論　文

完全故障検出効率を保証するコントローラの非スキャンテスト容易化設計法

大竹　哲史[†]　　増澤　利光[†]　　藤原　秀雄[†]

A Non-Scan DFT Method for Controllers to Provide Complete Fault Efficiency

Satoshi OHTAKE[†], Toshimitsu MASUZAWA[†], and Hideo FUJIWARA[†]

あらまし　本論文では，有限状態機械（FSM）から論理合成されたコントローラに対して，完全故障検出効率を保証する非スキャンテスト容易化設計法を提案する．従来より，完全故障検出効率を達成するテスト生成法として，コントローラの組合せ回路部のテストパターンを生成し，スキャンフリップフロップを利用してそのテストパターンを印加するスキャン方式が利用されている．しかし，スキャン方式の場合，テスト系列が長くなる，実動作速度でテスト実行できない，などの問題が指摘されている．本論文では，コントローラの組合せ回路部のテストパターンをスキャンフリップフロップを用いず，FSMの状態遷移を利用して印加する手法を提案する．提案手法は，回路の実動作速度でテスト実行可能で，従来法と比べてテスト実行時間が短い．また，ベンチマークによる実験結果から，面積オーバヘッドも小さいことを示す．

キーワード　非スキャンテスト容易化設計，完全故障検出効率，コントローラ，実動作速度テスト

1. まえがき

VLSIのテストは，一般に困難な問題で費用がかかる．VLSIの回路規模の増大に伴い，テスト費用の削減及びテストの質の向上はますます重要になっている．テスト費用の評価尺度には，テスト生成時間や，テスト実行時間がある．また，テストの質の評価尺度として，故障検出効率（fault efficiency）がある．これは，テストに用いるテスト生成アルゴリズムによって生成されるテスト系列（またはテストパターン集合）が，与えられた回路中のテスト生成の対象となる故障を検出できる割合で，次式で定義される．

$$\frac{[\text{検出できた故障数}]+[\text{冗長と判明した故障数}]}{[\text{全故障数}]}$$

ここで，冗長故障とは，回路の動作に影響を与えない故障で，テスト生成アルゴリズムがその判定を行う．特に，故障検出効率が100%のとき，完全故障検出効率という．本論文で対象とする故障モデルは，単一縮退故障とする．

組合せ回路のテスト生成に関しては，効率の良いテスト生成アルゴリズム[1]が提案されており，実用的なテスト生成時間で完全故障検出効率をもつテストパターン集合を生成することができる．これに対して，順序回路の場合，フリップフロップ（FF）数をnとすると状態数が2^nになるため，順序回路のテスト生成には膨大な時間がかかり，一般に完全故障検出効率をもつテスト系列を生成するのは極めて困難である．そのため，順序回路をテストが容易な回路に設計変更するテスト容易化手法が用いられている．テスト容易化手法として，完全スキャン設計法（full-scan design for testability）や非スキャンテスト容易化設計法（non-scan design for testability）などが提案されている．

順序回路は，組合せ論理と状態レジスタ（FFの集合）から構成される．完全スキャン設計法[1]では，状態レジスタのすべてのFFをスキャンFF（回路の外部から値を制御及び観測できるFF）に置き換える．これにより，テスト実行の際に自由に状態レジスタの値を制御及び観測できるようになる．したがって，テスト生成の対象回路（テスト生成モデル）は，状態レジスタを擬似外部入力及び擬似外部出力に置き換えた組合せ回路となり，この組合せ回路に対してテスト生

[†] 奈良先端科学技術大学院大学情報科学研究科，生駒市
Graduate School of Information Science, Nara Institute of Science and Technology, Ikoma-shi, 630-0101 Japan

成することで順序回路のテスト系列を得ることができる．これにより，組合せ回路用のテスト生成アルゴリズムを用いて，実用的なテスト生成時間で完全故障検出効率をもつテスト系列を得ることができる．しかし，テスト実行において，スキャンFFに対してその値の制御及び観測を逐次的に行うので，状態レジスタのFF数が多くなると，テスト実行時間が長くなるという問題がある．更に，回路の実動作速度でテスト実行（at-speed test）できないという問題もある．Maxwellら[2]は，縮退故障に対するテスト系列を回路の実動作速度で印加すれば，遅いクロックで印加するよりもVLSIの欠陥をより多く検出できることを示した．したがって，回路の実動作速度でのテスト実行は重要である．

実動作速度でのテスト実行を可能とする方法として，非スキャンテスト容易化設計法が提案されている．Patelら[3]は，状態レジスタを，外部入力数だけのFFからなる状態レジスタ（SR1）と，残りのFFからなる状態レジスタ（SR2）に分割し，テスト実行の際にSR1に外部から値を設定できるように，マルチプレクサ（MUX）を追加してテスト容易化している．この手法では，回路の実動作速度でテスト実行可能である．また，外部入力数が状態レジスタのFF数以上の場合（SR2がない場合）は，完全スキャン設計法と同様に，組合せ回路用のテスト生成アルゴリズムを用いて完全故障検出効率をもつテスト系列を得ることができる．しかし，外部入力数が状態レジスタのFF数より小さい場合（SR2が存在する場合）は，テスト生成モデルが順序回路となるので，順序回路用のテスト生成アルゴリズムが必要となり，完全スキャン設計法に比べてテスト生成時間が長く，完全故障検出効率をもつテスト系列が得られる保証はない．更に，テスト生成モデル中のSR2の値を組合せ論理を介して制御しなければならないので，テスト系列にはSR2の値を制御するための系列が含まれ，テスト実行時間が長くなるという問題が生じる．

本論文では，VLSIのコントローラ（制御回路）の非スキャン方式によるテスト容易化設計の一手法を提案する．提案手法では，状態レジスタを擬似外部入力と擬似外部出力に置き換えた，組合せ回路に対してテスト生成を行う．得られたテストパターン集合の各テストパターンは，外部入力の値と擬似外部入力（状態レジスタ）の値で構成されている．テストパターンの状態レジスタの値が，与えられた回路の通常動作の状態遷移を用いて設定できるものについては，その機能を用いて設定する．通常動作の状態遷移を用いてテストのための状態に設定できないものについては，遷移できない状態を生成する論理を別途追加し，それを用いて設定する．本手法は，(1) 組合せ回路用のテスト生成アルゴリズムでテスト生成可能で，(2) 完全故障検出効率を保証し，(3) 従来法と比べてテスト実行時間が短く，(4) 回路の実動作速度でテスト実行可能という特色がある．また，本論文ではベンチマークを用いた実験により，テストのための付加回路の面積オーバヘッドは小さいことを示す．

以下，**2.** で用語を定義し，**3.** で提案手法を概説する．次に，**4.** でテスト容易化設計法を提案し，**5.** でテスト系列生成法及びテスト実行法を提案する．**6.** では提案手法と従来法との比較を行う．

2. 諸 定 義

コントローラ（制御回路）は，レジスタ転送レベルでは，有限状態機械（FSM, Finite State Machine）として記述される（図1参照）．FSMは，リセット状態をただ一つもち，リセット信号を入力することにより，リセット状態に遷移する．コントローラを実現する際には，FSMを論理合成し，図2のような順序回路を求める．FSMの論理合成において，FSMの各状態を状態レジスタのどの値で表すかが決定される．これを状態割当てと呼ぶ．本論文では，簡単のため，FSMの各状態に対応する状態レジスタの値は，一意に定まるものとする．論理合成の手法によっては，FSMの一つの状態に複数の状態レジスタの値が対応づけられることがある．この場合，異なる状態レジスタ値は異なる状態を表すものとみなし，論理合成された順序回路から状態遷移を求めることにより，本論文で提案する

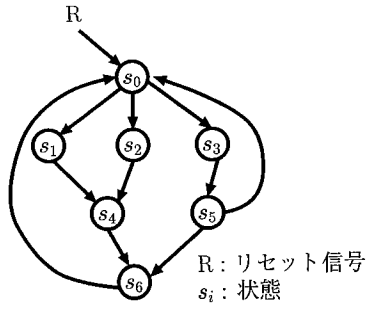
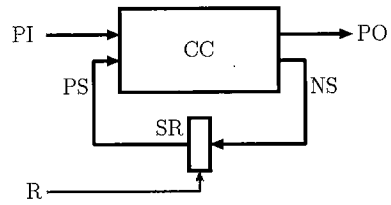
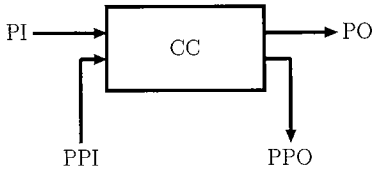


図1 有限状態機械（FSM）
Fig. 1 A finite state machine (FSM).



PI：外部入力　　CC：組合せ論理部 PS：現状態
PO：外部出力　　SR：状態レジスタ NS：次状態
R：リセット信号

図2　FSM を論理合成して得られた順序回路
Fig. 2　A sequential circuit synthesized from an FSM.



PPI：擬似外部入力　PPO：擬似外部出力

図3　組合せテスト生成モデル
Fig. 3　A combinational test generation model.

手法を適用できる．

FSM を論理合成して得られた順序回路（図 2）の状態レジスタが表現できる状態数は，状態レジスタの FF 数を n とすると 2^n である．この状態を以下のように二つに分類する．

[定義 1]　FSM を論理合成して得られた順序回路（図 2）の状態レジスタが表現できる状態のうち，FSM のリセット状態から到達可能な状態を有効状態と呼び，到達不可能な状態を無効状態と呼ぶ．　□

本論文では，図 2 の順序回路の組合せ論理部（CC）の単一縮退故障を対象としたテストについて考察する．本論文のテスト手法は，完全故障検出効率を保証するために，順序回路から組合せテスト生成モデルと呼ぶ組合せ回路を構成し，組合せテスト生成モデルに対してテスト生成を行い，求めたテストパターンを順序回路に適用できるように，順序回路のテスト容易化設計を行う．まず，組合せテスト生成モデルを定義する．

[定義 2]　FSM を論理合成して得られた順序回路（図 2）の状態レジスタを，擬似外部入力（PPI）及び擬似外部出力（PPO）に置き換えた回路を，この順序回路の組合せテスト生成モデルと呼ぶ（図 3 参照）．　□

組合せテスト生成モデルに対してテスト生成して得られた各テストパターンは，外部入力（PI）に対応する部分と，PPI に対応する部分に分割できる．PPI の値は，順序回路では状態レジスタに対応するので，順序回路の外部から直接には制御できない．そこで，テストパターンを PPI の値によって以下の二つに分類する．

[定義 3]　テストパターンの PPI（状態レジスタ）の値が有効状態であるならば，そのテストパターンを有効テストパターンと呼び，有効テストパターンでないテストパターンを無効テストパターンと呼ぶ．　□

テストパターンの PPI に対応する部分は，有効状態または無効状態である．そこで，テストパターンに現れる状態を以下のように定義する．

[定義 4]　有効テストパターンに現れる有効状態を有効テスト状態と呼び，無効テストパターンに現れる状態を無効テスト状態と呼ぶ．　□

3.　提案手法の概説

本章では，提案する手法を概説する．

提案する手法は，与えられた FSM（図 1）を論理合成した順序回路（図 2）の組合せ回路部分に対し，完全故障検出効率をもつテスト系列を高速にテスト生成でき，かつ，回路の実動作速度でテスト実行可能となるようにテスト容易化する手法である．

提案する手法では，完全故障検出効率をもつテスト系列を高速に生成するために，順序回路の組合せテスト生成モデル（図 3）に対して，組合せテスト生成アルゴリズムを適用する．生成された各テストパターンは，擬似外部入力（PPI）の値と外部入力（PI）の値で構成されている．このテストパターンを順序回路に適用するには，PPI の値を順序回路の状態レジスタに設定しなければならない．有効テストパターンに含まれる有効テスト状態（PPI の値の部分）は，与えられた FSM のリセット状態から遷移できるので，合成された順序回路の機能を用いて状態レジスタに設定する．

一方，無効テストパターンに含まれる無効テスト状態（PPI の値の部分）は，与えられた FSM のリセット状態から遷移できないので，合成された順序回路の機能を用いて状態レジスタに設定することはできない．そこで，すべての無効テスト状態を生成する論理を，合成された順序回路に別途付加する．ここで，テストのための回路を別途付加するとは，もとの順序回路の組合せ回路部分を変更せずに付加するということである．もとの順序回路の組合せ回路部分を変更しないこ

とにより，最初に求めたテストパターン集合が，もとの順序回路の組合せ回路部分に対して完全故障検出効率をもつことを保証している．このテスト容易化設計法を **4.** で提案する．

生成された各テストパターンは，テスト容易化された順序回路に対して，次のように時間展開して印加される．有効テストパターンについては，はじめに状態レジスタに有効テスト状態（PPI の値の部分）を設定するための入力系列を外部入力から印加し，次に外部入力からテストパターンの PI の値の部分を印加する．一方，無効テストパターンについては，はじめに無効テスト状態を生成する論理によって状態レジスタに無効テスト状態（PPI の値の部分）を設定し，次に外部入力からテストパターンの PI の値の部分を印加する．この，テスト系列生成及びテスト実行法は **5.** で詳しく述べる．

4. テスト容易化設計法

本章では，コントローラの非スキャンテスト容易化設計法を提案する．提案手法は，コントローラの組合せ回路部分の単一縮退故障に対して完全故障検出効率を保証する．

4.1 テスト容易化設計の手順

FSM が与えられた場合のテスト容易化設計の手順を以下に示す．

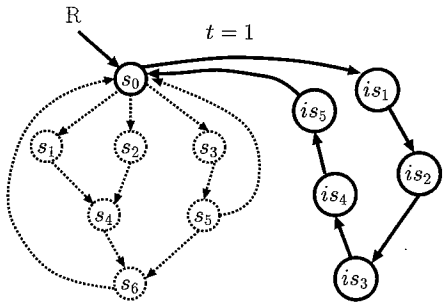
1. 論理合成：与えられた FSM（図 1）を論理合成し，順序回路（図 2）を得る．ここで，状態割当の結果を知ることができるものとする．

2. 組合せテスト生成：合成された順序回路（図 2）の組合せテスト生成モデル（図 3）に対して組合せ回路用のテスト生成アルゴリズムを用いてテスト生成する．

3. 無効テスト状態の抽出：2. で得られたテストパターンを，それらの PPI の値によって，有効テストパターン，無効テストパターンに分類する．具体的には，1. で得られた状態割当の結果より，テストパターンの PPI の値が FSM の有効状態に対応する値であれば，それを有効テストパターンとし，対応しない値であれば無効テストパターンとする．有効テストパターンに現れる状態を有効テスト状態とし，無効テストパターンに現れる状態を無効テスト状態とする．有効テスト状態については，**5.1** の有効テストパターンの印加の際に用いる．

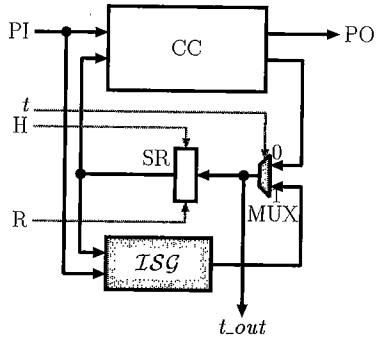
4. 無効テスト状態生成論理を付加：与えられた FSM（図 1）に対して，リセット状態からすべての無効テスト状態を通る経路ができるように遷移を追加する（図 4 参照）．ここで，状態レジスタの FF 数を n とすると，状態レジスタの表現できる状態数は 2^n であるが，無効テスト状態数は，たかだかテストパターン数である．無効テスト状態はテスト実行において，無効テストパターンの印加に利用するだけなので，無効テスト状態への遷移は，状態遷移モード切換信号 $t=1$ のとき，組合せ論理（CC）に無関係に遷移させる．次に，FSM に追加した無効テスト状態に関する遷移の部分だけを論理合成して，無効テスト状態生成論理（$\mathcal{ISG}$, Invalid State Generator）を求め，1. で既に合成されている回路（図 2）に図 5 のようにマルチプレクサ（MUX）を介して追加する．

5. ホールド機能の付加：状態レジスタ SR にホール



s_i：有効状態　　is_j：無効テスト状態
t：状態遷移モード切換信号

図 4 無効テスト状態への遷移を付加した FSM
Fig. 4 An FSM which is augmented by adding transitions to invalid test states.



H：ホールド信号　　$\mathcal{ISG}$：無効テスト状態生成論理
t_out：状態出力信号　t：状態遷移モード切換信号

図 5 無効テスト状態生成論理を付加したコントローラ
Fig. 5 A controller which is augmented by adding invalid state generator.

ド機能（ホールド信号 H）をもたせる．ホールド機能は，テスト系列長を短くするために用いる．詳細は 5. で述べる．

FSM が与えられない場合にも，論理レベル回路が与えられれば FSM 抽出プログラム（例えば，SIS [4] の状態遷移図抽出プログラム）を用いることによって，本手法を適用することができる．

4.2 遅延オーバヘッド

テスト容易化設計手続き 4. と 5. において，状態レジスタの直前に MUX を挿入し，また，状態レジスタにホールド機能をもたせている．そのために，コントローラの通常動作時にも遅延が生じる．しかし，この遅延オーバヘッドは完全スキャン設計法の場合と同等である．また，この遅延は，コントローラ設計の最初から評価でき，この遅延をあらかじめ考慮してコントローラを設計及び合成できる．

一方，$\mathcal{ISG}$ は，コントローラの通常動作には影響を与えない．$\mathcal{ISG}$ を合成する際に，$\mathcal{ISG}$ の遅延が組合せ論理部の遅延よりも小さくなるようにすれば，$\mathcal{ISG}$ を付加する前の回路の実動作速度でテスト実行可能となる．ベンチマークを用いた実験では，$\mathcal{ISG}$ を付加する必要のあるすべての回路に対して，$\mathcal{ISG}$ の遅延は組合せ論理部の遅延よりも小さいことを示している（**6.4** 参照）．

4.3 面積オーバヘッド

本手法において，$\mathcal{ISG}$，MUX，状態レジスタのホールド機能のための論理が面積オーバヘッドとなる．MUX 及びホールド機能による面積オーバヘッドは完全スキャン設計法と同じである．本手法では，テスト容易化設計の手続き 4. において，無効テスト状態への遷移を追加する．本手法において，テスト生成して得られたテストパターン集合の完全故障検出効率を保証するためには，すべての無効テスト状態に遷移できればよく，無効テスト状態への遷移順序は，故障検出効率に影響を与えない．しかし，その遷移順は，$\mathcal{ISG}$ 面積に影響を与えると考えられる．したがって，無効テスト状態の適切な遷移順を考えることにより，テスト容易化のための面積オーバヘッドを縮小することができる．

Patel ら [3] の手法では，外部入力から直接状態レジスタの一部の FF に値を設定できるようにしている．本手法においても，直接状態レジスタの一部の FF に値を設定できるように回路を構成することもできる．例えば，図 6 のように回路を構成すれば，状態レジスタの FF のうち，外部入力から直接値を設定できない FF についてのみ無効テスト状態の値を $\mathcal{ISG}$ で生成すればよい．これにより，$\mathcal{ISG}$ で生成しなければならない値が減るので，$\mathcal{ISG}$ の面積を小さくすることができる．このようにしても，テスト容易化のための MUX 数は変わらない．

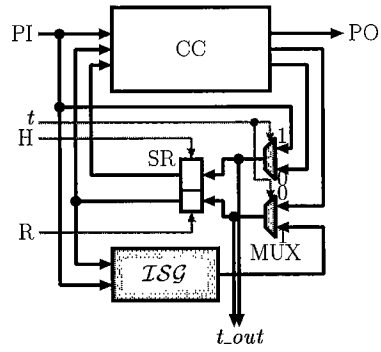


図 6　無効テスト状態生成に外部入力を利用する場合
Fig. 6　An example of an invalid test state generation using primary inputs.

4.4 観 測 点

図 5，図 6 のように，*t_out* を観測点としてテスト用の外部出力としている．しかし，*t_out* のビット幅分の外部出力ピンを確保できない場合も考えられる．VLSI は一般にコントローラ（制御部）とデータパス（演算部）からなる．コントローラのテストの際には，データパスの機能を用いないので，データパスの外部出力が十分にあれば，MUX を介して *t_out* をデータパスの外部出力から出力できる．また，データパスの外部出力が十分に存在しない場合には，XOR ゲート木を用いて *t_out* のパリティのみを出力する方法も考えられる．これにより，観測のための外部出力ピン数を減らすことができる．この方法を用いた場合，故障の影響が *t_out* の奇数本の信号線に伝搬された場合に検出でき，偶数本に伝搬された場合には検出できない．しかし，一つの故障に対し，その故障を検出するテストパターンは，求めたテストパターン集合に一般に複数存在する．それらのテストパターンに故障の影響が奇数本の信号線に伝搬されるようなテストパターンが一つでも存在すればこの故障は検出できることになり，ほとんどの故障が XOR 木を用いても検出できると期待できる．文献 [5] では，ベンチマーク回路を用いた実験結果より，ほとんどの場合で XOR 木を用いても検出故障数が減らないことを示している．

以下では，t_outは何らかの方法で観測できるものとする．

5. テスト系列生成法及びテスト実行法

本章では，テストパターンの印加方法及び出力応答の観測方法を提案する．

5.1 有効テストパターンの印加及び観測

各有効テストパターンは，回路の機能（通常の状態遷移）を用いて印加する．ある有効テストパターンの印加について考える．はじめに，リセット状態から，その有効テストパターンの PPI の値に対応する有効テスト状態 s_i への状態遷移系列を求める．状態遷移モード切換信号 $t=0$ とし，求めた状態遷移系列を印加し，状態レジスタに s_i を設定する（図 7 参照）．ここで，組合せ論理部（CC）に故障が存在する場合，必要な有効テスト状態を状態レジスタに設定できない可能性がある．しかし，この状態遷移系列を印加する際に，状態レジスタに取り込まれる値を，状態レジスタの直前の状態出力信号 t_outから観測することで，故障を検出することができる．したがって，有効テスト状態を設定するための入力系列は，CC のテスト系列をかねている．次に，外部入力からそのテストパターンの PI の値を印加する（図 7 参照）．このようにして，各有効テストパターンは組合せ論理部（CC）に印加できる．

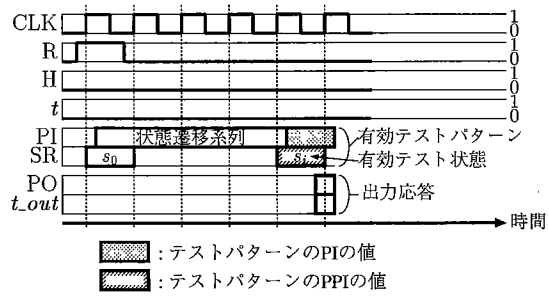


図 7 有効テストパターンの印加
Fig. 7 Applying valid test patterns.

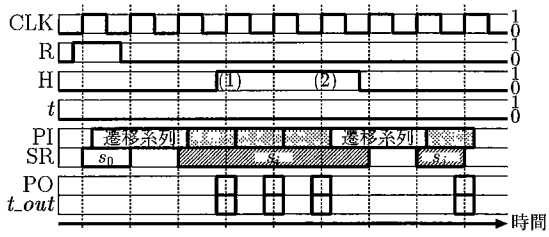


図 8 状態レジスタのホールド機能を用いた場合
Fig. 8 Applying valid test patterns using hold mode of state register.

[テスト系列長の短縮]

すべての有効テストパターンの印加に要するテスト系列長の短縮について考える．はじめに，同じ有効テスト状態 s_i を含む有効テストパターンが複数存在する場合について考える．この場合，**4.1** のテスト容易化の手続き 5. で付加した状態レジスタのホールド機能を用いて，状態レジスタに s_i を設定した後，その値をホールドし，s_i を含む複数の有効テストパターンの PI の値を次々と印加（図 8 のホールドのタイミング（1）参照）することができる．これにより，各有効テストパターンに対してリセット状態からの状態遷移を行うのではなく，各有効テスト状態に対して 1 回ずつリセット状態から遷移すればよくなり，有効テストパターンの印加に要するテスト系列長を小さくできる．

次に，有効テスト状態 s_i から別の有効テスト状態 s_j へ，通常の状態遷移によって，リセットを用いるよりも短い遷移系列で遷移できる場合について考える．この場合，s_i を含むすべての有効テストパターンを印加し終えた次の時刻まで s_i をホールドしておき，続けてこの遷移系列を印加すれば，次の有効テスト状態 s_j にリセットを用いるよりも短い遷移系列で遷移することができる（図 8 のホールドのタイミング（2）参照）．これにより，各有効テスト状態へリセット状態から遷移するよりも，すべての有効テストパターンの印加に要するテスト系列長を小さくできる．

したがって，リセット状態から始まる，すべての有効テスト状態を含む状態遷移系列（有効テスト状態走査系列と呼ぶ）を求め，これを用いて状態レジスタに有効テスト状態を設定していけばよい．ここで，FSM の任意の状態において，リセット信号を印加することでリセット状態に遷移できるので，必ず有効テスト状態走査系列は存在する．このとき，すべての有効テストパターンの印加に要するテスト系列長は，有効テスト状態走査系列長を L_{vt}，有効テストパターン数を N_{vp} とすると，$L_{vt}+N_{vp}$ である．求められた有効テストパターン集合に対し，すべての有効テストパターンの印加に要するテスト系列長を最短にするには，最短の有効テスト状態走査系列を求めればよい．

[有効テストパターンの印加]

すべての有効テストパターンの印加に要するテスト系列が最短となる有効テストパターンの印加の手順を

以下に示す．

1. 有効テストパターン集合の分割：**4.1** のテスト容易化の手続き 3. で得られた有効テストパターン集合に対して，有効テストパターンの PPI の値（有効テスト状態）によって分割する．

2. 最短の有効テスト状態走査系列の生成：FSM に対して，すべての有効テスト状態へ少なくとも 1 回遷移する最短の状態遷移系列を求める．これは，すべての有効テスト状態を頂点とする重み付き完全有向グラフにおいて，巡回セールスマン問題（TSP, Traveling Salesman Problem) [6] を解くことと同じである．ここで，頂点間の辺の重みは，対応する有向テスト状態間の最短遷移系列長とする．この最短遷移系列長は，FSM の状態を頂点とし，FSM の遷移を辺とした有向グラフにおいて，有向テスト状態に対応する頂点間の最短経路問題を解くことによって得られる．TSP は，NP 困難問題であるが，これまでに，TSP を解く多くのヒューリスティックアルゴリズムが提案されているので，それらを用いて最短有効テスト状態走査系列を求めることができる．

3. 有効テストパターンの印加：各有効テスト状態は，2. で得られた最短の有効テスト状態走査系列を用いて 1 回ずつ状態レジスタに設定する．有効テスト状態走査系列を印加し，状態レジスタにある有効テスト状態 s_i が設定されたら，状態レジスタの値をホールドし（図 8 のホールドのタイミング（1）参照），1. で得られた s_i を含むすべての有効テストパターンの PI に対応する値を PI から印加する．そして，状態レジスタのホールドを解除し（図 8 のホールドのタイミング（2）参照），引き続き有効テスト状態走査系列を印加し，次に現れる有効テスト状態 s_j まで遷移する．この操作を繰り返すことによって，すべての有効テストパターンを印加することができる．

[出力応答の観測]

有効テストパターンに対する出力応答は，有効テストパターンを印加した直後に外部出力と状態出力 *t_out* で観測できる．ここで，*t_out* の観測は，各有効テストパターンの印加の直後だけでなく，有効テスト状態走査系列の中も行う．これは，本節のはじめに述べたように，状態レジスタに値を設定するための状態遷移系列は CC のテスト系列を兼ねているためである．

5.2 無効テストパターンの印加及び観測

各無効テストパターンは，無効テスト状態生成論理（$\mathcal{ISG}$，図 4 参照）を用いて，有効テストパターンの場合と同様の方法で印加する．以下では，簡単のため，$\mathcal{ISG}$ を用いた遷移は外部入力に無関係に行うものとする．

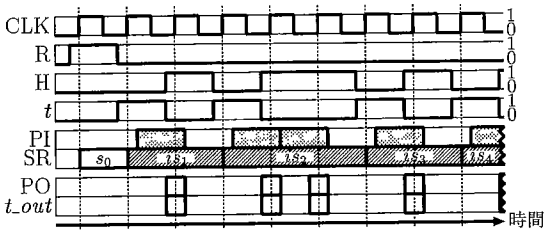


図 9 無効テストパターンの印加
Fig. 9 Applying invalid test patterns.

4.1 で述べたように，リセット状態からすべての無効テスト状態を通る経路ができるように遷移を追加したので，リセット状態 s_0 から無効テスト状態 is_1 へは状態遷移モード切換信号 $t=1$ としてクロックを一つ進めることによって遷移できる（図 9 の状態遷移モード切換信号 t のタイミング参照）．is_1 に遷移した後，状態レジスタの値をホールドして，is_1 を含む無効テストパターンの PI の値を PI から次々と印加できる（図 9 のホールドのタイミング参照）．次の無効テスト状態 is_2 へは，ホールドを解除してクロックを進めることで遷移できる．この操作を繰り返すことによって，すべての無効テストパターンを印加することができる．また，無効テストパターンに対する出力応答は，無効テストパターンを印加した直後に，外部出力と状態出力 *t_out* で観測できる．ここで，状態遷移モード切換信号 $t=0$ とすることで，出力応答を *t_out* に伝搬できる（図 9 の状態遷移モード切換信号 t のタイミング参照）．

$\mathcal{ISG}$ は，リセット状態から無効テスト状態のみを順に生成するので，リセット状態からの，すべての無効テスト状態が現れる最短状態遷移系列長は，最初のリセット信号の入力を含めると，無効テスト状態数 +1 となる．したがって，すべての無効テストパターンの印加に要するテスト系列長は，無効テスト状態数を N_{is}，無効テストパターン数を N_{ip} とすると，$N_{is}+N_{ip}+1$ であり，最悪の場合でも $N_{ip}\times 2+1$ である．

5.3 テスト容易化のための付加回路のテスト

$\mathcal{ISG}$ は通常動作では用いないので，テストのための機能（無効テスト状態の生成）が実現されているかをテストするだけでよい．具体的には，状態遷移モード切換信号 $t=1$ として，リセット状態からテストパ

ターンに含まれる無効テスト状態を，決められた順に出力できるかどうかを調べればよい．これは，無効テストパターンを用いたテスト実行の際に，状態出力信号 t_{out} から観測することによって行える．したがって，無効テストパターンを用いたテスト実行時に，同時に $\mathcal{ISG}$ のテストを行える．

また，状態レジスタの直前に付加する MUX のテストについては，FSM を論理合成して得られた順序回路の組合せテスト生成モデルを作る際に MUX を付加し，MUX を含めた回路に対してテスト生成することによって，MUX の故障を検出することができる．

6. 従来法との比較

本章では，完全スキャン設計法 [1] 及び Patel らの非スキャンテスト容易化設計法 [3] と本手法を，テスト生成時間，故障検出効率，テスト実行時間及び面積オーバヘッドの点について比較し，比較実験結果を示す．

6.1 完全スキャン設計法

完全スキャン設計法では，テスト生成モデルが組合せ回路となり，完全故障検出効率をもつテストパターン集合を高速に求めることができる．しかし，完全スキャン設計法では，スキャンを用いてテストパターンを印加するので，生成されたテストパターンの印加及び出力応答の観測に要するテスト実行時間（クロック数）は，テストパターン数を N_{pat}，FF 数を N_{FF} とすると，$N_{pat} \times (N_{FF}+1)+N_{FF}$ となり，状態レジスタの FF 数が大きくなると長くなる．また，面積オーバヘッドは，状態レジスタの FF をスキャン FF に置き換えるので，一つの MUX の実現に要する面積を A_{MUX} とすると，$N_{FF} \times A_{MUX}$ となる．

外部ピンオーバヘッドは，スキャンイン，スキャンアウト，MUX 切換信号の 3 本である．

完全スキャン設計法では，スキャンシフト動作には，システムクロックよりも遅いスキャン用のクロックを用いるため，実動作速度でテスト実行できない．

6.2 Patel らの非スキャンテスト容易化設計法

Patel らの手法は，与えられた順序回路の FF を，外部入力数分だけ外部入力から直接制御できるように MUX を付加して設計変更する方法である．外部入力から直接制御する FF の選択は，自己ループ以外のループを切ることができるように選び，更に可制御性を計算して求めている．観測点については，回路中の信号線の可観測性を計算し，観測が困難な信号線を XOR ゲート木を用いることにより直接観測できるようにしている．

Patel らの手法では，外部入力数が状態レジスタの FF 数よりも大きい場合は，テスト生成モデルが組合せ回路となり，完全故障検出効率をもつテストパターン集合を高速に求めることができる．面積オーバヘッドは，状態レジスタのすべての FF を外部入力から制御するので，$N_{FF} \times A_{MUX}$ となる．Patel らの手法では，各テストパターンを印加するのに外部入力を用いて状態レジスタの値を設定するので，各パターンについて 2 システムクロックサイクルを要する．そのため，テスト生成して得られたテストパターンの印加及び出力応答の観測に要するテスト実行時間（クロック数）は，$N_{pat} \times 2+1$ である．

外部入力数が状態レジスタの FF 数よりも小さい場合は，テスト生成モデルが順序回路となり，完全故障検出効率をもつテスト系列を生成することは，一般に困難である．また，生成されたテスト系列には，外部入力から直接制御できない FF を初期化する系列が含まれるため，生成されたテスト系列長が長くなるという問題が生じる．面積オーバヘッドは，状態レジスタの外部入力数だけの FF を外部入力から制御するので，外部入力数を N_{PI} とすると $N_{PI} \times A_{MUX}$ となる．この場合のテスト生成して得られたテスト系列の印加及び出力応答の観測に要するテスト実行時間（クロック数）は，テスト生成して得られたテスト系列長を L_{seq} とすると，$L_{seq} \times 2+1$ となる．

観測のための XOR 木の面積オーバヘッドは，観測点数を N_{OP}，XOR 面積を A_{XOR} とすると，$(N_{OP}-1) \times A_{XOR}$ となる．また，外部ピンオーバヘッドは，状態レジスタのホールド/ロード信号，MUX 切換信号，及び観測のための XOR 木の出力の 3 本である．

Patel らの手法では，実動作速度でテストパターンの印加ができるので，実動作速度でテスト実行可能である．

6.3 本 手 法

本手法は，テスト生成モデルが組合せ回路となり，完全故障検出効率をもつテストパターン集合を高速に求めることができる．面積オーバヘッドは，無効テスト状態が存在する場合は，$\mathcal{ISG}$ の面積を $A_{\mathcal{ISG}}$ とすると，$N_{FF} \times A_{MUX}+A_{\mathcal{ISG}}$ である．$\mathcal{ISG}$ は，たかだかテストパターン数の無効テスト状態を生成するための組合せ論理である．6.4 では，この面積を実験によって評価する．また，無効テスト状態が存在しない

場合は，面積オーバヘッドはない．

テスト生成して得られたテストパターンの印加及び出力応答の観測に要するテスト実行時間（クロック数）は，無効テスト状態が存在する場合は，有効状態走査系列長を L_{vt}，無効テスト状態数を N_{is} とすると，$L_{vt}+N_{is}+N_{pat}+2$ となる．また，無効テスト状態が存在しない場合は，$L_{vt}+N_{pat}+1$ となる．

外部ピンオーバヘッドは，無効テスト状態が存在する場合は，ホールド/ロード信号，MUX 切換信号，状態出力 *Lout*（N_{FF}）の $2+N_{FF}$ 本となる．また，無効テスト状態が存在しない場合は，MUX 切換信号が不要なので，$1+N_{FF}$ となる．外部ピンオーバヘッドを小さくするために，*Lout* をデータパスの外部出力と共有した場合は，$N_{FF}\times A_{MUX}$ だけ面積オーバヘッドが増え，外部ピンオーバヘッドは，*Lout* の代わりに，新たに付加した MUX の切換信号となり，無効テスト状態が存在する場合は 3 本，存在しない場合は 2 本となる．

本手法でも，実動作速度でテストパターンの印加ができるので，実動作速度でテスト実行可能である．

6.4 実験結果

MCNC'91 の FSM ベンチマーク [7] を用いた比較実験結果を示す．用いた FSM ベンチマークの特性，FSM ベンチマークを論理合成した結果を表 1 に示す．これらは，FSM ベンチマーク（55 個）のうち，論理合成して得られた回路の組合せ回路部分に対してテスト生成したとき，無効テスト状態が存在したもの（35 個）である．用いた計算機は S-4/20 model 712 (Fujitsu) である．また，合成ツールは AutoLogic II (MentorGraphics) で，AutoLogic II に附属のサンプルライブラリーを用いて合成した．ここで，回路面積の単位はゲート数で，用いたライブラリーセル面積をゲート数に換算したものである．用いたテスト生成ツールは，TestGen (SunRise) である．

表 2 に各手法を用いた場合のテスト生成結果を示す．Patel らの手法では，ループの切断，可制御性及び可観測性を計算して外部入力から直接制御する FF 及び観測点を求めているが，ここでは，観測点を完全スキャン設計及び本手法と同様にすべての FF とした．また，外部入力から制御する FF については，すべての組合せに対してテスト生成を行った．表 2 では，最も故障検出効率が高いものの中で，最もテスト生成時間の短いものを示している．Patel らの手法を用いたとしても，これよりも故障検出効率，テスト生成時間の点で良い結果は得られないと考えられる．本論文で提案する手法は，組合せテスト生成モデルに対して組合せ回路用のテスト生成アルゴリズムを用いているので，テスト生成時間は完全スキャン設計法と同じである．また，回路名の先頭に“＊”のあるものは，外部入力数が FF 数以上の回路で，Patel らの手法でも組合せ回路用のテスト生成アルゴリズムを用いている．完全スキャン設計法と本手法は，すべての回路で完全故障検出効率を保証し，Patel らの手法では，回路名の先頭に“＊”のあるもののみ完全故障検出効率を保証している．本実験では，Patel らの手法の故障検出効率は，すべての回路に対して 100%という結果が得られている．完全スキャン設計法及び Patel らの手法のテスト実行時間は，**6.1** 及び **6.2** で述べた計算式で求めた．本手法のテスト実行時間は，TSP を解く簡

表 1　FSM ベンチマーク特性・論理合成後の面積
Table 1　FSM benchmark characteristics and areas after logic synthesis.

回路名	状態数	入力数	出力数	FF 数	回路面積
bbara	10	4	2	4	410.30
bbsse	16	7	7	4	781.20
bbtas	6	2	2	3	87.60
beecount	7	3	4	3	331.50
dk14	7	3	5	3	295.10
dk16	27	2	3	5	510.40
dk27	7	1	2	3	92.00
dk512	15	1	3	4	220.80
ex1	20	9	19	5	2740.50
ex2	19	2	2	5	416.90
ex3	10	2	2	4	192.80
ex4	14	6	9	4	479.20
ex5	9	2	2	4	183.70
ex7	10	2	2	4	189.60
keyb	19	7	2	5	1835.40
lion9	9	2	1	4	322.10
opus	10	5	6	4	567.60
planet1	48	7	19	6	2791.10
planet	48	7	19	6	2791.10
pma	24	8	8	5	1068.60
s1488	48	8	19	6	6190.20
s1494	48	8	19	6	6242.80
s1	20	8	6	5	2396.00
s208	18	11	2	5	2361.30
s27	6	4	1	3	416.30
s298	218	3	6	8	8720.80
s386	13	7	7	4	1241.10
s420	18	19	2	5	2217.50
s510	47	19	7	6	1184.20
s820	25	18	19	5	4411.00
s832	25	18	19	5	4543.70
sse	16	7	7	4	781.20
styr	30	9	10	5	2748.90
tma	20	7	6	5	802.70
train11	11	2	1	4	364.50

表2 各手法のテスト生成結果の比較
Table 2 Test generation results of each method.

回路名	テスト生成時間（秒）			テスト実行時間（クロック数）		
	スキャン	Patel	本手法	スキャン	Patel	本手法
*bbara	0.99	0.99	0.99	334	131	87
*bbsse	1.79	1.79	1.79	399	159	101
bbtas	0.16	0.22	0.16	71	57	27
*beecount	0.67	0.67	0.67	199	95	60
*dk14	0.46	0.46	0.46	231	113	68
dk16	1.08	4.94	1.08	623	783	153
dk27	0.21	0.28	0.21	71	115	31
dk512	0.34	0.81	0.34	199	233	72
*ex1	14.82	14.82	14.82	1613	527	312
ex2	0.75	2.60	0.75	485	631	123
ex3	0.48	0.87	0.48	244	303	71
*ex4	0.95	0.95	0.95	304	119	78
ex5	0.44	0.78	0.44	244	277	71
ex7	0.35	0.52	0.35	194	207	60
*keyb	16.79	16.79	16.79	1409	491	270
lion9	0.45	0.89	0.45	239	319	65
*opus	1.11	1.11	1.11	394	151	106
*planet1	11.56	11.56	11.56	1574	453	405
*planet	12.72	12.72	12.72	1574	453	405
*pma	4.17	4.17	4.17	947	315	200
*s1488	72.07	72.07	72.07	3149	871	629
*s1494	78.26	78.26	78.26	2981	859	633
*s1	15.11	15.11	15.11	1241	433	262
*s208	31.51	31.51	31.51	1607	497	301
*s27	0.88	0.88	0.88	199	97	61
s298	254.01	2853.94	254.01	9890	19999	2446
*s386	3.59	3.59	3.59	514	207	123
*s420	22.48	22.48	22.48	1439	465	273
*s510	3.56	3.56	3.56	916	269	194
*s820	48.50	48.50	48.50	2225	727	442
*s832	50.91	50.91	50.91	2297	787	450
*sse	1.73	1.73	1.73	399	159	101
*styr	16.37	16.37	16.37	1367	475	293
*tma	2.41	2.41	2.41	653	229	156
train11	0.62	1.27	0.62	274	359	77

単なアルゴリズムを計算機上に実装して有効テスト状態走査系列を求め，**6.3** で述べた計算式により算出したものである．テスト系列長はすべての回路で本手法が他の二つの手法と比べて短いことがわかる．特に，s298 では，本手法が完全スキャン設計法の 1/4，Patel らの手法の 1/8 という結果が得られている．TSP を解く効率の良いアルゴリズムを用いれば，更に本手法のテスト系列長を小さくできると考えられる．

表 3 に $\mathcal{ISG}$ 面積と $\mathcal{ISG}$ 遅延を示す．$\mathcal{ISG}$ は本手法にのみ存在する．ここでは，図 6 のように外部入力から状態レジスタの一部の FF の値を直接制御できるようにした場合の $\mathcal{ISG}$ の面積オーバヘッドを示している．また，$\mathcal{ISG}$ は，外部入力に無関係に無効テスト状態へ遷移するように合成したものである．表 2 の回路名に“*”のある回路は，$\mathcal{ISG}$ が不要なので，ここでは示していない．表中の“無効テスト状態”の欄の“状態数”は無効テスト状態数，“ビット幅”は $\mathcal{ISG}$ を用いて制御する（外部入力から直接制御しない）状態レジスタ中の FF 数，“生成数”は $\mathcal{ISG}$ で生成する値の数を示している．ここで，外部入力から直接制御する FF は適当に選んでいる．また，$\mathcal{ISG}$ の無効テスト状態の出力順も適当に決めている．$\mathcal{ISG}$ の面積の単位はゲート数で，括弧内は表 1 で示した回路面積との比である．$\mathcal{ISG}$ が必要なものは全体の 31.4%で，それらの $\mathcal{ISG}$ の面積オーバヘッドは，最小のもので 0.37%，最大でも 8.32%，平均では 3.5%とわずかである．外部入力から制御する FF を適切に選ぶことにより，$\mathcal{ISG}$ で生成する値の数を小さくすることができ，更に $\mathcal{ISG}$ 面積オーバヘッドを縮小することができると考えられる．また，**4.3** で述べたように，$\mathcal{ISG}$ で生成する値の出力順について考慮すれば，更に $\mathcal{ISG}$ の面積オーバヘッドを縮小することができると考えられる．

表3 無効テスト状態数・$\mathcal{ISG}$ 面積・$\mathcal{ISG}$ 遅延
Table 3 The number of invalid test states, $\mathcal{ISG}$ area and $\mathcal{ISG}$ delay.

回路名	無効テスト状態			$\mathcal{ISG}$ 面積・（ゲート数（比））	遅延（段数）	
	状態数	ビット幅	生成数		CC	$\mathcal{ISG}$
bbtas	3	1	2	1.20 (1.36%)	9	3
dk16	6	3	6	39.30 (6.13%)	13	8
dk27	2	2	2	1.20 (1.30%)	8	3
dk512	3	3	3	7.00 (6.06%)	13	5
ex2	23	3	8	34.70 (8.32%)	13	9
ex3	7	2	4	11.70 (6.06%)	12	7
ex5	8	2	4	12.90 (7.02%)	12	7
ex7	10	2	4	14.10 (6.80%)	11	7
lion9	8	2	4	1.20 (0.37%)	14	7
s298	33	5	28	255.60 (2.16%)	34	19
train11	6	2	4	11.70 (3.20%)	14	7

表 3 の“遅延（段数）”の欄中の“CC”及び“$\mathcal{ISG}$”は，FSM から合成された順序回路の組合せ回路部分及び $\mathcal{ISG}$ それぞれの最大ライブラリーセル段数を示している．これは，組合せ回路部分及び $\mathcal{ISG}$ の遅延と考えることができる．したがって，実験で用いた $\mathcal{ISG}$ の必要なすべての回路に対して，$\mathcal{ISG}$ の遅延は組合せ回路部分の遅延よりも小さいことがわかる．

各手法の面積オーバヘッドを表 4 に示す．面積オーバヘッドの単位はゲート数で，それぞれの手法に対して，**6.1**，**6.2** 及び **6.3** で示した計算式で求めた．ここで，MUX 面積，XOR 面積は対応するライブラリーセル面積をゲート数に換算したもので，それぞれ 3.40 ゲート，4.30 ゲートとしている．表中の Patel の

表4　各手法の面積オーバヘッドの比較
Table 4　Area overheads of each method.

回路名	スキャン	Patel (iMUX + oXOR)	本手法 (iMUX + $\mathcal{ISG}$ + oMUX)
bbara	13.60	26.50 (13.6+ 12.9)	27.20 (13.6+ 0.0+ 13.6)
bbsse	13.60	26.50 (13.6+ 12.9)	27.20 (13.6+ 0.0+ 13.6)
bbtas	10.20	15.40 (6.8+ 8.6)	21.60 (10.2+ 1.2+ 10.2)
beecount	10.20	18.80 (10.2+ 8.6)	20.40 (10.2+ 0.0+ 10.2)
dk14	10.20	18.80 (10.2+ 8.6)	20.40 (10.2+ 0.0+ 10.2)
dk16	17.00	24.00 (6.8+ 17.2)	73.30 (17.0+ 39.3+ 17.0)
dk27	10.20	12.00 (3.4+ 8.6)	21.60 (10.2+ 1.2+ 10.2)
dk512	13.60	16.30 (3.4+ 12.9)	34.20 (13.6+ 7.0+ 13.6)
ex1	17.00	34.20 (17.0+ 17.2)	34.00 (17.0+ 0.0+ 17.0)
ex2	17.00	24.00 (6.8+ 17.2)	68.70 (17.0+ 34.7+ 17.0)
ex3	13.60	19.70 (6.8+ 12.9)	38.90 (13.6+ 11.7+ 13.6)
ex4	13.60	26.50 (13.6+ 12.9)	27.20 (13.6+ 0.0+ 13.6)
ex5	13.60	19.70 (6.8+ 12.9)	40.10 (13.6+ 12.9+ 13.6)
ex7	13.60	19.70 (6.8+ 12.9)	41.30 (13.6+ 14.1+ 13.6)
keyb	17.00	34.20 (17.0+ 17.2)	34.00 (17.0+ 0.0+ 17.0)
lion9	13.60	19.70 (6.8+ 12.9)	28.40 (13.6+ 1.2+ 13.6)
opus	13.60	26.50 (13.6+ 12.9)	27.20 (13.6+ 0.0+ 13.6)
planet1	20.40	41.90 (20.4+ 21.5)	40.80 (20.4+ 0.0+ 20.4)
planet	20.40	41.90 (20.4+ 21.5)	40.80 (20.4+ 0.0+ 20.4)
pma	17.00	34.20 (17.0+ 17.2)	34.00 (17.0+ 0.0+ 17.0)
s1488	20.40	41.90 (20.4+ 21.5)	40.80 (20.4+ 0.0+ 20.4)
s1494	20.40	41.90 (20.4+ 21.5)	40.80 (20.4+ 0.0+ 20.4)
s1	17.00	34.20 (17.0+ 17.2)	34.00 (17.0+ 0.0+ 17.0)
s208	17.00	34.20 (17.0+ 17.2)	34.00 (17.0+ 0.0+ 17.0)
s27	10.20	18.80 (10.2+ 8.6)	20.40 (10.2+ 0.0+ 10.2)
s298	27.20	40.30 (10.2+ 30.1)	310.00 (27.2+255.6+ 27.2)
s386	13.60	26.50 (13.6+ 12.9)	27.20 (13.6+ 0.0+ 13.6)
s420	17.00	34.20 (17.0+ 17.2)	34.00 (17.0+ 0.0+ 17.0)
s510	20.40	41.90 (20.4+ 21.5)	40.80 (20.4+ 0.0+ 20.4)
s820	17.00	34.20 (17.0+ 17.2)	34.00 (17.0+ 0.0+ 17.0)
s832	17.00	34.20 (17.0+ 17.2)	34.00 (17.0+ 0.0+ 17.0)
sse	13.60	26.50 (13.6+ 12.9)	27.20 (13.6+ 0.0+ 13.6)
styr	17.00	34.20 (17.0+ 17.2)	34.00 (17.0+ 0.0+ 17.0)
tma	17.00	34.20 (17.0+ 17.2)	34.00 (17.0+ 0.0+ 17.0)
train11	13.60	19.70 (6.8+ 12.9)	38.90 (13.6+ 11.7+ 13.6)

欄の括弧内は面積オーバヘッドの内訳を示しており，“iMUX”，“oXOR”は，それぞれ状態レジスタの直前に付加するMUXの面積オーバヘッド，観測のためのXOR木の面積オーバヘッドを示している．本手法の欄では，状態出力 t_{out} をデータパスの外部出力と共有した場合の面積オーバヘッドを示している．本手法の欄の括弧内もPatelの欄と同様に面積オーバヘッドの内訳を示している．“iMUX”，“$\mathcal{ISG}$”，“oMUX”は，それぞれ状態レジスタの直前に付加するMUXの面積オーバヘッド，$\mathcal{ISG}$ 面積オーバヘッド，観測のためのMUXの面積オーバヘッドを示している．状態出力 t_{out} を外部ピンで観測できる場合は，“oMUX”は不要となる．

また，ここで示していないFSMベンチマーク（20個）については，無効テスト状態が存在しないので，“iMUX”及び“$\mathcal{ISG}$”は不要である．この場合，“oMUX”が必要な場合でも完全スキャン設計法と同じ面積オーバヘッドである．

このように，実験結果より，本手法は $\mathcal{ISG}$ を付加する必要のある回路でも平均3.5%の $\mathcal{ISG}$ 面積オーバヘッドで，完全故障検出効率をもつテストパターン集合を高速に求めることができることを示した．更に，すべての回路において．得られたテストパターン集合の印加に要するテスト実行時間は，従来法と比べて小さいことを示した．

7. むすび

完全スキャン設計法では完全故障検出効率をもつテスト系列を高速に求めることができるが，実動作速度でテスト実行できないという問題があった．更に，スキャンシフト動作を必要とするため，テスト実行時間が長いという問題もあった．一方，従来の非スキャンテスト容易化設計法では，実動作速度でテスト実行可能であるが，完全故障検出効率をもつテスト系列を求めるのは一般に困難であるという問題があった．本論文では，完全故障検出効率をもつテスト系列を高速に求めることができ，かつ，実動作速度でテスト実行可能なテスト容易化設計法及びテスト実行法を提案した．従来法との比較実験では，すべてのFSMベンチマークに対して，従来法よりもテスト実行時間が短いことを示した．また，$\mathcal{ISG}$ を付加する場合でも，$\mathcal{ISG}$ 面積オーバヘッドは平均3.5%とわずかであることを示した．今後の課題としては，更に面積オーバヘッドを縮小するために，$\mathcal{ISG}$ 面積最小化が挙げられる．

謝辞　本研究に関し，多くの貴重な意見をいただいた本学の井上智生助手，井上美智子助手はじめ情報論理学講座の諸氏に感謝する．本研究は一部，（株）半導体理工学研究センター（STARC）との共同研究による．

文　献

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

[2] P.C. Maxwell, R.C. Aitken, V. Johansen, and I. Chiang, “The effect of different test sets on quality level prediction: When is 80% better than 90%?,” in Proc. of International Test Conference, pp.358–364, 1991.

[3] V. Chickermane, E.M. Rudnick, P. Banerjee, and J.H. Patel, “Non-scan design-for-testability techniques for sequential circuits,” in Proc. of 30th ACM/IEEE Design Automation Conference, pp.236–241, 1993.

[4] E.M. Sentovich, K.J. Singh, L. Lavagno, C. Moon,

R. Murgai, A. Saldanha, H. Savoj, P.R. Stephan, R.K. Brayton, A. Sangiovanni-vincentelli, "SIS: A system for sequential circuit synthesis," Technical Report UCB/ERL-M92/41, University of California, Berkeley, 1992.

[5] H. Fujiwara and A. Yamamoto, "Parity-scan design to reduce the cost of test application," IEEE Transaction on Computer-Aided-Design, vol.12, no.10, pp.1604–1611, 1993.

[6] M.R. Garey and D.S. Johnson, "Computer and Intractability," W.H. Freeman and Company, 1979.

[7] S. Yang, "Logic synthesis and optimization benchmarks user guide," Technical Report 1991-IWLS-UG-Saeyang, Microelectronics Center of North Carolina, 1991.

[8] 和田弘樹，増澤利光，K.K. Saluja，藤原秀雄，"完全故障検出効率を保証するデータパスの非スキャンテスト容易化設計法，" 信学技報，VLD97-79～99, 1997.

(平成 10 年 3 月 18 日受付，6 月 16 日再受付)

大竹　哲史　(学生員)

平 7 電通大・電通・情報卒．平 9 奈良先端大・情報科学・博士前期課程了．現在同博士後期課程在学中．日本学術振興会特別研究員．テスト生成アルゴリズム，テスト容易化設計，テスト容易化高位合成に関する研究に従事．

増澤　利光　(正員)

昭 57 阪大・基礎工・情報卒．昭 62 同大大学院博士後期課程了．同年同大情報処理教育センター助手．同大基礎工助教授を経て，平 6 奈良先端大・情報科学助教授，現在に至る．平 5 コーネル大客員準教授 (文部省在外研究員)．分散アルゴリズム，並列アルゴリズム，テスト容易化設計，テスト容易化高位合成に関する研究に従事．ACM，IEEE，EATCS，情報処理学会各会員．工博．

藤原　秀雄　(正員)

昭 44 阪大・工・電子卒．昭 46 同大大学院博士後期課程了．阪大工学部助手，明治大理工学部教授を経て，現在奈良先端大情報科学教授．昭 56 ウォータールー大客員助教授．昭 59 マッギル大客員準教授．論理設計，高信頼設計，設計自動化，テスト容易化設計，テスト生成，並列処理，計算複雑度に関する研究に従事．著書 "Logic Testing and Design for Testability" (The MIT Press) など．大川出版賞．IEEE，情報処理学会各会員．工博．IEEE Fellow.