ISSN 2186-5647

静的テスト圧縮のための多重目標故障テスト生成を用いたＭバイＮアルゴリズム

○原　侑也・山崎紘史・細川利典（日大）・吉村正義（京都産大）

1. はじめに

近年，半導体微細化技術の進歩に伴い，大規模集積回路（Large Scale Integrated circuits LSI）が大規模化・複雑化し，テストコストの増加が問題になっている[1]．テストコストはテストパターン数に比例するため，テストパターン数を削減することにより，テストコストを削減することが期待できる．

テストパターン数を削減する手法として，テスト圧縮[2][3]が提案されている．小規模の回路では，ほぼ最小のテスト集合を得るテスト圧縮法が過去に提案されている[2]．しかしながら，大規模回路において最小のテスト集合を得るためには計算量が多く，現実的な計算時間での適用は困難である．

テスト圧縮手法には，テスト生成中にテスト圧縮を行う動的圧縮法[2]と，テスト生成後にテスト圧縮を行う静的圧縮法[3]がある．大規模な回路に適用可能な静的圧縮法の一つとして，冗長なテストパターンをN(N＜M)個生成し，このテストパターンを初期テスト集合に追加することにより，他のテストパターンをＭ個削除することで結果的にテストパターン数を削減する手法であるＭバイＮアルゴリズムが提案されている[4][8]．この手法において，故障検出率[5]を低下させずにテストパターンを圧縮するには，Ｎ個のテストパターンにおいて，削除するＭ個のテストパターンのみで検出される複数の故障を全て検出することが必要条件である．

ＭバイＮアルゴリズムでは，削除するテストパターンをM(N<M)個選択する．次に，選択したテストパターンでのみ検出できる故障をすべて検出できるＮ個のテストパターンを生成し，選択したＭ個のテストパターンを削除する．すなわち，再生成するテストパターンが生成可能か否かは，選択したＭ個のテストパターンに強く依存する．そのため，選択するＭ個のテストパターンの組み合わせによって，最終的なテストパターン数が異なる．　本論文では，必須割当てと複数目標故障テスト生成(Multiple Target Test Generarion:MTTG)[7]を用いたＭバイ Ｎアルゴリズムを提案する．

本論文の構成は，2章でＭバイＮアルゴリズムに必要な予備知識について説明し，3章でＭバイＮアルゴリズムについて説明する．4章では提案手法である必須故障の必須割当てに着目したテストパターン選択法とテストパターン再生成について説明し，5章で実験結果を示し，6章で結論と今後の課題について述べる．

2. 諸定義

本章ではＭバイ Ｎアルゴリズムに必要な必須故障，必須割当てを定義する．

2.1. 必須故障

必須故障[4]とは，与えられた回路に対して生成されたテスト集合Tにおいて，故障ｆがテストパターン tp(tp∈ T)によって検出可能であるが，tp 以外のテストパターンでは検出不可能であるとき，ｆをｔｐの必須故障という．一方，テストパターン tp が必須故障を持たないとき，テストパターン tp を冗長パターンという．テスト集合Tが冗長パターンを含まないとき，Tを極小テスト集合といい，ＭバイＮアルゴリズムでは初期テスト集合として極小テスト集合を用いる．

また Ｔ の部分集合T′(∈ T)でのみ検出可能な故障をT′の必須故障と定義する．

2.2. 必須割当て

必須割当てとは故障を検出するために必要不可欠な信号線の論理値割当てであり，必須割当ては’0’,’1’,’X’の3値で表現される．

図1に必須割当ての例を示す．故障 f1 は信号線ｅの0縮退故障，故障 f2 は信号線ｅの1縮退故障を示している．これらの故障を検出するために，必要不可欠な論理値を各信号線に割当てる．その結果，故障 f1 の必須割当ては，信号線 a，b，e，ｇが1であり，信号線ｆが0である．故障 f2 の必須割当ては，信号線 e，f，ｇが0である．

AN M by N Algorithm Using Multiple TargetTest Generation for Static Test Compaction

Yuya HARA,Hiroshi YAMAZAKI,
Toshinori HOSOKAWA and Masayoshi MATSUNAGA

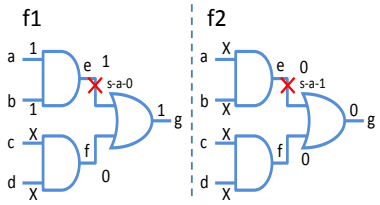

図 1. 必須割当て例

3. M バイ N アルゴリズム

本章では M バイ N アルゴリズム[4][8]について説明する．M バイ N アルゴリズムは，極小テスト集合 T に追加しても故障検出率が向上しない冗長なテストパターンを N(N<M)個追加し，テスト集合 T から M 個のテストパターンを削除するアルゴリズムである．

例として 2 バイ 1 アルゴリズムの説明をする．まず削除するテストパターン tp1 と tp2 を選択し，テスト集合T_1とする．追加するテストパターンの生成は，T_1の必須故障に着目して行う．T_1の複数の必須故障を同時に検出することが可能なテストパターンが生成可能か否かを判定する．すなわち，tp1 と tp2 のみで検出される故障全てを検出するテストパターンを生成する必要がある．したがって，T_1の必須故障を検出可能なテストパターン tp3 が生成可能か否かの判定を行う．可能であれば生成したテストパターン tp3 をテスト集合 T に追加し，tp1 と tp2 をテスト集合Tから削除する．テスト生成が不可能であると判定された場合は，テストパターンの再選択を行い，同様の処理を実行する．

4. 必須割当てと複数目標故障検出テスト生成を用いた M バイ N テスト圧縮

本章では，提案手法である必須割当て情報を用いた M バイ N アルゴリズムを説明する．提案手法では必須割当て情報を用いて同時検出可能グラフを作成する．

次に，同時検出可能グラフを用いて各テストパターンの必須故障の必須割当て情報を基に削除するテストパターンを選択し，選択したテスト集合の必須故障を充足可能性問題を用いた複数目標故障テスト生成を実行する．

4.1 同時検出可能グラフ

(定義 1:必須割当てベクトル)

信号線$(l_1, l_2, \cdots, l_{L-1} l_L)$をもつ回路の全信号線の論理値(0, 1, X)割当てベクトルLTを考える(L は全信号線数)．LTの信号線l_iの値をLT(i)と表記する$(1 \leqq i \leqq L)$．故障 f の必須割当てされる信号線をl_jとすると，LT(j)に必須割当てされる値(0 又は 1)を設定し，必須割当てされない信号線をl_kとすると，LT(k)に X を設定する$(1 \leqq j, k \leqq L)$．このとき LT を f の必須割当てベクトルという．

(定義 2:必須割当て両立可能)

信号線$(l_1, l_2, \cdots, l_{L-1} l_L)$をもつ回路の故障$f_1$と$f_2$の必須割当てベクトル$LT_1$，$LT_2$を考える(L は全信号線数)．$LT_1$，$LT_2$の信号線$l_i$の値をそれぞれ$LT_1(i)$, $LT_2(i)$と表記する．任意の i(1≦i≦L)について，次の二つの条件のうちいずれかを満たすとき，T_1とT_2は必須割当て両立可能という．

(1) $LT_1(i) = LT_2(i)$

(2) $LT_1(i) = X$，又は $LT_2(i) = X$

$LT_1(i)$と$LT_2(i)$が必須割当て両立可能であるときに，実際に併合して生成される必須割当てベクトルLTは表 1 に示す演算$\cap_T$を用いると任意の i$(1 \leqq i \leqq L)$について，次の式で表すことができる．

$$LT(i) = LT_1(i) \cap_T LT_2(i)$$

(定義:3　同時検出可能グラフ)

同時検出可能グラフは無向グラフ G=(V, E, t)であり，頂点 v∈V はテストパターンを表し，各頂点にはラベル$t: V \rightarrow LT$(LTは必須割当てベクトルを表す)が付けられている．また，辺$(u, v) \in E(u, v \in V, u \neq v)$は t(u)と t(v)が必須割当て両立可能であることを表す．

表 1. 圧縮演算$\cap_T$

$\cap_T$	0	1	X
0	0	φ	0
1	φ	1	1
X	0	1	X

図 2 に同時検出可能グラフの例を示す．例として，全信号線数が 6 本のテスト対象回路のテスト集合のテストパターン数は 6 個であり，各テストパターンの必須故障集合の必須割当てがv_1は 010X1X，v_2は 010X11，v_3は 01XXX1，v_4は 110X11，v_5は X10X01，v_6は 110XX1 とする．各頂点のラベルはテストパターンの必須故障集合の必須割当てであり，各辺は頂点の必須割当てが両立していることを表し，　2 個のテストパターンから成るテスト集合の必須故障を全て検出できるテストパターンが生成可能である可能性があることを示している．

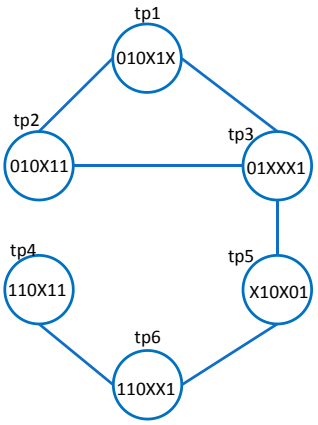

図 2. 同時検出可能グラフ

4.2 テストパターン選択

(定義 4:クリーク必須故障)

クリーク必須故障とは,同時検出可能グラフのあるクリークにおいて,クリークに含まれるテストパターンでしか検出できない故障を表す.

(定義 5:ハミング距離)

ハミング距離とは,必須割当て両立可能な必須割当てベクトル間の距離を表す.信号線$(l_1, l_2, \cdots, l_{L-1} l_L)$をもつ回路の故障$f_1$と$f_2$の必須割当てベクトル$LT_1$, LT_2が必須割当て両立可能であるとき,任意の$i(1 \leqq i \leqq L)$について,次の二つの条件のうちいずれかを満たすとき, LT_1とLT_2間のハミング距離は 1 増加する.

(1) $LT_1(i) = 0$(又は 1)かつ $LT_2(i)$=X

(2) $LT_1(i)$ = X かつ $LT_2(i) = 0$(又は 1)

本提案手法において,削除するテストパターンの選択は同時検出可能グラフと各クリーク必須故障の必須割当てベクトルを用いる.新たなテストパターンを生成できるか否かは,クリーク必須故障の必須割当てベクトルにおいて衝突が発生するか否かである. したがって, 3 バイ 2 アルゴリズムなど,複数のクリーク必須故障を複数のテストパターンで検出するには、どのテストパターンでどのクリーク必須故障を検出するかが重要になる.

提案手法では生成した同時検出可能グラフから最大クリーク抽出を行う.次に,抽出したクリークのテスト集合のクリーク必須故障を,クリークのノード数よりも小さなグループに分割し,グループごとにテスト生成を行う.グループ分割には各クリーク必須故障の必須割当てベクトル間でのハミング距離を用いて行う.各グループのハミング距離が大きくなるにつれ,
各グループの必須割当てベクトルの X の割合が減少しテストパターン生成に失敗する可能性が高まる.したがって,各グループのハミング距離が小さくなるようにグループ分割を行う.グループ数の初期値を 1 とし,テストパターンの生成に成功したならば,クリークに含まれる全てのテストパターンを初期テスト集合から削除し,新たに生成したテストパターンを初期テスト集合に追加する.テストパターンの生成に失敗した場合は,グループ数を 1 個増加させて,同様の処理を行う.

生成した同時検出可能グラフの,最大クリークサイズが 4 であり,最大クリークに含まれるテストパターンのクリーク必須故障が f1,f2,f3,f4 であり, 各故障の必須割当てベクトルが f1:11X0XX, f2:1110X1, f3:XX1001, f4:XX1X01 である場合の例について説明する.まず,クリークに含まれる全てのクリーク必須故障を検出する 1 個のテストパターンを生成する.生成に成功した場合はクリークに含まれるテストパターンを初期テスト集合から削除し,生成したテストパターンを追加する.失敗した場合はグループ数を 2 に増加させ,クリーク必須故障のグループ分割を行う.まず,各クリーク必須故障間の必須割当てベクトルのハミング距離を計算する.次に,ハミング距離が最大の必須故障のペアを選択し, 2 個のグループにそれぞれを追加する. f1,f4 間のハミング距離が 6 で最大であるので, f1 と f2 をそれぞれグループ G1,G2 に追加する. よって G1={f1} となり, G1 の必須割当ては G1: 11X0XX となる. 同様に G2={f4}, G2:XX1X01 となる. 次に, まだグループに分類されていないクリーク必須故障 f2,f3 について考える. それぞれのグループの必須割当てベクトルとのハミング距離が最小なクリーク必須故障を 1 個選択し,ハミング距離が最小なグループに追加する. f3 と G2 間でのハミング距離が 1 となり最小であるので f3 を G2 に追加する.よって,G2={f3,f4},G2:XX1001 となる. 同様に処理を行い,f2 を G1 に追加し,G1={f1,f2}, G1:1110X1 となる. 全ての故障をグループに分類したので各グループのクリーク必須故障を検出するテストパターンを生成する.テストパターンが全て生成された場合,クリークに含まれるテストパターンを初期テスト集合から削除し,生成したテストパターンを追加する.生成に失敗した場合はグループ数を 3 に増加させ,同様にクリーク必須故障のグループ分割を行い、テスト生成を行う. グループ数がクリークサイズ−1 まで処理を行い,テスト生成に失敗した場合は,他のクリークを抽出し同様の処理を行う.

4.3 複数目標故障テスト生成

新しく追加するテストパターンの生成においては,グループごとに含まれるクリーク必須故障を検出可能な 1 個のテストパターンを生成する必要がある.本論文では,充足可能性問題を用いた複数目標故障テスト生成[7]を用いる.

充足可能性問題を用いた複数故障の同時検出のためのテスト生成は,まず回路情報と故障情報からテスト生成回路モデルを作成する.次に,作成したモデルを CNF に変換し充足可能性の判定を行うことでテスト生成する.

初めにテスト対象となる回路に対して,正常回路と同時に検出する故障数だけ故障回路を用意する.次に正常回路とそれぞれの故障回路の入力に共通の外部入力を接続する.外部出力に故障の影響を伝搬させるためには,正常回路の外部出力値と各故障回路の外部出力値が異なる必要がある.そのため,片方の外部出力を 1 (真), もう一方を 0(偽)にする必要があるため, 正常回路の外部出力と各故障回路の外部出力を EXOR ゲートの入力に接続する.故障の影響の観測は少なくとも 1 個の外部出力に伝搬できればよいため, 正常回路と各故障回路の外部出力を接続した EXOR ゲー

トの出力を OR ゲートの入力に接続する．次に，各 OR ゲートの出力が常に 1(真)になる情報を付加する．以下のように生成したテスト生成回路モデルを CNF に変更し充足可能性を判定し，充足可能と判定されればテスト生成が成功となり外部入力に割当てた値がテストパターンとなる．

図 3 に 2 個の故障を検出するテストパターンを生成するためのテスト生成回路モデルを示す．

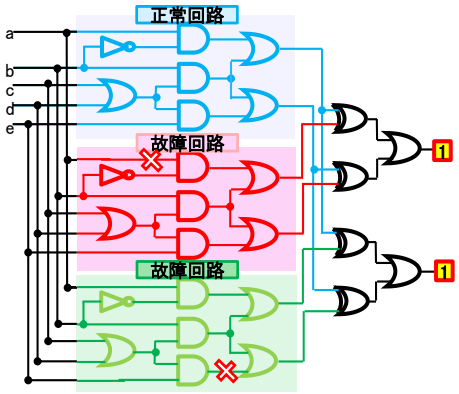

図 3．テスト生成回路モデル

5．実験結果

本論文で提案した多重目標故障テスト生成を用いた M バイ N アルゴリズムを 3 バイ N アルゴリズムとして実装し，実験を行った．実験では故障モデルを単一縮退故障とし，ISCAS'85 ベンチマーク回路を用いた．

表 1 は初期テスト集合のうち各テストパターンで検出される必須故障数が 40 以下のテストパターンのみで同時検出可能グラフを作成し，3 バイ N アルゴリズムを実行した結果である．またテスト再生成には研究室内で開発した SAT-ATPG を用いた．初期テスト集合は，Synopsys 社の TetraMax を用いて動的圧縮を行ったテスト集合を用いた．L.B[6]はテストパターン数の下界である．[4]は文献[4]の 2 バイ 1 アルゴリズムの結果である．

表 2. 実験結果

回路名	初期テスト集合	L.B[6]	提案手法	[4]
c880	30	12	20	21
c1355	85	84	84	84
c1908	109	99	102	106
c2670	64	42	45	45
c3540	126	80	90	91
c5315	74	37	47	44
c6288	22	6	14	14
c7552	101	52	88	80

6．おわりに

本論文では，必須割当てと複数目標故障検出テスト生成を用いた M バイ N テスト圧縮法について提案した．

今後の課題としては，テスト圧縮の時間削減と M の数を増加させた実験とその評価，またテストパターン分解を用いた M バイ N テスト圧縮などが挙げられる

参考文献

[1]Y. Sato, T. Ikeya, M. Nakao, and T. Nagumo, “A bist approace for Very Deep Sub-Micron (VDSM) Defects”, Proc. International Test Conference, pp. 283-291, 2000.

[2]Y. Matsunaga, “MINT-An exact algorithm for finding minimum test set”, IEICE trans. Fundamentals vol. E76-A, pp1652-1658, 1993.

[3]Seiji Kajihara, Irith Pomeranz, Kozo Kinoshita and Sudhakar M. Reddy
” Cost-Effective Generation of Minimal Test Sets for Stuck-at Faults in Combinational Logic Circuits”, 30th ACM/IEEE Design Automation Conference, pp102-106, 1993.

[4]Seiji Kajihara, Irith Pomeranz, Kozo Kinoshita and Sudhakar M. Reddy
” On Compaction Test Sets by Addition and Removal of Test Vectors”, IEEE, pp. 202-207, 1994.

[5]M. Abramovici, M. A. Breuer and A. D. Friedman,” Digital systems testing and testable design”, IEEE Press, 1995.

[6] I. Pomerara, and S. M. Reddy,
″Generalization of Independent Fault Sets for Transition Faults,″ 1992 IEEE VLSI Test Symposium, pp. 7-12, April 1992.

[7]Eggersgluss, S, Univ. of Bremen, Bremen, Germany , Krenz-Baath, R. , Glowatz, A. , Hapke, F. “A NEW SAT-based ATPG for Generating Highly Compacted Test Sets”
IEEE, pp. 230 - 235, 2012.

[8] Ilker Hamzaoglu and Janak H. Patel.
“ Test Set Compaction Algorithms for Combinational Circuits”,
IEEE Council on Electronic Design Automation. pp. 957-963