A Fundamental Theorem for Exor Circuits

Marek A. Perkowski
Department of Electrical Enginering
Portland State University, P.O. Box 751, Portland, Oregon, 97207, USA

Abstract

The paper presents a theorem that forms a foundation to all well-known and all possible new canonical circuits with EXOR output gates. Let M be a $2^n \times 2^n$ *binary matrix with columns corresponding to minterms and rows corresponding to a family of Boolean functions of n variables. M[i,j] = 1 means that the Boolean function of row "i" includes the minterm corresponding to column "j". If the rows are linearly independent with respect to bit-by-bit EXOR operation, then the family is called "orthogonal family of Boolean functions". The functions from the family are called "orthogonal" functions. The theorem states that for any orthogonal family of* 2^n *Boolean functions* f_i *of n variables represented as a* $2^n \times 2^n$ *matrix M, there exists a canonical three-level realization* $F = f_0 \cdot S_0 \oplus ... \oplus f_{2^n-1} \cdot S_{2^n-1}$*, where functions* f_i *are the given orthogonal functions, and coefficients* S_i *are determined by multiplying matrix* M^{-1} *by the vector of minterms FV of function F. Each such canonical expansion creates also an universal cell that can be used in multi-level trees, Directed Acyclic Graphs (DAGs), and generalized functional decision diagrams. Generalizations to multi-output incompletely specified functions F, and the concept of partitioned AND/OR/EXOR PLAs are also presented. Finally we illustrate practicality of these concepts in application to Fine Grain Field Programmable Gate Arrays (FPGAs) and to cellular logic in general.*

1 Introduction.

While I compared matrices of all well-known canonical AND/EXOR forms, the striking observation was that we always deal with a binary matrix of rows that are linearly independent with respect to bit-by-bit EXORing of them. Therefore, I tried to see, what would happen if I take *any* orthogonal matrix † instead of particular orthogonal matrices of the well-known forms. It resulted, that *for each such matrix there exist always one associated canonical form*, and the circuit realizations of these forms are EXORs of all kinds of gates, *not only EXORs of AND gates.* Quite amazingly, there seem to be no papers in the literature that would make this observation and find some practical uses of it. This simple invention generalizes the whole concept of "Reed-Muller logic" to a much broader family of canonical functions. Moreover, it helps to find new families of AND/EXOR canonical forms as well.

The paper has two parts. The first part is a self-sufficient theoretical contribution introducing the theorem, which we will call the Orthogonal Expansion Theorem. Section 2 presents the orthogonal matrices for some well-known and new forms, and presents the Theorem. The generalization for incompletely specified functions is done in section 3.

The second part of the paper presents circuit applications. Section 4 discusses the multi-level circuits based on the Theorem. Section 5 introduces the new FPGA technologies in which the discovered by us families of forms can find immediate practical applications. Open research problems are listed in section 6.

[0]This work was supported by NSF Grant MIP-9110772.

[0]† In the above abstract, I defined the concept of an "orthogonal family of Boolean functions" and its associated binary matrix.

2 The Orthogonal Expansion Theorem.

Let us first introduce and illustrate the concept of an orthogonal family of Boolean functions. We create a $2^n \times 2^n$ matrix M with columns corresponding to minterms (for a function with n variables we have 2^n columns). The rows correspond then to some Boolean functions of n variables. A 1 in the intersection of a row "i" and column "j" means that minterm "j" is in function "i". The set of rows can be linearly independent with respect to EXOR operation (i.e. rows are bit-by-bit exored). If a set of 2^n rows is linearly independent then there is one and only one matrix M^{-1}, inverse to M with respect to exoring operation. In such case, the family of Boolean functions corresponding to rows will be called the "orthogonal family of Boolean functions" (or set of orthogonal Boolean functions), and the matrix will be called an "orthogonal matrix".

Example 2.1. Let us observe that the set of minterms of two-variable functions is the orthogonal family, with M being a unitary 4×4 matrix **1** (with ones on a diagonal and zeros otherwise).

Example 2.2. For the standard Reed-Muller form of two variables, A and B, the set of orthogonal functions is 1, A, B, AB. This is expressed as the following matrix M:

$$M = \begin{array}{c} \begin{array}{cccc} \overline{A}\cdot\overline{B} & \overline{A}\cdot B & A\cdot\overline{B} & AB \end{array} \\ \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{array}.$$

The first row corresponds to function 1, the second row corresponds to function A (since it covers minterms $A \cdot \overline{B}$ and A $\cdot$ B), the third row corresponds to function B, the fourth row corresponds to function $A \cdot B$. The circuit corresponding to this form has an EXOR gate in the first (output) level and AND gates in the second level.

Example 2.3. For the Fixed-Polarity Reed-Muller form of polarity A=0, B=1, the set of orthogonal functions is $1, \overline{A}, B, \overline{A}B$.

$$M = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix}.$$

The second row corresponds to function $\overline{A}$ and the fourth row corresponds to function $\overline{A} \cdot B$. The circuit corresponding to this form has an EXOR gate in the first level, AND gates in the second level, and NOT gates in the third level.

Example 2.4. Let us now take some set of orthogonal functions that will not correspond to AND/EXOR/(NOT) realization. For instance, set of functions 1, A, AB, A+B is orthogonal. It is described by the following matrix M.

$$M = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix}.$$

The third row corresponds to function $A \cdot B$ and the fourth row corresponds to function $A + B$. The circuit corresponding to this form has an EXOR gate in the first level, AND gates in the second level, and *arbitrary functions* (AND and OR in this case) in the third level - Fig. 2.1.

In each of the above examples one can check that all Boolean functions of 2 variables can be obtained by EXOR-ing the rows. It can be also easily checked that all above matrices M are linearly independent, and each M has EXACTLY ONE inverse matrix M^{-1}. It can be then concluded that arbitrary linearly independent (invertible) matrix M over {0,1} defines an orthogonal family of functions. So, for any orthogonal matrix there exists exactly one "orthogonal expansion" and one canonical form. This creates an extremely large family of canonical expansions that generalize the well-known canonical AND/EXOR expansions.

Let us denote the vector of minterms by FV. CV denotes the vector of coefficients for some given orthogonal form represented by M. Given is an arbitrary orthogonal set of 2^n Boolean functions f_i of n variables. This set can be represented as a $2^n \cdot 2^n$ orthogonal matrix M with functions f_i as rows, i = 0, ..., $2^n - 1$.

Theorem 1. Given is function F($x_1, ..., x_n$) represented as a vector FV with 2^n coordinates corresponding to its minterms in natural binary ordering. There exists a canonical three-level realization

$F(x_1, ..., x_n) = f_0 \cdot S_0 \oplus ... \oplus f_{2^n-1} \cdot S_{2^n-1}$,

where functions f_i are the given orthogonal functions, and coefficients S_i are determined by the coefficient vector CV = $M^{-1} \times FV$.

Theorem 1A. Given is a function F($x_1, ..., x_m$) such that the set of input variables $x_1, ..., x_m$ includes properly the set $x_1, ..., x_n$. There exists an unique expansion

$F(x_1, ..., x_m) = f_0(x_1, ..., x_n) \oplus ... \oplus f_{2^n-1}(x_1, ..., x_n) \cdot SF_{2^n-1}(x_{n+1}, ..., x_m)$,

where functions f_i are the given orthogonal functions of n variables, and the coefficient functions (called also the "data input functions") SF_i of the remaining input variables are determined from the coefficient vector CV = $M^{-1} \times$ FV.

Example 2.5. Let us compare expansion of a two-variable function in two bases; the standard minterm basis, and the basis of orthogonal functions: $f_1 = 1$, $f_{AB} = AB$, $f_B = B$, and $f_{A+B} = (A + B)$:

$m_0\ \overline{A}\ \overline{B}\ \oplus\ m_1\ \overline{A}\ B\ \oplus\ m_2\ A\ \overline{B}\ \oplus\ m_3\ A\ B = S_1 \cdot (1)$

By substituting A = 0, B = 0, we get $m_0 = S_1$. By substituting A = 0, B = 1, we get $m_1 = S_1 \oplus S_B \oplus S_{A+B}$. By substituting A = 1, B = 0, we get $m_2 = S_1 \oplus S_{A+B}$. By substituting A = 1, B = 1, we get $m_3 = S_1 \oplus S_{AB} \oplus S_B \oplus S_{A+B}$. Hence we obtain the following equation for minterms: $FV = M \times CV =$

$$= \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} S_{AB} \\ S_B \\ S_{A+B} \\ S_1 \end{bmatrix} = \begin{bmatrix} m_0 \\ m_1 \\ m_2 \\ m_3 \end{bmatrix}$$

Therefore CV = $M^{-1} \times FV =$

$$\begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} m_0 \\ m_1 \\ m_2 \\ m_3 \end{bmatrix} = \begin{bmatrix} S_{AB} \\ S_B \\ S_{A+B} \\ S_1 \end{bmatrix}$$

Example 2.6. Let the function F be represented by a vector FV^T = [0 1 1 0]. CV =

$$\begin{bmatrix} 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} S_{AB} \\ S_B \\ S_{A+B} \\ S_1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

Then F = $f_{AB} \oplus f_{A+B} = (AB) \oplus (A+B)$, illustrated in Figure 2.2.

Let us also observe the following: (1) Theorem 1 is a special case of Theorem 1A, when n = m. (2) The canonicity of the forms based on these expansions results directly from the Theorem. (3) The uniqueness of functions SF_i for multi-level synthesis with universal cells results also from the Theorem. (4) it can be shown that for every ESOP circuit for function F there exists at least one set of orthogonal functions in which this ESOP realization is the canonical solution to F. (5) see [11, 13] for formulas and examples of how to find functions SF_i.

It is important to note that the above examples do not imply the computer realization of the algorithms. For any kind of expansion, the matrices M and M^{-1} can be calculated just once and next stored in disk storage. The rows of these matrices represent Boolean functions, so storing them as minterms is not efficient. Although we use minterms and binary matrices in above explanation, our computer implementation is based on representing all these functions as BDDs [23, 1]. For instance, while calculating a coordinate "j" of vector CV (the coefficient CV_j), one finds an intersection of a BDD corresponding to row "j" of matrix M^{-1} with the BDD corresponding to function F. If the number of true minterms in this intersection BDD is odd, the value of the coefficient CV_j" is 1, otherwise it is 0. To help counting the number of minterms we modified the concept of the BDD by adding to each node a pointer to the number of minterms covered by it. Since the BDDs represent basically sets of disjoint cubes, for several orthogonal sets it is possible to find rules to generate relatively quickly the BDDs corresponding to rows of M and M^{-1} by generalizing the "disjoint cube" based methods from [6, 15, 21, 22].

In case of multi-output functions, the expansion is applied concurrently to the vector of functions, by applying the above expansion to each output function of the vector separately.

3 Generalization to Incompletely Specified Functions.

Let us observe that the outlined method can be easily generalized to incompletely specified functions F. To do this, every "don't care" minterm of function F is represented in the Function Vector FV of function F with a different symbol d_i. By multiplying the transform matrix M^{-1} by such a vector one obtains a Coefficient Vector CV with symbolic coordinates. Each coordinate of this vector is an EXOR of symbols d_i, and sometimes a 1. Symbol manipulation based on basic laws of Boolean algebra is used to simplify this vector ($A \oplus A = 0$, etc). Next, the solution with the smallest number of non-zero coefficients of the Coefficient Vector is found by finding the best assignment of values 0 and 1 to all d_i's from the coordinates. This is done by a heuristic tree searching algorithm that subsequently selects variables d_i and assigns them values (similarly to the satisfiability algorithms).

Example 3.1. Assuming a set of orthogonal functions: $f_1 = 1$, $f_a = a$, $f_b = b$, $f_{a+b} = a + b$, and function F from the Karnaugh map in Figure 3.1, the symbolical Coefficient Vector is calculated as follows: $CV = M^{-1} \times FV =$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} 0 \\ 1 \\ 1 \\ d \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \oplus d \\ 1 \oplus d \\ d \end{bmatrix}$$

By substituting d = 0 the Coefficient Vector becomes: CV = $[0110]^T$, with two non-zero coordinates. By substituting d = 1 it becomes: CV = $[0001]^T$, with one non-zero coordinate. The canonical solution to function F based on matrix M^{-1} is thus

F = $S_1 \cdot 1 \oplus S_a \cdot a \oplus S_b \cdot b \oplus S_{a+b} \cdot (a+b) = 0 \cdot 1 \oplus 0 \cdot a \oplus 0 \cdot b \oplus 1 \cdot (a+b)$ = (a+b).

In this particular case the output is an EXOR of 0 and a single orthogonal function, (a+b).

Similarly to the algorithms mentioned in section 2, the heuristic algorithm to solve this problem represents Boolean functions as the BDDs, and not the binary vectors. Its strength comes from the efficiency of manipulating the BDDs of many variables [23, 1].

4 Universal Cells and Their Use in Multi-Level Circuits.

The techniques based on Theorem 1 allow to realize functions of n variables. When the Boolean function F has $m > n$ variables, the $2^n \cdot 2^n$ orthogonal matrices are used to realize function F in multi-level realization by using Theorem 1A. Each such matrix M_r specifies an "universal cell" to be used in an orthogonal expansion e_r. Fig. 4.1 illustrates an universal cell with four inputs and three outputs. One can treat it as the generalization of multiplexers and AND/EXOR cells. Such cells are used to realize Davio expansions, while realizing the circuit as a multi-level DAG created from Kronecker Functional Decision Diagrams (KFDDs) [16, 22, 23]. Input variables are given to the "address inputs" and functions SF_i to the "data inputs" (notice a different use of vertical buses than in Fig. 5.1).

Fig. 4.2 illustrates a DAG with three levels of single-output universal cells. The boxes represent universal cells, labels "e_r" inside the boxes denote types of expansions used in them.

Let us observe that the DAG from Figure 4.2 is a generalization of the well-known DAG circuits created with multiplexers [22, 30]. While the multiplexer gate is used in (tree and DAG) realizations based on Shannon expansion, the multiplexers and AND/EXOR gates are used in Reed-Muller Trees, Reed-Muller DAGs, Kronecker-Reed-Muller Trees, Kronecker Reed-Muller DAGs, and other multilevel circuits based on Shannon and two Davio (ring) expansions [10, 11, 13]. Other more powerful universal functions and multilevel structures that use them have been created based on general "orthogonal" and "orthonormal" expansions [11, 13].

Let us observe that a gate based on a set of orthogonal functions F = $f_0 \cdot S_0 \oplus \ldots \oplus f_{2^n-1} \cdot S_{2^n-1}$ can be easily generalized to a multi-output gate, called "EXOR PLA of orthogonal functions" - see Fig. 4.1. In particular, it can be realized as regular "EXOR of AND/OR/wire gates" shown in Fig. 4.3. Such multi-output gates can be used to develop new FPGAs, that would combine advantages of standard partitioned PLAs, fine-grain FPGAs, and EXOR-based universal gates. We call them the "partitioned AND/OR/EXOR PLAs", and hope that they will become commercially available. Among synthesis methods that one can create for such new "Universal Gates" the most natural seem the generalizations of the tree and DAG methods from the literature. One will be thus able to distinguish the "Levelized" Trees and DAGs (generalizations of the Reed-Muller Trees, Kronecker-Reed-Muller Trees, etc), and the "Permuted" Trees and DAGs (generalizations of structures introduced in [29]). Levelized Trees and DAGs are those in which variables are partitioned to groups, and only one group of variables will be available at any level of the structure (see Fig. 4.2). Permuted Trees and DAGs are those in which various sets of variables can be at any level.

In "Disjoint Permuted Trees and DAGs" (see Fig. 4.4), the variables are still partitioned to disjoint sets (a,b, c,d, e,f, and g,h), but various sets can appear at the same level of the expansion (like sets c,d and e,f).

In "Overlapped Permuted Trees and DAGs" (see Fig. 4.5), the variables are partitioned to non-disjoint sets (a,b, b,c,d, a,e, and e,g), and several variables can appear at the same level of the expansion.

The Levelized Trees and DAGs are the base to create the new concept of *Orthogonal Decision Diagrams (ODDs)* introduced here for the first time. ODDs are the generalizations of the KFDD decision diagrams introduced in [11, 13, 23, 16, 22]. While in every node of KFDD one of three possible orthogonal expansions for a *single variable* is applied, in node of ODD applied is one of all possible orthogonal expansions for some selected *number of variables*. The size of such diagrams is always not worse than that of the KFDDs, but their applicability is limited by the lack of fast heuristic algorithms to select the best orthogonal expansion for the given function.

5 Fine Grain FPGAs and Sea-of-Gates Logic.

In our former papers [10, 11, 13] we introduced several families of canonical and multi-level circuits, also for the multiple-valued logic. We formulated the generalized "Orthogonal" and "Orthonormal" expansions, that include an extremely wide category of new canonical circuits and multi-level circuits. It was, however, not until we became familiar with new Fine-Grain (Cellular) Field Programmable Gate Array architectures, that we realized the big practical importance of some of those new expansions, as well as the expansions introduced above.

New FPGAs, called "Fine Grain", "Cellular", or "sea of gates" FPGAs, have been recently available or announced by Concurrent Logic (CLI, now part of ATMEL), Plessey, Toshiba, Algotronix and Motorola, but very few logic synthesis methods for these devices have been published. The methods include: adaptation of general mapping methods to cell libraries [26, 27], creation of Reed-Muller trees [29, 30], Kronecker Reed-Muller trees and KFDDs [11, 13, 16, 22], which are next regularly mapped to the ATMEL architecture.

In this paper we use the ATMEL 6000 series FPGAs as an illustration, but the methods are general and can be used in any of those architectures and *also in sea-of-gates Gate Array design.* (It was observed by Motorola that one of the important design issues is the development of such FPGAs and associated design tools that the migration from FPGA to ASIC sea-of-gates gate arrays would be easy and would preserve the timing [28]).

It can be observed that regular arrays of cells available in ATMEL allow to emulate PLA-like structures (Fig. 5.1), with all the input variables in the local buses of the AND-plane and all the output variables in the OR-plane. In general, the AND plane of the classical AND/OR PLA will be called the "input plane" and the OR plane will be called the "output plane". The outputs of the input plane will be called "internal variables". Although this approach allows to realize efficiently some (multi-output) Boolean functions, it does not use fully the resources of the ATMEL chip. It can be then observed that since the two-input EXOR gates are available as personalizations of the CLBs ("Configurable Logic Blocks") of the ATMEL 6000, also the EXOR PLA can be easily realized (Fig. 5.2). The EXOR PLA has thus AND gates in the input plane and EXOR gates in the output plane. The EXOR PLAs have been discussed in literature for years [17] but with the exception of [7] they have been not realized in ASIC hardware nor in PLD devices.

Now, one can appreciate that a regular two-dimensional arrangement of two-input gates, AND, OR and EXOR, is fundamental to both these designs. It is then obvious that multi-level AND/OR/EXOR designs not worse than OR-based and EXOR-based PLAs can be easily realized in ATMEL 6000. In this design, called by us a "generalized PLA", two-input AND-, EXOR-, and OR-gates exist in both the input and output planes.

Unfortunately, until very recently the methods to minimize Boolean functions to such structures have been absent, and PSU's team was the only group that has developed several general methods to minimize this kind of logic. These methods can be divided to three categories: exhaustive heuristic search [3, 4, 9, 12, 15, 22, 25, 30], rule-based methods [26, 27], and spectral methods [6, 14, 19, 20, 21, 31].

In spectral methods we assume that some canonical form of the Boolean function must be found, which minimizes the "envelope area" of the "generalized PLA". This paper additionally assumes that the rows in the output plane correspond to EXOR gates. Now, the columns in the input plane can be still combinations of AND, OR and EXOR gates. However, because we do not want to waste area for the "connection cells", we assume that local buses are entirely used for leading the input variables to gates in the input plane, and for leading the internal variables to the output plane. We assume also that one (horizontal) bus is used for one input variable, and a single vertical bus is used for an internal variable. In the input plane, each column takes one input from the cell located above it, and the another input from the horizontal bus. A cell in row X can be also configured to be a "wire", which means, the output of the cell located directly above the cell in row X is AND-ed in row X with constant 1 (see Fig. 5.1).

Most importantly, we assume that the selection of gates in the columns of the input plane is done in such way that all possible column functions create an *orthogonal set of functions* - see Fig. 5.3. In such special case the PLA-like structure from Fig. 5.3 will be called the *"orthogonal PLA"*. The input plane will be called the *"orthogonal plane"*.

The requirement of the orthogonality of the column functions means that (assuming certain order of input variables) every Boolean function is realized in one and not more than one way. It is then a *canonical representation* of this function. There is an astronomical number of orthogonal sets of functions that can be realized in this way in cellular architectures [14].

6 Open Research Questions.

The fundamental question is the following. Given an orthogonal function base, how to find the values of the binary spectral coefficients S_i for each function f_i, i = 0, ..., $2^n - 1$. There are two main applications of these coefficients. (1) In the case of "orthogonal PLA" synthesis, they are used to select orthogonal functions to be programmed in the orthogonal plane. (2) In the case of multilevel circuits from "universal gates", created based on generalized orthogonal expansions [13], the matrix M^{-1} determined to find the coefficients is also useful to find the "data inputs" to the universal cells at every level of the circuit. Efficient solutions to those problems must be found. Next, similarly as in Fixed-Polarity Reed-Muller algorithms one would go through all possible forms of certain type to find the best one. Since the number of all canonical expansions found by us is astronomical and only some of them have practical importance, we analyse those families of orthogonal functions that have easy circuit realizations [14]. It is important to find, among all families of orthogonal functions such sets of families which have circuit realizations giving certain advantages, such as speed, area or testability.

The Theorem can be useful to solve the following problems: (1) given is a set of $K < 2^n$ functions of n variables how to add the minimum number of additional functions to be able to realize given function F as an EXOR of all those primary and additional functions. (2) How to find efficiently the realization of the given function in particular canonical expansions. (3) find find such sets of families which have circuit realizations that have the above-mentioned advantages; for instance in which the orthogonal functions have regular layouts, like those presented in sections 4 and 5.

Since AND/EXOR canonical forms have universal tests and very good testability properties, it would be interesting to investigate whether these properties hold for the new forms. Since the cellular realizations of the new forms are never larger than such realizations of AND/EXOR forms, and the number of tests is definitely larger than for the Reed-Muller forms, investigating the size/testability trade-offs will be useful.

The presented approach opens several important problems; some of the most fundamental questions are:

1. How to create efficiently the matrix M? Can one use concepts similar to Kronecker Product of Matrices in order to quickly generate the matrix M? Are there fast recursive algorithms to create it?

2. How to create efficiently the matrix M^{-1}? Can one adopt the Kronecker-like methods of AND/EXOR circuits? Are there any special fast methods of finding the M^{-1}?

3. Can problems 1 and 2 be solved for some particular new families of orthogonal functions (other than the known AND/EXOR ones)? †

4. What are the new practically interesting families of orthogonal functions?

5. How to create efficient methods for circuits that have arbitrary combinations of AND, OR and wires in "orthogonal plane" and individually negated inputs to gates in the plane.

6. Can the structure of orthogonal matrices be used to divide the space of *all* ESOPs to families of canonical expansions?

7. Development of efficient ODD algorithms: to create ODDs from netlists, and to execute all Boolean operations on them. Finding good variable-ordering and expansion-selection heuristics. Investigation of ODD diagrams based on levelized and non-levelized orthogonal expansions.

8. Development of BDD- or ODD-based algorithms to represent efficiently operations on Boolean functions while finding M^{-1}, instead of representing these functions and operations by using minterms and matrices.

9. Development of more efficient methods to handle don't cares, that would be based on the principles from section 3.

10. Development of practical logic synthesis/physical design algorithms for partitioned AND/OR/EXOR PLAs.

At present we work on some of those problems [14], and we found some efficient partial solutions to them.

7 Conclusions.

The main contribution of this paper was to present a broad new family of *canonical forms* and corresponding *universal gates*. Since the previously found canonical forms of Boolean functions proved to be useful both in theory and in practice, I hope that these newly discovered forms will find applications as well.

Since people usually associate canonical forms (other than trivial sum-of-minterms) with AND/EXOR logic, it must be emphasized here that the new forms are AND/OR/EXOR and not only AND/EXOR [14].

There is also a chance that the Orthogonal Expansion Theorem will help to define a new subject area in logic synthesis research. Since the name "AND/EXOR circuits" is used equivalently with "Reed-Muller Logic", and the new circuits have OR gates as well, a new name must be found for them. I would like to propose to call this new research area the *"Orthogonal Logic"*. Let us observe that all problems of Reed-Muller Logic have counterparts in this new Orthogonal Logic: defining new canonical forms in terms of matrices, factorization of matrices, fast algorithms, use as general Boolean function representation, and more... The Orthogonal Logic is the extension and generalization of the Reed-Muller Logic.

[0]† The answer is positive!

8 Acknowledgment.

The author would like to thank Dr. Richard Rudell for very careful reading of the early manuscript and asking questions that helped greatly to improve the presentation of this paper.

References

[1] E.M. Clarke, X. Zhao, M. Fujita, and Y. Matsunaga, "Fast Walsh Transform Computation with Binary Decision Diagrams", *Proceedings of this Workshop.*

[2] Concurrent Logic Inc., CLI 6000 Series Field Programmable Gate Arrays, *Preliminary Information,* Dec. 1, 1991, Rev. 1.3.

[3] L. Csanky, M. Perkowski, I. Schaefer, "Canonical Restricted Mixed-Polarity Exclusive Sums of Products", *Proc. of the IEEE ISCAS'92*, pp. 17-20.

[4] L. Csanky, M. Perkowski, I. Schaefer: "Canonical Restricted Mixed-Polarity Exclusive-Or Sums of Products and the Efficient Algorithm for their Minimization", *IEE Proceedings,* Pt.E, Vol. 140, No. 1, Jan. 1993, pp. 69-77.

[5] M. Davio, J.P. Deschamps, and A. Thayse, Discrete and Switching Functions, *McGraw Hill,* 1978.

[6] B.J. Falkowski, and M.A. Perkowski, "One more way to calculate generalized Reed-Muller expansions of boolean functions," *Intern. J. Electr.* Vol. 71, No. 3, 1991, pp. 385-396.

[7] J. Froessl, and B. Eschermann, "Module Generation for AND/XOR-Fields (XPLAs)," *Proc. IEEEE ICCD '91,* pp. 26-29, 1991.

[8] D.H. Green, "Families of Reed-Muller Canonical Forms", *Intern. J. of Electr.,* pp. 259-280, February 1991, No 2.

[9] M. Helliwell, and M. Perkowski, "A Fast Algorithm to Minimize Multi-Output Mixed-Polarity Generalized Reed-Muller Forms", *Proc. of the IEEE/ACM DAC,* 1988, paper 28.2, pp. 427-432,

[10] M. Perkowski, P. Dysko, and B. Falkowski, "Two Learning Methods for a Tree Search Combinatorial Optimizer", *Proc. of the IEEE Intern. Phoenix Conf. on Comp. and Comm.,* Scottsdale, Arizona, March 1990, pp. 606-613.

[11] M. Perkowski and P. Johnson, "Canonical multi-valued input Reed-Muller Trees and Forms," *Proc. 3rd NASA Symposium on VLSI Design,* Oct. 1991, Moscow, Idaho, pp. 11.3.1-11.3.13.

[12] M. Perkowski, L. Csanky, A. Sarabi, and I. Schaefer, "Minimization of Mixed-Polarity Canonical AND/EXOR Forms", *Proc. of ICCD'92*, 32-36.

[13] M.A. Perkowski, "The Generalized Orthonormal Expansion of Functions with Multiple-Valued Inputs and Some of its Applications", *Proc. of the ISMVL'92,* pp. 442-450.

[14] M. Perkowski, A.Sarabi, F. Beyl, "XOR Canonical Forms of Switching Functions", *Proceedings of this Workshop.*

[15] A. Sarabi, M.A. Perkowski: "Fast Exact and Quasi-Minimal Minimization of Highly Testable Fixed-Polarity AND/XOR Canonical Networks", *Proc. of the IEEE/ACM DAC,*, 1992, pp. 30-35.

[16] A. Sarabi, P.F. Ho, K. Iravani, W.R. Daasch, and M. Perkowski, "Minimal Multi-level Realization of Switching Functions based on Kronecker Functional Decision Diagrams", *Proc. IWLS '93, International Workshop on Logic Synthesis.*

[17] T. Sasao, and Ph. Besslich, "On the Complexity of Mod-2 Sum PLA," *IEEE Transactions on Computers,* vol. 39, No. 2, pp. 262 -266, Febr. 1990.

[18] T. Sasao, "Optimization of Multiple-Valued AND-EXOR Expressions using Multiple-Place Decision Diagrams," *Proc. IEEE 22nd ISMVL,* 1992.

[19] I. Schaefer, and M. Perkowski, "Synthesis of Multi-Level Multiplexer Circuits for Incompletely Specified Multi-Output Boolean Functions with Mapping Multiplexer Based FPGAs", accepted to *IEEE Trans. on CAD,* in October 1992.

[20] I. Schaefer, and M.A. Perkowski, "Multiple-Valued Input Generalized Reed-Muller Expansions", *accepted to IEE Proceedings,* Pt.E, in June 1992.

[21] I. Schaefer, and M.A. Perkowski, "Multiple-Valued Input Generalized Reed-Muller Forms", *Proc. of the IEEE ISMVL'91*, pp. 40-48.

[22] I. Schaefer, M. Perkowski, and H. Wu, "Multi-level Logic Synthesis for Cellular FPGAs Based on Orthogonal Expansions," *Proceedings of this workshop.*

[23] I. Schaefer, *Ph.D. Thesis*, PSU, 1992.

[24] U. Kebschull, E. Schubert, and W. Rosenstiel, "Multilevel Logic Synthesis based on Functional Decision Diagrams", *Proc. IEEE Euro-DAC,* pp. 43-47, 1992.

[25] N. Song, and M. Perkowski, "EXORCISM-MV-2: Minimization of Exclusive Sum of Products Expressions for Multiple-Valued Input Incompletely Specified Functions", *Proc. ISMVL '93.*

[26] N. Song and M. Perkowski, "A Method of Logic Mapping for Fine Grain FPGAs", *Proc. IWLS '93.*

[27] N. Song and M. Perkowski, "A new approach to mapping for Fine Grain FPGAs," *submitted to Journal on VLSI Design.*

[28] R. Wilson, "FPGAs are closing in on gate array business", *Electronic Enginering Times, October 1992.*

[29] L-F. Wu, and M. Perkowski, "Minimization of Permuted Reed-Muller Trees for Cellular Logic Programmable Gate Arrays", *Proc. of the 2nd Intern. Workshop on Field-Programmable Logic and Applications, FPL'92,* Vienna, Austria, August 31-September 2, 1992, pp. 7/4.1-7/4.4.

[30] H. Wu, M. Perkowski, and N. Zhuang, "Synthesis of Multiplexer Directed-Acyclic-Graph network with application to FPGAs and BDDs," *Proc. IWLS '93.*

[31] H. Wu, M. Perkowski, and N. Zhuang, "Canonical Restricted Partially-Mixed-Polarity Reed-Muller Expansion and its Fast Computation with application to AT 6000 Series FPGA mapping," *Submitted to IEEE Trans. on Computers*, 1993.