

機能的時間展開モデルを用いたテスト生成

日大生産工(院) ○杉木 一也 日大生産工 細川利典
九大 吉村正義

1 まえがき

LSI(Large Scale Integrate circuit)の高集積化・大規模化が進むにつれて、テスト生成が困難になってきており、高い故障検出効率の達成を保障するためには何らかのテスト容易化設計(Design For Testability:DFT)を施すことが必要である。

現在、LSIのテスト容易化設計技術で広く取り入れられているのがスキャン設計[1,2]である。しかしながら、ハードウェアオーバーヘッドが大きい、テスト実行時間が長いという欠点がある。また、さまざまな電化製品に組み込まれる暗号回路に対してスキャンチェーンを用いた攻撃が可能であることが報告されている[3,4]。その他にIP(Intellectual Property)コアに対してもスキャンチェーンを用いた回路の解析が行われる可能性がある。そのために、スキャン設計せずにテストパターンを生成し、高い故障検出効率を達成することが、ハードウェアオーバーヘッド、テストコスト、セキュリティの安全性において最良の解決策である。しかしながら、一般にスキャン設計をしない順序回路に対するテスト生成は困難である。

時間展開を用いた一般的な順序回路のテスト生成[5,6]は対象の故障を励起し、故障の影響が外部出力まで伝搬されるまで、回路の入力から出力側へ時間展開を繰り返す。故障の影響が外部出力へ伝搬されると、次に故障を励起・伝播するために割り当てた内部の信号線の値を正当化するために後方追跡や一致操作を行う。これらの処理は正当化のための要求が全て外部入力の値の割り当てによって充足されるまで時間展開を回路の出力側から入力側へと繰り返す。また、これらの処理の過程で信号線への値の割り当てに対して決定木を生成しながら処理を行い、必要な値の割り当てにおいて矛盾が発生した場合は、バックトラックを行う。そして、バックトラック回数が規定の値を越えるとその故障に対するテスト生成を打ち切る。一般に順序回路はフィードバックループを持ち、単純に回路構造にのみ注目してフィードバックループの時間展開を行うと展開時刻数の規定値を超えるまで、際限無く時間展開を繰り返す。

近年、LSI設計はさらなる設計生産性の向上のためにレジスタ転送レベル(Register Transfer Level:RTL)回路の記述より抽象度の高い動作記述を動作合成[7]を実行することによって RTL回路を生成する方法が取り入れられている。このようにして生成されたRTL回路はデータパス部とコントローラ部から構成された回路として生成できる。そして、このようなRTL回路のテストのためにデータパスとコントローラをそれぞれ分

離して、それぞれ適したテスト容易化設計[8]を行いテストする手法が提案されている。このうちデータパスのテスト手法[9]はデータパス内部のモジュール単位でテストを行う階層テスト[10]を行う方法に基づいている。ただし、これらの手法は回路のビット幅が均一であることが条件とされ、適用可能な回路の範囲が狭い。また、テスト容易化の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。

設計者が動作記述を用いて設計するにあたり、回路に実装するアルゴリズムやサイクル数を決定する。その後、動作合成によって生成されたRTL回路の機能検証時に設計情報をもとに検証に必要なテストパターン作成する。この機能検証パターンを利用することで、機能動作が考慮されたデータパス回路への制御信号線の入力系列や状態信号線の出力系列およびサイクル数を抽出できる。本論文では、これら機能情報を利用して通常動作に基づいてデータパス回路のテスト生成を行うモデルである機能的時間展開モデルとそのテスト生成法を提案する。

2 順序回路の機能動作解析

例えば、コントローラとデータパスから構成される順序回路の2つの動作について機能検証パターンを用いて、検証を行った結果、表1に示すような制御信号線の系列と状態信号線の系列が得られたとする。図1はコントローラとデータパス間の制御信号線の系列と状態信号の系列の例を示している。それぞれの動作を行った時に各時刻でデータパスの制御信号端子A,B,Cの入力系列と状態信号端子P,Qの出力系列が示されている。たとえば、表1の動作1の1時刻目の制御信号線

表1：RTL回路の動作仕様

		動作仕様				
		端子	時刻			
動作 1	A	0	0	0	0	0
	B	0	0	1	1	1
	C	0	1	0	0	0
	P	1	1	1	1	1
	Q	1	0	0	0	0
動作 2	A	1	1	0	0	0
	B	0	1	0	0	1
	C	0	1	0	1	1
	P	1	1	1	0	0
	Q	1	1	0	1	1
動作 3	A	1	0	0	0	0
	B	1	0	0	0	1
	C	0	1	0	1	0
	P	1	0	0	0	0
	Q	1	1	1	1	0

Test Generation Using Functional Time Expansion Model

Kazuya SUGIKI, Toshinori HOSOKAWA, and Masayoshi YOSHIMURA

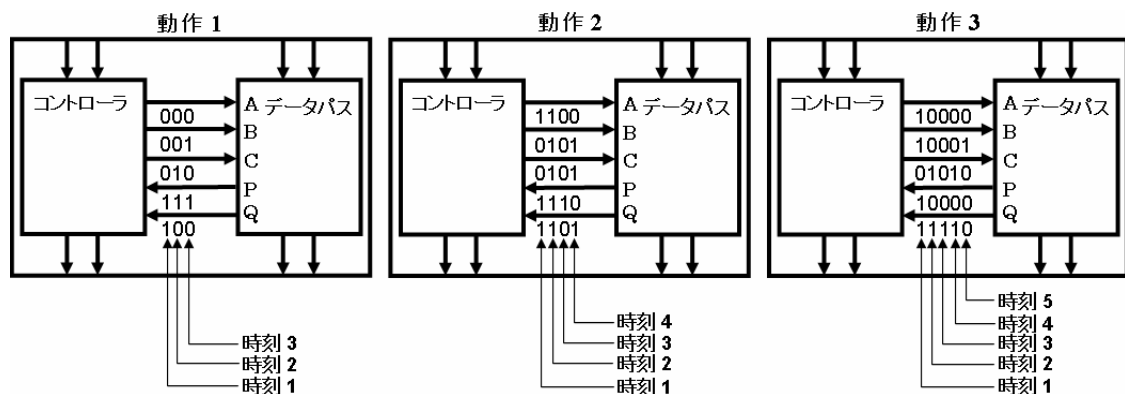


図1：各動作時のコントローラとデータベース間の信号系列

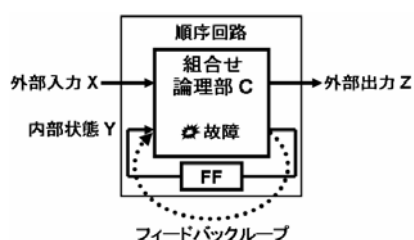


図2：順序回路

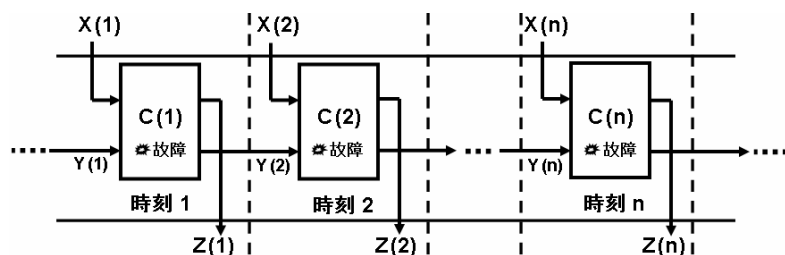


図3：時間展開モデルと多重故障

と状態信号線の値は、 $(A,B,C,P,Q)=(1,0,0,1,1)$ となり、2時刻目では $(A,B,C,P,Q)=(0,0,1,1,0)$ となり、3時刻目では $(A,B,C,P,Q)=(0,1,0,1,0)$ となる。こうして制御系列と状態系列および演算に必要なサイクル数を抽出することができる。

3 機能的時間展開モデル

一般の順序回路は図2に示すようにフィードバックループを持ち、順序回路の自動テスト生成ツール(ATPG: Automatic Test Pattern Generation)内部ではテスト生成時に順序回路のフィードバックループの時間的な接続関係を考慮し、図3に示す連続する組合せ回路となるように時間展開を行う。図3の $C(n)$ は時刻 n での組合せ回路に対応し、 $X(n)$ 、 $Y(n)$ および $Z(n)$ はそれぞれ時刻 n の外部入力、内部状態、外部出力の値を表す。時間展開モデルを用いてテスト生成を考えると、図2に示す順序回路で単一故障であっても、図3に示す時間展開モデルでは同一の組合せ論理部 $C(n)$ が繰り返し現れるため、ATPGは多重故障を扱う必要がある。また、従来の順序ATPGのように回路構造にのみ着目して時間展開モデルを生成した場合、展開時刻数の制約を与えない限り、フィードバックループ部分を際限なく展開することができる。そのために時間展開数が増え、テスト生成が困難になる場合がある。

しかしながら、回路設計における上流工程での機能検証から得られる検証パターンの機能的な情報を利用することで、時間展開モデルの展開時刻数を決定し、データベースの制御信号線の入力系列と状態信号線の出力系列を得ることができる。そして、データベースの時間展開モデルを動作時間分展開し、時間展開モデル上の各時刻の制御信号線と状態信号線に対して抽出した制御信号線の入力系列と状態信号線の出力系列の値を

制約として与える。このモデルを機能的時間展開モデルと呼び、このモデルに対してテスト生成することで機能的な動作環境上でのテスト生成を可能にする。従来、順序回路内部のデータベースのテストを行うためには、制御信号線と状態信号線の値を外部の入力から制御を試みながら、テスト生成を行うのに対して、本手法は複雑な計算を必要とせず、制御信号線と状態信号線に機能検証によって抽出した値を割り当てることでデータベース内部の信号線の値を決定できる。その結果、テスト生成は値の割り当てがない残りの外部入力に割り当てる値を決定するだけでよい。

本手法は図4のように対象回路に対して状態信号線を可観測するDFTを施した環境下でテスト生成を行う。図5はゲートレベルのデータベース回路の例を示しており、外部入力はデータ入力 $PI1, PI2$ と制御信号線 A, B, C の5入力と外部出力はデータ出力 $PO1, PO2$ と状態信号線 P, Q の4出力を持つ。図5で示したデータベース回路に表1の動作1を実行させたときの機能的時間展開モデルを生成した例を図6に示す。動作仕様の機能情報をもとに時間展開数3の時間展開モデルを生成し、各時刻で制御信号線と状態信号線に制約を与える。図6において四角で囲われた信号線の値は与えられた制約値によって一意に信号線の値が決定されたことを示す。時刻2のゲート $G4$ の入力信号線の0縮退故障を励起し、時刻3の $PO2$ に故障の影響を伝搬させる場合を考える。機能的な制約がない場合は、故障を時刻2で励起するために、時刻2において、 $PI1, PI2$ のいずれかに1を、または A に0の値割り当てを考慮することができ、3つの選択肢のいずれかを選ばなければならない。一方、機能的な制約がある場合には、時刻2の A に制約値0が与えられているため、故障の励起するための信号線の値割り当ての選択を行う必要がない。このように制御信号線と状態

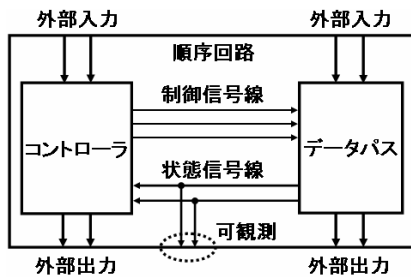


図4：ゲートレベルのデータパス回路例

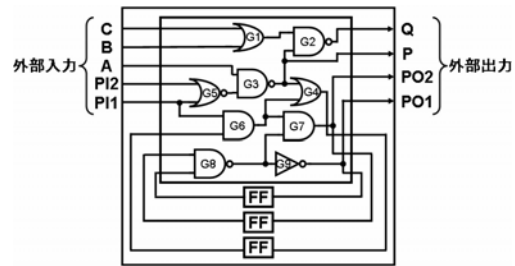


図5：ゲートレベルのデータパス回路例

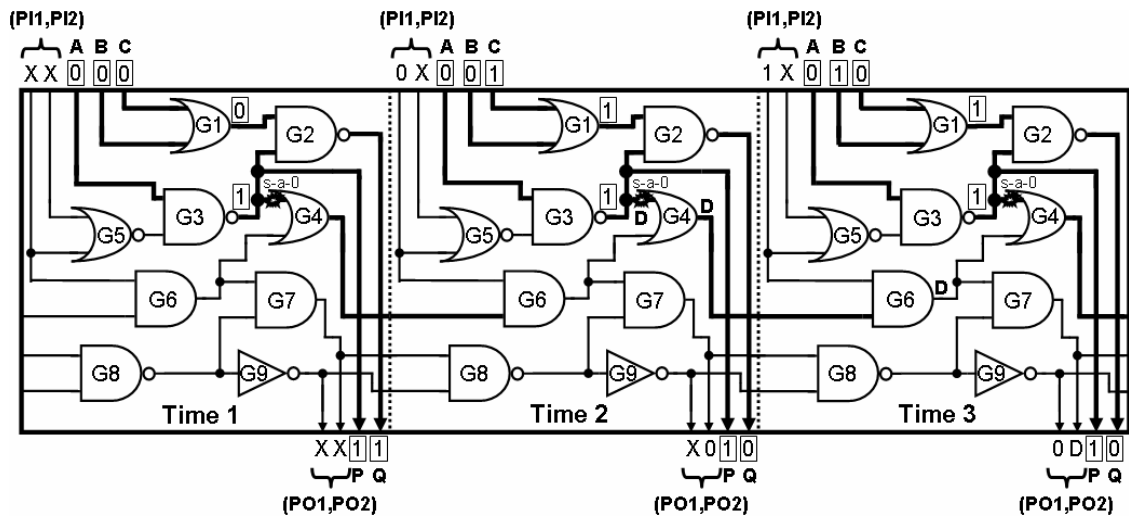


図6：機能的時間展開モデルを用いたテスト生成

信号線に制約を与えることで、値を割り当てる信号線の選択枝を削減することができる。問題を単純にするために回路の初期の内部状態がリセット信号などで初期されているものとして、テスト生成時の解空間の大きさを時間展開モデル中の各時刻の外部入力の値の組合せとして考える。図6の例で機能的時間展開モデルの外部入力の組合せの数は $2^6=64$ 通りになる。それに対して、通常の時間展開モデルの入力の組合せの数は制御信号線が可制御であったとしても $2^{15}=32768$ 通りになる。また、実際にはコントローラを通じて制御信号線の値を決定しなければならない。このことから機能的時間展開モデルのテスト生成の解空間は通常の時間展開モデルのそれと比べて非常に小さくなることがわかる。機能的時間展開モデルを用いてテスト生成を行った場合は、G4の入力信号線の0縮退故障を検出するために時刻2のPI1に0の値割り当てと時刻3のPI1に1の値割り当てを行うだけで、テスト生成を終えることができる。

4 機能的時間展開モデルを用いたテスト生成アルゴリズム

Step1. 機能検証パターンで回路をシミュレーションした時の制御信号線の入力系列と状態信号線の出力系列及び動作サイクル数を機能情報として抽

出する。

Step2. 機能情報からテスト生成時の時間展開数と制約の情報が付加された機能的時間展開モデルを生成し、多重故障を考慮した組合せテスト生成を行う。

Step3. Step2で生成したテストパターンを順序回路のテスト系列に変換する。

5 実験結果

本手法を8ビット除算器に対して適用し、2種類の機能的時間展開モデルを使ってテスト生成を行った。実験はHP Workstation xw4200 (CPU: Pentium4 3.4GHz, メモリ：3GB, OS: Redhat enterprise 4)上で行い、機能的時間展開モデルを用いたテスト生成可能なツールSTAGYを用いた。また、比較対象とする従来法のテスト生成ツールにはSynopsys社のTetraMAXを使った。表2は8ビット除算器のデータで、#GATEがゲート数、#PIが外部入力線数、#POが外部出力線数、#FFがFF数、#controlが制御信号線数、#statusが状態信号線数を表している。表3は提案手法による8bit除算器のテスト生成で利用した2つの機能的時間展開モデルでのテスト生成結果である。モデルは機能的時間展開モデルの種類、CPUtimeはテスト生成時間、DETECTは検出された故障数、FEは故障検出効率を表している。表4は従

来法と提案手法とのテスト生成時間と故障検出効率について結果を比較したもので、CPUtimeはテスト生成時間、FEは故障検出効率を示している。比較すると提案手法はTetraMAXよりも564倍の速さでテスト生成を行い、故障検出効率についても11.25%向上させることができています。ただし、本手法は一部の機能的時間展開モデル(2個)を用いてテスト生成を行っているために、実験において未検出だった故障が他のモデルで検出可能かどうか検証していない。そのために、冗長故障の判定ができないために故障検出効率と故障検出率が等しくなっている。実験結果より提案手法のテスト生成は回路の動作サイクル数と制御信号線と状態信号線の制約をもちいることでテスト生成の解空間を削減し、テスト生成時間を大幅に短縮し、かつ高い故障検出効率を達成することができた。

6 おわりに

本論文では、RTL 設計されたデータパスを論理合成した順序回路に対するテスト生成モデルとして機能的時間展開モデルを提案した。機能的時間展開モデルは、機能検証パターンを利用することで得た動作サイクル数、制御信号線の入力系列、状態信号線の出力系列の情報を組み込んでいる。また機能的時間展開モデルを用いたテスト生成法を提案し、解空間を削減し、テスト生成を高速化させることができた。今後の課題としては、評価する回路を増やすことと、テスト生成により効果的な機能検証パターンの選択などが挙げられる。

「参考文献」

- [1] H.Fujiwara, "Logic Testing and Design for Testability", The MIT Press, 1985.
- [2] M.Abramovici, M.A.Breuer, and D.Friedman, "Digital systems testing and testable design", IEEE Press, 1995
- [3] Bo Yang, Kaijie Wu, Ramesh Karri, "Scan Based Sied Channel Attack on Dedicated Hardware Implementations of Data Encryption Standard," Test Conference, 2004. Proceedings. ITC 2004. International, pp.339-344, 2004
- [4] Bo Yang, Kaijie Wu, Ramesh Karri, "Secure Scan: A Design-for-Test Architecture for Crypto Chips," Annual ACM IEEE Design Automation Conference, pp.135-140, 2005.
- [5] W.T.Cheng, "The back algorithm for sequential test generation," Proc. 1988 IEEE Int. Conf. on Computer Design, pp. 66-69, Oct. 1988.
- [6] T.M. Niermann and J.H.Patel, "HITEC : A Test Generation Package for Sequential Circuit," in Proc. Of the European Design Automation Conf., Feb.1991, pp.214-218.
- [7] T.-C.Lee, N.K.Jha, and W.H.Wolf, "Behavioral synthesis of highly testable data paths under the non-scan and partial scan environments," in Proc. Design Automation Conf., pp. 292-297, June 1993.
- [8] R.Inoue, T.Hosokawa, H.Fujiwara, "A Test

Generation Method for State Observable FSMs to Increase Defect Coverage under the Test Length Constraint," IEEE Workshop on RTL and High Level Testing, pp.79-86, Oct. 2007.

[9] H.Iwata, T.Yoneda, and H.Fujiwara, "A DFT Method for RTL Data Paths Based on partially strong testability to Guarantee Complete Fault Efficiency," in Proc. of the 14th Asian Test Symposium, pp.306-311, Dec. 2005.

[10] B.T.Murray and J.P.Hayes, "Hierarchical test generation using precomputed tests for modules," IEEE Trans. Computer-Aided Design, vol. 9, pp. 594-603, June 1990.

表 2 : 8ビット除算器のデータ

#GATE	#PI	#PO	#FF	#control	#status
573	24	16	44	9	2

表 3 : 機能的時間展開モデルのテスト生成結果

モデル	CPUtime (秒)	DETECT	FE
モデル 1	3.78	963	93.13
モデル 2	2.99	967	93.52

表4 : 順序ATPGテスト生成結果比較

ATPG	CPU time (秒)	FE
STAGY	6.76	94.29
TetraMAX	3811.27	83.04