含意操作に基づく順序回路の冗長故障判定

藤本 幸宏[1] ・ 渡邊 孝博[2]

[1] 日本電信電話株式会社

[2] 知能情報システム工学科

論理回路の冗長故障は，回路の故障検査(テスト)では検出できない故障であり，これをあらかじめ検査項目の故障リストから除いておくことは効率的なテスト生成に欠かせない．そこで，本研究では，特に回路が複雑になる順序回路に対して，2種類の効率的な冗長故障判定方法の検討を行う．我々はこれらの2種類の方法を実装し，ISCAS'89ベンチマーク回路を用いて実験を行った．その結果，冗長故障判定を行わない時と比較して，テスト生成全体の効率を向上させることが確認できた．

Key Words: *Test generation, Redundant faults, Implication procedure, Chen method*

1. はじめに

LSIのテスト設計の分野では，論理回路の大規模化・高機能化に伴い，故障検査用テストパターンの作成は困難さを増しており，設計期間を短縮する上で大きな障害となっている．一般に，論理回路は，組合せ回路と記憶素子(フリップフロップ)を持つ順序回路であり，テスト生成をする場合，時刻に依存して各ゲートにおける入出力値の変化を考慮しなければならない．回路の階層的な分割や並列処理手法の導入による問題解決も試みられているが，未だ十分とは言えず，現状では順序回路における自動テストパターン生成(ATPG)は，LSI設計工程において最も計算パワーを必要とする処理の1つとなっている．

現在，順序回路のテスト生成を容易にするため，多くはスキャン回路設計等のテスト容易化設計を取り入れ，テスト生成問題を組合せ回路の問題に置き換えて扱っている．しかしながら，性能重視のRISCプロセッサやマイコン，ゲートアレイ等のASICにおいては，スキャン回路のオーバヘッド削減の要望がある．しかし，スキャン回路を持たない順序回路に対して，従来のテストパターン生成手法では，非常に多くの時間を必要とし，大規模な回路になると，実用的な時間での処理が不可能になる．そこで，高速に高品質なテストパターンを生成できる順序回路用テスト生成システムが望まれている．

本研究では順序回路のテスト生成を効率良く行うために，テスト生成の前処理として冗長故障を判定し，それらを取り除いてテスト生成をする含意操作法とChenの方法の2種類の方法について検討を行った．

尚，本研究で前提とする故障は，単一縮退故障(信号線が0または1に固定してしまう故障がただ1ヶ所発生)であり，実験に用いたデータはISCAS'89ベンチマーク回路である．

以下，第2章では準備として，故障，テスト生成，冗長故障について説明する．第3章では，今回用いた冗長故障判定方法である含意操作法とChenの方法について説明する．第4章では実験及び結果についての考察を行う．

2. 準備

(1) 故障

回路を構成する要素に物理的欠陥があれば，設計した論理が正しくても回路が正しい動作をしなくなる．このような物理的欠陥が回路の故障の要因である．論理回路の論理機能が故障によって別の論理機能に変化してしまう故障を論理故障あるいは単に故障と言う．

また，一般にテスト生成を行う故障のモデルとして，縮退故障がある．これは，回路中の信号値が0に固定してしまう0縮退故障(s-a-0故障)と1に固定してしまう1縮退故障(s-a-1故障)のことである．

(2) テスト生成

実際に回路に故障が存在するか否か，すなわち故障検出をするためには，回路に入力パターン(入力系列)を与え，それに対する回路の出力パターンや出力パターン(出力応答系列)を観測する．さらに，回路に故障

が存在することが判った時，それがどのような故障であるかを調べることを故障診断といい，故障検出や故障診断を総称してテストという．テストを行うための入力パターンをテストパターンという．テストパターン(系列)は，テストの対象となる回路の論理機能や回路を形成する要素の論理機能，要素間の接続情報，といった回路情報を元に作成される．故障検出の入力系列を作成することをテスト生成という．

(3) 冗長故障

故障回路の入出力応答が，正常な回路の入出力応答と等しい場合，入出力端子だけから見る限り，その故障を検出することは出来ない．これを冗長故障と呼ぶ．その冗長故障は，テスト生成の時、特に順序回路のテスト生成では困難を生じる．なぜならば，そのようなテストできない冗長故障を認識することは，非常に多くの時間を必要とし，場合によっては，限られた時間内には成功しないかもしれないからである．(その故障に対する全てのテストパターンを入力しても，テスト生成が不可能であるような場合．) そのような故障の探索に時間を費すことはテスト生成の効率を減らす原因となる．

ここで，図1を用いて冗長故障の例を説明する．

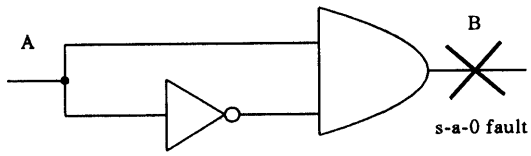

Fig.1 Redundant fault

この回路が正常な回路であるとして，信号線Aにどのような入力値を与えても，信号線Bは，0という値に固定される．一方，信号線Bにs-a-0故障があったとしても，やはり正常時と値が同じであるので，回路において故障が起きたのかどうかを判断することが出来ない．すなわち，信号線Bのs-a-0故障は，テスト生成不可能な冗長故障と判断できる．

3. 冗長故障の判定方法

(1) 含意操作法

含意操作[1]とは回路の各信号線に値(0または1)を設定し，一意的に値が決まる信号線を順次決定して行く操作である．この含意操作の過程において矛盾が生じる，すなわち，同一信号線に異なる値の設定の要求が生じる場合がある．これは，当初の設定値が論理的に無理であることを意味し，その信号線は当初の否定値に固定される．

つまり，その信号線における固定値と同じ値の縮退故障が生じた場合，正常時と同じ値を出力する故障となるので，出力側で故障が起きたのかどうか判断することが出来ない．この含意操作の例を図2に示す．

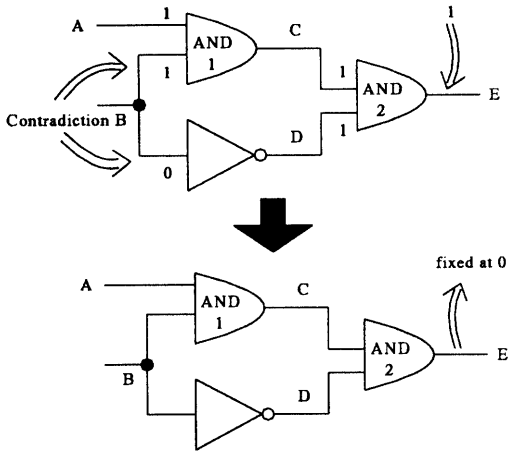

Fig.2 Implication step

例えば，信号線Eに論理値1を設定し，含意操作を行う．AND2の出力Eを1に設定するには，その入力C,Dを1に設定する必要がある．しかし，Dを1に設定することは，Bが0であることを意味し，一方，Cを1に設定することは，A，Bが1であることを意味するので，Bで矛盾が生じる．従って，Eを論理値1に設定できないことから，論理0に固定される信号線と判断することが出来る．この信号線Eのように正常時の論理値が固定される信号線を固定値信号線と呼ぶ．

このような固定値信号線ではその固定値に等しい縮退故障を検出できないため，これが冗長故障となる．

また，実際には順序回路を対象としているので，多時刻の回路を扱わなければならない．そのため，順序回路のフィードバックモデルを時間軸に展開し，全体としてループの無い組み合わせ回路として考える．

含意操作法による冗長故障判定の流れを図3に示す．

(2) Chen の方法

Chen の方法[2]では，まず，各信号線をどのような値に設定できるかを表す信号線の状態と，初期状態が分からないフリップフロップ (FF) の状態を調べるために，4つの状態 (値) を定義する．次に，時刻によって変化する各信号線の状態を調べるために，設定した状態を回路中に伝搬させる．そして，それらの情報を元に，ある状態 (値) に設定できない信号線と初期化できないFFを認識する．次に各信号線の状態を用いて

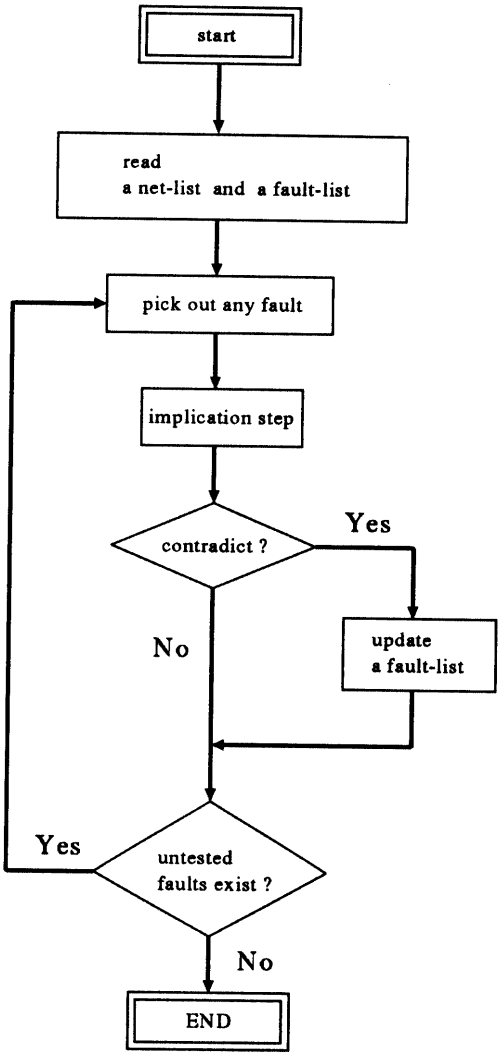

Fig.3 Implication procedure

5種類の冗長故障を判定する.

a) 状態の定義

まず, 次のような4種の状態を定義する.

- U:(Uncontrollable): 0にも1にも設定できない.
- U1:(Uncontrollable-1) : 1に設定できない.
- U0:(Uncontrollable-0) : 0に設定できない.
- C:(Controllable): 0にも1にも設定できる.

ここでは例として入力がA,Bである2入力のANDゲートの出力の状態遷移表を記す.

B \ A	C	U0	U1	U
C	C	C	U1	U1
U0	C	U0	U1	U
U1	U1	U1	U1	U1
U	U1	U	U1	U1

Fig.4 State transition for AND logic

b) 状態の伝搬

ここでは, 入力列を変化させることによって, 各信号線がどのような値に設定できるのか(あるいは, 設定できないのか)を調べるために, 定義した4種の状態を用いて, 以下のstep-1からstep-3の処理で状態を回路中に伝搬させる.

step-1

Primary Inputは, 0にも1にも設定できるのでCを, そしてFFの最初の出力値は不明であるので, Uを割り当てる.

step-2

step-1で割り当てた状態から, 定義した状態遷移表を用いて, 各信号線の状態をFFの入力まで計算する.

step-3

それぞれのFFについて, 入出力の状態が同じであるかを調べる. もし, 全てのFFの入出力の状態が同じであれば, 状態は既に安定した, と判断し, 処理を終了する. もし, そうでないならばクロックを進め, step-2で再び計算を続ける.

c) 冗長故障の判定

b) で得られた回路の各信号線の状態を用いて以下の type-1 から type-5 の 5 種類の冗長故障を判定する.

type-1

U0 である信号線の s-a-1 故障と, U1 である信号線の s-a-0 故障を, type-1 の冗長故障と定義する (図 5).

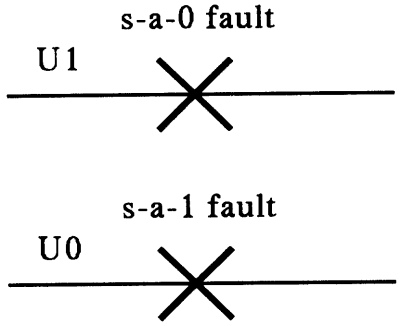

Fig.5 Redundant fault(type-1)

type-2

図 6 のような入力数 2 以上の AND ゲートにおいて入力 A が U1, すなわち, 1 には設定できないとする.

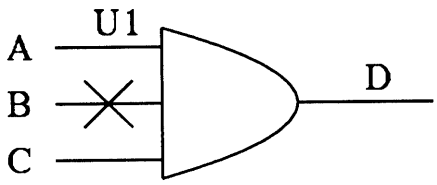

Fig.6 Redundant fault(type-2)

この時, ゲートを通じて B の故障を伝搬させようとすれば, B 以外の入力は, 全て 1 に設定できなければならないが, A が U1 であるので, B の故障を伝搬させることが出来ない.

つまり, AND ゲートか NAND ゲートの入力の内の 1 つでもが U1 の時, または OR ゲートか NOR ゲートの入力の内の 1 つでも U0 の時には, それ以外の全ての入力は type-2 の冗長故障と定義する.

type-3

l_f を故障の起こる信号線とし, l_f から Primary Output へ向けて故障の影響が伝搬する全てのパスの集合を P とする.

このとき, P の中のそれぞれのパスで, 故障の影響が伝搬するのを妨げられるならば, l_f を type-3 の冗長故障と定義する.

type-4

このタイプの故障は, 実際に回路に故障を注入して, その影響がどのように伝搬するのかを調べる. 故障の場所から Primary Output への全てのパスで, その伝搬が妨げられるなら, type-4 の冗長故障と定義する.

type-5

ある信号線から Primary Output へのパスが存在しなければ, その信号線の故障の情報を伝搬することが出来ない. これを type-5 の故障と定義する.

4. 実験

冗長故障をあらかじめテスト生成の前に判定することにより, どれだけの効果が得られるのか, を確かめるために含意操作法を用いた場合と, Chen の方法を用いた場合についての実験を行った.

実験には ISCAS'89 ベンチマーク回路とその故障リスト[3]を利用した. 実験全体の流れを図 7 に示す. 判定した冗長故障は, テスト生成が不可能なので, テスト生成の対象から外して考える, つまり, 故障リストから除去し, 更新された故障リストに対してテスト生成を行う. 判定した冗長故障及び処理時間を評価項目とした. また, 冗長故障判定を行わないでテスト生成を行った場合との比較を行った.

尚, 実験に使用した計算機は Sun-AS4050(48Mbyte) でプログラム言語は C 言語である. テスト生成には既存のソフトである HITEC[4]を用いた.

テスト生成として HITEC を用いた理由は, プログラムの各ステップが分かれて記述されており, この実験フローに組み込んで使用することが容易であったこと, および含意操作, Chen の方法ともに, HITEC の前処理として用いられた事例報告がなかったことによる.

(1) 含意操作実験

ここでは, 含意操作を行って, 冗長故障を判定し, それらを故障リストから除去し, テスト生成を行った場合と, ただ単にテスト生成を行った場合についての実験を行った.

含意操作で判定した冗長故障数 (図 7 の A) と HITEC で判定した冗長故障数 (図 7 の B) の比較を, 図 8 にまとめる.

これらより, s208.1, s641, s713 の回路の回路においては, 大部分の冗長故障を判定したと判断できる. 特に s713 では, HITEC によって見付けた全ての冗長故障を判定し, s5378 では, HITEC で見付けたよりさらに多くの冗長故障を判定した. s5378 を除いて, 含意操作で判定した冗長故障は, いずれも HITEC に含

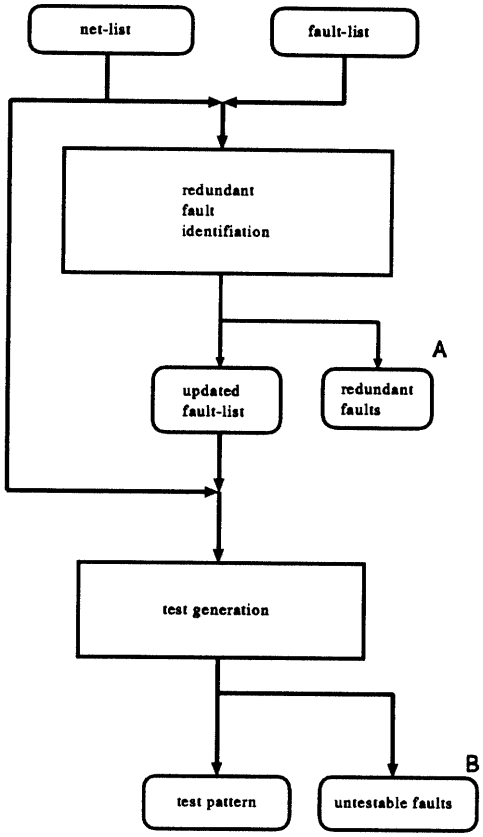

Fig.7 Experiment flow

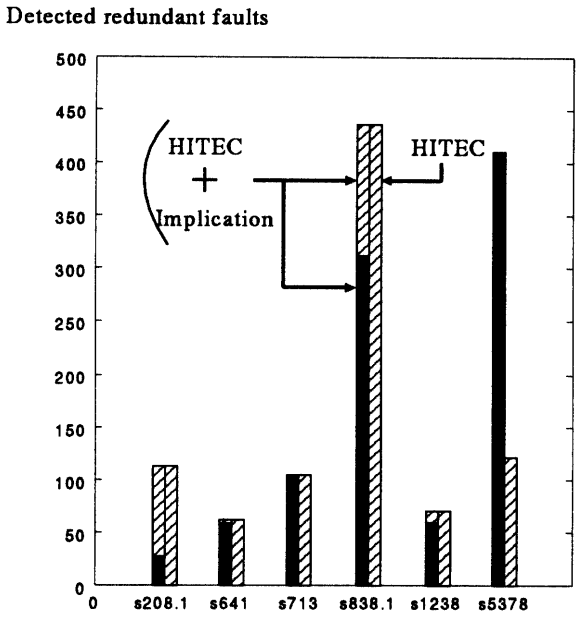

Fig.8 Comparison between Implication and HITEC

まれていた.

次に, HITEC との処理時間の比較を図9にまとめる.

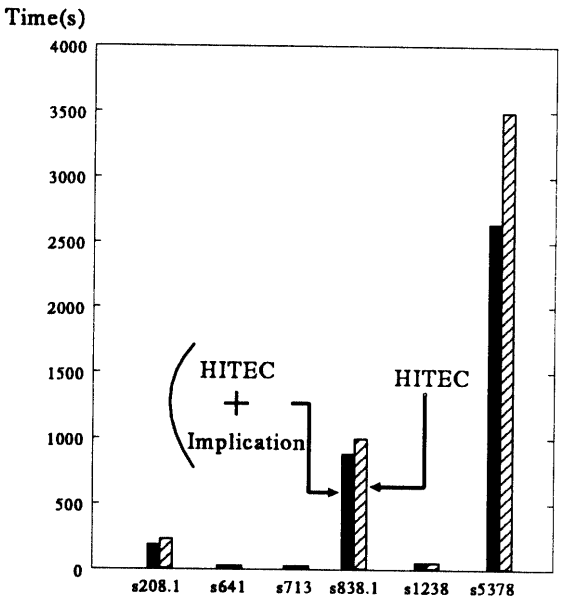

Fig.9 Processing time for test generation

これらより, いずれの回路の場合でも, 含意操作に要する時間は比較的少ないと言える. また, s5378 など, 回路が大規模になれば, テスト生成を単独で行う時と比べてはるかに時間を削減でき, 含意操作に要する時間の割合は少なくなった.

そして, s641 の回路では, 含意操作で 60 個の冗長故障を判定し, HITEC では, 残りの 3 個の冗長故障を判定したが, この冗長故障が冗長故障判定とテスト生成にどれだけ影響を及ぼすのかを調べた. つまり, 初めからこの 3 個の冗長故障を取り除いて含意操作, テスト生成をする追加実験を行った. その結果, 60 個の冗長故障判定に要した時間は, 2.663s, テスト生成に要した時間は, 11.542s となった. このとき, 含意操作で 1 個の冗長故障の判定に要した平均時間は, $2.663/60 = 0.044$s で, 3 個の冗長故障の判定に要した平均時間は, $(3.472 - 2.663)/3 = 0.207$s となり, この 3 個の故障が, 冗長判定に時間を要する, 見付けにくい故障であることが分かった. 回路中にはこのように見付けにくい故障が含まれているので, それらの性質を調査し, より効率良く判定する方法について検討する必要がある.

(2) Chen の実験

ここでは Chen の方法を取り入れた実験について述べる. 方針は含意操作の場合と同様である.

判定した冗長故障数の比較を図 10 にまとめる.

このことから，s713より小規模な回路ではHITECによって判定できる冗長故障がまだ残っているが，s838.1以上の大規模な回路ではChenの方がより多くの冗長故障を判定したことが分かる．特に，s5378では，805個も多くの冗長故障を判定した．

次に，処理時間の比較を図11にまとめる．

いずれの回路においても，Chenにかかった時間はテスト生成と比較して，少ないと言える．特に，s5378のような大規模な回路ほど，時間が削減されていることが分かる．

(3) 考察

図12及び図13に、含意操作(HITEC+Implication), Chen(HITEC+Chen),HITECのみの各方法について，判定した冗長故障数及び処理時間の比較をそれぞれ示す．

s208.1，s641，s713では，判定した冗長故障数はいずれの方法を用いても同じであった．また，処理時間は，HITEC > Chen > 含意操作の順となり，これらの回路に対しては含意操作が有効であると言える．

s838.1では，判定した冗長故障数は，HITEC = 含意操作 < Chenの順となり，処理時間はHITEC > 含意操作 > Chenの順となり，これらの回路では，判定した冗長故障数，処理時間ともにChenの方法が有効であると言える．

s1238では判定した冗長故障数はいずれの方法でも同じであった．処理時間はHITEC > 含意 > Chenの順となり，Chenの方法が有効であると言える．

s5378では判定した冗長故障数は，Chen > 含意操作 > HITECの順となり，処理時間はHITEC > 含意操作 > Chenの順となり，Chenの方法が有効であると言える．

このことから，含意操作，Chenのいずれの方法もHITECと比べて判定した冗長故障数，処理時間とも良い結果が得られている．含意操作とChenを比べると，比較的小規模な回路では含意操作の方が処理時間は少ないが，大規模な回路になれば，Chenの方が処理時間が少なく，しかもより多くの冗長故障を判定していることが分かった．

従って，それぞれの信号線に対して値を割り当て，矛盾が生じないかどうかを調べる含意操作よりも，各信号線の状態を計算した後に冗長故障判定を行うChenの方法の方が大規模な回路に対してはより有効であると言える．

そして，回路の規模が大きくなるからといって，必ずしもしそれと比例して処理時間が増える訳ではない．特にs838.1では，Primary Outputが1であり，しかもFFが他の回路よりも多いという特殊な回路構造を

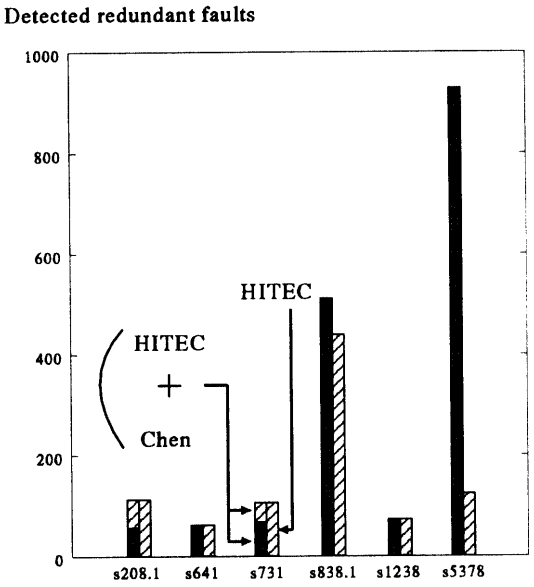

Fig.10 Comparison between Chen and HITEC

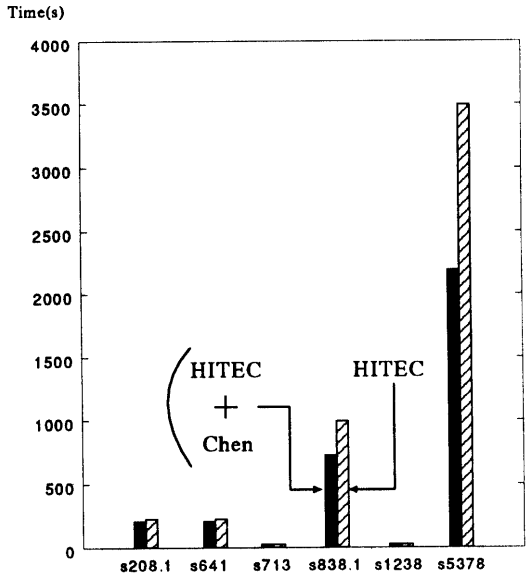

Fig.11 Processing time for test generation

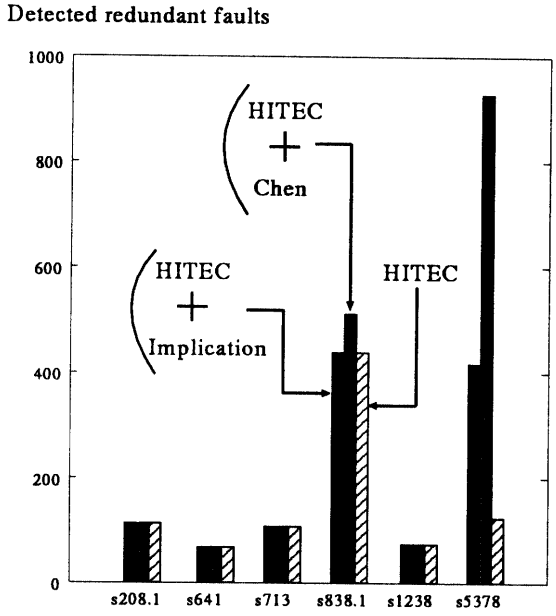

Fig.12 Comparison of detected redundant faults

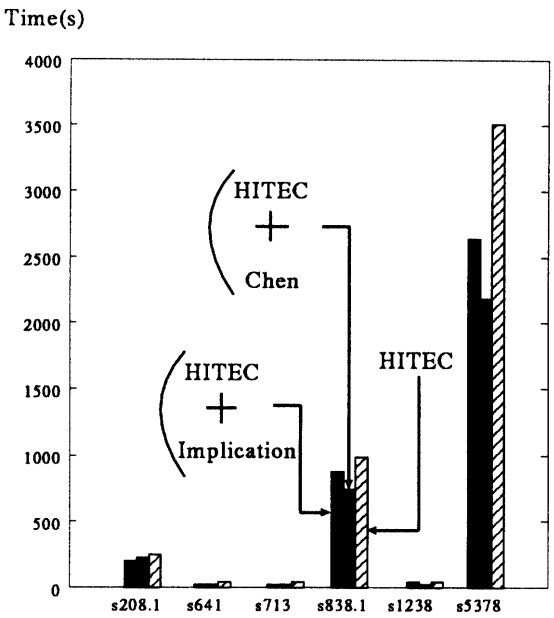

Fig.13 Comparison of processing times

しているので，どの方法を用いても同規模の回路よりもはるかに処理時間が多かった．

また，b) の含意操作実験で行ったように，回路によっては見付けにくい冗長故障があるので，それらの性質についてさらに調査する必要がある．

5. むすび

本研究では，順序回路におけるテスト生成の前処理として冗長故障を判定する2つの方法についての検討を行った．具体的には，テスト生成の前の段階で，冗長故障をできるだけ判定し，それらをテスト生成の対象から外して考えることにより，テスト生成の負担をできるだけ削減しようという方法である．1つは，含意操作を取り入れた方法，もう1つは，Chen の方法を取り入れた方法である．

どちらの方法も，信号線がどのような値になるのかを調べる方法であったが，テスト生成の段階で冗長故障を認識するよりもより多くの冗長故障を判定でき，テスト生成全体の処理時間を削減できることが分かった．

特に，Chen の方法では，今回実験した中で，最も多くの冗長故障を判定し，その結果としてテスト生成の効率を向上できることが確認できた．

今後の課題としては，以下のことが挙げられる．

- さらに大規模な回路においても，有効性を検討すること．
- 2つの方法を併用するなどして，更に多くの冗長故障判定が可能か否かを検討すること．
- 含意操作法の拡張，すなわち，含意操作で判定した冗長故障の情報を出力側へと伝搬し，その影響を他の信号線に伝えることによって他の冗長故障を見つけ出す方法を検討すること．

参考文献

1) E.Auth and M.H.Schulz, "A Test-Pattern-Generation Algorithm for Sequential Circuits", *IEEE Design & Test of Computers,* pp.72-85,1991.

2) H.-C.Liang , C.L.Lee and J.E.Chen, "Identifying Untestable Faults in Sequential Circuits", *IEEE Design & Test of Computers,* pp.14-23,1995.

3) F.Brglez,D.Bryan and K.Kozminski, "Combinational Profiles of Sequential Circuits", *Porc. IEEE Int. Symposium on Circuits and Systems,* pp.1929-1934,1989.

4) T.Niermann and J.H.Patel, "HITEC :A Test Generation Package for Sequential Circuits", *Proc. European Design Automation Conf.,* EDAC Association, Washington,D.C.,1991.

(1997.10.15 受理)

A REDUNDANT FAULT IDENTIFICATION METHOD FOR SEQUENTIAL CIRCUITS BASED ON IMPLICATION PROCEDURE

Yukihiro FUJIMOTO , Takahiro WATANABE

Redundant faults in logic circuits are faults that cannot be detected in fault simulation. Therefore removing any redundant fault from a fault list prior to test pattern generation is indispensable to efficient circuit testing.

In this paper two efficient methods are discussed to identify redundant faults in sequential circuits. We implemented these methods and experimented using ISCAS'89 benchmark circuits. Experimental results show that the both can improve the efficiency of a whole test generation system compared to a system without redundant-fault identification.