Technical Report CS77006-R

Nonorthogonal Transforms for Logical Design

Jon C. Muzio*

June 1977

University of Manitoba

and Virginia Polytechnic Institute and State University

*This work was performed while the author was on sabbatical at the University of Bath and Virginia Polytechnic Institute and State University. Grateful acknowledgement is made of the financial assistance of NATO under their Senior Fellowship Programme.

Nonorthogonal Transforms for Logical Design

Several authors have given detail study to the use of orthogonal transforms for logic design purposes and for the classification of switching functions (see [1] - [5]). The majority of these have used Rademacher-Walsh transforms to deduce the relationship of a given function with various XOR functions of the variables. The purpose of this report is to investigate whether a useful technique may be based on nonorthogonal transforms constructed to match the particular modules which it is desired to use to realize the function. The primary advantage of the use of orthogonal transforms is their ease of inversion enabling the recovery of the original function. In our case where the transforms are not even usually square it will be necessary to keep a copy of the specification of the function.

The approach is to assume that we are given a function which is to be realized in terms of some given subfunction. A transform corresponding to this subfunction is constructed and it is used to indicate which particular variables should be substituted into the subfunction as an initial module to realize the function.

Two algorithms will be presented, neither of which is amenable to hand calculation. The first tends to produce decompositions which have many levels but few modules at each level while the second leads to realizations with fewer levels but more modules at each level. Two rather small examples are considered in some detail in section 3.

The method presented is currently under development to determine exactly which types of problems it is applicable to. It is clear that in

certain situations it will fail altogether and it is not suggested that it will provide an adequate design method on its own. Rather it should be used in conjunction with a number of other approaches to generate a useful total design package (see, for example, [2], [6], and [7]). The advantage of this approach is its ability to work with an arbitrary module which may be quite large.

1.1 Notation and Definitions

Our concern is with switching functions of n variables $f(x_1 \cdots x_n)$, which map ordered n-tuples of 0's and 1's into {0,1}. These ordered n-tuples are called "vertices". Two vertices which differ in exactly one position are called "adjacent". Sets of adjacent vertices are termed "cubes", defined as follows: a cube $c_1=c_{11}\ c_{12} \cdots c_{1n}$, $c_{1i}\varepsilon\{0,1,X\}$ includes all those vertices $a_1 \cdots a_n$ such that $a_i=c_{1i}$ if $c_{1i}\varepsilon\{0,1\}$ else $a_i\varepsilon\{0,1\}$ (for each i, $1\leq i\leq n$).

A switching function of n variables may be defined by a table listing all its 2^n vertices in order with the value taken by the function being listed as the (n+1)th column. This (n+1)th column will be called the specification vector $\underset{\sim}{F}$ for a function f. Alternatively the function may be defined for all those cubes and vertices for which it takes the value 1 (the "on-array") and for all the cubes and vertices for which it takes the 0 (the "off-array). The functions considered may be either totally specified (assigned a value in {0,1} for all 2^n vertices) or partially specified (there exist vertices for which the function is not assigned a value). If the function is partially specified a realization will consist of any completely specified function that agrees with the partially specified function at all vertices for which the latter is defined.

The "intersection of two cubes $c_1 = c_{11} \ldots c_{1n}$ and $c_2 = c_{21} \ldots c_{2n}$ is defined by:

$c_1 \wedge c_2 = c_{11} \cap c_{21}, c_{12} \cap c_{22}, \ldots, c_{1n} \cap c_{2n}$

where "$\cap$" is defined by the table shown.

p ∩ q		0	1	X	q
	0	0	∅	0	
p	1	∅	1	1	
	x	0	1	X	

The intersection of two cubes is called their "intersection cube" (though note that it is not a genuine cube).

The "distance" between two cubes is the number d of distinct occurences of ∅ in their intersection cube. Such as intersection cube will be referred to as a distance-d intersection cube. When convenient the positions occupied by the ∅'s will also be noted, for example 0 ∅ 0X1 ∅ 1 is a distance 2 intersection cube in x_2 and x_6.

1.2 Redundant Variables

The algorithms to be developed will be heavily involved with the checking for redundant variables and two different procedures will be used for this.

Initially the simplest test is for single redundant variables, as follows:

1. Evaluate all distance 1 intersection cubes formed by the intersection of one cube in the on-array and one cube in the off-array.

2. A necessary and sufficient condition for a variable x_i to be redundant is that this list contains no distance 1 intersection cubes in x_i.

These conditions can easily be widened for larger sets of variables, for example:

3. The variables x_i and x_j are both redundant if and only if

a) there are no distance 1 intersection cubes in either x_i or x_j; and

b) there are no distance 2 intersection cubes in x_i and x_j.

For the second algorithm we need a rather more complex technique since a large number of redundant variables may have been introduced at the previous stage. As our requirement is for a near minimal set of non-redundant variables it is essential to look at every intersection cube. Each intersection cube yields a list of variables which cannot all simultaneously be removed although all but one may be. For example, if c_1=011XX1010 and c_2=00X10X100 then $c_1 \wedge c_2$=0 Ø 1101ØØ0 which gives the set of variables $\{x_2,x_7,x_8\}$ which cannot all be simultaneously redundant. Each intersection cube leads to such a set though in practice most of the sets can be removed since if we have two such sets A,B with $A \subseteq B$ then B is redundant. The problem remaining is to choose a minimal number of elements from some set of sets so that at least one is chosen from each set. This is, of course, the classic covering problem and may be solved using a prime implicant table or any similar technique. For our application it is quite adequate to deduce a reasonable near minimal set rather than spending a large amount of time to find a minimal set. Indeed the minimal set will not always lead to the better solution.

2. The Construction and Use of the Transform

The construction of the transform is straightforward, and it is built

in a similar fashion to the Rademacher-Walsh transform for XOR functions (see [3]). When a function $f(x_1,\ldots,x_n)$ with a specification vector $\underset{\sim}{F}$ and a desired subfunction or module $M(y_1,\ldots,y_m)$, $(m\leq n)$, our aim is to deduce a reasonable substitution of the x variables into M as a good initial decomposition of f (assuming such a subfunction is reasonable for the realization of f).

Each row of the transform is based on a single function $M(y_1,\ldots,y_m)$ with each $y_i\varepsilon\{x_1,\ldots,x_n,0,1\}$. Only substitutions leading to nontrivial distinct functions are included and inverses of functions included are unnecessary. Suppose there are k such functions. Then the transform will contain k rows. In practice various heuristic arguments may be employed to remove rows representing functions that are unlikely to be useful.

The 2^n columns of the transform correspond to the 2^n combinations of the variable $x_1,\ldots,x_n$ and the exact ordering of these columns does not matter as long as the specification vector for f is based on the same ordering. We will assume that column j $(1\leq k\leq n)$ corresponds to the assignment $x_k = a_k$ for each k $(1\leq k\leq n)$ where

$$j = \sum_{k=1}^{n} a_k 2^{n-k}$$

Suppose row i corresponds to the function $M(x_{i1},x_{i2},\ldots,x_{im})$. Then

$$T = \begin{cases} 1 \text{ if } M(a_{i1},\ldots,a_{im})=0 \\ -1 \text{ if } M(a_{i1},\ldots,a_{im})=1 \end{cases}$$

Essentially the rows of the transform are just the specification vectors for the functions concerned coded by $0\rightarrow 1, 1\rightarrow -1$.

As an example suppose our interest is in a 3 variable function

$f(x_1,x_2,x_3)$ to be realized using the subfunction $M(y_1,y_2) = y_1 + \bar{y}_2$. For this function we have the following nontrivial substitutions: $M(x_1,x_2)$, $M(x_1,x_3)$, $M(x_2,x_1)$, $M(x_2,x_3)$, $M(x_3,x_1)$, $M(x_3,x_2)$, $M(x_1,1)$, $M(x_2,1)$, $M(x_3,1)$. All other substitutions lead to functions that are either trivial, duplicates, or inverses of functions listed above (the last three are just the three variables themselves).

Consequently the transform will have 9 rows and 8 columns, as shown below. Each row is labelled by the function leading to it and the combination of variables for

$$
\begin{array}{c|cccccccc}
 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
x_1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
x_2 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\
x_3 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
\hline
M(x_1,x_2) & -1 & -1 & 1 & 1 & -1 & -1 & -1 & -1 \\
M(x_1,x_3) & -1 & 1 & -1 & 1 & -1 & -1 & -1 & -1 \\
M(x_2,x_1) & -1 & -1 & -1 & -1 & 1 & 1 & -1 & -1 \\
M(x_2,x_3) & -1 & 1 & -1 & -1 & -1 & 1 & -1 & -1 \\
M(x_3,x_1) & -1 & -1 & -1 & -1 & 1 & -1 & 1 & -1 \\
M(x_3,x_2) & -1 & -1 & 1 & -1 & -1 & -1 & 1 & -1 \\
x_1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\
x_2 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\
x_3 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1
\end{array}
= \underset{\sim}{T}
$$

each column is listed at the top.

Having constructed $\underset{\sim}{T}$ we multiply it by a coded version of $\underset{\sim}{F}$ in that 0 is represented by -1, 1 is represented by 1, and don't cares (if there are any) are represented by 0's. This yields the vector $\underset{\sim}{R}$ viz $\underset{\sim}{R} = \underset{\sim\sim}{TF}$.

If $\underset{\sim}{T}$ is one of the orthogonal Rademacher-Walsh transforms then $\underset{\sim}{R}$ will be the spectrum of the function. For our purposes $\underset{\sim}{R}$ will be referred to as a correlation vector since its entries measure the agreement or disagreement of f with the functions used to build the transform. Note that since it will usually not be possible to easily recover $\underset{\sim}{F}$ from $\underset{\sim}{R}$ it is essential to retain a copy of $\underset{\sim}{F}$. In practice it is most convenient to store this in the form of the on and off arrays for the function. It is possible that an evaluation technique similar to that used in [8] might be more efficient for generating $\underset{\sim}{R}$. The entries in R give a measure of the agreement or disagreement of f with the possible M functions. This procedure works on the assumption that functions corresponding to the larger absolute values in $\underset{\sim}{R}$ are the preferable initial subfunctions. If the sign of the entry in $\underset{\sim}{R}$ is negative the agreement is with the function and if positive it is with the inverse of the function.

At this state the two methods diverge so the algorithms will be specified.

Algorithm 1

1. For a given module (or modules) form the transform $\underset{\sim}{T}$ of the correct size.

2. Evaluate $\underset{\sim}{R}=\underset{\sim}{T}\underset{\sim}{F}$ where $\underset{\sim}{F}$ is the specification vector for the function of n variables to be realized.

3. Investigate the modules corresponding to the large absolute value coefficients in $\underset{\sim}{R}$.

4. Select one such module as a subfunction and append it to the function as an extra variable.

5. Check the new function for redundant variables other than the newly appended subfunction.

6.* If none are redundant delete the module used and return to 4. If one or more are redundant delete them and return to 1 or 2 (as appropriate).

7. Continue the procedure until the function has been completely decomposed.

* After performing 6 a number of times it may become apparent that it is necessary to procede to the next stage carrying n+1 variables. Currently a number of loose heuristic methods for detecting when this is advantageous have been used.

The approach described above is serial in the sense that a single module is selected at each stage. The disadvantage of this approach is a tendency for the realization to have many levels of modules with few modules at each level. In practical terms the opposite situation is often preferable, viz a realization with fewer levels and more modules at each level. This is the aim of the second algorithm.

Algorithm 2

1. For a given module (or modules) form the transform $\underset{\sim}{T}$ of the correct size.

2. Evaluate $\underset{\sim}{R}=\underset{\sim}{T}\underset{\sim}{F}$ where $\underset{\sim}{F}$ is the specification vector for the function of n variables to be realized.

3. Investigate the modules corresponding to the large absolute value coefficients in $\underset{\sim}{R}$.

4. Select some set of m of these modules and append them to the

function as extra variables (the value of m will depend on the values in $\underset{\sim}{R}$). These m variables are called the 'new variables'.

5. Check the new function for redundant variables deducing a minimal or near minimal size set of nonredundant variables. If there are several sets of the same size choose one with the smallest number of new variables.

6. Return to 1 or 2 as appropriate, unless 5 has been a complete failure in which case return to 4 and consider a different set of new variables.

7. Continue the procedure until the function has been completely decomposed.

The decision as to whether to procede with an n+1 variable problem in algorithm 1 depends on the nature of the particular function under study. It is also possible to use the method in conjunction with the technique described in [7] so that we can determine initially which subfunctions will lead to redundancies. In practice all modules characterized by large values in R are investigated prior to proceding with an n+1 variable problem.

The algorithms will be illustrated using the transform $\underset{\sim}{T}$ defined above for $M(y_1,y_2)=y_1 + \bar{y}_2$. For convenience both $\underset{\sim}{F}$ and $\underset{\sim}{R}$ will be written as row vectors. To illustrate the various stages Karnaugh maps will be used (blank entries indicating 'don't cares').

We use algorithm 1 initially. Suppose $f(x_1,x_2,x_3)=\bar{x}_1\bar{x}_2+\bar{x}_1x_3$ and is completely specified.

ON x_1	x_2	x_3
0	0	X
0	X	1

OFF x_1	x_2	x_3
1	X	X
X	1	0

x_1 \ x_2x_3	00	01	11	10
0	1	1	1	0
1	0	0	0	0

$\underset{\sim}{F}$=[1 1 -1 1 -1 -1 -1 -1], $\underset{\sim}{R}=\underset{\sim}{T}\underset{\sim}{F}$=[2 6 2 2 -2 -2 6 2 -2]

Based on the first entry of 6 let $\bar{x}_4 = M(x_1, x_3)$ giving

ON x_1 x_2 x_3 x_4					OFF x_1 x_2 x_3 x_4				
c_1	0	0	0	0	c_3	1	X	X	0
c_2	0	X	1	1	c_4	X	1	0	0

Since the only distance 1 intersection cubes are

$$c_1 \wedge c_3 = \emptyset\ 000$$

and $c_1 \wedge c_4 = 0\ \emptyset\ 00$ it follows that x_3 is redundant leaving

ON x_1 x_2 x_3			OFF x_1 x_2 x_3		
0	0	0	1	X	0
0	X	1	X	1	0

x_1 \ x_2x_4	00	01	11	10
0	1	1	1	0
1	0			0

(note that the function is only partially specified now)

$\underset{\sim}{F}$=[1 1 -1 1 -1 0 -1 0] and $\underset{\sim}{R}$=[0 4 -2 2 -4 -4 4 2 -4].

We will consider the first three entries of size 4 in $\underset{\sim}{R}$.

(a) Let $\bar{x}_5 = M(x_1, x_4)$, giving

ON x_1 x_2 x_4 x_5				OFF x_1 x_2 x_4 x_5			
0	0	0	0	1	X	0	0
0	X	1	1	X	1	0	0

and the redundancy check gives x_4 redundant, leaving

ON x x x	OFF x x x
0 0 0	1 X 0
0 X 1	X 1 0

which is unchanged from the function at the previous stage, so this is obviously not a useful module.

(b) One of the heuristic rules normally followed is to prefer modules

that introduce variables which have not previously been used (in this case x_2). The only one of the three possible choices for x_5 which meets this criteria is $x_5=M(x_4, x_2)$, giving

ON	x_1	x_2	x_4	x_5
c_1	0	0	0	1
c_2	0	X	1	1

OFF	x_1	x_2	x_4	x_5
c_1	1	0	0	1
c_2	X	1	0	0

The only distance 1 intersection cube is $c_1 \wedge c_3 = \emptyset 001$ and since there are no distance 2 intersection cubes in x_2 and x_4 both are redundant leaving

ON	x_1	x_5
	0	1

OFF	x_1	x_5
	1	1
	X	0

so $f(x_1, x_2, x_3) = \bar{M}(x_1, x_5)$.

(c) The third size 4 entry in R yields $x_5=M(x_4, x_1)$ giving

ON	x_1	x_2	x_4	x_5
	0	0	0	1
	0	X	1	1

OFF	x_1	x_2	x_4	x_5
	1	X	0	0
	0	1	0	1

and x_1 is redundant leaving

ON	x_2	x_4	x_5
	0	0	1
	X	1	1

OFF	x_2	x_4	x_5
	X	0	0
	1	0	1

x_2 \ x_4x_5	00	01	11	10
0	0	1	1	
1	0	0	1	

$\underset{\sim}{F}$=[-1 1 0 1 -1 -1 0 1] and $\underset{\sim}{R}$=[2 4 -4 0 -2 0 -2 4 4].

The two early entries of size 4 lead to

(i) $\bar{x}_6 = M(x_2, x_5)$

ON	x_2	x_4	x_5	x_6
	0	0	1	1
	0	1	1	1
	1	1	1	0

OFF	x_2	x_4	x_5	x_6
	X	0	0	0
	1	0	1	0

Both x_2 and x_5 are redundant leaving

ON x_4 x_6	OFF x_4 x_6
0 1	0 0
1 1	
1 0	

so that $f(x_1, x_2, x_3)=M(x_4, \bar{x}_6)$

(ii) $x_6 = M(x_4, x_2)$

ON x_2 x_4 x_5 x_6	OFF x_2 x_4 x_5 x_6
0 0 1 1	0 0 0 1
X 1 1 1	1 0 0 0
	1 0 1 0

Both x_2 and x_4 are redundant leaving

ON x_5 x_6	OFF x_5 x_6
1 1	0 1
	0 0
	1 0

so that $f(x_1, x_2, x_3)=\bar{M}(\bar{x}_5, x_6)$.

We have a number of different decompositions of the function in terms of the module M. Some of the realizations are not very efficient partly because f is a function of only 3 variables so the range of values for the correlation coefficients is not large enough to give good discrimination in the choice of modules.

To illustrate algorithm 2 it will be applied following the first stage of the above example. We had

ON x_1 x_2 x_4	OFF x_1 x_2 x_4
0 0 0	1 X 0
0 X 1	X 1 0

with $x_4 = \bar{M}(x_1, x_3)$.

$\underset{\sim}{R}$=[0 4 -2 2 -4 -4 4 2 -4]

Consider the first three size 4 entries simultaneously,

viz $x_5=\bar{M}(x_1, x_4)$, $x_6=M(x_4, x_2)$, $x_7=M(x_4, x_1)$.

ON	x_1	x_2	x_4	x_5	x_6	x_7
c_1	0	0	0	0	1	1
c_2	0	X	1	1	1	1

OFF	x_1	x_2	x_4	x_5	x_6	x_7
c_3	1	0	0	0	1	0
c_4	1	1	0	0	0	0
c_5	0	1	0	0	0	1

To deduce a minimal set of nonredundant variables it is necessary to consider all the intersection cubes, viz

$c_1 \wedge c_3 = \emptyset 0001\emptyset$	$\{x_1, x_7\}$
$c_1 \wedge c_4 = \emptyset\emptyset 00\emptyset\emptyset$	$\{x_1, x_2, x_6, x_7\}$
$c_1 \wedge c_5 = 0\emptyset 00\emptyset 1$	$\{x_2, x_6\}$
$c_2 \wedge c_3 = \emptyset 0\emptyset\emptyset 1\emptyset$	$\{x_1, x_4, x_5, x_7\}$
$c_2 \wedge c_4 = \emptyset 1\emptyset\emptyset\emptyset$	$\{x_1, x_4, x_5, x_6, x_7\}$
$c_2 \wedge c_5 = 01\emptyset\emptyset\emptyset 1$	$\{x_4, x_5, x_6\}$

The sets following the intersection cubes are the sets of variables which the intersection cubes dictate cannot all be redundant e.g. c_2 c_5 indicates we must retain at least one of x_4, x_5, x_6. In this case it is obvious that the minimal size nonredundant sets of variables are $\{x_1, x_6\}$ and $\{x_6, x_7\}$. The general problem at this stage is a set covering problem which can be attacked by any of the prime implicant table techniques. In this example the two possibilities give:

ON	x_1	x_6
	0	1

OFF	x_1	x_6
	1	1
	1	0
	0	0

$f(x_1, x_2, x_3)=\bar{M}(x_1, x_6)$

ON	x_6	x_7
	1	1

OFF	x_6	x_7
	1	0
	0	0
	0	1

$f(x_1, x_2, x_3)=\bar{M}(x_6, \bar{x}_7)$

The former is the better solution and would have been chosen by the algorithm since it uses fewer new variables.

Note that although algorithm 2 introduces more extra variables than algorithm 1 we never apply the transform to this n+m variable problem and, because of the greater scope for redundancy this approach will tend to converge to a solution faster than algorithm 1. For some functions all the correlation coefficients are small and of equal magnitude usually indicating the inappropriateness of the module under consideration. It is also clear that for certain functions it is not possible to deduce a good initial module using this approach.

3. Examples

Two examples will be considered to illustrate the algorithms. Some rows of the transforms which would have been included in a computer implementation will be omitted here in an attempt to keep the examples to a reasonable size. For a similar reason the number of variables in the examples is kept to a maximum of 5. Neither algorithm is followed exclusively, rather several alternate paths through the examples are discussed. $\underset{\sim}{F}$ and $\underset{\sim}{R}$ are written as row vectors when this is convenient.

Example 3.1

The function to be realized is totally specified and given by

ON x_1	x_2	x_3	x_4	OFF x_1	x_2	x_3	x_4
1	1	X	X	0	0	X	X
X	1	1	0	0	X	0	X
1	X	1	X	0	X	X	1
1	X	X	0	X	0	0	1

The operator module used is one based on a sole sufficient operator of Wesselkamper [9] which is defined by

$$S(x,y,z) = \begin{cases} z & \text{if } x=y \\ x & \text{if } x\neq y \end{cases}$$

It is clear that $S(x,y,z) = S(z,y,x)$. We shall construct a transform giving the correlation with all the possible S functions. Rows corresponding to S functions with one or two variables constant will be omitted in the example.

R_{ij}^{k} will be used to denote the correlation coefficient corresponding to $S(x_i, x_k, x_j)$.

Since $S(x,y,z) = S(z,y,x)$ and $S(x,\bar{y},z)$ is the majority function there are only 4 functions to consider in the transform for any group of three variables, e.g. for x_1, x_2, x_3, these are $S(x_1, x_2, x_3)$, $S(x_2, x_3, x_4)$, $S(x_3, x_1, x_2)$, and $S(x_1, \bar{x}_2, x_3)$. All other possible substitutions of these three variables lead to functions that are equal to one of the above or its inverse.

Coefficients corresponding to $S(x_i, \bar{x}_j, x_k)$ will be denoted by R_{ijk}.

We have

$$\underset{\sim}{F} = [-1\ -1\ -1\ -1\ -1\ -1\ 1\ -1\ 1\ -1\ 1\ 1\ 1\ 1\ 1\ 1]$$

and hence

$$\begin{bmatrix} R_{23}^{1} \\ R_{24}^{1} \\ R_{34}^{1} \\ R_{13}^{2} \\ R_{14}^{2} \\ R_{34}^{2} \\ \vdots \end{bmatrix} \begin{bmatrix} 1\ 1\ -1\ -1\ -1\ -1\ -1\ -1\ 1\ 1\ 1\ 1\ 1\ 1\ -1\ -1 \\ 1\ -1\ 1\ -1\ -1\ -1\ -1\ -1\ 1\ 1\ 1\ 1\ 1\ -1\ 1\ -1 \\ 1\ -1\ -1\ -1\ 1\ -1\ -1\ -1\ 1\ 1\ 1\ -1\ 1\ 1\ 1\ -1 \\ 1\ 1\ -1\ -1\ 1\ 1\ 1\ 1\ -1\ -1\ -1\ -1\ 1\ 1\ -1\ -1 \\ 1\ -1\ 1\ -1\ 1\ 1\ 1\ 1\ -1\ -1\ -1\ -1\ 1\ -1\ 1\ -1 \\ 1\ -1\ -1\ -1\ 1\ 1\ 1\ -1\ 1\ -1\ -1\ -1\ 1\ 1\ 1\ -1 \\ \vdots \end{bmatrix} \begin{bmatrix} -1 \\ -1 \\ -1 \\ -1 \\ -1 \\ -1 \\ \vdots \end{bmatrix} = \begin{bmatrix} 4 \\ 6 \\ 4 \\ -4 \\ -4 \\ 4 \\ \vdots \end{bmatrix}$$

$$
\begin{bmatrix} \vdots \\ R^3_{12} \\ R^3_{14} \\ R^3_{24} \\ R^4_{12} \\ R^4_{13} \\ R^4_{23} \\ R_{123} \\ R_{124} \\ R_{134} \\ R_{234} \end{bmatrix} =
\begin{bmatrix}
 & & & & & & & & & & & & & & & \\
1 & 1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & -1 & -1 \\
1 & -1 & 1 & 1 & 1 & -1 & 1 & 1 & -1 & -1 & 1 & -1 & -1 & -1 & 1 & -1 \\
1 & -1 & 1 & 1 & -1 & -1 & 1 & -1 & 1 & -1 & 1 & 1 & -1 & -1 & 1 & -1 \\
1 & 1 & 1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & -1 & -1 & -1 \\
1 & 1 & -1 & 1 & 1 & 1 & -1 & 1 & -1 & 1 & -1 & -1 & -1 & 1 & -1 & -1 \\
1 & 1 & -1 & 1 & -1 & 1 & -1 & -1 & 1 & 1 & -1 & 1 & -1 & 1 & -1 & -1 \\
1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 \\
1 & 1 & 1 & 1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & -1 & -1 & -1 & -1 \\
1 & 1 & 1 & -1 & 1 & 1 & 1 & -1 & 1 & -1 & -1 & -1 & 1 & -1 & -1 & -1 \\
1 & 1 & 1 & -1 & 1 & -1 & -1 & -1 & 1 & 1 & 1 & -1 & 1 & -1 & -1 & -1
\end{bmatrix}
\begin{bmatrix} \vdots \\ 1 \\ -1 \\ 1 \\ -1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} =
\begin{bmatrix} \vdots \\ -4 \\ -4 \\ 4 \\ -12 \\ -12 \\ -4 \\ -12 \\ -4 \\ -4 \\ -4 \end{bmatrix}
$$

Several of the coefficients have equal values resulting from the symmetries within the chosen function. The strategy we will adopt is to consider the three cases corresponding to the entries of 12 in R using both algorithms. For comparison we shall also consider the resulting functions if we chose two of the coefficients with value 4.

(a) Using algorithm 1.

(i) R^4_{12}=-12 so let x_5=S(x_1, x_4, x_2) giving

ON	x_1	x_2	x_3	x_4	x_5		OFF	x_1	x_2	x_3	x_4	x_5
c_1	1	1	X	X	1		c_5	0	0	X	X	0
c_2	X	1	1	0	1		c_6	0	X	X	1	0
c_3	1	X	X	0	1		c_7	X	0	0	1	0
c_4	1	0	1	1	0		c_8	0	1	0	0	1

with distance 1 intersection cubes

$c_1 \wedge c_8$ = Ø1001

$c_2 \wedge c_8$ = 01Ø01

$c_3 \wedge c_8 = \emptyset 1001$

$c_4 \wedge c_7 = 10\emptyset 10$

$c_4 \wedge c_6 = \emptyset 0110$

$c_4 \wedge c_5 = \emptyset 0110$

so that x_2, x_4 and x_5 are all individually redundant. Since there are no distance 2 intersection cubes in x_2 and x_4 both are redundant leaving:

ON x_1	x_3	x_5	OFF x_1	x_3	x_5
1	X	1	0	X	0
X	1	1	X	0	0
1	1	0	0	0	1

which is easily shown by the 3-variable transform to be $S(x_1, \bar{x}_3, x_5)$.

(ii) R^4_{13}=-12 so let $x_5=S(x_1, x_4, x_3)$ giving

ON x_1	x_2	x_3	x_4	x_5	OFF x_1	x_2	x_3	x_4	x_5
1	1	0	1	0	0	0	1	0	1
X	1	1	0	1	0	X	0	X	0
1	X	1	X	1	0	X	X	1	0
1	X	X	0	1	X	0	0	1	0

and a redundancy check reveals that x_3 and x_4 are both redundant leaving a function identical to that in the previous case. This just reflects the symmetry of x_2 and x_3 in the original function.

(iii) R_{123}=-12 so let $x_5=S(x_1, \bar{x}_2, x_3)$ giving

ON x_1	x_2	x_3	x_4	x_5	OFF x_1	x_2	x_3	x_4	x_5
1	1	X	X	1	0	0	X	X	0
X	1	1	0	1	0	X	0	X	0
1	X	1	X	1	0	1	1	1	1
1	0	0	0	0	X	0	0	1	0

and the usual redundancy check reveals that both x_2 and x_3 are redundant leaving

ON x_1 x_4 x_5	OFF x_1 x_4 x_5
1 X 1	0 X 0
X 0 1	0 1 1
1 0 0	X 1 0

which is $S(x_1, x_4, x_5)$.

(b) We consider the same three cases simultaneously using algorithm 2. Let $x_5=S(x_1, x_4, x_2)$, $x_6=S(x_1, x_4, x_3)$, $x_7=S(x_1, \bar{x}_2, x_3)$.

ON	x_1	x_2	x_3	x_4	x_5	x_6	x_7	OFF	x_1	x_2	x_3	x_4	x_5	x_6	x_7
c_1	1	1	1	X	1	1	1	c_9	0	0	0	X	0	0	0
c_2	1	1	X	0	1	1	1	c_{10}	0	0	X	1	0	0	0
c_3	1	1	0	1	1	0	1	c_{11}	0	0	1	0	0	1	0
c_4	X	1	1	0	1	1	1	c_{12}	0	0	X	1	0	0	0
c_5	1	1	X	0	1	1	1	c_{13}	0	X	0	1	0	0	0
c_6	1	X	1	0	1	1	1	c_{14}	0	1	1	1	0	0	1
c_7	1	0	0	0	1	1	0	c_{15}	X	0	0	1	0	0	0
c_8	1	0	1	1	0	1	1	c_{16}	0	1	0	0	1	0	0

To deduce a minimal set of nonredundant variables requires the calculation of all 64 intersection cubes, most of which give redundant conditions. The following sets of variables are all those which these 64 intersection cubes dictate cannot all be redundant. $\{x_1, x_5, x_6\}$, $\{x_1, x_6, x_7\}$, $\{x_1, x_5, x_7\}$, $\{x_1, x_3, x_5\}$, $\{x_1, x_2, x_6\}$, $\{x_1, x_4, x_7\}$, $\{x_2, x_5, x_7\}$, $\{x_3, x_6, x_7\}$, $\{x_4, x_5, x_6\}$.

The minimal sets of nonredundant variables are easily determined by a covering table and are $\{x_1, x_2, x_6\}$, $\{x_1, x_3, x_5\}$, $\{x_1, x_4, x_7\}$,

$\{x_1, x_5, x_6\}$, $\{x_1, x_5, x_7\}$, $\{x_1, x_6, x_7\}$, $\{x_2, x_5, x_7\}$, $\{x_3, x_6, x_7\}$, $\{x_4, x_5, x_6\}$, $\{x_5, x_6, x_7\}$.

The solutions selected by algorithm 2 would be the first three sets listed since these have the smallest number of new variables. These are exactly the solutions deduced in (a).

(c) We investigate two low value correlation coefficients to see what sort of realizations they lead to.

(i) $R^3_{14}=-4$ and let $x_5=S(x_1, x_3, x_6)$

ON	x_1	x_2	x_3	x_4	x_5	OFF	x_1	x_2	x_3	x_4	x_5
c_1	1	1	0	1	1	c_6	0	X	0	0	0
c_2	X	1	1	0	0	c_7	0	X	0	1	1
c_3	1	X	1	0	0	c_8	0	0	1	0	0
c_4	1	X	1	1	1	c_9	0	X	1	1	0
c_5	1	X	0	0	1	c_{10}	X	0	0	1	1

It is easily seen that none of x_1, x_2, x_3, x_4 is redundant since

$c_1 \wedge c_7 = \emptyset 1011$

$c_1 \wedge c_{10} = 1\emptyset 011$

$c_2 \wedge c_6 = 01\emptyset 00$

$c_2 \wedge c_9 = 011\emptyset 0$

so the only redundant variable is x_5 and its introduction does not appear to have been useful.

(ii) Consider $R_{134}=-4$ and let $x_5=S(x_1, \bar{x}_3, x_4)$.

ON	x_1	x_2	x_3	x_4	x_5	OFF	x_1	x_2	x_3	x_4	x_5
c_1	1	1	X	1	1	c_5	0	0	X	0	0
c_2	0	1	1	0	0	c_6	0	X	0	X	0

c_3	1	X	1	X	1	c_7	0	X	1	1	1
c_4	1	X	0	0	0	c_8	1	0	0	1	1

and we have

$$c_3 \wedge c_7 = \emptyset X111$$

$$c_1 \wedge c_8 = 1\emptyset011$$

$$c_2 \wedge c_6 = 01\emptyset00$$

so that x_1, x_2, and x_3 are not redundant. However x_4 is redundant, its removal resulting in

ON x_1	x_2	x_3	x_5	OFF x_1	x_2	x_3	x_5
1	1	X	1	0	0	X	0
0	1	1	0	0	X	0	0
1	X	1	1	0	X	1	1
1	X	0	0	1	0	0	1

and this function turns out to be rather more difficult to attack than the original function.

However on occasion the choice of a small value may lead to a reasonable solution, e.g. $x_5=S(x_2, \bar{x}_3, x_5)$ gives a good solution although $R_{234}=-4$.

Example 3.2

A 5-variable totally specified function defined by the arrays below and illustrated in the Karnaugh map.

ON x_1	x_2	x_3	x_4	x_5	OFF x_1	x_2	x_3	x_4	x_5
X	X	1	X	1	X	X	0	X	0
X	X	X	1	1	X	X	X	0	0
1	X	1	1	X	X	0	0	0	X
1	1	X	X	1	0	X	0	0	X
X	1	1	1	X	0	0	X	X	0

x_2x_3 \ $x_1x_4x_5$	000	001	011	010	100	101	111	110
00	0	0	1	0	0	0	1	0
01	0	1	1	0	0	1	1	1
11	0	1	1	1	0	1	1	1
10	0	0	1	0	0	1	1	0

The module used is a 3-input majority function, $M(i,j,k)=x_ix_j+x_ix_k+x_jx_k$. R_{ijk} will be used to denote the correlation coefficient with $M(i,j,k)$. Neither inverted inputs nor constant inputs will be considered in the example in order to avoid it becoming too complex. For 5 variables there will be 10 possible 3-input majority functions so the transform has 10 rows.

Now

$$\underset{\sim}{R} = \begin{bmatrix} R_{123} \\ R_{124} \\ R_{125} \\ R_{134} \\ R_{135} \\ R_{145} \\ R_{234} \\ R_{235} \\ R_{245} \\ R_{345} \end{bmatrix} = \underset{\sim\sim}{TF} = \begin{bmatrix} 12 \\ 12 \\ 12 \\ 12 \\ 20 \\ 20 \\ 12 \\ 20 \\ 20 \\ 28 \end{bmatrix}$$

where $\underset{\sim}{T}$ is defined as

$$\begin{bmatrix}
-1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 \\
-1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 \\
-1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 \\
-1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 \\
-1 & -1 & -1 & -1 & -1 & 1 & -1 & 1 & -1 & -1 & -1 & -1 & -1 & 1 & -1 & 1 \\
-1 & -1 & -1 & 1 & -1 & -1 & -1 & 1 & -1 & -1 & -1 & 1 & -1 & -1 & -1 & 1 \\
-1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 \\
-1 & -1 & -1 & -1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & 1 & 1 & 1 & 1 \\
-1 & -1 & -1 & 1 & -1 & -1 & -1 & 1 & -1 & 1 & 1 & 1 & -1 & 1 & 1 & 1 \\
-1 & -1 & -1 & 1 & -1 & 1 & 1 & 1 & -1 & -1 & -1 & 1 & -1 & 1 & 1 & 1
\end{bmatrix} \quad \ldots\ldots$$

$$\begin{bmatrix}
-1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
-1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
-1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
-1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 \\
-1 & 1 & -1 & 1 & 1 & 1 & 1 & 1 & -1 & 1 & -1 & 1 & 1 & 1 & 1 \\
-1 & 1 & 1 & 1 & -1 & 1 & 1 & 1 & -1 & 1 & 1 & 1 & -1 & 1 & 1 & 1 \\
-1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 \\
-1 & -1 & -1 & -1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & 1 & 1 & 1 & 1 \\
-1 & -1 & -1 & 1 & -1 & -1 & -1 & 1 & -1 & 1 & 1 & 1 & -1 & 1 & 1 & 1 \\
-1 & -1 & -1 & 1 & -1 & 1 & 1 & 1 & -1 & -1 & -1 & 1 & -1 & 1 & 1 & 1
\end{bmatrix}$$

and

$$\underset{\sim}{F} = [-1\ -1\ -1\ -1\ \ 1\ -1\ \ 1\ -1\ -1\ -1\ \ 1\ -1\ \ 1\ \ 1\ \ 1\ \ 1$$
$$-1\ -1\ -1\ \ 1\ -1\ \ 1\ \ 1\ \ 1\ -1\ \ 1\ -1\ \ 1\ -1\ \ 1\ \ 1\ \ 1]$$

(a) For algorithm 1 the largest coefficient is R_{345} so we let x_6= M(3,4,5) giving

ON	x_1	x_2	x_3	x_4	x_5	x_6		OFF	x_1	x_2	x_3	x_4	x_5	x_6
c_1	X	X	1	X	1	1		c_8	X	X	0	X	0	0

c_2	X	X	X	1	1	1	c_9	X	X	X	0	0	0
c_3	1	X	1	1	X	1	c_{10}	X	0	0	0	X	0
c_4	1	1	0	0	1	0	c_{11}	0	X	0	0	X	0
c_5	1	1	1	X	1	1	c_{12}	0	0	1	1	0	1
c_6	1	1	X	1	1	1	c_{13}	0	0	0	X	0	0
c_7	X	1	1	1	X	1	c_{14}	0	0	X	0	0	0

Note that this six variable function is only partially specified.

The distance 1 intersection cubes are

$c_1 \wedge c_{12}$ = 0011Ø1

$c_2 \wedge c_{12}$ = 0011Ø1

$c_3 \wedge c_{12}$ = Ø01101

$c_4 \wedge c_8$ = 1100Ø0

$c_4 \wedge c_7$ = 1100Ø0

$c_4 \wedge c_{10}$ = 1Ø0010

$c_4 \wedge c_{11}$ = Ø10010

so that x_3, x_4, and x_6 are singly redundant. Since there are no distance 2 intersection cubes in x_3 and x_4 both are redundant and may be deleted giving

ON x_1	x_2	x_5	x_6	OFF x_1	x_2	x_5	x_6
X	X	1	1	X	X	0	0
1	X	X	1	X	0	X	0
X	1	X	1	0	X	X	0
1	1	1	0	0	0	0	1

This 4-variable function which remains is treated similarly except that the transform only has 4 rows in it, namely

$$\underset{\sim}{R} = \begin{bmatrix} R_{125} \\ R_{126} \\ R_{156} \\ R_{256} \end{bmatrix} = \underset{\sim\sim}{TF} = \begin{bmatrix} 4 \\ 12 \\ 12 \\ 12 \end{bmatrix}$$

where

$$\underset{\sim}{T} = \begin{bmatrix} -1 & -1 & -1 & -1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & 1 & 1 & 1 & 1 \\ -1 & -1 & -1 & -1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & 1 & 1 & 1 & 1 \\ -1 & -1 & -1 & 1 & -1 & -1 & -1 & 1 & -1 & 1 & 1 & 1 & -1 & 1 & 1 & 1 \\ -1 & -1 & -1 & 1 & -1 & 1 & 1 & 1 & -1 & -1 & -1 & 1 & -1 & 1 & 1 & 1 \end{bmatrix}$$

and $\underset{\sim}{F} = [-1\ -1\ -1\ 1\ -1\ 1\ -1\ 1\ -1\ 1\ -1\ 1\ -1\ 1\ 1\ 1]$

which would indicate the equal potential usefullness of M(1,2,6), M(1,5,6), and M(2,5,6). This is hardly surprising since a detail look at the function reveals a symmetric relation between the variables. We consider only x_7 = M(1,2,6) for this reason. This gives

ON	x_1	x_2	x_5	x_6	x_7	OFF	x_1	x_2	x_5	x_6	x_7
c_1	0	0	1	1	0	c_5	1	1	0	0	1
c_2	1	X	X	1	1	c_6	X	0	X	0	0
c_3	X	1	X	1	1	c_7	0	X	X	0	0
c_4	1	1	1	0	1	c_8	0	0	0	1	0

with the distance 1 intersection cubes

$c_1 \wedge c_6 = 001\emptyset0$

$c_1 \wedge c_7 = 001\emptyset0$

$c_1 \wedge c_8 = 00\emptyset10$

$c_2 \wedge c_5 = 110\emptyset1$

$c_3 \wedge c_5 = 110\emptyset1$

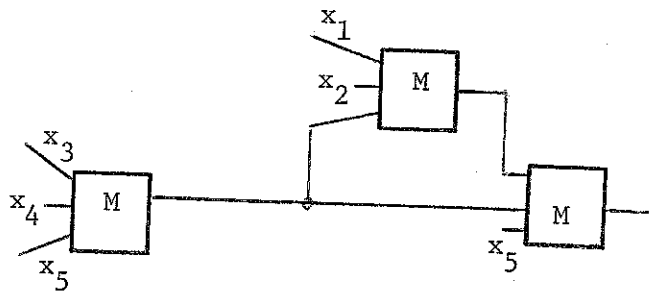
$c_4 \wedge c_5 = 11\emptyset01$

Since there are no distance 2 intersection cubes in x_1 and x_2 both of them are redundant giving

ON x_5	x_6	x_7	OFF x_5	x_6	x_7
1	1	0	0	0	1
X	1	1	X	0	0
1	0	1	0	1	0

which is M(5,6,7).

This gives the module circuit below



We will briefly consider the effect of an alternative choice which could have been made.

(i) If x_7 = M(1,2,5) (corresponding to the smaller correlation coefficient).

The only redundant variable turns out to be x_7 itself. If we persist with this approach and apply the original 5-variable transform to this function (retaining x_7) the resulting high value correlation coefficients are R_{126}, R_{156}, and R_{256} and the resulting circuits are similar to the one above, with x_7 remaining redundant at the end.

(ii) One further possibility is to only remove x_4 rather than both x_3 and x_4 after the first stage of the example.

ON x_1	x_2	x_3	x_5	x_6	OFF x_1	x_2	x_3	x_5	x_6
X	X	X	1	1	X	X	X	0	0

1	X	1	X	1		X	0	0	X	0
1	1	0	1	0		0	X	0	X	0
X	1	1	X	1		0	0	1	0	1

The 5 variable transform can be applied to this function giving:

$$\begin{bmatrix} R_{123} \\ R_{125} \\ R_{126} \\ R_{135} \\ R_{136} \\ R_{156} \\ R_{235} \\ R_{236} \\ R_{256} \\ R_{356} \end{bmatrix} = \begin{bmatrix} 8 \\ 8 \\ 16 \\ 12 \\ 12 \\ 20 \\ 12 \\ 12 \\ 20 \\ 20 \end{bmatrix}$$

The high value coefficients are R_{156}, R_{256}, and R_{356}. The first two avoid x_3 and will lead to the earlier circuit.

To specifically include x_3 we choose $x_7 = M(3,5,6)$ giving

ON x_1	x_2	x_3	x_5	x_6	x_7		OFF x_1	x_2	x_3	x_5	x_6	x_7
X	X	X	1	1	1		X	X	X	0	0	0
1	X	1	X	1	1		X	0	0	X	0	0
1	1	0	1	0	0		0	X	0	X	0	0
X	1	1	X	1	1		0	0	1	0	1	1

The redundancy check indicates that x_3, x_6 and x_7 are singly redundant and that x_3 and x_6 may both be deleted. This results in a function which is identical to the one we had after deleting x_3 and x_4 in the first stage of the example, so all this accomplishes is the introduction of an extra redundant module into the circuit.

(b) Returning to the first stage to apply algorithm 2 there were four entries of 20 and one of 28 in the initial $\underset{\sim}{R}$. Including all of these we let $x_6 = M(3,4,5)$, $x_7 = M(2,4,5)$, $x_8 = M(2,3,5)$, $x_9 = M(1,4,5)$ and $x_{10} = M(1,3,5)$.

The extra variables require some expansion of the specification arrays giving:

ON	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
c_1	0	0	1	0	1	1	0	1	0	1
c_2	1	1	1	X	1	1	1	1	1	1
c_3	0	1	1	0	1	1	1	1	0	1
c_4	1	0	1	0	1	1	0	1	1	1
c_5	X	X	1	1	1	1	1	1	1	1
c_6	1	1	X	1	1	1	1	1	1	1
c_7	0	1	0	1	1	1	1	1	1	0
c_8	1	0	0	1	1	1	1	0	1	1
c_9	0	0	0	1	1	1	1	0	1	0
c_{10}	1	1	1	1	X	1	1	1	1	1
c_{11}	1	0	1	1	0	1	0	0	1	1
c_{12}	1	1	0	0	1	0	1	1	1	1
c_{13}	0	1	1	1	0	1	1	1	0	0

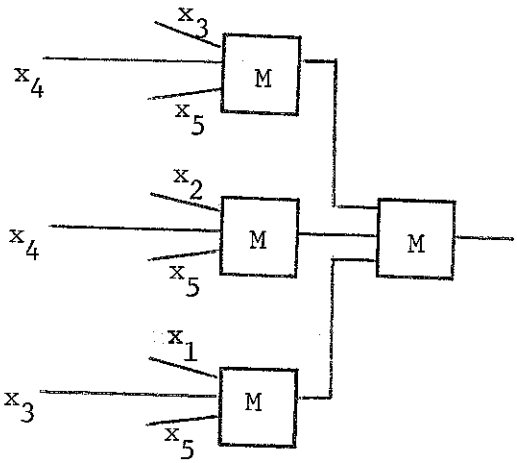
OFF	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
c_{14}	1	1	0	1	0	0	1	0	1	0
c_{15}	0	0	0	X	0	0	0	0	0	0
c_{16}	1	0	0	1	0	0	0	0	1	0
c_{17}	0	1	0	1	0	0	1	0	0	0
c_{18}	X	X	0	0	0	0	0	0	0	0
c_{19}	0	0	X	0	0	0	0	0	0	0
c_{20}	1	0	1	0	0	0	0	0	0	1
c_{21}	0	1	1	0	0	0	0	1	0	0
c_{22}	1	1	1	0	0	0	0	1	0	1
c_{23}	0	0	0	0	X	0	0	0	0	0
c_{24}	0	1	0	0	1	0	1	1	0	0
c_{25}	0	0	1	1	0	1	0	0	0	0
c_{26}	1	0	0	0	1	0	0	0	1	1

and the 169 intersection cubes yield the following sets of variables which cannot all be redundant:

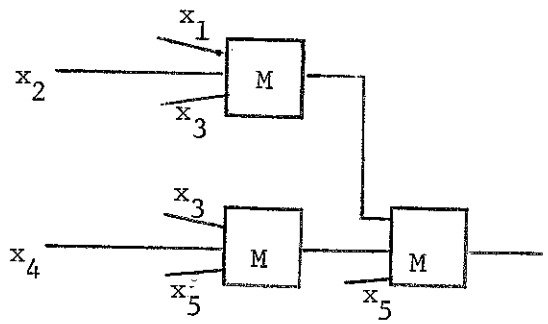
$\{x_3, x_6, x_{10}\}$, $\{x_4, x_6, x_9\}$, $\{x_1, x_9, x_{10}\}$, $\{x_3, x_6, x_8\}$, $\{x_4, x_6, x_7\}$, $\{x_2, x_7, x_8\}$, $\{x_1, x_5, x_6, x_8\}$, $\{x_1, x_2, x_5, x_6\}$, $\{x_1, x_5, x_6, x_7\}$, $\{x_3, x_4, x_5, x_6\}$, $\{x_3, x_5, x_7, x_9\}$, $\{x_4, x_5, x_8, x_{10}\}$, $\{x_5, x_6, x_8, x_{10}\}$, $\{x_5, x_6, x_7, x_9\}$, $\{x_5, x_6, x_8, x_9\}$, $\{x_5, x_6, x_7, x_{10}\}$, $\{x_2, x_4, x_6, x_8, x_9\}$, $\{x_5, x_7, x_8, x_9, x_{10}\}$, $\{x_2, x_5, x_6, x_{10}\}$.

From these using a simple covering table it transpires that the only two minimal sets of nonredundant variables are $\{x_6, x_7, x_{10}\}$ and $\{x_6, x_8, x_9\}$ giving solutions M(6,7,10) and M(6,8,9).

As we would expect the solutions from algorithm 2 use fewer levels but more modules at each level, the circuit for the first solution being as shown below.



The simplest solution, which was not deduced by these methods, appears to be that shown below.



References

[1] C.R. Edwards, "The application of the Rademacher-Walsh transform to Boolean function classification and threshold logic synthesis," Trans. IEEE, Vol. C-24, pp. 48-62, 1975.

[2] C.R. Edwards, "Digital synthesis under function symmetries and mapping methods," (forthcoming).

[3] S.L. Hurst, "Logical Processing of Digital Signals," Crane Russak, New York (forthcoming).

[4] S.L. Hurst, "Application of Chow parameters and Rademacher-Walsh matrices in the synthesis of binary functions," Comp. J., Vol. 16, pp. 165-173, 1973.

[5] S.L. Hurst, "The detection of symmetries in Boolean functions using Rademacher-Walsh spectral coefficients," (forthcoming).

[6] D.M. Miller and J.C. Muzio, "A fast method for determining the two-place decompositions of a binary function," Proc. 4th Man. Conf. on Num. Math., pp. 293-308, 1974.

[7] J.C. Muzio, "The logical decomposition of switching functions by the construction of redundant variables," Technical Report, Virginia Poly. Inst., Dept. Comp. Sci., 1977.

[8] J.C. Muzio, "The direct evaluation of large magnitude spectral coefficients," Technical Report, Virginia Polytechnic Inst., Dept. Computer Science, 1977

[9] T.C. Wesselkamper, "A sole sufficient operator," Notre Dame J. Formal Logic, Vol. 16, pp. 86-88, 1975.