



## NAND セルの構成とその故障検出について\*

柴 谷 善 郎\*\* 久 津 輪 敏 郎\*\* 江 端 克 彦\*\*\*

### Abstract

A Nand cell which corresponds to logical expression simplified by Karnaugh Maps is proposed. This cell has two input, two output and four control terminals.

Eight logical functions can be realized at the vertical output terminal by setting three control signals. The horizontal output can be equal to either horizontal or vertical input by using a control signal.

Fault detecting procedures are considered for each signal line in the cell. A general function-realization using two dimensional cellular array is presented. The detection and the reparation of faulty rows, columns are discussed for the cellular array.

### 1. ま え が き

集積回路, LSI など半導体技術の発達にともない, 可変型論理回路(セル)の配列によって任意の論理関数を実現する問題が研究されている<sup>1)</sup>. この中でセル構造の類別や関数実現可能性などの理論的取扱いはかなり多いが, セルの2次元配列についての具体例はあまり見られない. 現在までに提案された具体例<sup>2), 3)</sup>では, 構造の複雑さ, 制御方法, 配列法等に問題が残されている.

本文では, これらの問題点を改善でき, 簡単な構成で任意関数を実現できるように, カルノー図から得た最簡略表現に対応する NAND 形セルを提案する. そしてセル配列による任意関数実現の一般的方法を述べ, 単一セルの故障検出法を考察する. さらにセル配列における故障の検出およびその補償について論ずる.

単一セルの故障検出については, セル内の各信号線の0, 1縮退故障を検出できる検査入力ベクトルの組を決定する. いままでのセルの故障検出ではセルの入・出力での縮退故障またはセル内の各ゲートの縮退

故障が主に考えられているが<sup>4)</sup>, 本文ではより完全に近い故障検出を目的として, 各信号線の縮退故障を対象とした.

セル配列での故障検出は, 故障の補償と関連づけて考えるべきである. セル配列は将来 LSI で構成されるものと考えられるが, このとき配列内の故障セルを正常なセルと取替えるのは容易でないから, 配列内での故障セルを含む行, 列の機能を正常な行, 列におきかえるという方法を探るのが妥当であろう. この観点から本文では, 単一セルの故障検出法を用いて故障セルを含む行, 列を検出し, それを補償する方法を論じている.

### 2. セルの構造

#### 2.1 セルの垂直論理回路

Fig. 1 (次頁参照)に示された2入力2出力セルの2次元配列によって任意の論理関数を実現するためには, セルの垂直出力  $z_v$  に Table 1 (次頁参照)の6個の2変数論理関数を実現できることが必要十分な条件である<sup>2)</sup>. また, セルの2次元配列のためには, 水平出力  $z_h$  に  $x$  または  $y$  を実現できることが望ましい. いま制御入力  $S_1, \dots, S_2, S_3$  を用いて, 垂直出力  $z_v$  に Table 1 の6個の関数と定数  $\{0, 1\}$  とを実現し, 水平出力  $z_h$  に  $x$  または  $y$  を実現するセルの構成を検討する.

一般に  $t$  個の論理関数  $\{f_1(x, y), \dots, f_t(x, y)\}$  を

\* On the Constitution and Fault Detection of NAND Cells by Yoshio SHIBATANI, Toshiro KUTSUWA (Dept. of Electronics, Faculty of Engineering, Osaka Institute of Technology.) and Katsuhiko EBATA (Dept. of Electronics, Faculty of Engineering, Hiroshima-Denki Institute of Technology)

\*\* 大阪工業大学工学部電子工学科

\*\*\* 広島電機大学工学部電子工学科

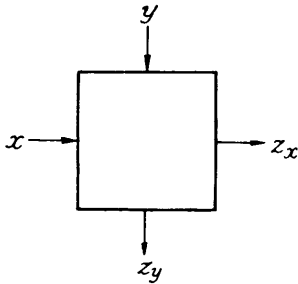


Fig. 1 2 inputs and 2 outputs cell.

Table 1 Logical function  $f_i(x, y)$ .

$i$	$f_i(x, y)$	
i	$\bar{x}\bar{y}$	または $\bar{z}_y$
ii	$x\bar{y}$	" $z_y$
iii	$\bar{y}$	" $y$
iv	$x\oplus y$	" $x\oplus\bar{y}$
v	$\bar{x}\vee\bar{y}$	" $\bar{x}\vee y$
vi	$x\vee\bar{y}$	" $x\vee y$

可変できる制御入力の個数  $r$  は

$$r \geq \log_2 t \quad (1)$$

の整数で与えられる。論理関数  $f_i(x, y)$  と制御入力ベクトル  $(S_r, \dots, S_2, S_1)_v$  とに 1 対 1 の対応をつけることにより回路を構成することができる。すなわち回路の出力で実現される合成論理関数  $F(x, y, S)$  は

$$F(x, y, S) = \bigvee_{i=1}^t f_i(x, y) \cdot (S_r, \dots, S_2, S_1)_v \quad (2)$$

で表現される。ここで  $S_j (j=r, \dots, 2, 1)$  は 0 または 1 であり、 $v$  は  $(S_r, \dots, S_2, S_1)$  を 10 進化した数値を示す。 $f_i(x, y)$  と  $(S_r, \dots, S_2, S_1)_v$  の対応のつけ方によって回路構成は異なるが、カルノー図の助けによって最簡略の回路構成を決定することができる<sup>5)</sup>。

セルの垂直出力  $z_v$  に 8 個の論理関数 (定数 0, 1 を含む) を実現するには、式 (1) より 3 個の制御入力  $S_3, S_2, S_1$  が必要であり、 $f_i(x, y)$  と  $(S_3, S_2, S_1)_v$  との最簡略の対応を求めれば Fig. 2 のようになる。すなわち  $\{f_i(x, y)\}$  として、

$$\{f_i(x, y)\} = \{\bar{x}\bar{y}, xy, y, x\oplus\bar{y}, \bar{x}\vee y, x\vee y, 1, 0\}$$

を選ぶとき、8 個の 1 が隣接する組による 1 項と 4 個の 1 が隣接する 3 組による項との論理和の形となり、合成論理関数  $F(x, y, S)$  は

$$F(x, y, S) = xS_2S_1 \vee yS_2 \vee xyS_1 \vee \bar{x}\bar{y}S_3 \quad (3)$$

で表わされる。これに対応する NAND 回路構成は Fig. 3 となる。

Table 1 の関数および  $\{0, 1\}$  の 8 個の論理関数を実現するための簡略表現は式 (3) だけではない。

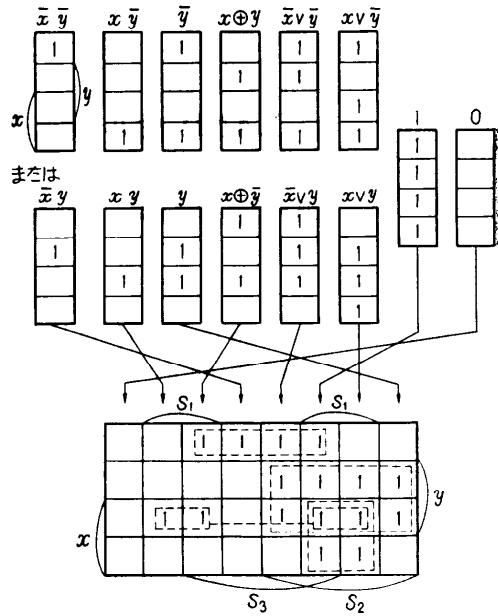


Fig. 2 Combination of  $f_i(x, y)$  with  $(S_1, S_2, S_3)_v$ .

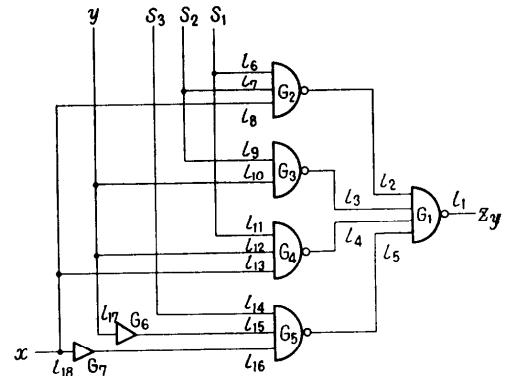


Fig. 3 Vertical logical circuit using NAND gates.

$\{f_i(x, y)\} = \{\bar{x}\bar{y}, xy, \bar{y}, x\oplus\bar{y}, \bar{x}\vee y, x\vee y, 1, 0\}$  を選べば、

$$F(x, y, S) = \bar{y}S_3 \vee xyS_2 \vee \bar{x}S_3S_1 \vee \bar{x}\bar{y}S_1 \quad (4)$$

が得られる。また、

$$\{f_i(x, y)\} = \{\bar{x}y, x\bar{y}, \bar{y}, x\oplus y, \bar{x}\vee\bar{y}, x\vee\bar{y}, 1, 0\}$$

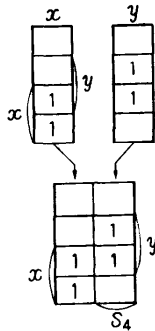
のとき、

$$F(x, y, S) = \bar{y}S_3 \vee x\bar{y}S_3 \vee xS_3S_2 \vee \bar{x}yS_1 \quad (5)$$

となる。式 (3), (4), (5) を比較すると、どれも項数 4, 位数 (変数の総和) 11, 補数化された変数名の数の 2 で表現されており、回路構成に必要なゲート数は 7 である。しかし、補数化された変数の総和が、そ

**Table 2** Logical operation of vertical logical circuit.

垂直インデックス $v$	制御入力ベクトル $S_3, S_2, S_1$			出力関数 $z_v = F(x, y, S)$
0	0	0	0	0
1	0	0	1	$xy$
2	0	1	0	$y$
3	0	1	1	$x \vee y$
4	1	0	0	$\bar{x}\bar{y}$
5	1	0	1	$x \oplus \bar{y}$
6	1	1	0	$\bar{x} \vee y$
7	1	1	1	1



**Fig. 4** Map of vertical output.

それぞれ2, 4, 3であることから, 式(3)の表現が他より簡略であるとみることができる。

**Table 2** は Fig. 3 の回路における制御入力ベクトル  $(S_3, S_2, S_1)$  と出力関数  $F(x, y, S)$  の関係を示す。制御入力ベクトルを10進数化した  $v(v)_{10} = (S_3, S_2, S_1)_2$  を垂直インデックスと呼び,  $v$  を用いてセルの垂直方向の論理動作を表わす。

**2.2 セルの水平論理回路**

水平出力  $z_x$  に  $x$  または  $y$  を出す回路については, 式(1)より1個以上の制御入力が必要である。これを  $S_4$  とするとき, **Fig. 4** から

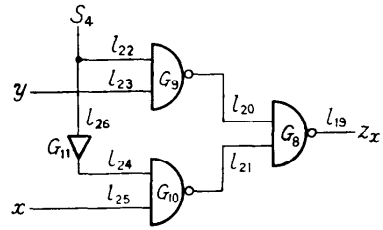
$$z_x = x\bar{S}_4 \vee yS_4$$

$$\text{または } z_x = xS_4 \vee y\bar{S}_4$$

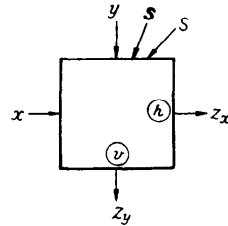
が得られる。したがって **Fig. 5** の回路構成が得られ, 制御入力  $S_4$  と水平出力  $z_x$  の関係は **Table 3** で表わされる。

**Table 3** Logical operation of horizontal logical circuit.

水平インデックス $h$	制御入力 $S_4$	水平出力 $z_x$
0	0	$x$
1	1	$y$



**Fig. 5** Horizontal logical circuit.



**Fig. 6** Representation of the cell.

Fig. 3 の垂直論理回路と Fig. 5 の水平論理回路とを1個のセルにまとめ, これを **Fig. 6** のように表示する。⓪, ①には垂直, 水平のインデックスを記入して, 両方向の論理動作を表わす。

**3. セル配列による論理関数実現の一般的方法**

2変数の任意の論理関数が, 前述のセルを2個縦続接続することによって, 実現されることは明らかである<sup>2)</sup>。3変数の任意の論理関数  $f(x_1, x_2, x_3)$  については,

$$f(x_1, x_2, x_3) = f(x_1, x_2, 0)\bar{x}_3 \vee f(x_1, x_2, 1)x_3 \tag{6}$$

に展開できるから, **Fig. 7** (a) (次頁参照) の構成によって実現できる。ここで  $F_3, J_2, K_2$  はそれぞれ  $f(x_1, x_2, x_3), f(x_1, x_2, 0), f(x_1, x_2, 1)$  を表わす。 $J_2, K_2$  は2変数の論理関数であり, セルを2個縦続して実現できる。式(6)はまた, つぎのように書ける。

$$F_3 = \bar{J}_2 \bar{x}_3 \vee K_2 x_3$$

したがって上側のセルで  $K_2$  と  $\bar{J}_2$  ( $J_2$  の補数) をつくることにより, Fig. 7 (b) の構成で  $F_3$  を実現することもできる。

一般に  $n$  変数の論理関数は

$$f(x_1, x_2, \dots, x_n) = f(x_1, x_2, \dots, x_{n-1}, 0)\bar{x}_n \vee f(x_1, x_2, \dots, x_{n-1}, 1)x_n$$

したがって

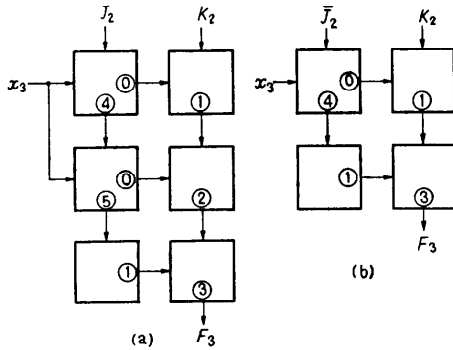


Fig. 7 Realization of  $F_3 = j_2 \bar{x}_3 \vee K_2 x_3$

$$F_n = J_{n-1} \cdot \bar{x}_n \vee K_{n-1} \cdot x_n$$

で表される. 上の方法によって  $n-1$  変数関数  $K_{n-1}$ ,  $J_{n-1}$  (または  $\bar{J}_{n-1}$ ) を実現し, Fig. 7 で  $J_2, K_2, x_3$  をそれぞれ  $J_{n-1}, K_{n-1}, x_n$  におきかえることにより  $n$  変数の論理関数を実現できる.

以上は任意関数実現の一般的方法を述べたものであるが, 特定の論理関数を実現するには, より小さい配列でよいことは当然である.

4. 単一セルの故障検出

セル配列によって任意の論理関数を実現する場合, 配列内の各セルに故障がないか調べる必要がある. セルの故障検出においては, 一般の論理回路での故障検出と同様に, セル内の信号線が0または1に縮退する形の故障を扱う.

ここで, 一つの信号線が二つ以上に分岐している場合には, 各信号線の故障を別々に検査しなければならない. なぜなら, Fig. 3 において信号線  $S_1$  が  $l_6, l_{11}$  に分岐して  $G_2, G_4$  への入力線となると,  $l_6$  の断線 ( $G_2$  入力線の1縮退故障) は  $l_{11}$  すなわち  $G_4$  入力線によっては検査できず, 逆に  $l_{11}$  の故障も  $G_2$  の入力線で検査できないからである.

一方,  $G_6$  の入(出)力線  $l_{17} (l_{16})$  については, 入力(出力)線の0縮退故障が出力(入力)線の縮退故障と等価であるから, 入力線または出力線のどちらか一方について縮退故障を検査すればよい.  $G_7$  の入(出)力線  $l_{18} (l_{16})$  についても同様である.

4.1 垂直論理回路の故障検出

本文のセルの垂直論理回路はカルノー図を用いて最簡略化された NAND ゲート回路であり, 出力側から1段目を OR ゲートに2段目を AND ゲートにおきかえると, 出力関数を変化させることなく冗長のない

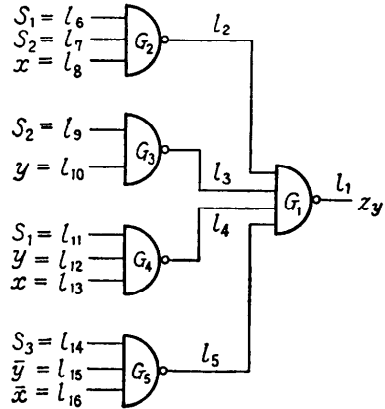


Fig. 8 2 Level Nand circuit equivalent to Fig. 3.

AND—OR 回路に変えることができる. また前述の線の分岐を考慮して, Fig. 8 の等価な2レベル NAND ゲート回路におきかえることができる. そこで2レベル NAND ゲート回路の故障検出法を考察し, それを本セルの回路に適用する. Fig. 8 の2レベル

Table 4 Fault-testing table for vertical logical circuit.

検査入力 インテック クス	検査入力 ベクトル x y S <sub>3</sub> S <sub>2</sub> S <sub>1</sub>	ゲート G <sub>2</sub>			ゲート G <sub>3</sub>		ゲート G <sub>4</sub>			ゲート G <sub>5</sub>					
		x l <sub>6</sub>	S <sub>2</sub> l <sub>7</sub>	S <sub>1</sub> l <sub>8</sub>	y l <sub>10</sub>	S <sub>2</sub> l <sub>9</sub>	x l <sub>11</sub>	y l <sub>12</sub>	S <sub>1</sub> l <sub>13</sub>	y l <sub>15</sub>	S <sub>3</sub> l <sub>14</sub>	S <sub>2</sub> l <sub>16</sub>			
0 縮 退 検 査	l <sub>4</sub>	0	0	1	0	0						*	*	*	
	l <sub>6</sub>	0	0	1	0	1							*	*	*
	l <sub>8</sub>	0	0	1	1	0							*	*	*
	l <sub>7</sub>	0	0	1	1	1							*	*	*
	l <sub>10</sub>	0	1	0	1	0		*	*						
	l <sub>11</sub>	0	1	0	1	1		*	*						
	l <sub>14</sub>	0	1	1	1	0		*	*						
	l <sub>15</sub>	0	1	1	1	1		*	*						
	l <sub>19</sub>	1	0	0	1	1	*	*	*						
	l <sub>23</sub>	1	0	1	1	1	*	*	*						
1 縮 退 検 査	l <sub>25</sub>	1	1	0	0	1				*	*	*			
	l <sub>26</sub>	1	1	0	1	0				*	*				
	l <sub>29</sub>	1	1	1	0	1				*	*				
	l <sub>30</sub>	1	1	1	1	0				*	*				
	l <sub>0</sub>	0	0	0	0	0									*
	l <sub>1</sub>	0	0	0	0	1									*
	l <sub>2</sub>	0	0	0	1	0				*					*
	l <sub>3</sub>	0	0	0	1	1	*			*					*
	l <sub>5</sub>	0	1	0	0	0				*					*
	l <sub>9</sub>	0	1	0	0	1				*	*				*
検 査	l <sub>12</sub>	0	1	1	0	0				*	*				*
	l <sub>13</sub>	0	1	1	0	1				*	*				*
	l <sub>17</sub>	1	0	0	0	1	*					*			*
	l <sub>18</sub>	1	0	0	1	0	*			*					*
	l <sub>20</sub>	1	0	1	0	0				*	*				*
	l <sub>21</sub>	1	0	1	0	1	*			*					*
	l <sub>22</sub>	1	0	1	1	0	*			*	*				*
	l <sub>24</sub>	1	1	0	0	0				*	*				*
	l <sub>28</sub>	1	1	1	0	0				*	*				*

NAND ゲート回路について、故障検査表<sup>6)</sup>をつくれれば Table 4 (前頁参照)を得る。表中の列方向には各ゲート名とそれへの入力線を書き、行方向に検査入力ベクトルを書く。ここでゲート入力線の0縮退(出力線の1縮退)故障、ゲート入力線の1縮退(出力線の0縮退)故障を検出する入力の組合せを、それぞれ0縮退、1縮退検査入力ベクトルとよび検査表に区別して書き入れる。

0縮退検査入力ベクトルについては、一つのNANDゲート  $G_k$  の出力が0で、それ以外のゲート出力が1である入力組合せを求める。この入力ベクトルは  $G_k$  の各入力線の0縮退(出力線の1縮退)故障を検出できる。したがって検査表においてその入力ベクトルと入力線とが対応する位置に、検出可能を示す\*を書き入れる。この場合検出可能な入力線(出力線)が0縮退(1縮退)故障ならば回路出力  $z_v = F(x, y, S)$  は0となり、正常なとき  $z_v$  は1である。

1縮退検査入力ベクトルはつぎのようにして求められる。 $z_v = F(x, y, S) = 1$  となる入力ベクトルに隣接する入力ベクトル(一つの入力変数だけを前者の補数にした入力ベクトル)がNANDゲート  $G_k$  の出力を1にし、回路出力  $z_v$  を0にすると、この入力ベクトルは  $G_k$  の一つの入力線の1縮退(出力線の0縮退)故障を検出できる。故障検出される入力線は、上で補数化した入力変数に対応する入力線であって、これが1縮退故障ならば  $G_k$  の出力は0、 $z_v$  は1となる。故障がなければ  $G_k$  の出力は1で、 $z_v$  は0である。

Table 4 から各ゲートの入、出力線の0縮退、1縮退故障を検出可能な検査入力ベクトルの最小集合を求める。0縮退検査入力ベクトルとしては、

$$\begin{aligned} & ((t_4, t_5, t_6, t_7)), \\ & ((t_{10}, t_{11}, t_{14}, t_{15}, t_{26}, t_{30})), \\ & ((t_{19}, t_{23})), ((t_{25}, t_{29})) \end{aligned}$$

が必要である。ここで(( ))はその中のどれか一つを採択することを示す。

1縮退検査入力ベクトルの最小集合は、

$$\begin{aligned} & t_3, t_{13}, ((t_{24}, t_{28})), \\ & ((t_{22} \text{ と } ((t_{17}, t_{21})), t_{21} \text{ と } ((t_{18}, t_{22})))) \end{aligned}$$

である。

ここで検査入力ベクトルの簡潔な組合せとして Table 5 の  $T_1, \dots, T_5$  を用いるとき、回路出力  $z_v = F(x, y, S)$  が全て  $\phi \equiv (1, 0)$  ならば垂直論理回路は正常であり、 $0 \equiv (0, 0)$ 、 $1 \equiv (1, 1)$  または  $\eta \equiv (0, 1)$  となる出力があれば回路に故障があることになる。 $T_1, \dots,$

Table 5 Combination of detecting input vectors.

入力ベクトルの組	x	入 y	S <sub>3</sub>	力 S <sub>2</sub>	S <sub>1</sub>	正 常 出 力
$T_1 = (t_7, t_8)$	0	0	$\phi$	1	1	$\phi$
$T_3 = (t_{13}, t_{13})$	0	1	1	$\phi$	1	$\phi$
$T_4 = (t_{23}, t_{23})$	1	1	1	0	$\phi$	$\phi$
$T_4 = (t_{23}, t_{21})$	1	0	1	$\phi$	1	$\phi$
$T_5 = (t_{23}, t_{22})$	1	0	1	1	$\phi$	$\phi$

Table 6 Output response in the case of single fault.

出力 応 答 インデックス	検 査 入 力					故 障 状 態
	T <sub>1</sub>	T <sub>2</sub>	T <sub>3</sub>	T <sub>4</sub>	T <sub>5</sub>	
R <sub>0</sub>	$\phi$	$\phi$	$\phi$	$\phi$	$\phi$	正 常
R <sub>2</sub>	$\phi$	$\phi$	$\phi$	$\phi$	1	$t_6=1$
R <sub>4</sub>	$\phi$	$\phi$	$\phi$	0	0	[[ $t_6=0, t_7=0, t_8=0, t_2=1$ ]]
R <sub>6</sub>	$\phi$	$\phi$	$\phi$	1	$\phi$	[[ $t_7=1, t_{13}=1$ ]]
R <sub>8</sub>	$\phi$	$\phi$	$\phi$	1	1	[[ $t_{13}=0, t_{16}=1$ ]]
R <sub>9</sub>	$\phi$	$\phi$	0	$\phi$	$\phi$	[[ $t_{11}=0, t_{12}=0, t_{13}=0, t_4=1$ ]]
R <sub>10</sub>	$\phi$	$\phi$	1	$\phi$	$\phi$	$t_{11}=1$
R <sub>17</sub>	$\phi$	0	$\phi$	$\phi$	$\phi$	[[ $t_6=0, t_{10}=0, t_2=1$ ]]
R <sub>14</sub>	$\phi$	1	$\phi$	$\phi$	$\phi$	[[ $t_{17}=0, t_{13}=1, t_{11}=1$ ]]
R <sub>12</sub>	$\phi$	1	1	$\phi$	$\phi$	$t_6=1$
R <sub>31</sub>	0	$\phi$	$\phi$	$\phi$	$\phi$	[[ $t_{14}=0, t_{13}=0, t_{16}=0$ ]] [[ $t_{17}=1, t_{13}=1, t_6=1$ ]]
R <sub>121</sub>	0	0	0	0	0	$t_1=0$
R <sub>102</sub>	1	$\phi$	$\phi$	$\phi$	$\phi$	[[ $t_6=1, t_{14}=1$ ]]
R <sub>104</sub>	1	$\phi$	$\phi$	$\phi$	1	$t_{10}=1$
R <sub>202</sub>	1	1	1	1	1	[[ $t_2=0, t_8=0, t_4=0$ ]] [[ $t_6=0, t_1=1$ ]]

$T_5$  に対する出力応答から、Fig. 3 の各信号線の単一故障状態を調べれば Table 6 のようになる。Table 6 の [[ ]] は、単一故障の意味から、その中のどれか一つが示された値に縮退することを表わす。全て単一故障が  $T_1, \dots, T_5$  に対する出力応答から調べられ、多重の故障もこの出力応答によって検出できる。一方、セルの故障診断(セル内での故障箇所の決定)は  $T_1, \dots, T_5$  だけでは行なえないが、セル配列がLSI等で構成されるとき故障箇所を修理して使用するのには現実に困難であるから、ここで述べた故障検出のみで十分であるといえる。

4.2 水平論理回路の故障検出

垂直論理回路の場合と同様に故障検査表を作ると Table 7 (次頁参照)が得られる。 $G_9, G_{10}$  の入力線での0縮退(出力線での1縮退)故障の検出には、

$$((u_3, u_7)), ((u_4, u_6))$$

が必要であり、入力線の1縮退(出力線の0縮退)故障の検出には  $u_2, u_5$  が必要である。簡潔な検査入力ベクトルの組合せを求めると Table 8 (次頁参照)が得られる。

Table 8 の  $U_1, U_2$  について回路出力が  $\phi, \phi$  と

**Table 7** Fault-testing table for horizontal logical circuit.

検査入力 インデックス	検査入力ベクトル			ゲート $G_9$		ゲート $G_{10}$	
	$x$	$y$	$S_4$	$y$ $I_{23}$	$S_4$ $I_{22}$	$x$ $I_{23}$	$S_4$ $I_{24}$
0 縮退検査	$u_3$	0	1	1	*	*	
	$u_4$	1	0	0			*
	$u_6$	1	1	0			*
	$u_7$	1	1	1	*	*	
1 縮退検査	$u_0$	0	0	0			*
	$u_1$	0	0	1	*		
	$u_2$	0	1	0		*	
	$u_5$	1	0	1	*		*

**Table 8** Combination of detecting input vectors.

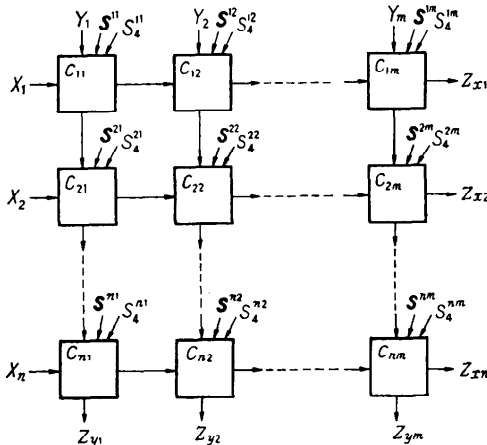
入力ベクトルの組	$x$	$y$	$S_4$	正常出力
$U_1=(u_7, u_4)$	1	$\phi$	1	$\phi$
$U_2=(u_6, u_2)$	$\phi$	1	0	$\phi$

るとき回路は正常であり、それ以外の出力のとき回路に故障があることを示す。Table 6 と同様に、 $U_1, U_2$  に対する出力応答から故障状態を調べることができる<sup>7)</sup>。

**5. セル配列での故障検出**

まず、つぎの条件を設定する。

- a. セル配列は Fig. 9 のように表わされる。セル相互間の接続線は内部接続線であると考え、検査は配列の外部入力端子  $X_i, Y_j$ 、制御入力端子  $S^{ij}=(S_3^{ij}, S_2^{ij}, S_1^{ij})$ 、 $S_4^{ij}$  および外部出力端子  $Z_{xi}, Z_{yj}$  を用いて行なう。ここで  $i=1, 2, \dots, n, j=1, 2, \dots, m$ 、である。



**Fig. 9** Cellular array.

- b. 故障の対象はセルの各信号線の 0, 1 縮退故障とし、各セルの垂直、水平の論理回路を別々に検査する。

- c. 各セルの垂直論理回路については  $i=1$  から  $n$  へ、水平論理回路については  $j=1$  から  $m$  へと順次検査する。

**5.1 垂直、水平両方向の活性化**

すべてのセルの垂直インデックス  $v$  を 2 に設定し、外部垂直入力  $Y_j(j=1, 2, \dots, m)$  を 0, 1 に変化させ、外部垂直出力  $Z_{vj}$  が 0, 1 に変化するか調べる。もし変化しない列があれば、その列内のセルに故障が存在する。

すべてのセルの水平インデックス  $h$  を 0 に設定し、外部水平入力  $X_i(i=1, 2, \dots, n)$  を 0, 1 に変化させ、外部水平出力  $Z_{xi}$  が 0, 1 に変化するか調べる。もし変化しない行があれば、その行内のセルに故障が存在する。

これにより故障が検出されない行、列について、つぎのように詳細な故障検出を行う。

**5.2 垂直方向の故障検出**

**5.2.1 入力の設定**

$i(1 \leq i \leq n)$  行のセルについて垂直論理回路の故障を検出するためには、Table 5 に示した検査入力ベクトルの組  $T_1, \dots, T_5$  を各セル  $C_{ij}(j=1, \dots, m)$  に供給しなければならない。 $T_k(k=1, \dots, 5)$  は入力  $x, y, S_3, S_2, S_1$  の組合せであり、制御入力  $S_3, S_2, S_1$  には外部制御入力  $S^{ij}=(S_3^{ij}, S_2^{ij}, S_1^{ij})$  を利用できる。一方  $x$  については  $j=1$  を除き外部入力  $X_i$  を直接利用できず、 $y$  については  $i=1$  を除き外部入力  $Y_j$  を直接利用できない。そこでセル  $C_{ij}(j=1, \dots, m)$  の水平インデックス  $h$  を 0 にしておき、 $X_i=x$  の外部入力を与える。 $y$  については、 $1, \dots, i-1$  の各行のセルについて垂直インデックス  $v$  を 2 にしておき、 $Y_j=y$  の外部入力を与える。セル  $C_{ij}(j=1, \dots, m)$  の垂直出力を下端の外部出力  $Z_{vj}$  まで伝搬させるために、 $i+1 \leq l \leq n$  である全ての  $l$  について  $C_{lj}$  の垂直インデックス  $v$  を 2 に設定する。

**5.2.2 故障検出**

上の入力設定のもとに、 $i=1$  から  $n$  へ向ってセル  $C_{ij}(j=1, \dots, m)$  の故障検出を行う。 $C_{ij}$  に検査入力ベクトルの組  $T_1, \dots, T_5$  を供給するとき、 $Z_{vj}$  での出力応答が  $\phi, \dots, \phi$  ならば  $C_{ij}$  の垂直論理回路は正常である。これ以外の出力応答ならば  $C_{ij}$  に故障が存在する。

### 5.3 水平方向の故障検出

5.1 で故障検出されない行および 5.1 と 5.2 で故障検出されない列について、各セルの水平論理回路の故障検出を行う。  $j=1$  から  $m$  へ向って順次検査を行うが、  $j$  列のセル  $C_{ij}(i=1, \dots, n)$  に検査入力ベクトルの組  $U_1, U_2$  を供給するために、  $j-1$  以下の各列のセルについて水平インデックス  $h$  を 0 にして  $X_i = x$  の外部水平入力を供給し、  $C_{ij}(i=1, \dots, n)$  の垂直インデックス  $v$  を 2 に設定して  $Y_j = y$  の外部垂直入力を供給する。  $S_4$  には外部制御入力  $S_4^{ij}$  を用いる。  $C_{ij}(i=1, \dots, n)$  の水平出力を外部水平出力  $Z_{xi}$  まで伝搬させるために、  $j+1$  以上の各列のセルについて水平インデックス  $h$  を 0 にする。  $U_1, U_2$  に対する  $Z_{xi}$  の出力応答が  $\phi, \phi$  ならば  $C_{ij}$  の水平論理回路は正常であり、これ以外の出力応答の時  $C_{ij}$  に故障がある。

## 6. セル配列の故障補償

垂直論理回路に故障のあるセルを含む列（故障列）および水平論理回路に故障のあるセルを含む行（故障行）が見い出された場合、故障列の機能の一つ右の列におきかえ、故障行の機能の一つ下の行におきかえる。すなわち故障列の各セルについて水平インデックス  $h$  を 0 にすることにより故障列の各セルを水平方向のブスにし、一つ右の列の各セルについてその垂直インデックス  $v$  を故障列の各セル本来の垂直インデックス  $v$  におく。そして一つ右の列の垂直入力を故障列本来の垂直入力にする。これに応じて故障列より右の列の機能の一つ右へ順次おきかえる。

故障行の各セルについて垂直インデックス  $v$  を 2 にして垂直方向のブスにし、一つ下の行の各セルについてはその垂直、水平インデックス  $v, h$  を故障行のセル本来のインデックスに等しくする。そして水平入力も一つ下の行へおきかえる。同様に故障行より下の行の機能の一つ下へと順次おきかえる。

## 7. むすび

カルノー図からセルの垂直、水平論理回路の最簡略表現を求め、これに対する NAND セルを提案した。

このセルの垂直出力には、3 個の制御入力を用いて、任意関数実現に必要な十分な 6 個の関数と定数 0, 1 とを実現する。水平出力には 1 個の制御入力によって水平入力  $x$  または垂直入力  $y$  を出力として出すことができる。一つのセルにこのような機能を持せることにより、従来のセル配列で問題であったエッジジャンパやメインおよびコレクタ・アレイという 2 種の配列が不要になり、一つの配列で両方の機能を持せることができるようになった。

セルの故障検出については、従来の方法よりも完全に近い故障検出を目的として、セル内の各信号線の 0, 1 縮退故障を対象にした。最簡略表現から得られた NAND セルであるので、各信号線についての縮退故障の検出は容易であり、垂直論理回路は 5 個、水平論理回路は 2 個の検査入力ベクトルの組によって故障を検出できることを明らかにした。

セル配列での故障検出については、単一セルの故障検出の方法を利用して故障セルを含む行、列を検出する方法を示し、それに対する補償法を明らかにした。これらは、セル配列の LSI 化という目的に適合する有効な方法であると考えられる。

## 参 考 文 献

- 1) 野口, 大泉; “Cellular Logic の研究について” 信学誌, 54, 2, pp. 206~220 (昭 46).
- 2) R. C. Minnick; “Cutpoint Cellular Logic”, IEEE Trans. on EC. EC-13, pp. 685~698 (1964).
- 3) R. C. Minnick; “Cobweb Cellular Arrays”, Proceedings-Fall Joint Computer Conference, pp. 327~341 (1965).
- 4) 山本, 樹下; “故障検査を考慮した組合せ回路の一構成法”, 信学論 (D) 55-D, 1 (昭 47-01).
- 5) 江端, 柴谷, 久津輪; “可変論理回路構成法の一考察”, 信学会電子計算機研究会資料 (昭 50-03).
- 6) Z. Kohavi, D. A. Spires; “Designing Set of Fault Detection Tests for Combinational Logic Circuits”, IEEE Trans on C. C-20, pp. 1463~1469 (1971).
- 7) 柴谷, 久津輪, 江端; “NAND 形セル配列について”, 信学会電子計算機研究会資料 (昭 50-03). (昭和 50 年 3 月 27 日受付)