ISSN 2186-5647

2-37

機能的時間展開モデル生成アルゴリズムの評価

日大生産工(院)　○増田哲也　日大生産工(院)　西間木淳
日大生産工　細川利典　大阪学院大　藤原秀雄

1. まえがき

近年，半導体集積技術の発達に伴い，設計される大規模集積回路(Large Scale Integrated circuits:LSI)の高機能化，高集積化，複雑化が急速に進展している[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法として，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[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るため，設計生産性に優れる．また，LSI の高集積化に伴い，一般的な LSI テスト手法であるスキャンテストのテスト実行時間は急激に増加している．そのため，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法が提案されている[3]．これらの手法は RTL 回路のデータパスのみに着目した手法である．そのため，データパスに対する高い故障検出効率を実現するためには，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 RTL 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 テスト容易化設計手法も提案されている[4]．文献[4]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実現させるようにコントローラを拡大する．これまで提案されてきた順序回路のテスト生成アルゴリズム[5][6]は，回路構造のみからテスト系列を生成する．それゆえ，テスト容易な構造に基づいて拡大したコントローラの機能通りにテスト生成するとは限らない．文献[7,8]では，コントローラの状態遷移を網羅するようにデータパスの機能的時間展開モデル[8]を生成し，テスト生成を行う方法が提案されている．この手法ではコントローラの機能に基づいて機能的時間展開モデルを生成するため，時間展開数 k はデータパスのレイテンシに依存し，大きくなる可能性がある．時間展開数 k が大きくなると，その機能的時間展開モデルにおけるテスト生成の探索空間が増大し，テスト生成が困難になる．文献[9]ではデータパスのテスト容易な構造に着目し，そのテスト容易な構造に基づいてテスト生成を行うための機能的 k 時間展開モデルを生成する．さらに，生成された機能的 k 時間展開モデルの動作を実現するためにコントローラを拡大する．生成される機能的 k 時間展開モデルによって，コントローラ拡大の面積オーバーヘッドは異なる．それゆえ，機能的 k 時間展開モデルの生成においては，コントローラ拡大による面積オーバーヘッドを考慮する必要がある．さらに，生成される機能的 k 時間展開モデルはそれぞれテスト容易性が異なる．また，機能的 k 時間展開モデル生成アルゴリズムが異なると，生成される機能的 k 時間展開モデルも異なる．従って，テスト容易性および面積オーバーヘッドを考慮した機能的 k 時間展開モデル生成アルゴリズムを使用することが，機能的 k 時間展開モデル生成において重要となる．

本論文では，機能的 k 時間展開モデル生成アルゴリズムを提案し，その評価を行う．本論文で提案するアルゴリズムは，コスト関数を基に機能的 k 時間展開モデルを生成する．コスト関数は生成された機能的 k 時間展開モデルのテスト容易性とコントローラ拡大による面積オーバーヘッドの評価のために使用される．コスト関数を用いることで，小さな面積オーバーヘッドで，かつ高い故障検出率を期待できる機能的 k 時間展開モデル生成を実現する．本論文で提案するアルゴリズムによって生成された機能的 k 時間展開モデルのテスト容易性とコントローラ拡大のオーバーヘッドを実験により評価した．以下，機能的 k 時間展開モデルのことを機能的 k-TEM と呼ぶ．

2. 機能的 k 時間展開モデル

本節では機能的 k-TEM を説明する．機能的 k-TEM とは，RTL データパスのテスト容易な構造に基づいてテスト生成を実行するための時間展開モデルである．図 1(a)に ex2[10]のス

An Evaluation of a Functional Time Expansion Model Generation Algorithm
Tetsuya MASUDA, Jun NISHIMAKI, Toshinori HOSOKAWA and Hideo FUJIWARA

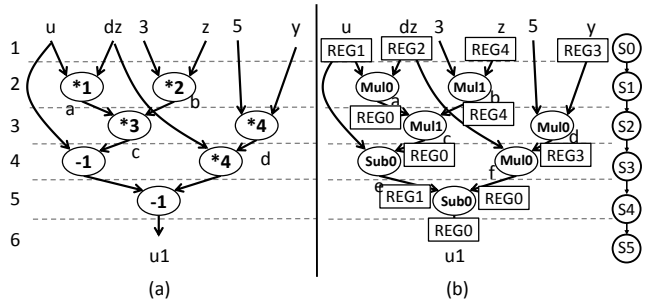

図 1. ex2 のバインディング済 DFG

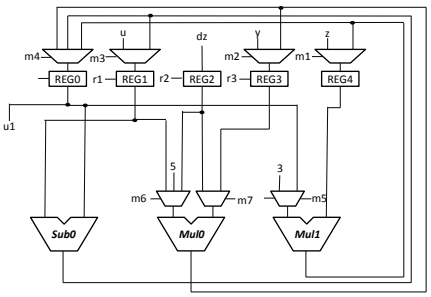

図 2. ex2 の RTL データパス

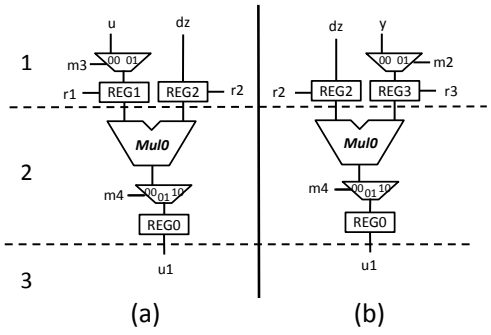

図 3. 機能的 3-TEM 例

ケジューリング済データフローグラフ(Data Flow Graph：DFG)を示す．図 1(b)に，ex2 のスケジューリング，演算器バインディング及びレジスタバインディング済の DFG とコントローラの動作を示す．図 1(b)の各ノードに付加された Sub0，Mul0，Mul1 の記述は，DFG 内の演算操作をバインディングした演算器名を示す．演算器名 Mul0 及び Mul1 は乗算器を意味し，演算器名 Sub0 は減算器を意味する．図 1(b)の各エッジに示された REG1，REG2，REG3，REG4，REG5 の記述は，DFG 内の変数をバインディングしたレジスタ名を示す．また，図 1(b)のコントローラに記された S0，S1，…，S5 は状態を意味し，各状態を結ぶエッジは状態遷移を意味する．図 2 に，図 1(b)で示した ex2 のスケジューリング及びバインディング済 DFG 情報より生成される RTL データパスを示す．機能的 k-TEM 生成では，図 2 で示したような RTL データパスのテスト容易な構造に着目する． 図 3 に図 2 の RTL データパスのテスト容易な構造に基づいて生成された乗算器 Mul0 をテストするための機能的 3-TEM の例を示す．図 3(a)および図 3(b)は異なる機能的 3-TEM であるが，二つのモデルはともに乗算器 Mul0 を含んでいる．すなわち，図 3 の機能的 3-TEM を用いることで Mul0 に対するテスト生成が可能となる．

次に，図 3 における機能的 3-TEM を実現するためのコントローラ拡大を考える．機能的 k-TEM を用いてテストを実行するためには，生成された機能的 k-TEM の動作を実現するためのコントローラの拡大が必要となる場合がある．コントローラ拡大による面積オーバーヘッドに関して図 3(b)を用いて説明する．図 3(b)の機能的 3-TEM は 1 サイクル目に外部入力 dz の値を REG2 にロードし，外部入力 y の値を REG3 へロードする．2 サイクル目に REG2 および REG3 の値を Mul0 へ印加し，REG0 に演算結果を取り込む．最後に 3 サイクル目にその出力結果を外部出力 u1 で観測する．上記のような動作を実現する制御信号系列をコントローラは 3 サイクルにわたり状態遷移させる必要がある．しかしながら，コントローラの機能動作のみを利用して，図 3(b)の動作を実現することは不可能である．そのため，コントローラの拡大を実行する．図 4 に図 3(b)の機能的 3-TEM の動作を実現するために拡大したコントローラの例を示す．図 4 のコントローラはテスト用の機能として新たに状態 S0 から状態 S3 への遷移が追加されている．この遷移を利用することで，図 3(b)の機能的 3-TEM の動作が実現できる．コントローラの本来の機能に新たに状態遷移を追加したため，コントローラに面積オーバーヘッドが発生する．次に図 3(a)の機能的 3-TEM の動作を実現するためのコントローラ拡大を考える．図 3(a)の機能的 3-TEM は 1 サイクル目に外部入力 u の値を REG1 にロードし，外部入力 dz の値を REG2 へロードする．2 サイクル目に REG1 および REG2 の値を Mul0 へ印加し，3 サイクル目にその出力結果を REG0 から外部出力 u1 へ伝搬する．上記のような制御信号系列を出力する機能をコントローラが有していない場合，拡大が必要となる．しかしながら，図 1(b)のコントローラに着目すると，図 3(a)の機能的 3-TEM は

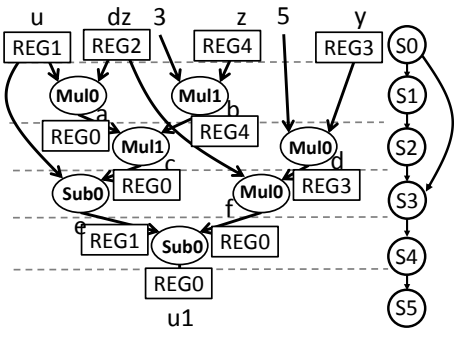

図 4. コントローラ拡大例

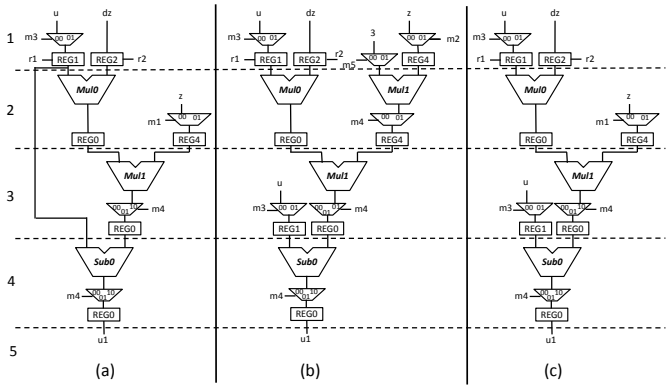

図 5. 機能的 5-TEM 例

状態 S0→状態 S1→状態 S2 の 3 サイクルの機能動作(状態遷移)によって実現可能である．すなわち，図 3(a)の機能的 3-TEM を動作させるためのコントローラの拡大は不要である．したがって，図 3(a)の機能的 3-TEM を生成することで，コントローラにテスト用の機能を追加せずに Mul0 のテストが可能となる．

次に，生成される機能的 k-TEM のテスト容易性について考える．図 5 に機能的 5-TEM の例を示す．図 5 の(a)，(b)，(c)の機能的 5-TEM は乗算器 Mul0，Mul1，減算器 Sub0 を含んでいる．そのため，三つの演算器に対してテスト生成が可能となるが，そのテスト容易性はそれぞれ異なる．図 5(a)は機能的 5-TEM に再収斂構造が存在し，図 5(b)は定数制御の影響を受ける演算器が存在する．図 5(c)は再収斂構造も定数制御演算器も存在しない機能的 5-TEM である．図 5(a)を用いてテスト生成する場合，時刻 2 の Mul0 の左入力および時刻 4 の Sub0 の左入力の値割り当てが衝突する可能性が生じる．次に，図 5(b)を用いてテスト生成する場合を考える．図 5(b)では時刻 1 で定数値 3 が Mul1 の左入力へ与えられる．すなわち，時刻 2 の Mul1 の出力は 3 の倍数に固定され，その影響が時刻 3 の Mul1，時刻 4 の Sub0 の出力まで及ぶ．定数制御の影響を受ける演算器によって値が固定され，テスト生成時に値の制御が困難となる．図 5(c)は上記で示したような再収斂構造や定数制御演算器が存在しないため，図 5(a)，(b)と比較してテスト容易であるといえる．したがって，テスト容易性を考慮する場合，図 5 で示した三つの機能的 5-TEM のうち，図(c)の機能的 5-TEM を選択することが適切である．本節で示した，機能的 k-TEM 生成におけるコントローラの面積オーバーヘッドおよび機能的 k-TEM のテスト容易性を機能的 k-TEM 生成時に考慮することで，小さな面積オーバーヘッドで，かつ高い故障検出率を期待できる機能的 k-TEM を生成できる．

3. 機能的時間展開モデル生成アルゴリズム

機能的 k-TEM 生成時にテスト容易性およびコントローラ拡大における面積オーバーヘッドを考慮することが重要となる．本論文で示すアルゴリズムは前節で説明した各要素を考慮にいれたコスト関数を用いる．コスト関数を用いて生成された機能的 k-TEM のコストを計算し，コストが最も低くなる機能的 k-TEM を選択する．さらに選択された機能的 k-TEM を基にコントローラを拡大する．本アルゴリズムで使用するコスト関数を以下に示す．

$$COST = k \times \alpha RC \times \beta(C_{add} + C_{sub}) \times \gamma(C_{mul} + C_{div}) \times T$$

α, β, γは係数であり，kは時間展開数，RCは再収斂構造数，C_{add}は定数制御を受ける加算器数，C_{sub}は定数制御を受ける減算器数，C_{mul}は定数制御を受ける乗算器数，C_{div}は定数制御を受ける除算器数であり，それぞれテスト容易性を評価するコストとなる．また，Tはコントローラ拡大による追加状態遷移数であり，面積オーバーヘッドのコストを表す．

図 6 に機能的 k-TEM 生成アルゴリズムを示す．各手順について以下に説明を行う．

1 行目：データパス内に存在する演算器の集合を取得する．

2 行目：演算器の数だけ 3 行目，4 行目を繰り返す．

3 行目：テスト対象演算器を決定する．

4 行目:テスト対象演算器を含む機能的 k-TEM を全探索する．ただし，k 未満の機能的 k-TEM を含む．

5 行目：各演算器に対して生成されたすべての機能的 k-TEM のコストを上記のコスト関数を用いて計算し，コストの総和が最小となる機能的 k-TEM 集合を選択する．ただし，選択された集合 M 中の機能的 k-TEM にはすべての演算器が含まれる．

6 行目：選択された機能的 k-TEM の動作を実現するため，コントローラを拡大する．

7 行目：5 行目で取得した機能的 k-TEM 集合 M を返す．

```
Generate Easily Testable Functional k TEM (Datapath, Controller, k)
{
1   FUNC = Get_Use_Functional_Unit(Datapath);

2   For each Functional Unit i = 0  to j
3       Test_Funci = Get_Test_Target_Functional_Unit(FUNC, i);
4       Fk-TEMseti = Get_Easily_Testable_Functional_k_TEM(Datapath, Test_Funci, k);
    end For

5   M = Select_Fk-TEM(Fk-TEMset1, Fk-TEMset2,..., Fk-TEMsetj, Controller);

6   Augment_Controller(M, Controller);

7   return M;
}
```

図 6. 機能的時間展開モデル生成アルゴリズム

4. 実験結果

本節では，本論文で示したアルゴリズムを用いた実験結果を示す．実験回路として DFG ベースの回路 ex2[10]，ex4[10]，DFCT[11]の三個の回路を対象とした．本論文で示したコスト関数とアルゴリズムを使用して，各回路の機能的 k-TEM 生成を実行し，故障検出率とテスト生成時間，およびコントローラ拡大による面積オーバーヘッドを評価した．なお，コスト関数の係数はα=3，β =5，γ =20 とした．比較対象となる従来手法として，コントローラを拡大せず，機能的時間展開モデルを用いないで順序回路テスト生成を実行する手法を挙げる．従来手法および提案手法に対するテスト生成は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[8]を利用し，データパス内部の演算器にのみ縮退故障を設定した．また，テスト生成では故障シミュレーションを実行した．表 1 に本手法で生成した機能的 k-TEM の詳細(最長機能動作レイテンシ，時間展開数 k，モデル数 m，追加状態遷移数，面積オーバーヘッド)を示す．各回路において時間展開数が 4 となる機能的 k-TEM が生成でき，すべての回路において機能動作よりも少ない時間展開数でテスト生成が可能となった．さらにコントローラ拡大による回路全体の面積オーバーヘッドも ex2 ではオリジナルの回路と比較して 0.2%，ex4 では 0.12%，DFCT では 0.18%で抑えることができた．次に各回路に対してテスト生成を実行した結果を表 2，表 3 に示す．故障検出率は ex2，ex4 において提案手法のほうが高い故障検出率を達成できた．またテスト生成時間は，ex2 では従来手法から約 35%，ex4 では約 87%，DFCT では約 62%削減できた．提案手法は，コントローラの拡大と，テスト容易性の高い機能的 k-TEM の生成できたため，高速かつ高い故障検出率を達成することができたと考える．

表 1. 面積オーバーヘッド

回路名	機能動作レイテンシ	時間展開数 k	kTEM数m		追加状態遷移数	面積オーバヘッド(%)
			k=3	k=4		
ex2	6	4	2	0	3	0.20
ex4	5	4	1	1	1	0.12
DFCT	7	4	2	1	3	0.18

表 2. 故障検出率

回路名	故障検出率(%)	
	従来手法	提案手法
ex2	99.98	100.00
ex4	99.27	99.59
DFCT	99.47	99.35

表 3. テスト生成時間

回路名	テスト生成時間(sec)	
	従来手法	提案手法
ex2	1630	1045
ex4	427	55
DFCT	5261	1959

5. まとめ

本論文ではデータパス回路の構造に着目した機能的 k-TEM 生成法として，機能的 k-TEM のテスト容易性およびコントローラ拡大による面積オーバーヘッドを考慮したアルゴリズムを提案し，評価した．機能的 k-TEM に対するテスト容易性と面積オーバーヘッドを評価するコスト関数を定義し，コスト関数を基に機能的 k-TEM 生成を実行した．本手法を用いた場合，ex2，ex4，DFCT 回路において，面積オーバーヘッドは平均で約 0.16%に抑えることができた．また，故障検出率は平均で 0.07%向上し，テスト生成時間は平均で約 60%削減できた．

今後の課題として，CDFG ベースの回路等の大規模回路での実験や，機能的 k-TEM 生成アルゴリズムで使用するコスト関数の強化等が挙げられる．

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] M.C.McFarland, A.C.Parker, R.Camposano, “The high-level synthesis of digital systems”, Proc. IEEE, pp.301-318, 1990.

[3] 高崎智也,井上智生,藤原秀雄,” 無閉路部分スキャン設計に基づくデータパスのテスト容易化高位合成におけるバインディング手法“，電子情報通信学会論文誌(DI)，Vol.J83-D-I No.2，pp.282-292,2000.

[4] L.M.FLottes, B.Rouzeyre, L.Volpe,”A CONTROLLER RESYNTHESIS BASED METHODS FOR IMPROVING DATAPATH TESTABILITY”, IEEE International Symposium on Circuits and Systems, pp. 347 -350, May 2000.

[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., pp.214-218, Feb.1991 .

[7] Toshinori Hosokawa, Teppei Hayakawa, and Masayoshi Yoshimura, “ A Comprehensive Functional Time Expansion Model Generation Method for Datapaths Using Controllers” , The Eleventh Workshop on RTL and High Level Testing (WRTLT'10), pp.131-138, Dec. 2010.

[8] Kazuya Sugiki, Toshinori Hosokawa, and Masayoshi Yoshimura, ”A Test Generation Method for Datapath Circuits Using Functional Time Expansion Models ”, The Ninth Workshop on RTL and High Level Testing (WRTLT’08), pp.39 -44, Nov. 2008.

[9] Yusuke Kodama, Jun Nishimaki, Tetsuya Masuda, Toshinori Hosokawa and Hideo Fujiwara, “A Controller Augmentation Method to Generate Easily Testable Functionl k-Time Expansion Models for Data `Path Circuits”, The Ninth Workshop on RTL and High Level Testing(WRTLT’13), 2013.

[10] M.T.-C.Lee, “High-Level Test Synthesis of Digital VLSI Circuits”, Artech House Publishers, 1997.

[11] Ian G. Harris and Alex Oraigln ,“Testability Improvement in High-Level Synthesis Through Reconvergence Reduction”, In Pro-ceedings of the Asilomar Conference on Signals, Systems and Computers, pp.1919-203 ,1996.