Scan 5618
5619

Butler
1978

2 scrp

Tandem Networks of Universal Cells

JON T. BUTLER, MEMBER, IEEE

Abstract—**The tandem network, a row-like interconnection of cells is described. Such networks are more general than the widely studied irredundant cascade, but less general than a universal network based on, for example, the Shannon decomposition. The analysis is facilitated by the introduction of three cell-network interconnections, whose functions are characterized by certain attributes of the partition matrix of the realized function. Partition matrices have been used to significant advantage in characterizing disjunctive decompositions. They are also important in the analysis of nondisjunctive decompositions, but their application is cumbersome in such cases. It is shown that certain nondisjunctive decompositions can be handled easily as operations on partition matrices. A counting technique is developed which shows that the number of functions realized by tandem networks is significantly larger than that realized by cascades. In addition, a synthesis technique is shown for constructing a tandem network to realize a given function.**

Index Terms—**Cascades, disjunctive decomposition, disjunctive networks, networks of flexible cells, nondisjunctive decompositions, tandem networks.**

I. Introduction

During the past fifteen years considerable effort has been devoted to the study of combinational networks of flexible cells. The cascade or TRIB [1], [2] has been widely studied, and within the past several years, the more general tree or disjunctive net [3]–[6] has been examined. Except for a few instances, however (e.g., [7]), little work has been done on other cell interconnections. Studies of more general networks are important because it has been shown [8], [12]–[14] that the fraction of functions realized by irredundant cascades and by irredundant disjunctive nets is very small even for moderate sizes (4 or more inputs). Even for redundant cascades, cascades in which a single input can drive more than one cell, the fraction of realizable functions approaches 0 as the number of inputs grows arbitrarily [9].

In this paper, properties of a new interconnection, the tandem net, are presented. The tandem network is a special case of the redundant disjunctive network, but is more general than the irredundant cascade. In particular, it is shown that a tandem network realizes significantly more functions than a similar cascade. The increase in function count comes at the expense of approximately three times as many cells. The analysis and synthesis of tandem networks is based on the properties of functions realized by three cell-network interconnections, discussed in the following sections.

II. Cell-Network Interconnection #1—N_1

Consider the $x_1 \cdots x_{k-1} | x_k$ partition matrix[1] of a function $f(x_1, x_2, \cdots, x_k)$, as shown in Fig. 1, where C_0 and C_1 represent the $x_k = 0$ and $x_k = 1$ columns, respectively.[2] Let 0 and 1 represent the columns of all 0's and all 1's, respectively. Let X denote a column with at least one 0 and at least one 1, and $\bar{X}$ its complement. The concatenation of two columns will represent a complete partition matrix. Thus, for example, 01 represents $f(x_1, x_2, \cdots, x_k) = x_k$ and XX represents a function independent of x_k, but dependent on at least one of the remaining variables.

For the cell-network interconnections to be analyzed, it is convenient to classify realized functions according to Table I. Note that if $f(x_1, x_2, \cdots, x_k)$ is realized by a network of universal cells, so also is the complement function $\bar{f}(x_1, x_2, \cdots, x_k)$ (add an inverter to the output). Thus, if a Type D function, say $0X$, is realized, so also is $1\bar{X}$. Furthermore, if $f(x_1, x_2, \cdots, x_k)$ is realized, so also is $f(x_1, x_2, \cdots, \bar{x}_k)$ (add an inverter to input x_k). Thus, if $0X$ is realized, so also is $X0$. As a result, Type D functions occur as a group of four, $0X$, $X0$, $1\bar{X}$, and $\bar{X}1$. That is, if one member of the group is realized by some network N, all members are realized by N. Note that the "dual" group of four $0\bar{X}$, $\bar{X}0$, $1X$, and $X1$ may or may not be realized. For the analysis to follow, it is convenient to subdivide Type D functions into two groups, Type $D1$ and $D2$. If $0X$, $X0$, $1\bar{X}$, $\bar{X}1$, is realized but the dual group is not, the four realized functions are said to be *Type D1*. If $0X$, $X0$, $1\bar{X}$, $\bar{X}1$, $0\bar{X}$, $\bar{X}0$, $1X$, and $X1$ are all realized, the eight functions are said to be *Type D2*.

Type E functions have the form XY, where $X \neq Y$ and $X \neq \bar{Y}$, and are divided into two groups. In particular, if XY, $\bar{X}\bar{Y}$, YX, and $\bar{Y}\bar{X}$ only are realized, each of the four is classified as *Type E1*. On the other hand, if XY, $X\bar{Y}$, $\bar{X}Y$, $\bar{X}\bar{Y}$, YX, $Y\bar{X}$, $\bar{Y}X$, and $\bar{Y}\bar{X}$ are realized, they are all *Type E2* functions.

Lemma 4 of [5] shows that the trivial functions 00 ($f = 0$) and 11 ($f = 1$) are realized by all nets, as are functions on a single variable, 01 ($f = x_k$) and 10 ($f = \bar{x}_k$). Thus, the four Type A functions are realized by all nets of universal cells.

Manuscript received December 10, 1975; revised February 18, 1977 and November 15, 1977.

The author is with the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL 60201.

[1] For a discussion of partition matrices see Curtis [10].

[2] Alternatively one could consider the Shannon decomposition about x_k, $f(x_1, x_2, \cdots, x_k) = \bar{x}_k f(x_1, x_2, \cdots, 0) + x_k f(x_1, x_2, \cdots, 1)$, but the partition matrix is used for ease of representation.

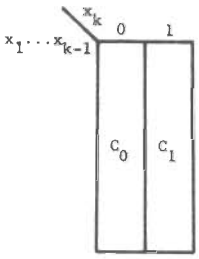

Fig. 1. The $x_1 \cdots x_{k-1} | x_k$ partition matrix.

TABLE I
CLASSIFICATION OF FUNCTIONS

Type	Form of Partition Matrix
A	00, 01, 10, 11
B	XX and $\bar{X}\bar{X}$
C	$X\bar{X}$ and $\bar{X}X$
D	$0X$, $X0$, $1X$, $X1$
$D1$	$0X$, $X0$, $1\bar{X}$, and $\bar{X}1$ are realized but not $0\bar{X}$, $\bar{X}0$, $1X$, and $X1$
$D2$	$0X$, $X0$, $1\bar{X}$, $\bar{X}1$, $0\bar{X}$, $\bar{X}0$, $1X$, and $X1$ are all realized
E	XY, $X \neq Y$ and $X \neq \bar{Y}$
$E1$	XY and $\bar{X}\bar{Y}$ are realized but not $X\bar{Y}$ and $\bar{X}Y$
$E2$	XY, $X\bar{Y}$, $X\bar{Y}$, and $\bar{X}Y$ are all realized

For the remaining functions, however, net structure determines how many of each type are realized.

The first cell-network interconnection to be considered is shown in Fig. 2. The functions realized by N_1, the overall network, can be viewed as the set of functions realized by M operated on by the set of functions realized by Cell 1. For example, if M realizes XY, a Type E function, and Cell 1 realizes $\bar{a}x_k$, N_1 realizes $0\bar{Y}$, a Type D function. The operation, in this case, is to make the $x_k = 0$ column all 0's and to complement the $x_k = 1$ column. In all, the sixteen functions realized by Cell 1 correspond to every combination of the operations shown in Table II on the two columns in the partition matrix. Since there are 2 columns, there are $2^4 = 16$ different combinations of operations, each corresponding to a function realized by Cell 1.

Consider by type the functions realized by the overall network N_1. All Type A functions are, of course, realized. Type B and C functions are realized only when M realizes a function corresponding to operations 3 and 4 exclusively. Furthermore, Type B and C functions can be viewed as arising from Type B functions only. To illustrate this, suppose M realizes $X\bar{X}$, a Type C function, and Cell 1 realizes the coincident product, $a \odot x_k$.[3] Thus, N_1 realizes $\bar{X}\bar{X}$. However, from Lemma 3 of [5], if a network of universal cells realizes $f(x_1, x_2, \cdots, x_k)$, it also realizes $f(x_1, x_2, \cdots, 0)$. Thus, M must also realize XX. But for this case, if Cell 1 realizes $\bar{a}$, N_1 also realizes $\bar{X}\bar{X}$.

The number of Type B functions realized by M is $m - 2$, where m is the number of functions independent of x_k. Each function, in turn, gives rise, under operations 3 and 4, to four

[3] The coincident product $a \odot b$ is given as,

$$a \odot b = ab + \bar{a}\bar{b}.$$

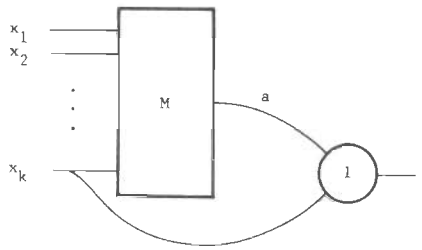

Fig. 2. Cell-network interconnection $\#1(N_1)$.

TABLE II
OPERATIONS ON COLUMNS IN THE PARTITION MATRIX

1	Set all entries in the column to 0
2	Set all entries in the column to 1
3	Complement the column
4	Leave the column unchanged

functions XX, $\bar{X}\bar{X}$, $X\bar{X}$, and $\bar{X}X$. However, $\bar{X}\bar{X}$ will produce this set of four, and by Lemma 1 of [5], M also realizes $\bar{X}\bar{X}$. Thus, there are a total of $2(m-2)$ functions of the form XX, $\bar{X}\bar{X}$, $X\bar{X}$, and $\bar{X}X$. Of these, one half or $m-2$ are Type B and $m-2$ are Type C.

Type D functions can arise in a number of ways, all of which require that M realize a function with at least one nontrivial column. By an argument similar to that for Type B and C functions, it is seen that the Type D functions $0X$, $X0$, $1X$, and $X1$ can be generated by Operations 1 and 2, where XX is a Type B function realized by M. Thus, there are $4(m-2)$ Type D functions. These are all Type $D2$ because $\bar{X}\bar{X}$, which is also realized by N, produces $0\bar{X}$, $\bar{X}0$, $1\bar{X}$, and $\bar{X}1$. These have already been included in the count $4(m-2)$, since the $m-2$ Type B functions realized by M include both XX and $\bar{X}\bar{X}$.

A Type E function can only be realized by cell-network interconnection $\#1$ when M realizes a Type E function. If M realized a Type $E2$ functions, the operations of Cell 1 produce no new Type E functions. However, if M realizes a Type $E1$ function XY (and $\bar{X}\bar{Y}$), then operations 3 and 4 produce the four functions XY, $\bar{X}\bar{Y}$, $X\bar{Y}$, and $\bar{X}Y$. These operations, in effect, convert the Type $E1$ functions realized by M into Type $E2$ functions.

Thus, we have the following result.

Theorem 1: The functions realized by cell-network interconnection $\#1$, N_1, are as follows.

Type A: 00, 01, 10, 11
Type B: XX, $\bar{X}\bar{X}$
Type C: $X\bar{X}$, $\bar{X}X$
Type $D1$: None
Type $D2$: $0X$, $X0$, $1X$, $X1$
Type $E1$: None
Type $E2$: XY, $X\bar{Y}$, $\bar{X}Y$, $\bar{X}\bar{Y}$

where XX and XY are Type B and E functions, respectively, realized by M.

Note that functions realized by N_1 which are independent of x_k are precisely the functions realized by M which are independent of x_k. This observation and Theorem 1 yield the following.

Lemma 1: Let r'_i and r_i be the number of Type i functions realized by cell-network interconnection #1, N_1, and M (the subnetwork), respectively. Let m' and m be the number of functions realized by N_1 and M, respectively, which are independent of x_k. Then

$$r'_A = 4 \tag{1}$$

$$r'_B = m - 2 \tag{2}$$

$$r'_C = m - 2 = r'_B \tag{3}$$

$$r'_{D1} = 0 \tag{4}$$

$$r'_{D2} = 4(m-2) = 4r'_B \tag{5}$$

$$r'_{E1} = 0 \tag{6}$$

$$r'_{E2} = 2r_{E1} + r_{E2}, \tag{7}$$

and the total number of functions realized by N_1, is

$$n' = 2r_{E1} + r_{E2} + 6m - 8. \tag{8}$$

Furthermore,

$$m' = m. \tag{9}$$

This result shows that cell-network interconnection #1 realizes, in general, more functions than the subnetwork M. An exception is the case where M is also a cell-network interconnection #1. In this interconnection, the two universal cells combine to form a two-input one-output network. But all the functions realized by this are realized by a single universal cell.

It is interesting to compare the results of Theorem 1 to the case where x_k of Fig. 1 connects only to Cell 1. From [5], the number of functions n'' of such a network is

$$n'' = 6m - 8$$

comparing this with (8) shows that the additional functions realized by cell-network interconnection #1 stem from Type E functions realized by M.

III. Cell-Network Interconnection #2—N_2

The cell-network interconnection to be considered in this section is shown in Fig. 3. As with cell-network interconnection #1, it is useful to view the functions realized by the overall network, N_2, as the result of operations on functions realized by subnetwork, M. The $x_{k-1} \cdots x_1 \mid x_k$ partition matrix of a function realized by M is shown in Fig. 4. Since x_k is not a primary input to M, the $x_k = 0$ and $x_k = 1$ columns are identical.

The operations involved are those listed in Table II. In particular, Cell 1 determines which pair (of the 16 possible) of operations are involved and Cell 2 determines how they are distributed over the four columns in Fig. 4. As an example, if Cell 1 realizes the exclusive OR function, then operations 3 and 4 of Table II are involved. If Cell 2 realizes $x_{k-1}x_k$, then the three columns corresponding to $x_{k-1}x_k = 00$, 01, 10 remain unchanged while the column $x_{k-1}x_k = 11$ is complemented. Thus, if Fig. 4 is the matrix realized by M, the overall network realizes

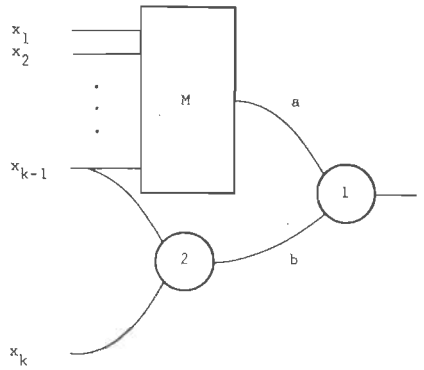

Fig. 3. Cell-network interconnection #2(N_2).

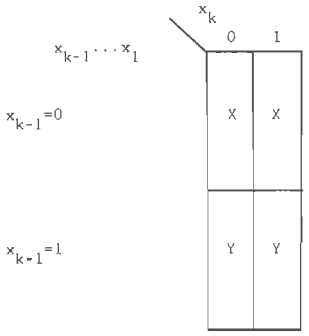

Fig. 4. Partition matrix of a function realized by M.

$$\begin{matrix} XX \\ Y\bar{Y} \end{matrix}$$

Consider by type the functions realized by N_2.

Type B Functions

Type B functions occur when Cell 2 realizes a function independent of x_k, while M and Cell 1 range over their realizable functions. Since Cell 1 is universal, the four functions independent of x_k realized by Cell 2, 0, 1, x_k, and $\bar{x}_k$, add no more functions than if Cell 2 realized just x_k, and the functions realized by N_2 which are independent of x_k are precisely those realized by the cell-network interconnection #1. Of these, two $\begin{smallmatrix}0&0\\0&0\end{smallmatrix}$ and $\begin{smallmatrix}1&1\\1&1\end{smallmatrix}$, are Type A. The remaining are Type B. Thus, from Lemma 1,

$$r'_B = 2r_{E1} + r_{E2} + 6m - 10.$$

Furthermore, m', the number of functions realized by cell-network interconnection 2 which are independent of x_k, is

$$m' = r'_B + 2.$$

Type C Functions

Type C functions are realized in a variety of ways. Table III shows the combinations of functions realized by M which produce distinct Type C functions. All such functions occur when Cell 1 realizes the exclusive OR or coincident product function. Note that Type $D2$ functions realized by M produce twice as many Type C functions per source function as do Type $D1$ functions, because the operations, in effect,

TABLE III
TYPE C FUNCTIONS REALIZED BY CELL-NETWORK INTERCONNECTION #2

Entry Number	M Realizes	Number of Functions Realized by M	Form of Type C Functions Realized by N_2	Number Realized
1	$\begin{matrix}XX\\XX\\XX\end{matrix}$ (Type B)	$m-2$	$\begin{matrix}X\bar{X} & X\bar{X}\\X\bar{X} & \bar{X}X\\X\bar{X} & \bar{X}X\end{matrix}$	$2(m-2)$
2	$\begin{matrix}11\\XX\end{matrix}$ (Type $D1$)	r_{D1}	$\begin{matrix}10 & 10\\X\bar{X}\end{matrix}$	$2r_{D1}$
3	$\begin{matrix}11\\XX\end{matrix}$ (Type $D2$)	r_{D2}	$\begin{matrix}10\\X\bar{X} & X\bar{X}\end{matrix}$	r_{D2}
4	$\begin{matrix}YY\\XX\end{matrix}$ (Type $E1$)	r_{E1}	$\begin{matrix}Y\bar{Y} & \bar{Y}Y\\X\bar{X}\end{matrix}$	$2r_{E1}$
5	$\begin{matrix}YY\\11\end{matrix}$ (Type $E2$)	r_{E2}	$\begin{matrix}Y\bar{Y}\\01 & 10\end{matrix}$	r_{E2}
6	00 (Type A)	2	10 01	2

TABLE IV
TYPE D FUNCTIONS REALIZED BY CELL-NETWORK INTERCONNECTION #2

Entry Number	Function Realized by M[a]	Cell 1 Realizes	Form of Type D Realized	Number of Type D Functions Produced	Type Realized
1	$\begin{matrix}XX\\XX\\XX\end{matrix}$ (Type B)	$a \cdot b$	$\begin{matrix}X0 & 00 & 0X & 00\\00 & X0 & 00 & 0X\\X0 & 00 & 0X & 00\end{matrix}$	$4(m-2)-2r_{D1}-r_{D2}$	$D1$
2	$\begin{matrix}XX\\XX\\XX\end{matrix}$ (Type B)	$a \cdot b$	$\begin{matrix}00 & X0 & 00 & 0X\\X1 & 11 & 1X & 11\end{matrix}$	$2r_{D1}+r_{D2}$	$D2$
3	$\begin{matrix}XX\\XX\\XX\end{matrix}$ (Type B)	$a+b$	$\begin{matrix}11 & X1 & 11 & 1X\\X1 & 11 & 1X & 11\end{matrix}$	$4(m-2)-2r_{D1}-r_{D2}$	$D1$
4	$\begin{matrix}XX\\XX\\XX\end{matrix}$ (Type B)	$a+b$	$\begin{matrix}11 & X1 & 11 & 1X\\X0 & 0X\end{matrix}$	$2r_{D1}+r_{D2}$	$D2$
5	$\begin{matrix}XX\\XX\\XX\end{matrix}$ (Type B)	$a \cdot b$	$\begin{matrix}X0 & 0X\\X1 & 1X\end{matrix}$	$2(m-2)$	$D2$
6	$\begin{matrix}XX\\01 & 10\end{matrix}$ (Type B)	$a+b$	$\begin{matrix}X1 & 1X\\00 & 00 & 01 & 10\end{matrix}$	$2(m-2)$	$D2$
7	01 10 (Type A)	$a \cdot b, a+b$	$\begin{matrix}01 & 10 & 00 & 00\\10 & 01 & 11 & 11\\11 & 11 & 10 & 01\end{matrix}$	8	$D2$
8	$\begin{matrix}XX & 00\\00 & XX\\XX & 11\\11 & XX\\XX\end{matrix}$ (Type D)	$a \cdot b, a+b$	$\begin{matrix}X1 & 1X & 01 & 10\\01 & 10 & X1 & 1X\\X0 & 0X & 10 & 01\\10 & 01 & X0 & 0X\\X0 & 0X & X1 & 1X\end{matrix}$	$2r_{D1}+2r_{D2}$	$D2$
9	$\begin{matrix}YY\\XX\end{matrix}$ (Type E)	$a \cdot b, a+b$	$\begin{matrix}Y0 & 0Y & Y1 & 1Y\\X0 & 0X & X1 & 1X\end{matrix}$	$4r_{E1}+4r_{E2}$	$D2$
10	$\bar{X}\bar{X}$ (Type C)	$a \cdot b, a+b$	$\bar{X}0\ 0\bar{X}\ \bar{X}1\ 1\bar{X}$	$4r_C$	$D2$

[a] The function types specified for the functions realized by M are with respect to x_{k-1}. Thus, $\begin{matrix}XX\\11\end{matrix}$ is a Type D function with respect to x_{k-1} (but a Type B function with respect to x_k).

convert Type $D1$ functions to $D2$. A similar statement is true of Type E functions, also. Totaling the functions in Table III yields

$$r'_C = 2r_{D1} + r_{D2} + 2r_{E1} + r_{E2} + 2m - 2.$$

Type D Functions

Type D functions are also tabulated by examining the combinations of functions types realized by M and operations on the partition matrices of such functions. Table IV shows the combinations which result in Type D functions. The number of functions realized by type is

$$r'_{D1} = -4r_{D1} - 2r_{D2} + 8(m-2)$$
$$r'_{D2} = 4r_C + 6r_{D1} + 4r_{D2} + 4r_{E1} + 4r_{E2} + 4m.$$

Type E Functions

The combinations of functions realized by subnetwork M and connecting cells which produce Type E functions are shown in Table V.

The total number of functions realized is

$$r'_{E1} = 12r_C + 4r_{D1} + 12r_{E1} + 12r_{E2} + 12(m-2)$$
$$r'_{E2} = 4r_{D1} + 6r_{D2} + 4r_{E1} + 2r_{E2} + 4(m-2).$$

TABLE V
TYPE E FUNCTIONS REALIZED BY CELL-NETWORK INTERCONNECTION #2

Entry Number	Function Realized by M	Cell 1 Realizes	Form of Type E Function	Number of Type E Functions Realized	Type Realized
1	XX XX (Type B) XX	$a \cdot b$ $a + b$	$0X$ $X0$ $0X$ $X0$ XX XX $X0$ $0X$ XX XX $X0$ $0X$ $1X$ $X1$ $1X$ $X1$ XX XX $X1$ $1X$ XX XX $X1$ $1X$	$12(m-2)$	$E1$
2	XX XX (Type B) XX	$a \oplus b$	XX $\bar{X}X$ $X\bar{X}$ $X\bar{X}$ $\bar{X}X$ XX XX $X\bar{X}$ $0X$ $X0$ $0X$ $X0$ XX XX	$4(m-2)$	$E2$
3	$\bar{X}\bar{X}$ (Type C)	$a \cdot b$ $a + b$	$\bar{X}0$ $0\bar{X}$ $\bar{X}\bar{X}$ $\bar{X}\bar{X}$ $\bar{X}0$ $0\bar{X}$ $1X$ $X1$ $1X$ $X1$ XX XX $\bar{X}1$ $1\bar{X}$ $\bar{X}\bar{X}$ $\bar{X}\bar{X}$ $\bar{X}1$ $1\bar{X}$	$12r_C$	$E1$
4	XX 11 (Type $D1$) YY 00 (Type $D1$)	$a \cdot b$ $a + b$	$0X$ $X0$ $X0$ $0X$ 11 11 01 10 $1Y$ $Y1$ $Y1$ $1Y$ 00 00 10 01	$4r_{D1}$	$E1$
5	XX 11 (Type $D2$)[a] YY 00 (Type $D2$)	$a \cdot b$ $a + b$	$0X$ $X0$ $X0$ $0X$ 11 11 01 10 $1Y$ $Y1$ $Y1$ $1Y$ 00 00 10 01	$4r_{D2}$	$E2$
6	XX 00 (Type D)[a]	$a \oplus b$	XX XX $\bar{X}X$ $X\bar{X}$ 01 10 00 00 $\bar{X}\bar{X}$ $\bar{X}\bar{X}$ $X\bar{X}$ $X\bar{X}$ 10 01 11 11	$4r_{D1} + 2r_{D2}$	$E2$
7	XX YY (Type E)	$a \oplus b$	XX XX $\bar{X}X$ $X\bar{X}$ $Y\bar{Y}$ $\bar{Y}Y$ YY YY $\bar{X}\bar{X}$ $\bar{X}\bar{X}$ $X\bar{X}$ $\bar{X}X$ $\bar{Y}Y$ $Y\bar{Y}$ $\bar{Y}\bar{Y}$ $\bar{Y}\bar{Y}$	$4r_{E1} + 2r_{E2}$	$E2$
8	XX YY (Type E)	$a \cdot b$ $a + b$	$0X$ $X0$ $0X$ $X0$ XX XX $Y0$ $0Y$ YY YY $Y0$ $0Y$ $1X$ $X1$ $1X$ $X1$ XX XX $Y1$ $1Y$ YY YY $Y1$ $1Y$	$12(r_{E1} + r_{E2})$	$E1$

[a] The functions which are produced by N_2 from Type D functions realized by M of the form $\begin{smallmatrix}11\\XX\end{smallmatrix}$ and $\begin{smallmatrix}00\\XX\end{smallmatrix}$ are not shown for the sake of compactness.

Collating the results derived so far yields the following.

Lemma 2: The number of functions realized by N_2, cell-network interconnection #2 by type is

$$r'_A = 4 \tag{10}$$

$$r'_B = 2r_{E1} + r_{E2} + 6m - 10 \tag{11}$$

$$r'_C = 2r_{D1} + r_{D2} + 2r_{E1} + r_{E2} + 2m - 2 \tag{12}$$

$$r'_{D1} = -4r_{D1} - 2r_{D2} + 8m - 16 \tag{13}$$

$$r'_{D2} = 4r_C + 6r_{D1} + 4r_{D2} + 4r_{E1} + 4r_{E2} + 4m \tag{14}$$

$$r'_{E1} = 12r_C + 4r_{D1} + 12r_{E1} + 12r_{E2} + 12m - 24 \tag{15}$$

$$r'_{E2} = 4r_{D1} + 6r_{D2} + 4r_{E1} + 2r_{E2} + 4m - 8 \tag{16}$$

for a total n' of

$$n' = 16r_C + 12r_{D1} + 9r_{D2} + 24r_{E1} + 20r_{E2} + 36m - 56 \tag{17}$$

functions. Also,

$$m' = 2r_{E1} + r_{E2} + 6m - 8 \tag{18}$$

where r'_i and r_i are the number of Type i functions realized by N_2 and M, respectively, and where m' and m are the number of functions realized by N_2 and M, respectively, which are independent of x_{k-1} and x_k, respectively.

IV. ANALYSIS OF FUNCTIONS REALIZED BY TANDEM NETWORKS

Fig. 5 shows cell-network interconnection #3 which is composed of a subnetwork M and three cells. This network is a combination of cell-network interconnections #1 and #2 and the number of functions realized can be calculated easily by substituting (15), (16), and (18) into (1)–(9). Thus, the following is derived.

Lemma 3: The number of functions realized by N_3, cell-network interconnection #3 by type is

$$r'_A = 4 \tag{19}$$

$$r'_B = m' - 2 \tag{20}$$

$$r'_C = r'_B \tag{21}$$

$$r'_{D1} = 0 \tag{22}$$

$$r'_{D2} = 4r'_B \tag{23}$$

$$r'_{E1} = 0 \tag{24}$$

$$r'_{E2} = 24r_C + 12r_{D1} + 6r_{D2} + 28r_{E1} + 26r_{E2} + 28m - 56 \tag{25}$$

for a total n' of

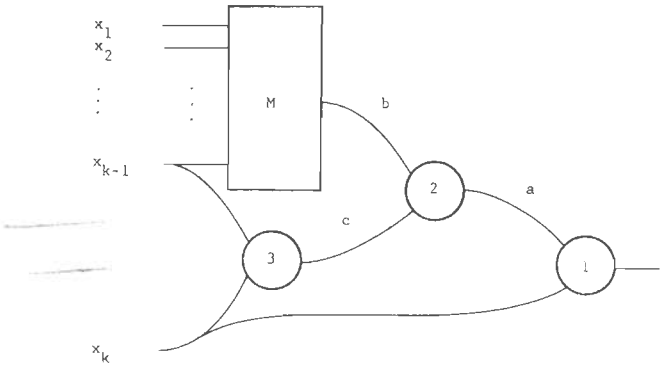

Fig. 5. Cell-network interconnection $\#3(N_3)$.

$$n' = 24r_C + 12r_{D1} + 6r_{D2} + 40r_{E1} + 32r_{E2} + 64m - 112 \tag{26}$$

functions. Also,

$$m' = 2r_{E1} + r_{E2} + 6m - 8 \tag{27}$$

where r_i' and r_i are the number of Type i functions realized by N_3 and M, respectively, and where m' and m are the number of functions realized by N_3 and M, respectively, which are independent of x_k and x_{k-1}, respectively.

The tandem network, a row-like interconnection of universal cells, has as a basic unit, the three-cell structure of cell-network interconnection $\#3$. In particular, a k-input tandem network, as shown in Fig. 6, has $k-2$ three-cell units connected iteratively to a single universal cell. Lemmas 2 and 3 can be used to derive an expression for the number of functions realized by tandem networks as follows.

Theorem 2: The number of functions $r_i(k)$ of Type i realized by a k-input tandem network is

$$r_A(k) = 4 \tag{28}$$

$$r_B(k) = r_C(k) = m(k) - 2 \tag{29}$$

$$r_{D1}(k) = 0 \tag{30}$$

$$r_{D2}(k) = 4m(k) - 8 \tag{31}$$

$$r_{E1}(k) = 0 \tag{32}$$

$$r_{E2}(k) = C_1(\alpha_1 - 6)\alpha_1^k + C_2(\alpha_2 - 6)\alpha_2^k + \tfrac{152}{49} \tag{33}$$

for a total $n_{\tan}(k)$ of

$$n_{\tan}(k) = C_1\alpha_1^{k+1} + C_2\alpha_2^{k+1} + \tfrac{48}{49} \tag{34}$$

where $m(k)$, the number of functions realized by a k-input tandem net which are independent of x_k is

$$m(k) = n_{\tan}(k-1) \tag{35}$$

and where

$$C_1 = \frac{-1202 + 363\sqrt{11}}{1960\sqrt{11}}, \qquad C_2 = \frac{1202 + 363\sqrt{11}}{1960\sqrt{11}}$$

$$\alpha_1 = 16 + 4\sqrt{11}, \qquad \alpha_2 = 16 - 4\sqrt{11}.$$

Proof: For a two-input tandem net (a single universal cell) we have

$$r_A(2) = 4, \quad r_B(2) = 2, \quad r_C(2) = 2,$$
$$r_{D1}(2) = 0, \quad r_{D2}(2) = 8, \quad r_{E1}(2) = 0,$$
$$r_{E2}(2) = 0, \quad m(2) = 4.$$

The values for other k can be found by the successive application of (19)–(25) and (27) to the initial values given by above. Since for both sets of equations, $r_B(k) = m(k) - 2$ and $r_{E1}(k) = 0$, two simultaneous independent recurrence equations result. These are

$$r_{E2}(k) = 76m(k-1) + 26r_{E2}(k-1) - 152 \tag{36}$$

$$m(k) = 6m(k-1) + r_{E2}(k-1) - 8. \tag{37}$$

Solving (36) and (37) for their closed form solutions with the initial conditions of $r_{E2}(2) = 0$ and $m(2) = 4$ yields, upon substitution into (19)–(27), the desired result. Q.E.D.

Table VI shows the number of functions, $n_{\tan}(k)$, realized by k-input tandem nets as given by (34). Also shown is the number of functions, $n_{\text{cas}}(k)$, realized by a k-input cascade $n_{\text{cas}}(k) = (2 \cdot 6^k + 8)/5$ (from [1]). As the table entries show, tandem nets realize significantly more functions than cascades. In fact,

$$\lim_{k\to\infty} \frac{n_{\text{cas}}(k)}{n_{\tan}(k)} = 0.$$

Because a k-input cascade is imbedded in a k-input tandem net, the functions realized by the latter include all those realized by the former. The additional functions are produced at a cost of approximately three times the number of cells.

It is of interest to consider the number of k-variable functions $N_{\tan}(k)$ which are realized by at least one tandem network. Note that $n_{\tan}(k)$ is the number of functions realized by a single tandem network of k variables. An upper bound for $N_{\tan}(k)$ is

$$N_{\tan}(k) < k!\, n_{\tan}(k),$$

since a tandem network with a specific assignment of labels to the inputs realizes $n_{\tan}(k)$ functions and there are $k!$ ways to make such an assignment. This is a strict upper bound because the AND of all variables is realized by all $k!$ networks. Comparing the upper bound with the total number of functions on k variables $N_k = 2^{2^k}$ as k becomes arbitrarily large yields

$$\lim_{k\to\infty} \frac{N_{\tan}(k)}{N_k} = 0.$$

With respect to three variables, Table VI shows that 240 functions are realized by a single cascade. One then wonders about the remaining 16 functions. All of these, it turns out, are realized by other tandem networks. Thus, all three-input functions are tandem network realizable.

TABLE VI
NUMBER OF FUNCTIONS REALIZED BY A CASCADES AND TANDEM NETS OF k-INPUTS

k	$n_{cas}(k)$	$n_{tan}(k)$
2	16	16
3	88	240
4	520	6,448
5	3,112	187,184
6	18,664	5,474,096
7	111,976	160,196,400
8	671,848	4,688,357,168
9	4,031,080	137,211,717,424
10	24,186,472	4,015,706,384,176

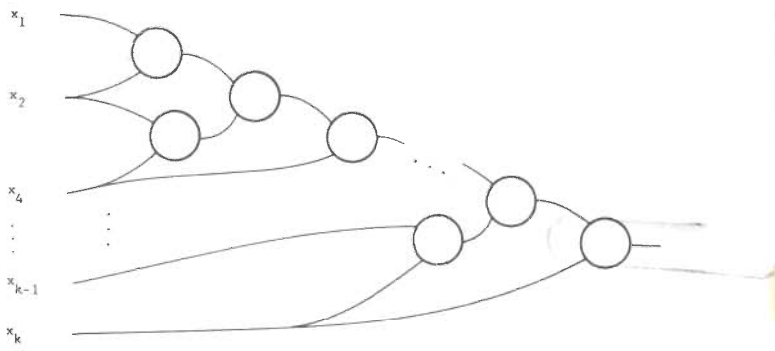

Fig. 6. A k-input tandem net.

V. SYNTHESIS OF TANDEM NETWORKS

The synthesis problem is stated as follows. Given a k variable function $f(X)$, where $X = \{x_1, x_2, \cdots, x_k\}$, assign functions to each cell in a k-input tandem network so that $f(X)$ is realized. Part of the problem in developing an economical algorithm is that a typical tandem function is realized in many ways. For example, the AND of k variables can be realized by a k-input tandem net in which all cells realize the two-input AND. It can also be realized by a cascade of AND gates, which is a tandem network where certain cells realize trivial functions. It is convenient, therefore, to introduce the *canonical tandem network*, which has the property that any function realized by an arbitrary tandem network is realized by a canonical network. As a result, it is only necessary to synthesize the canonical network. In the discussion to follow the canonical form is specified by six conditions.

Cells in a tandem network N are classified as follows:

Type 1: Both inputs connect to network inputs,

Type 2: One input only connects to a network input, and

Type 3: Both inputs connect to the outputs of other cells.

Let C, the *cascade* of N, denote the set of Type 2 and 3 cells. The structure of a tandem network N imposes a precedence on network inputs. That is, x_i *precedes* x_j if x_i labels an input which is higher than x_j. For example, in Fig. 6, x_1 precedes x_2, x_2 precedes x_3, etc.

Since each cell in the network can realize any of the 16 functions on two variables, trivial functions are possible. For example, Fig. 7(a) shows a network in which several cells realize trivial functions. In such cases, it will be convenient to eliminate redundant cells. Fig. 7(b) shows the network of Fig. 7(a) with redundant cells removed. In the networks to be considered from now on all cells will realize functions dependent on both inputs. The term gate will be used to denote such cells.

There are three basic two-input one-output functions, the AND, OR, and exclusive OR. All other functions on two variables can be realized from these by adding inverters to appropriate inputs. For the networks to be considered, it is important that only AND, OR, and exclusive OR gates occur in the cascade. If the output of an AND or OR gate in the cascade connects to the input of another gate in the cascade through an inverter, the inverter can be removed by applying de Morgan's theorem. For example, Fig. 7(c) shows the network of Fig. 7(b) with all such inverters removed. When exclusive OR gates are involved, an inverter at the output can be transferred to one of the inputs. Thus, a tandem network with only AND, OR, and exclusive OR gates in its cascade is no less general than an unrestricted tandem network. We have the following.

Condition 1: The cascade of a canonical tandem network consists only of AND, OR, and exclusive OR gates.

Because of this condition, the cascade of a canonical network will consist of a string of AND, OR, or exclusive OR gates, followed by a string of gates of a different type, etc. The synthesis algorithm will test at each step for strings of each gate type.

Cascades of AND *or* OR *Gates Whose Inputs are Network Inputs Only*

Let $f(X) = f(x_1, x_2, \cdots, x_k)$ denote a function dependent on k variables. x_i *0-masks* x_j if $f(X \mid 0 \rightarrow x_i)$[4] is independent of x_j. Similarly, x_i *1-masks* x_j if $f(X \mid 1 \rightarrow x_i)$ is independent of x_j. x_i is said to *mask* x_j if it 0-masks or 1-masks x_j.[5] Note that x_i masks itself.

The masking relation can be extended to sets of variables in a natural way. Let $Z \subseteq X$. Then x_i 0(1)-masks Z if x_i 0(1)-masks x_j, for all $x_j \in Z$. Let $Z_d \subseteq Z$, the *dominating set* of Z, denote the set $\{x_i\}$, where $x_i \in Z$ and x_i masks Z. As an example, consider the function realized by the networks in Fig. 7

$$f(x_1, x_2, x_3, x_4, x_5) = (x_1 + x_3)\bar{x}_2 x_5 + x_4 \bar{x}_5 + \bar{x}_4 x_5.$$

Here,

x_1 0-masks $\{x_1\}$ and 1-masks $Z_1 = \{x_1, x_3\}$,

x_2 0-masks $\{x_2\}$ and 1-masks $Z_2 = \{x_1, x_2, x_3\}$,

x_3 0-masks $\{x_3\}$ and 1-masks $Z_1 = \{x_1, x_3\}$,

x_4 0-masks $\{x_4\}$ and 1-masks $\{x_4\}$,

and x_5 0-masks $Z_3 = \{x_1, x_2, x_3, x_5\}$ and 1-masks $\{x_5\}$.

The dominating sets of Z_1, Z_2, and Z_3 are respectively $Z_{d1} = Z_1$, $Z_{d2} = \{x_2\}$, and $Z_{d3} = \{x_5\}$. It is interesting to

[4] $f(X \mid a \rightarrow x_i)$ denotes the function $f(X)$ with all occurrences of x_i replaced by a.

[5] The masking relation is the same as that used to synthesize fanout-free networks of AND's, OR's, and inverters [11].

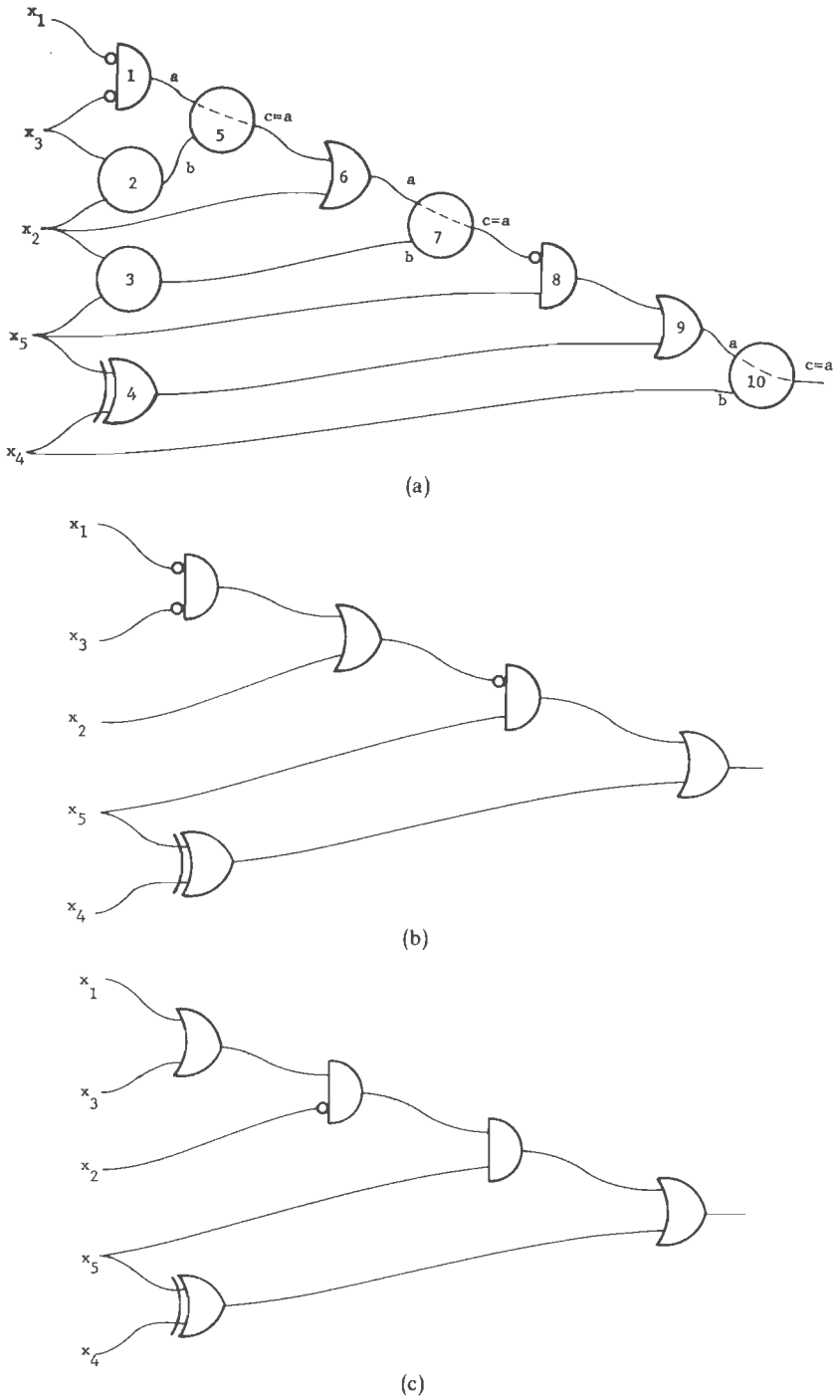

Fig. 7. (a)–(c) Three tandem networks which realize the same function.

note in Fig. 7(c) that an input to an AND or OR gate in the cascade masks itself and at least one other input. This observation indicates a method by which certain inputs to a tandem network can be identified.

Lemma 4: If x_i connects to a Type 2 gate which realizes a^*b^* or $a^* + b^*$, then x_i need not connect to a Type 1 gate which connects to any input preceding x_i, where a and b are the Type 2 gate inputs and $a^* = a$ or $\bar{a}$.

Proof: The proof is the same as that for a similar statement about redundant cascades [2, lemma 3, p. 854].

Q.E.D.

With respect to Fig. 7(a), Lemma 4 shows that Cells 2 and 3 can be removed, regardless of their function. (They were removed originally by inspection. Each connected to input b of a cell whose output was independent of b.) Cells 5 and 7 can also be removed. In the general case of such Type 3 cells, removal may require an inverter to be combined with an adjacent cell if the network function is to remain the same. Lemma 4 does not apply to Type 1 cells such as Cell 4 in Fig. 7(a).

Condition 2: In a canonical tandem network, if input x_i connects to a Type 2 cell (with inputs a and x_i) which realizes ax_i^* or $a + x_i^*$, it does not connect to a Type 1 cell whose other input is driven by a variable which precedes x_i.

Theorem 3 below shows that the masking relation can be used to extract a string of Type 2 AND or OR gates whose primary inputs satisfy Condition 2.

Theorem 3: $f(X)$ has the decomposition

$$f(X) = g(X - Z_d)x_{i1}^* x_{i2}^* \cdots x_{im}^* \quad (= g(X - Z_d) + x_{i1}^* + x_{i2}^* + \cdots + x_{im}^*), \quad (38)$$

for $x_{ij} \in Z_d$, where g is tandem realizable if and only if

1) $f(X)$ is tandem realizable,
2) X is a masked set with dominating set Z_d, and
3) $f(X) = 0(1)$ for all instances of masking.

Proof: (if) Expand $f(X)$ in the Shannon canonical form about $x_{i1}, x_{i2}, \cdots$, and x_{im}

$$\begin{aligned} f(X) = {} & f(X \mid 11 \cdots 1 \to x_{i1}x_{i2} \cdots x_{im}) \\ & \cdot x_{i1}x_{i2} \cdots x_{im} + f(X \mid 11 \cdots 0 \\ & \to x_{i1}x_{i2} \cdots x_{im})x_{i1}x_{i2} \cdots \bar{x}_{im} + \cdots \\ & + f(X \mid 00 \cdots 0 \to x_{i1}x_{i2} \cdots x_{im}) \\ & \cdot \bar{x}_{i1}\bar{x}_{i2} \cdots \bar{x}_{im}. \end{aligned} \quad (39)$$

Assume without loss of generality that all elements in Z_d 0-mask X (the case where some inputs 1-mask while others 0-mask follows in like manner).

If $f(X) = 0(1)$ in all instances of masking, then (39) reduces to

$$\begin{aligned} f(X) = {} & f(X \mid 11 \cdots 1 \to x_{i1}x_{i2} \cdots x_{im}) \\ & \cdot x_{i1}x_{i2} \cdots x_{im}(f(X \mid 11 \cdots 1 \to x_{i1}x_{i2} \cdots x_{im}) \\ & + \bar{x}_{i1} + \bar{x}_{i2} + \cdots + \bar{x}_{im}). \end{aligned} \quad (40)$$

Since $f(X)$ is tandem realizable, it is realized by a canonical tandem network whose decomposition follows directly from (40). Thus,

$$f(X \mid 11 \cdots 1 \to x_{i1}x_{i2} \cdots x_{im})$$

is tandem realizable.

(only if) The three conditions of the hypothesis follow directly from (38). Q.E.D.

It follows that one step in the synthesis algorithm is to test for a set Z_d of inputs which mask all of X. If $f(X) = 0$ or 1 in all instances of masking, then a string of AND or OR gates, respectively, can be extracted. The residue function $g(X - Z_d)$ must then be tested for tandem realizability.

Cascades of AND *or* OR *Gates Whose Inputs Come From Type 1 Gates*

Although Theorem 3 shows how to extract a string of Type 2 AND or OR gates from the cascade of a tandem net, it does not apply to a string of Type 3 gates. For example, in Fig. 7(c), the output gate composes a Type 3 gate string of length one. As a result, x_4 and x_5 do not mask all of the remaining inputs. However, when $x_4x_5 = 00$ or 11, the output is independent of all variables. This leads to the following definition.

x_ix_j *masks* x_k under $f(X)$, if there exists at least one assignment of values to x_i and x_j such that $f(X)$ is independent of x_k. x_ix_j *masks* $Y \subseteq X$ if there exists assignments of values to x_i and x_j such that x_ix_j masks x_k for all $x_k \in Y$. As an example, for the function realized by the networks of Fig. 7, x_4x_5 mask $\{x_1, x_2, x_3, x_4, x_5\}$. Note that if x_i masks Y, then x_ix_j mask Y, where x_j is another variable in X. Therefore, in the discussion to follow, *it will be assumed that if* x_ix_j *mask* Y *neither* x_i *nor* x_j *alone mask* Y.

Because it facilitates the synthesis algorithm it will be assumed that:

Condition 3: Any inverter in the lead between the input of a gate in the cascade of a canonical tandem network and the output of a Type 1 gate g is part of g. Furthermore, if g is an AND or OR gate the inverter is "absorbed" using de Morgan's theorem.

A further restriction on the form of the canonical network is obtained by the following observation. Fig. 8(a) shows part of tandem network N in which a Type 1 AND gate g_1 connects to an AND gate g_2 which is part of the cascade of N. Together, g_1 and g_2 realize a three input AND function. Fig. 8(b) shows a rearrangement of the AND gates which does not change the network function. Condition 2 applies, and so g_3 can be eliminated resulting in the network of Fig. 8(c). In general,

Lemma 5: Consider a Type 3 AND(OR) gate g in a cascade of a tandem network. Then g need not connect to a Type 1 AND(OR) gate unless that gate connects to primary inputs of lowest precedence.

Lemma 5 eliminates the consideration of certain structures and allows:

Condition 4: No Type 3 AND(OR) gate in a canonical tandem network connects to the output of a Type 1 AND(OR) gate g', except perhaps when g' connects to two inputs of lowest precedence.

Although Lemma 5 eliminates from consideration certain configurations, there remains the problem of identifying a cascade of AND or OR gates driven by Type 1 gates only. Theorem 4 addresses this problem.

Theorem 4: Consider a canonical tandem network whose output is derived from a cascade of Type 3 AND or a cascade of Type 3 OR gates, which receives inputs from Type 1 gates only. Let Y be the set of inputs to these Type 1 gates and $f(X)$ the network function realized. Then, all $x_i \in Y$ are involved in a pair x_ix_j which masks X, for some $x_j \in Y$. Furthermore, all instances of masking yield $f(X) = 0$ (AND) or $f(X) = 1$ (OR).

Proof: Consider $x_i \in Y$. x_i connects to at least one Type 1 gate g whose other input is x_j. If g connects to an AND gate then all values of x_i and x_j which map to 0 cause $f(X) = 0$. Thus, x_i and x_j mask X. The case for OR gates is identical except $f(X) = 1$. Q.E.D.

The synthesis algorithm for this case can be seen as follows. Fig. 9(a) shows a tandem network which realizes a function $f(X)$ in which X is masked by a number of pairs. Fig. 9(b) shows a graph in which vertices are labeled by variables in X, and where an edge connects x_i and x_j if x_ix_j masks X. For example, an edge appears between x_6 and x_{12} because these are applied to OR gate g_6 which in turn drives an AND gate in the cascade. An edge also appears between x_4

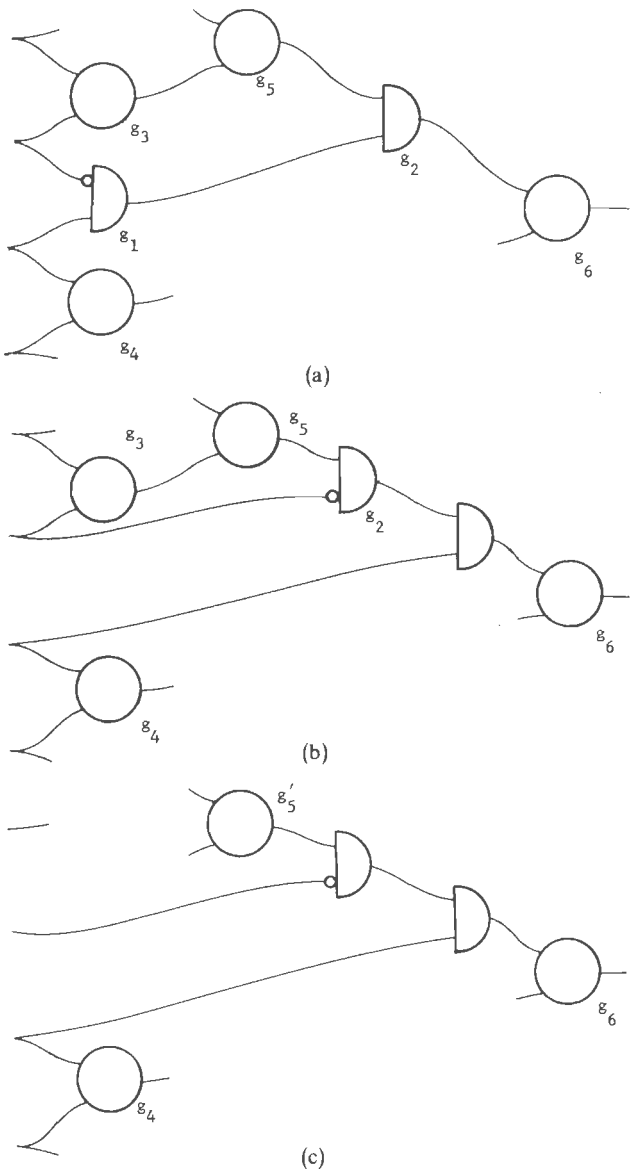

Fig. 8. (a)–(c) Eliminating a Type 1 AND gate.

and x_{12}, for example, because when $x_4 x_{12} = 10$, no choice of x_6 will cause both g_5 and g_6 to produce a 1. Thus, $f(X) = 0$ when $x_4 x_{12} = 10$, and so $x_4 x_{12}$ mask X.

With respect to the set of inputs $Y = X - \{x_5, x_{13}\}$ which are applied to Type 1 gates whose outputs are applied to the cascade of AND gates, edges in Fig. 9(b) indicate possible pairs of inputs which are applied to the same gate. It is necessary, therefore, to determine which pairs correspond to a gate such as $x_6 x_{12}$ and which do not, such as $x_4 x_{12}$.

The problem of determining which inputs are applied to exclusive OR gates and their complement, coincident product gates, can be handled by considering symmetry among variables. For example, in Fig. 9(a), x_1, x_2, and x_4 are symmetric. In particular, the output q is 0 for all assignments of values to $x_1 x_2 x_4$ except 000 and 111, since for such assignments at least one of the outputs of g_3 and g_4 is 0. Since q is "sensitized" to the output of g_1 only when all of $x_1 x_2 x_4$ are 0 or all are 1, it makes no difference which of x_1, x_2, or x_4, the input to g_2 connects. Similarly, the function realized is unchanged if the connection to g_5 is moved from x_4 to x_1 or x_2. If the network is to be a tandem network, however, the inputs to g_2 and g_5 cannot be applied to the same gate. Any permutation of the labels x_1, x_2, and x_4 within themselves leaves the network function unchanged.

Note also that if x_6 is replaced by $\bar{x}_6$, the four inputs, x_1, x_2, x_4, and $\bar{x}_6$, are symmetric. In general, a set of variables will be symmetric, to within a complementation of a variable, if they are applied to a set of adjacent exclusive OR and coincident product gates which share inputs with neighboring gates. Tests for symmetry among variables are quite straightforward. For example, the test can proceed algebraically or in an easily applied tabular method [15].

Thus, a step in the synthesis algorithm is to test for symmetry of variables in a prospective function $f(X)$. In particular, only variables which are part of the same subgraph need be tested. For example, in Fig. 9(b), x_2 and x_3 need not be considered since they can never be applied to the same gate in a tandem network. Let Y_i be a set of variables such that $x_j, x_k \in Y$ if x_k is symmetric to x_k or $\bar{x}_k$. Then, Y is divided into two disjoint subsets of symmetric variables Y_{i1} and Y_{i2} such that for any $x_j \in Y_1$, and $x_j \in Y_2$, x_j is symmetric to $\bar{x}_k$. Transform the given function $f(X)$ into a function $f'(X)$ in which each occurrence of x_j such that $x_j \in Y_{i2}$ is complemented. Note that the new function is tandem realizable if and only if the original is. In the example of Fig. 9(a), this operation amounts to placing an inverter at x_6 or x_1, x_2, and x_4 (depending on how Y_{i2} is chosen).

Formally,

Theorem 5: For some $f(X, Y)$, let

$$g(X, x_i) = f(X, Y \mid x_i \to x_j \; \forall x_j \in Y), \tag{41}$$

where $x_i \in Y$. Then $f(X, Y)$ has the decomposition

$$f(X, Y) = g(X, x_i) S^m_{0,m}(Y)(g(X, x_i) + \bar{S}^m_{0,m}(Y)),^6$$

if and only if

$$\begin{aligned} f(X, Y) &= g(X, 0) && \text{if } x_j = 0 \; \forall x_j \in Y \\ f(X, Y) &= g(X, 1) && \text{if } x_j = 1 \; \forall x_j \in Y \text{ or} \\ f(X, Y) &= 0(1) && \text{otherwise.} \end{aligned} \tag{42}$$

Proof: (For the first case only. The other follows in like manner.) (iff) Express $f(X)$ in its Shannon canonical form:

$$f(X, Y) = \bar{x}_{i1} \bar{x}_{i2} \cdots \bar{x}_{im} g(X, 0) + x_{i1} x_{i2} \cdots x_{im} g(X, 1), \tag{43}$$

for $x_{ij} \in Y$. Without loss of generality assume $x_i = x_{i1}$. ORing the right side of (43) with $\bar{x}_{i1} \bar{x}_{i2} \cdots \bar{x}_{im} g(X, 1) x_{i1} (= 0)$ and $x_{i1} x_{i2} \cdots x_{im} g(X, 0) \bar{x}_{i1} (= 0)$, collecting terms and rearranging yields

$$f(X, Y) = g(X, x_{i1}) S^m_{0,m}(Y) = g(X, x_i) S^m_{0,m}(Y).$$

(only iff) Equation (42) follows directly from (43). Q.E.D.

The application of Theorem 5 to tandem networks is based on the fact that $S^m_{0,m}(Y)(\bar{S}^m_{0,m}(Y))$ is realized by a

[6] $S^m_{0,m}(Y)$, where $m = |Y|$, the symmetric function on variable set Y, is 1 if and only if no variables in Y are 1 (0 subscripts) or if all m variables are 1 (1 subscript).

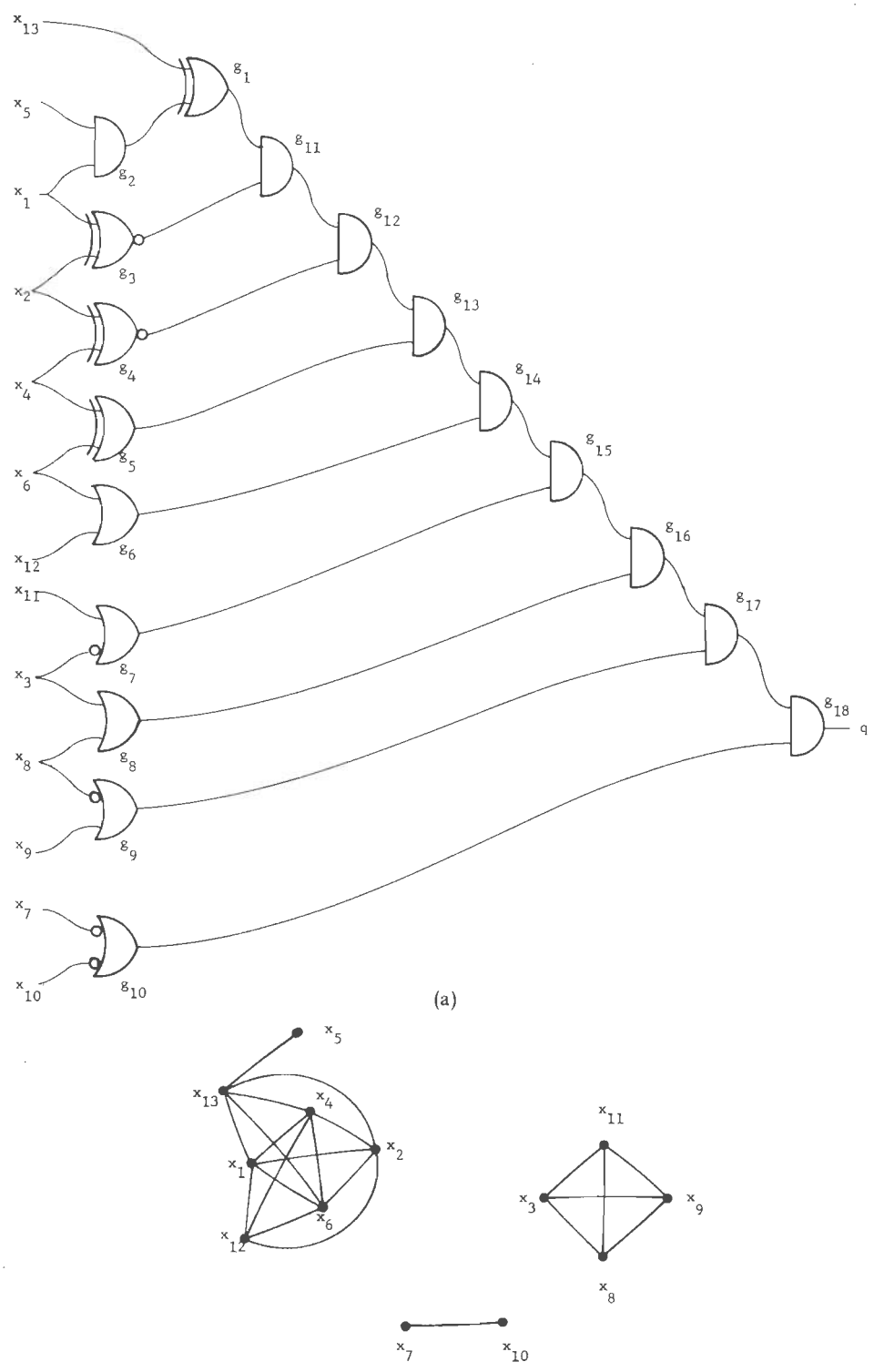

Fig. 9. (a) and (b) Example of a tandem network with a cascade of AND gates whose inputs are the outputs of Type 1 gates.

cascade of AND(OR) gates whose inputs come from coincident product (exclusive OR) gates.

Lemma 6: Let $f(X, Y)$ have the decomposition

$$f(X, Y) = g(X, x_i)S^m_{0,m}(Y)(g(X, x_i) + \bar{S}^m_{0,m}(Y)) \quad (44)$$

where $g(X, x_i)$ is given as (41) and $m = |Y|$. Then $f(X, Y)$ is tandem realizable if and only if $g(X, x_i)$ is realized by a canonical tandem net in which x_i is or can be applied to a Type 1 gate whose output is applied to a cascade of AND(OR) gates which produces the network output.

Theorem 5 and Lemma 6 form the basis for a test of exclusive OR or coincident product gates. When such gates are removed, only AND or OR gates remain. For example, if this is done to the function realized by the net of Fig. 9(a), the network of Fig. 10(a) is obtained. Here, x_1, x_2, x_4, and $\bar{x}_6$

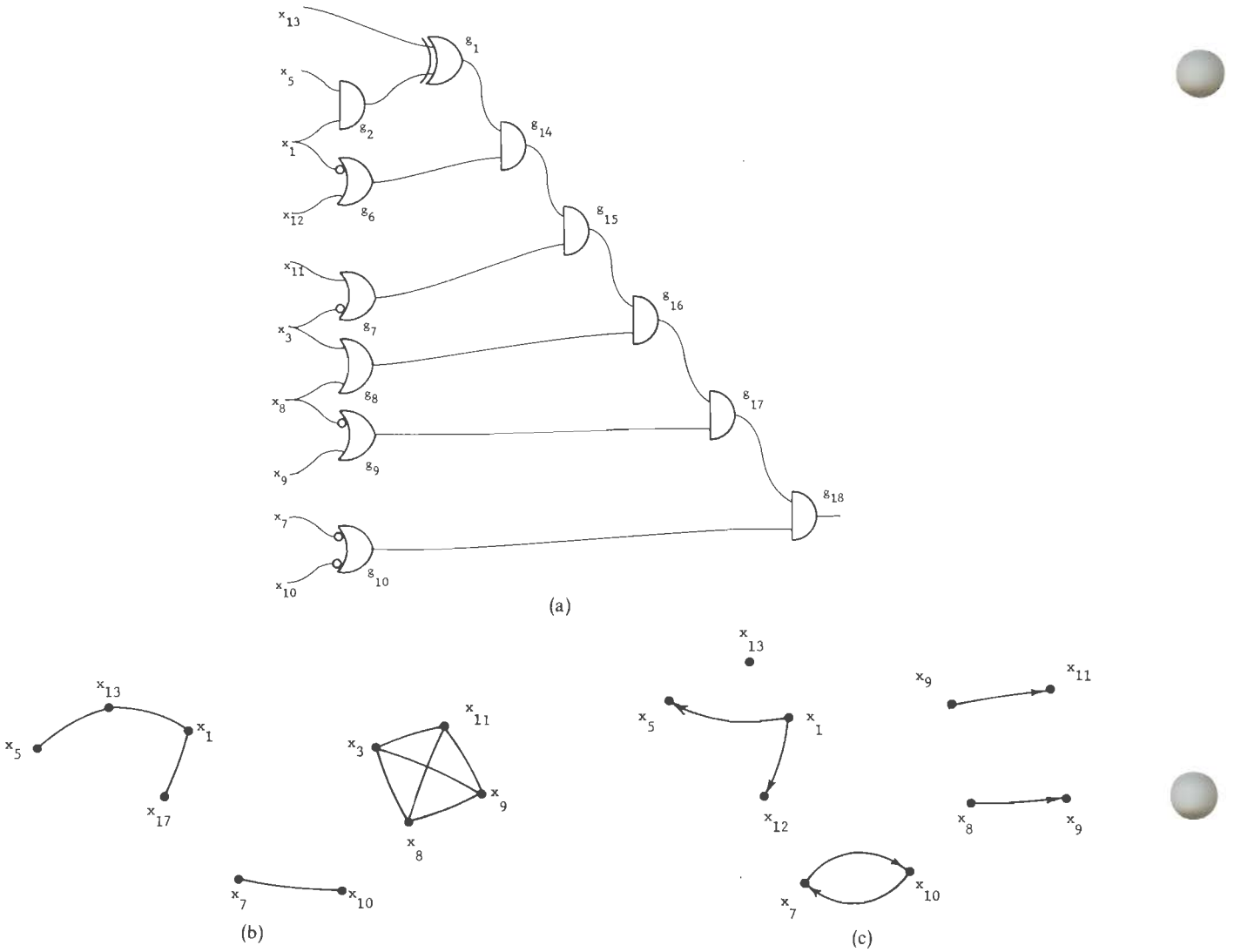

Fig. 10. (a)-(c) Network of Fig. 9 with exclusive OR and coincident product gates removed.

have been replaced by x_1. The inverter on the input of g_6 is the result of complementing x_6.

It is necessary now to determine which edges in the pair masking graph correspond to Type 1 AND or OR gates. This problem can be handled by finding variables which are masked by one other variable. Fig. 10(c) shows this relationship for the function realized by the network of Fig. 10(a). That is, if x_i masks x_j, a directed arc connects x_i to x_j. Notice that inputs which are applied to only one gate g are masked by exactly one other input (the other input to g). In general,

Lemma 7: Let $f(X)$ be a function with the following properties.

a) $x_i x_j$ masks X and in all instances of masking $f(X) = 0(1)$.

b) x_j masks x_i.

Then,

$$f(X) = g(X - x_i)h(x_i, x_j)(g(X - x_i) + h(x_i, x_j)),$$

where $f(X)$ is tandem realizable if and only if $g(x - x_i)$ is, for

$$h(x_i, x_j) = x_i^* + x_j^*(x_i^* x_j^*).$$

Lemma 7 shows that for a prospective function, an OR or AND gate can be extracted leaving a function of one fewer variable which is tandem realizable if and only if the original one is. The function $h(x_i, x_j)$ is determined by the values of x_i and x_j which mask $f(X)$. For example, in the function realized by the net of Fig. 10(a), $f(X) = 0$ when x_8 and x_9 (two inputs which satisfy the criteria of Lemma 7) are 1 and 0, respectively. Thus, $h(x_8, x_9) = \bar{x}_8 + x_9$.

Applying Lemma 7 repeatedly will remove all Type 1 OR gates of Fig. 10(a). At the end of this stage of the synthesis algorithm, it is necessary to specify that the input, x_1, of the network left to be synthesized is to be preceded by all others. If such a requirement is imposed on this section of the tandem net by the extraction of a previous cascade, it is necessary to verify that the input under consideration can indeed be preceded by all others. For example, if x_8 of Fig.

10(a) must be preceded by all other inputs, one must conclude the function is not realizable since there is no way to place x_8 at the bottom without affecting the "tandemness" of the network. Of the inputs, only x_7, x_9, x_{10}, and x_{11} can be precedee by all others.

Cascades of Exclusive OR *Gates*

In this section, tandem networks whose output is a cascade of exclusive OR gates are considered. A result similar to Lemma 5 for AND(OR) cascades is also true of exclusive OR gates.

Lemma 8: If x_i and x_j are applied to a Type 1 gate whose output is applied in turn to a Type 3 exclusive OR gate in a cascade of such gates, then x_i and x_j need not connect to a Type 2 exclusive OR gate in the cascade.

Proof: Fig. 11(a) shows two exclusive OR gates in cascade which connect to x_i and a gate g_1, also driven by x_i. The rearrangement of exclusive OR gates shown in Fig. 11(b) does not alter the realized function. The combination of g_1 and the exclusive OR gate realizes a two-input function which can be realized by a generally different gate g'_1 as shown in Fig. 11(c). Q.E.D.

Thus,

Condition 5: In a canonical tandem network if a Type 3 exclusive OR gate connects to the output of a gate whose inputs are x_i and x_j, no adjacent exclusive OR gate connects to either x_i or x_j.

Condition 5, like its counterpart for AND and OR gates, Condition 4, eliminates the need in the synthesis algorithm to consider certain configurations. Lemma 9 below also results in a simplification.

Lemma 9: Consider a Type 3 exclusive OR gate g in a cascade of exclusive OR gates. Then g need not connect to a Type 1 exclusive OR or OR gate, unless that gate connects to network inputs of least precedence.

Proof: The proof for the Type 1 exclusive OR gate is similar to that for Lemma 8. The proof for Type 1 OR gate follows from the fact that if an input to an exclusive OR gate is complemented the function realized is the same as that obtained by complementing any other input. In particular, place an inverter on the output of each Type 1 OR which inputs the cascade. Apply de Morgan's theorem, changing these into AND gates. For each inverter so placed, insert an inverter in the input of the first exclusive OR gate (which receives from inputs of least precedence). If necessary use de Morgan's theorem to eliminate an inverter. If a Type 1 gate is there it may become an OR gate. Q.E.D.

Condition 6: A Type 3 exclusive OR gate in the cascade of a canonical tandem network does not connect to the output of a Type 1 exclusive OR or OR gate, except when the latter connects to two inputs of lowest precedence.

When the output of a tandem network is produced by a cascade of exclusive OR gates, masking of all variables does not occur (except pair masking occurs in a trivial case where a single exclusive OR gate composes the cascade). Thus, there is the problem of identifying variables which are associated with the cascade and those which are not. Since a cascade of exclusive OR gates must be preceded by a (possibly empty) cascade of AND or OR gates, certain variables will be masked by a single variable or a pair. However, within variables applied to the exclusive OR cascade certain of these will be masked. For example, Fig. 12(a) shows a network in which three variables, x_4, x_6, and x_9 are masked by one, x_6. It is important to note, however, that this is an extreme case; in general, of the inputs associated with a cascade of exclusive OR gates, at most three can be masked by a single variable (or by a pair of variables). Thus, if four or more variables X are masked by a single variable or by a pair, these need not be considered. Under this criterion none of the inputs in Fig. 12(a), for example, can be disregarded.

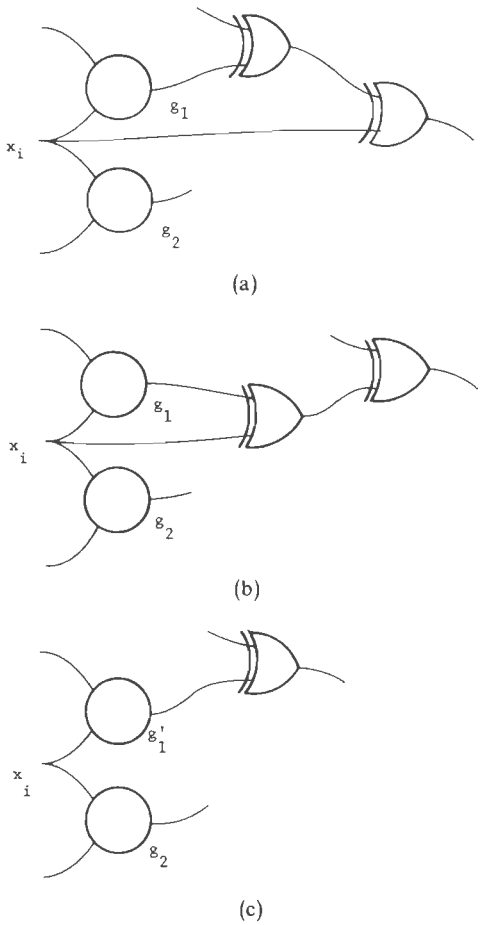

Fig. 11. (a)–(c) A partial tandem network.

Of the variables under consideration, any x_i applied to exactly one Type 1 gate g must be masked by the other input to g. Thus, Fig. 12(b) indicates x_2, x_7, x_4, x_9, x_3, and x_{10} as prospective inputs. It is then necessary to determine which inputs indeed drive Type 1 gates whose outputs are applied to an exclusive OR gate in the cascade. As a step in this direction, consider

Lemma 10: $f(X)$ has the decomposition

$$f(X) = g(X - x_j) \oplus x_i^a x_j, \tag{45}$$

where $g(X - x_j \mid a \to x_i) = f(X \mid \bar{a} \to x_i, 0 \to x_j)$ if and only if

x_j is a-masked by x_i and

$$f(X \mid \bar{a} \to x_i, 0 \to x_0) = \bar{f}(X \mid \bar{a} \to x_i, 1 \to x_j), \quad (46)$$

where $a \in \{0, 1\}$ and $x_i^0 = x_i$ and $x_i^1 = \bar{x}_i$.

Proof: (if) It follows from (46) that

$$f(X \mid \bar{a} \to x_i) = f(X \mid \bar{a} \to x_i, 0 \to x_j)\bar{x}_j + \bar{f}(X \mid \bar{a} \to x_i, 0 \to x_j)x_j$$

$$f(X \mid \bar{a} \to x_i) = f(X \mid \bar{a} \to x_i, 0 \to x_j) \oplus x_j. \quad (47)$$

Since x_j is a-masked by x_i,

$$f(X \mid a \to x_i) = f(X \mid a \to x_i, 0 \to x_j) = f(X \mid a \to x_i, 1 \to x_j). \quad (48)$$

Expressing $f(X)$ in the Shannon canonical form about x_i and substituting (47) and (48) yields

$$f(X) = [f(X \mid \bar{a} \to x_i, 0 \to x_j) \oplus x_j]x_i^a + [(f(X \mid a \to x_i, 0 \to x_j)]x_i^{\bar{a}} \quad (49)$$

which is (45) for $f(X \mid a \to x_i, 0 \to x_j) = g(X - x_j \mid a \to x_i)$.

(only if) Since $g(X - x_j)$ is independent of x_j, it follows that x_i a-masks x_j. Furthermore, (46) follows directly from (45). Q.E.D.

If a function $f(X)$, where $|X| > 3$ has no single variable or a pair which mask X, then the output of a tandem net which realizes $f(X)$, if indeed any exist, cannot come from a cascade of AND or OR gates. Thus, if $f(X)$ has the decomposition of (45) and is tandem realizable so also is $g(X - x_j)$. Furthermore, if $g(X - x_j)$ is tandem realizable and x_i is an appropriately chosen input, $f(X)$ is tandem realizable. Thus,

Theorem 6: Let $f(X)$ be a function in which no single input or pair masks X for $|X| > 3$ and let it have the decomposition

$$f(X) = g(X - x_j) \oplus x_i^a x_j.$$

Then $f(X)$ is tandem realizable if and only if $g(X - x_j)$ is realized by a canonical tandem network where x_j is or can be input to a single Type 1 AND gate g which drives a cascade of exclusive OR gates whose output is the network output.

The case where a single network input is applied to the input of an exclusive OR gate in the cascade is quite straightforward. In particular,

Lemma 11: $f(X)$ has the decomposition

$$f(X) = g(X - x_i) \oplus x_i$$

if and only if

$$f(X \mid 0 \to x_i) = \bar{f}(X \mid 1 \to x_i)$$

and we have,

Theorem 7: Let $f(X)$ have the decomposition

$$f(X) = g(X - x_i) \oplus x_i.$$

Then, $f(X)$ is tandem realizable if and only if $g(X - x_i)$ is.

The extraction of a cascade of exclusive OR gates may require that a specific input of the preceding cascade be preceded by all other inputs to that cascade. If such a requirement has been imposed on the present cascade, it is then necessary to determine if the specific input can indeed be preceded by all others. In general, any primary input to an exclusive OR gate can be preceded by all others. Also acceptable is any end variable of a set of variables applied to Type 1 gates, as long as the other end variable is not required to be the associated with the preceding cascade. The single exception to the latter statement occurs when there is a single string of Type 1 gates. For example, in Fig. 12(a), inputs, x_4, x_9, x_5, x_8, x_3, and x_{10} can be preceded by any other input.

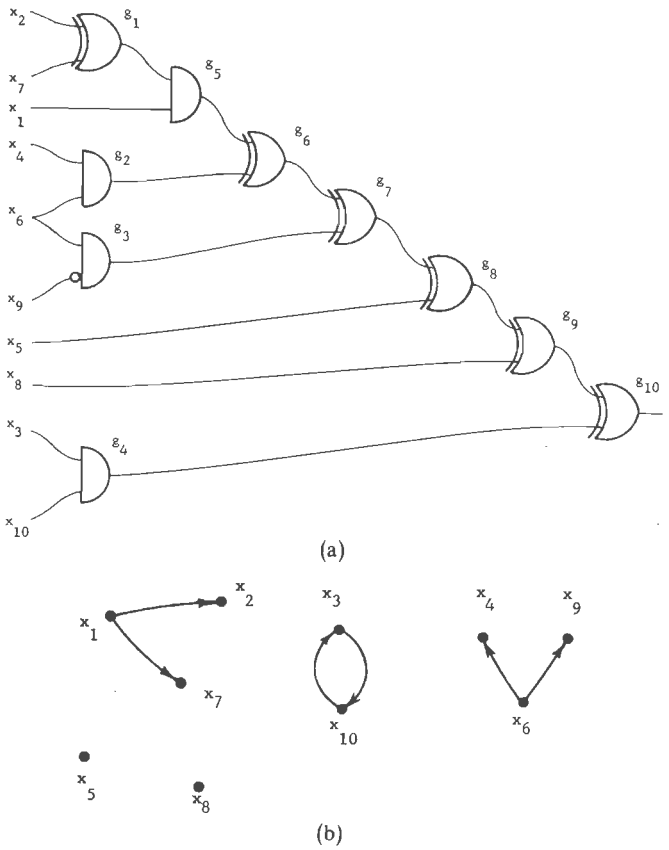

Fig. 12. (a)–(c) Example of a cascade of exclusive OR gates.

Summary of Synthesis Algorithm

The synthesis algorithm can be summarized as follows: Given a function f to be tested.

1) Set $g = f$.

2) If g is trivial, f is tandem realizable, and the algorithm halts successfully.

3) If masking of the entire variable set exists, extract a cascade of AND or OR gates, and go to 5). Otherwise, go to 4).

4) Determine if a cascade of exclusive OR gates can be extracted, do this, and go to 5). Otherwise, go to 6).

5) If a variable specified in a previous step cannot be preceded by all other variables, go to 6). Otherwise, set g equal to the residue function and go to 2).

6) Halt. f is not tandem realizable.

VI. Summary and Conclusions

This paper has focused on the tandem network, a network of universal cells whose function set includes all functions realized by the irredundant cascade. In particular, the relationship between functions realized by three cell-network interconnections and functions realized by the subnetwork was shown. This analysis leads naturally to a characterization of functions realized by the tandem net and to a counting technique for the number of functions realized. This number is significantly larger than for irredundant cascades. Finally, a synthesis technique was shown for assigning functions to cells in a tandem network such that a given function is realized.

A very interesting topic is the application of the partition matrix operation to more general interconnections than the ones considered here. For example, the problem of similar cell-network interconnections involving three and more input universal cells seems tractable. A problem of somewhat greater difficulty is the characterization of functions realized by interconnections of general networks with common input leads in terms of the functions realized by the individual networks. It appears that more advanced techniques are required to handle the proliferation of operations which result.

References

[1] K. K. Maitra, "Cascaded switching networks of two-input flexible cells," *IRE Trans. Electron. Comput.*, vol. EC-11, pp. 136–143, Apr. 1962.

[2] H. S. Stone and A. J. Korenjak, "Canonical form and synthesis of cellular cascades," *IEEE Trans. Electron. Comput.*, vol. EC-14, pp. 852–862, Dec. 1965.

[3] A. Marouka and N. Honda, "Logical networks of flexible cells," *IEEE Trans. Comput.*, vol. C-22, pp. 347–358, Apr. 1973.

[4] ——, "The range of flexibility of tree networks," *IEEE Trans. Comput.*, vol. C-24, pp. 9–28, Jan. 1975.

[5] J. T. Butler and K. J. Breeding, "Some characteristics of universal cell nets," *IEEE Trans. Comput.*, vol. C-22, pp. 897–903, Oct. 1973.

[6] K. Chakrabarti and D. Kolp, "Fan-in constrained tree networks of flexible cells," *IEEE Trans. Comput.*, vol. C-23, pp. 1238–1249, Dec. 1974.

[7] R. H. Urbano, "Structure and function in polyfunctional nets," *IEEE Trans. Comput.*, vol. C-17, pp. 152–173, Feb. 1968.

[8] J. T. Butler, "On the number of functions realized by cascades and disjunctive networks," *IEEE Trans. Comput.*, vol. C-24, pp. 681–690, July 1975.

[9] J. Sklansky, A. J. Korenjak, and H. S. Stone, "Canonical tributary network," *IEEE Trans. Electron. Comput.*, vol. EC-14, pp. 961–963, Dec. 1965.

[10] H. A. Curtis, *Design of Switching Circuits*. Princeton, NJ: Van Nostrand, 1962, pp. 277–279.

[11] J. P. Hayes, "The fanout structure of switching functions," *J. Assoc. Comput. Mach.*, vol. 22, pp. 551–571, Oct. 1975.

[12] ——, "Enumeration of fanout-free Boolean functions," *J. Assoc. Comput. Mach.*, vol. 23, pp. 700–709, Oct. 1976.

[13] K. L. Kodandapani and S. C. Seth, "On combinational networks with restricted fanout," *IEEE Trans. Comput.* to be published.

[14] E. A. Bender and J. T. Butler, "Asymptotic approximations for the number of fanout-free functions," *IEEE Trans. Comput.*, to be published.

[15] S. S. Yau and Y. S. Tang, "On identification of redundancy and symmetry of switching functions," *IEEE Trans. Comput.*, vol. C-20, pp. 1609–1613, Dec. 1971.

Jon T. Butler (M'68) was born in Baltimore, MD, on December 26, 1943. He received the B.E.E. and M.Eng.(E.E.) degrees from Rensselaer Polytechnic Institute, Troy, NY, in 1966 and 1967, respectively. He received the Ph.D. degree from Ohio State University, Columbus, OH, in 1973.

From 1967 to 1970 he served as an Officer in the Air Force Avionics Laboratories, Dayton, OH, where he conducted research and managed contracts in the areas of bionics techniques and radar countermeasures. From 1973 to 1974 he was a National Research Council Postdoctoral Associate at the Avionics Laboratories. Presently, he is an Assistant Professor in the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL. His research interests include digital design, switching theory, and cellular automata.

Dr. Butler is a member of Tau Beta Pi, Eta Kappa Nu, and Sigma Xi.