レジスタ転送レベルでのデータフロー依存型回路の階層テスト容易化設計法

永井 慎太郎,　　大竹 哲史,　　藤原 秀雄

奈良先端科学技術大学院大学　情報科学研究科
〒630-0101　奈良県生駒市高山町8916-5
TEL: 0743-72-5226　FAX: 0743-72-5229
E-mail: {*shinta-n, ohtake, fujiwara*}*@is.aist-nara.ac.jp*

あらまし　本論文では，コントローラの機能を考慮したデータパスの階層テスト容易化設計法を提案する．データパスの階層テスト生成法では，各回路要素に対してテスト生成およびテストプラン生成を行う．テストプランとは，外部入力から回路要素の入力へテストベクトルを正当化し，その応答を外部出力へ伝搬するための制御ベクトルの時系列をいう．提案手法では，拡張データフローグラフを用いてコントローラから制御ベクトル系列を抽出し，これを用いてテストプランを構成する．これにより，データパスへテストプランを供給するための付加回路の面積を小さくできる．提案手法はさらに，実動作速度でのテスト実行(at-speed test)が可能で，データパスに対して完全故障検出効率を保証できる．

A Method of Design for Hierarchical Testability for Data Flow Intensive Circuits at Register Transfer Level

Shintaro Nagai ,　Satoshi Ohtake　and　Hideo Fujiwara

Graduate School of Information Science, Nara Institute of Science and Technology
8916-5, Takayama, Ikoma, Nara 630-0101
TEL: +81-743-72-5226　FAX: +81-743-72-5229
E-mail: {*shinta-n, hiroki-w, ohtake, fujiwara*}*@is.aist-nara.ac.jp*

Abstract　This paper proposes a non-scan DFT method for hierarchical testability of a register-transfer level data path using control vector sequences generated by an original controller. In hierarchical test generation, a test plan for each module in the data path is generated. The test plan consists of a control vector sequence that can justify any value to the inputs of the module under test from some primary inputs and can propagate its output value to a primary output. In order to generate a control vector sequence for a test plan from the original controller, we extract an extended test control data flow graph from the data path and the controller. In our proposed method, the area overhead for a hierarchically testable data path is smaller than our previous work since the area overhead for the test controller to supply such test plans to the data path is small. Furthermore, our proposed method can achieve complete fault efficiency and at-speed testing.

1 はじめに

近年のVLSI回路の大規模化・高集積化により，回路のテストは困難な問題になっている．そのため，回路をテストの容易な回路に変更するテスト容易化設計の研究が進められている．テスト容易化設計では，テスト容易化のための付加回路の面積オーバヘッドをできるだけ小さく抑え，テスト生成やテスト実行にかかる時間の短縮や，故障検出効率[1]の向上が目標である．

組合せ回路に対しては実用的なテスト生成時間で完全故障検出効率[2]を達成できるテスト生成アルゴリズムが提案されている[1]．これに対して，順序回路では一般に実用的なテスト生成時間で高い故障検出効率を得るのは困難である．そのため，順序回路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が可能となるように回路のテスト容易化を行う手法が提案されている．

代表的なテスト容易化設計法として完全スキャン設計法がある．完全スキャン設計法では，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を行い，高い故障検出効率を達成できるが，実動作速度でのテスト実行ができない．また，回路の大規模化に伴い，テスト生成の対象となる回路の規模が大きくなると，テスト生成時間が長くなり，故障検出効率が低くなる可能性がある．さらに完全スキャン設計法では，長いテスト実行時間を要する．

完全スキャン設計法での問題点を解消するために，レジスタ転送(RT)レベルでのデータパスを対象としたテスト生成法やテスト容易化設計法が提案されている[2]~[8]．これらの手法はデータパスの階層テスト生成法[9]に基づいている．階層テスト生成法では，ゲートレベルにおいて組合せ回路で構成される回路要素単体に対してテスト生成を行い，RTレベルで各回路要素に対してテストプラン生成を行う．テストプランとは，外部入力から回路要素の入力へテストベクトルを正当化し，その回路要素の出力応答を外部出力へ伝搬するための制御ベクトルの時系列をいう．階層テスト生成法では組合せ回路で構成される回路要素単体に対してテスト生成を行うので，テスト生成の対象回路の規模が小さく，短いテスト生成時間で完全故障検出効率を達成できる．また，回路の通常動作時のデータ転送に用いる信号線上でテストベクトルの正当化および出力応答の伝搬を行うので，完全スキャン設計法に比べてテスト実行時間が短く，実動作速度でのテスト実行が可能である．しかし一般に各回路要素に対してテストプランが存在するとは限らないので，これらの手法ではテスト対象の回路要素に対してテストプランが存在するようにデータパスのテスト容易化を行っている．

データパスの強可検査テスト容易化設計法[2]では，レジスタのホールド機能および演算器のスルー機能を用いてデータパスを設計変更する．この手法では，テスト生成時間およびテスト実行時間は完全スキャン設計法に比べて短く，実動作速度でのテスト実行が可能である．強可検査法[3]では，テストプランを通常動作に用いるコントローラの機能を考慮せずに構成しているので，テストプランをデータパスへ供給するためのテストコントローラおよびマルチプレクサ(MUX)を回路内部に付加している．しかし強可検査法では，テストコントローラや付加したMUXの面積が大きいため，回路全体の面積オーバヘッドが完全スキャン設計法に比べて大きいという問題がある．

強可検査法でのテストコントローラの面積を削減するために，データパスの新しいテスト容易性として固定制御可検査性を導入し，固定制御可検査性に基づくテスト容易化設計法(固定制御可検査法)を提案した[4]．固定制御可検査法では，演算器のスルー機能，MUXおよびバイパスレジスタを用いてデータパスを設計変更する．固定制御可検査法は，強可検査法での利点を持ち，強可検査法よりも面積オーバヘッドを削減している．しかし，依然として回路全体の面積オーバヘッドが完全スキャン設計法に比べて大きいという問題がある．

コントローラの機能を利用したデータパスのテスト容易化設計法[5]では，データパスの各回路要素に対してテストプランの存在を保証するために，レジスタのホールド機能および演算器のスルー機能を用いてデータパスを設計変更する．また，データパスの各回路要素に対してテストプランを供給する機能をコントローラに付加しており，完全故障検出効率を保証できる．

Genesis[6]~[8]では，データパス中の各演算器に対するテストプランをコントローラの通常動作時の出力系列(以下では，コントローラの制御系列と呼ぶ)を用いて構成するためのテスト容易化設計法を提案している．Genesisではコントローラの制御系列を抽出するために，データフローグラフを生成し，そのデータフローグラフ上で，データパス中の各演算器に対するテストプランをコントローラの制御系列で構成できるかどうかを解析している．テストプランを構成できない場合は，MUXを用いてデータ

---

[1] 回路中のテスト生成の対象となる全故障数に対する，テスト生成アルゴリズムによって生成されたテスト系列が検出可能な故障数とテスト不可能と判明した故障数の和の割合をいう．

[2] 故障検出効率が100%の場合をいう．

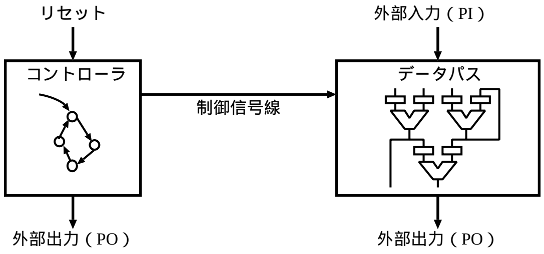


図 1: RT レベルにおけるデータフロー依存型回路

パスを設計変更する．各演算器のテストプランは付加した MUX の制御ベクトルとコントローラの制御系列で構成されるため，テストプランをデータパスへ供給するための回路の面積オーバヘッドが小さい．しかしデータパス中の MUX や，付加した MUX をテストの対象としておらず，それらの MUX に対してテストプランを生成しないため，完全故障検出効率を保証できない．

本論文では，データパス中の組合せ回路で構成される全ての回路要素に対して階層テストを実現するためのテスト容易化設計法を提案する．提案手法では，テストプランをデータパスへ供給するための機能をコントローラに付加せずに，コントローラの制御系列を用いてテストプランを構成する．提案するテスト容易化設計法では，演算器のスルー機能，定数発生器器および MUX を用いてデータパスを設計変更する．Genesis ではデータパスに対して完全故障検出効率を保証できないのに対して，提案手法では Genesis と同等の面積オーバヘッドで，強可検査法および固定制御可検査法と同様に完全故障検出効率を保証できる．

2　諸定義

2.1　データフロー依存型回路

RT レベルにおいて，回路はコントローラとデータパスから構成される．データフロー依存型回路では，コントローラとデータパスは制御信号線のみで接続され (図??)，コントローラはリセット入力のみを持つ．RT レベルにおいてコントローラは状態遷移図，データパスは回路要素および回路要素を相互に接続する信号線で記述される．回路要素は，外部入力，外部出力，定数入力，ホールド機能を持つレジスタと持たないレジスタ，MUX，加算器や乗算器などの演算器に分類される．以下では，組合せ回路で構成される MUX および演算器を組合せ回路要素と呼ぶ．各回路要素は入出力を持ち，入力はデータを入力するデータ入力とコントローラから制御値を入力する制御入力に分類され，出力はデータを出力するデータ出力がある．信号線は回路要素のデータ入出力を接続するためのデータ信号線とコントローラと制御入力を接続するための制御信号線に分類される．本論文では，以下の条件を満たすデータパスを対象とする．

A1: 回路要素の各データ入出力のビット幅は全て等しい．

A2: 各回路要素は，1 個または 2 個のデータ入力，1 個のデータ出力，高々1 個の制御入力を持つ．

2.2　階層テスト生成法

階層テスト生成法は，次の 2 段階から成る．第 1 段階ではゲートレベルにおいて，各組合せ回路要素単体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ベクトルを生成する．第 2 段階では RT レベルまたは動作記述レベルにおいて，テストベクトルを外部入力から組合せ回路要素へ正当化し，その出力応答を外部出力へ伝搬するためのテストプラン (制御ベクトル系列) を求める．階層テスト生成が可能なデータパスの性質として，強可検査性 [2] が提案されている．

定義 1 (強可検査性) データパス DP 中の各組合せ回路要素 M に対してテストプラン TP が存在し，TP で外部入力から M の入力ポートへ任意の値を正当化，かつ M の出力ポートから任意の値を外部出力へ伝搬できるとき，DP は強可検査であるという．
□

強可検査データパスには，次の特長がある．

- 各組合せ回路要素単体に対して，組合せ回路用のテスト生成アルゴリズムを適用するので，短いテスト生成時間で完全故障検出効率を達成できる．
- 各組合せ回路要素に対してテストプランの存在を保証しているので，データパス全体に対して完全故障検出効率を達成できる．

2.3　拡張データフローグラフ

ここでは，新しいデータフローグラフとして拡張データフローグラフを導入する．拡張データフローグラフ (以下では ETCDF，Extended Test Control Data Flow と呼ぶ) は，有向グラフ $G = (V, E, c)$ で与えられる．V は演算を表す頂点の集合，E はデータ転送を表す辺の集合，c:$V \mapsto \mathcal{N}$($\mathcal{N}$:自然数) は制御

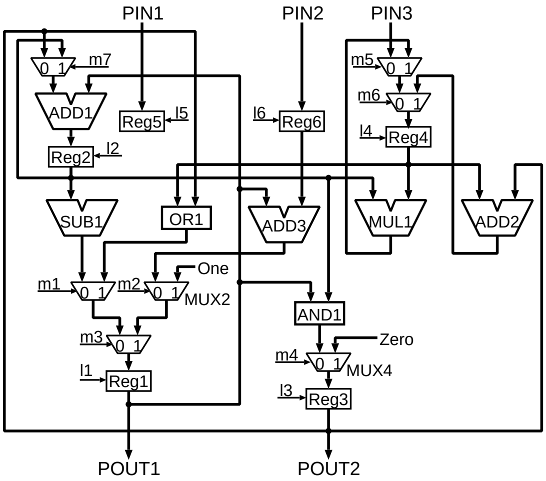


図 2: $Tseng$ データパス

Input	State		Outputs												
reset	PS	NS	l1	l2	l3	l4	l5	l6	m1	m2	m3	m4	m5	m6	m7
1	Any	S1	1	0	1	1	1	1	0	1	1	1	0	0	0
0	S1	S2	0	1	0	0	0	0	0	0	0	0	0	0	0
0	S2	S3	1	0	0	1	0	0	0	0	0	0	1	0	0
0	S3	S4	1	1	0	1	0	0	0	0	1	0	0	1	1
0	S4	S5	1	0	1	0	0	0	1	0	0	0	0	0	0
0	S5	S1	0	0	0	1	1	1	0	0	0	0	0	0	0

図 3: $Tseng$ コントローラ状態遷移表

ステップである．頂点 $v \in V$ は，データパスの外部入力，外部出力，定数入力，演算器または MUX に対応する．辺 $e(v_i, v_j) \in E$ が $c(v_i) \neq c(v_j)$ を満たすならば，辺 e は 2 つの演算頂点 v_i と v_j に対応する回路要素間に存在するレジスタとそれらの回路要素を接続するデータ信号線に対応する．辺 $e(v_i, v_j) \in E$ が $c(v_i) = c(v_j)$ を満たすならば，辺 e はデータ信号線に対応する．

ETCDF は 2 種類の頂点で構成される．1 つは実行される演算に対応する頂点で，その頂点に対応する回路要素の入力に値が伝搬され，レジスタまたは外部出力にその回路要素の出力応答を伝搬する．もう 1 つは実行されない演算に対応する頂点 (区別するために以下では，ダミー頂点と呼ぶ) で，そのダミー頂点に対応する回路要素の出力応答がレジスタおよび外部出力に伝搬しない．ETCDF は 2 種類の辺で構成される．1 つは実行される 2 つの演算頂点を接続する辺である．もう 1 つはレジスタとダミー頂点またはダミー頂点と MUX に対応する頂点間を接続する辺 (区別するために以下では，ダミー辺と呼ぶ) である．

ここで，演算器のスルー機能について考える．ある制御ステップで，スルー機能付き演算器 m の機能がスルー演算に選択されている場合，m に対応する頂点は ETCDF 上には表さず，スルー演算によるデータフローについては単に辺で表す．

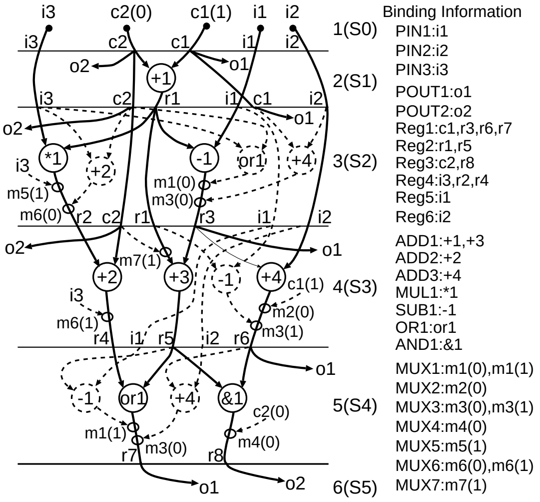


図 4: 拡張データフローグラフ ($Tseng$)

ベンチマーク回路 $Tseng$ のデータパスとコントローラの状態遷移表をそれぞれ，図 2と図 3に示す．制御ステップ数を 6 としたときの $Tseng$ に対する ETCDF を図 4に示す．図 4において，大きい頂点は演算器に対応し，小さい頂点は MUX に対応する．図 4において，破線で示した頂点がダミー頂点を表し，破線で示した辺がダミー辺を表す．Genesis[8] のデータフローグラフ (TCDF，Test Control Data Flow) では演算器に対応する頂点のみ表しているのに対して，提案手法の ETCDF では演算器および MUX に対応する頂点を表している．

ETCDF 上のダミー頂点およびダミー辺に関する回路の機能は回路外部に影響しないが，テストプランを構成するためのテスト容易化の際にこの機能を考慮することで，面積オーバヘッドを軽減することができる．また ETCDF では，コントローラのリセットは最初の制御ステップでのみ実行されるものとする．

3 提案手法

提案するテスト容易化設計法では，演算器のスルー機能，定数発生器および MUX を用いて，与えられたデータパスを強可検査データパスに設計変更する．テストプランをデータパスへ印加するのに全ての制御信号線 (付加した回路要素の制御信号線も含む) を付加回路によって制御すれば，付加回路の面積が大きくなる．そのため提案手法では，データパスに付加した回路要素以外の回路要素はコントローラの制御系列を用いて制御する．提案手法は以下の 2 ステップからなる (図 5参照)．

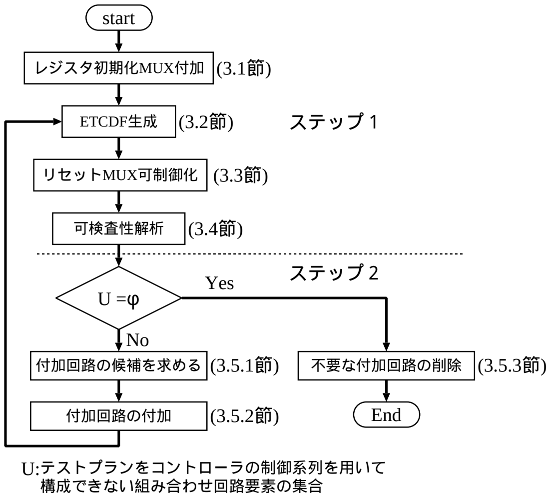


図5: テスト容易化設計手続き

ステップ1: データパス中の各組合せ回路要素に対してコントローラの制御系列を用いてテストプランを構成できるかどうかを調べる．

ステップ2: ステップ1でテストプランを構成できなかった組合せ回路要素に対してテスト容易化を行う．

ステップ1では，コントローラの制御系列を抽出するために，コントローラとデータパスからETCDFを生成する(3.2節)．ETCDF上で，テスト対象となる組合せ回路要素に対応する頂点の入力と出力に直接接続する辺の可制御性および可観測性を調べることで，テストプランをコントローラの制御系列を用いて構成できるかどうかを調べる(3.4節)．

ステップ2では，テストプランをコントローラの制御系列を用いて構成できるように，データパスを設計変更する(3.5節)．テスト容易化で用いる付加回路としては，面積オーバヘッドをできるだけ小さく抑えるために，演算器のスルー機能，定数発生器およびMUXを用いる．このとき，各組合せ回路要素に対するテストプランの付加回路の制御については，1つの制御ベクトルで構成できるようにする．

テストプランの供給方法を図6に示す．提案手法では各テストプランについて，付加回路の制御を1つの制御ベクトルで構成するので，付加回路の制御用のハードウェアをレジスタ(テストレジスタと呼ぶ)のみで構成できる．従って提案手法では，強可検査法および固定制御可検査法に比べて少ない面積でテストプランの供給を実現している．テストレジスタの故障によって，データパスの付加回路に誤った制御ベクトルが供給される場合があるので，データパスのテストができない可能性がある．そのため，データパスをテストする前にテストレジスタからデータパスの付加回路へ供給する制御ベクトルを外部で観測する必要がある．そこで，データパスの外部出力側にMUXを付加し(図6中のMUX1)，テストレジスタの値を観測する．テストレジスタの故障を検出すればテストは終了する．このMUXはテストレジスタから制御する(図6中のS_0)．テストピンはテストレジスタのホールド制御用の外部ピンである．

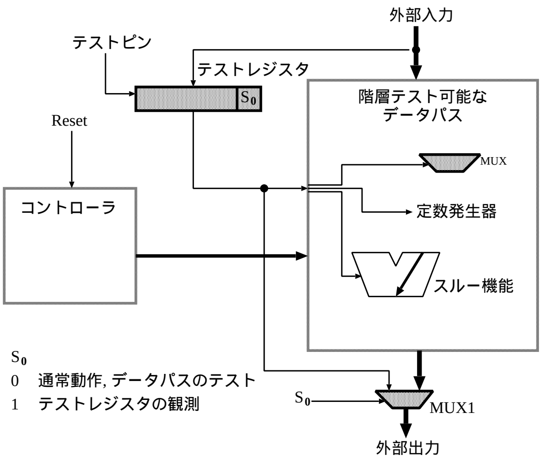
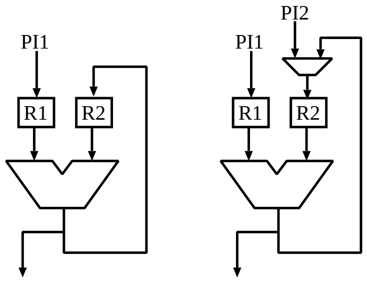


図6: テストプランの供給方法



(a)設計変更前　(b)設計変更後

図7: レジスタ初期化MUX付加

3.1 レジスタ初期化MUX付加

データパス中の全ての組合せ回路要素に対応する頂点をETCDF上に表すことができるように前処理としてデータパスを設計変更する．与えられたデータパス中のあるレジスタを外部から何らかの値に初期化できないとき，外部からそのレジスタまでのデータフローは存在しない．前処理では，初期化できないレジスタの直前にMUXを付加して，外部入力とそのMUXを接続し，外部から初期化できるように設計変更する．例えば，図7(a)のレジスタR2は外部から初期化できないので，R2の直前にMUXを付加し，外部入力とそのMUXを接続する(図7(b))．テストの際(テストプランの生成の時も含む)に，外部入力から初期化できないレジスタへのデータフローが存在するように，付加したMUXの制御を固定する．

3.2　ETCDF 生成

コントローラに接続された制御信号線以外の制御信号が決められたとき，ETCDF の生成は，以下の ETCDF 生成手続きおよび ETCDF 更新手続きにより行う．

ETCDF 生成手続き: 与えられたデータパスおよびコントローラの状態遷移表をそれぞれ，DP および FSM とする．また，l を状態遷移回数，s_0 をリセット状態とする．レジスタ初期化 MUX 付加により，外部入力または定数入力から全てのレジスタへのデータフローが存在することが保証されている．従って，全てのレジスタを初期化するのに必要な状態遷移を起こせば，全ての回路要素に対応する頂点を ETCDF 上に表現することができる．ここで，l は与えられるものとする．ただし，l はレジスタ初期化に必要な状態遷移回数以上の整数値とする．s_0 に対応する 1 番目の制御ステップから l 番目の制御ステップまで ETCDF 更新手続きを繰り返す．

ETCDF 更新手続き (i 番目の制御ステップを ETCDF に追加する手続き): $i-1$ 番目の制御ステップまでの ETCDF を G_{i-1} とする．s_{i-1}，s_i をそれぞれ，$i-1$,i 番目の制御ステップに対応する FSM の状態とする．E_{i-1} を $i-1$ 番目の制御ステップの辺の集合とする．s_i において，G_{i-1} に追加する頂点の集合および辺の集合をそれぞれ，V_i，E_i とし，各集合は最初は空とする．このとき，以下の手続きによって得られる V_i，E_i を G_{i-1} に加える．

1. s_i でホールドモードになっているレジスタに対応する辺が E_{i-1} に存在すれば，その辺を E_i に追加する．

2. 次の条件を同時に満たす演算器に対応する頂点を V_i に追加する．

- **c1:** 演算器の入力へ外部入力または E_{i-1} の辺に対応するレジスタからの経路が存在する．
- **c2:** 演算器の出力から外部出力または s_i でロードモードになっているレジスタへの経路が存在する．

3. 2 で V_i の頂点として追加した演算器へレジスタを介さない経路をもつ外部入力および外部出力に対応する頂点を V_i に追加する．

4. 2 で V_i に追加した演算器間，演算器とレジスタ間または演算器と外部入出力間に存在する MUX を考える．MUX の s_i で選択されていない入力とレジスタまたは外部入力との接続関係を調べ，MUX の選択されていない入力へ E_{i-1} の辺に対応するレジスタからの経路があるならば，この MUX に対応する頂点 v を V_i に追加する． さらに v に対応する MUX の選択されていない入力とその入力への経路を持つレジスタ間に演算器があれば，その演算器をダミー頂点として V_i に追加する．

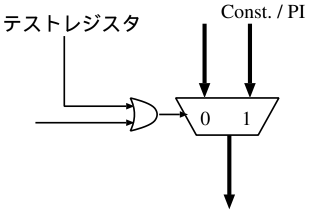


図 8: リセット MUX 可制御化

5. 各頂点間を接続する辺，ダミー辺およびロードモードになっているレジスタに対応する辺を E_i に追加する．

3.3　リセット MUX 可制御化

リセット状態において制御信号が v，他の状態において制御信号が $\bar{v}$ であるような MUX をリセット MUX と呼ぶことにする (図 2中の MUX2,MUX4)．リセット MUX は，レジスタを定数または外部入力から初期化するために用いられる．ETCDF の 1 番目の制御ステップにリセット MUX に対応する頂点が現れない場合は，リセット状態にのみ活性化されるリセット MUX の故障を検出するためのテストプランを生成することができない．この場合にはテストレジスタから任意の制御値をリセット MUX の制御入力に与えるための回路をリセット MUX の制御信号線上に付加する (図 8)．

リセット状態において制御信号が 1，他の状態において制御信号が 0 のリセット MUX を考える．制御信号 0 が印加される制御ステップで，リセット MUX を通してデータ転送が行われるとき，リセット MUX に対応する頂点がその制御ステップに現れる．その頂点に対してテストプランの存在を保証すれば，リセット MUX の制御入力に 1 を印加することで，リセット状態にのみ活性化されるリセット MUX の故障を検出できる．

リセット MUX の制御のための付加回路については，テストレジスタから制御する．テストプランの生成の際には，リセット MUX に対してリセット状態で制御信号が v のときのみ活性化されるリセット MUX の故障を検出するためのテストプランを生成することはできないが，$\bar{v}$ が印加される制御ステッ

表 1: テストプラン生成対象頂点の入出力辺に対する尺度

可検査性解析 対象回路要素	入力辺	出力辺
2 入力演算器	C_g, C_g	O
1 入力演算器	C_g	O
MUX(制御信号 0) (左入力を選択)	左入力:C_{all1}, 右入力 C_0	O
	左入力:C_0, 右入力:—	O
MUX(制御信号 1) (右入力を選択)	左入力:C_0, 右入力 C_{all1}	O
	左入力:—, 右入力:C_0	O

—:尺度を付けないことを表す．

プに存在するリセット MUX に対応する頂点に対してテストプラン p を生成できる．リセット状態で制御信号が v のときのみ活性化されるリセット MUX の故障を検出するためのテストプランは，付加回路を用いて制御入力に v を与えることによって p で代用できる．

3.4 可検査性解析

データパス中の各組合せ回路要素 m に対してテストプランをコントローラの制御系列を用いて構成できるかどうかを調べるために，ETCDF 上で m に対応する頂点 n の入力に接続された辺 (入力辺) の可制御性および出力に接続された辺 (出力辺) の可観測性を調べる．可検査性解析は次の 2 段階からなる．第 1 段階では，n の各入出力辺に対して表??に基づいて尺度を付ける．尺度とは，レジスタまたはデータ信号線上の値の制御および観測が可能かどうかを示す性質である．尺度として以下の 6 つの性質を用いる．

一般可制御性: 辺 x に任意の値を制御可能かどうかを示す尺度をいい，$C_g(x)$ で表す．

0 可制御性: 辺 x に値 0 を制御可能かどうかを示す尺度をいい，$C_0(x)$ で表す．

1 可制御性: 辺 x に値 1 を制御可能かどうかを示す尺度をいい，$C_1(x)$ で表す．

全 1 可制御性: 辺 x に全て 1 からなる値を制御可能かどうかを示す尺度をいい，$C_{all1}(x)$ で表す．

可観測性: 辺 x の値を観測可能かどうかを示す尺度をいい，$O(x)$ で表す．

可検証性: 辺 x の値を観測可能かどうか，または，辺 x に任意の値を制御可能かどうかを示す尺度をいい，$V(x)$ で表す．

演算器のテストでは，ETCDF 上で対応する頂点の全ての入力辺が一般可制御性 C_g を満たし，かつ出力辺が可観測性 O を満たさなければならない．MUX のテストでは，制御入力と各データ入力に対して，(0,all1,0)，(0,0,–)，(1,0,all1)，(1,–,0) の 4 通りのテストパタンを正当化できれば，MUX の全ての故障をテストできる．ここで「–」はどんな値でもよいことを意味する．表??の MUX において，上段は MUX の制御信号線上の故障を検出するために各入出力辺に付ける尺度，下段は MUX のデータ信号線上の故障を検出するために各入出力辺に付ける尺度を表す．また，表??の MUX において「—」は尺度を付ける必要がないことを表す．

第 2 段階では，尺度が付いた頂点 n の入出力辺ごとに，その尺度を尺度変換表 (表??) に基づいて別の辺に対する尺度に変換していき，外部入力または外部出力に接続する辺に到達するまで尺度の変換を繰り返す (以下では，尺度変換と呼ぶ)．尺度変換表は，尺度変換において，各演算器に対応する頂点に付いた尺度を，その演算器の関数に基づいて別の辺の尺度にどう変換できるかを示した表である．尺度変換は以下の 2 ステップで行う．ここで，頂点 n に対して尺度を付けた辺の集合を $\mathbf{M}(n)$ とする．また，n の尺度変換によって尺度が付いた全ての辺の集合を $\mathbf{M}'(n)$ とし，$\mathbf{M}'(n)$ は最初は空とする．

ステップ 1: $\mathbf{M}(n) = \phi$ であれば尺度変換は終了し，そうでなければ以下を行う．$\mathbf{M}(n)$ から 1 個の尺度の付いた辺 e を削除し，その辺を $\mathbf{M}'(n)$ に追加する．辺 e が外部入力または外部出力に接続する辺であればステップ 1 へ戻り，そうでなければステップ 2 を行う．

ステップ 2: 辺 e の尺度を尺度変換表に基づいて他の辺の尺度に変換し，尺度が付いた辺を $\mathbf{M}(n)$ に追加してステップ 1 へ戻る．

テスト対象の組合せ回路要素 m に対応する頂点 n とは別の，n より上流の m に対応する頂点 n' を介して値を伝搬するようなテストプランを構成したとき，m を通して m のテストのための値を伝搬しなければならない．そのため，m がその値を伝搬できるかを検証するために，頂点 n' の出力辺の可観測性を調べる必要がある．

また，テスト対象の組合せ回路要素 m' に対応する頂点 k の出力辺に付けた可観測性を外部出力に対応する頂点の入力辺まで尺度変換を行う際には，k とは別の，k より下流の m' に対応する頂点の出力辺への尺度変換を行わない．すなわち，m' を含む閉路を通らずに，m' の出力応答を外部出力へ伝搬する．これにより，不要な探索を削減することができる．

一般に組合せ回路要素に対応する頂点は ETCDF 上に複数存在するが，少なくとも 1 つの頂点に対す

表 2: 尺度変換表

尺度	+	−	×	÷	**AND**	**OR**	**XOR**	**NOT**	**MUX**
$C_g(z)$	$C_g(x)$, $V(y)$ $C_g(y)$, $V(x)$	$C_g(x)$, $V(y)$ $C_g(y)$, $V(x)$	$C_g(x)$,$C_1(y)$ $C_g(y)$,$C_1(x)$	$C_g(x)$,$C_1(y)$	$C_g(x)$,$C_{all1}(y)$ $C_g(y)$,$C_{all1}(x)$	$C_g(x)$,$C_0(y)$ $C_g(y)$,$C_0(x)$	$C_g(x)$,$V(y)$ $C_g(x)$,$V(x)$	$C_g(x)$	$C_g(z)$
$C_1(z)$	$C_1(x)$,$C_0(y)$ $C_1(y)$,$C_0(x)$	$C_1(x)$,$C_0(y)$	$C_1(x)$,$C_1(y)$	$C_1(x)$,$C_1(y)$ $C_{all1}(x)$, $C_{all1}(y)$	$C_1(x)$,$C_{all1}(y)$ $C_1(y)$,$C_{all1}(x)$ $C_1(x)$,$C_1(y)$	$C_1(x)$,$C_0(y)$ $C_1(y)$,$C_0(x)$ $C_1(x)$,$C_1(y)$	$C_0(x)$,$C_1(y)$ $C_0(y)$,$C_1(x)$	$C_g(x)$	$C_1(z)$
$C_0(z)$	$C_0(x)$,$C_0(y)$	$C_0(x)$,$C_0(y)$ $C_1(x)$,$C_1(y)$ $C_{all1}(x)$, $C_{all1}(y)$	$C_0(x)$ $C_0(y)$	$C_0(x)$	$C_0(x)$ $C_0(y)$	$C_0(x)$,$C_0(y)$	$C_0(x)$,$C_0(y)$ $C_1(x)$,$C_1(y)$ $C_{all1}(x)$, $C_{all1}(y)$	$C_{all1}(x)$	$C_0(z)$
$C_{all1}(x)$	$C_{all1}(x)$,$C_0(y)$ $C_{all1}(y)$,$C_0(x)$	$C_{all1}(x)$,$C_0(y)$	$C_{all1}(x)$,$C_1(y)$ $C_{all1}(y)$,$C_1(x)$	$C_{all1}(x)$,$C_1(y)$	$C_{all1}(x)$, $C_{all1}(y)$	$C_{all1}(x)$ $C_{all1}(y)$	$C_{all1}(x)$,$C_0(y)$ $C_{all1}(y)$,$C_0(x)$	$C_0(x)$	$C_{all1}(z)$
$O(x)$	$V(y)$,$O(z)$	$V(y)$,$O(z)$	$C_1(y)$,$O(z)$	$C_1(y)$,$O(z)$	$C_{all1}(y)$,$O(z)$	$C_0(y)$,$O(z)$	$V(y)$,$O(z)$	$O(z)$	$O(x)$
$O(y)$	$V(x)$,$O(z)$	$V(x)$,$O(z)$	$C_1(x)$,$O(z)$	$C_g(x)$,$O(z)$	$C_{all1}(x)$,$O(z)$	$C_0(x)$,$O(z)$	$V(x)$,$O(z)$	-	$O(y)$
$V(x)$	$V(y)$,$O(z)$	$V(y)$,$O(z)$	$C_1(y)$,$O(z)$	$C_1(y)$,$O(z)$	$C_{all1}(y)$,$O(z)$	$C_0(y)$,$O(z)$	$V(y)$,$O(z)$	$O(z)$	$V(x)$
$V(y)$	$V(x)$,$O(z)$	$V(x)$,$O(z)$	$C_1(x)$,$O(z)$	$C_g(x)$,$O(z)$	$C_{all1}(x)$,$O(z)$	$C_0(x)$,$O(z)$	$V(x)$,$O(z)$	-	$V(y)$
$V(z)$	$V(x)$,$V(y)$	$V(x)$,$V(y)$	$V(x)$,$V(y)$	$V(x)$,$V(y)$	$V(x)$,$V(y)$	$V(x)$,$V(y)$	$V(x)$,$V(y)$	$V(x)$	$V(z)$

z = x (operation) y

る尺度変換で以下の問題が生じなければ，テストプランをコントローラの制御系列を用いて構成できる．

1. 定数入力に一般可制御性が付く，または異なる定数の可制御性が付く場合
2. ETCDF 上のある 1 つの辺において，尺度の衝突が生じる場合

ここで尺度の衝突とは，可検査性解析において尺度変換で複数の異なる尺度が 1 つの辺に対して割り当てられる場合，または，複数の一般可制御性または複数の可検証性が 1 つの辺に対して割り当てられる場合をいう．後者の場合，頂点 n に対応する回路要素の各データ入力に異なる任意の値が印加できない．可観測性は n の 1 つの出力辺に要求され，その可観測性は n とは別の頂点の出力辺の可観測性に変換されるので，1 つの辺に可観測性が同時に要求されることはない．

3.5 データパスの階層テスト容易化設計

可検査性解析でテストプランをコントローラの制御系列を用いて構成できないと判定された組合せ回路要素の集合を **U** とする．**U** 中のある組合せ回路要素 m に対応する頂点は複数存在し，各頂点に対して考えられる尺度変換も複数存在するので，3.4節で述べた問題点を解消するために必要な付加回路の候補を尺度変換ごとに求める (3.5.1節)．用いる付加回路は，演算器のスルー機能，定数発生器および MUX である．スルー機能は，加算器や乗算器などの演算器に対しては，マスク素子を用いることで低面積で実現できる．マスク素子を用いてスルー機能を実現できない場合は，MUX を用いてスルー機能を実現する．定数発生器は，出力に定数を発生するマスク素子で実現できる．m に対してテストプランを生成できることを保証するために必要な付加回路の面積は，尺度変換ごとに異なる．そこで提案手法では，m に対応する全ての頂点に対して考えられる尺度変換ごとに必要な付加回路の面積を表す重みを付ける．各付加回路の重みを，マスク素子と定数発生器は 1，MUX は 3 とする．**U** 中の全ての組合せ回路要素の全ての尺度変換に対して付加回路の候補を求めた後に，重みの小さい順に付加回路をデータパスに付加する (3.5.2節)．全ての組合せ回路要素に対してテストプランをコントローラの制御系列を用いて構成できることを保証した後に，不要な付加回路が存在するかどうかを調べ，不要な付加回路があれば削除する (3.5.3節)．

3.5.1 付加回路候補の求め方

1 つの尺度変換に対して，3.4節に示した問題点が発生していれば，それぞれの場合に応じて，以下のように付加回路の候補を求める．

問題点 (1):

i) 一般可制御性が定数入力に接続する辺に付いた場合: 定数入力の直後に MUX を付加すれば，外部入力から任意の値を直接制御できるようになる．この場合はこの MUX を候補とする．

ii) 定数 α の可制御性が定数 $c(\neq \alpha)$ に接続する辺に付いた場合: 定数 c の直後に α を発生する定数

発生器を付加すれば，定数αを制御できるようになる．この場合はこの定数発生器を候補とする．

問題点 (2):

$i)$ 1つの辺に一般可制御性が2回以上要求された場合:

- **a)** テスト対象回路要素がデータ入力を1つ持つ場合，そのデータ入力の直前にMUXを付加すれば，外部入力から任意の値を直接制御できるようになる．この場合はこのMUXを候補とする．

- **b)** テスト対象回路要素がデータ入力を2つ持つ場合，その回路要素に対応する頂点の入力辺までに再収斂経路[3]の多い入力辺をeとする．その回路要素のeの終点に対応するデータ入力の直前にMUXを付加してこの問題が解決できる場合は，このMUXを候補とする．

  これで解決できなければ，両方のデータ入力の直前にMUXを付加することにより，この問題を解決できる．この場合はこれらのMUXを候補とする．

$ii)$ $i)$以外で，1つの辺に複数の尺度が要求された場合:

ここで，テストプラン生成対象の組合せ回路要素をmとする．

- **a)** mとは別の2つのデータ入力を持つ回路要素m'に対応する頂点をn'とする．n'の入力辺をe_aおよびe_b，出力辺をe_oとする．e_oに付いた尺度をp_iとし，e_aのp_iとe_bのp'_iに尺度変換されているとする．e_bから外部入力に対応する頂点の出力辺までの間に尺度の衝突が発生している場合，e_aに対応するm'のデータ入力にスルー機能を付加すれば，e_bに無関係に，e_oのp_iをe_aのp_iに尺度変換できるので，この問題を解決できる．この場合はこのスルー機能を候補とする．

- **b)** mとは別の2つのデータ入力を持つ回路要素m''に対応する頂点をn''とする．n''の入力辺をe'_aおよびe'_b，出力辺をe'_oとする．e'_aに付いた尺度をp_jとし，e'_oのp_jとe'_bのp'_jに尺度変換されているとする．e'_bから外部入力に対応する頂点の出力辺までの間に尺度の衝突が発生している場合，e'_aに対応するm''のデータ入力にスルー機能を付加すれば，e'_bに無関係に，e'_aのp_jをe'_oのp_jに尺度変換できるので，この問題を解決できる．この場合はこのスルー機能を候補とする．

外部入力からmの1つのデータ入力まで値を伝搬するのに3個[4]以上スルー機能を必要とする場合，または，mのデータ出力から外部出力まで値を伝搬するのに3個以上スルー機能を必要とする場合は，付加回路の面積の増大を防ぐため，それらのスルー機能を候補とせずに1個のMUXを候補として選ぶことを考える．前者の場合，mのデータ入力の直前にMUXを付加すれば，外部入力から値を直接制御できるようになる．後者の場合，MUXを外部出力の直前に付加し，mのデータ出力とそのMUXを接続すれば，外部出力で値を直接観測できるようになる．これらの場合はこのMUXを候補とする．

mとは別の2入力演算器の両方のデータ入力にスルー機能が同時に候補として選ばれた場合には，同時に実現することができないので，どちらか一方の候補を選ばなければならない．候補として選ばれなかったスルー機能をMUXと置き換えて，外部入力からのmへのテストのための値の伝搬および外部出力へのmの出力応答の伝搬を行うことを考える．ただし，付加回路の面積の増大を防ぐために，できるだけ多くのスルー機能を1個のMUXに置き換える．外部入力からmの1つのデータ入力へテストのための値を伝搬するのにスルー機能を最も多く必要とするならば，そのデータ入力の直前にMUXを付加すれば，外部入力から値を直接制御できるようになる．mのデータ出力から外部出力へ出力応答を伝搬するのにスルー機能を最も多く必要とするならば，MUXを外部出力の直前に付加し，そのデータ出力とそのMUXを接続すれば，外部出力で値を直接観測できるようになる．これらの場合はこのMUXを候補とする．

3.5.2 付加回路の決め方

$\mathbf{U}$中の全ての組合せ回路要素に対応する各頂点について，全ての尺度変換ごとに求めた付加回路の候補の面積を表す重みを計算する．$\mathbf{U}$が空になるまで以下の手続きを繰り返す．

1. 重みの最も小さい尺度変換を選び，その尺度変換で候補となっている付加回路を全てデータパスに付加する．

2. 付加回路の制御の全ての組合せについてETCDFを生成し，$\mathbf{U}$に付加回路を追加する．

3. **2**で生成した各ETCDF上で，$\mathbf{U}$中の全ての組

---

[3] 再収斂経路とは，ETCDF上の頂点v_1および2つのデータ入力を持つv_1とは異なる頂点v_2に対して，v_1を始点，v_2を終点とする任意の異なる経路の対をいう．

[4] 1個のMUXの面積は3個のスルー機能の面積に等しい．

合せ回路要素に対応する各頂点に対して可検査性解析を行う．

3.5.3 付加回路の削除

3.5.2節で付加した回路要素のうち，不要な回路要素があるかどうかを調べる．付加回路を削除しても，コントローラの制御系列を用いて全ての組合せ回路要素に対してテストプランを構成できれば，その付加回路を削除する．

4 従来法との比較

本節では，Genesis[8]，強可検査法 [3]，固定制御可検査法 [4] および提案手法を比較する．これらの手法はデータパスの階層テストを実現するための非スキャンテスト容易化設計法であり，実動作速度でのテスト実行が可能である．

Genesis:

コントローラの制御系列を用いて，各演算器のテストプランを構成している．付加した回路要素 (MUX) の制御については，各演算器のテスト間は付加した MUX の制御を固定にしているため，制御用の回路をレジスタのみで構成でき，少ない面積でテストプランを供給できる．しかし，MUX や付加した MUX をテストの対象としておらず，それらの MUX に対してテストプランを生成しないため，全ての組合せ回路要素に対して完全故障検出効率を保証できない．

強可検査法:

はじめにコントローラと独立に (制御入力を自由に制御できるものとして) データパスをテスト容易化する．次に制御入力へテストプランを供給できることを保証するために，制御入力に MUX およびテストプランを生成するテストコントローラを付加する．回路内部に付加したテストコントローラからデータパスへテストプランを供給できるので，実動作速度でのテスト実行が可能となる．また，全ての組合せ回路要素に対して完全故障検出効率を保証できる．テストプラン長を考慮して設計変更しているので，テスト実行時間が短くなる．強可検査法でのテストプランでは，制御ベクトルが時刻ごとに変化するので，強可検査法ではテストコントローラを順序回路で構成している．そのため，強可検査法ではテストコントローラや MUX の面積が大きくなり，面積オーバヘッドが大きい．

固定制御可検査法:

各テストプランをテストベクトルの正当化，テスト，出力応答の伝搬の 3 つのフェーズに分けて，各フェーズにおける制御系列を 1 個の制御ベクトルで構成しているので，各テストプランは高々3 個の制御ベクトルで構成できる．これにより，テストコントローラを組合せ回路で構成できるので，固定制御可検査法では強可検査法に比べてテストコントローラの面積をより小さくすることができる．しかし，制御信号線上に付加した MUX の面積が強可検査法と同程度であるため，依然として面積オーバヘッドが大きい．

提案手法:

各テストプランを Genesis と同様にコントローラの制御系列を用いて構成している．従って，提案手法での面積オーバヘッドは強可検査法および固定制御可検査法よりも大幅に削減できる．提案手法では，コントローラの制御系列を用いて各組合せ回路要素に対してテストプランを構成するため，テスト実行時間は強可検査法および固定制御可検査法でのテスト実行時間よりも長くなる．Genesis では全ての組合せ回路要素に対して完全故障検出効率を保証できないのに対して，提案手法では強可検査法および固定制御可検査法と同様に完全故障検出効率を保証できる．

5 実験結果

Genesis，強可検査法，固定制御可検査法および提案手法を，面積オーバヘッド，テスト生成時間およびテスト実行時間について比較した．実験に使用した RT レベルベンチマーク回路は，LWF と JWF[10]，Paulin[8] および Tseng[11] である．これらの回路はデータフロー依存型回路である．4 つの回路の特性を表??に示す．#PI，#PO はコントローラおよびデータパスそれぞれの外部入力数および外部出力数を表す．コントローラに関して，#State および#Control はそれぞれ，状態数および制御出力数を表す．データパスに関して，#Reg. および#Mod. はそれぞれ，レジスタ数および演算器数を表す．論理合成ツールには AutoLogicII(Mentor Graphics) を使用した．実験ではコントローラの状態数を k とすると，$2k+1$ 番目の制御ステップまでの ETCDF を生成した．

テスト容易化設計に伴う付加回路のデータパスに対する面積オーバヘッドおよび外部ピンオーバヘッドを表??に示す．DP はデータパスのテスト容易化に伴う付加回路の面積オーバヘッド，TC はテストプランをデータパスへ供給するための付加回路の面積オーバヘッド，MUX はデータパスの外部出力に付加した MUX の面積オーバヘッドを示す．回路全体の面積オーバヘッドは，回路全体に対する，デー

表 3: コントローラ/データパス回路の特性

回路	面積 ($gate$)	コントローラ					データパス					
		#PI	#PO	#State	#Control	面積 ($gate$)	#PI	#PO	bit	#Reg.	#Mod.	面積 ($gate$)
LWF	1986	1	0	4	8	58	32	32	16	5	3	1924
JWF	6875	1	0	8	38	200	80	80	16	14	3	6672
Paulin	24966	1	0	6	16	124	64	64	32	7	4	24834
Tseng	15033	1	0	5	13	95	96	64	32	6	7	14930

表 4: 面積オーバヘッド

回路	面積オーバヘッド (%)																外部ピンオーバヘッド (#)			
	Genesis*				強可検査法				固定制御可検査法				提案手法				Genesis	強可検査法	固定制御可検査法	提案手法
		DP	TC	MUX		DP	TC	MUX		DP	TC	MUX		DP	TC	MUX				
LWF	— (0)	— (0)	— (0)	— (0)	34.5	5.7	24.3	4.5	30.5	5.6	20.4	4.5	7.3	3.8	2.3	1.2	0	3	4	1
JWF	— (0)	— (0)	— (0)	— (0)	32.8	1.6	26.4	4.8	37.7	6.0	26.5	5.2	2.1	1.1	0.7	0.4	0	3	4	1
Paulin	— (1.3)	— (1.0)	— (0.2)	— (0.1)	8.0	2.0	5.4	0.6	6.1	2.3	3.1	0.7	2.5	1.6	0.6	0.3	1	3	4	1
Tseng	— (2.0)	— (1.7)	— (0.2)	— (0.1)	11.5	3.6	6.8	1.1	12.8	5.8	5.8	1.2	2.4	1.5	0.6	0.3	1	3	4	1

—: Genesis では演算器のみテスト容易化しているため，全体のテスト容易化による面積オーバヘッドを示せない．

*: Genesis では完全故障検出効率を達成できていないが，強可検査法，固定制御可検査法および提案手法では完全故障検出効率を達成．

表 5: テスト生成結果

回路	テスト生成時間 (秒)				テスト実行時間 (サイクル)			
	Genesis^{*1}	強可検査法 ^{*2}	固定制御可検査法 ^{*2}	提案手法	Genesis^{*1}	強可検査法	固定制御可検査法	提案手法
LWF	— (0.38)	0.83	0.85	0.78 (0.38)	— (78)	229	229	196 (78)
JWF	— (0.38)	0.84	1.13	0.85 (0.67)	— (78)	720	742	934 (78)
Paulin	— (2.44)	3.08	3.12	5.02 (4.51)	— (1524)	1120	1334	1937 (1542)
Tseng	— (2.83)	3.65	3.98	3.78 (3.26)	— (1131)	11351	1236	1547 (1281)

—: Genesis では MUX のテストをしていないため，全体のテスト生成時間およびテスト実行時間を示せない．

^{*1}, ( ): 演算器のみのテスト生成時間およびテスト実行時間．

^{*2}: テストプラン生成時間を含む．

タパスのテスト容易化に伴う付加回路の面積の割合を示す．回路全体の面積オーバヘッドについて，提案手法は強可検査法および固定制御可検査法に比べて大幅に削減しており，Genesis とほぼ同等である．外部ピンオーバヘッドについて，提案手法は Genesis と同様に，テストレジスタ (図 6) のロード/ホールド用のテストピン 1 本のみであり，強可検査法および固定制御可検査法に比べて小さい．

テスト生成結果を表??に示す．Genesis および提案手法での括弧内のテスト生成時間およびテスト実行時間は演算器のみのテスト生成結果であり，Genesis では MUX に対してテストしていない．テスト生成時間はテストパタン生成時間とテストプラン生成時間からなる．Genesis および提案手法では，組合せ回路要素に対するテストプランは手動で求めたため，表??におけるテスト生成時間は組合せ回路要素に対するテストパタン生成時間のみを示している．一方，強可検査法および固定制御可検査法では，テスト生成時間にテストプラン生成時間が含まれているが，テストプラン生成時間は全ての回路に対して約 0.1 ~0.2 秒であった．従って，表??においてテスト生成時間はほとんどテストパタン生成時間で占められていることが分かる．これについては，提案手法でも同様である．提案手法でのテストプラン生成時間は強可検査法および固定制御可検査法に比べて長くなる可能性があるが，表??から分かるように，提案手法のテスト生成時間は強可検査法および固定制御可検査法とほぼ同じである．

強可検査法および固定制御可検査法ではテスト実行時間が短くなるように設計変更を行っているのに対して，提案手法ではコントローラの制御系列を用いて，各組合せ回路要素のテストプランを構成できるように設計変更を行う．従って，提案手法のテスト実行時間は，強可検査法および固定制御可検査法より長い．

提案手法でのテスト生成時間およびテスト実行時間は強可検査法および固定制御可検査法に比べて長くなった．これは面積オーバヘッドとテスト生成時間およびテスト実行時間のトレードオフを示している．提案手法での面積オーバヘッドは Genesis とほぼ同等であり，強可検査法および固定制御可検査法に比べて大幅に削減できることを実験で示した．

6　あとがき

本論文では，コントローラの制御系列を用いてデータパス中の各組合せ回路要素に対してテストプランを構成するための階層テスト容易化設計法を提案した．Genesis ではデータパス中の MUX や付加回路をテストの対象としておらず，それらの回路要素に対してテストプランを生成していないため，完全故障検出効率を保証できない．これに対して，提案手法では強可検査法および固定制御可検査法と同様に完全故障検出効率を保証できる．さらに提案手法での面積オーバヘッドは Genesis とほぼ同等で，強可検査法および固定制御可検査法に比べて大幅に削減できた．今後の課題としては，コントローラがステータス入力やリセット以外の外部入力を持ち，異なるビット幅，多入力多出力回路要素を持つ RT レベル回路を対象としたテスト容易化設計法に拡張することなどがある．

謝辞　本研究に際し，多くの貴重な意見を頂いた，井上美智子助教授ならびに本学の情報論理学講座の諸氏に深く感謝します．本研究は一部、奈良先端科学技術大学院大学支援財団教育研究活動支援による研究助成，および，新エネルギー・産業技術総合開発機構 (NEDO) から半導体理工学研究センター (STARC) に委託された「SoC 先端設計技術の研究開発」の一部として奈良先端科学技術大学院大学に再委託され実施されています．

参考文献

[1] H. Fujiwara, Logic testing and design for testability, The MIT Press, Cambridge (1985).

[2] 和田弘樹, K. K. Saluja, 増澤利光, 藤原秀雄, "完全故障検出効率を保証するレジスタ転送レベルデータパスの非スキャンテスト容易化設計法", 電子情報通信学会論文誌, Vol.J82-D-I, No.7, pp.843-851 (July 1999).

[3] S. Ohtake, H. Wada, T. Masuzawa and H. Fujiwara, "A non-scan DFT method at register transfer level to achieve complete fault efficiecy," in *Proc. of ASP-DAC*, pp.599-604 (2000).

[4] 永井慎太郎, 和田弘樹, 大竹哲史, 藤原秀雄, "固定制御可検査性に基づく RTL 回路の非スキャンテスト容易化設計法", 電子情報通信学会論文誌, Vol.J84-D-I, No.5, pp.454-465 (May 2001).

[5] 鈴木和博, 井上美智子, 藤原秀雄, "コントローラの機能を利用したデータパスのテスト容易化設計," 信学技法, FTS2000-86, pp.1-8 (Feb. 2001).

[6] S. Bhatia and N. K. Jha, "Genesis:A behavioral synthesis system for hierarchical testability," in *Proc. of EDTC*, pp.272-276 (Feb. 1993).

[7] I. Ghosh, A. Raghunathan and N. K. Jha, "Design for hierarchical testability of RTL circuits obtained by behavioral synthesis," *IEEE Trans. on CAD*, Vol.16, No.9, pp.1001-1014 (Sep. 1997).

[8] I. Ghosh, A. Raghunathan and N. K. Jha, "A Design for testability technique for RTL circuits using control/data flow extraction," *IEEE Trans. on CAD*, Vol.17, No.8, pp.706-723 (Aug. 1998).

[9] B. T. Murray and J. H. Hayes, "Hierarchical test generation using pre computed tests for modules," *IEEE Trans. on CAD*, Vol.9, No.6, pp.594-603 (June 1990).

[10] M. Inoue, K. Noda, T. Higashimura, T. Masuzawa and H. Fujiwara, "High-level synthesis for weakly testable data paths," *IEICE Trans. on Inf & Syst.*, Vol.E81-D, No.7, pp.645-653 (July 1998).

[11] I. Ghosh, N. K. Jha and S. Bhawmik, "A BIST scheme for RTL controller-data paths based on symbolic testability analysis," in *Proc. of DAC*, pp.554-559 (1998).