COMBINATIONAL MULTIPLE-VALUED CIRCUIT DESIGN BY GENERALIZED DISJUNCTIVE DECOMPOSITION

Tatiana Kalganova

Belarusian State University of Informatics and Radioelectronics
Laboratory of Image Processing and Pattern Recognition.
Brovky St., 6, 220600, Minsk, Republic of Belarus,
Phone: (0375172) 491-981, Fax: (0375172) 495-106,
E-mail: jack@expert.belpak.minsk.by or pottosina@risq.belcaf.minsk.by

Abstract - A design of multiple-valued circuits based on the multiple-valued programmable logic arrays (MV PLA’s) by generalized disjunctive decomposition is presented. Main subjects are 1) Generalized disjunctive decomposition of multiple-valued functions using multiple-terminal multiple-valued decision diagrams (MTMDD’s); 2) Realization of functions by MV PLA-based combinatorial circuits.

Key words - Multiple-valued logic, generalized disjunctive decomposition; combinational multiple-valued circuits (MV circuits).

1. INTRODUCTION

The functional decomposition theory developed by Ashenhurst, Curtis, Roth, Karp and etc. and applied for binary and r-valued functions is employed in the design and testing of logic circuits [1, 2, 3]. The decomposition technique is to break the many variable function into several functions with fewer variables. These functions can be designed independently, and are relatively easier to design.

Direct application of classical decomposition theory to design the practical MV circuits involves two problems:

1) the usefulness of the decomposition for a multiple-valued functions.

2) the computation time and the memory requirements;

As for the first problem, the generalized disjunctive decomposition allows to increase the number of decomposable functions and solve this problem [9].

In this paper we develop an algorithm to solve the second problem. The necessary condition developed in [4] for switching functions is generalized for the multiple-valued case. We use MTMDD instead of decomposition tables. It allows to investigate (n-1) different partition of X for one MTMDD. MTMDD permits to examine the functions with many variables. The structure of Shannon tree generalized for r-valued functions is proposed too. MTMDD is built from multiple-valued Shannon tree and has the similar structure like in suggested tree.

The rest of paper is organized as follows. In section 2 we start with describing the basic notations and definitions. We then discuss the algorithm to find the good decomposition using MTMDD (Section 3). Here, the Shannon tree is also generalized for r-valued functions and the properties of MTMDD are considered too. Design of MV PLA-based combinatorial circuits is represented in Section 4. Section 5 is summary of this paper.

2. DEFINITION AND NOTATIONS

Functional decomposition of r-valued function is used for any algebraic representation of given function. It allows to choose any functionally complete algebra for analysis. Using the algebra developed by Rosser [5] allows to investigate the disjunctive decomposition of r-valued function. This algebra is chosen for study because of its simplicity at the realization of r-valued functions and their system on the combinational MV circuits [6].

Definition 1. The Rosser algebra is defined as follows: Let x and y be the r-valued variables. Then

(a) \( x \lor y = \text{MAX}(x, y) \) is the disjunction operator.

(b) \( x \land y = x \cdot y = \text{MIN}(x, y) \) is the conjunction operator.

(c) \( x' = \begin{cases} r-1, & x = s, \\ 0, & x \neq s, \end{cases} \)

where \( s \in \{0, 1, \ldots, r-1\} \), \( x' \) is the literal of a \( x \) variable.

Definition 2. Let \( X \) is the set of r-valued input variables. \( \{X_1, X_2\} \) is a partition of \( X \) when \( X_1 \cap X_2 = \emptyset \) and \( X_1 \cup X_2 = X \).

To represent r-valued functions with minimal expressions, the following notations are adopted:

\( E_r : \) the set of constants is an r-valued Rosser algebra, \( E_r = \{0, 1, \ldots, r-1\} \)

---

1 This work has been supported in part by Found of Fundamental Investigations (Republic of Belarus), Grant №M96-094.
\( X \) : the set of \( n \) \( r \)-valued input variables \{\( x_1, x_2, \ldots, x_n \}\), where \( n \) is the number of elements of \( X \).

\[ |X|: \text{the number of elements of the set } X \]

\[ \sigma : (\sigma_1, \sigma_2, \ldots, \sigma_n), \text{ where } \sigma_i \in \{0, 1, \ldots, r\}, i=1,2,\ldots, n. \]

\[ \tau : (\tau_1, \tau_2, \ldots, \tau_n), \in \sigma, \text{ where } \tau_i \in \{0, 1, \ldots, n_1\}, \]

\[ \eta : (\eta_1, \eta_2, \ldots, \eta_n), \in \sigma, \text{ where } \eta_i \in \{0, 1, \ldots, n_2\}. \]

**Definition 3.** Let \( f(X) \) be an \( r \)-valued function and \( \{X_i, X_2\} \) is a partition of \( X \). Then the projection of \( f(X) \) over \( X_i=\tau \) is the value of \( f(X) \) evaluated with \( X_i=\tau \).

**Definition 4.** Let \( f(X) \) be an \( r \)-valued function and \( \{X_i, X_2\} \) is a partition of \( X \). Let \(|X_1|=n_1 \) and \(|X_2|=n_2\). Then the expansion of \( f(X) \) over \( X_i \) is given by

\[ f(X) = \bigvee_{\tau=0}^{r^{n_1}-1} x_1^\tau_1 x_2^\tau_2 \ldots x_{n_1}^\tau_{n_1} f(\tau, X_2) \] (1)

and the projection of \( f(X) \) over \( X_i=\tau \) is an \( r \)-valued function such that

\[ f(\tau, X_2) = \bigvee_{\eta=0}^{r^{n_2}-1} x_1^\eta_1 x_2^\eta_2 \ldots x_{n_2}^\eta_{n_2} f(\tau, \eta) \] (2)

where \( f(\tau, \eta) \) is the value of \( f(X) \) evaluated with \( X_1=\tau \) and \( X_2=\eta \).

Equation (1) defines the Shannon expansion of \( r \)-valued function in Rosser algebra.

**Definition 5.** An \( r \)-valued function \( f(X) \) is said to have a generalized disjunctive decomposition with respect to \( X_i \) if there exist \( r \)-valued functions \( h_1, h_2, \ldots, h_k \) and \( g \) such that

\[ f(X) = g(h_1(X_1), h_2(X_2), \ldots, h_k(X_k), X_2) \] (3)

where \( \{X_1, X_2\} \) is a partition of \( X \).

An \( r \)-valued function \( f(X) \) is said to have a simple disjunctive decomposition if \( k=1 \), in other words

\[ f(X) = g(h(X)), X_2 \] (4)

where \( \{X_1, X_2\} \) is a partition of \( X \).

The number of elements in the bound and free set will be denoted by \( n_1 \) and \( n_2 \), respectively. The decomposition is said to be trivial if \( n_1 \) is 1 or \( n \) and if \( k=r^{n_1} \). A function that has a nontrivial generalized disjunctive decomposition is said to be decomposable.

**Definition 6.** An \( r \)-valued function \( f(X) \) is said to have a multiple disjunctive decomposition if there exist \( r \)-valued functions \( g, t \) and \( h \) such that

\[ f(X) = g(h(X_1), t(X_2), X_3) \] (5)

and iterative disjunctive decomposition if

\[ f(X) = g(h(t(X_1), X_2), X_3) \] (6)

where \( \{X_1, X_2, X_3\} \) is a partition of \( X \).

**Definition 7.** Let \( f(X) \) be a completely specified \( r \)-valued function and \( \{X_i, X_2\} \) is a partition of \( X \). Let \(|X_1|=n_1 \) and \(|X_2|=n_2\). The decomposition table of \( f(X) \) is the truth table of \( f(X) \) with \( r^{n_1} \) columns defined by set \( X_1 \) and \( r^{n_2} \) rows defined by set \( X_2 \).

**Definition 8.** The number of different column patterns in the decomposition table is called a column multiplicity of the decomposition, and denoted by \( \mu \).

Note that the column multiplicity defines the type of disjunctive decomposition. An \( r \)-valued function \( f(X) \) is said to have a simple disjunctive decomposition if \( \mu \) is less or equal to \( r \) and a generalized disjunctive decomposition if \( \mu \) is less or equal to \( r^{n_1} \), where \( n_1 \) is the number of elements in set \( X_1 \).

3. **GENERALIZED DISJUNCTIVE DECOMPOSITION USING MTMDD**

Generalized functional decomposition of switching (binary) functions using binary decision diagrams (BDD) was studied by T. Sasao [4]. For the same reason like in [4] the algorithm for an \( r \)-valued functions is developed. In this section we introduce a multiple-valued Shannon tree and MTMDDs obtained from Shannon tree, present a decomposition searching algorithm that apply MTMDD.

3.1 **MULTIPLE-VALUED SHANNON TREE**

By applying the multiple-valued Shannon expansion recursively to an \( r \)-valued function, we can represent a logic function by an expansion tree.

Fig. 2 shows an example of an expansion tree for a 3-variable 3-valued function, where the symbol \( S \) denotes the Shannon expansion. This tree is called a multiple-valued Shannon tree. The terminal nodes represent \( r \)-valued constants \( 0, 1, \ldots, r-1 \). Each edge has a literal of a variable as a label. A product (conjunction) of the literals from the root node to a terminal node represents a product term that is \( MIN \) of all literals composing it. For example, the left most path defines the product \( f_{000}x_1'h_2'x_3' \), where \( f_{000} \) is the value of \( f(X) \) evaluated with \( x_1=x_2=x_3=0 \). This tree shows the Eq.(2) for \( f(X) \).
A product term including all variables is called a minterm, and an expression consisting only of minterms is called a sum-of-product expression. Note that the products having zero coefficients disappear. Thus, the number of non-zero coefficients equals to the number of products in the expression.

3.2 MULTIPLE-TERMINAL MULTIPLE-VALUED DECISION DIAGRAMS

A multiple-valued decision diagram (MDD) is a generalization of a binary decision diagram, in which an internal node may have more than two children. An MDD having more than two kinds of terminal nodes (e.g., 0, 1, ..., r-1) is called a multiple-terminal MDD [10, 11].

The MTMDD describing r-valued logic functions is obtained by simplifying multiple-valued Shannon tree, using the following rules:

1. If two sub-graphs are isomorphic, delete one, and connect the severed edge to the remaining sub-graph (Fig. 3).

2. Delete the Shannon node if its r descendent nodes are identical (Fig. 4). Delete the 0-terminal node if f(X) is given by multiple-valued Shannon tree (Fig. 5).

Fig. 2 Multiple-valued Shannon tree

Fig. 3 Merging isomorphic sub-graphs

Fig. 4. Elimination of the multiple-valued Shannon node

Fig. 5. Elimination of the 0-terminal node

Fig. 6. Multiple-valued Shannon tree for 3-valued 3-variable function
there are isomorphic sub-graphs and Shannon nodes having $r$ descendent identical nodes. If we cannot simplify the graph anymore, then such a MTMDD is called reduced MTMDD (Fig. 9).

An ordered MTMDD is a MDTMD such that the input variables appear in a fixed order in all the paths of the graph, and that no variable appears more than once in a path.

Definition 9. An ordered MTMDD is a Quasi-Reduced MTMDD if every path from the root to the terminal nodes involves all variables, and has no isomorphic sub-graphs in the same level.

Definition 10. The path function of a node in MTMDD is an $r$-valued input 2-valued output function and represents the conditions that there is a path from root to the node. The sub-path function of a node in MTMDD is an 2-valued input $r$-valued output function and represents the conditions that there is a path from a node to the terminal node. A sub-path function of a node is the projection of $f(X)$ over $X_i=q, f(q, X_j)$.

3.3 THE DECOMPOSITION SEARCHING ALGORITHM

In this section we will describe an algorithm to find all the generalized disjunctive decompositions for $r$-valued functions. The algorithm consists of the two phases:

1. A MTMDD for given function is generated.
2. This MTMDD are tested for good decomposition.

Lemma. Let $\{X_1, X_2\}$ be a partition of $X$ and the quasi-reduced MTMDD for $f(X)$ is partitioned into two blocks as shown in Fig. 10. Let $q_i$ ($i=1, \ldots, t$) be the path functions of the nodes in the lower block that are adjacent to the boundary of the blocks. Then, $q_i \land q_j = 0$ ($i \neq j$) and $q_i \lor q_j = r-1$.

(Proof) Suppose that Fig. 10 is a complete $r$-valued decision tree of $n$ variables. Then $q_i$, $q_j$, ..., $q_t$ denote minterms of $n_i=|X_1|$ variables. These minterms satisfy the conditions of Lemma. Because of $q_i$ ($i=1, \ldots, t$) is an $r$-valued input binary function. This property is kept even if the isomorphic sub-trees are merged in the quasi-reduced MTMDD. (Q.E.D.)

Example. Let us consider the quasi-reduced MTMDD shown in Fig. 8. This MTMDD is partitioned into two blocks such that the upper block contains the nodes for $X_1$ and the lower block contains the nodes for $X_2$, where $X_1=(x_1, x_2)$ and $X_2=(x_3)$. The paths functions of the nodes are:
For the left \( x_1 \) node: \( q_1 = x_1^0 x_2^2 \lor x_1^2 \)
For the right \( x_1 \) node: \( q_2 = x_1^0 x_2^2 \lor x_1^4 \)

It is clear that \( q_1 \) and \( q_2 \) satisfies the conditions of Lemma.

**Theorem.** Let \( \{X_i, X_j\} \) be a partition of \( X \). Suppose that the reduced ordered MTMDD for \( f(X) \) is partitioned into two blocks such as shown in Fig. 10. Let \( t \) be the number of the nodes in the lower block that are adjacent to the boundary of the two blocks, and \( \mu \) be the column multiplicity of the decomposition \( f(X) = g(h_t(X_1),..., h_1(X_2)) \). Then, \( t \geq \mu \).

**(Proof)** Consider quasi-reduced MTMDD instead of reduced ordered MTMDD. Let \( q_1, q_2, ..., q_t \) be the boundary nodes in the lower block (1). Let the path functions of the nodes be \( q_1(X_1), q_2(X_1), ..., q_t(X_1) \). By the property of quasi-reduced MTMDD, \( q_i, (i=1, ..., t) \) are all disjoint, and for the \( \alpha \) and \( \beta \) inputs of lower block such that \( \alpha, \beta \in q_i \), \( g(h_t(\alpha),..., h_1(\alpha), X_2) = g(h_t(\beta),..., h_1(\beta), X_2) \). This implies that \( t \geq \mu \). (2) Let Fig. 10 be the multiple-valued Shannon tree of an \( r \)-valued \( n \)-variable function. Consider the MTMDD which is obtained from multiple-valued Shannon tree by merging isomorphic sub-trees in the lower block. Let \( m \) be the number of the nodes in the lower block that are adjacent to the boundary of the blocks. Then, by the definition of the column multiplicity \( m \geq \mu \). The quasi-reduced MTMDD is obtained by further merging this MTMDD. Note that the number of the nodes will not increase by the reduction of the MTMDD. Thus, \( m \geq \mu \). The reduced ordered MTMDD is obtained by further reducing the quasi-reduced MTMDD. Note that in two MTMDDs \( m \) is the same. Theorem is proved from (1) and (2). (Q.E.D.)

**Example.** Fig. 11 shows the three different partitions of 3 valued 5 variable function:
1) \( X_1 = \{x_1, x_2\}, X_2 = \{x_3, x_4, x_5\} \)
2) \( X_1 = \{x_1, x_2, x_3\}, X_2 = \{x_4, x_5\} \)
3) \( X_1 = \{x_1, x_2, x_3, x_4\}, X_2 = \{x_5\} \)

By Theorem, the column multiplicity for these decompositions are four, three and two, respectively. The first partition gives the generalized disjunctive decomposition, the second and third partitions - simple disjunctive decomposition.

### 3.4 Design of the MV PLA - Based Combinatorial Circuits

In this section, we deal with design of MV PLA-based combinational circuits using generalized disjunctive decomposition.

---

**Fig. 11 Determination of column multiplicity using MTMDD**

Let us solve the following problem. Let \( f(X) \) be an \( r \)-valued function of \( n \) variables. Let \( \{X_1, X_2\} \) be a partition of \( X \) and \( f(X) = g(h_t(X_1),..., h_1(X_2)) \) be generalized disjunctive decomposition, where \( k \) is the fixed number. Reduce the size and the number of MV PLA’s in combinational circuit that realize given function.

In order to solve this problem, we use the following.

To obtain the column multiplicity \( \mu \) (\( i=1,..., \sum_{i=1}^{n-2} C_n^i \) we use reduced ordered MTMDD and Theorem. For an \( n \) variable \( r \)-valued function, we have to obtain the column multiplicities for \( \sum_{i=1}^{n-2} C_n^i \) different partitions of \( X \). This is be done efficiently by permuting the order of the input variables in MTMDDs. To realize efficient MV circuit in terms of minimal size, we use the following algorithm.

**Algorithm 3.1.**

**Input data:** Reduced ordered MTMDD

**Output data:** Combinational MV PLA- based circuit

1. Obtain \( \mu \).
2. When \( \mu = r^0 \) = 1, \( X \) does not support function \( f(X) \) and realize the function by the circuit in Fig. 12 (a).
3. When \( \mu = r^r \), realize the function by the circuit in Fig. 12 (b) using simple disjunctive decomposition.
4. When \( \mu = r^r \), realize the function by the circuit in Fig. 12 (c) using generalized disjunctive decomposition.

...
6. Otherwise ($\mu > r^k$), execute the generalized disjunctive decomposition for $g(h_1(X_i),...,h_k(X_i),X_2)$ function and realize $f(X)$ by the circuit shown in Fig. 12 (e). Here, the iterative disjunctive decomposition is used.

Note, that the $X_i$ and $X_2$ set is formed for the minimal column multiplicity $\mu$ obtained at the execution of the searching algorithm.

**ALGORITHM 3.2.**

**Input data:** Reduced ordered MTMDD

**Output data:** Minimal column multiplicity, $\mu$.

1. For $f(X)$ obtain the column multiplicities $\mu_i$ ($i=1,...,(n-2)!$) for all permutations of input variables.
2. Execute the minimal column multiplicity.
3. Derive the partition of $X$, using the smallest multiplicity.
4. According to the column multiplicity, use one of the realizations in Fig. 12. For each column pattern, assign a different $r$-valued vector of $k$ elements. Let $h_1, h_2, ..., h_k$ be sub-functions obtained by the generalized disjunctive decomposition.
5. When $\mu > r^k$, realize function $g$ by using step 1 through step 5 until $\mu$ is equal to or smaller than $r^k$.

4. **Summary**

A decomposition searching algorithm has been presented. This algorithm is an extension of the switching function decomposition algorithm given in [4]. The direct extension of the binary version requires a definition of Shannon tree and decision diagrams produced from this tree for $r$-valued functions. The method is easy to implement on a digital computer. The analysis of the time complexity of proposed algorithm is not considered because the size of paper is limited.

Future work is carried out in frame of generalization the developed algorithm for multiple-valued system and cascaded decomposition.

5. **References**