A COMBINATIONAL NETWORK SYNTHESIS METHOD BASED ON THRESHOLD LOGIC

By

P. ARATÓ

Department of Process Control, Technical University, Budapest

Received July 25, 1976

Presented by Prof. Dr. A. FRIGYES

1. Introduction

During the research in threshold logic several synthesis methods have been proposed [1, 2, 3]. In spite of the efforts, a great number of practical problems remained unsolved. The main difficulties are the testing of linear separability, the lack of a practical compound synthesis method of non-separable Boolean functions, and the practical realization of threshold gates for mass production.

The scope of this paper is to find a practical design method for combinational networks by making use of theoretical results and ideas from the threshold logic. Most of the theoretical results in threshold logic seem to be useful for a synthesis method of combinational networks with arbitrary combinational building blocks determined in advance, because the threshold gate can be considered as a generalization of the combinational gates.

The method described in this paper leads to a multilevel network of fix structure, in which the number of levels depends on the set of building blocks selected in advance. It means that the building blocks determine only the rate of convergence and not the decomposition structure of the network. The don't-care combinations of the Boolean function to be realized are really neglectable in the method and so they have an advantageous effect on the rate of convergence.

2. Terminology

Let $F(x)$ denote an arbitrary Boolean function, where x means the input vector with n bivalued components considered as real values:

$$x = (x_1, x_2, \ldots x_n)$$

Let x^1 = denote the set of input vectors for which $F(x \in x^1) = 1$,
x^0 = the set of input vectors for which $F(x \in x^0) = 0$,

x^{1j} = one of the input vectors $x \in x^1$,
x^{0p} = one of the input vectors $x \in x^0$.

The number of the vectors x^1 and x^0 depends on the truth table of $F(x)$. The input vectors corresponding to the don't-care combinations are obviously not among the vectors x^1 and x^0.

Let y^k denote the difference vector of the input vectors x^{1j} and x^{0p}:

$$y^k = x^{1j} - x^{0p}.$$

It has been shown [4, 8] that, by calculating all of the difference vectors the Boolean function can be characterized by a matrix **Y**, the rows of which are the difference vectors. Considering the special properties of the difference vectors [4, 8], the matrix **Y** can be transformed into a more concise form as follows:

$$\mathbf{R}^t = \begin{bmatrix} x^{1t} - x^{0p} \\ \cdots\cdots \\ x^{1t} - x^{1q} \end{bmatrix}$$

where x^{1t} is an arbitrary x^1-vector and x^{0p} is an arbitrary x^0-vector.

It is easy to prove that the rows of matrix $\mathbf{R}^t$ represent all of the difference vectors [8].

3. The Decomposition Structure

Let x^1_F and x^0_F denote the sets of x^1 and x^0 vectors of a Boolean function $F(x)$, respectively. Suppose another Boolean function $f_1(x)$ which determines subsets A, B and C, D of sets x^1_F and x^0_F, respectively, as follows:

$$A \cup B = x^1_F; \quad A \cap B = \emptyset \quad \text{(empty)};$$
$$C \cup D = x^0_F; \quad C \cap D = \emptyset \quad \text{(empty)},$$

(where $\cup$ denotes union and $\cap$ stands for disjunction);

$$f_1(x \in A) = 1; \quad f_1(x \in B) = 0;$$
$$f_1(x \in C) = 0; \quad f_1(x \in D) = 1.$$

Thus

$$x^1_{f1} = A \cup B; \quad x^0_{f1} = B \cup C$$

Let two other functions be defined on the subsets A, B, C, D as follows:

$$f_2\colon\ x^1_{f2} = A; \quad x^0_{f2} = D$$
$$f_3\colon\ x^1_{f3} = B; \quad x^0_{f3} = C$$

So it can be proved that

$$F(x) = f_1 f_2 + \bar{f}_1 f_3 . \tag{1}$$

This decomposition of $F(x)$ follows from the definition of subsets A, B, C, D as shown in the next table:

Subsets of input vectors	Output values of			
	f_1	f_2	f_3	F
A	1	1	—	1
B	0	—	1	1
C	0	—	0	0
D	1	0	—	0

In Figure 1 the KARNAUGH map of $F(x)$ is shown with f_1, f_2 and f_3, as input variables. Equation (1) follows from the map.

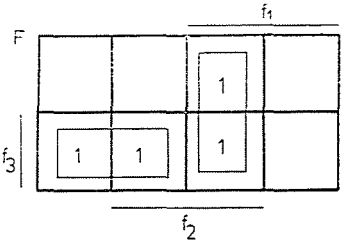


Fig. 1

There are several other possibilities for the definition of the function f_2 and f_3. For example, $F(x)$ can be expressed as

$$F(x) = f_2(f_1 + f_3), \tag{2}$$

if

$$f_2\colon\ x^1_{f_2} = AUB;\quad x^0_{f_2} = D.$$
$$f_3\colon\ x^1_{f_3} = B;\quad x^0_{f_3} = C.$$

In this case the truth table of the functions is as follows:

Subsets of input vectors	Output values of			
	f_1	f_2	f_3	F
A	1	1	—	1
B	0	1	1	1
C	0	—	0	0
D	1	0	—	0

Equation (2) can be derived from the Karnaugh map in that case as well.

A decomposition structure can be formed by successively defining a function f_1 for functions f_2 and f_3 on every level as it is shown in Figure 2, where K denotes a combinational network realizing Equations (1) or (2). Functions f_1 on each level may be similar or different, depending on the properties of the building blocks.

Defining the functions f_1 step by step, some of the subsets A,B,C,D may be found to be empty. These cases are summarized in Figure 3, where $K1$ and $K2$ denote networks according to Equations (1) and (2), respectively.

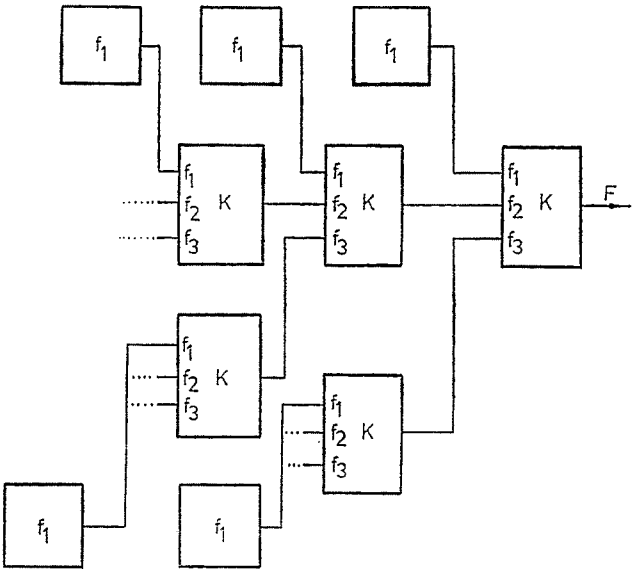


Fig. 2

The convergence of the decomposition method described above is ensured by the fact that the number of the input vectors of the function f_2 and f_3 is decreasing level by level. This follows from the definition of the sets $x^1_{f_2}$, $x^0_{f_2}$, $x^1_{f_3}$, $x^0_{f_3}$. In other words, the functions f_2 and f_3 have more and more don't-care input combinations level by level. It is obvious that, in the case when the function $F(x)$ to be realized is not completely specified, the rate of convergence will be the greatest if most of the input vectors are don't-care. So the don't-care combinations do not require special handling and are really neglectable.

4. Independence checking

For the computerization of the decomposition and design method it is important to know which variables do not affect the functions f_2 and f_3 on a given level. These variables are to be neglected because the next level

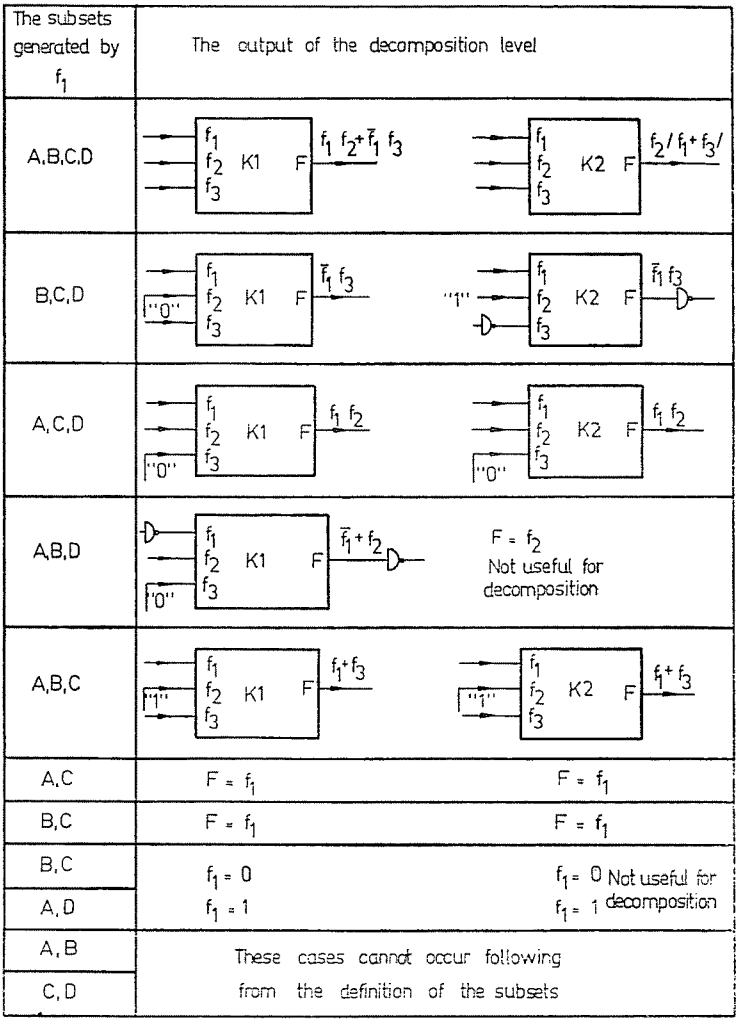


Fig. 3

is independent of them. This independence checking can be made with the use of the matrix **Y** or $\mathbf{R}^t$ [8], as it is summarized in the next statement:

A Boolean function $F(x)$ has at least one realization for each of the variables $x_{i1}, x_{i2}, \ldots x_{ir}$ which is independent at least of one of these variables, if and only if for the columns $i_1, i_2, \ldots i_r$ $(r \leq n)$ of the matrix **Y** of $F(x)$ no rows can be found in which there would be only one nonzero value in the columns $i_1, i_2 \ldots i_r$, respectively.

Proof:

For proving the necessity, assume that $F(x)$ has a realization which is independent of the variable x_i. In this case there exists no one x^1; x^0 vector

pair in which x^1 and x^0 are distinguished by x_i only. (In the opposite case $F(x)$ should depend on x_i.) Thus, among the difference vectors (the rows of $\mathbf{Y}$) there cannot be any vectors, with y_i only as a nonzero component and so the necessity is proved.

The sufficiency of the statement can be proven by assuming that the column i of $\mathbf{Y}$ has no nonzero values single in their row. In this case no x^1; x^0 vector pair can exist in which x^1 and x^0 are distinguished by x_i only. So each x^1 has its neighbour by x_i not among the vectors x^0, but among the vectors x^1 or among the vectors corresponding to the don't-care combinations. Thus $F(x)$ can be covered by prime implicants which do not contain the variable x_i and sufficiency is proved. By determining the cover of an incompletely specified function the don't-care combination vectors become x^1 or x^0. This fixing of the don't-care combinations can be done by many different ways and the different fixings may be contradictory to each other. For this reason, if more than one column $(i_1, i_2 \ldots i_r)$ of $\mathbf{Y}$ has the property mentioned in the statement, then it is not sure that there exists a realization of $F(x)$ independent of all the variables $x_{i1}, x_{i2}, \ldots x_{ir}$ simultaneously. Such a realization could afford a contradictory fixing of the don't-care combinations. Of course, the realization does exist for completely specified functions.

Example 4.1

A Boolean function is given by its combination vectors:

$$\begin{array}{cccccc} & x_1 & x_2 & x_3 & x_4 & \\ x^{11} = & (1, & 0, & 0, & 1) & \\ x^{12} = & (1, & 1, & 0, & 0) & \\ x^{13} = & (1, & 1, & 1, & 0) & \\ x^{14} = & (1, & 1, & 1, & 1) & \end{array} \qquad \begin{array}{cccccc} & x_1 & x_2 & x_3 & x_4 & \\ x^{01} = & (1, & 0, & 0, & 0) & \\ x^{02} = & (1, & 1, & 0, & 1) & \\ x^{03} = & (1, & 0, & 1, & 1) & \end{array}$$

Forming the matrix $\mathbf{Y}$,

$$\mathbf{Y} = \begin{array}{c} \begin{matrix} y_1 & y_2 & y_3 & y_4 \end{matrix} \\ \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & -1 & -1 \\ 0 & 1 & 0 & -1 \\ 0 & 1 & 0 & 0 \end{bmatrix} \end{array} \qquad \begin{array}{l} y^1 = x^{11} - x^{01} \\ y^2 = x^{12} - x^{01} \\ y^3 = x^{13} - x^{01} \\ y^4 = x^{14} - x^{01} \\ y^5 = x^{11} - x^{02} \\ y^6 = x^{12} - x^{02} \\ y^7 = x^{13} - x^{02} \\ y^8 = x^{14} - x^{02} \\ y^9 = x^{11} - x^{03} \\ y^{10} = x^{12} - x^{03} \\ y^{11} = x^{13} - x^{03} \\ y^{12} = x^{14} - x^{03} \end{array}$$

it can be concluded that the function has a realization independent of x_1, because all the values in column 1 are zeros. The *Karnaugh* map of the function is shown in Figure 4. A minimal disjunctive form derived from the map:

$$F = x_2x_3 + x_2\bar{x}_4 + \bar{x}_2\bar{x}_3\bar{x}_4 ,$$

which is independent of x_1.

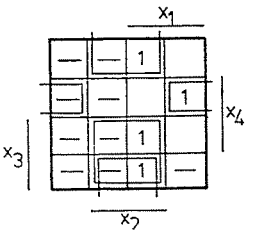


Fig. 4

Example 4.2

A Boolean function is given by its combination vectors and by the matrix **Y**:

	x_1	x_2	x_3		x_1	x_2	x_3	
x^{11} =	0,	1,	0)	x^{01} =	(0,	0,	0)	
x^{12} =	(1,	1,	0)	x^{02} =	(0,	1,	1)	
x^{13} =	(0,	0,	1)	x^{03} =	(1,	1,	1)	
x^{14} =	(1,	0,	1)	x^{04} =	(1,	0,	0)	

$$\mathbf{Y} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \\ 0 & 0 & -1 \\ 1 & 0 & -1 \\ 0 & -1 & 0 \\ 1 & -1 & 0 \\ 0 & 0 & -1 \\ -1 & -1 & 0 \\ 0 & -1 & 0 \\ -1 & 1 & 0 \\ 0 & 1 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

From **Y** we can conclude that the function has a realization independent of x_1, because column 1 does not contain a nonzero value single in its row. On the *Karnaugh* map (Figure 5) the minimal disjunctive form is illustrated

$$F = x_2\bar{x}_3 + \bar{x}_2x_3,$$

which is independent of x_1.

Example 4.3

Consider the **Y** matrix of the following function:

	x_1	x_2	x_3	x_4		x_1	x_2	x_3	x_4
x^{11} =	1,	1,	0,	1)	x^{01} =	0,	1,	0,	0)
x^{12} =	1,	1,	1,	1)	x^{02} =	0,	0,	0,	1)
x^{13} =	1,	1,	1,	0)	x^{03} =	1,	0,	0,	0)
x^{14} =	0,	1,	1,	1)					

$$\mathbf{Y} = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & -1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ -1 & 1 & 1 & 1 \end{bmatrix}$$

According to the statement, this function must have realizations each of which is independent of one variable at least. Selecting from the prime implicants signed on the Karnaugh map (Figure 6), the minimal disjunctive

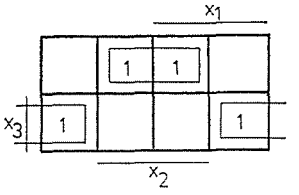


Fig. 5

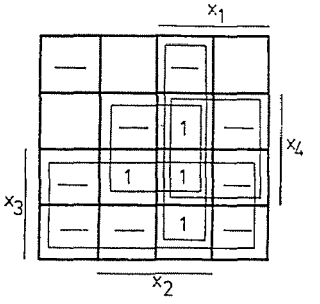


Fig. 6

forms can be derived as follow:

$$F = x_1x_2 + x_3$$
$$F = x_1x_4 + x_3$$
$$F = x_2x_4 + x_3$$
$$F = x_1x_2 + x_2x_4$$

Each of the above expressions is independent of one of the variables.

5. Defining the functions f_1

In the decomposition structure described above the rate of convergence and the simplicity of the levels are strongly influenced by the function f_1. For defining the most suitable f_1 on each decomposition level, optimality conditions ought to be exactly known and in most cases this is not easy for practical use. The optimality conditions are determined by the logical properties of the building blocks, the scale of integration, the restrictions on the propagation delay, hazards and wiring costs, etc. [3]. Instead of attempting to express exact optimality conditions the effect of some general types of f_1 is shown below.

5.1. f_1 *as a threshold function*

On each decomposition level the functions f_1 can be considered as, threshold functions. So the specification of f_1 is easy to change and to describe by determining the input weight and the threshold values [8] on each level.

Let $F^k(x)$ denote the function to be realized on the decomposition level k. The parameters of a suitable f_1 as a threshold function can be determined with the help of the matrix $\mathbf{Y}$ or $\mathbf{R}^t$ [8]. Forming the unity difference vectors, an advantageous weight vector w can be calculated [4, 8]:

$$w = \sum_{j=1}^{m} \frac{y^j}{\sum_{i=1}^{n} |y_i^j|} \tag{3}$$

where m is the number of the difference vectors of $F^k(x)$, n is the number of the variables of $F^k(x)$, and $|y_i^j|$ is the absolute value of the component i of the difference vector y^j.

The advantageous effect of this weight vector on the decomposition structure can be illustrated by the properties of the difference vectors [4, 8].

If, for the weight vector determined by Equation (3) a threshold domain (T_1, T_2) can be calculated such that

$$w \cdot (x \in x^1_{Fk}) \geq T_1$$
$$w \cdot (x \in x^0_{Fk}) < T_2$$

hold for all specified x-vectors, then f_1 realizes $F^k(x)$ as a threshold function and the decomposition procedure is finished on the k-th level. In this case the calculation of the threshold domain is based on the inequality

$$(wx^1)_{\min} > (wx^0)_{\max} .$$

The threshold values T_1 and T_2 can be placed somewhere between the above two values.

If, for the weight vector determined by Equation (3) no threshold domain can be determined by which $F^k(x)$ would be realizable, then the decomposition procedure must go on. In this case a threshold domain (T_1, T_2) must be calculated by which the subsets A,B,C,D are determinable for continuing the decomposition on the next level:

$$A\colon\ w \cdot (x \in x^1_{Fk}) \geq T_1$$
$$B\colon\ w \cdot (x \in x^1_{Fk}) < T_1$$
$$C\colon\ w \cdot (x \in x^0_{Fk}) \leq T_2$$
$$D\colon\ w \cdot (x \in x^0_{Fk}) > T_2$$

The suitable threshold domain (T_1, T_2) has an effect on the simplicity of the next levels. For this reason it is important to decide how to design the values T_1 and T_2. In that case the inequality

$$(wx^1)_{\min} \leq (wx^0)_{\max}$$

holds, and whatever values of T_1 and T_2 are determined, some of the input vectors remain unrealized on the k-th level and are left for the next levels. In the examples below the calculation of the threshold domain is made according to the expressions as follow:

$$T_1 = (wx^0)_{\max} \ ; \quad T_2 = T_1 - 1 \tag{4}$$

$$T_1 = T_2 + 1; \quad T_2 = (wx^1)_{\max} \tag{5}$$

$$T_1 = \text{entier}\left[\frac{(wx^1)_{\min} + (wx^0)_{\max}}{2}\right]; T_2 = T_1 - 1 \tag{6}$$

The calculation of the weight vectors is executed not only by Equation (3) but, in addition, according to the expressions

$$w = \sum_{j=1}^{m} y^j \tag{7}$$

$$w = \sum_{j=1}^{m} \frac{y}{|y^j|^2} \tag{8}$$

In the examples the components of the weight vectors are normalized as follows:

$$w_{i\,\mathrm{norm}} = \mathrm{entier}\left[\frac{10}{(w_i)_{\max}} \cdot w_i\right] \tag{9}$$

Of course, these ways of calculation are only illustrations of the design and decomposition method and can be changed depending on the building. blocks. For example, the procedure described above is applicable for the case in which f_1 is given in advance as a threshold gate with restricted parameters on all levels.

5.2. f_1 *as function of one variable only*

In this case the decomposition tree consists of networks K1 or K2 only. If $f_1 = x_i$ or $f_1 = \bar{x}_i$, then subsets A,B,C,D can be selected by the values of x_i or $\bar{x}_i$ in the combination vectors. The rate of convergence and the simplicity of the levels are not independent of which variable is chosen for $f = x_i$ or $f = \bar{x}_i$ on each level. For choosing a suitable variable, the matrices $\mathbf{Y}$ or $\mathbf{R}^t$ of $F^k(x)$ can be used [8]. For each variable the following formulas are to be calculated:

$$k - \left| k \cdot x_j^{11} - \sum_{p=1}^{k} x_j^{0p} \right| + \left| (l-1) \cdot x_j^{11} - \sum_{r=2}^{l} x_j^{1r} \right| \tag{10}$$

$$l + \left| k \cdot x_j^{11} - \sum_{p=1}^{k} x_j^{0p} \right| - \left| (l-1) \cdot x_j^{11} - \sum_{r=2}^{l} x_j^{1r} \right| \tag{11}$$

where x_j^{11}, x_j^{0p}, x_j^{1r} are the j-th components of the combination vectors (x^{11} is chosen arbitrarily);

l is the number of the combination vectors x^1,

k is the number of the combination vectors x^0.

Based on the above expressions, the next statement provides a tool for selecting a suitable x_i.

The subsets A and C generated by $f_1 = x_i$ or $f_1 = \bar{x}_i$ contain the most combination vectors of x^1_{Fk} and x^0_{Fk}, respectively, if the value of one of the Expressions (10) and (11) is the smallest with $j = i$.

The proof of this statement is based on the properties of the difference vectors and of the matrix $\mathbf{R}^t$ and detailed in [8].

The variables selected by the statement represent the best single-variable cover of the combination vectors of $F^k(x)$.

5.3. f_1 *as a special symmetrical function*

In the examples further on f_1 is restricted as a special symmetrical function on each level. This type of function can be described concisely as follows:

$$f_1 = S^n_{t,t+1,t+2,\ldots}(x_1 \ldots . x_n) \quad \text{or}$$
$$f_1 = S^n_{0,1,2,\ldots t-1}(x_1 \ldots . x_n),$$

where the lower indices $0, 1, 2, \ldots t-1, t+1 \ldots$ denote the symmetry numbers [1].

Functions of this type can be considered as threshold functions with input weights all equal to 1. The threshold domain depends on the value of t. It can be proved [1, 2] that this type of function forms a complete function class [3], if

$$t \neq \frac{n+1}{2}$$

which always holds for even values of t.

In this case the design procedure may be similar to that in 5.1 with f_1 as a special threshold function, the input weights of which are given in advance and only the threshold domain can be calculated on each level. This calculation becomes easy with the help of the matrix $\mathbf{Y}$ or $\mathbf{R}^t$, because it means only a comparison of the row sums [4, 8].

It is obvious that a more flexible design procedure can be formulated by allowing f_1 not to be dependent on all of the variables of $F^k(x)$. In this case a variable selection is necessary which is similar to that in 5.2, but more variables have to be selected on each level. The method of selection is based on the statement in 5.2. This selection can be considered as a generalization of the one-variable selection and is made by trials with all possible variable combinations. The effect of this method can be compared with the one-variable selection in the computer examples further on.

The design and decomposition method is summarized in the flow chart of Figure 7.

6. Examples

The design procedure outlined above is illustrated by some examples in which the defining of f_1 is made in the ways as mentioned in 5. Figures 8—14 show the Karnaugh maps of the functions to be realized and the result-network. Each result is characterized by the codes on the upper left parts of the figures as follow:

E: decomposition with network $K1$
B: decomposition with network $K2$
Y: calculating the weight vector by Expression (7)
X: calculating the weight vector by Expression (3)

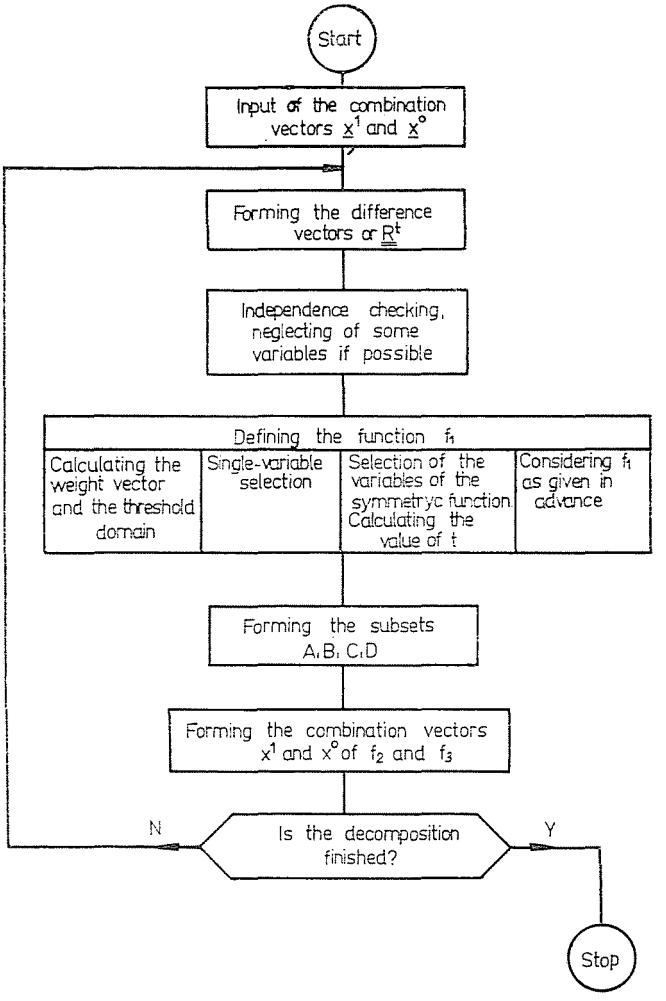


Fig. 7

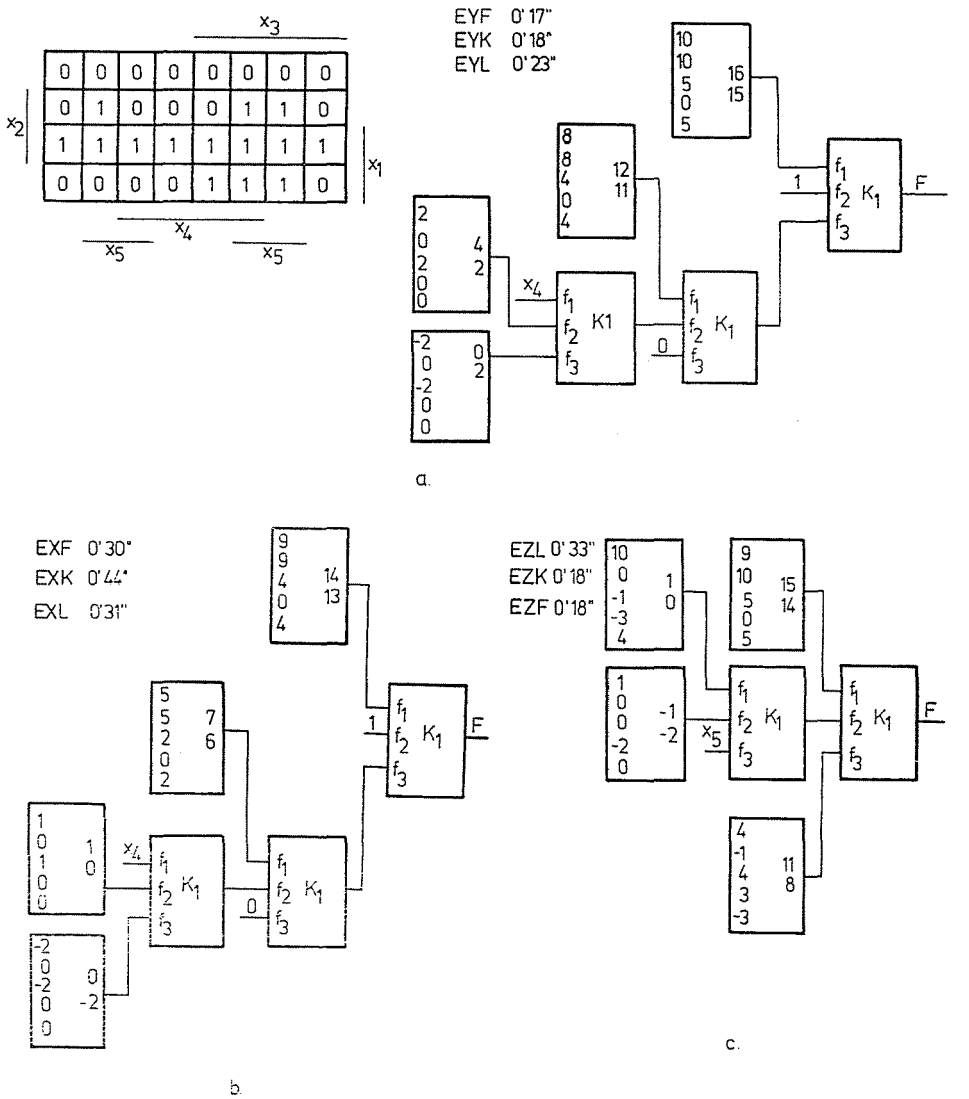

x3
x2
x1
x4
x5
x5
EYF 0'17"
EYK 0'18"
EYL 0'23"
K1
F
a.
EXF 0'30"
EXK 0'44"
EXL 0'31"
b.
EZL 0'33"
EZK 0'18"
EZF 0'18"
c.

Fig. 8

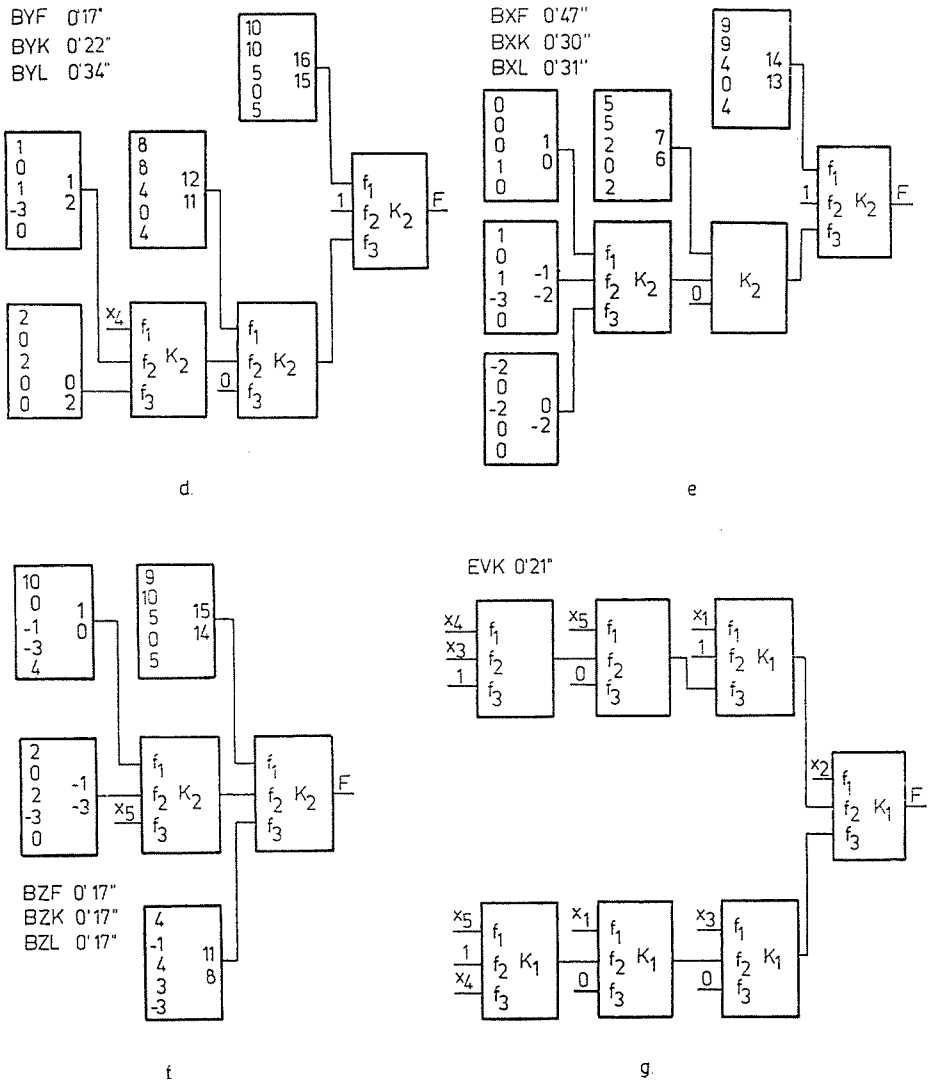
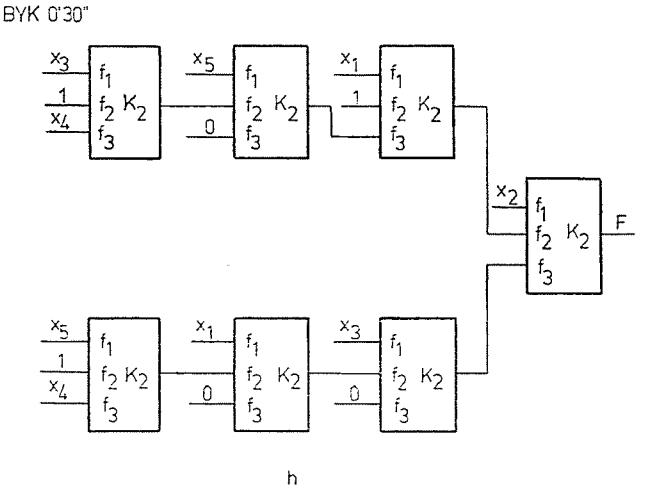

BYF 0'17"
BYK 0'22"
BYL 0'34"
BXF 0'47"
BXK 0'30"
BXL 0'31"
EVK 0'21"
BZF 0'17"
BZK 0'17"
BZL 0'17"
d
e
f
g

Fig. 8



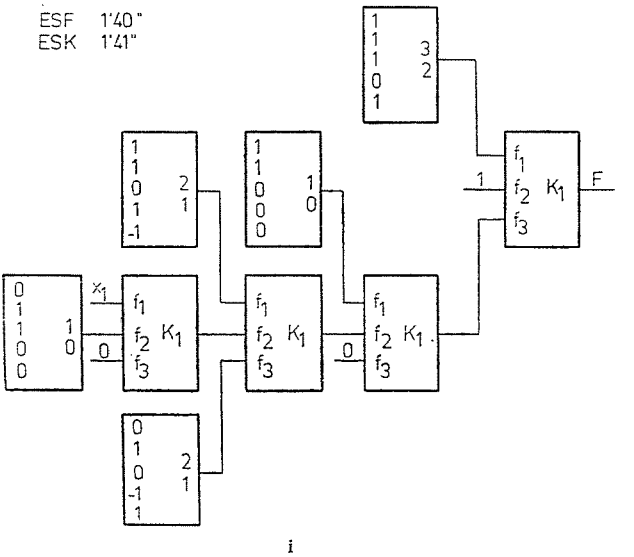
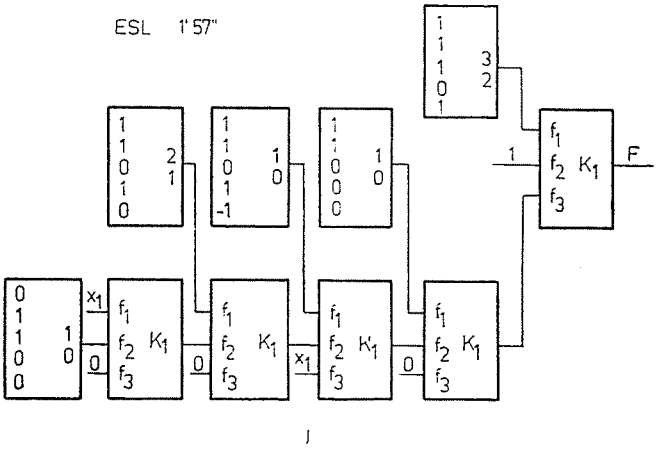
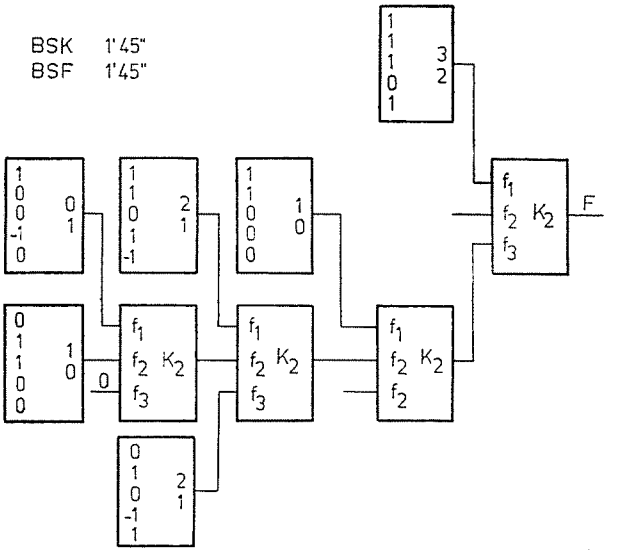


Fig. 8



j



k

Fig. 8

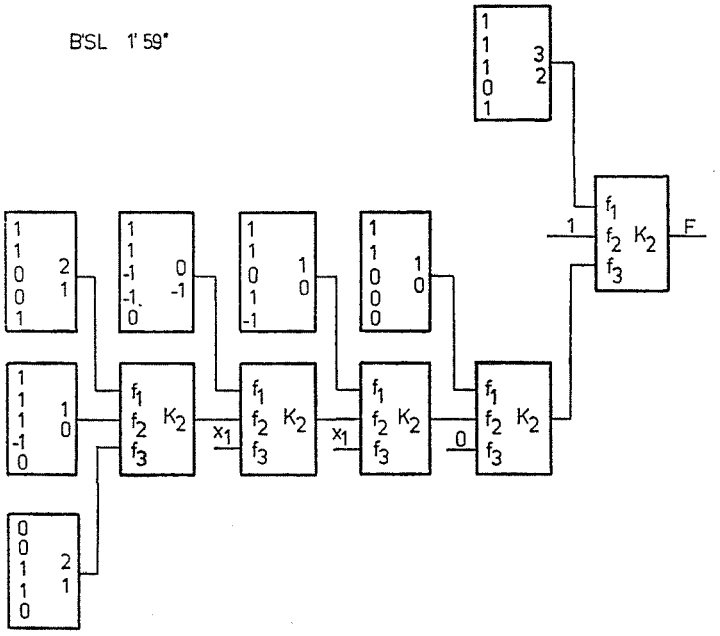


Fig. 8

Z: calculating the weight vector by Expression (8)
V: one-variable selection
S: f_1 as a special symmetric function
F: calculating the threshold domain by Expression (4)
K: calculating the threshold domain by Expression (4)
L: calculating the threshold domain by Expression (6)

The normalization of the weight vectors has been made in every case according to Expression (9). After the letter-code of the results the computing time is shown in each figure. The design program has been written for the computer ODRA 1204 in autocode MOST 2. If the function to be realized is a threshold function, then one of the known realizations is given under the Karnaugh map in the figures.

From the results of the examples it can be concluded that the way of calculating the weight vectors does not effect very much the rate of convergence with the normalization applied according to Expression (9). In some cases the multi-variable selection seems to be less advantageous than the single-variable selection (Figure 8).

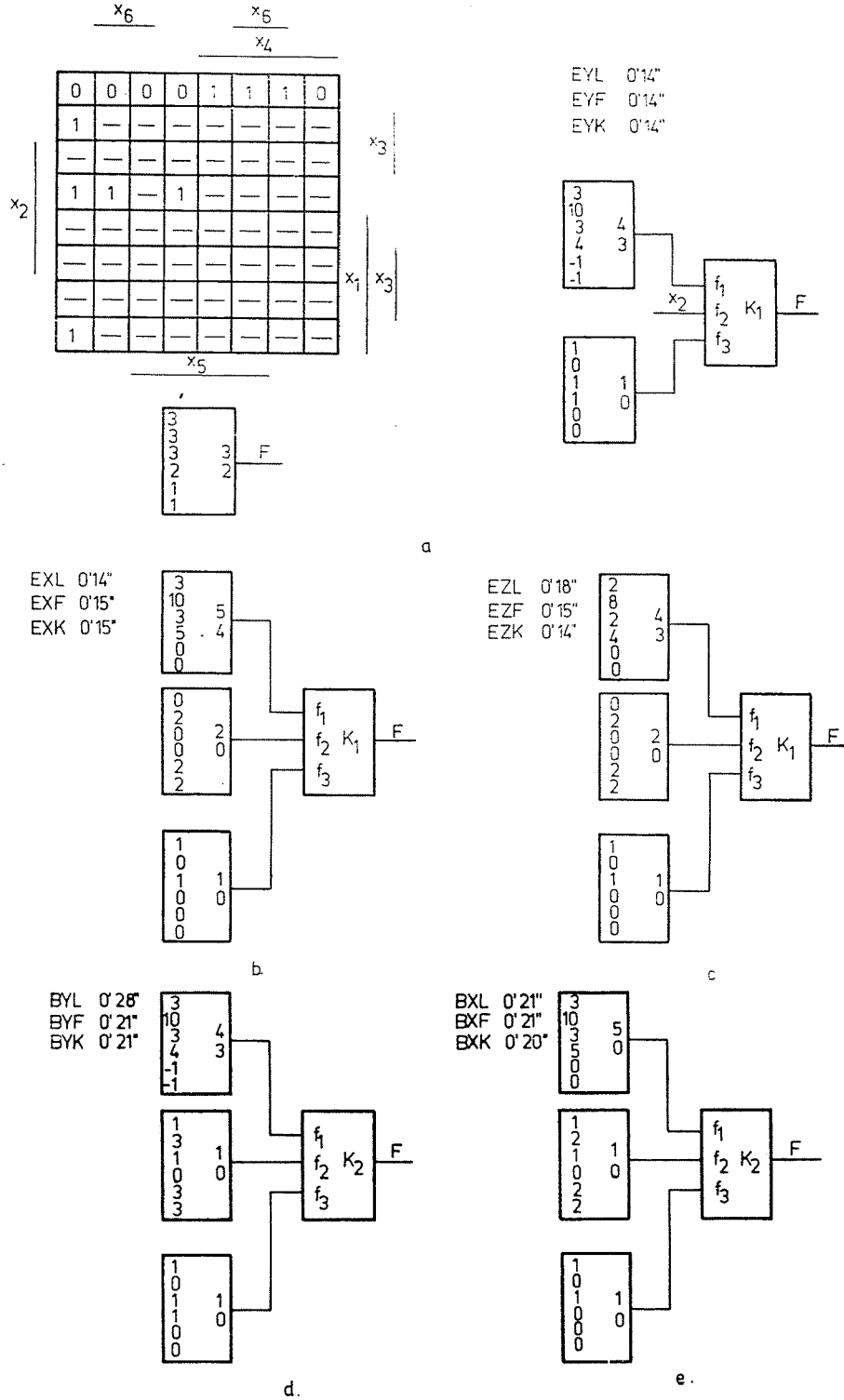

x6
x6
x4
x2
x3
x1
x3
x5
F
EYL 0'14"
EYF 0'14"
EYK 0'14"
x2
f1
f2
f3
K1
F
a
EXL 0'14"
EXF 0'15"
EXK 0'15"
b
EZL 0'18"
EZF 0'15"
EZK 0'14"
c
BYL 0'28"
BYF 0'21"
BYK 0'21"
K2
d
BXL 0'21"
BXF 0'21"
BXK 0'20"
e

Fig. 9

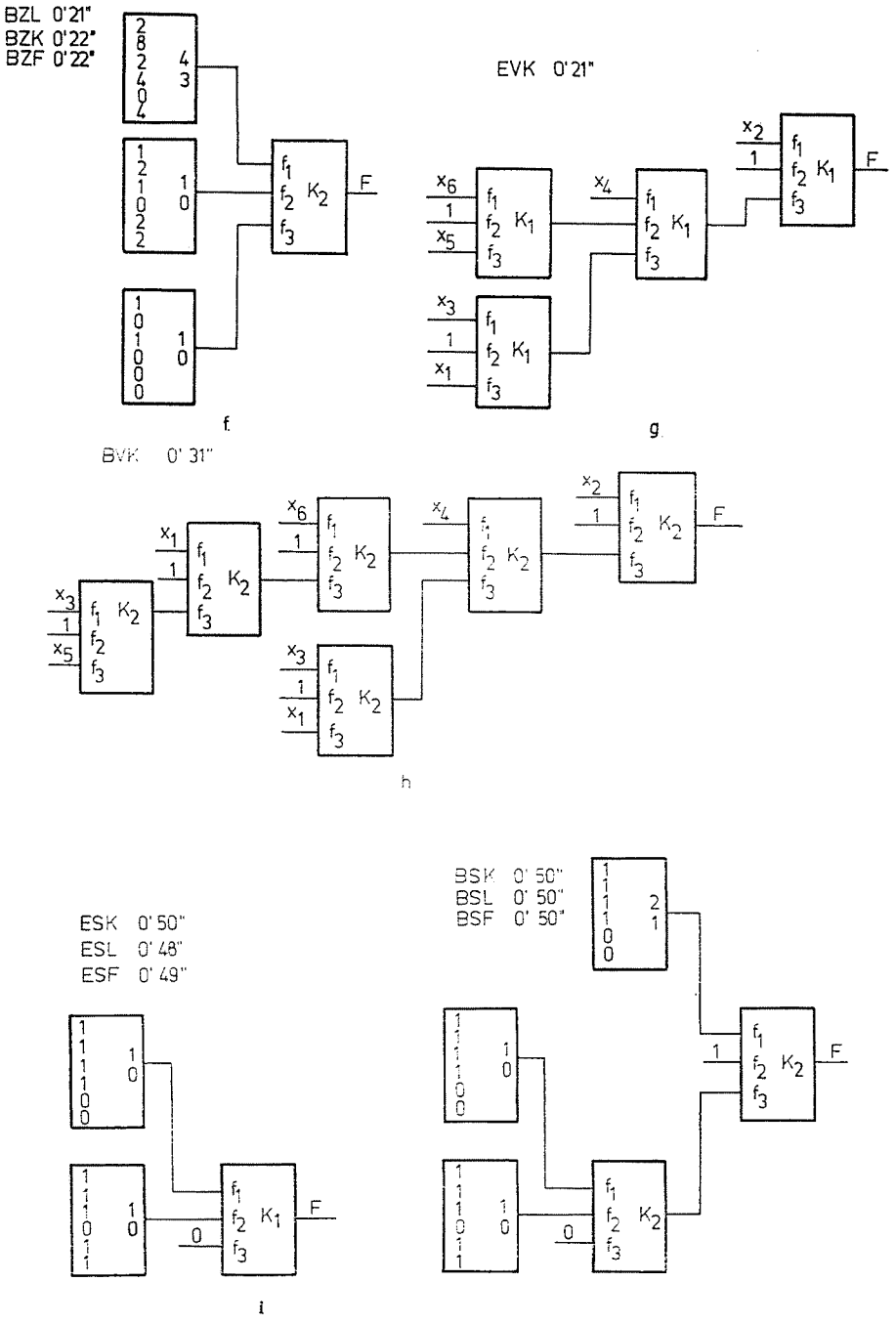
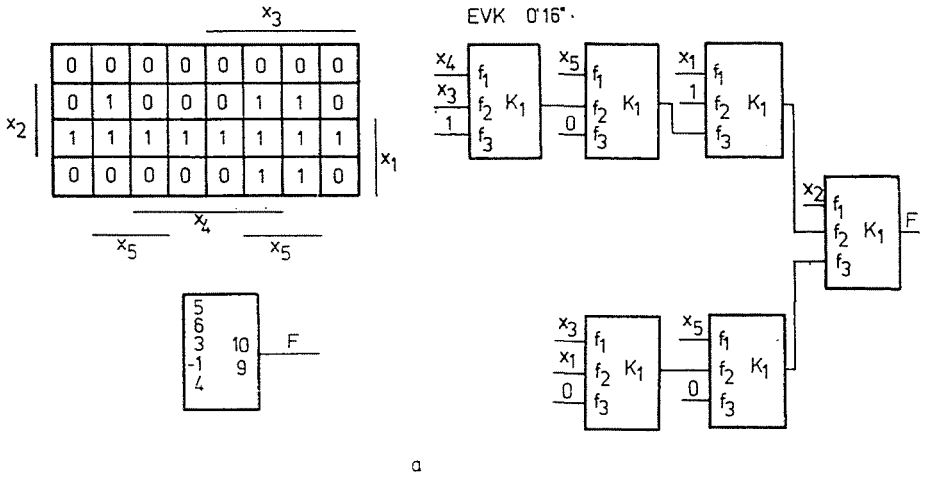

BZL 0'21"
BZK 0'22"
BZF 0'22"
EVK 0'21"
BVK 0'31"
ESK 0'50"
ESL 0'48"
ESF 0'49"
BSK 0'50"
BSL 0'50"
BSF 0'50"
f
g
h
i

Fig. 9



EYF 0' 15"
BYF 0' 15"

EZF 0' 14"
BZF 0' 14"

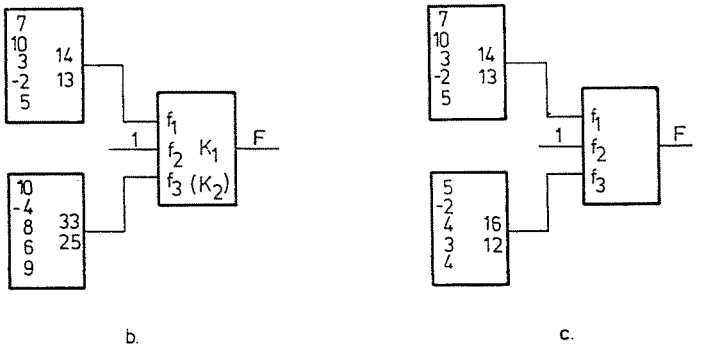


Fig. 10

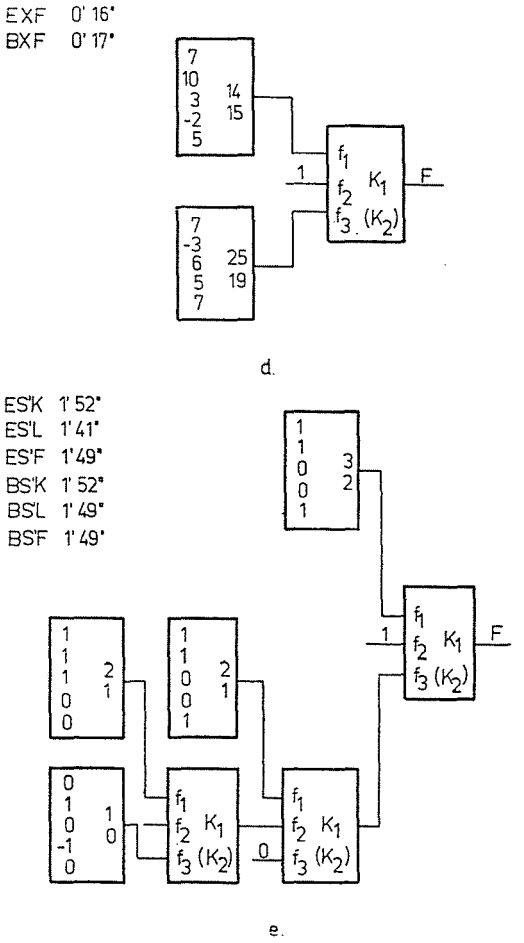

EXF 0' 16"
BXF 0' 17"
7
10
3
-2
5
14
15
7
-3
6
5
7
25
19
1
f_1
f_2 K_1
f_3 (K_2)
F
d.
ES'K 1' 52"
ES'L 1' 41"
ES'F 1' 49"
BS'K 1' 52"
BS'L 1' 49"
BS'F 1' 49"
1
1
0
0
1
3
2
1
1
1
0
0
2
1
1
1
1
0
0
2
1
0
1
1
0
-1
0
1
0
f_1
f_2 K_1
f_3 (K_2)
0
1
f_1
f_2 K_1
f_3 (K_2)
F
e.

Fig. 10

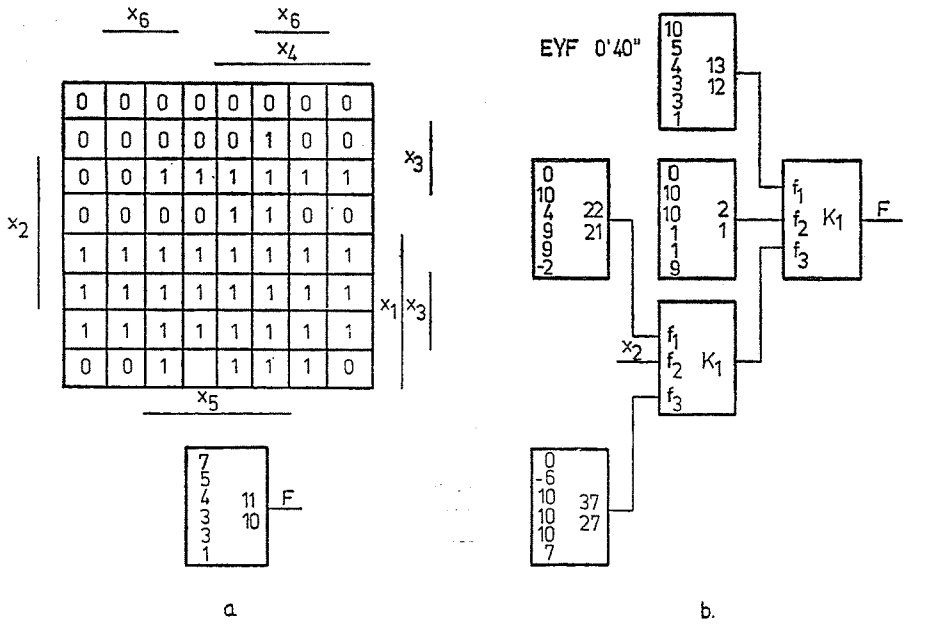
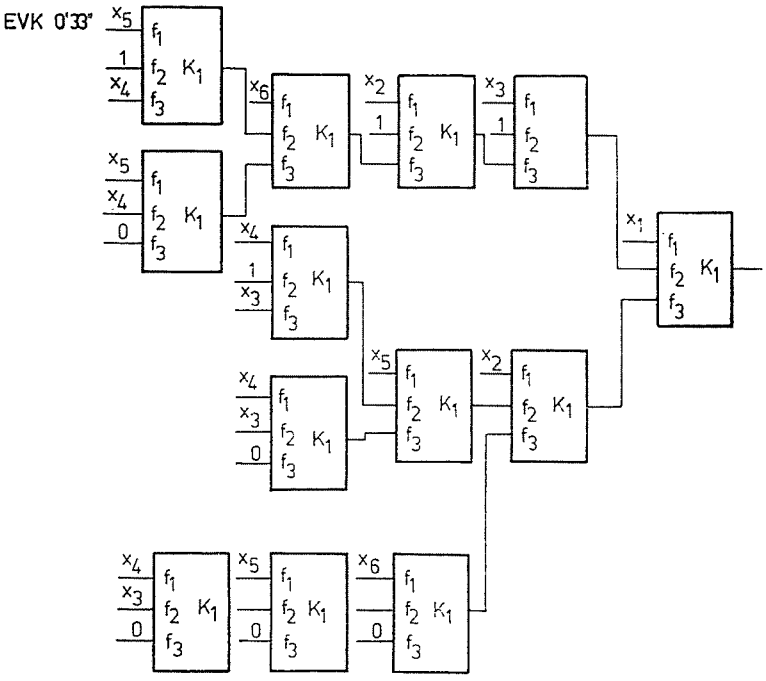

x6
x6
x4
x3
x2
x1
x3
x5
F
a
EYF 0'40"
22
21
13
12
2
1
f1
f2
K1
f3
F
x2
37
27
b.
EXF 0'36"
BXF 0'37"
12
11
c.

Fig. 11



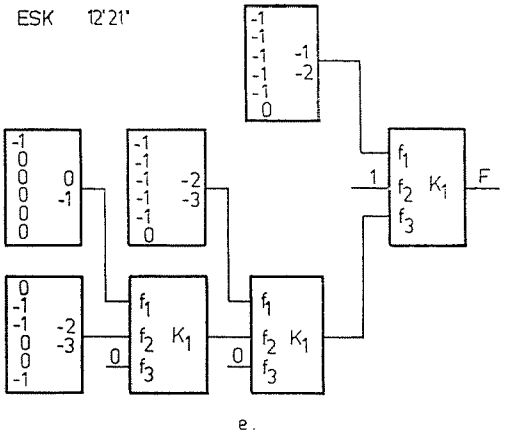
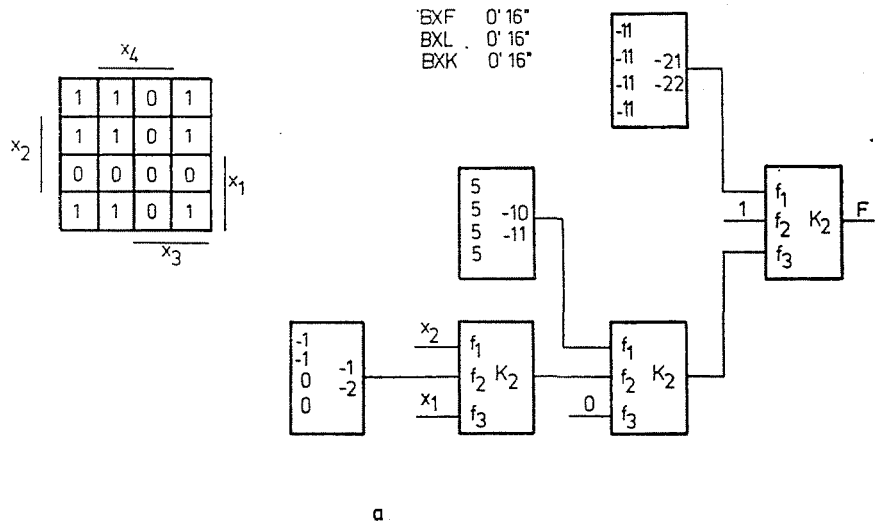
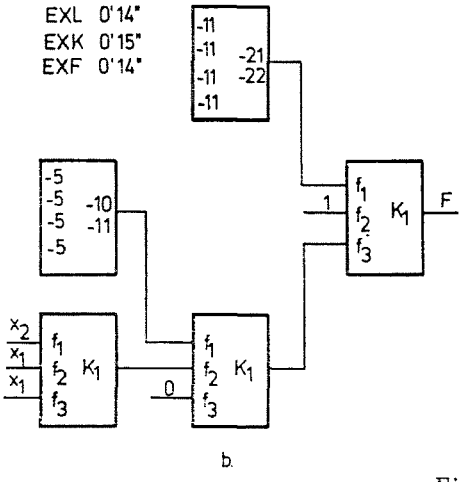


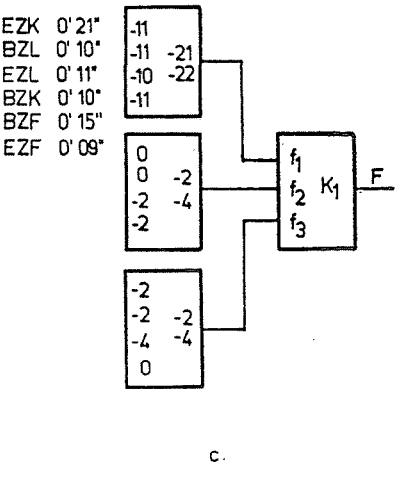
Fig. 11


x4
x2
x1
x3
1 1 0 1
1 1 0 1
0 0 0 0
1 1 0 1
BXF 0'16"
BXL 0'16"
BXK 0'16"
-11
-11 -21
-11 -22
-11
5
5 -10
5 -11
5
-1
-1 -1
0 -2
0
x2
x1
f1
f2 K2
f3
1
0
F

a


EXL 0'14"
EXK 0'15"
EXF 0'14"
-11
-11 -21
-11 -22
-11
-5
-5 -10
-5 -11
-5
x2
x1
x1
f1
f2 K1
f3
1
0
F

b


EZK 0'21"
BZL 0'10"
EZL 0'11"
BZK 0'10"
BZF 0'15"
EZF 0'09"
-11
-11 -21
-10 -22
-11
0
0 -2
-2 -4
-2
-2
-2 -2
-4 -4
0
f1
f2 K1
f3
F

c

Fig. 12

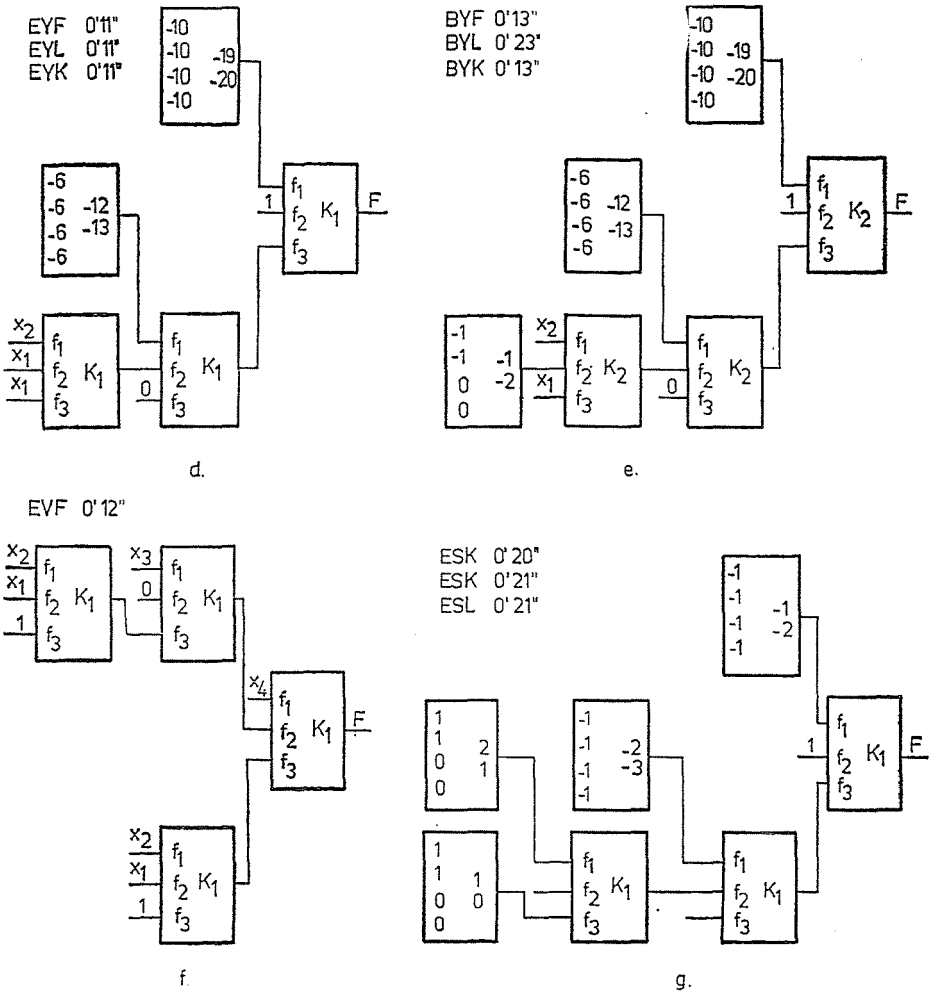

EYF 0'11"
EYL 0'11"
EYK 0'11"
BYF 0'13"
BYL 0'23"
BYK 0'13"
EVF 0'12"
ESK 0'20"
ESK 0'21"
ESL 0'21"
F
d.
e.
f.
g.

Fig. 12

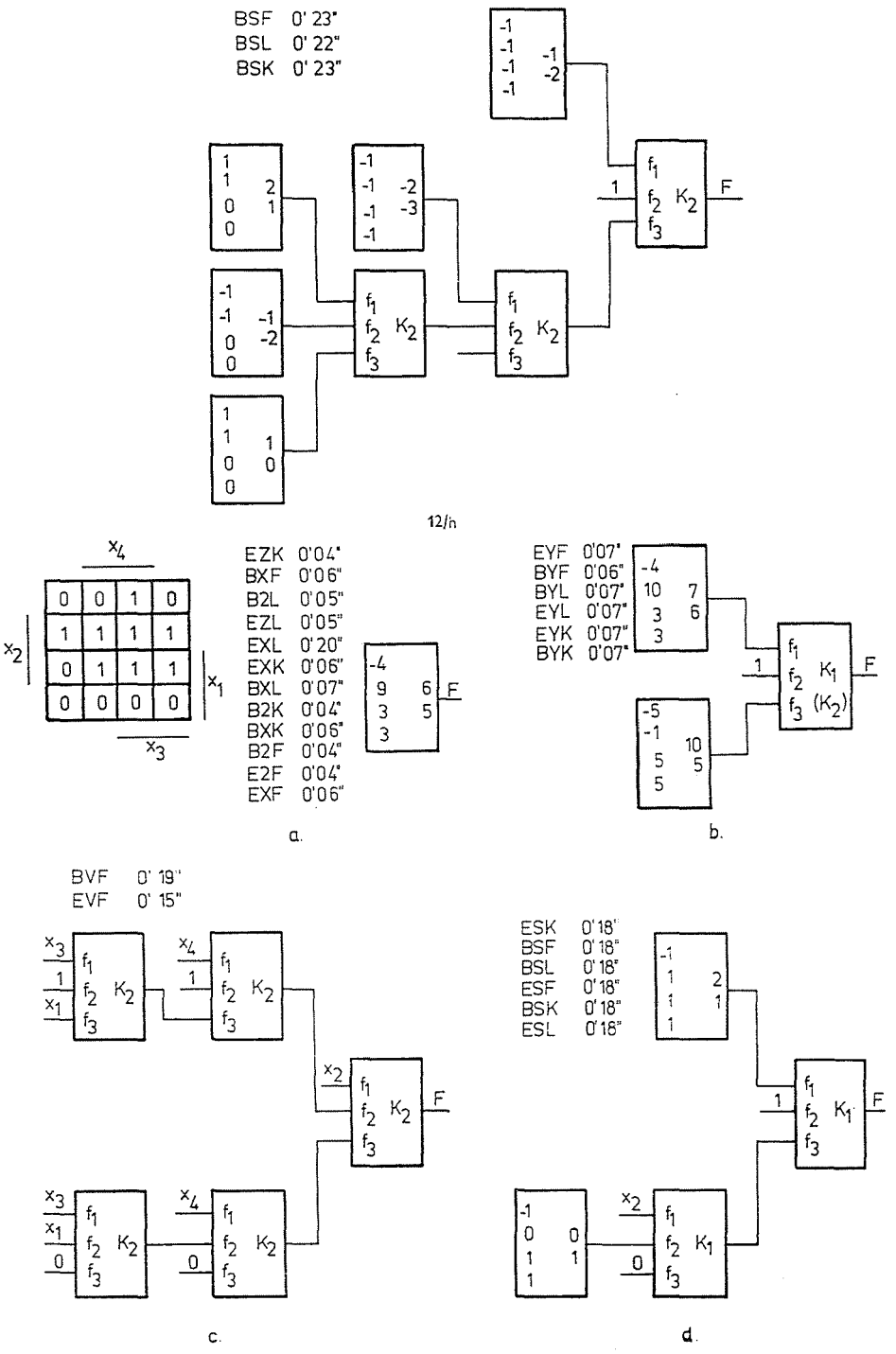

BSF 0'23"
BSL 0'22"
BSK 0'23"
12/h
EZK 0'04"
BXF 0'06"
B2L 0'05"
EZL 0'05"
EXL 0'20"
EXK 0'06"
BXL 0'07"
B2K 0'04"
BXK 0'06"
B2F 0'04"
E2F 0'04"
EXF 0'06"
EYF 0'07"
BYF 0'06"
BYL 0'07"
EYL 0'07"
EYK 0'07"
BYK 0'07"
a.
b.
BVF 0'19"
EVF 0'15"
ESK 0'18"
BSF 0'18"
BSL 0'18"
ESF 0'18"
BSK 0'18"
ESL 0'18"
c.
d.

Fig. 13

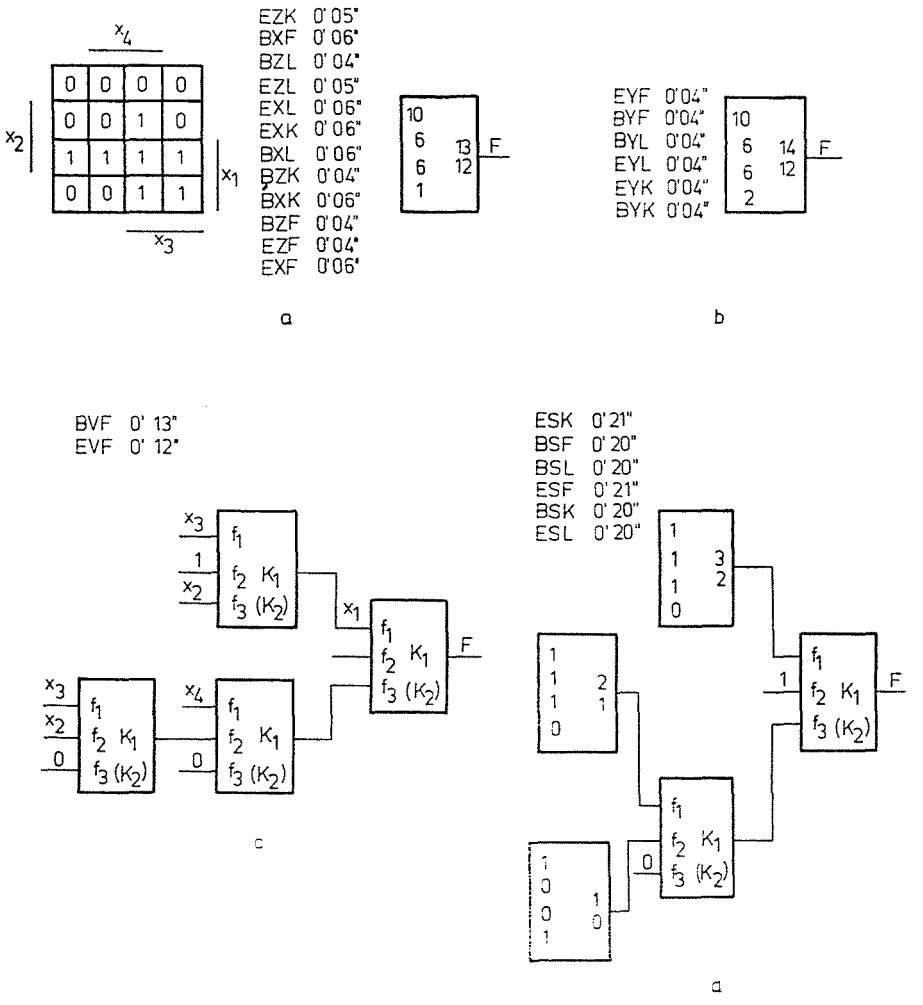


Fig. 14

7. Conclusions

The design method described above is applicable for the practical synthesis of combinational networks with arbitrary building blocks given in advance. The properties of the building blocks can be taken into consideration in the specification of the function f_1 for all levels or for each level separately. In the latter case the defining methods differ from the outlined ones only in the determination of the subsets A,B,C,D.

There is also another way of using arbitrary building blocks. The properties of the building blocks may be left out of consideration in the

specification of f_1 and after having got the resulting decomposition network, only the resulting functions f_1 and the networks $K1$ or $K2$ are to be realized by means of the arbitrary building blocks given in advance. For these possibilities the defining methods outlined in Part 5 are only illustrations and their aim is to test the procedure by some examples.

For the design of multiple output combinational networks the method described above can be used in two different ways:

a) The functions f_1 may be defined as common for several output functions on each level.

b) For the design of multiple-output networks the single-output method is applicable with constructing a proper single-output network [6, 7].

The elimination of the logical hazards is a very important task in a design procedure. In the design and decomposition method described in this paper the hazard elimination seems to be feasible by making special restrictions on the subsets A,B,C,D, and it is one of the subjects of further research.

Summary

In the paper a synthesis method is described, by which a design procedure can be constructed, using threshold gates with parameters given in advance. Therefore, the method is applicable for combinational network synthesis with arbitrary combinational gates chosen in advance. The method does not require a special handling of the not completely specified Boolean functions. The realization of the synthesis is made by a fixed multilevel decomposition structure. Some computer examples are given in order to illustrate the efficiency of the method.

References

1. LEWIS, P. M. II.—COATES, C. L.: Threshold Logic, New York, Wiley (1967).
2. DERTOUZOS, M. L.: Threshold Logic: a Synthesis Approach, MIT (1965).
3. MUROGA, S.: Threshold Logic (its Applications), Wiley-Interscience (1971).
4. ARATÓ, P.: Some Theorems for a New Synthesis Method in Threshold Logic, Periodica Polytechnica, El. Eng. 13, (1969) No. 4.
5. ARATÓ, P.: An Upper Bound for the Relative Gap of Threshold Functions, Periodica Polytechnica, El. Eng. 14, (1970) No. 3.
6. BARTEE, T. E.: Computer Design of Multiple Output Logical Networks, IRE Transactions EC. 10 (1961) pp. 21—30.
7. BEISTER, J.—ZIEGLER, R.: Zur Minimierung von Funktionenbündeln, NTG Fachtagung, Karlsruhe, 25—27. 9. 1974.
8. ARATÓ, P.: A Combinational Network Synthesis Method Based on Threshold Logic (a dissertation in Hungarian), Budapest, 1974.

Dr. Péter ARATÓ, H-1521 Budapest