ISSN 2186-5647

2-50

CDFG ベース動作記述に対する 演算器入出力順序深度削減のための テスト容易化バインディング法に関する研究

日大生産工　○中村　健太　日大生産工　細川　利典　大院大　藤原　秀雄

1. はじめに

近年，半導体集積技術の発展に伴い，設計される大規模集積回路（ Large Scale Integrated circuits：LSI)の大規模化，複雑化が急速に進んでいる[1]．これにより，LSI の設計コストの増大が問題視されている．LSI の設計生産性を向上させる手段として，動作レベルの回路記述からレジスタ転送レベル(Register Transfer Level：RTL)の回路を生成する動作合成が提案されている[1][2]．動作レベルの回路記述は，RTL の回路記述と比較して，高い抽象度で記述されているため，設計生産性に優れる．

一方，LSI の高速化，高集積化に伴い，一般に広く用いられているスキャンテストの実行時間が増加している．それに加え，フルスキャン設計[3]が施された回路に対するサイドチャネル攻撃手法[4]が提案されており，セキュリティ面でも問題があることが報告されている[4]．そのため，非スキャンテストで少ないテスト生成時間で高い故障検出効率のテストを実行することが理想的であり，非スキャンテストに基づくテスト容易性を考慮した動作合成が提案されている[5-7]．これらの手法は，RTL 回路のデータパスのテスト容易性に着目した手法である．データパスとコントローラがテスト時に分離されていることが前提となる．テスト時にデータパスとコントローラを分離するためには付加回路が必要である．一方，データパスとコントローラを分離しないことを前提とした RTL 回路に対するテスト容易化設計手法も提案されている[8]．

文献[8]では，データパスのテスト容易な構造に基づいて，テスト時にその動作を制御可能とするようにコントローラを拡大する．しかしながら，従来の順序回路のテスト生成アルゴリズムは，回路構造にのみ着目してテスト系列を生成する．すなわち，テスト容易な構造に基づいてテスト系列が生成されるとは限らない．

文献[9]ではデータパスのテスト容易な構造に着目したテスト容易化機能的 k 時間展開モデル[9]を生成し，コントローラ拡大を適用し，そのモデルの動作を制御可能とし，テスト容易化機能的 k 時間展開モデルを用いてテスト生成を実行する方法が提案されている．そのテスト生成時は，拡大したコントローラの機能に着目し，生成したテスト容易化機能的 k 時間展開モデルの動作を制御するように各時刻の制御信号・状態信号値を制約として与える．さらに，文献[9]の手法は，テスト時にコントローラとデータパスを分離せずにテスト容易な動作を制御できる．それにより，比較的小さな時間展開数で，かつ解空間の狭いモデルを生成することができ，テスト生成の効率化が図れる．しかしながら，文献[9]の手法は，データパス内の演算器のテスト生成のみを対象とした DFT 手法である．さらに，テスト容易化機能的 k 時間展開モデルを使用するために，各時刻の制御信号と状態信号の時系列値を制約値として与えることのできる専用の制約付テスト生成を必要とする．

文献[10]では，演算器だけではなくデータパス全体のテスト容易性を考慮し，回路構造に基づいた一般のテスト生成を用いて高い故障検出効率を達成するためにデータパス中のすべてのハードウェア要素（演算器，マルチプレクサ，レジスタ）に対応するテスト容易化 k 時間展開モデルの生成とその動作を実現するための無効テスト状態[11]の状態遷移を設計するというコントローラ拡大に基づく DFT 手法が提案されている．またその DFT 手法は，全ての状態レジスタと状態信号レジスタのみをスキャン設計したパーシャルスキャン設計を用いている．コントローラをスキャン設計することにより，無効テスト状態を含む任意の状態にシフト動作で遷移することが可能である．任意の状態から k サイクル間状態遷移を実行することにより，一般の順序テスト生成を用いたとしても，各ハードウェア要素に対するテスト容易化機能的 k 時間展開モデルが回路構造に基づくテスト生成が比較的実現容易である．

コントローラ拡大を行ったとしても，演算器入出力順序深度[12]が大きい演算器は，その演算器から外部入力，外部出力までに多くのレジスタを通るので，その演算器をテストするためのテスト容易化機能的 k 時間展開モデルの時間展開数が大きくなり，テスト生成が困難となる可能性がある．文献[12]では，生成されるテスト容易化機能的 k 時間展開モデルの時間展開数を削減するために演算器入出力順序深度を削減するバインディング法が提案されている．文献[12]は，データフローグラフ(Data Flow Graph：DFG)ベースの回路に対するテスト容易化バインディング法であり，コントロールデータフローグラフ(Control Data Flow Graph：CDFG)ベースの回路には対応していない．

本論文では，CDFG ベースの回路を対象としたテスト容易化バインディング法を提案し，演算器入出力順序深度の削減を目指す．

Study on Testability Binding Method for Reducing Operator In / Out Sequence Depth for CDFG Base Behavioral Specifications

Kenta NAKAMURA，Toshinori HOSOKAWA，and Hideo Fujiwara

2. 諸定義

本章では，演算器入出力順序深度削減のためのバインディングで使用する用語の定義を行う．

2.1. 演算器入出力順序深度

ある演算器の 1 個の入力から，全ての外部入力に到達するまでの各経路における最小のレジスタ数をその演算器の演算器入力順序深度という．特に，演算器の i (1 ≤ i ≤ 演算器の入力数) 番目の入力順序深度を演算器第 i 入力順序深度という．また，ある演算器の出力から，全ての外部出力（状態信号線を含む）に到達するまでの各経路における最小のレジスタ数をその演算器の演算器出力順序深度という．

図 1 は演算器 SUB0，MUL0，MUL1，レジスタ R0～R4，マルチプレクサ M0～M7 から構成されるデータパス例である．

MUL1 の演算器第 1 入力順序深度は，MUL1 の第 1 入力→M7→R3→MUL0→M6→R2→M2→y の経路に存在するレジスタ数が 2 個で，MUL1 の第 1 入力までの全経路中で最小であるので，MUL1 の演算器第 1 入力順序深度は 2 である．MUL1 の演算器第 2 入力順序深度は，MUL1 の第 2 入力→R1→M1→z の経路に存在するレジスタ数が 1 個で，MUL1 の第 2 入力までの全経路中で最小であるので，MUL1 の演算器第 2 入力順序深度は 1 である．MUL1 の出力順序深度は，MUL1 の出力→M1→R1→M4→SUB0→M0→R0→u1 の経路に存在するレジスタが 2 個で，MUL1 の出力から外部出力までの全経路中で最小であるので，MUL1 の出力順序深度は 2 である．

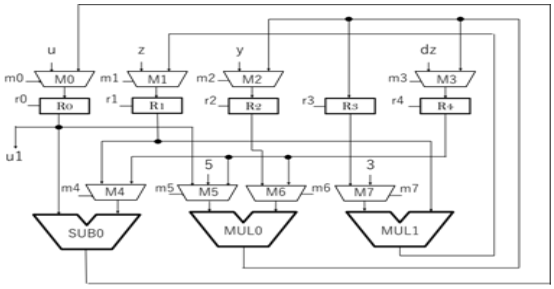

図 1 データパス図

2.2. 演算器両立グラフ

演算器バインディングにおいて，頂点集合 V と辺集合 E と頂点の重み w からなる無向グラフ G(V,E,w) を用いる（演算器両立グラフ）．前提として 1 つの演算に 1 つの演算器が割り当てられているものとする．頂点 v（v∈V）は演算器であり，辺（u,v）((u,v)∈E, u,v∈V, u≠v) は隣接頂点である演算器 u と v が共有可能であることを示す．各頂点は重みがラベル付され，重み w : V→Z+(Z+は非負の整数)で示される．頂点 v の重み w(v)は演算器 v の，演算器入出力順序深度である．演算器入力順序深度が無限大であるとき，その値は十分に大きな整数値（例．100）を用いて重みを計算する．

図 2 は演算器両立グラフの例であり，5 個の乗算器（*0～*4）の共有可能性を表現している．表 1 に，図 2 の各頂点の重みを示す．

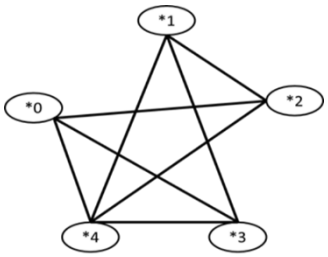

図 2 演算器両立グラフ

表 1 重み一覧

頂点	第1入力順序深度	第2入力順序深度	出力順序深度
＊0	1	1	4
＊1	100	1	4
＊2	2	2	3
＊3	100	1	3
＊4	1	2	2

2.3. レジスタ両立グラフ

レジスタバインディングに，頂点集合 V と辺集合 E と辺の重み w からなる無向グラフ G(V,E,w)を用いる（レジスタ両立グラフ）．前提として，1 つの変数には，1 つのレジスタが割り当てられているものとする．頂点 v（v∈V）はレジスタであり，辺（u,v）((u,v)∈E, u,v∈V, u≠v）は隣接頂点であるレジスタ u と v が共有可能であることを示す．各辺は重みがラベル付され，重み w：E→Z+ (Z+は非負の整数)で示される．辺(u,v)の重み w(u,v)は変数 u と v を共有化した際の，全演算器の演算器入出力順序深度の削減数の総和である．

図 3 は，レジスタ両立グラフの例であり，11 個のレジスタの共有可能性を表現している．表 2 は，図 3 に対して演算器バインディング（演算*0,*3,*4 を共有化，演算*1,*2 を共有化）が完了した後の，図 3 の辺の重みを示す．表 2 に，演算器入力および出力順序深度の総和を削減できる辺の重みのみを示す．

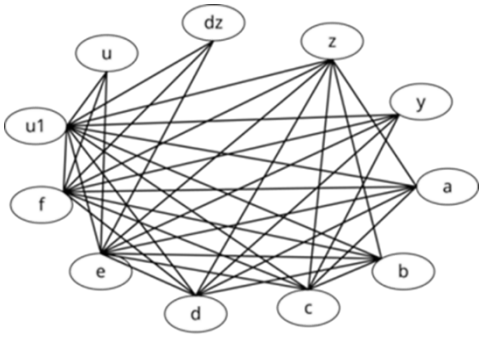

図 3 レジスタ両立グラフ

表 2 重み一覧

辺	重み	辺	重み	辺	重み	辺	重み
(b,u1)	1	(d,e)	1	(c,y)	2	(c,a)	1
(c,u1)	1	(d,f)	1	(d,u1)	2	(f,y)	2
(f,u1)	1	(c,u)	2	(c,b)	1	(f,a)	1
(a,u1)	2	(c,dz)	2	(f,u)	2	(f,b)	1
(a,e)	1	(c,z)	2	(f,dz)	2	(f,z)	2
(a,f)	1						

2.4. 演算器両立グラフの重み和最小最小クリーク分割問題

入力：演算器両立グラフ G

出力：演算器バインディング情報

制約：演算器の個数

最適化：各クリークを構成する頂点の重みの総和

を最小化する．

2.5. レジスタ両立グラフの重み和最大最小クリーク分割問題

入力：レジスタバインディンググラフ G
出力：レジスタバインディング情報
制約：レジスタの個数
最適化：各クリークを構成する辺の重みの総和を最大化する．

3. 演算器入出力順序深度削減指向テスト容易化バインディンアルゴリズム

本章では，テスト容易化バインディングのアルゴリズムの説明を行う．

3.1. テスト容易化バインディングアルゴリズム

図 4 にテスト容易化バインディングのアルゴリズムを示し，ステップ毎に説明する．

step1：DFG 内の演算数分の演算器を生成する．
step2：ループ変数 i を 0 に初期化する．
step3:i<CDFG の状態数を満たす場合は step4 へ，満たさない場合は step7 の処理を行う．
step4：CDFG の状態 i の DFG に対して，演算器両立グラフの重み和最小最小クリーク分割問題を解く．
step5：CDFG の状態 i の DFG に対して，演算器割当てを行う．
step6：ループ変数 i をインクリメントする．
step7：各状態の演算器をマージする．
step8：全変数分のレジスタを生成する．
step9：レジスタ両立グラフの重み和最大最小クリーク分割問題を解く．
ssep10：レジスタ割当てを行う．

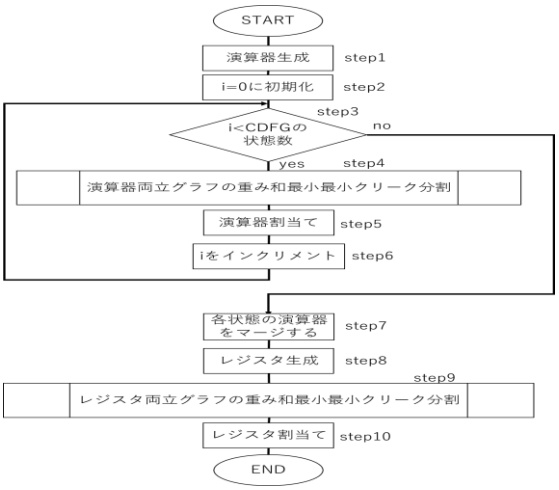

図 4 テスト容易化バインディングアルゴリズム

3.2. 演算器両立グラフの重み和最小最小クリーク分割

図 5 に演算器両立グラフの重み和最小最小クリーク分割のアルゴリズムを示し，ステップ毎に説明する．

step1：全てのクリーク集合の組み合わせを列挙した場合は step6, そうでない場合は step2 の処理を行う．
step2：クリーク分割を行う．
step3：step2 の解が演算器数最小制約を満たしている場合は step4, そうでない場合は step1 の処理を行う．
step4：step2 の解のクリークの重み和を計算する．
step5：step2 の解をクリーク集合の集合に追加する．
step6：クリーク集合の集合の中から重み和が最小の解を返す．

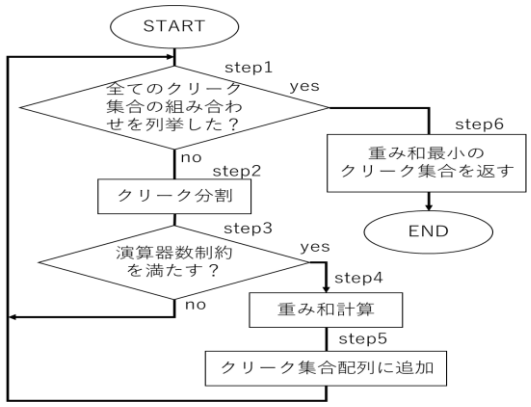

図 5 演算器両立グラフの重み和最小最小クリーク分割

3.3. レジスタ両立グラフの重み和最大最小クリーク分割

図 6 にレジスタ両立グラフの重み和最大最小クリーク分割のアルゴリズムを示し，ステップ毎に説明する．

step1：全てのクリーク集合の組み合わせを列挙した場合は step6, そうでない場合は step2 の処 s 理を行う．
step2：クリーク分割を行う．
step3：step2 の解がレジスタ数最小制約を満たしている場合は step4, そうでない場合は step1 の処理を行う．
step4：step2 の解のクリークの重み和を計算する．
step5：step2 の解をクリーク集合の集合に追加する．
step6：クリーク集合の集合の中から重み和が最大の解を返す．

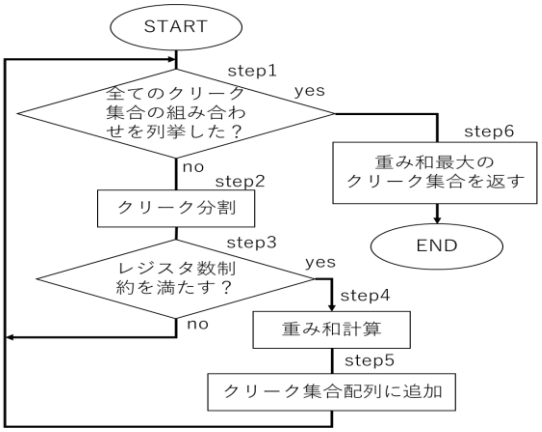

図 6 レジスタ両立グラフの重み和最大最少クリーク分割

4. 実験結果

本論文では，提案する CDFG ベースの回路を対象としたテスト容易化バインディング法の有効性を示すために 3 種類の動作合成ベンチマーク回路[11]で評価実験を行った．本実験では従来手法として，最小面積指向バインディング法であるレフトエッジアルゴリズム[2]によって生成された RTL 回路と，提案するバインディング法によって生成された RTL 回路をそれぞれ合成し実験を行った．評価基準には，演算器入出力順序深度を用いた．表 3 に演算器入出力順序深度の実験結果を示す．

演算器入出力順序深度を比較すると，Sehwa と Maha は従来手法と提案手法ともに最小値となった．つまり，従来手法と同じくテスト容易な RTL 回路を生成できた．

Kim の SUB1 を比較すると，第 2 入力順序深度が従来手法では外部入力に到達せず∞となっている．一方，提案手法では SUB1 の第 2 入力順序深度は 1 までに減らすことができた．Kim の SUB1 以外の演算器入出力順序深度はすべて同じ値となった．つまり，Kim に関しては，従来手法より提案手法のほうがテスト容易な RTL 回路を生成できたといえる．

Kim の LESS0 の第 2 入力順序深度は従来手法と提案手法ともに∞となった．これは，CDFG 中に比較演算が 1 個しかなく，比較演算の第 2 入力には定数しか入らないことが原因である．

表 3. 演算器入出力順序深度

従来手法					提案手法				
回路名	演算器名	第1入力	第2入力	出力	回路名	演算器名	第1入力	第2入力	出力
Sehwa	ADD0	1	1	1	Sehwa	ADD0	1	1	1
	SUB0	1	1	1		SUB0	1	1	1
	LESS0	1	1	1		LESS0	1	1	1
Maha	ADD0	1	1	1	Maha	ADD0	1	1	1
	SUB0	1	1	1		SUB0	1	1	1
	LESS0	1	1	1		LESS0	1	1	1
Kim	ADD0	1	1	1	Kim	ADD0	1	1	1
	ADD1	1	1	1		ADD1	1	1	1
	SUB0	1	1	1		SUB0	1	1	1
	SUB1	1	∞	1		SUB1	1	1	1
	LESS0	1	∞	1		LESS0	1	∞	1

5. まとめ

本論文では，CDFG ベースの回路を対象としたテスト容易化バインディング法を提案した．評価実験では，実験対象の 3 個の回路全てにおいて演算器入出力順序深度が同じ，または削減することができ，テスト容易な RTL 回路を生成できた．一方，今回マルチプレクサの入出力順序深度は考慮していないので，最小値ではないところが存在し，削減することができる．今後の課題は，すべてのハードウェア要素の入出力順序深度削減指向のバインディング法の提案が挙げられる．

参考文献

[1] 藤原 秀雄, ディジタルシステムの設計とテスト, 工学図書株式会社, 2004.

[2] 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 Publishers, 1992.

[3] H. Fujiwara, Logic Testing and Design for Testability, The MIT Press, 1985.

[4] R. Nara, K. Satoh, M. Yanagisawa, T. Ohtsuki, and N. Togawa, “Scan-Based Side-Channel Attack Against RSA Crypto-Systems Using Scan Signatures”, IEICE Trans. on Fundamentals, Vol. E93-A, No. 12, pp2481-2489, 2010.

[5] M.T.-C. Lee, W. H. Wolf, and N. K. Jha, “Behavioral Synthesis for Easy Testability in Data Path Scheduling”, in Proc. Int. Conf. on Computer-Aided Design, pp.616 -619, 1992.

[6] M.T.-C. Lee, W. H. Wolf, and N. K. Jha, “Behavioral Synthesis for Easy Testability in Data Path Allocation”, in Proc. Int. Conf. Computer Design, pp. 29 -32, 1992.

[7] M.T.-C. Lee, N. K. Jha, and W. H. Wolf, “Behavioral Synthesis for Highly Testable Datapaths Under the Non-Scan and Partial Scan Environments”, in Proc. Design Automation Conf., pp.292-297, 1993.

[8] L.M.FLottes , B.Rouzeyre , L.Volpe,”A Controller reSynthesis Based Methos For Improving Datapath Testability”, IEEE International Symposium on Circuits and Systems, pp. 347 -350, May 2000.

[9] T. Masuda, J. Nishimaki, T. Hosokawa and H. Fujiwara, "A Test Generation Method for Datapaths Using Easily Testable Functional Time Expansion Models and Controller Augmentation," IEEE the 24th Asian Test Symposium (ATS'15), pp. 37-42, Nov. 2015.

[10] 石山悠太, 細川利典, 山崎紘史, ” パーシャルスキャン設計を用いた k サイクルキャプチャテストのためのコントローラ拡大法”, Forum on Information Technology (FIT2018)

[11] S. Ohtake, T. Masuzawa, and H. Fujiwara, "A non-scan approach to DFT for Controllers Achieving 100% Fault Efficiency , " Journal of Electronic Testing: Theory and Applications (JETTA), Vol. 16, No. 5, pp.553-566, Oct. 2000.

[12] M. Sato, T. Masuda, J. Nishimaki, T. Hosokawa, and H. fujiwara, ” A Binding Method to Generate Easily Testable Functional Time Expansion Models”, 17th IEEE Workshop on RTL and High Level Testing(WRTLT’16)