システムＬＳＩ設計技術　95－4
（2000. 2. 9）

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

井筒稔　和田弘樹　増澤利光　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:{minoru-i, hiroki-w, masuzawa, fujiwara}@is.aist-nara.ac.jp

あらまし　本稿では，レジスタ転送レベルデータパスの組込み自己テスト方式（BIST）として，階層テストに基づく test per clock 方式の BIST を提案する．この手法では，テストパターン生成器，応答解析器をテスト対象回路の外部入力，外部出力のみに付加し，各組合せ回路要素に対して，データパスの経路を用いてテストパターン，応答を伝搬する．本稿では，この階層テストに基づく BIST が可能なデータパスとして，単一可制御データパスを定義し，与えられたデータパスを単一可制御データパスに設計変更するためのテスト容易化設計法を提案する．また，ベンチマーク回路を用いて，提案手法の評価を行う．

キーワード　テスト容易化設計, レジスタ転送レベルデータパス, 階層テスト, 組込み自己テスト

A DFT Method for BIST Based on Single-Control Testability of RTL Data Paths

Minoru Izutsu, Hiroki Wada, Toshimitsu Masuzawa, and Hideo Fujiwara

Graduate School of Information Science,
Nara Institute of Science and Technology
8916-5, Takayamacho, Ikoma, Nara 630-0101
Tel : +81-743-72-5226　Fax : +81-743-72-5229

e-mail:{minoru-i, hiroki-w, masuzawa, fujiwara}@is.aist-nara.ac.jp

Abstract This paper persents a new BIST (Built-In Self Test) method for register transfer level data paths based on both hiererchical testing and "test-per-clock" scheme. In the proposed method, test pattern generaters and response analysers are placed on primary inputs and primary outputs. Test patterns and corresponding responses are transmitted by using paths on a data path defiened for each module. The concept of single-control testable data path is introduced as a data path that the BIST based on hierarchical testing can be applied to. The design for testability method that transforms a data path into single-control testable one is also proposed. The proposed method is evaluated in experiment for some benchmarks.

Key words　Design For Testability, RTL Data Path, Hierarchical Test, Built-In Self Test

1 はじめに

VLSIの大規模化，複雑化に伴い，テストパターン生成，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）[1]の重要性がますます高まっている．BISTを実現するために，テスト対象回路の外部入力，外部出力に，それぞれ，テストパターン生成器，応答解析器を付加する．しかし，テスト対象回路に閉路が含まれている場合には，外部入出力にテストパターン生成器，応答解析器を付加するだけ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[2]．

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，テストパターン生成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パターンを印加できず，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）レジスタをテストパターン生成器，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)[3]，CBILBO（Concurrent BILBO）が用いられる．test per clock方式では，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（at-speed test）が可能である．このため，テスト実行時間が短く，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．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，文献[4]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示している．

本稿では，test per clock方式のBISTとして，階層テスト[5, 6]に基づく方法を提案する．この手法では，内部レジスタをBILBOやCBILBOに変更せず，テストパターン生成器，応答解析器は，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，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）ごとテストを行う．つまり，テストパターンをテストパターン生成器から各組合せ回路要素まで伝搬し，応答をその組合せ回路要素から応答解析器まで伝搬する．このテストパターン，応答の伝搬は，データパス中の経路を利用する．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．もしそのような経路がデータパス中に存在しなければ，データパスに経路を付加する必要がある．また，この経路に演算モジュールが現れる場合には，この演算モジュールの入力端子から出力端子へテストパターン，応答を伝搬できるように，この演算モジュールにスルー機能を付加する必要がある．

本稿では，階層テストに基づくtest per clock方式のBISTが可能なデータパスとして，単一制御可検査データパスを定義し，与えられたデータパスが単一制御可検査になるように設計変更するテスト容易化設計法を提案する．本稿で提案するBISTの特徴は以下の通りである．

- 高い故障検出率：テストはデータパス中の組合せ回路要素ごとに行われる．実際のデータパスで使用され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，テストパターンとしてランダムパターンを用いることにより，高い故障検出率が得られることが知られており[7]，本手法で高い故障検出率を得ることが期待できる．
- 低いハードウェア・オーバーヘッド：テストパターン生成器，応答解析器を外部入出力のみに付加するので，文献[4]の手法に比べ，ハードウェア・オーバーヘッドが小さいことが期待できる．
- test per clock方式：連続クロックで，テストパターンの生成，応答の解析が可能であり，実時間テストが可能である．

以下，2節では本稿で対象とするデータパスを定義し，データパスグラフについて述べる．3節ではデータパスの単一制御可検査性を定義する．4節では単一制御可検査性に基づくテスト容易化設計法を述べる．5節ではベンチマーク回路を用いた実験により，提案手法を評価する．

2 データパス

本稿で対象とするデータパスを定義する．

データパスは以下の構成要素から成る．

- 回路要素
- データ信号線:回路要素を相互に接続．
- 制御信号線:制御回路から回路要素へ制御信号を伝達．
- 状態信号線:回路要素から制御回路へ状態信号を伝達．

以下に各構成要素について説明する．

- **回路要素:** 回路要素は**外部入力**，**外部出力**，**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．

  回路要素に対してデータ信号線が接続される端子を**データ端子**，制御信号線が接続される端子を**制御端子**，状態信号線が接続される端子を**状態端子**と呼ぶ．データ端子は回路要素に信号を入力するためのデータ入力端子と回路要素からデータを出力するためのデータ出力端子に分類される．(以下，データ入力端子を入力端子，データ出力端子を出

力端子と呼ぶ.) データパス上のすべての回路要素のデータ端子は等しいビット幅を持つものとする.

外部入力, 外部出力： 外部入力 (PI) はデータパス外部からデータパスにデータを入力するための端子, 外部入力 (PO) はデータパスから外部にデータを出力するための端子である. 便宜上, 外部入力は出力端子のみを持ち, 外部出力は1つの入力端子のみを持つものとする.

マルチプレクサ： マルチプレクサは2つの入力端子と1つの出力端子, 1ビットの制御端子を持つ. 制御端子の値に従って, 対応する入力端子の値をそのまま出力端子に出力する.

演算モジュール： 演算モジュールは1つまたは2つの入力端子, 1つの出力端子, 高々1つの制御端子, 高々1つの状態端子を持つ. 入力端子に与えられた値に対して演算を行って出力端子に出力する.

観測モジュール： 観測モジュールは1つまたは2つの入力端子, 高々1つの制御端子, 1つの状態端子を持ち, 出力端子を持たない. 観測モジュールは制御回路に伝達するための状態信号を生成することのみを目的とした比較器 (コンパレータ) 等をモデル化したものである.

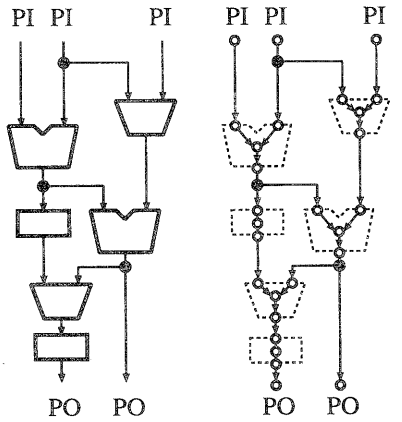
ラッチ, レジスタ： ラッチ, レジスタはいずれも記憶素子である. ラッチは1つの入力端子と1つの出力端子を持つ. 入力端子に与えられた値を記憶し, その値を次のクロックサイクルで出力端子に出力する. レジスタは1つの入力端子と1つの出力端子, 1ビットの制御端子を持つ. 制御端子の値によって, 入力端子の値を新たに記憶するか (ロード), 既に記憶している値を保持する (ホールド). 記憶している値は次のクロックサイクルで出力端子に出力する.

- **データ信号線：** データ信号線 (以降, 単に信号線と呼ぶ) は相異なる回路要素の出力端子と入力端子を接続する. 複数の信号線を同一の出力端子に接続できる (ファンアウト可能) が, 入力端子に接続する信号線は1本のみとする.
- **制御信号線, 状態信号線：** 制御信号線は制御回路からデータパス上の回路要素に対して制御信号を伝達し, 状態信号線は回路要素から制御回路に対して状態信号を伝達する.

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.



(a) データパス　(b) データパスグラフ

図 1: データパスとデータパスグラフ

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

  ここで A_1 はデータ信号線に対応し, $A_1 = \{(x, y) \in V_2 \times V_2 |$ 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の対応, 回路要素と出力端子の対応を表す. つまり, $A_2 = \{(x, u) \in V_2 \times V_1 | x$ 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 1(a) のデータパスに対するデータパスグラフを図 1(b) に示す. 経路, 単純経路, サイクルなどのグラフ用語をデータパスグラフに対しても用いる.

本稿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つの外部入力から到達可能であり, すべての出力端子は少なくとも1つの外部出力に到達可能であるものとする.

3 単一制御可検査性

本稿では, test per clock 方式の BIST で高い故障検出率を達成できるデータパスのテスト容易化設計法を提案する. 実際のデータパスで用いられ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) に対しては, ランダムパターンをテストパターンとして用いることにより, 高い故障検出率が得られる [7]. 比較器については, ランダムパターンでは高い故障検出率を得るのが困難だが, 制御点, 観測点を付加することにより, ランダムパターンで高い故障検出率を得ることができる [7]. このことから, データパス中の各組合せ回路要素 M に対して, 以下の2条件が成り立てば, M にランダムパターンを用いてテストを実行することにより, 高い故障検出率を得られる.

1. Mの各入力端子まで入力端子ごとに異なるテストパターン生成器で生成したランダムパターンを伝搬可能.
2. Mの出力端子の値を応答解析器まで伝搬可能.

本稿のテスト容易化設計法では，ハードウェア・オーバーヘッドを低く抑えるために，テストパターン生成器は外部入力のみに，応答解析器は外部出力のみに配置する．この制約上ですべての組合せ回路要素が上記の2条件を満たすための十分条件として，データパスの単一制御可検査性を次のように定義する．

[定義 1](単一制御可検査データパス DP)
対応するデータパスグラフにおいて，各組合せ回路要素 $M(\in V_1)$ に対して，以下の条件を満たす互いに共通部分を持たない単純経路 P_1, P_2, P_3が存在し，各経路に対応するデータパス上の経路に沿って任意の値が伝搬可能であるとき，データパス DPは単一制御可検査であるという．

- 外部入力を始点とし，Mの入力端子を終点とする経路 P_1および P_2*

  ただし，Mが1つの入力端子しか持たない場合，P_2を空経路とする．
- Mの出力端子を始点とし，外部出力を終点とする経路 P_3.

  ただし，Mが出力端子を持たない (観測モジュール) 場合，P_3を空経路とする． □

P_1, P_2を Mの**制御経路**，P_3を Mの**観測経路**とよぶ．

単一制御可検査データパスにおいて，組合せ回路要素 Mに対して，制御経路を用いて外部入力から連続したテストベクトルを印加し，観測経路を用いて Mの応答を連続して外部出力で観測できる．これらの経路上には，演算モジュールやマルチプレクサが現れるが，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†．この制御は，Mのテストの間固定しておけばよく，各組合せ回路要素に対して，1つの制御パターンで十分なので，単一制御可検査性とよんでいる．

4 テスト容易化設計法

4.1 テスト容易化設計

本節では，与えられたデータパスを単一制御可検査データパスに設計変更するためのテスト容易化設計法 (Design For Testability，以下 DFT) を示す．単一制御可検査データパスは，データパスの各組合せ回路要素 Mに対して，互いに共通部分を持たない制御経路と観測経路を持ち，それぞれの経路に沿って任意の値を伝搬できる．

与えられたデータパスにおいて，ある組合せ回路要素 Mに対して，互いに共通部分を持たない制御経路，観測経路が存在しない場合，単一制御可検査にするためには，データパスに新たな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，マルチプレクサを用いて実現する．また，制御経路，観測経路に M以外の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 2](単一制御可検査 DFT)
単一制御可検査のための DFT を次の最適化問題として定義する．
- 入力：データパス
- 出力：単一制御可検査データパス
- 最適化目標：付加する DFT 要素 (マルチプレクサ，スルー機能) のハードウェア量最小化 □

4.2 テスト容易化設計法

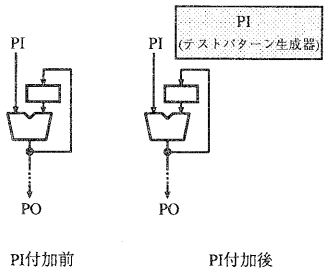
単一制御可検査 DFT のための発見的アルゴリズムを示す．本アルゴリズムは，以下の2段階からなる．
1. 制御経路の決定と DFT 要素付加
2. 観測経路の決定と DFT 要素付加

組合せ回路要素 Mに対して，互いに共通部分を持たない制御経路，観測経路が存在するかどうかを判定する問題は，本質的に2品種フロー問題である．2品種フロー問題は NP 完全であるため [8]，本アルゴリズムでは，制御経路と観測経路に分離して決定する．

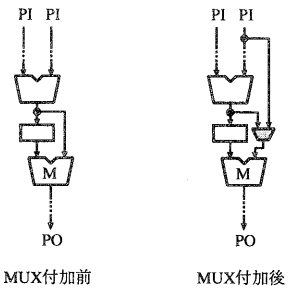
4.2.1 制御経路の決定と DFT 要素付加

各組合せ回路要素に対し，最小の付加ハードウェアで実現できる制御経路 (2入力組合せ回路要素の場合は，互いに共通部分を持たない2つの制御経路) を決定し，DFT 要素 (マルチプレクサ，スルー機能) を付加する．制御経路を求める組合せ回路要素は，一つづつ処理していくが，先の処理で付加した DFT 要素は後の処理でも利用できるので，組合せ回路要素を処理する順序によって，全体のハードウェア・オーバーヘッドは異なる．全体のハードウェア・オーバーヘッドを低く抑えるには，なるべく必要性の高い DFT 要素から付加していくことが望ましい．そこで，前処理として，必ず付加が必要な DFT 要素を付加する．次に，外部入力に近い組合せ回路要素から順に制御経路を決定する．これは，外部入力に近い組合せ回路要素ほど，制御経路の選択肢が少なく，そこで付加する DFT 要素は必要性が高いと考えられるからである．また，外部入力に近い部分の DFT 要素ほど，以降の処理で再利用できる可能性が高いと考えられるからである．

*P_1と P_2の始点は相異なる．
†演算モジュールでは，テスト容易化設計で付加するスルー機能を利用する．



(a) PI の付加



(b) MUX の付加

図 2: 前処理

・前処理

(a) データパスに外部入力が 1 つしか存在しない場合：組合せ回路要素に対して，互いに共通部分を持たない 2 つの制御経路が存在しないことは明らかである．これを解決するため，外部入力の付加を行う (図 2(a))．実際には，外部入力の代りにテストパターン生成器をデータパスに付加する．しかしながら，以下のアルゴリズム記述の便宜上，外部入力の付加として扱う．

(b) 2 入力組合せ回路要素 Mのそれぞれの入力端子に対して，データパスグラフでその先祖を辿ったときに最初に現れる 2 入力組合せ回路要素が同じものであるとき：Mに対して，互いに共通部分を持たない 2 つの制御経路が存在しない．これを解決するため，データパスにマルチプレクサを付加することにより，新たな経路を付加する (図 2(b))．

図 3に示す手続きで，2 入力組合せ回路要素 Mに対して，データパスグラフ上で先祖を辿り，Mに対してマルチプレクサの付加が必要かどうかを判定する．

2 入力組合せ回路 Mに対するマルチプレクサの付加は以下のように行う (図 2(b))．

マルチプレクサの付加：Mの入力端子を x, yとすると，xの直前にマルチプレクサを挿入し，マルチプレクサのもう一方の入力端子は外部入力に接続する．このとき，yへ到達可能な外部入力が 1 つしかない場合には，その外部入力以外の外部入力に接続する．

```
/* Mは対象となる 2 入力組合せ回路要素 */
/* M1, M2 は Mに入力を与えている組合せ回路要素 */
/* terminalは端子を格納する変数.  */

preprocess(M){
terminal = Mの一方の入力端子 it_1;
M1 = it_1に接続している回路要素;
while(M1 が 1 入力回路要素){
 terminal = Mの入力端子;
 M1 = terminalに接続している回路要素;
}
terminal = Mの他方の入力端子 it_2(≠ it_1);
M2 = it_2に接続している回路要素;
while(M2 が 1 入力回路要素){
 terminal = Mの入力端子;
 M2 = terminalに接続している回路要素;
}
if (M1 = M2)
 return TRUE; /* マルチプレクサの付加が必要 */
else
 return FALSE;
}
```

図 3: 前処理の手続き

・制御経路の決定

外部入力に近い組合せ回路要素から順に，制御経路を決定し，DFT 要素を付加する．具体的には，以下の方法で対象とする組合せ回路要素を決定し，その組合せ回路要素に対して，データパスグラフ上で最小費用流問題を解くことで，最小の付加ハードウェアで実現できる制御経路を決定する．

対象となる組合せ回路要素の決定方法：組合せ回路要素 Mの入力端子の先祖を辿ったときに最初に現れる組合せ回路要素 (Mが 2 入力組合せ回路要素の場合は，それぞれの入力端子の先祖を辿ったときに最初に現れる組合せ回路要素 M_a, M_bの双方) が既に制御経路の決定している組合せ回路要素のとき，Mを対象とする．(便宜上，外部入力は，制御経路を決定済み組合せ回路要素とみなす．) この場合，制御経路を実現するためにマルチプレクサの付加は必要ない．このような組合せ回路要素が存在しない場合は，M_a, M_bの一方だけが既に制御経路が決定済みの 2 入力組合せ回路要素 Mが存在し，Mを対象とする．この条件を満たす組合せ回路要素が複数ある場合には，データパスグラフにおいて，Mから M'へ到達可能な場合，Mを優先する．さらに，互いに到達可能，あるいは互いに到達不可能の組合せ回路要素に関しては，マルチプレクサを優先する．

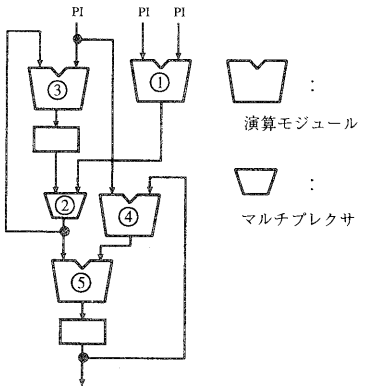
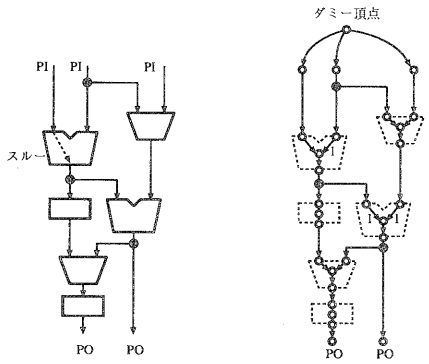


図 4: 制御経路の決定順



演算モジュールに対するスルー機能のコストはすべて1とした．
コストを記入していない辺のコストはすべて0．

図 5: データパスグラフのコスト

図 4にこの手続きの適用例を示す．組合せ回路要素に付加した番号は選択された順番を示す．

このように優先度を決めることにより，外部入力に近い組合せ回路要素から順に制御経路が決定されることとなり，スルー機能を共有しやすくなるので，ハードウェア・オーバーヘッドが低くなることが期待できる．

制御経路の決定方法：組合せ回路要素 Mに対して，最小の付加ハードウェアで実現できる制御経路を以下のように決定する．

データパスグラフに全ての外部入力への辺を持つダミー頂点を付加する．各演算モジュール u の各入力端子 x に対し，辺 $(x,u) \in A_2$のコストを，x の値を u の出力に伝搬するスルー機能を実現するのに必要な付加ハードウェア量 (既にスルー機能が付加済のときはコスト 0 とする) とし，他のすべての辺のコストを 0 とする．また，すべての辺の容量を 1 とする (図 5)．

このグラフにおいて，ダミー頂点を始点，Mを終点とする最小費用流問題を解くことにより，最小の付加ハードウェアで実現できる制御経路を決定する．つまり，M が 1 入力組合せ回路要素のときは，流量 1 の最小費用流が Mの制御経路を表す．また，Mが 2 入力組合せ回路要素のときは流量 2 の最小費用流が Mの 2 つの制御経路を表す．このとき，データパスグラフの作り方から，この 2 つの制御経路は共通部分をもたない．

上記の最小費用流問題において，2 入力組合せ回路要素 Mに対し，流量 2 のフローが存在しないことがある．この場合，Mに対して互いに共通部分を持たない 2 つの制御経路が存在しないということであり，マルチプレクサを付加することにより，新たな経路を付加する．マルチプレクサの付加は前処理と同様に行う．

マルチプレクサの付加：Mの入力端子を x,yとすると，x の直前にマルチプレクサを挿入し，マルチプレクサのもう一方の入力端子は外部入力に接続する．

4.2.2 観測経路の決定

観測経路は，各組合せ回路要素 Mに対して，制御経路と共通部分を持たず，最小の付加ハードウェアで実現できる観測経路を求める．ただし組合せ回路要素 Mが出力端子を持たない観測モジュールの場合，観測経路は空経路とする．制御経路の場合と同様に，観測経路を求める組合せ回路要素は一つづつ処理していく．ただし，外部出力に近い組合せ回路要素から順に観測経路を決定し，DFT 要素を付加する．具体的には，以下の方法で対象とする組合せ回路要素を決定し，その組合せ回路要素に対してデータパスグラフ上で最小費用流問題を解くことで，最小の付加ハードウェアを実現できる観測経路を決定する．

対象とする組合せ回路要素の決定方法：組合せ回路要素 Mの出力端子 ot の子孫を辿ったときに最初に現れる組合せ回路要素の集合を $\mathcal{M}$ とする．$\mathcal{M}$ に既に観測経路の決定している組合せ回路要素が含まれるとき，対象とする組合せ回路要素を Mとする．ただし，この条件を満たす組合せ回路要素が複数ある場合には，$\mathcal{M}$ が観測経路を決定済の MUX，もしくは観測経路決定済の演算モジュールで，otから辿った入力端子にスルー機能が付加されているものを優先する．さらに，そのような組合せ回路要素が複数存在する場合には，組合せ回路要素 M' から組合せ回路要素 Mに到達可能な場合には Mを優先する．

図 6にこの手続きの適用例を示す．組合せ回路要素に付加された番号は選択された順番を示す．

観測経路の決定方法：組合せ回路要素 Mに対して，最小の付加ハードウェアで実現できる観測経路を以下のように決定する．データパスグラフに全ての外部出力からの辺を持つダミー頂点を付加する．さらに Mの制御経

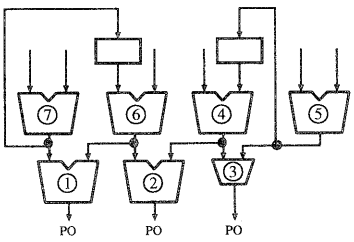


図 6: 観測経路の決定順

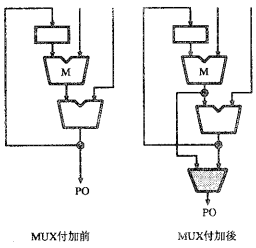


図 7: MUX の挿入

路に現れるすべての辺を取り去る．こうして得られたグラフに，制御経路を求めるときと同様のコスト，容量を与える (既に付加済のスルー機能のついてはコスト 0 とすることに注意)．このグラフで，Mを始点，ダミー頂点を終点とする流量 1 の最小費用流が，Mの観測経路を表す．

上記の最小費用流問題において，流量 1 のフローが存在しないことがある．この場合，Mの制御経路と共通部分を持たない観測経路が存在しないということであり，マルチプレクサを付加することにより，新たな経路を付加する．マルチプレクサの付加は以下のように行う (図 7)．

マルチプレクサの付加：対象となる組合せ回路要素 M の出力端子を z，任意の外部出力を d とする．d の直前にマルチプレクサを挿入し，他方の入力に zを接続する．

このマルチプレクサの制御経路は，前述の制御経路の決定方法に従う．

5 適用例

従来手法として文献 [4] による方法，および提案手法 (単一制御可検査テスト容易化設計法) を，テスト容易化設計後のハードウェア・オーバーヘッドについて比較した．実験に使用した RTL ベンチマーク回路は，GCD と Paulin，および 3rd Lattice Wave Filter(LWF) と 4th IIR である．これらのベンチマーク回路の回路特性を表 1に示す．「circuit」は回路名を表し，「#PI」，「#PO」，「#Reg.」，「#MUX」，「#OP」はそれぞれ外部入力数，外部出力数，レジスタ数，マルチプレクサ数，演算モジュール数を表す．ハードウェア・オーバーヘッドの算出に必要な回路面積は，論理合成ツール「AutoLogicII(Mentor Graphics)」および ALTERA 社の論理合成ライブラリを用いて求めた．表 1の Area にデータパスのビット幅が 16 ビットの場合の回路面積 (gate equivalent) を示す．

表 1: データパスの回路特性

circuit	GCD	Paulin	LWF	4th IIR
#PI	2	2	2	1
#PO	1	2	2	1
#Reg.	3	7	5	12
#MUX	4	11	5	3
#OP	1	4	3	5
Area	530.6	3818.9	735.0	1728.3

各手法でのテスト容易化設計に伴う付加回路によるハードウェア・オーバーヘッドを表 2に示す．

ただし，従来手法，提案手法とも全ての外部入出力に対してテストパターン生成器および応答解析器を付加するので，外部入出力で発生するハードウェア・オーバーヘッドは等しい．よって，表 2のハードウェア・オーバーヘッドの項は，外部入出力に付加されるテストパターン生成器および応答解析器を含まないものとした．

ほとんどの事例で，提案手法のハードウェア・オーバーヘッドは従来手法より低いことが分かった．

6 むすび

レジスタ転送レベルデータパスの組込み自己テストとして，階層テストに基づく test per clock 方式の BIST を提案した．提案方式は実動作速度でのテストが可能な BIST 方式であり，低いハードウェア・オーバーヘッドでデータパス上の組合せ回路要素に対して高い故障検出率を得ることができる．

また，実験によって，提案手法によって生じるハードウェア・オーバーヘッドは従来手法のものより低いことを確認した．

今後の課題としては，テストパターン数と故障検出率の関係について調べることがあげられる．従来手法では，複数の組合せ回路要素で構成される無閉路な部分回路に対してランダムパターンによるテストを実行するが，提案手法では，各組合せ回路要素毎にランダムパターンによるテストを実行する．よって，従来手法よりも高い故障検出率が得られるものと期待される．

謝辞 本研究に際し，多くの貴重な意見を頂いた広島市立大学の井上智生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，(株) 半導体理工学研究センター (STARC) との共同研究，および，文部省科学技術研究費補助金・基盤研究 B(2)(代表者：

表 2: ハードウェア・オーバーヘッド (HW/OH)

		従来手法 [4]			提案手法			
circuit	bit width	HW/OH(%)	#BILBO	#CBILBO	HW/OH(%)	#MUX	#THRU	#LFSR
GCD	8	89.30	1	1	28.00	2	1	0
	16	74.54			27.53			
	32	66.67			27.09			
Paulin	8	80.74	0	6	28.34	10	7	0
	16	40.25			16.65			
	32	20.07			9.14			
LWF	8	43.23	0	1	35.10	4	3	0
	16	34.86			33.32			
	32	30.87			32.47			
4th IIR	8	41.47	0	2	35.89	5	6	1
	16	29.65			28.87			
	32	25.70			26.89			

藤原秀雄，課題番号 09480054) の研究助成による．

参考文献

[1] P. H. Bardell, W. H. McAnney, J. Savir:"Built-In Test for VLSI:Pseudorandom Techniques," Wiley-Interscience (1987).

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

[3] 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).

[4] A. P. Stroele, H. -J. Wunderlich:"Hardware-optimal test register insertion", IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems, Vol. 17, No. 6, pp.531-539 (1998).

[5] 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).

[6] 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).

[7] I. Ghosh, N. K. Jha, S. Bhawmik:"A BIST scheme for RTL controller-data paths based on symbolic testability analysis," Proc. Design Automation Conf., pp.554-559 (1998).

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