Application of Dual Karnaugh Mapping for processing of logical switching functions and designing combinational switching circuits

Junso TOYODA

Department of Electronic Science, Okayama University of Science
Ridaicho 1-1, Okayama 700, JAPAN

(Received September 24, 1982)

Abstract

A graphic methods of logical processing in which Karnaugh Map and Maxterm Map, proposed in this paper and can be said as modified Inversed Karnaugh Map, are cooperatively utilized is discussed. It is shown that the methods is very efficient to get complete sum or complete product and also to convert from sum-of-product to product-of-sum, and *vice versa*. Some of applications are shown for processing logical switching functions and combinational switching circuits.

Introduction

This paper is concerned with expanding availabilities of graphic methods. Maxterm Map, which is cooperatively utilized with Karnaugh Map in Dual Mapping, is addressed by wholly supplemental numerics to that of Karnaugh Map, and each cell of the Map has logical AND connectve property with any of the others. We use symboles : (used as AND) to represent sum-of-product and symboles + (used as OR) to represent product-of-sum on a Map, former being addressd by Appendix 1 and latter by Appendix 2. Two forms of a switching function sum-of-product and product-of-sum can be displayed, if desired, on a map without any of doubled plotting nor spaced plotting of :'s and +'s based on these arrangement of addressing. The Map on which :'s are to be plotted normally by address of Appendix 1 (simplified as "m") is called as Minterm Map (simplified as Map(m),). The Map on which +'s are to be plotted normally by address of Appendix 2 (simplified as "M") is called as Maxterm Map (simplified as Map(M)). The graphic methods, based on these arrangement of mapping, can be more useful.

1 Minterm mapping and Maxterm mapping

Of the mapping we arrange two of styles, variable-wise-mapping and term-wise-mapping, former for getting simple sum or simple products latter for getting switching functions of 2'nd order.

We give value 0 to complement of variables of simple product to be displayed prior to variable-wise-mapping on Map(m), and then we take logical AND operation with the value of each cell of the Map. Plotting is performed as to get the records of the result of succesive AND operations using :'s for only the cells valued as 1. Appendix 3 shows how minterm m_{15} (ABCD) is gotten on Map(m) which has initially all 1's. Similarly we give value 1 to complement of variables of simple sum prior to variable-wise-mapping on Map(M), and then we take logical OR operation with the value of each cell of the Map. Plotting is performed as to get the records of the result of successive OR operations using +'s for only the cells valued as 0. Appendix 4 shows how maxterm M_{15} $(A+B+C+D)$ is gotten on Map(M) which has initially all 0's.

Term-wise-mapping on to Map(m) is performed giving value 1 only to a term (simple product) of switching function to be displayed, and taking logical OR operation between the value of each cell of the Map. Plotting is performed as to get the records of the result of successive term-wise OR operations using :'s for only the cells valued as 1. Similarly, term-wise-mapping on to Map(M) is performed giving value 0 only to a term (simple sum) of switching function to be displayed, and taking logical AND operation between the value of each cell of the Map. Plotting is performed as to get the records of the result of successive term-wise AND operations using +'s for only the cells valued as 0. Logical properties of Dual Mapping are summarrized on Appendix 9.

The mapping can be said, being done as to get the records of result of logical operations, a synthesizing of logical information of original map and that of function to be displayed.

Appendix 5 shows mapping of function $J=\alpha+\beta+\gamma=A\overline{B}\overline{D}+ABD+\overline{A}B\overline{C}$ on to Map(m), which already has information of $I=\delta+\varepsilon+\lambda=ACD+B\overline{C}D+\overline{A}C\overline{D}$. Appendix 6 shows mapping of function $T=\alpha\beta\gamma=(A+B+C)(A+\overline{C}+\overline{D})(\overline{A}+\overline{B}+D)$ on to Map(M), which already has information of $S=\delta\varepsilon=(A+B+\overline{D})(B+C+\overline{D})$. It is convenient to use temporal names for terms included in function to be mapped, such as α, β, γ, ,,, , and then rearrange the names for resultant function, using such as a, b, c, ,,, , according to logical connective property of the Map.

Mapping process can be illustrated as following,

Suffix of "m"	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Old Map(m)	0	0	1	0	0	1	1	0	0	0	0	1	0	1	0	1
J_0(m)	0	0	0	0	1	1	0	0	1	0	1	0	0	1	0	1
Mapping logic	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
New Map(m)	0	0	1	0	1	1	1	0	1	0	1	1	0	1	0	1
Plotting			:		:	:	:		:		:	:		:		:
Suffix of "M"	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Old Map(M)	1	0	1	0	1	1	1	1	1	0	1	1	1	1	1	1
T_0(m)	0	0	1	0	1	1	1	0	1	0	1	1	0	1	0	1
Mapping logic	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:
New Map(M)	0	0	1	0	1	1	1	0	1	0	1	1	0	1	0	1
Plotting	+	+		+				+		+			+		+	

where $J_0(m)$ and $T_0(M)$ are used as minterm expression and maxterm expression for function J and T respectively.

We can have those of resultant function X and Y as follows,

$$X_0(m) = m_2 + m_4 + m_5 + m_6 + m_8 + m_{10} + m_{11} + m_{13} + m_{15} \tag{1}$$

$$Y_0(M) = M_{15} M_{14} M_{12} M_8 M_6 M_3 M_1 \tag{2}$$

Rearranged form which can be seen on Appendix 5 or Appendix 6 are as follows,

$$X(m) = A\overline{B}\overline{D} + ABD + \overline{A}B\overline{C} + ACD + \overline{A}C\overline{D} \quad (= a + b + c + g + h) \tag{3}$$

$$Y(M) = (A+B+C)(A+\overline{C}+\overline{D})(\overline{A}+\overline{B}+D)(B+C+\overline{D}) \quad (= xyzu) \tag{4}$$

These are one of simplifyed sum-of-product and product-of-sum respectively. Of (4), we can see, that it is the only minimum product-of-sum.

These treatment are quite similar to logical operation of switching circuit of 2'nd order. We can see that variable-wise-mapping corresponds to 1'st stage and term-wise-mapping corresponds to 2'nd stage of both forms of AND-OR or OR-AND switching circuit. Accordingly it is easy to design switching circuit directly from a map whichever sum-of-product or product-of-sum.

2 Boolean Algebraic background of Dual Mapping

We know that, by canonical expansion theorem of Boolean Algebrae, arbitrary logical function Z can be expressed either by sum-of-minterm $Z_0(m)$ or product-of sum $Z_0(M)$, of which logical formulaic expression shown as following,

$$Z = Z_0(m) = \sum_0^{2^n-1} (k_i : m_i) \quad (5) \qquad \overline{Z}_0(m) = \sum_0^{2^n-1} (\overline{k}_i : m_i) \quad (6)$$

$$Z=Z_0(\mathrm{M})=\prod_{0}^{2^n-1}(\mathrm{k}_i+\mathrm{M}_{2^n-1-i}) \qquad (7) \qquad \overline{Z}_0(\mathrm{M})=\prod_{0}^{2^n-1}(\overline{\mathrm{k}}_i+\mathrm{M}_{2^n-1-i}) \qquad (8)$$

$$\mathrm{m}_i=\overline{\mathrm{M}}_{2^n-1-i} \qquad (9)$$

where n is number of variables; k_i are constants valued as 1 or 0 and whose suffix corresponds to that of minterm; Σ and Π are used as successive OR sum and successive AND product respectively.

Now, we can see, inspecting Map(M) of Appendix 6, that Y(M) of (4) is the only minimum form of product-of-sum of Y. But we can find, of Appendix 5, another form of sum-of-product for X as follows,

$$\mathrm{X}_1(\mathrm{m})=\mathrm{A}\overline{\mathrm{B}}\overline{\mathrm{D}}+\mathrm{B}\overline{\mathrm{C}}\mathrm{D}+\overline{\mathrm{A}}\mathrm{B}\overline{\mathrm{D}}+\mathrm{ACD}+\overline{\mathrm{A}}\mathrm{C}\overline{\mathrm{D}} \quad (=\mathrm{a}+\mathrm{d}+\mathrm{e}+\mathrm{g}+\mathrm{i}) \qquad (10)$$

We can also see, by further inspection, that the pattern of : for X and that of + for Y(M) have wholly supplemental appearance. Accordingly we can have jointed form of two maps as is illustrated on Appendix 7. This facts hows that X(m) or X_1(m) addressed by "m" and Y(M) addressed by "M" should be different expression of a function Z. Accordingly we can write hereafter as follows,

$$\mathrm{X}_1(\mathrm{m})=\mathrm{Z}_1(\mathrm{m}) \qquad \mathrm{X}(\mathrm{m})=Z_2(\mathrm{m}) \qquad \mathrm{Y}(\mathrm{M})=Z(\mathrm{M}) \qquad (11)$$

It is important that, as for jointed form of Map such as Appendix 7, the Map has partially different address system, "m" for : and "M" for + . And is also important tbat there are logical AND between : of written minterm and 1 of valued cell where addressed by "m" and also logical OR between + of written maxterm and 0 of valued cell where addressed by "M". Existence of these constants 1 or 0 means none of inconsistency, because 0's are used for absent terms on plotting Z(m), and 1's are used for absent terms on plotting Z(M).

We also can have a map in seperated form such as illustrated on Appendix 8, and also arrange, hereafter, that Space on Map(M) means 1 for inspecting convenience.

It is very useful, based on these arrangement, that we can get four of logical informations from a Dual Map by selecting proper address for reading out : , +, and Space Pattern as illustrated on Table 1, on which reading out notations are seen.

Table 1 tells us that logical informations of Z are gotten as Z/ : (m)"m" (sum-of-product) or Z/0(m)"M" (product-of-sum) on Map(m), and also tells us that of $\overline{Z}$ are gotten as Z/0(m)"m" by expansion law or as Z/ : (m)"M" by Morgan's law. Accordingly Z/0(m)"M" is gotten as negation of $\overline{\mathrm{Z}}$/0(m)"m" by Morgan's law or as negation of $\overline{\mathrm{Z}}$/ : (m)"M" by expansion law. Procedure and result of reading out

Table 1 Logical information of Function Z onseperated Dual Map.

Symboles / Read by	Map (m) :	Map (m) Space	Map (M) +	Map (M) Space
"m"	$Z_0(m)=Z/:(m)$"m"	$\overline{Z}_0(m)=Z/0(m)$"m"	$\overline{Z}_0(m)=Z/+(M)$"m"	$Z_0(m)=Z/0(M)$"m"
"M"	$\overline{Z}_0(M)=Z/:(m)$"M"	$Z_0(M)=Z/0(m)$"M"	$Z_0(M)=Z/+(M)$"M"	$\overline{Z}_0(M)=Z/0(M)$"M"

(NOTATION) $Z/:(m)$"m" means logical information of Z of plotted: pattern on Map (m) read out by minterm address "m". $Z/0(M)$"M" means logcical information of Z of Space pattern on Map (M), which has plotted +'s, read out by maxterm address "M".

+ or Space from Map(M) is the same as from Map(m), and also we can have four of logical informations of Z.

Note that the logical informations gotten on Table 1 are wholly correspond to those of fundamental equation (5)～(8). Accordingly we can rewrite all of them using the read out notation illustrated on Table 1, of which formulaically as follows,

$$Z_0(m)=Z/:(m)\text{"m"}=m_2+m_4+m_5+m_6+m_8+m_{10}+m_{11}+m_{13}+m_{15} \quad (12)$$

$$Z_0(M)=Z/+(M)\text{"M"}=M_{15}M_{14}M_{12}M_8M_6M_3M_1 \quad (13)$$

$$Z/0(m)\text{"m"}=Z/0(M)\text{"m"}=m_0+m_1+m_3+m_7+m_9+m_{12}+m_{14}=\overline{Z}_0(m) \quad (14)$$

$$Z/0(M)\text{"M"}=Z/0(m)\text{"M"}=M_{13}M_{11}M_{10}M_9M_7M_5M_4M_2M_0 \quad =\overline{Z}_0(M) \quad (15)$$

We know (13) has the only minimum form of (4). On the other hand there are many ways to get simplified form of (12), but any one of them should be included in (13), because $Z_0(M)=Z_0(m)$. We have, by expanding (13) as follows,

$$\begin{aligned} Z(M) &= (A\overline{D}+B+C)(A\overline{B}+AD+\overline{A}\overline{C}+\overline{B}\overline{C}+\overline{C}D+\overline{A}\overline{D}+\overline{B}\overline{D}) \\ &= A\overline{B}\overline{D}+ABD+\overline{A}B\overline{C}+B\overline{C}D+\overline{A}B\overline{D}+A\overline{B}C+ACD+\overline{A}C\overline{D}+\overline{B}C\overline{D} \\ &= a+b+c+d+e+f+g+h+i = Z(\hat{m}) \end{aligned} \quad (16)$$

where $Z(\hat{m})$ is used as complete sum of Z.

Note that (16) includes all of terms of (3) and (10), moreover the new term f. Confirmation of (16) can be seen on Fig. 1, Prime Implicants Table by Quine-McCluskey methodes.[1) 2)] Truth Table of complete sum and Map(m, M) of (16) are shown on Fig. 2, on which we can see that each terms of (16) are cyclic, that

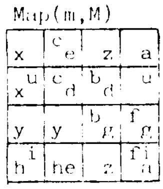
Fig. 1 Prime Implicants Table for Z(m)

	m_2	m_4	m_5	m_6	✓ m_8	✓ m_{10}	m_{11}	m_{13}	m_{15}
(a) $(A:\overline{B}:\overline{D})$					+	+			
b $(A:B:D)$								+	+
g $(A:C:D)$							+		+
d $(B:\overline{C}:D)$			+					+	
e $(\overline{A}:B:\overline{D})$		+		+					
f $(A:\overline{B}:C)$						+	+		
c $(\overline{A}:B:\overline{C})$		+	+						
h $(\overline{A}:C:\overline{D})$	+			+					
i $(\overline{B}:C:\overline{D})$	+					+			

Fig. 2 Truth Table and Cyclic Map for Z(m) and Z(M)

	A	B	C	D
(a)	1	0	–	0
b	1	1	–	1
g	1	–	1	1
d	–	1	0	1
e	0	1	–	0
f	1	0	1	–
c	0	1	0	–
h	0	–	1	0
i	–	0	1	0

Map(m, M)



is there are many combinations which form the same pattern of : on Map(m). We could have found (10) as one of these combinations avoiding duplicated utilization of terms and observing common use of $\bar{D}$ of essential term (a).

We can have another combinations by expanding Petrick function[3) 4)], as following,

$$\text{Petrick Function} = a(h+i)(c+e)(c+d)(e+h)(f+g)(b+d)(b+g) \quad (17)$$

$$= \underline{a}(\underline{hcbg} + \underline{hcdg} + \underline{hcbf} + hedbg + \underline{hedg} + hedbf + eicbg + eicdg + eicbf + \underline{eidg} + eidbg + ,,,,$$

Underlined five are to be selected, of which additional three combnations are as follows,

$$Z_3(m) = A\bar{B}\bar{D} + \bar{A}B\bar{C} + B\bar{C}D + ACD + \bar{A}C\bar{D} \quad (= a+c+d+g+h) \quad (18)$$

$$Z_4(m) = A\bar{B}\bar{D} + ABD + \bar{A}B\bar{C} + A\bar{B}C + \bar{A}C\bar{D} \quad (= a+b+c+f+h) \quad (19)$$

$$Z_5(m) = A\bar{B}\bar{D} + B\bar{C}D + \bar{A}B\bar{D} + ACD + \bar{A}C\bar{D} \quad (= a+d+e+g+h) \quad (20)$$

All of them are implied by $Z(\hat{m})$ of (16), but are different each other. Thus we can have following relations.

$$Z(M) = Z(\hat{m}) \; ; \quad Z_1(m) \subseteqq Z(\hat{m}) \; ; \quad Z_2(m) \subseteqq Z(\hat{m}) \; ;$$

$$Z_3(m) \subseteqq Z(\hat{m}) \; ; \quad Z_4(m) \subseteqq Z(\hat{m}) \; ; \quad Z_4(m) \subseteqq Z(\hat{m}) \; ; \quad (21)$$

Another example of cyclic prime implicants of product-of-sum which has dual relationship with (21) is seen on Function L, of which Truth Table of prime implicants and cyclic map are shown on Fig. 4, and formulaically given as follows,

$$L_0(M) = L/+(M)\text{“}M\text{”} = M_{14}M_{12}M_{11}M_{10}M_9\, M_7\, M_6\, M_5\, M_1 \quad (22)$$

$$L_1(M) = (A+B+\bar{D})(A+C+\bar{D})(A+\bar{B}+D)(\bar{A}+B+C)(\bar{A}+\bar{C}+D) \quad (= xyvwr) \quad (23)$$

$$L_2(M) = (A+B+\bar{D})(B+C+\bar{D})(A+\bar{B}+C)(\bar{A}+B+D)(\bar{B}+\bar{C}+D) \quad (= xzupq) \quad (24)$$

$$L_3(M) = (A+B+\bar{D})(A+\bar{B}+C)(\bar{A}+B+C)(\bar{B}+\bar{C}+D)(\bar{A}+\bar{C}+D) \quad (= xuwqr) \quad (25)$$

$$L_4(M) = (A+B+\bar{D})(A+\bar{B}+C)(\bar{A}+B+C)(\bar{A}+B+D)(\bar{B}+\bar{C}+D) \quad (= xuwpq) \quad (26)$$

$$L_5(M) = (A+B+\bar{D})(A+\bar{B}+C)(A+\bar{B}+D)(\bar{A}+B+C)(\bar{A}+\bar{C}+D) \quad (= xuvwr) \quad (27)$$

$$L_0(m) = L/0(M)\text{“}m\text{”} = m_0 + m_2 + m_7 + m_{11} + m_{12} + m_{13} + m_{15} \quad (28)$$

$$L(m) = \bar{A}\bar{B}\bar{D} + AB\bar{C} + ACD + BCD \quad (= a+b+c+d) \quad (29)$$

We can see that (29) is the only minimum sum-of-product of L.

Confirmation by Prime Implicants Table is shown on Fig. 3, and we can have complete product $L(\hat{M})$ as following

$$L(\hat{M}) = (A+B+\bar{D})(A+C+\bar{D})(B+C+\bar{D})(A+\bar{B}+C)(A+\bar{B}+D)(\bar{A}+B+C)$$
$$(\bar{A}+B+D)(\bar{B}+\bar{C}+D)(\bar{A}+\bar{C}+D) \quad (= xyzuvwpqr) \quad (30)$$

We can have $L(\hat{m})$ by expanding (30), as following

$$L(\hat{m}) = (A+\bar{D}+BC)(A+\bar{B}+CD)(\bar{A}+B+CD)(\bar{C}+D+\bar{A}\bar{B})(B+\bar{D}+AC)$$

$$=(A+\overline{D}+BC)(CD+AB+\overline{A}\overline{B})(\overline{C}+D+\overline{A}\overline{B})(B+\overline{D}+AC)$$

$$=\overline{A}\overline{B}\overline{D}+AB\overline{C}+ACD+BCD \tag{31}$$

We can confirm, by (31), that L(m) of (29) is the only minimum sum-of-product of L. Expanded form of (23)~(27), $L_1(M)$ for example, are as follows,

$$L_1(M)=(xv)(yw)r=(A+BD+\overline{B}\overline{D})(C+\overline{A}\overline{D}+\overline{B}\overline{D}+AB)(\overline{A}+\overline{C}+D)$$

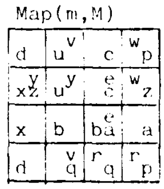
$$=a+b+c+d+ABD \tag{31'}$$

Fig. 3 Prime Implicants Table for L(M)

		M_{14} ✓	M_{12} ✓	M_{11}	M_{10}	M_9	M_7	M_6	M_5	M_1
(x)	$(A+B+\overline{D})$	✓	✓							
y	$(A+C+\overline{D})$	✓			✓					
z	$(B+C+\overline{D})$	✓						✓		
u	$(A+\overline{B}+C)$			✓	✓					
v	$(A+\overline{B}+D)$			✓		✓				
w	$(\overline{A}+B+C)$						✓	✓		
p	$(\overline{A}+B+D)$						✓		✓	
q	$(\overline{B}+\overline{C}+D)$					✓				✓
r	$(\overline{A}+\overline{C}+D)$								✓	✓

Fig. 4 Truth Table and Cyclic Map for L(M) and L(m)

	A	B	C	D
(x)	1	1	–	0
y	1	–	1	0
z	–	1	1	0
u	1	0	1	–
v	1	0	–	1
w	0	1	1	–
p	0	1	–	1
q	–	0	0	1
r	0	–	0	1



Thus we can write down the dual relations of (21) for L as following,

$$L(m)=L(\hat{M})\ ;\quad L_1(M)\supseteqq L(\hat{M})\ ;\quad L_2(M)\supseteqq L(\hat{M})\ ;$$

$$L_3(M)\supseteqq L(\hat{M})\ ;\quad L_4(M)\supseteqq L(\hat{M})\ ;\quad L_5(M)\supseteqq L(\hat{M})\ ; \tag{32}$$

It is important that both of (16) and (31) include none of redundant terms such as seen of none underlined terms of (17) on expanding Petrick Function. It means that the only minimum function, be it whatever sum-of-product or product-of-sum, precisely correspond to complete sum or complete product. Thus reading out minimum function on Dual Map offers useful means to get complete sum or complete product, and is applicable to synthesize or to decompose switching functions and to design a switching circuits.

3 Sythesizing or Decomposing a switching functions on Dual Map

Direct synthesizing a sum-of-product with a product-of-sum, whatever be it logical AND or logical OR, can be easilly done on Dual Map, because functional transformation is easy. Example, as for synthesizing Exclusive-OR Function $K=Z\overline{L}+\overline{Z}L$, are shown on Fig. 5 ~ Fig. 8.

Fig. 5 Synthesis P(M)=Z(M) : L(m)

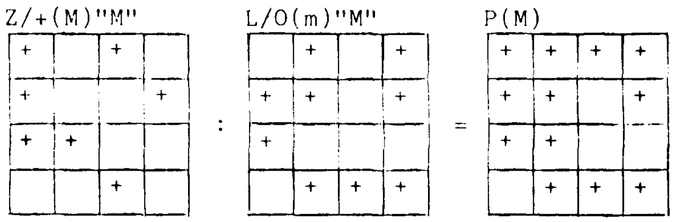


Fig. 6 Synthesis Q(M)=Z(M) : L(m)

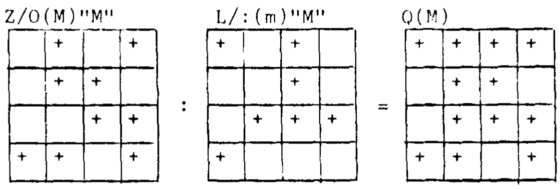


Fig. 7 Synthesis R(m)=P(m)+Q(m)

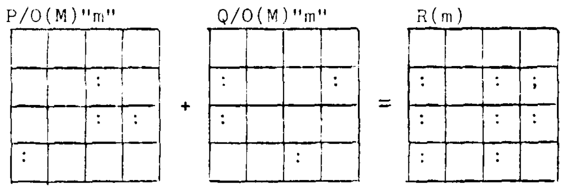
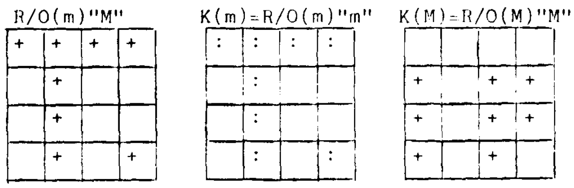


Fig. 8 Function K as $\overline{\text{R}}$(m) ; and as $\overline{\text{R}}$(M)



We can have, for given Z(M) and L(m), function P(M) directly from Fig. 5 as following,

$$\begin{aligned} P(M) &= Z(M) : L(m) = (Z/+(M)\text{“M”}) : (L/0(m)\text{“M”}) \\ &= (A+\bar{D})(C+B)(\bar{B}+D)(\bar{A}+D) \end{aligned} \tag{33}$$

$$P(m) = P/0(M)\text{“m”} = ABD+ACD+\bar{A}\bar{B}C\bar{D} \tag{34}$$

L/0(m)“M” of (33) include nine of cyclic prime implicants as had been shown, but we need not worry about because they are complete product. We also can have function Q(M) directly from Fig. 6 as follows,

$$\begin{aligned} Q(M) &= \bar{Z}(M) : \bar{L}(M) = (Z/0(M)\text{“M”}) : (L/:(m)\text{“M”}) \\ &= (\bar{B}+\bar{D})(B+D)(C+D)(A+\bar{B})(\bar{A}+B+\bar{C}) \end{aligned} \tag{35}$$

$$Q(m) = Q/0(M)\text{“m”} = \bar{A}\bar{B}D+\bar{B}\bar{C}D+ABC\bar{D} \tag{36}$$

Then, PARITY-OR function $R=ZL+\bar{Z}\bar{L}$ can be gotten on Fig. 7, forming OR-SUM of Fig. 5 and Fig. 6, as follows,

$$R(m) = (P/0(M)\text{“m”})+(Q/0(M)\text{“m”}) = AD+\bar{B}D+\bar{A}\bar{B}C+ABC \tag{37}$$

$$R(M) = R/0(m)\text{“M”} = (A+\bar{B})(C+D)(\bar{A}+B+D) \quad \text{(see Fig. 8)} \tag{38}$$

Thus we can have Exclusive-OR function K, by negating R, as shown on Fig. 8, of which formulaicaly as follows,

$$K(m) = R/0(m)\text{“m”} = \bar{A}B+\bar{C}\bar{D}+A\bar{B}\bar{D} \ ; \quad K(M) = R/:(m)\text{“M”} \tag{39}$$

$$K(M) = R/:(m)\text{“M”} = (\bar{A}+\bar{D})(B+\bar{D})(A+B+\bar{C})(\bar{A}+\bar{B}+\bar{C}) \tag{40}$$

Decomposition of a function to a form of (simple product)-OR-(pro-duct-of sum) is easilly done by Dual Mapping. Fig. 9 shows a decomposition of L to form of $ABD+L'(M)$, for given $L(m)$ of (29), of which illustrated as following,

$$L(M)=L/:(m)\text{"}m\text{"}=ABD+L/0(m)\text{"}M\text{"}=ABD+L_1(M) \quad (41)$$

$$\text{or } ABD+L_2(M) \text{ or } ABD+L_3(M) \text{ or } ABD+L_4(M) \text{ or } ABD+L_5(M)$$

Fig. 10 shows a decomposition of Z to form of $ABD+ACD+Z'(M)$ for given $Z(M)$ of $Z(M)=Y(M)=xyzu$ of (4), of which illustrated as follows,

$$Z(M)=Z/+(M)\text{"}M\text{"}=Z/0(M)\text{"}m\text{"}=ABD+ACD+(a+c+h)/0(m)\text{"}M\text{"}$$

$$=ABD+ACD+(\overline{A}+\overline{B})(\overline{C}+\overline{D})(\overline{A}+\overline{D})(A+B+C) \quad (42)$$

Fig. 11 shows decomposition of K to form of $(\overline{A}+\overline{B}+\overline{C}):K'(m)$, for given of (40), of which illustrated as follows,

$$K(M)=(\overline{A}+\overline{B}+\overline{C}):(B+\overline{D})(A+B+\overline{C})(\overline{A}+C+\overline{D}) \quad (43)$$

$$=(\overline{A}+\overline{B}+\overline{C}):K(m)=(\overline{A}+\overline{B}+\overline{C}):K/0(M)\text{"}m\text{"}$$

$$=(\overline{A}+\overline{B}+\overline{C})(\overline{A}B+\overline{C}\overline{D}+BC+A\overline{D})$$

Fig. 9 Decomposition $L(M)=e+L'(M)$

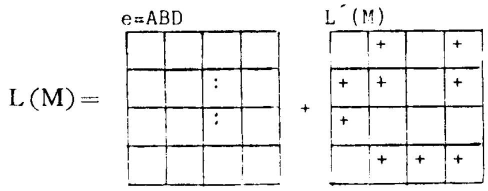


Fig. 10 Decomposition $Z(M)=b+g+Z'(M)$

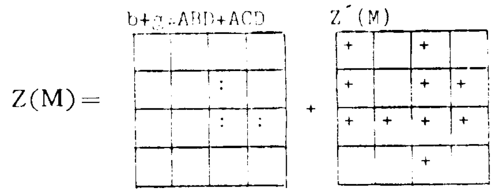
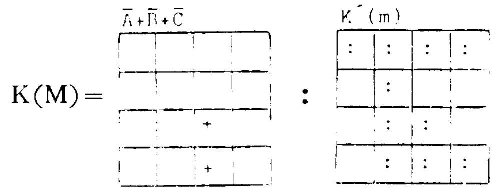


Fig. 11 Decomposition $K(M)=(\overline{A}+\overline{B}+\overline{C}):K'(m)$



4 Designing minimum switching circuits of 2'nd order by Dual Mapping

Minimal sum-of-product is not always minimal of switching circuit of 2'nd order, and also the same is as for minimum propuct-of-sum. we should inspect both forms of given function. On this point of view Dual Mapping offers very useful means, being easy to inspect both forms on same field of a map. We could have had minimum product-of-sum $Z(M)$ as the only minimal one of given function Z, and also $L(m)$ of product-of-sum as the only minimal one of given L. Moreover, we can design a circuit directly from a map whatever be it sum-of-product or product-of-sum. Discrimination which one be selected is very easy on Dual Map. We simply select the only minimal one without worrying about cyclic prime implicants, because the only minimal one is always just the same as complete sum or complete product, as had been shown.

OR-AND arrangement for Z(M) and AND-OR arrangement for L(m) are shown on Fig. 12 and Fig. 13 respectively. Both of them are the only minimum one of switching circuit of 2'nd order. Sometime we are forced to utilize either one of OR-AND or AND-OR of the case such as common use of logic elements. We are not sure, without any of restriction, which one to be selected of $Z_1(m) \sim Z_5(m)$ or $L_1(M) \sim L_5(M)$. Temporally selected examples are shown on Fig. 14 and on Fig. 15.

Fig. 12
OR-AND for Z(M)

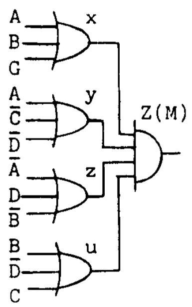


Fig. 13
AND-OR for L(m)

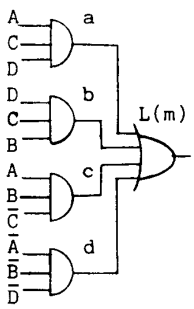


Fig. 14
AND-OR for $Z_1(m)$

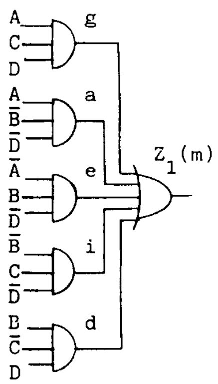
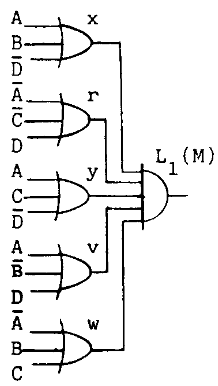


Fig. 15
OR-AND for $L_1(M)$



Once the minimum OR-AND or AND-OR found out, we can transform to NOR-NOR or to NAND-NAND arrangement by simply replacing all of logic elements. Newly gotten NOR-NOR (or NAND-NAND) has just the same input variables and number of logic elements of original OR-AND (or AND-OR).

There is restriction on designing NOR-NOR-NOR (simplified as 3-NOR) or NAND-NAND-NAND (3-NAND) that all of input variables should not be negated variables. We can design the input stage of 3-NOR or 3-NAND by utilizing logic elements as NOT elements for all of the input vaiables. But we seek for common use of logic elements to have more simplified form of the input circuit, utilizing following Boolean Alge-beaic relations.

$$A\bar{B}=A(\bar{A}+\bar{B})\ ; \qquad \bar{A}B=B(\bar{A}+\bar{B})\ ; \tag{44}$$

$$A+\bar{B}=A+\bar{A}\bar{B}\ ; \qquad \bar{A}+B=B+\bar{A}\bar{B} \tag{45}$$

(44) shows that paired left hand side AND term have common output $\bar{A}+\bar{B}$ of preceeding stage, and (45) shows that paired left hand side OR terms have common output $\bar{A}\bar{B}$ of prceeding stage. We can observe availabilities of these relations on truth tables illustrated on Table 2 and on Table 3[5] on which 0 factored as (0)

correspond to negated variables of left hand side of (44) and (45). 0 which d'nt factored by ( ) means the output of NOT elements to be utilized as possible as commonly. Designing 1'st stage of 3-NOR or 3-NAND are performed by finding the most effective combinations of these two kind of 0's. Designed 3-NOR for Z(M) and 3-NAND for L(m) are shown on Fig. 16 and on Fig. 17 respectively. Designed

Table 2 Truth table of 1'st stage of 3-NOR for Z(M) and 3-NAND for L(m)

		A	B	C	D
Z(M)	y	1	—	0	$(0)^1$
	z	$(0)^1$	$(0)^2$	—	1
	u	—	1	1	$(0)^2$
L(m)	c	1	1	0	—
	d	0	0	—	0

suffix correspond to No. of the gate

Table 3 Truth Table of 1'st stasge of 3-NAND for $Z_1(m)$ and 3-NOR for $L_1(M)$

		A	B	C	D
$Z_1(m)$	a	1	$(0)^1$	—	0
	e	$(0)^1$	1	—	0
	i	—	$(0)^2$	1	0
	d	—	1	$(0)^2$	1
$L_1(M)$	x	1	1	—	$(0)^1$
	r	$(0)^1$	—	$(0)^2$	1
	y	1	—	1	$(0)^2$
	v	1	$(0)^3$	—	$(0)^1$
	w	$(0)^3$	1	1	—

Fig. 16
3-NOR Arrangement of Z(M)

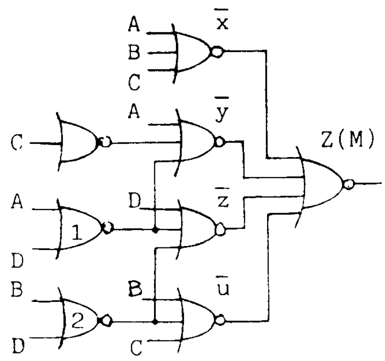
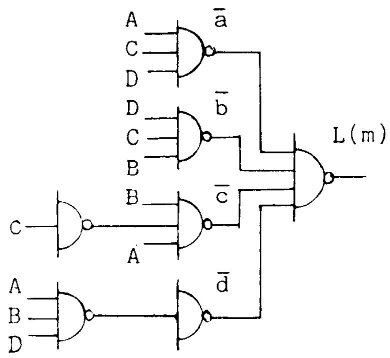
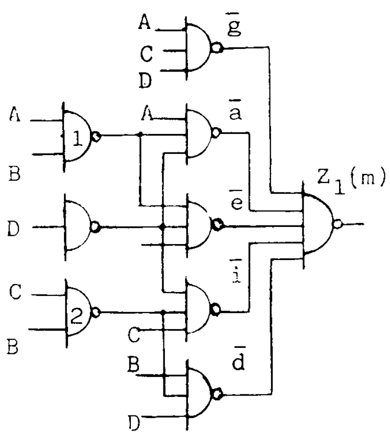


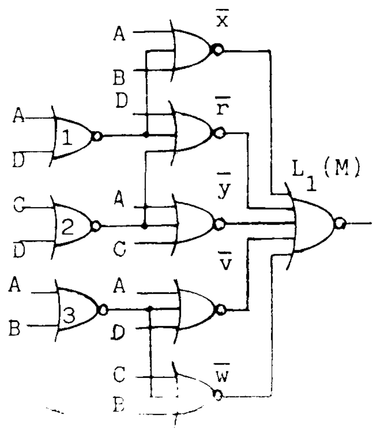
Fig. 17
3-NAND Arrangement of L(m)



Flg. 18 3-NOR for $Z_1(m)$



Flg. 19 3-NAND for $L_1(M)$



3-NAND for $Z_1(m)$ and 3-NOR for $L_1(M)$ are also shown on Fig. 18 and on Fig. 19 respectively.

It can be said, arranging all of above discussion, that Dual Mapping offers useful means to process switching functions and to design switching circuits, and is expected widely use of related fields.

The auther would like thank Professor R. Hashimoto and other members of Department of Electronic Science, Okayama University of Science, for their advices of this paper.

Reference

1) McCluskey, E. Minimization of Boolean functions. Bell Syst. tech. J., 1956, 35, 1417–1444.
2) Quine, W.V. The ploblem of simplifying truth function. Am. math. Mon., 1952, 59, 521–531.
2) Quine, W.V. The ploblem of simplifying truth function. Am. math. Mon., 1952, 59, 521–531.
3) Petrick, S.R. A direct determination of the irredundant forms of a Booiean function from the set of prime implicants. Air Force Cambridge Research Center report Bedford, Mass. 1956. AFCRD-TR-56-110.
4) Pyne, I.B. and McCluskey, E. Reduction of redundancy in solving prime implicants tables. I.R.E. Transactions on electronic comp. 1962. EC 11, 473–482.
5) D.T. Ellis. A synthesis of combinational logic with NAND or NOR elements. IEEE Trans. Comput. EC-14, No. 5, October, 1965.

Appendix 1 Minterm address "m"

m_0	m_4	m_{12}	m_8	m_{16}	m_{20}	m_{28}	m_{24}
m_1	m_5	m_{13}	m_9	m_{17}	m_{21}	m_{29}	m_{25}
m_3	m_7	m_{15}	m_{11}	m_{19}	m_{23}	m_{31}	m_{27}
m_2	m_6	m_{14}	m_{10}	m_{18}	m_{22}	m_{30}	m_{26}

Appendix 2 Maxterm address "M"

M_{31}	M_{27}	M_{19}	M_{23}	M_{15}	M_{11}	M_3	M_7
M_{30}	M_{26}	M_{18}	M_{22}	M_{14}	M_{10}	M_2	M_6
M_{28}	M_{24}	M_{16}	M_{20}	M_{12}	M_8	M_0	M_4
M_{29}	M_{25}	M_{17}	M_{21}	M_{13}	M_9	M_1	M_5

Appendix 3 Formation of m_{15} by Veriable-wise Mapping on Map (m)

Original Map				: A =				: B =				: C =				: D = m_{15}			
1	1	1	1	0	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0
1	1	1	1	0	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0
1	1	1	1	0	0	1	1	0	0	1	0	0	0	1	0	0	0	1	0
1	1	1	1	0	0	1	1	0	0	1	0	0	0	1	0	0	0	0	0

Appendix 4 Formation of M_{15} by Variable-wise Mapping on Map(M)

Original Map				+A=				+B=				+C=				+D=M_{15}			
0	0	0	0	0	0	1	1	0	1	1	1	0	1	1	1	0	1	1	1
0	0	0	0	0	0	1	1	0	1	1	1	0	1	1	1	1	1	1	1
0	0	0	0	0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1
0	0	0	0	0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1

Appendix 5 Term-wise-mapping on Map(m)

$$X=a+b+c+g+h=A\overline{B}\overline{D}+ABD+\overline{A}B\overline{C}+ACD+\overline{A}C\overline{D}$$

Original Map				Mapping X				Plotting				Rearranging			
0	0	0	0	0	γ	0	α		:		:		c		a
0	ε	ε	0	0	γ	β	0		:	:			c	b	
0	0	δ	δ	0	0	β	0			:	:			bg	g
λ	λ	0	0	0	0	0	α	:	:		:	h	h		a

Space =0

Appendix 6 Term-wise mapping on Map(M)

$$Y= xyzu=(A+B+C)(A+\overline{C}+\overline{D})\ (\overline{A}+\overline{B}+D)\ (B+C+\overline{D})$$

Original MaP				Mapping Y				Plotting				Rearranging			
1	1	1	1	α	1	γ	1	+		+		x		z	
δ_ε	1	1	ε	α	1	1	1	+			+	xu			u
δ	1	1	1	β	β	1	1	+	+			y	y		
1	1	1	1	1	1	γ	1			+				z	

Space =1

Appendix 7 Jointed Map(m, M)

By terms

x	c	z	a
x u	c	b	u
y	y	b g	g
h	h	z	a

By symboles

+	:	+	:
+	:	:	+
+	+	:	:
:	:	+	:

Appendix 8 Seperated Map

by :

	:		:
	:	:	
		:	:
:	:		:

Space=0

by +

+		+	
+			+
+	+		
		+	

Space=1

Appendix 9 Logical properties of Dual Map

	logic between cell	value of operand / others	plotting	variable-wise mapping logic	term-wise mapping logic
Map(m)	OR	1 / 0	: (=1)	AND	OR
Map(M)	AND	0 / 1	+ (=0)	OR	AND