MASK 方法에 依한 論理函數의 最小化

論 文
28—11—2

On the Minimization of the Switching Function by the MASK Method

趙 東 燮*·黃 熙 隆**
(Dong Sub Cho, Hee Yeung Hwang)

Abstract

This paper deals with the computer program of finding the minimal sum-of-products for a switching function by using the MASK method derived from the characteristics of the Boolean algebra. The approach differs from the previous procedures in that all the prime implicants are determined only by the bit operation and the minimal sum-of-products are obtained by the modified Petrick method in this work. The important features are the relatively small amount of the run time and the less memory capacity to solve a problem, as compared to the previous computer programs.

Ⅰ. Introduction

The minimization of switching functions involving many variables is a difficult task. For these cases, the Karnaugh map approach breaks down and tabular or algorithmic method due to McCluskey, based on an original technique due to Quine, is used as a hand computation technique or better still programmed for a digital computer. The basic idea of the minmization is to examine each term to see if the Boolean theorem $AB+A\bar{B}=A$ can be applied. This results in a complete list of all the prime implicants for the function concerned. Then the set of the prime implicants that covers the function is obtained from the larger set of all prime implicants.

Several papers have treated the problem by this process and this paper also handles the problem in a similar way. Nevertheless, it differs from the previous work in identifying the prime implicants. This difference can make the minimization process more efficient, since only those prime implicants that are not involved in other prime implicants are generated directly by the MASK method presented in Section Ⅱ. In the selection process of finding the minimal sum-of-products, a minmum covering of the given minterms are determined by the modified Petrick method presented in Section Ⅲ. In Section Ⅳ computer program techniques and the flow chart are described.

Ⅱ. Identification of the Prime Impilcants

Almost all of the procedures in references [1]-[23] treat the minimization process as two seperate parts. First, all of the prime implicants are generated from the given minterms. This is PI identification. Then the set of prime implicants that best cover the switching function is chosen from the larger set of all prime implicants. This is PI selection.

In this section, the identification process for the prime implicants is discussed. There are several computer algorithms for the minimization of the switching function, but the basic idea is the same. According to the Boolean theorem $AB+A\bar{B}=A(B+\bar{B})=A$, all the prime implicants are

* 正會員：서울大 大學院
** 正會員：서울大 工大 副敎授
接受日字：1979年 8月2日

generated from the combinable minterms. Conversely, the Boolean canonic form of minterms are directly obtained by using the following property of the prime implicant and the Mask method.

1) Property of the Prime Implicant

Definition:

Let $F(x_1, x_2, \ldots, x_n)$ be a Boolean canonic minterm of n variables $x_1, x_2, \ldots, x_n$. The partial derivative of F with respect to x_i, $1 \leqq i \leqq n$, is defined as

$$\frac{dF}{dx_i} = F(x_1, x_2, \ldots, x_i, \ldots, x_n) \oplus F(x_1, x_2, \ldots, \bar{x}_i, \ldots, x_n).$$

Theorem 1: (Property of the prime implicant)

If the prime implicant has the n eliminated literals and the number of the input variables is m, Boolean canonic minterms whose number is $\sum_{r=0}^{n} {}_nC_r (=2^n)$ satisfy the following equation,

$$\sum_{i=1}^{n} \frac{dF(G(X_{p1}, X_{p2}, \ldots, X_{p(m-n)}), X_1, X_2, \ldots, X_n)}{dX_i} = 0$$

where $G(X_{p1}, X_{p2}, \ldots, X_{p(m-n)})$ denotes the prime implicant and X_i indicate the eliminated bit positions and $\frac{d\,F}{d\,X_i}$ is the partial derivative (or Boolean difference) of F with respect to X_i (see Reference [24]).

Proof:

First, for $n=1$, there exists an eliminated bit X_1 satisfying,

$$\frac{dF(G(X_{p1}, X_{p2}, \ldots, X_{p(m-1)}), X_1)}{dX_1}$$
$$= F(G(X_{p1}, X_{p2}, \ldots, X_{p(m-1)}), X_1) \oplus F(G(X_{p1}, X_{p2}, \ldots, X_{p(m-1)}), \bar{X}_1)$$
$$= F(G(X_{p1}, X_{p2}, \ldots, X_{p(m-1)}), 1) \oplus F(G(X_{p1}, X_{p2}, \ldots, X_{p(m-1)}), 0)$$
$$= 1 \oplus 1 = 0. \qquad (1)$$

Thus the equality holds for $n=1$.

Now assume that the equality holds for $n=k$

$$\sum_{i=1}^{k} \frac{dF(G(X_{p1}, X_{p2}, \ldots, X_{p(m-k)}, X_1, X_2, \ldots, X_k)}{dX_i} = 0 \qquad (2)$$

To prove that it holds for $n=k+1$, the partial derivative can be applied.

$$\sum_{i=1}^{k+1} \frac{dF(G(X_{p1}, X_{p2}, \ldots, X_{p(m-k-1)}), X_1, X_2, \ldots, X_k, X_{k+1})}{dX_i}$$
$$= \sum_{i=1}^{k+1} \frac{dF(G(X_{p1}, X_{p2}, \ldots, X_{p(m-k-1)}), X_{k+1}, X_1, X_2, \ldots X_k)}{dX_i}$$

Let $[G(X_{p1}, X_{p2}, \ldots, X_{p(m-k-1)}), X_{k+1}]$ and $[G(X_{p1}, X_{p2}, \ldots, X_{p(m-k-1)}), X_1, X_2, \ldots, X_k]$ be $H(X_{p1}, X_{p2}, \ldots, X_{p(m-k)})$ and $P(X_{p1}, X_{p2}, \ldots, X_{p(m-1)})$, respectively. Thus it yields,

$$\sum_{i=1}^{k} \frac{dF(H(X_{p1}, X_{p2}, \ldots, X_{p(m-k)}), X_1, X_2, \ldots, X_k)}{dX_i} + \frac{dF(P(X_{p1}, X_p, \ldots, X_{p(m-1)}), X_{k+1})}{dX_{k+1}}$$
$$= 0 + 0 = 0 \text{(from equation (1) and (2)).}$$

Q.E.D.

Example 1:

Consider the Boolean canonic minterms from the prime implicant, $X_1\bar{X}_2X_4$, whose eliminated bits are X_3 an X_5.

From the above theorem,

$$\frac{dFG(X_1, \bar{X}_2, X_4,)X_3, X_5}{dX_3} = (X_1, \bar{X}_2, X_3, X_4, X_5) \oplus G(X_1, \bar{X}_2, \bar{X}_3, X_4, X_5)$$

$$\frac{dFG(X_1, \bar{X}_2, X_3, X_4, X_5)}{dX_5} = G(X_1, \bar{X}_2, X_3, X_4, X_5) \oplus G(X_1, \bar{X}_2, X_3, X_4, \bar{X}_5)$$

$$\frac{dFG(X_1, \bar{X}_2, \bar{X}_3, X_4, X_5)}{dX_5} = FG(X_1, \bar{X}_2, \bar{X}_3, X_4, X_5) \oplus F(X_1, X_2, \bar{X}_3, X_4, X_5)$$

Hence, the canonic minterms are $X_1\bar{X}_2X_3X_4X_5$, $X_1\bar{X}_2X_3X_4\bar{X}_5$, $X_1\bar{X}_2\bar{X}_3X_4X_5$ and $X_1\bar{X}_2\bar{X}_3X_4\bar{X}_5$.

And note that the number of the generated minterms is 2^n, where n is the number of the eliminated bits.

2) Mask Method

From the above discussion, the prime implicant can be obtained by using the Mask Method that is described by the following theorem 2.

Definition:

If the given set of minterms are arranged in ascending order of decimal equivalence of each binary minterm the lowest minterm is called LM and the highest minterm is said to be HM.

Theorem 2: (Computer algorithm for finding the prime implicants)

The prime implicant can be generated from the given set of minterms if and only if the following relations hold.

Relation 1: Let the lowest minterm of the selected set of the given minterms be LM and the highest minterm HM. LM .AND. HM=LM

where .AND. denotes the bit operation.

Relation 2: Let the result of LM. EX-OR. HM be MASK which shows the position of the eliminated bits. After performing the OR operation (OR masking) with MASK for all through the selected set of minterms, the number of the same result equal to HM is 2^n.

Proof: From the property of the prime implicant (Theorem 1), there exists a prime implicant which satisfies the following equation,

$$\sum_{i=1}^{n} \frac{dF(G(X_{p1}, X_{p2}, \ldots, X_{p(m-n)}), X_1, X_2, \ldots, X_n)}{dX_i} = 0$$

where the lowest minterm and the highest minterm of the given set of minterms is $F(G(X_{p1}, X_{p2}, \ldots, X_{p(m-n)}), 0, 0, \ldots, 0)$ and $F(G(X_{p1}, X_{p2}, \ldots, X_{p(m-n)}), 1, 1, \ldots, 1)$ respectively.

Since each minterm (LM and HM) has the same prime implicant term, without loss of generality relation 1 holds for these two minterms.

$F(G(X_{p1}, X_{p2}, \ldots, X_{p(m-n)}), 0, 0, \ldots, 0)$ .AND.
$F(G(X_{p1}, X_{p2}, \ldots, X_{p(m-n)}), 1, 1, \ldots, 1)$
$=F(G(X_{p1}, X_{p2}, \ldots, X_{p(m-n)}), 0, 0, \ldots, 0)$

And to find the position of the eliminated bits, the EXCLUSIVE-OR operation is applied to these two minterms satisfying Relation 1

$F(G(X_{p1}, X_{p2}, \ldots, X_{p(m-n)}), 0, 0, \ldots, 0)$ .EX-OR.
$F(G(X_{p1}, X_{p2}, \ldots, X_{p(m-n)}), 1, 1, \ldots, 1)$
$=F(G(0, 0, \ldots, 0), 1, 1, \ldots, 1)$

From this result, the eliminated bit positions which are set to 1 are obtained. Next, the 2^n minterms are checked by setting the eliminated bit positions, since they have the same term, $G(X_{p1}, X_{p2}, \ldots, X_{p(m-n)})$.

To computerize this, OR masking method is used for the purpose of setting the eliminated bit positions to 1. And then, there must be 2^n terms which are equal to HM if the given set of the minterms has the prime implicant.

Q.E.D.

Example 2:

Extract the prime implicant from the following minterms by the theorem 2, called the MASK method,

0000···LM
0010
0101
1000
1010···HM.

Step 1: Check if the relation 1 is satisfied.
0000 .AND. 1010=0000

Step 2: Check if the relation 2 is satisfied.
MASK=0000 .EX-OR. 1010=1010
And then perform the OR operation with the given set of minterms.
0000 .OR. 1010=1010*
0010 .OR. 1010=1010*
0101 .OR. 1010=1111
1000 .OR. 1010=1010*
1010 .OR. 1010=1010*
Check if the number of the same values which are equal to HM is 2^2.
In This example,
the number of 1's is 2 and the number of the same value as HM is 4.

Step 3: If the given set of minterm satisfies step 1 and 2, the prime imlicant is obtained by eliminating the bits (whose positions are shown in MASK) of LM or HM.
LM·····0000
MASK·····1010
Hence, the prime implicant is XOXO.

For a switching function, its prime implicants can be generated by this MASK method. Because of its testing between LM and HM, the smaller cost of a prime implicant is generated rapidly by the MASK method. The essential advantages of the MASK method are the rapid generation of the prime implicants and a less amount of run time and memory capacity of its computer program.

The logical opereations used in the MASK method cannot be simply performed manually because of the inability of man to handle too many bits, but they can be performed very easily by

digital computer. Hence, the MASK method is well suited for the computer programming of the prime implicants generation.

Ⅲ. Selection of the minimal sum-of-products

To select the minimal subset of the prime implicants, the prime implicant table should be constructed with the prime implicants that are not dominated The selection process for the minimal sum-of-products of a given Boolean switching function can be carried out as follows.

1) Determination of the minterms which are not reduced to the prime implicants. The number of checks is determined columnwise for each minterm. And then if the number is 0, the corresponding minterm is not reduced.
2) Determination of the essential prime implicants: If the counted number is 1, the corresponding prime implicant is the essential prime implicant.
3) Selection of the chosen prime implicants from the cyclic table by the modified Petrick method: The Petrick method is very useful in selecting the chosen prime implicants, but the selection of a minimum subset of prime implicants is a very difficult problem, even for digital computer (especially for the cyclic table with the large number of prime implicant). To computerize the Petrick method, it can be modified as follows.

1) Modified Petrick Method.

In this method, each row of the prime implicant table is considered as a binary variable and a product-of-sums expression is derived for the complete table. This function is called a prime implicant function (P), and each variable corresponds to a prime implicant of the original switching function (see Fig. 1).

From Fig. 1, to account for all the terms in the table, the prime implicant function P is obtained as follow,

$$P=(A+B+D)(B+E)(A+B)(C+D)(A+D)(A+C).$$

Since this is a Boolean expression, it may be reduced in the normal way.

Minterms / PI's	m_1	m_2	m_3	m_4	m_5	m_6
A	V		V		V	V
B	V	V	V			
C				V		V
D	V			V	V	
E		V				

Fig. 1. The cyclic prime implicant table.

Taking the complementation on P,

$$\bar{P}=\overline{(A+B+D)(B+E)(A+B)(C+D)(A+D)(A+C)}$$
$$=\overline{(A+B+D)}+\overline{(B+E)}+\overline{(A+B)}+\overline{(C+D)}+\overline{(A+D)}+\overline{(A+C)}$$
$$=\bar{A}\bar{B}\bar{D}+\bar{B}\bar{E}+\bar{A}\bar{B}+\bar{C}\bar{D}+\bar{A}\bar{D}+\bar{N}\bar{C}$$
$$=\bar{A}(\bar{B}\bar{D}+\bar{B}+\bar{D}+\bar{C})+\bar{B}\bar{E}+\bar{C}\bar{D}.$$

Assuming the lowest cost of the function $\bar{P}$, the result is

$$\bar{P}=\bar{A}+\bar{B}\bar{E}+\bar{C}\bar{D}.$$

Thus $P=A(B+E)(C+D)$ and the chosen prime implicants are

A, B or E, C or D.

2) Procedure for PI selection by the Modified Petrick method

To computerize this procedure, following steps are given.

Step 1: Count the number of the checks in each row of the prime implicant table and let it be the original cost.

Step 2: Count the number of the checks in each column of the cycle prime implicant table and let it be K.

Step 3: If the value of K is 0, the corresponding minterm is selected as the not-reduced minterm, and if the value of K is 1, the prime implicant which implies the check is selected as the essential prime implicant.

Step 4: Eliminate the checks in the corresponding minterm columns which are reduced to the essential prime implicants.

Step 5: Count the number of the checks in each row of the cyclic prime implicant table and let it be the new cost of the corresponding prime implicants.

Step 6: Select the chosen prime implicant which has the maximum value of a new cost. If all the minterms are covered, go to the Step 7. Otherwise, go to the Step 5.

Step 7: Stop.

Ⅳ. Computer programming by the MASK method and the modified petrick method

The minimization algorithm, called the MASK method, generate the prime implicants from the given set of minterms. All the operations of the MASK method are the bit operations, and so they can be directly programmed by the assembly language. But, for convenience sake, they are programmed by the FORTRAN language.

In the computer programming the logical operations are performed by the FORTRAN as follows.

1) Programmings for the following statements:

a. IF LM .AND. HM=LM GO TO 1 OTHERWISE GO TO 2

```
      DO 10 I=1, N
      IE(LM(I)-HM(I)) 2, 10, 10
   10 CONTINUE
      GO TO 1
```

where the number of variables is N.

b. LET LM .EX-OR. HM BE MASK

```
      DO 20 I=1, N
      IF(LM(I)+HM(I).EQ. 1) GO TO 30
      MASK(I)=0
      GO TO 20
   30 MASK(I)=1
   20 CONTINUE
```

c. OR-MASK WITH MASK THROUGH LM TO HM AND COUNT THE SAME VALUES EQUAL TO HM

```
      COUNT=0
      DO 50 I=1, M
      DO 40 J=1, N
      IF(MASK(J).EQ. 1) GO TO 40
      IF(K(I, J).NE.HM(J)) GO TO 50
   40 CONTINUE
      COUNT=COUNT+1
   50 CONTINUE
```

where K(1, J) is LM and K(M, J) is HM.

2) Programming for the modified Petrick method:

Let the original cost be OCOST and the new cost NCOST. The procedure selecting the chosen prime implicants is programmed by the FORTRAN language as follows.

```
      CP=1
      OBIG=OCOST(1)
      NBIG=NCOST(1)
      DO 60 I=2, N
      IF(NBIG-NCOST(I)) 44, 55, 60
   44 NBIG=NCOST(I)
      OBIG=OCOST(I)
      CP=I
      GO TO 60
   55 IF(OBIG-OCOST(I)) 77, 60, 60
   77 OBIG=OCOST(I)
      CP=I
   60 CONTINUE
```

where CP is the chosen prime implicant number and N is the number of prime implicants in the prime implicant table.

The flow chart for the computer programming of the switching function by the MASK method is shown in Fig. 2. The prime implicant identification section is executed by the modified Petrick method.

All the minterms inoluding don't care minterms are read in decimal number. And then they are converted to the binary form in order to perform the bit operation. Applying the MASK method to the selected set of minterms, the prime implicants are generated simply and the prime implicat table is easily constructed.

From the prime implcant table, at first not-reduced minterms and the essential prime implicants are generated. Then the cyclic prime implicant table is obtained and the modified Petrick method is applied to this cyclic table. In this procedure the chosen prime implicants are taken but these are not unique solution because of the

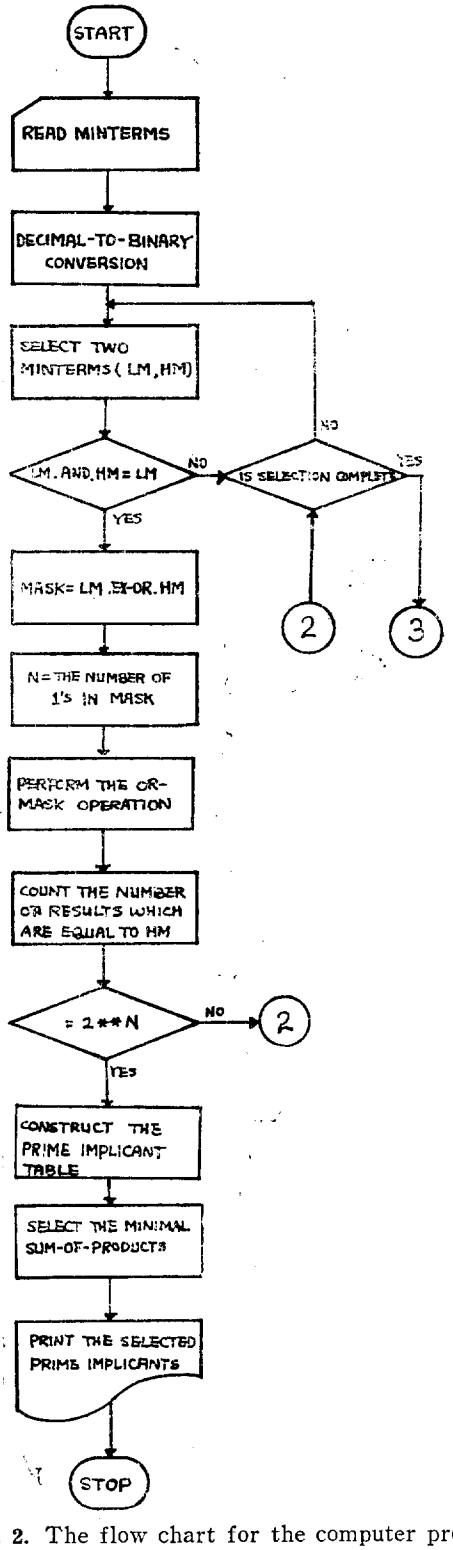

Fig. 2. The flow chart for the computer programming of the switching function.

optional selection from the prime implicants which have the same cost.

The computer program of minimizing the switching function is listed in Appendix.

V. Conclusions

The MASK method, described in Section II, contains two new algorithms for identifying the prime implicant. This method is well suited for the computer program and found to be more efficient than other computer program, since algorithms are handled by the bit operation. In fact, this program is programmed by the FORTRAN language but the computation time and the memory capacity will be much less reduced if programmed by the assembly language.

By comparison with the Quine-McCluskey(QM) tabular procedure, the QM procedure identifies a larger implicant only after exhaustively identifying all of the smaller implicants that subsume it. But a larger prime implicant of the given set of minterms is directly obtained by the applying two algorithms.

As a comparison, aset of minimization problems is run by using both the computer program in appendix of reference[18] and FORTRAN version of the MASK method. The results of this comparison are listed in Table 1. These are only loose comparisons. Interested readers are urged to con-

Table. 1 Results of comparison between the MASK method and another minimization program.(Both programs are executed on FACOM 230-28S)

Example	Number of variables	Execution time (sec) Program in [18]	Execution time (sec) MASK method	Ratio
1	3	0.374	0.466	0.80
2	3	2.017	0.512	3.94
3	4	2.179	0.840	2.59
4	4	2.503	0.839	2.98
5	5	5.476	1.239	4.42
6	5	4.833	1.078	4.48
7	6	17.894	2.803	6.38
8	6	22.389	3.391	6.60
9	7	145.979	32.678	4.47
10	7	134.231	26.042	5.15

tact the authors for details. Table 1 shows that the MASK method is faster in all but one case. These are concluded as follows;

1) The computer program by the MASK method gives answers close to the minimal cost and requires the less computation timeas compared to others in the same variables. (see the results of Table 1)
2) As the input variables increase, the computing time increase at a lower rate.
3) If programmed by the assembly language, a much less amount of the computation time and the memory spaces will be required.
4) For the multiple output simplification cases, the MASK method can also be applied.

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. 1028~1038, Nov. 1978.

[4] N.N. Necula, "An algorithm for the automatic approximate minimization of Boolean function," 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 Tran. 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, N.J. 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 Co-

puters, Vol. EC-16, pp. 446～456, August 1967.

[20] A. Avoboda, "Ordering of implicants," IEEE Tran. 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.

[23] 黃熙隆, A new approach to the minimization of switching functions by the simple table method," 대한전기학회지, 제28권 제 6 호, pp. 61～77, 1979 6월.

[24] S.C. Lee, Modern Switching Theory and Digital Design, Prentice-Hall, Englewood Cliffs, N,J. 1978.