연구논문집 제11권 제1호 p.23~42, 1980

Cost criteria와 MASK 方法에 의한 多出力함수의 최소화

張 師 元
전자계산학과

〈요 약〉

本 論文은 multi-output switching Function minimization에 적용하여 MASK 方法을 Algorithm에 관해 Program化를 시도하였다.

multi-output switching function의 minimization은 많은 변수의 input과 Function이 증가할 때는 어려운 경우가 된다. 여기서는 PI를 결정하기 위해 새로운 方法을 제시하며 그리고 모든 minterm을 CHeck하기 위해서 반드시 필요한 PI 선택과정을 PI 결정과 더불어 행함으로써 각 Minterm에 대해 모든 PI를 결정하는 수고를 덜고 있다.

本論文에서는 Multi-output Switching Function에 대해서, Optimal한 Selection을 intersection Table Cost Table, Snbcost Table에 의해서 구해지며, 이 Proceclure는 복잡하지 않으며 실질적으로 COMPUTER Program 응용된다.

On the minimization of the multi-ouput Sw!tching Function by the MASK method and the cost criteria

Sa Won *Chang*
Dept of Computer science

I. 서 론

Logic 설계에서, 다수의 확실한 Problem을 Multi-output Switching Function이 가지고 있다.

Single-output Switching Function로는 이미 개발된 方法中의 하나는 각 기능을 독립적으로 완전하게 사용되어졌다. 그러나 좀 더 복잡한 Function에서는 총 gate가 보다 작은 gate의 결과를 포함한다. 이 기법의 목적은 Optimal COST를 가진 multi-output Switching Function의 minimization 이다.

本 論文에서는 Optimal COST가 Intersection Table COST Table Sub COST TABLE에 의해 착안되었다.

COST Criterion은 Intersection Table을 Cover 한 Prime implicant(PI) Selection의 우선순에 따른다. 그러면 Logic Design에 있어서의 그 설계 절차에 대해 알아보고, MASK 方法을 써서 多出力설계 문제를 처리하겠다.

II. 이 론

1. Procedure of Logic design

Logic Function의 최소화는 Logic Function을 Boolean Function으로 치환하여 Boolean Operation

에 의하여 이루어진다. 이 Boolean의 성질을 이용하여 MASK 方法이 나오게 되며, 최소화는 다음과 같은 성질에 근거를 두고 있다.

Boolean Algebra

$A\bar{B}C+ABC=AC(\bar{B}+B)=AC(1)=AC$

$AB+A\bar{B}=A(B+\bar{B})=A(1)=A$

A	B	C
1	0	1
1	1	1
1	×	1

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

위의 과정은 변수 갯수가 적고 Minterm수가 적으면 아주 간단하지만 변수의 수가 증가함에 따라 점점 더 어려워진다.

이러한 난점을 해결하기 위해서 다음과 같은 여러 方法이 연구되어 왔다.

(1) Karnaugh Mapping Method

(2) Consensus

(3) Conjunctive/Disjuntive Manipulations

(4) ToPological Method

(5) Quine-Mccluskey (Q.M) Method

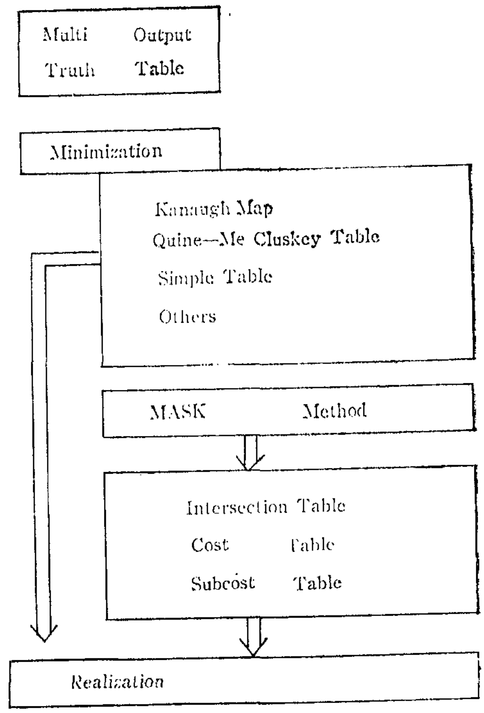
(6) MASK Method

本論文의 方法은 다음과 같은 과정을 거처서 논리함수의 최소화를 해결하고 있다.

2. Procedure of Logic Design

모든 문제에서와 같이 Truth Table을 작성하고 주어진 Truth Table에서 다음의 Minimal〈최소화〉 절차를 거친 다음 실제 회로를 설계합니다 만은 本文에서는 최근 발표된 MASK Method를 써서 Multi-ouput design을 하고 있읍니다.

MASK METHOD을 multi-output switching Function(多重出論理함수)에 적용하기 위해서는 Intersection TABLE, COST TABLE, SUBCOST TABLE 등을 만들고 있읍니다.



Ⅲ

1. PI identification process

이 과정은 주어진 논리함수를 표현하는데 적용될 수 있는 모든 PI를 구해내는 과정이다.

2. Identification of the Prime Implicants

Almost all of the procedures in references 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 algorthms for the minimization of the switching function, but the basic idea is the same. According to the Boolean theorem $AB+\bar{B}=A(B+\bar{B})=A$; all the prime implicants are generated from the combinable minterms. Conversely, the Boolean canonical form of minterms are directly obtained from the following property of the prime implicant.

⟨Definition:⟩

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

$$\frac{dF}{dx_i} = F(x_1, x_2, \cdots, x_i, \cdots, x_n) \oplus F(x_1, x_2, \cdots, \bar{x}_1, \cdots, x_n)$$

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,

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

where $G(X_{p1}, X_{p2}, \cdots, X_{p(m-n)})$ denotes the prime implicant and X_i in dicate the eliminated bit positions and $\frac{dF}{dX}$ 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,

$$\frac{dF(GX_{p1}, X_{p2}, \cdots, X_{p(m-1)}), X_1)}{dX_1}$$
$$= F(G(X_{p1}, X_{p2}, \cdots, X_{p(m-1)}), X_1) \oplus F(G(X_{p1}, X_{p2}, \cdots, X_{p(m-1)}), X_1)$$
$$= F(G(X_{p1}, X_{p2}, \cdots, X_{p(m-1)}), X_1) \oplus F(G(X_{p1}, X_{p2}, \cdots, X_{p(m-1)}), 0)$$
$$= 1 \oplus 1 = 0$$

Thus the equality holds for $n=1$.

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

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

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

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

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

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

Q.E.D.

Example 1:

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

From the above theorem,

$$\frac{pF(X_1\overline{X}_2X_4, X_3, X_5)}{dX_3} = F(X_1\overline{X}_2X_3X_4, X_5) \oplus F(X_1\overline{X}_2\overline{X}_3X_4, X_5)$$

$$\frac{dF(X_1\overline{X}_2X_3X_4, X_5)}{dX_5} = F(X_1\overline{X}_2X_3X_4X_5) \oplus F(X_1\overline{X}_2X_3X_4\overline{X}_5)$$

$$\frac{dF(X_1\overline{X}_2\overline{X}_3X_4, X_5)}{dX_5} = F(X_1\overline{X}_2\overline{X}_3X_4X_5) \oplus F(X_1\overline{X}_2\overline{X}_3X_4\overline{X}_5)$$

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

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

From the above discussion, the prime implicant can be obtained by using the following theorem 2 derived from theorem 1.

Definition:

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

Example 2:

Extract the prime implicant from the following minterms by the theorem above, 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*

1100 .OR. 1010=1010*

Check if the number of the same values which are equal to HM is 2^n.

the number of 1's······2

the number of the same values······4

Step 3 : If the given set of minterms satisfies step 1 and step 2, the prime implicant 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.

Fr a switching function, its prime implioants can be generated by this MASK cmethod. Because of its testing betwee LM and HM, the smaller cost of a pnime implicant is generated rapidly by the MASK method. The essential advattages of the MASK method are the rnpid generation of the prime implicant and a less amount of run time and memory capacity of its computer program.

The logical operations 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.

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

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 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,

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

where the lowest minterm and the highest minterm of the given set of minterms is

$F(G(X_{p1}, X_{p2}, \cdots, X_{p(m-n)}),\ 0, 0, \cdots, 0)$ and

$F(G(X_{p1}, X_{p2}, \cdots, X_{p(m-n)}), 1, 1, \cdots, 1)$

respectively.

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

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

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

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

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, $G(X_{p1}, X_{p2}, \cdots, X_{p(m-n)})$.

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

Q.E.D.

Ⅳ. PI selection process

이 과정은 PI identification 과정에서 구한 Prime Implicant들 중에서 논리함수를 표현할 수 있는 최소한의 Prime Impliсant(PI)를 선택하는 과정이다 모든 PI들은 MASK Method에 의해 산출되고 각 commonality(공통성)은 Intersection Table로부터 구해진다.

1. The Mask Method

Boolean Algebra의 성질을 이용하는 데에서 MASK 方法이 나온다. 주어진 8개의 Minterm이 하나의 Prime Implicant로 구성될 때 주어진 Boolean Algebra를 써서 줄이는 절차는 다음 도표와 같읍니다.

The Mask Method

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

$\boxed{1\ 0\ 0\ \times}$ $\boxed{1\ 0\ 1\ \times}$ → $\boxed{1\ 0\ \times\ \times}$

$\boxed{1\ 1\ 0\ \times}$ $\boxed{1\ 1\ 1\ \times}$ → $\boxed{1\ 1\ \times\ \times}$

$\boxed{1\ \times\ \times\ \times}$

여기서 직접 Prime Implicant를 만들기 위해서 다음과 같은 Operation이 행하여진다.

A	B	C	D		
1	0	0	0	1	
1	0	0	1	2	OR-MASK
1	0	1	0	3	WITH 0 1 1 1
1	0	1	1	4	
1	1	0	0	5	
1	1	0	1	6	
1	1	1	0	7	
1	1	1	1	8	

〈Fig 3〉

1. LM과 HM가 AND Operation이 되며 그 결과는 LM이 나온다. 이것이 MASK 方法의 첫번째 방법이다.

M(1) .AND. M(8)=M(1)

```
 1 0 0 0
 1 1 1 1
---------
 1 0 0 0
```

2. Fig 4

M(1). AND. M(8)=M(1)

```
 1 0 0 0
 1 1 1 1
---------
 1 0 0 0
```

M(1) .EX-OR. M(8)=MASK((0 1 1 1))

〈Fig 4〉

2. MASK Bit position을 찾아내는 方法입니다. 여기서는 Bit By Bit operation을 하게 됩니다. 그러면, 실例로 주어진 6개의 minterm에서 하나의 Prime Implicant를 선택(select)하는 2-Bit가 없어지는 例의 과정을 보겠읍니다.

2-Bit가 없어지는 例

A	B	C	D	E	F
1	0	0	1	0	0
1	0	0	0	1	0
1	0	0	1	1	0
1	0	1	0	0	1
1	0	1	1	0	0
1	0	1	1	1	0

(1) LM과 HM을 AND operation 하면 LM가 된다

```
 100100
 101110
--------
 100100
```

(2) 첫번째 조건이 만족되기 때문에 EX-OR operation을 해주면 우리는 MASK-position을 찾아낼 수 있다.

```
 100100
 101110
--------
 001010
```

그 결과를 가지고 全體에 주어진 minterm에 대해 OR-Masking을 하게 되면 다음 Table과 같이 됩니다.

1	0	1	1	1	0	*	OR-MASK
1	0	1	0	1	0		
1	0	1	1	1	0	*	
1	0	1	0	1	1		
1	0	1	1	1	0	*	
1	0	1	1	1	0	*	
1	0	×	1	×	0		

마지막으로 Check 해 보아야 할 것은 *의 갯수와 MASK Bit의 추측된 Bit(eliminated의 갯수)를 Check 합니다.

MASK Bit의 추측된 Bit의 갯수가 n이라고 가정하면 조건 (2)를 만족하는 minterm은 2^n개이다. 여기서는 *의 갯수는 2^n와 같게 되었기 때문에 *로 표시된 Minterm은 다음과 같은 Prime Implicant로 구성됨을 알 수 있읍니다.

1 0 × 1 × 0

지금까지의 MASK Method를 정리해 보면 다음과 같다.

MASK Method ; Provided that given minterms are reduced to one prime implicant they must satisfy the following conditions:

LM: the lowest minterm

HM: the highest minterm

① LM .AND. HM=LM

② Let the result of LM .EX-OR. HM be MASK, and then Count the number of 1's in MASK and let at be N. $M(K)$ .OR. MASK $=M(2{**}N)$ for $K=1, 2, \cdots, 2{**}N$ where $M(1)=$ LM and $M(2{**}N)=$HM

2. Construction of Intersection table

모든 出力에 때해 design 하고자하는 3개의 함수와 각 Function에 주어진 Minterm이 다음과 같을 때 intersection Table은 다음 표와 같읍니다.

Construction of Intersection Table

$F_\alpha(A, B, C, D)=\Sigma_m(0, 2, 8, 10)$

$F_\beta(A, B, C, D)=\Sigma_m(0, 1, 2, 11)$

$F_\gamma(A, B, C, D)=\Sigma_m(1, 4, 8)$

	0	1	2	4	8	10	11
F_α	1		1		1	1	
F_β	1	1	1				1
F_γ		1		1	1		

〈Fig 8〉

For example, F_γ에 대해 살펴보면, Minterm이 1, 4, 8이기 때문에 Intersection Table에서 1, 4, 8 자리에 다음과 같이 각각 1이 배치됩니다.

*intersection Table의 長點은 직접 commonality 〈공통성〉을 찾아낼 수 있다는데 있읍니다.

3. Cost Table & Subcost Table

Cost Table은 prime Implicant chart 옆면에 상응하는 Prime Implicant에 의해 covered(취재)된 check의 필요한 번호목록을 수록하므로써 구성된다

Optimal selection 그 방법은 Prime Implicant의 highest cost가 채택되고 subcost table이 구성된다

sub cost table은 Prime Implicant selection의 두번째 중요성을 가진다.

Ⅴ. Intersection Table

이 Intersection table은 多出力에 대해서는 MASK 方法과 같이 사용됩니다. 지금까지의 方法은 각각의 function에 대해서 Prime Implicant를 구한 다음 design 〈설계〉를 해왔지만, Intersection Table을 사용하여 한꺼번에 세개의 function에서 가지고 있는 Prime Implicant를 구할 수 있다는 장점이 있다

Intersection Table

(1) Intersection Table represents the commonality of minterms and is used in order to generate the multi-output prime implicants.

(2) Cost Table

Cost is defined as the number of the covered minterms in the Intersection Table for any prime implicant or minterm.

원래 Cost는 3가지로 정의되어 있는데 本論文에서는 gate Cost로서 入力을 위주로 한 line Cost로 정의한다. 그러면 Cost Table이 어떻게 구성되는지 알아보자.

여기서 언급되는 Cost란 Prime Minterm에 의해서 cover 되는 minterm의 갯수를 나타내고 있읍니다.

그러면 앞의 Intersection Table이 다음과 같이 주어졌을 때 MASK 方法에 의해 다음과 같은 3개의 Prime Implicant가 generating 〈나오게〉 됩니다

					0	1	2	4	8	10	11					
				F_α	1		1		1	1						
				F_β	1	1	1				1					
				F_γ		1		1	1							
A	B	C	D									α	β	γ		
×	0	×	0		1		1		1	1		1			4	3
0	0	×	0		1		1					1	1		4	0
0	0	0	×		1	1							1		0	0

2	2	2	2	1	1	1	← Cost Table
0	0	0	0	0	0	0	← subcost Table

이 각각의 Prime Implicant의 Cost를 우측에 표시하면 위와 같읍니다.

실 例로서 첫번째 구해진 Prime Implicant에서 Cost를 구하는 方法을 말하면 minterm 0, 2, 8, 10을 Cover 하고 있기에 4개의 minterm을 cover 하므로 cost가 4가 된다.

또 두번째 Prime Implicant에서는 minterms 0, 1을 cover 하고 있는데 F_α, F_β에 공유하고 있으므로 cover 되는 minterms의 갯수는 4가 되므로 cost는 4가 된다.

그리고 minterms에 대하여 cost를 구해보면 Intersection Table에서 각각의 minterm이 갖고 있는 1의 갯수와 일치하게 된다. 위와 같은 방법으로 cost Table이 작성됩니다.

(3) Subcost Table

Subcost Table

Subcost is defined as the number of the remaining minterms which are not covered by the selected prime implicant or minterm

그러면 Subcost Table 作成에 대해서 설명합시다 제일 첫번째 prime implicant subcost의 값을 정(定)해 보겠읍니다. minterms 0, 2, 8 10은 cover가 되겠고, 그 minterm의 column에서 cover 안된 것이 3개 남아 있다. 이 값(갯수)이 Subcost의 값이다. 그러므로, Subcost의 값은 3이 되고 이와 같은 方法이 그 나머지에 적용됩니다.

例.

그러면 직접 문제를 풀어봅시다.

Selection of the Prime Implicants and Minterms

Selection of the Prime Implicants and Minterms

						0	1	2	4	8	10	11									
							*		*	*		*									
					F_α	1		1		1	1										
					F_β	1	1	1				1									
					F_γ		1		1	1											
	A	B	C	D									α	β	γ						
*	×	0	×	0		1		1		1	1		1			4	3	2	1	0	1
*	0	0	×	0		1		1					1	1		4	0	0	0	0	0
	0	0	0	×		1	1							1		2	2	1	1	1	1

2	2	2	1	2	1	1
0	0	0	0	0	0	0

0	2	0	1	2	1	1
0	0	0	0	0	0	0

0	2	0	1	1	0	1
0	0	0	0	0	0	0

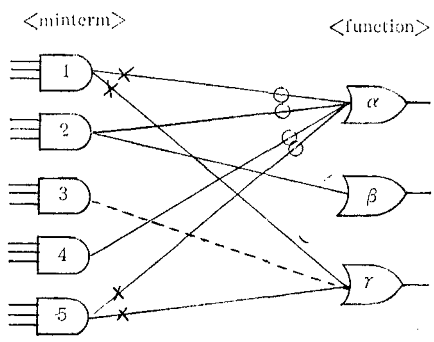
제일 먼저 MASK 方法에 의해 Prime Implicant가 위와 같이 주어져 있읍니다. 오른편(우측)에는 Commonality가 적혀 있읍니다. 앞 chart에서 설명했듯이 똑 같은 方法으로 Cost Table과 Subcost Table이 작성됩니다.

이와 같이 주어진 정보로부터 가장 optimal한 selection을 하기 위해서 Cost가 제일 높은것(큰값)을 선택(selection)하고 있읍니다. 그런데 위의 cost Table에서 제일 높은 cost의 값은 4이므로 4를 선택하는데 여기서는 똑같은 값이 2개 있읍니다. 이 똑같은 값 중에서 어느 것이 optimal 한 것인지 모르기 때문에 앞 chart에서 설명했듯이 subcost를 利用하여 optimal 한 것을 찾고 있읍니다. Subcost의 optimal를 利用하여 optimal한 것을 찾고 있읍니다. Subcost의 optimal은 값이 가장 작은 것이어야 합니다. 그 이유는 다음과 같읍니다.

즉 주어진 intersection Table의 1을 될 수 있는 한 많이 cover하기 위해서는 Subcost의 가장 작은 값을 selection 합니다. 실例로 다음과 같은 Intersection Table과 Cost, Subcost Table 값이 주어졌다고 가정하자.

	1	2	3	4	5
α					
β					
γ					

						4(3)
						4(0)



4(3) K1 (x)
4(0)* K1(0)
4(0)

위의 Fig에서 보면

① 5개의 AND gate와 3개의 OR gate 필요하며 각각을 Connection 합니다. 이 Connection을 가장 많이 줄이는 것이(즉 Cost를 가장 Maximum한 것을 찾는 것) Primary objective(主目的)이다.

② Connection이 똑같이 4개일 때는 어느 것을 먼저 선택하느냐 하는 문제는 (즉 cost의 값이 同一할 때)

i) α함수의 OR-gate를 선택하면 4번 AND gate가 줄어든다. (위 그림에서)

ii) X-표시를 한 부분을 선택한다면 ①·⑤번 gate가 줄어든다. 이러한 原則에 입각하여 Subcost Table이 작성되며 남아 있는 Connection의 수가 Subcost의 값이 된다. 그러므로 위의 에에서 0 표시한 부분의 Subcost는 3이고 X- 표시 부분의 Subcost는 0가 된다. 따라 Subcost의 값이 제일 적은 Prime Implicant나 minterm을 Selection 하는 것이 유리하다. 결과적으로 주어진 Intersection Table의 1을 될 수 있는 한 많이 Cover 하기 위해서는 Subcost의 가장 작은 값을 Selection 하게 됩니다.

위와 같은 원칙에 입각하면, 위의 도표에서는

1. 두번째 Prime Implicant가 선택됩니다. 두번째 Prime Implicant가 select 됨에 따라 새로운 Cost Table과 Subcost Table를 作成하게 된다.

2. 그 다음 Optimal한 Selection을 첫번째 Prime Implicant가 됩니다. 여기에 따른 Prime Implicant가 선택되었기에 또 다른 Cost Table이 작성됩니다.

3. 가장 높은 Cost의 값이 "2"가 되기 때문에 1번Minterm(m_1)이 선택되고, 이와 같은 과정을 계속하면 4, 8, 11번 Minterm(m_4, m_8, m_{11})이 Selection 됩니다.

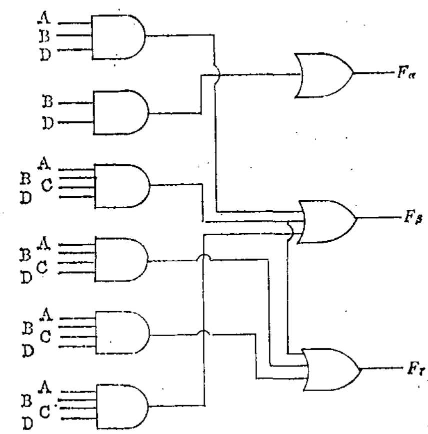
4. 그러면 지금까지 선택된 Prime Implicant를 다음 표로 만듭니다.

〈Realization of Multi-output Switching Function〉

A	B	C	D	α	β	γ	0	1	2	4	8	10	11
0	0	×	0	1	1	0	1		1				
×	0	×	0	1	0	0	1		1		1	1	
0	0	0	1	0	1	1		1					
0	1	0	0	0	0	1				1			
1	0	0	0	1	0	1					1		
1	0	1	1	0	1	0							1

위의 도표에서 살펴보면 첫번째 Prime Implicant와 두번째 Prime Implicant는 α에 대해 포함관계에 있기 때문에 첫번째 Prime Implicant의 α는 필요없게 됩니다. 또 두번째 Prime Implicant와 다섯번째 Prime Implicant는 포함관계에 있기 때문에 마찬가지로 α에 대해 필요없게 됩니다.

결과로서 이 선택된 Prime Implicant로서 回路를 구성해 보면 아래와 같은 Hardware Implementation이 됩니다.



*F_α의 OR-gate는 사실상 필요없는 것이 됩니다.

*Multi-output switching Function including Don't care 다음에 Don't care를 가지고 있는 multi-output switching function의 설계에 대해 알아보자.

Multi-output Switching Function including Don't care

	A	B	C	D		0	1	2	4	8	10	11	α	β	γ						
					F_α	d		1		d	1										
					F_β	1	1	1				d									
					F_γ		d		1	1											
*	×	0	×	0		1		1		1	1		1			2	3	1	1	1	1
*	0	0	×	0		1		1					1	1		3	0	0	0	0	0
	0	0	0	×		1	1							1		2	0	1	0	0	0
						1	1	2	1	1	1	0									
						0	0	0	0	0	0	0									
						0	1	0	1	1	1	0									
						0	0	0	0	0	0	0									
						0	0	0	1	1	1	1									
						0	0	0	0	0	0	0									

위와 같은 Intersection Table이 주어진다. 이 intersection table에서 generating 되는 Prime Implicant는 다음의 3개가 된다. 3개의 Prime Implicant의 Comonality의 우측과 같다. 여기서 Cost Table을 작성할 때 Don't care가 없는 multi-output switching function과 차이점은 Cost value 계산시 Don't care는 계산(counter) 하지 않는 것이다. 이러한 원칙에서 첫번째 Prime Implicant의 Cost를 계산해 보면 minterm 2, 10 두개가 cover 되므로 cost value는 2가 된다. 마찬가지 방법으로 첫번째 Prime Implecant에 의해 cover 되지 않는 minterm, 즉, Subcost는 3개가 되므로 3이 된다.

똑 같은 작업이 나머지 2개의 Prime Implecant와 7개의 minterm에 대해서도 진행됩니다. 그러면 위와 같은 Cost Table과 Subcost Table이 작성되고, 여기서 제일 먼저 선택되는 것은 두번째 Implement가 된다.

Ⅵ. 결 론

지금까지 MASK 方法과 Intersection Table, Subcost Table을 사용한 multi-output function design에 대해 설명하였읍니다. 本 論文에서 제시하고 있는 MASK 方法과 Table은 computer program으로 처리하기 쉽고 MASK Method에 의해 Prime Implicant를 generating 시킬 때 computer time이 다른 方法보다는 평균 5배가 빠를 것으로 예측한다. computer program은 보유하고 있으며, 다음과 같은 利點이 있다.

Advantage

1. Due to the Cost Criteria, minimal gates are required.
2. By the MASK method, mluti-output prime implicants are easily generated.
3. To reduce the number of connections, included commonality is eliminated from the list of selected terms.
4. Will suited for computer program.

References

1. F.J. Hill and G.R. Peterson, Introduction to switching theory and logical design, Wiley, New York, 1974.
2. V.T. Rhyne, Fundamentals of digital system, Prentice-Hall, Englewood Cliffs, NJ. 1973.
3. M.M. MaPo, Computer logic design, Prentice-Hall, Englewood Cliffs, NJ. 1972.
4. T.L. Booth, Digital networks and computer systems, John Wiley, 1972.
5. Hee Yeung Hwang. "A new approach to the minimization of switching functions by the Simple table method," 대한전기학회지, 제28권 제6호, pp.451—467, 1979.
6. Givone, Introduction to switching circuit theory, McGraw Hill New York, 1970.
7. G. Karnaugh, "The map method for synthesis of combinational logic circuits," AIEE Trans. Commun. Electron', pt.1, Vol.72, pp.593~599, Nov. 1853.
8. E.J. McCluskey, Jr., "Minimization of Boolean Functions." Bell Syst. Tech. J., Vol.35, pp.1417~1414, Nov. 1956.
9. 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.
10. N.N. Necula, "An algorithm for the automatic approximate minimization of Boolean functions," IEEE Trans. Comput., Vol. C-17, pp.770~782, Aug. 1968.
11. H.A. Curtis, "Simplified decomposition of Boolean functions," IEEE Trans. Comput., Vol.C-25, pp.1033~1076, Oct. 1976.
12. Sureshander, "Minimization of switching functions-A fast technique," IEEE Trans. Comput.(Corresp.), Vol.C-24., pp.753~756, July 1975. 13. 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.
14. F.M. Brown. "Equational realizations of switching functions," IEEE Trans. Comput,

Vol. C-24, pp. 1054~1066, Nov. 1975.

15. 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.
16. S.R. Das, "Comments on' A new algorithm for generating prime implicants," IEEE Trans. Comput., Vol. C-20, pp. 1614~1615, Dec. 1971.
17. J.G. Bredeson and D.T. Hulena, "Generation," of prime implicant by direct multiplication IEEE Trars, Comput., Vol. C-20, pp. 475 ~476, Apr. 1971.
18. H.R. Hwa, "A method for generating prime implicants of a Boolean expression," IEEE Trans. Comput., Vol. C-23, pp. 637~644, June 1974.
19. B.L. Hulme and R.B. Worrell, "A prime implicant algorithm with factoring," IEEE Trans. Comput., Vol. C-24, pp. 1129~1131, Nov. 1975.
20. 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.
21. F.J. Hill and G.R. Peterson, Introduction to switching theory and Logical design, Wiley, New York, 1974, pp. 97~174.
23. Taylor L. Booth, Digital Network and Computer Systems, Wiley, New York, 1971, pp. 93~157.
23. John B. Peatman, The Design of Digital System, McGraw Hill, New York, 1972, pp. 56~116.
24. V.T. Rhyne, Fundamentals of Digital Systems Design, Englewood Cliffs, NJ: Prentice-Hall, 1973, pp. 163~165.
25. W.V. Quine, "A way to simplify truth functions," Amer. Math. Mon. vol. 62, pp. 627~631, Nov. 1955.
26. P. Tison, "Generalization of consensus theorn and application to tht minimization of boolean functions," IEEE Trars. Electronic Computers, vol. EC-16, pp. 446~456, August 1967.
27. A. Avoboda, "Ordering of implicants," IEEE Trans. Electronic Computers (Short notes), vol. EC-16, pp. 100~150, February 1967.
28. 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.
29. 黃熙隆, A new approach to the minimization of switching functions by the simple Table method," 대한전기학회지, 제2권 제6호, pp. 61 ~77, 1979. 6월.
30. S.C. Lee, Modern Switching Theory and Digital Design, Prentice-Hall, Englewood Cliffs, N.J. 1978.

Appendix The Computer Program for the Minimization of the Switching Funotion

```
      BOS/VS   FORTRAN 5   -770127- V01-L14                                   79.08.11
OPTIONS IN EFFECT (FORTS1)
  OBJECT,NOSTACK,SOURCE,NOMAP,NOISN,EBCDIC,NOAUTODBL,NOSEQUENCE,NOASTERISK,NODENSEPRINT
  OPT(0),FMTAREA(   256),FLAG(W),DEBUG(STD)

      INTEGER INT(64,6),ID(200),D(100),R(100),CL(100),CY(50,50),DA(64,6)
     *,L(100),NF(10),XL(50)/50*0/,X1/'  0'/,X2/'  1 '/,X3/'  X '/,X(10)/
     *'  A ','  B ','  C ','  D ','  E ','  F ','  G ','  H ','  I ','
     *J '/,XS(2)/'SELE','CTED'/,DIF,COST(50),P(50,8),DJ,FM,CHECK(10),COM
     *M(50),SUBPI(10,10),DLI,COST1(50),N
     *,FNA(5)/' FA ',' FB ',' FC ',' FD ',' FE '/      ,LSUB(8)
      COMMON J1,N,ND,FM,ICOUNT,NE,NF,L,ID,D,MO,M,CY,R,CL,CHECK,ICHECK,DA
     *,DJ,P,COST,COST1,INT,SUBPI,COMM,NOF,NFI,ND3,ICOST,NEC
C
C
C
C
C
C
C
C
C
C        ****************************************************************
C        *                                                              *
C        *      MULTIPLE-FUNCTION MINIMIZATION PROGRAMMING              *
C        *                                                              *
C        ****************************************************************
C
C        (A)PURPOSE : MINIMIZATION OF MULTI-OUTPUT BOOLEAN FUNCTION
C
C        (B)THEORY    *  MINIMIZATION THEORY  OF MONO-FUNCTION IS PRESENTED
C                        PREVIOUSLY.
C
C                     *  IN THE CASE OF MULTIPLE-FUNCTION  COST CRITERION IS VARIED
C                        AS THE COMMONALITY OR NO. OF PRIME IMPLICANT IS CHANGED.
C
C
C        (C)PROCEDURE  1)READ  INPUT DATA AND REARRANGE THEM.
C                      2)SET DA TABLE
C                      3)FIND PRIME IMPLICANT IN DA TABLE
C                      4)THE SUBSET OF PRIME IMPLICANT WHICH HAS HIGHER COMMONOLITY
C                        THAN THE FIRST PRIME IMPLICANT  SHOULD BE SELECTED.
C                      5)IF THERE ARE MORE PRIME IMPLICANTS BRANCHED FROM THE FIRST
C                          PRIME IMPLICANT, REPEAT PROCEDURE 4)
C                      6)REPEAT PROCEDURE 3),4),5) UNTIL ALL OF THE PRIME IMPLICANT
C                        ARE FOUND.
C                      7)NOW  SELECT APPROPRIATE PRIME IMPLICANT  UNTIL  ALL THE
C                        MINTERMS  ARE COVERED.
C            PROCEDURE  2)----7) SHOULD BE ITTERATED  NOF TIMES(NOF:NO.OF FUNCTION)
C        (D)A LITTLE COMPLICATE SUBROUTIMES,ARRAIES PARAMETERS ARE INCLUDED IN THIS
C             PROGRAM AS FOLLOWS.
C     SUBROUTINE      PI : FIND PRIME IMPLICANT IN DA TABLE
C                     ACHECK : CONSTRUCT P.I. CHART AND CHCK THE 1'S IN DA TABLE
C                     MCOST : COMMONALITY AND COST OE CHOSEN P.I. AND SUBP.I.
C                     SUBPR : SELECT SUBP.I. OF HIGHER COMMONALITY.
C          NOF --- NO. OF FUNCTION
C          N   --- NO. OF VARIABLES
C          ARRAY    INT --- INPUT AND INTERSECTIN TABLE
C                   D   --- TRUE OR DON'T CARE MINTERMS
C                   DA  --- DA TABLE
C                   COST--- COST OF EACH PRIME IMPLICANT
C                   COMM--- COMMUNALITY OF EACH P.I.
C                   CL  --- USED FOR CHECKING THE MINTERMS
C                   CY  --- PRIME IMPLICANT CHART
```

```
FACOM  BOS/VS   FORTRAN S   -770127- V01-L14    FTMAIN                                    79.08.1
C
C
C
C
C
C
C
C
      NOF=3
      N=4
      PRINT 1001,NOF,(FNA(I),I=1,NOF)
      NM=2**N
      DO 20 J=1,NOF
   20 READ(5,10)  (INT(I,J),I=1,NM)
   10 FORMAT(40I2)
      DO702 I=1,NM
      DO 701 J=1,NOF
      IF(INT(I,J).NE.0) GO TO 703
  701 CONTINUE
      GO TO 702
  703 K=I-1
      PRINT 704 ,K,(INT(I,JJ),JJ=1,NOF)
  702 CONTINUE
      PRINT 1011,N
      PRINT 1103
      DO 30 IF=1,NOF
      M=0
      DO 40 I=1,NM
      IF(INT(I,IF).EQ.0) GO TO 40
      M=M+1
      D(M)=I-1
      IF(I.EQ.1) GO TO 40
      ID(I-1)=M
   40 CONTINUE
C
C
C    SET DA TABLE
C
      DO 50 I=1,M
      DO 43 JA=1,N
   43 DA(I,JA)=0
      I1=I+1
      DO 50 J=I1,M
      DIF=D(J)-D(I)
      DO 60 NA=1,N
      IF(DIF-2**(N-NA))  60,65,50
   60 CONTINUE
   65 IF((D(I)/DIF)/2*2.NE.D(I)/DIF) GO TO 50
      DO 70 II=1,N
      KII=2**(II-1)
      IF(DIF.EQ.KII) GO TO 75
   70 CONTINUE
   75 DA(I,II)=1
   50 CONTINUE
C
C
C
C    FIND THE FIRST PRIME IMPLICANT IN DA TABLE.
C
      MQ=0
      DO 100 I=1,M
      DO 100 J=1,N
      DO 2000 II=1,N
 2000 CHECK(II)=0
      ICOUNT=0
      IF(DA(I,J).EQ.0)  GO TO 100
      IF(DA(I,J).EQ.1)  ICOUNT=1
      NE=1
      NF(NE)=J
      ND=2
      L(1)=I
      L(2)=ID(D(I)+2**(J-1))
   41 J1=J+1
      ICHECK=0
      CALL PI
      IF(FM.EQ.0)  GO TO 85
      DO 84 K=1,ND
      LK=L(K)
   84 LSUB(K)=LK
      ND3=ND
      NEC=0
      CALL ACHECK
      CALL MCOST
```

```
      FACOM  BOS/VS   FORTRAN:S   -770127- V01-L14      FTMAIN
          CALL   SUBPR
C
C
C
C   ARE THERE MORE PRINE IMPLICANTS BRANCHED FROM THE FIRST P.I.?
   85 IF(NE.EQ.1) GO TO 100
      DO 86 K=1,ND
      LK=LSUB(K)
   86 L(K)=LK
      NE1=NE-1
      DO 201 NA=1,NE1
      IF(NF(NE).EQ.N) GO TO 201
      J1=NF(NE1-NA+1)+1
  205 NE=NE1-NA+1
      ND=2**(NE)
      NDD=ND
      ICHECK=1
      ICOUNT=0
      CALL  PI
      IF(ND.LE.NDD)  GO TO 201
      IF(FM.EQ.0) GO TO 200
      CALL ACHECK
      ND3=ND
      CALL MCOST
      CALL  SUBPR
  200 IF(NF(NE1-NA+2).EQ.N)  GO TO 201
      J1=NF(NE1-NA+2)+1
      GO TO 205
  201 CONTINUE
  100 CONTINUE
      DO 120 I=1,M
  103 IF(CL(I).NE.0) GO TO 120
      MQ=MQ+1
      CY(MQ,I)=1
      DJ=D(I)
      NEC=-1
      NEC=0
      CALL ACHECK
      ICHECK=0
      CALL MCOST
      PRINT 1120,MQ,(D(L(II)),II=1,ND)
  120 CONTINUE
      DO  155 K=1,M
      ICOST=COST(K)
  155 COST1(K)=ICOST
      DO 156 K=1,MQ
  156 XL(K)=0
C
C
C
C
C
C   PRIME IMPLICANT CHART IS CONSTRUCTED AND NOW SELECT THE NECESSARY P.I
C
  157 MAX=0
      ICOST=0
      ICOMM=0
      DO 160 K=1,MQ
      IF(XL(K).EQ.1) GO TO 160
      IF(MAX-COST1(K))  167,165,160
  165 IF(ICOST-COST(K)) 167,166,160
  166 IF(ICOMM-COMM(K)) 160,160,167
  167 MAX=COST1(K)
      II=K
      ICOMM=COMM(K)
      ICOST=COST(K)
  160 CONTINUE
      XL(II)=1
      DO 170 K=1,M
      IF(CY(II,K).EQ.0) GO TO 170
      CL(K)=-3
      DO 175 KK=1,MQ
  175 CY(KK,K)=0
  170 CONTINUE
      DO 180 II=1,MQ
      KCOUNT=0
      DO 185 JJ=1,M
  185 KCOUNT=KCOUNT+CY(II,JJ)
      ICOMM=COMM(II)
      COST1(II)=KCOUNT*ICOMM
  180 CONTINUE
      DO 190 II=1,M
      IF(CL(II).NE.-3) GO TO 157
  190 CONTINUE
```

```
FACOM BOS/VS   FORTRAN S   -770127- V01-L14      FTMAIN                          79.08
      C   PRINT OUTPUT
      C
      C
      C
      C
      C
      C
            PRINT 1005,IF,FNA(IF)
            PRINT 1003,(X(I),I=1,N)
            DO 410 I=1,MQ
            IF(XL(I)) 420,420,430
        430 PRINT 1004,I,COST(I),(P(I,MM),MM=1,N),XS
            GO TO 410
        420 PRINT1004,I,COST(I),(P(I,MM),MM=1,N)
        410 CONTINUE
        704 FORMAT(/45X,I4,5(2X,I4))
       1001 FORMAT(///////36X,'########### MULTIPLE FUNCTION MINIMIZATION PRO
           *GRAM   ###########'/////37X,'THIS IS MULTIPLE FUNCTION MINIMIZATION
           * PROGRAM WHICH MINIMIZES'//37X,'THE COST CRITERIA (NO.OF GATES) OF
           * THE TOTAL FUNCTION USING DA'//37X,'TABLE AND COST TABLE METHOD.'/
           */////37X,'THIS MULTIPLE FUNCTION CONTAINS ',I4,' FUNCTIONS'//40X,'
           *NAMELY'/////43X,'MINTERM',10(2X,A4))
       1011 FORMAT(///37X,'AND EACH FUNCTION CONTAINS',I4,' VARIABLES')
       1111 FORMAT(/////37X,'THE LIST OF MINTERMS OF EACH FUNCTIONS ARE AS FOL
           *LOWS WHERE'/40X,' TRUE MINTERM, FALSE MINTERM AND REDUNDANCY MIN'
           *'TERMS ARE'/40X,'REPRESENTED BY 1, 0 AND -1 RESPECTIVELY.'////44X
           *,'MIN.NO ',5A4)
       1002 FORMAT(10X,' ( ',I4,' )=',I2)
       1003 FORMAT(//44X,'    NO.    COST',10A4)
       1004 FORMAT(/44X,4X,I2,I5,2X,10A4)
       1005 FORMAT(/////37X,' **** FUNCTION ** ',I3,' **** ',A4)
       1103 FORMAT(//////37X,'THE FOLLOWING IS THE LIST OF THE PRIME IMPLICANT
           *S OBTAINED FROM '//39X,'DA TABLE WHERE PRIME IMPLICANTS SELECTED F
           *ROM COST TABLE ARE   '//37X,'   LABLED AS SELECTED       ')
       1150 FORMAT(/////10X,'    P.I.                    MINTERM'//10X,'   NO.  COST C
           *OMM.',30I3)
       1160 FORMAT(/16X,' CHECK(-3)   ',30I3)
       1120 FORMAT(/10X,3I6,30I3)
         30 CONTINUE
            STOP
            END
 DATA SIZE =     305, PROCEDURE SIZE =      2784
 NO DIAGNOSTICS GENERATED (FTMAIN)
 END OF COMPILATION (FTMAIN)

FACOM BOS/VS   FORTRAN S   -770127- V01-L14                                        79.08.11
OPTIONS IN EFFECT (FORTS1)
   OBJECT,NOSTACK,SOURCE,NOMAP,NOISN,EBCDIC,NOAUTODBL,NOSEQUENCE,NOASTERISK,NODENSEPRINT,
   OPT(0),FMTAREA(  256),FLAG(W),DEBUG(STO),
            SUBROUTINE PI
            INTEGER INT(64,6),ID(200),D(100),R(100),CL(100),CY(50,50),DA(64,6)
           *,L(100),NF(10),XL(50)/50*0/,X1/'  0'/,X2/'  1 '/,X3/'  X '/,X(10)/
           *'  A ','  B ','  C ','  D ','  E ','  F ','  G ','  H ','  I ',
           *J '/,XS(2)/'SELE','CTED'/,DIF,COST(50),P(50,8),DJ,FM,CHECK(10),COM
           *M(50),SUBPI(10,10),DLI,COST1(50),N
            COMMON J1,N,ND,FM,ICOUNT,NE,NF,L,ID,D,MQ,M,CY,R,CL,CHECK,ICHECK,DA
           *,DJ,P,COST,COST1,INT,SUBPI,COMM,NOF,NFI,ND3,ICOST,NEC
      C     THIS SUBROUTINE IS USED FOR FINDING PRIME IMPLICANT IN DA TABLE
      C
      C
      C
      C
      C
            IF(J1.GT.N) GO TO 45
            DO 95 JA=J1,N
            IF(ICHECK.EQ.0) GO TO 86
            IF(ND.GT.NDO)  GO TO 86
            IF(CHECK(JA).EQ.1)  GO TO 95
         86 FM=ICOUNT
            JCOUNT=0
            DO 90 NN=1,ND
            IF(DA(L(NN),JA)) 90,87,88
         87 JCOUNT=1
            GO TO 90
         88 ICOUNT=ICOUNT+1
         90 CONTINUE
            IF(JCOUNT.EQ.0)  GO TO 91
            ICOUNT=FM
            GO TO 95
         91 NE=NE+1
            NF(NE)=JA
            FM=ICOUNT
            DO 93 K=1,ND
         93 L(ND+K)=ID(D(L(K))+2**(JA-1))
            ND=ND*2
         95 CONTINUE
            IF(FM.NE.0) GO TO 44
            DO 100 JJ=1,NE
            IF(DA(L(II),NF(JJ)).NE.1) GO TO 100
            FM=1
        100 CONTINUE
         44 RETURN
         45 FM=ICOUNT
            RETURN
            END
DATA SIZE =    189, PROCEDURE SIZE =      493
NO DIAGNOSTICS GENERATED (PI     )
END OF COMPILATION (PI     )
```

```
          FACOM   BOS/VS    FORTRAN S    -770127- V01-L14                                    79.08.2.
   OPTIONS IN EFFECT (FORTS1)
      OBJECT,NOSTACK,SOURCE,NOMAP,NOISN,EBCDIC,NOAUTODBL,NOSEQUENCE,NOASTERISK,NODENSEPRINT,
      OPT(0),FMTAREA(   256),FLAG(W),DEBUG(STD)

 0001           SUBROUTINE MCOST
 0002           INTEGER INT(64,6),ID(200),D(100),R(100),CL(100),CY(50,50),DA(64,6)
               *,L(100),NF(10),XL(50)/50*0/,X1/' 0'/,X2/' 1 '/,X3/' X '/,X(10)/
               *' A ',' B ',' C ',' D ',' E ',' F ',' G ',' H ',' I ','
               *J '/,XS(2)/'SELE','CTED'/,DIF,COST(50),P(50,8),DJ,FM,CHECK(10),COM
               *M(50),SUBPI(10,10),DLI,COST1(50),N
 0003           COMMON J1,N,ND,FM,ICOUNT,NE,NF,L,ID,D,MQ,M,CY,R,CL,CHECK,ICHECK,DA
               *,DJ,P,COST,COST1,INT,SUBPI,COMM,NOF,NFI,ND3,ICOST,NEC
          C
          C
          C
          C      THIS SUBROUTINE IS FOR THE COST TABLE.
          C
          C      COST IS COMPUTED BY MULTIPLYING COMMONALITY BY THE NO. OF MINTERMS.
          C
 0004           ICOST=0
 0005           IF(ND3.EQ.1) GO TO 136
 0006           DO 130 J3=1,NOF
 0007           DO 135 I3=1,ND3
 0008           DLI=D(L(I3))+1
 0009           IF(INT(DLI,J3).EQ.0) GO TO 130
 0010       135 CONTINUE
 0011           ICOST=ICOST+1
 0012       130 CONTINUE
 0013           IF(ICHECK) 150,140,137
 0014       136 DLI=D(L(1))+1
 0015           DO 230 J3=1,NOF
 0016           IF(INT(DLI,J3).EQ.0) GO TO 230
 0017           ICOST=ICOST+1
 0018       230 CONTINUE
 0019           IF(ICHECK) 150,140,137
 0020       137 IF(ICOST.LE.NFI) GO TO 150
 0021           CALL ACHECK
 0022       140 COMM(MQ)=ICOST
 0023           COST(MQ)=ND3*ICOST
 0024      3001 FORMAT(10X,20I4)
 0025       150 RETURN
 0026           END

 91  DATA SIZE =     170, PROCEDURE SIZE =        365
 94  NO DIAGNOSTICS GENERATED (MCOST )
 96  END OF COMPILATION (MCOST )
          FACOM   BOS/VS    FORTRAN S    -770127- V01-L14                                    79.
   OPTIONS IN EFFECT (FORTS1)
      OBJECT,NOSTACK,SOURCE,NOMAP,NOISN,EBCDIC,NOAUTODBL,NOSEQUENCE,NOASTERISK,NODENSEPRINT
      OPT(0),FMTAREA(   256),FLAG(W),DEBUG(STD)

 0001           SUBROUTINE ACHECK
 0002           INTEGER INT(64,6),ID(200),D(100),R(100),CL(100),CY(50,50),DA(64,6)
               *,L(100),NF(10),XL(50)/50*0/,X1/' 0'/,X2/' 1 '/,X3/' X '/,X(10)/
               *' A ',' B ',' C ',' D ',' E ',' F ',' G ',' H ',' I ','
               *J '/,XS(2)/'SELE','CTED'/,DIF,COST(50),P(50,8),DJ,FM,CHECK(10),COM
               *M(50),SUBPI(10,10),DLI,COST1(50),N
               *,SUBPII(25,8)
 0003           COMMON J1,N,ND,FN,ICOUNT,NE,NF,L,ID,D,MQ,M,CY,R,CL,CHECK,ICHECK,DA
               *,DJ,P,COST,COST1,INT,SUBPI,COMM,NOF,NFI,ND3,ICOST,NEC
          C
          C
          C
          C
          C
          C
          C      SET PRIME IMPLICANT CHART (ARRAY CY)
          C
          C      AND PRIME IMPLICANTS ARE CONVERTED TO BINARY FORM(LIKE '0 1 X 1 X 0').
          C
          C
          C
          C
          C
          C         CHECK THE MINTERMS IN DA TABLE    (1) ----> (-1)
          C
 0004           MQ=MQ+1
 0005           DO 63 K=1,M
 0006        63 CY(MQ,K)=0
 0007           DO 65 K=1,ND3
 0008           CL(L(K))=1
 0009           CY(MQ,L(K))=1
 0010        65 CONTINUE
 0011           DO 70 JJ=1,NE
 0012           NFNE=NF(JJ)
 0013           CHECK(NFNE)=1
 0014           DO 70 II=1,ND3
 0015           IF(DA(L(II),NFNE).EQ.1) DA(L(II),NFNE)=-1
 0016        70 CONTINUE
 0017           DJ=D(L(1))
 0018           DO 26 K=1,N
 0019           KT=2**(N-K)
 0020           IF(DJ.LT.KT) GO TO 25
 0021           P(MQ,K)=X2
 0022       210 DJ=DJ-KT
 0023           GO TO 26
 0024        25 P(MQ,K)=X1
 0025        26 CONTINUE
 0026           IF(NEC.EQ.-1) GO TO 200
 0027           DO 24 K=1,NE
 0028        24 P(MQ,N-NF(K)+1)=X3
 0029      1120 FORMAT(/10X,120('.')//10X,'PRIME IMPLICANT ***',I4//10X,'CHECKED M
               1INTERMS ARE ',20I4)
 0030      1130 FORMAT(/10X,120('.')//10X,'MINTERM    NO.   DA TABLE IN ASCENDING
               1ORDER'/)
 0031      1140 FORMAT(10X,2I7,10I4)
 0032       200 RETURN
 0033           END
 991  DATA SIZE =     579, PROCEDURE SIZE =        545
 994  NO DIAGNOSTICS GENERATED (ACHECK)
 996  END OF COMPILATION (ACHECK)
```

```
FACOM  BOS/VS   FORTRAN S   -770127- V01-L14                                        79.08.
OPTIONS IN EFFECT (FORTS1)
   OBJECT,NOSTACK,SOURCE,NOMAP,NOISN,EBCDIC,NOAUTODBL,NOSEQUENCE,NOASTERISK,NODENSEPRINT,
   OPT(0),FMTAREA(  256),FLAG(W),DEBUG(STD)

      SUBROUTINE SUBPR
      INTEGER INT(64,6),ID(200),D(100),R(100),CL(100),CY(50,50),DA(64,6)
     *,L(100),NF(10),XL(50),DIF,COST(50),P(50,8),DJ,FM,CHECK(10),COMM(50
     *),SUBPI(10,10),DLI,COST1(50),SUBPII(25,8),NDI(8),SUM,COMM2
     *,IN1(4),IN2(4),SUBPI2(10,10),X3/ '  X '/
      COMMON J1,N,ND,FM,ICOUNT,NE,NF,L,ID,D,MQ,M,CY,R,CL,CHECK,ICHECK,DA
     *,DJ,P,COST,COST1,INT,SUBPI,COMM,NOF,NFI,ND3,ICOST,NEC
      IN1(1)=1
      IN2(1)=2
      IN1(2)=1
      IN2(2)=3
      IN1(3)=2
      IN2(3)=4
      IN1(4)=3
      IN2(4)=4
C
C
C
C
C
C
C
C
C
C
C
C
C   IN THIS SUBROUTINE SUB P.I.'S ARE FOUND AND IF THE SUBPI IS IN HIGHER
C
C   COMMONALITY,IT SHOULD BE INSERTED INTO PRIME IMPLICANT CHART.
C
      NE1=NE+1
      DO 99 I=1,NE1
      IND=ND/2**(I-1)
   99 NDI(I)=IND
      DO 110 I=1,ND
      LI=L(I)
  110 SUBPI(1,I)=LI
      IA1=1
      II1=0
      DO 115 I=1,NE
      NDI1=NDI(I)
      NDI2=NDI(I+1)
      SUM=4**(I-1)
      DO 120 II=1,SUM
      II1=II1+1
      DO 130 IB=1,NDI1
      LIB=SUBPI(II1,IB)
      L(IB)=LIB
  130 CONTINUE
      DO 131 K=1,NDI1
  131 R(K)=0
      ICHECK=-1
      ND3=NDI1
      CALL MCOST
      COMM2=ICOST
C
C
      IF(NDI1.NE.2) GO TO 90
      SUBPII(1,1)=L(1)
      SUBPII(2,1)=L(2)
      GO TO 100
   90 ND4=NDI1/4
      ICOUNT=0
      DO 95 I2=1,4
      DO 95 J2=1,ND4
      ICOUNT=ICOUNT+1
      SUBPI2(I2,J2)=L(ICOUNT)
   95 CONTINUE
      DO 101 LL=1,4
      II=IN1(LL)
      KI=1
      DO 125 K=1,NDI2
      L(K)=SUBPI2(II,KI)
      IF(KI.NE.ND4) GO TO 124
      II=IN2(LL)
      KI=1
```

```
          FACOM  BOS/VS   FORTRAN S   -770127- V01-L14     SUBF
         GO TO 125
     124 KI=KI+1
     125 CONTINUE
         DO 200K=1,NDI2
     200 SUBPII(LL,K)=L(K)
     101 CONTINUE
    1120 FORMAT(10X,30I4)
C
C
C
C
     100 M4=4
         IF(NDI2.EQ.1) M4=2
         DO 160 IK=1,M4
         IA1=IA1+1
         DO 150 IB=1,NDI2
         ISUBPI=SUBPII(IK,IB)
         SUBPI(IA1,IB)=ISUBPI
         L(IB)=ISUBPI
     150 CONTINUE
         ICHECK=-1
         ND3=NDI2
         CALL MCOST
    1100 FORMAT(/10X,30I4)
         IF(ICOST.LE.COMM2) GO TO 160
         IF(IK.EQ.4) GO TO 154
         DO 151 K=1,NDI2
         IF(R(K).EQ.0) GO TO 152
     151 CONTINUE
         GO TO 160
     152 DO 153 K=1,NDI2
     153 R(K)=1
     154 NEC=+1
         CALL ACHECK
         IF(NDI2.EQ.1) GO TO 159
         NEC=NDI2/2
         DO 158 K=1,NEC
         KT=2**(K-1)
         KT2=(ALOG(FLOAT(L(KT)-L(1)))/ALOG(2.0)+0.3)
     158 P(MQ,N-KT2)=X3
     159 ICHECK=0
         CALL MCOST
     160 CONTINUE
     120 CONTINUE
     115 CONTINUE
C
C
C
         RETURN
         END
FT991  DATA SIZE =    840, PROCEDURE SIZE =      1124
FT994  NO DIAGNOSTICS GENERATED (SUBPR )
FT996  END OF COMPILATION (SUBPR )
```

********** MULTIPLE FUNCTION MINIMIZATION PROGRAM ***********

THIS IS MULTIPLE FUNCTION MINIMIZATION PROGRAM WHICH MINIMIZES THE COST CRITERIA (NO.OF GATES) OF THE TOTAL FUNCTION USING DA TABLE AND 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 FUNCTION 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	X	1	1	SELECTED →	$\bar{X}_1 X_3 X_4$
2	2	0	1	1	1		
3	8	X	1	X	1	SELECTED →	$X_2 X_4$
4	3	1	1	1	1		
5	3	1	1	1	1		
6	6	1	1	1	X	SELECTED →	$X_1 X_2 X_3$

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

NO.	COST	A	B	C	D		
1	8	X	1	X	1	SELECTED →	$X_2 X_4$
2	3	1	1	1	1		
3	3	1	1	1	1		
4	4	1	X	1	0	SELECTED →	$X_1 X_3 \bar{X}_4$
5	3	1	1	1	0		
6	6	1	1	1	X		

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

NO.	COST	A	B	C	D		
1	2	0	X	0	0	SELECTED →	$\bar{X}_1 \bar{X}_3 \bar{X}_4$
2	4	1	X	1	0	SELECTED →	$X_1 X_3 \bar{X}_4$
3	3	1	1	1	0		
4	6	1	1	1	X	SELECTED →	$X_1 X_2 X_3$

```
FACOM   BOS/VS        -770201- V01-L15       SYSTEM MESSAGE LIST                   79.0

          SEQ.          CONTROL DATA / MESSAGE
           1            *JOB MWAN.H.R.ACCT='T79-00-001'.LIST=1.TIME=600.PRINT=50
           2              JB105 BOJ HWAN.H.R
           3            *FORTS1
           4              JB110 BOS FORTS1
           5              FT994  NO DIAGNOSTICS GENERATED (FTMAIN)
           6              FT994  NO DIAGNOSTICS GENERATED (PI     )
           7              FT994  NO DIAGNOSTICS GENERATED (MCOST  )
           8              FT994  NO DIAGNOSTICS GENERATED (ACHECK)
           9              FT994  NO DIAGNOSTICS GENERATED (SUBPR  )
          10            /END
          11              FT998  NO SERIOUS DIAGNOSTICS THIS STEP
          12              JB111 EOS FORTS1
          13            *EXEC FTMAIN.RLIB
          14              JB110 BOS FTMAIN
          15              JB111 EOS FTMAIN
          16              JB106 EOJ HWAN.H.R
```