時間展開モデルを用いた無閉路順序回路の動的テスト系列圧縮方法の解析

細川利典† 吉村正義† 太田光保†

無閉路順序回路に対するテスト系列生成は，時間展開モデルを用いて生成することができ，生成された無閉路順序回路全体のテスト系列から逆変換パターンを生成し，故障シミュレーションを実行して，未検出故障数を削減することにより，動的に無閉路順序回路のテスト系列長を短縮することができる．本論文では，逆変換パターンで多くの未検出故障を検出するために，逆変換パターンはドントケア（X）を含む可能性があるという点に着目して，逆変換パターンと ATPG パターンを時間展開モデルの圧縮バッファを用いて圧縮する動的テスト系列圧縮方法を提案する．またテスト系列圧縮はテスト系列中の X の数が多いほど圧縮効率が高くなる可能性があるという点に着目したテスト系列圧縮方法を提案する．実際の回路に対して本提案手法を適用した結果，テスト系列長を 34〜67%まで短縮することができた．

An Analysis of Dynamic Test Sequence Compaction Methods for Acyclic Sequential Circuits Using a Time Expansion Model

Toshinori Hosokawa,† Masayoshi Yoshimura† and Mitsuyasu Ohta†

A test sequence for an acyclic sequential circuit can be generated using a time expansion model, and the length of the test sequence can be dynamically shortened by performing fault simulation with reversely transformed patterns which are generated from the whole test sequence for an acyclic sequential circuit in order to detect undetected faults. In this paper, we present the dynamic test sequence compaction method: a reversely transformed pattern and an ATPG pattern are compacted in a compaction buffer for a time expansion model focusing on the point that a reversely transformed pattern may have don't cares (Xs) in order to detect more undetected faults . We also present the test sequence compaction method focusing on the point that compaction efficiency may be high as the number of Xs in a test sequence increases. Experimental results for practical circuits showed that our presented methods could shorten the length of test sequences by 34 to 67%.

1. はじめに

近年の LSI の大規模化，複雑化により LSI のテスト設計の自動化は必要不可欠である．一般の順序回路のテスト系列生成（ATPG）は困難な問題であり，高い故障検出効率を得るためにはスキャン設計に代表されるテスト容易化設計（DFT）が必要である．多くの LSI 設計に普及しているスキャン設計は，LSI 中のすべてのフリップフロップ（FF）をスキャン FF で構成するフルスキャン設計[1]と一部の FF のみをスキャン FF で構成するパーシャルスキャン設計[2)〜5)]がある．スキャン設計では，スキャン FF を等価的に外部入出力と見なせるので，スキャン FF を取り除いた残りの回路（核回路）に対して ATPG を行えばよい．よって，フルスキャン設計では，核回路が組合せ回路になるので，組合せ ATPG アルゴリズムで ATPG 可能となり，ほぼ完全な故障検出効率が得られるが，面積，性能劣化，消費電力のオーバヘッドが大きくなる．一方，パーシャルスキャン設計はフルスキャン設計に比べて前述のオーバヘッドを削減できるが，核回路が順序回路になるので，一般には組合せ ATPG 不可能であり，順序 ATPG を必要とする．したがって，パーシャルスキャン設計では，高い故障検出効率の達成はスキャン FF に置き換える FF の同定方法に大きく依存する．文献 2) では平衡構造，文献 3) では内部平衡構造，文献 4)，5) では無閉路構造を核回路とするパーシャルスキャン設計方法とそのテスト系列生成方法が

† 松下電器産業株式会社半導体開発本部
Corporate Semiconductor Development Division, Matsushita Electric Industrial Co., Ltd.

提案され，順序回路である核回路を組合せ ATPG 可能とし，フルスキャン設計よりも小さいハードウェアオーバヘッドで同等の故障検出効率が得られた．

高い故障検出効率を得るテスト系列を高速に生成することに加え，LSI テスターでのテスト時間を短縮すること，すなわちテスト系列長を短縮することが重要である．テスト系列長を短縮する技術としてテストパターンの圧縮技術[6)~10)]がある．特に文献 10) は時間展開モデル[4)]を用いた無閉路順序回路のテスト系列の圧縮方法を提案している．

本論文では，文献 10) で提案した逆変換故障シミュレーションによる動的テスト系列圧縮方法と，新たに生成された無閉路順序回路のテスト系列とすでに存在する無閉路順序回路全体のテスト系列との圧縮方法をそれぞれ解析し，それぞれの方法を改善するために**逆変換パターン**と **ATPG パターン**による動的テスト系列圧縮方法と，故障検出情報を参照したテスト系列変換方法および ATPG パターン中のドントケア（X）の値の決定方法を提案する．

本論文は，次のような構成になっている．まず，2 章で時間展開モデルを用いた無閉路順序回路のテスト系列生成方法と今回提案するテスト系列圧縮方法の動機について述べる．3 章では時間展開モデルを用いた無閉路順序回路のテスト系列生成アルゴリズムについて述べる．4 章では逆変換パターンと ATPG パターンによる動的テスト系列圧縮方法について述べる．5 章では故障検出情報の参照によるテスト系列変換方法と ATPG パターン中の X の値の決定方法について述べる．6 章では，実際の回路に対して 4，5 章で提案した方法を適用した実験結果を示し，その考察を行う．7 章では，本論文のまとめと今後の課題について述べる．

2. 時間展開モデルを用いたテスト系列生成

2.1 テスト系列生成

時間展開モデルを用いた無閉路順序回路のテスト系列生成方法[4)]について，例を用いて説明する．**図 1** は無閉路順序回路 S の例を示す．図 1 において，FF1～FF5 は FF，A～E は組合せ論理部，$PI1$～$PI3$ は外部入力，$PO1$，$PO2$ は外部出力である．

図 2 は図 1 の S の時間展開モデル $C(S)$ である．$C(S)$ は S の各外部出力から外部入力まで，入力方向に組合せ論理部を展開した組合せ回路である．$C(S)$ の組合せ論理部，外部入力，外部出力はラベル t（非負の整数）を持つ．一般に時間展開モデルにおいて，組合せ論理部 a のラベルを $l(a)$，組合せ論理部 b のラベルを $l(b)$ とし，a と b が接続しているとすると

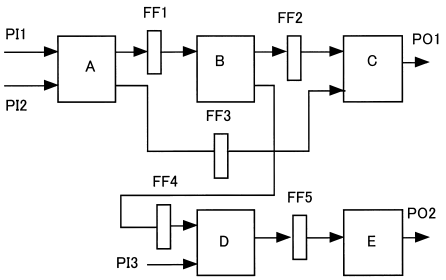

図 1 無閉路順序回路：S
Fig. 1 Acyclic sequential circuit: S.

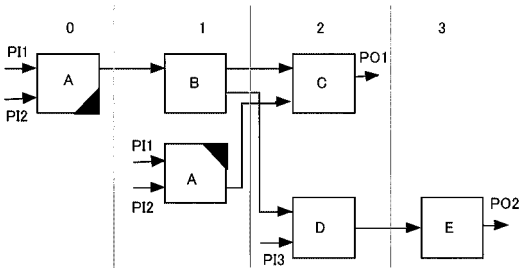

図 2 時間展開モデル：$C(S)$
Fig. 2 Time expansion model: $C(S)$.

き，対応する無閉路順序回路において，a と b の間には $|l(a) - l(b)|$ 個の FF が存在することを表している．図 2 において，$C(S)$ の上部に書かれている数字は，その列に存在する組合せ論理部，外部入力，外部出力のラベルを表す．また，組合せ論理部の黒塗りの部分は，外部出力または他の組合せ論理部の入力のいずれにも到達不可能である信号線または論理ゲートを表し，時間展開モデルから除去されている．

無閉路順序回路の故障 f_a に対応する時間展開モデルの故障 f_e は，f_a の存在する組合せ論理部と対応する時間展開モデルの各組合せ論理部の同じ位置（信号線）に存在する 1 つの多重故障となる．たとえば，図 1 の S の A 中の故障は，図 2 の $C(S)$ に示すように A が 2 つ存在するので，$C(S)$ では 1 つの 2 重故障として扱う．ここで，無閉路順序回路の故障集合を F_a，F_a に対応する時間展開モデルの故障集合を F_e としたとき，以下のことが成り立つ[4)]．

(1) 無閉路順序回路の任意の故障 $f_a \in F_a$ について，f_a に対応する時間展開モデルの故障 $f_e \in F_e$ がただ 1 つ存在する．

(2) 故障 $f_a \in F_a$ に対するテスト系列が存在するとき，かつそのときに限り，故障 f_a に対応する故障 $f_e \in F_e$ に対するテストパターンが存在する．

(3) 故障 $f_e \in F_e$ に対するテストパターンは，対応

表 1 テスト系列
Table 1 Test seuence.

(a) T_1

時刻	PI1	PI2	PI3
0	0	1	X
1	1	0	X
2	X	X	1
3	X	X	X

(b) T_2

時刻	PI1	PI2	PI3
0	0	0	X
1	1	1	X
2	X	X	0
3	X	X	X

(c) C(S)の基本テンプレート

時刻	PI1	PI2	PI3
0	○	○	X
1	○	○	X
2	X	X	○
3	X	X	X

する故障 f_a に対するテスト系列に変換可能である．

すなわち，時間展開モデルの故障に対して生成したテストパターンは無閉路順序回路のテスト系列に変換可能で，その変換したテスト系列で対応する無閉路順序回路の故障を検出することができる．

図 2 の $C(S)$ のある故障に対して生成されたテストパターンを $(PI1(0), PI2(0), PI1(1), PI2(1), PI3(2)) = (0,1,1,0,1)$ とすると（$PIi(t)$：ラベル t に存在する外部入力 PIi の値），S における時刻 0 の $PI1$ の値は 0，時刻 0 の $PI2$ の値は 1，時刻 1 の $PI1$ の値は 1，時刻 1 の $PI2$ の値は 0，時刻 2 の $PI3$ の値は 1 となり，S のテスト系列は**表 1** (a) に示されるテスト系列になる．

2.2 テスト系列圧縮

$C(S)$ で生成された 2 つのテストパターン $v_1(0,1,1,0,1)$，$v_2(0,0,1,1,0)$ を S のテスト系列に変換した 2 つのテスト系列をそれぞれ T_1，T_2 とする．テスト系列 T_1，T_2 をそれぞれ表 1 (a)，(b) に示す．まず T_1（無閉路順序回路全体のテスト系列とする）が生成され，その次に T_2 が生成されたとすると，T_1 中に X の部分が存在するので，T_2 は T_1 の時刻 2 から入力できる．すなわち，T_2 は T_1 にスキュー 2 で圧縮可能[10]である．ここでスキューとは 2 つのテスト系列の時刻のずれを表す．したがって，**図 3** に示すように T_1 と T_2 を圧縮した結果，無閉路順序回路全体のテスト系列（以後全体テスト系列）T_1 は T に更新される．さらに，新たに生成されたテスト系列は，全体テスト系列とスキュー 0 から順に圧縮可能であるか否かを調べ，圧縮可能であれば圧縮して全体テスト系列を更新する[10]．

また，T_1，T_2 に示すように，テスト系列において 0 または 1 が決定している箇所と X である箇所は，すべてのテスト系列について一定である．ただし，時間展開モデルのテストパターンには X が含まれないものとする．0 または 1 が決定している箇所を○で表す

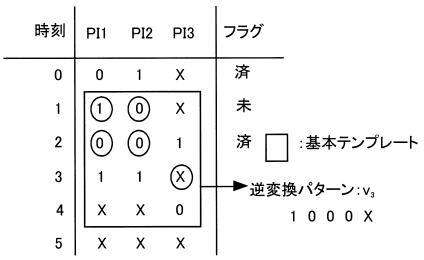

図 3 全体テスト系列：T
Fig. 3 Whole test sequence: T.

と，S の各テスト系列は表 1 (c) に示すテスト系列になる．表 1 (c) のテスト系列を $C(S)$ に対する基本テンプレート[10]という．

図 3 の T に着目すると，時刻 1〜時刻 4 のテスト系列は T_1，T_2 とは別のテスト系列であり，このテスト系列で S に対して故障シミュレーションを実行すれば新たに未検出故障を検出できる可能性がある．文献 10) では，全体テスト系列からまだ故障シミュレーションを実行していないテスト系列を時間展開モデルのテストパターンに逆変換し，その逆変換したテストパターンで時間展開モデルの未検出故障に対して故障シミュレーションを実行し，検出できた故障を未検出故障から削除して ATPG の対象となる故障数を削減することによる動的テスト系列圧縮方法が提案されている．具体的には，図 3 に示すように，全体テスト系列の各時刻にフラグを設ける．時刻 t のフラグの「済」は，時刻 t から時刻 $t+l-1$ のテスト系列（l は基本テンプレート長）を時間展開モデルのテストパターンへ逆変換したテストパターンで，故障シミュレーションを実行したことを表す．時刻 t のフラグの「未」は，時刻 t から時刻 $t+l-1$ のテスト系列を時間展開モデルのテストパターンへ逆変換したテストパターンで，故障シミュレーションを実行していないことを表す．時刻 t のフラグに何も記入されていない場合は，時刻 $t+l-1$ のテストパターンが存在しないことを表す．時刻 t のフラグが「未」である時刻 t〜時刻 $t+l-1$ のテスト系列は**逆変換可能**であるという．逆変換は逆変換可能なテスト系列を基本テンプレートと見なし，基本テンプレートの○の部分の値を抽出して，基本テンプレートの時刻 t の PIi の値から時間展開モデルのラベル t の外部入力 PIi の値を生成する．全体テスト系列から逆変換されたテストパターンを**逆変換パターン**という．

図 3 の T の逆変換可能な時刻 1〜時刻 4 のテスト系列を逆変換して，時間展開モデルのテストパターン $v_3(1,0,0,0,X)$ を生成する．文献 10) では，v_3 の X

の部分にランダムに 0 または 1 を設定していたが，逆変換パターンはランダムパターンであるので未検出故障数が少なくなると，新たな故障検出は困難になる．そのため，逆変換パターンの X の部分の値の決定方法がテスト系列圧縮の効率を高くする点において重要である．一方，時間展開モデルは組合せ回路であるので，テストパターン圧縮バッファ（以後圧縮バッファ）を用いた組合せ回路の動的テストパターン圧縮[6]を行うことができる．逆変換パターンと圧縮バッファ中のテストパターンを圧縮することによって逆変換パターン中の X の値を決定する方法を提案する．この圧縮方法を 4 章で述べる．

一方，時間展開モデルのテストパターンは，表 1 (c) に示すようなテスト系列長と X の数が一定のテスト系列に変換され，全体テスト系列と圧縮を行う．ここで，変換されたテスト系列中の X の数が多いほうが全体テスト系列との圧縮において圧縮の効率が高くなることが推測できる．故障検出情報を参照したテスト系列変換方法と時間展開モデルのテストパターン中の X の値の決定方法を 5 章で述べる．

3. 時間展開モデルを用いた無閉路順序回路のテスト系列生成アルゴリズム

時間展開モデルを用いた無閉路順序回路のテスト系列生成アルゴリズムはすでに文献 4) で提案されている．本章では，テスト系列圧縮を含んだ時間展開モデルを用いた無閉路順序回路のテスト系列生成アルゴリズムについて説明する．本アルゴリズムは以下に示す特徴を持つ．

(1) 時間展開モデルの故障に対して生成されたテストパターンは圧縮バッファを用いて圧縮される．
(2) 全体テスト系列から逆変換パターンが生成され，その逆変換パターンで時間展開モデルの未検出故障に対して故障シミュレーションが実行される．
(3) 無閉路順序回路の全体テスト系列と新たに生成されたテスト系列が圧縮される．

(2) のアルゴリズムについては 4 章で，(3) のアルゴリズムについては 5 章で詳細に述べる．本章ではテスト系列圧縮を含んだ全体のテスト系列生成アルゴリズムの概要を説明する．

図 4 にテスト系列圧縮を含む時間展開モデルを用いた無閉路順序回路のテスト系列生成アルゴリズムを示す．まず無閉路順序回路 S の時間展開モデル $C(S)$ を生成する．次に $C(S)$ の未検出故障集合 F の中に ATPG 未処理の故障がなくなるまで以下の処理を繰り返す．F から未検出でかつ ATPG 処理を行っていない故障 f を 1 つ選択し，ATPG を行いテストパターン v を生成する．以後，ある未検出故障 (f) を検出するために生成したテストパターン (v) を **ATPG パターン**と呼ぶ．次に v を $C(S)$ の圧縮バッファ $TBUF$ 中のテストパターンと圧縮する（関数 Test_pattern_compaction）．もし $TBUF$ から 1 個のテストパターン tp が出力されているならば，tp 中の X に 0 または 1 を設定し，$C(S)$ の未検出故障に対して故障シミュレーションを実行し，検出できた故障を F から削除する．次に tp を S のテスト系列 TP に変換し，全体テスト系列 T と最小のスキュー値で圧縮可能な箇所で TP を圧縮する．次に更新された T に逆変換可能なテスト系列が存在すれば，逆変換パターンを生成し，その逆変換パターンで $C(S)$ の未検出故障に対して故障シミュレーションを実行し，検出できた故障を F から削除する．逆変換，故障シミュレーション，未検出故障の削除の一連の処理を T から逆変換可能なテスト系列が存在する限り繰り返す（関数 Reverse_transform）．

次に F の中に ATPG 未処理故障がなくなったとき，$TBUF$ が空になるまで以下の処理を繰り返す．まず $TBUF$ から 1 個のテストパターン tp を取り出す．次に tp 中の X に 0 または 1 を設定し，$C(S)$ の未検出故障に対して故障シミュレーションを実行する．検出できた故障を F から削除する．次に tp を S のテスト系列 TP に変換し，全体テスト系列 T と最小のスキュー値で圧縮可能な箇所で TP を圧縮する．次に

```
Test_generation_acyclic(S)
{
    Generate a time expansion model C(S) corresponding to S.
    for (undetected fault f ∈ F)
    {
        Select a fault f from undetected fault set F.
        Generate a test pattern v for f in C(S).
        if ((tp = Test_patten_compaction(v)) != NULL)
        {
            Set 0s and/or 1s to Xs in a test pattern tp.
            Perform fault simulation for C(S) with tp.
            Remove detected faults from F.
            Transform tp into a test sequence TP.
            Compact TP with a whole test sequence T for S.
            Reverse_transform();
        }
    }
    While(a test pattern compaction buffer TBUF is not empty)
    {
        Take a test pattern tp from TBUF.
        Set 0s and/or 1s to Xs in tp.
        Perform fault simulation for C(S) with tp.
        Remove detected faults from F.
        Transform tp into a test sequence TP.
        Compact TP with a whole test sequence T for S.
        Reverse_transform();
    }
}
```

図 4 時間展開モデルを用いたテスト系列生成アルゴリズム
Fig. 4 Test sequence generation algorithm using a time expansion model.

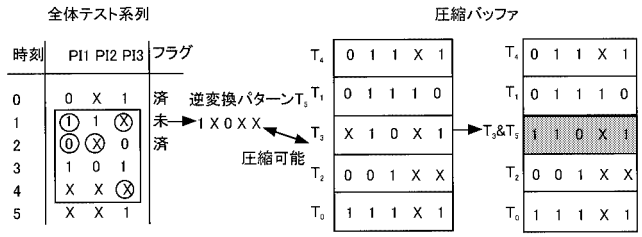

図 5 逆変換パターンと ATPG パターンの動的圧縮
Fig. 5 Dynamic compaction by a reversely transoformed pattern and an ATPG pattern.

上記の関数 Reverse_transform と同様の処理を行う．

ここで，*TBUF* を用いた時間展開モデルの動的テストパターン圧縮方法（関数 Test_pattern_compaction）のアルゴリズムを簡単に説明する．*TBUF* の長さを L とし，*TBUF* 中の i 番目のテストパターンを $TBUF(i)$ と表記する $(0 \leq i < L)$．*TBUF* 中には最大 L 個のテストパターンが存在する．テストパターンは X の数が多い順にソートされている．関数の引数であるテストパターン v が $TBUF(i)$ と圧縮可能であるか否かを判定し，圧縮可能であれば v と $TBUF(i)$ を圧縮し $TBUF(i)$ を更新する．*TBUF* 中のどのテストパターンとも v が圧縮不可能である場合，*TBUF* 中のテストパターン数が L 未満であれば，*TBUF* に v を加える．もし *TBUF* 中のテストパターン数が L 個であるならば，$TBUF(L-1)$ と v の X の数を比較し，v の X の数が多ければ $TBUF(L-1)$ を出力し，v を $TBUF(L-1)$ に加える．v の X の数が少なければ v を *TBUF* から出力されたテストパターンとする．

4. 逆変換パターンと ATPG パターンによる動的テスト系列圧縮方法

4.1 動的テスト系列圧縮方法の基本概念

ここでは，逆変換パターン中に存在する X の値の決定方法，すなわち図 4 の未検出故障集合 F に ATPG 未処理の故障が存在するときの関数 Reverse_transform() のアルゴリズムについて述べる．文献 10) では，逆変換パターンの X の部分にランダムに 0 または 1 を設定していた．逆変換パターンは，ある故障を検出することを目的に生成した決定的パターンではないランダムパターンであるので，未検出故障数が比較的少数になるとほとんど故障を検出しなくなる可能性がある．

一方，時間展開モデルは組合せ回路であるので，時間展開モデルのテストパターンは圧縮バッファを用いて動的に圧縮することができる（図 4 の関数 Test_pattern_compaction に相当）．圧縮バッファには，ATPG パターンが圧縮されている．ここで本章で用いる図（**図 5**，**図 6**）は外部入力 $PI1(0)$，$PI3(0)$，$PI1(1)$，$PI2(1)$，$PI3(3)$ を持つ，基本テンプレート長 4 の時間展開モデルを対象にしている．また圧縮バッファ中のテストパターンの値は左から $PI1(0)$，$PI3(0)$，$PI1(1)$，$PI2(1)$，$PI3(3)$ の順に並んでいる．

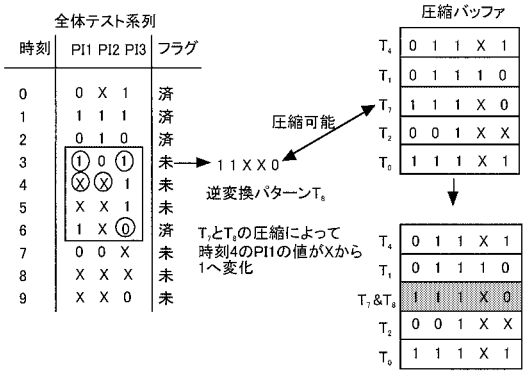

図 6 動的圧縮の解析
Fig. 6 Analysis of dynamic compaction.

逆変換パターンをランダムパターンから決定的パターンにすることによって，より多くの故障検出を行うことを可能にするための方法を提案する．圧縮バッファ中に存在するテストパターンは ATPG パターンであり，ATPG パターンは決定的パターンである．そこで，逆変換パターンと圧縮バッファに存在する ATPG パターンを圧縮し，逆変換パターンをランダムパターンから決定的パターンにすることを考える．図 5 を用いて逆変換パターンと ATPG パターンによる動的テスト系列圧縮方法を説明する．図 5 の逆変換可能である時刻 1〜時刻 4 のテスト系列を逆変換し，逆変換パターン $T_5(1, X, 0, X, X)$ を生成する．次に T_5 と圧縮バッファの中に存在する ATPG パターンとの圧縮を試みる．図 5 の場合，T_3 と T_5 が圧縮可能であるので，T_3 と T_5 を圧縮し T_3 を $T_3\&T_5$ に更新する．$T_3\&T_5$ は逆変換パターンと ATPG パターンを含んいることから決定的パターンとなる．以後，逆変換パターンと ATPG パターンを圧縮してできたテストパターンを**逆変換 ATPG パターン**と呼ぶ．逆変換パ

ターンを含まない ATPG パターンを単に ATPG パターンと呼ぶ．T_3&T_5 は逆変換 ATPG パターンであるので，テスト系列変換された後は必ず全体テスト系列の時刻 1〜時刻 4 のテスト系列と圧縮されなければならない．

4.2 圧縮の効率向上のための提案方法

次に圧縮の効率を向上させるために，逆変換パターンと圧縮バッファに存在する ATPG パターンとの圧縮アルゴリズムに関して，次の 3 つの場合についてその処理方法の解析を行う．

（解析 1） 新たな ATPG パターンが圧縮バッファ中の全テストパターンと圧縮不可能であるときの圧縮バッファから出力させるテストパターンの選択方法

図 5 を用いて解析を行う．新たな ATPG パターン T_6 が生成され，圧縮バッファ中の ATPG パターンと圧縮を試みるが，どのテストパターンとも圧縮不可能であるとする．このとき，圧縮バッファから逆変換 ATPG パターン (T_3&T_5) を出力し，T_6 を圧縮バッファに加える．なぜならば，逆変換 ATPG パターン (T_3&T_5) は，すでに存在する全体テスト系列の一部（時刻 1〜時刻 4 のテスト系列）であり，そのテスト系列で故障シミュレーションを実行し，未検出故障数を削減することにより，圧縮バッファ中の不要な ATPG パターンを削除できる可能性があるからである．ここで，不要な ATPG パターンとは未検出故障に対して故障シミュレーションを実行しても検出できない ATPG パターンである．たとえば，ある ATPG パターンで検出可能な未検出故障が，ある逆変換 ATPG パターンですべて検出可能である場合，その ATPG パターンは不要なテストパターンとなる．もしその ATPG テストパターンを先に圧縮バッファから出力して，未検出故障に対して故障シミュレーションを実行した場合，逆変換 ATPG パターンで故障シミュレーションを実行していないので，未検出故障を検出できる可能性があり，不要な ATPG パターンとならなくなる．

また逆変換 ATPG パターンでできるだけ多くの未検出故障を検出するために，新たに生成された ATPG パターンは優先的にまず逆変換 ATPG パターンと圧縮を試みる．

（解析 2） 全体テスト系列から逆変可能なテスト系列が複数存在するときの逆変換された複数の逆変換パターンと圧縮バッファ中の ATPG パターンとの圧縮の正当性

図 6 を用いて解析を行う．図 6 において，テスト系列長 10 の全体テスト系列が存在し，また圧縮バッファに 5 個の ATPG パターン $(T_4, T_1, T_7, T_2, T_0)$ が存在する．ここで，逆変換可能である時刻 3〜時刻 6 のテスト系列を逆変換して逆変換パターン $T_8(1, 1, X, X, 0)$ を生成し，圧縮バッファ中の ATPG パターン T_7 と圧縮し，T_7 は逆変換パターン T_8 を含んだ ATPG パターン T_7&T_8 に更新される．ここでほかに逆変換可能であるテスト系列は時刻 4〜時刻 7 のテスト系列と時刻 5〜時刻 8 のテスト系列がある．それぞれのテスト系列を逆変換すると逆変換パターン $T_9(X, 1, X, X, X)$ と $T_{10}(X, 1, 1, X, X)$ が生成される．逆変換 ATPG パターン (T_7&T_8) は，時刻 3〜時刻 6 のテスト系列であるので，T_9，T_{10} は異なる逆変換パターン T_8 を含む ATPG パターンとは圧縮できない．また，T_9 は圧縮バッファ中の T_4 と圧縮可能であるが，逆変換パターン T_8 と T_7 の圧縮の結果から，実際には時刻 4 の $PI1$ の値が X から 1 に変化しているので，T_9 は $1, 1, X, X, X$ となり T_4 とは圧縮できない．このような矛盾を防ぐために，1 つの逆変換パターンが圧縮バッファから出力されて全体テスト系列が更新されるまで，逆変換可能なテスト系列が存在しても逆変換を行わないようにする．

（解析 3） 逆変換パターンが圧縮バッファ中の全 ATPG パターンと圧縮不可能であるときの逆変換パターンの処理方法

逆変換パターン v_1 が圧縮バッファ中の ATPG パターンのいずれとも圧縮不可能であるとき，まだほかに逆変換可能なテスト系列が存在するならば，そのテスト系列を逆変換して逆変換パターン v_2 を生成し（v_1 はすてる），圧縮バッファ中の ATPG パターンと圧縮を試みる．なぜならば，最初に生成した逆変換パターン v_1 の X の部分にランダムに 0 または 1 を設定することで，v_2 の X の数が少なくなる可能性があり，結果として v_2 も圧縮バッファ中のいずれの ATPG パターンとも圧縮が不可能になる可能性があるためである．すべての逆変換パターンが圧縮バッファ中のいずれの ATPG パターンとも圧縮不可能であるとき，再度 1 パターンずつ逆変換パターンを生成し，逆変換パターンの X にランダムに 0 または 1 を設定して故障シミュレーションを実行する．（解析 1）で説明したように圧縮バッファに存在する不要な ATPG パターンを認識可能にするために，ATPG パターンと圧縮できなくても逆変換パターンで故障シミュレーションを実行し，未検出故障数を削減しておく．

5. 全体テスト系列との圧縮の効率化方法

時間展開モデルの未検出故障に対して故障シミュレーションを実行した ATPG パターンは，無閉路順

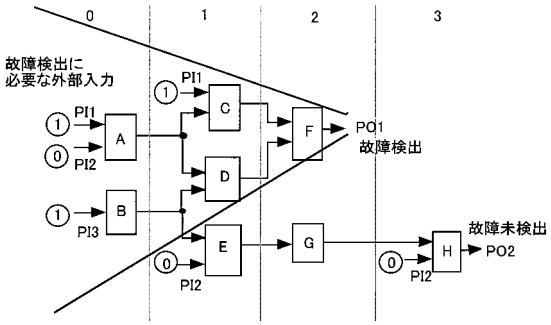

図 7 故障検出に必要な外部出力
Fig. 7 Necessary primary inputs ofr fault detection.

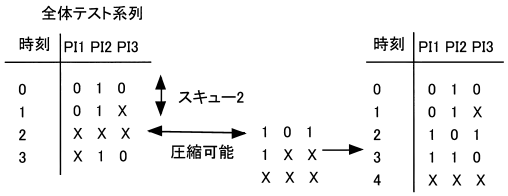

図 8 故障検出情報を参照したテスト系列変換方法
Fig. 8 Test sequence transformation method by referring to fault detection inofomation.

序回路のテスト系列に変換され，すでに存在する全体テスト系列と圧縮される．このとき，最小のスキュー値で圧縮される．テスト系列の圧縮において，X の数が多いほどより効率的な圧縮が可能であると考えられる．本章では，すでに存在する全体テスト系列とできるだけ小さなスキュー値で圧縮できることを目的としたテスト系列変換方法と ATPG パターンの X の値の決定方法を提案する．

5.1 故障検出情報を参照したテスト系列変換方法

圧縮バッファ中のある 1 つの ATPG パターンは複数の目標故障を検出する ATPG パターンが圧縮されている可能性がある．しかしながら，その目標故障のいくつかはすでに他のテストパターンで検出されている可能性があるので，圧縮バッファ中の ATPG パターン中のいくつかの外部入力には故障検出に無関係な値が割り当てられている可能性がある．

図 7 に時間展開モデルを示す．図 7 において，$PI1$〜$PI3$ は外部入力，$PO1$，$PO2$ は外部出力，A〜H は組合せ論理部を示す．テストパターン $(PI1(0), PI2(0), PI3(0), PI1(1), PI2(1), PI2(3)) = (1,0,1,1,0,0)$ で図 7 の時間展開モデルの未検出故障に対して故障シミュレーションを実行した結果，$PO1$ では故障が検出され，$PO2$ では故障が検出されなかったとする．$PO1$ で故障を検出するために必要でありうる外部入力は $PO1$ から到達可能な外部入力 $PI1(0)$，$PI2(0)$，$PI3(0)$，$PI1(1)$ である．$PO2$ では故障を検出しないので，$PO1$ から到達不可能な外部入力 $PI2(1)$，$PI2(3)$ の値と $PO2$ の期待値は必要ないことが分かる．

そこで時間展開モデルの未検出故障に対して故障シミュレーションを実行した後，時間展開モデルの ATPG パターンを無閉路順序回路のテスト系列に変換するときに，時間展開モデルの外部出力での故障検出情報を参照して，故障検出に関係しない外部入力の値を X にしてテスト系列変換を行う方法を提案する．以下に故障検出情報を参照したテスト系列変換方法のアルゴリズムを示す．

ステップ 1 時間展開モデルにおいて，故障を検出している全外部出力から到達可能な外部入力の和集合を求める．

ステップ 2 ステップ 1 で求めた集合の補集合に相当する外部入力の値を X にする．

ステップ 3 時間展開モデルのテストパターンを対応する無閉路順序回路のテスト系列に変換する．

ステップ 4 最大時刻の全外部出力で故障が検出されていないならば，最大時刻のテストパターンを削除する．最大時刻のテストパターンが削除されなくなるまでステップ 4 を繰り返す．

図 8 は図 7 の時間展開モデルの外部出力の故障検出情報を参照してテスト系列変換して全体テスト系列と圧縮する例である．通常のテスト系列変換[4]を行ったテスト系列と全体テスト系列とは圧縮できない（テスト系列長 8）．一方，図 7 の $PO1$ での故障検出に関係しない $PI2(1)$，$PI2(3)$ の値を X にして，テスト系列変換を行うと図 8 に示すような長さ 3 のテスト系列となる．このテスト系列と全体テスト系列はスキュー 2 で圧縮可能であり（テスト系列長 5），全体テスト系列長は短くなる．

5.2 ATPG パターン中の X の値の決定方法

圧縮バッファから出力された ATPG パターンには，X が含まれる可能性がある．全体テスト系列との圧縮を効率化するために，X の値の決定方法を提案する．

図 9 を用いて X の値の決定方法を説明する．図 9 は外部入力 $PI1(0)$，$PI3(0)$，$PI1(1)$，$PI2(1)$，$PI3(3)$ を持つ，基本テンプレート長 4 の時間展開モデルを対象としている．圧縮バッファから出力された ATPG パターン $v_1(X,X,1,0,1)$ を考える．v_1 を X を含んだままテスト系列変換を行うと図 9 のテスト系列 T_1 になる．図 9 に示す全体テスト系列と T_1 の圧縮を試みて，スキュー 3 で圧縮したと仮定すると，T_1 は T_1' となり，T_1' を逆変換すると $v_1'(1,1,1,0,1)$ がで

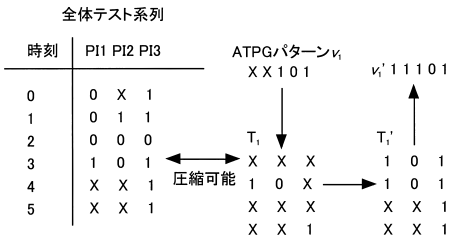

図 9 ATPG パターン中の X の値の決定方法
Fig. 9 X decision method for an ATPG pattern.

きる．この例の場合，v_1 の X の部分の値がすべて決定されたので，時間展開モデルの未検出故障に対して故障シミュレーションを実行する．もしまだ v_1' に X の部分が残るならば，ランダムに 0 または 1 を設定する．実行後の v_1' は無閉路順序回路のテスト系列に変換され，全体テスト系列とスキュー 3 で圧縮できることが保証される．圧縮の結果，全体のテスト系列長は 7 になる．一方 v_1 の X の部分の値にランダムに 0 または 1 を設定したとし，その結果 $v_2(1,0,1,0,1)$ が得られたとすると，v_2 で時間展開モデルの未検出故障に対して故障シミュレーションを実行した後で無閉路順序回路のテスト系列に変換されたテスト系列を T_2 とする．T_2 は全体テスト系列とは圧縮不可能であり，全体のテスト系列長は 10 になる．このように，X を含んだままの ATPG パターンをテスト系列変換して，全体テスト系列と圧縮を試みて，圧縮可能な最小のスキュー値で圧縮したと仮定して得られるテスト系列を逆変換することによって，ATPG パターン中の X の値を決定する．この方法により，全体テスト系列長を短縮できる可能性がある．

6. 実 験 結 果

本章では 4，5 章で提案したテスト系列圧縮方法を実際の回路で評価した結果について報告する．**表 2** は，評価に用いる実際の回路の特性を示す．表 2 において，CN，PI，PO，FF，GATE はそれぞれ回路名，回路の外部入力数，外部出力数，FF 数，ゲート数（ATPG ツールのプリミティブ数）を表す．また SR は回路を無閉路順序回路にするために必要である回路中の FF 数に対するスキャン FF 数の割合（単位%），SD は回路の最大順序深度を表す．TEM は時間展開モデルのゲート数（ATPG ツールのプリミティブ数）を表す．時間展開モデルは文献 4) に示すヒューリスティックアルゴリズムを用いて回路の順序深度 +1 のラベル数を持つゲート数最小化を指向して生成された組合せ回路である．実験に用いた計算機は UltraUA30（動作速度：296 MHz，SPECint95：12.1，SPECfp：18.3）である．ATPG ツールは社内の多重故障対応の組合せ ATPG[4] を用いた．また実験に使用する全回路について 99%以上の故障検出効率を達成し，どの評価モードでも故障検出効率はほぼ一定である．ATPG 時間とテスト系列長を考慮した結果，圧縮バッファ長の値は 100 が適当であったので，その結果について述べる．

表 2 回路特性
Table 2 Properties of circuits.

CN	PI	PO	FF	GATE	SR	SD	TEM
#1	81	22	502	16618	55.6	7	26926
#2	342	11	1550	21190	48.8	14	31359
#3	336	376	1710	31095	48.2	5	55756
#4	513	48	2381	34731	5.4	45	165151
#5	515	556	5363	67329	32.8	9	90879

表 3 R モード実験結果
Table 3 Experimantal results of mode R.

CN	R-TL	R-CT	R-D
#1	5366	1108	4.0
#2	842	292	15.3
#3	2178	1618	13.1
#4	3194	1735	5.9
#5	12460	3839	4.1

表 4 D モード実験結果
Table 4 Experimantal results of mode D.

CN	D-TL	D-CT	D-D	CR
#1	4365	1180	4.8 (15.1)	9.7
#2	675	352	19.5 (424.6)	4.8
#3	2124	1664	13.7 (101.9)	3.8
#4	1529	1454	12.7 (61.5)	5.5
#5	6810	2951	8.4 (17.0)	11.2

6.1 D モードの評価

表 3，**表 4** に逆変換パターンと ATPG パターンによる動的テスト系列圧縮方法を評価した実験結果を示す．表 3 において，R-TL，R-CT，R-D はそれぞれ，逆変換パターン，ATPG パターン中の X にランダムに 0 または 1 を設定し，故障検出情報を参照しないでテスト系列変換を行うモード（R モード）を適用したときのテスト系列長，全体のテスト系列生成時間（単位：秒），逆変換パターン 1 つあたりの平均検出故障数である．

表 4 において D-TL，D-CT，D-D，CR はそれぞれ，逆変換パターンと ATPG パターンによる動的テスト系列圧縮を行い，ATPG パターン中の X にランダムに 0 または 1 を設定し，故障検出情報を参照しないでテスト系列変換を行うモード（D モード）を適用したときのテスト系列長，全体のテスト系列生成時間

（単位：秒），逆変換パターン 1 つあたりの平均検出故障数（括弧内数字は逆変換 ATPG パターン 1 つあたりの平均検出故障数），すべての逆変換パターン数に対する逆変換 ATPG パターン数の割合（単位：%）である．

逆変換パターンと圧縮バッファ中に存在する ATPG パターンを圧縮することによって，表 3，表 4 に示すように R モードのテスト系列長を 2〜52%短縮することができた．効果の高い#4，#5 は D モードの逆変換パターン 1 つあたりの検出故障数が R モードの 2 倍以上増加し，逆変換 ATPG パターンの割合も比較的高いことが分かる．#1 は#4 よりも，逆変換 ATPG パターンの割合は高いが，逆変換 ATPG パターン 1 つあたりの平均検出故障数が#4 の 25%と少なくなるため効果は#4 ほど高くないと考えられる．#4，#5 について，生成したテスト系列長が短くなったことによる故障シミュレーションの短縮時間が，逆変換パターンと ATPG パターンの圧縮にかかる時間よりも大きいため，テスト系列生成時間は R モードより 13〜30%速くなっている．その他の回路は R モードとほぼ同程度の時間であった．

6.2 T モードの評価

表 3，**表 5** に全体テスト系列との圧縮効率化方法を評価し，R モードと比較した実験結果を示す．表 5 において，T-TL，T-CT，XR，XDR はそれぞれ，逆変換パターン中の X にランダムに 0 または 1 を設定し，ATPG パターン中の X に対しては 5.2 節で提案した方法を用いて値を決定し，故障検出情報を参照してテスト系列変換を行うモード（T モード）を適用したときのテスト系列長，全体のテスト系列生成時間（単位：秒），1 つの ATPG パターンのテスト系列変換において，時間展開モデルの全外部入力数に対するテスト系列変換時に X にできた外部入力数の割合の平均値（単位：%），時間展開モデルの全外部入力数に対する 1 つの ATPG パターン中の X についてランダムに 0 または 1 を割り当てなかった X の数の割合の平均値（単位：%）である．表 3，表 5 から全体テスト系列との圧縮の効率化を行うことで，テスト系列長を R モードの 24〜64%短縮することができ，特に圧縮効果が高かった#2，#5 は XR，XDR の両方の値が比較的高いことが分かる．XR の値は故障検出情報を参照してテスト系列変換を行う方法の効果を示し，XDR の値は 5.2 節で提案した方法の効果を示している．#1 は XR の値が高く，XDR の値が低いので，さらに解析を行った．#1 に関しては故障検出情報を参照してテスト系列変換を行う方法のみを適用した場合，テスト系列長は 2804（48%削減）となり，XR が高いために圧縮の効率が高くなってると考えられ，5.2 節で提案した方法のみを適用した結果，テスト系列長は 4742（12%削減）となり，XDR の値が低いため圧縮の効率が低いと考えられる．テスト系列生成時間については，故障検出情報の参照する処理時間のオーバヘッドと ATPG パターンの X の値を決定する処理時間のオーバヘッドが，テスト系列長短縮のための故障シミュレーション時間の短縮分よりも大きいと考えられ，#2，#4 を除く回路で全体のテスト系列生成時間が R モードの 10〜34%増加した．

表 5 T モード実験結果
Table 5 Experimental results of model T.

CN	T-TL	T-CT	XR	XDR
#1	2901	1486	83	0.5
#2	306	276	85	10.5
#3	1654	1828	93	5.1
#4	1531	1753	50	36.0
#5	6218	4219	92	8.1

表 6 D モードと T モードの組合せ実験結果
Table 6 Experimental results of combination of mode D and T.

CN	DT-TL	DT-CT
#1	2781	1461
#2	292	340
#3	1439	1769
#4	1251	1621
#5	4089	3302

6.3 D モードと T モードを組み合わせた評価

表 3，**表 6** は D モードと T モードを組み合わせて評価し，R モードと比較した実験結果を示している．表 6 において，DT-TL，DT-CT はそれぞれ逆変換パターンと ATPG パターンによる動的テスト系列圧縮を行い，ATPG パターン中の X に対しては 5.2 節で提案した方法を用いて値を決定し，故障検出情報を参照してテスト系列変換を行うモードを適用したときの，テスト系列長と全体のテスト系列生成時間（単位：秒）である．表 6 からモード D とモード T を組み合わせることにより，さらにテスト系列長を短縮でき，全体のテスト系列長を R モードの 34〜67%短縮することができた．テスト系列生成時間については，圧縮効果が高くかつ R モードのテスト系列長が比較的長い#4，#5 はそれぞれ R モードに比べて 7%，14%高速化された．その他の回路については R モードの 9〜32%増加した．

7. おわりに

本論文では，逆変換パターンと ATPG パターンによる動的テスト系列圧縮方法と，故障検出情報の参照によるテスト系列変換方法および ATPG パターンの X の値の決定方法を提案した．実験結果から，以下の効果を得ることができ，提案方法の有効性を示すことができた．

(1) 逆変換パターンと ATPG パターンによる動的テスト系列圧縮方法を適用することで，テスト系列長を 2〜52%削減できた．

(2) 故障検出情報を参照したテスト系列変換方法と ATPG パターンの X の値の決定方法を適用することで，テスト系列長を 24〜64%削減することができた．

(3) (1) と (2) の方法を組み合わせて適用することで，テスト系列長を 34〜67%削減することができた．

今後の課題として，同じ無閉路順序回路に対する複数の時間展開モデルとテスト系列圧縮方法の効果の関係について考察することがあげられる．

謝辞 本研究に対して貴重なコメントをいただきました広島市立大学の井上智生助教授に深く感謝いたします．

参考文献

1) Abramovici, M., Breuer, M.A. and Friedman, A.D.: *Digital Systems Testing and Testable Design*, Computer Science Press (1990).
2) Gupta, R., Gupta, R. and Breuer, M.A.: The BALLAST methodology for structured partial scan design, *IEEE Trans. Comput.*, Vol.39, No.4, pp.538–544 (1990).
3) Takasaki, T., Inoue, T. and Fujiwara, H.: Partial scan design methods based on internally balanced structure, *IEEE Proc. Asia and South Pacific Design Automation Conf.*, pp.211–216 (Feb. 1998).
4) Inoue, T., Hosokawa, T., Mihara, T. and Fujiwara, H.: An optimal time expansion model based on combinational ATPG for RT level circuits, *IEEE Proc. Asian Test Symp.*, pp.190–197 (Dec. 1998).
5) Kunzmann, A. and Wunderlich, J.J.: An analytical approach to the partial scan problem, *Journal of Electronic Testing Theory and Applications*, Vol.1, No.2, pp.163–174 (1990).
6) Goel, P. and Rosales, B.C.: Test generation and dynamic compaction of tests, *IEEE Proc. Int. Test Conf.*, pp.189–192 (Oct. 1979).
7) Kajihara, S., Pomeranz, I., Kinoshita, K. and Reddy, S.M.: Cost- effective generation of minimal test sets for stuck-at faults in combinational logic circuits, *IEEE Trans. Computer-Aided Design*, pp.1496–1504 (Dec. 1995).
8) Pomeranz and Reddy, S.M.: Vector restoration based static compaction of test sequences for synchronous sequential circuits, *IEEE Proc. Int. Conf. on Computer Design*, pp.360–365 (Oct. 1997).
9) Niermann, T.M., Roy, R.K., Patel, J.H. and Abraham, J.A.: Test Compaction for Sequential Circuits, *IEEE Trans. Computer Aided Design*, Vol.11, No.2, pp.260–267 (1992).
10) 細川，井上，平岡，藤原：時間展開モデルを用いた無閉路順序回路のテスト系列圧縮方法，電子通信学会論文誌（D-1），Vol.J82-D-1, No.7, pp.869–878 (1999).

(平成 11 年 9 月 22 日受付)
(平成 12 年 2 月 4 日採録)

細川 利典（正会員）
昭和 39 年生．昭和 62 年明治大学工学部電子通信工学科卒業．同年松下電器産業（株）入社．論理シミュレーションエンジン，テストパターン生成，故障シミュレーション，テスト容易化設計，上流テストの研究開発に従事．

吉村 正義
昭和 48 年生．平成 10 年大阪大学大学院基礎工学研究科物理系専攻システム工学分野修士課程修了．同年松下電器産業（株）入社．テスト容易化設計の研究開発に従事．

太田 光保
昭和 37 年生．昭和 61 年広島大学工学部第 2 類卒業．同年松下電器産業（株）入社．IP ベーステスト，テストパターン生成，テスト容易化設計の研究開発に従事．