Exact ESCT minimization for functions of up to six input variables - PRELIMINARY VERSION

Dimitrios Voudouris Marinos Sampson
George Papakonstantinou
Dept of Electrical and Computer Engineering, Division of Computer Science
Computing Systems Laboratory, National Technical University of Athens
157 80, Zografou Campus, Greece

Abstract

In this paper an efficient algorithm for the synthesis and exact minimization of ESCT(Exclusive or Sum of Complex Terms) expressions for Boolean functions of at most six variables is proposed. This kind of logical expressions can be mapped to a special cellular architecture, called Reversible Wave Cascade Architecture. This topology is useful, because it has been proved to be reversible and moreover it may help in the design of quantum circuits. The proposed algorithm is the first one to give solution to the problem of finding minimal ESCT expressions for switching functions of up to six input variables.

1 Introduction

For many years, logic synthesis was based on AND, OR and NOT gates. However, for arithmetic, error correcting and telecommunication applications, the use of XOR (eXclusive OR) gates can reduce the complexity of logic circuits. The most general (and most powerful) AND-XOR expression is the "Exclusive or Sum Of Products" (ESOP) expression where a function is represented as the XOR sum of logical products (Logical ANDs of variable literals) [1]. The natural evolution to these expressions are the "Exclusive or Sum of Complex Terms" (ESCT) expressions where every term may additionally use the logical OR and XOR operations.

An ESCT expression of a Boolean function is a XOR sum of terms. These terms are similar to logical products, but the Boolean function between the variable literals is not restricted only to logical AND but also includes logical OR and XOR. In other words, an ESCT expression of m terms, for a single-output Boolean function of n input variables, has the following form: $Q = \sum_{i=1}^{i \leq m} \oplus G_n^i(x_n, G_{n-1}^i(x_{n-1}, \ldots G_1^i(x_1, 0)))$, where G_j^i is an arbitrary single-output Boolean function of 2 input variables. Each $G_n^i(x_n, G_{n-1}^i(x_{n-1}, \ldots, G_1^i(x_1, 0)))$ term is called a complex term. A more formal definition is given in section 2.

This kind of expressions have been introduced by K. K. Maitra [2] in 1962, while studying cellular architectures. He proposed a special kind of those, which we now call the Reversible Wave Cascade (Fig. 1). An ESCT expression is an

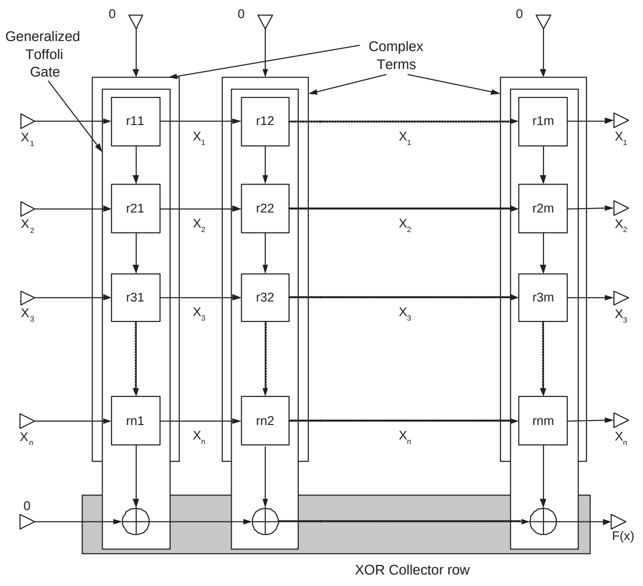


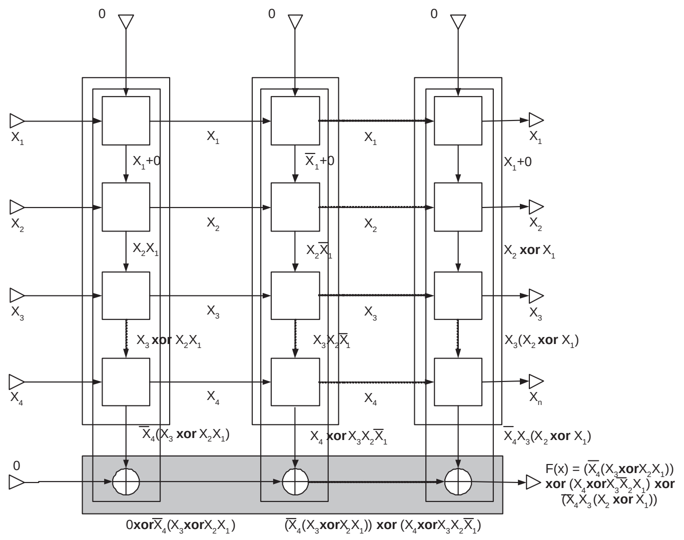
Figure 1: Reversible wave cascade architecture

expression of a Boolean function suited for direct mapping to such an architecture.

***Example* 1** *Let $f(x_1, x_2, x_3, x_4)$ be a single-output 4-variable Boolean function. An ESCT expression of f is:*$(x_1x_2 \oplus x_3)\bar{x_4}$ $\oplus(\bar{x_1}x_2x_3 \oplus x_4)$ $\oplus((x_1 \oplus x_2)x_3\bar{x_4})$*. This ESCT expression can be directly mapped to the Reversible Wave Cascade of Fig 2.*

It has been proved [3] that an ESCT expression can be directly mapped to reversible logic gates and more specifically to Generalized Toffoli gates (refer to Definition 6). A logic gate is called reversible, if it has the same number of inputs and outputs, and maps each input vector into a unique output vector and vice versa. Moreover both fan-in and fan-out are forbidden. One of their important properties is that they consume minimal amounts of power, due to the fact that they loose no information[4]. However, it should be noted that this fact is technology dependent. For current CMOS technology the power lost, because of information loss, is minimal, therefore reversible logic does not bring any real advantage. In Ref [5] it was shown that all quantum logic gates must be reversible. Due to this, the reversible wave cascades is a very attractive architecture for the implementation of reversible logic circuits and perhaps quantum logic. In Fig. 1, a Reversible Wave Cascade is shown, representing an ESCT expression. It can be observed that each column of this architecture (a complex term), along with an additional XOR function can be considered as a Generalized Toffoli gate (refer to Definition 6).

The reversible wave cascade is a cellular architecture suitable for mapping ESCT expressions. Currently, there are no such commercial architectures available. Although an ESCT expression is suited for mapping into such a cellular architecture, it can be mapped to some modern LUT (Look Up Table) FPGAs



0
0
0
X_1
X_1
X_1
X_1
X_1+0
$\overline{X}_1+0$
X_1+0
X_2
X_2
X_2
X_2
X_2X_1
$X_2\overline{X}_1$
X_2 **xor** X_1
X_3
X_3
X_3
X_3
X_3 **xor** X_2X_1
$X_3X_2\overline{X}_1$
$X_3(X_2$ **xor** $X_1)$
X_4
X_4
X_4
X_n
$\overline{X}_4(X_3$ **xor** $X_2X_1)$
X_4 **xor** $X_3X_2\overline{X}_1$
$\overline{X}_4X_3(X_2$ **xor** $X_1)$
0
F(x) = $(\overline{X}_4(X_3$**xor**$X_2X_1))$ **xor** $(X_4$**xor**$X_3\overline{X}_2X_1)$ **xor** $(\overline{X}_4X_3(X_2$ **xor** $X_1))$
0**xor**$\overline{X}_4(X_3$**xor**$X_2X_1)$
$(\overline{X}_4(X_3$**xor**$X_2X_1))$ **xor** $(X_4$**xor**$X_3X_2\overline{X}_1)$

Figure 2: Reversible wave cascade for function of Example 1.

(Field Programmable Gate Arrays). In Ref [6] a superset of ESCT expressions have been mapped to the ATMEL 6000 series FPGA. Nevertheless, such a mapping may degenerate certain features of these expressions, such as reversibility.

There were algorithms developed in the past, for mapping switching functions to Cellular Array (CA) architectures and then minimizing the number of the produced complex terms. In references [6], [7], [8], [9], [10], techniques like variable reordering and cube transformation or even multi-valued logic are used, in an effort to minimize the number of complex terms, though their architectures differ from the one presented in this paper and are more complex. In references [11], [12] a systematic method was presented to produce architectures very similar to the Reversible Wave Cascade architecture, by extending the EXORLINK [13] operation to complex terms. In Ref [3] it was proved that a Reversible Wave Cascade is considered to be a reversible gate (more specifically a Generalized Toffoli gate - Definition 6). Moreover, an algorithm is proposed for mapping arbitrary Boolean functions to Reversible Wave Cascades but it was not implemented. A similar reversible architecture is described in [14], with major objective to minimize the number of garbage inputs, although at that time no algorithm for the mapping and minimization of ESCT expressions to such architectures had been proposed. In Ref [15] two algorithms were presented that produced ESCT expressions for a single-output Boolean function while minimizing the number of terms in the produced expression. The first one guarantees minimality for functions up to 5 variables. The second algorithm applied the first algorithm on groups of cascades, inside the cellular array, as a term transformation operation. This procedure was repeated several times, over different groups of cascades, in order to heuristically minimize their number. These algorithms have been improved in [16] with the introduction of relative terms and in [17] they have been extended for multi-output Boolean functions.

In this paper we introduce an algorithm that can produce minimal expressions (the ones with the least number of complex terms) for a single-output switching function f with at most 6 variables in a practical computing time. To the best of the authors knowledge, this is the first algorithm in the related literature, that can find minimal ESCT expressions for switching functions of 6 variables.

2 Theoretical background

In this section we provide some background definitions. An expression of a switching function suitable for mapping to a Maitra cascade (cell chain) is called a Maitra term[2]. A more formal definition[3] follows:

Definition 1 *A complex Maitra term (complex term or Maitra term for simplicity) is recursively defined as follows:*

1. *Constant* 0 *or* 1 *Boolean function is a Maitra term.*
2. *A literal is a Maitra term.*
3. *If M_i is a Maitra term, x^* is a literal, and G is an arbitrary two-variable Boolean function (Maitra cell), then $M_{i+1} = G(x^*, M_i)$ is a Maitra term.*

Additionally, it is required that each variable appears in each Maitra term only once. In other words a complex term is: $P = G_n(x_n^*, G_{n-1}(x_{n-1}^*, \ldots G_1(x_1^*, 0)))$, where x_i^* are literals and x_i are the variables P depends on. It is noted that in this previous definition x_i^* does not need to be in its negative form, since for every single output two variable Boolean function $G_n(x, y)$, there exists another single output two variable Boolean function $G_n^{'}$ such that: $G_n(x, y) = G_n^{'}(\bar{x}, y)$.

Definition 2 *An ESCT (Exclusive-or Sum of Complex Terms) expression (sometimes also called Maitra expression) for a switching function, is a XOR sum of complex terms:*

$$f = \sum_{i=1}^{m} \oplus M_i,$$

where M_i are complex terms and m is their number inside the expression. The same variable ordering is used for every M_i.

***Example* 2** *Let $f(x_1, x_2, x_3) = x_1x_2x_3$, where x_3 is the most significant variable and x_1 is the least significant one. The expression $x_1x_2x_3$ is an ESCT expression for f. Another ESCT expression for f is: $(\bar{x_1} + \bar{x_2}) \oplus (\bar{x_1} + \bar{x_2} + x_3)$, where $(\bar{x_1} + \bar{x_2}), (\bar{x_1} + \bar{x_2} + x_3)$ are complex terms.*

It can be observed that a complex term defines a specific variable ordering for its variables. For P, x_n is the most significant variable and x_1 is the least significant ($x_n, x_{n-1}, \ldots, x_2, x_1$ is the order of variable significance). If the variable ordering changes then it is possible that the new expression produced may not be a complex term. This will be clarified with the next example:

***Example* 3** *Let $f(x_1, x_2, x_3, x_4) = x_4x_3(x_2 \oplus x_1)$. If we define the following variable ordering: x_4, x_3, x_2, x_1, where every variable is more significant than its right adjacent, then f is the complex term $P = G_4(x_4, G_3(x_3, G_2(x_2, G_1(x_1, 0))))$,*

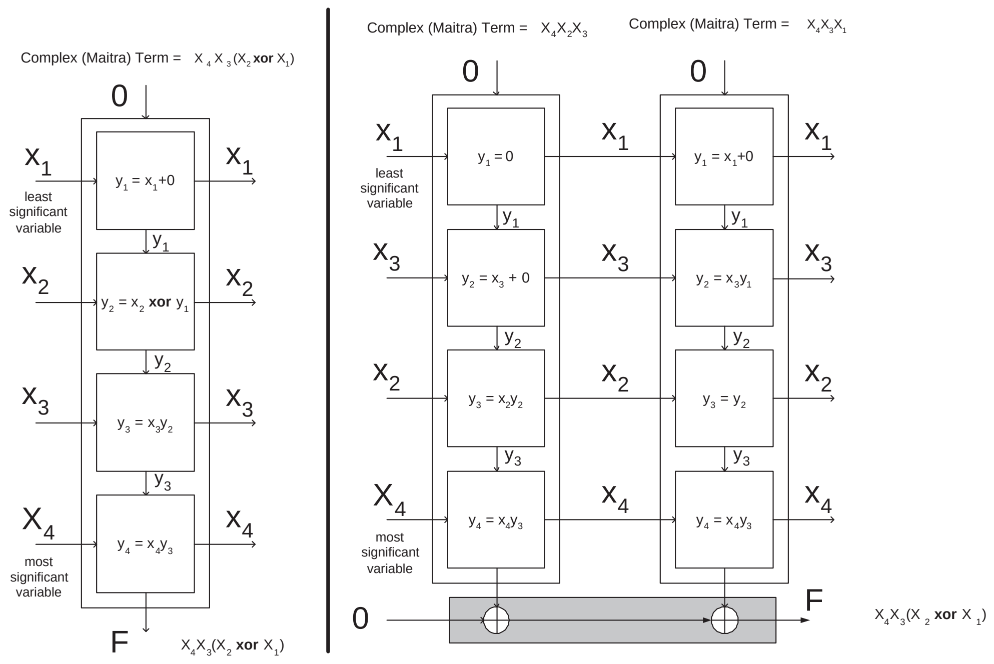


Figure 3: Reversible Wave Cascades for different variable orderings of function: $f(x_1, x_2, x_3, x_4) = x_4x_3(x_2 \oplus x_1)$.

where $G_4 = x_4G_3, G_3 = x_3G_2, G_2 = x_2 \oplus G_1, G_1 = x_1 + 0$. If we define a different variable ordering such as: x_4, x_2, x_3, x_1, where every variable is more significant than its right adjacent, then function $f = x_4x_2x_3 \oplus x_4x_3x_1$, using this variable ordering, can not be expressed as one complex term $P = G_4(x_4, G_2(x_2, G_3(x_3, G_1(x_1, 0))))$, since we cannot define any suitable single output two variable Boolean function G_2, with inputs x_2, G_3. Nevertheless, using this variable ordering, we can express the f function as the XOR sum of two complex terms: $P_1 \oplus P_2 = x_4x_2x_3 \oplus x_4x_3x_1$. The reversible wave cascades that correspond to the different variable orderings of Boolean function f are presented in Fig. 3. The leftmost Reversible Wave Cascade corresponds to the first variable ordering, while the rightmost one to the second variable ordering.

Definition 3 *A minimal (or exact) expression of a switching function $f(x_1, \ldots, x_n)$ of n variables is defined as the ESCT expression which has the least number of terms comparing to every other ESCT expression for this function.*

***Example* 4** *The ESCT expression $(\bar{x_1}+\bar{x_2}) \oplus (\bar{x_1}+\bar{x_2}+x_3)$ of Boolean function f, which was presented in example 2, is not minimal. Another ESCT expression of f is $x_1x_2x_3$ which is minimal, since it contains only one complex term $(x_1x_2x_3)$ and there can be no other ESCT expression for this function with fewer complex terms (an ESCT expression having zero complex terms represents only a constant Boolean function and f is, obviously, not constant).*

Definition 4 *The ESCT weight $w(f)$ (or simply weight) of a switching function $f(x_1, \ldots, x_n)$ of n variables, is defined as the number of complex terms in a minimal ESCT expression of f.*

***Example* 5** *In the previous example, the weight of the function f is equal to 1 (e.g. the number of complex terms in the minimal expression $x_1x_2x_3$ of f).*

Definition 5 *A switching function is called cascade realizable, if it has weight 1.*

It can easily be proved that every two-variable switching function is cascade realizable, since each Maitra cell can realize any two-variable function.

An ESCT expression can be directly mapped to a special cellular architecture, called Reversible Wave cascade (Fig. 1). A complex (Maitra) term is mapped to a column in the reversible wave cascade, excluding the last XOR cell. Each column is composed of cells ($r_{ij}, 1 \leq i \leq n, 1 \leq j \leq m$). The horizontal input to each cell is a variable and is propagated to the horizontal output of the cell. The vertical input is the output of the previous cell in the same column (or the constant 0 in the case of the first cell of each column). The outputs of each column are connected to the XOR collector, thus obtaining the function $F(x)$. Every such cell implements a single output two variable switching function (this function corresponds to the Maitra cell of Definition 1).

An ESCT expression is composed of reversible Generalized Toffoli gates [3].

Definition 6 *A $k * k$ generalized Toffoli gate is defined [3] as: $P_1 = A_1, P_2 = A_2, \ldots, P_{n-1} = A_{n-1}, P_n = f_{n-1}(A_1, A_2, \ldots, A_{n-1}) \oplus A_n$, where A_i are the inputs of the gate, P_i are the outputs of the gate and f_{n-1} is an arbitrary switching function of $n-1$ variables.*

In Ref [3] it was proved that a $k * k$ generalized Toffoli gate is reversible. Obviously, a Maitra cascade, composed of n cells, plus the corresponding XOR collector cell, is a $(n+1) * (n+1)$ Toffoli gate (Fig. 1) where: $P_1 = A_1 = x_1, \ldots, P_n = A_n = x_n, A_{n+1} = 0$ (or the output of another Toffoli gate) , $P_{n+1} = f_n(A_1, A_2, \ldots, A_n) \oplus A_{n+1}$ and $f_n(A_1, A_2, \ldots, A_n) = G_n(G_{n-1}(\ldots G_1(0, x_1), x_{n-1}), x_n)$ ($G_i, i \leq n$ are Boolean functions defined in Definition 1). It follows that a Reversible Wave Cascade is a reversible logic circuit because it is composed of reversible gates.

Is is obvious that the mapping of an ESCT expression to a reversible circuit is a trivial procedure. Algorithms that create ESCT expressions for switching functions are very attractive, since, in essence, they produce reversible circuits. Moreover, algorithms that create minimal ESCT expressions for switching functions are even more useful, because they produce smaller reversible circuits, resulting in, possibly, smaller production costs. Even more, as the technology in chip manufacturing advances, the energy consumption of such circuits will be diminished, since they loose no power due to information loss, and the heat dissipation caused by technological limits will gradually be reduced. This will, possibly, make architectures, like the reversible wave cascade, even more attractive. Unfortunately, in today's VLSI technologies (like CMOS) the power loss, even in reversible circuits, is still great due to heat dissipation.

It has been proved[18] that a Maitra cell doesn't need to implement every two-variable switching function. A set of only six functions is sufficient (complete set). Of course, there are many equivalent such sets[19]. We have adopted

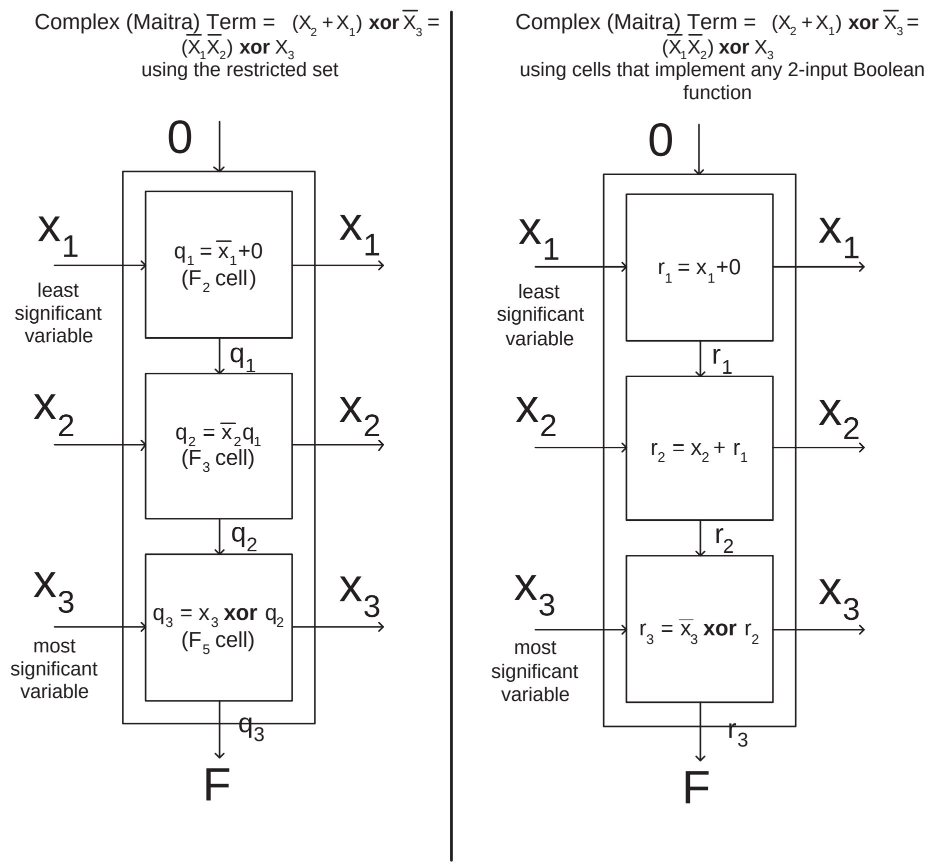

Complex (Maitra) Term = $(X_2 + X_1)$ xor $\overline{X_3}$ = $(\overline{X_1}\overline{X_2})$ xor X_3
using the restricted set
0
X_1
least significant variable
$q_1 = \overline{x_1}$+0
(F_2 cell)
X_1
q_1
X_2
$q_2 = \overline{x_2}q_1$
(F_3 cell)
X_2
q_2
X_3
most significant variable
$q_3 = x_3$ xor q_2
(F_5 cell)
X_3
q_3
F
Complex (Maitra) Term = $(X_2 + X_1)$ xor $\overline{X_3}$ = $(\overline{X_1}\overline{X_2})$ xor X_3
using cells that implement any 2-input Boolean function
0
X_1
least significant variable
$r_1 = x_1$+0
X_1
r_1
X_2
$r_2 = x_2 + r_1$
X_2
r_2
X_3
most significant variable
$r_3 = \overline{x_3}$ xor r_2
X_3
r_3
F

Figure 4: Reversible Wave Cascades for example 6.

(in the rest of the paper) one of them which can be seen in Table 1. The cascades that use cells which implement any switching function, from the complete set, are called Restricted Maitra Cascades and lead to smaller representations, since only three bits per Maitra cell are required for the representation of each cell instead of four. From this point on, without loss of generality, when we mention Maitra cascades, we will refer to restricted Maitra cascades.

Example **6** *Let $f(x_1, x_2, x_3) = (x_1 + x_2) \oplus \overline{x_3}$ (x_3 is the most significant variable and x_1 is the least significant one). This function can be expressed as one cascade, shown in the rightmost part of Fig. 4, where each cell can realize every possible single-output two variable Boolean function. In this case, we choose functions: $r_1 = 0 + x_1, r_2 = r_1 + x_2, r_3 = r_2 \oplus \bar{x_3}$. This function, however, can be realized using cells from the set defined in Table 1. In this case: $f(x_1, x_2, x_3) = (x_1 + x_2) \oplus \bar{x_3} = \overline{(x_1 + x_2)} \oplus x_3 = \bar{x_1}\bar{x_2} \oplus x_3$ and the cells used are: $q_1 = 0 + \bar{x_1}$(cell of index 2), $q_2 = q_1\bar{x_2}$(cell of index 3), $q_3 = q_2 \oplus x_3$(cell of index 5). This restricted cascade is shown in the leftmost part of Fig. 4.*

Table 1: Cell index set

Cell index(r)	$F_r(x, y)$
1	$x + y$
2	$\overline{x} + y$
3	$\overline{x}y$
4	xy
5	$x \oplus y$
6	y

It is assumed that the first cell in every cascade has one of its inputs connected to 0 (for symmetry reasons). Hence, the first cell of a chain needs to be of index 1, 2 or 6 only[15].

2.1 Representation

Next, we will present the definitions of the representation we use, for the rest of this paper, for an arbitrary Boolean function and an arbitrary ESCT expression.

Definition 7 *The minterm representation (MT) of a switching function f with n variables, is a bitvector of size 2^n where the i-th bit is 1 if the i-th minterm of f is 1.*

It can easily be observed that the MT representation of a Boolean function depends on its assumed variable ordering. This will be clarified with the next example.

***Example* 7** *Let $f(x_1, x_2, x_3) = (x_1 x_2) + x_3$. The Truth Table of f is shown in Table 2:*

Table 2: Truth table for function of example 7.

1st var ordering				2nd var ordering			
x_3	x_2	x_1	$f(x_1, x_2, x_3)$	x_1	x_2	x_3	$f(x_1, x_2, x_3)$
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	1
0	1	0	0	0	1	0	0
0	1	1	1	0	1	1	1
1	0	0	1	1	0	0	0
1	0	1	1	1	0	1	1
1	1	0	1	1	1	0	1
1	1	1	1	1	1	1	1

If we use the following variable ordering: x_3, x_2, x_1 (x_3 is the most significant variable and x_1 is the least significant), then the MT representation of f is 11111000 *(the leftmost digit represents the minterm $x_3 x_2 x_1$ and the rightmost the minterm $\bar{x_3}\bar{x_2}\bar{x_1}$). If we use the following variable ordering: x_1, x_2, x_3 (x_1 is the most significant variable and x_3 is the least significant), then the MT representation of f is* 11101010 *(the leftmost digit represents the minterm $x_1 x_2 x_3$ and the rightmost the minterm $\bar{x_1}\bar{x_2}\bar{x_3}$).*

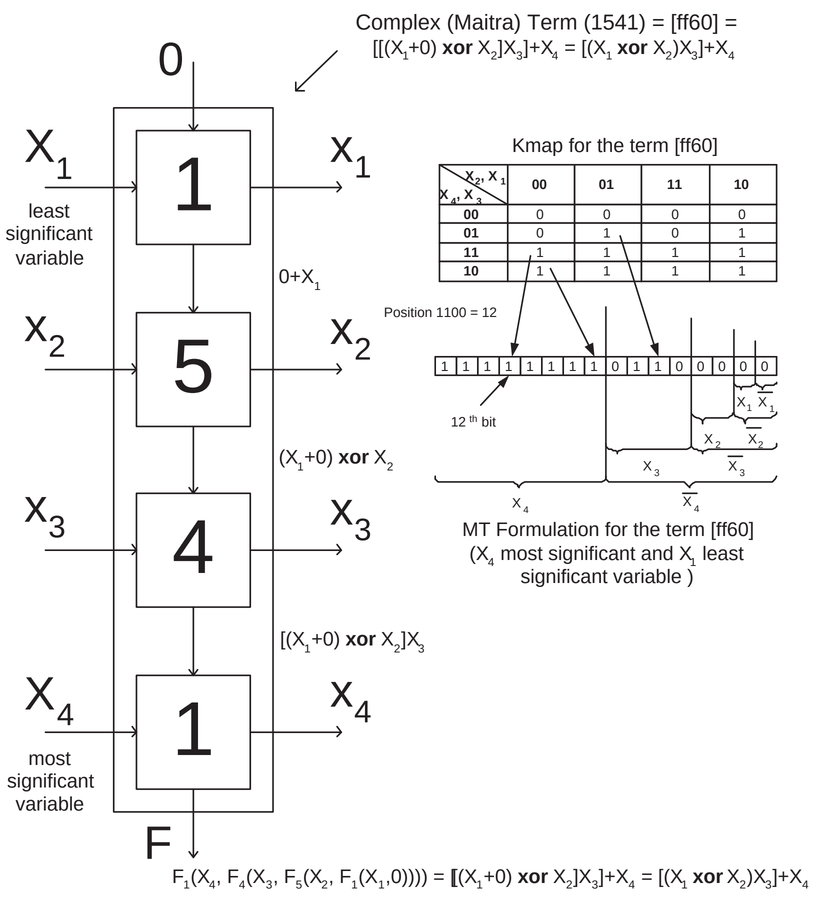

Complex (Maitra) Term (1541) = [ff60] =
[[(X1+0) xor X2]X3]+X4 = [(X1 xor X2)X3]+X4
0
X1
least significant variable
1
5
4
1
X2
X3
X4
most significant variable
0+X1
(X1+0) xor X2
[(X1+0) xor X2]X3
F
Kmap for the term [ff60]
Position 1100 = 12
12 th bit
MT Formulation for the term [ff60]
(X4 most significant and X1 least significant variable)
F1(X4, F4(X3, F5(X2, F1(X1,0)))) = [(X1+0) xor X2]X3]+X4 = [(X1 xor X2)X3]+X4

Figure 5: Maitra Cascade representations.

For the rest of this paper, the MT representation of a Boolean function will be enclosed in brackets and will be displayed using hex digits. The hexadecimal notation is used in place of the binary one in order to improve the readability of the paper. In the previous example the first MT representation will be: $[f8]$ and the second: $[ea]$.

A complex term is characterized by its cells, since its first input is the constant 0. Therefore, we can represent it by a series of cells.

Definition 8 *The cell representation of a complex term, is a series of numbers, corresponding to the series of indices of Maitra cells (Table 1) that belong to the complex term. The leftmost cell corresponds to the cell with the constant input 0 (corresponds to the least significant variable of the complex term) and the rightmost to the one closest to the XOR collector (corresponds to the most significant variable of the complex term).*

***Example* 8** *Let's assume function* $f(x_1, x_2, x_3, x_4) = (x_1 \oplus x_2)x_3 + x_4 = [ff60]$.

This function is one complex term, if we use the following variable ordering: x_4, x_3, x_2, x_1 (x_4 it the most significant variable and x_1 is the least significant). It can be rewritten as: $f = F_1(x_4, F_5(x_5, F_1(x_1, 0))) = (x_4 + (x_3(x_2 \oplus (x_1 + 0))))$. Using cells from the previously defined cell set, it can be represented as: (1541) (the left-most cell is the one with one of its inputs hardwired to constant 0). The Maitra cascade corresponding to this complex term is shown in Fig. 5 (along with it's Karnaugh map and its MT representation).

For the rest of this paper the cell representation of a complex terms will be enclosed in parenthesis (as opposed to the minterm representation, which is enclosed in brackets).

Every Boolean function can be expressed with the help of its subfunctions (they are defined below) through relations, known as Boolean decompositions (or expansions). The Boolean decompositions form the backbone of our theoretical approach for the minimization of Boolean functions.

Definition 9 *Let $f(\boldsymbol{x})$ be a switching function and $\boldsymbol{x}$ the vector of its variables. Let x_i be one of the variables in the vector $\boldsymbol{x}$. Then $f(x_1, x_2, \ldots, x_i = 0, \ldots)$, $f(x_1, x_2, \ldots, x_i = 1, \ldots)$ and $\{f(x_1, x_2, \ldots, x_i = 0, \ldots) \oplus f(x_1, x_2, \ldots, x_i = 1, \ldots)\}$ are subfunctions of f, regarding variable x_i. For simplicity, in the rest of this paper, they will be referred as f_0, f_1 and f_2 respectively and x_i will be referred as x.*

A Boolean function f can be expressed as (Shannon, Positive Davio and Negative Davio respectively):

- $f(\mathbf{x}) = \bar{x}f_0 \oplus xf_1$
- $f(\mathbf{x}) = xf_2 \oplus f_0$
- $f(\mathbf{x}) = \bar{x}f_2 \oplus f_1$

The next definition presents the generator tree, which is the standard decomposition procedure used later by our minimization algorithms.

Definition 10 *Let f be an n-variable switching function. By creating the subfunctions f_1, f_0, f_2 of f and then the subfunctions of f's subfunctions and so on (recursively), a ternary tree is generated. The leftmost branch of each subtree represents the f_1 subfunction of the subtree's root, the middle one represents the f_0 subfunction and the rightmost represents the f_2 subfunction. This decomposition is applied until the constant 0 or 1 function is encountered or a leaf (a two variable Boolean function) is obtained. This tree is named the generator tree.*

***Example* 9** *Let $f(x_1, x_2, x_3, x_4) = [(x_1 + \bar{x_2})\bar{x_3}x_4] \oplus [\bar{x_1}\bar{x_2}x_3\bar{x_4}]$ or in MT formulation: $[0b10]$. Assuming the following variable ordering: x_4, x_3, x_2, x_1, where x_4 is the most significant variable and x_1 is the least significant one, its generator tree can be seen in Fig. 6.*

Theorem 1 (Complement complex term) *The complement function of a complex term is also a complex term. In the complement term, each cell with*

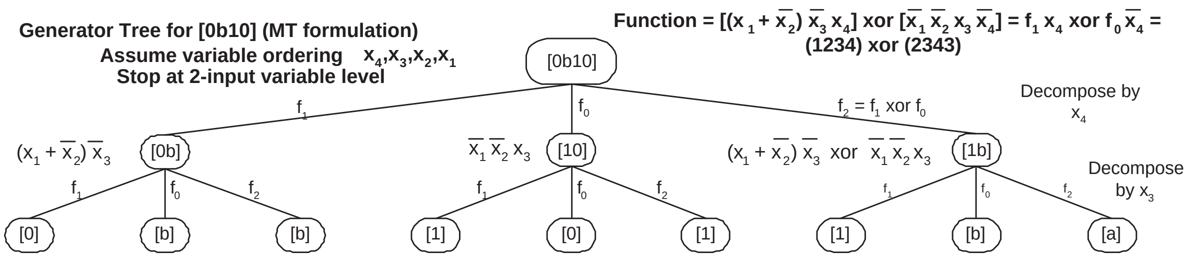

Generator Tree for [0b10] (MT formulation)
Assume variable ordering x_4,x_3,x_2,x_1
Stop at 2-input variable level
Function = $[(x_1 + \overline{x_2})\,\overline{x_3}\,x_4]$ xor $[\overline{x_1}\,\overline{x_2}\,x_3\,\overline{x_4}] = f_1 x_4$ xor $f_0 \overline{x_4}$ =
(1234) xor (2343)
[0b10]
Decompose by x_4
f_1
f_0
$f_2 = f_1$ xor f_0
$(x_1 + \overline{x_2})\,\overline{x_3}$
[0b]
$\overline{x_1}\,\overline{x_2}\,x_3$
[10]
$(x_1 + \overline{x_2})\,\overline{x_3}$ xor $\overline{x_1}\,\overline{x_2}\,x_3$
[1b]
Decompose by x_3
f_1
f_0
f_2
[0]
[b]
[b]
[1]
[0]
[1]
[1]
[b]
[a]

Figure 6: Generator tree.

index belonging to set {1,3} or {2,4} is replaced by the other member of the same set. Cells with index belonging to the set {5,6} remain the same in the complement term. Especially for the cells that have one of their inputs equal to the constant 0, index 1 changes to index 2 and vice versa.

Proof. It can easily be proved exhaustively, using recursion (start from a simple cell). Q.E.D.

***Example* 10** *The complement of a function composed of two complex terms:* $f = [0bb4] = (1234) \oplus (2453)$ *is:* $\overline{f} = (1234) \oplus (2453) \oplus 1 = \overline{(1234)} \oplus (2453) = (2412) \oplus (2453) = (1234) \oplus \overline{(2453)} = (1234) \oplus (1251) = [f44b]$.

Theorem 2 (Complex term $\oplus x$) *The result of the XOR-sum of a complex term* $f = F_n(x_n, F_{n-1}(x_{n-1}, \dots F_1(x_1, 0)))$, *where* F_i *are Maitra cells, with* x_n *(the variable corresponding to the last cell of the complex term) is also a complex term.*

Proof. It can easily be proved exhaustively, considering all possible cell types. Q.E.D.

***Example* 11** *Let* $f(x_1, x_2, x_3, x_4) = (x_1 + \bar{x_2})\bar{x_3}\bar{x_4} = (1233)$ *(x_4 is the most significant variable), then:* $f(x_1, x_2, x_3, x_4) \oplus x_4 = (x_1 + \bar{x_2})\bar{x_3}\bar{x_4} \oplus x_4 = (x_1 + \bar{x_2})\bar{x_3}\bar{x_4} \oplus \bar{x_4} \oplus 1 = ((x_1 + \bar{x_2})\bar{x_3} \oplus 1)\bar{x_4} \oplus 1 = (\bar{x_1}x_2 + x_3)\bar{x_4} \oplus 1 = (x_1 + \bar{x_2})\bar{x_3} + x_4 = (1231)$.

Corollary 1 (Complex term $\oplus \overline{x}$) *The result of the XOR-sum of a complex term* $f = F_n(x_n, F_{n-1}(x_{n-1}, \dots F_1(x_1, 0)))$, *where* F_i *are Maitra cells, with* $\overline{x_n} = x_n \oplus 1$ *(the variable corresponding to the last cell of the complex term) is also a complex term.*

Proof. It is a consequence of Theorems 1 and 2, since $\bar{x} = x \oplus 1$. *Q.E.D.*

***Example* 12** *Let* $f(x_1, x_2, x_3, x_4) = (1233)$ *(x_4 is the most significant variable), then:* $f(x_1, x_2, x_3, x_4) \oplus \overline{x_4} = (1233) \oplus x_4 \oplus 1 = (1231) \oplus 1 = (2413)$.

The rules to create such expressions are presented in Table 3, for complex terms: $F_p(x_n, y)$, $F_q(x_n, y_1) = F_p(x_n, y) \oplus x_n$ and $F_r(x_n, y_2) = F_p(x_n, y) \oplus \overline{x_n}$. In Table 3 y, p is the initial input and the cell index of term F_p and $(y_1, q), (y_2, r)$ are the initial inputs and the cell indices of terms $F_p \oplus x_n, F_p \oplus \bar{x_n}$ (for cell indices, see Table 1).

Table 3: XOR-sum of a complex term with X or $\bar{X}$

p	y_1	q	y_2	r
1	y	3	$\bar{y}$	1
3	y	1	$\bar{y}$	3
2	$\bar{y}$	2	y	4
4	$\bar{y}$	4	y	2
5	y	6	$\bar{y}$	6
6	y	5	$\bar{y}$	5

***Example* 13** *Let $f = (x_1 \oplus x_2)x_3 + \overline{x_4}$ be a complex term or in the cell representation form: (1542). The complement term of f is: $\bar{f} = (2524)$. The XOR-sums of f with x_4 and $\overline{x_4}$ are: $f \oplus x_4 = (2522)$ and $f \oplus \overline{x_4} = (1544)$ respectivelly.*

Corollary 2 *The XOR sum of a switching function f with x, $\overline{x}$ and 1 produces a new switching function with the same weight as f (x is f's most significant variable).*

Proof. It is easily derived from Theorem 1 and 2. Q.E.D.

2.2 Minimization theorems

Definition 11 *An equivalent expression (F_2) of a Maitra expression (F_1) for a switching function $f(x_1, \dots x_n)$ is an expression produced by applying Theorem 1, 2 or Corollary 1 to pairs of terms in the expression F_1.*

***Example* 14** *Let: $F_1 = f(x_1, x_2, x_3, x_4) = ((x_1 + \bar{x_2})\bar{x_3}x_4) \oplus (\bar{x_1}\bar{x_2}x_3\bar{x_4}) = (1234) \oplus (2343)$ (x_4, x_3, x_2, x_1 is the variable significance order, where x_4 is the most significant and x_1 is the least significant one). An equivalent expression of F_1 by applying Theorem 3 is: $F_2 = f(x_1, x_2, x_3, x_4) = (((x_1 + \bar{x_2})\bar{x_3}x_4) \oplus x_4) \oplus ((\bar{x_1}\bar{x_2}x_3\bar{x_4}) \oplus x_4) = ((((x_1 + \bar{x_2})\bar{x_3}) \oplus 1)x_4) \oplus ((\bar{x_1}\bar{x_2}x_3\bar{x_4}) \oplus \bar{x_4} \oplus 1) = ((((x_1 + \bar{x_2})\bar{x_3}) \oplus 1)x_4) \oplus (((\bar{x_1}\bar{x_2}x_3) \oplus 1)\bar{x_4} \oplus 1) = (\overline{((x_1 + \bar{x_2})\bar{x_3})})x_4 \oplus (\overline{(\bar{x_1}\bar{x_2}x_3)}\bar{x_4} \oplus 1 = (x_1 + x_2 + \bar{x_3})x_4 \oplus (\bar{x_1}\bar{x_2}x_3 + x_4))$. Using the cell representation: $F_2 = \{(1234) \oplus x_4\} \oplus \{(2343) \oplus x_4\} = (2414) \oplus (2341)$.*

Theorem 3 *Each minimal expression of a switching function f can always be written in one of the following **compact** forms:*

- *First Compact Form:*
$$f = F_p(x_n, y) \tag{1}$$
(with one subfunction constant e.g. 0 or 1)

- *Second Compact Form:*
$$f = F_p(x_n, y) \oplus F_q(x_n, z) \tag{2}$$

- *Third Compact Form:*
$$f = F_p(x_n, y) \oplus F_q(x_n, z) \oplus F_r(x_n, g) \tag{3}$$

Table 4: Cell indices and inputs for Theorem 3

First Compact Form			Second Compact Form				Third Compact Form					
p	y	constant subfunction	p	q	y	z	p	q	r	f_0	f_1	f_2
1	f_0	$f_1 = 1$	3	4	f_0	f_1	3	4	6	$y \oplus g$	$z \oplus g$	$y \oplus z$
2	f_1	$f_0 = 1$	1	2	$\overline{f_0}$	$\overline{f_1}$	1	2	6	$\bar{y} \oplus g$	$\bar{z} \oplus g$	$\bar{y} \oplus \bar{z}$
3	f_0	$f_1 = 0$	1	4	f_0	$\overline{f_1}$	1	4	6	$\bar{y} \oplus \bar{g}$	$z \oplus \bar{g}$	$\bar{y} \oplus z$
4	f_1	$f_0 = 0$	3	2	$\overline{f_0}$	f_1	3	2	6	$y \oplus \bar{g}$	$\bar{z} \oplus \bar{g}$	$y \oplus \bar{z}$
5	f_0	$f_2 = 1$	3	6	f_2	f_1	1	4	6	$y \oplus g$	$\bar{z} \oplus g$	$y \oplus \bar{z}$
6	f_0	$f_2 = 0$	1	6	$\overline{f_2}$	$\overline{f_1}$	1	4	5	$y \oplus g$	$z \oplus g$	$y \oplus z$
			1	5	f_2	f_1	3	4	5	$y \oplus g$	$\bar{z} \oplus g$	$y \oplus \bar{z}$
			3	5	$\overline{f_2}$	$\overline{f_1}$	3	2	6	$\bar{y} \oplus g$	$z \oplus g$	$\bar{y} \oplus z$
			4	6	f_2	f_0	3	2	5	$\bar{y} \oplus g$	$\bar{z} \oplus g$	$\bar{y} \oplus \bar{z}$
			2	6	$\overline{f_2}$	$\overline{f_0}$	1	2	5	$\bar{y} \oplus g$	$z \oplus g$	$\bar{y} \oplus z$
			4	5	$\overline{f_2}$	f_0	3	4	6	$\bar{y} \oplus \bar{g}$	$\bar{z} \oplus \bar{g}$	$\bar{y} \oplus \bar{z}$
			2	5	f_2	$\overline{f_0}$	3	4	5	$\bar{y} \oplus \bar{g}$	$z \oplus \bar{g}$	$\bar{y} \oplus z$
							1	4	6	$\bar{y} \oplus \bar{g}$	$\bar{z} \oplus \bar{g}$	$\bar{y} \oplus \bar{z}$
							1	2	6	$y \oplus \bar{g}$	$z \oplus \bar{g}$	$y \oplus z$
							1	2	5	$y \oplus \bar{g}$	$\bar{z} \oplus \bar{g}$	$y \oplus \bar{z}$
							3	2	5	$y \oplus \bar{g}$	$z \oplus \bar{g}$	$y \oplus z$

where the valid combinations of cell indices and their corresponding inputs are presented in Table 4.

According to Theorem 3 every minimal ESCT expression of a Boolean function can be written in one of the three compact forms. Table 4 contains all the possible cases, referred by Theorem 3.

The horizontal lines in each column (Compact form) of Table 4 denote forms which are equivalents.

The main minimization theorem, which is the theoretical backbone of our minimization algorithms XMin5 and XMin6 follows.

Theorem 4 *A minimal expression of a switching function f of n variables with weight at most W, can be obtained by merging (XOR summing) switching functions of weight at most $\lfloor W/3 \rfloor$, with the subfunctions of f.*

Q.E.D.

2.3 Additional Theorems

Next we give two lemmas.

Lemma 1 *If f_i, f_j are subfunctions of a Boolean function f of $n+1$ variables, k_1, k_2, k_3 are switching functions of n variables and:*

- $f_i = k_1 \oplus k_3$
- $f_j = k_2 \oplus k_3$

- $w(f) = w(k_1) + w(k_2) + w(k_3)$

then $[w(f_i)+w(f_j)]-w(f) = [w(f_i)+w(f_j)]-[w(k_1)+w(k_2)+w(k_3)] \leq w(k_3)$.

The above lemma holds for every possible pair of subfunctions (f_0, f_1), (f_0, f_2), (f_2, f_1) and denotes that the actual weight of a Boolean function f minus the estimation of its weight produced as $w(f_i) + w(f_j)$, where $f_i, f_j = f_0, f_1, f_2, i \neq j$ is always bounded. If the estimation of the weight is $w(f_0) + w(f_1)$ or $w(f_0) + w(f_2)$ or $w(f_2) + w(f_1)$ then the bound, according to Table 4, is $w(g)$, $w(y)$, $w(z)$ respectively (for example: $f_0 = y \oplus z, f_1 = z \oplus g$, therefore, according to Lemma 1, $w(f_0) + w(f_1) - w(f) \leq w(g)$, since $k_3 = g$).

This fact may be used as a pruning technique in our proposed algorithms.

3 The Minimization Algorithms

Based on the previous theorems and lemmas, we present two algorithms (XMin5 and XMin6) that can find minimal expressions for switching functions of 5 and 6 variables respectively.

First, we will describe XMin5 intuitively. As Theorem 3 suggests, every minimal expression of a Boolean function is in one of the three compact forms. Since we do not know which one, XMin5 must evaluate all three cases. Initially, XMin5 checks if the Boolean function is in the first compact form of the Theorem 3. In this case, as Theorem 3 suggests, one of the subfunctions f_0, f_1, f_2 of the Boolean function is constant (1 or 0). Therefore, the weight of the Boolean function is equal to the weight of the non-constant subfunction.

If the Boolean function has no constant subfunctions then each minimal expression of the Boolean function is either in the second or the third compact form of Theorem 3. Initially, XMin5 assumes that all minimal ESCT expressions of the input Boolean function are in the second compact form. In this case the weight of the function, according to Theorem 4, will be: $w = MIN(w(f_0) + w(f_1), w(f_1) + w(f_2), w(f_0) + w(f_2))$ (refer to the proof of Theorem 4).

Of course, in the general case, the actual weight of our input function may be even less than $MIN(w(f_0) + w(f_1), w(f_1) + w(f_2), w(f_0) + w(f_2))$, in which case there will be at least one minimal expression of our input function in the third compact form of Theorem 3. Theorem 4 shows us how to find this expression. For every possible g function with number of input variables equal to that of subfunctions f_0, f_1, f_2 and weight less or equal to the weight of our input function divided by three, we must create the following XOR sums: $f_0 \oplus g, f_1 \oplus g, f_2 \oplus g$. The expressions which we will produce are of the following form: $f_{k1}(x_n, g) \oplus f_{k2}(x_n, f_i \oplus g) \oplus f_{k3}(x_n, f_j \oplus g)$, where $k1, k2, k3 = p, q, r$ and $f_i, f_j = f_0, f_1, f_2$ and $k1 \neq k2 \neq k3, f_i \neq f_j$ (Table 4). The minimal expression we are searching for, is produced when we select the appropriate g function, so that this expression has the least possible number of complex terms among all the others. Since our input function has 5 input variables, then according to Corollary **??**, the g function must have weight at most 1, thus it must be a complex term. We will call K terms, those g complex terms which produce minimal expressions for our input function. The resulting weight of our input function will be: $w = w(f_i \oplus K) + w(f_j \oplus K) + w(K) = w(f_i \oplus K) + w(f_j \oplus K) + 1$ ($w(K) = 1$, since K is a complex term) (third compact form).

In this last case, since there is no profound way to find the appropriate K complex term, we must check all possible 4 variable complex terms to determine if one or more of them produce minimal expressions for our input function in the third compact form of Theorem 3.

Algorithm XMin6 is similar to XMin5 but works for 6-variable Boolean functions. XMin6, like XMin5, uses the results of Corollary **??**, Corollary 2 and algorithm XMin5 in order to find the weight of a switching function (and at least one minimal expression for it). During the phase where XMin6 tries to determine if there is a minimal expression in the third compact form of Theorem 3, it uses 5-variable Boolean function with weight at most 3 (due to Corollary **??**).

4 Conclusions

An algorithm has been proposed in this paper for the exact minimization of ESCT expressions for single-output switching functions of up to 6 variables. To the best of the authors' knowledge this is the first algorithm in the literature that can find minimal ESCT expressions for switching functions of 6 variables and moreover in a practical computing time.

References

[1] T. Sasao "Switching Theory For Logic Synthesis", Kluwer Academic Publishers, 1999.

[2] K.K. Maitra "Cascaded switching networks of two-input flexible cells" IRE Trans. Electron. Comput., pp. 136-143, 1962.

[3] A. Mishchenko, M. Perkowski, "Logic Synthesis of Reversible Wave Cascades", In proc. of International Workshop on Logic And Synthesis 2002, New Orleans, Louisiana, June 4-7, 2002, pp. 197-202.

[4] C. Bennet "Logical Reversibility of Computation", IBM Journal of Research and Development, 17, 1973, pp. 525-532.

[5] J. Preskill "Lecture notes in quantum computing", http://www.Theory.caltech.edu/ preskill/ph229

[6] A. Sarabi, N. Song, M. Chrzanowska-Jeske, M. Perkowski"A comprehensive approach to logic synthesis and physical design for two-dimensional logic arrays", In proc. of Design Automation Conference 1994, pp. 321-326.

[7] I. Schaefer, M. Perkowski, H. Wu "Multilevel logic synthesis for cellular FPGAs based on orthogonal expansions", Proc, IFIP WG 10.5 Workshop on Applications of the Reed-Muller Expansion in Circuit Design, Sept. 1993, Hamburg, Germany, pp. 42-51.

[8] G. Lee "Logic synthesis for celullar architecture FPGA using BDD", Proc. Asia and South Pacific Design Automation Conf. 1997, Jan 1997, pp 253-258 .

[9] G. Lee, R. Drechsler "ETDD-based Synthesis of term-based FPGAs for incompletely specified Boolean functions", Proc. Asia and South Pacific Design Automation Conf. 1998, Feb 1998, pp 75-80.

[10] P. Lindgren, R. Drechsler, B. Becker "Look-up table FPGA synthesis from minimized multi-valued pseudo kronecker expressions", in proc. of the 28th IEEE International Symposium on Multiple-Valued Logic, 27-29 May 1998, pp. 95 - 100.

[11] N. Song, M. Perkowski "Minimization of exclusive sums of multi-valued complex terms for logic cell arrays", in proc. of the 28th IEEE International Symposium on Multiple-Valued Logic, 27-29 May 1998, pp. 32 - 37.

[12] N. Song, M. Perkowski "A new approach to and/or/exor factorization for regular arrays", Proc. 1998 Euromicro, Vasteras, Sweden, August 25-27, 1998, pp. 269-276.

[13] M. Helliwell, M. Perkowski "A fast algorithm to minimized multi-output mized polarity generalized reed-muller forms", Proc 25th ACM IEEE Conference on Design Automation, Atlantic city, New Jersey, USA, 1988, pp. 427-432.

[14] D. Maslov, G. Dueck "Garbage in Reversible Designs of Multiple Output Functions", in proc. 6th International Symposium on Representations and Methodology of Future Computing Technology, Trier, Germany, March 2003, pp. 162-170.

[15] D. Voudouris, S. Stergiou, G. Papakonstantinou "Minimization of reversible wave cascades", IEICE Trans. on Fundamentals, Vol E88-A, No. 4, 2005/04, pp. 1015-1023.

[16] D. Voudouris, G. Papakonstantinou "Maitra Cascade Minimization", in proc. of the 6th IWSBP, September 2004, Freiberg (Sachsen), Germany, pp. 209 - 220.

[17] D. Voudouris, M. Kalathas, G. Papakonstantinou "Decomposition of Multi-output Boolean Functions", HERCMA 2005, Athens, Hellas, HERMIS Journal Vol 6-2005, pp. 154-161.

[18] R. C. Minnick, "Cutpoint cellular logic" IEEE Trans. Electron. Comput., vol. EC-13, Dec, 1964, pp. 685-698.

[19] G. Papakonstantinou "Cascade Transformation", IEEE Transactions on computers, Jan 1976, 25(1), pp. 93 - 95.

[20] G. Papakonstantinou, F. Gritzali "Modulo-2 expressions of switching functions", Electronic Letters, vol 12, issue 10, May 1976, pp. 244 - 245.

[21] T. Hirayama, Y. Nishitani, T. Sato "A faster algorithm of minimizing AND-EXOR expressions", IEICE Trans. on Fund., Vol E85-A, No. 12, pp. 2708-2714, Dec. 2002.

[22] S. Stergiou, D. Voudouris, G. Papakonstantinou "Exact and Heuristic MVESOP Minimization Algorithms", IEICE Trans. on Fundamentals, Vol E87-A No.5, pp.1226.

[23] A. Mishchenko, M. Perkowski "Fast Heuristic Minimization of exclusive-sums-of-products", in proc. of the 5th International Workshop on Applications of the Reed Muller Expansion in Circuit Design, Stackville, Mississippi, Aug. 2001, pp. 242 - 250.