Simple Table 方法에 의한 論理函數 最小化의 新方法

論 文
28—6—3

A New Approach to the Minimization of Switching Functions by the Simple Table Method

黃 熙 隆
(Hee Yeung Hwang)

Abstract

This paper is concerned with minimization process of the binary logic function. This paper describes an algorithm called the SIMPLE TABLE METHOD that well suited to minimization a switching function of any number of variables. For the Simple Table construction, a theorem based upon the numerical properties of the logic function is derived from the relationships governing minterms of the given function. Finally the minimal sum of products can be obtained in terms of the Direct Method or the Indirect Method from the table and table characteristics derived from the Simple Table. The properties and table characteristics used in this paper are described. All the minterms of a switching function are manipulated only by decimal numbers, not binary numbers. Some examples are used as a vehicle to guide the readers who are familiar with the Karnaugh map and Quine-McCluskey tabular method to this New method. These examples not only treat how to handle Don't Care miterms but also show the multiple output functions.

Index terms: prime implicants (PI's), minterms, directions of adjacency (DA), don't care minterms, required adjacency directions (RAD), RAD tree, tuple coupling, intersection table, cube, vertices, Simple Table, DA table.

I. INTRODUCTION

The basc tenet of the switching-function minimization problem, when expressed in AND/OR terms, is $AB+A\bar{B}=A(B+\bar{B})=A(1)=A$ (1) It has been a difficult task to identify such combinable pairings and obtain a minimal sum of products of the given switching-function as the number of switching variables increases.

A wide variety of techniques for identifying and selecting prime implicants and/or prime implicates have been developed, such as [1]~[14], [19]~[22]. Almost all of the methods listed in the references treat the minimization process as two separate parts. First, all of the prime implicants (PI's) that apply to the function are generated. This is PI identification. Second, The minimal set of PI's that best covers the function is chosen from the larger set of all PI's. This is PI selection. This paper also handles the problem in a similar way.

*正會員：서울工大 電子計算機 工學科 副敎授(當學會編修委員)
接受日字；1979年 5月 7日

For the first part, The Simple Tables generating all the prime implicants of a switching function of any number of variables is introduced. In the present paper, the tables of 1's number of minterms as in the Quine-McCluskey tablar mehtod [2], Hwa [12] and other [4],[6] is not used at all. But in this correspondence, all minterms are manipulated only by the decimal numbers.

Theorems are presented to establish easily the Simple Table configuration that are necessary and sufficient for existence of all combinable pairs and prime implicants.

For the second part, minimal sum of products are determined by a simple examination of a Simple Table in terms of the Direct Method or the Indirect Method. In the Indirect Method, since all the directions of adjacency (DA) relations among minterms and all degree of consensus of [6] are expressed completely in the DA table that derived from the Simple Table, hence the PI selection process becomes evidently easier than [3],[4],[9],[14], The minimal sum of products can be obtained well known methods such as Karnaugh Map method and the Quine-McCluskey tabular method. Manual application of all minimization algorithms of them is possible only when the number of variables is small. But in this new method, the inability for men to handle a switching function of large variables clearly reduced.

This Simple Table Method is suitable both for solving a problem by hand or programming it on a digital computer. The memory requirement and a large number of basic operations in this method are much less than any existng known method.

Applications of the Simple Table method to Don't Care minterms and Multioutput functions are also fulfilled systematically without difficulty.

II. THEORY

The basis of minimization theory for the Simple Table Method is as follows.

THEORM 1.

Let $B=(b_n,\cdots b_2,\ b_1)$ and $T=(t_n,\cdots t_2,\ t_1)$ be the base minterm and test minterm where b_i and t_i $(i=1,2,\cdots n)$ are their logic values (0,1). If two minterms B and T are the same in every position but i-th (i.e. $b_i+t_i=1$ for $i=1,2,\cdots n$), canonical form of Boolean function F reduced to one form;

$$F=B+T=(t_n,\ t_{n-1},\cdots,\ -,\ t_{i-1},\cdots t_2,\ t_1)$$
$$=(b_n,\ b_{n-1},\cdots,\ -,b_{i-1},\cdots b_2,\ b_1),$$

where symbol '−' represents the eliminated bit.

Theorem 1 is identical to (1).

THEOREM 2.

Let the decimal integers assigned to the minterms B and T be Bd, Td respectively and the difference R, $(R=T_d-B_d)$, be equal to 2^i $(i=0, 1,2,\cdots n-1)$.

Iff the integer part of $Bd/(Td-Bd)$ is even number, two minterms Td and Bd satisfy the theorem 1.

PROOF:

There are only two cases stisfying $Td-Bd=2^i$ $(i=0,1,\cdots n-1)$ among minterms without loss of generality. They are as follows.

CASE 1.

$$Td=(t_n,\cdots t_{i+2},1,\ t_i,\cdots t_2,\ t_1)$$
$$Bd=(b_n,\cdots b_{i+2},\ 0,\ b_i,\cdots b_2,\ b_1)$$

Where $t_j=b_j$ for $j=1,2,\cdots i,\ i+2\cdots n$.

CASE 2.

$$T=(t_n,\cdots t_{i+3},1,\ 0,\ t_i,\cdots t_2,\ t_1)$$
$$B=(b_n,\cdots b_{i+3},\ 0,\ 1,\ b_i,\cdots b_2,\ b_1)$$

Where $t_j=b_j$ for $j=1,\cdots i,\ i+3,\cdots n$.

If two minterms Td and Bd satisfy the Theorem 1, it corresponds to the CASE 1. The integer part of $Bd/(Td-Bd)$ becomes even number as follows.

$$Bd=b_n*2^{n-1}+b_{n-1}*2^{n-2}+\cdots b_{i+2}*2^{i+1}+0*2^i+b_i*2^{i-1}+\cdots+b_2*2^1+b_1*2^0$$

$$Bd/(Td-Bd)=Bd/2^i$$
$$=b_n*2^{n-i-1}+b_{n-1}*2^{n-i-2}+\cdots+b_{i+2}*2+b_i*2^{-1}+\cdots+b_2*2^{1-i}+b_1*2^{-i}$$
$$=\underbrace{2(b_n*2^{n-i-2}+b_{n-1}*2^{n-i-3}+\cdots+b_{i+2})}_{\text{Even number of Integer part.}}+b_i*2^{-1}+\cdots+b_2*2^{1-i}+b_1*2^{-i}$$

Conversely, if two minterms Td and Bd whose difference is equal to 2^i do not satisfy the theorem 1, it corresponds to the CASE 2 clearly, And then in CASE 2, the integer part of $Bd/(Td-Bd)$ becomes odd number as follows.

$$Bd=b_n*2^{n-1}+b_{n-1}*2^{n-2}+\cdots+0*2^{i+1}+1*2^i+b_i*2^{i-1}+\cdots+b_2*2^1+b_1*2^0$$

$$Bd/2^i=b_n*2^{n-i-1}+b_{n-1}*2^{n-i-2}+\cdots+1+b_i*2^{-1}+\cdots+b_2*2^{1-i}+b_1*2^{-i}$$

$$=\underbrace{2(b_n*2^{n-i-1}+b_{n-1}*2^{n-i-2}+\cdots+b_{i+3})}_{\text{Odd number of Integer part.}}+1+b_i*2^{-1}+\cdots+b_2*2^{1-i}+b_1*2^{-i}$$

Q.E.D.

III. CONSTRUCTION OF THE SIMPLE TABLE

Consider a Boolean function defined by its truth table.

Table 1 shows the truth table for a function $F(X1,X2,X3,X4)$ for $n=4$ variables as an example.

From the Table 1, the Simple Table is generated as shown in Table 2.

This Table shows that all the possible prime implicants (which satisfy theorem 2) are represented.

The procedurea are divided into the following steps to construct the Simple Table.

Step 1: Select all minterms including don't care minterms and put them in ascending order as shown in Table 2.

Let the base minterms be arranged vertically.

Let the test minterms be arranged horizontally.

Step 2: Let R(representing the difference between Td and Bd be $Td-Bd$ for each Bd and all $Td \geqq Bd$.

Step 3: If R satisfies theorem 2, fill the corresponding cell with the value R and encircle it.

Step 4: If the Simple Table has Don't care minterm row as shown in Table 2,

These minterms are designated on the Table by the symbol "V" in their associated decimal integer columns.

Step 5: In order to check all minterms covered by the prime implicants, the check list is provided as shown in table 2.

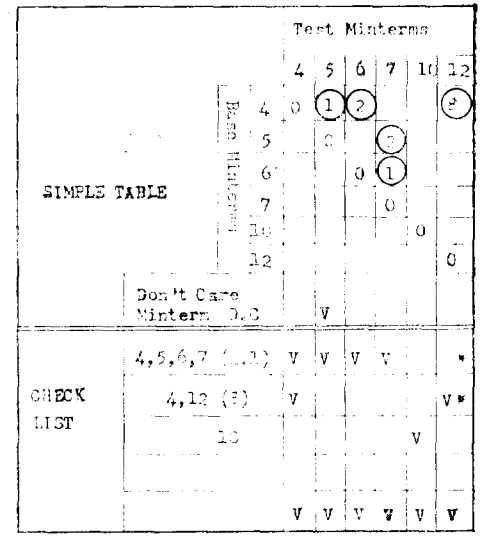
Table 1.

Truth Table for a function $F(X_1,X_2,X_3X_4)=\Sigma m(4,6,10,12)+d(5)$

	X_1	X_2	X_3	X_4	F
0	0	0	0	0	0
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	d
6	0	1	1	0	1
7	0	1	1	1	1
8	1	0	0	0	0
9	1	0	0	1	0
10	1	0	1	0	1
11	1	0	1	1	0
12	1	1	0	0	1
13	1	1	0	1	0
14	1	1	1	0	0
15	1	1	1	1	0

Table 2.

Construction of the Simple Table and Check List.



* represents the reduced prime implicants.

IV. CONSTRUCTION OF THE CHECK LIST

In the Simple Table constructed above, there are all possible prime implicants which cover the Boolean function.

The Check List can be generated from the Simple Table.

In this paper, two methods will be proposed to generate the Check List. One is the Direct Method and the other is the Indirect Method. Each property of the Simple Table leads to each rule that is used for the Direct Method. The problem of finding the minimal sum of products from the Check List is discussed in details in the literature [2]& [9].

1) Direct Method

Using the properties of the Simple Table, all the possible prime implicants are selected from the Simple Table directly and check in Check List by rules explained in section (B) of IV.

Properties and ckeck rules are introduced here and these are applied to the direct method.

(A) Procedure

To select the possible prime implicants and check in the Check List, the following Steps are taken.

Step 1: Count the number of each R which is the same and let it be K. Find the maximum value of N such that $2^N \leq K < 2^{N+1}$

Step 2: Using table properties and check rules (as shown in section (B) in IV) which derived from the Simple Table and showed the standard form for each N, search for all the possible prime implicants which are suitable for every check rule in order to obtain the minimal sum of products from the Simple Table.

Step 3: When the prime implicants are found, check in the check-list the minterms which are covered by these prime implicants.

Step 4: If all the minterms are covered, go to Step 5. Otherwise let N be $N-1$ and go to Step 2.

Step 5: This algorithm is completed.

(B) General properties of the Simple Table and check rules

In a given Boolean function of n variables such that $F(X1, X2, \cdots Xn)=\Sigma m\,(A,B,C,D\cdots)$ where the alphabet letters are assigned to denote the decimal minterm numbers, the general properties of the Simple Table are constructed as follows.

Property 1.

Let two minterms be A,B and assume that they satisfy the following condition;

$A,B\ (2^i)$

where $A<B$.

The Simple Table is built as shown in Fig 1 for two two minterms.

This property is applied for $N=0$.

To check minterms which are covered, following Rule 1 is presented.

Rule 1:

Move 2^i leftward and upward and check the corresponding minterms.

Then mark "V" in the Check List.

		Test minterms	
Base minterms		A	B
	A	0	2^i
	B		0
A,B(2^i)		V	V *

Fig 1. The Simple Table for $N=0$

Property 2.

If there exist four minterms satisfying the following conditions;

$A,B(2^i)$ $A,C(2^j)$

$C,D(2^i)$ $B,D(2^j)$ where $A<B<C<D$ and $2^i<2^j$.

The Simple Table is constructed as shown in Fig. 2 for four minterms.

This property is applied for $N=1$.

To check minterms covered by A,B,C,D (2^i, 2^j) term the following rule 2 is given.

Rule 2.

From the Fig. 2 the following five rules are

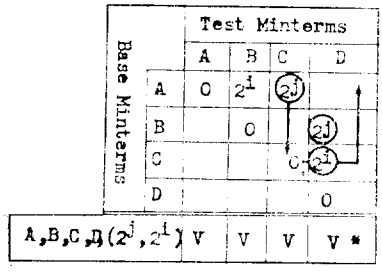


Fig. 2 The Simple Table for $N=1$ (2-cube)

obtained as follows.

1) Select two 2^i and moves the left 2^i down until 0 is found.
2) If 0 is found, move right to the right 2^i column.
3) If there exists 2^i, four terms are reduced to one.
4) To check the four terms, move the two 2^i'S selected above leftward and upward
5) Corresponding minterms are checked in the check list.

To extend the property to higher order of N, the characteristies of the Simple Table structure are introduced.

If all the bit position numbers decided by the theorem 2 are given, the set of the greatest value of them constitute diagonal elements and the smaller values of them are arranged in the same form of N-1 case as shown in Fig. 3.

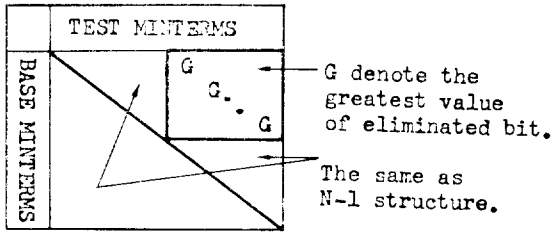


Fig. 3 General structure of the Simple Table for N.

Now that the structure of the Simple Table is generated above, it can be applied for the case of higher order N.

Property 3.

If there are eight minterms which satisfy the following conditions

$A,B(2^i)$ $A,C(2^j)$ $A,E(2^k)$
$C,D(2^i)$ $B,D(2^j)$ $B,F(2^k)$
$E,F(2^i)$ $E,G(2^j)$ $C,G(2^k)$
$G,H(2^i)$ $F,H(2^j)$ $D,H(2^k)$

where $A<B<C<D<E<F<G<H$, and $2^i<2^j<2^k$,

the Simple Table is obtained as shown in Fig. 4.

This property is applied for $N=2$, and eight minterms are reduced to one prime implicant $A\sim H$ $(2^i, 2^j, 2^k)$.

To check the eight minterms, the following ruls 3 is given.

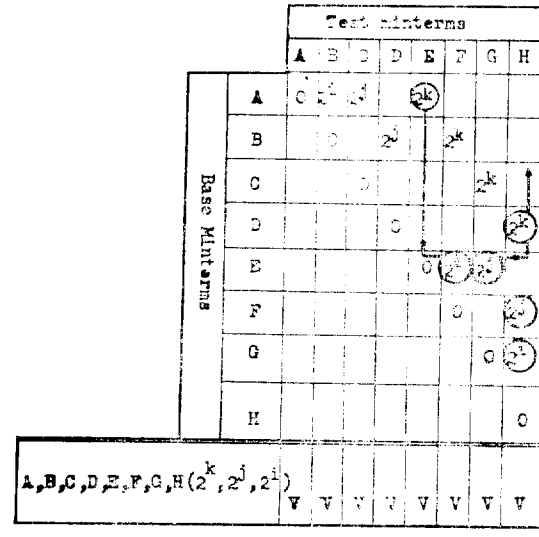


Fig. 4. The Simple Table for $N=2^3$ (3-cube)

RULE 3.

1) For the case of $N=2$ the Simple Table of Fig. 4 is obtained by the similar method. From Fig. 4, the following rules and methods are derived.

   Select the leftmost 2^k and move down until 0 appears.
2) If 0 appears, move right to the rightmost 2^k column to find of the same values in the column as appeared in the row.
3) If there are two same values, eight minterms are reduced to one.
4) To check the eight minterms, the following methods are introduced.

   Method 1.

   In above Step 2), select the rightmost value

appeared in the row and move down until 0 appears, then do as like in Rule 2.

Method 2.

Move the 2^j upward and choose the 2^k in the corresponding column. Move the 2^j leftward until 0 appears and then move upward and choose the 2^k in corresponding column.

Method 3.

Move selected the 2^k upward and leftward to check the minterms covered in the Check List below the Simple Table.

Property 4.

If there are sixteen minterms which satisfy the following conditions, the Simple Table is constructed as shown in Fig. 5 from the diagonal elements and appropriate arrangement.

$A,B(2^i)$ $A,C(2^j)$ $A,E(2^k)$ $A,I(2^l)$
$C,D(2^i)$ $B,D(2^j)$ $B,F(2^k)$ $B,J(2^l)$
$E,F(2^i)$ $E,G(2^j)$ $C,G(2^k)$ $C,K(2^l)$
$G,H(2^i)$ $F,H(2^j)$ $D,H(2^k)$ $D,L(2^l)$
$I,J(2^i)$ $I,K(2^j)$ $I,M(2^k)$ $E,M(2^l)$
$K,L(2^i)$ $J,L(2^j)$ $J,N(2^k)$ $F,N(2^l)$
$M,N(2^i)$ $M,O(2^j)$ $K,O(2^k)$ $G,O(2^l)$
$O,P(2^i)$ $N,P(2^j)$ $L,P(2^k)$ $H,P(2^l)$

where $A<B<C<D<E<F<G<H<I<J<K<L<M<N<O<P$ and $2^i<2^j<2^k<2^l$.

To check the sixteen minterms the following Rule 4 is given.

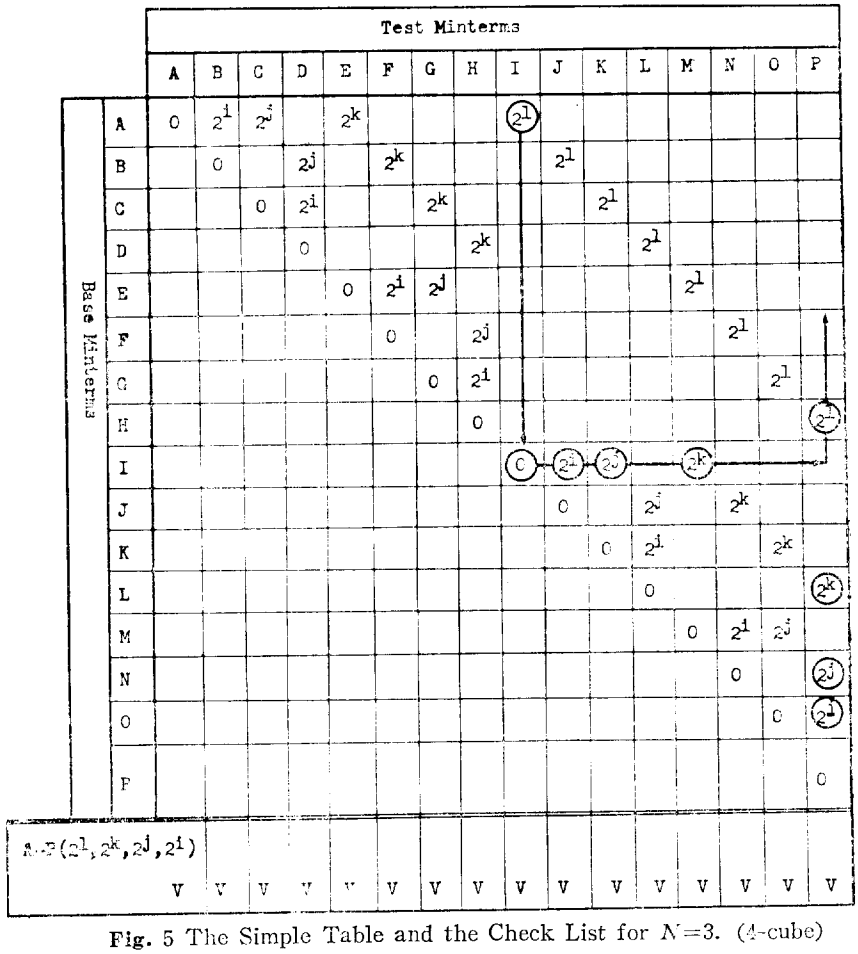


Fig. 5 The Simple Table and the Check List for $N=3$. (4-cube)

RULE 4.

For the case of $N=3$, the Simple Table of Fig. 5 is obtained by the similar methods for $N<3$. From Fig. 5, the following rules and methods are derived.

1) Select the leftmost 2^l and move down until 0 appears.
2) If 0 appears, move right to the rightmost 2^l column to find the same values in the column

as appear in the row.

3) If there are three same values, sixteen minterms are reduced to one.

4) In order to check the sixteen minterms, the following methods are introduced.

Method 1.

In Step 2), select the rightmost value appeared in the row and move down until 0 appears.

Then, do as like in Rule 3 and Rule 2.

Method 2.

Do as in the similiar way (for $2^i, 2^j$, and 2^k) (as presented in RULE 3).

To check the 2^l move the 2^k upward and choose the 2^l in the corresponding column.

Then move the 2^k leftward and move upward if there there exist 0 and choose the 2^l in corresponding column.

Method 3.

Move the selected 2^l upward and leftward.

Check minterms covered in the check list below the Simple Table.

We can extend these rules for $N \geqq 4$, similarly

2) Indirect Method

In order to determine prime implicants easily, anothr table of Directions of Adjacency (DA) can be obtained from Simple Table.

This table is referenced as the DA table in this paper.

(A) Construction of the DA Table (Directions of Adjacency).

The method to construct the DA table is represented in the following Steps using an example.

The steps of constructing the DA Table

Step 1. Draw the DA Table as shown in Fig. 6.

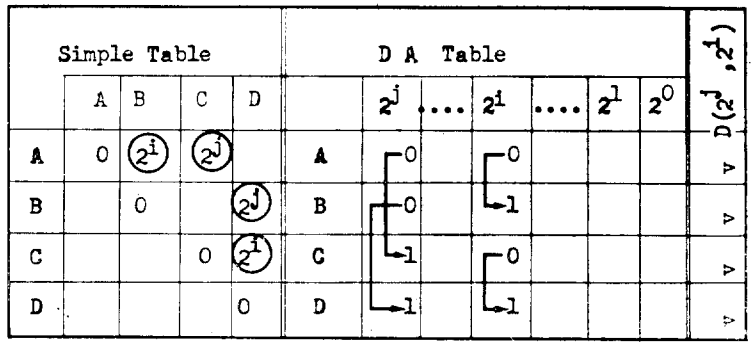


Fig.6 DA table

Step 2. For each base minterm(for example B in Fig.6) in the Simple Table, examining the corresponding row to the rightside of zero difference the checked differences (for example 2^i in Fig. 6) are found. Fill the related cell in the DA Table with symbol 0. For the same test minterm (for example B), checked differences (for example 2^i) in the vertical column can be found and fill the related cell in the DA Table with symbol 1.

Step 3: Every couple of symbols 0 and 1 that are created for each checked difference are connected by anarrowed line as shown in Fig.6

(B) Properties of the DA table.

DA table, obtained from the Simple Table by above steps, has the following properties.

Property 1: Every couple of symbols 0 and 1 for two minterms that are connected with line (for example minterms A and B) means that two minterms are the same except for corresponding bit (in this example 2^i) and two minterms can be reduced to one eliminating that bit.

Property 2: All the possible reducible relations are represented completely in the DA Table. All information for minterms of logic functions can be obtained from the DA Table.

Property 3: The 1-cube, 2-cube, 3-cube and the higher cube relations can be found

easily in the DA Table, which will be described Section (C) later.

Property 4: The symbol 0 in the DA Table means that this minterm has larger one among minterms of the logic function with which can be reduced to one. The symbol 1 in the DA Table means that this minterm has adjacent smaller one as a couple.

Property 5: The DA Table can be made directly without drawing the Simple Table.

Property 6: Symbol 0 and 1's for every minterm's bits of the DA Table are equal to original bits of the same minterm when expressed in binary numbers.

(C) Construction of Check List from DA table

For minimization purposes, like [9], we define that any minterm m_i is ADJACENT to another input minterm m_j, if input combinations that they represent differ only one variable.

In this work the same terminologies such as Directions of adjacency and Required adjacency direction (RAD) in [9] are used for the simplicity. However, all the RAD's relations for minterms of a given switching function are represented by the symbols 0, 1 and the arrowed lines completely in the Table of the Directions of adjacency (DA) without doing the process of seaching one by one as in [3], [9].

In the Table of the Directions of Adjacency, the symbol $[_1{}^0$ in a minterm of a given switching function represents one RAD relation with the other minterms. Since all RAD relations of a given minterms are shown in the Table of DA, the procedure of determining the minimal sum of products becomes much easier than other works [1] to [14] and the concept of RAD tree is somewhat different from the conventional one of [9].

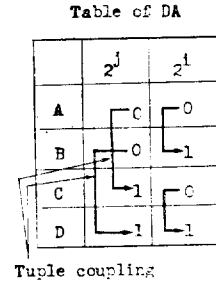
In order to define a RAD tree of cubes in the DA table., let the i-th bit of the binary form for a minterm be the i-th node of the tree and an arrowed line for 1-cube relation be a branch of the tree. When a pair of minterms have a 1-cube relation in their any k-th bit of binary form, some another bit (for example j-th bit) of them have the same property of RAD's (i.e. the same bist of two minterms are 1,1, or 0 and 0. This ralation is referred to the tuple-coupling of 1-cubes), then connect the two nodes by a line as a stem of the RAD tree (for this example i-th and j-th nodes are connected by a line).

The node which has the property of a tuple coupling can be reduced to one branch because of 1-cube of two minterms as shown in Fig. 8 constructed from Fig. 7 A doubling of the tuple-coupling relation of four minterms and the higher cases can also be reduced to branch of the tree succesively by the same fashion. The additin of each node doubles the number of vertices. For example, RAD trees of 2-cube relation of four minterms and 3-cube of eight mintermsre shown ain Fig. 7 through Fig. 10.

The knowledge of the RAD's and RAD trees of cubes applicable to a given minterm is a most effective starting point for the search for the Prime Implicant's (PI's) that cover that minterm.

The number of RAD's immediately defines the order of the largest cube that may possibly cover that minterm.

If a minterm has j RAD's, then the largest PI that can cover it is a j-cube having 2j O-cubes at it's vertices.



Fgi. 7. The Table of DA of two-cube (corresponds to Fig. 2.).

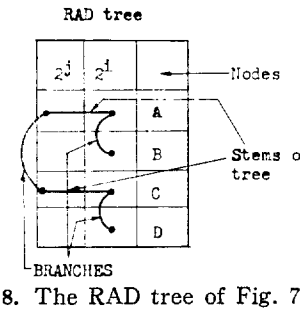


Fig. 8. The RAD tree of Fig. 7.

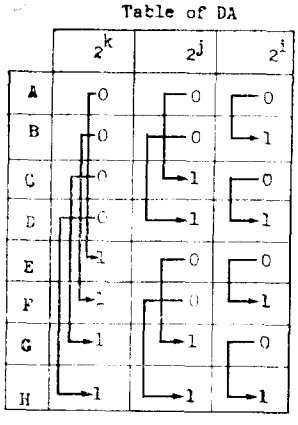


Fig. 9. The Table of DA of 3-cube from Fig. 4.

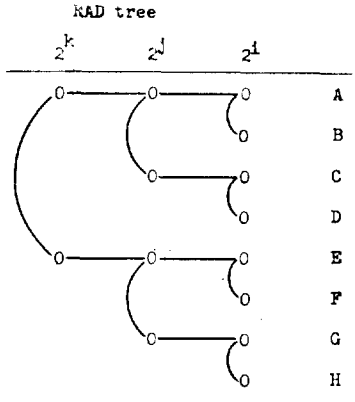


Fig. 10. RAD tree of Fig. 9.

Once we know how to construct the RAD trees in the DA Table we can find largest trees of the most nodes or the highest cube.

After the construction of RAD's trees of all the possible cubes the check list can be made and then the minimal sum of products are determined by the methods treated in other's references. There are another kind of trees that have not cube relations by only connecting a stem between the nodes of a minterm.

A procedure how to find the RAD trees of any possible cubes in the DA table for both the manual and the computer method is proposed according to the following steps.

Step 1 : Select the minterms whose number of RAD's are the largest in the table of the DA.

Step 2 : Among the minterms obtained in Step 1, select a pair of minterms with 1-cube relation (for example the i-th bit) from the top of the DA. Table

Step 3 : Search a tuple-coupling relation of RAD's for the two minterms obtained in Step 2.

If any one or more tuple-couplings are found (for example i-th bit of them), then a stem of the RAD tree between two nodes is connected. Thus 2-cube relation of four minterms is obtained easily (see Fig. 8).

Step 4 : Search the doubling of the tuple coupling relations of RAD's among the minterms obtained in Step 3.

If any one is found (for example the k-th bit of four minterms), then one more stem of RAD tree extended to the k-th node and eight minterms that have 3-cube relation are obtained clearly (see Fig. 10). Otherwise, go to Step 6.

Step 5 : Continue above Step 4 by doubling the tuple-coupling relation again until the largest tree is obtained.

Step 6 : In order to get another tree of the same order of cube, select another pair of minterms obtained in Step 2 and are not covered by the tree in Step 5. Then go to Step 2 through Step 6. If anothe pair of minterms that have the same order of cube are not found, go to Step 7.

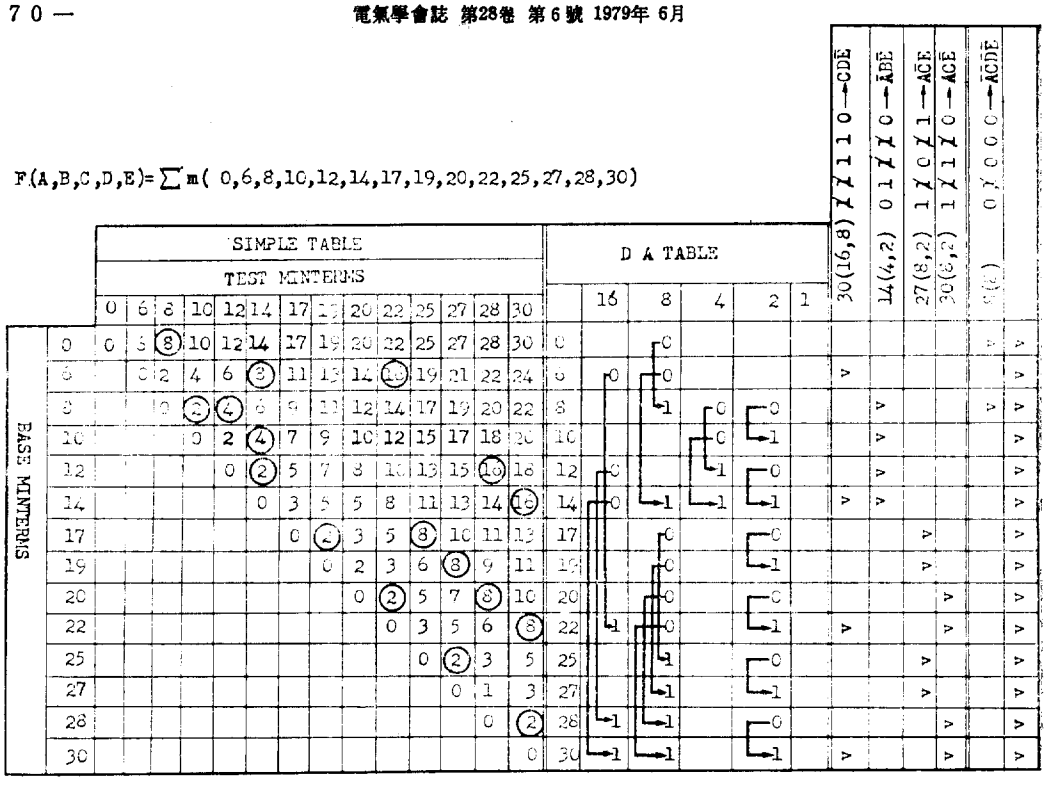
Step 7 : Another tree of any lower order cube can be obtained by repeating Step 1 through Step 7.

Three examples are shown in Fig. 11~Fig. 13. for the Indirect Method use. These are taken from (15), pp. 148~157.

V. APPLICATIONS OF THE SIMPLE TABLE METHOD

The methods of constructing the Simple Table and Check List are developed in Sections Ⅲ and Ⅳ. In this Chapter we deal with some applications of the Simple Table Method to the various types of given switching functions such as a single output function, a function including Don't Care minterms and multi-output functions.

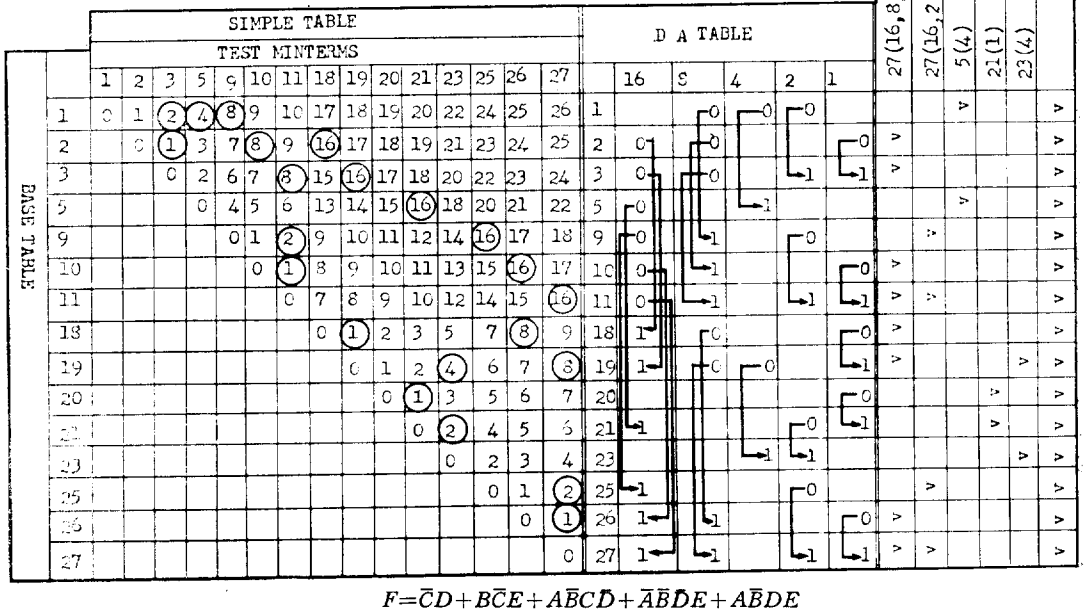
$F(A,B,C,D,E)=\sum m(\ 0,6,8,10,12,14,17,19,20,22,25,27,28,30)$



$$F=CD\bar{E}+\bar{A}B\bar{E}+A\bar{C}E+AC\bar{E}+\bar{A}\bar{C}\bar{D}\bar{E}$$

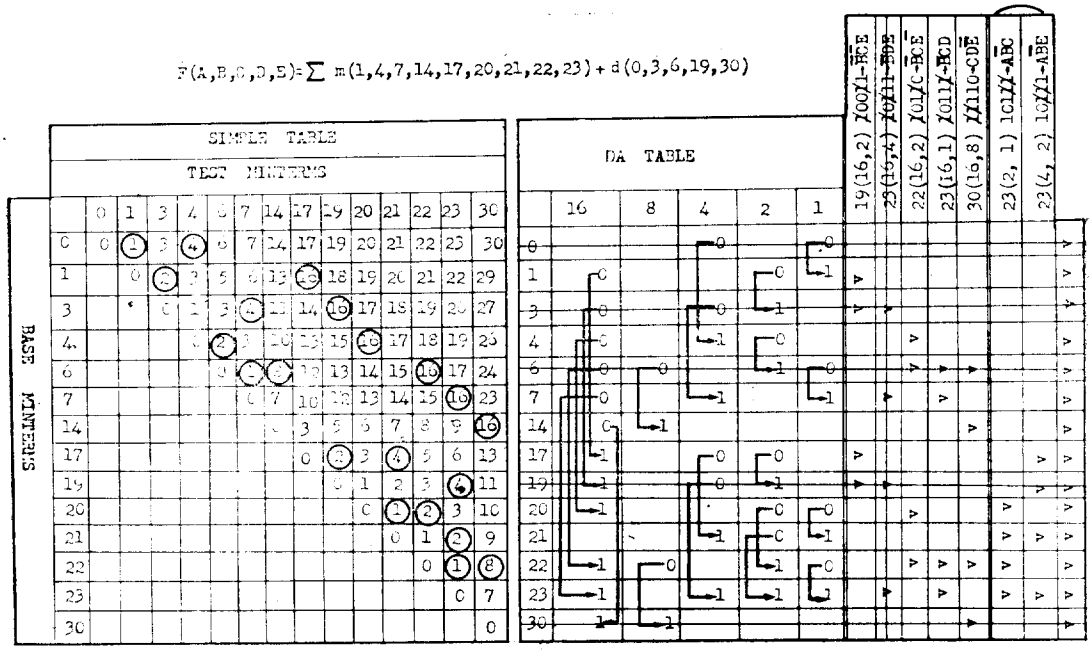
Fig. 11. An example for the DA table use.

$F(A,B,C,D,E)=\sum(1,2,3,5,9,10,11,18,19,20,21,23,25,26,27)$.



$$F=\bar{C}D+B\bar{C}E+A\bar{B}C\bar{D}+\bar{A}\bar{B}\bar{D}E+A\bar{B}DE$$

Fig. 12. An example for the DA table use.



$$F=CD\bar{E}+\bar{B}\bar{C}E+\bar{B}C\bar{E}+\bar{B}CD+\begin{cases}A\bar{B}E\\ \text{or}\\ A\bar{B}C\end{cases}$$

Fig. 13. An example for DA table use. (This example is identical to that of Fig. 16)

1) Application to Single-Output function

To determine the minimal sum of products, for a function $f(X_1,X_2,X_3,X_4)=\sum m(0,2,3,6,7,8,9,10,13)$, To determine the minemal sum of products we first construct the Simple Table and then build the Check List by using steps in Chap. Ⅲ and Ⅳ.

Solution

(i) Construction of Simple Table

Step 1 : Arrange minterms horizontally and vertically in ascending order.

Step 2 : For $Td \geq Bd$, find the value of difference, $R=Td-Bd$.

Step 3 : Encircle the values $2^n(n=0,1,2,\ldots)$ satisfying the Theorem 2.

Step 4 : Go to Step 5 because of not having Don't care minterms.

Step 5 : Provide Check List below the Simple Table.

The Simple Table is constructed in Fig. 14 for this switching function.

To determine the minimal sum of products form for $F(X_1,X_2,X_3,X_4)=\sum m(0,2,3,6,7,8,9,10,13)$, for which the Simple Table is shown in Fig. 14.

(ii) Construction of the Check List

Step 1 : Maximum value of N such that $2^n \leq K < 2^{n+1}$ is 1.

Step 2 : Using the Rule 2 for $N=1$, scan the table and seach for all the possible prime implicants. There are two cases satisfying the Rule 2. Each case is expressed by an arrow.

Step 3 : Check the minterms covered by these prime implicants in the Check List.

Step 4 : Use the Rule 1 for N=O.

Step 2′ : Select the rightmost value 4.

Step 3′ : Check in the Check List.

Step 4′ : All the minterms are covered.

Step 5 : Stop.

The Check list obtained by these steps is drawn below the Simple Table.

(iii) Dertermination of the minimal sum of products from the Check List.

By means of the Check List below the Simple Table, the minimal sum of products can be selected in a way similar to the Quine-Mc

BASE MINTERMS \ TEST MINTERMS	0	2	3	6	7	8	9	10	13
0	0	2	3	6	7	(8)	9	10	13
2		0	1	(4)	5	6	7	(8)	11
3			0	3	(4)	5	6	7	10
6				0	(1)	2	3	4	7
7					0	1	2	3	6
8						0	1	(2)	5
9							0	1	(4)
10								0	3
13									0
0,2,8,10 (8,2)	v	v				v		v	*
2,3,6,7 (4,1)		v	v	v	v				*
9,13 (4)							v		v *
	v	v	v	v	v	v	v	v	v

X_1	X_2	X_3	X_4	
χ	0	χ	0	→ $\bar{X}_2\,\bar{X}_4$
0	χ	1	χ	→ $\bar{X}_1\,X_3$
1	χ	0	1	→ $X_1\,\bar{X}_3\,X_4$

$$\therefore F = \bar{X}_2 \cdot \bar{X}_4 + \bar{X}_1 \cdot X_3 + X_1 \cdot \bar{X}_3 \cdot X_4$$

* Represents one of the minimal sum of products. This example is taken from [15], pp. 148~151.

Fig. 14. The Simple Table and the Check List for the simple out-put function.

BASE MINTERMS \ TEST MINTERMS	1	3	9	10	11	32	36	96	128	129	192	203
1	0	(2)	(8)	9	10	31	35	95	127	128	191	200
3		0	6	7	(8)	29	33	93	125	126	189	198
9			0	1	(2)	23	27	87	119	120	183	192
10				0	(1)	22	26	86	118	119	182	191
11					0	21	25	85	117	118	181	190
32						0	(4)	(64)	96	97	160	169
36							0	60	92	93	156	165
96								0	32	33	96	105
128									0	(1)	(64)	73
129										0	63	72
192											0	11
203												0
1,3,9,11 (8,2)	v	v	v		v							*
10,11(1)				v	v							*
32,36(4)						v	v					*
32,96(64)						v	v	v				*
128,129 (1)									v	v		*
128,192 (64)									v		v	*
203												v *
	v	v	v	v	v	v	v	v	v	v	v	v

X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8
0	0	0	0	∅	0	∅	1
0	0	0	0	1	0	1	∅
0	0	1	0	0	∅	0	0
0	∅	1	0	0	0	0	0
χ	0	0	0	0	0	0	0
1	χ	0	0	0	0	0	0
1	1	0	0	1	0	1	1

$F(X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8) = X_1X_2X_3X_4X_6X_8 + X_1X_2X_3X_4X_5X_6X_7 + X_1X_2X_3X_4X_5X_7X_8 + X_1X_3X_4X_5X_6X_7X_8 + X_2X_3X_4X_5X_6X_7X_8 + X_1X_3X_4X_5X_6X_7X_8 + X_1X_2X_3X_4X_5X_6X_7X_8$

Fig. 15. An example of the Single out-put function by the Direct Method.

Cluskey tabular method and the Petrick method.

For this function, the solution is

$F(X_1, X_2, X_3, X_4)=X_2X_4+X_1X_3+X_1X_3X_4$

Now a minimal sum of products of a large n-variable Boolean function is solved in the following example. It proves the simple and quick nature of the Simple Table method in comparison with other methods.

Example 1.

To determine the minimal sum of products for a function $F(X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8)=\sum m(1, 3, 9, 10, 11, 32, 36, 96, 128, 129, 192, 203)$

The Simple Table and Check List are constructed in Fig. 15.

From the Check List, we can determine the following minimal sum of products.

$F(X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8)=X_1X_2X_3X_4X_6X_8+X_1X_2X_3X_4X_5\bar{X}_6X_7+X_1\bar{X}_2X_3\bar{X}_4X_5\bar{X}_7X_8+X_1X_3X_4X_5\bar{X}_6\bar{X}_7X_8+\bar{X}_2X_3X_4X_5X_6X_7X_8+\bar{X}_2\bar{X}_3X_4X_5X_6X_7X_8+X_1X_3\bar{X}_4X_5X_6X_7X_8+X_1X_2\bar{X}_3\bar{X}_4X_5\bar{X}_6X_7X_8$

2) Application to the Don't Care minterms

As in the case with the Quine-McCluskey tablular method, the Simple Table method can be extended to functions including Don't Care minterms.

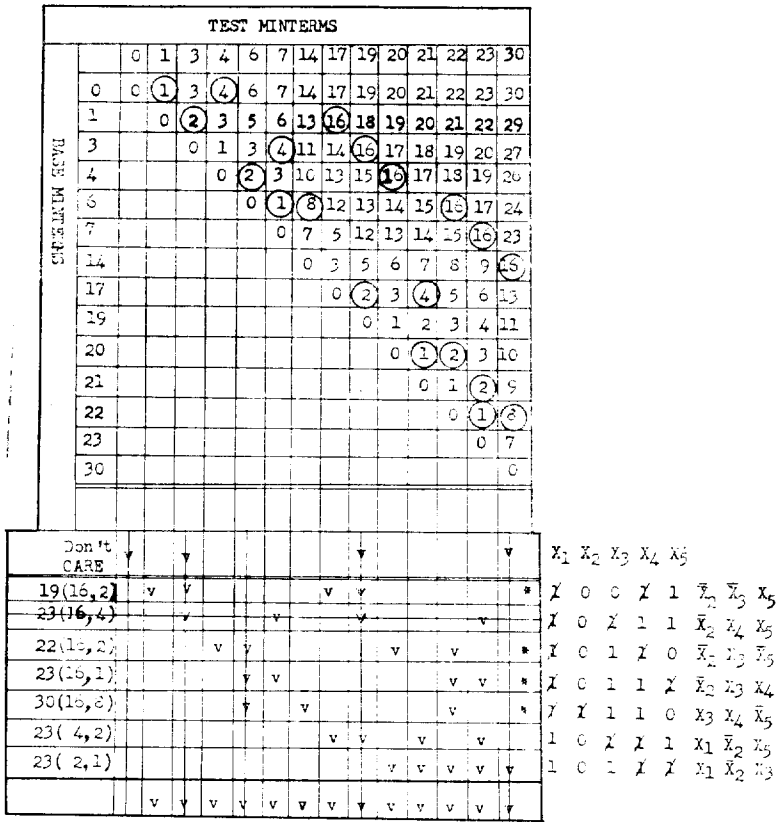
In the process of selecting the minimal sum of products, Don't Care minterms are removed.

If prime implicants which are selected cover all the minterms except the Don't Care minterms, the minimal sum of products are obtained.

An example for Don't Care minterms is given in Fig. 16.

Example 2. For a given function

$F(X_1, X_2, X_3, X_4, X_5)=\sum m(1, 4, 7, 14, 17, 20, 21, 22, 23)+d(0, 3, 6, 19, 30)$, first arrange the minterms in ascending order regardless of Don't Care minterms



$$F=X_3X_4X_5+X_2X_3X_5+X_2X_3X_5+\bar{X}_2X_3X_4+\begin{cases}X_1\bar{X}_2X_5\\ \text{or}\\ X_1\bar{X}_2X_3\end{cases}$$

Fig. 16 An example for the Don't Care Minterms.

and construct the Simple Table and Check List as in the Table-Note that the Don't Care minterm row is inserted between the Simple Table and the Check List.

The solution for this example is

$$F=X_3X_4X_5+X_2X_3X_5+X_2X_3X_5+X_2X_3X_4+\begin{cases}X_1X_2X_5\\ \text{or}\\ X_1X_2X_3\\ \text{optional}\end{cases}$$

This example is taken from [15], pp. 157 ~159.

3) Application to Multi-Output Function.

The adaptation the Simple Table Method to Multi-Output function is quite similar to the Single out-put function.

Considerable savings can often be achieved by the sharing of hardware among the multi-output functions. It is clear that minimum total number of gates are required. To find the minimal sum of product realization for each function, at first draw the intersection table (as shown in Table 3.) of a given functions.

Then locate this intersection table between the Simple Table and the Check List of each function for corresponding column minterms.

Next make the check list, just as in the single-output Simple Table Method. Only differnences are in that (1) we select the common letter from the intersection table in checking minterms which are reduced to one prime implicant and write in corresponding row and (2) in selecting prime implicants. The following procedure is used to guarantee an optimum realization (minimum total cost).

Step 1. Select the prime implicants group containing the same number of the common letters.

Step 2. Obtain the prime implicants from the selected group just as in the single output function.

Step 3. If all the minterms are covered, go to Step 5. Otherwise go to Step 4.

Step 4. Select the next group which has the lower common letters, and go to step 2.

Step 5. Find the dominated prime implicants

Table.3 The Intersection Table for the Multi-ouput function.

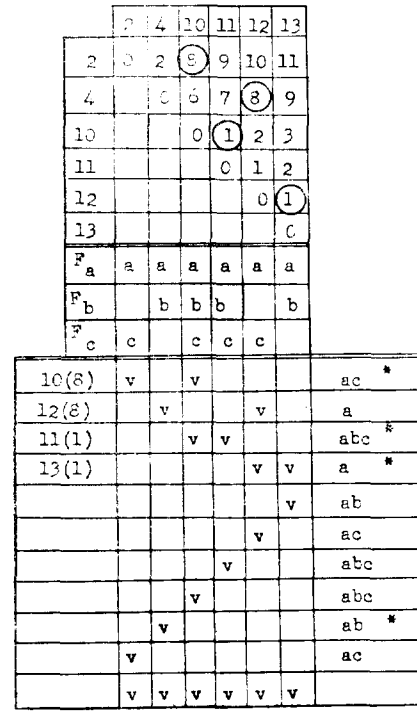
Out-put Functions		Minterms								
		0	1	3	4	5	7	8	9	12
	F_a	a	a	a		a	a			
	F_b		b	b	b			b	b	
	F_c	c		c			c		c	c

Fa=∑m(0,1,3,5,7)
Fb=∑m(1,3,4,8,9)
Fc=∑m(0,3,7,9,12)

Output Functions		Minterms								
		1	2	3	4	5	10	11	12	13
	F_a		a		a		a	a	a	a
	F_b				b	b	b	b		b
	F_c	c	c	c			c	c	c	

Fa=∑m(2,4,10,11,12,13)
Fb=∑m(4,5,10,11,13)
Fc=∑m(1,2,3,10,11,12)

Fig. 17 The Intersection Table.



Fa=$X_2X_3X_4+X_1X_2X_3+X_1X_2X_3+X_1X_2X_3X_4$

Fig. 18 The Multi-output function.

	4	5	10	11	13	
4	0	(1)	6	7	9	
5		0	5	6	(8)	
10			0	(1)	3	
11				0	2	
13					0	
Fa	a		a	a	a	
Fb	b	b	b	b	b	
Fc			c	c		
5(1)	v	v				b
11(1)			v	v		abc *
13(8)		v			v	b *
					v	ab
				v		abc
			v			abc
		v				b
	v					ab *
	v	v	v	v	v	

Fb=$X_1X_2X_3+X_2X_3X_4+X_1X_2X_3X_4$

Fig.19 The Multi-output function

and eliminate them.

Step 6. Function realization.

An example for this case is solved as follows. This example is taken from [15], pp. 161~166.

For a given multiple output function, first the intersection table is constructed as in Fig. 17. and then from this intersection table we can obtain each another intesection table that inserted between the Simple Table and the Check List.

The Simple Table and the intersection table and the Check List are given Fig. 18~Fig. 20 for each function.

Using the Step in Chap. V-(3), we can select the minimal sum of products for each function.

VI. CONCLUSIONS

Being the simple Table Mehtod based upon the numerical properties of Theorem 2 and its tabular representaion, the techniques described in this paper have been shown to be fundamentally different from those given in [1]~[14], [19]~[22] that based upon the conventional boolean concepts. The experiments with the various examples showed that this Simple Table Method is efficient. One of the greatest important features of this paper is that the switching functions are simply handled, no matter how many variables there may be, by both the manual and the computer method.

This method does neither need large capacity of memory as the algorithms [1], [4], [21], [22], nor require a large number of basic operations as [3], [4], [6], [20]~[22]. In general, the conclusions of this paper can be summarized as follows.

1) A theroem governing the adjacent relations among minterms is derived for the construction of the Simple Table.

2) Since all the directions of adjacency relations among minterms are represented completely

	1	2	3	10	11	12	
1	0	1	(2)	9	10	11	
2		0	(1)	(8)	9	10	
3			0	7	(8)	9	
10				0	(1)	2	
11					0	1	
12						0	
Fa		a		a	a	a	
Fb				b	b		
Fc	c	c	c	c	c	c	
11(8,1)		v	v	v	v		c
11(1)				v	v		abc *
3(2)	v		v				c *
3(1)		v	v				c
10(8)		v		v			ac *
12						v	ab *
11					v		abc
10				v			abc
3			v				c
2		v					ac
1	v						c

Fc=$X_1X_2X_3+X_1X_2X_4+X_2X_3\bar{X}_4+X_1X_1X_3X_4$

Fig. 20 The Multi-output function.

in the DA table, and the DA table can be obtained easily from the Simple Table, hence PI selecttion process becomes evidently easier by the constructing semantic trees of any possible number of cube.

3) The Direct method for PI selection is also represented.

4) All the prime implicants are listed in the Simple Table.

5) It is easy to idetify the largest PI, for its graphical structure of the Simple Taple and the DA table.

6) Because of its systematic steps, it can be applied to the computer program with the less memories and the less number of operations.

7) Owing to its simplicity it could be understood by students and logic designers with ease.

8) Since all the minterms are handled by the decimal number hence it would be easy to check.

9) Since only the decimal numbers corresponding to minterms are used in the Simple Table Method, the various concepts such as degree of consensus, distance between minterms and symbols of covering as [3],[6] are not used at all.

10) The DA table can be constructed directly from the switching function by applying theorem

11) Applications to Don't Care minterms and multiple-output functions by introducing the Don't Care minterm row and the Intersection table are made.

The usefulness of this Simple Table Method will be found in various areas such as logic designs, sequential circuits and circuit analysis.

ACKNOWLEDGEMENT

The author wishes to express his sincere thanks to Prof.s Yang Heung Seuck,Park Young Moon and a graduate student Cho Dong Seop at Dept. of Electrical Engineering, College of Eng. S.N.U. for their helpful suggestions and advices in preparing this paper.

REFERENCES

[1] G. Karnaugh, "The map method for synthesis of combinational logic circuits," AIEE Trans. Commun. Electron., pt.1, Vol. 72, pp. 593~599. Nov. 1953.

[2] E.J. McCluskey, Jr., "Minimization of Boolean Functions." Bell Syst. Tech. J., Vol. 35, pp.1417~1414, Nov. 1956.

[3] Zosimo Arevalo and J.G. Bredeson, "A method of simplify a Boolean function into a near minimal sum-of products for programmable logic arrays," IEEE Trans. Comput Vol. C-27, pp. 1028~1039, Nov. 1978.

[4] N.N.Necula, "An algorithm for the automatic approximate minimization of Boolean functions," IEEE Trans. Comput., Vol. C-17, pp· 770~782, Aug. 1968.

[5] H.A. Curtis, "Simplified decomposition of Boolean functions," IEEE Trans. Comput., Vol. C-25, pp. 1033~1076, Oct. 1976.

[6] Sureshander, "Minimization of switching functions-A fast technique," IEEE Trans. Comput. (Corresp.), Vol. C-24., pp. 753~756, July 1975.

[7] B. Reusch, "Generation of prime implicants from subfunctions and a unifying approach to the covering problem," IEEE Trans. Comput., Vol. C-24, pp.924~930, Sept. 1975.

[8] F.M. Brown, "Equational realizations of switching functions," IEEE Trans. Comput., Vol. C-24, pp. 1054~1066, Nov. 1975.

[9] V.V. Rhyne, P. Noe, M. Mckinney and U. W. Pooch, "A new technique for the fast minimization of switching functions," IEEE Trans. Comput., Vol. C-26, pp. 757~764, Aug.1977.

[10] S.R.Das, "Comments on'A new algorithm for generating prime implicants," IEEE Trans. Comput., Vol. C-20, pp. 1614~1615, Dec. 1971.

[11] J.G. Bredeson and D.T. Hulene, "Generation," of prime implicant by direct multiplication IEEE Trans, Comput., Vol. C-20, pp. 475~476, Apr. 1971.

[12] H.R.Hwa, "A method for generating prime implicants of a Boolean expression," IEEE Trans. Comput., Vol. C-23, pp. 637~644, June 1974.

[13] B.L. Hulme and R.B. Worrell, "A prime

implicant algorithm with factoring," IEEE Trans. Comput., Vol. C-24, pp.1129~1131, Nov. 1975.

[14] J.R. Slagle, C.L.Chang, and R.C.T.Lee, "A new algorithm for generating prime implicants," IEEE Trans. Comput., Vol. pp.304~310, Apr. 1970.

[15] F.J.Hill and G.R. Peterson, Introduction to switching theory and Logical design, Wiley, New York, 1974, pp. 97~174.

[16] Taylor L. Booth, Digital Network and Computer Systems, Wiley, New York, 1971, pp. 93~157.

[17] John B. Peatman, The Design of Digital System, McGraw Hill, New York, 1972, pp. 56~116.

[18] V.T. Rhyne, Fundamentals of Digital Systems Design, Englewood Cliffs, NJ:Prentice-Hall, 1973, pp. 163~165.

[19] W.V. Quine, "A way to simplify truth functions," Amer. Math. Mon. vol. 62, pp. 627~631, Nov. 1955.

[20] P. Tison, "Generalization of consensus theory and application to the minimization of boolean functions," IEEE Trans. Electronic Computers, vol. EC-16, pp. 446~456, August 1967.

[21] A. Avobcda, "Ordering of implicants," IEEE Trans. Electronic Computers (Short notes), vol. EC-16, pp. 100~150, February 1967.

[22] N.N. Necula, "A numerical procedure for determination of the prime implicants of a Boolean function," IEEE Trans. Electronic Computers (Correspondence), vol. EC-16, pp. 687~689 October 1967.