設計自動化 59－7
(1991. 10. 9)

実数値シミュレーションを利用した順序回路テスト生成手法

彦根 和文† 池田 光二† 畠山 一実† 林 照峯† 高倉 正博‡

†(株)日立製作所 日立研究所 ‡日立エンジニアリング(株)

あらまし 実数値に拡張した論理シミュレーションを利用する順序回路用テスト生成手法を提案する。この手法はテスト生成問題を実数値の最適化問題に置き換え，一種の収束計算を行うことによりテストパターンを生成するものである。一般に順序回路のテスト入力は複数時刻にわたるパターン系列となることから，テストパターン系列全体を導出するために，複数時刻分の入力パターン系列毎に収束計算処理を行う。最後にISCAS'89 ベンチマーク回路を用いた評価結果を示す。

Seqential Circuit Test Generation by Real Number Simulation

Kazunori Hikone† Mitsuji Ikeda† Kazumi Hatayama† Terumine Hayashi†

Masahiro Takakura‡

† Hitachi Research Laboratory, Hitachi, Ltd. ‡ Hitachi Enginering, Co., Ltd.

Abstract This paper presents a test pattern generation method for sequential circuits using a real number simulation, called extended logic simulation. This method formulates a test generation problem as a real number optimization problem and generates a test pattern using a convergence calculation. In general, a test pattern sequence for a sequential circuit has several time periods. Here each convergence calculation is achieved for constant time periods to lead the whole test pattern sequence. Finally experimental results for ISCAS '89 benchmark circuits are given.

１．はじめに

論理回路の大規模化と多様化が進むにつれて設計期間の短縮が重要な課題となっている。回路の故障検査に必要なテストパターンの設計においては，設計工数削減のためテスト生成の高性能化が重要となっている。

一般に論理ＬＳＩは組合せ回路部分と記憶素子部分からなる順序回路であるが，順序回路のテスト生成は非常に困難である。そこで，現在，論理ＬＳＩの多くはスキャン設計方式などのテスト容易化設計手法を取り入れることにより，順序回路のテスト生成問題を比較的容易な組合せ回路の問題に変換して扱っている。しかし，マイクロプロセッサ等ではテスト容易化設計手法を取り入れることが不可能な状況が生じる。このような順序回路に対しては従来からＤアルゴリズム［１］やＰＯＤＥＭ法［２］など経路活性化手法を時間軸展開の概念を用いて順序回路テスト生成に拡張する方法が提案されている。また，シミュレーションを利用したコスト関数を用いてテストパターン系列を導く方法［３］，［４］も提案されている。しかし，いずれの方法でも必ずしも十分な性能は得られていない。

本論文では，論理値を０から１までの実数で表わし，乗算を基調とした演算を行うシミュレーション（連続値論理シミュレーションと呼ぶ）を利用する順序回路テスト生成手法を提案する。

以下では，連続値論理シミュレーションとテスト生成の考え方について述べ，次に単一クロック同期式順序回路を対象とした順序回路テスト生成手法について述べる。最後に本手法のISCAS '89 ベンチマーク回路に対する評価結果を示す。

２．実数値シミュレーションとテスト生成

2.1 連続値論理シミュレーション

論理回路の入力パターンに対する出力値は論理シミュレーションを行うことにより得られる。しかし，この方法では論理シミュレーションの結果，回路出力論理値が期待した値になっていないとき，入力パターンが回路出力値を期待した値に設定するパターンからどれくらい離れているかが分からない。図１に示す３つの３入力ＡＮＤゲートの出力論理値はいずれも０である。図中，(c)は３入力線のうち１つを１にすれば，出力論理値を１にできる。(b)は２つを１にすれば出力論理値を１にできる。また(a)では３入力全部を１にしなければ出力論理値を１にできない。このように，図１(a)，(b)，(c)は出力論理値が０でも，出力論理値を１にするしやすさに差がある。この差を表現するために［０，１］の実数値を用いたシミュレーション手法を導入する。このシミュレーション手法は従来の２値論理を扱うシミュレーションに対して，０，１の論理値を０から１までの連続した実数値として扱うため連続値論理シミュレーションと呼ぶ。

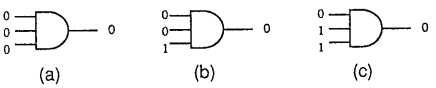

図１．　出力論理値を１にするしやすさの違い

連続値論理シミュレーションでは実数値を扱うために，論理回路の入力端子に与える論理値０，１をそれぞれ，0.0＋ε，1.0－ε（0.0<ε<0.5）とする。そして，各論理ゲートに対し，図２に示す演算規則を適用して，その出力値を計算する。例えば図１(a)，(b)，(c)の各出力値はε＝0.01とした場合，次のようになる。

（ａ）0.01×0.01×0.01=0.000001
（ｂ）0.01×0.01×0.99=0.000099
（ｃ）0.01×0.99×0.99=0.009801

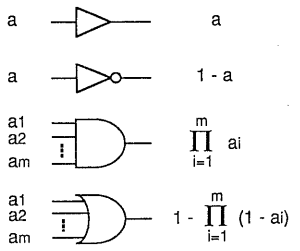

図２．　ゲートの実数値演算規則

このように，２値の論理シミュレーションでは同じ論理値０であるのに対して，連続値論理シミュレーションでは１にするしやすさに差があることを表現できる。

2.2 連続値論理シミュレーションとテストパターン

一般に，与えられた入力パターンが検出したい故障のテストパターンであるかどうかの判定は，正常回路と故障回路に対する論理シミュレーションの出力結果を比較することにより行われる。しかし，この方法ではシミュレーションの結果，入力パターンがテストパターンでなかったとき，そのパターンがテストパターンと比較してどのくらい異なるかが分からない。以下では，この異なる度合いを距離と呼ぶ。

入力したパターンとテストパターンの距離を連続値論理シミュレーションを使って，表現することができる。すなわち，論理回路Ｃに対してその故障回路Ｃｆを構成し，入力パターンをＣ，Ｃｆに与えてシミュレーションを行った結果から正常回路と故障回路の出力値の差を求めることにより得られる。このとき，故障回路のシミュレーションでは故障箇所が１縮退故障のときは 1.0，０縮退故障の時は 0.0をそれぞれ与えることによって正常回路との差を得る。

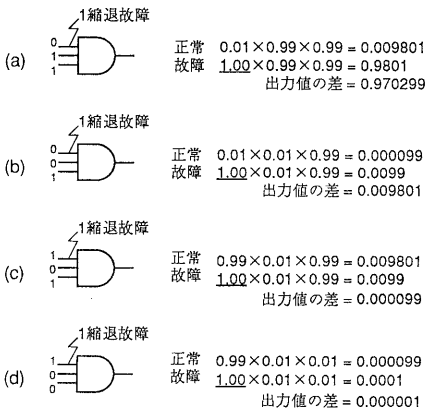

図３．　正常回路と故障回路での出力値の差

図３に示す３入力AND回路の入力端子の１縮退故障に対するテストパターンは (0,1,1) である。パターン (0,1,1) (0,0,1) (1,0,1) (1,0,0)を正常回路と故障回路についてそれぞれシミュレーション行い，その出力値の差を計算すると，図３に示すように出力値の差はテストパターン(0,1,1)で最も大きく，テストパターンと異なるビット数が多くなるほど小さくなる。これによって，入力パターンとテストパターンの距離が表現可能となる。

2.3 収束計算処理と組合せ回路テスト生成

前節に示した距離の考え方を利用した組合せ回路中の１故障に対するテストパターン生成方法［５］の概略フローを図４に示す。テストパターンに対する入力パターンのコストは正常回路と故障回路における出力値の差を全出力端子について求め，その総和の逆数とする。

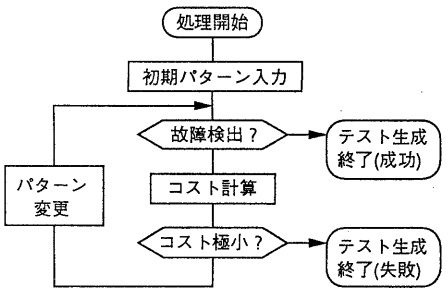

図４．　収束計算処理によるテスト生成方法

３．順序回路テスト生成

3.1 順序回路への適用

一般に順序回路中の任意の故障に対するテスト入力は，１時刻以上にわたるパターンの時系列となる。また，その系列長が特定できないため，テストパターン系列全体に関してのコスト計算に基づく収束計算処理適用は困難である。そこで，時間軸の前方向のみに順次パターン系列を決定していくことで，テストパターン系列を生成する考え方（前方向テスト生成：図５）を導入し，各パターン系列決定段階において収束計算処理を用いる。前方向テスト生成では，処理対象の時刻におい

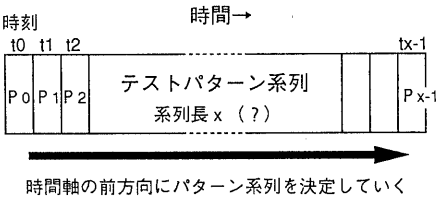

図５．　前方向テスト生成

て故障が検出できないときは，故障信号をより外部出力端子に近い記憶素子へ伝搬させるようなパターン生成を行い，回路状態を遷移して行く。

3.2 複数時刻系列単位のパターン生成

上記の前方向テスト生成では，順序回路にパターン系列を与えることによってきまる回路出力値や記憶素子状態値を評価してパターン系列の決定を行う。この場合，１時刻分のパターン印加による外部出力と回路状態からコストを求め，コスト極小となる入力パターンを選択する方法が最も単純であるが，図６のように，状態３から故障検出可能な状態への状態遷移を起こす入力パターンのコストが大きい場合，故障検出不可能な状態に陥ってしまうことがある。このような状態に陥ることを減少させるため，複数時刻にわたるパターン系列のコストを求めることにする（図７）。そして，このパターン系列に対して収束計算処理を行うことで，複数時刻先まで考慮したパターン系列生成に基づく前方向テスト生成を行うことができる。

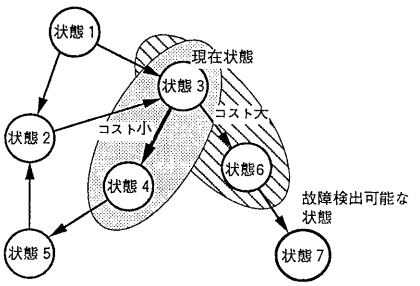

図６．　１時刻先までの評価

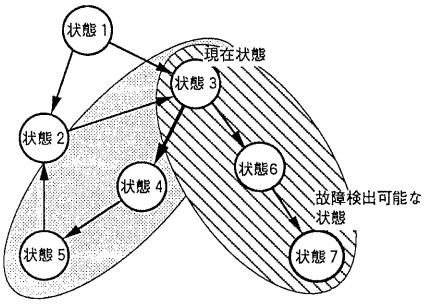

図７．　複数時刻先までの評価

3.3 コスト計算

順序回路Ｃに対して印加する複数時刻にわたる入力パターン系列のコスト計算方法を次に示す。

出力数ｍ，記憶素子数ｎの順序回路Ｃが正常な場合と故障ｆを有する場合を考える。任意の時刻ＴのＣに対して，系列長ｋのパターン系列Ｐを与えて連続値論理シミュレーションを行った結果，時刻ｔにおける正常時および故障時の回路出力値を$Vt_1(t)$～$Vt_m(t)$，$Vf_1(t)$～$Vf_m(t)$，ｋ時刻後の記憶素子における実数値を正常時，故障時それぞれSt_1～St_n，Sf_1～Sf_nとする。このとき，故障ｆに対する入力パターン系列ＰのコストC_{seq}を次式で定義する。

$$C_{seq} = \frac{1}{D_{out} + D_m}$$

ここで，

$$D_{out} = \sum_{t=T}^{T+k-1} \sum_{i=1}^{m} \left| Vt_i(t) - Vf_i(t) \right|$$

$$D_m = \sum_{j=1}^{n} \left\{ W_j \times \left| St_j - Sf_j \right| \right\}$$

上式中のW_jは，記憶素子ｊにおける正常時，故障時の出力値の差に対する重みで，記憶素子ｊから外部出力端子への経路の内，経路中に存在する記憶素子の数が最も少ない経路の記憶素子数（ｊを含む）をnf_jとしたとき，

$$W_j = \frac{1}{2^{nf_j}}$$

と定義する。

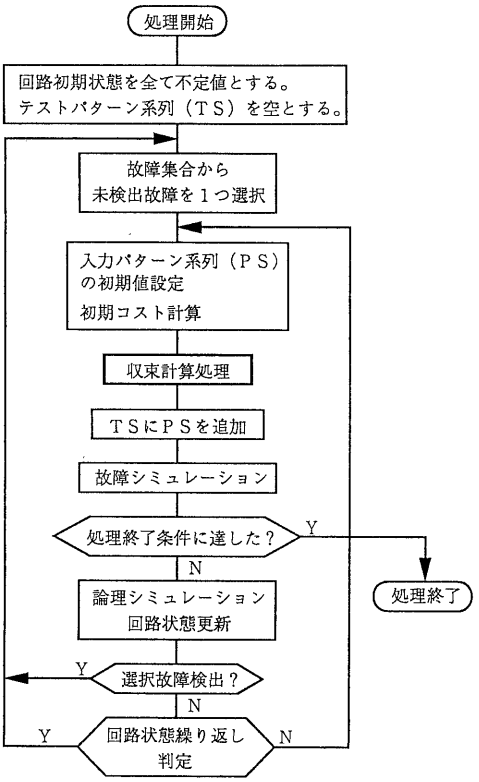

図８．　順序回路テスト生成手法

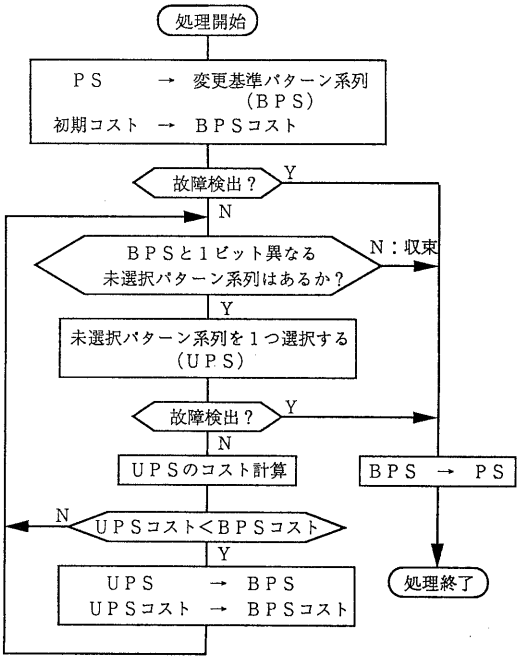

図９．　収束計算処理

この重み付けによって出力値の差の評価優先度は外部出力が最優先となる。また，外部出力に近い記憶素子の優先度が高くなるため，故障信号を外部出力端子に近い記憶素子へ誘導するパターン系列生成が図れる。

3.4 テスト生成方法

上記のパターン系列生成とコスト計算方法を用いた順序回路テスト生成手法の処理手順を図８に示す。ここで，ＴＳはテストパターン系列，ＰＳは収束計算処理の対象となるパターン系列を表す。また，回路状態繰り返し判定処理は，同一故障に対する処理中，ＰＳを論理シミュレーションした後の回路状態をチェックし，過去の状態履歴中に同じ状態があれば処理対象の故障を変更する。これによって，再び同じ状態が繰り返されることを回避している。

図９に本テスト生成手法で用いられる収束計算処理の手順を示す。ここで，ＢＰＳはパターン系列処理中の変更基準パターン系列，ＵＰＳはコスト計算のために選択されたパターン系列である。収束計算処理では，変更基準パターン系列（ＢＰＳ）に対して１ビット異なるパターン系列をパターン系列のコスト比較対象とする。そして，コストが減少するパターン系列をＢＰＳとして処理を繰り返し，比較対象の未選択パターン系列がなくなるまで処理を続ける。

4．評価実験

以上示した手法について，ISCAS'89 ベンチマーク順序回路に対して評価実験を行った。

4.1 実験における設定事項

(1) 収束計算処理を行う単位パターン系列の系列長

収束計算処理の単位となるパターン系列の系列長は，順序回路Cの記憶素子jを通る外部入力端子から外部出力端子への経路の内，経路中に存在する記憶素子の数が最も少ない経路の記憶素子数をff_jとするとき，

$$\text{系列長} = 2 \times (1 + \mathrm{MAX}_j\, ff_j)$$

で与える。ここで，$(1 + \mathrm{MAX}_j\, ff_j)$は入力パターンの影響が全記憶素子に伝わり，回路外部出力端子に現れる最小時刻数を示し，単位パターン系列長をその2倍とした。

(2) 論理値に対応させる実数値

外部入力端子に与える入力パターンの論理値に対応した実数値は，$\varepsilon = 0.01$とし，論理値0，1，Xに対して，0.01，0.99，0.5とする。また，回路の記憶素子に与える初期値は，各パターン系列に対するシミュレーション開始時刻における記憶素子状態の論理値0，1，Xに対して，同様に 0.01，0.99，0.5 とし，系列に対するシミュレーション途中では前時刻の回路状態を実数値シミュレーションの結果得られる値をそのまま受け渡すものとする。

(3) 終了条件

・総テストパターン系列長が最大テスト系列長(5000パターンとする)を超えた場合
・1000パターン以上連続して故障が検出されなかった場合

4.2 実験結果

ISCAS'89ベンチマーク回路を用いた評価実験結果を表1に示す。実験は大型計算機HITAC M-680H上で行った。この結果，S344，S349，S1196，S1238，S1488，S1494で90%以上の故障検出率が実用的な計算時間内で得られた。また，経路活性化法に基づく方法では故障検出率，計算時間共に良くないS1423に対しても，ほぼ90%の故障検出率が得られ，他の回路についても良好な結果が

表1． 実験結果

						生成単位系列長 自動					生成単位系列長 = 1			
回路名	素子数	FF数	入力数	出力数	故障数	生成時間 CPU(sec)	検出 故障数	検出率 (%)	テスト 系列長	生成系列 単位長	生成時間 CPU(sec)	検出 故障数	検出率 (%)	テスト 系列長
S208	96	8	11	2	215	54.70	132	61.40	1268	4	7.71	132	61.40	228
S298	119	14	3	6	308	22.61	265	86.04	320	8	5.07	265	86.04	1260
S344	160	15	9	11	342	78.29	329	96.20	336	6	16.86	329	96.20	1845
S349	161	15	9	11	350	150.80	335	95.71	1434	6	12.21	333	95.14	803
S382	158	21	3	6	399	64.02	343	85.96	1608	8	6.36	277	69.42	939
S386	159	6	7	7	384	70.60	314	81.77	796	4	19.25	308	80.21	1809
S400	162	21	3	6	424	62.89	368	86.79	1304	8	4.28	359	84.67	275
S420	196	16	19	2	430	263.23	173	40.23	2424	4	28.13	172	40.00	392
S444	181	21	3	6	474	90.49	420	88.61	2512	8	4.23	404	85.23	300
S526	193	21	3	6	555	93.68	434	78.20	1920	8	15.46	433	78.02	3402
S526N	194	21	3	6	553	92.33	432	78.12	2336	8	10.10	432	78.12	1801
S641	379	19	35	24	467	638.34	404	86.51	784	4	83.33	404	86.51	624
S713	393	19	35	23	581	624.39	476	81.93	572	4	135.55	476	81.93	1366
S820	289	5	18	19	850	1320.93	695	81.76	4432	4	102.88	474	55.76	2285
S832	287	5	18	19	870	1125.15	695	79.89	3152	4	159.59	455	52.30	4260
S838	390	32	35	2	857	389.53	246	28.70	332	4	82.10	247	28.82	165
S1196	529	18	14	14	1242	516.76	1238	99.68	1208	4	130.81	1229	98.95	1943
S1238	508	18	14	14	1355	1246.94	1273	93.95	4644	4	220.54	1274	94.02	4728
S1423	657	74	17	5	1515	4697.77	1363	89.97	1820	10	279.37	1306	86.20	4468
S1488	653	6	8	19	1486	540.84	1396	93.94	1792	4	162.66	1354	91.12	4410
S1494	647	6	8	19	1506	884.44	1422	94.42	3476	4	106.44	1333	88.51	2326
S5378	2779	179	35	49	4603	61971.87	3490	75.82	4450	10 (*)	829.41	1469	31.91	1627

(*) S5378は、系列長を強制的に10とした。

得られた。表中，複数時刻にわたるパターン系列を評価する方法（生成単位系列長＞１）は，１時刻単位に評価する方法と比較して，総じて故障検出率は高くなっている。これは，１時刻先だけの評価ではテストパターン系列生成途中でコストが大きい入力パターンを選択しなければ故障を検出できる状態に達することができない場合があり，複数時刻先までの評価を行うことで生成能力が向上すること示している。また，複数時刻パターン系列を評価する方法が１時刻単位に評価する方法に比べてテスト生成時間が長くなるのは，収束計算処理において選択対象のパターン系列の全ビット長が１時刻のパターンのビット長よりも多くなるため，コストが収束するまでのパターン系列変更回数が多くなるためと考えられる。

５．おわりに

実数値に拡張した論理シミュレーションを利用する順序回路用テストパターン系列生成手法を提案し，その評価実験を行いその有効性を確認した。今後は，故障検出率向上のために生成単位系列長，重み付け係数の計算方法，入力パターンの論理値に対応する実数値を決定するεパラメータの設定方法等のヒューリスティックの検討を進める予定である。また，本手法は処理構造が単純なため，並列処理やベクトル処理の適用により大きな効果が得られるものと考えられ，今後その検討も進める予定である。

参考文献

[1] 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. Compt., Vol. EC-16, No.5, pp. 567-579, 1967.

[2] P. Goel, "AN Implicit Enumeration Algorithm to Generate Tests for Combinational Logic Circuits," IEEE Trans. Compt., Vol.C-30, No. 3, PP. 215-222, 1981.

[3] K. T. Cheng and V. D. Agrawal, "A Sequiential Circuit Test Generator Using Threshold-Value Simulation," FTCS-18 Digest of Papers, pp. 24-29, 1988.

[4] V. D. Agrawal, K. T. Cheng, and P. Agrawal, "CONTEST: A Concurrent Test Generator for Sequential Circuits," Proc. 25th DA Conf., pp. 84-89, 1988.

[5] 池田，畠山，林，"バックトラック処理不要な組合せ回路テスト生成手法，" 信学技報 FTS89-35, pp. 53-58, 1989.