

衝突回避による固定制御可検査性に基づくテスト容易化設計法の改良について

An Improvement of a DFT Method Based on Fixed-control Testability by Conflict Avoidance

川原 侑大[†]
Yuudai Kawahara市原 英行[‡]
Hideyuki Ichihara井上 智生[‡]
Tomoo Inoue

あらまし

大規模集積回路に対するテスト生成を効率よく行う方法として、階層テスト生成 [1] がある。本論文では、階層テスト容易化設計法の 1 つである、固定制御可検査性に基づくテスト容易化設計法 [4] の改良法を提案する。タイミング衝突 (1 つの外部入力から 1 つの回路要素の異なる 2 つの入力までの制御経路の順序深度が等しいこと) の少ない制御林を生成するためのアルゴリズムを用いることで、回避するために必要な回路要素の付加を削減できる。さらに、提案手法では付加したバイパス MUX を用いたテストプラン長の削減について考察する。テストプラン長を削減することで、テスト実行時間の削減が期待できる。また、計算機実験では、ベンチマーク回路を用いて従来法との比較を行い、提案手法の有効性を示す。

1. はじめに

今日の大規模集積回路 (VLSI) に対するテスト費用の削減は重要な課題の 1 つである。テスト生成に要する費用を削減するためのテスト容易化設計法として、完全スキャン設計が広く用いられている。しかし、完全スキャン設計は膨大なテスト実行時間を要し、フリップフロップをスキャンフリップフロップに置き換えるためのハードウェアオーバーヘッドが大きいという問題がある。一方、論理合成技術の発展により、今日の VLSI 設計は、レジスタ転送レベル (RTL) で行うことが一般的になってきた。RTL のデータパス部に対するテスト生成法として階層テスト生成 [1] がある。階層テスト生成は、大きく 2 つのステップから成る: (1) データパスを構成する各回路要素に対するテスト生成、(2) 各回路要素に対して (1) で生成されたテストパターンを、外部入力から回路要素の入力に伝達し、その出力応答を回路要素の出力から外部出力まで伝達するための制御系列 (テストプラン) の生成。

上述した階層テスト生成は、ゲートレベル上のテスト生成に比べて、短い時間でテスト生成をすることが期待できる。また、完全スキャン設計と比較して、短い時間でテスト実行が可能である。しかし、テストプラン生成は、回路の構造によっては多くの時間を要する場合がある。そこで、テストプラン生成を容易に行うためのテスト容易化設計法 [3] が提案されている。

文献 [3] では、テストプラン生成が容易に行える RTL データパスの性質として、強可検査性を示している。この手法は、データパスが強可検査となるように演算モジュールのスルー機能やレジスタのホールド制御を付加する。しかし、ホールド制御を付加することは、データパスのハードウェアオーバーヘッドの増大や、レジスタ制御の増加によるテストプラン生成回路の面積の増大を引き起こす。そこで、テストプラン生成におけるハードウェアオーバーヘッドの削減を目的とした手法 [4]-[5] が提案されている。

文献 [4] は、テストプラン生成回路の面積の削減が可能な手法である。強可検査性 [3] を満たすデータパスが、さらに固定可制御可検査性 [4] を満たすとき、そのデータパ

スに対するテストプランは簡単になるため、テストプラン生成回路の縮小に効果的である。そこで、文献 [4] ではデータパスが固定制御可検査性を満たすようにスルー機能やバイパス回路を付加する。以下、この手法のことを FXDFT と呼ぶ。

一方、文献 [5] は、レジスタのホールド制御の削減が可能な手法である。強可検査性に基づくテスト容易化設計法 [3] の一部である制御林生成アルゴリズムの問題点に着目し、これを改良することでテストプラン実行時に必要となるレジスタのホールド制御を削減できる。以下、この手法のことを HODFT と呼ぶ。

本論文では、HODFT を応用した、固定制御可検査性に基づくテスト容易化設計法 [4] の改良法を示す。HODFT のアルゴリズムと同様に 1 つの回路要素への制御経路の順序深度ができるだけ異なるように制御林を生成する。これは、データパスを固定制御可検査にするために必要なバイパス回路の付加が削減できることを意味する。結果として、データパスのハードウェアオーバーヘッドを削減しながらテストプラン生成回路の規模を削減できるものと期待できる。また、付加したバイパス回路を利用して、テストプラン実行時に衝突を発生することなくテストプラン長を削減できることを示す。

計算機実験では、提案手法がデータパス及びテストプラン生成回路のハードウェアオーバーヘッドの削減に効果的であること、及びテストプラン長を削減できることを示す。

2. 背景

2.1 固定制御可検査性に基づくテスト容易化設計法

RTL データパスは演算モジュール、レジスタ、マルチプレクサから成る。階層テスト生成のステップ 1 で生成されるテストパターンに依存せず、効率良くテストプラン生成を行える手法として、強可検査性に基づくテスト容易化設計法 [3] が提案されている。データパス中の全ての回路要素に対して以下の 2 点を満たすとき、そのデータパスは強可検査であるという。

- 強可制御性: 回路要素の入力端子に対して外部入力から任意の値を伝達可能
- 強可観測性: 回路要素の出力端子のとりうる任意の値を外部出力まで伝達可能

文献 [3] では、この定義を満たすように (1) 演算器に対するスルー機能: 他の入力に依存せず、入力を出力へ通過 (スルー) させ、(2) レジスタに対するホールド機能: 任意の時間、値を保持可能にする、ための 2 種類の DFT 要素を追加する。これらの DFT 要素を付加することで、データパス中の全回路要素に対するテストプランを生成できる。

データパス中の全ての回路要素に対するテストプランの発生を容易にするために、データパス・コントローラ間にテストコントローラを付加する [2]。テストプランはテストコントローラ内のテストプラン生成回路で生成する。文献 [3] で追加するホールド制御は、テストパターンやその出力応答を伝搬させるための制御系列が複雑化し、それを発生するテストプラン生成回路の規模の増大を招く。

そこで、文献 [4] では、テストプラン生成回路の規模を小さくするデータパスの性質として固定制御可検査性を

[†]広島市立大学大学院 情報科学研究科[‡]広島市立大学 情報科学部

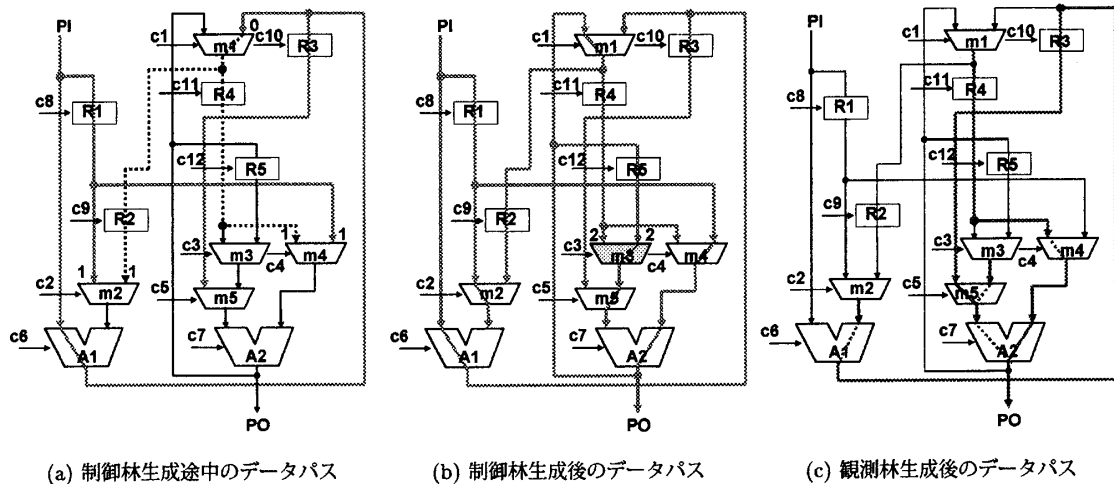


図1: 制御林・観測林の生成

定義している。固定制御可検査性を満たすデータパスは、テストプランが簡単になるため、付加するテストプラン生成回路の面積の縮小に有効である。FXDFTでは、テストプランを以下の3つのフェーズに分類する。

正当化フェーズ 外部入力から、テストの対象となる回路要素 M の直前のレジスタ $R(CP)$ までの経路を活性化化する制御系列

テストフェーズ $R(CP)$ から M の入力へ正当化し、 M の出力応答を M の直後のレジスタ $R(OP)$ まで伝搬する制御系列

伝搬フェーズ $R(OP)$ から外部出力への経路を活性化化する制御系列

FXDFTでは、データパスが固定制御可検査性であるための条件として、以下の2つを定義している。

- データパスは強可検査である。
- データパス中の各回路要素に対するテストプランの正当化/テスト/伝搬の各フェーズは、各々1個の制御ベクトルで構成できる。

データパスがこの性質を満たすように(1)スルー機能、(2)バイパス回路の2つを追加する。文献[3]では、タイミング衝突(以下、衝突と呼ぶ)が発生した際は、ホールド制御を追加することで解決するが、データパスが固定制御可検査性を満たすようにするには、テストプランの各フェーズは固定でなければならないため、ホールド制御は追加できない。そこで、FXDFTでは、バイパス回路を付加することによって衝突を解消する。

3. 衝突数削減を指向した固定制御可検査データパスの設計

前節では、固定制御可検査性に基づくテスト容易化設計法[4]について説明したが、この手法では衝突を回避するためにバイパス回路を付加しなければならない。したがって、データパスのハードウェアオーバーヘッドを増加させる可能性がある。一方、文献[3]の制御林生成アルゴリズムにおける問題点に着目し、改良した手法HODFTがある。HODFTは、各回路要素までの制御経路の順序深度ができるだけ異なるように制御林を生成するため、衝突を削減することができる。衝突数の削減は、データパスを固定制御可検査にするために必要なバイパス回路の付加を削減できることを意味する。そこで、本論文では衝突を削減することが可能な手法であるHODFTを応用した固定制御可検査

データパスの設計法を提案する。さらに、本論文では付加したバイパスMUXを用いて、テストプラン実行時に衝突が発生することなく、各回路要素のテストプラン長を短縮できることを示す。提案手法は以下のステップで行う。

1. 制御林の生成
2. 観測林の生成
3. DFT要素の追加
4. テストプラン長の削減

ステップ1で制御林を生成する。制御林とは、外部入力からデータパス中の各回路要素の入力までの制御経路(木)の集合(林)である。また、制御林を用いて全ての回路要素の正当化フェーズを実行できる。ステップ2で観測林を生成する。観測林とは、各回路要素から外部出力までの観測経路の集合である。この観測林を用いて、全ての回路要素に対する伝搬フェーズを実行できる。ステップ3では、ステップ1, 2で生成した経路を用いてテストプランを実行するために必要なDFT要素を付加する。そして、ステップ4では付加したDFT要素を利用してテストプラン長の短縮を行う。

3.1 制御林の生成

HODFTと同様の方法で制御林を生成する。データパス中のMUX・演算器を頂点、信号線を辺とする。外部入力を始点として順序深度に基づく幅優先探索を行い、外部入力に近い回路要素から制御経路を探索・決定する。回路要素の入力から出力へ通過する経路を決定する際に、その先の回路要素で衝突が発生するかどうかをチェックする。

- 出力先の回路要素の入力で衝突が発生しない場合、入力から出力への経路を決定(スルーの決定)する。
- 衝突が発生するとき、決定を延期する。他の入力からの経路も調べる。
- いずれの入力からの経路も、出力先で衝突が起これるときは、最小の衝突数となる経路を選択する。

図1(a)を用いて制御林の生成を説明する。この図は制御林生成の途中の様子である。図中の回路要素の入力に示した数字は順序深度(外部入力から回路要素の入力までに、テストパターンを伝達するのに要する時間)を示す。この時点では、外部入力から $m1$ の右入力までの経路は決定済みである。次に、 $m1$ のどちらかの入力を $m1$ の出力へスルーするかを決定しなければならない。右入力までの経路はすでに決定しているので、 $m1$ の右入力をスルーした場合について考える。右入力をスルーした場合、その先

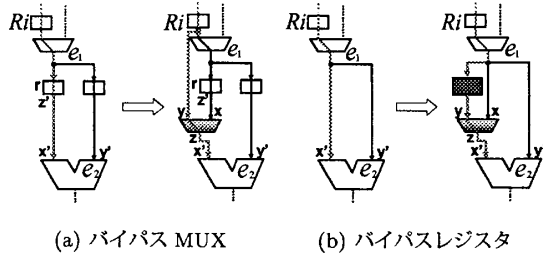


図2: バイパス回路

の経路は点線で示した通りである。この場合、 $m2$ と $m4$ で衝突が発生することがわかる。衝突数が2となるため、 $m1$ のスルーの決定は延期する。探索を続けると、 $m1$ の左入力をスルーすると衝突が $m3$ の1個となることがわかる。(この時、 $m3$ の右・左入力の順序深度は互いに2である。(図1(b))) 従って、 $m1$ は左入力をスルーする。以上の方法で制御林を生成すると図1(b)のようになる。

3.2 観測林の生成

生成した観測林を図1(c)に示す。観測林は文献[3]-[5]と同様の方法で生成する。次節で述べるDFT要素を削減するために、3.1で生成した制御林との共有部分ができるだけ多くなるように観測林を生成する。

3.3 DFT要素の追加

求めた制御林と観測林に基づいて、データパス中の各回路要素ごとに、テストプランを求める。このとき、テストプランを実現するために必要なDFT要素を求めて追加する。提案手法においてDFT要素は、FXDFTと同様にスルー機能とバイパス回路を追加する。制御経路上で演算モジュールを通過する必要がある際は、スルー機能を追加する(図1(b)のA1,A2)。また、衝突を回避するために、バイパス回路(バイパスMUXもしくはバイパスレジスタ)を付加する。付加するバイパス回路を図2に示す。バイパスMUXは、制御経路の順序深度を1減らすことで衝突を解消する(図2(a))。制御経路上に独立したレジスタがない場合は、バイパスMUXを付加できないため、バイパスレジスタを付加する(図2(b))。バイパスレジスタは制御経路の順序深度を1増やすことで衝突を解消する。

図1に示したデータパスにDFT要素を追加した様子を図3(a)に示す。この例では、 $m3$ の制御経路で衝突が発生しているが、衝突を解消するために外部入力から $m3$ の右入力までの経路上にバイパスMUXを付加する。この結果、 $m3$ の右入力の順序深度を1減らすことができるため、衝突を解消できる。一方、図3(b)はFXDFTの手法を用いた際のデータパスの様子である。 $m2$, $m4$, $A2$ の制御経路上で発生した衝突を解消するために $tM1$ を付加し、 $m5$ の衝突を解消するために $tM2$ を付加する。以上より、提案手法は制御林生成の工夫によって、FXDFTに比べて衝突回避に用いるバイパス回路を削減できることがわかる。

3.4 バイパスMUXを用いたテストプラン長削減

前節で述べたバイパスMUXは、基本的に制御・観測経路上で衝突を起こす各回路要素に対して、その衝突を回避するために追加される。そのため、経路上で衝突を起こさない他の回路要素の経路上では使用しないことが前提であった[4]。しかし、バイパスMUXを使用しないテストプランの時も、経路上にバイパスMUXが存在すれば、このバイパスMUXを適切に用いてテストプランを短縮できる時がある。そこで本節では、付加したバイパスMUXを利用したテストプラン長の削減について考察する。

バイパスMUXを利用して、テストプラン長を削減できることを、図4を用いて説明する。図中の $tM1-tM4$ はバ

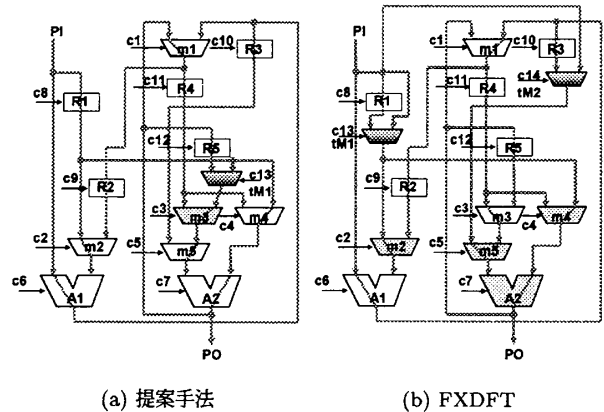


図3: DFT適用後のデータパス

イパスMUX、 $M1$, $M2$ は演算モジュール、 $R1-R5$ はレジスタを示す。なお、これらのバイパスMUXは $M1$, $M2$ 以外の回路要素に対する経路の衝突を解消するためのものであり、FXDFTにおいて、 $M1$, $M2$ のテストプランでは使用しない。 $M1$, $M2$ の各入力の隣に記した文字と数字は、その入力に与える値と順序深度を示す。

ここでは、 $M1$ のテストプラン長削減について説明する。図4の太線は $M1$ のテストプラン実行時に使用する経路を示す。 PI から $M1$ までの各経路は制御経路である。一方、 $M1$ の出力応答 z を PO に伝搬するために、 $M2$ の左入力から $M2$ の出力へ値をスルーするための値 c を $M2$ の右入力に入力する。この時、 PI から $M2$ に c を与えるための経路を補助経路と呼ぶ。

ここで、 $M1$ のテストプラン実行時の入出力系列を表1に示す。FXDFTでは、時刻0,1,2に PI より a, b, c を入力し、テストプラン長は5となる。表中の $tM1, tM3, tM4$ は、これらのバイパスMUXによって経路をバイパスした際の入出力系列を示す。 $tM1, tM3, tM4$ を用いたときは、時刻2から各入力を開始すればよく、これはテストプラン長を3に短縮できたことを意味する。また、全てのバイパスMUXを用いたときは、 $M1$ の両入力の順序深度が0となる。これは、 $M1$ に a と b を同時に入力できないことを意味する。このように、使用するバイパスMUXの組合せによっては、テストプランを実行できなくなる場合がある。したがって、テストプランが実行できなくなる場合を除いて、経路上にあるバイパスMUXを適切に選択して利用することでテストプランを削減できる。この手法では、既に付加しているバイパスMUXを用いるため、新たなハードウェアオーバーヘッドはない。

ここで述べたアイデアは、前述3.3のDFT要素の追加を終えた後に適用する。各回路要素のテストプランについて、バイパスMUXを利用したプラン長の削減の可能性を

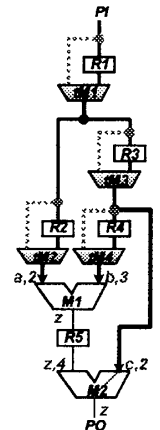


図4: プラン短縮の例

表1: 入出力系列

time	FXDFT		$tM1, tM3, tM4$	
	PI	PO	PI	PO
0		b		
1	a			
2	c		a	
3			b	
4		z	c	z

表 2: ベンチマーク回路

circuits	PIs	POs	MUXs	ops.	regs.
4thIIR	1	1	3	5	12
Dct_f	4	7	26	40	11
Gcd	2	1	4	4	3
Jwfl	1	1	25	3	14
Jwf2	5	5	25	3	14
Lwfl	1	1	6	3	3
Lwf2	2	2	5	3	5
Mpeg	7	16	207	161	241
Paulin	2	2	11	4	7

調べ、可能ならばテストプランを再構成する。

4. 実験結果

4.1 実験方法

提案手法の有効性を確認するため、ベンチマーク回路を用いて計算機実験を行った。(計算機:SUN Blade2000 ワークステーション) 実験では、従来法 FXDFT[4], HODFT[5] で DFT した結果と、提案手法を用いて DFT した結果を比較した。実験に用いた 9 個のベンチマーク回路の回路特性を表 2 に示す。左から順に、それぞれの回路の外部入力数、外部出力数、マルチプレクサ数、演算器数、レジスタ数である。

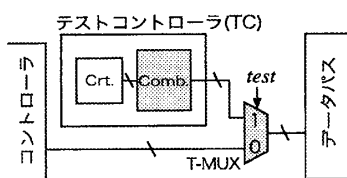


図 5: テストコントローラのモデル

図 5 に示すようなモデルを用いて行った。テストコントローラ (TC) とコントローラからの制御信号線を選択する T-MUX が図 5 のように挿入されている。T-MUX の面積は 2 入力 1 出力の MUX を 7 として算出している。

4.2 実験結果

実験結果を表 3 に示す。FX, HO はそれぞれ手法 FXDFT, HODFT を、ours は提案手法を示す。表中の「衝突数」はタイミング衝突数を示し、「DFT 要素」は衝突の解消に用いるバイパス回路の付加数を示す。なお、数字に * を記したものは、ホールド制御の付加数を示す。HODFT ではバイパス MUX の代わりにホールド制御を付加するため、この付加数を表記した。「テストプラン長」は、データパスを構成する各回路要素のテストプランの合計を示す。また、「テスト実行サイクル」は回路要素ごとにテストパターンを求め、その回路要素のテストプラン長を乗算することで求めた。「TPG の面積」は、図 5 に示したとおり、テストコントローラ内のテストプラン生成回路及び T-MUX の面積の合計を示す。提案手法の括弧をつけた数字は 3.4 の手法を用いない時の数字を示す。数字が 1 つしかない欄は、3.4 を適用しても同じ結果であったものである。

提案手法は FXDFT と比較して、衝突数が削減できていることがわかる。この結果、衝突の解消に用いる DFT 要素が FXDFT より削減できる。(HODFT とは追加する DFT 要素の種類が異なるため比較できない。)

テスト実行サイクルについて考察する。提案手法は、FX, HO と比較して、ほとんどの回路でテスト実行サイクルを削減している。しかし、Jwfl のテスト実行サイクルが FX より増加した。これは、制御生成時に経路決定の延期が多く、各回路要素ごとの順序深度が増え、テストプラン

表 3: 実験結果

回路	手法	衝突	DFT 要素	テストプラン長	テスト実行サイクル	TPG 面積
4thIIR	FX	1	1	67	4121	55
	HO	0	*1	62	4136	63
	ours	0	0	61	4113	33
Dct_f	FX	4	4	180		372
	HO	2	*2	164	n.a.	361
	ours	2	2	176		342
Gcd	FX	0	0	26	1918	7
	HO	0	0	26	1918	19
	ours	0	0	26	1918	7
Jwfl	FX	13	4	126	3433	353
	HO	5	*7	141	3913	411
	ours	5	2	133	3548	363
				(142)	(3936)	(320)
Jwf2	FX	4	3	97	2751	296
	HO	0	0	92	2636	282
	ours	0	0	92	2636	257
Lwfl	FX	2	1	33	1164	109
	HO	0	*1	33	1197	101
	ours	0	0	32	1151	83
Lwf2	FX	1	1	28	1169	42
	HO	1	*1	27	1151	47
	ours	1	1	28	1169	42
Mpeg	FX	0	0	10889		2066
	HO	0	0	11005	n.a.	2017
	ours	0	0	11005		2062
Paulin	FX	2	2	74	2871	135
	HO	1	*1	68	2762	140
	ours	1	1	71	2808	119

長が増加したことが原因であると思われる。また、Lwf2, Paulin のテスト実行サイクルは HO より増加した。これは、ホールドの使用に伴うテストプラン長の増加に比べて、付加したバイパス MUX のテストを行うためのテストプラン数の増加が大きいことが原因であると思われる。

さらに、提案手法は HODFT と比較して TPG の面積が削減できている。これより、固定制御可検査性の有効性が確認できる。また、FXDFT と比べて Jwfl を除く全ての回路で TPG の規模を縮小できることがわかる。これは、データパス中の付加 DFT 要素を削減したことによって、テストプランが簡単になったためだと考えられる。

また、節 3.4 で示したテストプラン長削減の効果は、Jwfl に対する結果に示されている。

5. まとめと今後の課題

本論文では、固定制御可検査性において、タイミング衝突をできるだけ回避することで、より小さいオーバーヘッドでテストプラン生成を実現する手法を提案した。計算機実験を行い、従来法と提案手法を比較した。提案手法は、データパスで発生するタイミング衝突を削減できるため、回避に必要なバイパス MUX の付加数を削減できる。また、テストプラン生成回路の面積も削減できる。

今後は、RTL 回路全体の面積求めた上で提案手法を評価することが課題として挙げられる。

参考文献

- [1] B.T. Murray and J.P. Hayes, "Hierarchical test generation using pre computed tests for modules," IEEE Trans. Comput.-Aided Des., Integrated Circuits & Syst., vol.9, no.6, pp.594-603, June 1990.
- [2] I.Ghosh, A. Raghunath and N.K. Jha, "Design for hierarchical testability of RTL circuit obtained by behavioral synthesis," Proc. Int. Conf. on Computer Design, pp.173-179, 1995.
- [3] 和田弘樹, 増澤利光, K.K. Saluja, 藤原秀雄, "完全故障検出効率を確保するレジスタ転送レベルデータパスの非スキャンテスト容易化設計法", 信学論 (D-1), vol.J82-D-1, no.7, pp.843-851, July 1999
- [4] 永井慎太郎, 和田弘樹, 大竹哲史, 藤原秀雄, "固定制御可検査性に基づく RTL 回路の非スキャンテスト容易化設計法", 信学論 (D-1), vol.J84-D-1, no.5, pp.454-465, 2001.
- [5] 岡本直己, 市原英行, 井上智生, 細川利典, 藤原秀雄, "強可検査性に基づくデータパスのテストプラン生成アルゴリズムの改良について", 電子情報通信学会技術研究報告, Vol. 103, No. 476, pp. 13-18, 2003.