Odd/Even Cube Covering for Minimizing ESOP Circuits*

M. A. Thornton
Dept of Electrical and Computer Engr.
Mississippi State University
Mississippi State, MS 39762 USA
mitch@ece.msstate.edu

B. Q. Vu
Dept of Computer Science and Engr.
University of Arkansas
Fayetteville, AR 72701 USA
bqv@engr.uark.edu

R. Drechsler
Institute of Computer Science
University of Freiburg
Freiburg, Germany
drechsle@informatik.uni-freiburg.de

Abstract- **A method for minimizing *Exclusive-OR Sum of Product* (ESOP) forms of Boolean functions is described. ESOP forms have been proven to be require fewer products than the more common *Sum of Products* (SOP) forms based on the inclusive-OR operator for many classes of functions. Since finding the optimal ESOP solution requires the solution of the well-known *set covering problem*, a heuristic method is developed. A prototype version of this technique has been implemented and the results are compared to a more mature ESOP minimization algorithm.**

1 Introduction

A fundamental problem in the area of switching theory is that of two-level minimization of Boolean expressions. In general, the two-level minimization problem consists of finding an equivalent expression with exactly two algebraic operations allowed (in addition to the use of the unary inversion operator) such that the resulting expression satisfies some constraint. Two well-known examples are finding the Sum of Products (SOP) and Product of Sums (POS) forms subject to the constraint that as few logic OR operations as possible are present in the SOP form, or, as few logic AND operations as possible occur in the POS form. This subset of two-level minimization problems is loosely referred to as "minimizing a Boolean function" by many logic designers.

Although many well known techniques have been developed for minimizing SOP and POS forms [6] [7], this problem is provably hard. It can be shown that finding the exact solution to the two-level minimization problem involves finding the solution to the subproblems commonly known as the *set covering problem* (for unate functions) and the *minimum cost assignment problem* (for binate functions). These subproblems have been shown to be members of the class of NP-complete problems. Thus, there are no known deterministic algorithms available that run in less than an exponential amount of time.

Despite the difficulty in finding the exact minimal forms of the SOP and POS representations, there are very good heuristic methods available that often find the minimal, or at least, near-minimal solutions [1]. The existence of these techniques has enabled the development and widespread use of logic synthesis tools that are prevalent in modern design environments [3].

It has been shown that the Exclusive-OR Sum of Products (ESOP) form can often represent functions with fewer product terms than the more well-known SOP form [11] [8]. The ESOP form expresses a function as a collection of product terms (or cubes) that are combined with the XOR operator. Minimizing the ESOP form also falls into the general class of set covering problems, however, there are not good generalized heuristic minimization techniques available as is the case for the SOP and POS forms. Many researchers are currently investigating techniques for minimizing ESOP forms. Other techniques have been proposed recently, but they are still considered immature [10] [5]. Since the discovery of such a method would have important consequences with regard to the practical problems of logic synthesis, logic verification, state minimization and other digital logic problems, we are motivated to develop alternative methods for solving the problem.

A heuristic ESOP minimization technique is described here. The method is derived by modifying the tabulation method that is used for SOP minimization [7]. The implementation of this technique is discussed using a cube list representation and using Binary Decision Diagrams (BDD) [2].

The organization of the paper describes the theory behind this approach in detail and compares it

*This project was supported by NSF grants CCR-9633085, SBE-9815371 and DAAD grant 315/PPP/gü-ab.

with another popular approach for minimizing ESOP forms. Next, experimental results are presented that compare this approach with another well-known method. These results indicate that our technique generates results that are similar to those produced by the other method [10]. Finally, we present conclusions and discuss possible future extensions to this approach.

2 Description of the Method

An ESOP minimization technique based on odd/even cube covering is described here. This technique is based on two fundamental relations as given in Equations 1 and 2.

$$\bigoplus_{i=1}^{2m} c = 0 \tag{1}$$

$$\bigoplus_{i=1}^{2m+1} c = c \tag{2}$$

Where c represents an instance of a specific product term (cube or logical AND) of a set of literals, $\oplus$ represents the exclusive-OR operation and m is some non-negative integer. Equation 1 essentially states that including a cube in a cover set for an even number of times is equivalent to not including it at all. Likewise, including a cube in a cover set an odd number of times is equivalent to including it with an instance of one. This gives a degree of freedom in finding a minimal ESOP cover since the existence of an instance of a cube in the cover set does not necessarily mean it is in the overall on-set of the function being minimized.

As an example, consider the function, $F(w,x,y,z) = \sum(0,1,4,10,11,13,14)$. Figure 1 shows two Karnaugh maps. The map on the left contains circles that result in the minimized SOP form, $F = \overline{w}\,\overline{x}\,\overline{y} + \overline{w}\,\overline{y}\,\overline{z} + \overline{w}\,x\,y\,z + w\,x\,\overline{y}\,z + w\,y\,\overline{z} + w\,\overline{x}\,y$. The SOP form requires 6 AND gates and the overall expression is comprised of 9 literals. The Karnaugh map on the right contains circles that obey Equations 1 and 2 since all 0 values are circled an even number of times and all 1 values are circled for an odd number. Thus, the 0 values cancel out due to the XOR relation in Equation 1 and the logic-1 values remain. This results in the expression, $F = \overline{w}\,\overline{y} \oplus x\,z \oplus w\,y$ which only requires 3 AND gates and is comprised of 6 literals.

While the Karnaugh map example is useful for illustrating the theory behind our approach, it is impractical for even moderately sized Boolean functions which are commonly encountered in typical designs. For this reason, we developed an experimental computer program that operates over a structure similar to the cover table used in the tabular (or Quine-McCluskey) approach [7] for SOP minimization. This technique is described using two different types of function representations, *cube lists* and *Binary Decision Diagrams* (BDDs).

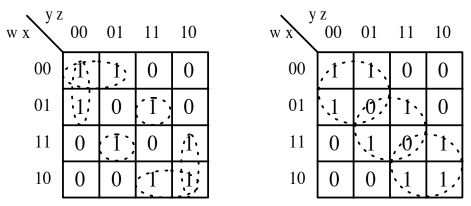

Figure 1: Karnaugh Map Illustrating SOP Versus ESOP Forms

2.1 Cube List Approach

A cube list is a symbolic representation of a collection of product terms that provide a cover for a function, in our case an ESOP cover. The cube list approach is very similar to the well-known tabulation method where a set of minterms describing the function to be minimized is used as the initial input. It is summarized in the following three steps:

1. Pairs of minterms are examined to determine if they differ by only a single literal. If they do, a new cube is added to the list that has a "don't care" in place of the literal. This process is then applied to the new set of cubes and is continued until no new product terms may be added.

2. The list of cubes resulting from step 1 is reduced in size by eliminating redundant product terms. This is the step where a solution to the cover problem is found through the use of heuristics.

3. The reduced cube list is further optimized by searching for pairs that satisfy particular algebraic properties. When such a pair is found, they may be replaced by another pair containing more "don't care" values, or, they may be "merged" into a single term.

With the exception of step 3, the method outlined above could be interpreted as a high-level description of the tabulation method for SOP circuits. However, due to the different properties of the inclusive-OR versus the exclusive-OR operation, each step must be implemented in a slightly different manner.

In step 1, after new cubes are added to the list, the pair of terms that produced each new cube cannot automatically be deleted as is the case with the SOP tabulation method. Rather, a count of the total number of cubes in the new list must be made that cover the original set. Only if the count is an odd number, $\{1, 3, 5, \ldots\}$, may the original cube be deleted. This is due to the relationships in Equations 1 and 2. Furthermore, when a new cube is created, it is added to the list only if it does not exist there already. This is important since the occurrence of two identical cubes would cause a cancellation since $x \oplus x = 0$. For the SOP formulation, this is not a problem since $x + x = x$.

The second phase of the process involves the creation of a cover table followed by solving the cover problem. This is also done in a similar manner to the SOP tabulation method where "essential products" are chosen for the final list. However, as was the case in step 1, care must be taken to ensure that all minterms are covered an odd number of times, thus the odd/even counting process is also integrated into this procedure.

In the event that the essential cubes do not cover all minterms, a situation occurs that is referred to as a "cyclic cover". This is a case where there is no clear choice in which product term to add to the final list because none of the candidates are essential and it is an instance of the cover problem. The heuristic we employ is to chose the product term that covers the **fewest** minterms. This is just the opposite of a common heuristic used in the related SOP tabulation method where implicants are chosen that cover the most minterms. However, our heuristic was based on the fact that each time an implicant is added to the final list, the odd/even counting process must be invoked. If a product was added that covered many minterms that were covered previously, an even count would occur for these resulting in adding yet another product term for each to make the count become odd. By adding implicants that cover the fewest minterms, we minimize the number of additional products that must be added to maintain an odd minterm cover count.

After step 2, an ESOP is represented by the resulting cube list that is guaranteed to cover each minterm of f an odd number of times and each minterm of $\overline{f}$ exactly zero times. However, as was shown in the Karnaugh map example, it is permissible to cover minterms of $\overline{f}$ an even number of times, since these function zeros will cancel due to the XOR operation. Furthermore, it is often desirable to include this in the resultant ESOP since it can help to produce a smaller list of larger cubes. For these reasons, the third step is included in our technique.

Step 3 makes use of two algebraic properties, $\overline{x}y \oplus x\overline{y} = x \oplus y$ and $x \oplus \overline{x} = 1$. Each pair of cubes in the list produced in step 2 is examined for common factors. If a non-null common factor is found, the corresponding non-common factors of the two cubes are checked to determine if they satisfy the algebraic properties. The first property causes each cube to expand its' cover set resulting in the replacement of the current two cubes with two larger ones. The second property allows the two cubes to be merged into a single term.

2.2 BDD Based Approach

As is well-known, each Boolean function $f : \mathbf{B}^n \rightarrow \mathbf{B}$ can be represented by a *Binary Decision Diagram* (BDD) [2], i.e. a directed acyclic graph where a Shannon decomposition is carried out in each node. For functions represented by reduced, ordered BDDs, efficient manipulations are possible. BDDs are the state-of-the-art data structure in VLSI CAD and have been used in many applications [4].

BDDs representing Boolean functions can be directly interpreted as a two-level form, if the 1-paths, i.e. the paths from the root of the graph to the terminal 1 node, are interpreted as cubes. The resultant cube lists may be considered to be either SOP or ESOP since the decomposition carried out in each node of the BDD is disjoint. Unfortunately, for some functions, the number of 1-paths might be very large, even though the ESOP is small (see e.g. function $f = x_1 \oplus \ldots \oplus x_n$). This occurs because the disjoint cube list does not cover a single minterm more than once. This restricts the size of the resulting cubes. Furthermore, in the case of ESOP, the disjoint cube list does not contain any even-numbered coverings of minterms for $\overline{f}$. Although these restrictions exist, the disjoint cube list obtained from the BDD will be a better initial starting point since the cube based method discussed above relies on an initial list of minterms (i.e. $0 - cubes$) which are not only disjoint, but the smallest possible cube as well.

The transformations discussed for cube lists above can be carried out on BDDs based on synthesis operations. Since the variable ordering of BDDs plays an important role not only for the size, but also for the number of 1-paths, the reordering algorithms have to be assimilated. An alternative proposed in [12] is to give up the strict BDD structure and also allow e.g. more than two outgoing edges per node.

A different approach is to formulate the covering problem as a SAT instance and build the BDD for the resulting problem [9]. This technique is mainly limited by the size of the graph and can only be applied to functions with up to 5 variables.

3 Implementation

An experimental version of this technique has been implemented based on the use of cube lists. The initial input is a list of covering minterms in the *PLA* format used by *ESPRESSO*. Presently, the implementation only supports single-output functions, although it can be easily extended to handle multiple-output functions by utilizing multiple-valued logic concepts.

During the occurrence of a cyclic cover, the very simple heuristic of choosing the product term that covers the *least* number of minterms was used. Also, the process of examining pairs of cubes needed in steps 1 and 3 of the technique were implemented by maintaining the cubes lists in an order of increasing Hamming weight and forming pairs by beginning with the cubes of lower weight first. Altering the order in which the cube pairs are examined can affect the outcome of the ESOP result.

4 Experimental Results

This program was run using standard benchmark circuits and the results were compared to the ESOP minimizer *EXMIN2* [10]. Table 1 contains results when a 5-bit arithmetic squaring circuit was minimized using our approach versus *EXMIN2*. The leftmost column indicates the function output as given in the order present in the *PLA* input file. The other two columns contain the number of product terms in the ESOP form for each output resulting from each technique. While the overall number of products is the same, it should be noted that *EXMIN2* optimized for the multi-output circuit in one pass while our experimental version optimized each output independently. Of the 34 product terms produced by *EXMIN2*, 14 are shared, thus only 21 distinct product terms were produced. In the case of our technique, only 4 product terms were shared, and this occurred by coincidence, thus we produced 30 distinct product terms.

5 Conclusion

A technique for determining an ESOP form for a Boolean function has been developed based on the primitive operation of odd/even cube covering. This technique has been implemented and the results have been compared to other ESOP heuristic minimizers. Methods for implementing this technique using both cube lists and BDDs have been described. Experimental results indicate that this approach yields roughly equal results to more mature ESOP minimization techniques. This result is encouraging since our current implementation is based on very simple heuristics.

In the future, we plan to incorporate multi-output function capabilities for this method and to implement it based on BDDs rather than cube lists. We also plan to investigate the use of other heuristics for solution of the cover table problem and the order in which pairs of product terms are selected.

Table 1: Results of Finding ESOP Forms for a 5-bit Squaring Circuit

Output	Odd/Even Cube Covering	EXMIN2
0	3	2
1	4	3
2	5	6
3	6	5
4	6	5
5	4	7
6	2	2
7	1	2
8	1	1
9	1	1

References

[1] R.K. Brayton, G.D. Hachtel, C. McMullen, and A.L. Sangiovanni-Vincentelli. *Logic Minimization Algorithms for VLSI Synthesis.* Kluwer Academic Publishers, 1984.

[2] R.E. Bryant. Graph - based algorithms for Boolean function manipulation. *IEEE Trans. on Comp.*, 35(8):677–691, 1986.

[3] O. Coudert. Two-level logic minimization: an overview. *INTEGRATION, the VLSI Jour.*, 17(2):97–140, 1994.

[4] R. Drechsler and B. Becker. *Binary Decision Diagrams - Theory and Implementation.* Kluwer Academic Publishers, 1998.

[5] M. Helliwell and M. Perkowski. A fast algorithm to minimize multi-output mixed-polarity generalized reed-muller forms. In *Design Automation Conf.*, pages 427–432, 1988.

[6] M. Karnaugh. The map method for synthesis of combinational logic circuits. *Transactions of A.I.E.E.*, 72(pt. 1):593–599, Nov. 1953.

[7] E.J. McCluskey. Minimization of Boolean functions. *Bell System Technical Jour.*, 35, 1956.

[8] U. Rollwage. The complexity of mod-2 sum PLA's for symmetric functions. *IFIP WG 10.5 Workshop on Applications of the Reed-Muller Expansion in Circuit Design*, pages 6–12, 1993.

[9] T. Sasao. An exact minimization of AND-EXOR expressions using BDDs. *IFIP WG 10.5 Workshop on Applications of the Reed-Muller Expansion in Circuit Design*, pages 91–98, 1993.

[10] T. Sasao. EXMIN2: A simplification algorithm for Exclusive-OR-Sum-of products expressions for multiple-valued-input two-valued-output functions. *IEEE Trans. on CAD*, 12:621–632, 1993.

[11] T. Sasao and Ph. Besslich. On the complexity of mod-2 sum PLAs. *IEEE Trans. on Comp.*, 39:262–266, 1990.

[12] Y. Ye and K. Roy. Graph-based synthesis algorithms for AND/XOR networks. In *Design Automation Conf.*, pages 107–112, 1997.