(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第3672546号
(P3672546)
(45)発行日　平成17年7月20日(2005.7.20)　(24)登録日　平成17年4月28日(2005.4.28)

(51)Int.Cl.[7]	F I		
G01R 31/3183	G01R	31/28	Q
G01R 31/28	G06F	17/50	670G
G06F 17/50	G01R	31/28	F

請求項の数 18　(全 23 頁)

(21)出願番号　特願2002-265519(P2002-265519)
(22)出願日　平成14年9月11日(2002.9.11)
(65)公開番号　特開2004-101419(P2004-101419A)
(43)公開日　平成16年4月2日(2004.4.2)
審査請求日　平成14年9月11日(2002.9.11)

(73)特許権者　396023993
株式会社半導体理工学研究センター
神奈川県横浜市港北区新横浜３丁目１７番地２　友泉新横浜ビル６階
(74)代理人　100077517
弁理士　石田　敬
(74)代理人　100092624
弁理士　鶴田　準一
(74)代理人　100100871
弁理士　土屋　繁
(74)代理人　100082898
弁理士　西山　雅也
(74)代理人　100081330
弁理士　樋口　外治

最終頁に続く

(54)【発明の名称】テストパターン発生器における最適初期値の決定方法および装置

(57)【特許請求の範囲】
【請求項１】
集積回路テストのためにテストパターン発生器に入力する最適初期値を決定するための方法であって、
ａ）任意に与えられた第１の初期値を用いて故障シミュレーションおよび逆順故障シミュレーションを行って、予め決められたｎ個の故障を検出するための第１のテスト長および第１の極小テスト長を求めるステップと、
ｂ）前記第１の初期値、第１のテスト長および第１の極小テスト長に基づいて前記第１の極小テスト長よりも短いテスト長を得ることが可能な第２の初期値を算出するステップと、
ｃ）前記算出された第２の初期値を用いて第２の故障シミュレーションを行って前記ｎ個の故障を検出するための第２のテスト長を求めるステップと、
ｄ）前記第２のテスト長と前記第１の極小テスト長を比較し、前記第２のテスト長が前記第１の極小テスト長と同じか長い場合は前記第１の極小テスト長を第２の極小テスト長とみなし、前記第２のテスト長が前記第１の極小テスト長より短い場合は、前記第２の故障シミュレーションに対して逆順故障シミュレーションを行って第２の極小テスト長を求めるステップと、
ｅ）前記第２の初期値、第２のテスト長および第２の極小テスト長を前記第１の初期値、第１のテスト長および第１の極小テスト長とみなして前記ステップｂ）、ｃ）およびｄ）を繰り返して最小テスト長を求めるステップ、

を具備し、前記ステップｅ）で求めた最小テスト長を提示する初期値を前記集積回路のテストにあたって前記テストパターン発生器における初期値として決定する事を特徴とする、テストパターン発生器における最適初期値の決定方法。

【請求項２】

前記ステップｅ）において、前記ステップｂ）、ｃ）およびｄ）は、前記極小テスト長が予め決められた値に達するまで繰り返されるものであることを特徴とする、請求項１に記載の集積回路テストにおける最適初期値決定方法。

【請求項３】

前記ステップｅ）において、前記ステップｂ）、ｃ）およびｄ）は、前記第１または第２の初期値が予め選定されたテストパターン系列の最終値に到達するまで繰り返されることを特徴とする、請求項１に記載のテストパターン発生器における最適初期値の決定方法。

【請求項４】

前記第１の初期値をテストパターン系列のｉ番目の値、第１のテスト長をＬ、第１の極小テスト長をＬｍ、前記ステップｂ）における第２の初期値をテストパターン系列のｉ２番目とする場合、ｉ２は、

ｉ２＝ｉ＋Ｌ－Ｌｍ＋１

として算出されることを特徴とする、請求項１乃至３の何れか１項に記載の、テストパターン発生器における最適初期値の決定方法。

【請求項５】

前記テストパターン発生器は、線形帰還シフトレジスタを用いたテスト／クロック構成を有することを特徴とする、請求項１乃至４の何れか１項に記載の、テストパターン発生器における最適初期値の決定方法。

【請求項６】

前記テストパターン発生器は、線形帰還シフトレジスタとスキャンチェーンを組み合わせたテスト／スキャン構成を有することを特徴とする、請求項１乃至４の何れか１項に記載の、テストパターン発生器における最適初期値の決定方法。

【請求項７】

被テスト回路と、テストパターン系列を発生して前記被テスト回路に出力するテストパターン発生器と、前記被テスト回路からのテスト結果を検出するテスト応答圧縮器とを１チップ内に組み込んだ集積回路のテスト装置において、前記テストパターン発生器は、

ａ）任意に与えられた第１の初期値を用いて故障シミュレーションおよび逆順故障シミュレーションを行って、予め決められたｎ個の故障を検出するための第１のテスト長および第１の極小テスト長を求めるステップと、

ｂ）前記第１の初期値、第１のテスト長および第１の極小テスト長に基づいて前記第１の極小テスト長よりも短いテスト長を得ることが可能な第２の初期値を算出するステップと、

ｃ）前記算出された第２の初期値を用いて第２の故障シミュレーションを行って前記ｎ個の故障を検出するための第２のテスト長を求めるステップと、

ｄ）前記第２のテスト長と前記第１の極小テスト長を比較し、前記第２のテスト長が前記第１の極小テスト長と同じか長い場合は前記第１の極小テスト長を第２の極小テスト長とみなし、前記第２のテスト長が前記第１の極小テスト長より短い場合は、前記第２の故障シミュレーションに対して逆順故障シミュレーションを行って第２の極小テスト長を求めるステップと、

ｅ）前記第２の初期値、第２のテスト長および第２の極小テスト長を前記第１の初期値、第１のテスト長および第１の極小テスト長とみなして前記ステップｂ）、ｃ）およびｄ）を繰り返して最小テスト長を求めるステップ、

を実行することにより求めた、前記最小テスト長を提示する初期値を備える事を特徴とする、集積回路のテスト装置。

【請求項８】

前記テストパターン発生器は、線形帰還シフトレジスタを用いたテスト／クロック構成で

あることを特徴とする、請求項７に記載の集積回路のテスト装置。
【請求項９】
前記テストパターン発生器は、線形帰還シフトレジスタとスキャンチェーンを組み合わせたテスト／スキャン構成であることを特徴とする、請求項７に記載の集積回路のテスト装置。
【請求項１０】
集積回路テストのためにテストパターン発生器に入力する最適初期値を決定するための方法であって、
ａ）任意に与えられた第１の初期値を用いて第１の故障シミュレーションを行って、予め決められた目標テスト長L_{fix}において検出される第１の故障数ｎを求めるステップと、
ｂ）前記第１の故障シミュレーションに対して逆順故障シミュレーションを行って前記第１の故障数ｎを検出するのに必要な第１の極小テスト長を求めるステップと、
ｃ）前記第１の初期値、前記目標テスト長L_{fix}および第１の極小テスト長に基づいて、前記第１の極小テスト長を提示する第２の初期値を算出するステップと、
ｄ）前記算出された第２の初期値を用いて第２の故障シミュレーションを行って前記目標テスト長L_{fix}において検出される第２の故障数ｎｔを求めるためのステップと、
ｅ）前記第１および第２の故障数ｎおよびｎｔを比較し、ｎｔ≦ｎの場合は前記第１の故障数ｎを極大の検出故障数とし、ｎｔ＞ｎの場合、ｎｔ≦ｎと成るまで前記ステップｂ）～ｄ）を繰り返して極大の検出故障数を求めるステップと、
ｆ）前記極大の故障検出数を提示する初期値の、テストパターン系列における次の初期値を用いて、前記極大の検出故障数を検出するために必要なテスト長Ｌを第３の故障シミュレーションによって求めるステップと、
ｇ）前記テスト長Ｌを前記目標のテスト長L_{fix}と比較し、$L \geqq L_{fix}$の場合、前記第３の故障シミュレーションの初期値、テスト長Ｌおよび目標テスト長L_{fix}に基づいて、次に前記目標テスト長L_{fix}よりも短いテスト長の得られる可能性のある初期値を算出するステップと、
ｈ）前記ステップｇ）で算出された初期値に対して、$L < L_{fix}$と成るまで前記ステップｆ）およびｇ）を繰り返すステップと、
ｉ）前記ステップｇ）または前記ステップｈ）において$L < L_{fix}$を提示する初期値を前記第１の初期値とみなして前記ステップａ）～ｈ）を繰り返し、前記目標テスト長において検出される最大の故障数を得るためのステップ、
を具備し、前記ステップｉ）で求めた最大故障検出数を提示する初期値を前記集積回路のテストにあたって前記テストパターン発生器における初期値として決定する事を特徴とする、テストパターン発生器における最適初期値の決定方法。
【請求項１１】
前記ステップｉ）において、前記ステップａ）～ｈ）は、検出された故障数が予め決められた望ましい値となるまで繰り返されるものであることを特徴とする、請求項１０に記載の集積回路テストにおける最適初期値の決定方法。
【請求項１２】
前記ステップｉ）において、前記ステップａ）～ｈ）は、前記何れかの初期値が予め選定されたテストパターン系列の最終値に到達するまで繰り返されることを特徴とする、請求項１０に記載のテストパターン発生器における最適初期値の決定方法。
【請求項１３】
前記第１の初期値をテストパターン系列のｉ番目の値、予め決められた目標テスト長をL_{fix}、第１の極小テスト長をＬｍ、前記ステップｃ）における第２の初期値をテストパターン系列のｉ２番目の値とする場合、ｉ２は、
$i2 = i + L_{fix} - Lm$
として算出されることを特徴とする、請求項１０乃至１２の何れか１項に記載の、テストパターン発生器における最適初期値の決定方法。

【請求項１４】
前記テストパターン発生器は、線形帰還シフトレジスタを用いたテスト／クロック構成を有することを特徴とする、請求項１０乃至１３に記載の、テストパターン発生器における最適初期値の決定方法。
【請求項１５】
前記テストパターン発生器は、線形帰還シフトレジスタとスキャンチェーンを組み合わせたテスト／スキャン構成を有することを特徴とする、請求項１０乃至１３に記載の、テストパターン発生器における最適初期値の決定方法。
【請求項１６】
被テスト回路と、テストパターン系列を発生して前記被テスト回路に出力するテストパターン発生器と、前記被テスト回路からのテスト結果を検出するテスト応答圧縮器とを１チップ内に組み込んだ集積回路のテスト装置において、前記テストパターン発生器は、
ａ）任意に与えられた第１の初期値を用いて第１の故障シミュレーションを行って、予め決められた目標テスト長L_{fix}において検出される第１の故障数ｎを求めるステップと、
ｂ）前記第１の故障シミュレーションに対して逆順故障シミュレーションを行って前記第１の故障数ｎを検出するのに必要な第１の極小テスト長を求めるステップと、
ｃ）前記第１の初期値、前記目標テスト長L_{fix}および第１の極小テスト長に基づいて、前記第１の極小テスト長を提示する第２の初期値を算出するステップと、
ｄ）前記算出された第２の初期値を用いて第２の故障シミュレーションを行って前記目標テスト長L_{fix}において検出される第２の故障数ｎｔを求めるためのステップと、
ｅ）前記第１および第２の故障数ｎおよびｎｔを比較し、ｎｔ≦ｎの場合は前記第１の故障数ｎを極大の検出故障数とし、ｎｔ＞ｎの場合、ｎｔ≦ｎと成るまで前記ステップｂ）～ｄ）を繰り返して極大の検出故障数を求めるステップと、
ｆ）前記極大の故障検出数を提示する初期値の、テストパターン系列における次の初期値を用いて、前記極大の検出故障数を検出するために必要なテスト長Ｌを第３の故障シミュレーションによって求めるステップと、
ｇ）前記テスト長Ｌを前記目標のテスト長L_{fix}と比較し、$L \geqq L_{fix}$の場合、前記第３の故障シミュレーションの初期値、テスト長Ｌおよび目標テスト長L_{fix}に基づいて、次に前記目標テスト長L_{fix}よりも短いテスト長の得られる可能性のある初期値を算出するステップと、
ｈ）前記ステップｇ）で算出された初期値に対して、$L < L_{fix}$と成るまで前記ステップｆ）およびｇ）を繰り返すステップと、
ｉ）前記ステップｇ）または前記ステップｈ）において$L < L_{fix}$を提示する初期値を前記第１の初期値とみなして前記ステップａ）～ｈ）を繰り返し、前記目標テスト長において検出される最大の故障数を得るためのステップ、
を実行することにより求めた、前記最大故障数を提示する初期値を備える事を特徴とする、集積回路のテスト装置。
【請求項１７】
前記テストパターン発生器は、線形帰還シフトレジスタを用いたテスト／クロック構成を有することを特徴とする、請求項１６に記載の、集積回路のテスト装置。
【請求項１８】
前記テストパターン発生器は、線形帰還シフトレジスタとスキャンチェーンを組み合わせたテスト／スキャン構成を有することを特徴とする、請求項１６に記載の、集積回路のテスト装置。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】
本発明は、大規模集積回路（ＶＬＳＩ）を含めた集積回路のテストにあたって、被テスト回路に印加するテストパターンを発生するためのテストパターン発生器において、効率的

にテストを行うための最適な初期値を決定する方法に関する。

【０００２】

【従来の技術】

半導体プロセスの微細化に従い、集積回路（ＬＳＩ、ＶＬＳＩを含む）のテストに対するコストが増加してきている。このままＬＳＩの動作速度や集積度が上昇すると、現在よりも高価なテスタを長時間使用しなければならなくなる。高価なテスタの使用時間を減らす為に、スキャン設計方式や組込み自己テスト（Ｂｕｉｌｔ－Ｉｎ・Ｓｅｌｆ－Ｔｅｓｔ）方式（以下、ＢＩＳＴ方式）に代表されるテスト容易化設計によってテスタの負担を軽くする手法が研究されている。

【０００３】

ＢＩＳＴ方式は、テストパターン発生器（Ｔｅｓｔ・Ｐａｔｔｅｒｎ・Ｇｅｎｅｒａｔｏｒ、以下、ＴＰＧ）とテスト応答圧縮器（Ｔｅｓｔ・Ｒｅｓｐｏｎｓｅ・Ｃｏｍｐａｃｔｏｒ、以下、ＴＲＣ）、を被テスト回路と同一チップ上に集積しテストを行う。これによりチップ外部のテスタの負担を軽減することができる。ＴＰＧとして線形帰還シフトレジスタ（Ｌｉｎｅａｒ・Ｆｅｅｄｂａｃｋ・Ｓｈｉｆｔ・Ｒｅｇｉｓｔｅｒ、以下、ＬＦＳＲ）が、ＴＲＣとしてマルチプルインプットシグネチャーレジスタ（Ｍｕｌｔｉｐｌｅ・Ｉｎｐｕｔ・Ｓｉｇｎａｔｕｒｅ・Ｒｅｇｉｓｔｅｒ、以下、ＭＩＳＲ）が使われることが多い。

【０００４】

ＬＦＳＲを使った擬似ランダムパターン発生器は、テスタを使わずにテストパターンを印加することができるが、９５％を超えるような高い故障カバレージに到達するためには、大量のテストパターンを必要とし、その結果テストに要する時間、すなわちテスト長が長くなってしまう欠点がある。この欠点を克服する試みが数多く報告されている。重み付き擬似乱数発生手法は、被テスト回路（Ｃｉｒｃｕｉｔ・Ｕｎｄｅｒ・Ｔｅｓｔ、以下ＣＵＴ）に与えるテストパターンの各ビットに重み付けを行う方法であり、より多くの故障を検出できるように、“０”もしくは“１”の発生する確率を変える手法である。（例えば、非特許文献１参照。）

一方、観測点、検査点を挿入する方法は、観測点や検査点を被テスト回路に組み込み、可観測性、可制御性を改善しテスト品質を向上させる手法である。この方法はＣＵＴに観測点や検査点を挿入するため、ＣＵＴの回路構造を変更することになり、遅延などを変える事になるという欠点がある。（例えば、非特許文献２参照。）

上記の手法だけでは限られたテスト長で十分な故障カバレージを得ることが困難である。そこで擬似乱数とＡＴＰＧ（Ａｕｔｏｍａｔｉｃ・Ｔｅｓｔ－Ｐａｔｔｅｒｎ・Ｇｅｎｅｒａｔｉｏｎ）ベクトルを組み合わせるＢＩＳＴも研究されている。（例えば、非特許文献３、４および５参照。）

テストキューブのドントケアを利用してＬＦＳＲ用のシードとして符号化、これをチップ内のＬＦＳＲで復号化してテストを行う方法が研究されている。リシーディング法では、（テストキューブの特定されたビット数＋２０）ビットの長さのＬＦＳＲを使い、テストキューブの符号化を行っている。

【０００５】

また複数の帰還多項式を持つＬＦＳＲをテストキューブの復号化に使う方法としてＭＰ－ＬＦＳＲがある。この方法は（テストキューブの特定されたビット数＋４）ビットのＭＰ－ＬＦＳＲによって、テストキューブの符号化を行っている。（例えば、非特許文献６参照。）

ビットフリップＢＩＳＴ、およびビットフィックスＢＩＳＴでは、ＬＦＳＲの出力するテストパターンの中で、テストキューブに近いものを探し、一部のビットを反転、あるいは固定させることで、テストキューブをＣＵＴに与える方法である。この方法はビットを反転、もしくは固定させる回路が必要となる。

【０００６】

リシーディング法、ビットフリップＢＩＳＴ、ビットフィックスＢＩＳＴを使ったＢＩＳ

Ｔでは、まず検出の容易な故障を擬似ランダムテストによって検出してから、ＡＴＰＧベクトルのリシードを行っている。これらの方法は、まず例えば１万パターン程度の擬似ランダムテストを行う。そして擬似ランダムテストによって検出できなかった残存故障に対して、ＡＴＰＧベクトルを印加する。このテストベクトルの追加は、テスタもしくは、ＢＩＳＴでは内蔵したＲＯＭからＡＴＰＧベクトルを与える必要があり、残存故障が多くなるとテストコストを増大させてしまう。

【０００７】

ビットフリップＢＩＳＴやビットフィックスＢＩＳＴ用回路のハードウエア量を減らすためには、擬似ランダムテストの品質を上げ、残存故障数を減らすことが望ましい。残存故障が少なくなると、テスタを使う場合は必要となるテスタのストレージ容量が減り、ＢＩＳＴでは内蔵するＲＯＭの規模が軽くなる。

【０００８】

擬似ランダムテストの品質を上げるために、シードの選択によって擬似ランダムテストの品質を上げる方法が考えられる。あらかじめ求めたテストキューブを利用する方法が研究されている。これはＬＦＳＲの出力するテストパターンとテストキューブのハミング距離によってシードのテスト品質を評価している。これにより、妥当な計算量によって高いテスト品質のシードを得ている。（例えば、非特許文献７～１０参照。）

【０００９】

【非特許文献１】

Ｒ．Ｋａｐｕｒ，Ｓ．Ｐａｔｉｌ，Ｔ．Ｊ．Ｓｎｅｔｈｅｎ，Ｔ．Ｗ．Ｗｉｌｌｉａｍｓ，“Ｄｅｓｉｇｎ・ｏｆ・ａｎ・Ｅｆｆｉｃｉｅｎｔ・Ｗｅｉｇｈｔｅｄ・Ｒａｎｄｏｍ・Ｐａｔｔｅｒｎ・Ｇｅｎｅｒａｔｉｏｎ・Ｓｙｓｔｅｍ，“Ｉｎｔ’ｌ．Ｔｅｓｔ・Ｃｏｎｆ．，ｐｐ．４９１－５００，１９９４）

【００１０】

【非特許文献２】

Ｈ－Ｃ．Ｔｓａｉ，Ｋ－Ｔ．Ｃｈｅｎｇ，Ｓ．Ｂｈａｗｍｉｋ，“Ｉｍｐｒｏｖｉｎｇ・Ｔｈｅ・Ｔｅｓｔ・ｑｕａｌｉｔｙ・ｆｏｒ・Ｓｃａｎ－ｂａｓｅｄ・ＢＩＳＴ・Ｕｓｉｎｇ・Ａ・Ｇｅｎｅｒａｌ・Ｔｅｓｔ・Ａｐｐｌｉｃａｔｉｏｎ・Ｓｃｈｅｍｅ，”ＤＡ・Ｃｏｎｆ，ｐｐ．７４８－７５３，１９９９

【００１１】

【非特許文献３】

Ｇ．Ｋｉｅｆｅｒ・ａｎｄ・Ｈ－Ｊ．Ｗｕｎｄｅｒｌｉｃｈ，“Ｄｅｔｅｒｍｉｎｉｓｔｉｃ・ＢＩＳＴ・ｗｉｔｈ・Ｓｃａｎ・Ｃｈａｉｎｓ”，Ｉｎｔ’ｌ・Ｔｅｓｔ・Ｃｏｎｆ，ｐｐ．１０５７－１０６４，１９９８

【００１２】

【非特許文献４】

Ｇ．Ｋｉｅｆｅｒ，Ｈ．Ｖｒａｎｋｅｎ，Ｅ．Ｊ．Ｍａｒｉｎｉｓｓｅｎ，Ｈ－Ｊ．Ｗｕｎｄｅｒｌｉｃｈ，“Ａｐｐｌｉｃａｔｉｏｎ・ｏｆ・Ｄｅｔｅｒｍｉｎｉｓｔｉｃ・Ｌｏｇｉｃ・ＢＩＳＴ・ｏｎ・Ｉｎｄｕｓｔｒｉａｌ・Ｃｉｒｃｕｉｔｓ”、Ｉｎｔ’ｌ・Ｔｅｓｔ・Ｃｏｎｆ，ｐｐ．１０５－１１４，２０００

【００１３】

【非特許文献５】

Ｄ．Ｄａｓ、Ｎ．Ａ．Ｔｏｕｂａ，“Ｒｅｄｕｃｉｎｇ・Ｔｅｓｔ・Ｄａｔａ・Ｖｏｌｕｍｅ・Ｕｓｉｎｇ・Ｅｘｔｅｒｎａｌ／ＬＢＩＳＴ・Ｈｙｂｒｉｄ・Ｔｅｓｔ・Ｐａｔｔｅｒｎｓ”Ｉｎｔ’ｌ・Ｔｅｓｔ・Ｃｏｎｆ，ｐｐ．１１５－１２２，２０００

【００１４】

【非特許文献６】

Ｓ．Ｈｅｌｌｅｂｒａｎｄ，Ｓ．Ｔａｒｎｉｃｋ，Ｊ．Ｒａｊｓｋｉ、Ｂ．Ｃｏｕｒｔｏｉｓ，“Ｇｅｎｅｒａｔｉｏｎ・ｏｆ・ｖｅｃｔｏｒ・ｐａｔｔｅｒｎｓ・Ｔｈｒｏｕｇｈ・Ｒｅｓｅｅｄｉｎｇ・ｏｆ・Ｍｕｌｔｉｐｌｅ－Ｐｏｌｙｎｏｍｉａｌ・Ｌｉｎｅａ

r・Feedback・Shift・Registers”, Int’l. Test・Conf, pp. 120－129, 1992
【００１５】
【非特許文献７】
I. Bayraktaroglu, K. Udawatta, A. Oraologlu, “An・Examination・of・PRPG・Selection・Approaches・for・Large, Industrial・Design”, Asia・Test・Symposium, pp. 440－444, 1998
【００１６】
【非特許文献８】
I. Bayraktaroglu and A. Oraologlu, “Selecting・a・PRPG：Randomness, Primitiveness, or・Sheer・Luck?”, Asia・Test・Symposium, pp. 373－379, 2001
【００１７】
【非特許文献９】
C. Fagot, O. Gascuel, P. Girard, C. Landrault、“A・Ring・Architecture・Strategy・for・BIST・Test・Pattern・Generation”、Asia・Test・Symposium, pp. 418－423, 1998
【００１８】
【非特許文献１０】
C. Fagot, O. Gascuel, P. Girard、C. Landrault, “On・Caluclating・Efficient・LFSR・Seeds・for・Built－In・Self・Test”Europe・Test・Conf, pp. 4－14, 1999
【００１９】
【発明が解決しようとする課題】
ＬＦＳＲ等をテストパターン発生器として用いる上記のような集積回路のテスト方法において、テストパターン発生器に与える初期値が異なると、所定の故障カバレッジを達成するまでのテスト長が異なることが知られている。従って、テスト時間を最短にする初期値を予め選定することが出来れば、集積回路の量産時のテスト時間が短縮され、テストに要するコストの大幅削減が可能となる。
【００２０】
ところがこれまで、パターン発生器の初期値については殆ど注意が払われず、単に適当な値、例えば、１１１…１（オール１）、が選択されているのみである。従って、このように特別な考慮を払うことなく選択した初期値が不適当なものであった場合、テスト長が長大になる可能性を有している。
【００２１】
本発明はかかる点に注目して成されたものであり、テストパターン発生器において、目的の故障カバレージに最短テスト長で到達するテストパターン発生器に与える初期値の選定方法を提案する。本方法を用いると、初期値の計算に必要なオフラインの計算時間は必要であるものの、チップのハードウエアを増加させることなくテスト時間を短くすることができる。
【００２２】
さらに本発明は、ある固定されたテスト長に対して最大故障検出数が得られるテストパターン初期値の選定方法を提案する。このことは残存故障数が小さくなることを意味する。本方法を用いると、オフラインでの計算時間を必要とするものの、ハードウエアの追加は無く、残存故障を減らすことができる。ビットフリップ法を適用する場合には、そのハードウエア量を削減することにつながる。

【００２３】
【課題を解決するための手段】
本発明は、前記課題を解決するために、ａ）任意に与えられた第１の初期値を用いて故障シミュレーションおよび逆順故障シミュレーションを行って、予め決められたｎ個の故障を検出するための第１のテスト長および第１の極小テスト長を求めるステップと、ｂ）前記第１の初期値、第１のテスト長および第１の極小テスト長に基づいて前記第１の極小テスト長よりも短いテスト長を得ることが可能な第２の初期値を算出するステップと、ｃ）前記算出された第２の初期値を用いて第２の故障シミュレーションを行って前記ｎ個の故障を検出するための第２のテスト長を求めるステップと、ｄ）前記第２のテスト長と前記第１の極小テスト長を比較し、前記第２のテスト長が前記第１の極小テスト長と同じか長い場合は前記第１の極小テスト長を第２の極小テスト長とみなし、前記第２のテスト長が前記第１の極小テスト長より短い場合は、前記第２の故障シミュレーションに対して逆順故障シミュレーションを行って第２の極小テスト長を求めるステップと、ｅ）前記第２の初期値、第２のテスト長および第２の極小テスト長を前記第１の初期値、第１のテスト長および第１の極小テスト長とみなして前記ステップｂ）、ｃ）およびｄ）を繰り返して最小テスト長を求めるステップ、を具備し、前記ステップｅ）で求めた最小テスト長を提示する初期値を前記集積回路のテストにあたって前記テストパターン発生器における初期値として決定する事を特徴とする。
【００２４】
前記第１の構成を有する本発明の方法によれば、テストパターン発生器に対して入力する初期値として、予め決められた数の故障を最短時間で検出することができる値を選択することができる。従って、この値を集積回路テストにあたってテストパターン発生器の初期値として選択すれば、初期値として特別な考慮を払わない場合に比べて、テスト時間を大幅に短縮することが出来、量産時のコスト削減効果が大きい。
【００２５】
本発明はまた、前記課題を解決するために、ａ）任意に与えられた第１の初期値を用いて第１の故障シミュレーションを行って、予め決められた目標テスト長L_{fix}において検出される第１の故障数ｎを求めるステップと、ｂ）前記第１の故障シミュレーションに対して逆順故障シミュレーションを行って前記第１の故障数ｎを検出するのに必要な第１の極小テスト長を求めるステップと、ｃ）前記第１の初期値、前記目標テスト長L_{fix}および第１の極小テスト長に基づいて、前記第１の極小テスト長を提示する第２の初期値を算出するステップと、ｄ）前記算出された第２の初期値を用いて第２の故障シミュレーションを行って前記目標テスト長L_{fix}において検出される第２の故障数ｎｔを求めるためのステップと、ｅ）前記第１および第２の故障数ｎおよびｎｔを比較し、ｎｔ≦ｎの場合は前記第１の故障数ｎを極大の検出故障数とし、ｎｔ＞ｎの場合、ｎｔ≦ｎと成るまで前記ステップｂ）～ｄ）を繰り返して極大の検出故障数を求めるステップと、ｆ）前記極大の故障検出数を提示する初期値の、テストパターン系列における次の初期値を用いて、前記極大の検出故障数を検出するために必要なテスト長Ｌを第３の故障シミュレーションによって求めるステップと、ｇ）前記テスト長Ｌを前記目標のテスト長L_{fix}と比較し、$L \geqq L_{fix}$の場合、前記第３の故障シミュレーションの初期値、テスト長Ｌおよび目標テスト長L_{fix}に基づいて、次に前記目標テスト長L_{fix}よりも短いテスト長の得られる可能性のある初期値を算出するステップと、ｈ）前記ステップｇ）で算出された初期値に対して、$L < L_{fix}$と成るまで前記ステップｆ）およびｇ）を繰り返すステップと、ｉ）前記ステップｇ）または前記ステップｈ）において$L < L_{fix}$を提示する初期値を前記第１の初期値とみなして前記ステップａ）～ｈ）を繰り返し、前記目標テスト長において検出される最大の故障数を得るためのステップ、を具備し、前記ステップｉ）で求めた最大故障検出数を提示する初期値を前記集積回路のテストにあたって前記テストパターン発生器における初期値として決定する事を特徴とする。
【００２６】
前記第２の構成を有する本発明の方法によれば、テストパターン発生器に対して入力する

初期値として、予め決められたテスト長の範囲内で最大個数の故障を検出することが可能な値を選択することができる。従って、この値を集積回路テストにあたってテストパターン発生器の初期値として選択すれば、初期値として特別な考慮を払わない場合に比べて、テストの品質を大幅に改善することができ、量産時のコスト削減効果が大きい。

【００２７】

【発明の実施の形態】

擬似ランダムパターン発生器としてＬＦＳＲを使ったテストパターン発生器は、初期状態をセットした後にクロックを与えるとテストパターン系列を発生する。このようなテストパターン発生器を用いたＢＩＳＴとして、図１に示されるようなｔｅｓｔ－ｐｅｒ－ｃｌｏｃｋ構成が挙げられる。図１において、１はＬＦＳＲ２によって構成したテストパターン発生器、３はテストパターンが入力される被テスト回路（ＣＵＴ）、さらに４はテスト応答圧縮器を示す。ＬＦＳＲ２内部のフリップフロップの状態が決定すると、その後出力されるテストパターン系列が一意に決まる。ＬＦＳＲ２のフリップフロップの初期状態を決定する情報をシードと呼ぶ。

【００２８】

なお、テストパターン発生器１にＬＦＳＲを用いたｔｅｓｔ－ｐｅｒ－ｃｌｏｃｋ構成のＢＩＳＴでは、ＬＦＳＲ２に入力されるシードとＬＦＳＲ２からＣＵＴ３に出力されるテストパターンとは、同じ物となる。

【００２９】

図１のようなＬＦＳＲを使った単一縮退故障テストを考える。単一縮退故障テストを対象とする、被テスト回路は組合せ回路である。

【００３０】

ｍ次の原始多項式によって構成されるＬＦＳＲはＣＵＴに2^m-1個の異なるテストパターンを出力する。ＬＦＳＲからのｍビット２元ベクトルは、ＧＦ（2^m）上の元α^i（$0 \leqq i \leqq 2^{m-1}-2$）とみなすことができる。ここでｉは指数を表す。

【００３１】

シードα^0に対して、ＬＦＳＲはまず、テストパターンα^0を出力し、つぎにα^1、以下同様にα^2、α^3、…とテストパターンを出力する。シードがα^1であるときテストパターンα^1を出力し、続いてα^2を出力する。シードα^iに対してｊ番目に出力するテストパターンはα^{i+j-1}と表される。

【００３２】

ＣＵＴに存在しうる検出可能な単一縮退故障の集合をＦとする。あるテストパターンの故障カバレージをＣ％と表すと、

【数１】

$$\frac{|F| \times C}{100}$$

個の故障が検出される。ただし｜｜は集合の要素数を表す。

【００３３】

F_aは、Ｆに含まれる故障の中で、テストパターンαが検出する故障の集合である。

【００３４】

シードα^iに対し、テストパターンα^{i-1}、α^{i-2}・・・と発生させる方法を逆順テストパターン発生と呼ぶことにする。Ｌ（α^i、n_{fix}）はシードα^iに対し目標故障検出数＝n_tを達成するテスト長を返す関数である。n_{fix}＝｜Ｆ｜となることが望ましいが、n_{fix}として｜Ｆ｜よりやや小さい値を設定することが現実的である。

【００３５】

L_R（α^{i+1}、n_{fix}）はシードα^{i+1}に対し逆順テストパターン発生を行い、目標故障検出数n_tを達成するテスト長を返す関数である。Ｌ（α^i、n_{fix}）、L_R（α^{i+1}、n_{fix}）

は故障シミュレーションを実行することにより求めることができる。
【００３６】
あるＣＵＴとその目標検出故障数n_{fix}、およびＬＦＳＲの多項式が与えられたとき、テスト長が最も短くなるＬＦＳＲのシードを最短テスト長シードα^{i*}と呼ぶ。このとき、$i^{*}=argmin\{L(\alpha^{i}, n_{fix})\}$である。
【００３７】
次に、シードとＬＦＳＲから出力されるテストパターン系列との関係について考察する。
【００３８】
ＩＳＣＡＳ８９ベンチマークのｓ３８６回路の組合せ回路部分に、テスト／クロックのテストを適用することを考える。この回路の入力幅は１３ビットである。各シードに対して１００％故障カバレージを達成するテスト長についての実験結果を、図２に示す。ｓ３８６回路の検出可能故障数は３８４個である。横軸はシード番号、縦軸は１００％故障カバレージを達成するテスト長（クロック）を示す。例えばシードをα^{0}とした時の１００％故障カバレージを達成するテスト長Ｌは、図２より$L(\alpha^{0}, 384)=1783$であることが判る。
【００３９】
図２から、ＬＦＳＲを使ったテスト／クロック構成における、シードとテスト長に関する特徴がわかる。つまり指数が１つ増える度にテスト長は、
１）１つだけ短くなる、
２）同じ、または
３）より長くなる、
のいずれかである。
【００４０】
この特徴をより理解しやすくするために、図３に、図２を簡略化した例を示す。図３において、縦軸は１００％故障カバレージを達成するテスト長Ｌ、横軸はシードの指数ｉを示す。図３では、説明を簡単にするためＣＵＴの検出可能故障数を１００としている。
【００４１】
図３において、シードα^{0}の時のテスト長は$L(\alpha^{0}, 100)=8$である事がわかる。次に、シードをα^{1}とした時、α^{1}からα^{7}までの７パターンによって、α^{0}によって検出される故障の全てが検出されたものとする。すなわち、
【００４２】
【数２】

$$\left|\mathbf{F}_{a^1}\cup\mathbf{F}_{a^2}\cup\cdots\cup\mathbf{F}_{a^7}\right|=\left|\mathbf{F}_{a^0}\cup\mathbf{F}_{a^1}\cup\cdots\cup\mathbf{F}_{a^7}\right|=\left|\mathbf{F}\right|$$

とする。このとき$L(\alpha^{1}, 100)=7$となる。
【００４３】
同様にα^{2}からα^{7}の６パターンが検出する故障が、α^{1}によって検出される故障を全て含むとき、すなわち、
【００４４】
【数３】

$$\left|F_{a^0}\cup F_{a^1}\cup\cdots\cup F_{a^7}\right|=\left|F_{a^1}\cup F_{a^2}\cup\cdots\cup F_{a^7}\right|=\left|F\right|$$

とすると、α^{2}がシードであるときのテスト長は、$L(\alpha^{2}, 100)=6$となる。同様にしてシードがα^{3}の場合のテスト長$L(\alpha^{3}, 100)=5$である。
【００４５】
一方、シードα^{4}のとき、パターンα^{4}からα^{7}が検出する故障は１００未満であって、シ

ードα^{3}が検出する故障に達しないものとする。すなわち、

【００４６】

【数４】

$$\left|F_{a^4} \cup F_{a^5} \cup \cdots \cup F_{a^7}\right| < \left|F_{a^3} \cup F_{a^4} \cup \cdots \cup F_{a^7}\right| = \left|F\right|$$

とする。このとき、シードα^{4}に対して、シードα^{3}が検出する故障を全て検出するまで、テスト長が延びることになる。図３の例ではテスト長は５から１０に延びている。

【００４７】

今、以下の条件、

Ｌ（α^i、ｎ$_{fix}$）＜Ｌ（α^{i+1}、ｎ$_{fix}$）

が成立するとき、α^{i}を極小テスト長シードと呼び、Ｌ（α^i、ｎ$_{fix}$）を極小テスト長Ｌ$_{min}$と呼ぶ。以下、極小テスト長となるシードを極小テスト長シードα^{imin}と表し、そのシードの指数をｉ$_{min}$と表す。このようにして全指数に対してテストを行って見出した極小テスト長のうち、最小のものが最短テスト長Ｌｓとなる。

【００４８】

図３の例では、極小テスト長シードはα^{3}であり、このとき極小テスト長は５である。図２の例では、極小テスト長は１３個存在し、図中左から、１６４８、２１０４、１２９８、１４６６、２５０５、２２１１、２８２３、２２８３、２４９７、１８６６、１６１５、２０００、１８６７であり、従って、最短テスト長Ｌｓは１２９８となる。

【００４９】

以上の様にして求めた最短テスト長Ｌを与えるシードを、図１の回路におけるＬＦＳＲ２の初期シードとして選定すると、ＣＵＴ３のテストは最短時間で終了することとなる。一方、ＣＵＴ３のテストにおいて、従来のテスト方法で行われていたように、特にシードに注意を払わず任意のシードを選定した場合、図２の例では、例えばテスト長３０００を超えるシードが選定される可能性がある。このようなシードを選択すると、ＶＬＳＩの量産時においてテストに要する時間が長大となり、テストのコストが増加する。従って、最短テスト長を有するシードを選定することが、量産時のＶＬＳＩテストコスト削減に、大きく貢献する。

【００５０】

しかしながら、図２に示す様に、テストパターン系列全てのシードに対してテスト長を求めることは、ＶＬＳＩの回路規模が非常に大きなものであるため、実際には殆ど不可能である。従って、最適なシードを選定するための何らかの効率的で実際的な方法が希求されている。

【００５１】

本発明では、上述した、シードとテスト長との関係における特徴を利用して、実際に利用可能な方法で最適初期値を選定するための方法を提供するものである。

【００５２】

以下に、本発明の方法に従って、予め設定された数の故障ｎ$_{fix}$を最短テスト長によって求めるための初期シードを選定する手順について、図面を参照して説明する。なお、この場合のシードを最短テスト長シード選定方法と呼ぶ。

【００５３】

まず、本選定方法を、極小テスト長を求めるステップＡと、より短い極小テスト長の存在を確認するステップＢに分けて説明する。ｉは現在の指数を表し、ｉ＝０と初期化する。つまり指数０のシードから処理をはじめる。

【００５４】

［ステップＡ］

ステップＡでは、極小テスト長とそのシードを求める。

【００５５】

指数ｉ以上で最も早く現れる極小テスト長を求める。まずシードα^{i}のときのテスト長Ｌｉ（α^{i}、n_{fix}）を、故障シミュレーションによって求める。次に、逆順テストパターン発生による故障シミュレーションを行い、L_R ｉ（α^{i+Li-1}、n_{fix}）を求める。これにより、極小テスト長Ｌｍが求められる。極小テスト長となる指数ｉｍはテスト長Ｌｉから極小テスト長Ｌｍを引いたものに、ｉを加えたもの、すなわち、Ｌｉ－Ｌｍ＋ｉ、となる。

【００５６】

図４にステップＡの例を示す。まずＳｔｅｐ－Ａ－１でシードα^{0}に対して、故障シミュレーションを行って、テスト長Ｌ（α^{0}、n_{fix}）を求める。このとき、図４の例では、テスト長Ｌ（α^{0}、n_{fix}）＝８となる。続いて、Ｓｔｅｐ－Ａ－２で、最大故障検出数n_{fix}を得たシードα^{7}を初期シードとして逆順故障シミュレーションを行い、テスト長L_R（α^{7}、n_{fix}）を求める。この値が極小テスト長Ｌｍになる。図４に示す例ではＬｍ＝５となる。

【００５７】

この極小テスト長Ｌｍを実現する指数ｉｍは、Ｌ－Ｌｍ＋ｉ＝８－５＋０＝３となる。この計算ステップをＳｔｅｐ－Ａ－３と表す。つまりシードα^{L-Lm+i}、即ちα^{3}が、極小テスト長５によって目標故障カバレージを達成するシードであるとわかる。

【００５８】

［ステップＢ］

ステップＢでは、ステップＡで求めた極小テスト長よりも短い極小テスト長を達成するシードが存在するかどうか調べる。

【００５９】

指数$i_{min}+1$のシードは、極小テスト長Ｌｍよりも長いテスト長となる。

【００６０】

まず指数、ｉ＝$i_{min}+1$、と設定し、テスト長ｌ＝Ｌ（α^{i}、n_{fix}）を求める。このテスト長Ｌから、次に極小テスト長l_{min}よりも短いテスト長が現れる可能性のある指数が計算できる。その指数はｉ＋ｌ－l_{min}＋１である。これを新たな指数ｉとする。計算したシードα^{i}のテスト長ｌ＝（α^{i}、n_{fix}）を求める。テスト長ｌが極小テスト長l_{min}よりも短ければ、Ｓｔｅｐ－Ａに戻り、再び極小テスト長を求める。そうでなければ、再び極小テスト長l_{min}よりも短いテスト長が現れる可能性のある指数を計算する。これを、極小テスト長l_{min}より短いテスト長が現れるまで続ける。これを、ｍビットのＬＦＳＲの場合、指数（$2^{m}-1$）まで続ける。ｉ＝$2^{m}-2$となったときの極小テスト長l_{min}が目標故障カバレージを達成する最短テスト長l^{*}であり、i_{min}はその最短テスト長となるシードの指数i^{*}を示す。

【００６１】

図５及び図６によって、ステップＢの動作を説明する。図５及び図６のいずれの場合も、すでにシードα^{3}が極小テスト長l_{min}＝５を達成するとする。つまりi_{min}＝３である。

【００６２】

図５の場合を説明する。Ｓｔｅｐ－Ｂ－１にて、シードα^{4}に対するテスト長Ｌ（α^{4}、n_{fix}）を求める。図５の例では、１０となる。極小テスト長l_{min}＝５であるので、Ｌ（α^{4}、n_{fix}）、…Ｌ（α^{9}、n_{fix}）≧５である。すなわちα^{4}からα^{9}は最小テスト長を与えるシードの候補とはならない。

【００６３】

Ｓｔｅｐ－Ｂ－２で次にl_{min}よりも短いテスト長となる可能性のあるシードの指数を求める。この場合、ｉ＝４＋１０－５＋１＝１０となり、シードはα^{10}となる。シードα^{10}に対するテスト長Ｌ（α^{10}、n_{fix}）を求める。図５の例では、テスト長Ｌ＝４である。つまり指数１１のシードを使うことにより、さらに短いテスト長によって目標故障検出数n_{fix}が得られる可能性がある。この場合ステップＡに戻り、再び極小テスト長l_{min}を求める。

【００６４】

図６の場合を説明する。Ｓｔｅｐ－Ｂ－１とＳｔｅｐ－Ｂ－２の動作は図５と同じである。Ｓｔｅｐ－Ｂ－１にて、シードα^{4}に対するテスト長Ｌ（α^{4}、n_{fix}）を求める。図６の例ではテスト長は１０となる。次にＳｔｅｐ－Ｂ－２では直前のＳｔｅｐ－Ａで求めた極小テスト長l_{min}よりも短いテスト長となる可能性のあるシードの指数を計算する。図６でも図５と同様、シードα^{10}となる。Ｓｔｅｐ－Ｂ－３においてこのシードα^{10}のテスト長Ｌ（α^{10}、n_{fix}）を求める。図６の例ではＬ（α^{10}、n_{fix}）＝７である。この場合、Ｓｅｔｐ－Ｂ－２に戻り、極小テスト長l_{min}よりも短いテスト長となる可能性のあるシードの指数を再び求める。

【００６５】

図７に最短テスト長シードをもとめるプログラムの一例を示す。

【００６６】

図８および９は、以上に示した最短テスト長シードを求める手順を要約して示すフローチャートである。以下、このフローチャートに沿って説明する。

【００６７】

まず、ステップＳ１でシードの指数ｉを０に設定する。次に、ステップＳ２において、シードをα^{0}として故障シミュレーションを実行し、予め設定したn_{fix}個の故障を検出するために必要となるテスト長Ｌを求める。図４の例ではテスト長Ｌは８である。

【００６８】

次のステップＳ３では、ステップＳ２で実行した故障シミュレーションに対して、逆順故障シミュレーションを行って、n_{fix}個の故障を検出するために必要な極小テスト長Ｌｍを求める。図４の例では、極小テスト長Ｌｍは５であることが判る。逆順故障シミュレーションの初期値ｉは、ｉ＝ｉ＋Ｌ－１である。

【００６９】

ステップＳ３で極小テスト長Ｌｍが求められると、ステップＳ４において、この極小テスト長Ｌｍを与えるシードの指数を計算する。ステップＳ５では、ステップＳ４で計算された指数を有するシードを選定する。ステップＳ６において、シードα^{i}で得られるテスト長Ｌｍが、集積回路テストにおいて満足し得る値であるか否かを判断する。Ｌｍの値が満足されるものである場合、ステップＳ７において処理を終了し、最適シードとしてステップＳ５で得られたα^{i}を設定する。

【００７０】

次に、ステップＳ８において、指数ｉがテストパターン系列Ｓの最終値に達しているか否かが判定される。達している場合（ステップＳ８のＹＥＳ）は、ステップＳ９で処理を終了し、ステップＳ５で得られたシードを最適値として選択する。

【００７１】

次に、ステップＳ１０において、指数ｉを１進め、ステップＳ１１でその指数がテストパターン系列Ｓの最終値に達していないことを確認した後、指数ｉを有するシードをステップＳ１２において選択し、ステップＳ１３で、ステップＳ２と同じ故障シミュレーションを行う。図５の例では、この故障シミュレーションは、シードα^{4}に対して行われている。なお、ステップＳ１１で指数ｉがテスト系列Ｓの最終値を超えている場合（ステップＳ１１のＹＥＳ）は、ステップＳ１４で処理を終了し、ステップＳ５で求めたシードα^{i}を最適値とする。

【００７２】

ステップＳ１３での故障シミュレーションによって、シードα$^{i+L-Lm+1}$を用いた場合のテスト長Ｌが得られる。図５の場合、このテスト長Ｌは１０である。

【００７３】

図５のステップＢ－２で説明したように、ステップＳ１３で新たなテスト長Ｌが得られると、今の指数ｉから、ｉ＝ｉ＋Ｌ－Ｌｍまでの間のシードは、極小テスト長Ｌｍよりも小さいテスト長を与えるシードとはなり得ない。従って、ステップＳ１５で指数をｉ＝ｉ＋Ｌ－Ｌｍ＋１に進め、ステップＳ１６でこの指数がテストパターン系列Ｓの最終値を超えないことを確かめた後、ステップＳ１７でステップＳ２と同じ故障シミュレーションを行

う。なお、ステップＳ１６で指数ｉがテストパターン系列Ｓの最終値を超える場合は、ステップＳ１８で処理を終了し、ステップＳ５で求めたシードα^{i}を最適値とする。

【００７４】

次のステップＳ１９では、ステップＳ１７で実行した故障シミュレーションの結果として得られたテスト長Ｌが極小テスト長Ｌｍよりも短いか否かを判断する。実際はテスト長は１クロックずつ減少しているので、ステップＳ１９ではＬ＝Ｌｍ－１であるか否かを判断しても良い。ステップＳ１９でＹＥＳの場合が図５の例に相当し、従ってこの場合は、ステップＳ３に戻って、逆順故障シミュレーションを行い、新たな極小テスト長Ｌｍを求める。

【００７５】

一方、ステップＳ１９でＮＯの場合は、図６の例に相当するので、ステップＳ１５に戻って、次に、極小テスト長Ｌｍよりも短いテスト長を得る可能性のある指数を演算し、その指数に対してステップＳ１６以下を繰り返して実行する。この繰り返しは、ステップＳ１９においてＬ＜Ｌｍと判断されるまで行われる。

【００７６】

以上の処理を繰り返して、最小（最短）のテスト長Ｌｍを求める。

なお、図５のシードα^{4}において、故障シミュレーションによりテスト長Ｌが求められると、この故障シミュレーションにおいてテストパターンα^{10}までの逆順故障シミュレーションを行って、その間にこれまでに求めた極小テスト長よりも短い極小テスト長が存在するか否かを見ることもできる。もし、より短い極小テスト長が存在すれば、その値を用いて、より短いテスト長を得ることが可能な次のシードの指数を計算することができる。この方法を使えば、最短テスト長シードの計算時間をさらに短くすることも可能である。

【００７７】

表１は、全シードに対して故障シミュレーションを行って最短テスト長シードを求めた場合と、上記本発明の最短テスト長シード選択法を適用した場合の比較を示している。実験は、ｓ３８６回路、多項式２００３３について行っている。表１から明らかなように、全シードについて故障シミュレーションを行った場合は、８１９１回の故障シミュレーションを必要とし、その計算時間は約４０分であった。一方、本発明の方法を適用した場合は、１４回の故障シミュレーションしか必要とせず、その計算時間は約４秒であった。

【００７８】

【表１】

表1　最短テスト長シード選択法の故障シミュレーション回数と計算時間（ｓ３８６／多項式２００３３）

	故障シミュレーション回数	計算時間
全シードの故障シミュレーション	8191回	約 40分
最短テスト長シード選択法	14回	約 ４秒

表２に、表１の実験で得られた最短テスト長シードによるテスト長と、シードの選択を考慮しなかった場合のテスト長の比較を示す。表１および表２より明らかなように、本発明の方法によれば、最短テスト長シードを非常に短い時間で得ることができるので、集積回路テストに要する時間が大幅に短縮される。

【００７９】

【表２】

表２　最短テスト長シードと、シードの選択を考慮しなかった場合のテスト長（ｓ３８６／多項式２００３３）

シード	テスト長
最短テスト長シード	1293
00‥01	1783
All 1	2230
0101‥	1971
1010‥	2949
ランダム14回の中の最短テスト長	1706

図１０は、図１に示したテスト／クロック構成のＢＩＳＴとは異なる、テスト／スキャン構成のＢＩＳＴを示す図である。本構成のＢＩＳＴは、テストパターン発生器１を、ＬＦＳＲ２とスキャンチェーン５で構成した事を特徴とする。この構成によれば、被テスト回路（ＣＵＴ）３の信号入力数とＬＦＳＲ２のビット数が独立となるため、小さなビット数のＬＦＳＲ２によって、集積度の高い回路のテストを行う事ができる利点を有している。例えば、１０００ビットの入力のある被テスト回路を２０ビットのＬＦＳＲでテストすることが可能となる。

【００８０】

図１０のテスト／スキャン構成もシードによって、出力するテストパターン系列を変化させることがでる。ところが、この構成ではＬＦＳＲのシードが、そのままテストパターンとして現れない。しかしシードによって、出力されるテストパターン系列が決定されることに代わりはない。ＬＦＳＲの段数がm、スキャンチェーン５の長さがｋビットであるとき、ＰＲＰＧはｋビットのテストパターンp_0、p_1、・・・を発生する。

【００８１】

シードをＬＦＳＲに設定後にテストパターンp_jを生成するシードをβ_jと表す。シード$\beta_j=\alpha^{(jxk) \bmod (2^m-1)}$となる。すなわち$\beta_0=\alpha^0$、$\beta_1=\alpha^k$、$\beta_2=\alpha^{2k}$、・・・である。このときｋと、（$2^m-1$）が互いに素である必要がある。互いに素となっていない場合は、スキャンチェーンにダミーのＦｌｉｐ・Ｆｌｏｐを追加することで互いに素となるように調整する。

【００８２】

図１１の例では、シードβ_0、・・・、β_4がそれぞれ、１０パターン目までに出力するテストパターンが示されている。例えばシードβ_0は、まずテストパターンp_0を出力する。続いてp_1を出力する。シードβ_0が１０パターン目に出力するテストパターンはp_9である。シードβ_1は、まずp_1を出力し、続いてp_2を出力する。このようにすることによって、図１０のようなｔｅｓｔ－ｐｅｒ－ｓｃａｎ構成も図１のｔｅｓｔ－ｐｅｒ－ｃｌｏｃｋ構成と同様の手法によって、目標故障カバレージを達成する最短テスト長を得ることが出来る。

【００８３】

以下、本発明の第２の実施形態について説明する。本実施形態は、ある固定されたテスト時間内に、最大の故障を検出することができる最適シードの選定方法に関する。複雑な回路の場合、９５％を超える故障カバレージの達成が難しく、あるテスト長を設定し、そのテスト長l_{fix}に対し、最も多くの故障を検出するシードを選ぶ方法が現実的である。このシード

【数５】

$$\beta_{j^*} = \arg\max_{\beta_j} \left| F_{\beta_j} \cup F_{\beta_{j+1}} \cup \cdots \cup F_{\beta_{j+l_{fix}-1}} \right|$$

を最大故障検出数シードと呼ぶ。

【００８４】

最大故障検出数シードの選定法は、前記最短テスト長シードの選定法を拡張したものである。最短テスト長シードの選定法との違いは、極小テスト長を見つけるたびに、動的に目標故障検出数を更新していくところである。

【００８５】

本方法の説明に先立ち、関数を定義する。Ｄ（α^i、ｌ）はシードα^iに対しテスト長ｌのとき、検出される故障検出数を返す関数である。Ｄ（α^i、ｌ）はＬ（α^i、n_{fix}）L_R（α^{i+l}、n_{fix}）、と同様に故障シミュレーションを実行することにより求めることができる。

【００８６】

テスト長l_{fix}を与えると、そのテスト長で最大故障検出数となるシードを選定する。あるテスト長l_{fix}における目標故障検出数を求めるＳｔｅｐ－Ａ＊と、より多くの故障が検出できるシードの存在を予測するＳｔｅｐ－Ｂ＊の２つのステップに分けられる。

【００８７】

β_jのシード番号をｊとする。シード番号０のシードから選定処理を始めるので、ｊ＝０となる。

【００８８】

Ｓｔｅｐ－Ａ＊では、テスト長l_{fix}における目標故障検出数を求める。

【００８９】

まず、シードβ_jのテスト長l_{fix}による故障検出数を目標故障検出数n_tとする。n_t＝Ｄ（β_j、l_{fix}）。次に、目標故障検出数n_tとなる極小テスト長を

【数６】

$$L_R(\beta_{i+l_{fix}-1}, n_t)$$

から求める。これにより、極小テスト長l_{min}が求められる。極小テスト長となるシード番号j_{min}は、j_{min}＝ｊ＋l_{fix}－l_{min}となる。このときのj_{min}に対してシードβ_{imin+1}のテスト長がl_{fix}よりも長くなるまでＳｔｅｐ－Ａ＊を続ける。

【００９０】

β_{imin+1}のテスト長が、目標テスト長l_{fix}よりも長いテスト長のときはＳｔｅｐ－Ｂ＊に移る。目標故障検出数n_{fix}を達成するテスト長ｌ＝L_F（β_j、n_{fix}）がl_{fix}よりも短くなるシード番号を計算する。ｌが計算どおりのテスト長l_{fix}－１となるまで、予測を繰り返し、予測どおりのテスト長となったとき、再び目標故障検出数n_tを設定しなおす。

【００９１】

これを、ｍビットのＬＦＳＲの場合、シード番号2^m－２まで続ける。このときの目標故障検出数n_tがあるテスト長l_{fix}における最大故障検出数n^*であり、シード番号j_{min}が最大故障検出数シード番号j^*となる。

【００９２】

図１２に最大故障検出数シードをもとめるプログラムの１例を示す。

【００９３】

図１３および１４は、以上に示した最大故障検出数シードを求める手順を要約して示すフローチャートである。また、図１５は、このフローチャートの説明のための図面である。以下、これらの図面を参照して、本発明の第２の実施形態を説明する。

【００９４】
まず、ステップＰ１においてシード番号ＪをＪ＝０に設定する。次にステップＰ２で、シード番号０のシードβ０について故障シミュレーションを実行し、予め決められたテスト長L_{fix}（図１５の場合は、L_{fix}＝１０）において検出される故障数ｎを求める。
【００９５】
ステップＰ３では、ステップＰ２で求めたｎの値が、最大故障数として満足できるものか否かを判断する。満足される場合は、ステップＰ４で処理を終了し、この最大故障数ｎを与えるシードβ０を、テストパターン発生器に入力する初期値とする。
【００９６】
ステップＰ３でＮＯの場合は、ステップＰ５において、ステップＰ２で行われた故障シミュレーションに対し逆順故障シミュレーションを行って、ｎ個の故障数を検出するのに必要となる極小テスト長Ｌｍを求める。図１５の場合、極小テスト長Ｌｍは７である。この極小テスト長を与えるシードの番号は、$J+L_{fix}-Lm$として計算される。ステップＰ６では、シード番号$J+L_{fix}-Lm$がテストパターン系列Ｓの最終値を超えるか否かを検出する。超える場合は、ステップＰ７で処理を終了し、最後に求めた極小テスト長を与えるシードを最適初期値として選定する。
【００９７】
ステップＰ６で、シード番号$J+L_{fix}-Lm$がテストパターン系列Ｓの範囲内である場合（ステップＰ６のＮＯ）、ステップＰ８において、シード番号Ｊを、$J=J+L_{fix}-Lm$とし、ステップＰ９においてこの番号のシードを用いて故障シミュレーションを行い、目標テスト長L_{fix}内で何個の故障が検出されるかを検出する。これは、最短テスト長シード検出方法の場合と異なり、極小テスト長Ｌｍを与えるシードの場合、極小テスト長Ｌｍから目標テスト長L_{fix}の間に新しい故障が見つかる可能性が存在するためである。
【００９８】
図１５の場合、シードβ３に対してテスト長１０を与えて故障シミュレーションを行うと、テストパターンｐ１０～ｐ１２において新しい故障が発見される可能性がある。
【００９９】
次のステップＰ１０では、ステップＰ９で求めた故障数ｎｔをステップＰ２で求めた故障数ｎと比較する。ｎｔ＞ｎの場合は、ステップＰ１１でｎをｎｔとした後、ステップＰ３を経てステップＰ５に戻って再び逆順故障シミュレーションを行って、故障数ｎｔを検出するために必要となる極小テスト長を求める。さらに、ステップＰ６以下を実行して、新たな故障数ｎｔを、ｎｔ≦ｔと成るまで求める。
【０１００】
以上の処理によって、目標テスト長L_{fix}で検出される故障数ｎが極大となり、それ以上更新されなくなると、ステップＰ１２において、極大故障数を提示するシードに対し、シード番号を１だけ進める。図１５の例では、シードβ３で検出される故障数ｎｔを、ｎｔ≦ｔとした場合、次のシードはβ４である。ステップＰ１３においてこのシード番号がテストパターン系列の最終値を超えないか否かを判断し、超える場合はステップＰ１４で処理を終了し、超えない場合はステップＰ１５に進む。
【０１０１】
ステップＰ１５では、新たな番号のシードに対して故障シミュレーションを実行して、これまでに求めた極大の故障数ｎを得るために必要なテスト長Ｌを求め、ステップＰ１６でこのテスト長Ｌを目標テスト長L_{fix}と比較する。図１５の場合、シードβ４のテスト長Ｌは１３であるので、$L>L_{fix}$である。この場合、図１５から明らかなように、シードβ８において初めてテスト長$L<L_{fix}$と成る可能性がある。ステップＰ１７では、このような、次にテスト長$L<L_{fix}$となる可能性のあるシードの番号を算出している。
【０１０２】
ステップＰ１８で、ステップＰ１７で算出されたシード番号がテストパターン系列の最終

値を超えないか否かを判定して、超えない場合、ステップＰ１５に戻って算出された番号のシードに対して故障シミュレーションを行う。このステップは、ステップＰ１６において、テスト長ＬがＬ＜L_{fix}と成るまで繰り返される。なお、ステップＰ１８でＹＥＳの場合は、ステップＰ７、Ｐ１３の場合と同様にステップＰ１９で処理を終了する。

【０１０３】

ステップＰ１６でテスト長Ｌ＜L_{fix}となると、このときのシードを用いて再びステップＰ２以下を実行することにより、さらに目標テスト長L_{fix}内で検出される新たな故障数が得られる。従って、このようにして得られた新たな故障数が予め決められた目標の故障検出数に達するか、あるいは、シード番号がテストパターン系列の最終値を超えるまで、上記の処理を繰り返して、最大故障検出数を得る。このときのシードがテストパターン発生器の最適初期値となる。

【０１０４】

なお、図１５のシードβ４において、故障シミュレーションによりテスト長Ｌが求められると（図１４のステップＰ１５）、この故障シミュレーションにおいてテストパターンｐ１４までの逆順故障シミュレーションを行って、その間にこれまでに求めた極小テスト長よりも短い極小テスト長が存在するか否かを見ることもできる。もし、より短い極小テスト長が存在すれば、その値を用いて、目標テスト長L_{fix}内でより多くの故障数を検出することが可能な次のシード番号を計算することができる。この方法を使えば、最大故障検出数シードの計算時間をさらに短くすることも可能である。

【０１０５】

上記最大故障検出数シードの選定に関する説明は、図１０および図１１に示すｔｅｓｔ-ｐｅｒ-ｓｃａｎ構成に関して行っているが、この選定方法が、ｔｅｓｔ－ｐｅｒ－ｃｌｏｃｋ構成のＢＩＳＴに関しても同様に適用できることは勿論である。

【０１０６】

表３は２０ビット原始多項式ＬＦＳＲを使ったｃ７５５２回路の２０多項式に対して、シードα^0と本発明の方法で求めた最大故障検出数シードの場合について比較する表である。本表では、シードα^0（００・・０１）と最大故障検出数シードの平均故障検出数と最小の残存故障数、１多項式あたりの処理時間を示す。

【０１０７】

【表３】

表３　ｃ７５５２　２０ビットＬＦＳＲ／２０多項式の平均故障検出数

テスト長	シード00・・01		最大故障検出数シード		
	平均	最小	平均	最小	
	残存故障数	残存故障数	残存故障数	残存故障数	処理時間(秒)
1000	464	418	369	359	4121
5000	351	319	277	259	4262
10000	304	270	243	227	5034

目標テスト長１０００で、シードα^0を使った場合、平均の残存故障数は４６４であり、最小の残存故障数は４１８である。最大故障検出数シードは平均で３６９個、最小の残存故障数は３５９である。１多項式あたりの処理時間は４１２１秒であった。

【０１０８】

テスト長が１００００の場合、２０個の原始多項式の中で最も故障検出数の大きい最大故障検出数シードを持つ多項式は、残存故障数が２２７である。これは、シードα^0の平均残存故障数の３０４よりも７７個多くの故障を検出している。

【図面の簡単な説明】

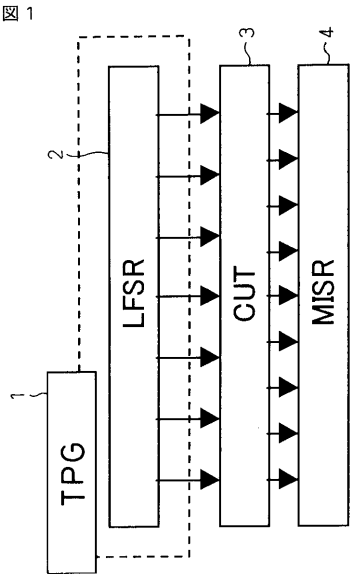
【図１】テスト／クロック構成のＢＩＳＴを示す図。

【図２】シードの指数とテスト長の関係を示す図。

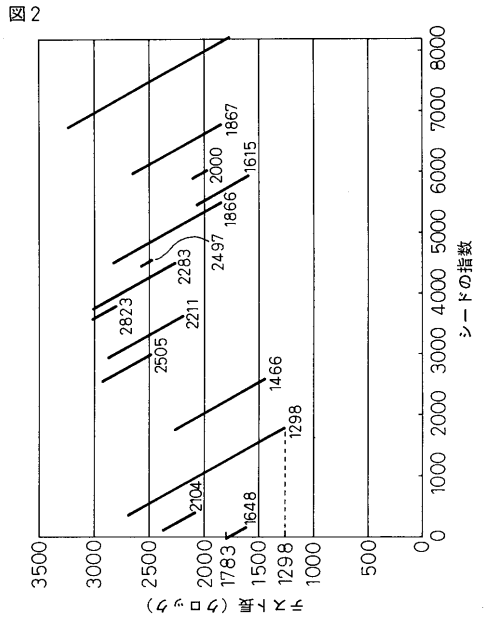
【図３】図２のシードの指数とテスト長の関係を簡略化して示す図。

【図４】本発明の１実施形態に係る方法の説明に供する図。
【図５】本発明の１実施形態に係る方法の説明に供する図。
【図６】本発明の１実施形態に係る方法の説明に供する図。
【図７】本発明の１実施形態を実施するプログラムの一例を示す図。
【図８】本発明の１実施形態に係る方法の手順を示すフローチャートの前半部分を示す図。
【図９】図８に示すフローチャートの後半部分を示す図。
【図１０】テスト／スキャン構成のＢＩＳＴを示す図。
【図１１】テスト／スキャン構成のＢＩＳＴにおけるシードとテストパターンの関係を示す図。
【図１２】本発明の他の実施形態を実施するプログラムの一例を示す図。
【図１３】本発明の他の実施形態に係る方法の手順を示すフローチャートの前半部分を示す図。
【図１４】図１３に示すフローチャートの後半部分を示す図。
【図１５】図１３、１４に示すフローチャートの説明に供する図。
【符号の説明】
１…テストパターン発生器
２…ＬＦＳＲ
３…被テスト回路
４…テスト応答圧縮器
５…スキャンチェーン

【図１】

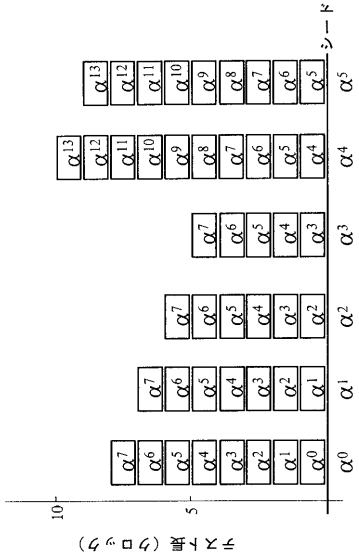


【図２】



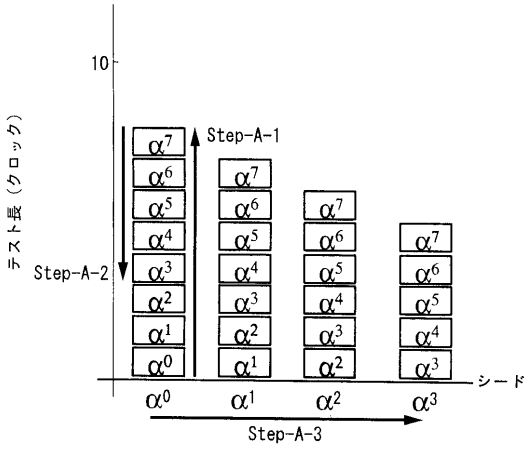
【図３】

図３



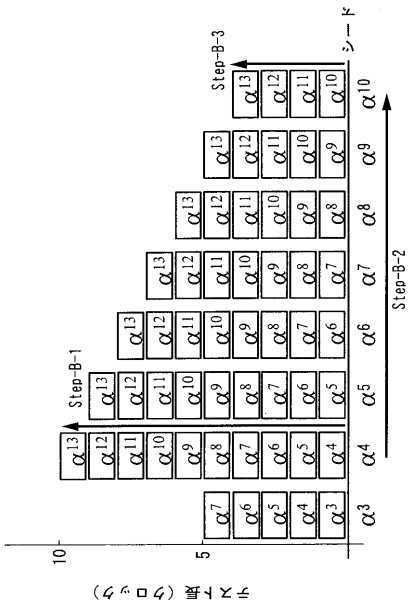
【図４】

図４



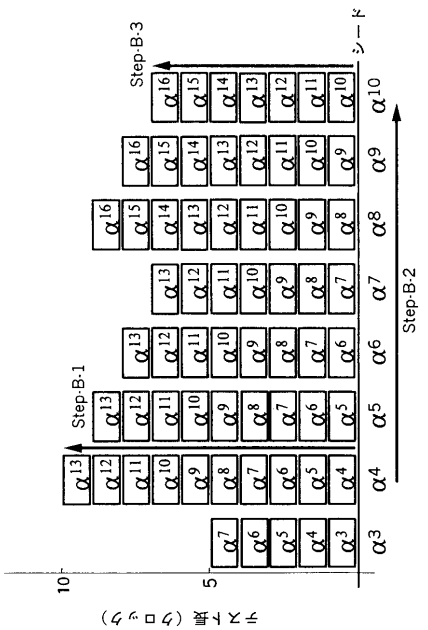
【図５】

図５



【図６】

図６



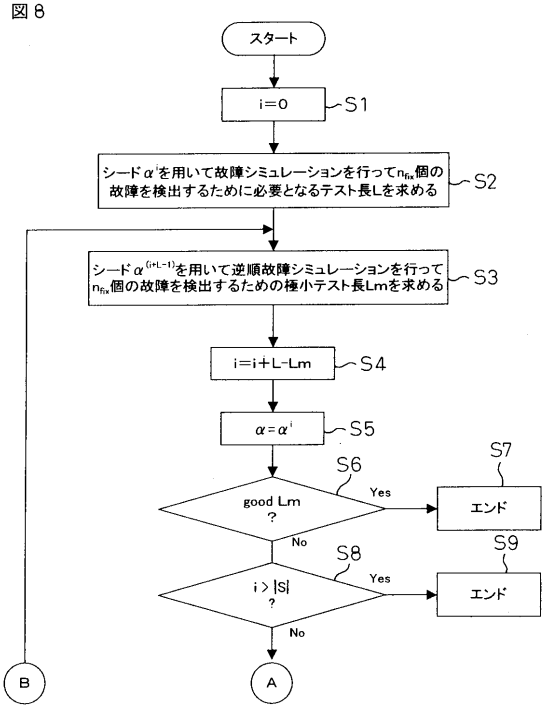
【図７】

図７

```
procedure SEARCH_ISTAR
begin
  i := 0;
  l := L(α^i, n_fix);
  while( i < 2^n − 1 )do
  begin
    l_min := L_R(α^(i+l−1), n_fix);
    i_min := i + l − l_min;
    i := i_min + 1;
    l := L(α^i, n_fix);
    repeat
      i := i + l − l_min + 1;
      l := L(α^i, n_fix)
    until( l = l_min − 1 and i < 2^n − 1 )
  end;
  l* := L_min;
  i* := i_min
end.
```

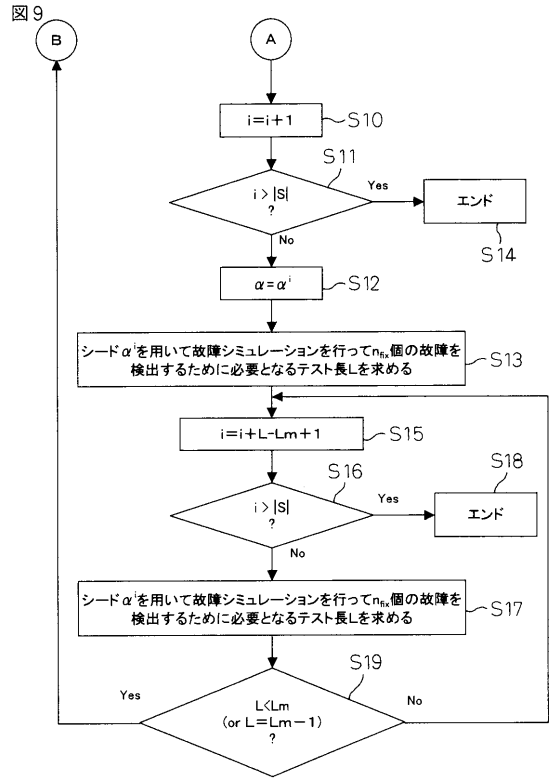
【図８】

図８



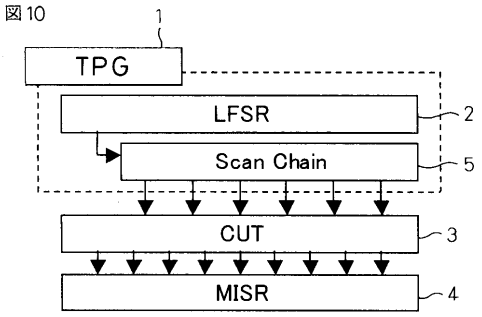
【図９】

図９



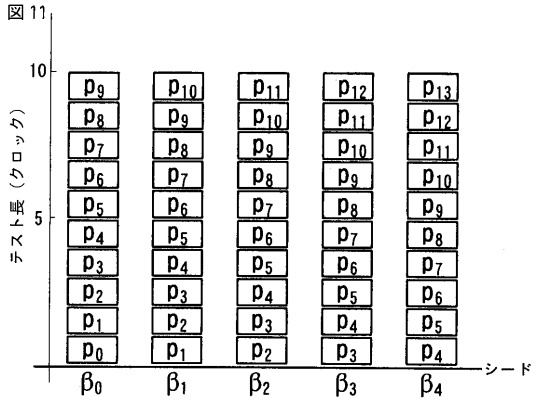
【図１０】

図１０



【図１１】

図１１



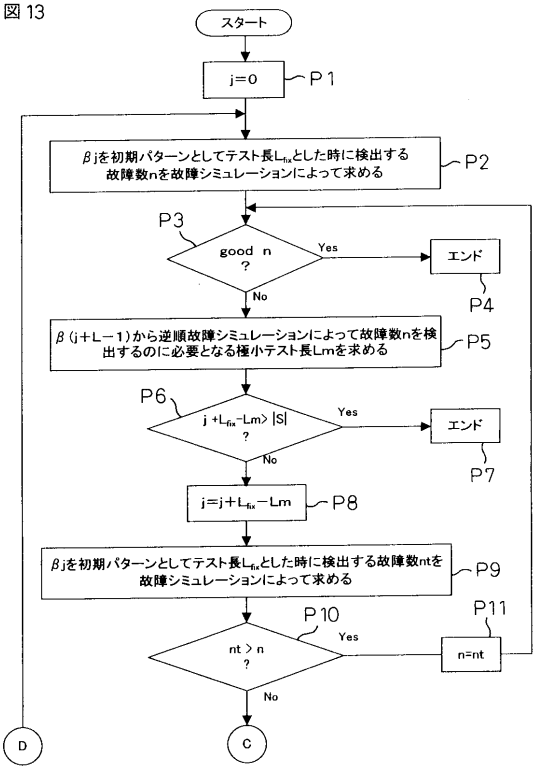
【図１２】

図 12

```
procedure SEARCH_JSTAR
begin
  j := 0;
  repeat
    repeat
      n_t := N(β_j, l_fix);
      l_min := L_R(β_{j+L_fix-1}, n_t);
      j := j + l_fix - l_min;
      j_min := j;
      j := j + 1;
      l := L(β_j, n_t)
    until( l < l_fix and j < 2^n - 1 );
    repeat
      j := j + l - l_fix + 1;
      l := L(β_j, n_t)
    until( l > l_fix - 1 and j < 2^n - 1)
  until( j < 2^n - 1 );
  n* := n_t;
  j* := j_min
end.
```

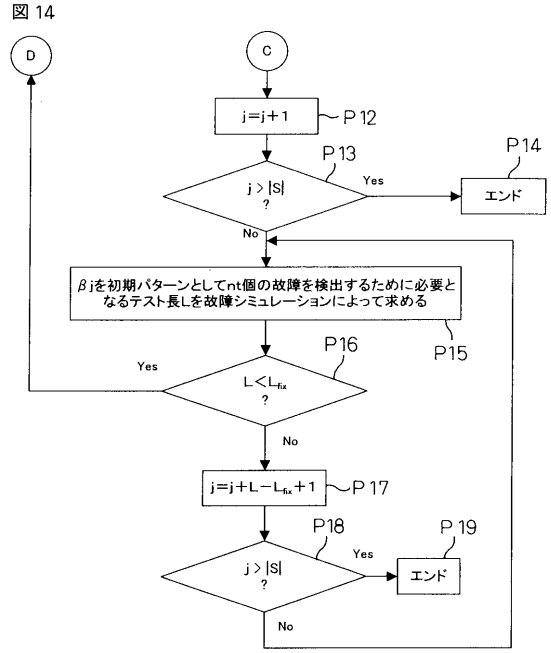
【図１３】

図 13



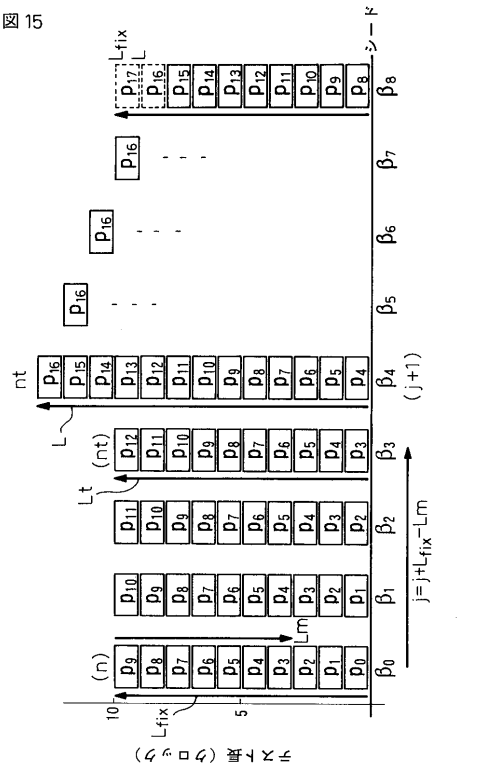
【図１４】

図 14



【図１５】

図 15



フロントページの続き

(72)発明者　市野　憲一
東京都世田谷区上馬５－８－２４
(72)発明者　新井　雅之
東京都小平市小川東町１８２１－２３
(72)発明者　福本　聡
神奈川県川崎市多摩区菅５－１０－２５－１０２
(72)発明者　岩崎　一彦
神奈川県横浜市港北区岸根町４０９－６－５０４
(72)発明者　正田　剛史
東京都国分寺市戸倉４－７－４８
(72)発明者　佐藤　正幸
群馬県高崎市倉賀野町４５９８－１６

審査官　下中　義之

(56)参考文献　特開２００２－２５０７５５（ＪＰ，Ａ）
特開平０９－３１８７１１（ＪＰ，Ａ）
特開平０３－２９６６７６（ＪＰ，Ａ）
特開平０２－０１５５００（ＪＰ，Ａ）
特表２００３－５３５３４３（ＪＰ，Ａ）
国際公開第０１／０９２９０３（WO，Ａ１）

(58)調査した分野(Int.Cl.^{7}，ＤＢ名)
G01R 31/28
G01R 31/3183
G06F 17/50　670