MINIMIZATION OF MULTIPLE-VALUED INPUT MULTI-OUTPUT MIXED-RADIX EXCLUSIVE SUMS OF PRODUCTS FOR INCOMPLETELY SPECIFIED BOOLEAN FUNCTIONS

Marek Perkowski, Martin Helliwell, Pan Wu
Portland State University, Department of Electrical Engineering
P.O. Box 751, Portland, OR, 97207, tel. (503) 464-3806x23

ABSTRACT

A new concept of a mixed-radix multiple-valued input exclusive sum of products (MRESP) is presented and some possible circuit realizations for the new concept are discussed. The algorithm starts from a Boolean function and generates an approximate MRESP form and the appropriate multi-output circuit. Such circuits can have smaller complexity than both the EXOR forms with mixed polarity, the PLAs with decoders, and the networks with two-variable function generators. They are also easily testable.

1. INTRODUCTION

There has been a growing interest in recent years regarding the design of logic circuits using EXOR gates. Particular interest is in the minimization of the Generalized Reed-Muller Forms (GRM) with fixed polarity of variables or EXOR forms with mixed polarity of variables [4,22,23,25,51,57,30]. Functions realized by such circuits can have fewer gates, fewer connections, and take up less area in the VLSI realization [4,22]. More importantly, such circuits are easily testable. They are also used in self-testing circuits [2,4,21,25,35,41,42,45,49,50,56]. Besslich writes [4] :

> "secondly, the testability of circuits is significantly improved [49]. The gains from this second advantage may even exceed possible disadvantages in such cases where the EXOR realization is more costly than the equivalent vertex (sum of product) form. Applications have so far not become very popular because of the practical difficulties in the design procedure".

Circuits of this type find applications in linear machines, arithmetic and communication circuits, encrypting schemes, coding schemes for error control and synchronization, sequence generation for process identification, system testing, etc.

The problem of the minimization of such circuits is very important but it was traditionally treated as extremely difficult. Since optimal solutions can be found only for functions with not more than 5 variables [43], the interest is in approximate solutions. Few authors [4,22,51] have implemented computer programs. The paper [30] presents a new and efficient algorithm for mixed polarity EXOR forms and the corresponding program. The algorithm from [30] will be improved here, and also extended for the case of the logic with multiple-valued inputs that generalizes the classical Boolean logic and finds many important applications in logic design [8,12,13,17,18,33,37,38,39,44,52-55,59-65]. We assume that the number of truth values for each variable can be different.

We present a solution to a problem here that has not yet been formulated and solved in the literature: mixed-radix minimization of multiple-output, multiple-valued, incompletely specified functions. The only paper that relates to multiple-valued PLA-like logic realization with OR-plane elements that generalize exors is [13]. The difference of our approach is, however, that we use normal EXOR gates not modsums, that the output in our approach is binary, the function is incompletely specified and multi-output, and that we give a constructive algorithm to find a solution.

This paper describes an approximate method that yields especially good results for the minimization of strongly unspecified multi-output logic functions with multiple-valued inputs and binary-valued outputs. Such circuits can have smaller complexity than both the mixed polarity EXOR forms [30], PLA with decoders from [61], and networks with two variable function generators from [64]. The most important advantage to this method is that it can produce circuits that are superior (in terms of speed or area) to those obtained using other methods for the logic with multiple-valued inputs and are also very easily testable. Although the method in question can be applied to any type of such a function and each variable can have an arbitrary number of logic values, an example of the method using 4-valued inputs will be presented [61,64]. This variant finds, among others, applications in the minimization of PLAs in which pairs of inputs are implemented via 2-by-4 decoders (2 inputs, 4 outputs).

2. BOOLEAN LOGIC WITH MULTIPLE-VALUED INPUTS

A multiple-valued input, two-valued output, incompletely specified switching function f (*multiple-valued function*, for short) is a mapping $f(X_1, X_2, \cdots, X_n)$: $P_1 \times P_2 \times \cdots P_n \rightarrow B$, where X_i is a *multiple-valued variable*, $P_i = \{0, 1, \ldots, p_i - 1\}$ is a *set of truth values* that this variable may assume, and $B = \{0, 1, -\}$ (- denotes a *don't care value).* This is a generalization of an ordinary n-input switching function f: $B^n \rightarrow B$.

Definition 1. For any subset $S_i \subseteq P_i$, $X_i^{S_i}$ is a *literal* of X_i representing the function such that

$$X_i^{S_i} = \begin{cases} 1 & \text{if } X_i \in S_i \\ 0 & \text{if } X_i \notin S_i. \end{cases}$$

Definition 2. A product of literals, $X_1^{S_1} X_2^{S_2} \ldots X_n^{S_n}$, is referred to as a *product term* (also called *term* or *product* for short). A product term that includes literals for all function variables $X_1, X_2, \ldots, X_n$ is called a *full term.* A sum of products is denoted as a *sum-of-products expression (SOPE)* while a product of sums is called a *product-of-sums expression (POSE).* An EXOR of products will be called a *Mixed-Radix Multiple-Valued Exclusive Sum of Produts Form* (*MRESP* for short).

Switching functions with multiple-valued inputs, two-valued outputs, find several applications in logic design, pattern recognition, and other areas. In logic design, they are primarily used for the minimization of PLAs that have 2-bit decoders on the inputs. A Programmable Logic Array (PLA) with r-bit decoders directly realizes a SOPE of a 2^r-valued input, two-valued output, function [59,60,61]. An *EXOR-based PLA,* is a PLA in which EXOR gates replace OR gates in the OR plane. An *EXOR-based Programmable Logic Array (EXPLA)* with r-bit decoders directly realizes a MRESP of a 2^r-valued input, two-valued output, function [59,60,61].

3. DEFINITIONS AND BASIC PROPERTIES

Any multiple-output function can be represented and minimized as a function with a single two-valued output. Let us consider a Boolean function F with *multiple output,* that is, $F(X_1, \ldots, X_{n-1}) = (f_0, \ldots, f_{m-1})$. We define a single-output switching function F on n variables where variables $X_1, \ldots, X_{n-1}$ are multiple-valued and the variable X_n takes the values $\{0, \ldots, m-1\}$. $F(X_1, \ldots, X_{n-1}, X_n) = F_{X_n}(X_1, \ldots, X_{n-1})$ where $F_i(X_1, \ldots, X_{n-1})$ denotes the i-th *projection* of $F(X_1, \ldots, X_{n-1})$, that is, f_i. We will, therefore, consider, henceforth, only single-output switching functions and n will denote the number of input variables.

Example 1. Given is a 3-input 2-output binary function F(a, b, c) = (f_0, f_1) with binary inputs from Fig. 1a. This function is described by an array of disjoint ON-cubes from Fig. 1b. After transforming this array to the ON-cubes array of 4-input function F(a, b, c, X) with 2-valued input X, the array is as in Fig. 1c, see also a Karnaugh map from Fig. 1d. As the result of the minimiza-

tion of function F(a, b, c, X), the MRESP is $F = a^1b^1 \oplus c^1b^1X^1 \oplus c^1X^0 = ab \oplus cbX^1 \oplus cX^0$ (Fig. 1e). Substituting X = 0 to this solution, we obtain function f_0(a, b, c) = $ab \oplus c$. Substituting X = 1 we obtain f_1(a, b, c) = $ab \oplus bc$ (Fig. 1f). The corresponding circuit is presented in Fig. 1g. Another solution is F(a, b, c, X) = $ab\bar{c} \oplus \bar{b}cX^0 \oplus \bar{a}bc$ (Fig. 1h), which, after separation to single functions, is f_0(a, b, c) = $ab\bar{c} \oplus \bar{b}c \oplus \bar{a}bc$, f_1(a, b, c) = $ab\bar{c} \oplus \bar{a}bc$. The corresponding circuit is in Fig. 1i. As we see, a better solution is obtained when both the number of terms and the number of literals are minimized.

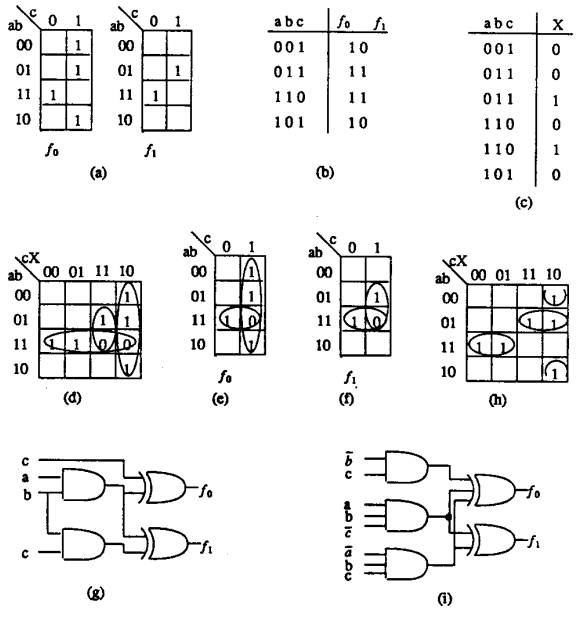


Fig. 1

A switching function with multiple-valued inputs is uniquely determined by its *truth table*, or by an expression given in SOPE or POSE. A multiple-valued input, two-valued output, function will be simply called a *function* from now. Any literal of the form $X_i^{P_i}$ is identically equal 1. Hence, we often write $X_i^{P_i} X_j^{S_j}$ as $X_j^{S_j}$. Although our method can be applied to a multiple-valued input function with any number of input values, we will concentrate on 4-valued examples from now on as P_i = P = {0, 1, 2, 3}.

Definition 3. A map of an n-variable, p-valued input, two-valued output function consists of p^n *cells*. Cells that contain a 1 will be called *true minterms* (1-cells) while cells that contain an 0 will be called *false minterms (0-cells of the map).*

The maps that we will use in this paper generalize the concept of the Karnaugh maps for the case of multiple-valued input functions. When our method is used for a completely specified function it uses an *array ON of arbitrary disjoint cubes (product terms,* ON-cubes, cubes of minterms). These can be minterms, full terms, or any disjoint product implicants [44]. In the case of an incompletely specified function the function is represented as the *array ON of disjoint ON-cubes* and the *array DC of disjoint DC-cubes (DC-cubes are cubes of don't cares).*

Example 2. Given is a 2-valued input, single 2-valued output function f(a, b, c, d), presented in the Karnaugh map (K-map) shown in Fig. 2a. Using classical approach to Boolean function minimization (Fig. 2b), the SOPE from Fig. 2c is found.

Example 3. Fig. 3a shows a four-valued input, two-valued output function. This function corresponds to the function from the previous example.

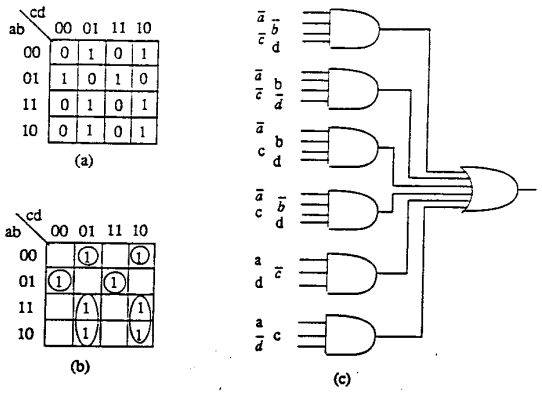


Fig. 2.

Variables a and b have been paired to a 4-valued variable X and variables c and d to a 4-valued variable Y. The encoding of the variables' values is as follows: 00 - 0, 01 - 1, 10 - 2, 11 - 3. Therefore, $\bar{a}\bar{b}$ is the value 0 of variable X, $\bar{a}b$ is the value 1 of variable X, $a\bar{b}$ is the value 2 of variable X, and ab is the value 3 of variable X. Similarly, the values are assigned for 4-valued variable Y as seen in Fig. 3a. Using any method of multiple-valued input function minimization, an expression: f = $X^{023} Y^{12} + X^1 Y^{03}$ is found whose SOPE circuit realization is shown in Fig. 3b.

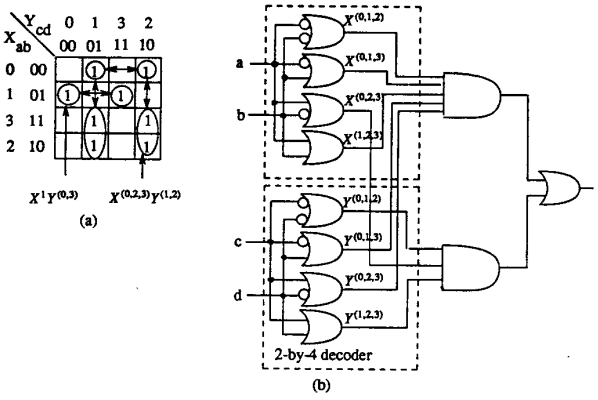


Fig. 3.

4. MULTIPLE-VALUED INPUT MIXED-RADIX EXCLUSIVE SUM FORMS

Let A, B, C denote any values of multiple-valued input literals, or any functions on them. The following operations hold for multiple-valued input algebra:

1. Associative laws:
   1A. $A \oplus (B \oplus C) = (A \oplus B) \oplus C$ 1B. $A(BC) = (AB)C$
2. Distributive law: $A(B \oplus C) = AB \oplus AC$
3. Commutative laws: 3A. $A \oplus B = B \oplus A$ 3B. $AB = BA$
4. Identities: 4A. $A \oplus 0 = A$ 4B. $X_i^{S_i} \oplus 1 = X_i^{P_i - S_i}$ 4C. $A0 = 0$

   4D. $A \oplus A = 0$ 4E. $X_r^{S_i} \oplus X_r^{S_j} = X_r^{(S_i - S_j) \cup (S_j - S_i)}$

   4F. $X_i^{S_i} \oplus X_i^{P_i - S_i} = X_i^{P_i} = 1$

4G. $X_i^{S_i} X_j^{P_i - S_i} \oplus X_i^{P_i - S_i} X_j^{S_j} = X_i^{S_i} \oplus X_j^{S_j}$

4H. if $S_i \supset R_i$ and $S_j \cap R_j = \phi$
then

$$X_i^{S_i} X_j^{S_j} \oplus X_i^{R_i} X_j^{R_j} = X_i^{S_i \cup R_i} X_j^{S_j \cup R_j} \oplus X_i^{S_i - R_i} X_j^{R_j}$$

4I. if $S_i \supset R_i$ and $S_j \supset R_j$ then

$$X_i^{S_i} X_j^{S_j} \oplus X_i^{R_i} X_j^{R_j} = X_i^{S_i - R_i} X_j^{S_j} \oplus X_i^{R_i} X_j^{S_j - R_j}$$
$$= X_i^{S_i} X_j^{S_j - R_j} \oplus X_i^{S_i - R_i} X_j^{R_j}$$

Definition 4. The *degree of a term* is the number of literals in it that are not identically equal to 1.

For instance the degree of $X^{23} Y^{02}$ is two. The degree of $X^{01} Y^{0}$ is two in three-valued algebra, but it is one in a two-valued algebra, since $X^{01} Y^{0} = 1\, Y^{0} = Y^{0}$.

Definition 5. The *distance of two cubes* is the number of variables for which the corresponding literals have disjoint sets of truth values (in other words - the number of variables for which the sets S_i and R_i are *disjoint*).

For instance, the distance of cubes $X^{01}Y^{02}Z^{123}$ and $X^{12}Y^{1}Z^{13}$ in the four-valued algebra is one, since sets {0, 1} and {1, 2} of variable X are non-disjoint, so as the sets {1, 2, 3} and {1, 3} of variable Z. The sets {0, 2} and {1} of variable Y are disjoint.

Our primary goal of MRESP synthesis is to *minimize the number of terms* (inputs to EXOR gates). For the circuit with the minimum number of terms our secondary goal is to minimize the total number of inputs. Therefore the *cost function C* to be minimized is:

$$C = NT + \frac{NI}{NI_{in}}$$

where:
- NT is the total number of terms in the solution,
- NI is the total number of literals in the solution,
- NI_{in} is the total number of the literals in the initial function.

In addition, *the cost function C1* for folded realizations of EXPLA or for standard-cell realizations is:

$$C1 = NT + \frac{NI1}{NI1_{in}}$$

where:
- $NI1$ is the total number of input wires to AND and EXOR gates in the solution,
- $NI1_{in}$ is the total number of input wires to AND and EXOR gates in the initial function.

For instance, literal X^{012} as an input to an AND gate requires a single wire for the 2-by-4 decoder realization of logic with 4-valued inputs. X^{01} is realized as $X^{012} X^{013}$. It, therefore, requires two wires. Similarly $X^{0} = X^{012} X^{013} X^{023}$ requires three wires.

The multiple-valued mixed-radix MRESPs that are discussed here, like RM forms and GRM forms, also have very good testability properties [58]. Tests are function independent and can easily be created from standard tests for classical Reed Muller forms [49]. No systematic study however done by other authors, similar to that of [49], [56] and [58], is known to us.

5. THE MULTIPLE-VALUED XLINK OPERATION.

5.1. Primary xlinking.

Basic operation of our system is the operation of **xlinking** that generalizes several operations known from previous papers.

Let us first observe that any two minterms m1 and m2 in a m-valued map can be linked by a chain of groups, where

- each group includes two adjacent map cells (*adjacent* are cells of distance 1),
- the first group includes m1, the last group includes m2,
- any two subsequent groups of the chain include a single common minterm.

It can easily be proven that the EXOR of minterms m1 and m2 is equal to the EXOR of all the groups from the chain. Figure 4 is a map of three 3-valued variables. It shows an example of a chain from minterm 000 to minterm 222. Let us observe that there are usually many such chains from m1 to m2 (there are d ! of them, where d is the distance of m1 and m2).

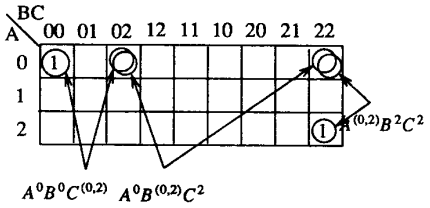


Fig. 4.

The first principle of our method to minimize MRESP forms is based on the idea that any two minterms can be expanded into an EXOR sum of one or more terms containing fewer literals.

The chaining operation expands the well-known Boolean rules of merging and exclusion used in the Quine-McCluskey, or other, logic design algorithms. It finds one of the shortest paths between two nodes of a multiple valued hypercube. The number of generated terms equals the distance of these nodes.

We will give a *systematic procedure for finding xlinks of full terms* below. The application of this procedure will be called **xlinking** (pronounced crosslinking). The result of the procedure will be called the **xlink** (crosslink) of the two original full terms. The full terms and the product terms are represented in the computer in the positional notation that allows for both 2-valued and m-valued minimization [61]. It can be easily verified that the xlinking can be found if the truth value sets for each variable are either disjoint or equal.

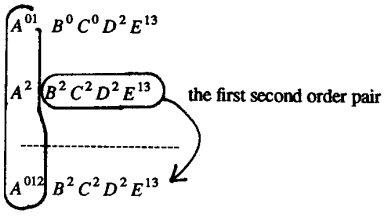
To find the xlink of a pair of two full terms, for example $A^{01} B^{0} C^{0} D^{2} E^{13}$ and $A^{2} B^{2} C^{2} D^{2} E^{13}$, in a 4-valued function, we write them vertically like this:

$A^{01} B^{0} C^{0} D^{2} E^{13}$

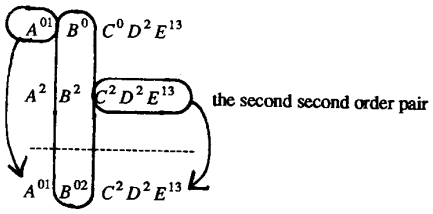
$A^{2} B^{2} C^{2} D^{2} E^{13}$
↑↑↑

Each time when the polarities of the literal are disjoint from full term to full term in the pair it is denoted by an arrow. Each arrow will give rise to one term of the xlink. Let us now consider each arrow separately. The above initial pair of full terms can then be expanded to three **second order pairs**, for variables A, B, and C respectively, as shown below.

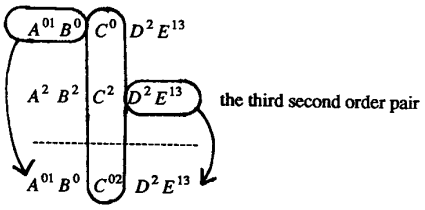
For variable A, the term $A^{012}\ B^{2} C^{2} D^{2} E^{13}$ is created as follows:



For variable B, the term $A^{01}\ B^{02}\ C^{2} D^{2} E^{13}$ is created as follows:



For variable C, the term $A^{01}B^{0}\ C^{02}\ D^{2}E^{13}$ is created as follows:

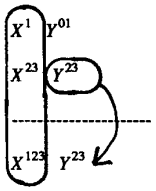


Under each pair of literals of disjoint sets of polarities under consideration (A in the first pair, B in the second, C in the third pair), we write the multiple-valued literal, with the set of truth values being the sum of the respective sets from the literals of the cubes. To create the result of xlink for a second order pair, **we copy the part of the term to the left of the dash from the top full term. The part to the right of the dash is copied from the bottom full term as shown.** The xlink of the initial pair of full terms is an EXOR of xlink terms of the second order pairs for each variable of different polarities. Therefore

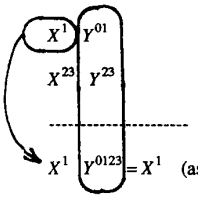
$A^{01}B^{0}C^{0}D^{2}E^{13} \oplus A^{2}B^{2}C^{2}D^{2}E^{13} = A^{012}B^{2}C^{2}D^{2}E^{13} \oplus A^{01}B^{02}C^{2}D^{2}E^{13} \oplus A^{01}B^{0}C^{02}D^{2}E^{13}$

This procedure can easily be further extended for any two terms in which for every two correspondingly different literals for the same variable: $X_i^{S_j}$ and $X_i^{S_k}$ the sets S_j and S_k are disjoint. For example, terms $X^{1}Y^{01}Z^{13}$ and $X^{23}Y^{23}Z^{13}$ of function of 4-valued variables X, Y, Z, V can be xlinked (in other words are **xlinkable terms or xlinkable cubes),** since

- for X: $\{1\} \cap \{2, 3\} = \phi$.
- for Y: $\{0, 1\} \cap \{2, 3\} = \phi$.
- for Z: $\{1, 3\} = \{1, 3\}$.
- for V: the truth values {0, 1, 2, 3} are the same.



and



(assuming 4-valued input logic)

Xlink of $X^{1}Y^{01}$ and $X^{23}Y^{23}$ is then $X^{123}Y^{23} \oplus X^{1}$.

In other words, all indices of literals must be either equal or disjoint. This type of xlinking for any two xlinkable terms will be called **primary xlinking.** It can easily be checked that primary xlink is not executable if at least one of the variables with different sets includes overlapping sets. For instance in

$X^{01}Y^{13}Z^{01}$

$X^{23}Y^{01}Z^{01}$

Variable Y has different values {1, 3} and {0, 1} that are overlapping, i.e. {1, 3} $\cap$ {0, 1} = {1}. Therefore, xlink of $X^{01}Y^{13}Z^{01}$ and $X^{23}Y^{01}Z^{01}$ is impossible.

Now we are able to formulate the definition of primary xlinking.

Definition 6. Let $C_S = X_1^{S_1} \dots X_n^{S_n}$ and $C_R = X_1^{R_1} \dots X_n^{R_n} \neq C_S$ be two cubes in which for each j, j = 1, ..., n it holds: $S_j = R_j$ or $S_j \cap R_j = \phi$. Let X_{j_i}, i = 1, ..., r be the variables for which $S_{j_i} \cap R_{j_i} = \phi$. Let X_{l_k}, k = 1, ..., n - r be the variables for which $S_{l_k} = R_{l_k}$.

The *primary xlink* of cubes C_S and C_R is defined by the following formula:

$$C_S \textcircled{1} C_R = \bigoplus_{i=1}^{r} \left[\left(X_{j_1}^{S_{j_1}} \cdots X_{j_{i-1}}^{S_{j_{i-1}}} X_{j_i}^{S_{j_i} \cup R_{j_i}} X_{j_{i+1}}^{R_{j_{i+1}}} \cdots X_{j_r}^{R_{j_r}} \right) \left(\prod_{k=1}^{n-r} X_{l_k}^{S_{l_k}} \right) \right]$$

Properties of primary xlinking.

1. Cubes C_S and C_R are of the same degree.
2. Cubes C_S and C_R are specified for the same variables (the complete sets of truth values are for the same variables in the cubes, in other words, the "-" symbols stand in the same positions).
3. r is the distance of C_S and C_R.
4. Primary xlinking of the distance two cubes corresponds to the formula 4G applied forward.

5.2. Secondary Xlinking.

Let us now introduce another type of xlinking. This new operation permits two terms of different degrees to be xlinked. As we have seen, the primary xlinking reduces the degree of the terms. By the use of the primary xlinking, together with the laws

1) A $\oplus$ 0 = A,
2) A $\oplus$ A = 0,
3) $\oplus$ is commutative and associative,

we will be able to formulate a **secondary xlinking** that xlinks terms of degrees differing by one.

Secondary xlinkable terms are two terms that satisfy these conditions:

- their degrees differ by one,
- there exist exactly one variable for which the truth value sets associated with one term are included in the truth value sets associated with the other term, for all other variables they are equal or disjoint.

For example, the terms $A^{0}B^{1}C^{012}D^{1}E^{1}$ and $A^{0}B^{0}C^{1}D^{1}E^{0}$ are secondary xlinkable since for variables A, B, C, D, E we have, respectively {0} = {0}, {1} $\cap$ {0} = ϕ, {0, 1, 2} $\supset$ {1}, {1} = {1}, {1} $\cap$ {0} = ϕ.

The number of disjoint sets is the distance of cubes.

Let us now consider an example of secondary xlinking. It uses the primary xlinking and the above three laws.

Let us assume the function with 3-valued variable C and 2-valued variables A, B, D, and E, shown in Fig. 5.

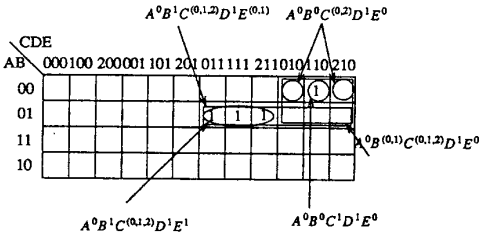


Fig. 5.

The secondary xlinking will be applied to:
$A^0B^1C^{012}D^1E^1 \oplus A^0B^0C^1D^1E^0$.

First, let us see that variable's C set {0, 1, 2} in $A^0B^1C^{012}D^1E^1$ includes set {1} of variable C in $A^0B^0C^1D^1E^0$. Let us then create a term that is adjacent with respect to variable C to the term $A^0B^0C^1D^1E^0$: this will be a term $A^0B^0C^{02}D^1E^0$.

Now we EXOR the previous EXOR sum with the zero term:
$A^0B^1C^{012}D^1E^1 \oplus A^0B^0C^1D^1E^0 \oplus (A^0B^0C^{02}D^1E^0 \oplus A^0B^0C^{02}D^1E^0)$.

Next we apply the fact that exoring is associative:
$A^0B^1C^{012}D^1E^1$
$\oplus (A^0B^0C^1D^1E^0 \oplus A^0B^0C^{02}D^1E^0) \oplus A^0B^0C^{02}D^1E^0$.

We xlink the terms in the parantheses:
$A^0B^1C^{012}D^1E^1 \oplus A^0B^0C^{012}D^1E^0 \oplus A^0B^0C^{02}D^1E^0$.

We xlink the first two terms:
$A^0B^{01}C^{012}D^1E^0 \oplus A^0B^1C^{012}D^1E^{01} \oplus A^0B^0C^{02}D^1E^0$
$= A^0D^1E^0 \oplus A^0B^1D^1 \oplus A^0B^0C^{02}D^1E^0$.

The above operation is illustrated in the Karnaugh map of Fig. 5. Remember that each cell covered by an even number of groups is a "zero" cell (false minterm) and one covered by an odd number of groups is a "one" cell (minterm).

The above sequence of transformations was shown for the sake of explanation but the execution of the secondary xlink is very straightforward in our implementation of the xlink operation. Its execution is therefore speedy.

We are now able to formulate the operation of secondary xlinking.

Definition 7. Let $C_S = X_1^{S_1} \cdots X_i^{S_i} \cdots X_n^{S_n}$ and $C_R = X_1^{R_1} \cdots X_i^{R_i} \cdots X_n^{R_n}$ be two cubes for which there exists exactly one variable X_i such that $S_i \supset R_i$ and other variables have disjoint or equal truth value sets.
The *secondary xlinking* of cubes C_S and C_R
$C_S \textcircled{\scriptsize D} C_R = C_N \oplus (C_S \textcircled{\scriptsize 1} (C_R \textcircled{\scriptsize 1} C_N))$
where
$C_N = X_1^{R_1} \cdots X_i^{S_i - R_i} \cdots X_n^{R_n}$.

In the current algorithm the secondary xlinking is used on cubes that have distance 0, 1, and 2 only, but it can be very easily expanded for cubes of arbitrary distance. It is obvious from the above definition that the difference of cubes' degrees is one. Secondary xlinking of distance two cubes corresponds to the formula 4H applied forward.

5.3. Unlinking.

Unlinking operations are inverse to the xlinking operations. They are necessary in the algorithm to permit for iteration of xlinking in order to find a global minimum or at least some local minimum close to it. There are now two types of unlinking operators in the system:

- *primary unlinking,*
- *secondary unlinking.*

The primary unlinking is an inverse operation to the primary xlinking of cubes of distance two. It corresponds to the formula 4H applied in the reverse direction.

The secondary unlinking is an inverse operation to the secondary xlinking of cubes of distance two. It corresponds to the formula 4I applied forward.

6. ALGORITHM.

The algorithm currently used in our program is quite simple. The idea is to carry out all primary and secondary xlinks possible, in some reasonable order, giving priority to xlinking least distant groups first. Next groups are decreased by primary and secondary unlinking operations, the small groups are re-ordered and the procedure is iterated. Since there are many xlinks and unlinks possible, the algorithm uses permutations in successive calls of xlinking and unlinking procedures, in order to provide searching the larger space of solutions (functions F). Random reordering of cubes in F serves the same goal. PRIMARY_XLINK(F, DISTANCE) executes primary xlinks of distance DISTANCE in array F.

Algorithm to Minimize MRESP Forms.

Input: Arrays ON and DC of disjoint cubes for a single output multi-valued input function.

1. F := ON $\cup$ DC.
2. SOLUTION := ON, MIN_COST := COST(ON).
3. Sort randomly F.
4. DISTANCE := 1, F := PRIMARY_XLINK(F, DISTANCE),
   DISTANCE := 2, F := PRIMARY_XLINK(F, DISTANCE).
5. Repeat 4 until no primary xlinks of distance 2 can be executed.
6. F1 := ON - { $cube_i \in$ ON | $cube_i \subseteq$ DC }
7. If COST(F1) < MIN_COST
   then (SOLUTION := F1; MIN_COST := COST(F1)).
8. DISTANCE := DISTANCE + 1,
   F := PRIMARY_XLINK(F, DISTANCE).
9. If primary xlink was executed go to 4 else go to 8.
10. Repeat 8,9 until DISTANCE > MAX_DISTANCE.
11. Execute steps 6, 7.
12. DISTANCE := 1, F := SECONDARY_XLINK(F, DISTANCE).
13. Execute steps 6, 7.
14. DISTANCE := 2, F := SECONDARY_XLINK(F, DISTANCE).
15. Execute steps 6, 7.
16. If secondary xlink was executed in 12 or 14 then go to 4.
17. F := UNLINKING(F).
18. Execute steps 6, 7.
19. If time has not exceeded the allotted time limit then go to 3
    else return SOLUTION and its cost.

It is important to do xlinking on terms of high degree first. This allows for the results of the first xlinks to be compared for xlinking to groups of the same degree before those groups are xlinked to groups of lower degree. After performing secondary xlinks, it is required to check for primary xlinks again, because the secondary xlinks may contain primary xlinkable groups. Currently only some of the possible unlinks are executed, which still proved to be sufficient to improve the quality of solutions.

In the case of multi-output function, the function is first transformed from multi-output array to arrays ON and DC of disjoint cubes, using one of the methods mentioned in section 3. After execution of the above algorithm, the resultant MRESP is separated into the component single-output functions by substituting all possible values to the multiple-valued variable X_n whose values 0, ..., m-1 correspond to the component outputs.

Example 4. Let us realize the function from Example 3 as the mixed-polarity EXOR form. The algorithm returns the solution f = $\bar{a}b + c\bar{d} + \bar{c}\bar{d}$ from Fig. 6a. The respective circuit is presented in Fig. 6b.

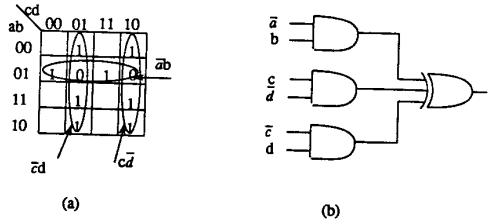


Fig. 6.

Example 5. Let us realize function f from previous examples assuming the same variable pairing as in Examples 2, 3 and 4. X = (a, b), Y = (c, d). Two solutions are found:

f = $X^1 \oplus Y^{12}$ (Fig. 7a)

f' = $X^{023} \oplus Y^{03}$ (Fig. 8a)

The realization of f in EXPLA is shown in Fig. 7b, The realization of f' in EXPLA is shown in Fig. 8b. Figures 7c and 8c present the standard cell realizations of these functions using the function generators for X and Y.

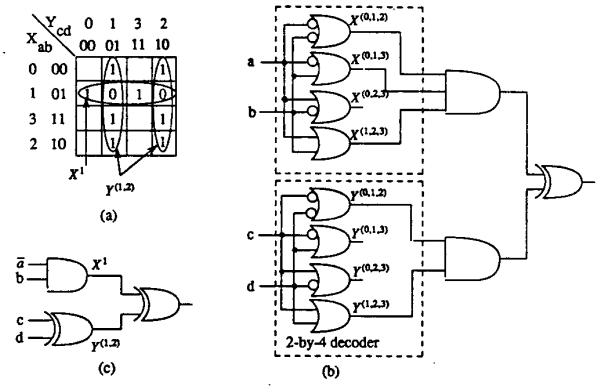


Fig. 7.

7. CONCLUSION

A new concept of MRESP along with the method to minimize such forms has been introduced. Such a network concept was not yet introduced in the literature. There are also no algorithms to minimize such networks. Even for the particular case of MRESP - the mixed-polarity binary-output EXOR forms [22,30], no algorithms have been published until now for the multi-output and incompletely specified functions. The method was tested on many examples and the computational complexity and quality of the algorithm will be a subject of a separate paper.

8. ACKNOWLEDGMENT

The authors would like to thank the anonymous referees for their criticizm that helped to improve this paper.

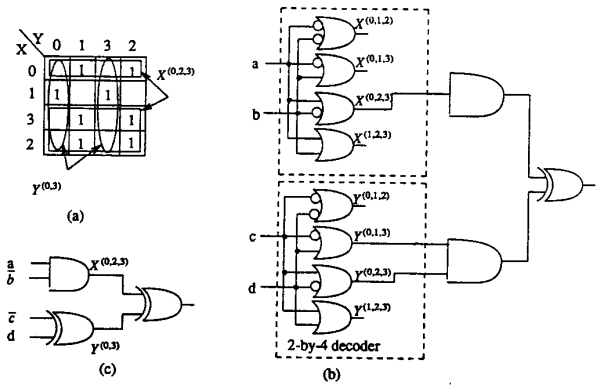


Fig. 8.

9. LITERATURE

[1] Akers, S.B.: "On a theory of Boolean functions", *J. SIAM*, Vol. 7., pp. 487-498, December 1959.

[2] Bender, E., Butler, J., and H. Kerkhoff: "Comparing the sum verses the max for use multiple-valued PLA's", *Proc. of ISMVL-85.*

[3] Bennetts, R.G., and D. Lewin: "Fault diagnosis of digital systems - a review", *Comput. J., Vol. 14., pp. 199-206, 1971.*

[4] Besslich, Ph.W.: "Efficient Computer Method for EXOR Logic Design", *Proc. IEE*, Vol. 130, Part E, CDT, No. 6., pp. 203-206, 1983.

[5] Besslich P.: "Heuristic minimization of multiple-valued logic functions: a direct cover approach", *IEEE Trans. on Comp., February 1986, pp. 134-144.*

[6] Bhavsar, D., and R.W. Heckelman: "Self-testing by polynomial division", *Proc. 1981 IEEE Test Conference*, pp. 208-216, 1981.

[7] Bioul, G., Davio, D., and J.P. Deschamps: "Minimization of ring-sum expansions of Boolean functions", *Philips Research Report*, Vol. 28, pp. 17-36, 1973.

[8] Brayton, R.K. Hachtel, G.D. McMullen, C.T. and A.L. Sangiovanni-Vincentelli: *Logic Minimization Algorithms for VLSI Synthesis*, Boston, MA, Kluwer, 1984.

[9] Brayton, R.K., Camposano, R., De Micheli, G., Otten, R.H.J.M., and J. Van Eijndhoven: "The Yorktown Silicon Compiler System", Chapter 7 in Gajski, D., (ed), *Silicon Compilation*, 1987.

[10] Calingaert, P.: "Switching function canonical forms based on commutative and associative binary operations", *AIEE Trans.* Vol. 79, pp. 808-814, January 1961.

[11] Carter, W.C., Wadia, A.B., and D.C. Jessep, Jr.: "Implementation of Checkable Acyclic Automata by Morphic Boolean Functions", *Proc. of the Symp. on Computers and Automata*, Polytechnic Institute of Brooklyn, p. 645, 1971.

[12] Chan, A.H.: "Using decision trees to derive the complement of a binary function with multiple-valued inputs," *IEEE Trans. on Comp.*, Vol. C-36, pp. 212 - 214, Febr. 1987.

[13] Ciesielski, M.J. Perkowski, M.J., and S. Yang: "PALMINI-MV: An Approach to Multiple-Valued Logic Minimization Based on Graph Coloring". *Technical Report. Univ. of Massachusetts.*

[14] Cohn, M.: "Inconsistent canonical forms of switching functions", *IRE Trans. Electron. Comput.*, Vol. EC-11, p. 284, April 1962.

[15] Davio, M.: "Ring-Sum Expansions of Boolean Functions", *Proc. of the Symp. on Computers and Automata*, Polytechnic Institute of Brooklyn, April 13-15, pp. 411-418, 1971.

[16] Davio, M., Deschamps, J.P., and A. Thayse: *"Discrete and Switching Functions"*, McGraw-Hill Book Co., Inc., New York, 1978.

[17] DeMicheli, G., Brayton, R.K. and A. Sangiovanni-Vincentelli: "Optimal State Assignment for Finite State Machines", *IEEE Trans. on CAD*, Vol. CAD-4, No. 3, July 1985, pp. 268-284.

[18] DeMicheli, G.: "Symbolic Design of Combinational and Sequential Logic Circuits Implemented by Two-Level Logic Macros", *IEEE Trans. on CAD*, Vol. CAD-5, No. 4., Oct. 1986, pp. 597-616.

[19] Dueck, G., and D. Miller: "A 4-valued PLA using a Modsum", *Proc. of ISMVL-86.*

[20] Dueck, G., and D. Miller: "A direct cover multiple-valued minimization using the truncated sum", *Proc. of ISMVL-87.*

[21] Even, S., Kohavi, I., and A. Paz: "On minimal modulo-2 sum of products for switching functions", *IEEE Trans. on Electron. Computers*, pp. 671-674, October 1967.

[22] Fleisher, H., Tavel, M., and J. Yeager: "Exclusive-OR representations of Boolean functions", *IBM J. Res. Develop.*, Vol. 27, pp. 412-416, July 1983.

[23] Fleisher, H., Tavel, M., and J. Yeager: "A Computer Algorithm for Minimizing Reed-Muller Canonical Forms", *IEEE Trans. on Computers*, Vol. C-36, No. 2, February 1987.

[24] Fujiwara, H.: *"Logic Testing and Design for Testability"*, Computer System Series, The MIT Press, 1986.

[25] Green, D.H., and K.R. Dimond: "Polynomial representation of non-linear feedback shift-registers", *Proc. IEE*, Vol. 117, No. 1., pp. 56-60, 1970.

[26] Green, D.H., and I.S. Taylor: "Multiple-valued switching circuit design by means of generalized Reed-Muller expansions", *Digital Processes*, No. 2, pp. 63-81, 1976.

[27] Green, D.H., and M. Edkins: "Synthesis procedures for switching circuits represented in generalized Reed-Muller form over a finite field", *Computer and Digital Techniques*, Vol. 1., No. 1., pp. 22-35, 1978.

[28] Green, D.: *"Modern Logic Design"*, Electronic Systems Engineering Series, 1986.

[29] Hayes, J.P.: "On modifying logic networks to improve their diagnosability", *IEEE Trans. Comput.* Vol. C-23, No. 1, pp. 56-62, 1974.

[30] Helliwell, M., and M.A. Perkowski: "A Fast Algorithm to Minimize Multi-Output Mixed-Polarity Generalized Reed-Muller Forms", *Proc. 25-th ACM/IEEE Design Automation Conference*, paper 28.2, pp. 427-432, June 12- June 15, 1988.

[31] Helliwell, M., and M.A. Perkowski: "New Algorithms and Hardware Accelerators for Exact and Aproximate Minimization of Mixed Polarity Reed-Muller Forms". *Diades Group Report* 34/1988, PSU.

[32] Helliwell, M., and M.A. Perkowski: "Application of Secondary Crosslinking Operation to Minimize Mixed Polarity Reed-Muller Forms". *Diades Group Report* 32/1988, PSU.

[33] Hong, S.J., Cain, R.G., and D.L. Ostapko: "MINI: A heuristic approach for logic minimization", *IBM J. Res. Develop.*, Vol. 18, pp. 443 - 458, Sept. 1974.

[34] Hurst, S.I.: *"Logical processing of digital signals"*, Edward Arnold, London: Crane-Russak, N.Y., 1978.

[35] Kodandapani, K.L.: "A note on easily testable realizations for logical functions", *IEEE Trans. Comp.*, Vol. C-23, pp. 332-333, 1974.

[36] Kodandapani, K.L., and R.V. Setlur: "A note on minimum Reed-Muller canonic forms of switching functions", *IEEE Trans. Comp.*, Vol. C-26, pp. 310-313, 1977.

[37] Kuo, Y.S., and ?. Chau, "Generating essential primes for a boolean function with multiple-valued inputs", *Proceedings of DAC' 86*, 1986.

[38] Kuo, Y.S.: "Generating essential primes for a boolean function with multiple-valued inputs," *IEEE Trans. on Comp.*, Vol. C-36, pp. 356 - 359, March 1987.

[39] Lee, E.B., and M.A. Perkowski: "Concurrent Minimization and State Assignment of Finite State Machines", *Proceedings of the 1984 Intern. Conf. on Systems, Man, and Cybernetics*, IEEE, Halifax, Nova Scotia, Canada, October 9-12, 1984.

[40] Mukhophadhyay, A., and G. Schmitz: "Minimisation of exclusive-OR and logical equivalence switching circuits", *IEEE Trans. Comp.*, Vol. C-19, No. 2. , pp. 132-140, February 1970.

[41] Muller, D.E.: "Application of Boolean algebra to switching circuit design and to error detection", *IRE Trans. Electron. Comp.*, Vol EC-3, pp. 6-12, September 1954.

[42] Page, E.W.: "Minimally testable Reed-Muller canonical forms", *IEEE Trans. Comput.*, Vol. C-29, No. 8., pp. 746-750, 1980.

[43] Papakonstantinou, G.: "Minimization of modulo-2 sum of products", *IEEE Trans. on Computers.*, Vol. C-28, pp. 163-167, February 1979.

[44] Perkowski, M.: *"Digital Design Automation System DIADES. Documentation"*, Dept. EE, Portland State University, Portland, OR, 1988.

[45] Pradhan, D.K.: "Universal test sets for multiple fault detection in AND-EXOR arrays", *IEEE Trans. on Comput.*, Vol. C-27, No.2., pp. 181-187, 1978.

[46] Pomper, G., and J. Armstrong: "Representation of multivalued functions using the direct cover approach", *IEEE Trans. on Comp., Sept. 1981, pp. 674-679.*

[47] Pradhan, D.K.: *"Fault-Tolerant Computing. Theory and Techniques. Vol. I."* Prentice-Hall, 1987.

[48] Ramamoorthy, C.V.: "Procedures for minimization of "exclusive or" and "logical equivalence" switching circuits", *IEEE Symp. on Switching Circuit Theory and Logical Design*, pp. 143-149, October 1965.

[49] Reddy, S.M.: "Easily testable realization for logic functions", *IEEE Trans. Comput.*, Vol. C-21, pp. 1183-1188, Nov. 1972.

[50] Reed, I.S.: "A class of multiple-error-correcting codes and their decoding scheme", *IRE Trans. Inf.Th.*, Vol. PGIT-4, pp. 38-49, 1954.

[51] Robinson, J.P., and C.L. Yeh: "A Method for modulo-2 Minimization", *IEEE Trans. on Computers*, Vol. C-31, pp. 800-801, August 1982.

[52] Roth, P.: *"Computer Logic, Testing and Verification"*, Rockville, MD: Computer Science, 1980.

[53] Rudell, R.L., and A.L. Sangiovanni-Vincentelli: "ESPRESSO-MV: algorithms for multiple-valued logic minimization, *Proc. IEEE Custom Integrated Circuits Conf.*, 1985.

[54] Rudell, R.: *"Multiple-Valued Logic Minimization for PLA Synthesis"*, M.S. Report, June 5, 1986. University of California, Berkeley California 94720.

[55] R.L. Rudell, and A.L. Sangiovanni-Vincentelli, "Multiple-valued minimization for PLA optimization," *Proc. Intern. Symp. on Multiple-Valued Logic*, pp. 198-208, May 26-28, Boston, MA, 1987.

[56] Saluja, K.K., and S.M. Reddy: "Fault detecting test sets for Reed-Muller canonic networks", *IEEE Trans. Comput.*, Vol. C-24, No. 10., pp. 995-998, 1975.

[57] Saluja, K.K., and E.H. Ong: "Minimization of Reed-Muller canonic expansion", *IEEE Trans. Comput.*, Vol. C-28, pp. 163-167, February 1979.

[58] Sarabi, A., and M.A. Perkowski: "New Algorithms for Exact and Aproximate Minimization of Fixed Polarity Reed-Muller Forms for Incompletely Specified Boolean Functions". *Diades Group Report* 35/1988, PSU.

[59] Sasao, T.: "An application of multiple-valued logic to a design of Programmable Logic Arrays", *Proc. 8th Intern. Symp. on Multiple-Valued Logic (ISMVL)*, 1978.

[60] Sasao, T.: "Multiple-valued decomposition of generalized boolean functions and the complexity of programmable logic arrays," *IEEE Trans. Comput.*, Vol. C-30, pp. 635-643, Sept. 1981.

[61] Sasao, T.: "Input variable assignment and output phase optimization of PLA's," *IEEE Trans. Comput.*, Vol. C-33, pp. 879 - 984, Oct. 1984.

[62] Sasao, T.: "Input Variable Assignment and Output Phase Optimization of PLA's", *IEEE Trans. on Comp.*, Vol. C-33, No. 10, pp. 879-894, October 1984.

[63] Sasao, T.: "An algorithm to derive the complement of a binary function with multiple-valued inputs," *IEEE Trans. on Comp.*, Vol. C-34, pp. 131 - 140, Febr. 1985.

[64] Sasao, T.: "MACDAS: Multi-level AND-OR circuit synthesis using two-variable function generators", *23-rd Design Automation Conference*, Las Vegas, pp. 86-93, June 1986.

[65] Sasao, T.: "Bounds on the average number of products in the minimal Sum-of-Products expressions for multiple-valued input two-valued output functions," *Proc. Intern. Symp. on Multiple-Valued Logic*, pp. 260-267, May 26-28, Boston, MA, 1987.

[66] Schmokler, M.S.: "Mod-2 Sums of Products", *IEEE Trans. on Comp.*, Vol. C-18, No. 10., October 1969.

[67] Trimalai, P., and J. Butler: "Analysis of minimization techniques for multiple-valued functions", *Proc. of ISMVL-88.*

[68] Wu, X., Chen, X., and S.L. Hurst: "Mapping of Reed-Muller coefficients and the minimisation of Exclusive-OR switching functions", *Proc. IEE*, part E, vol. 129, pp. 15-20, January 1982.