UIT Report Vol.11. No.2, pp.195~207, 1980 연구논문집 제11권 제2호 pp.195~207, 1980

MASK方法과 COST TABLE에 의한 논리함수의 최소화에 대한 계산기 利用에 관한 연구

張 師 元
전 자 계 산 학 과
(1980. 6. 10 접수)

〈요 약〉

論理함수를 최소화하는 方法에 대해서는 지금까지 많은 논문들이 발표되어왔다. 대표적으로 Karnaugh-Map이 창안된 후 Quine-McClusky의 Tabular Method가 창안되었으나 이러한 方法들은 변수의 증가에 따라 문제의 해결이 점점 어려워 진다는 공통점을 안고 있다.

따라서 이러한 난점을 극복하기 위해서 여러 方法으로 연구하던 중에, MASK 方法과 COST TABLE 결정法에 대한 이론이 창안되면서, 손쉽게 多出力 함수를 구할수가 있다. 모든 최소화된 多 出力함수를 구하는데 있어서, PRIME IMPLICANTS를 결정하는 단계와 ESSENTIAL PRIME IMPLICANTS를 구하는 두 과정이 필요한데, MASK 方法은 PRIME IMPLICANTS를 求하는 方法으로, COST 결정법은 2단계인 ESSENTIAL PRIME IMPLICANTS를 求하는 方法으로 사용된다. MASK 方法과 COST 결정법에 의한 COST 산출方法이 무엇이며, 이 MASK 方法과 COST 결정법을 利用하여, 多出力함수의 최소화 design을 program 하는데 있다.

On the computerization of minimizing the Logic circuit by the MASK and the COST Table Method

Chang Sa Won
Det. of computer science
(Received June 10 1980)

〈**Abstract**〉

The minimization of the multi-output switching functions involving many variables and many functions is a difficult task. This paper is concerned with a new minimization procedure based on the MASK method and a new Cost Table that allows this process to be implemented with reduced computational effort. This procedure is applicable to both manual and computer programed minimization. the details of this algorithm is presented and illustrated by some example.

I. 서 론

Switching 함수의 최소화는 논리회로의 설계에 있어서 중요한 역할을 담당한다. 이미 최소의 S.O.P (Sum of products)을 구하는 데에는 여러가지의 方法(Algorithm)이 연구되어 왔으나 Switching 함수의 최소에 대해서 이미 설계된 계산기로서는 기억용량과 실행시간 같은점에 한계성을 가진다.

그러므로 本論文에서는 MASK 方法이라고 불리는 bit by bit operation(즉, AND, OR와 Exclusive-OR Operation)을 이용하여 PI(prime Implicants)를 찾아내는데 있다.

多出力論理함수에 대해 P.I(Prime Implicants)의 최적의 선택 (optimal selection)은 교차표(Intersection Table) 및 가격표 (Cost Table)와 부가격표(Subcost Table)을 사용하므로서 실행된다.

최적(optimal)의 가격(cost)은 교차표(Intersec-

tion Table), 가격표(Cost table), 부가격표(subcost Table)를 사용하므로서 고려될 수 있다.

가격의 판정(COST Criterion)은 교차표(Intersection Table)을 교차(cover)하는 P.I(Prime Implicant) 선택의 우선순서에 기인된다.

Ⅱ. P.I를 구하는 方法(The Method of Generating Prime Implicants)

switching 함수의 모든 P.I(Prime Implicants)를 찾아내는 方法 (Algorithm)은 다음과 같다.

만일 여러개의 주어진 minterms 〈민텀〉들에서 한개의 P.I(Prime Implicant)로 줄어 들수 있다면 다음의 조건들을 만족시켜야만 한다.

(1) $M(i)$ .AND. $M(f)=M(i)$

(2) $M(t)$ .OR. $T=M(f)$ (For $t=i,\ i+1,\cdots,\ f$, and $T=M(i)$ .Ex-OR. $M(f)$ 〈민텀〉

여기서 $M(i)$와 $M(f)$는 각각 최소의 (initial) minterm〈밀텀〉와 최후의 민텀(final minferm)을 나타낸다.

*. AND. .EX-OR.와 .OR. 같은 operations은 bit by bit operation을 실행한다.

〈AND operation의 例〉:

$$\begin{array}{rr} & 1\ 0\ 1\ 0\ 1\ 1 \\ \text{AND} & 0\ 1\ 1\ 0\ 1\ 0 \\ \hline & 0\ 0\ 1\ 0\ 1\ 0 \end{array}$$

만약 제거된 빗트(eliminuted bit)의 數가 n이라면 조건 〈2〉를 만족하는 민텀들(minterms)의 數는 2^n개 되어야 한다.

(例) (ii) Modified petrick method.

MODIFIED PETRICK METHOD

pHme Implicants \ minterms	6	7	15	38	46	47
a	✓	✓				
b			✓			✓
c		✓	✓			
d					✓	✓
e				✓	✓	
f	✓			✓		

Fig. 1 modified petrick method

$(a+f)(a+c)(b+c)(e+f)(d+e)(b+d)=1$

$abe+abdf+acde+bcef+cdf=1$

$\overline{(a+f)(a+c)(b+c)(e+f)(d+e)(b+d)}=1$

$\bar{a}.\bar{f}+\bar{a}.\bar{e}+\bar{b}.\bar{c}+\bar{e}.f+\bar{d}\bar{e}+\bar{b}\bar{d}=0$

$\bar{a}.(\bar{f}+\bar{e})+\bar{b}(\bar{c}+\bar{d})+\bar{e}(\bar{f}+\bar{d})=0$

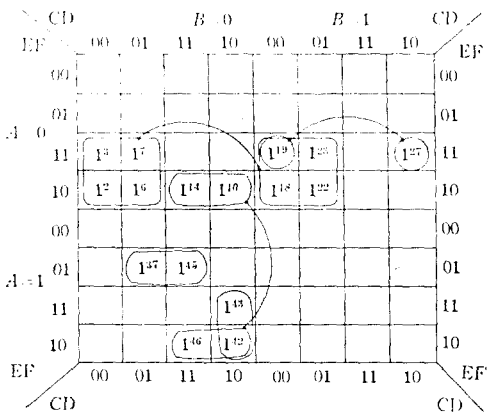
$\therefore\ a=b=e=1$

〈example〉

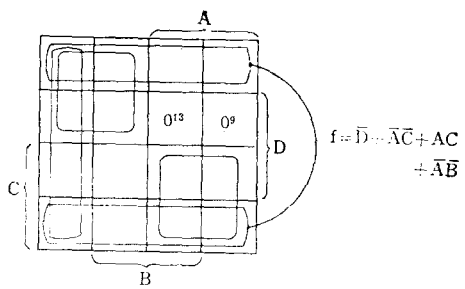
(iii) Karnaugh-map

$f(A,B,C,D,E,F)=\Sigma m(2,3,6,7,10,14,18,19,22,23,27,37,42,43,45,46)$

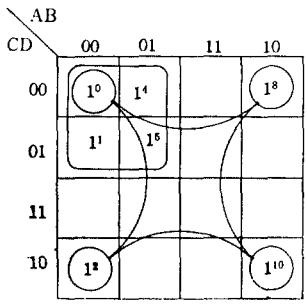
〈solution〉



$f=\bar{A}B\bar{D}EF\leftarrow m_{19}+m_{27}$
$+A\bar{B}D\bar{E}F\leftarrow m_{37}+m_{45}$
$+A\bar{B}CDE\leftarrow m_{42}+m_{43}$
$+\bar{B}CE\bar{F}\leftarrow m_{10}+m_{14}+m_{42}+m_{46}$
$+\bar{A}\bar{C}E\leftarrow\Sigma m(2,3,6,7,18,19,22,23)$

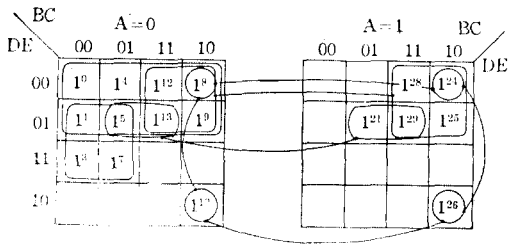


2. $f(A,B,C,D)=\pi M(7,9,13)$
〈solution〉



3. $f(A,B,C,D)=\Sigma m(0,1,2,4,5,8,10)$
〈solution〉

4. $f(A,B,C,D,E)=\Sigma m(0,1,3,4,5,7,8,9,10,12,13,21,24,25,26,28,29)$



$f=C\bar{D}E\leftarrow\Sigma m(5,13,21,29)$
$+\bar{A}\bar{B}E\leftarrow\Sigma m(1,3,5,7)$
$+\bar{A}\bar{D}\leftarrow\Sigma m(0,1,4,5,8,9,12,13)$
$+B\bar{C}\bar{E}\leftarrow\Sigma m(8,10,24,26)$
$+B\bar{D}\leftarrow\Sigma m(8,9,12,13,24,25,28,29)$

Fig. 2 Karnaugh-map method

Table 1 Minimization with 'don't care' condition

List 1

	U	*V*	*W*	*X*	*Y*	*Z*
5	0	0	0	1	0	1✓
9	0	0	1	0	0	1✓
20	0	1	0	1	0	0✓
40	1	0	1	0	0	0√
7	0	0	0	1	1	1✓
11	0	0	1	0	1	1✓
13	0	0	1	1	0	1✓
21	0	1	0	1	0	1✓
25	0	1	1	0	0	1✓
37	1	0	0	1	0	1✓
41	1	0	1	0	0	1✓
42	1	0	1	0	1	0✓
56	1	1	1	0	0	0✓
15	0	0	1	1	1	1✓
27	0	1	1	0	1	1✓
29	0	1	1	1	0	1✓
39	1	0	0	1	1	1✓
43	1	0	1	0	1	1✓
45	1	0	1	1	0	1✓
53	1	1	0	1	0	1✓
57	1	1	1	0	0	1✓
58	1	1	1	0	1	0✓
31	0	1	1	1	1	1✓
47	1	0	1	1	1	1✓
59	1	1	1	0	1	1✓
61	1	1	1	1	0	1✓
63	1	1	1	1	1	1✓

List 2

	U	*V*	*W*	*X*	*Y*	*Z*
5, 7	0	0	0	1	—	1✓
5, 13	0	0	—	1	0	1✓
5, 21	0	—	0	1	0	1✓
5, 37	—	0	0	1	0	1✓
9, 11	0	0	1	0	—	1✓
9, 13	0	0	1	—	0	1✓
9, 25	0	—	1	0	0	1✓
9, 41	—	0	1	0	0	1✓
20, 21	0	1	0	1	0	— *A*
40, 41	1	0	1	0	0	—✓
40, 42	1	0	1	0	—	0✓
40, 56	1	—	1	0	0	0✓
7, 15	0	0	—	1	1	1✓
7, 39	—	0	0	1	1	1✓
11, 15	0	0	1	—	1	1✓
11, 27	0	—	1	0	1	1✓
11, 43	—	0	1	0	1	1✓
13, 15	0	0	1	1	—	1✓
13, 29	0	—	1	1	0	1✓
13, 45	—	0	1	1	0	1✓
21, 29	0	1	—	1	0	1✓
21, 53	—	1	0	1	0	1✓
25, 27	0	1	1	0	—	1✓
25, 29	0	1	1	—	0	1✓
25, 57	—	1	1	0	0	1✓
37, 39	1	0	0	1	—	1✓
37, 45	1	0	—	1	0	1✓
37, 53	1	—	0	1	0	1✓
41, 43	1	0	1	0	—	1✓
41, 45	1	0	1	—	0	1✓
41, 57	1	—	1	0	0	1✓
42, 43	1	0	1	0	1	—✓

	U	V	W	X	Y	Z
42,58	1	—	1	0	1	0✓
56,57	1	1	1	0	0	—✓
56,58	1	1	1	0	—	0✓
15,31	0	—	1	1	1	1✓
15,47	—	0	1	1	1	1✓
27,31	0	1	1	—	1	1✓
27,59	—	1	1	0	1	1✓
29,31	0	1	1	1	—	1✓
29,61	—	1	1	1	0	1✓
39,47	1	0	—	1	1	1✓
43,47	1	0	1	—	1	1✓
43,59	1	—	1	0	1	1✓
45,47	1	0	1	1	—	1✓
45,61	1	—	1	1	0	1✓
53,61	1	1	—	1	0	1✓
57,59	1	1	1	0	—	1✓
57,61	1	1	1	—	0	1✓
58,59	1	1	1	0	1	—✓
31,63	—	1	1	1	1	1✓
47,63	1	—	1	1	1	1✓
59,63	1	1	1	—	1	1✓
61,63	1	1	1	1	—	1✓

List 3

	U	V	W	X	Y	Z
5, 7/13,15	0	0	—	1	—	1✓
5, 7/37,39	—	0	0	1	—	1✓
5,13/ 7,15	0	0	—	1	—	1✓
5,13/21,29	0	—	—	1	0	1✓
5,13/37,45	—	0	—	1	0	1✓
5,21/13,29	0	—	—	1	0	1✓
5,21/37,53	—	—	0	1	0	1✓
5,37/ 7,39	—	0	0	1	—	1✓
5,37/13,45	—	0	—	1	0	1✓
5,37/21,51	—	—	0	1	0	1✓
9,11/13,15	0	0	1	—	—	1✓
9,11/25,27	0	—	1	0	—	1✓
9,13/11,15	0	0	1	—	—	1✓
9,13/25,29	0	—	1	—	0	1✓
9,13/41,45	—	0	1	—	0	1✓
9,25/11,27	0	—	1	0	—	1✓
9,25/13,29	0	—	1	—	0	1✓
9,25/41,57	—	—	1	0	0	1✓
9,41/11,43	—	0	1	0	—	1✓
9,41/13,45	—	0	1	—	0	1✓
9,41/25,27	—	—	1	0	0	1✓

	Z	Y	X	W	V	U
40,41/42,43	1	0	1	0	—	—✓
40,41/56,57	1	—	1	0	0	—✓
40,42/41,43	1	0	1	0	—	—✓
40,56/41,57	1	—	1	0	0	—✓
7,15/39,47	—	0	—	1	1	1✓
7,39/15,47	—	0	—	1	1	1✓
11,15/27,31	0	—	1	—	1	1✓
11,15/43,47	—	0	1	—	1	1✓
11,27/15,31	0	—	1	—	1	1✓
11,43/15,47	—	0	1	—	1	1✓
11,43/27,59	—	—	1	0	1	1✓
13,15/29,31	0	—	1	1	—	1✓
13,15/45,47	—	0	1	1	—	1✓
13,29/15,31	0	—	1	1	—	1✓
13,29/45,61	—	—	1	1	0	1✓
13,45/15,47	—	0	1	1	—	1✓
13,45/29,61	—	—	1	1	0	1✓
21,53/29,61	—	1	—	1	0	1✓
25,29/27,31	0	1	1	—	—	1✓
25,29/57,61	—	1	1	—	0	—✓
25,57/27,59	—	1	1	0	—	1✓
25,57/29,61	—	1	1	—	0	1✓
37,39/45,47	1	0	—	1	—	1✓
37,45/39,47	1	0	—	1	—	1✓
37,45/53,61	1	—	—	1	0	1✓
37,45/45,61	1	—	—	1	0	1✓
41,43/45,47	1	0	1	—	—	1✓
41,45/43,47	1	0	1	—	—	1✓
41,45/57,61	1	—	1	—	0	1✓
41,57/43,59	1	—	1	0	—	1✓
41,57/45,61	1	—	1	—	0	1✓
42,43/58,59	1	—	1	0	1	—✓
42,58/43,59	1	—	1	0	1	—✓
42,58/45,61	1	—	1	—	0	1✓
56,57/58,59	1	1	1	0	—	—✓
56,58/57,59	1	1	1	0	—	—✓
15,31/47,63	—	—	1	1	1	1✓
15,47/31,63	—	—	1	1	1	1✓
27,31/59,63	—	1	1	—	1	1✓
27,59/31,63	—	1	1	—	1	1✓
29,31/61,63	—	1	1	1	—	1✓
29,61/31,63	—	1	1	1	—	1✓
43,47/59,63	1	—	1	—	1	1✓
43,59/47,63	1	—	1	—	1	1✓
45,47/61,63	1	—	1	1	—	1✓
45,61/47,63	1	—	1	1	—	1✓

	U	V	W	X	Y	Z	
57, 59/61, 63	1	1	1	—	—	1	✓
57, 61/59, 63	1	1	1	—	—	1	✓

List 4

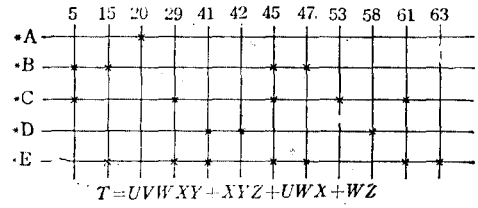
	U	V	W	X	Y	Z	
5, 7/13, 15/37, 39/45, 47	—	0	—	1	—	1	*B*
5, 7/37, 39/13, 15/45, 47	—	0	—	1	—	1	
5, 13/ 7, 15/37, 39/45, 47	—	0	—	1	—	1	
5, 13/21, 29/37, 45/53, 61	—	—	—	1	0	1	C
5, 13/37, 45/ 7, 15/39, 47	—	0	—	1	—	1	
5, 13/37, 45/21, 53/29, 61	—	—	—	1	0	1	
5, 21/13, 29/37, 45/53, 61	—	—	—	1	0	1	
5, 21/37, 53/13, 29/45, 61	—	—	—	1	0	1	
5, 37/ 7, 39/13, 15/45, 47	—	0	—	1	—	1	
9, 11/13, 15/25, 29/27, 31	0	—	1	—	—	1	✓
9, 11/13, 15/41, 43/45, 47	—	0	1	—	—	1	✓
9, 11/25, 27/13, 15/29, 31	0	—	1	—	—	1	✓
9, 4/25, 27/41, 57/43, 59	—	—	1	0	—	1	✓
9, 13/11, 15/25, 29/27, 31	0	—	1	—	—	1	✓
9, 13/11, 15/41, 43/45, 47	—	0	1	—	—	1	✓
9, 13/25, 29/11, 15/27, 31	0	—	1	—	—	1	✓
9, 13/25, 29/41, 45/57, 61	—	—	1	—	0	1	✓
9, 13/41, 45/11, 15/43, 47	—	0	1	—	—	1	✓
9, 13/41, 45/25, 29/57, 61	—	—	1	—	0	1	✓
9, 25/11, 27/13, 15/27, 31	0	—	1	—	—	1	✓
9, 25/41, 57/11, 43/27, 59	—	—	1	0	—	1	✓
9, 25/41, 57/13, 29/45, 61	—	—	1	—	0	1	✓
9, 41/11, 43/13, 15/45, 47	—	0	1	—	—	1	✓
9, 41/11, 43/25, 57/27, 29	—	—	1	0	—	1	✓
40, 41/42, 43/56, 57/58, 59	1	—	1	0	—	—	*D*
40, 41/56, 57/42, 43/58, 59	1	—	1	0	—	—	
11, 15/27, 31/43, 47/59, 63	—	—	1	—	1	1	✓
11, 15/43, 47/27, 31/59, 63	—	—	1	—	1	1	✓
11, 43/27, 59/15, 31/47, 63	—	—	1	—	1	1	✓
13, 15/29, 31/45, 47/61, 63	—	—	1	1	—	1	✓
13, 15/45, 47/29, 31/61, 63	—	—	1	1	—	1	✓
13, 29/45, 61/15, 31/47, 63	—	—	1	1	—	1	✓
25, 29/27, 31/57, 59/61, 63	—	1	1	—	—	1	✓
25, 29/57, 61/27, 31/59, 63	—	1	1	—	—	1	✓
25, 57/27, 29/29, 31/61, 63	—	1	1	—	—	1	✓
41, 43/45, 47/57, 59/61, 63	1	—	1	—	—	1	✓
41, 45/57, 61/43, 47/59, 63	1	—	1	—	—	1	✓
41, 57/43, 59/45, 47/61, 63	1	—	1	—	—	1	✓

List 5

9, 11/13, 15/25, 29/27/31							
41, 43/45, 47/57, 59/61, 63	—	—	1	—	—	1	*E*
9, 11/13, 15/41, 43/45, 47							
25, 29/27, 31/57, 59/61, 63	—	—	1	—	—	1	
9, 4/25, 27/41, 57/43, 59							
13, 15/29, 31/45, 47/61, 63	—	—	1	—	—	1	
9, 13/25, 29/41, 45/57, 61							
11, 15/27, 31/43, 47/59, 63	—	—	1	—	—	1	

〈**Prime implicant set**=$\bar{U}V\bar{W}X\bar{Y}$, $\bar{V}XZ$, $X\bar{Y}Z$, $UW\bar{X}$, WZ〉

Table 2 Prime Implicant table with 'don't cares'



If the reader draws the Karnaugh map for this problem it will be clear that the map method involves far less computation, and the use of 'don't care' conditions is immediately apparent. Hower, the strength of the tabular method lies in its easy extension to large-variable problems.

Other tabular techniques have been developed for minimizing Boolean functions, notably those due to Roth,[7] Curtis,[8] and Samson and Mills.[9] The McCluskey method, however, is simple to apply and is well suited for both hand and machine computation.

Ⅲ. 다중출력 함수의 디자인

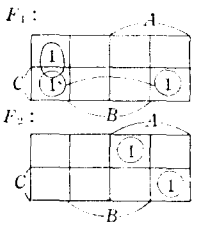
대부분의 COMBINATIONAL 회로의 디자인 문제는 A SET OF SWITCHING FUNCTION이라고 일컬어지는 2개 이상의 다중 출력 신호를 가지고 있다. 이러한 다중 출력 함수를 디자인 하는 문제에 가장 간단하게 도달 할 수 있는 방법은, 함수들의 입력을 서로 연관시키지 않고, 독립적으로 그 출력 함수의 주어진 입력에만 의존하여 구하는 방법이다. 그러나 이러한 방법은 다중 출력 함수의 주어진 입력들을 서로 연관시켜서 디자인 하는 방법보다 경제적으로 비용이 많이 든다.

다음 두개의 출력 함수를 비교해 보자.

$F_1(A,B,C)=\Sigma m\ (0,1,5)$

$F_2(A,B,C)=\Sigma m\ (5,6)$

두 출력 함수를 입력을 연관시키지 않고, 독립적으로 구해보면,

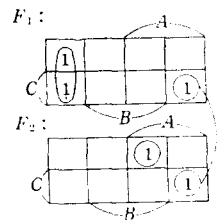


$F_1 = \bar{A}\bar{B}+\bar{B}C$ (6 c.u)

$F_2 = A\bar{B}C+AB\bar{C}$ (8 c.u)

로서 총 14 C.U가 소요된다.

두 출력 함수를 입력을 서로 연관시켜서 구해보면



$F_1 = \bar{A}\bar{B}+ABC$ (7 c.u)

$F_2 = ABC+AB\bar{C}$ (5 c.u)

로서 총 12 C.U가 소요된다.

여기서 C.U는 GATE의 입력의 수가 비용인 비용의 단위이다.

위에서 본 바와 같이 다중 출력 함수를 주어진 입력을 서로 연관시켜서 디자인 하면, 입력을 연관시키지 않고 독립적으로 각각 다중출력 함수를 디자인하는 방법보다 비용이 적게듬을 알 수 있다. 다중출력 함수를 디자인하는 일반적인 방법은 다음의 두 단계를 거쳐야 한다.

1. PRIME IMPLICANTS를 결정
2. 다중출력 함수를 표시하기 위한 ESSENTIAL PRIME IMPLICANTS를 결정.

위의 두 단계를 처리하는 방법은 여러가지 이론에 따라 다르다. 이 논문에서는 1단계로 MASK 방법, 2단계로 COST 결정법을 사용하여, 다중 출력 함수의 최소화 디자인을 PROGRAM 하였다.

Ⅳ. MASK 방법

MASK 방법은 PRIME IMPLICANTS를 구하는, 과거의 이론보다 획기적이며 간단한 방법이다.

과거의 모든 이론이 그러하듯이 MASK 방법도 입력을 줄일 수 있는 가장 간단하면서도 기본적인 BOOLEAN ALGEBRA인

$$AB+A\bar{B}=A(B+\bar{B})=A$$

에 기본적인 이론적 근거를 두고 있으며, 입력이나 출력의 수가 증가하더라도, 손 쉽게 PRIME IMPLICANTS를 구할 수 있다. MASK방법은 다음과 같은 과정에 의해 PRIME IMPLICANTS를 구한다.

1. 입력으로 주어진 MINTERM 중에서 두 개의 민텀(MINTERM)

   LM : LOWEST MINTERM

   HM : HIGHEST MINTERM

   을 선택한다.

2. 선택된 두개의 MINTERM : LM, HM이 다음의 관계를 만족시키는 가를 확인한다.

   LM .AND. HM=LM

3. 선택된 두 개의 MINTERM이 위 관계를 만족시키면, MASK를 다음 관계식에 의해서 구한다.

   MASK=LM .EX-OR. HM

   MASK를 구한다음, MASK 속의 1의 갯수를 N이라고 하면 LM과 HM 사이의 MINTERM의 갯수가 LM과 HM을 포함하여 2^N 이상이 되어야 한다.

4. 위의 관계를 만족하면 LM과 HM을 포함한 LM과 HM사이의 모든 MINTERM들과 MASK를 OR OPERATION 시킨다.

   MINTERM .OR. MASK=HM

   위의 관계를 만족시키는 MINTERM의 갯수가 2^N이 되면 만족시키는 MINTERM들로 PRIME IMPLICANT를 형성한다.

MASK 방법은 위의 과정을 모든 MINTERM이 비교될 때까지 반복하여 PRIME IMPLICANT를 형성한다.

MASK 방법의 MASK란 제거된 BIT를 표시하는 방법이며, MASK 속의 1의 갯수는 제거된 BIT의 수를 나타낸다. MASK 방법은 손으로 처리하는 것보다 계산기로 처리하기 편한 방법이다.

Ⅴ. COST 결정법

PRIME IMPLICANTS가 MASK 방법에 의해서 구해진 다음에는, 비용이 가장 적게 드는 다중 출력 함수를 표시하기 위한 ESSENTIAL PRIME IMPLICANT를 결정해야 한다. COST 결정법이란 비

용이 가장 적게 드는 다중 출력 함수의 ESSENTIAL PRIME IMPLICANT를 구하는 방법으로, 과거로부터 여러가지 발표된 COST 결정법들이 있지만, 이 논문에서는 다음과 같은 방법으로 COST를 결정한다.

1. PRIME IMPLICANT가 포함하고 있는 MINTERM 중에서, 그 PRIME IMPLICANT가 COVER하는 함수들 각각의 MINTERM의 수를 합한 수를 COST로 하며, 각 함수들의 DON'T CARE MINTERM의 수는 제외한다.

Ⅵ. FLOW CHART

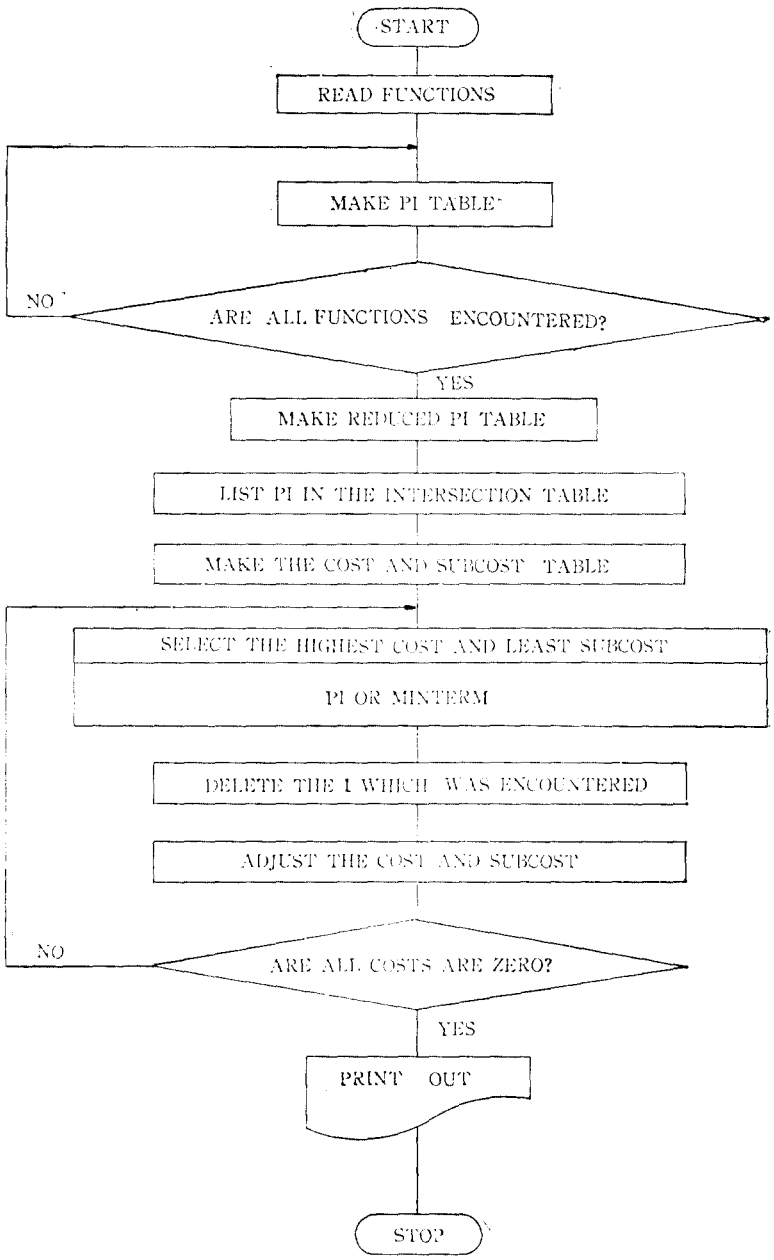


Fig. 3 The overall flow chart for the computer program.

또한 각 MINTERM이 COVER하는 함수의 수를 COST로 잡으며, 그 MINTERM을 DON'T CARE MINTERM으로 하는 함수의 수는 제외한다.

2. PRIME IMPLICANT가 포함하고 있는 MINTERM 중에서, 그 PRIME IMPLICANT가 COVER하지 않는 함수들 각각의 MINTERM의 수의 합을 SUBCOST로하며, 역시 각 함수들의 DON'T CARE MINTERM의 수는 제외한다.

위 COST 결정법에 의해서 COST와 SUBCOST가 결정되면, 우선 COST가 가장 큰 PRIME IMPLICANT를 ESSENTIAL PRIME IMPLICANT로 택한다. COST가 크다는 것은, 그 값을 갖는 PRIME IMPLICANT가 각 함수들의 MINTERM을 많이 포함한다는 것을 의미하므로, 그 PRIME IMPLICANT는 비용이 적게드는 다중출력함수의 최소화 디자인의 ESSENTIAL PRIME IMPLICANT로 적당하기 때문이다.

COST 비교시 가장 큰 COST를 갖는 PRIME IMPLICANT가 여러개 있으면, 그 중에서 SUBCOST가 가장 적은 것을 ESSENTIAL PRIME IMPLICANT로 택한다. SUBCOST가 적다는 것은 그 값을 갖는 PRIME IMPLICANT가 COVER하는 함수가 상대적으로 많다는 것을 의미하므로 그 PRIME IMPLICANT를 나타내는 GATE를 여러 출력 함수가 공동사용함으로써 비용이 적게 들기 때문이다. 또한 COST 비교시 PRIME IMPLICANTS와 MINTERM들이 동시에 가장 큰 값을 가진다면, MINTERM들의 SUBCOST가 더 작더라도, 이때는 PRIME IMPLICANTS 중에서 ESSENTIAL PRIME IMPLICANT를 구한다.

SUBCOST까지 같은 경우에는 임의 결정으로 임의로 하나의 PRIME IMPLICANT를 ESSENTIAL PRIME IMPLICANT로 택한다.

Ⅲ. 계산기 처리를 위한 PROGRAMING

PROGRAM은 MASK 방법과 COST 결정법의 원리를 적용하였으며, 다음에 보이는 여러가지 표(TABLE)를 만들어 가는 과정에 의하여 PROGRAMING 된다.

PROGRAM은 다음에 보이는 바와 같이 여러가지 다양한 표를 가지고 있으며, 그런 표가 의미하는 내용은 다음과 같다.

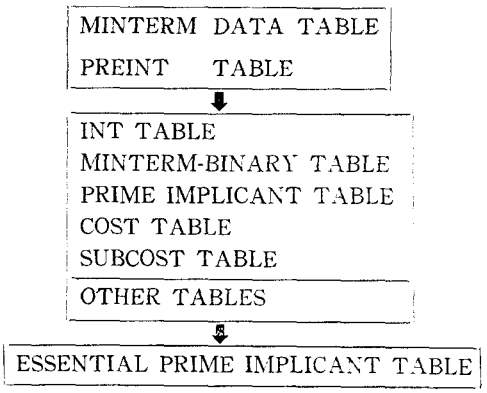


Fig. 4 Procedure of Logic design

- MINTERM DATA TABLE : 각 함수들에게 입력으로 주어진 10진수 MINTERM들을 종합하여 크기에 의거하여 순서적으로 배열하여 놓은 표이다.
- PREINT TABLE : 주어진 10진수 MINTERM이 COVER 하는 함수들을 진입력과 DON'T CARE 입력을 구별하여 놓은 표이다.
- INT TABLE: PREINT TABLE에 주어진 함수들의 DON'T CARE 입력을 진입력으로 표시하여 놓은 표이다.
- MINTERM-BINARY TABLE: MINTERM DATA TABLE에 주어진 10진수 MINTERM들을 2진수 MINTERM으로 바꾸어 표시해 놓은 표이다.
- PRIME IMPLICANT TABLE: MASK 방법에 의해 구해진 여러개의 PRIME IMPLICANT를 표시하는 표이다.
- COST TABLE: COST 결정법에 의해 구해진 PRIME IMPLICANT와 MINTERM들의 COST를 나타내는 표이다.
- SUBCOST TABLE: COST 결정법에 의해 구해진 PRIME IMPLICANT와 MINTERM들의 SUBCOST를 표시하여 놓은 표이다.
- ESSENTIAL PRIME IMPLICANT TABLE: COST 결정법에 의해 구해진 ESSENTIAL PRIME IMPLICANTS를 표시하여 놓은 표이다.

다중 출력 함수가 다음과 같이 진입력과 DON'T CARE 입력을 같이 포함하는 함수로 주어졌다고 하자.

$$F_1(A,B,C,D)=\Sigma m(0,5,7,14,15)+d(1,6,9)$$
$$F_2(A,B,C,D)=\Sigma m(13,14,15)+d(1,6,9)$$

	F_1	F_2	F_3
0	1		1
1	×	×	1
5	1		1
6	×	×	
7	1		1
9	×	×	×
13		1	×
14	1	1	×
15	1	1	

PREINT TABLE

F_1	F_2	F_3
1		1
1	1	1
1		1
1	1	
1		1
1	1	1
	1	1
1	1	1
1	1	

INT TABLE

MINTERM DATA TABLE

A	B	C	D
0	0	0	0
0	0	0	1
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	1
1	1	0	1
1	1	1	0
1	1	1	1

MINTERM BINARY TABLE

$F_3(A,B,C,D)=\Sigma m(0,1,5,7)+d(9,13,14)$

주어진 다중 출력 함수의 입력에 의해 4개의 TABLE이 구해졌다. 다음 단계는 구해진 INT TABLE과 MINTERM BINARY TABLE을 사용하여, MASK 방법에 의해서 PRIME IMPLICANTS를 차례차례 구하여 PRIME IMPLICANT TABLE 완성시킨다. MASK 방법에 의하여 PRIME IMPLICANTS를 구할때 단일 출력 함수와는 달리, 구하여진 PRIME IMPLICANT가 포함하는 MINTERM들이 공통된 함수들을 공유해야만 된다는 것이다. 그래야만 구해진 PRIME IMPLICANT가 실질적인 것이되며, 함수를 공유하지 못하는 PRIME IMPLICANT는 버려져야 한다.

PRIME IMPLICANT TABLE과 동시에 구해진 PRIME IMPLICANT가 공유하는 함수가 무엇인지 나타내는 TAGA TABLE과 구해진 PRIME IMPLICANT가 나타내는 GATE의 입력 논리 및 제거된 BIT를 표시하는 TAGB TABLE을 구해야 한다.

- TAGA TABLE: 각 PRIME IMPLICANT가 포함하고 있는 모든 MINTERM을 공유하고 있는 함수들을 표시해 놓은 표이다.
- TAGB TABLE: 각 PRIME IMPLICANT가 나타내는, GATE의 입력 논리및 제거된 BIT들을 표시해 놓은 표이다.

0	1								
1	1	1	1	1					
5		1		1			1		
6					1	1			
7					1		1		
9		1	1					1	
13		1						1	1
14					1	1			1
15					1			1	1

Fig.5 PRIME IMPLICANT TABLE

F_1	1	0	1	1	1	1	1	0	0	1
F_2	0	0	1	0	0	1	0	1	1	1
F_3	1	1	1	1	0	0	1	1	0	0

Fig.6 TAGA TABLE

A	0	×	×	0	×	×	0	1	1	1
B	0	×	0	×	1	1	1	×	1	1
C	0	0	0	0	1	1	×	0	×	1
D	×	1	1	1	×	0	1	1	1	×

Fig.7 TAGB TABLE

PRIME IMPLICANT TABLE을 구할때 PRIME IMPLICANT가 포함하는 MINTERM 들을 공유하는 함수들이 존재하면, 그 다음 단계로서 구한 PRIME IMPLICANT가 앞서 구해는 것들에의 포함관계를 꼭 확인해 보아야 한다. 포함 관계를 따질때는 PRIME IMPLICANT가 앞서 구해논 것에 포함되며, 동시에 TAGA TABLE의 공유된 함수들이 같은가를 확인하여야 한다. 이러한 두 관계가 동시에 성립되어야, 구한 PRIME IMPLICANT는 실질적인 것이 못되어 제거된다. INT TABLE과 MINTERM-BINARY TABLE을 사용하여, MASK방법에 의해 PRIME IMPLICANT TABLE이 구해졌다. 그러면 그 다음 단계로서 COST 결정법을 써서 COST와 SUBCOST TABLE을 만들어 ESSENTIAL PRIME IMPLICANTS를 구하는 과정이 된다.

COST와 SUBCOST들은 PREINT TABLE PRIME IMPLICANT TABLE, TAGA TABLE을 사용하여 구하며, 구하는 방법은 COST 결정법에 설명한 바와 같이 구한다.

COST와 SUBCOST TABLE이 완성되면 COST 결정법에서 언급한 바와 같이 COST와 SUBCOST들을 비교하여 ESSENTIAL PRIME IMPLICANT을 구하며, 또 다른 ESSENTIAL PRIME IMPLICANT를 구할 COST와 SUBCOST TABLE을 작성하기 위해서, 구한 ESSENTIAL PRIME IMPLICANT가 COVER하는, TAGA TABLE의 함수들의 MINTERM을 제거하여 버린다.

제거된 MINTERM의 함수들이 표시된 새로운 PREINT TABLE, PRIME IMPLICANT TABLE,

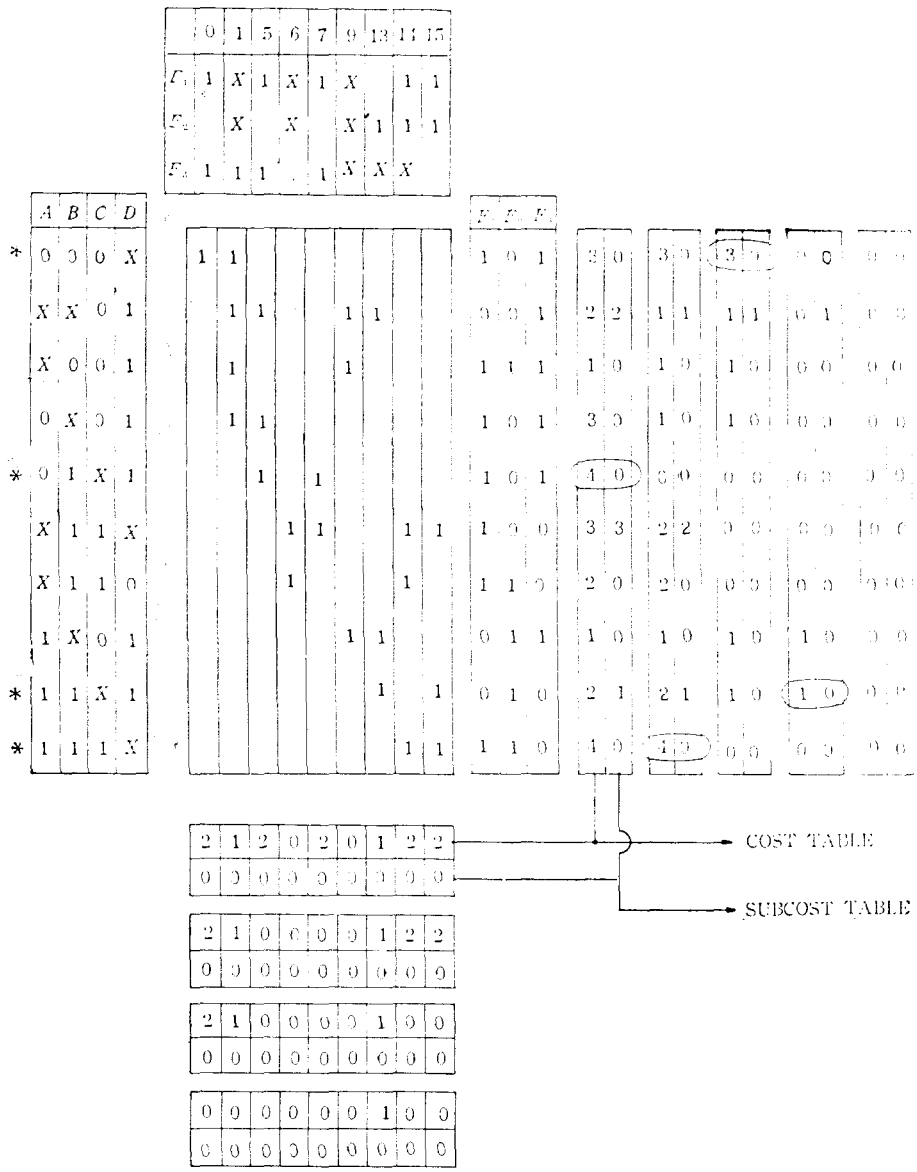


Fig. 8 Selection of the prime Implicant and minterms

	0	1	5	6	7	9	13	14	15
F_1	1	×	1	×	1	×		1	1
F_2		×		×		×	1	1	1
F_3	1	1	1		1	×	×	×	

	0	1	5	6	7	9	13	14	15
	1	1							
			1		1				
							1		1
								1	1

A	B	C	D
0	0	0	×
0	1	×	1
1	1	×	1
1	1	1	×

F_1	F_2	F_3
1	0	1
1	0	1
0	1	0
1	1	0

Fig. 9 Essential Prime Implicant

TAGA TABLE을 사용하여, COST 결정법에 의해 새로운 COST와 SUBCOST TABLE을 작성하여, ESSENTIAL PRIME IMPLICANT를 구한다. 이러한 과정은 PREINT TABLE의 모든 MINTERM의 함수들이 제거될때 즉 모든 PRIME IMPLICANT의 COST들이 0의 값을 가질때까지 반복되어 실시된다.

COST 결정법에 의하여 모든 ESSENTIAL PRIME IMPLICANTS가 구해졌다. 다음 단계는 ESSENTIAL PRIME IMPLICANT TABLE을 작성하는 것이다.

〈Fig. 9와 같은〉 ESSENTIAL PRIME IMPLICANT TABLE이 완성되었다. 그 다음에는 각각의 함수마다 독립적으로, ESSENTIAL PRIME IMPLICANT가 서로 포함 관계를 가지고 있는가를 확인하고, 만약 포함 관계가 성립한다고 하면, 포함되는 ESSENTIAL PRIME IMPLICANT를 그 함수에서는 제거하여 버린다. 그 이유는 포함되는 ESSENTIAL PRIME IMPLICANT가 여러 함수가 공통으로 사용하는 것이더라도, 그것은 그 함수에게는 두번씩 필요없는 입력을 넣어주는 결과가 되어, 다중 출력 함수의 결선비용이 커지므로, 포함되는 ESSENTIAL PRIME IMPLICANT는 제거해 버린다.

최종적 결과로서 다중 출력 함수를 표시해 보자.

$$F_1=\bar{A}C\bar{B}+\bar{A}BD+ABC$$

$$F_2=ABD+ABC$$

$$F_3=\bar{A}C\bar{B}+\bar{A}BD$$

다중 출력 함수를 GATE로 표시해 보자. 〈Fig. 10 참조〉

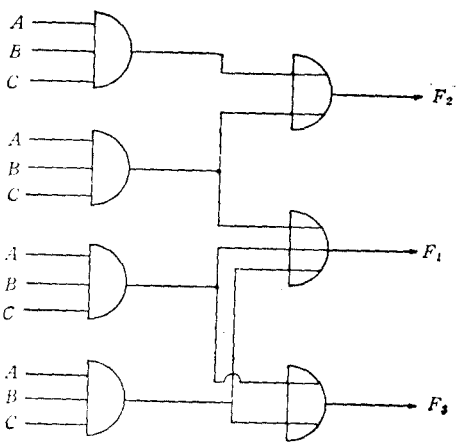


Fig. 10 Reduced multi-output logic network relations

Ⅵ. 결 론

여태까지 MASK 방법과 COST 결정법에 의한 다중 출력 함수의 최소화 디자인에 관하여 논의하였다.

MASK 방법과 COST 결정법에 의한 다중출력함수의 최소화 디자인방법은 다음과 같은 이점이 있다.

1. MASK 방법은 어느 방법보다 계산기에 적용하기에 적합한 방법으로, 어느 방법보다도 더 빨리 다중출력함수의 최소화 디자인을 계산기에 의해 구할 수 있다. 즉 다른 어떤 방법보다도 다중 출력함수의 최소화 디자인을 계산기로 처리하는 시간이 단축된다.

2. COST 결정법에 의해서 결정되는, 다른 이론과 달리 COST와 새로운 SUBCOST라는 개념을 도입하였기 때문에, 다른 어떤 이론에 의해서 디자인되는 다중출력 함수보다 더욱 간단하게 다중출력함수가 디자인 되므로, 전체적인 GATE와 결선 비용

이 적게 들어 경제적이다.

3. 계산기에 의한처리를 위한 PROGRAMING에서 논의한 바와 같이 다중출력함수의 최소화 디자인 절차는 여러가지 표를 순서에 따라 기계적으로 작성하여 나가기 때문에 이해하기 쉽고, 또한 손으로 처리하는 데도 다른 방법보다 간단하고, 빠르게 처리된다.

그러나, 여태까지 발표된 이론들에 제시된 모든 방법보다, 다중 출력 함수를 최소화 디자인하는데 있어서, 이해하기 쉽고, 경제적으로 비용이 적게들고, 손이나 계산기로도 처리하기 쉬운 MASK 방법과 COST 결정법은 약간의 문제점을 안고 있다.

앞으로 계속 심사 숙고하여 처리할 문제점은 COST와 SUBCOST까지 같을때, ESSENTIAL PRIME IMPLICANT를 임의 결정하는 OPTION의 경우, 보다 경제적이고 다중출력함수를 더욱 더 최소화시키는 선택방법을 찾아내는 것이다.

참 고 문 헌

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. B.L. Hulme and R.B. Worrell, "A phine inplicant algorithm with factoring," IEEE Trans Comput., Vol, C-24, pp.1129~1131, Nov. 1975.
4. V.T.Rhyne: "Fundamentals of Digital System," Prentice-Hall Englewood Clifjs MJ. 1973.
5. D.D. Givone: "Introduction to Switching Circuit Theory," Mcglaw Hill, New York 1970.

Computer Program Results

**********MULTIPLE FUNCTION MINIMIZATON PROGRAM**********

THIS IS MULTIPLE FUNCNION MINIMIZATION PROGRIAM WHICH MINIMIZES THE COST CRITERIA (NO. OF GATES) OF THE TOTAL FUNCTION USING MASK METHOD COST TABLE METHOD.

THIS MULTIPLE FUNCTION CONTAINS 3 FUNCTIONS

NAMELY

MINTERM	FA	FB	FC
0	0	0	1
3	1	0	0
4	0	0	1
5	1	1	0
7	1	1	0
10	0	1	1
13	1	1	0
14	1	1	1
15	1	1	1

AND EACH FUNCNION CONTAINS 4 VARIABLES

THE FOLLOWING IS THE LIST OF THE PRIME IMPLICANTS OBTAINED FROM DA TABLE WHERE PRIME IMPLICANTS SELECTED FROM COST TABLE ARE LABLED AS SELECTED

**** FUNCTION ** 1 ***** FA

NO.	COST	A	B	C	D	
1	2	0	×	1	1	SELECTED→$\overline{X}_1X_3X_4$
2	2	0	1	1	1	
3	8	×	1	×	1	SELECTED→X_2X_4
4	3	1	1	1	1	
5	3	1	1	1	1	
6	6	1	1	1	×	SELECTED→$X_1X_2X_3$

**** FUNCTION ** 2 ***** FB

NO.	COST	A	B	C	D	
1	8	×	1	×	1	SELECTED→X_2X_4
2	3	1	1	1	1	
3	3	1	1	1	1	
4	4	1	×	1	0	SELECTED→$X_1X_3\overline{X}_4$
5	3	1	1	1	0	
6	6	1	1	1	×	

**** FUNCTION ** 3 ***** FC

NO.	COST	A	B	C	D	
1	2	0	×	0	0	SELECTED→$\overline{X}_1\overline{X}_3\overline{X}_4$
2	4	1	×	1	0	SELECTED→$X_1X_3\overline{X}_4$
3	3	1	1	1	0	
4	6	1	1	1	×	SELECTED→$X_1X_2X_3$