

組合せ回路における 同時検査入力生成アルゴリズム

高松雄三

佐賀大学工学部

樹下行三

広島大学総合科学部

検査系列を効率よく生成することを目的として、同時性(concurrency)という概念に基づいた新しい検査系列生成アルゴリズム(CONTアルゴリズム:CONcurrent Test generation algorithm)を提案する。このCONTアルゴリズムは、(1)目標故障の検査入力を生成する過程で、同時にその検査入力で検出される可能性のある故障を求める、(2)目標故障の検査入力を生成中バックトラックが発生したとき、回路の入力を変えるというバックトラックは行わないで同時に処理を進めている故障を利用して目標故障を変更する、の特徴を有している。したがって、ここで述べるCONTアルゴリズムは、検査入力生成後故障シミュレーションを行うという従来の手法とは異なるものである。

WGDA 36-3 "Concurrent Test Generation Algorithm
for Combinational Logic Circuits", (in Japanese)

by Yuzo TAKAMATSU (Department of Electronic Engineering, Faculty of Science and Engineering, Saga University, 1 Honjo-machi, Saga, 840 Japan) and Kozo KINOSHITA (Department of Information and Behavioral Sciences, Faculty of Integrated Arts and Sciences, Hiroshima University, Hiroshima, 730 Japan)

A new test generation algorithm for combinational circuits, referred to as the CONcurrent Test generation algorithm (CONT algorithm), is presented. The CONT algorithm has following characteristics: (1) It aims at generating a test pattern that can detect faults as many as possible by making fault lists concurrently in the process of test generation. (2) When backtrack occurs in the process of test generation, the primary input value is not changed but a target fault is switched to another one using the fault list. Experimental results show that the CONT algorithm generates test patterns which achieve about 95% fault coverage without changing the primary input value. Some experimental results show that the CONT algorithm generates test patterns faster and more efficient than a fault simulator.

1. まえがき

論理回路のVLSI化に伴って、その検査系列を生成する時間、費用がますます増大する傾向にある。検査対象となる回路は一般に順序回路であり、その検査系列生成は組合せ回路のそれに比べて更に複雑である。現在、これらの問題を解決する一つの方法として実用化されている手法にスキャンバス方式^{[11]・[12]}がある。この方式により回路設計を行うことにより、検査系列生成の問題は組合せ回路のそれに帰着させることができる。したがって、効率の良い組合せ回路の検査系列生成アルゴリズムを開発することは重要な問題である。

組合せ回路の検査系列生成アルゴリズムについては、これまで、多くの手法が提案されている^{[3] - [16]}。これらの検査系列生成アルゴリズムの多くは生成時間の高速化をめざしているが^{[8] - [12]}、検査入力数の縮小化に注目した検査系列生成アルゴリズムも提案されている^{[13] - [15]}。

筆者らは前に、同時性(concurrency)という概念に基づいて、検査系列生成時間の高速化、および検査入力数の縮小化を目的としてCONTアルゴリズム(Concurrent Test generation algorithm)を提案した^[16]。このCONTアルゴリズムには次のような新しい手法を導入した。

(1) 目標故障の検査入力生成中、その検査入力で検出される可能性のある故障を求めながら検査入力を生成する。

(2) 目標故障の検査入力生成中、バックトラックが発生したときは回路の入力を変えないで目標故障を変更し、新しい目標故障の検査入力の生成を試みる。

筆者らは、このCONTアルゴリズムを隣接ビットペアのデータ構造を用いてプログラム化し、文献[16]でその結果を報告した。しかしながら、データ構造として用いた隣接ビットペア表現では、演算結果に未使用の隣接ビットペアが生じること、また、同時に挿入できる故障の数が15個までであること、などから検査入力数は縮小することができたが、検査系列生成時間に問題が残っていた。

そこで、ここではこれらの問題点を改良した新しいCONTシステムを述べる。まず、2章でCONTシステムの概要を述べる。CONTシステムは、2つの検査系列生成モードを有しており、それらの詳細を3、4章で述べる。5章では、CONTシステムの実験結果を示す。その結果、回路の入力を変えるという検査系列生成アルゴリズムのバックトラック^[10]なしで、約95%以上の故障検出率を達成する検査入力を生成することができた。さらに、ある回路に対しては、従来の故障シミュレーションよりも高速に、かつ検査入力数をより少なく生成することができた。

ここで述べるCONTシステムは、検査入力生成と

故障シミュレーションを同時に行おうとするものであり、検査入力生成と故障シミュレーションを別々に行う従来の検査入力生成システムとは異なる手法である。

2. CONTシステム

以下、本文で扱う回路は、AND, OR, NOT, NOR, NAND およびEXCLUSIVE-ORから構成されている組合せ回路である。また、対象とする故障は、単一の縮退故障(s-a-0またはs-a-1)である。CONTシステムで用いる信号値は、Dアルゴリズムと同様の0、1、X(未定義)、Dおよび \bar{D} の5値である。

図1にCONTシステムの全体の構成図を示す。図に示すようにCONTシステムは、同時検査入力生成モードTGM-C(Test Generation Mode with Concurrency)と単一目標故障検査入力生成モードTGM-S(Test Generation Mode with a Single target fault)の2つの検査系列生成モードをもっている。

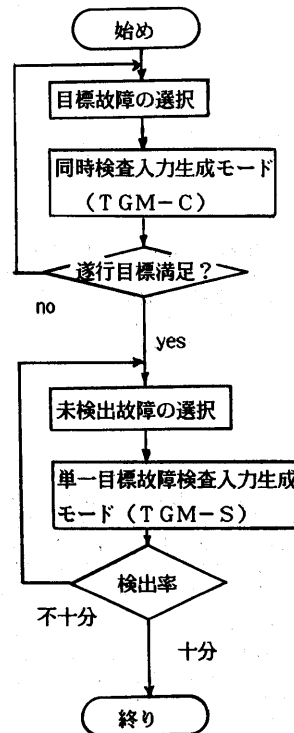


図1 CONTシステムの構成図

TGM-Cでは、目標故障を仮定し、その検査入力を生成する過程で、生成される検査入力で検出される可能性のある故障の故障リストを作成する。この処理の過程でバックトラックが発生したときは、回路の入力を変更するというバックトラックは行わないで、目標故障を変更して検査入力生成アルゴリズムを実行する。

一方、TGM-SはTGM-Cの目標故障を変更するというバックトラックを行わないで、単一の目標故障に対して検査入力を生成するモードである。したがって、TGM-Sのバックトラックは回路の入力を変更するというバックトラックを行う。

TGM-Cでは、目標故障をその検査入力生成の過程で変更するので完全な検査入力生成アルゴリズムではない。したがって、CONTシステムではTGM-Cをある時点でTGM-Sに切り換え、完全な検査系列生成アルゴリズムとして構成する。このTGM-CからTGM-Sへの切り換えは、システムの遂行目標により行う。遂行目標としては、TGM-Cを実行した目標故障の数、故障検出率、そのとき同時に検出された故障の数などであり、これらは、使用者が決めることができる。

3. 同時検査入力生成モード (TGM-C)

3.1 TGM-Cのアルゴリズム

TGM-CはPODEMアルゴリズム^[10]と同様の経路活性化法に基づく検査系列生成アルゴリズムである。したがって、以下の操作を基本としている。

(1) 含意操作

決められた回路の入力信号値により、一意に決められる回路の信号線の信号値を決定する操作である。

(2) バックトラック

現在決められている回路の入力信号値で目標故障の検査入力が生成できないとき、他の選択を行う操作である。

(3) 後方追跡

ある初期目標 (目標信号線と目標信号値) を達成する回路の入力信号値を決める操作である。

TGM-Cではこれらの基本操作に以下のような新しい操作を加えている。

(1) 含意操作中にゲートの故障リストを作成する操作を行う。ここで故障リストとは、そのゲートの出力信号線で検出できる故障のリストであり、同時シミュレーション^[17]のそれとほぼ同様である。この操作を故障リストの作成操作という。

(2) バックトラックが発生したとき、回路の入力の信号値を変更するというバックトラックは行わないで、目標故障を変更する。この操作を目標故障の変更操作という。

(3) 後方追跡における初期目標は、PODEMでは目標故障線、D先端 (Dフロンティア^[6]) であるが、TGM-Cではバックトラックが発生したとき、初期目標として故障リストを用いる。

これらの他にTGM-Cには、次の操作も導入している。

(4) 故障の収集操作

生成された検査入力で検出される故障を求める操作である。これは含意操作の過程で同時に作成した故障リストを用いて行う。

図2にTGM-Cの流れ図を示している。このTGM-Cによる検査入力生成の概要は次のとおりである。

まず、目標故障を仮定し、その故障信号 (Dまたは \bar{D}) を回路の出力へ伝播するような回路の入力信号値を後方追跡により決める。次に、その回路の入力信号値により含意操作を行いながら故障リストの作成操作を行う。この後方追跡、含意操作をDまたは \bar{D} が回路の出力線に伝播するまで繰り返す。この操作の過程で

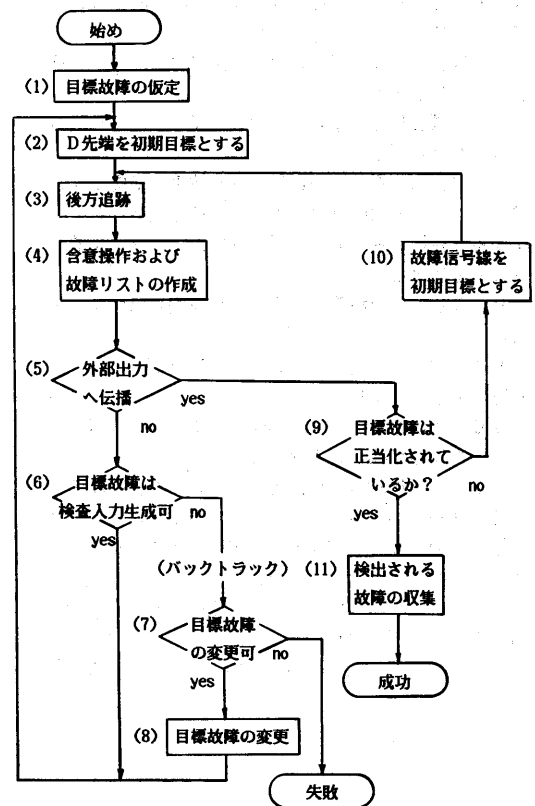


図2 TGM-Cの流れ

バックトラックが発生したときは目標故障を変更する。こうして、目標故障の信号値Dまたは \bar{D} が回路の出力に伝播した後、回路の出力ゲートの故障リストにより、生成した検査入力で検出される故障を収集する。

これまでの準備によりTGM-Cのアルゴリズムは次のように記述される。

TGM-Cのアルゴリズム：

1：検査入力を生成する故障信号線、故障値を仮定し、信号値D (s-a-0) または \bar{D} (s-a-1) を設定する。以下、これを目標故障と呼ぶ。

2：一つのD先端を選び、それを伝播するような初期目標を決める。

3：初期目標を達成するような回路の入力信号値を後方追跡により決定する。

4：後方追跡で割り当てられた入力信号値により、含意操作を行う。この操作の過程で各ゲートに対する故障リストを作成する。ここで、故障リストはそのゲートの入力線側のすべてのゲートの故障リストが作成されているときのみ作る（故障リストの作成については、3.2節(A)で述べる）。

5：Dまたは \bar{D} が回路の出力に伝播すれば9へ。そうでなければ6へ。

6：目標故障の検査入力が生成可能ならば2へ戻る。そうでなければ7へ。

7：現在の目標故障が新しい目標故障に変更できれば8へ。そうでなければ、TGM-Cは失敗として終了する（目標故障の変更アルゴリズムは、3.2節(B)で述べる）。

8：新しい目標故障にDまたは \bar{D} を割り付け2へ戻る。

9：目標故障の信号値が確定していれば11へ。そうでないときは10へ。

10：目標故障の信号値が確定していない信号線、信号値を初期目標として3へ戻る。

11：目標故障の検査入力が生成された。回路のすべてのゲートの故障リストを完成するため、生成された検査入力のうち未定義となっている入力に0または1をランダムに割り付け含意操作を行う。この後、回路の出力ゲートの故障リストから生成した検査入力で検出できる故障を収集する。

[TGM-Cのアルゴリズム終了]

3.2 TGM-Cにおける基本操作

(A) 故障リストの作成と伝播

これまで述べたようにTGM-Cでは含意操作の過程でゲートに対して故障リストを作成する。故障リストは、そのゲートの入力側のすべてのゲートの故障リストが生成されているときのみ作成する。図3にTGM-Cにおける故障リストの作成および伝播のようすを示す。ここで、回路の信号A=B=1およびC=0はすでに決まっているとする。また、故障はそれぞれの信号線のs-a-0およびs-a-1を考える。

そうすると、以下のようにしてゲートGiの故障リストが作成できる。信号線AおよびBの故障はゲートGiの出力線Eで検出できるので、これらの故障をLiに登録する。ここで、ゲートGiの故障リストをLiで表し、また、Liの意味は次のとおりである。

Li		
	(s-a-0)	
A	0	故障Aが発生したときGiの出力は0
B	0	故障Bが発生したときGiの出力は0
	(s-a-0)	

こうして、図3に示すようなLiの故障リストが作成される。

次に、Gjの故障リストLjを作成する。信号線Eの故障および信号線A、Bの故障はゲートGjの出力線Fで検出できるので、これらの故障をLjに加えて図3に示すようなLjを作る。このように故障を次のゲートの故障リストに加えて行くことを故障リストの伝播と呼ぶ。こうして、故障リストを作成、伝播して行くことにより、目標故障の検査入力が生成されたとき、その検査入力で検出できる故障を回路の出力ゲートの故障リストから収集することができる。

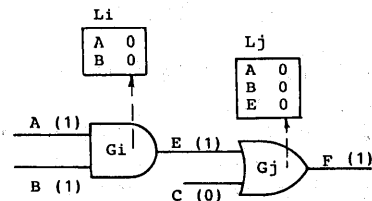


図3 故障リストの作成と伝播

(B) 目標故障の変更

目標故障の検査入力生成の過程でバックトラックが発生したとき、TGM-Cではその目標故障を次のアルゴリズムによって新しい目標故障に変更する。

目標故障の変更アルゴリズム：

S1：これまで設定されている信号値Dを1に \bar{D} を0に設定し、合意操作をする。

S2：次の条件を満足するゲートG_iを求める。

(i)ゲートG_iは故障リストが作成されている、かつ、

(ii)ゲートG_iに続くゲートの故障リストは作成されていない。

このようなゲートG_iがなければ、目標故障の変更は失敗とする。

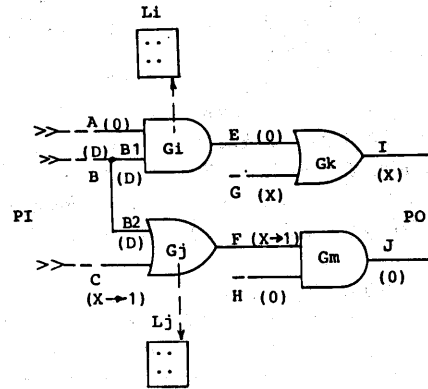
S3：S2で求めたゲートの中から、そのゲートの故障リストが回路の出力まで伝播できるようなゲートG_jを選びS4へ。このようなゲートG_jがなければ、目標故障の変更は失敗とする。

S4：ゲートG_jの出力線を目標故障とし、信号値D(G_jの出力値が1のとき)または \bar{D} (G_jの出力値が0のとき)を割り付ける。

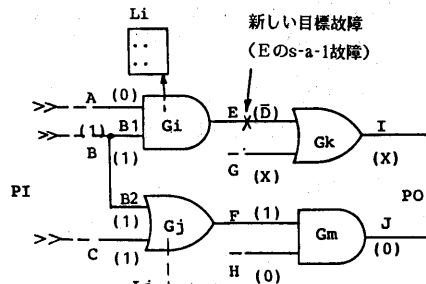
【目標故障の変更アルゴリズム終了】

以上の方法により、得られた新しい目標故障に対してTGM-Cは検査入力生成アルゴリズムの実行を続ける。

図4に目標故障の変更アルゴリズムの実行例を示そう。いま、図4(a)に示すような信号値がすでに設定されているものとする。このとき、検査入力生成の過程で信号線Cの値が1に設定されたとしよう。そうすると、D先端がなくなり、バックトラックが生じる。そこで、前述の目標故障変更アルゴリズムにより、目標故障を変更することを試みる。S1により、Dを1に設定し合意操作を行うと図4(b)のようになる。S2からゲートG_iとG_jを選ぶ。G_jから回路の出力Jへの経路は、H=0であるからL_jを回路の出力Jへ伝播できない。しかし、G_iから回路の出力Iへの経路は、L_iを回路の出力Iへ伝播できる可能性がある。したがって、S3によりG_iを選ぶ。S4からG_iの出力線を新しい目標故障とし、信号線Eに \bar{D} を設定する。このときの回路の信号値および新しい目標故障を図4(b)に示している。



(a)



(b)

図4 目標故障の変更

(C) 検出される故障の収集

検査入力が生成された後、その検査入力で検出できる故障を収集する。このとき、回路のすべてのゲートの故障リストを完成するため、生成された検査入力の中で未定義の入力については、ランダムに0または1を割り付け合意操作を行う。この後、回路の出力ゲートの故障リストから、その検査入力で検出される故障を収集する。

3.3 TGM-Cの実行例

図5の簡単な例を用いてTGM-Cのアルゴリズムを説明しよう。ここで、故障は回路のそれぞれの信号線の故障s-a-0およびs-a-1である。

1：目標故障を信号線Bのs-a-0とする。

2: D先端の集合 {B1, B2} からB1を選ぶ。
 3: B1のDを伝播するため、Eから後方追跡により入力X1を1に決める。

4: X1=1, X2=Dにより含意操作を行う。A=1, B1=DからL1を作成する。

2から6の繰返し: D先端のE2を選び、それを伝播する初期目標Iから後方追跡を行い、X3=0を決める。このX3=0により含意操作を行った後の信号値および作成された故障リストを図5(a)に示している。

いま、検査入力生成の過程でG=1が設定されたと仮定しよう。そうすると、D先端がなくなり、バックトラックが生じる。そこで、3.2節(B)の目標故障の変更アルゴリズムにより、目標故障を変更する。S1により、DおよびD̄をそれぞれ、1および0に設定し含意操作を行うと、図5(b)のようになる。S2とS3でG4を選ぶ。なぜなら、L4が作成されており、G4から回路の出力Zへの経路はL4を伝播できる可能性があるからである。そこで、G4の出力線Iのs-a-1故障を新しい目標故障とし、信号線IにD̄を設定する。目標故障の変更に成功したので、TGM-Cの手順2へ戻る。

2から6の手順を繰り返し実行した後、目標故障Iのs-a-1に対して図5(c)で示すような検査入力が生成されたと仮定しよう。

11: 故障リストL5およびL6から、生成された検査入力で検出される故障、すなわち信号線B2のs-a-0および信号線F、I、K、Hのs-a-1が求められる。

4. 単一目標故障検査入力生成モード (TGM-S)

TGM-Cはその検査入力生成の過程で目標故障を変更しているので、検出可能な故障に対して検査入力を生成できるという意味で完全な検査入力生成アルゴリズムではない。したがって、CONTシステムでは、TGM-Cをある時点でTGM-Sに切り換えるという手法を用いている。すなわち、TGM-Sでは、TGM-Cで検査入力が生成されなかった故障に対して検査入力を生成する。したがって、TGM-Sは目標故障は変えないで、単一の目標故障に対して検査入力の生成を試み、バックトラックが発生したときは、回路の入力信号値を変えるというバックトラックを行うことにより検査入力を生成する。

こうして、CONTシステムは全体として完全な検査系列生成アルゴリズムとして構成している。TGM-CからTGM-Sへの切り換えは、システムの遂行目標により行う。遂行目標としては、例えば、TGM

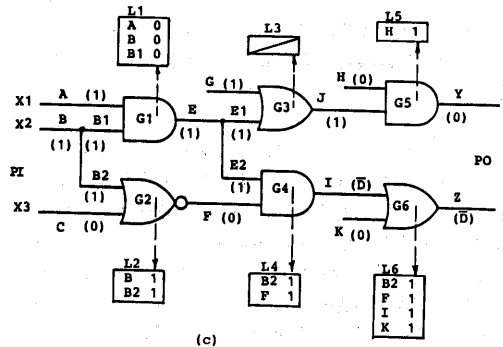
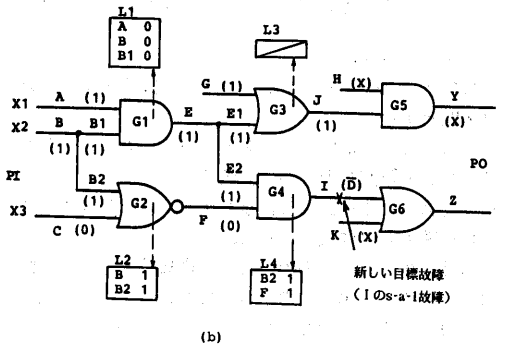
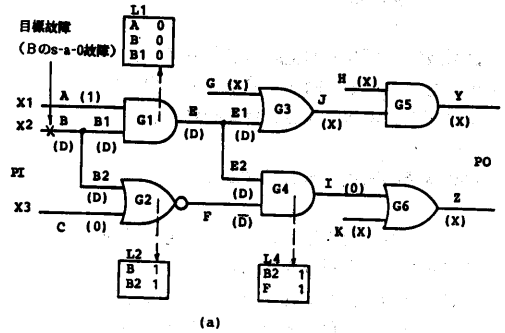


図5 TGM-Cの実行例

-Cを実行した目標故障の数、故障検出率、そのとき同時に検出された故障の数、などである。これらは使用者が決めることができるが、本論文での実験で用いた遂行目標は、TGM-Cの効果を調べるためTGM-Cを実行した目標故障の数とした。

5. 実験結果

これまで述べたCONTシステムをプログラム化して、サンプル回路^[18]に適用してアルゴリズムの有効性を評価した。実験では、遂行目標としてTGM-Cを実行する目標故障の数をういた。すなわち、すべての故障を一度目標故障としてTGM-Cを実行し、その後TGM-Sに切り換え、TGM-Cで未検出となった故障に対して検査入力の生成を行った。

表1には、TGM-CおよびCONTシステム全体の、それぞれの検出率、生成時間(比)、生成された検査入力の数を示している。ここで、生成時間は、回路C432のTGM-Cの計算時間を基準として表している。表1から分るように、TGM-Cのみで回路C1355を除いて約95%以上の故障検出率を達成する検査入力を生成することができた。

また、TGM-Cの有効性を調べるため、故障シミュレーション^[19](FSIMと表す)との比較を行った。その結果を表2に示す。この実験では、TGM-Cと故障シミュレーションで、それぞれ80%の検出率を得るまでの検査系列生成を行い、その生成時間および生成された検査入力の数を求めた。表2から、TGM-Cは、C3540とC6288を除いて故障シミュレーションより高速であり、さらに、生成した検査入力の数がすべての回路に対して少ない。

TGM-Cと故障シミュレーションの検査入力生成の様子を回路C2670について図6に示している。TGM-Cでは、80.2%、90.0%の検出率を得る検査入力を、それぞれ43および116、計算時間3.06秒、7.70秒、で生成した。一方、故障シミュレーションでは、80.1%の検出率を達成する検査入力79を計算時間5.42秒で生成した。しかしながら、故障シミュレーションでは、81.2%以上の検出率を得ることには失敗した。なお、使用計算機はFACOM-M382である。

表1 CONTシステムの実行結果

回路名	検出率(%)		計算時間(比)		検査入力の数	
	TGM-C	全体	TGM-C	全体	TGM-C	全体
C432	99.2	99.2	1.00	1.48	66	66
C499	98.8	98.9	1.50	2.48	69	70
C880	100.0	100.0	2.05	2.05	81	81
C1355	86.2	99.5	11.2	17.5	113	187
C1908	98.5	99.5	8.90	10.4	159	172
C2670	94.6	96.7	13.0	24.1	149	175
C3540	95.0	96.2	24.7	35.5	204	225
C5315	98.7	98.9	24.8	28.8	163	170
C6288	99.5	99.6	20.8	22.3	36	39
C7552	96.2	98.3	62.1	92.0	246	337

表2 TGM-Cと故障シミュレーションとの比較

回路名	計算時間 (比) FSIM/TGM-C	80%の検出率を得る 検査入力の数	
		TGM-C	FSIM
C1908	1.19	51	59
C2670	1.63	48	79
C3540	0.90	77	92
C5315	1.63	30	51
C6288	0.20	5	12
C7552	1.50	38	65

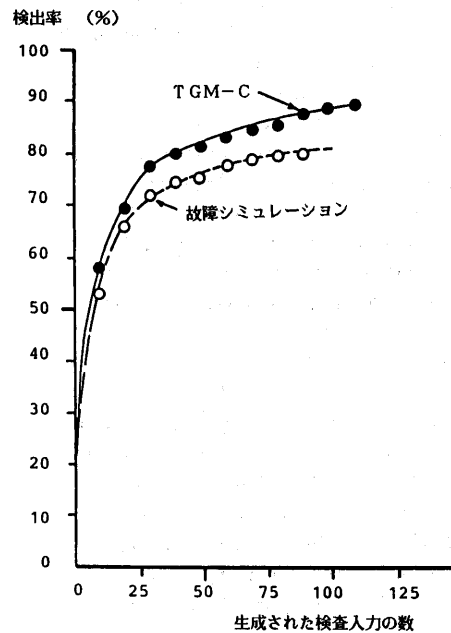


図6 TGM-Cと故障シミュレーションとの比較例(C2670)

6. むすび

本論文では、コンカレント性という概念に基づく新しい検査系列生成システムを提案した。このシステムは、TGM-CとTGM-Sの2つの検査系列生成モードを有している。前者のTGM-Cは、目標故障の検査入力生成の過程でその検査入力により検出可能と思われる故障を求めながら処理を進め、バックトラックの発生に対しては目標故障の変更で対処するモードである。後者のTGM-Sは、一つの目標故障に対して検査入力を生成するモードである。

実験結果で示したように、TGM-Cのみで高い故障検出率を得る検査入力を生成できる。このTGM-Cは、検査入力生成と故障シミュレーションを並行して行っていると考えられる。

本論文で述べたTGM-Cの手法を故障シミュレーションの手法に応用すると、効率のよい故障シミュレータが実現されるものと思われる。現在、これらについて考察中であり別の機会に発表する予定である^[20]。

参 考 文 献

- [1] Eichelberger, E.B. and Williams, T.W.: A Logic Design Structure for LSI Testability, Proc. 14th Design Automation Conf., pp.462-468 (1977).
- [2] Yamada, A., Wakatsuki, N., Fukui, T. and Funatsu, S.: Automatic System Level Test Generation and Fault Location for Large Digital Systems, Proc. 15th Design Automation Conf., pp.347-352 (1978).
- [3] Armstrong, D. B.: On Finding a Nearly Minimal Set of Fault Detection Tests for Combinational Logic Nets, IEEE Trans. Electron. Comput., Vol. EC-15, No. 1, pp.66-73 (1966).
- [4] Sellers, F. F., Hsiao, M. Y. and Bearson, L. W.: Analyzing Errors with the Boolean Difference, IEEE Trans. Comput., Vol. C-17, No. 7, pp.676-683 (1968).
- [5] Kinoshita, K., Takamatsu, Y. and Shibata, M.: Test Generation for Combinational Circuits by Structure Description Functions, Proc. 10th Int. Symp. on Fault-Tolerant Computing, pp.152-154 (1980).
- [6] Roth, J. P.: Diagnosis of Automata Failures: A Calculus and a Method, IBM J. of Res. and Dev., vol.10, No. 4, pp.278-291 (1966).
- [7] Muth, P.: A Nine-valued Circuit Model for Test Generation, IEEE Trans. Comput., Vol. C-25, NO. 6, pp.630-636 (1976).
- [8] Cha, C. W., Donath, W. E. and Ozguner, F.: 9-V Algorithm for Test Pattern Generation of Combinational Digital Circuits, IEEE Trans., Vol. C-27, No. 3, pp.193-200 (1978).
- [9] Murakami, M., Shiraki, N. and Hirakawa, K.: Logic Verification and Test Generation for LSI Circuits, Proc. Int. Test Conf., pp.467-472 (1980).
- [10] Goel, P.: An Implicit Enumeration Algorithm to Generate Tests for Combinational Logic Circuits, IEEE Trans. Comput., Vol. C-30, No. 3, pp.215-222 (1981).
- [11] Fujiwara, H. and Shimono, T.: On the Acceleration of Test Generation Algorithms, IEEE Trans., Comput., Vol. C-32, No. 12, pp.1137-1144 (1983).
- [12] Takamatsu, Y. and Kinoshita, K.: An Efficient Test Generation Method by 10-V Algorithm, Proc. Int. Symp. on Circuits and Systems, pp.679-682 (1985).
- [13] Airapetian, A. N. and McDonald, J. F.: Improved Test Set Generation Algorithm for Combinational Circuit Control, Proc. 9th Int. Sympo. on Fault-Tolerant Computing, pp.133-136 (1979).
- [14] McDonald, J. F. and Benmehrez, C.: Test Set Reduction using the Subscripted D-Algorithm, Proc. Int. Test Conf., pp.115-121 (1983).
- [15] Yau, C.W.: Concurrent Test Generation using AI Techniques, Proc. Int. Test Conf., pp.722-731 (1986).
- [16] 高松、樹下: コンカレント性を用いた検査系列生成アルゴリズム, 情報処理学会論文誌, Vol.27, No.12, pp.1254-1258 (1986).
- [17] Ulrich, E. G. and Baker, T.: The Concurrent Simulation of Nearly Identical Digital Networks, Proc. 10th Design Automation Workshop, pp.145-150 (1973).
- [18] Brglez, F. and Fujiwara, H.: A Neutral Netlist of 10 Combinational Benchmark Circuits and a Target Translator in FORTRAN, Special Session on ATPG and Fault Simulation, Proc. Int. Symp. on Circuits and Systems, pp.663-698 (1985).
- [19] Takamatsu, Y. and Kinoshita, K.: On the Effect of Size of Fault Word in Parallel Fault Simulation, JIP, Vol.9, No.3 (1986).
- [20] 高松、樹下: アルゴリズム型検査入力生成向き故障シミュレータ、信学会フォールトトレラント研資、(1987).