2

Multiple-Level Logic Optimization with Boolean Relations

Bernd Wurth[*]
Institute of Electronic Design Automation
Technical University of Munich, 80290 Munich, Germany
Tel.: +49-89-2105-3646. Fax: +49-89-2105-3696.
email: `wurth@regent.e-technik.tu-muenchen.de`

Norbert Wehn
Siemens AG, Semiconductor Division
81549 Munich, Germany
Tel.: +49-89-4144-4619. Fax: +49-89-4144-4888.
email: `wehn@vincent.hl.siemens.de`

Abstract

Exact and heuristic techniques are presented to calculate the Boolean relation for an arbitrary subcircuit in a multiple-level logic circuit with an external don't care set. We are not restricted to process subcircuits which are only driven by primary inputs. The new techniques keep BDD sizes small and therefore allow the calculation of Boolean relations for many circuits of nontrivial size which could not be dealt with before. The efficiency of the techniques is demonstrated on various benchmark circuits. The developed techniques are applied to multiple-level logic optimization.

Keywords

Boolean Relation, Observability Relation, Don't Cares, Multiple-Level Logic Optimization, Binary Decision Diagrams

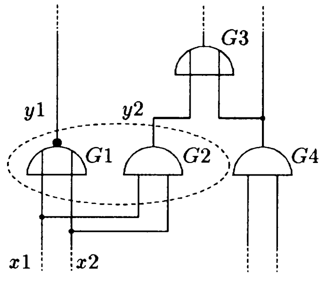
[*] The first author was supported by an Ernst von Siemens-grant given by Siemens AG.

1 INTRODUCTION

The ability to calculate and exploit *don't care* conditions has long been recognized an important technique in multiple-level logic synthesis. A variety of methods has been presented (Bartlett, 1988; Muroga, 1989; Damiani, 1993). Whereas a *don't care* set is calculated for a single node and specifies all the flexibility for implementing the node's function, *Boolean relations* describe all the flexibility for implementing an arbitrary multiple-output subcircuit. It has been pointed out before (Brayton, 1990) that Boolean relations are superior to don't care sets as don't care sets cannot capture the complete flexibility for implementing a multiple-output subcircuit. The reason is that for the calculation of a node's maximal don't care set, the functions of all the other nodes are not allowed to change. Boolean relations, however, allow the simultaneous modification of all nodes of a subcircuit. The example of Figure 1 illustrates the advantage of using Boolean relations.

Figure 1 shows a small section of a multiple-level logic circuit. The primary inputs and outputs of the circuit are omitted. We consider the subcircuit consisting of gate $G1$ and $G2$. The table on the right side specifies the output vectors which may be computed by the subcircuit if the respective input minterm is applied. This table represents a Boolean relation. For an input minterm with several output vectors, the values of the circuit's primary outputs do not depend on the output vector actually computed by the subcircuit. This flexibility, expressed by the Boolean relation, is due to the embedding of the subcircuit in a larger circuit. Here, the only degree of freedom exists for the input minterm $(1,1)$, as the subcircuit may compute either the vector $(0,1)$ or $(1,0)$. The actual implementation of the subcircuit computes the vector $(0,1)$. If the subcircuit is modified to compute the vector $(1,0)$ instead, gate $G1$ is replaced by an XNOR gate, and gate $G2$ by constant 0. Gate $G3$ can be replaced by a wire from $G4$, thus yielding a total gain of two gates. It is important to note that the subcircuit cannot be optimized by don't care methods because two node functions have to be modified at the same time to exploit the degree of freedom. Methods for the minimization of Boolean relations have been developed which find good two-level implementations (Watanabe, 1991). We, in contrast, are concerned with the calculation of the Boolean relation for a subcircuit.

Techniques have been presented (Chen, 1992) which calculate Boolean relations for multiple-output subcircuits in terms of primary input variables. Other methods (Cerny, 1977; Savoj, 1993) calculate the Boolean relation of a subcircuit in terms of internal



input minterm $(x1,x2)$	allowed output vector $(y1,y2)$
$(0,0)$	$(1,0)$
$(0,1)$	$(0,0)$
$(1,0)$	$(0,0)$
$(1,1)$	$(0,1),(1,0)$

Figure 1 Boolean relation for subcircuit $(G1,G2)$.

network variables. With regard to the specification of a subcircuit, the latter methods are therefore more general. Cerny and Marin laid the foundation for a theory of Boolean relations (Cerny, 1977). They present formulas to compute maximal Boolean relations for the subcircuits of a serially or parallelly two-partitioned circuit. These techniques can be used to calculate the Boolean relation for an arbitrary subcircuit. Recently, Savoj and Brayton expanded on these ideas and presented algorithms to compute maximal and sub-optimal ("compatible") Boolean relations (Savoj, 1993).

In this paper we focus on the problem of calculating maximal Boolean relations for subcircuits in a multiple-level logic circuit to use them for optimization. External don't care sets are exploited. A Boolean relation is expressed in terms of the subcircuit's outputs and inputs. New exact and heuristic techniques are presented for the calculation of Boolean relations. These methods are applicable to nontrivial circuits. We present results for the calculation and exploitation of Boolean relations.

The article is organized as follows. In Section 2 the basic terminology used throughout the paper is explained. Techniques to compute Boolean relations are discussed in Section 3. The application to multiple-level logic optimization is described in Section 4. We give conclusion and directions for future work in Section 5.

2 TERMINOLOGY

This section introduces some terminology for BDD (Binary Decision Diagrams) operations and Boolean relations. For the definition of a Boolean network, which is used to model a combinational circuit, we refer to standard literature (Brayton, 1990).

2.1 Boolean functions

We are dealing with Boolean functions $f : B^n \rightarrow B$, where $B = \{0,1\}$. Let $\mathbf{x} = (x_1, \ldots, x_n)$ denote a vector of Boolean variables and $\mathbf{x}^k \in B^n$ the k^{th} minterm, i.e., $\mathbf{x}^0 = (0, \ldots, 0)$, $\mathbf{x}^1 = (0, \ldots, 0, 1)$ etc. The number of elements of a vector is denoted by $|\mathbf{x}|$, here $|\mathbf{x}| = n$. A vector of Boolean functions is given by $\mathbf{f} = (f_1, \ldots, f_m)$. We will denote the Boolean function obtained from $f(\mathbf{x}, \mathbf{y})$ by assigning the minterm $\mathbf{x}^k$ as $h^k(\mathbf{y})$, i.e., $h^k(\mathbf{y}) = f(\mathbf{x}^k, \mathbf{y})$.

The *consensus operator* of a Boolean function $f(\mathbf{x}, \mathbf{y})$ with respect to the vector $\mathbf{x}$ is given by $\mathcal{C}_{\mathbf{x}} f = h^0(\mathbf{y}) \cdot h^1(\mathbf{y}) \cdot \ldots \cdot h^{2^n-1}(\mathbf{y}) = \prod_{k=0}^{2^n-1} h^k(\mathbf{y})$.

The *smoothing operator* of a Boolean function $f(\mathbf{x}, \mathbf{y})$ with respect to the vector $\mathbf{x}$ is given by $\mathcal{S}_{\mathbf{x}} f = h^0(\mathbf{y}) + h^1(\mathbf{y}) + \ldots + h^{2^n-1}(\mathbf{y}) = \sum_{k=0}^{2^n-1} h^k(\mathbf{y})$. The consensus and smoothing operations can be computed efficiently if applied to BDD representations of Boolean functions.

2.2 Boolean relations

A *Boolean relation* R is a binary relation from B^n to $B^m : R \subseteq B^n \times B^m$. We say that an output vector $\mathbf{y}^l \in B^m$ is *allowed* for an input minterm $\mathbf{x}^k \in B^n$ iff $(\mathbf{x}^k, \mathbf{y}^l) \in R$, e.g., two output vectors are allowed for the input minterm $(1,1)$ of the Boolean relation in Figure 1. For each Boolean relation R there is a *characteristic function* $\Phi : B^n \times B^m \rightarrow B$ such that $\Phi(\mathbf{x}^k, \mathbf{y}^l) = 1$ iff $(\mathbf{x}^k, \mathbf{y}^l) \in R$. As an example, the characteristic function of the

Boolean relation in Figure 1 is $\Phi(\mathbf{x},\mathbf{y}) = \overline{x_1} \cdot \overline{x_2} \cdot y_1 \cdot \overline{y_2} + \overline{x_1} \cdot x_2 \cdot \overline{y_1} \cdot \overline{y_2} + x_1 \cdot \overline{x_2} \cdot \overline{y_1} \cdot \overline{y_2} + x_1 \cdot x_2 \cdot \overline{y_1} \cdot y_2 + x_1 \cdot x_2 \cdot y_1 \cdot \overline{y_2}$.

We are dealing with characteristic functions of Boolean relations and represent them by BDDs. Because of the close correspondence between Boolean relations and characteristic functions, we sometimes use the term Boolean relation for a characteristic function. If not stated otherwise, the terms Boolean relation and characteristic function refer to a *maximal* Boolean relation, i.e., no input minterm/output vector-pair could be added to the Boolean relation without violating the circuit's specification.

Let a network N implement a Boolean function $\mathbf{z} = \mathbf{f}(\mathbf{x})$. The characteristic function which is the relational representation of the multiple-output function $\mathbf{z}$ is

$$\Phi_N^I(\mathbf{x},\mathbf{z}) = \prod_{i=1}^{|\mathbf{Z}|} (z_i \overline{\oplus} f_{z_i}(\mathbf{x})) . \tag{1}$$

A Boolean function $\mathbf{f} : B^n \to B^m$ is *compatible* with R iff $\forall \mathbf{x}^k \in B^n$, $(\mathbf{x}^k, \mathbf{f}(\mathbf{x}^k)) \in R$. The goal of Boolean relation minimization (Watanabe, 1991) is to find a function with minimum cost which is compatible with R.

3 NEW TECHNIQUES TO COMPUTE BOOLEAN RELATIONS

In this section, new techniques to calculate the Boolean relation for a subcircuit are derived. These techniques exploit the fact that in many practical cases a combinational circuit's specification is not an arbitrary relation, but a multiple-output function plus an *external don't care* set. An *external don't care* set denotes all input vectors that are never produced by the environment at the circuit's inputs. For example, during the synthesis of the combinational logic of a finite state machine, the set of unreachable states can be used as an external don't care set. Existing methods (Cerny, 1977; Savoj, 1993) handle arbitrary relational specifications. This, however, implies unnecessarily expensive computations in the above-mentioned case of a specification with an external don't care set.

Our new exact technique allows the computation of maximal Boolean relations in much larger circuits than existing methods if the circuit is specified with or without an external don't care set. This is achieved by the application of a new formula for the calculation of the *satisfiability relation*, yielding smaller BDD representations, and the use of efficient image computation techniques. Experimental results are given to compare the new techniques with existing approaches.

3.1 Satisfiability relation

In (Cerny, 1977) formulas are given to calculate the characteristic functions for the subcircuits in a serially or parallelly two-partitioned circuit. Two formulas needed in the sequel are restated here without explanation. A network N, serially decomposed into the subnetworks L and U (lower and upper part) is given, see Figure 2(a). The characteristic functions for L, U and N are $\Phi_L(\mathbf{x},\mathbf{y})$, $\Phi_U(\mathbf{y},\mathbf{z})$ and $\Phi_N(\mathbf{x},\mathbf{z})$.

If $\Phi_N(\mathbf{x},\mathbf{z})$ and $\Phi_U(\mathbf{y},\mathbf{z})$ are given and $\Phi_U = \Phi_U^I$ according to equation (1), the maximal Boolean relation $\Phi_{L_{max}}(\mathbf{x},\mathbf{y})$ is

$$\Phi_{L_{max}}(\mathbf{x},\mathbf{y}) = \mathcal{S}_{\mathbf{z}}\left(\Phi_N(\mathbf{x},\mathbf{z}) \cdot \Phi_U^I(\mathbf{y},\mathbf{z})\right). \tag{2}$$

As this Boolean relation follows from the restricted observability of the outputs of L at the primary outputs, it is called *observability relation.* The observability relation determines for each minterm $\mathbf{x}$ a set of allowed vectors $\mathbf{y}$, for which the upper part U computes allowed values at the primary outputs $\mathbf{z}$.

If $\Phi_N(\mathbf{x},\mathbf{z})$ and $\Phi_L(\mathbf{x},\mathbf{y})$ are given, the maximal Boolean relation $\Phi_{U_{max}}(\mathbf{y},\mathbf{z})$ is calculated as

$$\Phi_{U_{max}}(\mathbf{y},\mathbf{z}) = \mathcal{C}_{\mathbf{x}}\left(\Phi_N(\mathbf{x},\mathbf{z}) + \overline{\Phi_L(\mathbf{x},\mathbf{y})}\right). \tag{3}$$

As this Boolean relation follows from the restricted satisfiability of the inputs of U from the primary inputs, it is called *satisfiability relation.*

Whereas equation (2) exploits the fact that $\Phi_U = \Phi_U^I$, formula (3) holds for general Boolean relations Φ_L. We will use a formula for the satisfiability relation which exploits $\Phi_L = \Phi_L^I$.

Lemma 1 *If $\Phi_U(\mathbf{y},\mathbf{z})$ and $\Phi_L^I(\mathbf{x},\mathbf{y})$ are given, and $\Phi_N(\mathbf{x},\mathbf{z}) = \mathcal{S}_{\mathbf{y}}(\Phi_L^I(\mathbf{x},\mathbf{y}) \cdot \Phi_U(\mathbf{y},\mathbf{z}))$, the maximal Boolean relation $\Phi_{U_{max}}(\mathbf{y},\mathbf{z})$ is calculated as*

$$\Phi_{U_{max}}(\mathbf{y},\mathbf{z}) = \Phi_U(\mathbf{y},\mathbf{z}) + \overline{\mathcal{S}_{\mathbf{x}}\left(\Phi_L^I(\mathbf{x},\mathbf{y})\right)}. \tag{4}$$

Proof. We restate the lemma in the following way: Iff $\mathbf{z}^l$ is an output vector allowed for the input minterm $\mathbf{y}^k$, then $\Phi_{U_{max}}(\mathbf{y}^k,\mathbf{z}^l) = 1$. There are two cases: First, the input minterm $\mathbf{y}^k$ is computed by L for some primary input vector, i.e., $\mathbf{y}^k$ is satisfiable. Then no additional degree of freedom is added to the Boolean relation of U by the presence of subnetwork L. We have $\Phi_{U_{max}}(\mathbf{y}^k,\mathbf{z}^l) = 1$ iff $\Phi_U(\mathbf{y}^k,\mathbf{z}^l) = 1$ because the second term in equation (4) evaluates to 0. Second, the input minterm $\mathbf{y}^k$ cannot be computed by L, i.e., $\mathbf{y}^k$ is not satisfiable. Then any vector $\mathbf{z}^l$ is an allowed output of U. It holds that $\Phi_{U_{max}}(\mathbf{y}^k,\mathbf{z}^l) = 1$ because the second term in equation (4) evaluates to 1. $\square$

This formula, which can also be applied if an external don't care set is present, is relevant for multiple-level logic optimization because during resynthesis of U some network L is given. Consequently, $\Phi_L = \Phi_L^I$, which is exploited by equation (4). Using equation (4) instead of equation (3) has significant practical advantages as will be shown below.

3.2 Boolean relation of a subcircuit

Our goal is to calculate the Boolean relation for an arbitrary subcircuit or *block* of a Boolean network N'. Through structural analysis of network N', a subnetwork N is obtained which contains all nodes functionally related to the considered block. The network N can be further divided into three subnetworks, i.e., the block B, the lower part L and the upper part U. The subnetwork U consists of all the nodes which may influence the

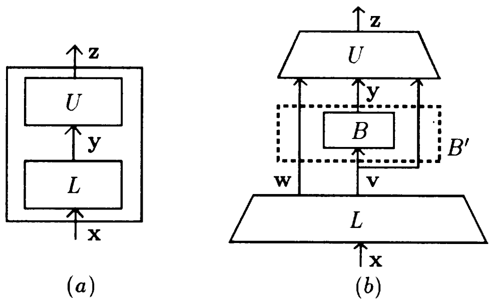


Figure 2 Decomposed network N.

observability of the block's outputs at the primary outputs. Likewise, the subnetwork L holds all the nodes which may influence the *satisfiability* of the block inputs or the inputs of U. Figure 2(b) shows the network N with its three parts. As the figure illustrates, block inputs can directly go into U, and some outputs of L may directly feed U.

For notational convenience, the same symbol, e.g. v_i, is used to denote a node and its variable in the Boolean network. We will denote the Boolean function of a node w_i in terms of the primary inputs $\mathbf{x}$ by $f_{w_i}(\mathbf{x})$. The Boolean functions of other nodes are denoted analogously. For each primary output z_i two Boolean functions, $f^a_{z_i}$ and $f^b_{z_i}$, are needed in the sequel. The function $f^a_{z_i}$ is the Boolean function of the primary output node z_i in terms of the variables $\mathbf{v}$ and $\mathbf{w}$, $f^a_{z_i}(\mathbf{v},\mathbf{w})$. The function $f^b_{z_i}$ is the function of z_i in terms of the variables $\mathbf{v},\mathbf{w}$ and $\mathbf{y}$, $f^b_{z_i}(\mathbf{v},\mathbf{w},\mathbf{y})$.

The new technique for the calculation of a Boolean relation is based on the following theorem.

Theorem 1 *Given a block B in a network N, which may have an external don't care set $exdc(\mathbf{x})$, the characteristic function of the maximal Boolean relation of B is given by*

$$\Phi_{B_{max}}(\mathbf{v},\mathbf{y}) = \mathcal{C}_{\mathbf{w}}\left(\Phi_{obs}(\mathbf{v},\mathbf{w},\mathbf{y}) + \overline{\Phi_{sat}(\mathbf{v},\mathbf{w})}\right), \tag{5}$$

with the observability relation $\Phi_{obs}(\mathbf{v},\mathbf{w},\mathbf{y}) = \prod_{i=1}^{|\mathbf{Z}|}\left(f^a_{z_i}(\mathbf{v},\mathbf{w})\overline{\oplus}f^b_{z_i}(\mathbf{v},\mathbf{w},\mathbf{y})\right)$, and the satisfiability function $\Phi_{sat}(\mathbf{v},\mathbf{w}) = \mathcal{S}_{\mathbf{X}}\left(\overline{exdc(\mathbf{x})}\cdot\prod_{i=1}^{|\mathbf{V}|}(v_i\overline{\oplus}f_{v_i}(\mathbf{x}))\cdot\prod_{i=1}^{|\mathbf{W}|}(w_i\overline{\oplus}f_{w_i}(\mathbf{x}))\right)$.

Proof. Combination of equation (2) and equation (4) yields the Boolean relation for B' (see Figure 2(b)):

$$\Phi_{B'_{max}}(\mathbf{v},\mathbf{w},\mathbf{y}) = \mathcal{S}_{\mathbf{Z}}\left(\Phi_{BU}(\mathbf{v},\mathbf{w},\mathbf{z})\cdot\Phi_U(\mathbf{v},\mathbf{w},\mathbf{y},\mathbf{z})\right) + \overline{\mathcal{S}_{\mathbf{X}}\left(\Phi_L(\mathbf{x},\mathbf{v},\mathbf{w})\right)}. \tag{6}$$

Using $\Phi_{BU} = \Phi^I_{BU}$ of the cascade of B and U, $\Phi_{BU}(\mathbf{v},\mathbf{w},\mathbf{z}) = \prod_{i=1}^{|\mathbf{Z}|}\left(z_i\overline{\oplus}f^a_{z_i}(\mathbf{v},\mathbf{w})\right)$, and similarly $\Phi_U = \Phi^I_U$, $\Phi_U(\mathbf{v},\mathbf{w},\mathbf{y},\mathbf{z}) = \prod_{i=1}^{|\mathbf{Z}|}\left(z_i\overline{\oplus}f^b_{z_i}(\mathbf{v},\mathbf{w},\mathbf{y})\right)$, we obtain for the first term on the right side of equation (6) the *observability relation*:

$$\begin{aligned}\Phi_{obs}(\mathbf{v},\mathbf{w},\mathbf{y}) &= \mathcal{S}_{\mathbf{Z}}\left(\Phi_{BU}(\mathbf{v},\mathbf{w},\mathbf{z})\cdot\Phi_U(\mathbf{v},\mathbf{w},\mathbf{y},\mathbf{z})\right)\\ &= \mathcal{S}_{\mathbf{Z}}\left(\prod_{i=1}^{|\mathbf{Z}|}\left(z_i\overline{\oplus}f^a_{z_i}(\mathbf{v},\mathbf{w})\right)\cdot\prod_{i=1}^{|\mathbf{Z}|}\left(z_i\overline{\oplus}f^b_{z_i}(\mathbf{v},\mathbf{w},\mathbf{y})\right)\right)\\ &= \prod_{i=1}^{|\mathbf{Z}|}\left(f^a_{z_i}(\mathbf{v},\mathbf{w})\overline{\oplus}f^b_{z_i}(\mathbf{v},\mathbf{w},\mathbf{y})\right).\end{aligned}$$

For the (not complemented) second term in equation (6), use of $\Phi_L = \Phi_L^I$ according to equation (1) and consideration of external don't cares yields the *satisfiability function* Φ_{sat}. To obtain $\Phi_{B_{max}}$ from $\Phi_{B'_{max}}$, formula (8) in (Cerny, 1977) is applied: $\Phi_{B_{max}}(\mathbf{v},\mathbf{y}) = \mathcal{C}_{\mathbf{W}}(\Phi_{B'_{max}}(\mathbf{v},\mathbf{w},\mathbf{y}))$. $\square$

Note that the onset of Φ_{sat} is the image of the possible primary input minterms, which are in the offset of $exdc(\mathbf{x})$, by the functions $\mathbf{f}_w, \mathbf{f}_v$. For comparison, equation (7) shows how the Boolean relation for block B is computed in (Cerny, 1977) and similarly in (Savoj, 1993),

$$\Phi_{B_{max}}(\mathbf{v},\mathbf{y}) = \mathcal{C}_{\mathbf{W},\mathbf{X}}\left(\Phi_{LB}(\mathbf{x},\mathbf{v},\mathbf{w},\mathbf{y}) + \overline{\Phi_L^I(\mathbf{x},\mathbf{v},\mathbf{w})}\right), \quad (7)$$

where the characteristic function for the cascade of B and L is denoted by Φ_{LB}. The derivation of equation (7) is given in the appendix. Note that equation (7) has also been used in (Kukimoto, 1992) in the context of design rectification for FPGAs.

Despite their structural similarity, the two formulas (5) and (7) differ in several important aspects: First, the observability relation Φ_{obs} in formula (5) depends on the internal variables $\mathbf{v},\mathbf{w},\mathbf{y}$. The corresponding function Φ_{LB} in formula (7) additionally depends on the primary input variables $\mathbf{x}$. Furthermore, the satisfiability function Φ_{sat} in formula (5) depends on the internal variables $\mathbf{v},\mathbf{w}$ only, whereas the function Φ_L in formula (7) also depends on the variables $\mathbf{x}$. In the worst case, if the block is located at the primary outputs, Φ_L and Φ_{LB} represent the functionality of almost the complete network N. As a consequence, the BDD representation of the argument of the consensus operation is much larger for formula (7) than for formula (5). Finally, to calculate the satisfiability function Φ_{sat} in formula (5), the characteristic function Φ_L need not be calculated explicitly. This is achieved by the use of image computation techniques, which have originally been developed for automata verification (Coudert, 1990). The satisfiability function is computed using image computation techniques as shown in (Savoj, 1991). Note that Φ_L has to be calculated explicitly in formula (7).

These differences are essential, because during the calculation of the block's characteristic function according to formula (5) much smaller BDDs are created. This enables us to handle larger circuits. Due to smaller BDD sizes, the calculation according to the new technique also proves to be significantly faster. However, since our technique is based on the exploitation of a special kind of specification (multiple-output function plus external don't care set), we cannot deal with arbitrary relational specifications.

We have implemented both methods according to the new formula (5) and formula (7)

Table 1 Comparison of Boolean relation computation techniques

Circuit	*In*	*Out*	*Literals*	*BDD nodes*			*CPU time in sec*		
				previous	*new*	*Ratio*	*previous*	*new*	*Ratio*
apex7	49	37	351	52776	12387	0.23	413.4	37.6	0.09
b9	41	21	256	NF	65846	-	-	27.1	-
c8	28	18	301	2450	826	0.34	3.6	1.4	0.39
cordic	23	2	194	32327	1987	0.06	163.7	5.5	0.03
count	35	16	174	1392	524	0.38	3.3	1.6	0.48
example2	85	66	432	124280	17346	0.14	64.2	32.0	0.50
f51m	8	8	319	70	70	1.0	0.1	0.1	1.0
i4	192	6	340	NF	1158	-	-	4.4	-
k2	45	45	3047	131533	20961	0.16	868.6	248.0	0.29
lal	26	19	257	1101	799	0.73	2.7	2.4	0.89
pcler8	27	17	102	102583	28187	0.27	47.5	8.3	0.17
term1	34	10	977	NF	149592	-	-	139.7	-
ttt2	24	21	592	2307	2034	0.88	5.9	3.8	0.64
s298	17	20	244	143862	6227	0.04	200.9	16.8	0.08
s344	24	26	269	NF	47622	-	-	42.2	-
s444	24	27	352	NF	35460	-	-	89.7	-
s526	24	27	445	NF	57609	-	-	284.0	-
s641	54	42	539	NF	63027	-	-	1751.2	-

using the same shared ROBDD package, which is based on (Brace, 1990). For the ordering of the BDD variables, a simple depth first search heuristic was used.

Table 1 shows results for the computation of Boolean relations for some MCNC benchmark circuits. The first column gives the name of the circuit. The next two columns give the number of primary inputs and outputs. For sequential circuits, the number of flip-flops is added to the number of inputs and outputs. The fourth column shows the circuit size in literals. The next two columns show the maximum number of BDD nodes during the calculation of a block's Boolean relation, where **previous** denotes the method based on formula (7), **new** the new technique based on formula (5). The following column **Ratio** shows the ratio between the BDD sizes obtained with the new and the previous method. The last three columns show the CPU time in seconds for the two different methods and again the ratio. An entry NF signifies that the shared ROBDD grew larger than 150000 nodes. Calculation was aborted at that point. The circuits have been partitioned into two-output blocks using a heuristic described in the next section. All experiments have been performed on a DECstation 5000/200.

There are some circuits, e.g. f51m, where both methods yield identical maximal BDD sizes. This happens when the characteristic functions $\Phi_{B'}$ of the blocks and not the intermediate functions have the largest BDD representation.

The results emphasize two aspects. First, Boolean relations can generally be calculated with significantly smaller BDDs and also much faster using the new technique. Many circuits, e.g. almost all sequential circuits in Table 1, cannot be handled by the previously known method. Secondly, the BDDs for large circuits are usually still large. Even the new method fails for the large sequential circuits not given in Table 1. This is mainly due to the large number of outputs w_i of subnetwork L which directly feed U.

To tackle this problem, we propose a heuristic. The functions f_{w_i} involved in the image computation strongly influence BDD sizes and computation time. Therefore, the outputs w_i of the lower part L are partitioned into two sets W' and W'', where W' contains a prespecified number of nodes which are structurally "close" to the block. By calculating the image only for the functions of the nodes in V and W', it is assumed that the functions of the nodes in W'' may produce any vector. This simplification decreases the result quality. The Boolean relations are not maximal any more, but their computation is in general sped up and becomes possible for large circuits. Note that Table 1 gives only results for the calculation of maximal Boolean relations.

4 APPLICATION TO MULTIPLE-LEVEL LOGIC OPTIMIZATION

We apply the new techniques to multiple-level logic optimization. Our algorithm is as follows. A network is first partitioned into blocks. The nodes in a block have to be "related" to one another in order to successfully exploit Boolean relations. We cluster nodes such that the nodes of a block have a large number of common successors and predecessors. For each block, the maximal Boolean relation according to formula (5) is calculated and exploited.

To exploit the Boolean relation, the optimal function compatible with the Boolean relation has to be found. Our cost function is the number of literals. As we do not have a Boolean relation minimizer at hand, we find a good implementation by exhaustively optimizing compatible functions with ESPRESSO (Brayton, 1994). First, however, we check if the Boolean relation allows to set a node function to constant 0 or 1, or to replace one block output function by another one. These situations usually yield high gain.

The algorithm was applied to a number of sequential benchmark circuits. First, for each circuit the set of unreachable states was calculated. These unreachable states served as external don't care set during optimization. The circuits were preprocessed by the first part of the SIS script *script.rugged*, which basically performs cube and kernel extraction. The circuits after preprocessing are the initial circuits.

We simplified the initial circuits with our algorithm. For comparison, the rest of the SIS script *script.rugged* was applied. Whereas our algorithm calculates and exploits Boolean relations, the command *full_simplify*, which is called in the script *script.rugged*, employs node simplification by don't care methods. Table 2 shows the obtained results. The column **Init** lists the number of literals of the initial circuits. The column **SIS 1.1** shows the results obtained by *script.rugged*, the column **BooRel** the number of literals obtained by our algorithm.

The size of the initial circuits could be reduced significantly using Boolean relations. BooRel performs better than SIS in terms of literals. However, CPU times are larger for BooRel because the computation of Boolean relations is more expensive than the computation of don't care sets. We believe that even better results can be obtained if the following problems are resolved.

First of all, the problem of partitioning a circuit has to be examined more thoroughly. Experiments showed that the optimization quality is very sensitive to the partitioning. Furthermore, a Boolean relation minimizer is necessary.

Table 2 Optimization results (in literals)

Circuit	*In*	*Out*	*Init*	*SIS 1.1*	*BooRel*
s298	17	20	126	97	89
s344	24	26	154	142	137
s382	24	27	166	150	137
s386	13	13	136	120	111
s400	24	27	160	143	136
s444	24	27	158	140	138
s526	24	27	222	154	135
s641	54	42	193	185	153

5 CONCLUSION

We have developed techniques to calculate the Boolean relation for a multiple-output subcircuit with arbitrary inputs. We exploit the fact that in many cases the circuit is specified by a multiple-output function and an external don't care set, which is a special relational specification. Experimental results show the superiority of our technique in terms of computation time and space to previous approaches. With the new technique, the BDDs remain much smaller during the calculation of characteristic functions. This enables us to apply the technique to many circuits of non-trivial size which could not be dealt with before. We applied the computation of Boolean relations to multiple-level logic optimization. Other possible applications are design rectification for FPGAs (Kukimoto, 1992) and technology mapping. Ongoing research concerns the partitioning of circuits into subcircuits, which has shown a significant influence on the quality of optimization results.

6 APPENDIX

With equation (2), the characteristic function Φ_N for the complete network, and the characteristic function Φ_U^I according to equation (1) for the upper part, we get the characteristic function for the subnetwork consisting of B and L:

$$\Phi_{LB}(\mathbf{x},\mathbf{v},\mathbf{w},\mathbf{y}) = \mathcal{S}_{\mathbf{z}}\left(\Phi_N(\mathbf{x},\mathbf{z}) \cdot \Phi_U^I(\mathbf{v},\mathbf{w},\mathbf{y},\mathbf{z})\right).$$

With formula (3), formula (8) in (Cerny, 1977), and the characteristic function according to equation (1) for the lower part, Φ_L^I, formula (7) is obtained:

$$\Phi_{B_{max}}(\mathbf{v},\mathbf{y}) = \mathcal{C}_{\mathbf{w},\mathbf{x}}\left(\Phi_{LB}(\mathbf{x},\mathbf{v},\mathbf{w},\mathbf{y}) + \overline{\Phi_L^I(\mathbf{x},\mathbf{v},\mathbf{w})}\right).$$

7 REFERENCES

Bartlett, K. A.; Brayton, R. K.; Hachtel, G. D.; Jacoby, R. M.; Morrison, C. R.; Rudell, R. L.; Sangiovanni-Vincentelli, A. and Wang, A. R. (1988) Multilevel logic simplification using implicit don't cares. *IEEE Transactions on Computer-Aided Design*, **7**, 723-40.

Brace, K. S.; Rudell, R. L. and Bryant, R. E. (1990) Efficient implementation of a BDD package. *27th ACM/IEEE Design Automation Conference DAC*, 40-5.

Brayton, R. K.; Hachtel, G. D.; McMullen, C. T. and Sangiovanni-Vincentelli, A. L. (1984) *Logic minimization algorithms for VLSI synthesis.* Kluwer Academic Publishers, Boston, MA.

Brayton, R. K.; Hachtel, G. D. and Sangiovanni-Vincentelli, A. L. (1990) Multilevel logic synthesis. *Proceedings of the IEEE*, **78**, 264-300.

Cerny, E. and Marin, M. A. (1977) An approach to unified methodology of combinational switching circuits. *IEEE Transactions on Computers*, **26**, 745-56.

Chen, K.-C. and Fujita, M. (1992) Efficient sum-to-one subsets algorithm for logic optimization. *29th ACM/IEEE Design Automation Conference, DAC* 443-8.

Coudert, O. and Madre, J. C. (1990) A unified framework for the formal verification of sequential circuits. *IEEE/ACM International Conference on Computer-Aided Design ICCAD*, 126-9.

Damiani, M. and Micheli, G. D. (1993) Don't care set specification in combinational and synchronous logic circuits. *IEEE Transactions on Computer-Aided Design*, **12**, 365-88.

Kukimoto, Y. and Fujita, M. (1992) Rectification method for lookup-table type FPGAs. *IEEE/ACM International Conference on Computer-Aided Design ICCAD*, 54-61.

Muroga, S.; Kambayashi, Y.; Lai, H. C. and Culliney, J. N. (1989) The transduction method - design of logic networks based on permissible functions. *IEEE Transactions on Computers*, **38**, 1404-24.

Savoj, H.; Brayton, R. K. and Touati, H. J. (1991) Extracting local don't cares for network optimization. *IEEE International Conference on Computer-Aided Design ICCAD*, 514-7.

Savoj, H. and Brayton, R. K. (1993) Observability relations for multi-output nodes. *Proceedings International Workshop on Logic Synthesis*

Watanabe, Y. and Brayton, R. K. (1991) Heuristic minimization of multiple-valued relations. *IEEE/ACM International Conference on Computer-Aided Design ICCAD*, 129-32.

8 BIOGRAPHY

Bernd Wurth, Ernst von Siemens scholar, is working towards his Ph.D. degree at the Department of Electrical Engineering of the Technical University of Munich, Germany. He studied Electrical Engineering at the University of Illinois at Urbana-Champaign and the Technical University of Munich, Germany, where he received a Dipl.-Ing. degree in 1992. His research interests are in logic synthesis and optimization and in test methods.

Norbert Wehn received the engineering degree and the Ph.D. degree in electrical engineering from the Darmstadt University of Technology, Germany, in 1984 and 1989, respectively. Since 1984 he has been working in the VLSI group at the Darmstadt University of Technology at CAD for physical design problems, defect-tolerant circuits and architectural synthesis. In 1991, he joined Siemens and is now working in the Semiconductor Division on multimedia. Dr. Wehn has authored or coauthored more than 20 papers on international conferences and journals in the cited fields.