ISSN 2186-5647



2-36

# 順序回路のテスト不可能故障判定の評価

日大生産工(院)　○秋山　正碩　　日大生産工(院)　山崎　紘史
日大生産工　　細川　利典　　京産大　　吉村　正義

## 1. はじめに

近年大規模集積回路(Large Scale Integrated circuits：LSI)の大規模化・複雑化に伴い，歩留まりの低下が問題となっている．歩留まり向上策の 1 個として過剰テストの抑制が挙げられる．一般的に過剰テストとは，LSI の機能動作に影響を与えない故障を検出するためのテストのことである．LSI のテストとして広く使用されているフルスキャンテスト[1][2]では，回路内のフリップフロップの値を自由に設定・観測できるため，本来ならば機能動作ではあり得ない内部状態となり，また外部出力で正常回路の状態と故障回路の状態を区別できない場合も区別可能となり，過剰テストを行っていると考えられる．

過剰テストを抑制するテストとして順序回路のテスト生成を用いた非スキャンベーステストがある．一般の順序回路のテスト生成アルゴリズム[3]は回路構造のみに着目して，組合せ回路部を時間展開したモデルを用いている．

順序回路のテスト生成において，テスト不可能故障の判定に多くの時間を費やす．それゆえ，テスト不可能故障を高速に判定することが重要であり，順序回路のテスト不可能故障の判定法が数多く提案されている[4-8]．

文献[4]では回路構造に着目し，テスト不可能故障を判定する FIRE アルゴリズムが提案されている．この手法は回路構造から各信号線に論理値 0 と 1 を割当てる必要のある 2 つの故障集合を含意操作を用いて探索し，2 つの集合間衝突が発生する故障をテスト不可能故障とする手法である．また FIRE を拡張した手法[5]として，論理ゲートの割当て不可能状態からテスト不可能故障を判定するアルゴリズムも提案されている．例えば正常な AND ゲートではすべての入力が 1 となったとき出力は 1 となる．そのためすべての入力が 1 で出力が 0 という割当ては不可能状態となる．そのため，各入力で 1 を割当てる必要のある故障集合と出力で 0 を割当てる必要のある故障集合の積集合がテスト不可能故障となる．

文献[6-7]では最初の時刻の擬似外部入力を可制御，最終時刻の擬似外部出力を可観測とした k 時間展開モデルを用いてテスト不可能故障を判定する手法が提案されている．これは順序回路を時間展開したテスト生成モデル用いた手法であり，時間展開数 k が大きくなるほど本来の順序回路の機能動作に近づき，より多数のテスト不可能故障を判定できる可能性が高まるが，解の探索空間が増大し，テスト不可能故障の判定に時間を要する手法である．

文献[8]では回路の無効状態を用いた手法を用いたテスト不可能故障判定アルゴリズムを提案している．無効状態とは順序回路の機能動作においてとり得ないフリップフロップの状態である． この手法は状態信号が回路構造的に到達不可能な状態を同定しておくことで，その状態を検出に必要とする故障をテスト不可能故障と判定する手法である．

本論文では，過去に提案された手法を組み合せて，判定するテスト不可能故障数を増加させるテスト不可能故障判定フローを提案する．またテスト不可能故障を削除した故障集合に対して機能的時間展開モデルを用いたテスト生成法を適用して，テスト生成時間を評価する．

## 2. テスト不可能故障判定に関する従来研究

順序回路におけるテスト不可能故障の判定法が過去に多数提案されている．2．1 節に回路構造からテスト不可能故障を判定する FIRE アルゴリズムを説明する．2．2 節にマルチサイクルキャプチャテスト生成モデルを用いたテスト不可能故障判定法を説明する．2．3 節にフリップフロップ出力信号線における無効状態からテスト不可能故障を判定する手法を説明する．

### 2．1 FIRE

文献[4]で提案されている FIRE アルゴリズムは，回路内の信号線に 0 を割当てる必要のある故障集合と，1 を割当てる必要のある故障集合を求め，2 個の故障集合の積集合からテスト不可能故障を判定する手法である．

図 1 に FIRE によるテスト不可能故障判定の例を示す．図 1 の回路において検出のために信号線 a に論理値を割当てる必要のある故障を考える．まず信号線 a に 0 を割当てる必要のある故障集合を求める．このとき図 1(a) に示すように a に 1 を割当てて含意操作を行

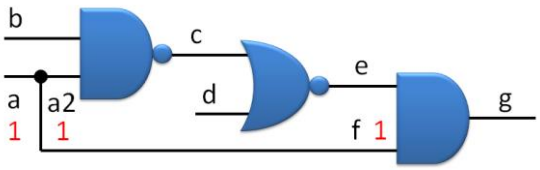
---

An Evaluation of Untestable Fault Identification for Sequential Circuits
Masahiro AKIYAMA,　Hiroshi YAMAZAKI,　Toshinori HOSOKAWA and Masayoshi

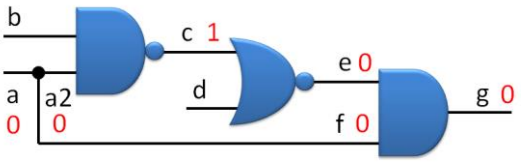
う．このとき a=1 のとき一意に決定する信号線の値は，その値と同じ縮退故障が発生したとき a=1 となると検出できない故障と考えることができるため，a=0 が検出するために必要である故障の集合となる．同様に a=1 を検出のために必要とする故障も a=0 から含意操作を行い故障集合を得る．このとき，信号線 a2，c，e，f は各ゲートの入力における制御値となっている．このとき他の入力に縮退故障があった場合，制御値によって伝搬不可能となる．そのため，a=1 を必要とする故障集合は図 1(b)に示された信号線値だけでなく，b/0，b/1，d/0，d/1，e/1，f/1 も含まれる．そして a=0 を必要とする故障集合と a=1 を必要とする故障集合の積集合から，図 1(c)に示すように f/1 がテスト不可能故障であると判定できる．

FIRE アルゴリズムは各信号線における 0 または 1 割当てが必要な故障集合からテスト不可能故障を判定していたが，これに各ゲートにおいてテスト不可能故障集合を求める拡張 FIRE アルゴリズム[5]が提案されている．

表 1 に拡張 FIRE アルゴリズムによるテスト不可能故障判定例を示す．表 1 では回路中の 2 入力 AND ゲート(入力信号線 X，Y，出力信号線 Z)において，信号線 X，Y，Z の 0，1 割当てが必要な故障集合を示している．各信号線単体からはテスト不可能故障が判定できないが，ゲートの不可能状態{X=1，Y=1，Z=0}を考えたとき，3 個の値割当てが必要な故障 G/0 がテスト不可能故障であると判定できる．




(a) a=1 による信号線割当て




(b) a=0 による信号線割当て

| 必要割当て | 故障集合 |
|---|---|
| a=0 | {a/1, a2=1, f=1} |
| a=1 | {a=0, a2=0, b=0,b=1, c=1, d=0, d=1, e=0, e=1, f=0, f=1, g=0} |
| 共通 | {f=1} |

(c) テスト不可能故障判定

図 1. FIRE によるテスト不可能故障判定

## 2. 2 マルチサイクルキャプチャテスト生成モデルを用いたテスト不可能故障判定

文献[6][7]ではマルチサイクルキャプチャテスト生成モデルを用いたテスト不可能故障の判定アルゴリズムを提案している．k=3 とした順序回路のマルチサイクルキャプチャテスト生成モデルを図 2 に示す．k 時間展開した際に各時刻の外部入力および 0 時刻目の疑似外部入力を制御可能とし，各時刻の外部出力および最終時刻の疑似外部出力を観測可能としている．この回路モデルに対し，単一故障[6]と多重故障[6]を探索し，少なくともどちらかでテスト不可能となった故障はテスト不可能故障と判定する．図 3(a)に単一故障モデル，図 3(b)に多重故障モデルの図を示す．

単一故障モデル，多重故障モデルは縮退故障モデルを仮定している．単一故障モデルは最終時刻にのみ故障影響があり，それ以外の時刻は正常回路とするモデルであり，多重故障モデルはすべての時刻において同じ信号線に同じ縮退故障が存在するモデルである．これらの故障モデルにおいてテスト不可能な故障は，順序回路においてもテスト不可能と判定することができる．このとき時間展開数が多いほど，本来の順序回路の機能動作に近づきテスト不可能故障の判定数が増加する可能性があるが，解の探索空間が増大し判定に要する時間も増大する．また 1 時間展開のとき検出できるテスト不可能故障は，順序回路における組合せ冗長故障となる．

表 1. 拡張 FIRE アルゴリズムによるテスト不可能故障判定

| | | | |
|---|---|---|---|
| 入力 | X | 0が必要 | { A/1, D/1 } |
| | | 1が必要 | { A/0, C/0, E/1, G/0 } |
| | Y | 0が必要 | { B/1, F/0 } |
| | | 1が必要 | { B/0, C/0, E/1, F/1, G/0 } |
| 出力 | Z | 0が必要 | { A/1, B/1, C/1, D/1, F/0, G/0 } |
| | | 1が必要 | { C/0, E/1 } |

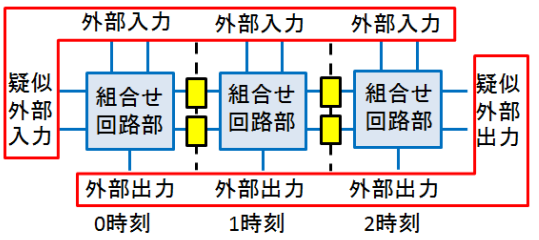
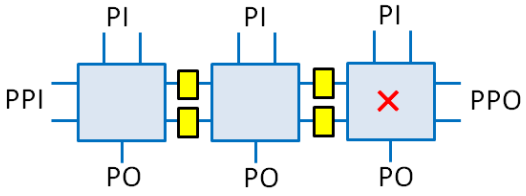
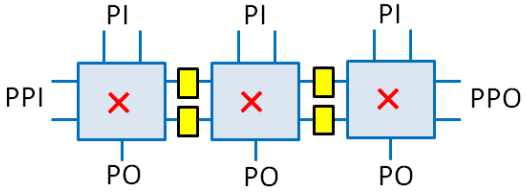



図 2. マルチサイクルキャプチャテスト生成モデルの回路モデル




(a) 単一故障モデル




(b) 多重故障モデル

図 3. 故障モデル

## 2. 3 無効状態を用いたテスト不可能故障判定法

文献[8]では順序回路の内部状態における無効状態と用いてテスト不可能故障を判定するアルゴリズムが提案されている．この手法ではまず 2. 1 節で説明した拡張 FIRE アルゴリズムにより判定されたテスト不可能故障を用いて無効状態を判定する．これはテスト不可能故障と判定された故障に対し擬似外部入力に 3 値で値割当てを行い外部出力で故障が検出されたとき，その擬似外部入力の割当ては無効状態と判定する．しかしながら大規模回路ではすべての割当てを試すことは困難であるため，文献[8]では 5000 個のランダムパタンを生成し，無効状態か否かを判定している．

得られた無効状態からテスト不可能故障を判定するために，無効状態を制約に加える．例えば 3 個のフリップフロップに対し{0, 1, 0}という割当てが無効状態であるとき，フリップフロップに{0, 1, 0}が割当てられた段階でバックトラックを行うようにする．これによりフリップフロップに{0, 1, 0}の無効状態を必要とするようなパターンを生成せずに ATPG を実行できる．

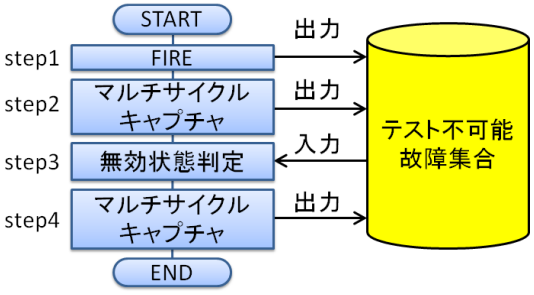



図 4. テスト不可能故障判定フロー

## 3. 提案テスト不可能故障判定

2 章で説明した 3 種類のテスト不可能故障判定アルゴリズムを用いたテスト不可能故障判定フローを図 4 に示す．

step1 では拡張 FIRE アルゴリズムを用いて回路構造から判定可能なテスト不可能故障を判定しテスト不可能故障集合を生成する．

step2 では step1 で検出したテスト不可能故障を除いた故障に対し，マルチサイクルキャプチャテスト生成モデルを用いたテスト不可能故障の判定を行い，step1 で判定したテスト不可能故障集合に追加する．

step3 では step1 と step2 から求めたテスト不可能故障を用いて無効状態を判定する．

step4 では step3 で判定した無効状態を回避する制約を付けて，既に判定されたテスト不可能故障を除いた故障に対し，マルチサイクルキャプチャテスト生成モデルを用いたテスト不可能故障判定を行い，最終的なテスト不可能故障集合を求める．

## 4. 実験結果

マルチサイクルキャプチャテスト生成モデルを用いてテスト不可能故障を判定し，得られたテスト不可能故障から無効状態を判定し，再びマルチサイクルキャプチャテスト生成モデルを用いてテスト不可能故障を判定した．実験に使用したパソコンは Windows 7，メモリ 8GB，CPU は Intel Core i7-2600 CPU で 3. 4GHz である．対象回路は ISCAS’89 ベンチマーク回路である．

表 3 に各時間展開におけるテスト不可能故障数を示す．表は左から回路名，代表故障数，各時間展開数におけるテスト不可能故障，文献[9]のテスト不可能故障数である．表中のハイフンは時間が足りず実験結果が得られなかったことと，文献[9]に記載されていないことを示す．この結果から s386 では[9]と同数のテスト不可能故障数を判定でき，s820 では文献[9]で打ち切られた故障がテスト不可能故障に含まれていることを示せた．

表 4 に無効状態を制約としてテスト生成(k=1)することでテスト不可能故障をより多く判定できた回路を示す．マルチサイクルキャプチャテスト生成のみの実験であるため，無効状態を追加しても最大でテスト不可能故障数は 2 個しか増加しなかったが，無効状態を用いたテスト不可能故障数の判定が有効であることを示した．

## 5. おわりに

本論文ではテスト不可能故障を判定するアルゴリズムを提案し，さらにマルチサイクルキャプチャテスト生成モデルを用いたテスト不可能故障判定アルゴリズムと，無効状態判定アルゴリズムを実装し，実験を行った．実験結果から拡張 FIRE アルゴリズムなしでも s386 ではすべてのテスト不可能故障を検出できたが，s510 など，全くテスト不可能故障を判定できない回路も存在した．今後の課題として拡張 FIRE アルゴリズムを実装し，より多くのテスト不可能故障を判定する．そして得られたテスト不可能故障からより多くの無効状態を判定し，この無効状態を利用してさらなるテスト不可能故障判定を行う．

## 参考文献


[1]. H. Fujiwara “Logic Testing and Design for Testability”, The MIT Press, 1985

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

[3]. MICHAEL H. SCHULZ, ERWIN TRISCHLER, THOMAS M. SAPFERT, “SOCRATES：a highly efficient automatic test pattern generation system”, IEEE Transactions on Computer-Aided Design, Vol. 7, No1, Jan, 1988, pp. 126-137

[4]. M. A. Iyer, D. Long, and M. Abramvici, “FIRE：a fault independent combinational redundancy identification algorithm”, IEEE Trans, VLSI systems, June 1996, pp. 295-301

[5]. Michael S. Hsial, “Maximizing Impossibi lities for Untestable Fault Identification”, IEEE Press, Mar 2002, pp. 949-953

[6]. Vishwani D. Agrawal and Srimat T. Chakradhar, “Combinational ATPG theorems for identifying untestable faults in sequential circuits”, IEEE Trans, Vol. 14, Sep 1995, pp. 1155-1160

[7]. 大森 悠翔，小河 広志，細川 利典，吉村 正義，山崎 浩二，“マルチサイクルキャプチャテストを用いたフルスキャン設計回路の縮退故障テスト生成”，電子情報通信学会，DC 研究会，Feb 2008

[8]. Xiao Liu and Michael S. Hsiao, “Constrained ATPG for Broadside Transition Testing”, IEEE Press, Nov 2003, pp. 175-182

[9]. T. Nierman, J. H. Patel, “HITEC：a test generation package for sequential circuits”, IEEE Press, Feb 1991, pp. 214-218


表 4. 無効状態を用いた追加判定テスト不可能故障数

| 回路名 | 代表故障数 | MAX | +無効状態 | [9] |
|---|---|---|---|---|
| s208 | 215 | 57 | 59 | 78 |
| s349 | 350 | 8 | 9 | 13 |
| s420 | 430 | 210 | 212 | 251 |
| s838 | 857 | 522 | 524 | - |

表 3. k 時間展開におけるテスト不可能故障数

| 回路 | 代表故障数 | テスト不可能故障判定時間展開数 | | | | | | [9] | 回路 | 代表故障数 | テスト不可能故障判定時間展開数 | | | | | | [9] |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 1 | 5 | 10 | 15 | 20 | 25 | | | | 1 | 5 | 10 | 15 | 20 | 25 | |
| s208 | 215 | 0 | 57 | 57 | 57 | 57 | 57 | 78 | s526 | 555 | 1 | 25 | 27 | 27 | 28 | 28 | - |
| s298 | 308 | 0 | 27 | 35 | 35 | 35 | 35 | - | s641 | 467 | 0 | 0 | 0 | 0 | 0 | 0 | 62 |
| s344 | 342 | 0 | 3 | 5 | 5 | 5 | 5 | 11 | s713 | 581 | 38 | 38 | 38 | 38 | 38 | 38 | 105 |
| s349 | 350 | 2 | 6 | 8 | 8 | 8 | 8 | 13 | s820 | 850 | 0 | 35 | 35 | 35 | 35 | 35 | 32 |
| s382 | 399 | 0 | 4 | 4 | 8 | 8 | 8 | - | s832 | 870 | 15 | 52 | 52 | 52 | - | - | 47 |
| s386 | 384 | 0 | 70 | 70 | 70 | 70 | 70 | 70 | s838 | 857 | 0 | 522 | 522 | 522 | - | - | - |
| s400 | 424 | 6 | 10 | 10 | 14 | 14 | 14 | - | s953 | 1,079 | 0 | 10 | 10 | 10 | - | - | 990 |
| s420 | 430 | 0 | 210 | 210 | 210 | 210 | 210 | 251 | s1196 | 1,242 | 0 | 0 | 0 | 0 | - | - | 3 |
| s444 | 474 | 14 | 18 | 18 | 22 | 22 | 22 | - | s1238 | 1,355 | 69 | 69 | 69 | 69 | - | - | 72 |
| s510 | 564 | 0 | 0 | 0 | 0 | 0 | 0 | 564 | | | | | | | | | |