

Rectangle Covering Factorization of ESOPs Into Scan-Based Levelized Circuits with Universal Test Set

Ugur Kalay¹, Marek A. Perkowski¹, Douglas V. Hall¹, B. Steinbach², Shah Amran Shahjahan¹

¹ Dept. of Electrical and Computer Engineering
Portland State University
P.O. Box 751, Portland, OR 97207-0751
ugurkal@ee.pdx.edu

² Freiberg University of Mining and Technology
Institute of Computer Science
D-09596 Freiberg, Germany
steinb@informatik.ba-freiberg.de

Abstract

Easily testable two-level AND-EXOR circuits have been investigated by many researchers. However, two-level AND/EXOR circuits (i.e. ESOP) can be area-consuming and slow; therefore they should be factorized. In this paper, a new AND-EXOR factorization method based on rectangle covering is presented. A factorized multi-level AND/EXOR circuit can be partitioned into planes of ESOP circuits, which can be tested separately using the scheme given in [10]. Therefore, in this paper we also show a new scan-based easily testable realization that uses a universal test set, and that suggests a deterministic Built-in Self Test (BIST).

1. INTRODUCTION

The transformation from a two-level form (i.e. SOP, POS) into a multi-level form can lead to very substantial reductions in area, complexity, and delay. Also, many logic gates have a certain fan-in limit, so it is not always practical to implement circuits using two-level expressions despite their high testability, ease of minimization, and regular layout realizations in programmable logic arrays (PLAs). Thus, multilevel circuits are used in practice, and one of the most often used ways to obtain them in industrial CAD systems is the iterated factorization methods [1, 2, 3, 6, 19, 20, 21, 27].

It is well-known that AND/EXOR circuits are in general more easily testable than AND/OR circuits [10, 13, 15, 16, 17]. However, there is not much work on multi-level AND/EXOR synthesis that targets area and testability optimization. Several researchers developed tools for multi-level logic synthesis based on EXOR gates. Saul gives an algorithm for multi-level synthesis of Reed-Muller representations in [19, 21]. His algorithm is based on a simple factorizer without much optimization. Chattopadhyay (et al.) created a tool for multi-level AND/EXOR design [6], but they did not use factorization and did not consider testability. Rajski (et al.) showed that the factorization of AND/OR expressions

preserves tests, which means that the factorized logic network can be tested with the same set of tests as the original network [14]. Tsai (et al.) showed that a very simple and restricted factorization of AND/EXOR expressions preserves some tests and is useful in realization of arithmetic functions [27]. However, the authors admit the poor quality of their factorization algorithm. Lee (et al.) discussed testability of multi-level AND/EXOR circuits [11], but these circuits are a very special case of circuits discussed by us here. Steinbach (et al.) proposed a method to design multi-level AND/OR/EXOR circuits that are fully testable [23], but their test set is not universal.

Other methods were proposed to increase the testability of a multi-level network during logic optimization by checking for a special type of Boolean resubstitution [4, 5, 7, 8], or after optimization by test point insertion [24, 25]. Toubia (et al.) also proposed a test point insertion that is performed during the logic optimization in [26]. However, none of the above methods specifically considered AND/EXOR designs, or they targeted only random pattern testability.

The goal of this paper* is to present a complete method to solve the problem of quasi-optimum factorization of AND/EXOR expressions, and propose a scan-based testing scheme which requires a minimal and universal test set. Brayton (et al.) solved the factorization problem for Sum-of-Products (SOP) expressions by using a rectangle covering technique [3]. They utilized the concept of overlapping rectangles to come up with optimal solutions. Overlapping rectangle techniques can also be used for EXOR-Sum-of-Products (ESOP) factorization. Therefore, in this paper we will define a rectangle covering approach, *Even-Odd Rectangle Covering* for the factorization of ESOPs. This is the first time that the rectangle covering approach has been used for ESOP circuits factorization.

In section 2, the basic information on factorizing SOP

*This research has been partially funded by Oregon Board of Higher Education.

and ESOP forms is given. Section 3 presents our method for efficiently factorizing ESOPs using rectangle covers. Section 4 discusses the testability of factorized ESOP forms. Conclusions and future directions are given in Section 5.

2. PRELIMINARIES

2.1. Basic Definitions

A *Boolean variable* is any variable whose domain is the set $\{0,1\}$. A *literal* is a Boolean variable or its complement (e.g. a or $\bar{a}$). A *cube* is simply a product of literals. However, it cannot consist of a literal and its negation (i.e. $ab\bar{a}$ is not a cube). An *expression* is a sum or EXOR-sum of cubes, e.g. $F = abc + bd\bar{f} + gh$ or $F = abc \oplus \bar{a}\bar{b}\bar{c}$.

The *support of an expression* is the set of all variables that occur in all literals of the expression, for instance for the expression $ab + \bar{b}\bar{c}$, the support set is $\{a,b,c\}$. Two or more expressions are said to be *disjoint* when they have no common elements in their support sets. For example, if $F = (a+b)$ and $G = (c+d)$, then $\text{sup}(F) \cap \text{sup}(G) = \emptyset$.

2.2. Factorization

A *factorized form* is an efficient representation of an equivalent Sum-of-Products (SOP) or EXOR-Sum-of-Products (ESOP) expression. It contains a smaller number of literals than the original SOP (ESOP) expression.

An *algebraic factorization* of a SOP (ESOP) expression can be obtained by simply factoring out (parenthesizing) the common literals, cubes, or higher-order SOP (ESOP) expressions in the original expression. For example, given the ESOP function

$$E = ac \oplus bc \oplus ad \oplus bd \oplus af \oplus bf,$$

an algebraic factorization is

$$\begin{aligned} E &= a(c \oplus d) \oplus b(c \oplus d) \oplus f(a \oplus b) \\ &= (a \oplus b)(c \oplus d) \oplus f(a \oplus b). \end{aligned}$$

An *optimum algebraic factorization* of an expression is the one with the least number of literals [9]. By definition, an optimum factorization is the product of disjoint SOPs (ESOPs), and/or Sum-of (EXOR-Sum-of) these products [3]. The factorized ESOP expression above is not optimal because it can be factorized further into a fewer number of literals, to give,

$$[E] = (a \oplus b)(c \oplus d \oplus f).$$

As our notation, the function name of an optimally factorized expression is written in brackets.

In some cases, other axioms of Boolean logic can be incorporated into simple algebraic factorization to obtain better factorizations. The **identity** law allows the following substitutions in an expression:

$$\begin{aligned} \bar{a} + 1 &= 1, \\ a + 1 &= 1, \\ a \oplus 1 &= \bar{a}, \\ \bar{a} \oplus 1 &= a. \end{aligned}$$

Example 1: Given the SOP function

$$F = ab + bd + ac + a\bar{b}c,$$

the factorized expression is

$$\begin{aligned} F &= b(a + d) + ac(1 + \bar{b}) \\ &= b(a + d) + ac. \end{aligned}$$

Example 2: Given the ESOP function

$$E = ab \oplus bd \oplus ac \oplus a\bar{b}c$$

the factorized expression is

$$\begin{aligned} [E] &= b(a \oplus d) \oplus ac(1 \oplus \bar{b}) \\ &= b(a \oplus d) \oplus acb \\ &= b(a \oplus d \oplus ac) \\ &= b(a(1 \oplus c) \oplus d) \\ &= b(a\bar{c} \oplus d). \end{aligned}$$

Notice that the non-factorized ESOP function of Example 2, E , has the same product terms as the non-factorized SOP function of Example 1, F . Yet, we see that E has been reduced to fewer literals than F . In this example, the difference in literal counts is only one. However, in larger functions the difference may be quite dramatic.

Another algebraic rule that we can use along with simple algebraic factorization is the **idempotence** law, which can be interpreted as adding more terms to the original expression. The addition of terms is acceptable as long as the new terms do not change the logical function implemented by the original expression. For AND/OR functions, better factorizations may be obtained by repeating one or more existing terms in the expression;

$$z = y + x + \underbrace{x + \dots + x}_{\text{additional terms}} = y + x.$$

Example 3: Given the SOP function

$$F = \underline{gd + gb + fd + bf} + \underline{ec + cf} + be,$$

after simple algebraic factorization,

$$F = (g + f)(d + b) + c(e + f) + be,$$

which has 9 literals. Next, we add an extra term, “ bf ”, and then factorize the expanded expression,

$$\begin{aligned} F &= \underline{gd + gb + fd + bf} + \underline{bf + be + ec + cf} \\ [F] &= (g + f)(d + b) + (e + f)(b + c), \end{aligned}$$

which has 8 literals. Thus, using the idempotence law we saved one literal and obtained the optimal factorized form.

For AND/EXOR expressions, we may also obtain better factorizations by repeating one or more existing or non-existing terms in the expression, provided that the terms are added in even sets only;

$$\begin{aligned} z &= y \oplus x \oplus \underbrace{x \oplus \cdots \oplus x}_{\text{additional existing terms}} = y \oplus x, \\ z &= y \oplus x \oplus \underbrace{z \oplus \cdots \oplus z}_{\text{additional non-existing terms}} = y \oplus x. \end{aligned}$$

Example 4: Given the ESOP function

$$E = zg \oplus gh \oplus xf \oplus yf \oplus yz \oplus xh,$$

after simple algebraic factorization,

$$E = z(g \oplus y) \oplus f(x \oplus y) \oplus h(g \oplus x),$$

which has a cost of 9 literals. However, after we add an extra non-existing term (“ xz ”) twice to the function,

$$E = \underline{xz \oplus xf \oplus yz \oplus yf} \oplus \underline{xz \oplus xh \oplus gz \oplus gh},$$

is factorized to

$$[E] = (x \oplus y)(z \oplus f) \oplus (x \oplus g)(z \oplus h),$$

which has a cost of 8 literals.

As shown in the above examples, SOP (ESOP) minimization rules (i.e. identity, idempotence) can be a part of the factorization procedure. In the next section, we will see how these laws can be applied to the rectangle covering method modified for ESOP factorization.

3. ESOP FACTORIZATION USING RECTANGLE COVERS

In this section the Even-Odd Rectangle Covering method is presented. First, some fundamentals to our approach will be given.

3.1. Further Definitions

A *cube free* expression is the one that cannot be divided evenly by a cube [9]. For example, $xy \oplus z$ is cube-free, but $xy \oplus zy$ is not cube-free because it can be evenly divided by the cube y . Also, for an expression to be cube-free, it must consist of at least two cubes (e.g. xyz is never cube-free because it can be divided evenly by any subset of its literals).

A *kernel* is defined as the cube-free quotient, K , of any expression, F , divided by a cube, C , such that, $K = F/C$ [9]. For example, if $F = xyz \oplus xywv \oplus tzv$ and $C = xy$, then $K = F/xy = z \oplus wv$, which is a cube-free quotient, and therefore a kernel. Also, note that a kernel must consist of at least two cubes due to the cube-free criterium. For completeness, we denote the cube C as the *co-kernel*, which divides the function F to obtain the kernel K .

We define the *level of a kernel*, such that a kernel is of *level-0* if it has no kernels except itself, and a kernel is of *level- n* if it has at least one kernel of *level- $(n-1)$* , but no kernels (except itself) of *level- n* or greater [9]. For example, for a factorized function

$$F = c(\bar{b}(f \oplus e) \oplus gb) \oplus d(e(b \oplus a) \oplus g \oplus f),$$

the kernels of different levels are:

$$\begin{aligned} K^0(f) &= f \oplus e && (\text{level-0 kernel}) \\ K^1(f) &= \bar{b}(f \oplus e) \oplus gb && (\text{level-1 kernel}) \\ K^2(f) &= F && (\text{level-2 kernel}), \end{aligned}$$

where $K^0(f) \subset K^1(f) \subset K^2(f)$.

A *CoKernel-Cube matrix*, M , is a matrix such that each row corresponds to a unique co-kernel of the expression and each column corresponds to a unique cube of the kernels found from the expression. A cell in the matrix, M_{RC} , is k if the cube C exist in the co-kernel R , where k represents a unique number of a term in the original expression; otherwise, the cell M_{RC} is left empty. A *rectangle* is simply a covering of any number of different combinations of rows and columns containing non-empty cells. A *prime rectangle* is a rectangle such that none of its cubes are covered by other rectangles. A *redundant rectangle* is a rectangle such that all of its cubes are covered by other rectangles. Figure 1 illustrates different rectangles in a CoKernel-Cube matrix. Prime rectangles, non-prime rectangles, and redundant rectangles are denoted in the figure by ‘A’, ‘B’, and ‘C’, respectively. Notice that a rectangle is not necessarily formed by only the elements on its corners. Therefore, all the elements of a rectangle are circled and the circles are connected with lines. This can be observed in the rectangle formed by the elements $\{M_{32}, M_{33}, M_{42}, M_{43}, M_{52}, M_{53}\}$ in Figure 1.

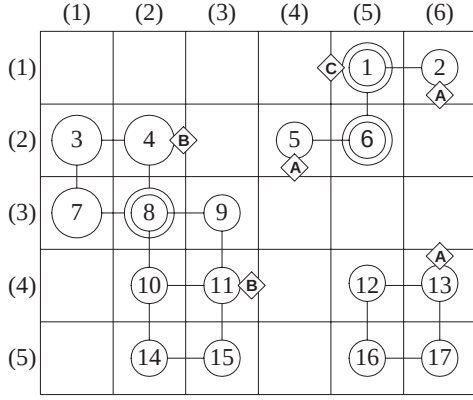


Figure 1. Rectangles in a CoKernel-Cube matrix.

Example 5: For the function $E = ab \oplus ac \oplus ad \oplus bc \oplus bd$, numbering each term we obtain: $ab = 1$, $ac = 2$, $ad = 3$, $bc = 4$, $bd = 5$. The CoKernel-Cube matrix is shown in Figure 2.

	(1)	(2)	(3)	(4)
	a	b	c	d
(1) a		1	2	3
(2) b	1		4	5
(3) c	2	4		
(4) d	3	5		

Figure 2. The CoKernel-Cube matrix for Example 5.

CoKernels	Cubes	Kernels
a	c, d	$c \oplus d$
b	c, d	$c \oplus d$
c	a, b	$a \oplus b$
d	a, b	$a \oplus b$

For the rectangle covers connected with dashed lines, we obtain the non-optimal factorized form,

$$E = a(b \oplus c \oplus d) \oplus b(c \oplus d).$$

For the rectangle covers connected with solid lines, we obtain the optimal factorized form,

$$[E] = (a \oplus b)(c \oplus d) \oplus ab.$$

Notice that we did not form the rectangle $\{M_{12}, M_{13}\}$ to find the optimal factorized form, rather we created a

single-cube rectangle, $\{M_{12}\}$. However, in the case of Figure 1 the non-prime rectangles shown are necessary to obtain the optimal solution.

3.2. Odd Rectangle Covering

Odd rectangle covering is a natural consequence of allowing an existing term of an ESOP expression to be repeated in odd sets, which occurs when an existing term is added even number of times. Therefore, in the CoKernel-Cube matrix, the repeated term is overlapped odd number of times. The following example illustrates odd rectangle covering.

Example 6: For the expression given below

$$E = \begin{matrix} ac & bc & ad & bd & cd & de & ae & ce & af & ef \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \end{matrix}$$

the CoKernel-Cube matrix is shown in Figure 3. In the figure, term $M_{41}(ad)$, denoted by '3', is overlapped three times (odd covering), which also means that it is added twice to the original expression.

	(1)	(2)	(3)	(4)	(5)	(6)
	a	b	c	d	e	f
(1) a			1	3	7	9
(2) b			2	4		
(3) c	1	2		5	8	
(4) d	3	4	5		6	
(5) e	7		8	6		10
(6) f	9				10	

Figure 3. The CoKernel-Cube matrix for Example 6.

The factorized form obtained without odd rectangle covering (dashed lines),

$$E = (a \oplus b)(c \oplus d) \oplus d(c \oplus e) \oplus e(a \oplus c) \oplus f(a \oplus e),$$

has 13 literals. The factorized form with odd rectangle covering (solid lines),

$$[E] = (a \oplus b)(c \oplus d) \oplus (a \oplus c)(d \oplus e) \oplus (a \oplus e)(d \oplus f),$$

has 12 literals. Hence, there is a literal savings of one, by overlapping the term "ad" three times.

3.3. Even Rectangle Covering

Even rectangle covering is a natural consequence of allowing a non-existing term to be added to the original ESOP expression in even sets ($x \oplus x = 0$).

Example 7: For the expression given below

$$E = \underset{1}{bc} \oplus \underset{2}{ad} \oplus \underset{3}{bd} \oplus \underset{4}{ce} \oplus \underset{5}{af} \oplus \underset{6}{ef},$$

the CoKernel-Cube matrix is shown in Figure 4. The non-existing term $M_{31}(ac)$, denoted by '7', is added and overlapped twice (even covering). The factorized form

	(1) <i>a</i>	(2) <i>b</i>	(3) <i>c</i>	(4) <i>d</i>	(5) <i>e</i>	(6) <i>f</i>
(1) <i>a</i>			7	2		5
(2) <i>b</i>			1	3		
(3) <i>c</i>	7	1				4
(4) <i>d</i>	2	3				
(5) <i>e</i>			4			6
(6) <i>f</i>	5					6

Figure 4. The CoKernel-Cube matrix for Example 7.

without even rectangle covering (dashed lines) is:

$$E = c(b \oplus e) \oplus d(a \oplus b) \oplus f(a \oplus e),$$

which has 9 literals. The optimal factorized form with even rectangle covering (solid lines) is:

$$[E] = (a \oplus b)(c \oplus d) \oplus (c \oplus f)(a \oplus e),$$

which has 8 literals, and saves one literal. Note that the non-existing addition technique of AND/OR expressions does not exist for AND/OR expressions. If the original expression in the above example was a SOP with the same product terms, the factorized expression would be,

$$F = c(b + e) + d(a + b) + f(a + e),$$

with 9 literals. The next example has cubes larger than only single-literal cubes for the rows and columns of the CoKernel-Cube matrix.

Example 8: Given the expression

$$E = \underset{1}{ab}xyz \oplus \underset{2}{cd}xyz \oplus \underset{3}{ab}fg \oplus \underset{4}{nm}cd \oplus \underset{5}{nm}uv \oplus \underset{6}{uv}fg$$

CoKernels	Kernels
<i>ab</i>	$xyz \oplus fg$
<i>cd</i>	$xyz \oplus nm$
<i>xyz</i>	$ab \oplus cd$
<i>fg</i>	$ab \oplus uv$
<i>nm</i>	$cd \oplus uv$

the CoKernel-Cube Matrix is shown in Figure 5.

	<i>ab</i>	<i>cd</i>	<i>xyz</i>	<i>fg</i>	<i>uv</i>	<i>nm</i>
<i>ab</i>			1	3		
<i>cd</i>			2	7		4
<i>xyz</i>	1	2				
<i>fg</i>	3	7			6	
<i>uv</i>				6		5
<i>nm</i>		4			5	

Figure 5. The CoKernel-Cube matrix for Example 8.

Term “ $cdfg$ ”, denoted by 7, is added twice to obtain the optimal factorized form below,

$$[E] = (ab \oplus cd)(xyz \oplus fg) \oplus (fg \oplus nm)(cd \oplus uv).$$

3.4. Even-Odd Rectangle Covering with Tautology Test

In the examples given above, we have dealt with ESOP expressions including only positive polarity (non-negated) variables. In addition, only the idempotence law (adding terms) has been adopted. However, the identity law is also important when factorizing mixed polarity ESOP expressions.

Example 9: Given the ESOP function

$$E = \underset{1}{ac} \oplus \underset{2}{\bar{b}c} \oplus \underset{3}{\bar{a}d} \oplus \underset{4}{bd}$$

the even-odd rectangle covering shown in Figure 6 yields,

	<i>a</i>	$\bar{b}$	$\bar{a}$	<i>b</i>
<i>c</i>	1	2		
<i>d</i>			3	4

Figure 6. The CoKernel-Cube matrix for Example 9.

$$E = c(a \oplus \bar{b}) \oplus d(\bar{a} \oplus b),$$

which is not optimal. If we use the identity law and re-factorize the expression algebraically,

$$\begin{aligned} E &= c((\bar{a} \oplus 1) \oplus (b \oplus 1)) \oplus d(\bar{a} \oplus b) \\ &= c(\bar{a} \oplus b \oplus 1 \oplus 1) \oplus d(\bar{a} \oplus b) \\ &= c(\bar{a} \oplus b) \oplus d(\bar{a} \oplus b) \\ [E] &= (\bar{a} \oplus b)(c \oplus d), \end{aligned}$$

which is optimal. First time we tried factorization we could not perform further factorization because we did not recognize that $(a \oplus \bar{b}) = (\bar{a} \oplus b)$, an identity. It may be even harder to determine on which variable to apply the identity law when there are larger and more complex terms in the expression. For example, $\bar{a} \oplus a\bar{b} \oplus cd = \bar{c} \oplus ab \oplus c\bar{d}$ is not easy to observe. However if we apply identity to all variables to convert them into positive polarity, we obtain a unique (canonical) representation, *Positive Polarity Reed Muller* (PPRM) [18], and we can easily determine the cubes that have common co-kernels. The equality of the two expressions in the above example can then be shown as:

$$\begin{aligned} \bar{a} \oplus a\bar{b} \oplus cd &= \bar{c} \oplus ab \oplus c\bar{d} \\ (a \oplus 1) \oplus a(b \oplus 1) \oplus cd &= (c \oplus 1) \oplus ab \oplus c(d \oplus 1) \\ a \oplus 1 \oplus ab \oplus a \oplus cd &= c \oplus 1 \oplus ab \oplus cd \oplus c \\ ab \oplus cd \oplus 1 &= ab \oplus cd \oplus 1. \end{aligned}$$

Therefore, to be able to use even-odd rectangle covering for the mixed polarity ESOP functions, we need to apply identity law ($\bar{x} \oplus 1 = x$) to the negated variables to obtain the PPRM form, and include the cube '1' in the CoKernel-Cube matrix. The following example illustrates this process.

Example 10: Given the ESOP function

$$E = ac\bar{d} \oplus aef \oplus \bar{b}e\bar{f} \oplus \bar{b}cd \oplus a\bar{c} \oplus \bar{b}\bar{c},$$

the even-odd rectangle covering without applying the identity law gives:

$$E = a(\bar{c} \oplus c\bar{d} \oplus ef) \oplus \bar{b}(\bar{c} \oplus e\bar{f} \oplus cd),$$

which is not optimal. If we apply identity before the rectangle covering, we obtain the PPRM form,

$$E = \begin{matrix} 1 & a & b & cd & ef & acd & aef & bcd & bef, \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \end{matrix}$$

and according to the rectangle covering shown in Figure 7, we obtain the factorized form,

$$[E] = (a \oplus b \oplus 1)(cd \oplus ef \oplus 1),$$

	1	a	b	cd	ef
1	1	2	3	4	5
a	2			6	7
b	3			8	9
cd	4	6	8		
ef	5	7	9		

Figure 7. The CoKernel-Cube matrix for Example 10.

which is the optimal positive polarity factorized form. We can obtain a mixed polarity factorized form as

$$\begin{aligned} [E] &= (a \oplus \bar{b})(\bar{c} \oplus ef) \\ &= (a \oplus \bar{b})((\bar{c} + \bar{d}) \oplus ef) \\ &= (a \oplus \bar{b})(\bar{c} \oplus c\bar{d} \oplus ef). \end{aligned}$$

Notice that we obtained the optimal mixed polarity factored form by applying the identity law again ($x \oplus 1 = \bar{x}$) to the final factored form found from the rectangle covering. However, the kernels obtained by this process may not be minimal ESOPs. Therefore, an ESOP minimizer such as given in [22] can be used on these cubes. Since the minimized kernels may not be cube-free, a further even-odd rectangle covering should be performed. This loop of even-odd rectangle covering and ESOP minimization should be performed until an optimally factorized form with minimal kernels is obtained.

3.5. Nested Iterations of Rectangle Covering

From the definition, we know that after we factor out (parenthesize) the co-kernels in the original expression, the kernels are cube-free. However, the factored out co-kernels may not be cube-free once they are combined in a factor term. In this case, the even-odd rectangle covering should be performed on the non-cube-free factor terms iteratively. This is also the case where multiple-level kernels are formed in the factorized expression. The following example shows the iterative process.

Example 11: Given the ESOP function below,

$$E = \begin{matrix} h & a & df & aef & bdf & bef & bfg & cdf & cef, \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \end{matrix}$$

we obtain the following factorized form from the first iteration of rectangle covering shown in Figure 8.

$$E = h \oplus (af \oplus bf \oplus cf)(d \oplus e) \oplus bfg.$$

	1	a	b	c	d	e	g
h	1						
af					2	3	
bf					4	5	6
cf					7	8	
df		2	4	7			
ef		3	5	8			

Figure 8. The rectangle covering in the first iteration.

Notice that the factorized form is not optimal because it has the non-cube-free term,

$$E' = af \oplus bf \oplus cf,$$

on which the second iteration is performed. We obtain the following factorized form for the non-cube-free term according to the rectangle covering shown in Figure 9.

$$E' = f(a \oplus b \oplus c).$$

	a	b	c
f	1	2	3

Figure 9. The rectangle covering in the second iteration.

The third iteration is performed again on the entire expression,

$$E = h \oplus f(a \oplus b \oplus c)(d \oplus e) \oplus bfg.$$

We obtain the following factorized form according to the rectangle covering shown in Figure 10.

$$[E] = h \oplus f((a \oplus b \oplus c)(d \oplus e) \oplus bg).$$

	1	(a⊕b⊕c)(d⊕e)	bg
h	1		
f		2	3

Figure 10. The rectangle covering in the third iteration.

The iterations stop at this point since there is no non-cube-free factor terms remaining. Notice in the final form that $(a \oplus b \oplus c)$ is a *level-0* kernel, and $((a \oplus b \oplus c)(d \oplus e) \oplus bg)$ is a *level-1* kernel.

3.6. Rectangle Covering for Multi-Output Functions

The following example generalizes the Even-Odd Matrix Covering method for multi-output functions.

Example 12: Given the multi-output ESOP function

$$\begin{aligned} X &= af \oplus bde \oplus bf \oplus cde \oplus cg \oplus ag \\ Y &= ace \oplus af \oplus bce \oplus bf \\ Z &= ade \oplus cde, \end{aligned}$$

CoKernels

de
f
g
ce
f
de

Kernels

b ⊕ c for X
a ⊕ b for X
c ⊕ a for X
a ⊕ b for Y
a ⊕ b for Y
a ⊕ c for Z,

the combined CoKernel-Cube matrix is shown in Figure 11.

		a	b	c	ce	de	f	g
X	a					13	1	6
	b					2	3	
	de	13	2	4				
	f	1	3					
	c					4		5
Y	g	6		5				
	a				7		8	
	b				9		10	
Z	ce	7	9					
	f	8	10					
Z	de	11		12				

Figure 11. The combined CoKernel-Cube matrix for Example 12.

Term “ ade ”, denoted by 13, is added twice for optimum factorized form for function X . The entire factorized network then is:

$$\begin{aligned} [X] &= U(de \oplus f) \oplus V(de \oplus g) \\ [Y] &= U(ce \oplus f) \\ [Z] &= Vde \\ U &= a \oplus b \\ V &= a \oplus c, \end{aligned}$$

with a total of 19 literals. Although each function is factorized separately, creating a combined CoKernel-Cube matrix helps recognize common factors (U and V) for further reduction of the literal count.

4. TESTABILITY OF FACTORIZED ESOPS

In this section, a scan-based multi-level AND/EXOR testing scheme that requires a minimal and universal test set is presented. Remember that an optimal factorized AND/EXOR expression consists of products of disjoint ESOPs and/or EXOR-sums of these products. Therefore, if the disjoint ESOPs in the factorized expression are identified, they can be separated during synthesis as individual ESOP planes with scan registers inserted between them. As a result, they can be tested separately since each ESOP plane will have total controllability on its inputs, and total observability on its output.

4.1. Identification of ESOP Planes

By definition, each kernel is in fact an ESOP plane. Different levels of kernels (ESOPs) can then be identified as shown in the following example.

Example 13: If the factorized function is given as below,

$$[E] = l((a \oplus b\bar{c})(d \oplus \bar{e}\bar{f} \oplus \bar{g}h) \oplus kmn \oplus \bar{k}\bar{m}\bar{n}) \oplus \bar{p}r \oplus \bar{s}\bar{t},$$

the level-0 ESOPs are:

$$\begin{aligned} E_1^0 &= a \oplus b\bar{c} \\ E_2^0 &= d \oplus \bar{e}\bar{f} \oplus \bar{g}h, \end{aligned}$$

where E_j^i represents the i^{th} kernel of level- j . The level-1 ESOP is:

$$E_1^1 = E_1^0 E_2^0 \oplus kmn \oplus \bar{k}\bar{m}\bar{n},$$

and the level-2 ESOP is:

$$E_1^2 = [E] = lE_1^1 \oplus \bar{p}r \oplus \bar{s}\bar{t}.$$

4.2. Implementation of an ESOP Plane

An easily testable ESOP realization with increased controllability and observability with a few additional logic gates, and a minimal-universal test set for the realization were introduced in [10]. The testing scheme requires only $(n+6)$ test patterns to detect a single stuck-at fault in the internal lines or the primary inputs/outputs of the circuit, where n is the number of primary inputs. Figure 12 shows the generic highly testable ESOP implementation and the universal test set as was given in [10].

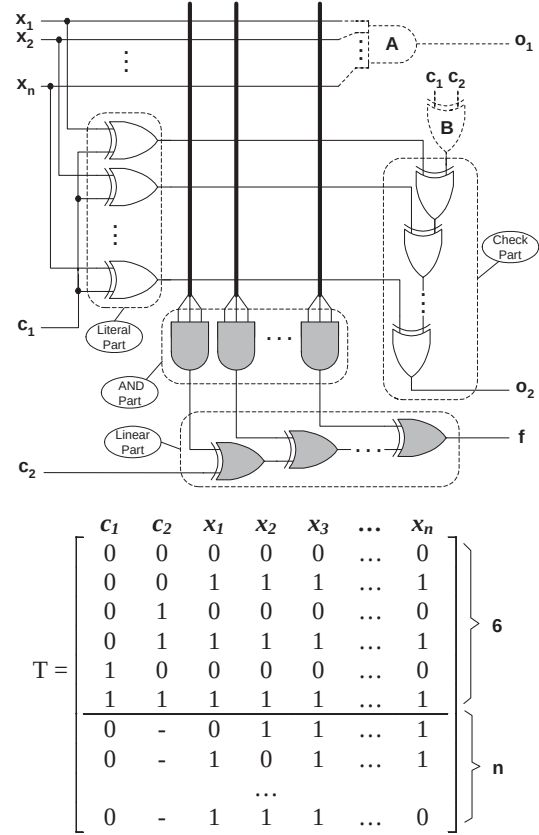


Figure 12. Highly testable generic ESOP implementation and the universal test set as presented in [10].

The realization requires a cascade implementation of the EXOR level in order to be able to obtain a universal test set. As also illustrated in [10], regular (universal) and a minimal number of test patterns allow generating them internally with a special *ESOP Deterministic Pattern Generator*, EDPG. In the complete BIST (Built-in Self Test) circuitry proposed in [10], EDPG applies the $(n+6)$ tests to the highly testable ESOP circuit, and the results are collected and compacted in a multiple-input signature register for the final signature analysis.

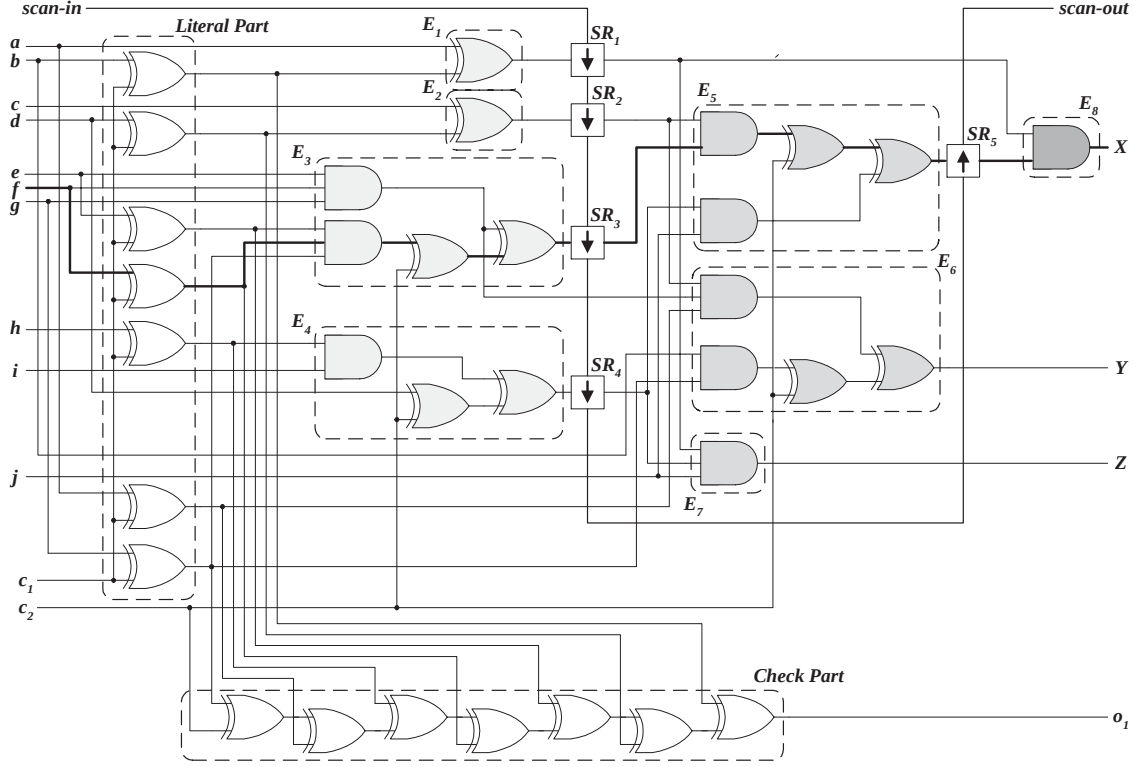


Figure 14. Highly testable multi-level AND/EXOR implementation.

The literal part implements the complements of the input variables as c_1 is set to '1'. During testing c_1 is set to '0' for some tests and '1' for others. The AND part and the linear part implements the realized ESOP function when $c_1 = 1$. The additional EXOR cascade in the check part, the AND gate "A", and the EXOR gate "B" are added for improved testability of the realization. Notice that the implementation requires a minimal number of controllability inputs and observability outputs (c_1, c_2, o_1, o_2), which is one of the main goals of scan-based design.

4.3. Multi-Level Testing with Scan Registers

Scan registers (SR_{1-5}) are inserted at the outputs of ESOP planes (E_{1-8}) –unless it is a primary output– to control the logic value applied to the next level, and to observe the output value of each ESOP plane via the scan chain. Figure 13 shows the implementation of a scan block.

The physical separation of levels by means of inserting scan blocks allows the ESOP planes of the same level to be tested together, in parallel. The test patterns are applied to the ESOP planes from the primary circuit inputs in parallel by an EDPG, and also applied from the scan blocks by shifting them serially into the scan chain. Sim-

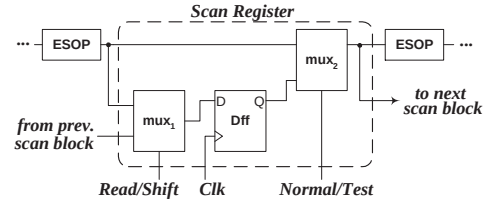


Figure 13. Scan block implementation.

ilarly, the results are collected at a multiple-input signature register, from the primary circuit outputs and from the scan blocks by shifting them out serially. The testing of the scan block are performed separately by shifting in another set of test patterns through the scan chain.

Example 14: Figure 14 shows the proposed highly testable multi-level AND/EXOR realization for the multi-output function given below.

$$\begin{aligned}
 [X] &= U[V(efg \oplus \overline{e}\overline{f}\overline{g}) \oplus jW] \\
 [Y] &= b\overline{g} \oplus \overline{a}efgV \\
 [Z] &= jUW \\
 U &= a \oplus \overline{b} \\
 V &= c \oplus \overline{d} \\
 W &= \overline{h}i \oplus d.
 \end{aligned}$$

ESOP planes are identified as follows. The level-0 ESOPs are:

$$\begin{aligned} E_1^0 &= c \oplus \bar{d} \\ E_2^0 &= efg \oplus \bar{e}\bar{f}\bar{g} \\ E_3^0 &= \bar{h}i \oplus d \\ E_4^0 &= a \oplus \bar{b}, \end{aligned}$$

the level-1 ESOPs are:

$$\begin{aligned} E_1^1 &= E_1^0 E_2^0 \oplus j E_3^0 \\ E_2^1 &= [Y] = b\bar{g} \oplus \bar{a}efg E_1^0 \\ E_3^1 &= [Z] = j E_3^0 E_4^0, \end{aligned}$$

and the level-2 ESOP is:

$$E_1^2 = [X] = E_4^0 E_1^1.$$

The major advantages of our scan-based multi-level AND/EXOR implementation over the traditional scan-based multi-level AND/OR design are:

1. The minimal number of test patterns, $(n + 6)$, can be generated internally and inexpensively [10].
2. The universal test patterns can be shared by the ESOP planes at each level. This allows simple shifting of the patterns through the scan chain with the identification of the ESOP planes.
3. To reduce the area overhead, primary circuit inputs and outputs are not included in the scan-chain. Also note that scan blocks are inserted only at the outputs of the ESOP planes, not at arbitrary nodes in the entire realization.

A delay analysis can be performed on the example circuit by referring to the below set of library components produced by LSI Logic Corporation [12].

Component	Delay(ns)
<i>AND2</i>	0.15 + $\sim 0.04/OutputLoad$
<i>AND-3</i>	0.21 + $\sim 0.04/OutputLoad$
<i>EXOR-2</i>	0.3 + $\sim 0.04/OutputLoad$
<i>mux 2 × 1</i>	0.26 + $\sim 0.03/OutputLoad$

Notice that during normal mode of operation the only extra delay added to the original multi-level AND/EXOR circuit is caused by **mux2** of the scan block, and by the single EXOR gate used in the literal part as an inverter. Today's technology makes it possible to implement a 2×1 multiplexer slower than an AND gate but faster than an EXOR gate. In the realization in Figure 14, a critical delay path is highlighted as bold lines. The delay of the path is calculated as 2.95 ns.

To show the delay improvement of our multi-level implementation over the 2-level cascaded-EXOR implementation given in [10], we first obtain the ESOP expression for function X , which has the critical delay path,

$$\begin{aligned} E &= acefg \oplus ac\bar{e}\bar{f}\bar{g} \oplus a\bar{d}efg \oplus a\bar{d}\bar{e}\bar{f}\bar{g} \oplus aj\bar{h}i \oplus \\ &\quad ajd \oplus \bar{b}cefg \oplus \bar{b}c\bar{e}\bar{f}\bar{g} \oplus \bar{b}\bar{d}efg \oplus \bar{b}\bar{d}\bar{e}\bar{f}\bar{g} \oplus \\ &\quad \bar{b}j\bar{h}i \oplus \bar{b}jd. \end{aligned}$$

The expression has 12 product terms. Therefore, the realization given in [10] requires 12 cascaded EXOR-2 gates in the linear part. The critical delay path, in this case, one AND-3 gate in the AND part and 12 EXOR-2 gates in the linear part; giving a total delay of 4.33 ns. This shows a significant improvement of our multi-level implementation over two-level implementations.

5. CONCLUSIONS AND FUTURE WORK

We defined the Even-Odd Rectangle Covering method that allows to obtain efficiently factorized multi-level AND/EXOR circuits. We showed that such circuits, when realized as partitioned ESOP planes with cascaded EXORs and scan registers inserted between the levels, can be upgraded to highly testable multi-level AND/EXOR realizations with universal test sets. As a result, a deterministic BIST circuit can be utilized to perform the entire testing of the circuit internally, as described in [10].

In terms of area, the effectiveness of the scan block insertion of our method depends on the size of the ESOP planes. Therefore, the factorization method can be optimized to obtain larger ESOP planes. This also involves the modification of the AND/EXOR minimizer that is used on the ESOP expression before the factorization, or that is used on each ESOP plane after the factorization.

As an alternative, not discussed in this paper, the EXOR gates in each ESOP plane can be realized as a tree [17], instead of a cascade [10]. This may yield faster circuits. However, the advantage of universal test set will be lost, leading to more complex BIST circuits, or to externally controlled scan-designs.

As our future work, a software tool will be developed to compare the performance of the factorization method presented in this paper with the methods given in [6, 19, 21, 23, 27]. Another direction of our future work will be to perform some measurements on several benchmark circuits to compare our easily testable multi-level AND/EXOR realization in terms of area, delay, power, and especially testability, to other test and implementation schemes. Such schemes involve multi-level arbitrary logic realizations with externally controlled or BIST-based scan test designs.

References

- [1] R. K. Brayton. Factoring Logic Functions. *IBM J. Res. Develop.*, 31(2), Mar. 1987.
- [2] R. K. Brayton and C. McMullen. The Decomposition and Factorization of Boolean Expressions. In *Proc. Int. Symp. Circuits and Systems*, May 1982.
- [3] R. K. Brayton, R. Rudell, A. Sangiovanni-Vincentelli, and A. Wang. MIS: A Multiple-Level Logic Optimization System. *IEEE Trans. Computer-Aided Design*, CAD-6(6), Nov. 1987.
- [4] S. C. Chang, M. Marek-Sadowska, and K. T. Cheng. Perturb and Simplify: Multilevel Boolean Network Optimizer. *IEEE Trans. Computer-Aided Design*, 15(12):1494–1504, Dec. 1996.
- [5] M. Chatterjee, D. Pradhan, and W. Kunz. LOT: Logic Optimization with Testability - New Transformations Using Recursive Learning. In *Proc. Int. Conf. Computer-Aided Design*, pages 318–325, 1995.
- [6] S. Chattopadhyay, S. Roy, and P. P. Chaudhuri. KGP-MIN: An Efficient Multilevel Multioutput AND-OR-XOR Minimizer. *IEEE Trans. Computer-Aided Design*, 16(3):257–265, Mar. 1997.
- [7] C. H. Chiang and S. K. Gupta. Random Pattern Testable Logic Synthesis. In *Proc. Int. Conf. Computer-Aided Design*, pages 125–128, 1994.
- [8] L. A. Entrena and K. T. Cheng. Combinational and Sequential Logic Optimization by Redundancy Addition and Removal. *IEEE Trans. Computer-Aided Design*, 14(7):909–916, Jul. 1995.
- [9] G. D. Hachtel and F. Somenzi. *Logic Synthesis and Verification Algorithms*. Kluwer Academic Publishers, Boston, 1996.
- [10] U. Kalay, M. A. Perkowski, and D. V. Hall. A Minimal Universal Test Set for Self Test of EXOR-Sum-of-Products Circuits. *submitted to IEEE Trans. Comp.*, 1998.
- [11] G. Lee, M. Hwang, M. Irwin, and R. Owens. Testability of a Class of Multi-Level Reed-Muller Circuits. In *Int. Workshop on Applications of the Reed-Muller Expansion in Circuit Design*, pages 134–141, 1993.
- [12] LSI Logic Corporation. *LCA/LEA500K Array-Based Products Databook*, fourth edition, May 1997.
- [13] D. K. Pradhan. Universal Test Sets for Multiple Fault Detection in AND-EXOR Arrays. *IEEE Trans. Comp.*, 27(2):181–187, Feb. 1978.
- [14] J. Rajske and J. Vesudevamurthy. The Testability-Preserving Concurrent Decomposition and Factorization of Boolean Expressions. *IEEE Trans. Computer-Aided Design*, 11(6):778–793, Jun. 1992.
- [15] S. M. Reddy. Easily Testable Realizations for Logic Functions. *IEEE Trans. Comp.*, C-21:1183–1188, Nov. 1972.
- [16] K. K. Saluja and S. M. Reddy. Fault Detecting Test Sets for Reed-Muller Canonic Networks. *IEEE Trans. Comp.*, C-24:995–998, Oct. 1975.
- [17] T. Sasao. Easily Testable Realizations for Generalized Reed-Muller Expressions. *IEEE Trans. Comp.*, 46(6):709–716, June 1997.
- [18] T. Sasao. *Switching Theory for Logic Synthesis*. Kluwer Academic Publishers, Boston, 1999.
- [19] J. M. Saul. An Algorithm for the Multi-Level Minimization of Reed-Muller Representations. In *Proc. IEEE Int. Conf. on Computer Design*, pages 634–637, October 1991.
- [20] J. M. Saul. Towards a Mixed Exclusive-/Inclusive-OR Factored Form. In *Int. Workshop on Applications of the Reed-Muller Expansion in Circuit Design*, pages 1–5, Sept. 1993.
- [21] J. M. Saul. Logic Synthesis Based on Exclusive-OR Sums. In *Proc. Reed-Muller Colloquium*. University of Bristol, Dec. 1995.
- [22] N. Song and M. Perkowski. Minimization of Exclusive Sum of Products Expressions for Multi-Output Multiple-Valued Input, Incompletely Specified Functions. *IEEE Trans. on Computer Aided Design*, 15(4):285–395, Apr. 1996.
- [23] B. Steinbach and M. Stöckert. Design of Fully Testable Circuits by Functional Decomposition and Implicit Test Pattern Generation. In *IEEE VLSI Test Symposium*, pages 22–27, Cherry Hill, NJ, 1994.
- [24] N. Tamarapalli and J. Rajske. Constructive Multi-Phase Test Point Insertion for Scan-Based BIST. In *Proc. Int. Test Conf.*, pages 649–658, 1996.
- [25] N. A. Touba and E. J. McCluskey. Automated Logic Synthesis of Random Pattern Testable Circuits. In *Proc. Int. Test Conf.*, pages 174–183, 1994.
- [26] N. A. Touba and E. J. McCluskey. Test Point Insertion Based on Path Tracing. In *Proc. VLSI Test Symp.*, pages 2–8, 1996.
- [27] C. Tsai and M. Marek-Sadowska. Multilevel Logic Synthesis for Arithmetic Functions. In *Proc. Design Automation Conf.*, pages 242–247, 1996.