A transformation of multiple-valued input two-valued output functions and its application to simplification of exclusive-or sum-of-products expressions

Tsutomu SASAO
Department of Computer Science and Electronics
Kyushu Institute of Technology
Iizuka 820, Japan

Abstract

This paper presents a transformation for p-valued input functions. The number of products in minimum exclusive-or sum-of-products expressions (ESOPs) is invariant under this transformation. Algorithms for reducing the number of product terms in ESOPs using this transformation are presented for $p = 2$ and $p = 4$. Arithmetic functions are simplified to show the ability of this approach.

1 Introduction

Recently automatic logic synthesis tools are extensively used in VLSI design. Most logic synthesis tools use AND and OR gates as basic logic elements, and they derive multi-level logic circuits from AND-OR two-level circuits. Thus the minimization of sum-of-products expressions (SOPs), which corresponds to the minimization of AND-OR two-level circuits, is vitally important in such tools. It is known that arithmetic and error correcting circuits can be realized with many fewer gates if EXOR gates are available as well as AND and OR gates. Such circuits can be derived from AND-EXOR two-level circuits. So the minimization of Exclusive-OR sum-of-products expressions (ESOPs), which corresponds to the minimization of AND-EXOR two-level circuits, is also important. It has been long conjectured that ESOPs require fewer products than SOPs to represent same functions [4, 11, 13]. Our recent research shows that ESOPs require fewer products than SOPs to realize randomly generated functions and symmetric functions [18, 16]. To realize an arbitrary function of 6 variables, an ESOP requires only 16 products, whereas an SOP require 32 products [8]. As for the 4-variable functions, ESOPs require, on the average, 3.66 products, whereas SOPs require 4.13 products [7]. Although there exists a class of functions whose ESOP realization require more products than SOP's [15], we believe that the ESOP's will be important tool in efficient logic design.

Table 1: Number of AND gates and connections to realize arithmetic functions by AND-OR and AND-EXOR circuits

Data	# of AND gates				# of connections			
	AND-OR		AND-EXOR		AND-OR		AND-EXOR	
Name	1bit	2bit	1bit	2bit	1bit	2bit	1bit	2bit
ADR4	75	17	31	11	423	139	168	99
LOG8	123	98	96	94	1019	1162	785	1090
MLP4	121	85	61	52	889	910	441	467
NRM4	120	70	73	56	887	799	602	618
RDM8	76	52	31	26	406	431	181	208
ROT8	57	38	35	28	389	414	280	353
SQR8	180	147	114	112	1398	1675	809	1181
WGT8	255	54	54	25	2078	530	356	207

The number of products in AND-OR two-level circuits can be reduced by adding decoders to the inputs (i.e., AND-OR PLAs with two-bit decoders)[17]. In a similar way, the number of products in AND-EXOR two-level circuits can be reduced by adding decoders to the inputs (i.e., AND-EXOR PLAs with two-bit decoders)[16, 12]. Table1 compares the number products and literals to represent arithmetic functions of 8-inputs by AND-OR circuits and AND-EXOR circuits with one and two-bit decoders, where a one-bit decoder generates true and complemented variables. This table implies the circuits based on ESOP require fewer gates than the ones based on SOPs. The minimization of AND-EXOR PLAs with decoders can be done by the minimization of ESOPs with p-valued inputs ($p \geq 2$) [16, 12]. So, an efficient minimization algorithm for ESOPs is useful for the design of such circuits. However, no efficient method is known for deriving minimum ESOPs (MESOPs) even for the two-valued case. Several heuristic algorithms have been developed for near minimum ESOPs [14, 5, 6, 18, 16, 2, 3].

The first topic of this paper is an L-transformation which is useful for minimizing ESOPs. Suppose that F_1 is an ESOP of a logic function, and G_1 is obtained by the transformation. Let F_m be an MESOP of F_1. Then, a MESOP for G is obtained by applying the same transformation to F_1 (Fig.1). Fig.2 shows an example of L-transformation. Let F_1 be an given expression, which is not so simple to minimize. If we interchange literals 1 and x in F_1, then we have expression G_1. Because $x \cdot y \oplus \bar{x} \cdot y = (x \oplus \bar{x}) \cdot y = 1 \cdot y = y$, G_1 is simplified

$$\begin{tikzcd}
\text{ESOP} & & \text{MESOP} \\
F_1 \arrow[rr, Rightarrow, "\text{minimization}"] \arrow[d, leftrightarrow] & & F_m \arrow[d, leftrightarrow] \\
G_1 \arrow[rr, Rightarrow, "\text{minimization}"'] & \text{L-transformation} & G_m
\end{tikzcd}$$

Figure 1: Concept of L-transformation

$$\begin{tikzcd}
\text{ESOP} & & \text{MESOP} \\
F_1 = x \cdot 1 \oplus 1 \cdot y \oplus \bar{x} \cdot y \arrow[rr, Rightarrow] \arrow[d, leftrightarrow] & & F_m = x \cdot \bar{y} \arrow[d, leftrightarrow] \\
G_1 = 1 \cdot 1 \oplus x \cdot y \oplus \bar{x} \cdot y \arrow[r, Rightarrow] & G_2 = 1 \oplus y \arrow[r, Rightarrow] & G_m = 1 \cdot \bar{y}
\end{tikzcd}$$

$$1 \longleftrightarrow x$$

Figure 2: Example of L-transformation

to G_2, and finally, to G_m, which is MESOP. And if we interchange literals 1 and x in G_m, then we have expression F_m, which is MESOP for F_1. The L-transformation produces an equivalence relation, and the numbers of products in MESOPs are the same for the functions in the same equivalence class. Therefore, any function in an equivalence class can be minimized to obtain the MESOPs for other functions in the class. Because some functions are easier to minimize than others in the same equivalence class, we can transform the given function into another one to make the function easier to minimize.

The second topic of this paper are fast simplification methods for ESOPs. We present algorithms to reduce the number of non-zero outputs for $p = 2$ and $p = 4$. Because these algorithms are simple and fast, they can be used for generating initial solutions for (near) minimal ESOPs. They produce ESOP's with fewer products than other method such as [1].

This paper is organized as follows:

2 defines the ESOPs and shows basic properties. **3** introduces the L-transformations and shows several examples. **4** presents fast transformation algorithms to reduce the number of non-zero outputs for $p = 2$ and $p = 4$. **5** shows simplification results of arithmetic functions for $p = 2$ and $p = 4$. **6** discusses the applications of L-transformation to logic minimization .

2 Definition and Basic Properties

In this section, we show some definitions of multiple-valued input two-valued output functions and basic properties of ESOPs.

Definition 1 *Let $P_i = \{0, 1, \cdots, p_i - 1\}$, $p_i \geq 2(i = 1, 2, \cdots, n)$, and $B = \{0, 1\}$. $f : \times_{i=1}^{n} P_i \to B$ is a* multiple-valued input two-valued output function . *For simplicity we assume that $P_i = P$ and $p_i = p(i = 1, 2, \cdots, n)$. A multiple-valued input two-valued output function is simply called a* function .

Definition 2 *Suppose that $S \subseteq P$. X^S is a* literal *of X, where*

$$X^S = \begin{cases} 0 & (X \notin S) \\ 1 & (X \in S) \end{cases}$$

When S contains only one element, $X^{\{i\}}$ is sometimes denoted by X^i. A logical product of literals

$$X_1^{S_1} X_2^{S_2} \cdots X_n^{S_n}$$

is a product term. *An exclusive-or sum-of-products*

$$\sum_{(S_1, S_2, \cdots, S_n)} \oplus X_1^{S_1} X_2^{S_2} \cdots X_n^{S_n}$$

is an exclusive-or sum-of-products expression*(ESOP), where $\sum_{(S_1, S_2, \cdots, S_n)} \oplus$ denotes the exclusive-or for tuples $(S_1, S_2, \cdots, S_n)$.*

Definition 3 *An ESOP for a function f with the minimum number of products is* minimum ESOP*(MESOP) for f. The number of products in an ESOP F is denoted by $\tau(F)$. The number of products in MESOP for f is denoted by $\tau(f)$.*

When $p = 2$, MESOP is also called a *minimal mod 2 sum-of-products*[11]. No minimization method is known except for the exhaustive method [3, 7] even for $p = 2$.

Lemma 1 *An arbitrary n-variable function $f(X_1, X_2, \cdots, X_n)$ is uniquely represented by the ESOP*

$$f = \sum_{(a_1, a_2, \cdots, a_n)} \oplus f(a_1, a_2, \cdots, a_n) \cdot X_1^{a_1} X_2^{a_2} \cdots X_n^{a_n},$$

where $\sum \oplus$ denotes the exclusive-or for all the combinations of $a_i \in P(i = 1, 2, \cdots, n)$, and $f(a_1, a_2, \cdots, a_n) \in \{0, 1\}$.

Definition 4 *Let S be a subset of $P = \{0, 1, \cdots, p-1\}$. $\vec{a} = (a_0, a_1, \cdots, a_{p-1})$ is called a* characteristic vector *of S, where*

$$a_i = \begin{cases} 0 & (i \notin S) \\ 1 & (i \in S). \end{cases}$$

Lemma 2 *An arbitrary n variable function $f(X_1, X_2, \cdots, X_n)$ can be uniquely represented as*

$$f = \sum_{j=0}^{p-1} \oplus X_1^j f(j, X_2, \cdots, Xn)$$

This is a Shannon expansion of p-valued logic function.

Example 1 *Consider the 4-valued input function shown in Fig.3. The Shannon expansion of this function is:*

$$X_1^0 \cdot X_2^{\{0,1\}} \oplus X_1^1 \cdot X_2^{\{1,2\}} \oplus X_1^2 \cdot X_2^2. \qquad (1)$$

		X_1			
		0	1	2	3
	0	1	0	0	0
	1	1	1	0	0
X_2	2	0	1	1	0
	3	0	0	0	0

Figure 3: Example for 4-valued input function

3 L-Transformation

In this section, we introduce L-transformation under which the number of products in MESOP's are invariant.

Theorem 1 (Expansion Theorem)
A necessary and sufficient condition for an arbitrary function $P^n \rightarrow B$ to have a unique representation of

$$f = X^{S_0} \cdot h_0 \oplus X^{S_1} \cdot h_1 \oplus \cdots \oplus X^{S_{p-1}} \cdot h_{p-1} \qquad (2)$$

is that the matrix $M = \begin{bmatrix} \vec{a_0} \\ \vec{a_1} \\ \vdots \\ \vec{a_{p-1}} \end{bmatrix}$ *be regular, where* $\vec{a_i}$ $(i = 0, 1, \cdots, p-1)$ *is a characteristic vector of* S_i.

(Proof) By rewriting (2), we have

$$f = \sum_{j=0}^{p-1} \oplus X^{S_j} \cdot h_j.$$

A Shannon expansion of f is

$$f = \sum_{j=0}^{p-1} \oplus X^j \cdot f_j.$$

Let I be a $p \times p$ unit matrix. Then the above two relations can be expressed as

$$[h_0, h_1, \cdots, h_{p-1}] \cdot M = [f_0, f_1, \cdots, f_{p-1}] \cdot I.$$

When M is regular, the inverse matrix M^{-1} exists, and

$$[h_0, h_1, \cdots, h_{p-1}] = [f_0, f_1, \cdots, f_{p-1}] \cdot M^{-1}.$$

This shows that $[h_0, h_1, \cdots, h_{p-1}]$ is uniquely represented. If M is not regular, f cannot be represented uniquely. (Q.E.D.)

Example 2
1) Consider an expansion of a 4-valued input function:

$$f = X^0 \cdot h_0 \oplus X^1 \cdot h_1 \oplus X^2 \cdot h_2 \oplus X^3 \cdot h_3.$$

Because the matrix $M = \begin{bmatrix} 1\,0\,0\,0 \\ 0\,1\,0\,0 \\ 0\,0\,1\,0 \\ 0\,0\,0\,1 \end{bmatrix}$ *is regular, the above expression has a unique representation. For example, the function shown in Fig.3 can be shown as (1).*
2) Consider another expansion of a 4-valued input function:

$$f = X^0 \cdot h_0 \oplus X^1 \cdot h_1 \oplus X^2 \cdot h_2 \oplus X^{\{1,2\}} \cdot h_3.$$

In this case, the matrix $M = \begin{bmatrix} 1\,0\,0\,0 \\ 0\,1\,0\,0 \\ 0\,0\,1\,0 \\ 0\,1\,1\,0 \end{bmatrix}$ *is not regular, and the above expression cannot represent the function uniquely. In fact, The function in Fig.3 can be represented as either*

$$f = X_1^0 \cdot X_2^{\{0,1\}} \oplus X_1^1 \cdot X_2^{\{1,2,3\}} \oplus X_1^2 \cdot X_2^{\{2,3\}} \oplus X_1^{\{1,2\}} \cdot X_2^3,$$

or

$$f = X_1^0 \cdot X_2^{\{0,1\}} \oplus X_1^1 \cdot X_2^{\{1,2\}} \oplus X_1^2 \cdot X_2^2 \oplus X_1^{\{1,2\}} \cdot 0.$$

Definition 5 *Let a function f be expanded as*

$$f = \sum_{j=0}^{p-1} \oplus X^j \cdot f_j.$$

Consider the following transformation which converts sub-functions $[f_0, f_1, \cdots, f_{p-1}]$ into $[g_0, g_1, \cdots, g_{p-1}]$ by a regular matrix M:

$$\begin{bmatrix} g_0 \\ g_1 \\ \vdots \\ g_{p-1} \end{bmatrix} = M \cdot \begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_{p-1} \end{bmatrix} = \begin{bmatrix} \vec{m_0} \\ \vec{m_1} \\ \vdots \\ \vec{m_{p-1}} \end{bmatrix} \cdot [\vec{f}\,]^t \qquad (3)$$

Then

$$g = \sum_{j=0}^{p-1} \oplus X^j \cdot g_j$$

is an L-transformed function generated by M, *where* $g_j = \vec{m_j} \cdot [f_0, f_1, \cdots, f_{p-1}]^t$.

Definition 6 *Let*

$$F = \sum_{\vec{k} \in B^p, \vec{k} \neq 0} \oplus X^{S_{\vec{k}}} \cdot F(\vec{k}) \qquad (4)$$

be an arbitrary ESOP, where $S_{\vec{k}}$ is a subset of P, and $\vec{k}$ is the the characteristic vector of $S_{\vec{k}}$. Let L be a matrix with p rows and $2^p - 1$ columns, where each column denotes the characteristic vector of a literal. Then,

$$L_a = M \cdot L = M \cdot [L(0), L(1), \cdots, L(2^p - 1)]$$

is a matrix showing the literals after transformation. The ESOP

$$G = \sum_{\vec{k} \in B^p, \vec{k} \neq 0} \oplus X^{SL_a(\vec{k})} \cdot F(\vec{k}), \qquad (5)$$

is an L-transformed expression generated by M, *where $La(\vec{k})$ represents a column vector $\vec{k}$ transformed by M, and $SL_a(\vec{k})$ denotes a literal $S_{\vec{k}}$ transformed by M.*

Lemma 3 *Let f be an arbitrary function, and g be an L-transformed function. Let F be an arbitrary ESOP for f, and G be the L-transformed expression generated by the same matrix. Then, G represents g.*

(Proof) Let (4) be an ESOP for f. Let $C_j(j = 0, 1, \cdots, p-1)$ be the set of the column vectors of the matrix L whose j-th component is 1. Because F represents a function f,

$$F(X = j) = f_j = \sum_{\vec{k} \in C_j} \oplus F(\vec{k}). \tag{6}$$

From here, we will show that (5) represents the function g. By (3) and (6), the relation between g and f is given by

$$\begin{aligned} g_j &= \vec{m}_j \cdot [f_0, f_1, \cdots, f_{p-1}]^t \\ &= \vec{m}_j \cdot [\sum_{\vec{k}_0 \in C_0} \oplus F(\vec{k}_0), \sum_{\vec{k}_1 \in C_1} \oplus F(\vec{k}_1), \cdots, \\ &\quad \sum_{\vec{k}_{p-1} \in C_{p-1}} \oplus F(\vec{k}_{p-1})]^t \\ &= m_{j0}(\sum_{\vec{k}0 \in C_0} \oplus F(\vec{k}_0)) \oplus m_{j1}(\sum_{\vec{k}1 \in C_1} \oplus F(\vec{k}_1) \oplus \cdots \\ &\quad \oplus m_{jp-1}(\sum_{\vec{k}_{p-1} \in C_{p-1}} \oplus F(\vec{k}_{p-1})) = \sum_{\vec{m}_j \cdot L(\vec{k}) = 1} \oplus F(\vec{k}) \end{aligned}$$

On the other hand, the function represented by G is

$$\begin{aligned} G(X = j) &= \sum_{L_a(\vec{k}) \geq (0, \cdots, \underbrace{1}_{j}, \cdots, 0)^t} \oplus F(\vec{k}) \\ &= \sum_{M \cdot L(\vec{k}) \geq (0, \cdots, \underbrace{1}_{j}, \cdots, 0)^t} \oplus F(\vec{k}) \\ &= \sum_{mj \cdot L(\vec{k}) = 1} \oplus F(\vec{k}) \end{aligned}$$

Therefore, $G(X = j)$ represents the function g_j. Hence G represents the function g. (Q.E.D.)
Lemma 3 shows that the transformation defines a new function independently of the representation of the original function. Thus, we have the following:

Theorem 2 (Transformation Theorem) *Let F_1 and F_2 be ESOPs for a function f. Let G_1 and G_2 be ESOPs which are obtained by an L-transformation from F_1 and F_2, respectively. Then G_1 and G_2 represent the same function.*

This is a multiple-valued extension of the two-valued transformation theory [3, 15].

Theorem 3 *Let F be an ESOP for a function f. Let G be the ESOP obtained by an L-transformation from F by a regular matrix M. Let g be a function represented by G. Then $\tau(f) = \tau(g)$.*

(Proof) By Theorem 2 , G represents a function g independently of the representation of F. G is an ESOP obtained by an L-transformation from F, and the number of the products in the two ESOPs are the same, i.e., $\tau(F) = \tau(G)$. Therefore, if F is an MESOP, then G is also an MESOP. Hence $\tau(f) = \tau(g)$. (Q.E.D.)
Because an L-transformation for a variable corresponds to a regular $p \times p$ matrix, it is associative. For literals corresponding to different variables, the transformation is commutative. The total number of different L-transformations for p-valued n-variable functions is
(# of different regular $p \times p$ matrices)^p.
Figs.1 and 2 illustrate the above theorem.

Example 3 *When $p = 2$, f represents an ordinary two-valued logic function $f : B^n \to B$. A Shannon expansion of f is*

$$f(X_1, X_2, \cdots, X_n) = X_i^0 f_0 \oplus X_i^1 f_1.$$

1) When $M_1 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$ is the transformation matrix, we have

$$\begin{bmatrix} g_0 \\ g_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} f_0 \\ f_1 \end{bmatrix}.$$

From above, we obtain $g_0 = f_0$ and $g_1 = f_0 \oplus f_1$. Next, by the Shannon expansion of g, we have

$$\begin{aligned} g(X_1, X_2, \cdots, X_n) = \quad & X_i^0 g_0 \oplus X_i^1 g_1 \\ = X_i^0 f_0 \oplus X_i^1 (f_0 \oplus f_1) = & X_i^{\{0,1\}} \cdot f_0 \oplus X_i^1 f_1 \end{aligned}$$

The above expression shows that interchanging the literals X_i^0 and $X_1^{\{0,1\}}$ in the ESOP for f gives the ESOP for g. Let $L = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix}$ be a matrix representing the $2^2 - 1$ different literals. A matrix representing the literals after the L-transformation is given by

$$M_1 \cdot L = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}.$$

This equation shows that literals are transformed as $X_i^0 \leftrightarrow X_i^{\{0,1\}}$. Let

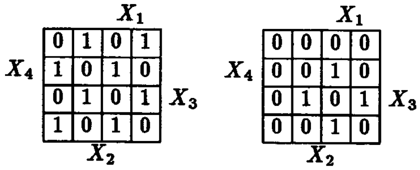
$$F = X_i^0 h_0 \oplus X_i^1 h_1 \oplus X_i^{\{0,1\}} \cdot h_2$$

be an arbitrary ESOP for f. By applying the L-transformation M_1 to F, we have

$$M_1(F) = X_i^{\{0,1\}} h_0 \oplus X_i^1 h_1 \oplus X_i^0 \cdot h_2$$

Because F represents a function f, we have $f_0 = F(X = 0) = h_0 \oplus h_2$ and $f_1 = F(X = 1) = h_1 \oplus h_2$. By the definition of g, we obtain

$$g = X_i^0 f_0 \oplus X_i^1 (f_0 \oplus f_1).$$



0	1	0	1
1	0	1	0
0	1	0	1
1	0	1	0

(a)Original function

0	0	0	0
0	0	1	0
0	1	0	1
0	0	1	0

(b)Transformed function

Figure 4:

From this, we have

$$\begin{aligned} g &= X_i^0(h0 \oplus h_2) \oplus X_i^1(h_0 \oplus h_1) \\ &= X_i^{\{0,1\}} \cdot h_0 \oplus X_i^1 \cdot h_1 \oplus X_i^0 \cdot h_2 = M_1(F). \end{aligned}$$

This also shows that the ESOP obtained from F with the L-transformation M_1 represents function g.

2) When $M_2 = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$ is the transformation matrix, we have

$$M_2 \cdot L = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}.$$

This equation shows that literals are interchanged as $X_i^1 \leftrightarrow X_i^{\{0,1\}}$. If X^i is represented by x and $X^{\{0,1\}}$ is represented by 1, then we have the L-transformation in Fig.2.

Example 4 *When $M_3 = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$ is the transformation matrix, we have*

$$M_3 \cdot L = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}.$$

This shows that literals are interchanged as

$$X_i^0 \rightarrow X_i^{\{0,1\}} \rightarrow X_i^1 \rightarrow X_i^0.$$

Consider the function f shown in Fig.4(a). The MESOP is

$$\begin{aligned} f = {} & X_1^1 \cdot X_2^{\{0,1\}} \cdot X_3^{\{0,1\}} \cdot X_4^{\{0,1\}} \\ & \oplus X_1^{\{0,1\}} \cdot X_2^1 \cdot X_3^{\{0,1\}} \cdot X_4^{\{0,1\}} \\ & \oplus X_1^{\{0,1\}} \cdot X_2^{\{0,1\}} \cdot X_3^1 \cdot X_4^{\{0,1\}} \\ & \oplus X_1^{\{0,1\}} \cdot X_2^{\{0,1\}} \cdot X_3^{\{0,1\}} \cdot X_4^1 \end{aligned}$$

However, it is not easy to find the MESOP from the map. Apply transformation M_3 to the variables of f, and we have

$$\begin{aligned} g = {} & X_1^0 \cdot X_2^1 \cdot X_3^1 \cdot X_4^1 \oplus X_1^1 \cdot X_2^0 \cdot X_3^1 \cdot X_4^1 \\ & \oplus X_1^1 \cdot X_2^1 \cdot X_3^0 \cdot X_4^1 \oplus X_1^1 \cdot X_2^1 \cdot X_3^1 \cdot X_4^0 \end{aligned}$$

Fig.4(b) shows the map for g, which has fewer minterms and is easier to minimize than f.

Example 5 *When $p = 4, f$ denotes a 4-valued input function $f^n : P \rightarrow B$, where $P = \{0,1,2,3\}$. Let*

$$L = \begin{bmatrix} 1&0&0&0&1&1&1&0&0&0&0&1&1&1&1 \\ 0&1&0&0&1&0&0&1&1&0&1&0&1&1&1 \\ 0&0&1&0&0&1&0&1&0&1&1&1&0&1&1 \\ 0&0&0&1&0&0&1&0&1&1&1&1&1&0&1 \end{bmatrix}$$

be the matrix representing the $2^4 - 1$ different literals.

1) When $M_4 = \begin{bmatrix} 0&1&0&0 \\ 1&0&0&0 \\ 0&0&1&0 \\ 0&0&0&1 \end{bmatrix}$ is the transformation matrix, we have

$$M_4 \cdot L = \begin{bmatrix} 0&1&0&0&1&0&0&1&1&0&1&0&1&1&1 \\ 1&0&0&0&1&1&1&0&0&0&0&1&1&1&1 \\ 0&0&1&0&0&1&0&1&0&1&1&1&0&1&1 \\ 0&0&0&1&0&0&1&0&1&1&1&1&1&0&1 \end{bmatrix}$$

This equation shows that the function g is obtained by replacing the literals in the ESOP for f as follows:

$$X^{\{0\}} \leftrightarrow X^{\{1\}}, X^{\{1,2\}} \leftrightarrow X^{\{0,2\}},$$
$$X^{\{1,3\}} \leftrightarrow X^{\{0,3\}}, and\, X^{\{0,2,3\}} \leftrightarrow X^{\{1,2,3\}}.$$

2) When $M_5 = \begin{bmatrix} 1&0&0&0 \\ 0&1&0&0 \\ 0&0&1&0 \\ 1&0&0&1 \end{bmatrix}$ is a transformation matrix, literals are transformed as follows:

$$X^{\{0\}} \leftrightarrow X^{\{0,3\}}, X^{\{0,1\}} \leftrightarrow X^{\{0,1,3\}},$$
$$X^{\{0,2\}} \leftrightarrow X^{\{0,2,3\}}, and\, X^{\{0,1,2\}} \leftrightarrow X^{\{0,1,2,3\}}.$$

4 Transformation Algorithms

4.1 General Strategy

L-transformations classify a set of logic functions into equivalence classes. All the functions in an equivalence class have MESOPs with the same number of products. An MESOP for a function is easily obtained from an MESOP for another function in the same equivalence class. Because some functions are easier to minimize than others, we can transform a given function into another one which is easier to minimize.

Definition 7 *Two functions f and g are* L-equivalent *if g can be obtained from f by a series of L-transformations.*

Example 6 *The set of two-variable two-valued logic functions can be classified into 3 classes by L-transformations as follows:*

- Class 0: 0
- Class 1: $X_1^1X_2^1, X_1^0X_2^1, X_1^1X_2^0, X_1^0X_2^0, X_1^1, X_1^0, X_2^1, X_2^0, 1$

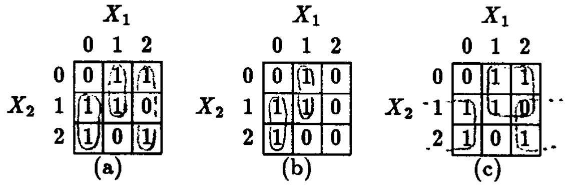


Figure 5: Example 4.2

- Class 2: $X_1^1 \oplus X_2^1, X_1^0 \oplus X_2^1, X_1^1 X_2^0 \oplus X_2^0,$ $X_1^1 X_2^1 \oplus X_2^0, X_1^0 X_2^1 \oplus X_2^0, X_1^0 X_2^0 \oplus X_2^1.$

Note that the functions in the Class i *require i products in MESOPs.*

For $p = 2$, the number of different regular matrices is 6, and the number of n-variable functions which are L-equivalent to a given function is at most 6^n. For $p = 3$, the number of different regular matrices is 168, and the number of n-variable functions which are L-equivalent to a given function is at most $(168)^n$. For $p = 4$, the number of different regular matrices is 20160, so the number of n-variable functions which are L-equivalent to a given one is up to $(20160)^n$.

Example 7 *1. Consider the 3-valued 2-variable function shown in Fig.5(a). The ESOP with three products is easy to derive:*

$$f = X_1^{\{0\}} \cdot X_2^{\{1,2\}} \oplus X_1^{\{1\}} \cdot X_2^{\{0,1\}} \oplus X_1^{\{2\}} \cdot X_2^{\{0,2\}}.$$

However, it is not easy to find a two-product solution from this map.

2. Next, consider the L-transformation $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}$,

$X_1^{\{0\}} \leftrightarrow X_1^{\{0,2\}}, X_1^{\{1\}} \leftrightarrow X_1^{\{1,2\}}, X_1^{\{0,1\}} \leftrightarrow X_1^{\{0,1,2\}}.$

The above function is transformed to

$$\begin{aligned} g &= X_1^{\{0,2\}} \cdot X_2^{\{1,2\}} \oplus X_1^{\{1,2\}} \cdot X_2^{\{0,1\}} \oplus X_1^{\{2\}} \cdot X_2^{\{0,2\}} \\ &= X_1^{\{0\}} \cdot X_2^{\{1,2\}} \oplus X_1^{\{1\}} \cdot X_2^{\{0,1\}} \\ &\quad \oplus X_1^{\{2\}} \cdot (X_2^{\{1,2\}} \oplus X_2^{\{0,1\}} \oplus X_2^{\{0,2\}}) \\ &= X_1^{\{0\}} \cdot X_2^{\{1,2\}} \oplus X_1^{\{1\}} \cdot X_2^{\{0,1\}}. \end{aligned}$$

Fig.5(b) is the map for g, which shows that g requires only two products.

3. Finally, consider the reverse transformation:

$X_1^{\{0\}} \leftrightarrow X_1^{\{0,2\}}, X_1^{\{1\}} \leftrightarrow X_1^{\{1,2\}}, X_1^{\{0,1\}} \leftrightarrow X_1^{\{0,1,2\}}.$

The previous ESOP is transformed to

$$f = X_1^{\{0,2\}} \cdot X_2^{\{1,2\}} \oplus X_1^{\{1,2\}} \cdot X_2^{\{0,1\}}.$$

Fig.5(c) shows the map of the reduced ESOP.

In general, it is not an easy task to obtain a simplified ESOP from the given ESOP. Also, it is not easy to find a good transformation for a given ESOP (or a function). From here, we assume that truth tables of n-variable functions are given. In this case, the ESOPs are canonical and the numbers of non-zero outputs are at most p^n . We also assume that functions with fewer non-zero outputs are easier to minimize than ones with more non-zeros.

Definition 8 *An* optimal transformation *is one which minimizes the number of non-zero outputs in the truth table.*

The optimal transformation generates an ESOP which is relatively easy to minimize by heuristic AND-EXOR minimizers such as EXMIN [16].

4.2 Algorithm for p=2

For $p = 2$, there are 6 different transformations. But only three transformations need to be examined. Other three can be obtained by negating the variables of the first ones. The three transformations to be examined are:
identical , $X^{\{1\}} \leftrightarrow X^{\{0,1\}}$, and $X^{\{0\}} \leftrightarrow X^{\{0,1\}}$.
For a given function, generate 3^n different truth tables and find one with minimum non-zero outputs, and we can get the optimal L-transformation. To find such transformation is essentially same as the minimization of pseudo-Kronecker expansion [3, 9, 10]. Therefore, a similar algorithm can be used. It is clear that L-transformation will produce ESOPs with fewer products than the method shown in [1], since the latter considers only 2^n different combinations. The following algorithms find an optimal L-transformation in a reasonable computation time up to 14 variables by a SUN-4 workstation. The computation time is proportional to $n \cdot 3^n$, and memory requirement is proportional to 3^n.

Algorithm 1 (Extended truth table) *1. Let n be the number of inputs and m be the number of outputs, respectively. Let f be a truth table of 2^n elements, where each element is an m-bit vector. The indeces of f are represented by n-bit vectors, and they run from $(0,\cdots,0)$ to $(1,\cdots,1)$.*

2. Obtain the extended truth table g with 3^n elements as follows: The indeces of g are represented by n-trit vectors, and they run from $(0,\cdots,0)$ to $(2,\cdots,2)$.

(a) For the entries with index vectors consisting of either 0 or 1:

$$g(a_1,\cdots,a_n) \leftarrow f(a_1,\cdots,a_n),$$

where $a_k \in \{0,1\}$ $(k = 1,2,\cdots,n)$.

(b) The remaining $3^n - 2^n$ entries are obtained as follows: For the entries with the index vectors where only one digit, say i-th, is 2:
$g(a_1,\cdots,2,\cdots,a_n)$
$\leftarrow g(a_1,\cdots,0,\cdots,a_n) \oplus g(a_1,\cdots,1,\cdots,a_n),$
where $a_k \in \{0,1\} (k = 1,2,\cdots,n, k \neq i)$, and $\oplus$ denotes bitwise ring sum operation.

Table 2: Extended truth table and Extended weight table

	X_1	X_2	X_3	f_0	f_1	W_a	W_b	W_c	W_d
0	0	0	0	0	1	1	2	4	6
1	0	0	1	0	1	1	1	3	5
2	0	0	2	0	0	0	1	2	7
3	0	1	0	1	1	1	2	4	5
4	0	1	1	1	1	1	2	4	5
5	0	1	2	0	0	0	0	1	8
6	0	2	0	1	0	1	2	4	5
7	0	2	1	1	0	1	2	3	4
8	0	2	2	0	0	0	1	1	7
9	1	0	0	0	0	0	1	3	7
10	1	0	1	0	1	1	1	2	6
11	1	0	2	0	1	1	2	4	5
12	1	1	0	0	1	1	2	4	6
13	1	1	1	0	1	1	2	3	5
14	1	1	2	0	0	0	0	2	7
15	1	2	0	0	1	1	2	3	5
16	1	2	1	0	0	0	1	3	5
17	1	2	2	0	1	1	2	2	6
18	2	0	0	0	1	1	1	3	5
19	2	0	1	0	0	0	2	3	5
20	2	0	2	0	1	1	1	2	6
21	2	1	0	1	0	1	2	4	5
22	2	1	1	1	0	1	2	3	4
23	2	1	2	0	0	0	0	1	7
24	2	2	0	1	1	1	2	3	4
25	2	2	1	1	0	1	1	4	5
26	2	2	2	0	1	1	1	1	7

(c) *For the entries with the index vectors where two digits, say i-th and j-th, are 2:*

$$g(a_1,\cdots,\underbrace{2}_{i},\cdots,\underbrace{2}_{j},\cdots,a_n)$$

$$\leftarrow\ g(a_1,\cdots,\underbrace{0}_{i},\cdots,\underbrace{2}_{j},\cdots,a_n)$$

$$\oplus g(a_1,\cdots,\underbrace{1}_{i},\cdots,\underbrace{2}_{j},\cdots,a_n),$$

where $a_k \in \{0,1\}$ $(k = 1,2,\cdots,n, k \neq i, k \neq j)$.

(d) *For the entries with the index vectors where 3, 4,$\cdots$, n-th digits are 2: Similar way to the above.*

Example 8 *The derivation of the extended truth table by Algorithm 1 for the function in Fig.6(a) is illustrated in Table2.*

2.a) For the rows 0,1,3,4,9,10,12,and 13, the entries of (f_0, f_1) *in Table2 are the same as that of Fig.6(a).*

2.b) The entry for the row 2 is obtained by EXORing the entries of rows 0 and 1: $(0,1) \oplus (0,1) = (0,0)$. *The entries for the rows 5,11,14, etc., are obtained in a similar way.*

2.c) The entry for the row 8 is obtained by EXORing the entries of rows 2 and 5: $(0,0) \oplus (0,0) = (0,0)$. *The entries for the rows 17, etc., are obtained in a similar way.*

2.d) The entry for the row 26 is obtained by EXORing the entries of rows 8 and 17: $(0,0) \oplus (0,1) = (0,1)$.

Algorithm 2 (Extended weight table)

```
1. For $i$=0 to $3^n$ do
      {$w_1(i) \leftarrow 0$ if $g(i) = 0$ else $w_1(i) \leftarrow 1$ }

2. For $k$=1 to $n$ do
      { For $j$=0 to $3^{k-1} - 1$ do
         { For $i$=0 to $3^{n-k} - 1$ do
            {$i_0 \leftarrow 3^{n-k}(3j + 0) + i$;
             $i_1 \leftarrow 3^{n-k}(3j + 1) + i$;
             $i_2 \leftarrow 3^{n-k}(3j + 2) + i$;
             $w_2(i_0) \leftarrow w_1(i_0) + w_1(i_2)$;
             $w_2(i_1) \leftarrow w_1(i_1) + w_1(i_2)$;
             $w_2(i_2) \leftarrow w_1(i_0) + w_1(i_1)$ }
         }
      For $i$=0 to $3^n$ do {$w_1(i) \leftarrow w_2(i)$ }
      }
```

3. *Let* $(a_1,\cdots,a_n)$ *be the index vector with the minimum weight. Obtain the L-transformed function as follows: If* $a_i = 0$ *then replace* $f(a_1,\cdots,\underbrace{1}_{i},\cdots,a_n)$ *by* $f(a_1,\cdots,\underbrace{0}_{i},\cdots,a_n) \oplus f(a_1,\cdots,\underbrace{1}_{i},\cdots,a_n)$

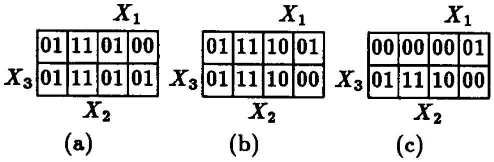
   If $a_i = 1$ *then replace* $f(a_1,\cdots,\underbrace{0}_{i},\cdots,a_n)$ *by* $f(a_1,\cdots,\underbrace{0}_{i},\cdots,a_n) \oplus f(a_1,\cdots,\underbrace{1}_{i},\cdots,a_n)$

   If $a_i = 2$ *then do not change.*

Example 9 *Table2 illustrates the derivation of the extended weight table for the function in Fig.6(a).*

1. For the column W_a: $W_a(i) \leftarrow 1$ *iff* (f_0, f_1) *of the i-th row is non-zero.*
   For the column W_b: $W_b(0) \leftarrow W_a(0) + W_a(18)$, $W_b(1) \leftarrow W_a(1) + W_a(19), \cdots$
   For the column W_c: $W_c(0) \leftarrow W_b(0) + W_b(6)$, $W_c(1) \leftarrow W_b(1) + W_b(7), \cdots$
   For the column W_d: $W_d(0) \leftarrow W_c(0) + W_c(2)$, $W_c(1) \leftarrow W_b(1) + W_b(2), \cdots$

2. *The minimum weight are 4 and they are in rows 7 and 22. If we use the row 7, the index vector is (0,2,1). For* x_1, *replace* $f(1, x_2, x_3)$ *with* $f(0, x_2, x_3) \oplus f(1, x_2, x_3)$ *Fig.6(b) shows the function after this operation.*
   For x_2, *do nothing.*
   For x_3, *replace* $f(x_1, x_2, 0)$ *with* $f(x_1, x_2, 0) \oplus f(x_1, x_2, 1)$
   Fig.6(c) shows the transformed function, which has only 4 non-zero outputs.



(a)

01	11	01	00
01	11	01	01

(b)

01	11	10	01
01	11	10	00

(c)

00	00	00	01
01	11	10	00

Figure 6: Example of L-transformation for p=2

4.3 Algorithm for p=4

For $p = 4$, there are 20160 different L-transformations. However, only $20160/(4!) = 840$ need to be examined among them. This number is still too large for exhaustive examination: in principle, we need to check up to $(840)^n$ different functions if we use the same approach as $p = 2$. So, for $p = 4$, we will give up the exhaustive method and consider only the subset of the transformations. For each variable, we find an L-transformation which maximally reduces the number of non-zero outputs. We iterate this procedure until no reduction is possible. This approach does not produce the optimal one for some functions, but requires computation time proportional to $n \cdot 4^n$. So, it is much faster than Algorithm 1 for the functions represented by the truth tables of the same size.

Algorithm 3 (Optimal Expansion for p=4)

1. *Expand the given function as*

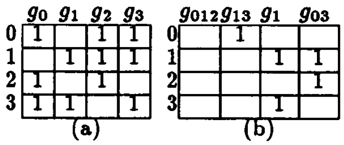
$$g = X^0 \cdot g_0 \oplus X^1 \cdot g_1 \oplus X^2 \cdot g_2 \oplus X^3 \cdot g_3.$$

2. *Compute the following functions:* $g_{01} = g_0 \oplus g_1$, $g_{02} = g_0 \oplus g_2$, $g_{03} = g_0 \oplus g_3$, $g_{12} = g_1 \oplus g_2$, $g_{13} = g_1 \oplus g_3$, $g_{23} = g_2 \oplus g_3$, $g_{012} = g_0 \oplus g_1 \oplus g_2$, $g_{013} = g_0 \oplus g_1 \oplus g_3$, $g_{023} = g_0 \oplus g_2 \oplus g_3$, *and* $g_{0123} = g_0 \oplus g_1 \oplus g_2 \oplus g_3$. *Also count the number of non-zero outputs of these functions* $|g_A|$, *where* $A \subseteq P$, *and* $P = \{0,1,2,3\}$. *Let* g_A, g_B, g_C, *and* g_D *be the functions which have smallest non-zero outputs, where* $A, B, C, D \subseteq P$. *Let* a, b, c *and* d *be the characteristic vectors of* A, B, C, *and* D, *respectively. Choose these sets so that the matrix* $M = \begin{bmatrix} \vec{a} \\ \vec{b} \\ \vec{c} \\ \vec{d} \end{bmatrix}$ *be regular.*

3. *Expand the function into*

$$g = X^{A'} \cdot g_A \oplus X^{B'} \cdot g_B \oplus X^{C'} \cdot g_C \oplus X^{D'} \cdot g_D,$$

where A', B', C' *and* D' *are subsets of* P *such that* $\vec{a}', \vec{b}', \vec{c}', \vec{d}'$ *are the characteristic vectors of them, and* $\begin{bmatrix} \vec{a}' \\ \vec{b}' \\ \vec{c}' \\ \vec{d}' \end{bmatrix} = (M^{-1})^t$.



(a)

	g_0	g_1	g_2	g_3
0	1		1	1
1		1	1	1
2	1		1	
3	1	1		1

(b)

	g_{012}	g_{13}	g_1	g_{03}
0		1		
1			1	1
2				1
3			1	

Figure 7: Example for Optimum Expansion for $p = 4$

Algorithm 4 (L-Transformation for p=4)

1. *Let* f *be the given function.* $g \leftarrow f$.
   $M_i \leftarrow I$ *(Unit matrix of* 4×4*)*$(i = 1, 2, ..., n)$.

2. *do 3 and 4 until the number of the non-zero outputs cannot be reduced.*

3. **For** *i=1,...,n* **do** *4.*

4. *Expand the function into*

$$g = X_i^0 \cdot g_{i_0} \oplus X_i^1 \cdot g_{i_1} \oplus X_i^2 \cdot g_{i_2} \oplus X_i^3 \cdot g_{i_3}.$$

   Obtain the optimal expansion by Algorithm 2:

$$g = X_i^{A'} \cdot g_A \oplus X_i^{B'} \cdot g_B \oplus X_i^{C'} \cdot g_C \oplus X_i^{D'} \cdot g_D,$$

   Let $g \leftarrow X_i^0 \cdot g_A \oplus X_i^1 \cdot g_B \oplus X_i^2 \cdot g_C \oplus X_i^3 \cdot g_D$, *and let*

$$M \leftarrow \begin{bmatrix} \vec{a} \\ \vec{b} \\ \vec{c} \\ \vec{d} \end{bmatrix},$$

   where $\vec{a}, \vec{b}, \vec{c}$, *and* $\vec{d}$ *are characteristic vectors of* A, B, C, *and* D, *respectively. Let* $M_i \leftarrow M \cdot M_i$. *Compute* $|g|$ *and check if the number of non-zeros are reduced.*

5. *Obtain an ESOP for* f *from* $M_i(i = 1, 2, ..., n)$.

Example 10 *Let's obtain an optimal expansion by using Algorithm4.*

1. *Suppose that the 4-valued 2-variable function*

$$g = X_1^0 \cdot g_0 \oplus X_1^1 \cdot g_1 \oplus X_1^2 \cdot g_2 \oplus X_1^3 \cdot g_3$$

   is given where g_0, g_1, g_2 *and* g_3 *are shown in Fig.7(a). Note that this function has 11 non-zero outputs.*

2. *Compute the functions as shown in Fig.8 . Also count the number of non-zero outputs in the functions. Fig.8 shows that* g_{012}*has the smallest number of non-zeros,* g_{13} *and* g_{023} *are the next. The characteristic vectors for* g_{012}, g_{13} *, and* g_{023} *, are (1,1,1,0), (0,1,0,1) and (1,0,1,1), respectively. Because they are linearly dependent, we choose* g_1 *instead of* g_{023} *to make the set of vectors linearly independent. Similarly, we choose* g_{03} *which has two*

	g_0	g_1	g_2	g_3	g_{01}	g_{02}	g_{03}	g_{12}	g_{13}	g_{23}	g_{012}	g_{013}	g_{023}	g_{123}	g_{0123}
0	1	0	1	1	1	0	0	1	1	0	0	0	1	0	1
1	0	1	1	1	1	1	1	0	0	0	0	0	0	1	1
2	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0
3	1	1	0	1	0	1	0	1	0	1	0	1	0	0	1
W	3	2	3	3	3	2	2	3	1	2	0	2	1	2	3

W:number of non-zeros

Figure 8: Example for optimum expansion for p=4

non-zeros. The transformation matrix, its inverse, and its transpose are

$$M = \begin{bmatrix}1110\\0101\\0100\\1001\end{bmatrix}, M^{-1} = \begin{bmatrix}0111\\0010\\1101\\0110\end{bmatrix}, (M^{-1})^t = \begin{bmatrix}0010\\1011\\1101\\1010\end{bmatrix}.$$

Therefore, $\vec{a'} = (0,0,1,0)$, $\vec{b'} = (1,0,1,1)$, $\vec{c'} = (1,1,0,1)$, *and* $\vec{d'} = (1,0,1,0)$, *and we have* $A' = \{2\}$, $B' = \{0,2,3\}$, $C' = \{0,1,3\}$, *and* $D' = \{0,2\}$.

3. *The optimal expansion is*

$$X^{\{2\}} \cdot g_{012} \oplus X^{\{0,2,3\}} \cdot g_{13} \oplus X^{\{0,1,3\}} \cdot g_1 \oplus X^{\{0,2\}} \cdot g_{03}.$$

The transformed function is

$$X^{\{0\}} \cdot g_{012} \oplus X^{\{1\}} \cdot g_{13} \oplus X^{\{2\}} \cdot g_1 \oplus X^{\{3\}} \cdot g_{03}.$$

Fig.7(b) shows the map of the transformed function. Note that it has only 5 non-zero outputs.

5 Experimental Results

To investigate the ability of L-transformation, we developed transformation programs, and simplified arithmetic functions. These programs treat multiple output functions. They are coded in FORTRAN, and run on SUN-4 work stations.

5.1 Benchmark Functions

Table 3 shows 8 arithmetic functions used in this experiment, which also appeared in Table 1. All the function have 8 inputs and multiple outputs, and originally have 255 or 225 product terms.

ADR4 is a 4 bit adder:$X + Y$; MLP4 is a 4 bit multiplier:$X \times Y$; LOG8 is a 8 bit logarithm circuit:$\log_2(X + 1)$; NRM4 calculates the distance of two 4-bit numbers: $SQRT(X^2 + Y^2)$; RDM8 is an 8 bit random number generator:$(5X + 1)$ mod 256; ROT8 is an 8-bit root circuit: $SQRT(X)$; SQR8 is an 8-bit square circuit:X^2; WGT8 counts the number of 1's in X and represents it in a binary number.

5.2 Transformations for p=2

For each function, we checked 3^8 different truth tables. Table3 also shows the number of non-zero outputs after transformation. The computation time is about 240 milliseconds for each function. MIN shows the number of products in near minimal ESOP obtained by EXMIN [16, 2]. For example, ADR4 is an 8 input 5 output function and originally has 255 products. Algorithm 1 found a 34 product solution, but EXMIN found a 31 product solution.

Table 3: Number of Products for Arithmetic circuits

INPUT DATA				$p = 2$		$p = 4$	
NAME	IN	OU	PT	TRNS	MIN	TRNS	MIN
ADR4	8	5	255	34	31	76	11
LOG8	8	8	255	171	96	167	94
MLP4	8	8	225	97	61	77	52
NRM4	8	5	255	157	73	118	58
RDM8	8	8	255	56	31	37	27
ROT8	8	5	255	83	35	53	28
SQR8	8	16	255	168	114	255	112
WGT8	8	4	255	107	54	62	25

IN : # of inputs OU : # of outputs

PT : # of non-zeros in the original truth table

TRNS : # of products after transformation

MIN : # of products after AND-EXOR minimization

5.3 Transformations for p=4

For each function, near optimal pairs of the input variables were obtained by a heuristic method [17] and a function with p=4 was generated. For each variable, Algorithm 3 checked 15 different literals to find a near optimal L-transformation. Table 3 also shows the number of non-zero outputs after L-transformation. The computation time is about 200 milliseconds for each function. For example, Algorithm4 found a 76 product solution for ADR4, but EXMIN obtained a 11 product solution. As for SQR8, it could not found better transformation than the canonical one.

6 Conclusion and Comments

In this paper, we presented the L-transformation for p-valued input two-valued output functions. The number of products in MESOPs is invariant under this transformation. Also, we showed that the L-transformation produce different functions with different minimization properties.

We defined the optimal L-transformation be one which transforms a given function into another one with minimum non-zeros. For p=2, we proposed an algorithm to find an optimum L-transformation. For p=4, we proposed a heuristic algorithm to find a near optimum L-transformation. By using these algorithms, we simplified eight arithmetic functions.

The first application of L-transformation is to obtain initial solutions for near minimum ESOPs. When the function is given as a truth table, both algorithms quickly reduce the number of products in ESOPs, although they cannot obtain the minimum. After this, EXMIN can be used to obtain the near minimum solutions. We could reduce about 20% of computation time for the functions shown in Table 3 by using this method.

The second application is to obtain very high quality solutions. Because L-transformations produces different functions with different minimization properties, we can obtain very high quality solution by generating equivalent functions iteratively. First, minimize the given ESOP by a heuristic minimizer such as EXMIN until the number of products cannot be reduced any more. Then, transform the ESOP to obtain an L-equivalent function. Thirdly, minimize the new ESOP again by EXMIN to obtain a better solution. Continue this procedure while we can improve the solution. A non-deterministic algorithm has been developed [2] where L-transformations are generated pseudo-randomly. This algorithm obtained better solutions than any other method for the functions in Table 3.

The third application is for parallel processing. Suppose that we have k processors. First, generate k functions which are L-equivalent to the given function. Then minimize each ESOP independently by using the parallel processors. Lastly, identify the ESOP with the minimum number of products, and obtain the ESOP for the original function.

The fourth application is the classification of logic functions. The L-transformations produce an RP equivalence classes . For p=2 and n=5, the number of the representative functions is 6936. We simplified all the representative functions, and verified that 9 products are sufficient to realize any functions of 5 variables. A straightforward investigation needs $2^{32} \approx 4.3 \times 10^9$ minimization. By using more sophisticated method, we also prove that 16 products are sufficient to realize any function of 6 variables[8].

The fifth application is to prove the minimality of the given ESOP. We found a method to derive the lower bounds on the number of products in MESOPs. By using this method, we proved the minimality for about 30 percents of 5 variable functions [15].

Acknowledgements

This work was supported in part by Grant in Aid for Scientific Research of the Ministry of Education, Science and and Culture of Japan. I thank Prof. Jon T.Butler for his kind help during my stay in Monterey. Mr.N.Koda and Dr. D. Brand provided me very useful comments. Professors P.Besslich, M.Davio, M.R. Mukerjee and J.Muzio sent me their latest papers.

References

[1] P.W. Besslich. Efficient computer method for ExOR logic design. *IEE Proc.*, 130:203–206, 1983. Part E.

[2] D. Brand and T. Sasao. On the minimization of AND-EXOR expressions. *IEICE Technical Report*, VLD 90-88, Dec. 1990.

[3] M. Davio, J-P Deschamps, and A.Thayse. *Discrete and Switching Functions.* McGraw-Hill International, 1978.

[4] S. Even, I. Kohavi, and A. Paz. On minimal modulo-2 sum of products for switching functions. *IEEE Trans. on Comput.*, 16:671–674, Oct. 1967.

[5] H. Fleisher, M. Tarvel, and J. Yeager. A computer algorithm for minimizing Reed-Muller canonical forms. *IEEE Trans. on Comput.*, C-36(2):247–250, Feb. 1987.

[6] M. Helliwel and M. Perkowski. A fast algorithm to minimize multi-output mixed-polarity generalized Reed-Muller forms. *Proc. Design Automation Conference*, 427–432, June 1988.

[7] N. Koda and T. Sasao. Four-variable AND-EXOR minimum expressions and their properties. *IEICE Technical Report*, FTS89(25):, Oct. 1989.

[8] N. Koda and T. Sasao. On the number of product terms of AND-EXOR minimum expressions. *IEICE Technical Report*, FTS 91:, July 1991. (in Japanese, to appear).

[9] P.K. Lui and J. Muzio. Boolean matrix transforms for the parity spectrum and the minimization of modulo-2 canoncial expansions. *University of Victoria*, (DCS-135-IR):, July 1990.

[10] M.R. Mukerjee. Minimization of ring-sum expansion of mixed polarity. *AMSE Sympo. on Modelling and Simulation*, Oct. 1990.

[11] A. Mukhopadhyay and G. Schmitz. Minimization of exclusive OR and logical equivalence of switching circuits. *IEEE Trans. on Comput.*, C-19(2):132–140, Feb. 1970.

[12] M. Perkowski, M. Helliwell, and P. Wu. Minimization of multiple-valued input multi-output mixed-radix exclusive sum of products for incompletely specified boolean functions. *Proc. Int. Sympo. on Multiple-Valued Logic*, 256–263, May 1989.

[13] J.P. Robinson and Chia-Lung Yeh. A method for modulo-2 minimization. *IEEE Trans. on Comput.*, C-31(8):800–801, Aug. 1982.

[14] K.K. Saluja and E.H. Ong. Minimization of Reed-Muller canonic expansion. *IEEE Trans. on Comput.*, C-28(7):535–537, July 1979.

[15] T. Sasao. Exclusive-or sum-of-products expressions: their properties and a minimization algorithm. *IEICE Technical Report*, VLD 90(87), Dec. 1990.

[16] T. Sasao. EXMIN: A simplification algorithm for exclusive-or- sum-of-products expressions for multiple-valued input two-valued output functions. *Proc. Int. Sympo. on Multiple-Valued Logic*, 128–135, May 1990.

[17] T. Sasao. Input variable assignment and output phase optimization of PLA's. *IEEE Trans. on Comput.*, C-33(10):879–894, Oct. 1984.

[18] T. Sasao and P.W. Besslich. On the complexity of MOD-2 sum PLA's. *IEEE Trans. on Comput.*, C-32(2):262–266, Feb. 1990.