# Biconditional-BDD Ordering for Autosymmetric Functions

Anna Bernasconi
Dipartimento di Informatica
Università di Pisa, Italy
anna.bernasconi@unipi.it

Valentina Ciriani
Dipartimento di Informatica
Università degli Studi di Milano, Italy
valentina.ciriani@unimi.it

Gabriella Trucco
Dipartimento di Informatica
Università degli Studi di Milano, Italy
gabriella.trucco@unimi.it

***Abstract*—Autosymmetric functions are particular "regular" Boolean functions that are exploited for logic optimization, since it is possible to reduce the number of variables and the number of points of the original autosymmetric function before its synthesis. In this paper we study this regularity in oder to derive a suitable variable ordering for Biconditional Binary Decision Diagrams (BBDDs). BBDDs are a new version of BDD that have EXOR of two variables (instead of a variable) in the nodes. These diagrams are employed for logic synthesis in new technologies such as silicon nanowires and DG-SiNWFETs. We show that it is possible to find a useful variable ordering for these functions and the experimental results validate our approach showing that in the $97\%$ of the cases we get an ordering that gives a number of nodes that is lower or equal to the one obtained with the standard ordering.**

## I. Introduction

Function "regularities" have been studied in different contexts of logic synthesis [2], [15], [16]. In particular, *partially symmetric* Boolean functions are Boolean functions that are invariant under the permutation of some input variables [19], [20]. A partially symmetric function that is invariant under the permutation of all variables is called *symmetric*. Symmetric and partially symmetric Boolean functions are often exploited in logic synthesis and are widely used in cryptology. The symmetric properties of these functions have been exploited for deriving suitable variable orderings for their representation via Ordered Binary Decision Diagrams (OBDDs) [14], [21].

In this paper we study a different kind of "symmetry" called autosymmetry [6], [7], [8], [9], [10] in order to find a suitable variable ordering for Biconditional Binary Decision Diagrams [3], [4], [5], which are a new variant of OBDDs used for representing EXOR rich functions. In particular, Biconditional Binary Decision Diagrams are Binary Decision Diagrams based on biconditional expansion:

$$f = (x_i \oplus x_j) f_{x_i \neq x_j} \; + \; (\overline{x}_i \oplus x_j) f_{x_i = x_j} \, .$$

Each node in a BBDD contains an EXOR of two variables (or a single variable that is in EXOR with the constant 1).

For example, Figure 2 shows a BBDD representation for the completely specified Boolean function $f = \{0000, 00001, 0010, 0100, 0101, 01111, 1000, 1010, 1011, 1101, 1110, 1111\}$. Each node contains a couple of variables. If the two variables are different in value (i.e., the EXOR of the two variables is true) then we consider the branch $!=$, otherwise (i.e., the EXOR is false) we consider the branch $=$. A dotted line corresponds to a complemented edge, i.e., the value of the corresponding sub-function is complemented.

Logic synthesis for emerging technologies is still at the beginning, but it is quite clear that there is a higher demand of new primitives. In this context, BBDDs have been exploited for a controllable-polarity DG silicon nanowires field effect transistors (SiNWFETs) and controllable-polarity DG-SiNWFETs [4]. Experimental results show that the BBDD prestructuring for circuits based on emerging technology devices is more effective than for standard CMOS.

In this context, the regularity of a Boolean function $f$ of $n$ variables is expressed by an *autosymmetry degree* $k$ (with $0 \leq k \leq n$), computed in polynomial time. While the extreme value $k = 0$ means no regularity, for $k \geq 1$ the function $f$ is said to be *autosymmetric*, and a new function $f_k$, called the *restriction* of $f$, is identified in polynomial time. In a sense, $f_k$ is "equivalent" to, but smaller than $f$, depends on $n - k$ variables $(y_1, \ldots, y_{n-k})$ only, and the number of points of $f_k$ is equal to the one of $f$ divided by $2^k$. Therefore, the minimization of $f_k$ is naturally easier than that of $f$. The new variables $y_1, \ldots, y_{n-k}$ are built as EXOR combinations of the original variables, that is $y_i = EXOR(X_i)$, with $X_i \subseteq \{x_1, \ldots, x_n\}$. These EXOR equations are called *reduction equations* and are exploited, in this paper, for deriving a suitable ordering for BBDDs.

For example, consider the completely specified Boolean function $f = \{0000, 00001, 0010, 0100, 0101, 01111, 1000, 1010, 1011, 1101, 1110, 1111\}$. The function $f$ is 2-autosymmetric and Figure 1 shows the the BBDD representation of $f$ using the standard variable ordering $x_1$, $x_2$, $x_3$, $x_4$. As shown in Section III, the autosymmetry property suggests that the each variables in these two sets $\{x_1, x_3\}$ and $\{x_2, x_4\}$ should be adjacent in the ordering. In fact, the BBDD representation of $f$, with ordering $x_1$, $x_3$, $x_2$, $x_4$, depicted in Figure 2 is more compact.

Although autosymmetric functions form a subset of all possible Boolean functions, a great amount of standard functions of practical interest fall in this class. Note that an autosymmetric function $f$ depends in general on all the $n$ input variables, however we shall be able to study $f$ in a $n-k$ dimensional space; i.e., $f$ is in general not degenerated, whereas all degenerated functions are autosymmetric.

Experimental results show that the ordering derived from reduction equations gives interesting results. In 97% of the considered benchmarks, we have that the number of nodes

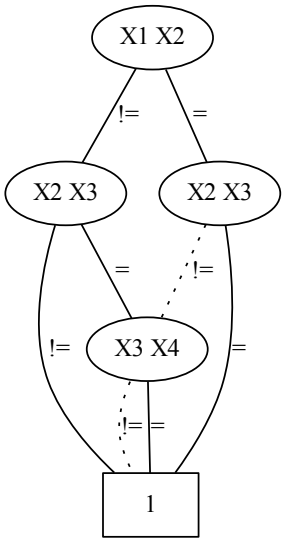


Fig. 1. BBDD of the 2-autosymmetric function $f = \{0000, 00001, 0010, 0100, 0101, 01111, 1000, 1010, 1011, 1101, 1110, 1111\}$ with variable ordering $x_1$, $x_2$, $x_3$, $x_4$.

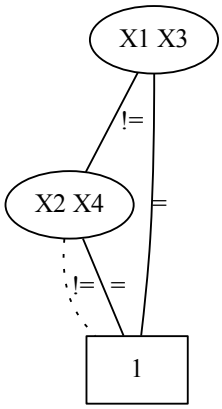


Fig. 2. BBDD of the 2-autosymmetric function $f = \{0000, 00001, 0010, 0100, 0101, 01111, 1000, 1010, 1011, 1101, 1110, 1111\}$ with variable ordering $x_1$, $x_3$, $x_2$, $x_4$.

in a BBDD with our ordering is lower or equal then that in the BBDD with a standard initial ordering (in the 56% of the benchmarks we get a strictly lower number of nodes).

The paper is organized as follows. Section II summaries the concepts of autosymmeric functions and Biconditional Binary Decision Diagrams. Section III shows the method for deriving a BBDD ordering for autosymmetric functions that have reduction equations containing EXORs with at most 2 literals. Section IV provides the experimental results and Section V concludes the paper.

## II. Preliminaries

### A. Autosymmetric Functions

In this section we briefly review autosymmetric functions that are introduced in [17] and further studied in [6], [7], [8], [9], [10]. For the description of these particular regular functions we need to summarize several concepts of Boolean algebra [13].

Given two binary vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, let $\boldsymbol{\alpha} \oplus \boldsymbol{\beta}$ be the elementwise EXOR between $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, for example $11010 \oplus 11000 = 00010$. We recall that $(\{0,1\}^n, \oplus)$ is a vector space, and that a *vector subspace* $V$ is a subset of $\{0,1\}^n$ containing the zero vector $\mathbf{0}$, such that for each $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ in $V$ we have that $\boldsymbol{v}_1 \oplus \boldsymbol{v}_2 \in V$. The vector subspace $V$ contains $2^k$ vectors, where $k$ is the *dimension* of $V$, and is generated by a basis $B$ containing $k$ vectors. Indeed $B$ is a minimal set of vectors of $V$ such that each point of $V$ is an EXOR combination of some vectors in $B$.

Now, let us consider a completely specified Boolean function $f : \{0,1\}^n \rightarrow \{0,1\}$; recall that $f$ can be described as the set of binary vectors in $\{0,1\}^n$ for which $f$ takes the value 1. Using this notation we can give the following definition. The function $f$ is *closed under* a vector $\boldsymbol{\alpha} \in \{0,1\}^n$, if for each vector $\boldsymbol{w} \in \{0,1\}^n$, $\boldsymbol{w} \oplus \boldsymbol{\alpha} \in f$ if and only if $\boldsymbol{w} \in f$.

For example, the function $f = \{0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1011, 1101, 1110\}$ is closed under $\boldsymbol{\alpha} = 0011$, as it can be easily verified.

It is easy to observe that any function $f$ is closed under the zero vector $\mathbf{0}$. Moreover, if a function $f$ is closed under two different vectors $\boldsymbol{\alpha}_1$, $\boldsymbol{\alpha}_2 \in \{0,1\}^n$, it is also closed under $\boldsymbol{\alpha}_1 \oplus \boldsymbol{\alpha}_2$. Therefore, the set $L_f = \{\boldsymbol{\beta}$: $f$ is closed under $\boldsymbol{\beta}\}$ is a vector subspace of $(\{0,1\}^n, \oplus)$. The set $L_f$ is called the *vector space* of $f$. For instance, the function $f$ of our previous example is closed under the vectors in the vector space $L_f = \{0000, 0011, 0101, 0110\}$.

For an arbitrary function $f$, the vector space $L_f$ provides the essential information for the definition of the autosymmetry property:

*Definition 1 ([9]):* A completely specified Boolean function $f$ is *$k$-autosymmetric*, or equivalently $f$ has *autosymmetry degree* $k$, $0 \leq k \leq n$, if its vector space $L_f$ has dimension $k$.

In general, $f$ is *autosymmetric* if its autosymmetry degree is $k \geq 1$. For instance, the function $f$ of our running example is 2-autosymmetric since its vector space $L_f$ has dimension 2.

We now define a special basis, called canonical, to represent $L_f$. Consider a $2^k \times n$ matrix $M$ whose rows correspond to the points of a vector space $V$ of dimension $k$, and whose columns correspond to the variables $x_1, x_2, \ldots, x_n$. Let the row indices of $M$ be numbered from 0 to $2^k - 1$. We say that $V$ is in *binary order* if the rows of $M$ are sorted as increasing binary numbers. We have:

*Definition 2 ([9]):* Let $V$ be a vector space of dimension $k$ in binary order. The *canonical basis* $B_V$ of $V$ is the set of points corresponding to the rows of $M$ with indices

$2^0, 2^1, \ldots, 2^{k-1}$. The variables corresponding to the first 1 from the left of each row of the canonical basis are the *canonical variables* of $V$, while the other variables are *non-canonical*.

It can be easily proved that the canonical basis is indeed a vector basis [12]. The canonical variables of $L_f$ are also called canonical variables of $f$.

*Example 1:* Consider the vector space $L_f$ of the function $f$ of our running example. We can arrange its vectors in a matrix in binary order:

| | $x_1$ | $x_2$ | $x_3$ | $x_4$ |
|---|---|---|---|---|
| **0** | 0 | 0 | 0 | 0 |
| **1** | 0 | 0 | 1 | 1 |
| **2** | 0 | 1 | 0 | 1 |
| **3** | 0 | 1 | 1 | 0 |

The canonical basis is composed of the vectors in position **1** and **2**, that are the vectors 0011 and 0101. The canonical variables of $f$ are $x_2$ (corresponding to the first 1 in 0101) and $x_3$ (corresponding to the first 1 in 0011). The remaining variables $x_1$ and $x_4$ are non-canonical.

For a vector $\boldsymbol{w} \in \{0,1\}^n$ and a subset $S \subseteq \{0,1\}^n$, consider the set $\boldsymbol{w} \oplus S = \{\boldsymbol{w} \oplus \boldsymbol{s} \mid \boldsymbol{s} \in S\}$. In a sense vector $\boldsymbol{w}$ is used to "translate" a subset $S$ through the vector space. We have:

*Definition 3:* Let $V$ be a vector subspace of $(\{0,1\}^n, \oplus)$. The set $A = \boldsymbol{\alpha} \oplus V$, $\boldsymbol{\alpha} \in \{0,1\}^n$, is an *affine space* over $V$ with *translation point* $\boldsymbol{\alpha}$.

Note that if $S$ is a vector subspace then $\boldsymbol{w} \in A$, because $S$ contains the zero vector $\mathbf{0}$, hence $\boldsymbol{w} \oplus \mathbf{0} = \boldsymbol{w}$. Moreover any other vector of $A$ could be chosen as $\boldsymbol{w}$, thus generating the same affine space.

There is a simple formula that characterizes the vector space associated to a given affine space $A$, namely [13]:

$$V = \boldsymbol{\alpha} \oplus A, \text{ with } \boldsymbol{\alpha} \text{ any point in } A.$$

That is, given an affine space $A$ there exists a unique vector space $V$ such that $A = \boldsymbol{\alpha} \oplus V$, where $\boldsymbol{\alpha}$ is any point of $A$.

As proved in [6], the points of a $k$-autosymmetric function $f$ can be partitioned into $\ell = |f|/2^k$ disjoint sets, where $|f|$ denotes the number of points of $f$; all these sets are affine spaces over $L_f$. I.e., $S = \boldsymbol{w} \oplus L_f$, where $S$ is any such a space and $\boldsymbol{w} \in f$. Thus:

$$f = \bigcup_{i=1}^{\ell} (\boldsymbol{w}^i \oplus L_f)$$

and for each $i, j$, $i \neq j$, $(\boldsymbol{w}^i \oplus L_f) \cap (\boldsymbol{w}^j \oplus L_f) = \emptyset$. The vectors $\boldsymbol{w}^1, \ldots, \boldsymbol{w}^\ell$ are chosen as all the points of $f$ where all the canonical variables have value 0.

*Example 2:* Consider the function $f = \{0000, 0001, 0010, 0011, 0100, 0101, 0110, 0111, 1000, 1011, 1101, 1110\}$ of our running example. By Example 1 the canonical variables of $f$ are $x_2$ and $x_3$. Thus, if we take the points of $f$ with all canonical variables set to 0, i.e., $\boldsymbol{w}^1 = 0000$, $\boldsymbol{w}^2 = 0001$, and $\boldsymbol{w}^1 = 1000$, we have

$$f = (0000 \oplus L_f) \cup (0001 \oplus L_f) \cup (1000 \oplus L_f),$$

where $L_f = \{0000, 0011, 0101, 0110\}$.

Autosymmetric functions can be reduced to "equivalent, but smaller" functions; in fact, if a function $f$ is $k$-autosymmetric, then there exists a function $f_k$ over $n - k$ variables, $y_1$, $y_2$, ..., $y_{n-k}$, such that

$$f(x_1, \ldots, x_n) = f_k(y_1, \ldots, y_{n-k}),$$

where each $y_i$ is an EXOR combination of a subset of $x_i$'s. These combinations are denoted $EXOR(X_i)$, where $X_i \subseteq X$, and the equations $y_i = EXOR(X_i)$, $i = 1, \ldots, n-k$, are called *reduction equations*. The function $f_k$ is called a *restriction* of $f$; indeed $f_k$ is "equivalent" to, but smaller than $f$, and has $|f|/2^k$ points only.

The restriction $f_k$ can be computed from $f$ and its vector space $L_f$ by first identifying the canonical variables, and then deriving the cofactor of $f$ where all the canonical variables are set to 0 (see [6] and [9] for more details). The reduction equations correspond to the homogeneous system of linear equations whose solutions define the vector space $L_f$, and they can be derived applying standard linear algebra techniques as shown in [6], [9].

*Example 3:* Consider the 2-autosymmetric function $f$ in our running example, with $L_f = \{0000, 0011, 0101, 0110\}$ and canonical variables $x_2$ and $x_3$. We can build $f_2$ by taking the cofactor $f_{x_2=0,x_3=0} = \{00, 01, 10\}$, that contains only 3 points and corresponds to the function $f_2(y_1, y_2) = \overline{y_1 y_2}$. The homogeneous system whose solutions are $\{0000, 0011, 0101, 0110\}$ is:

$$\begin{cases} x_1 = 0 \\ x_2 \oplus x_3 \oplus x_4 = 0 \end{cases}$$

Thus the reduction equations are given by

$$\begin{aligned} y_1 &= x_1 \\ y_2 &= x_2 \oplus x_3 \oplus x_4 \,. \end{aligned}$$

### B. Biconditional Binary Decision Diagrams

A *Binary Decision Diagram* (BDD) over a set of Boolean variables $X = \{x_1, x_2, \ldots, x_n\}$ is a rooted, connected direct acyclic graph, where each non-terminal (internal) node $N$ is labeled by a Boolean variable $x_i$ and has exactly two outgoing edges, the 0-edge and the 1-edge, pointing to two nodes called the 0-child and the 1-child of node $N$, respectively. Terminal nodes (leaves) are labeled 0 or 1. Binary decision diagrams are typically used to represent Boolean functions. Let $f$ be a completely specified Boolean function, and $f_{x_i}$ and $f_{\overline{x}_i}$ be the cofactors derived from $f$ substituting the variable $x_i$ with the values 1 and 0, respectively. The Shannon decomposition of $f$ around $x_i$ is:

$$f = x_i f_{x_i} \; + \; \overline{x}_i f_{\overline{x}_i} \,,$$

where $\overline{x}_i$ is the negation of the variable $x_i$. Any node in a BDD represents a Boolean function. The leaves represent the constant functions 0 and 1 and the root represents the entire Boolean function $f$. If the non-terminal node $N$ (with label $x_i$) represents the function $g$, then the 1-child of $N$ (resp. 0-child) represents the function $g_{x_i}$ (resp., $g_{\overline{x}_i}$). In other words, each internal node represents the Shannon decomposition with respect to its variable. The value of $f$ on the input $x_1, \ldots, x_n$ is found by following the path indicated in the BDD by the

values of $x_0, \ldots, x_{n-1}$. A *1-path* (resp. *0-path*) in a BDD is a path from the root to a leaf labeled by 1 (resp. 0).

A BDD is *ordered* if there exists a total order $<$ over the set $X$ of variables such that if an internal node is labeled by $x_i$, and its 0-child and 1-child have labels $x_{i_0}$ and $x_{i_1}$, respectively, then $x_i < x_{i_0}$ and $x_i < x_{i_1}$. A BDD is *reduced* if there exist no nodes whose 1-child is equal to the 0-child and there not exist two distinct nodes that are roots of isomorphic subgraphs. A reduced and ordered BDD is called *ROBDD*. The ROBDD is a canonical form; indeed, given a function $f : \{0,1\}^n \to \{0,1\}$ and a variable ordering $<$, there is exactly one ROBDD with variable ordering $<$ that represents $f$.

ROBDDs are usually quite compact, but there are functions whose ROBDD representation has a size, i.e., number of nodes, exponential in the number of input variables. Therefore, several extensions of BDDs have been considered (see [4] for a review of the main extensions). In this paper we consider a very recent extension, called *Biconditional Binary Decision Diagram* (BBDD), introduced in [3] and further studied in [4], [5]. In contrast to BDDs where each internal node represents a Shannon expansion, each internal node in a BBDD represents the *biconditional expansion*

$$f = (x_i \oplus x_j) f_{x_i \neq x_j} \; + \; (\overline{x}_i \oplus x_j) f_{x_i = x_j} \, ,$$

where $f_{x_i \neq x_j}$ and $f_{x_i = x_j}$ are the two cofactors derived from $f$ substituting $x_i$ with $\overline{x}_j$ and $x_j$, respectively. The two variables $x_i$ and $x_j$ are called the *Primary variable* (PV) and the *Secondary variable* (SV). In order to fully decompose a Boolean function by biconditional expansions, a boundary condition for the expansion of single variable functions is needed. This boundary condition can be obtained by fixing the secondary variable to the constant 1, so that biconditional expansion is reduced to Shannon expansion.

Thus, any non-terminal node $N$ (with variables $x_i$ and $x_j$) in a BBDD has exactly two outgoing edges, the $\neq$-edge and the $=$-edge, pointing to two nodes called the $\neq$-child and the $=$-child of node $N$, respectively: if $N$ represents the function $g$, then the $\neq$-child of $N$ (resp. $=$-child) represents the function $g_{x_i \neq x_j}$ (resp., $g_{x_i = x_j}$). For instance, the DDs depicted in Figures 1 and 2 are BBDDs.

To get an ordered BBDD a variable order must be imposed for PVs and a rule for the other variables assignment must be provided. In [4], [5], the authors propose to use the *Chain Variable Order* (CVO): given a Boolean function $f$ and an order $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ for the input variables of $f$, the CVO assigns PVs and SVs as

$$\begin{cases} PV_i = \pi_i \\ SV_i = \pi_{i+1} \end{cases} \text{for } 1 \leq i < n; \quad \begin{cases} PV_n = \pi_n \\ SV_n = 1 \end{cases} \text{for } i = n.$$

BBDDs ordered by the CVO are called OBBDDs. For example, while the BBDDs shown in Figures 1 has the standard variable ordering ($x_1$, $x_2$, $x_3$, $x_4$), the BBDD in Figure 2 has the ordering ($x_1$, $x_3$, $x_2$, $x_4$).

As for OBDD, OBBDDs can be reduced applying some reduction rules. In particular, an OBBDD is *weak Reduced* (*weak* ROBBDD) if it contains no nodes whose $\neq$-child is equal to the $=$-child and it contains no isomorphic subgraphs. Moreover, an OBBDD is *strong Reduced* (*strong* ROBBDD) if besides being weak reduced:

- it contains no empty levels, i.e., levels created by the CVO but containing no nodes as a result of the augmented functionality of biconditional expansion;
- subgraphs representing single variable functions are collapsed into a single BDD node driven by the Shannon expansion.

Both *weak* and *strong* ROBBDDs are canonical forms, as proved in [4].

In summary, BBDDs represent a generalization of BDDs, where the Shannon expansion is entirely substituted with the biconditional expansion. As a result, in a BBDD we can have both standard nodes labeled by single variables and nodes where the branching condition depends on two variables, rather than only one as in standard BDDs. Considering two variables per time enhances the expressive power of a decision diagram, as experimentally shown in [4], [5]: BBDDs are about 20% smaller, in terms of node count, with respect to BDDs built and sifted using the CUDD library [1].

Finally, recall that it is possible to create circuits from BDDs by directly mapping the BDD representing a Boolean function onto a network of multiplexers. As a result the depth of the circuit is quite large; in fact the depth is equal to the number of primary inputs. But as an advantage the circuits are quite compact for a large range of functions. The same can be done for BBDDs, that can be mapped onto networks of multiplexers and EXOR gates, both particularly suitable for emerging technologies based on graphene [18], [22].

## III. Variable ordering for Autosymmetric Functions

In this section we show how to derive a variable ordering for BBDD representations of autosymmetric functions. Section II-A shows that a $k$-autosymmetric function is associated to a vector space $L_f$ with dimension $k$ and a set of reduction equations.

In a Boolean space $\{0,1\}^n$ described by $n$ variables $x_1, \ldots, x_n$, let a 2-*EXOR* be an EXOR with at most 2 input variables, one of which possibly complemented. Given two Boolean variables $x_1, x_2$, all the possible 2-EXORs are essentially $x_1$, $\overline{x}_1$, $x_2$, $\overline{x}_2$, $(x_1 \oplus x_2)$ and $(x_1 \oplus \overline{x}_2)$ (in fact, $\overline{x}_1 \oplus x_2 = x_1 \oplus \overline{x}_2$, and $\overline{x}_1 \oplus \overline{x}_2 = x_1 \oplus x_2$).

In this paper we will consider autosymmetric functions whose reduction equations contain only 2-EXORs. This choice will give us a simple method for partitioning the variables in order to determine a useful ordering for BBDDs. Moreover, we have experimentally evaluated that the 98% of autosymmetric benchmark functions has this property.

*Example 4:* The following reduction equations contain 2-EXORs only:

$$\begin{cases} y_1 = x_2 \\ y_2 = x_1 \oplus x_3 \\ y_3 = x_1 \oplus x_5 \\ y_4 = x_6 \\ y_5 = x_7 \oplus x_8 \end{cases}$$

Recall that reduction equations are constructed from a homogeneous linear system. For instance, the reduction equations

in the above example correspond to the system[1]:

$$\left\{\begin{array}{rcl} x_2 & = & 0 \\ x_1 \oplus x_3 & = & 0 \\ x_1 \oplus x_5 & = & 0 \\ x_6 & = & 0 \\ x_7 \oplus x_8 & = & 0 \end{array}\right. = \left\{\begin{array}{lcl} x_2 & = & 0 \\ x_1 & = & x_3 \\ x_1 & = & x_5 \\ x_6 & = & 0 \\ x_7 & = & x_8 \end{array}\right.$$

From this system we derive the following equalities:

$$\begin{array}{lll} 0 & = x_2 & = x_6 \\ x_1 & = x_3 & = x_5 \\ x_7 & = x_8\,. & \end{array}$$

These equalities suggest a natural partition of the set $\{0, x_1, x_2, \ldots, x_9\}$:

$$\{\{0, x_2, x_6\}, \{x_1, x_3, x_5\}, \{x_7, x_8\}, \{x_4\}, \{x_9\}\}\,,$$

where each subset contains a set of variables (or the constant 0) that must have the same value, and where we have added the two singletons $\{x_4\}$ and $\{x_9\}$ representing the variables that can assume all the possible values. In particular, in our example $x_2$ and $x_6$ must be always equal to 0, while $x_1$, $x_3$, and $x_6$ must have the same value (0 or 1), moreover, $x_7$ and $x_8$ must have the same value.

In general we can give the following definitions.

*Definition 4:* Let $S$ be a homogeneous linear system containing only 2-EXORs:

$$\left\{\begin{array}{ll} x_{i_1} \oplus x_{j_1} & = 0 \\ & \vdots \\ x_{i_k} \oplus x_{j_k} & = 0 \\ x_{l_1} & = 0 \\ & \vdots \\ x_{l_m} & = 0 \end{array}\right.$$

The *equality system* is the system $E_S$ such that:

$$\left\{\begin{array}{ll} x_{i_1} & = x_{j_1} \\ & \vdots \\ x_{i_k} & = x_{j_k} \\ x_{l_1} & = 0 \\ & \vdots \\ x_{l_m} & = 0 \end{array}\right.$$

It is straightforward to verify that $S$ and $E_S$ have the same solutions since $x_i \oplus x_j = 0$ if and only if $x_i = x_j$.

*Definition 5:* Let $S_E$ be an equality system in the Boolean space described by the set of variables $\{x_1, x_2, \ldots, x_n\}$:

$$\left\{\begin{array}{ll} x_{i_1} & = x_{j_1} \\ & \vdots \\ x_{i_k} & = x_{j_k} \\ x_{l_1} & = 0 \\ & \vdots \\ x_{l_m} & = 0 \end{array}\right.$$

where $x_{i_h}, x_{j_h} \in \{x_1, x_2, \ldots, x_n\}$ for $1 \leq h \leq k$ and $x_{l_h} \in \{x_1, x_2, \ldots, x_n\}$ for $1 \leq h \leq m$. The *partition* derived from $E_S$ is the partition $P_S$ of the set $\{0, x_1, x_2, \ldots, x_n\}$ where, for any $x$ and $y$ in $\{0, x_1, x_2, \ldots, x_n\}$, $x$ and $y$ are in the same subset of the partition if and only if the equality $x = y$ can be derived from the system $S_E$.

We now show how the homogeneous linear systems and the corresponding partitioning of Boolean variables give a suitable ordering for BBDDs for autosymmetric functions whose homogeneous system contains 2-EXORs only. In fact, each EXOR node of a BBDD is an EXOR between two variables that are adjacent in the given ordering (the ordering CVO shown in Section II-B). From the partition we can describe the following ordering:

*Definition 6:* Let $P_S$ be a partition derived from an equality system $E_S$ such that $P_S = \{p_1, p_2, \ldots, p_s\}$ and each $p_i$ $(1 \leq i \leq s)$ is a subset of $\{0, x_1, x_2, \ldots, x_n\}$. An *ordering* $O_S$ derived from $P_S$ is $v_1, v_2, \ldots, v_s$ where each $v_i$ $(1 \leq i \leq s)$ represents the variables of $p_i$ in any order.

For example an ordering $O_S$ derived from $P_S = \{\{0, x_2, x_6\}, \{x_1, x_3, x_5\}, \{x_7, x_8\}, \{x_4\}, \{x_9\}\}$ of the previous example is

$$O_S = x_2, x_6, x_1, x_3, x_5, x_7, x_8, x_4, x_9\,.$$

Recall that a $n$ variable $k$-autosymmetric function indeed depends on $n - k$ new variables $y_1, y_2, \ldots, y_{n-k}$ that are linear combinations of the original variables $\{x_1, x_2, \ldots, x_n\}$ through the reduction equations. For this reason BBDD seem to be be quite suitable for such functions provided that any two variables, that are in EXOR in the reduction equations, are also adjacent in the ordering.

In order to show that the proposed ordering $O_S$ guarantees this property, we give the following example. Consider the reduction equations in Example 4, we have that $y_2 = x_1 \oplus x_3$ and $y_3 = x_1 \oplus x_5$. The proposed ordering $O_S = x_2, x_6, x_1, x_3, x_5, x_7, x_8, x_4, x_9$ seems to be not suitable since $x_1$ is next to $x_3$, but not to $x_5$. However, this is not a problem since $x_1 = x_3 = x_5$, therefore the reduction equations in Example 4 are completely equivalent to the following ones:

$$\left\{\begin{array}{ll} y_1 & = x_2 \\ y_2 & = x_1 \oplus x_3 \\ y_3 & = x_3 \oplus x_5 \\ y_4 & = x_6 \\ y_5 & = x_7 \oplus x_8 \end{array}\right.$$

where $y_3 = x_3 \oplus x_5$. Therefore, the ordering inside each subset is not important since the variables in the subset are equal.

We can finally state and prove the following proposition.

*Proposition 1:* Let $O_S$ be an ordering derived from a homogeneous linear system $S$ containing only 2-EXORs. It is always possible to find a homogeneous linear system $S'$ equivalent to $S$ (i.e., with the same solution of $S$) such that each 2-EXOR in $S'$ corresponds to a couple of adjacent variables in $O_S$.

*Proof:* Let $s$ a set of variables contained in the partition $P_S$. We construct $S'$ considering the following three cases:

1) If $s$ is a singleton, i.e., a set with exactly one element $x$, then $x$ does not occur in the homogeneous linear system $S$. We do not insert $x$ in $S'$.

[1]Recall that $x_i \oplus x_j = 0$ if and only if $x_i = x_j$.

2) If $s_x$ is the set containing the constant 0, for each variable $x \in s$ we have that $x = 0$ (i.e., $x$ is not in an EXOR with another variable). Thus, for each variable $x \in s$ we insert the equation $x = 0$ in $S'$ .
3) Otherwise, the set $s$ is not a singleton and contains variables only. In this case, each variable $x_i \in s$ is contained in at least an EXOR of $S$ with another variable $x_j$, which is in the same partition of $x_i$, (by construction of $O_S$), i.e., $x_j \in s$. Without loss of generality, let $x_1, x_2, \ldots x_h$ be the variables, contained in the set $s$, ordered following the ordering $O_S$. We can put in $S'$ the $h-1$ equations: $x_i = x_{i+1}$ with $1 \leq i \leq h-1$. It is straightforward to verify that these equations are enough to express the fact that the variables in $s$ must all have the same value.

The system $S'$ is therefore equivalent to $S$ and each EXOR of two variables in $S'$ corresponds to a couple of adjacent variables in $O_S$, by construction. ■

Of course, we can construct a set of reduction equations from the system $S'$ derived in Proposition 1 as described in Section II-A. Therefore, we can conclude that $O_S$ could be a suitable ordering for BBDDs, since each EXOR of two variables in the reduction equations is composed by a couple of variables that are adjacent in the ordering $O_S$.

For a complete example, consider the completely specified Boolean function $f = \{0000, 00001, 0010, 0100, 0101, 01111, 1000, 1010, 1011, 1101, 1110, 1111\}$. Note that $L_f = \{0000, 0101, 1010, 1111\}$ since $f$ is closed only under the vectors in $L_f$. Moreover, $f = 0000 \oplus L_f \cup 0001 \oplus L_f \cup 1000 \oplus L_f$. The homogeneous linear system $S$ corresponding to $L_f$ is:

$$\begin{cases} x_1 \oplus x_3 &= 0 \\ x_2 \oplus x_4 &= 0 \end{cases}$$

Finally, the ordering $O_S$ is $x_1$, $x_3$, $x_2$, $x_4$. The ordering suggested by the system gives a BBDD representation of $f$, in Figure 2, more compact that the BBDD obtained by the standard ordering $x_1$, $x_2$, $x_3$, $x_4$ in Figure 1.

## IV. Experimental results

### A. Experimental results

In this section we report the experimental results related to variable ordering for BBDD of autosymmetric functions. The experiments have been run on a Linux Intel Core i7, 3.40 GHz CPU with 8 GB of main memory. The benchmarks are taken from LGSynth93 [23]. Multioutput benchmarks have been synthesized minimizing each single output independently from the others.

The software used to derive the reduction equations used to determine the variable ordering for BBDD of autosymmetric functions is described in [6], [7], [8], [9]. As specified in Sect. III, since 98% of autosymmetric benchmark functions are described by reduction equations containing only 2-EXORs, we consider only this set of benchmarks. The general criterion used to specify the list of ordered variables is the following: first of all we insert in the ordered list each variable $x_i$ which is in EXOR with another variable $x_j$; then we consider all singletons, and finally all other variables.

TABLE I. Comparison of EXOR nodes and VAR nodes: ordered vs not ordered case with (ro 0; st 0) configuration.

| | not ordered | | ordered | |
|---|---|---|---|---|
| benchmark | EXOR nodes | VAR nodes | EXOR nodes | VAR nodes |
| add6(0) | 6 | 0 | 1 | 0 |
| add6(1) | 12 | 1 | 2 | 1 |
| add6(2) | 22 | 2 | 6 | 2 |
| add6(3) | 38 | 3 | 14 | 3 |
| add6(4) | 62 | 4 | 30 | 4 |
| add6(5) | 94 | 5 | 62 | 5 |
| adr4(1) | 26 | 1 | 16 | 3 |
| adr4(2) | 15 | 1 | 6 | 2 |
| adr4(3) | 8 | 1 | 2 | 1 |
| adr4(4) | 4 | 0 | 1 | 0 |
| al2(11) | 27 | 4 | 20 | 4 |
| alcom(5) | 5 | 3 | 3 | 3 |
| apla(5) | 11 | 1 | 10 | 1 |
| b7(5) | 3 | 1 | 3 | 1 |
| b9(3) | 327 | 6 | 262 | 2 |
| b11(5) | 3 | 1 | 1 | 1 |
| b12(6) | 100 | 4 | 31 | 4 |
| bcd.div3(0) | 2 | 0 | 2 | 0 |
| dc1(5) | 5 | 2 | 3 | 2 |
| dekoder(0) | 6 | 1 | 6 | 1 |
| dekoder(1) | 5 | 2 | 5 | 2 |
| dk27(8) | 7 | 1 | 7 | 1 |
| dk27(0) | 11 | 1 | 9 | 1 |
| dk27(1) | 11 | 1 | 9 | 1 |
| ex7(3) | 327 | 6 | 262 | 2 |
| exps(18) | 13 | 1 | 6 | 1 |
| exps(19) | 11 | 1 | 6 | 1 |
| f51m(6) | 1 | 0 | 1 | 0 |
| luc(3) | 8 | 3 | 5 | 3 |
| m1(8) | 7 | 1 | 4 | 1 |
| max1024(0) | 8 | 0 | 8 | 0 |
| max1024(1) | 16 | 0 | 16 | 0 |
| max1024(2) | 36 | 0 | 36 | 0 |
| max1024(3) | 55 | 0 | 55 | 0 |
| max1024(4) | 78 | 0 | 78 | 0 |
| max1024(5) | 103 | 0 | 103 | 0 |
| misex1(0) | 5 | 1 | 1 | 0 |
| mytest(0) | 1 | 0 | 1 | 0 |
| newcond(1) | 2 | 1 | 2 | 1 |
| newcwp(0) | 3 | 1 | 3 | 1 |
| newcwp(3) | 1 | 0 | 1 | 0 |
| p82(10) | 3 | 1 | 3 | 1 |
| pope.rom(18) | 9 | 1 | 4 | 1 |
| pope.rom(32) | 2 | 1 | 2 | 1 |
| pope.rom(35) | 2 | 0 | 2 | 0 |
| pope.rom(41) | 3 | 1 | 3 | 1 |
| pope.rom(47) | 6 | 1 | 3 | 1 |
| radd(0) | 4 | 0 | 1 | 0 |
| radd(1) | 8 | 1 | 2 | 1 |
| radd(2) | 14 | 2 | 6 | 2 |
| radd(3) | 22 | 3 | 14 | 3 |
| rd53(1) | 5 | 0 | 5 | 0 |
| risc(4) | 3 | 1 | 1 | 1 |
| squar5(6) | 3 | 1 | 3 | 1 |
| sqn(0) | 31 | 2 | 11 | 3 |
| sqr6(8) | 3 | 1 | 1 | 1 |
| t4(2) | 25 | 1 | 27 | 1 |
| t4(3) | 119 | 1 | 146 | 1 |
| wim(2) | 5 | 2 | 5 | 2 |
| Z5xp1(8) | 1 | 0 | 1 | 0 |
| Z9sym(0) | 18 | 0 | 18 | 0 |

For our experimentation we used the BBDD package, implemented in C language and presented in [5]. Since the BBDD package receives as input a Verilog description of a combinational logic network, flattened onto primitive Boolean operations, we have to translate our benchmarks in the proper format using ABC [11]. For each Verilog description of a single benchmark we save two different versions: the first one is the plain version with not ordered variables; the second one is the version where we specify a different variable order (based on the reduction equations previously described). We then run the BBDD package on both versions of the same benchmark, and to compare the obtained results. We consider

benchmark functions with at least an EXOR of two variables in the reduction equations.

Recall that BBDD contains two type of nodes: nodes that are EXOR of two variables (which we call here EXOR nodes) and nodes that contain a single variable, i.e., that are EXOR of one variable with the constant 1, (which we call here VAR nodes). In Table I we compare the number of EXOR nodes and the number of VAR nodes for not-ordered versus ordered bechmarks. The first column reports the name of the benchmarks followed by the number of the considered output (indicated in brackets). The following columns report, by groups of two, the number of EXOR nodes and the number of VAR nodes obtained running BBDD package. The first group (columns two and three) refers to plain benchmarks with not ordered variables, the second group (columns four and five) refers to benchmarks with ordered variables. To obtain the results reported in Tab. I, we ran the simulation requiring that the order of the variables is exactly the one specified in the input file that describes the benchmark. This is achieved by disabling dynamic and static reordering (otherwise, by default the BBDD package makes reordering to minimize the size).

The results are interesting, showing that ordering the variables is useful. In 97% of cases, imposing the ordering of the variables we obtain a number of EXOR nodes lower than or equal to the corresponding number of EXOR nodes of benchmark without variable ordering (the percentage is equal to 56% if we consider a number of nodes strictly lower than the corresponding number of nodes in not-ordered benchmarks). Similarly, the number of VAR nodes decreases or is equal to the number of VAR nodes obtained by the simulation of the not-ordered benchmark in 95% of the cases (5% if strictly lower than the not-ordered case). Of course, since the ordering we propose is the result of a heuristic approach, sometimes the results for ordered benchmarks could be worse than those relative to the not-ordered case, as it happens for instance for the benchmark $t4(3)$. The average gain obtained with the ordering of variables is about 23% in the case of EXOR nodes and 6% in the case of VAR nodes. Moreover, we ran the experiments without disabling dynamic and static reordering (using the i-sifting command setting, that is an iterative version of the standard sifting procedure). In this way the BBDD package makes its own reordering to minimize the size of the circuits. In this case, there is no difference in the results by changing the order of the variables in the input files describing the benchmarks, and the BBDD package finds similar solution.

## V. Conclusion

In this paper we have proposed a method for deriving an ordering for the BBDD representation of autosymmetric functions with reduction equations containing 2-EXORs, where BBDDs are a new DD data structure exploited for emerging technologies. In the experiments of this paper we have tested a function for autosimmetry and derived its reduction equations using a standard tool that also computes the restriction $f_k$, which is not relevant for BBDD ordering. Future work includes the design of "ad hoc" preprocessing functions that directly compute the BBDD ordering for an autosymmetric function. Another interesting direction is the study of new "symmetries" of Boolean functions that can give good informations for deriving ordering for BBDDs or other variations of BDDs.

## References


[1] CUDD: CU Decision Diagram Package Release 2.5.0.

[2] F. Aloul, A. Ramani, I. Markov, and K. Sakallah, "Solving Difficult SAT Instances in the Presence of Symmetry," in *ACM/IEEE 39th Design Automation Conference (DAC)*, 2002, pp. 731–736.

[3] L. G. Amarù, P. Gaillardon, and G. D. Micheli, "Biconditional BDD: a novel canonical BDD for logic synthesis targeting xor-rich circuits," in *Design, Automation and Test in Europe, DATE 13, Grenoble, France, March 18-22, 2013*, 2013, pp. 1014–1017.

[4] ——, "Biconditional binary decision diagrams: A novel canonical logic representation form," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 4, no. 4, pp. 487–500, 2014.

[5] ——, "An efficient manipulation package for biconditional binary decision diagrams," in *Design, Automation & Test in Europe Conference & Exhibition, DATE 2014, Dresden, Germany, March 24-28, 2014*, 2014, pp. 1–6.

[6] A. Bernasconi, V. Ciriani, F. Luccio, and L. Pagli, "Fast Three-Level Logic Minimization Based on Autosymmetry," in *ACM/IEEE 39th Design Automation Conference (DAC)*, 2002, pp. 425–430.

[7] ——, "Implicit Test of Regularity for Not Completely Specified Boolean Functions," in *IEEE/ACM 11th International Workshop on Logic & Synthesis (IWLS)*, 2002, pp. 345–350.

[8] ——, "Three-Level Logic Minimization Based on Function Regularities," *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 22, no. 8, pp. 1005–1016, 2003.

[9] ——, "Exploiting regularities for boolean function synthesis," *Theory Comput. Syst.*, vol. 39, no. 4, pp. 485–501, 2006.

[10] ——, "Synthesis of autosymmetric functions in a new three-level form," *Theory Comput. Syst.*, vol. 42, no. 4, pp. 450–464, 2008.

[11] R. Brayton and A. Mishchenko, "ABC: An Academic Industrial-Strength Verification Tool," in *CAV'10, Springer, LNCS 6174*, 2010, pp. 24–40.

[12] V. Ciriani, "A New Approach to Three-Level Logic Synthesis," Computer Science Department, University of Pisa, Technical Report TR-02-03, 2002, submitted.

[13] P. Cohn, *Algebra Vol. 1*. John Wiley & Sons, 1981.

[14] L. Heinrich-Litan and P. Molitor, "Least Upper Bounds for the Size of OBDDs Using Symmetry Properties," *IEEE Trans. Computers*, vol. 49, no. 4, pp. 360–368, 2000.

[15] V. Kravets and K. Sakallah, "Generalized Symmetries of Boolean Functions," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2000, pp. 526–532.

[16] ——, "Constructive Library-Aware Synthesis Using Symmetries," in *Design, Automation and Test in Europe Conference & Exhibition (DATE)*, 2001, pp. 208–213.

[17] F. Luccio and L. Pagli, "On a New Boolean Function with Applications," *IEEE Transactions on Computers*, vol. 48, no. 3, pp. 296–310, 1999.

[18] S. Miryala, V. Tenace, A. Calimera, E. Macii, M. Poncino, L. Amarú, G. De Micheli, and P.-E. Gaillardon, "Exploiting the Expressive Power of Graphene Reconfigurable Gates via Post-Synthesis Optimization," in *Proceedings of the 25th Edition on Great Lakes Symposium on VLSI*, ser. GLSVLSI '15, 2015.

[19] T. Sasao, "A new expansion of symmetric functions and their application to non-disjoint functional decompositions for LUT type FPGAs," in *International Workshop on Logic Synthesis*, 2000, pp. 105–110.

[20] ——, *Switching Theory for Logic Synthesis*. Kluwer Academic Publishers, 1999.

[21] J. Shi, G. Fey, and R. Drechsler, "BDD Based Synthesis of Symmetric Functions with Full Path-Delay Fault Testability," in *12th Asian Test Symposium (ATS 2003), 17-19 November 2003, Xian, China*, 2003, pp. 290–293.

[22] V. Tenace, A. Calimera, E. Macii, and M. Poncino, "One-pass Logic Synthesis for Graphene-based Pass-XNOR Logic Circuits," in *Proceedings of the 52Nd Annual Design Automation Conference*, ser. DAC '15, 2015.

[23] S. Yang, "Logic Synthesis and Optimization Benchmarks User Guide Version 3.0," Microelectronic Center, User Guide, 1991.