NAIST-IS-DT0161035

博士論文

階層 BIST のためのテスト容易化設計に関する研究

山口 賢一

2003 年 2 月 7 日

奈良先端科学技術大学院大学
情報科学研究科 情報処理学専攻

本論文は奈良先端科学技術大学院大学情報科学研究科に
博士 (工学) 授与の要件として提出した博士論文である。

山口 賢一

審査委員：　藤原 秀雄 教授
渡邉 勝正 教授
井上 美智子 助教授

階層 BIST のためのテスト容易化設計に関する研究*

山口 賢一

内容梗概

近年における半導体技術の向上に伴って大規模な論理回路を VLSI として製造することが可能となった．しかし論理回路の大規模化に伴って VLSI の信頼性を保証するテストに要する費用の増加が問題となっている．

この問題を解決する方法として，テスト系列生成と応答解析を VLSI 上で行なう組込み自己テスト法 (Built-In Self-Test. BIST) がある．さらに，回路速度の高速化に伴い，テストの際に実際の回路の動作速度でテストを行なう実動作速度テストが重要になってきている．実速度テスト可能な BIST として，回路内の一部のレジスタをテストパターンの発生や応答の観測を行うレジスタに設計変更するテスト容易化設計 (Design for Testability.DFT) が提案されている．しかし，この DFT に伴うハードウェアオーバヘッドは非常に大きく，実用に適したものではない．

ハードウェアオーバヘッドを削減する手法として，井筒らによって提案された単一制御可検査性 (Single-Control testability) に基づく手法がある．この手法は，階層 BIST に基づいている．RTL 回路のテスト生成においては，RTL とゲートレベルの 2 つの階層を利用する階層テスト生成がある. 同様に，BIST 方式においても RTL とゲートレベルの 2 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 で行なう．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ない故障検出率を評価

*奈良先端科学技術大学院大学 情報科学研究科 情報処理学専攻 博士論文，NAIST-IS-DT0161035, 2003 年 2 月 7 日.

する．この手法では，組合せ回路毎にテストを行なうので故障検出率が高く，レジスタを設計変更しないためハードウェアオーバーヘッドは小さい．しかし，組合せ回路要素を 1 つずつテストするためにテスト実行時間が大きく，対象となる回路がレジスタ転送レベルのデータパス部のみという問題点がある．

そこで，本論文ではこれらの問題点を解消する階層 BIST を実現する手法として，レジスタ転送レベルデータパスに対して単一制御並行可検査性および時分割単一制御並行可検査性というテスト容易性を提案し，これらの可検査性に基づく DFT を提案する．単一制御並行可検査性は，複数の組合せ回路要素を同時にテストするように単一制御可検査性を改善したものである．これによって，単一制御可検査性に基づく手法と同様に高い故障検出率を達成し，テスト実行時間を削減を実現した．また，時分割単一制御並行可検査性に基づく手法では，同時にテストする組合せ回路要素を決定する際に，各組合せ回路要素のテスト実行時間を考慮に入れることによって単一制御並行可検査性に基づく手法よりもさらにテスト実行時間を削減した．また，TPG からのパターンの印加と，応答の解析を行なうための経路の条件を緩和することで，単一制御可検査性や単一制御並行可検査性に比べて故障検出率を低下させることなく，ハードウェアオーバヘッドの削減も実現した．また，データパスの単一制御可検査性，単一制御並行可検査性および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，コントローラを含めたレジスタ転送レベル全体の BIST 法も提案した．これらのテスト容易化設計法では，非常に高い故障検出率を実用的なテスト実行時間で実現することができ，また，テスト容易性を実現するためのハードウェアオーバヘッドも小さい．

キーワード

単一制御並行検査性, 時分割単一制御並行可検査性，テスト容易化設計, 階層 BIST, レジスタ転送レベル

Studies on Design for Testability for Hierarchical BIST*

Kenichi Yamaguchi

Abstract

Along with the improvement in semiconductor technology, the scale of logic circuits is increasing. Also design costs and testing costs of the circuits are increasing.

In order to reduce the testing cost, built-in self test (BIST) method, which have test pattern generators and response analyzers in VLSI circuit, have been expected. Moreover, At-speed testing, which apply test patterns at the operational speed of the circuit, is important. In order to realize both BISTing and at-speed testing, some registers in circuit are enhanced so that they can generate test patterns and/or compact test responce. This design for testability (DFT) method requires too high hardware overhead to implement the large circuit.

In order to reduce hardware overhead, Idutsu et.al proposed DFT methods based on single-control testability of register transfer level(RTL) datapath for hierarchical BIST. A design is made hierarchical BISTed through the following two steps: (1)we genanate paths from TPG to input of combinational module and output of combinationl module to RA at RTL and (2)we apply fault simulation for target module at gate level. The advantages of this DFT method are high fault coverage and low hardware overhead. The drawback is long test application time, because the BIST tests only a single module at a time. Moreover this DFT method handdle only a datapath.

*Doctor's Thesis, Department of Information Processing, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-DT0161035, February 7, 2003.

In this thesis, we propose new testabilities for hirarchical BIST and DFT methods based on concurrent single-control testability and time division concurrent single-control testability of datapath, to remedy the drawback. The concurrent single-control testability is an extension of the single-control testability and has an advantage that test application time becomes shorter because multiple combinational modules can be tested at the same time. We also present DFT method for modifying a data path to a concurrently single-control testable one with low hardware overhead. The time division concurrent single-control testability is relax the conditions of control and observation paths. We also propose a DFT method based on time division concurrent single-control testability to reduce hardware overhead and test application time while keeping high fault coverage. Moreover, We propose a BIST architecture to test whole RTL circuits. These DFT methods can be realized a high fault coverage, low hardware overhead, and practical test application time.

Keywords:

single-control testability, concurrent single-control testability, design-for-testability, hierarchical BIST , register transfar level

関連発表一覧

- 学術論文誌

  1. 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,” 電子情報通信学会論文誌 (DI), Vol.J85-D-I, No.6, pp.527-537, Jun. 2002 .
  2. 山口賢一, 井上美智子，藤原秀雄, ”階層 BIST：低いオーバヘッドを実現する Test-per-clock 方式 BIST,” 電子情報通信学会論文誌, (条件付採録)

- 国際会議 (査読付き)

  1. K. Yamaguchi,H. Wada, T. Masuzawa, and H. Fujiwara, ”A BIST Method Based on Concurrent Single-Control Testability of RTL Data Paths,” Proc. of 10th Asian Test Symposium, pp.313-318, Nov. 2001 .
  2. K. Yamaguchi, M. Inoue, and H. Fujiwara, ”Hierarchical BIST: Test-Per-Clock BIST with low Overhead ,” Proc. of 3rd Workshop on RTL and High level Testing, pp. 42-47 Nov, 2002.

- 研究会報告

  1. 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容易化設計法,” 第 44 回 FTC 研究会，Jan. 2001.
  2. 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について,” 信学技報, (CPSY,FTS 2001-3), Apl. 2001 .

3. 山口賢一, 井上美智子，藤原秀雄, ”階層 BIST：低いオーバヘッドを実現する Test-per-clock BIST,” 第 47 回 FTC 研究会，Jul. 2001.
4. 山口賢一, 井上美智子，藤原秀雄, ”階層 BIST：低オーバヘッドを実現する Test-per-clock BIST,” 信学技報, (VLD2002-87, ICD2002-131, DC2002-39),Nov. 2002 .

目 次

1 序論 **1**

2 組込み自己テスト **4**

2.1. テストの評価尺度 4

2.2. レジスタ転送レベル回路 4

2.3. 組込み自己テスト方式 5

2.4. Wunderlich らの手法 8

2.5. 階層 BIST に基づく手法 9

2.5.1 井筒らの手法 9

3 諸定義 **11**

3.1. データパスグラフ 11

3.1.1 データパスグラフ 11

3.2. 単一制御可検査性 13

3.3. 単一制御並行可検査性 14

3.4. 時分割単一制御並行可検査性 16

3.4.1 時分割単一制御並行可検査性のアイデア 16

3.4.2 時分割単一制御並行可検査性 18

4 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 **21**

4.1. 問題の定式化 21

4.2. DFT アルゴリズムの概要 21

4.3. 単一制御可検査 DFT 23

4.4. テストスケジューリング及び観測経路に関する DFT 25

4.5. 制御経路に関する DFT 27

4.6. テストマルチプレクサに関する考察 27

4.7. 適用例 29
4.8. 実験結果 31
4.9. まとめ 34

5 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 35
5.1. 問題の定式化 35
5.2. DFT アルゴリズムの概要 35
5.3. カットエッジ除去 36
5.4. テストスケジューリング及び観測経路に関する DFT 38
5.5. 制御経路に関する DFT 41
5.6. テストセッション長決定 45
5.7. 制御経路の再決定とセッション長決定 45
5.8. BIST アーキテクチャ 46
5.9. 実験結果 48
5.10. まとめ 49

6 結論 52

謝辞 54

参考文献 55

図 目 次

2.1 レジスタ転送レベル回路 5
2.2 LFSR の概念図 6
2.3 BIST の概念図 6
2.4 組み込み自己テストの種類 8

3.1 データパスとデータパスグラフ 12
3.2 制御経路と観測経路のタイプ 17

4.1 単一制御可検査 DFT 24
4.2 $(k = 2)$ スケジューリング例 26
4.3 制御経路に関する DFT 28
4.4 Paulin 元回路 29
4.5 DFT 適用後 (k=2) 30

5.1 カットエッジ処理例 37
5.2 $(k = 2)$ テストスケジューリング例 39
5.3 制御経路の決定 44
5.4 BIST アーキテクチャ 46
5.5 Test-per-scan アーキテクチャ 50

表 目 次

3.1 組合せ回路特性 17
3.2 パターン数と標準偏差 20
3.3 テストスケジューリング例 20

4.1 Paulin に対するステージ 2 の適用過程 ($k = 2$) 30
4.2 回路特性 31
4.3 付加 TMUX およびスルー機能 31
4.4 ハードウェア・オーバヘッド 33
4.5 テストセッション 33

5.1 回路特性 48
5.2 ハードウェアオーバヘッドおよびテスト実行時間 ($k = 1$) 51
5.3 ハードウェアオーバヘッドおよびテスト実行時間 ($k = 2$) 51
5.4 データパス部のオーバヘッド 51
5.5 test-per-scan 方式との比較 51

第 1 章
序論

VLSI の大規模化，複雑化に伴い，論理回路のテストはますます重要な問題となってきている．論理回路のテストとは論理回路上の故障の有無を調べることを指す．

様々なテスト方式が提案されているが，現状では故障の有無で出力が異なるような入力系列 (テスト系列) を用いたテストが広く用いられている．これまでに論理回路の故障をテストする系列を生成するテスト生成アルゴリズム，およびその高速化に関する研究が多数報告されているが，テスト生成の問題は一般に NP 完全であることが知られており，大規模な回路に対して実用的な計算時間で全ての故障のテスト系列を求めることはできない．さらに，大規模な回路のテスト系列長は非常に長く，それらを記憶するためには大きな記憶領域が必要になる．そのため論理回路のテスト費用は増大する傾向にある．

テスト費用の問題を解決するために，テスト生成を行わずにテスト系列をハードウェアで発生する方法が考えられる．その一つの方法としてテスト系列生成，応答解析を VLSI 上で行う組込み自己テスト（Built-In Self-Test．以下，BIST）[2] があり，その重要性がますます高まっている．

BIST を実現するための簡単な手法として，テスト対象回路の外部入力 (Primary Input.PI)，制御入力 (Control Input.CI) にテストパターン発生器 (Test Pattern Generator.TPG)，外部出力 (Primary Outoput.PO) に，応答解析器 (Response Analyzer.RA) を付加する．しかし，テスト対象回路に閉路が含まれている場合には，このような手法では高い故障検出率を得ることができない．そのため，高い故障検出率を得るために，回路内部にテストのためのハードウェアを付加する方法が数多く提案されている．このように，回路の設計をテスト容易となるように変更することを**テスト容易化設計** (Design For Testability) と呼ぶ．

BIST は，test per scan 方式と test per clock 方式に分類できる．test per scan

方式では，回路中の（一部の）レジスタをスキャンレジスタに変更し，スキャン操作により，TPG で生成したテストパターンをスキャンレジスタにシフトインし，スキャンレジスタに格納された応答を RA にシフトアウトする．test per scan 方式では，ハードウェアオーバヘッドは小さいが，スキャン操作によりテスト系列を 1 ビットずつシフトインするので，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い．

一方，test per clock 方式では，回路中の（一部の）レジスタをテスト系列生成器，応答解析器に変更する．このようなテストレジスタとしては，BILBO（Built-In Logic Block Observer)[9]，CBILBO（Concurrent BILBO）[10] が用いられる．test per clock 方式では，連続クロックでテスト系列の生成／印加，応答の解析が可能であり，実動作速度テスト（at-speed test）が可能である．このため，test per scan 方式に比べて，実動作速度でのテスト実行時間が短く，さらに実動作速度でのテスト系列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．しかし，一般に，test per scan 方式に比べ，ハードウェア・オーバーヘッドが大きくなる．

test per clock 方式の BIST として，Wunderlich ら [11] は，回路中のすべての閉路が少なくとも 2 つの BILBO か 1 つの CBILBO を含むようにするテスト容易化設計法を提案している．この手法では, 設計変更するレジスタの数を少なくしてハードウェア・オーバヘッドを低く抑えながら, 高い故障検出率を得ようとしたものである．しかし，テスト対象回路にレジスタが含まれるような場合，高い故障検出率が得られない．

また, 井筒らは，単一制御可検査性というテスト容易性を提案し,test per clock 方式の BIST として，単一制御可検査性に基づく方法とそのテスト容易化設計法 (SC 法)[15] を提案している．この手法では，内部レジスタを BILBO や CBILBO に変更せず，TPG，RA は，それぞれテスト対象回路の外部入力，外部出力のみに付加する．そして，階層 BIST に基づき，データパス中の組合せ回路要素（演算器，マルチプレクサなど）ごとにテストを行う．RTL 回路のテスト生成においては，RTL とゲートレベルの 2 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 2 つの階層を利用した階層 BIST が考えられる．階層 BIST では 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 RA で観測するための経路の生成を RTL で行なう．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ない故障検出率を評価する．

SC 法では,Wunderlich ら [11] の手法と同等の故障検出率を得るために必要なハードウェア・オーバーヘッドは小さいが, 組合せ回路要素を 1 つずつテストするために, テスト実行時間が大きくなる．

そこで, 本論文では, 階層 BIST に基づき，SC 法に比べてテスト実行時間の短縮を目標とする．そのために，複数の組合せ回路要素を同時にテスト可能なデータパスとして単一制御並行可検査性を定義し，それに基づくテスト容易化設計法 (CSC 法) を提案する．

また，ハードウェアオーバヘッドをさらに削減するために，テストパターンの伝搬および応答の観測のために用いる経路の条件を CSC 法より緩和した時分割単一制御並行可検査性を定義し，それに基づくテスト容易化設計法 (TCSC 法) も提案した．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くテストスケジューリングを行ない，さらにテスト実行時間を短縮する．

また，データパスだけではなくコントローラも含んだレジスタ転送レベル全体に対して階層 BIST を実現するための BIST アーキテクチャを提案する．

第 2 章
組込み自己テスト

本章では，本論文で扱う論理回路のテストの評価尺度について述べる．次に組込み自己テスト方式およびその従来手法について述べる．

2.1. テストの評価尺度

論理回路のテストは, 回路の外部入力にテスト系列を印加し，その外部出力に現れる応答を期待値 (正常な回路からの出力) と比較することにより被テスト回路に故障が存在するかどうかを調べる．

テスト系列の質を評価する尺度としては**故障検出率**と**テスト実行時間**などがある．故障検出率は，印加するテスト系列によって，テスト対象故障のうちどれだけの故障が検出されるかを示す比率のことである．また，テスト実行時間はテスト系列の印加に要する時間のことである

2.2. レジスタ転送レベル回路

一般に RTL 回路は，コントローラとデータパスから構成される (図 2.1)*．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持ち，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

*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．通常 RTL 回路は，データパスとコントローラから構成される [19]．例えば，高位合成では，データパスとコントローラからなる RTL 回路を出力する．

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，2 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．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全て等しく，観測モジュール以外の全ての回路要素は 1 または 2 個の入力端子，1 個の出力端子，高々1 個の制御端子と観測端子を持ち，観測モジュールは 1 または 2 個の入力端子，高々1 個の制御端子と観測端子を持つ．また，すべての入力端子は，少なくとも 1 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 1 つの PO に到達可能である．

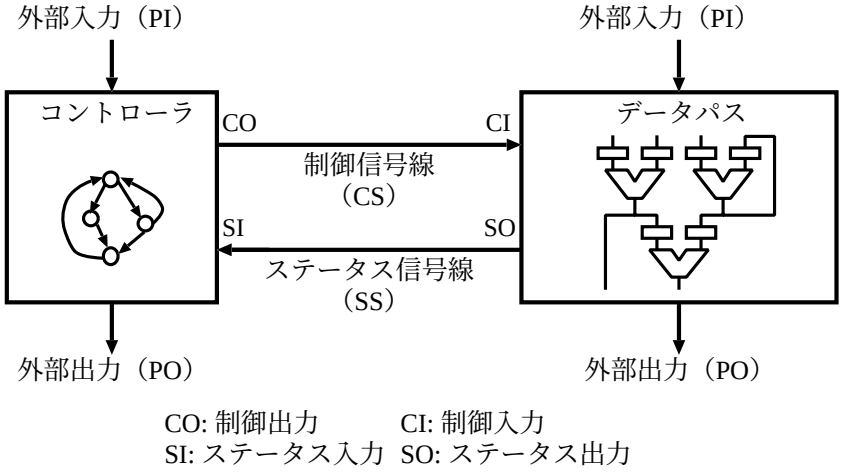


図 2.1 レジスタ転送レベル回路

2.3. 組込み自己テスト方式

外部のテスタを用いてテスト対象回路のテストを行う方式を**外部テスト方式**と呼ぶ．外部テスト方式では，生成しておいたテスト系列を外部テスタからテスト対象回路に与え，その応答を外部テスタで観測して故障の検出を行う．これに対

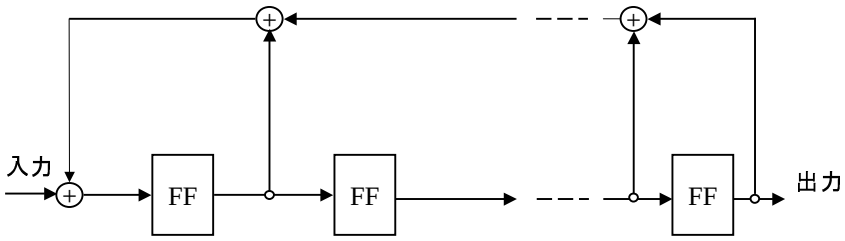


図 2.2 LFSR の概念図

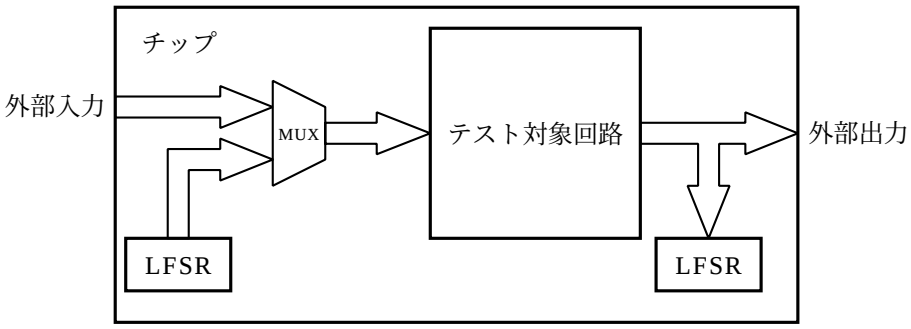


図 2.3 BIST の概念図

して，テストを行うための回路をテスト対象回路の内部に組み込んでテストを行う方式を**組込み自己テスト方式** (Built-In Self Test：以下 BIST) と呼ぶ．BIST 方式では，外部のテスタを用いずに，テストパタン生成器や応答解析器をチップ内に組み込むことにより回路のテストを行う．

一般に BIST では, 擬似ランダムパターンを発生する線型フィードバックシフトレジスタ (Linear Feedback Shift Register：以下 LFSR) をテストパターン生成器としてテスト対象回路の外部入力部に，テスト対象回路の出力を圧縮するための応答解析器としての LFSR を外部出力におく．

図 2.2に LFSR を, 図 2.3に BIST の概念図を示す.

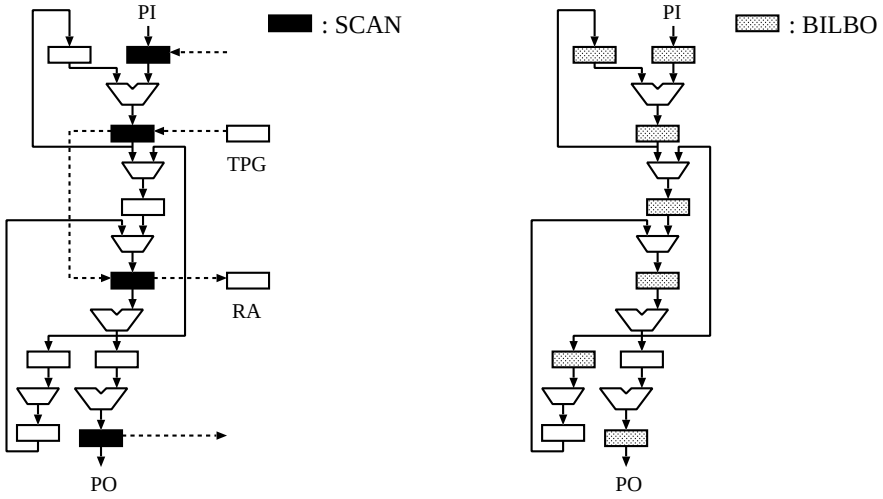
図 2.3の外部入力のマルチプレクサは, 通常動作とテスト動作との切り替えを行うための回路である.

LFSR は D フリップフロップのような遅延素子を直列に接続し，排他的論理和によるフィードバック回路を持つ単純な回路である．遅延素子の段数に応じた適切なフィードバック回路を選択することで，LFSR は全て 0 以外のとり得る全てのパターンを出力することができる．したがって，LFSR を用いて**擬似ランダムテスト** (Pseudo-random Test) や疑似全数テスト (Pseudo-exhaustive Test) を行うことができる.

テスト対象回路の外部入力にテスト系列を印加したときの出力系列を出力応答と呼ぶ．LFSR による出力応答の圧縮は，テスト対象回路の外部出力を LFSR の入力端子に接続することによって行う.

BIST は，図 2.4のように test per scan 方式と test per clock 方式に分類できる．test per scan 方式では，回路中の一部あるいは全てのレジスタをスキャンレジスタに設計変更する．スキャン操作により，テストパターン発生器で生成したテストパターンをスキャンレジスタにシフトし，スキャンレジスタに格納された応答を応答解析器にシフトする．test per scan 方式では，スキャン操作によりテストパターンを印加するので，テストパターン発生器で生成したパターンを連続したシステムクロックで印加できず，テスト実行時間も長い.

これに対して test per clock 方式は，回路中の一部あるいは全てのレジスタをテストパターン発生器，応答解析器に設計変更する．このようなテストレジスタとしては，BILBO[9]，CBILBO[10] が用いられる．BILBO は「通常のレジスタ」「テストパターン発生器 (TPG)」「応答解析器 (RA)」「シフトレジスタ」のうちのどれか 1 つのモードを実行することができる．モードの指定は制御信号線により行われる．CBILBO は「テストパターン発生器 (TPG)」と「応答解析器 (RA)」の 2 つのモードを同時に実行することができる．つまり,「テストパターン発生器 (TPG)」としてパターンを出力端子から出力すると同時に，入力端子に印加される応答を「応答解析器 (RA)」として処理することができる．一般に，test per clock 方式では test per scan 方式に比べ，ハードウェア・オーバーヘッドが大きくなる傾向があるしかし，連続クロックでテストパターンの生成／印加，応答の解析が可能であり，実時間テストが可能である．このため，テスト実行時間が短



(a)test per scan 方式　　(b)test per clock 方式

図 2.4 組み込み自己テストの種類

く，さらに，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．

2.4. Wunderlich らの手法

Wunderlich ら [11] は,test per clock 方式の BIST を提案している．この手法は，回路を無閉路な回路とみなせるように，回路中の全ての閉路が少くとも 2 つの BILBO[9] か 1 つの CBILBO[10] を含むように回路を設計変更する．そうすることで, 回路全体をいくつかの無閉路な部分回路に分割でき, 各部分回路は同時に実行可能なテストパターン発生器と応答解析器をそれぞれ入力, 出力部にもつことになる．つまり, 回路中の全ての閉路が TPG と RA を含むことが可能となり，比較的高い故障検出率が得られる．そして，この条件を満たすハードウェア・オーバーヘッド最小のテスト容易化設計として，閉路の共有度が高いレジスタを優先して BILBO ／ CBILBO に変更する方法を示している．

この手法では，TPG と RA の間の無閉路とみなした部分回路の回路規模が大

きい場合には，十分高い故障検出率を得ることができない．

2.5. 階層 BIST に基づく手法

RTL 回路のテスト生成においては，RTL とゲートレベルの 2 つの階層を利用する階層テスト生成 [12] がある．同様に，BIST 方式においても RTL とゲートレベルの 2 つの階層を利用した階層 BIST が考えられる．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な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 [18] 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．

2.5.1 井筒らの手法

井筒らの手法 (SC 法) では，データパスに対して単一制御可検査性を定義し, 回路中のレジスタを BILBO や CBILBO に設計変更することなく，TPG，RA をそれぞれ, 回路の外部入力と外部出力のみに付加する．これにより，Wunderlich ら [11] により提案された手法よりもハードウェアオーバーヘッドが低くなる．また，階層 BIST に基づいて，組合せ回路要素毎にテストするため，高い故障検出率を達成することができる.．つまり, 予め各組合せ回路要素 M に対して，外部入力を始点とし，M を終点とする経路 (**制御経路**) および M を始点として外部出力を終点とする経路 (**観測経路**) を決定しておく．そして，これらの経路に沿ってテス

トパターンを TPG から M まで伝搬し，応答を M から RA まで伝搬する．連続クロックでテストパターンの生成／印加，応答の解析を可能にするには，これらの経路が共通部分を持たないことが必要である．この手法では，回路内の各組合せ要素を 1 つずつテストするためにテスト実行時間が大きくなってしまう．

第 3 章
諸定義

3.1. データパスグラフ

3.1.1 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．

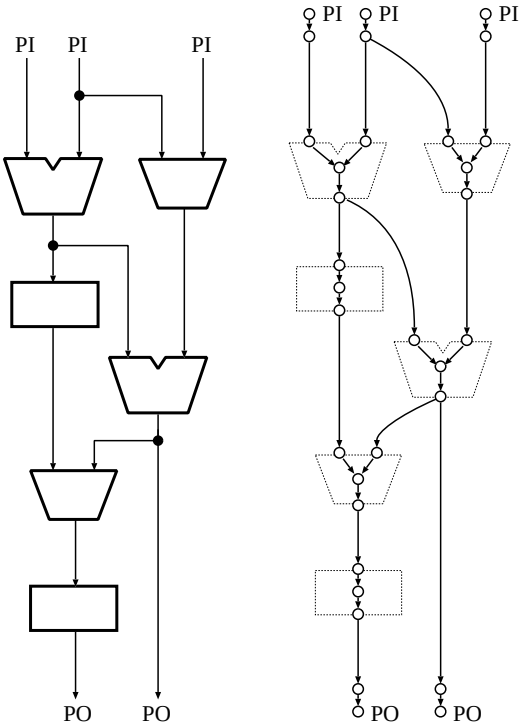
- $V = V_1 \cup V_2$

  ここで V_1 はデータパス中のすべての回路要素の集合，V_2 はデータパス中のすべてのデータ端子の集合とする．以下では，データパスグラフ G の頂点集合 V_1, V_2 をそれぞれ $G.V_1, G.V_2$, と表す．また，すべての組合せ回路要素の頂点集合を $V_C(\subseteq V_1)$ として表わす．

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

  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わす．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．以下では，データパスグラフ G の辺集合 A_1, A_2, A_3 をそれぞれ $G.A_1, G.A_2, G.A_3$ として表す．

図 3.1(a) のデータパスに対するデータパスグラフを図 3.1(b) に示す．本論文で対象とするデータパスは，そのデータパスグラフにおいてすべての入力端子は，少なくとも 1 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 1 つの PO に到達可能であるものとする．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．



(a)data path　　(b)data path digraph

図 3.1 データパスとデータパスグラフ

またデータパスグラフの拡張として, 演算器にスルー機能が実現されている場合のデータパスグラフについても考える．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．

そこで，データパスグラフ $G = (V, A)$ の部分グラフとして, スルー制約付データパス部分グラフ $G' = (V, A'(\subseteq A))$ を定義する．$A' = A_1 \cup A'_2 \cup A_3$ のように定義され,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, もしくはレジスタおよびラッチの入力端子と回路要素への対応を表す.

3.2. 単一制御可検査性

SC 法 [15] は，階層 BIST のためにレジスタ転送レベルデータパスの単一制御可検査性を提案し，与えられたデータパスを単一制御可検査にするための DFT 手法を提案した．データパスの単一制御可検査性を次のように定義している．

[**定義 1**] データパス中の任意の組合せ回路要素 M において，以下の条件を満たす 3 つの共通部分を持たない経路 P_1, P_2, P_3 が存在するなら，データパスは単一制御可検査であるという.

- 任意の値がそれぞれ P_1, P_2, P_3 を通って伝搬可能
- P_1, P_2 は TPG を始点とし，M*の入力端子を終点とする.
- P_3 は，M の出力端子を始点とし，RA を終点とする. ■

P_1, P_2 は M の制御経路，P_3 は M の観測経路である.

単一制御可検査データパスにおいて TPG と RA をそれぞれ PI と PO に置くことにより，組合せ回路要素 M に対して，制御経路を用いて PI から連続したテスト系列を印加し，観測経路を用いて M の応答を連続して PO で観測できる．ほとんどの組合せ回路要素 (加算器，減算器，乗算器，シフタ，マルチプレクサなど) はランダムパターンでテスト可能である．また，比較器などランダムパターンでテスト可能でない組合せ回路要素に関しては，テストポイント挿入手法によって

*P_1 と P_2 はそれぞれ異なる TPG を始点とする

ランダムパターンでテスト可能にできる [13]．従って，単一制御可検査データパスに対しては，BIST によって高い故障検出率を得ることができる．

制御経路，観測経路上の，演算モジュールやマルチプレクサは，この経路上の入力端子の値が出力端子に伝搬するように制御する†．これらを制御する信号 (テストプラン) は，M のテストの間固定しておけばよい．このように，各組合せ回路要素に対して，1 つのテストプランで十分なので，この性質のことを単一制御可検査性と呼ぶ．

3.3. 単一制御並行可検査性

ここでは，複数の組合せ回路要素を同時にテスト可能なように単一制御可検査性を拡張した単一制御並行可検査性を定義する．

[**定義 2**] $\mathcal{M}$ をデータパス中の組合せ回路要素の集合とする．スルー制約付データパス部分グラフ G' において組合せ回路要素の集合 $\mathcal{M}$ が以下の条件を満たすとき,$\mathcal{M}$ は単一制御並行可検査であるという．

- 以下の条件を満たす, 互いに共通部分を持たない木 $T_1, T_2, ..., T_m$ が存在．(各 T_i を**制御木**と呼ぶ．)
  - それぞれの木は PI を根とする．
  - 各組合せ回路要素 $M_i(\in \mathcal{M})$ の各入力端子は木の葉である．
  - 各組合せ回路要素 $M_i(\in \mathcal{M})$ の 2 つの入力端子は異なる木に属する．
- $G' - (T_1 \cup T_2 \cup ... \cup T_m)$ において，以下の条件を満たす，互いに共通部分を持たない経路 $P_1, P_2, ..., P_n$ が存在．(各 P_i を**観測経路**と呼ぶ．)
  - 各組合せ回路要素 $M_i(\in \mathcal{M})$ の出力端子が始点である．
  - 各経路の終点は PO である． ■

†演算モジュールが入力端子の値を出力端子に伝搬する機能 (スルー機能) を持たない場合は，テスト容易化設計でスルー機能を付加し，それを利用する．

$\mathcal{M}$ が単一制御並行可検査なら，TPG，RA をそれぞれ PI，PO に配置し，テスト系列，応答を制御木，観測経路を用いて伝搬することにより，$\mathcal{M}$ 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．さらに，単一制御可検査性と同様に，このテストの間, 制御木および観測経路に現れる演算モジュールおよびマルチプレクサの制御信号を固定しておけばよい．つまり, $\mathcal{M}$ に対して,1 つの制御パターンで十分である．

データパスのすべての組合せ回路要素を，単一制御並行可検査な組合せ回路要素の集合 $\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_l$ に分割できるとする．このとき，各 $\mathcal{M}_i$ に属する組合せ回路要素を同時にテストすれば，データパス全体のテスト実行時間は，全体の組合せ回路要素数ではなく，l に比例すると考えられる．このとき，各 $\mathcal{M}_i$ に属する組合せ回路要素の集合を**テストセッション**とよぶ．

データパス全体のテスト実行時間を短縮するには，テストセッション数が小さくなるように，すべての組合せ回路要素を単一制御並行可検査な集合に分割すればよい．一方，同時にテストされる組合せ回路は，相異なる応答解析器を利用するので，応答解析器が k 個しかなければ，高々k 個の組合せ回路要素が単一並行可検査となりうる．これらのことを考慮して，データパスの並行可検査性を以下のように定義する．

[定義 3](データパスの単一制御 $k-$ 並行可検査性)

スルー制約付データパス部分グラフ G' において，以下の条件を満たす組合せ回路要素 V_C 上の分割 $P(V_C)$ が存在するなら, データパス DP が単一制御 $k-$ 並行可検査であると定義する．

- $\forall S \in P(V_C)$ に対して,S(組合せ回路要素の集合) が単一制御並行可検査
- $\forall S \in P(V_C)$ に対して,$|S| \leq k$
- $|P(V_C)| = \lceil |V_C|/k \rceil$ ■

単一制御 $k-$ 並行可検査データパスでは，組合せ回路要素集合 V_C を，$\lceil |V_C|/k \rceil$ 個の部分集合（ただし，各部分集合は高々k 個の組合せ回路要素を含む）に分割し，各部分集合に属する組合せ回路要素を同時にテストできる．従って，テストセッション数が単一制御可検査データパスの場合に比べてほぼ $1/k$ に減少する．単一制御 $k-$ 並行可検査性は, 単一制御可検査性の一般化であり,k=1 の場合, 単

一制御 k-並行可検査性は単一制御可検査性となる．以下では,k の値を特に指定しないときは, 単一制御 $k-$ 並行可検査性のことを単一制御並行可検査性とよぶ.

3.4. 時分割単一制御並行可検査性

3.4.1 時分割単一制御並行可検査性のアイデア

本節では，時分割単一制御並行可検査性を定義する前に，時分割単一制御並行可検査性に基づく手法 (TCSC 法) で導入する SC 法や CSC 法と異なる新しいアイデアを説明する.

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 および 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)．3 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，3 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持たない.
- 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) の制御経路を持ち，かつ M の観測経路は制御経路と共通部分を持たない.
- 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 3.1 に示す組合せ回路要素に対して，データ入力 (DIN) には 32bit の LFSR を利用してパターンを与え，制

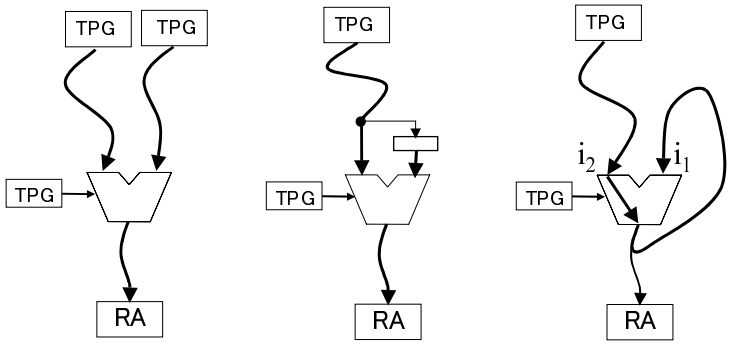


図 3.2 制御経路と観測経路のタイプ

御入力 (CIN) には 8bit 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 5 つの seed，特性多項式を用いて故障シミュレーションを行ない，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 3.2)．ただし，全ての入力端子に，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なった．

表 3.1 組合せ回路特性

	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 3.2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．

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，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，1 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 3.2 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，1 つの乗算器 MULT と加算器 ADD,3 つの MUX M1,M2,M3 を高々2 つずつテストする場合のテストスケジューリングを表 3.3 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 1 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，各セッション中の全ての組合せ回路要素に十分なパターンを必要とする． ■

3.4.2 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．

[**定義 4**] 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持たない木 $T_1, T_2, ..., T_i, ..., T_m$ 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．
- $T_1, T_2, ..., T_i, ..., T_m$ 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およ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持たない経路 $P_1, P_2, ..., P_n$ 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 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，$\mathcal{M}$ に属する全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, 制御経路およ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, 1 つのテストセッション $\mathcal{M}$ に対して,1 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 個とした時のデータパスの可検査性を以下のように定義する．

[**定義 5**](**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性である．
- 各テストセッションの要素数は高々k である.
- 各組合せ回路要素は，少なくとも 1 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よぶ.

表 3.2 パターン数と標準偏差

	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 3.3 テストスケジューリング例

	TCSC 法			CSC 法		
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

第 4 章
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

4.1. 問題の定式化

与えられたデータパスにおいて, 同時にテストする組合せ回路要素に対して, 共通部分を持たない制御木および観測経路が存在しない場合, 単一制御並行可検査にするためには, データパスに新しい経路を付加しなければならない．提案する DFT では，この経路付加は，マルチプレクサ (TMUX と呼ぶ) を用いて実現する．また，制御経路，観測経路に演算モジュールが現れる場合，任意の値を伝搬可能とするために，この演算モジュールにスルー機能を付加する．そこで，単一制御並行可検査のための DFT を，次の最適化問題として定式化する．

[定義 6](単一制御並行可検査 DFT)
単一制御並行可検査のための DFT を次の最適化問題として定義する．

- 入力：データパス, 並行度 k
- 出力：単一制御 k– 並行可検査なデータパス
- 最適化目標：付加する DFT 要素 (スルー機能，TMUX) のハードウェア量最小化. ■

4.2. DFT アルゴリズムの概要

単一制御並行可検査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 2 段階からなる．

ステージ 1 与えられたデータパスを単一制御可検査に設計変更する．この設計変更には，SC 法 [15] を改良したものを用いる．

ステージ 2 ステージ 1 で得られたデータパスを単一制御 $k-$ 並行可検査なデータパスに設計変更する．ここでは，全ての組合せ回路要素に対して制御経路と観測経路を決定するまで，以下の 3 つのステップを繰り返す．

1. スケジューリング: 同時にテストする組合せ回路要素として，未スケジューリング組合せ回路要素から k 個選択する．
2. 観測経路に関する DFT: (1) で選ばれた k 個の組合せ回路要素に対して，それぞれ観測経路を決定する．ここで，TMUX を付加した場合，付加した TMUX は未スケジューリング組合せ回路要素となり，後のスケジューリングの対象となる．
3. 制御経路に関する DFT: (1) で選ばれた k 個の組合せ回路要素に対して，それぞれ制御経路を決定する．(2) と同様に，付加した TMUX は未スケジューリング組合せ回路要素であり，後にスケジューリング対象となる．

4.3. 単一制御可検査 DFT

このステージの目的は，与えられたデータパスを最小のハードウェア・オーバヘッドで単一制御可検査にすることであり，SC 法を改良した手法を用いる．SC 法は，組合せ回路要素 1 つずつを順に処理し，互いに共通部分を持たない 2 つの制御経路と 1 つの観測経路を持つように DFT 要素を付加する．先に加えた DFT 要素は後の組合せ回路要素で再利用できるので，全体のハードウェア・オーバヘッドを削減するには，必要性の高い DFT 要素を早い段階で付加することが重要である．そこで，本論文の手法では，SC 法を適用する前に，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 7**] データパスグラフ G に対して,$e \in G.A_3$ を取り除いて得られるグラフを $G'(e)$ と表わす．つまり,$G'(e).V = G.V$ かつ $G'(e).A = G.A - \{e\}$ である．$G'(e)$ においてどの PI からも到達不能な組合せ回路要素 M が存在する場合，$e(\in G.A_3)$ 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． ■

データパスグラフがカットエッジ e を持つなら，e によって支配される組合せ回路要素は共通部分を持たない 2 つの制御経路を持つことができず，MUX を付加することにより，新たな経路を追加する．1 つの MUX を加えることにより，複数のカットエッジを解消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 8**] 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全ての組合せ回路要素を $D(e)$ と表す．ここで，G におけるカットエッジの半順序を次のように定義する．
カットエッジ e および e' において，$D(e) \subset D(e')$ のとき，またそのときに限り，$e < e'$ とする． ■

もし，2 つのカットエッジ e_1 と e_2 が $e_1 < e_2$ 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．このように提案手法では，最初に e_1 に対して TMUX を付加しておき，必要であれば e_2 に対しても TMUX を付加する．カットエッジがなくなるまで TMUX を付加し，その後は SC 法を適用して DFT を行なう．但し，提案手法は最初のステージでは制御経路および観測経路の決定は行なわない．図 4.1 に我々の DFT 手法の最初のステージを示す．

```
while (there exists a cut edge in G){
  e :=minimal cut edge in G;
  V(e) :=the set of all vertices unreachable from any PI in G(e);
  M(e) :=the maximal subgraph of G(e) with vertex set V(e);
  construct a spanning tree T of M(e)
  with the root u (u is the end vertex of e);
  V' := {v ∈ V(e) | v has an outgoing edge
  in M(e).E − T.E};
  choose any v from V' and insert a test MUX M
  in front of the v's input port;
  if (there exists a PI i unreachable to e)
    connect i to the other input port of M;
  else
    choose any PI i and connect i to
    the other input port of M;
  update G by adding M and the lines;
}
apply the DFT method in [15];
```

図 4.1 単一制御可検査 DFT

4.4. テストスケジューリング及び観測経路に関するDFT

テストスケジューリングは同時にテストする組合せ回路要素の集合 $\mathcal{M}$ を決定することである．提案する DFT アルゴリズムは，最小のハードウェア・オーバヘッドで観測経路を実現できるように $\mathcal{M}$ を決定する．観測経路を制御経路より先に決定するのは, 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．

テストスケジューリングにおいて，組合せ回路要素がすでに決定された集合 $\mathcal{M}$ に含まれるならスケジューリング済み，そうでないなら未スケジューリング組合せ回路要素という．ただし，全ての PO はスケジューリング済みとする．未スケジューリング組合せ回路要素から同時にテストする組合せ回路要素を決定する 2 ステップを以下に示す．

1. 候補組合せ回路要素の選択

   - k 個以上未スケジューリング組合せ回路要素が存在する場合: スケジューリング済み組合せ回路要素に隣接する全ての未スケジューリング組合せ回路要素を候補とする．候補数 ℓ が k 未満ならば，候補が k 個になるように，未スケジューリング組合せ回路要素を適当に候補に加える
   - $m(< k)$ しか未スケジューリング組合せ回路要素が存在しない場合：全ての未スケジューリング組合せ回路要素を候補とする．

2. 組合せ回路要素集合 $\mathcal{M}$ および $\mathcal{M}$ に含まれる組合せ回路要素の観測経路の決定

   次に示すようにデータパスグラフを構築し，流量 k（但し，候補数が $m(< k)$ 個なら流量 m）の最小費用流問題を解く．図 4.2 に $k = 2$ の例を示す．

   - 各候補組合せ回路要素に対する辺を持つダミー頂点を付加し始点とする．
   - 全ての PO からの辺を持つダミー頂点を付加し終点とする．
   - 各候補組合せ回路要素から PO へ TMUX に対応する辺を加える．

- 得られたグラフにおける各辺 (u, v) に対して，容量を $c(u, v) = 1$ と定義し，コスト $p(u, v)$ を以下のように定義する．

$$
\begin{aligned}
p(u, v) &= cost\_thru(u) && u \text{ が } v \text{ の入力端子.} \\
&= cost\_MUX && (u, v) \text{ が付加された MUX.} \\
&= 0 && \text{上記以外の場合.}
\end{aligned}
$$

  ($cost\_thru(u)$ は u から v の出力端子へのスルー機能のハードウェアコスト．スルー機能がすでに付加されている場合は，$cost\_thru(u) = 0$．$cost\_MUX$ は MUX のハードウェアコスト)

グラフの最小費用流は $\mathcal{M}$ を構成する組合せ回路要素数であり，最小費用流で用いる経路は，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．図 4.2 では，太線で表す経路が選ばれ，$cost\_thru(u) = 3$ で表されている $M1$ と $M4$ にスルー機能が付加される例である．

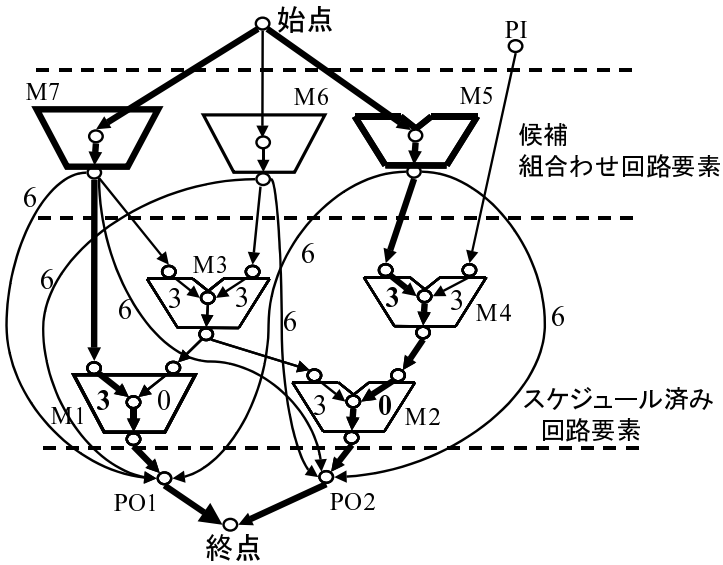


図 4.2 $(k = 2)$ スケジューリング例

4.5. 制御経路に関する DFT

与えられた組合せ回路要素集合 $\mathcal{M}$ に対して，$\mathcal{M}$ に含まれる全ての組合せ回路要素の制御経路を図 4.3 の手続きによって決定する．この手続きでは，$\mathcal{M}$ の要素は 1 つずつ考慮する．組合せ回路要素の制御経路は観測経路と同様に決定される：G から $\mathcal{M}$ に含まれる組合せ回路要素の全ての観測経路を除去して構成されるグラフ G' において，流量 2 の最小費用流問題を解く． G の代わりに G' において扱うのは，決定する制御経路がすでに決定されている観測経路と共通部分を持たないようにするためである．全ての制御経路が手続きによって互いに共通部分を持たない木を決定するので，各繰り返しステップの最後に G' を更新する．

図 4.3 において，組合せ回路要素に 2 つの共通部分を持たない制御経路が見つからない場合，TMUX M を制御経路を構成するために付加する．TMUX M は，付加された時には未スケジューリング組合せ回路要素とみなされ，M の制御経路と観測経路は後で決定する．データパスの構造によっては，M の入力端子にあらたにもう一つの TMUX が制御経路を決定するために加えられる可能性があり，さらに TMUX のための TMUX が付加され続ける可能性もある．連続した TMUX の付加を避けるために，必要に応じて図 4.3 の手続きに例外処理を行なう．この場合，グラフ G において M の制御経路の決定を妨げる観測経路を有する組合せ回路要素 $M' \in \mathcal{M}$ を検出し，M' から（TMUX を加えることで）PO に直接観測経路を生成することによって M' の観測経路を変更する．

4.6. テストマルチプレクサに関する考察

CSC 法では，ある組合せ回路要素 M の制御経路もしくは観測経路を新たに生成するために，TMUX（M' とする）を付加することがある．この TMUX M' もテストの対象であり，追加時には，M' は未スケジューリング組合せ回路要素とみなされる．つまり，M のスケジュールのために M' を加えることは，未スケジューリング組合せ回路要素数を減らすことには寄与しない．

そこで，TMUX M' を付加する代わりに，M を以降のテストセッションに移すことによって，全体として同じテストセッション数を小さいハードウェア・オー

```
G' := a digraph obtained from G by removing
  the observation paths of all modules in M
while (M ≠ ∅){
  choose any module v ∈ M;
  determine control paths P_1 and P_2 of v
  (by finding the minimum cost flow of two);
  add DFT elements for P_1 and P_2;
  for each module u on P_i (i = 1, 2)
    update G' by removing the outgoing edge
    from the input port of u that is not on P_i;
  M:=M-{v}
}
```

図 4.3 制御経路に関する DFT

バヘッドで実現できる可能性がある．ただし，M のテストのために M' を追加すると，M' を他の組合せ回路要素の制御経路および観測経路のために利用可能であり，他の TMUX の追加を防ぐことがある．このため，M' を付加しないことで，同じテストセッション数を小さいハードウェア・オーバヘッドで実現できるとは限らない．そこで，上記の手法の有効性を実験的に調査する．そのために，図 4.3において，M が前節で述べた例外処理を必要とする組合せ回路要素である場合，$M(\in \mathcal{M})$ を $\mathcal{M}$ から取り除き，未スケジューリング組合せ回路要素とするように変更した場合についても，実験結果を次節に示す．

ただし，上記の変更によって得られるデータパスは，変更しない場合と同じテストセッション数をより小さいハードウェア・オーバーヘッドで実現する場合であっても，k 並行可検査の定義を満たすとは限らない．これは，得られたデータパスの組合せ回路要素が少なくなるために，テストセッション数が $\lceil |V_c|/k \rceil$ となることを保証できないからである．

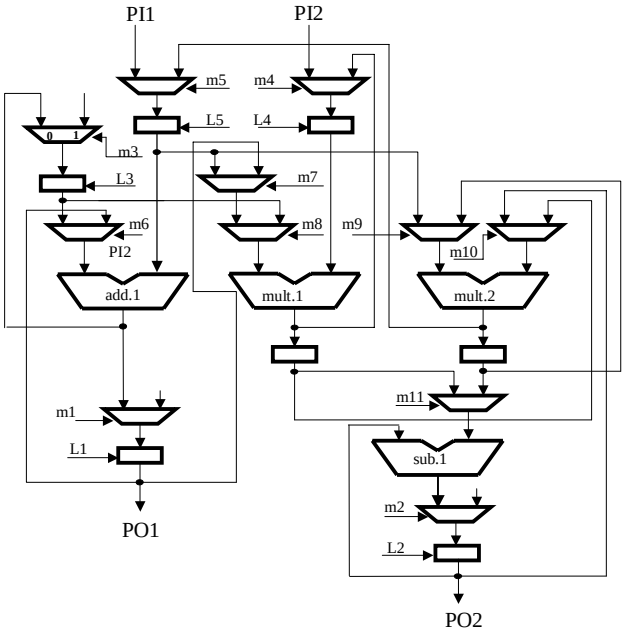


図 4.4 Paulin 元回路

4.7. 適用例

CSC 法をレジスタ転送レベルベンチマーク回路 Paulin(図 4.4) に適用したデータパスを図 4.5 に示す．図 4.5 は，並行度 k を 2 として得られた単一制御 2 並行可検査なデータパスである．図 4.5 における $m12, m13, m14, m15, m16, m17, m18$ は，ステージ 1 の処理により加えられた DFT 要素である．$m13$ は，図 4.4 の $sub1$ と $m2$ を接続する信号線が，カットエッジであるために加えられた MUX であり，$m12$ もカットエッジ処理によって加えられた MUX である．また，$m14, m15, m16, m17, m18$ は SC 法 [15] に基づいて挿入された MUX である．表 4.1 では，ステージ 2 の適用過程を示す. ID はテストセッションの ID を示し，test module は，セッションでテストされる組合せ回路要素を表す．DFTelements は，セッションを行なうために付加された DFT 要素を示す．例えば，add.1-r は，加算器 add.1 の右入力から出力へのスルー機能の付加を表す.

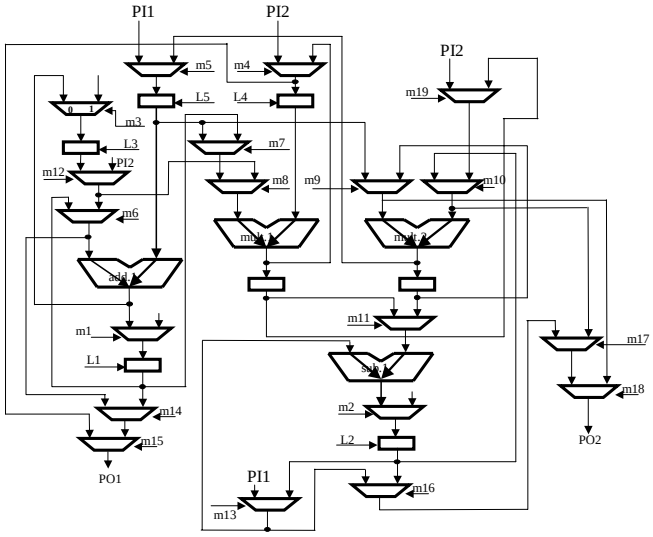


図 4.5 DFT 適用後 (k=2)

表 4.1 Paulin に対するステージ 2 の適用過程 ($k = 2$)

ID	test module		DFTelements
1	m15	m18	add.1-r,mult.1-r
2	m14	m17	
3	m1	m16	sub.1-r
4	m4	m2	mult.1-l
5	add.1	sub.1	
6	m6	m9	mult.2-r
7	m12	m11	mult.2-l
8	m5	m13	
9	m3	mult.1	
10	m8	m10	MUX(m19)
11	m7	mult.2	add.1-l
12	m19		

4.8. 実験結果

この節では，CSC 法の実験結果を示し，CSC 法と SC 法 [15] を比較する．実験に使用したレジスタ転送レベルベンチマーク回路は,Tseng および Paulin と 3rd Lattice Wave Filter(LWF) である．なお，SC 法では Tseng での適用結果について示されていないため，比較については Paulin と LWF で行なう．表 4.2にそれらのベンチマーク回路の特性を示す：#PI,#PO,#Reg,#MUX,#OP は それぞれ PI 数,PO 数, レジスタ数, マルチプレクサ数，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 AutoLogicII(Mentor Graphics) を用い，ALTERA 社の論理合成ライブラリを使用して，計算機上で論理合成を行うことによって得た.

表 4.2 回路特性

	#PI	#PO	#Reg	#MUX	#OP	Area(8bit)	Area(16bit)	Area(32bit)
Tseng	3	2	6	7	7	858.6	2499.4	8085.0
Paulin	2	2	7	11	4	1142.9	3818.9	13778.9
LWF	2	2	5	5	3	355.8	735.0	1493.4

表 4.3 は，DFT によって付加される TMUX とスルー機能数を示し，表 4.4 ではハードウェア・オーバヘッド (HW/OH) を示す．これらの表において，Modification については 4.6 節で示した変更によって得られた結果である．一般に BIST を実現する場合，PI および PO には TPG および RA を付加するので，表 4.4 にそ

表 4.3 付加 TMUX およびスルー機能

	SC method [15]		CSC methods							
	($k = 1$)		$k = 1$		$k = 2$		$k = 3$		Modification ($k = 3$)	
Circuit	#TMUX	#THRU	#TMUX	#THRU	#TMUX	#THRU	#TMUX	#THRU	#TMUX	#THRU
Tseng			8	8	10	11	15	8	14	8
Paulin	8	5	7	6	8	7	14	7	12	7
LWF	4	3	2	4	4	4	11	3	7	2

れらは含まないものとした．但し，k=3 の場合は, 新たに応答を観測するための LFSR を付加したため, そのハードウェア・オーバヘッドを加えたものを示している．表 4.4 より，CSC 法の $k = 1$ の場合，SC 法 [15] と比較してハードウェア・オーバヘッドが改善されていることがわかった．これは，今回追加したカットエッジに基づく処理が効果的に働いたためである．表 4.5 にテストセッション数を示す．表 4.4 および表 4.5 から，ハードウェア・オーバヘッドとテスト実行時間のトレードオフがわかる．提案手法の $k = 2$ の場合，SC 法 [15] よりもテストセッション数が削減できており，CS 法はテスト実行時間を改善している．しかしながら，TMUX のテストによってテストセッション数が増加したため，$k = 3$ におけるテストセッション数は予期していたよりも（組合せ回路要素数の約 1/3）より悪くなった．4.6 節で示した修正 DFT が今回実験した回路に関しては有効に働いたことが，表 4.4，4.5 から示される．故障検出率については示さないが，テスト対象組合せ回路要素に対して TPG からの値を伝搬し，その応答を RA へ伝搬する手法は SC 法 [15] と同じであるため，SC 法と同じ高い故障検出率が達成される．

表 4.4 ハードウェア・オーバヘッド

		SC method [15]	CSC methods			
		($k = 1$)	$k = 1$	$k = 2$	$k = 3$	Modification ($k = 3$)
Circuit	bit width	HW/OH(%)	HW/OH(%)	HW/OH(%)	HW/OH(%)	HW/OH(%)
Tseng	8	—	33.47	41.25	64.07	61.19
	16	—	22.61	27.83	42.43	40.48
	32	—	13.68	17.01	25.74	24.54
Paulin	8	22.50	19.15	26.17	46.38	41.98
	16	13.23	12.39	15.38	26.66	24.11
	32	7.27	6.80	8.66	14.48	11.00
LWF	8	35.10	23.66	37.55	109.13	78.92
	16	33.32	22.34	35.59	100.83	72.05
	32	32.47	21.71	34.66	96.87	68.78

表 4.5 テストセッション

	SC method [15]	CSC methods			
	($k = 1$)	$k = 1$	$k = 2$	$k = 3$	Modification ($k = 3$)
Circuit	session	session	session	session	session
Tseng		21	12	10	10
Paulin	23	22	12	10	10
LWF	12	10	6	7	6

4.9. まとめ

本章では，BIST のための RTL データパスの単一制御並行可検査性とその DFT について提案した．CSC 法の BIST の利点は高い故障検出率，低いハードウェア・オーバヘッド，短いテスト実行時間，システムクロックでテスト可能である．CSC 法では，MUX と演算モジュールを等価と扱っている．しかし，MUX のテスト系列長は演算モジュールのものより短い場合が多い．従って，組合せ回路要素毎のテスト時間を考慮したスケジューリングの考案が必要である．次章では，これらの問題点を解決するために時分割単一制御並行可検査性とその DFT について提案し，さらに RTL 全体に対して BIST を実現するようなアーキテクチャについて提案する．

第 5 章
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

5.1. 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 9**](**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およびセッション長，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．

5.2. DFT アルゴリズムの概要

単一制御並行可検査性を実現する 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関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 を行なう．

ステージ 2 テストセッション毎のテストスケジューリング，観測経路およ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

5.3. カットエッジ除去

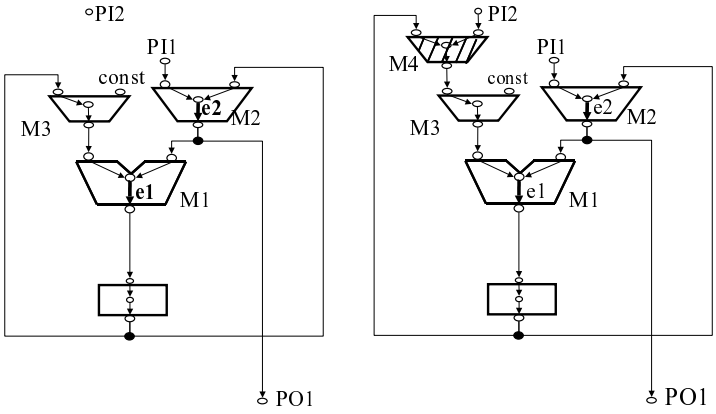
必要性の高い DFT 要素を早い段階で付加するために，スケジューリングに関わらず時分割単一制御並行可検査性を満たさないような組合せ回路要素に対して，以下に定義するカットエッジに基づいた処理を行う．

[**定義 10**] 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 $G'(e)$ と表わす．$G'(e)$ 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関わらず，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．1 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 11**] 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全ての組合せ回路要素を $D(e)$ と表す．e および e' において，$D(e) \subseteq D(e')$ のとき，またそのときに限り，$e \leq e'$ とする． ■

もし，2 つのカットエッジ e_1 と e_2 が $e_1 \leq e_2$ 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．



(a) カットエッジ処理前　(b) カットエッジ $e1$ 処理後

図 5.1　カットエッジ処理例

[例 2] 図 5.1(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．従って，$e1 \leq e2$ となり，まず，$e1$ に対してカットエッジ除去を行なう．$M3 \in D(e1)$ 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 5.1(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．■

5.4. テストスケジューリング及び観測経路に関するDFT

1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，すで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．

スケジューリング済みの組合せ回路要素に隣接する組合せ回路要素，及び PO と観測端子に隣接する全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 $\mathcal{M}$ および $\mathcal{M}$ 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．これは 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からである．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築する．

- 各候補組合せ回路要素への辺を持つダミー頂点を付加し始点とし，各 PO および観測端子からの辺を持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 も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なわない．
- 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．

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

($cost\_thru(u)$ 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．

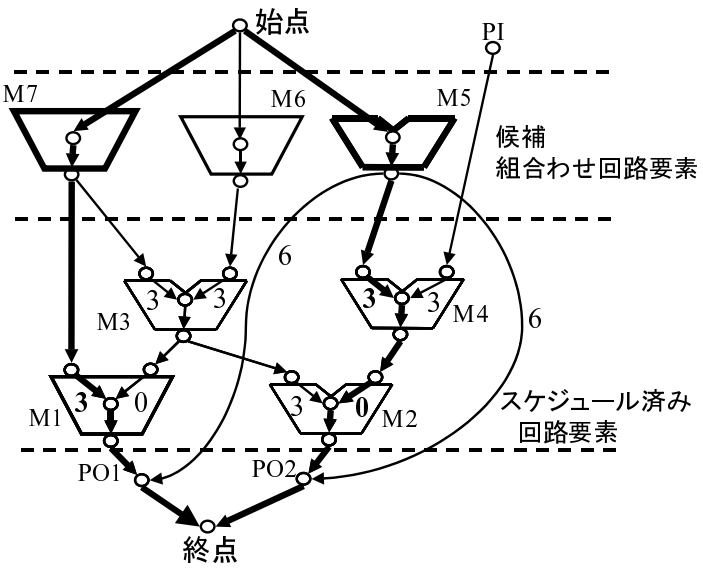


図 5.2 ($k = 2$) テストスケジューリング例

[例 3] 図 5.2 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，スルー機能を持たない辺がコスト 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 $M1$ 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． ■

5.5. 制御経路に関する DFT

1 つのテストセッション $\mathcal{M}$ に対して，$\mathcal{M}$ に含まれる全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 1 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 1 つずつ決める．各回路要素に対する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 1 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 への辺を持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およ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．(すで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，観測経路を求める時と同様のコスト，容量を与える．

[**例 4**] 図 5.3(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付ける．各経路は，順序深

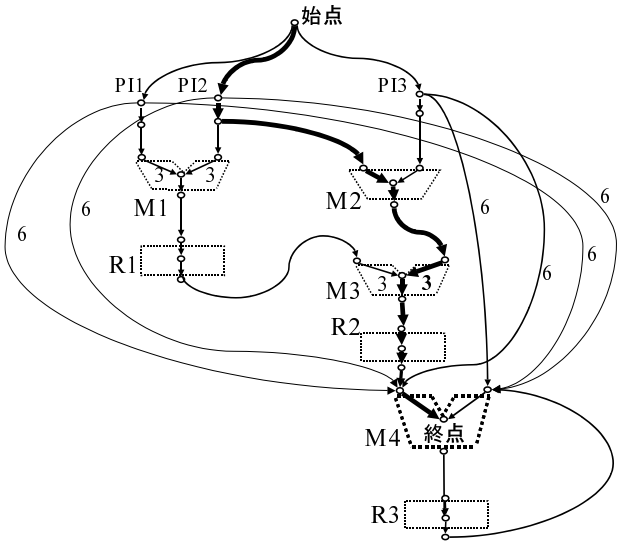
度に対応した頂点を通るように接続する．例えば，始点から終点までの順序深度が d の経路は，d に対応する頂点を通る．TMUX に対応する辺は順序深度 0 の経路として考える．

- 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 type3cost は，type3 を用いる場合の時間のコストと対象回路へのスルー機能を付加するハードウェアのコストである．

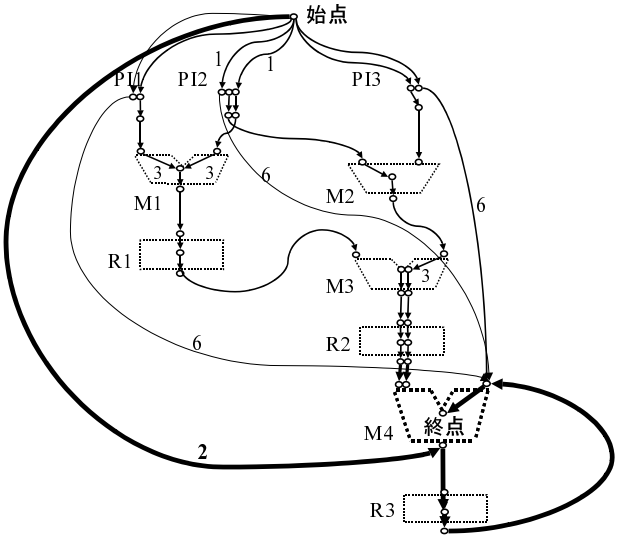
[**例 5**] 図 5.3(b) は，$M4$ の第 2 制御経路の決定例を示す．始点から $M4$ までの順序深度を考慮して端子を複製する．例えば，PI3 に対しては，TMUX による経路 (順序深度 0) と順序深度が 1 の経路が存在するので，2 つの端子を用意する．すでに同じセッションに対して決定している制御経路上の組合せ回路要素の制御経路に含まれない辺である $M2$ の右入力からの辺と $M3$ の左入力からの辺を削除する．始点から PI2 に対応する端子へは，順序深度 1 に対応する端子以外に接続し，コストを $type2cost(=1)$ とする．また, 始点から $M4$ の出力への辺を付加し，コストを $type3cost(=2)$ とする．$M4$ の右入力への 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． ■

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．データパス中の全ての組合せ回路

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．



(a) 第 1 制御経路の決定



(b) 第 2 制御経路の決定

図 5.3 制御経路の決定

5.6. テストセッション長決定

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 $\mathcal{M}_1$,$\mathcal{M}_2$,...,$\mathcal{M}_{s-1}$ がすでに決定し，$\mathcal{M}_s$ のセッション長を決定するとする．$l_j(j=1,2,...s)$ をセッション $\mathcal{M}_j$ のセッション長とする．$\mathcal{M}_{j,1}$,$\mathcal{M}_{j,2}$,$\mathcal{M}_{j,3}$ をそれぞれ $\mathcal{M}_j$ において type1, type2, type3 の経路を持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, $\mathcal{M}_p^{i,s}=\{j|M_i \in \mathcal{M}_{j.p}, 1 \leq j \leq s\}$ 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 $M_1,...,M_s$ 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 1 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．

5.7. 制御経路の再決定とセッション長決定

ステージ 2 では，テストセッション毎に制御経路および観測経路の決定を行なった．しかし例えば，あるテストセッションで type2 の制御経路を持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 の制御経路を持つというように，よりテスト実行時間の小さい経路を持つ可能性がある．DFT 要素が付加されたデータパスに対して 5.5 節で示した手法を用いて再度制御経路を決定する．但し，この際に新たに DFT 要素に対応する辺の付加は行なわない．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 [17] やテストポイント挿入 [18] などを行ない所望の検出率を達成できるようにするか，も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．

5.8. BIST アーキテクチャ

図 5.4 に提案したレジスタ転送レベル全体に対する階層 BIST アーキテクチャについて一例を示す．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 を行なう．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．また入力には 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．

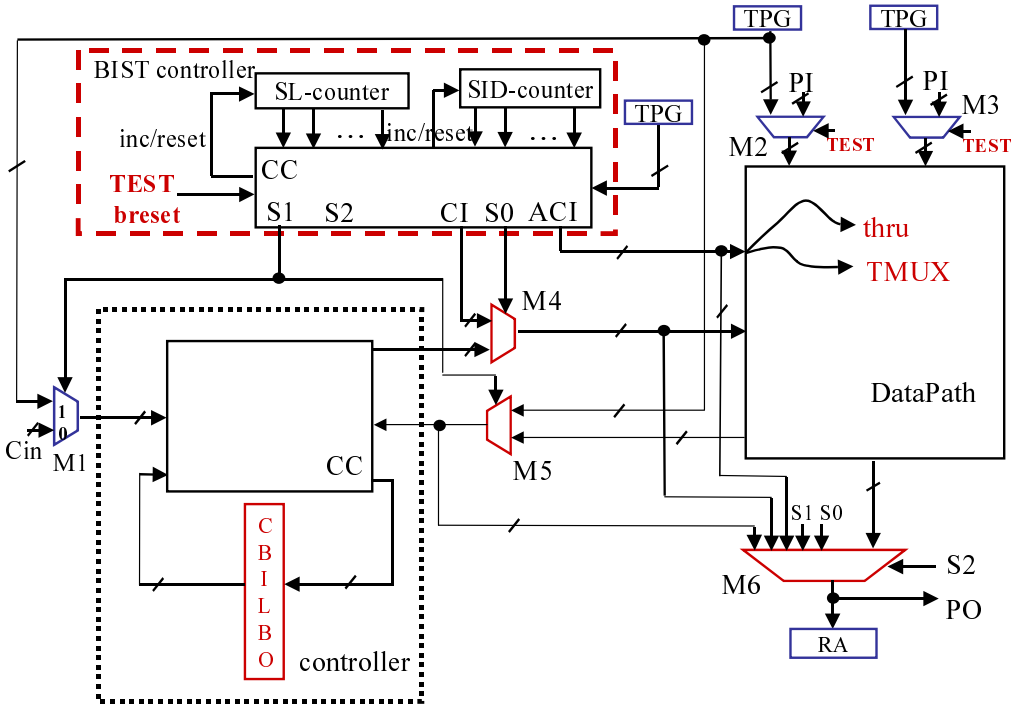


図 5.4 BIST アーキテクチャ

このアーキテクチャは **TEST** ピンと **breset** ピンの 2 つの外部ピンをもつ．TEST ピンが 1 の時に回路に対してテストが実行される．破線部の BIST コントローラは，データパスに付加した DFT 要素の制御信号，およ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 2 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，1 つのテストセッションと考える．BIST コントローラの動作について示す．

- 通常動作時:DFT 要素に対して，通常動作時に対応する制御信号を供給する．
- 演算モジュールのテスト時: テストセッションに応じたテストプランを供給される．
- 観測モジュールのテスト時: テストセッションに応じたテストプランを供給される．観測モジュールの出力を MUX M5 を介して RA で観測する．
- コントローラのテスト時：CBILBO をテストモードにする．

TPG，RA および 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 2 つの外部ピンと 3 つの MUX およ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．

5.9. 実験結果

提案手法と Wunderlich らの手法 [11] および CSC 法 [16] を比較する．使用した RTL 回路は,LWF，Paulin および Tseng(表 5.9) 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 5.1 回路特性

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 5.2，表 5.3 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 5.2) および並行度 $k = 2$ の場合 (表 5.3) 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，全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 要素が減少し，同時にテスト実行時間を減少させたためである．

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なった (表 5.4)．TCSC 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，組合せ回路要素毎にテストを行なうのに対し，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なうからである．しかし，テスト実行時間に関しては TCSC 法で時間を優先した場合ではかなり短くなっている．これらの結果から，提案した手法は短い時間で非常に高い故障検出率を達成することのできる BIST 法であると言える．

また，test-per-scan 方式の BIST と比較するために，図 5.5 のように，2 つのカウンタから構成される BIST コントローラを持ち，PI，PO には並列にパターンを印加できる LFSR，scan-in,scan-out には最上位ビットを出力する LFSR を接続するアーキテクチャを用いた．ここで，テスト対象回路 (Circuit under test. CUT) は RTL 回路全体であり，すべての FF をスキャン FF に置き換えるフルスキャン設計を行なうものとした．また，テスト実行時間は
(テスト実行時間) = (テストパターン数) × (スキャンパス長 + 1) + (スキャンパス長) ただし,(スキャンパス長) = (フリップフロップ数) として求めた．ハードウェアオーバヘッドは，元回路に対して，LFSR，スキャン FF，BIST コントローラ，スルー機能，および TMUX を付加した回路の面積の割合を示したものである．TCSC 法は，Test-per-scan 方式に比べて大幅にテスト実行時間を削減できている．また，ハードウェアオーバヘッドも LWF，Paulin の場合で，TCSC 法の方が小さい．

5.10. まとめ

本章では，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 および RTL 全体に対する階層 BIST アーキテクチャを提案した．TCSC 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．また，ハードウェアオーバヘッドは Test-per-scan 方式よりも小さくなる場合もあり，提案した手法は実動作速度テストを

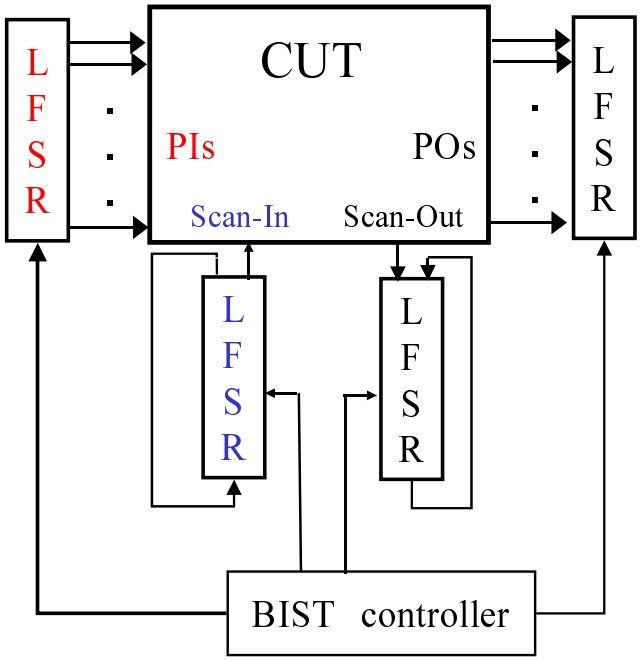


図 5.5 Test-per-scan アーキテクチャ

非常に小さなハードウェアオーバヘッドで実現することができる.

表 5.2 ハードウェアオーバヘッドおよびテスト実行時間 ($k = 1$)

回路	ハードウェアオーバヘッド (%)															テスト実行時間 (#clock)		
	CSC 法 [16]					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	[16]	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 5.3 ハードウェアオーバヘッドおよびテスト実行時間 ($k = 2$)

回路	ハードウェアオーバヘッド (%)															テスト実行時間 (#clock)		
	CSC 法 [16]					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	[16]	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 5.4 データパス部のオーバヘッド

		Wunderlich らの手法 [11]	CSC 法 [16]		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

表 5.5 test-per-scan 方式との比較

		test-per-scan 方式	TCSC 法			
			面積優先		時間優先	
			$k = 1$	$k = 2$	$k = 1$	$k = 2$
LWF	HW/OH(%)	103.05	79.90	81.13	79.90	81.13
	クロック数	21944	626	387	626	387
	FC(%)	100	100	100	100	100
Paulin	HW/OH(%)	24.14	20.95	21.72	22.50	23.28
	クロック数	196581	3136	2705	1995	1201
	FC(%)	99.99	99.99	99.99	99.99	99.99
Tseng	HW/OH(%)	36.75	42.48	43.41	43.68	43.41
	クロック数	156617	3411	1592	1783	1050
	FC(%)	99.99	99.99	99.99	99.99	99.99

第 6 章
結論

本論文では，階層 BIST に基づき実動作速度でのテストが可能な BIST を実現する手法を提案した．

2 章では，レジスタ転送レベル回路について述べ，組み込み自己テスト方式および実動作速度でのテストが可能な BIST の従来法として Wunderlich らの手法と階層 BIST に基づく井筒らの手法について述べた．

3 章では，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および時分割単一制御並行可検査性を定義した．

4 章で提案した単一制御並行可検査性に基づく手法 (CSC 法) は，単一制御可検査性に基づく手法の欠点であったテスト実行時間を改善し，またカットエッジ処理を新たに適用することでハードウェアオーバヘッドの削減も行なった．5 章で提案した時分割単一制御並行可検査性に基づく手法 (TCSC 法) では，CSC 法に比べて制御経路や観測経路の条件を緩和することで故障検出率を減らすことなくハードウェアオーバヘッドの削減を行なった．さらに，テスト実行時間を考慮したテストスケジューリングを行なうことによって，テスト実行時間の削減も行なった．これらの手法は，双方とも階層 BIST に基づいて組合せ回路要素毎にテストを行なうために，一般に BIST の欠点といわれる故障検出率の低さを大幅に補い，さらに TPG や RA を回路の外部からのみ用いることで小さなハードウェアオーバヘッドで BIST を実現できる．また，SC 法の欠点であったテスト実行時間も改善し，実用的なテスト実行時間でテストが可能である．

また，データパスが単一制御可検査，単一制御並行可検査およびの時分割単一制御並行可検査な場合に，RTL 全体に対して BIST を実現するためのアーキテクチャを提案した．

本手法の今後の課題としては，BIST アーキテクチャ特に BIST コントローラを改良することによって，ハードウェアオーバヘッドをより削減することが挙げ

られる．また，DFT 要素を付加する際に回路の動作速度への影響や，回路の消費電力を考慮する必要がある．さらに，データパス内のビット幅が異なった場合に対するテストなどがある．

謝辞

本研究の機会を与えて下さるとともに，本研究の全過程を通じて，方針や問題点などのあらゆる面において，懇切丁寧な御指導と御助言を賜わりました藤原秀雄教授に心から感謝の意を表します.

本研究を進めるにあたり，御指導と御助言を頂きました渡邉勝正教授に感謝の意を表します.

本研究の全過程を通じて，貴重な御討論，御助言を頂き，懇切丁寧に直接的な御指導を頂きました大阪大学基礎工学部の増澤利光教授ならびに本学の井上美智子助教授に深く感謝致します．本研究に際して，様々な御指導，御助言を頂きました大竹哲史助手ならびに米田友和助手に深く感謝致します.

本研究を進めるにあたり，貴重な御討論，御助言頂きました (株) 半導体理工学センタ (STARC) の小澤時典取締役研究推進部長，村岡道明 VCDS 開発室長ならびに細川利典氏,（株）日立製作所中央研究所の畠山一実氏，ソニー (株) セミコンダクタカンパニーシステム LSI 部門の小野寺岳志氏，三洋電機 (株) 研究開発本部マイクロエレクトロニクス研究所の小椋功氏ならびに (株) システムジェイディーの伊達博氏に感謝致します.

本研究に際して，様々な御討論，御助言を頂きました（株）日立製作所中央研究所の和田弘樹氏に感謝致します.

また，日頃より有益な御討論，御援助を頂きましたコンピュータ設計学講座 (旧情報論理学講座) の諸氏に感謝致します.

最後に，生活面で全面的に支えて頂いた両親および妻に感謝します.

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、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.

参考文献

[1] H. Fujiwara, *Logic Testing and Design for Testability*, The MIT Press (1985).

[2] P. H. Bardell, W. H. McAnney, J. Savir : *Built-In Test for VLSI : Pseudo-random Techniques*, Wiley-Interscience (1987).

[3] M. Abramovici, M. A. Breuer and A. D. Freedman : *Digital Systems TESTING and Testable DESIGN*, Computer Science Press (1990).

[4] B. Koenemann, J. Mucha, G. Zwiehoff : "Built-in test for complex digital integrated circuits," *IEEE Journal Solid-State Circuits*, Vol.SC-15, pp.315-318 (1980).

[5] 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," 電子情報通信学会論文誌 (DI), Vol.J82-D-I, No.7, pp.843-851 (1999).

[6] V.K.Agrwal : Multiple fault testing of large circuits by single fault test sets," *IEEE trans. Comput.*, C-30, 11, pp.855-856.(1981).

[7] M. R. Garey, D. S. Johnson : *Computers and Intractability : A Guide to the Theory of NP-Completeness*, Freeman(1979).

[8] P.Bardell and W.H.McAnney:"Self-Testing of Multichip Logic Modules," In Proc.1982 IEEE Test Conf., pp.200-204(1982).

[9] B.Koenemann, J.Mucha,and G.Zwiehoff:"Built-in logic block observation techniques," In Proc.1979 IEEE Test Conf., pp.37-41(1979).

[10] L.T.Wang and E.J McCluskey: "Concurrent Built-in logic block observer(CBILBO)," In Proc. the Int. Symp. on Circuits and Systems, pp.1054-1057 (1986).

[11] A.P.Stoele and H.J.Wunderlich, "Hardware-optimal test register insertion," IEEE Trans. on Computer-Aided Design of Intergrated Circuits and Systems, 17(6):531-539(1998).

[12] B. T. Murray and J. H. Hayes : "Hierarchical test generation using pre computed tests for modules," *IEEE Trans. on CAD*, VOL.16, NO.9, pp.1001-1014 (1990).

[13] I.Ghosh, N.K.Jha, and S.Bhawmik,"A BIST Scheme for RTL Circuits Based on Symbolic Testability Analysis," IEEE Trans. on Computer-Aided Design of Intergrated Circuits and Systems, 19(1):111-128(2000).

[14] 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 (DI), Vol.J84-D-I, No.5, pp.454-465 (2001).

[15] 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," 電子情報通信学会論文誌 (DI), Vol.J84-D-I, No.1, pp.69-77 (2001).

[16] 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，" 電子情報通信学会論文誌 (DI), Vol.J85-D-I, No.6, pp.527-537 (2002).

[17] B. Koenemann: "LFSR-Coded Test Patterns for Scan Design," *in Proceedings of the European Test Conference(ETC)*, pp.237-242 (1980).

[18] B. Krishnanmurthy:"A dynamic programming approach to the test point insertion problem," Proc. ACM/IEEE DAC, pp.695-705 (1987)

[19] Daniel D. Gajski, Nikil D. Dutt, Allen C-H Wu and Steve Y-L Lin:*High-level synthesis: introduction to chip and system design*(1992)

[20] D. Berthelot, M.L.Flotter, B.Rouzeyre:"BISTing Datapaths under Heterogeneous Test Scheme,"Journal of Electronic Testing Theory and Applications 14, 115-123(1999)