

## 組合せ論理回路のショート故障検出†

久津輪 敏 郎\*\* 江 端 克 彦†††

論理回路の故障検査では、主に縮退故障が対象にされている。しかしこれで表わせない故障も多い。ショート故障は縮退故障モデルで表わせないものの1つであり、回路部品の製造および使用段階でよく生ずる故障である。この故障検査はこれまでに若干検討され、テスト存在条件や SPOOF による検出手法などが示されているが、計算機処理には余り適していないようである。

そこで著者らは、計算機によって回路のショート故障の検査入力を決めるために、まず組合せ的検出手法を用い、これで検出されない故障に順序的検出手法を適用する方法を提案する。組合せ的手法とは故障挿入時と正常時の回路出力を比較して故障検査表をつくり、検出可能なすべての故障をカバーするより少数の入力ベクトルを決定することである。また順序的手法は入力ベクトルの順序対を用いて検出する方法である。

本論文ではこの方法を詳細に述べ、全体および主要部分の流れ図を示し、具体例について入力データと出力結果を示している。著者らのプログラムはメインおよび2つのサブプログラム合せて約550ステートメントからなり、比較的小規模の回路を対象としているが、大規模回路へも特別な困難なく適用可能である。

### 1. ま え が き

論理回路の故障検査では、主に論理素子の入出力線での縮退故障が対象とされている。縮退故障モデルは取扱いが簡単であり多くの物理的故障を良く表わすので有効であるけれども、これで表わせない故障も多い。

ショート故障は縮退故障モデルで表わせない故障の1つであり、つぎのようなところで生じる。すなわち IC チップ内の交叉している線間の絶縁不良、チップとパッケージ・ピン間の半田づけの際のハンダのブリッジ、プリント基板の印刷ミスによる線の接触、基板素子間の配線半田づけに伴う線の接触などである。このショート故障の検査は Friedman<sup>1)</sup>, William<sup>2)</sup>, Mei<sup>3)</sup> らによって検討され、山田<sup>4)</sup>によってテスト存在条件と SPOOF<sup>5)</sup>\* を用いた検出手法が示された。しかし、文献 1)~3) には具体的な検出方法は示されておらず、4) の SPOOF を用いる方法は、その表現形式から手計算には効果的であるが、計算機処理には余り適していないようである。

本論文では計算機によって回路のショート故障を検出する検査入力を自動的に決定するために、まず組合せ的検出手法を用い、これで検出されない故障について順序的検出手法を適用する方法を提案する<sup>6),7)</sup>。

ここで組合せ的検出手法とは、対象回路にショート故障を挿入した場合の回路出力と正常な回路の出力とを比較して異なるならば今供給している入力の組合せ(入力ベクトル)でこの故障を検出できるとする。そして検出可能なすべての故障を検査するためのより少数の入力ベクトルを決定することである。順序的検出手法は入力ベクトルの順序対を用いて検出することである。組合せ的手法は順序的手法に比べて簡単である上、大部分のショート故障がこれによって検出される。そこでまず組合せ的手法を用い、これで検出されないものについてのみ順序的手法を適用するのである。

### 2. ショート故障のモデル

回路は  $n$  個の原始入力、 $l$  個の出力をもち  $m$  個のゲートからなるとする。原始入力に番号  $1, 2, \dots, n$  をつけ、回路のゲート  $j_1$  から  $j_2$  への信号線があるとき、 $j_1 < j_2$  となるように各ゲートの出力線に番号  $n+1, \dots, j_1, \dots, j_2, \dots, n+m$  をつける。回路出力の任意の1つを  $z$  で表わし、 $l$  個の出力を区別する必要があるときのみ  $z_i$  などの添字をつける。ショート故障検出においては、1つのゲートから複数個の出力線がある場合でもそれらを区別する必要はなく同一の番号をつけられればよい。

ショート故障をつぎのように定義する。ショート故

† Fault Detections for Bridging Faults in Combinational Circuits by TOSHIRO KUTSUWA (Osaka Institute of Technology) and KATSUHIKO EBATA (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

\* SPOOF とは Structure and Parity-Observing Output Function の略号であり、回路の論理機能と接続関係を同時に表現できるようにするために、各入力変数に添え字として入力から出力までの信号線番号をつけ、否定素子を含む場合には信号線番号にさらに否定記号をつけて表わす論理関数の1つの表現形式である。

障は論理回路中で2つの信号線が偶然に結合されたときにおこり、結合点でワイヤド・アンドまたはワイヤド・オアの動作をすることである。

一般に正論理ならばワイヤド・アンドの動作をし、負論理のときワイヤド・オアの働きをする。なおこの故障モデルは Mei<sup>3)</sup>、山田<sup>4)</sup>らで示された定義と同じであって、2線がショートしたとき2線の中間の論理値をとるような回路素子の場合にはこのモデルを適用することはできない。

### 3. 検査入力決定法の概要および前処理

図1に故障検査入力を決定する手順を示す。大きな項目に分類すると、(1)回路接続情報の読み込み、(2)経路リストの作成、(3)正常な回路の真理値表作成、(4)組合せ的検出手法、(5)順序的検出手法となる。(4)、(5)については次章以下で述べる。

#### 3.1 回路接続情報の読み込み

検査対象となる回路の動作を計算機内で実現するた

めに、つぎのような回路の接続情報を各ゲート単位で読み込む。それはゲートの種類を示す記号(番号)、ゲートへの入力線の番号および出力線の番号である。接続情報を読み込む順序は、今読み込まれるゲートの入力線がすべて原始入力であるかまたは既に読み込んだゲートの出力線からなるような順序とする。この順序に従えば回路の真理値を求める演算が容易になるからである。ゲートを示す記号としては、AND、ORなどの文字列を用いてもよいが、番号化の方がプログラムが簡単になるので、NOT: 1, AND: 2, OR: 3, NAND: 4, NOR: 5の番号をつけている。また各ゲートの入力線数は便宜上6以下としたが、これの拡張は容易である。

#### 3.2 経路リストの作成

2つの信号線間のショート故障を考えると、その2線が同一の入出力経路上にない場合には、2線間に単にワイヤド・ロジックが挿入された組合せ回路として故障検出を考えればよい。同一経路上に2線がある場合にはショート故障によりワイヤド・ロジックが挿入されフィードバック・ループを形成し非同期順序回路になると考えなければならない。それには各経路のリストを知ることが必要である。そこで本文では先に読み込まれている回路接続情報に基づいて、つぎのように経路リストを求める。回路出力を起点にして、信号線番号の大きい方から小さい方へ順番に、接続情報の読み込み順と逆向きに信号線番号を検索する。信号線  $j$  が接続情報の出力側に見出されたならば、そのゲートの入力線の数だけ新しい経路スタックをつくり、 $j$ を含む経路に各入力線を結合して各スタックへ挿入する。これを信号線番号  $j=n+m$  から1まで繰返す。そして最後に各経路スタックの内容を逆に若い番号順にならべるとすべての経路のリストがえられる。

#### 3.3 真理値表の作成

まず正常な回路の出力を求めるために、10進数  $I=0$  について  $n$  桁の2進数に変換し、 $n$ 個の原始入力の値とする。そして論理演算のサブルーチンを用いて接続情報の読み込み順序に従って各信号線の論理値を求め、回路出力を求める。この操作を  $I=0$  から  $2^n-1$  まですべての入力組合せについて行い真理値表を作成する。

### 4. 組合せ的検出手法

#### 4.1 故障挿入および演算

任意の2線間にショート故障を挿入し、これによっ

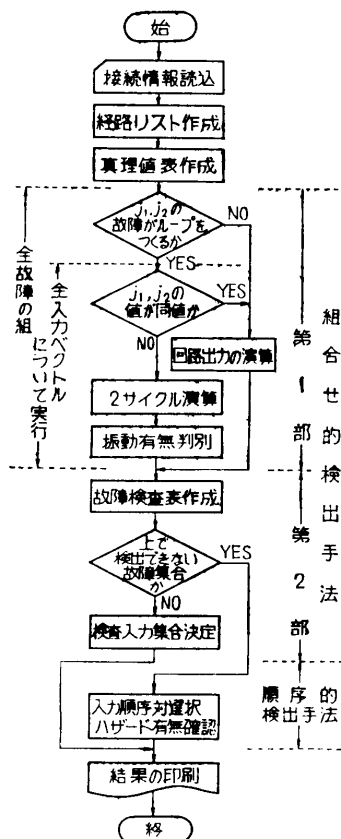


図1 ショート故障に対する検査入力を決定する方法のフローチャート

Fig.1 Flowchart of determination method of testing inputs for bridging faults.

てループが形成されるか否かを判別する。

4.1.1 ループを形成しない場合

これはショート故障する2線が同一経路上にない場合である。2線間にワイヤド・アンドまたはワイヤド・オアのゲートが挿入されたときとみなして回路出力を求める。この出力が正常な回路の出力と異なるならば、今供給している入力ベクトルで検出できる。したがって、まず  $I=0$  について、 $n$ 桁の2進数に変換し  $n$ 個の原始入力に割当て入力ベクトルとし、故障を挿入して接続情報の順序に従って論理演算を行い回路出力を求める。この操作を  $I=2^n-1$  までのすべてについて行う。

4.1.2 ループを形成する場合

これは2線  $j_1, j_2$  が同一経路上にある場合である。供給している入力ベクトルに関してつぎの3つのうちのいずれかの状態がおこる。

- i) 2線の論理値が同じである。
- ii) 論理値の異なる2線がショートして、2線の論理値が一定値(2線のワイヤド・アンドまたはオア)におさまる。
- iii) 論理値の異なる2線がショートしたことにより、ループを通じて論理値の振動がおこる。

i) の場合はショート故障の影響が回路出力に表われないので、今の入力ベクトルでは故障を検出できない。ii), iii) の場合には、ループの論理値の影響が回路出力に到達し、正常な回路出力と異なるならば故障を検出できる。

そこでまず  $I=0$  について  $n$ 桁の2進数に変換し  $n$ 個の原始入力に割り当てる。接続情報の順序に従って論理演算を行う。2線  $j_1$  と  $j_2$  の論理値を調べ等しければ回路出力の値をそのまま確保してこの入力  $I$  の検査を終了する。  $I=I+1$  として同じ検査を繰り返す。

2線の論理値が等しくない場合には、本来の経路による  $j_1$  から  $j_2$  への一方向の信号伝搬と、  $j_1, j_2$  がショートしたことによる両方向の伝搬経路が存在する。これらの論理動作を計算機で実現するためにつぎの2サイクルの論理演算を行い振動の有無を判別する。

いま  $j_1, j_2$  のショート故障が挿入される点すなわちワイヤド・ロジックの結合点を  $j_1', j_2'$  で示す。まず第1サイクルとして、  $j_1$  の論理値を一時確保したのち  $j_1$  と  $j_2$  (正常のときの値) の値を用いて  $j_1'$  にショート故障を挿入し、  $j_1'$  すなわち信号線番号  $j_1$

を入力側にもつ接続情報から順次論理演算を行い、接続情報の出力側に  $j_2$  が現われたときに、前に確保した  $j_1$  の値と  $j_2$  の現在の値を用いて  $j_2'$  に故障を挿入し、これを  $j_2'$  の値として接続情報に従ってさらに論理演算を行い回路出力を求める。

つぎに第2サイクルとして、再度  $j_1$  の値と  $j_2$  の現在の値を用いて  $j_1'$  にショート故障を挿入し、以下第1サイクルと同様に論理演算を行い回路出力を求める。第1サイクルと第2サイクルとで回路出力が異なるならば、ループで振動しそれが回路出力へ伝搬していることを示す。

**定理 1:** 同一経路上の2線のショート故障による振動の有無は2サイクル演算で判別できる。

(証明) 上の2サイクル演算は、  $j_1$  から  $j_2$  への本来の経路による一方向伝搬とショートによる2方向伝搬の論理動作を表わしており、回路は2値論理であって  $j_1$  と  $j_2$  の当初の値が異なっているのであるから、2サイクルで  $j_1', j_2'$  の変化がなければ3サイクル以降に変わることはない。  $j_1', j_2'$  が変化しそれが回路出力まで伝搬することによって判別可能となる。

(証明終)

図2(a), (b)によって2サイクル演算を説明する。ある入力ベクトルを供給して  $j_1$  の値が1となったとする。  $j_2$  の値は正常ならば0であるからそれと  $j_1$  を用いて  $j_1'$  に故障を挿入すると  $j_1'$  は0、これがインバータを通じて  $j_2$  は1、  $j_1$  と  $j_2$  のANDで  $j_2'$  は1となる。これが第1サイクルである。第2サイクルは  $j_1$  と  $j_2$  のANDで  $j_1'$  が1、インバータを通じて  $j_2$  は0、  $j_1$  と  $j_2$  のANDで  $j_2'$  は0となる。なお、さらに1サイクル行えば第1サイクルと同様  $j_1'=0, j_2=1, j_2'=1$  となり  $j_2$  の出力点  $j_2'$  は1, 0, 1, ...と振動することがわかる。ただし  $j_1, j_2$  間に存在するゲートの数(これを  $j_1, j_2$  間の距離と呼

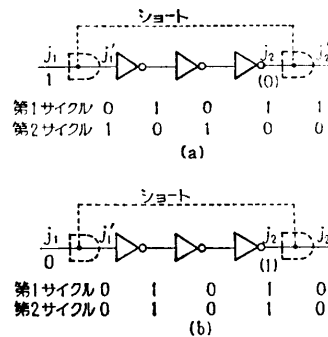


図2 2サイクル演算の説明  
Fig. 2 Illustration of the 2 cycle operations.

ぶことにする)が1の場合は伝搬遅延時間が短いため振動しないことがある。

他方, 別の入力ベクトルによって図2(b)のように  $j_1$  が0であったとする.  $j_2$  は正常ならば1, これから  $j_1'$  は0,  $j_2'$  は1,  $j_2'$  は  $j_1$ ,  $j_2$  から0, これが第1サイクルである. 第2サイクルも  $j_1'=0$ ,  $j_2=1$ ,  $j_2'=0$  となり, この場合は  $j_2'$  は一定値0におさまることがわかる。

以上, 簡単なインバータについて説明したが, 各種のゲートが組合せられた場合にもこの考え方を適用でき,  $j_1$  と  $j_2$  の間にゲートが何段もあるような複雑な場合にも同様に適用できる。

この4.1節が図1の流れ図における組合せ的検出手法第1部に対応する. この部分のより詳細な流れを図3に示す。

#### 4.2 故障の判定

各入力ベクトルについて正常な回路の出力と故障挿入時の出力とを比較する(排他的論理和をとる). この操作をすべての故障およびすべての入力ベクトルについて行い, 検出の可否を示す故障検査表をつくる. すなわち列方向に入力ベクトルを, 行方向に各ショート故障をとる. いま1つのショート故障がある入力ベクトルで検出可能ならば対応する位置に1を入れ, 検出されないならば0を, その入力ベクトルで振動を起すならば2を入れる(ただし  $j_1, j_2$  間の距離が1の場合は0を入れる). 要素が0ばかりからなる行の故障はどの入力ベクトルによっても検出できないことを示しており, これについては5章の順序的手法を利用できないかどうか調べる必要がある。

1をより多くもつ列の入力ベクトルほど検出能力が高いことを示す. 振動の2を検出可能な1と同等に扱うか否かは, 実際に回路出力点につながる観測器(例えばオシロスコープ, デジタルメモリなど)の観測能力やサンプル速度とも関係するので, それとの関連で判断する方がよい. 著者らのプログラムでは2も1と同様に扱っているが, 検出不能の0と同等に扱うための変更は容易である。

1と2を合せた検出能力の最も高い入力ベクトルを選択し, これでカバーされる故障を消去する. 残った故障について順次同様の操作を行えば, 組合せ的検出手法で検出可能なすべてのショート故障をカバーするための, より少数の入力ベクトル集合を決定できる。

なお, すべての故障をカバーする入力ベクトル集合のあらゆる組合せを求めて, その中で最少のベクトル集合を探すこともできないわけではないが, 故障の数, 入力ベクトルの数が多いときにはほう大な手数を要する. この点において上の方法は, 最少ではないがそれに近い集合が求まる実用的な方法である。

### 5. 順序的検出手法

#### 5.1 入力ベクトル順序対の選択

前章の組合せ的検出手法で検出されない故障について, 入力ベクトルの順序対による順序的検出手法を利用できないかどうかを調べる. 順序対についてつぎの定理が存在する。

**定理 2<sup>0</sup>**: 回路において入力ベクトル  $X_k^d$  が経路  $P_k$  の活性化条件で, かつ  $X_k^0, X_k^1$  間の入力変化でハザードが存在しなければ, 入力ベクトルの順序対  $X_k^0 \rightarrow X_k^1$  および  $X_k^1 \rightarrow X_k^0$  によって経路  $P_k$  上の任意の2線の単一ショート故障を検出できる。

ここで  $X_k^d$  は  $k$  番目の座標の値(原始入力  $k$  の値)が don't care であるような入力ベクトルであり,  $X_k^0, X_k^1$  は  $X_k^d$  の  $k$  番目の座標の値をそれぞれ 0, 1 とおいたものである.  $X_k^0 \rightarrow X_k^1$  か  $X_k^1 \rightarrow X_k^0$  かの選択はショート故障する  $j_1(j_1 < j_2)$  の値が  $0 \rightarrow 1$  となる方とする. なお,  $P_k$  上の任意の2線  $j_1, j_2$  の間に否定素子(NOT, NAND, NOR)が奇数個存在する場合には4章の組合せ的手法で検出できるので問題はないが, 否定素子が0または偶数個存在するとき4章の手法で検出できないことがあるので, それについてのみにこれを用いるのである。

これからつぎのことがいえる. 原始入力  $k(1 \leq k \leq n)$  からどれか1つの回路出力  $z$  に至る経路  $P_k(k, \dots, j_1, \dots, j_2, \dots, z)$  の活性化条件をもつ入力ベクトルとは,  $k$  以外の原始入力を固定して  $k$  のみを 0, 1 に変化させるとき, 経路内の各信号線の値が  $0 \rightarrow 1$  または  $1 \rightarrow 0$  に変化する入力ベクトルである.  $n$  個の原始入力がある回路の真理値表において活性化条件をもつ入力ベクトルとは, 真理値表\*の上から  $I$  番目の入力ベクトル(この  $k$  番目の座標の値が0であるもの)と,  $2^{n-I} + I$  番目の入力ベクトル(この  $k$  番目の座標の値は1である)とで経路内のすべての信号線の論理値が 0, 1 に変化するような入力ベクトルである。

したがって  $P_k$  内の2線  $j_1, j_2$  間のショート故障検出のための順序対を決定するには, 上の条件を満たす  $I$  番,  $2^{n-I} + I$  番の入力ベクトルをみつけ,  $j_1$  の値

\* 入力線番号  $1, 2, \dots, n$  は左から順に並んでいるとする。

が 0→1 になる順序対を選択する。そしてこの順序対によってハザードを生じないことを確認すればよいことになる。

5.2 順序対にもとづくハザード有無の確認

5.1 節で選択された順序対についてハザードの有無を判定する。ハザードには関数ハザードと論理ハザード<sup>\*)</sup>があるが、そのうち回路構成に起因する論理ハザードの方のみを調べればよい。論理ハザードをさらに静的ハザードと動的ハザードに分けて考える\*。静的ハザードについてはつぎの定理がある。

**定理 3<sup>\*)</sup>**: 回路が入力  $k$  の変化による入力状態の遷移で静的ハザードをもつ場合、回路は入力  $k$  から回路出力  $z$  への  $P$  経路、 $N$  経路をそれぞれ 1 つ以上もつ。

ここで  $P$  経路、 $N$  経路とは、経路中に含まれる否定素子の数が 0 または偶数個の経路、奇数個の経路である。 $P$  経路と  $N$  経路とで  $k$  の変化が伝わる速度がちがうことによって静的ハザードが起るのである。

これからつぎのことがいえる。

**定理 4**: 経路  $P_i(k, \dots, j, \dots, z)$  について  $k$  から回路出力  $z$  への経路活性化条件をもつ入力ベクトル順序対は、回路に静的ハザードを生じない。

(証明) いま  $k$  から  $j$  ( $k < j \leq z$ ) の間に  $P$  経路と  $N$  経路があり、 $P$  経路を上を活性化された経路とする。このときハザードを生ずるためには  $N$  経路も活性化 ( $k$  の変化が伝わる) されなければならないが、そうであれば  $k$  を 0, 1 に変化するとき  $j$  において  $P$ ,  $N$  経路は互いに相反する値をとり、 $j$  およびそれ以後の線には  $k$  の変化が伝わらない。すなわち  $P$  経路が活性化されているという仮定に反する。したがって  $N$  経路 (活性化している) は存在せず、静的ハザードは生じない。逆に  $N$  経路を上を活性化経路と仮定する場合も同様にいえる。 (証明終)

動的ハザードについて、つぎの定理がある。

**定理 5<sup>\*)</sup>**: 回路が入力  $k$  の変化による入力状態の遷移で動的ハザードをもつ場合、回路は  $P$  経路、 $N$  経路をそれぞれ 1 つ以上もち、 $P-D$  経路または  $N-D$  経路のどちらか片方のみを 1 つ以上もつ。

ここで  $P-D$ ,  $N-D$  経路とは、経路中に否定素子が 0 または偶数個、奇数個ある  $D$  経路\*\* であり、 $D$  経路は原始入力  $k$  から出力  $z$  への活性化経路に相当す

\* 静的ハザードは回路の素子が安定であるときの出力値が入力変化の前後で同じであり、動的ハザードは出力値が入力変化の前後で異なるハザードである。

\*\* 入力  $k$  の加わる入力端子から回路の出力端子まで  $k$  の値にかかわらず  $k$  または  $\bar{k}$  を伝達する経路を  $D$  経路という。

る。

この定理からつぎのことがいえる。すなわちこの定理は回路の出力点での判定について示されているが、回路出力  $z$  を経路中の任意の点  $j$  で置換えれば、経路中の任意の点での判定にも利用できるのは明らかである。この性質から  $j$  点で動的ハザードを生ずるときは、 $k$  から  $j$  間に  $P-D$  または  $N-D$  の活性化経路が存在し、ゲート  $j$  への入力点で静的ハザードを生ずる経路 ( $P$  と  $N$ ) が存在することである。

またハザードのループへの影響については、つぎのことがいえる。経路  $(k, \dots, j_1, \dots, j_2, \dots, z)$  中の  $j_1, j_2$  がショートしてループを形成するとき、 $k, j_2$  間のハザードはループの動作に影響を与える可能性をもつが、 $j_2, z$  間 ( $j_2$  を含まない) のハザードはループの動作に影響を与えない。ゆえに 5.1 節で選択した順序対が  $j_2$  より前で動的ハザードを生じないかを上の性質を用いて調べる。もし生ずる可能性をもつならば再度 5.1 節へ戻り新しく順序対を選択しなおす。

6. プログラムとその結果

プログラムはメインプログラム (ステートメント数 351) と論理演算 OPERAT (同 50), 順序的検出手法 JUNJO (同 154) のサブプログラムからなる。図 3 にはメインプログラム中の組合せ的検出手法第 1 部の部分の流れ図を示し、図 4 は論理演算 OPERAT を示す。

図 5 (b) は (a) の回路についての入力データを示

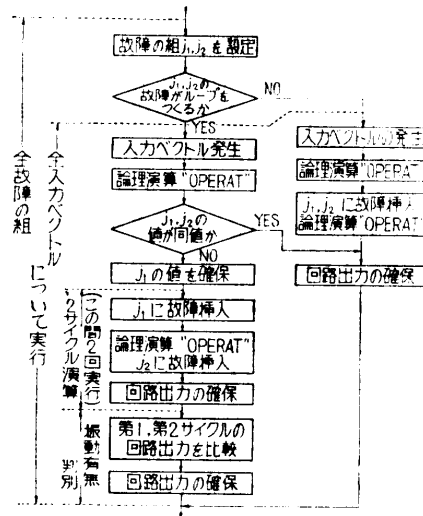


図 3 組合せ的検出手法第 1 部のフローチャート  
Fig. 3 Flowchart of combinational detecting procedure part 1.

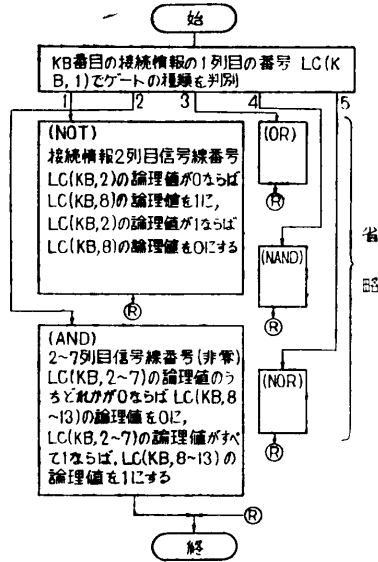


図 4 サブプログラム“OPERAT”の動作

Fig. 4 Operation of subprogram “OPERAT”.

す。第1行は原始入力数  $N$ , ゲート数  $JG$ , 信号線数  $JJ$  である。第2行から第10行までが接続情報であり, その第1列はゲートの種類を示す番号, 2~7列が入力線番号, 8~13列が出力線番号を示す。第11行は回路出力線の番号を示す。

演算の結果は, i) 経路リスト, ii) 真理値表, iii) 故障表, iv) 故障検査表, v) 組合せ的手法で検出されない信号線組合せ番号, vi) 順序的検出手法の結果 (選択した入力順序対など), vii) 組合せ的検出手法による入力ベクトルとそれによる回路出力およびこれらで検出される信号線組合せ番号, の順に出力される。図5(c)にはv), vi), vii)のみを示す。すなわち組合せ的手法で検出されない信号線組合せ番号は31, 44, ...である。これらに順序的手法を適用すると, 31すなわち  $j_1, j_2=3, 11$  のショート故障の場合は, 入力順序対13番→15番が検査入力であり回路出力は正常のとき1→0であり故障ならば出力が異なること

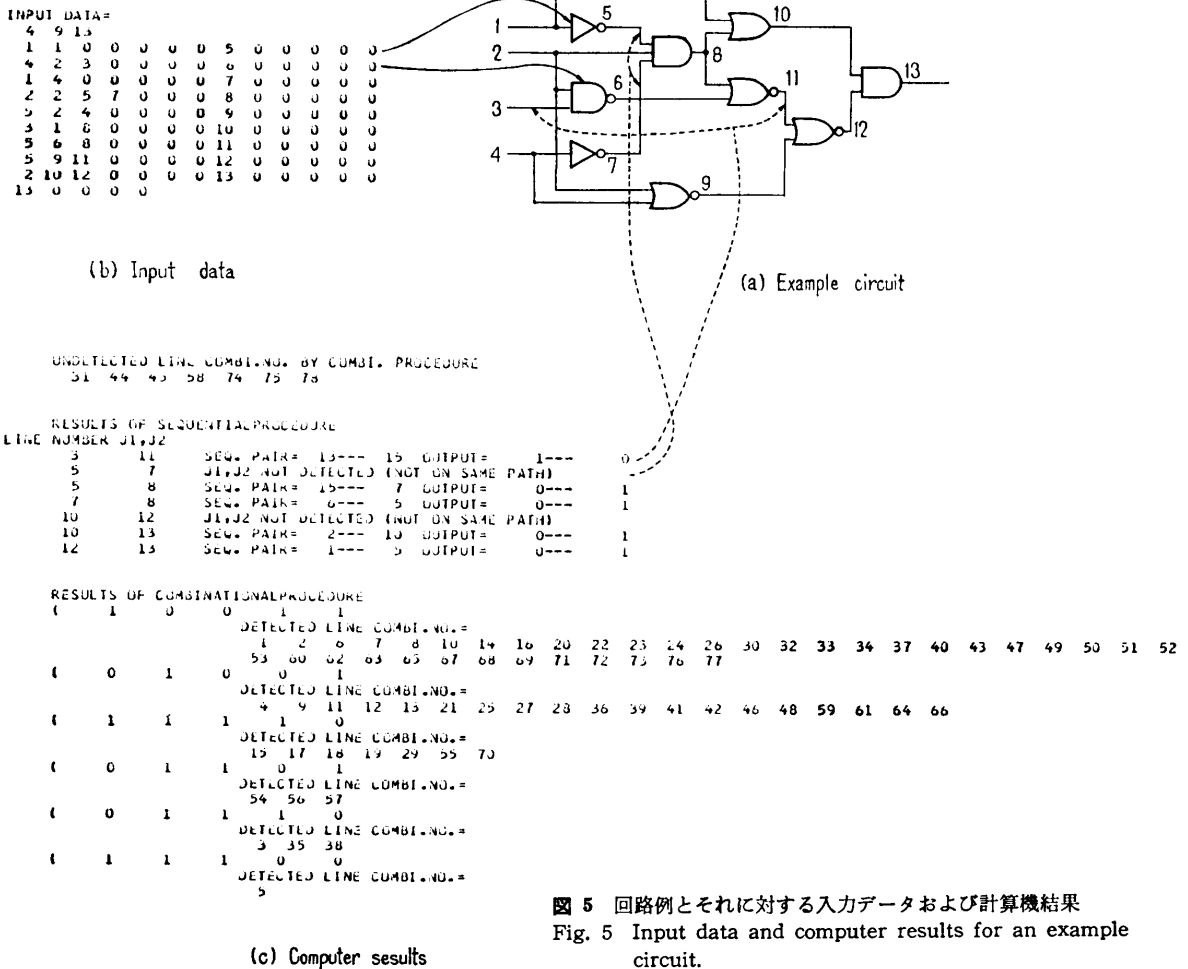


図 5 回路例とそれに対する入力データおよび計算機結果  
Fig. 5 Input data and computer results for an example circuit.

を示す.  $j_1, j_2=5, 7$  は同じ経路上にないため検出されないと打出されている. これは 5, 7 が同じ AND ゲートへの入力であり, ほかに他の入力になっていないから, この 2 線がショートすなわちワイヤド・アンドしてもゲート出力に影響が現われないためである.

組合せ的检测手法の結果は, 最初に選択される入力ベクトルが 1001 であって, 回路出力は正常なら 1, 出力がそうでないときこの入力ベクトルで信号線組合せ番号 1, 2, 6, ... のショート故障を検出できること, また 1001, 0100 など 6 個の入力ベクトルを用いれば組合せ的手法で検出可能なショート故障をすべてカバーできることを示す.

## 7. むすび

ショート故障の検査入力を計算機によって決定する方法, 主要部分の流れ図および実行例を示した. 著者らの作成したプログラムでは原始入力数 6, 回路出力数 5, ゲートの入力線数 6, 信号線数 25, 入出力経路長 10 までの回路を扱うようにしている. 扱える回路を拡張することは使用計算機の記憶容量とも関係するが, プログラムそのものの変更は容易である.

また, このプログラムではショート故障の対象を信号線のすべての組合せとしているが, 実際の使用にあたってはゲート, 信号線の配置上ショートする可能性の強い組合せを読み込み, それについてのみ実行すれば必要な記憶容量や計算時間も大幅に削減することがで

きよう. このための変更もまた容易である.

なお, 使用した計算機は IBM 370-125 (大阪工業大学中央研究所) であり, 先の例での占有記憶容量は 26.96 k Byte, 計算時間は 56 秒であった.

## 参 考 文 献

- 1) Friedman, A. D.: Diagnosis of Short-Circuit Fault in Combinational Circuits, IEEE Trans. Comput., C-23, 7, p. 746 (1974).
- 2) Williams, M. J. and Angell, J. B.: Enhancing Testability of Large-Scale Integrated Circuits via Test Points and Additional Logic, IEEE Trans. Comput. C-22, 1, p. 46 (1973).
- 3) Mei, K. C. Y.: Bridging and Stuck-at Faults, IEEE Trans. Comput., C-23, 7, p. 720 (1974).
- 4) 山田: 組合せ回路における信号線のショート故障のテスト, 信学会計算研資 EC 74-35, p. 45 (1974).
- 5) Clegg, F. W.: Use of SPOOF's in the Analysis of Faulty Logic Networks, IEEE Trans. Comput., C-22, 3, p. 229 (1973).
- 6) 久津論, 江端: 組合せ論理回路のショート故障検出, 昭和 53 年信学会総合全国大会 1425.
- 7) 久津論, 江端: 同上 (第 2 報), 昭和 53 年電気関係学会関西支部大会, G6-9.
- 8) 内藤: 論理式によるハザード検出と非同期回路の解析への応用, 信学会計算機研資 EC 74-7, p. 23 (1974).

(昭和 53 年 10 月 25 日受付)

(昭和 54 年 3 月 15 日採録)