設計自動化　５４－４
(1990.10. 8)

大規模論理回路用
テストパターン自動生成システム

畠山　一実†　　林　照峯†　　甲把　健†　　高倉　正博‡　　野上　和枝‡

†㈱日立製作所　　‡日立エンジニアリング㈱

あらまし　本論文では，トライステート素子，双方向エッジなどを含む大規模な論理回路に対する，実用的なテストパターン自動生成システムについて報告する。高品質なテストパターンを効率良く自動生成するためのアプローチとして，重み付き乱数パターン発生を用いている。試行データを用いた評価により，本アプローチの実用的な有効性を確認する。

A Test Pattern Generator
for
Large Scale Logic Circuits

Kazumi HATAYAMA† Terumine HAYASHI† Takeshi GAPPA†　Masahiro TAKAKURA‡ Kazue NOGAMI‡

† Hitachi, Ltd.　　‡ Hitachi Engineering, Co., Ltd.

Abstract　This paper presents a practical test pattern generator for large scale logic circuits including tri-state logic elements, bi-directional edges, and so on. A weighted random pattern generation approach is used for overcoming some problems, such as, increasing CPU times, decreasing fault coverage, and so on. Experimental results illustrate the effectiveness of our approach.

1．はじめに

論理回路の大規模化が進むにつれて，これを検査するためのテストパターンの設計が急激に困難化するが，これは設計期間を短縮する上で重大な問題となる。この対策としては，テスト容易化設計により，論理回路そのものをテストし易くする方法が良く用いられている。その中でもスキャン設計方式[1，2，3，4]は順序回路のテストパターン生成という非常に難しい問題を組合せ回路のテストパターン生成という比較的容易な問題に変換することができるため広く実用化されている。また，高品質のテストパターンを効率良く生成する手法として多くの組合せ回路用のテストパターン生成手法が提案されている。その中でも，単純な組合せ回路を対象とした手法としては，Dアルゴリズム[5]，PODEM法[6]，FANアルゴリズム[7]，SOCRATES法[8]などが完全かつ有効なアルゴリズムとして良く知られている。また，トライステート素子，双方向エッジなどを含む複雑な組合せ回路に対してもいくつかの手法が提案されている[9，10]。しかし，100kゲート級以上の大規模な論理回路を対象とした場合には，このようなアルゴリズムを利用した手法だけでは処理時間的に実用的ではなくなるのは自明である。したがって，これを補う手法として，乱数パターンを用いたテストパターン生成が必要である。乱数パターンによるテストパターン生成では大量のパターンを高速に故障シミュレーションする必要が生じるが，この点では，PPSFP(並列パターン単一故障伝播)法[11]，動的2次元並列法[12]など有効な手法が提案されており，実用的には問題ないと考えられる。

以下では，重み付き乱数パターン発生[13]を利用することにより，トライステート素子，双方向エッジなどを含む大規模かつ複雑な論理回路に対しても，品質の良いテストパターンを効率良く自動生成できるシステムについて報告する。

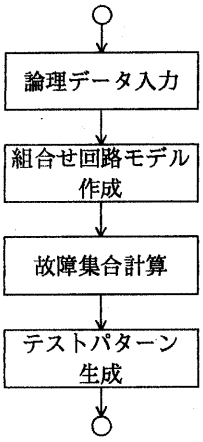

図1　テストパターン自動生成システムの処理の流れ

2．テストパターン自動生成システムの概要

ここでは，完全スキャン設計された論理回路[14]のみを処理の対象とするテストパターン自動生成システムを考える。図1はシステムの処理の流れを示したものである。システムでは，まず，スキャン回路を含む論理データを組合せ回路としてモデル化し，次に，これに基づいてテストパターン生成の対象故障集合を計算する。ただし，ここでは単一縮退故障のみを取り扱う。このようにして求められた回路モデル及び故障集合に基づいてテストパターンを生成するが，ここでは，2種類のアプローチを用いている。即ち，乱数パターンを利用したテストパターン生成とアルゴリズムによるテストパターン生成である。乱数方式では重み付き乱数パターンを発生し，これに対する故障シミュレーションを実施して有効テストパターン（新たな検出故障のあるパターン）を求める。ただし，ここでは処理時間を短縮するため，PPSFP法[11]をベースとした故障シミュレーション法を採用している。アルゴリズム方式では組合せ回路を対象とするテストパターン生成アルゴリズムをベースとした手法で個々の故障に対するテストパターンを生成し，生成されたテストパターンの故障シミュレーションにより同時検出故障を

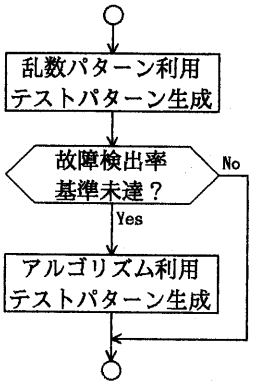

図２　テストパターン生成の処理の流れ

計算する。この２種類のアプローチは，図２に示すように順次実行される。

３．重み付き乱数パターンの利用

アルコリズム的なテストパターン生成手法によらない簡便なテストパターン生成手法として，擬似乱数を用いて入力パターンをランダムに０か１に割り当てる方法が知られている。しかし，この方法をそのまま双方向エッジを含む大規模論理回路を対象とした実用システムに適用した場合には，以下に示すような問題が生じる。

（１）多くのパターンの故障シミュレーションが必要となるため，これに要する計算機処理時間が莫大となる。
（２）双方向エッジに対しても入力パターンを与えた場合に，その双方向エッジが出力として用いられた場合には論理値の衝突（コンフリクト）を発生する危険がある。これを回避するためにはそのパターンを破棄する必要があり，その結果故障検出能力が低下する。

そこで，効率良くかつ品質の良い乱数パターンを発生させるために，以下のような方式を採用した。

乱数パターン生成では，一般にすべての入力エッジに対して擬似乱数の値によって決まった論理値を入力パターンとして与える。しかし，上記のように双方向エッジを含む回路に対して双方向エッジに対応するすべての仮入力エッジに入力パターンを与えることは無効なパターンを多数発生する可能性がある。これでは，故障シミュレーションが非効率的になってしまう。そこで，以下の方法により一部の双方向エッジピンに対しても乱数入力パターンを与える。

（１）通常動作では入力モードでしか使用されない双方向エッジに対しては一般の入力エッジと同様に入力パターンを与える。例えば，通常動作用の入力ピンがテスト用ピンとの兼用により双方向となる場合がこれにあたる。このケースでは，その双方向エッジに対応する仮出力エッジが固定的にＺ（ハイ・インピーダンス状態）となることにより判定される。
（２）入力エッジに与えられた入力パターンに対する論理シミュレーションを実施した結果，ある双方向エッジが入力モードになった場合，即ち，その双方向エッジに対応する仮出力エッジがＺとなった場合，これに対応する仮入力エッジに乱数入力パターンを追加する。

これにより，ある部分の双方向エッジに対して，コンフリクトの危険性なしに乱数入力パターンを与えることが可能となる。

乱数パターン生成は，大量のテストパターンを高速に発生できるという利点があるが，一方では，ある種の回路では故障を検出できる確率が非常に低くなるという欠点がある。例えば，図３ではＡＮＤ素子の出力の０縮退故障を検出するためには，その出力値を１にするパターン

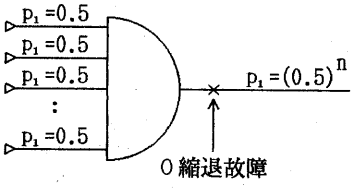

図３　乱数パターンで検出困難な故障例

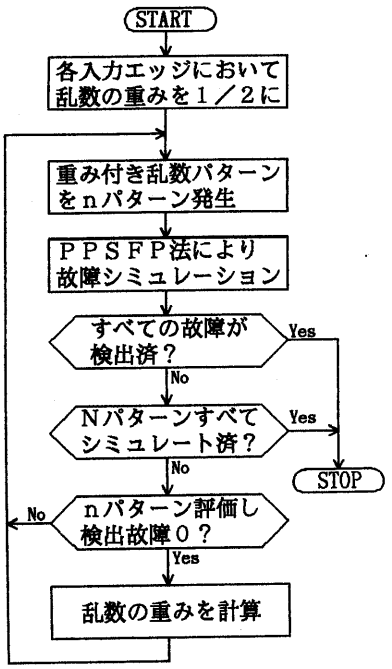

図４　重み付き乱数を用いた
テストパターン生成の処理手順

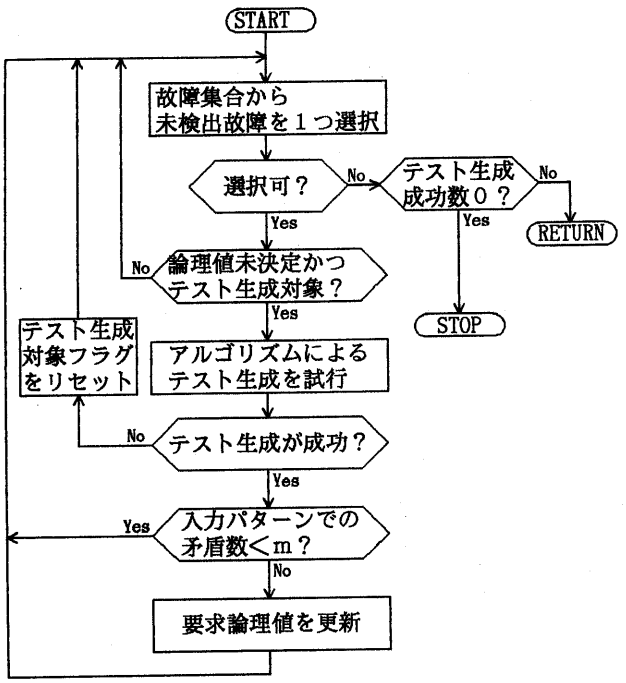

図５　乱数の重み計算の処理手順

が必要となる。しかし通常の乱数パターンでは出力値が１となる確率は非常に低くなるため，これを検出するテストパターンを見つけるのは困難である。そこで，入力値を０とするか１とするかを等確率としない重み付き乱数パターンが一部で用いられるようになっている。乱数の重み付け方法としては，テスタビリティ尺度に基づいて重みを計算する方法が一般的であるが，テストパターン生成アルゴリズムにより求めた入力パターンに基づいて重みを計算する方法も提案されている[１３]。ここで対象とする回路モデルは，トライステート素子，双方向エッジ，フリップフロップの組合せ回路モデルなど従来のテスタビリティ尺度では十分な評価が困難な素子が含まれるため，ここではアルゴリズムを用いたテストパターン生成に基づく方法を適用することにした。ここで用いる重み付き乱数を用いたテストパターン生成の処理手順を図４に示す。ただし，Ｎは発生する乱数パターン数の上限値を，また，ｎは故障シミュレーションで同時に評価するパターン数を表わす。

重み計算の処理手順を図５に示す。この方式の特徴は以下のとおりである。

（１）１組の重み付き乱数で多くの故障を検出できるようにするため，可能な限り多くの故障に対するテストパターン生成試行を実施する。即ち，論理値未確定部分にテストパターン生成の対象となる未検出故障が残っている限りテスト生成試行を継続している。これにより，多くの入力エッジに対して，故障検出の確率を向上させる方向に重み付けすることができる。ただし，一度テストパターン生成に失敗した故障は，それ以降はテストパターン生成試行の対象外としている。

（２）あるテストパターン生成試行の結果から求められた入力パターンが，それ以前に求められていた入力パターンと共通する入力エッジにおいて矛盾がないかどうかをチェックし，矛盾している個所が多数

(ここではm個所以上)の場合には，そのパターンは重み計算に使用しないようにしている。これは，矛盾するパターンをどちらも発生しやすくしようとして重み付けの効果が低下してしまうことを回避するための工夫である。

（3）各入力エッジでの乱数の重みを計算するため要求論理値を用いている。この要求論理値としては，0，1，Tの3種類の値を用いる。テストパターン生成試行を実施した結果求められた入力パターンに基づいて，表1に示すように更新される。重み付き乱数パターンの発生においては，このようにして求められた要求論理値に従って，入力エッジに1を与える確率を決定する。即ち，要求論理値0，1，Tに対応して，入力エッジに1が割り当てられる確率は各々1／16，15／16，1／2となる。

このようにして決定した重み付き乱数を用いることにより以下の効果が得られる。

（1）等確率の乱数パターンで検出困難な故障(例えば，図3の場合)に対しても，ある程度高い確率でこれを検出するパターンを発生できる。

（2）双方向エッジを入力モードに確定できる確率を高くすることが可能であるため，双方向エッジを多く含む回路に対しても高い検出能力をもつ乱数パターンを発生することができる。

表1　要求論理値の更新ルール

		入力パターン	
		0	1
現在値	0	0	T
	1	T	1
	T	T	T

4．試行結果及びその評価

以上示した方式を組み込んだテストパターン自動生成システムに対して14種の試行データを用いて評価実験を行なった。また，重み付き乱数を利用したことによる効果を明らかにするため，アルゴリズム利用のテストパターン生成のみを用いた評価実験も行なった。以上の試行結果を表2に示す。この結果をまとめると重み付き乱数利用による効果は以下のようになる。

（1）処理時間に関しては5～11倍の短縮を達成した。これは重み付き乱数パターンの故障検出能力が高い水準にあることを示している。

（2）検出率に関しては全体平均で99．4％を超える高い水準が得られた。これは，乱数パターンとして，アルゴリズムでは生成困難なパターンが偶然発生したため，一部の故障が検出されたことによる。

（3）テストパターン数に関しては，データによってバラツキがあるがほぼ同等である。

表2　テストパターン自動生成の試行結果

No.	ゲート規模	対象故障数	アルゴリズム手法のみ利用				重み付き乱数＋アルゴリズム			
			未検出故障数	検出率(%)	処理時間(※)	パターン数	未検出故障数	検出率(%)	処理時間(※)	パターン数
1	5k(3種)	5632	74	98.81	1.00	569	73	98.82	0.21	551
2	6k(3種)	7229	57	99.27	1.75	755	49	99.37	0.27	642
3	8k(2種)	12756	113	98.94	4.70	1470	53	99.52	0.52	1001
4	10k(3種)	13684	20	99.84	7.94	2216	13	99.91	0.71	1236
5	18k(3種)	28991	93	99.58	22.12	3867	66	99.70	3.59	3326
—	全体平均	8506	69	99.31	7.70	1797	51	99.46	1.10	1376

※：処理時間は相対値

これは乱数パターンを利用しても，有効なパターンのみを取り出すことにより，アルゴリズム生成パターンと同等の品質をもつテストパターンを生成できることを示している。

以上の試行結果から以下のことが言える。

（1）重み付き乱数パターンの利用は，計算機処理時間の短縮ばかりではなく，未検出故障の削減にも有効である。

（2）重み付き乱数パターンとアルゴリズムによるテストパターン生成を組み合わせることにより，処理時間，故障検出率とも十分な水準を達成することができる。

5．おわりに

トライステート素子、双方向エッジ等を含む大規模かつ複雑な論理回路を対象とするテストパターン自動生成システムに対して，重み付き乱数パターンを利用することにより処理効率の向上を図った。試行データを用いた評価の結果，アルゴリズムによるテストパターン生成のみを用いた場合に比べて，重み付き乱数パターンを利用した場合３～１６倍の高速化が達成された。この結果より，本システムは大規模論理回路に対して実用性があるとの見通しが得られた。

今後の課題としてはテストパターンの品質をさらに向上させるための故障検出能力の向上があるが，これににについては，アルゴリズム手法の中に高検出能力を得るための処理を導入する方法について検討する予定である。

参考文献

[1] S. Funatsu, N. Wakatsuki and T. Arima, "Test Generation Systems in Japan," Proc. 12th DA Conf., pp. 114-122, 1975.

[2] E.B. Eichelberger and T.W. Williams, "A Logic Design Structure for LSI Testability," Proc. 14th DA Conf., pp. 462-468, 1977.

[3] H. Ando, "Testing VLSI with Random Access Scan," Dig. COMPCON 1980, pp. 50-52, 1980.

[4] S. Kuboki, I. Masuda, T. Hayashi and S. Torii, "A 4K CMOS Gate Array with Automatically Generated Test Circuits," IEEE J. Solid-State Circ., Vol. SC-20, No.5, pp. 1018-1024, 1985.

[5] J. P. Roth, W. G. Bouricious and P. R. Schneider, "Programmed Algorithms to Compute Tests to Detect and Distinguish between Failures in Logic Circuits," IEEE Trans. Electron. Comput., Vol. EC-16, No. 5, pp. 567-579, 1967.

[6] P. Goel, "An Implicit Enumeration Algorithm to Generate Tests for Combinational Logic Circuits," IEEE Trans. Comput., Vol. C-30, No. 3, pp. 215-222, 1981.

[7] H. Fujiwara and T. Shimono, "On the Acceleration of Test Generation Algorithms," IEEE Trans. Comput., Vol.C-32, No. 12, pp. 1137-1144, 1983.

[8] M. H. Schulz and E. Auth, "Advanced Automatic Test Pattern Generation and Redundancy Identification Techniques," Dig. FTCS-18, pp. 30-35, 1988.

[9] T. Ogihara, S. Murai, Y. Takamatsu, K. Kinoshita and H. Fujiwara, "Test Generation for Scan Design Circuits with Tri-State Modules and Bidirectional Terminals," Proc. 20th DA Conf., pp. 71-78, 1983.

[10] N. Itazaki and K. Kinoshita, "Test Pattern Generation for Circuits with Three-State Modules by Improved Z-Algorithm," Proc. ITC-86, pp. 105-112, 1986.

[11] J. A. Waicukauski, E.B. Eichelberger D. O. Forlenza, E. Lindbloom and T. McCarthy, "Fault Simulation for Structured VLSI," VLSI System Design, Vol. 6, No. 12, pp. 20-32, 1985.

[12] N. Ishiura, M. Ito and S. Yajima, "High-Speed Fault Simulation Using a Vector Processor," Proc. ICCAD-87, pp. 10-13, 1987.

[13] J. A. Waicukauski, E. Lindbloom, E. B. Eichelberger and O. P. Forlenza, "A Method for Generating Weighted Random Test Patterns," IBM J. Res. Develop., Vol. 33, No. 2, pp. 149-161, 1989.

[14] T. Hayashi, K. Hatayama, Y. Kunitomo and S. Kuboki, "An Approach to Design Automation for Highly Testable Logic Circuits," Proc. ICCAD-86, pp. 98-101, 1986.