論　文

階層 BIST：低いオーバヘッドを実現する Test-per-clock 方式 BIST

山口　賢一[†*]　　井上美智子[†]　　藤原　秀雄[†]

Hierarchical BIST: Test-Per-Clock Scheme BIST with Low Overhead

Kenichi YAMAGUCHI[†*], Michiko INOUE[†], and Hideo FUJIWARA[†]

あらまし　レジスタ転送レベル（RTL）回路に対する階層組込み自己テスト（BIST）のためのテスト容易化設計法を提案する．階層 BIST とは，RTL 回路に対する BIST を RTL とゲートレベルの二つの階層で行う BIST を総称する概念である．RTL においては，テスト対象モジュールに対して，テストパターン発生器で発生したパターンを印加し，応答解析器で観測するための経路を生成する．ゲートレベルでは，テスト対象モジュールに対して故障シミュレーションによる故障検出率の評価を行う．階層 BIST に基づく手法の利点は，高い故障検出率，小さいハードウェアオーバヘッドを達成できることである．本論文では，階層 BIST のための可検査性として新たに時分割単一制御可検査性を提案し，テスト実行時間とハードウェアオーバヘッドの削減を行い，実験により提案手法の有効性を示す．

キーワード　テスト容易化設計，レジスタ転送レベル，組込み自己テスト，単一制御並行可検査性

1. まえがき

VLSI の大規模複雑化に伴い，テストデータ量やテスト時間の増大化が問題となっている．これらの問題を解決する方法として，組込み自己テスト法（Built-In Self-Test：BIST）が重要視されている．

BIST は，test-per-scan 方式と test-per-clock 方式に分類できる．test-per-scan 方式 [1] は，ハードウェアオーバヘッドが小さいが，スキャン操作を行うため，連続したシステムクロックでテスト系列を印加できず，テスト実行時間も長くなる．test-per-clock 方式は，テスト実行時間が短く，テストパターンの連続印加を必要とする遅延故障などのテストにも適用可能である．この方式では，Wunderlich ら [2] が，回路中のすべての閉路が少なくとも二つの BILBO [3] か一つの CBILBO [4] を含むようにするテスト容易化設計法（Desing for Testability. DFT）を提案しているが，ハードウェアオーバヘッドが大きくなる．

筆者らはレジスタ転送レベル（Register Transfer Level：RTL）データパスに対して単一制御可検査性（Single-Control testability）に基づく手法（SC 法）[5] や単一制御並行可検査性（Concurrent Single-Control testability）に基づく手法（CSC 法）[6] を提案した．SC 法，CSC 法及び本論文で提案する手法は，階層 BIST を実現する具体的な方法である．RTL 回路のテスト生成においては，RTL とゲートレベルの二つの階層を利用する階層テスト生成 [7] がある．同様に，BIST 方式においても RTL とゲートレベルの二つの階層を利用した階層 BIST が考えられる．階層 BIST ではテストパターン発生器（Test Pattern Generator：TPG）で発生したパターンをテスト対象モジュールに対して印加し，その応答を応答解析器（Response Analyzer：RA）で観測するための経路の生成を RTL で行う．ゲートレベルでは，テスト対象モジュールに対して故障シミュレーションを行い故障検出率を評価する．SC 法や CSC 法は，故障検出率が高く，Wunderlich らの手法 [2] に比べハードウェアオーバヘッドは小さい．SC 法では組合せ回路要素を一つずつテストするためにテスト実行時間が大きくなるため，CSC 法では，同時に複数の組合せ回路要素をテスト可能にした．しかし，CSC 法ではテストスケジューリングをハードウェアオーバヘッドに基づいて行うため，テスト実行時間が SC 法と比較して改善されない場合もあった．

† 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 of Science, Nara Institute of Science and Technology, 8916-5 Takayama-cho, Ikoma-shi, 630–0192 Japan
* 現在，奈良工業高等専門学校情報工学科

本論文では，SC 法や CSC 法と同等の故障検出率で，ハードウェアオーバヘッドとテスト実行時間を削減するためにデータパスに対して時分割単一制御並行可検査性（Time Division Concurrent Single-Control Testability）を提案する．この可検査性では，テストパターンの印加と応答の解析のために用いる経路の条件を緩和し，ハードウェアオーバヘッドの削減を行う．この可検査性に基づくテスト容易化設計法（TCSC 法）では，各組合せ回路要素単体でのテスト実行時間に基づきテストスケジューリングを行い，テスト実行時間を短縮する．データパスの時分割単一制御並行可検査性を実現するためのアーキテクチャを提案し，RTL 全体の BIST 法も提案する．

2. レジスタ転送レベル回路

本論文で対象とする RTL 回路は，コントローラとデータパスから構成される（図 1）(注 1)．コントローラは有限状態機械，データパスは回路要素と回路要素を接続する信号線で記述される．回路要素は，PI，PO，ラッチ，レジスタ，マルチプレクサ，演算モジュール，観測モジュールに分類される．このうち，マルチプレクサ，演算モジュール，観測モジュールを組合せ回路要素と呼ぶ．各回路要素は端子をもち，それぞれデータ端子，制御端子，観測端子に分類される．データ端子には，回路要素にデータを入力する入力端子と回路要素からデータを出力する出力端子がある．制御端子は，コントローラから制御信号を入力する端子である．観測端子は，コントローラへステータス信号を出力する端子である．信号線は，データ信号線，制御信号線，ステータス信号線に分類される．データ信号線は，二つの回路要素のデータ端子を接続する．制御信号線は，コントローラと制御端子を接続する．ステータス信号線は，観測端子とコントローラを接続する．

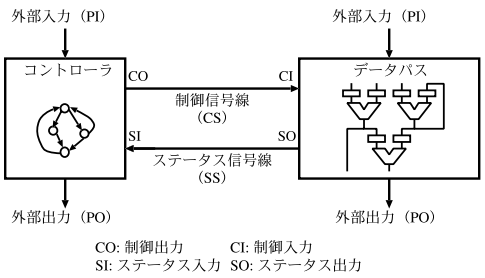


図 1　レジスタ転送レベル回路
Fig. 1　Resister transfer level circuit.

本論文で扱うデータパスは，各回路要素のデータ端子のビット幅がすべて等しく，観測モジュール以外のすべての回路要素は 1 または 2 個の入力端子，1 個の出力端子，たかだか 1 個の制御端子と観測端子をもち，観測モジュールは 1 または 2 個の入力端子，たかだか 1 個の制御端子と観測端子をもつ．また，すべての入力端子は，少なくとも一つの PI から到達可能であり，すべての出力端子は少なくとも一つの PO に到達可能である．

3. 階層 BIST

階層 BIST は，RTL とゲートレベルの二つの階層を利用する BIST 方式である．RTL ではテスト対象となるモジュールに対して，TPG から発生したテストパターンを組合せ回路要素に印加し，その応答を RA で観測するための経路を生成する．このとき，対象とする故障はゲートレベルなので，テスト対象モジュールに対してゲートレベルで故障シミュレーションを行い故障検出率を評価する．

本論文では，RTL におけるテスト容易化設計法を提案する．ゲートレベルに対しては，テストポイント挿入 [10] などの既存の手法を用いて，組合せ回路要素単体に対しては十分な故障検出率が得られるものとする．これらの手法では，データパスに対してはテストプランを生成する．テストプランとは，ゲートレベルの故障シミュレーションで与えたパターンと同じパターンを与えるために，データパス中の各組合せ回路要素に対して，TPG からのテストパターンの伝搬と RA での応答の観測のために与える制御信号線上の信号の時系列である．また，コントローラに対しては，本論文では状態レジスタを CBILBO に置き換えることにより階層 BIST を実現する．

4. 提案手法のアイデア

TCSC 法では，SC 法や CSC 法と異なる新しいアイデアを導入する．

4.1 制御経路と観測経路

TCSC 法では，SC 法や CSC 法と同様，TPG と RA を PI，PO 及び制御入力（CI）にのみ付加する．テスト対象組合せ回路 M に対して，TPG から M の

(注 1)：RTL を人手で設計する場合，データパスとコントローラに分離できない場合がある．しかし通常の RTL 回路は，データパスとコントローラに分離して構成する [8]．例えば，高位合成では，データパスとコントローラを分離した RTL 回路を出力する．

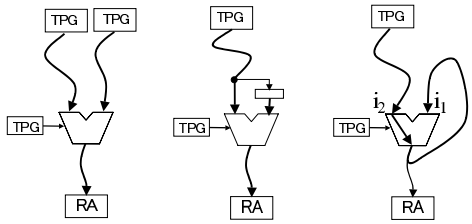


図 2　制御経路と観測経路のタイプ
Fig. 2　Type of control and observation paths.

入力端子への制御経路と M の出力端子から RA までの観測経路を単一の制御信号からなるテストプランで実現する経路として，type1 に加え，type2，type3 の経路も新たに考える（図 2）．三つのタイプの経路によって，各組合せ回路要素の異なる入力端子に TPG で発生した異なるテストパターンを印加することが可能となる．TCSC 法では，三つのタイプの経路を利用して，type1 のみ利用する SC 法，CSC 法に比べ付加する DFT 要素を削減する．

- type1：M の制御経路と観測経路は互いに共通部分をもたない．
- type2：M の異なる入力端子は同じ TPG を始点とする異なる順序深度（制御経路上のレジスタ数）の制御経路をもち，かつ M の観測経路は制御経路と共通部分をもたない．
- type3：M の一方の入力端子 i_1 の制御経路は，M の他方の入力端子 i_2 への制御経路と i_2 から M の出力端子への経路を通り，i_2 はスルー機能を有する．

type1，type2，type3 の経路を用いて実用時間内でテストが可能かを確かめるために各タイプでの故障検出率，パターン数を調べた．表 1 に示す組合せ回路要素に対して，データ入力（DIN）には 32 bit の LFSR を利用してパターンを与え，制御入力（CIN）には 8 bit の LFSR の上位 bit からパターンを与えた．異なる LFSR の特性多項式は異なるように設定し，各 LFSR に対して任意に選んだ五つの seed，特性多項式を用いて故障シミュレーションを行い，検出可能故障に対して故障検出率 100%を達成する平均パターン数とその標準偏差を求めた（表 2）．ただし，すべての入力端子に，TPG で生成されたパターンが印加された時点より計測を行った．表 2 より，必要となるパターン数（#P）は，type1，type2，type3 の順に小さいが，どのタイプでも実用時間内でテストが可能である．また標準偏差（SD）も必要となるパターン数に比べて小さいため，TPG の seed や特性多項式が異なっても必要となるパターン数が大幅に増加しない．

表 1　組合せ回路特性
Table 1　Characteristics of combinational ciruits.

	DIN	DOUT	CIN	
組合せ回路	#DIN	#DOUT	#CIN	bit 幅
MUX	2	1	1	1
加算器	2	1	1	2
減算器	2	1	1	2
乗算器	2	1	1	2
AND	2	1	1	2
OR	2	1	1	2

表 2　パターン数と標準偏差
Table 2　Number of patterns and standard deviations.

	type1		type2		type3	
組合せ回路	#P	SD	#P	SD	# P	SD
MUX	27	3.24	32	2.73	74	1.41
加算器	123	4.85	185	10.15	215	16.64
減算器	167	8.25	298	5.52	325	11.68
乗算器	680	14.20	902	12.63	1870	12.86
AND	100	1.22	158	5.10	198	7.52
OR	102	1.58	162	4.70	201	5.48

表 3　テストスケジューリング例
Table 3　Example of test scheduling.

	TCSC 法			CSC 法 [6]		
セッション	要素 1	要素 2	#P	要素 1	要素 2	#P
1	MULT	M1	27	MULT	M1	680
2	MULT	M2	32	M2	M3	27
3	MULT	M3	74	ADD	—	123
4	MULT	ADD	123			
5	MULT	—	424			
合計			680			826

4.2　テスト スケジューリング

TCSC 法では，複数の組合せ回路要素を同時にテストする．そのため，同時にテストする組合せ回路要素の集合（テストセッション）を決定するテストスケジューリングが必要となる．TCSC 法では，一つの組合せ回路要素を複数のセッションでテストすることによってテスト実行時間を削減する．

[例 1]　表 1 に示す平均パターン数で各組合せ回路要素がテストでき，一つの乗算器 MULT と加算器 ADD，三つの MUX M1，M2，M3 をたかだか二つずつテストする場合のテストスケジューリングを表 3 に示す．TCSC 法において，各セッションはそのセッション中の少なくとも一つの回路要素に十分なテストパターン数が与えられれば終了する．セッション 1 では，M1 に十分なテストパターン数が与えられ，MULT はセッション 2 以降でもテスト対象となっている．CSC 法で

は，各セッション中のすべての組合せ回路要素に十分なパターンを必要とする． □

5. 時分割単一制御並行可検査性

5.1 データパスグラフ

データパスに対してデータパスグラフ $G = (V, A)$ を次の有向グラフとして定義する．

- $V = V_1 \cup V_2$

ここで V_1 はすべての回路要素の集合，V_2 はすべてのデータ端子の集合とする．

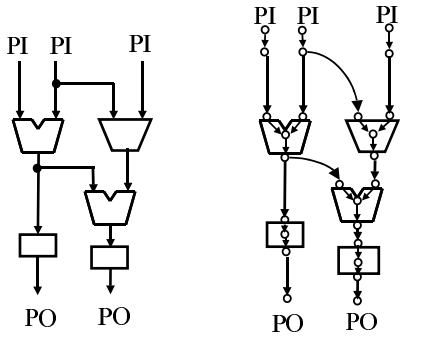
- $A = A_1 \cup A_2 \cup A_3$

ここで A_1 はデータ信号線を表し，$A_1 = \{(x, y) \in V_2 \times V_2 |$ 出力端子 x と入力端子 y がデータ信号線で接続 $\}$ とする．また，A_2，A_3 はそれぞれ，入力端子と回路要素を接続する信号線，回路要素と出力端子を接続する信号線を表す．すなわち，$A_2 = \{(x, u) \in V_2 \times V_1 | x$ は u の入力端子 $\}$，$A_3 = \{(u, x) \in V_1 \times V_2 | x$ は u の出力端子 $\}$ とする．図 3 (a) のデータパスに対するデータパスグラフを図 3 (b) に示す．また，データパスグラフはその対応するデータパスと同一視する．

データパスグラフ $G = (V, A)$ に対し，スルー制約付データパス部分グラフ $G' = (V, A'(\subseteqq A))$ を定義する．ここで，$A' = A_1 \cup A'_2 \cup A_3$ であり，A'_2 は，スルー機能を有する組合せ回路要素の入力端子と回路要素，若しくはレジスタ及びラッチの入力端子と回路要素への対応を表す．スルー機能は，演算モジュールにおいて入力端子と出力端子の間での任意の値の伝搬を保証する機能である．

5.2 時分割単一制御並行可検査性

階層 BIST として type1，type2，type3 を考慮した時分割単一制御並行可検査性を以下のように定義する．



(a) Data path (b) Data path digraph

図 3 データパスとデータパスグラフ
Fig. 3 A data path and its data path digraph.

[定義 1] スルー制約付データパス部分グラフ G' において，テストセッション $\mathcal{M}$ が，以下の条件を満たすとき，$\mathcal{M}$ は時分割単一制御並行可検査であるという．

- 以下の条件を満たす互いに共通部分をもたない木 $T_1, T_2, \ldots, T_i, \ldots, T_m$ が存在．
  - 各木の根は PI である．
  - $\mathcal{M}$ に属する各組合せ回路要素の各入力端子はいずれかの木に属する．
  - $\mathcal{M}$ に属する各組合せ回路要素の異なる入力端子は，異なる木に属する，または根からの順序深度が異なる．
- $T_1, T_2, \ldots, T_i, \ldots, T_m$ に現れる G' の入次数 2 以上の端子に対し，T_i に現れる入力辺以外の入力辺及びその入力辺にのみ到達可能な辺を G' から消去したグラフにおいて，以下の条件を満たす互いに共通部分をもたない経路 $P_1, P_2, \ldots, P_n$ が存在．
  - 各 P_i の始点は $\mathcal{M}$ に含まれる各組合せ回路要素の出力端子であり，終点は PO である． □

テストセッション $\mathcal{M}$ が時分割単一制御並行可検査なら，TPG，RA を PI，PO，CI に配置し，type1，type2，type3 の制御経路，観測経路を用いることにより，$\mathcal{M}$ に属するすべての組合せ回路要素を同時にテストできる．このテストの間，制御経路及び観測経路に現れる制御信号（テストプラン）を固定しておくことができる．つまり，一つのテストセッション $\mathcal{M}$ に対して，一つの制御パターンを与えれば，連続クロックでテスト系列の印加/応答の観測が可能となる．次に，同時にテストする組合せ回路要素の数を k 個としたときのデータパスの可検査性を以下のように定義する．

[定義 2]（データパスの時分割単一制御 k-並行可検査性） 以下の条件を満たす場合，データパスは時分割単一制御 k-並行可検査であると定義する．

- 各テストセッションが時分割単一制御並行可検査である．
- 各テストセッションの要素数はたかだか k である．
- 各組合せ回路要素は，少なくとも一つのセッションに含まれる． □

時分割単一制御 k-並行可検査性では，各組合せ回路要素は複数のテストセッションに属することが可能である．以下では，k の値を特に指定しないときは，時分割単一制御 k-並行可検査性のことを時分割単一制御

並行可検査性と呼ぶ．

6. 時分割単一制御並行可検査性を実現するためのテスト容易化設計法

与えられたデータパスを時分割単一制御並行可検査データパスに設計変更するため DFT を示す．

6.1 問題の定式化

時分割単一制御並行可検査性を実現するための DFT を，次の最適化問題として定式化する．

[定義 3]（時分割単一制御並行可検査 DFT）

- 入力：データパス，並行度 k，テストライブラリ
- 出力：時分割単一制御 k-並行可検査なデータパス，各テストセッション及びセッション長，テストプラン
- 最適化目標：ハードウェアオーバヘッド最小，テスト実行時間最小 □

データパスの DFT 要素は，新しい経路を付加するための MUX（TMUX と呼ぶ），スルー機能を考える．また，テストライブラリは各組合せ回路要素に対する目標故障検出率を達成するために必要となる各タイプの経路に対する平均テストパターン数の見積りである．セッション長は，各テストセッションに要するテスト実行時間である．

6.2 DFT アルゴリズムの概要

単一制御並行可検査 DFT のための発見的アルゴリズムを示す．本アルゴリズムは，以下の 3 段階からなる．

ステージ 1　カットエッジ除去：テストスケジューリングにかかわらず時分割単一制御並行可検査性を満たさない組合せ回路要素に対して DFT を行う．

ステージ 2　テストセッションごとのテストスケジューリング，観測経路及び制御経路のための DFT：以下の，（1）から（4）を各組合せ回路要素に十分な個数のパターンが印加されるまで繰り返す．

（1）テストスケジューリング：テストセッション $\mathcal{M}$ を決定する．

（2）観測 DFT：$\mathcal{M}$ に対する観測経路を求める．

（3）制御 DFT：$\mathcal{M}$ に対する制御経路を求める．

（4）セッション長決定：$\mathcal{M}$ のセッション長を求める．

ステージ 3　制御経路とセッション長の再決定

6.3 カットエッジ除去

必要性の高い DFT 要素を早い段階で付加するために，スケジューリングにかかわらず時分割単一制御並行可検査性を満たさないような組合せ回路要素に対して，以下に定義するカットエッジに基づいた処理を行う．

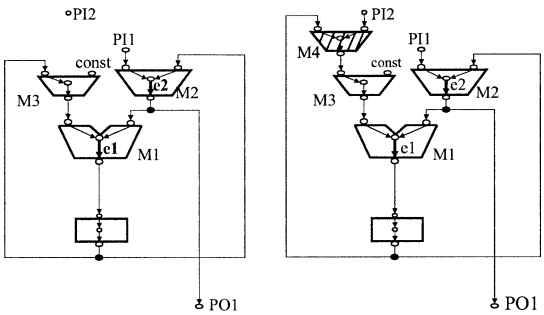
[定義 4]　データパスグラフ G に対して，回路要素と出力端子を接続する辺 e を取り除いて得られるグラフを $G'(e)$ と表す．$G'(e)$ においてどの PI からも到達不能であり，かつどの PO へも到達不能である組合せ回路要素 M が存在する場合，e をカットエッジと呼び，M は e によって支配されるという．□

e によって支配される組合せ回路要素はテストスケジューリングにかかわらず，時分割単一制御並行可検査性を満たすことができない．e を除去するために，e によって支配される組合せ回路要素の入力に任意の順序で TMUX を付加し，e に到達不能な PI が存在する場合はその PI から，なければ任意の PI と TMUX を接続する．これを e がカットエッジでなくなるまで繰り返す．一つの TMUX を加えることにより，複数のカットエッジを除去できる可能性がある．そこで，カットエッジに対して次の半順序関係を定義し，その半順序に従ってカットエッジを処理する．

[定義 5]　データパスグラフ G におけるカットエッジ e に対して，e によって支配されるすべての組合せ回路要素を $D(e)$ と表す．e 及び e' において，$D(e) \subseteqq D(e')$ のとき，またそのときに限り，$e \leqq e'$ とする．□

もし，二つのカットエッジ e_1 と e_2 が $e_1 \leqq e_2$ を満たすなら，e_1 のために加えられる MUX によって，e_2 がカットエッジでなくなることがある．

[例 2]　図 4 (a) では，$e1$ と $e2$ がカットエッジであり，$D(e1) = \{M3\}$，$D(e2) = \{M1, M3\}$ である．したがって，$e1 \leqq e2$ となり，まず，$e1$ に対してカット



(a) Before adding TMUX　(b) After adding TMUX for $e1$

図 4　カットエッジ処理例
Fig. 4　Example of adding TMUX for cut edge.

エッジ除去を行う．$M3 \in D(e1)$ の左入力の直前に TMUX $M4$ を付加し，$M4$ の他方の入力を PI2 に接続する．付加した後のグラフ（図 4 (b)）において，$e1$ だけでなく $e2$ もカットエッジではなくなっている．

□

6.4 テスト スケジューリング及び観測 DFT

一つのテストセッション $\mathcal{M}$ の決定（テスト スケジューリング），$\mathcal{M}$ に対する観測経路を同時に求める．観測経路を制御経路より先に求めるのは，制御経路は異なる組合せ回路要素で共有可能だが，観測経路は共有不能であり，可観測性のための DFT ハードウェア オーバヘッドが支配的になるためである．

組合せ回路要素に対して，既に目標となる故障検出率を達成するパターン数が印加されているものをスケジューリング済み，そうでないものを未スケジューリングであるという（詳細は **6.6**）．

PO，観測端子，またはスケジューリング済みの組合せ回路要素に隣接するすべての未スケジューリング組合せ回路要素を候補として，テストセッション $\mathcal{M}$ 及び $\mathcal{M}$ に含まれる組合せ回路要素の観測経路を求める．観測経路は，データパス中の経路，または組合せ回路要素 M の出力を TMUX を用いて直接 PO に接続する経路を用いて構成する．ただし，MUX の観測経路のためには TMUX による経路は付加しない(注 2)．それぞれの経路に要するハードウェアオーバヘッドをコストとして与え，流量 k（ただし，候補数が $m(< k)$ 個なら流量 m）の最小費用流問題を用いて観測経路を求める．データパスグラフ G から以下のように最小費用流問題の入力となるグラフを構成する．

- 各候補組合せ回路要素への辺をもつダミー頂点を付加し始点とし，各 PO 及び観測端子からの辺をもつダミー頂点を付加し終点とする．
- MUX 以外の各候補組合せ回路要素 M に対し，M の出力端子から各 PO へ辺を付加する．この辺は TMUX の付加を表す．ただし，直接接続する PO 若しくは観測端子が存在する場合，その PO や観測端子への辺の付加は行わない．
- 得られたグラフにおける各辺 (u, v) に対して，容量を 1 とする．コスト $p(u, v)$ を以下のように定義する．

$$
\begin{aligned}
p(u,v) &= cost\_thru(u) && u \text{ が } v \text{ の入力端子.}\\
&= cost\_MUX && (u,v) \text{ は TMUX に対応.}\\
&= 0 && \text{上記以外の場合.}
\end{aligned}
$$

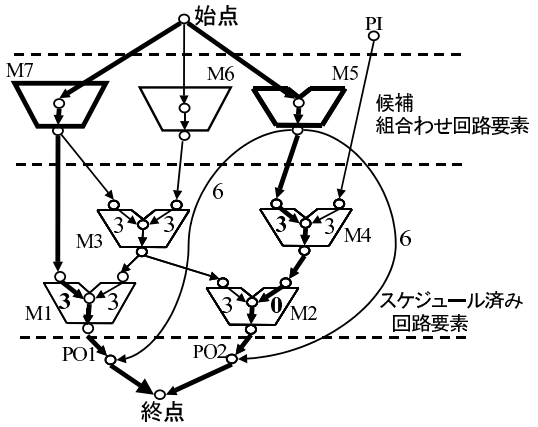


図 5 テスト スケジューリング例（$k = 2$）
Fig. 5 Example of test scheduling ($k = 2$).

（$cost\_thru(u)$ は u から v の出力端子へのスルー機能を付加するためのハードウェアコスト．付加済みの場合は，$cost\_thru(u) = 0$．$cost\_MUX$ は TMUX を付加するためのハードウェアコスト）

最小費用流で求まる経路は，最小のハードウェア オーバヘッドで実現できる観測経路を表す．観測経路のために，スルー機能と TMUX が付加される．

[例 3] 図 5 で，$k = 2$ であり $M1, M2, M3, M4$ がスケジューリング済みの場合を示す（簡単のためにレジスタを省略する）．$M5, M6, M7$ が候補組合せ回路要素である．MUX でない $M5$ の出力端子から，PO1 と PO2 へ TMUX に対する辺をコスト $cost\_MUX(= 6)$ として付加する．$M2, M3, M4$ の入力端子と回路要素を結ぶ辺のうち，スルー機能をもたない辺がコスト 3 となる．それ以外の辺のコストは 0 である．このグラフ上で流量 2 の最小費用流問題を解くと，図中太線の経路がコスト 6 で決定される．このセッションでは $M5$ と $M7$ が選択され，経路上のスルー機能に対応する辺である $M1$ の左入力と $M4$ の左入力にスルー機能が付加されることを意味する． □

6.5 制御 DFT

一つのテストセッション $\mathcal{M}$ に対して，$\mathcal{M}$ に含まれるすべての組合せ回路要素の制御経路を決定する．この手続きは，$\mathcal{M}$ に含まれる組合せ回路要素を一つずつ順に処理する．また，各回路要素に対しては入力端子への経路は一つずつ決める．各回路要素に対する

(注 2)：MUX のテストに TMUX を付加すれば，付加した TMUX のテストも必要となり，テスト実行時間を削減させることなく，ハードウェアオーバヘッドが増加してしまう．

最初のステップでは，ハードウェアコスト最小となる制御経路を一つ（第 1 制御経路と呼ぶ）決定する．次に，他方の制御経路（第 2 制御経路と呼ぶ）を順序深度を考慮して決定する．

第 1 制御経路は，データパスグラフから次に示すようなグラフを構成し，流量 1 の最小費用流問題を解くことで求まる．

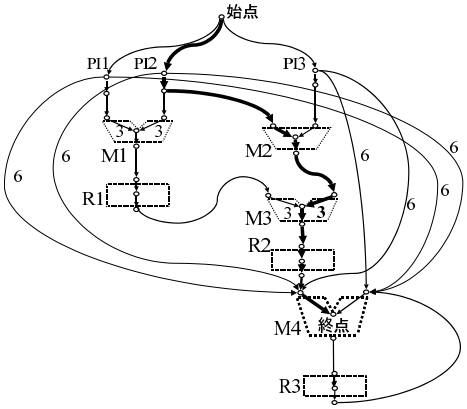
- 各 PI への辺をもつダミー頂点を付加し始点とし，対象となる組合せ回路要素 M を終点とする．
- $\mathcal{M}$ に属する組合せ回路要素の決定済みの制御経路及び観測経路上の組合せ回路要素から，制御経路と観測経路に属さない辺を削除する．(既に選ばれている経路と異なる制御信号を要する経路を選ばない)
- 対象要素 M が MUX 以外の場合，各 PI から対象要素の入力端子へ TMUX に対応する辺を加える．ただし，M に直接接続する PI が存在する場合，その PI からの辺は付加しない．
- 各辺に対し，観測経路を求めるときと同様のコスト，容量を与える．

[例 4]　図 6 (a) で，$M4$ の第 1 制御経路の決定例を示す．各 PI から演算モジュール $M4$ の各入力端子への TMUX に対応する辺を付加し，コストを $cost\_MUX(=6)$ とする．また，$M1, M3$ の各入力端子から回路要素の辺のコストを $cost\_thru(=3)$ とする．このグラフ上で流量 1 の最小費用流問題を解くと，始点から $PI2, M2, M3, R2$ と $M4$ の左入力を通って終点に至る経路が選択される．　□

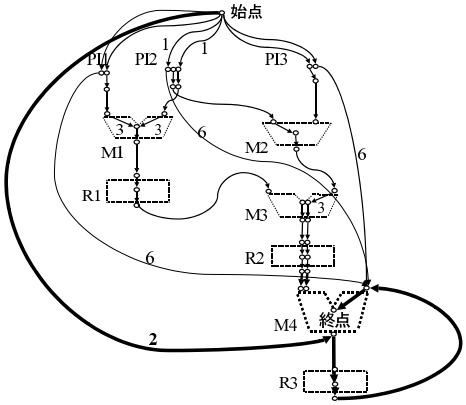
第 2 制御経路も流量 1 の最小費用流問題を解いて求める．ここで用いるグラフは，第 1 制御経路のためグラフと下記の点を除き同様である．

- 始点，終点以外の端子は，その端子を通り始点から終点に至る経路の順序深度の種類分複製し，各端子に順序深度を対応づける．各経路は，順序深度に対応した頂点を通るように接続する．例えば，始点から終点までの順序深度が d の経路は，d に対応する頂点を通る．TMUX に対応する辺は順序深度 0 の経路として考える．
- TMUX に対応する辺のうち，第 1 制御経路で決定した入力端子への辺を削除する．
- 終点へ接続する辺のうち，第 1 制御経路で利用した辺を削除する．
- 始点から，複製された PI 端子への辺のうち，第 1 制御経路と同じ PI で同じ順序深度のものを削除する．始点から，PI 端子への辺のコストは，第 1 制御経路と同じ PI であれば $type2cost$，そうでなければ 0 とする．$type2cost$ は type2 を用いる場合の時間のコストである．
- type3 を表すための辺として，始点から対象要素の出力への辺を付加する．この辺は，始点から対象要素までは第 1 制御経路を利用することを意味する．この辺のコストは，テスト対象回路へのスルー機能の付加に対応する $cost\_thru$ と type3 を用いる場合の時間のコストである $type3cost$ の和である．

[例 5]　図 6 (b) は，$M4$ の第 2 制御経路の決定例を示す．始点から $M4$ までの順序深度を考慮して端子を複製する．例えば，PI3 に対しては，TMUX による経路（順序深度 0）と順序深度が 1 の経路が存在するので，二つの端子を用意する．既に同じセッションに対して決定している制御経路上の組合せ回路要素の制御経路に含まれない辺である $M2$ の右入力からの辺と $M3$



(a) DFT for control path of first port



(b) DFT for control path od second port

図 6　制御経路の決定
Fig. 6　DFT for control paths.

の左入力からの辺を削除する．始点から PI2 に対応する端子へ順序深度 1 に対応する端子以外に接続し，コストを $type2cost(=1)$ とする．また，始点から $M4$ の出力への辺を付加し，コストを $type3cost(=2)$ とする．$M4$ の右入力へ TMUX に対応する辺を付加し，コストは $cost\_MUX(=6)$ とする．このグラフ上で，流量 1 の最小費用流問題を解くと，始点から $M4$ の出力端子と R3 を通り終点に至る経路が決定される．結果として，$M4$ は type3 の経路を利用してテストする． □

最小費用流問題を解く際に与えるコストである $type2cost$，$type3cost$，$cost\_thru$ 及び $cost\_MUX$ を調節することで，得られるデータパスに対して時間優先及び面積優先を選択することができる．

対象要素が MUX である場合，制御経路を生成するために TMUX を付加しないため制御経路を決定できない場合がある．この場合は，MUX をテストセッション $\mathcal{M}$ から削除し，未スケジューリングとする．データパス中のすべての組合せ回路要素はカットエッジ処理によってあるテストセッションでは時分割単一制御並行可検査性を満たすことが保証できるので，このような MUX もいずれスケジューリング済みとなる．

6.6 セッション長決定

セッション長をテストライブラリに基づき決定する．テストセッション $\mathcal{M}_\infty, \mathcal{M}_\in, \ldots, \mathcal{M}_{f-\infty}$ が既に決定し，$\mathcal{M}_f$ のセッション長を決定するとする．l_j $(j = 1, 2, \ldots, s)$ をセッション $\mathcal{M}_|$ のセッション長とする．$\mathcal{M}_{|,\infty}, \mathcal{M}_{|,\in}, \mathcal{M}_{|,\ni}$ をそれぞれ $\mathcal{M}_|$ において type1，type2，type3 の経路をもつ組合せ回路要素の集合とする．組合せ回路要素 M_i に対し，$N_{i,1}$，$N_{i,2}$，$N_{i,3}$ をそれぞれ type1，type2，type3 での M_i に要するテストパターン数とする．組合せ回路要素 $M_i, p = 1, 2, 3$ に対し，$\mathcal{M}^{\rangle,f}_{\checkmark} = \{||\mathcal{M}_\rangle \in \mathcal{M}_{|.\atop\checkmark}, \infty \leqq | \leqq f\}$ とする．このとき，

$$CL_{i,s} = \sum_{j \in \mathcal{M}_1^{i,s}} \frac{l_j}{N_{i,1}} + \sum_{j \in \mathcal{M}_2^{i,s}} \frac{l_j}{N_{i,2}} + \sum_{j \in \mathcal{M}_3^{i,s}} \frac{l_j}{N_{i,3}}$$

を $M_1, \ldots, M_s$ における M_i のテストパターン充足率と呼ぶ．M_i のテストパターン充足率が 1 になれば，M_i には十分なパターン数が印加され，スケジューリング済みと考える．M_s のセッション長は，M_s に属するある組合せ回路要素がスケジューリング済みとなる最小パターン数とする．すなわち，M_s へのテストパターンの印加は少なくとも一つの組合せ回路要素がスケジューリング済みとなる時点までとする．

6.7 制御経路の再決定とセッション長決定

ステージ 2 では，テストセッションごとに制御経路及び観測経路の決定を行った．しかし例えば，あるテストセッションで type2 の制御経路をもつ組合せ回路要素が，後のテストセッションで付加された DFT 要素を用いて type1 の制御経路をもつというように，よりテスト実行時間の小さい経路をもつ可能性がある．そこで，DFT 要素が付加されたデータパスに対して **6.5** で示した手法を用いて再度制御経路を決定する．ただし，この際に新たに DFT 要素に対応する辺の付加は行わない．最後に，各セッションに対して故障シミュレーションを行い，各セッション長を決定する．ここで，ある組合せ回路要素の検出率が所望の値に達しない場合は，リシーディング [9] やテストポイント挿入 [10] などを行い所望の検出率を達成できるようにするか，若しくは与えるパターン数に上限を与えてテストセッション長を決定する．

7. BIST アーキテクチャ

図 7 にレジスタ転送レベル全体に対する階層 BIST アーキテクチャの一例を示す．データパスに対しては，提案した時分割単一制御並行可検査 DFT を行う．コントローラを組合せ回路部と状態レジスタに分離し，状態レジスタを CBILBO に置き換える．入力には TPG の値が印加できるように MUX M5 を，出力には RA で応答を観測できるように MUX M6 を付加する．

このアーキテクチャは **TEST** ピンと **breset** ピンの二つの外部ピンをもつ．TEST ピンが 1 のときに回路に対してテストが実行される．破線部の BIST コ

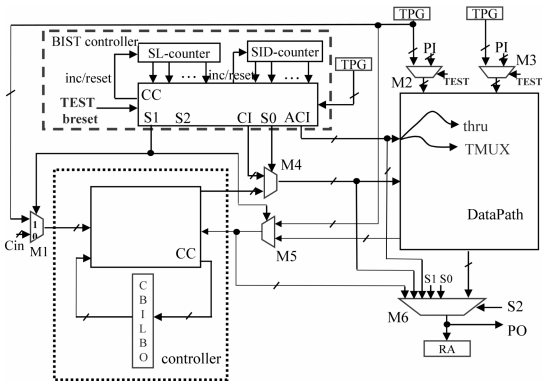


図 7 BIST アーキテクチャ
Fig. 7 BIST architecture.

ントローラは，データパスに付加した DFT 要素の制御信号，及びコントローラとデータパス間に加えられた MUX を制御する信号を出力する．BIST コントローラは二つのカウンタと組合せ回路から構成する．SID カウンタはテストセッション，SL カウンタは現在のテストセッションのテスト実行時間を示す．コントローラのテストは，一つのテストセッションと考える．TPG，RA 及び M1，M2，M3 は提案する BIST 回路の外部で実現可能であるため，提案するアーキテクチャは二つの外部ピンと三つの MUX 及びデータパス部とコントローラ部の DFT で実現可能である．

8. 実験結果

提案手法（TCSC 法）と Wunderlich らの手法 [2] 及び CSC 法 [6] を比較する．使用した RTL 回路は，LWF，Paulin 及び Tseng（表 4）である．#PI,#PO,#Reg,#MUX,#OP はそれぞれ PI 数，PO 数，レジスタ数，MUX 数，演算モジュール数を表す．回路面積の単位は gate equivalent で，論理合成ツールとして Design Compiler（Synopsys）を用いた．

表 5，表 6 に検出可能故障に対して 100%の故障検出率を達成する場合の，ハードウェアオーバヘッドとテスト実行時間を示す．TCSC 法，CSC 法に BIST アーキテクチャを適用した回路を対象とした．C，DP，BIST-C，MUX はそれぞれコントローラ部，データパス部，BIST コントローラ部，MUX M4，M5，M6 のハードウェアオーバヘッドである．面積優先時は，コストを $type2cost < type3cost < cost\_thru < cost\_MUX$ と与え，時間優先時は，演算モジュールのコストを $cost\_thru < cost\_MUX < type2cost < type3cost$ と与えた．提案手法では，並行度 $k = 1$ の場合（表 5）及び並行度 $k = 2$ の場合（表 6）ともにハードウェアオーバヘッドを削減した．これは，type1 だけでなく type2，type3 の経路を利用してテストしたためである．$k = 1$ の場合，時間優先時でもハードウェアオーバヘッドが減少した．これは MUX のテストに対して TMUX を付加しないためにテスト対象の組合せ回路要素数が減少したためである．$k = 2$ の場合，時間優先時では，すべての場合においてテスト実行時間を削減した．これは，提案したテストスケジューリングが有効に働いたためである．面積優先時でも，LWF，Tseng においてはテスト実行時間が減少している．これは，ハードウェアオーバヘッドを最小にするために，付加する DFT 要素がが減少し，同時にテスト実行時間を減少させたためである．

表 4　回路特性
Table 4　Circuit characteristics.

回路	面積（gate）	コントローラ						データパス					
		#PI	#PO	#State	#Status	#Control	面積（*gate*）	#PI	#PO	|bit|	#Reg.	#Mod.	面積（gate）
LWF	6835	1	0	4	0	8	67	64	64	32	5	3	6588
Paulin	36203	1	0	6	0	16	67	64	64	32	7	4	35333
Tseng	23000	3	2	5	0	13	102	96	64	32	6	7	22842

表 5　ハードウェアオーバヘッド及びテスト実行時間（$k = 1$）
Table 5　Hardware overhead and test application time ($k = 1$).

回路	ハードウェアオーバヘッド（%）															テスト実行時間（#clock）		
	CSC 法 [6]					TCSC 法（面積優先）					TCSC 法（時間優先）					CSC 法	TCSC 法	
		C	DP	BIST-C	MUX		C	DP	BIST-C	MUX		C	DP	BIST-C	MUX	[6]	面積優先	時間優先
LWF	38.43	1.20	19.50	13.14	4.59	25.51	1.20	7.33	12.38	4.59	25.51	1.20	7.33	12.38	4.59	540	626	626
Paulin	17.39	0.36	9.65	5.01	1.17	9.36	0.36	3.12	4.71	1.17	10.91	0.36	4.67	4.71	1.17	2247	3136	1955
Tseng	26.65	0.55	17.18	7.23	1.69	21.13	0.55	11.66	7.23	1.69	21.41	0.55	11.94	7.23	1.69	1868	3411	1783

表 6　ハードウェアオーバヘッド及びテスト実行時間（$k = 2$）
Table 6　Hardware overhead and test application time ($k = 2$).

回路	ハードウェアオーバヘッド（%）															テスト実行時間（#clock）		
	CSC 法 [6]					TCSC 法（面積優先）					TCSC 法（時間優先）					CSC 法	TCSC 法	
		C	DP	BIST-C	MUX		C	DP	BIST-C	MUX		C	DP	BIST-C	MUX	[6]	面積優先	時間優先
LWF	43.59	1.20	26.76	12.12	4.59	26.73	1.20	8.56	12.38	4.59	26.73	1.20	8.56	12.38	4.59	509	387	387
Paulin	17.55	0.36	10.97	4.63	1.59	10.14	0.36	3.90	4.71	1.17	12.46	0.36	5.45	4.71	1.17	2108	2705	1201
Tseng	30.53	0.55	20.03	7.44	2.51	22.05	0.55	12.85	6.96	1.69	25.13	0.55	17.62	6.96	1.69	1593	1592	1050

表 7　データパス部のハードウェアオーバヘッド
Table 7　Hardware overhead of data paths.

		Wunderlich らの手法 [2]	CSC 法 [6]		TCSC 法			
					面積優先		時間優先	
			$k=1$	$k=2$	$k=1$	$k=2$	$k=1$	$k=2$
LWF	HW/OH（%）	38.43	21.71	34.66	7.41	8.66	7.41	8.66
	クロック数	106	530	499	616	367	616	367
	FC（%）	100	100	100	100	100	100	100
Paulin	HW/OH（%）	22.49	8.66	14.88	3.14	3.92	4.69	5.47
	クロック数	701	2222	2093	3121	2680	1930	1176
	FC（%）	99.99	99.99	99.99	99.99	99.99	99.99	99.99
Tseng	HW/OH（%）	17.58	17.01	20.00	11.73	12.92	12.01	17.72
	クロック数	657	1843	1568	3386	1567	1758	1025
	FC（%）	99.25	99.99	99.99	99.99	99.99	99.99	99.99

Wunderlich らの手法はデータパスに対する手法であるため，その比較のためにデータパス部のみに対して評価を行った（表 7）．提案手法は，Wunderlich の手法に比べてハードウェアオーバヘッドが小さく，故障検出率（FC）が高いがテスト実行時間が長くなる．これは，提案手法では TPG と RA を回路の PI と PO にのみ置き，組合せ回路要素ごとにテストを行うのに対し，Wunderlich らの手法が回路内部のレジスタを BILBO や CBILBO に変更し閉路を含まない回路に対してテストを行うからである．

9. むすび

本論文では，階層 BIST のためのデータパスの時分割単一制御並行可検査性とその DFT 及び RTL 全体に対する階層 BIST アーキテクチャを提案した．提案手法は test-per-clock 方式に基づき，高い故障検出率，低いハードウェアオーバヘッド，短いテスト実行時間を実現する．今後の課題として提案手法の有効性を明らかにするため，より大規模な回路への提案手法の適用や消費電力を考慮する手法の提案などがある．

謝辞　本研究に際し，多くの貴重な意見を頂いた大阪大学の増澤利光教授，本学の大竹哲史助手，並びにコンピュータ設計学講座の皆様方に深く感謝致します．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，及び，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．また，21 世紀 COE プログラム「ユビキタス統合メディアコンピューティング」の支援を受けている．

文　献

[1] P. Bardell and W.H. McAnney, “Self-testing of multichip logic modules,” Proc. 1982 IEEE Test Conf., pp.200–204, 1979.

[2] A.P. Stoele and H.J. Wunderlich, “Hardware-optimal test register insertion,” IEEE Trans. Comput.-Aided Des. Intergr. Circuits Syst., vol.17, no.6, pp.531–539, 1998.

[3] B. Koenemann, J. Mucha, and G. Zwiehoff, “Built-in logic block observation techniques,” Proc. 1979 IEEE Test Conf., pp.37–41, 1979.

[4] L.T. Wang and E.J. McCluskey, “Concurrent built-in logic block observer (CBILBO),” Proc. Int. Symp. on Circuits and Systems, pp.1054–1057, 1986.

[5] 井筒　稔，和田弘樹，増澤利光，藤原秀雄，“単一制御可検査性に基づくレジスタ転送レベルデータパスの組込み自己テスト容易化設計法，” 信学論（D-I），vol.J84-D-I, no.1, pp.69–77, Jan. 2001.

[6] 山口賢一，和田弘樹，増澤利光，藤原秀雄，“レジスタ転送レベルデータパスの単一制御並行可検査性に基づく組込み自己テスト法，” 信学論（D-I），vol.J85-D-I, no.6, pp.527–537, June 2002.

[7] B.T. Murray and J.H. Hayes, “Hierarchical test generation using pre computed tests for modules,” IEEE Trans. Comput.-Aided Des. Integr. Circuts Syst., vol.16, no.9, pp.1001–1014, 1990.

[8] D.D. Gajski, N.D. Dutt, A.C-H Wu, and S.Y-L Lin, High-level synthesis: introduction to chip and system design, Kluwer Academic, 1992.

[9] B. Koenemann, “LFSR-coded test patterns for scan design,” Proc. European Test Conference (ETC), pp.237–242, 1980.

[10] B. Krishnanmurthy, “A dynamic programming approach to the test point insertion problem,” Proc. ACM/IEEE DAC, pp.695–705, 1987.

（平成 14 年 10 月 4 日受付，15 年 2 月 14 日再受付）

山口　賢一　（正員）

平 11 奈良高専専攻科・電子情報工学専攻了．平 13 奈良先端科学技術大学院大学博士前期課程了．平 15 同大博士後期課程了．博士（工学）．現在テスト容易化設計の研究に従事．平 15 より奈良高専情報工学科助手．IEEE 会員．

井上美智子　（正員）

昭 62 阪大・基礎工・情報卒．平元同大大学院博士前期課程了．同年（株）富士通研究所入社．平 7 阪大大学院博士後期課程了．奈良先端大助手を経て，現在同大助教授．分散アルゴリズム，グラフ理論，テスト容易化設計，高位合成の研究に従事．工博．IEEE，情報処理学会，人工知能学会各会員．

藤原　秀雄　（正員：フェロー）

昭 44 阪大・工・電子卒．昭 49 同大大学院博士課程了．同大・工・電子助手，明治大・工・電子通信助教授，情報科学教授を経て，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．論理設計論，フォールトトレランス，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」（MIT Press）など．大川出版賞，IEEE Computer Society Outstanding Contribution Award，IEEE Computer Society Meritorious Service Award など受賞．情報処理学会会員，IEEE Computer Society Golden Core Member，IEEE Fellow.