Efficient Computation of Exact and Simplified Observability Don’t care Sets for Multiple-Level Combinational Networks

Maurizio Damiani and Giovanni De Micheli


April 1990

Computer Systems Laboratory
Departments of Electrical Engineering and Computer Science
Stanford University
Stanford, CA 94305-4055

Abstract

This work presents new algorithms for the extraction of exact and approximate Observability Don’t Care sets (ODC sets) in a combinational multiple-level logic network. The proposed algorithms are efficient because they use local information, i.e. the computation of the ODC for a vertex in the network requires only the knowledge of the don’t care sets at the adjacent vertices.

Two approaches are proposed. The former computes the exact ODC sets. The latter computes ODC subsets and can be used when partial information on the don’t care sets is available. The algorithms for the simplified computation finds the largest subsets of the actual ODC sets, given the information available at the adjacent vertices.

Key Words and Phrases: Combinational Logic Synthesis, Logic Minimization, Boolean Networks, Don’t Care sets, Testing
Efficient Computation of Exact and Simplified Observability Don't care Sets for Multiple-Level Combinational Networks

1 Introduction

Over the past few years, the problem of computing efficiently and correctly the Observability Don't Care (ODC) sets has emerged as a central one in the synthesis of combinational networks [1], [2], [3]. The knowledge of the ODC sets is important in several respects, namely: 1) local minimization of functions in a Boolean network, 2) synthesis of 100% testable networks, 3) test pattern generation [4].

Bartlett proposed in [1] a computation of the ODC sets requiring the representation of the primary output expression in terms of the network intermediate variables. Such a representation may be subject to the explosion in size of the representation. Murata proposed in [3] exhaustive simulations of the circuit for determining the observability don't care sets of vertices with reconvergent fanout.

Other authors pointed out that it would be desirable and computationally much more efficient to derive the ODC set of a vertex of a Boolean network from the ODC sets of its direct fanout vertices [5], [7]. They showed, however, that a straightforward application of this idea could lead to erroneous results, because of the effect of reconvergent fanout. Therefore only approximate solutions have been proposed to compute the ODC sets [7], [8].

In this paper we show how an exact computation of ODC sets is indeed possible by using only local information. We propose an algorithm for such a computation that traverses the network backward from the primary outputs to the primary inputs processing each vertex only once.

Unfortunately this method, leading to an exact computation of the ODC sets, involves implicitly Boolean complementations and it is consequently prone to the well-known phenomenon of "combinational explosion". We therefore propose two other algorithms for the computation of subsets of the actual ODC sets, still based on local information. The first algorithm computes an ODC subset at each vertex of the Boolean network from the ODC subsets of its direct fanout vertices. The second one computes both subsets of the actual care and don't care sets from those of its fanout. It will be shown that the subsets computed by the second algorithm include those computed by the first one, thus providing a better approximation.

2 Definitions and Notations

2.1 Combinational Boolean Networks.

In this paper we model multiple-output combinational circuits by Boolean networks [1]. A Boolean network $N$ with $n$ input vertices and $m$ output vertices realizes a function $F : B^n \rightarrow B^m$ [1], where $B$ is the Boolean set $\{0, 1\}$. Underlining is used for denoting vector quantities throughout this paper.

The network is specified by its acyclic network graph $G = (V, E)$. The elements of the vertex set $V = V_I \cup V_G \cup V_O = \{v\}$ are in one-to-one correspondence with primary inputs, logic gates, and primary outputs, respectively. There is a directed edge $e$ from a vertex $\mu$ to a vertex $\nu$ if the output of the gate in $\mu$ is connected to
an input of the gate in $\nu$. In contrast to [1], and similarly to [3], we associate a network variable $y_i$ to each edge $e_i \in E$. An example of a circuit and of its associated Boolean network is shown in Fig. (1).

A network variable $y_i$ is said to be a fanout (fanin) variable of a vertex $\nu \in V$ if $e_i$ is an edge whose tail (head) end-point is $\nu$. We associate to each vertex $\nu$ an expression $f_\nu(y_1, y_2, \ldots, y_n)$ of its fanin edge variables. The expression $f_\nu$, describing the functionality of the gate in $\nu$, specifies all the fanout variables of $\nu$ in terms of the fanin variables.

2.2 Observability don't care sets.

By cutting an edge $e_i$ and by considering $y_i$ as a primary input variable, the new network $N'\epsilon$ realizes a function $\mathcal{E}'(\mathcal{E}, y) : B^{n+1} \rightarrow B^m$. Given a primary input assignment $\mathcal{E}_0$, the variable $y_i$ of $N$ is not observable if the vector equality [6]

$$\mathcal{E}(\mathcal{E}_0, 0) = \mathcal{E}(\mathcal{E}_0, 1).$$  

is satisfied. Recall that the cofactor $f|_{y_i}$ of a function $f$ with respect to $y_i$ is the function obtained by assuming $y_i = 1$. Similarly, $f|_{y_i}$ is the function obtained by setting $y_i = 0$. The vector function:

$$ODC_y = \frac{\partial \mathcal{E}}{\partial y_i}$$

therefore describes the observability of $y_i$. In particular, $y_i$ will be observable at the $k^{th}$ network output ($k = 1, \ldots, m$) if the $k^{th}$ component of $ODC_y$, is 0. The quantity $\frac{\partial \mathcal{E}}{\partial y_i}$ is usually termed Boolean difference [6] of $\mathcal{E}$ with respect to the (possibly internal) variable $y_i$.

The observability care set of $y_i$ ($ODC^c_y$) is defined by the complement of $ODC_y$, i.e. by $ODC^c_y$. Its $k^{th}$ component describes the network configurations that make the variable $y_i$ observable at the $k^{th}$ output.

Given a network, it is possible in principle to compute $ODC_y$ for any internal variable $y$ by flattening the network $N'\epsilon$ and applying Eq. (2). We show here that it is possible to avoid the flattening operation on the network and compute exact and approximate versions of the functions $ODC_y$ with a single traversal of the network.

For example, if the don't care set of a vertex $\nu$ is known, then it is easy to obtain an expression of $ODC_y$ for any fanin variable $y$ of $\nu$ from [1]:

$$ODC_y = ODC_y + (1, 1, \ldots, 1)^T \frac{\partial f_\nu}{\partial y}.$$

\[1\] Notice that Eq. (3) may contain internal network variables. These, however, may be resolved by back-substitution to obtain an expression of primary input variables only, so that there is no real contrast with the definition (2).
The vector \( (1, 1, \cdots, 1)^T \) is used to add \( \partial f_\nu / \partial y \) to all the components of the vector \( ODC_\nu \).

If the network has a tree structure, then it is possible to obtain all the ODC sets by traversing backwards the network and applying iteratively Eq. (3). If a vertex has reconvergent fanout, however, the observability conditions of the vertex do not coincide with those of its fanout variables. We present here how the observability don't care of a vertex can be derived from those of its fanout variables.

### 2.3 Observability don't care sets in presence of reconvergent fanout

Consider a vertex \( \nu \) with reconvergent fanout, and suppose that all the edges \( e_1, e_2, \cdots, e_n \), whose tail-end point is \( \nu \), are cut. Let \( E'(x, y_1, y_2, \cdots, y_n) \) be the function realized by the new network \( N' \), obtained by adding the variables \( y_1, y_2, \cdots, y_n \), corresponding to the cut edges, to the primary inputs. Then, the observability don't care set of \( \nu \) is described by the function

\[
ODC_\nu = E'(x, 0, \cdots, 0) \Theta E'(x, 1, 1) \quad (4)
\]

or, equivalently, by

\[
ODC_\nu = E'(y_1, y_2, \cdots, y_n) \Theta E'(y_1, y_2, \cdots, y_n). \quad (5)
\]

For the sake of simplicity, we describe first the case in which \( n = 2 \), so that there are only two fanout variables, \( y_1 \) and \( y_2 \). We will generalize the result down below.

The observability don't care set is described by the function

\[
ODC_\nu = E'(y_1, y_2) \Theta E'(y_1, y_2) \quad (6)
\]

By manipulating Eq. (6), \( ODC_\nu \) can be rewritten as

\[
ODC_\nu = \left( E'(y_1, y_2) \Theta E'(y_1, y_2) \right) \Theta \left( E'(y_1, y_2) \Theta E'(y_1, y_2) \right) \quad (7)
\]

where the term \( E'(y_1, y_2) \) has been "added and subtracted" in Eq. (6).

From Eq. (2), the first term within parentheses is \( ODC_{y_1} | y_2 \), while the second parentheses describe \( ODC_{y_2} | y_1 \). It then follows that

\[
ODC_\nu = ODC_{y_1} | y_2 \Theta ODC_{y_2} | y_1 \quad (8)
\]

Notice that in Eq. (5), \( ODC_\nu \) can be also rewritten as

\[
ODC_\nu = \left( E'(y_1, y_2) \Theta E'(y_1, y_2) \right) \Theta \left( E'(y_1, y_2) \Theta E'(y_1, y_2) \right) = ODC_{y_1} | y_2 \Theta ODC_{y_2} | y_1 \quad (9)
\]

from which we obtain the identity

\[
ODC_\nu = ODC_{y_1} | y_2 \Theta ODC_{y_2} | y_1 = ODC_{y_1} | y_2 \Theta ODC_{y_2} | y_1 \quad (10a)
\]

Similarly, for the care set we obtain

\[
OC_\nu = OC_{y_1} | y_2 \Theta OC_{y_2} | y_1 = OC_{y_1} | y_2 \Theta OC_{y_2} | y_1 \quad (10b)
\]

These identities will be used in Sect. (3), when deriving approximations to \( ODC_\nu \).

The algebraic manipulation above can be extended to the general case of \( n \) fanout variables as follows. It can be easily verified that for \( n \geq 2 \) the following identity can be derived from Eq. (5):

\[
ODC_\nu = \left( E'(y_1, y_2, \cdots, y_n) \Theta E'(y_1, y_2, \cdots, y_n) \right) \Theta
\]
This can be rewritten as:

\[
ODC_v = \bigoplus_{i=1}^{n} \mathcal{E}^i | y_1, \ldots, y_{i-1}, y_{i+1}, \ldots, y_n
\]

From Eq. (2), each term of the sum in Eq. (11) is easily recognized to be \( ODC_{y_i} | y_1, \ldots, y_{i-1}, y_{i+1}, \ldots, y_n \).

Eq. (12) can be thus rewritten as

\[
ODC_v = \bigoplus_{i=1}^{n} ODC_{y_i} | y_1, \ldots, y_{i-1}, y_{i+1}, \ldots, y_n
\]

(13)

For the care set, we have:

\[
OC_v = \bigoplus_{i=1}^{n} OC_{y_i} | y_1, \ldots, y_{i-1}, y_{i+1}, \ldots, y_n
\]

(14)

Similarly to the case of 2 fanout variables, changing the order in which the fanout variables are complemented results in different expressions of \( OC_v \). For \( n \) fanout variables, there are \( n! \) possible orderings, hence \( n! \) different possible expressions for \( ODC_v \). In particular, let \((i_1, i_2, \ldots, i_n)\) denote a permutation of \((1, 2, \ldots, n)\). Then, the following identities hold:

\[
ODC_v = \bigoplus_{j=1}^{n} ODC_{y_{i_1}} | y_{i_1}, y_{i_{j-1}}, y_{i_{j+1}}, \ldots, y_n
\]

\[
OC_v = \bigoplus_{j=1}^{n} OC_{y_{i_1}} | y_{i_1}, y_{i_{j-1}}, y_{i_{j+1}}, \ldots, y_n
\]

2.4 An exact algorithm for the ODC sets computation

Given the ODC set of a vertex \( v \), it is possible to compute the ODC set of all its fanin edges by means of Eq. (3). In turn, Eq. (13) allows us to compute a vertex ODC set, given those of its fanout variables. It is thus now possible to visit the Boolean network backwards from the primary outputs to its inputs and to determine the ODC sets of each vertex also in presence of reconvergent fanout.

The following algorithm performs the computation of the ODC sets. It uses the subset \( S \) of the vertices whose ODC set is known. Initially \( S \) is the set of primary output vertices with empty fanout set.

\[
\text{OBSERVABILITY}(G);
\]

\[
S := \{ \text{primary output vertices with empty fanout set} \};
\]

\[
\text{while } ( S \neq V ) \{ \text{select } v \in \{ V - S \} \text{ such that } FO(v) \subseteq S; \}
\]

\[
\text{foreach fanout variable } y_i \text{ of } v \{ \mu = \text{head vertex of edge } e_i \text{ computes } ODC_{y_i} = ODC_v + (\partial y_i / \partial y_i) \};
\]

\[
\text{compute } ODC_v \text{ by Eq. (3) } /\star /
\]

\[
S := S \cup \{ v \};
\]

\[
\text{compute } ODC_v \text{ by Eq. (13) } /\star /
\]

\[
S := S \cup \{ v \};
\]

4
The algorithm is linear in the number of edges. We illustrate here its operation on the circuit shown in Fig. (1). At the beginning, \( S = \{ u_4, u_3 \} \), and
\[
ODC_{u_4} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} ; ODC_{u_3} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}
\]
The first and second component of the vectors \( ODC \) describe the observability with respect to \( z_1 \) and \( z_2 \), respectively.

First the vertices \( u_2, u_3 \) are considered. By applying Eqs. (3) and (13)
\[
ODC_{u_3} = ODC_{y_3}\big|_{y_4} \oplus ODC_{y_4}\big|_{y_3} = \begin{pmatrix} 1 \\ y_5 \end{pmatrix} \oplus \begin{pmatrix} y_6 \\ 1 \end{pmatrix} = \begin{pmatrix} y_6 \\ y_5 \end{pmatrix}
\]
\[
ODC_{u_2} = ODC_{y_2}\big|_{y_3} \oplus ODC_{y_3}\big|_{y_2} = \begin{pmatrix} y_4 \\ 1 \end{pmatrix} \oplus \begin{pmatrix} 1 \\ y_3 \end{pmatrix} = \begin{pmatrix} y_4 \\ y_3 \end{pmatrix}
\]
Then \( S = \{ u_2, u_3, u_4, u_5 \} \) and \( u_1 \) is selected. Its fanout variables are \( y_1, y_2 \), and according to Eq. (3):
\[
ODC_{y_1} = \begin{pmatrix} y_6 + z_1 \\ y_5 + z_1 \end{pmatrix} = \begin{pmatrix} y_2 \bar{x}_4 + z_1 \\ y_2 + x_4 + z_1 \end{pmatrix}
\]
\[
ODC_{y_2} = \begin{pmatrix} y_4 + z_4 \\ y_3 + z_4 \end{pmatrix} = \begin{pmatrix} y_1 \bar{x}_4 + z_4 \\ y_1 + x_4 + z_4 \end{pmatrix}
\]
so that, using Eq. (13),
\[
ODC_{u_1} = ODC_{y_1}\big|_{y_2} \oplus ODC_{y_2}\big|_{y_1} = \begin{pmatrix} x_4 + z_1 \\ x_4 + z_1 \end{pmatrix} \oplus \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} x_1 z_4 \\ z_1 + z_4 \end{pmatrix}
\]
which is the correct result.

Finally the algorithm computes the ODC sets of the primary inputs, that may be used as external ODC sets for the minimization of the logic feeding the circuit.

The product of all the components of \( ODC_{y_1} \) gives the conditions for which the gate output at vertex \( v \) is not observable at any output. For example, in the case of the gate in \( u_1 \), the product of the components of \( ODC_{y_1} \), yields \( z_1 z_4 \).

3 Computation of approximate ODC sets

It is of practical interest to consider the case in which the ODC sets are approximated by subsets, because of their size. Note that excess approximations of the ODC sets are of no practical value for Boolean minimization. Therefore it is useful to derive ODC subsets at a vertex from the ODC sets of its fanout vertices. Unfortunately, Eq. (7) may not yield an ODC subset from subsets of the actual ODC sets of each edge variable.

For example, in the circuit of Fig. (1), if we assume
\[
ODC_{y_1} = \begin{pmatrix} z_1 \\ z_1 \end{pmatrix}
\]
(actually a subset of the true \( ODC_{y_1} \)), we find for \( ODC_{u_1} \) the estimate:
\[
ODC_{u_1} = \begin{pmatrix} z_1 z_4 + z_1 \bar{x}_4 \\ z_1 \end{pmatrix}
\]
which is no longer a subset of the true \( ODC_{u_1} \).
Two different approaches for the computation of ODC subsets are considered here. In particular we show two formulae that can replace Eq. (13) in the previous algorithm. The first formula computes both care and don’t care sets of a vertex from those of its fanout variables. The second one, simpler but less accurate, computes ODC subsets only.

We give first formulae for computing $ODC_\nu$ in the case of two fanout variables. We then present their generalization to larger fanout.

### 3.1 Propagation of care and don’t care subsets

Let $y_1$ and $y_2$ denote the variables corresponding to the edges in the fanout of a vertex $\nu$. Assume that only subsets $ODC_{y_1}, ODC_{y_2}$ of $ODC_{y_1}, ODC_{y_2}$ are available. Then, $ODC_\nu$ and $DC_\nu$, computed by:

$$ODC_\nu = ODC_{y_1}|_{y_2} ODC_{y_2}|_{y_1} + ODC_{y_1}|_{y_2} ODC_{y_2}|_{y_1} + ODC_{y_1}|_{y_2} ODC_{y_2}|_{y_1} + ODC_{y_2}|_{y_1} ODC_{y_1}|_{y_2}$$

and by

$$DC_\nu = DC_{y_1}|_{y_2} DC_{y_2}|_{y_1} + DC_{y_1}|_{y_2} DC_{y_2}|_{y_1} + DC_{y_1}|_{y_2} DC_{y_2}|_{y_1} + DC_{y_2}|_{y_1} DC_{y_1}|_{y_2}$$

are subsets of the true $ODC_\nu$ and $DC_\nu$, respectively.

To verify Eq. (17), it is sufficient to observe that, by expanding Eq. (10a) in sum of products form,

$$ODC_\nu = ODC_{y_1}|_{y_2} ODC_{y_2}|_{y_1} + ODC_{y_1}|_{y_2} ODC_{y_2}|_{y_1} + ODC_{y_1}|_{y_2} ODC_{y_2}|_{y_1} + ODC_{y_2}|_{y_1} ODC_{y_1}|_{y_2}$$

Similarly, it can be verified from Eq. (10b) that

$$OC_\nu = OC_{y_1}|_{y_2} OC_{y_2}|_{y_1} + OC_{y_1}|_{y_2} OC_{y_2}|_{y_1} + OC_{y_1}|_{y_2} OC_{y_2}|_{y_1} + OC_{y_2}|_{y_1} OC_{y_1}|_{y_2}$$

Since we assumed $ODC_{y_i} \subseteq ODC_{y_i}, OC_{y_i} \subseteq OC_{y_i}, i = 1, 2$, the relations $ODC_\nu \subseteq ODC_\nu$ and $DC_\nu \subseteq OC_\nu$ are immediately verified.

### 3.2 Computation of don’t care subsets only

If only subset of don’t care conditions $ODC_{y_i}$ are available, then, $ODC_\nu$ can be computed from:

$$ODC_\nu = ODC_{y_1}|_{y_2} ODC_{y_2}|_{y_1} + ODC_{y_1}|_{y_2} ODC_{y_2}|_{y_1}$$

To verify Eq. (19) it is sufficient to observe that its right member matches the first elements in the sum (17a). Therefore, $ODC_{y_\nu}$, as computed by Eq. (19), is also contained in $ODC_{\nu}$.

Note that the ODC sets computed by Eq. (17a) always include those derived from Eq. (19). Therefore, the propagation of both care and don’t care sets provide a better approximation to the exact ODC sets.

The ODC subsets computed by Eq. (17a), (17b) and (19) have a local maximality property, in the sense that no other cube of the cofactors $ODC_{y_1}|_{y_2}, OC_{y_1}|_{y_2}$ appearing in those equations can be proved to be always contained in $ODC_{y_\nu}$. 
Figure 2: Multi-way fork decompositions of a fanout tree.

### 3.3 Generalization to larger fanout

A vertex \( \nu \) with fanout \( n > 2 \) can always be replaced by a multi-way fork of interconnections, as shown in Fig. (2). Eqs. (17) (or, more simply, Eq. (19)) could be applied iteratively to this fork. In this way a subset of \( ODC_{\nu} \) can be obtained in an \( O(n) \) number of applications of Eqs. (17) or (19).

The result, however, depends on the selected tree. Consider, for example, the case of three fanout variables, shown in Fig. (2), and suppose, for simplicity, to apply Eq. (19) only. If the fanout tree of \( \nu \) is decomposed as in Fig. (2b), the resulting expression of \( ODC_{\nu} \) is

\[
ODC_{\nu} = ODC_{\nu_1} | y_{12}, ODC_{\nu_2} | y_{13}, ODC_{\nu_3} | y_{23} + ODC_{\nu_1} | y_{13}, ODC_{\nu_2} | y_{13}, ODC_{\nu_3} | y_{12} + ODC_{\nu_1} | y_{13}, ODC_{\nu_2} | y_{12}, ODC_{\nu_3} | y_{13}
\]

If it is decomposed as in Fig. (2c), we instead obtain the following expression, different from the previous one:

\[
ODC_{\nu} = ODC_{\nu_1} | y_{12}, ODC_{\nu_2} | y_{13}, ODC_{\nu_3} | y_{23} + ODC_{\nu_1} | y_{12}, ODC_{\nu_2} | y_{13}, ODC_{\nu_3} | y_{23} + ODC_{\nu_1} | y_{12}, ODC_{\nu_2} | y_{13}, ODC_{\nu_3} | y_{23}
\]

The largest expression that can be obtained, starting from the variable don't care subsets, is

\[
ODC_{\nu} = ODC_{\nu_1} | y_{12}, ODC_{\nu_2} | y_{13}, ODC_{\nu_3} | y_{23} + ODC_{\nu_1} | y_{12}, ODC_{\nu_2} | y_{13}, ODC_{\nu_3} | y_{23} + ODC_{\nu_1} | y_{12}, ODC_{\nu_2} | y_{13}, ODC_{\nu_3} | y_{23}
\]

larger than (or equal to) both estimates (20) and (21).

In the general case, by expanding Eq. (13) in sum-of-products form, it is immediate to verify that the product

\[
ODC_{\nu_1} | y_{12}, \ldots, y_{n}, ODC_{\nu_2} | y_{13}, \ldots, y_{n-1}, y_n \ldots ODC_{\nu_n} | y_{12}, \ldots, y_{n}
\]

is certainly contained in \( ODC_{\nu} \). Let \( (i_1, i_2, \ldots, i_n) \) denote a permutation of \( (1, 2, \ldots, n) \). From the expression (15) of the exact \( ODC_{\nu} \), associated to that permutation it follows that

\[
ODC_{\nu_1} | y_{i_1}, \ldots, y_{i_n}, ODC_{\nu_2} | y_{i_1}, \ldots, y_{i_n} \ldots ODC_{\nu_n} | y_{i_1}, \ldots, y_{i_n}
\]

is also always contained in \( ODC_{\nu} \). Let \( \pi_n \) denote the set of all possible permutations \( i = (i_1, i_2, \ldots, i_n) \) of \( (1, 2, \ldots, n) \). The sum

\[
\sum_{i \in \pi_n} ODC_{\nu_1} | y_{i_1}, \ldots, y_{i_n}, ODC_{\nu_2} | y_{i_1}, \ldots, y_{i_n} \ldots ODC_{\nu_n} | y_{i_1}, \ldots, y_{i_n}
\]
is therefore a subset of $\text{ODC}_v$.

Similarly to the case of two fanout variables, it can be shown that no other product of the variables appearing in Eq. (25) can be added, while maintaining the first member of Eq. (25) an ODC subset. From the expansion of Eq. (13), all candidate products involve in fact the complement of at least one fanout variable ODC subset, i.e., an OC superset. If such a product is added, it is no longer possible to guarantee $\text{ODC}_G \subset \text{ODC}_v$. Eq. (25) therefore represents the largest possible estimate.

Unfortunately, its computation requires a factorial amount of time, and is therefore not applicable to large fanouts. Note, however, that for most circuits the number of fanout stems for a node is typically small.

When subsets of the true $\text{OC}_{y_i}$ sets are also available, then the largest possible subsets of $\text{ODC}_G, \text{OC}_v$ can be computed as follows. A sum-of-products expression of $\text{ODC}_G$, $\text{OC}_v$ can be obtained by expanding the expressions (13) and (14). By substituting each $\text{ODC}_{y_i}$ and $\text{OC}_{y_i}$ with $\text{ODC}_{G_{y_i}}$ and $\text{OC}_{G_{y_i}}$, respectively, an expression of ODC and OC subsets is obtained. Different subsets are obtained by repeating the same procedure using a permutation of the variables $y_1, \ldots, y_n$. The largest possible subset is the sum of these subsets.

4 Relation to previous works.

Methods for computing observability don't care sets have been proposed in [8], [3] and [7]. None of these methods maintains separate components for the observability at different outputs, and therefore must either make some form of approximations in case of reconvergent fanout, or abandon the network traversal method.

The most "conservative" approach is taken in the program MIS [8]. There, the observability don't care of a gate in a vertex $\nu$ is computed by assuming the complete observability of its fanout gates. The observability $\text{ODC}_G^{\text{MIS}}$ of each fanout variable $y_i$ of $\nu$ is thus obtained from Eq. (3) by neglecting the first term of the sum. MIS computes the observability don't care set of $\nu$ as

$$\text{ODC}_G^{\text{MIS}} = \prod_{y_i} \frac{\partial F}{\partial y_i}$$

$$\text{ODC}_{y_i} = \text{ODC}_G + \frac{\partial F}{\partial y_i} = \text{ODC}_G + \sum_{j=i, j \neq i} y_j$$

which corresponds (with a different formalism) to Eq. (3.1) in [3]. The cases of not-reconvergent and reconvergent fanout are treated separately. In the case of not reconvergent fanout, the observability of the gate output $\nu$ is computed (cfr. Eq. (3.4) in [3]) from those of the fanout variables $y_i$ as

$$\text{ODC}_G = \prod_{i} \text{ODC}_{G_{y_i}}$$

In the case of reconvergent fanout, since the different components of the observability are not kept separate, the network traversal paradigm must be abandoned. Muroga proposes an approach based upon two exhaustive simulations of the circuit, with the reconvergent gate output stuck-at 0 and stuck-at 1, respectively. The observability don't care set for the gate is then computed as the set of primary input configurations that produce the same output in the two simulations. In other words, Eq. (1) is solved by "brute force".

In [7] ODC subsets are computed as follows. Variables are associated to gate outputs rather than edges. Eq. (3) is used to determine the observability of fanin variables for each vertex.
lines, the observability don't care sets associated to each variable are considered as estimates of the true ODC of the vertex. Note that, because of the different definition of variables, in presence of reconvergent fanout the observability of a vertex seems to depend on itself. Consider, for example, the circuit of Fig. 1, and assume that the gate in u_4 does not exist. The method finds two estimates of the observability of u_1 as

\[ ODC'_{u_1} = u_1 + z_1; \quad ODC''_{u_1} = u_1 + z_4 \]

This apparent dependency is eliminated by invoking the RESTRICT operation on the expressions \( ODC'_{u_1}, ODC''_{u_1} \) which drops all the cubes containing \( u_1 \) or \( \overline{u}_1 \). The RESTRICT operation on an expression \( expr \) with respect to a variable \( var \) can be defined as follows:

\[ RESTRICT(expr, var) \triangleq expr | var \cdot expr | \overline{var} \]

In the example of Fig. 1,

\[ RESTRICT(ODC'_{u_1}) = ODC'_{u_1} | u_1 \cdot ODC'_{u_1} | \overline{u}_1 = z_1; \]
\[ RESTRICT(ODC''_{u_1}) = ODC''_{u_1} | u_1 \cdot ODC''_{u_1} | \overline{u}_1 = z_4. \]  

RESTRICT is therefore similar to the cofactor operation of Eqs. (8) and (17). Eq. (17) shows, however, that only part of the cubes (containing either \( u_1 \) or \( \overline{u}_1 \)) actually needs to be dropped.

The estimates obtained after the application of RESTRICT are finally ANDed. In the above example, \( ODC_{u_1} = z_1 z_4 \). It has been proven in [7] that the final result is a subset of the true ODC set. The exact result is, in the above example, \( ODC_{u_1} = z_1 + z_4 \geq z_1 z_4 \).

5 Summary

In this paper we have presented a novel algorithm for the extraction of exact and approximate Observability Don't Care sets (ODC sets) in a combinational multilevel logic network. The proposed algorithms are efficient because they use local information, i.e. the computation of the ODC for a vertex requires only the knowledge of the don't care at the vertex immediate fanout. Two approaches have been proposed. The former computes the exact ODC sets. The latter computes ODC subsets and can be used when partial information on the don't cares is available.

6 Acknowledgement

This research was supported in part by a fellowship of the Rotary Foundation and by AT & T and DEC, jointly with NSF, under a PYI award program. We also acknowledge support from NSF under contract # MIP 8719546.

The helpful discussions with Frederic Mailhot and Michiel Ligthart are here gratefully acknowledged.

References


