ISSN 2186-5647

コントローラ拡大における無効テスト状態圧縮法

日大生産工（学部）　○池ヶ谷　祐輝　　日大生産工（院）　石山悠太
日大生産工　細川利典　　日大生産工　山崎紘史

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，複雑化が急速に進展している[1]．それに伴い，LSI のテスト生成が重要な課題となっている．高い故障検出効率を達成するためには，何らかのテスト容易化設計(Design for Testability : DFT)が必要である．本論文では，データパスとコントローラから構成されるレジスタ転送レベル(Register Transfer Level:RTL)回路に対する DFT 技術に着目する．

文献[2]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(Easily Testable Functional *k* Time Expansion Models : ETF*k*-TEM)[2]を用いたテスト生成手法が提案されている．文献[2]の手法では，テスト生成を行うために，データパスのテスト容易な構造に基づき ETF*k*-TEM を生成する．また，生成された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．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 ETF*k*-TEM の動作を実現する制御信号･状態信号系列(テスト動作制御･状態信号系列)[2]を制約として与える．ETF*k*-TEM を用いてテスト生成することにより，演算器内に存在する故障に対するテスト生成に関しては高速かつ高い故障検出効率を達成することが報告されている[2]．文献[3]では，演算器だけではなく，回路中の全てのハードウェア要素（演算器，マルチプレクサ，レジスタ）に対して ETF*k*-TEM を生成し，その動作を実現するためのテスト動作制御・状態信号系列を出力する状態遷移を無効テスト状態[4]にのみ設計する DFT 手法が提案されている．しかしながら，文献[3]の手法では，テスト動作制御・状態信号系列数の増加に伴い，コントローラ中の状態レジスタのビット幅を増加させる必要があるため，回路面積オーバヘッドが増大するという課題が挙げられる．

本論文では，文献[3]の手法におけるテスト動作制御・状態信号系列を圧縮することで，無効テスト状態数を削減する手法を提案する．本手法を用いてコントローラ中の状態レジスタのビット幅の増加数を削減し，テスト動作制御・状態信号系列を圧縮しない場合よりも少ない面積オーバヘッドで同等の故障検出効率を達成することを目指す．

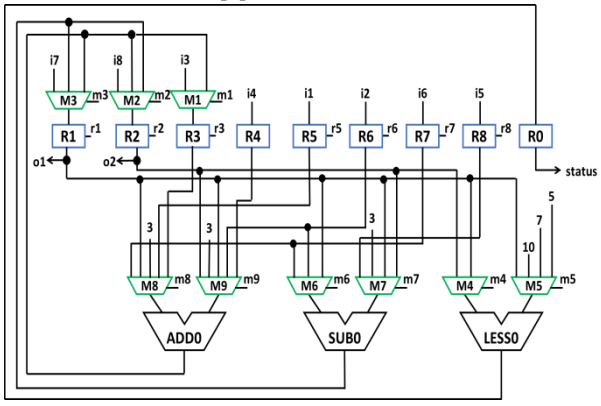

図 1. データパス例

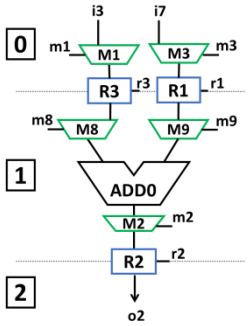

図 2. 表 1 で動作可能な ETF3-TEM 例

表 1. テスト動作制御・状態信号系列の例

	r1	r2	r3	r5	r6	r7	r8	m1	m2	m3	m4	m5	m6	m7	m8	m9	status
t0	1	X	1	X	X	X	X	0	XX	00	X	XX	XX	XX	XXX	XXX	X
t1	X	1	X	X	X	X	X	X	01	XX	X	XX	XX	XX	100	010	X
t2	X	X	X	X	X	X	X	X	XX	XX	X	XX	XX	XX	XXX	XXX	X

2.　テスト容易化設計手法

2-1 ETF*k*-TEM

ETF*k*-TEM[2]とは，データパスのテスト容易な構造に着目して生成された *k* サイクルテスト生成モデルである．ETF*k*-TEM は，データパスの全てのハードウェア要素 (演算器, マルチプレクサ, レジスタ) をテストできるように生成する必要がある．図 1 にデータパス例を示し，図 2 に図 1 の演算器 ADD0 をテスト可能な ETF3-TEM の例を示す．表 1 に図 2 の ETF3-TEM の動作を実現するための各時刻の制御信号値 $C \in \{0,1,X\}$ と状態信号値 $S \in \{0,1,X\}$ の時系列であるテスト動作制御・状態信号系列[2]を示す．図 1 において，*i*1~*i*8 は外部入力，*o*1，*o*2 は外部出力，*R*1，*R*2，*R*3，*R*5，*R*6，*R*7，*R*8 はホールドレジスタ，*R*0，*R*4 はレジスタ，*M*1~*M*9 はマルチプレクサ，*m*1~*m*9，*r*1~*r*7 は制御信号，*status* は状態信号，*ADD*0 は加算器，*SUB*0 は減算器，*LESS*0 は比較器である．表 1 において，*t*0~*t*2 は時刻を表し，図 2 における四角で囲まれた時刻 0~2 にそれぞれ対応している．ETF*k*-TEM において，外部入力または時刻 0 のスキャンレジスタからハードウェア要素の入力に何らかの値を伝搬でき，ハードウェア要素の出力から何らかの値を外部出力または時刻 2 のスキャンレジスタに伝搬できるものを，その ETF*k*-TEM で動作可能[3]であるという．マルチプレクサの左から *n* 番目の入力を入力 *n* と表す(*n* は非負の整数)．図 2 の

Invalid Test State Compaction Method Based on Controller Augmentation

Yuki IKEGAYA, Yuta ISHIYAMA, Toshinori HOSOKAWA and Hiroshi YAMAZAKI

ETF3-TEM において，*ADD*0，*R*1，*R*2，*R*3，*M*1 の入力 0，*M*2 の入力 1，*M*3 の入力 0，*M*8 の入力 4，*M*9 の入力 2 が動作可能である．

ETF*k*-TEM におけるハードウェア要素のテスト可能性に関しては文献[3]で述べられており，その説明は本論文では割愛する．

2-2 コントローラ拡大

コントローラ拡大[2][4][5]とは，コントローラに状態や状態遷移を追加するテスト容易化設計手法のことである．コントローラ中にはリセット状態から遷移し得ない状態が存在する場合があり，その状態を無効状態[4]という．文献[3]の手法では，コントローラの状態レジスタをスキャン設計することで，テスト時において，全状態から無効状態に遷移可能となる．したがって，本手法では，文献[3]の手法を適用することで，ETF*k*-TEM の動作を実現するためのテスト動作制御・状態信号系列を出力する状態遷移を無効状態にのみ設計する．この無効状態のことを無効テスト状態[4]と呼ぶ．データパスのテスト容易な構造に着目して生成された ETF*k*-TEM の動作を実現する機能は，コントローラに備わっていない可能性がある．よって，ETF*k*-TEM を考慮したテスト動作制御・状態信号系列を新たな状態遷移として無効テスト状態の状態遷移に設計することで，ETF*k*-TEM の動作を実現可能となる．なお，ETF*k*-TEM を考慮したコントローラ拡大時に無効状態数が不足する場合は，コントローラの状態レジスタのビット幅を増加させて，コントローラの無効状態数を増加させる．

3. 無効テスト状態の圧縮手法

3-1 提案手法の概要

テスト動作制御・状態信号系列をコントローラの無効テスト状態の状態遷移に設計する際，一つの状態遷移に各テスト動作制御・状態信号系列の 1 時刻分を割当てる．文献[3]の手法の場合，必要となる無効テスト状態数が増加し，状態レジスタのビット幅が増加する可能性がある．本手法では，テスト動作制御・状態信号系列をグラフを用いて圧縮する方法を提案し，無効テスト状態数を削減することでコントローラ中の状態レジスタのビット幅の増加数を削減する．

3-2 無効テスト状態の圧縮定義

制御信号値または状態信号値($P_0, P_1, \ldots, P_{w-1}$)をもつ二つのテスト動作制御・状態信号系列 *T*1，*T*2 を考える(w は制御信号線と状態信号線数の和)．*T*1，*T*2 の長さをそれぞれl_1, l_2とする．*T*1，*T*2 の時刻 *t* における制御信号値または状態信号値P_iの値をそれぞれ *T*1(*t*,*i*)，*T*2(*t*,*i*)と表記する．任意の $i(0 \le i < w)$ について，次の二つの条件のうちいずれかを満たす非負の整数 *k* が存在するとき，*T*2 は *T*1 にスキュー*k* で圧縮可能という．

(1) $k \ge l_1$

(2) $k \le t < \min\{l_1, k + l_2\}$ となる任意の t について，表 2 に示す圧縮演算 $\cap_c$ を$T1(t,i)$ と $T2(t-k,i)$に行った結果が 0,1,*X* のいずれかとなる．

表 2. 圧縮演算

	0	1	X
0	0	φ	0
1	φ	1	1
X	0	1	X

3-3 テスト動作制御・状態信号系列圧縮グラフ

テスト動作制御・状態信号系列圧縮グラフとは，無効テスト状態の圧縮を行うためのグラフである．このグラフは文献[6]で提案されたテストプラン両立グラフ[6]を応用している．テストプラン両立グラフは，階層テスト[7]のための圧縮グラフであるが，本手法ではテスト動作制御・状態信号系列に適用可能なように，[6]の方法を拡張する．テスト動作制御・状態信号系列圧縮グラフは頂点 $v \in V$，辺$(u,v) \in E$，$(u,v \in V,\ u \ne v,\ t(u) \ne t(v)$ or $j(u) \ne j(v))$，3 つの頂点のラベルからなる無効グラフ $G(V,E,j,t,l)$である．テスト動作制御・状態信号系列圧縮グラフの頂点 *v* は，各テスト動作制御・状態信号系列の 1 時刻分を表す．各頂点は 3 つのラベル $j: V \to n$ (*n* は自然数)，$t: V \to Z+$ (*Z*+は非負の整数)，$l: V \to n$ を持つ．頂点 *v* において，*j*(*v*)はテスト動作制御・状態信号系列，*t*(*v*)はテスト動作制御・状態信号系列の時刻，*l*(*v*)はテスト動作制御・状態信号系列の長さを表す．辺(*u*,*v*)は，頂点 *u* におけるテスト動作制御・状態信号系列と頂点 *v* におけるテスト動作制御・状態信号系列がスキュー$t(u)$-$t(v)$ ($t(u)$-$t(v) \ge 0$)で圧縮可能なことを表す．

表 3 は，ある ETF*k*-TEM の動作を実現するためのテスト動作制御・状態信号系列である．*T*1，*T*2，*T*3 はそれぞれ別々の ETF*k*-TEM から作成されている．図 3 は表 3 のテスト動作制御・状態信号系列 *T*1, *T*2, *T*3 をもとに作成されたテスト動作制御・状態信号系列圧縮グラフ例である．各頂点に与えられた 3 つの数字は，左からラベル *j*，*t*，*l* を表す．*T*1 を表す頂点のラベル *j* は 1，*T*2 を表す頂点のラベル *j* は 2，*T*3 を表す頂点のラベル *j* は 3 である．また，時刻 *t*0 を表す頂点のラベル *t* は 0，時刻 *t*1 を表す頂点のラベル *t* は 1, 時刻 *t*2 を表す頂点のラベル *t* は 2 である．さらに，*T*1 を表す頂点のラベル *l* は 3，*T*2 を表す頂点のラベル *l* は 2，*T*3 を表す頂点のラベル *l* は 2 である．(1,0,3)のラベルを持つ頂点は，長さ 3 のテスト動作制御・状態信号系列 *T*1 の時刻 *t*0 を表す．また，頂点(2,1,2)と頂点(1,0,3)の間には辺が存在する．これは，*T*2 と *T*1 はスキュー1 で圧縮可能であることを表す．

3-4 アルゴリズム

図 4 に提案手法の全体アルゴリズムを示す．はじめに，圧縮したテスト動作制御・状態信号系列を生成する．入力にテスト動作制御・状態信号系列 $T_j\ (j = 1, 2, \ldots, n)$ の集合 *T* を与える．まず，SC を *T* で初期化する(行 3)．SC はテスト動作制御・状態信号系列を部分的に圧縮した，または未圧縮の集合である．また，いくつかのテスト動作制御・状態信号系列を圧縮したテスト動作制御・状態信号系列を，部分的に圧縮したテスト動作制御・状態信号系列と呼ぶ．次に，SC の要素が 1 つになるまで行 5-8 の操

作を繰り返す(行 4). テスト動作制御・状態信号系列圧縮グラフ $G(V,E,j,t,l)$を SC から生成する(行 5). G に辺が存在しなければ, SC 内のそれぞれの部分的に圧縮したテスト動作制御・状態信号系列を連結して ST に格納し(行 6, 行 7), 新たな圧縮テスト動作制御・状態信号系列として ST を返す(行 8). 次に SC を ϕ で初期化する(行 10). G から一つのクリーク c_i を選択し, クリーク集合 C に加える($C = C \cup \{c_i\}$)(行 11). G から頂点 $u,v(\forall u \in c_i,\ j(u)=j(v)$となる$\forall v)$ と, u,v に接続されている辺を削除する. クリーク選択と頂点, 辺の削除は G が空になるまで繰り返す. C はクリークの重みの和が最小化するように選択される($\sum_{i=1}^{m} W(c_i)$, mはCのクリーク数, $W(c_i)$はc_iの重み). クリークの重み和は, 部分的に圧縮したテスト動作制御・状態信号系列の長さの総和と等しい. またc_iの重みは, 部分的に圧縮したテスト動作制御・状態信号系列の長さと等しい. $W(c_i)$は式(1)のように表現される.

$$W(c_i) = \max\left(t(v_{max}) - t(v) + l(v)\right)\ (\forall v \in c_i) \quad (1)$$

式(1)において, v_{max}は$\max(t(v))\ (\forall v \in c_i)$の頂点である.

次に, G から取り出すc_iを選択するための戦略を以下に示す.

(戦略 1)

サイズが最大となるクリークを選択する. これは, 圧縮するテスト動作制御・状態信号系列数を最大化することに等しい.

(戦略 2)

重みが最小となるクリークを選択する. これは, 部分的に圧縮したテスト動作制御・状態信号系列の長さを最小化することに等しい.

これらの戦略に基づいて, 最初の頂点を選択する近似アルゴリズムを以下に示す.

(H1： 最初の頂点を選択する近似アルゴリズム)

$v \in V$ は$|nbr(v)|$が最大となるように選択する. $|nbr(v)|$は v の隣接頂点集合要素数である. この近似アルゴリズムは戦略 1 で使用する.

表 3. テスト動作制御・状態信号系列の例 4

T1

	r1	r2	m1	m2	status
t0	X	1	1	X	X
t1	1	X	X	1	X
t2	X	X	X	0	X

T2

	r1	r2	m1	m2	status
t0	1	1	1	X	X
t1	X	X	X	0	X

T3

	r1	r2	m1	m2	status
t0	X	1	X	X	X
t1	X	X	X	1	X

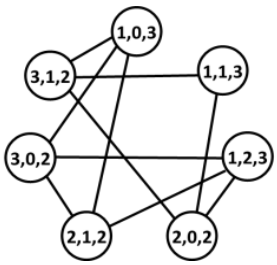

図 3. テスト動作制御・状態信号系列圧縮グラフ例

(H2： 最初の頂点を選択する近似アルゴリズム)

$v \in V$ は $l(v)$が最小となるように選択する. この近似アルゴリズムは戦略 2 で使用する.

(H1'： 最初の頂点以外の頂点を選択する近似アルゴリズム)

S をc_iのそれぞれの頂点 u の隣接集合の積集合とする. S を以下に示す.

$$S = \{v \in V - c_i | \forall u \in c_i, \exists (u,v) \in E\}$$

$v \in S$ は$|nbr(v)|$が最大となるように選択する($nbr(v) \in S$). この近似アルゴリズムは戦略 1 で使用する.

(H2'： 最初の頂点以外の頂点を選択する近似アルゴリズム)

$W(c_i)$が最小となるように選択した $v \in S$ をc_iに加える. この近似アルゴリズムは戦略 2 で使用する.

最初の頂点を選択するとき, H1 を用いて V から選択された頂点は, 頂点集合 V1 に格納される. また, H2 を用いて V1 から選ばれた頂点は, 頂点集合 V2 に格納される. 頂点は V2 からランダムに選択する. ここで, 最初の頂点を選択する 2 種類のアルゴリズム H1, H2 は, その順番を入れ替えて考えることができる. 最初の頂点以外の頂点を選択するとき, H1' を用いて V から選ばれた頂点は, 頂点集合 V3 に格納される. また, H2'を用いて V3 から選ばれた頂点は, 頂点集合 V4 に格納される. 頂点は V4 からランダムに選択する. ここで, 最初の頂点以外の頂点を選択する 2 種類のアルゴリズム H1', H2'は, その順番を入れ替えて考えることができる. それゆえ, これらのアルゴリズムの組み合わせを変えることで, クリークを選択するための 4 種類のアルゴリズムを考えることができる. これらのアルゴリズムの中で, $\sum_{i=1}^{m} W(c_i)$が最小となるものが適用され, G から C が選択される.

それぞれの$c_i \in C$において, 行 13, 行 14 の操作を繰り返す(行 12). 部分的に圧縮したテスト動作制御・状態信号系列 ST を c_iから生成する(行 13). そして, SC に ST を加える($SC = SC \cup \{ST\}$)(行 14). SC

```
1.  Generate_Compacted_Test_Operation_Control-Status_Signal_Sequence (T)
2.  {
3.    SC = T;
4.    while(|SC| > 1) {
5.      generate a test operation control−status signal sequence graph G from SC;
6.      if(G has no edges) {
7.        ST = concatenate partly compacted test operation control−status signal sequence in SC;
8.        return ST;
9.      }
10.     SC = ϕ;
11.     extract a clique set C from G;
12.     for(each clique c_i, c_i ∈ C) {
13.       ST = Compact_Test_Operation_Control−Status_Signal_Sequence(c_i);
14.       SC = SC ∪ {ST};
15.     }
16.   }
17.   return ST;
18. }
19. Compact_Test_Operation_Control−Status_Signal_Sequence(c_i)
20. {
21.   max_t = max(t(u)), u ∈ c_i;
22.   V' = {u |t(u) is equal to max_t,u ∈ c_i};
23.   max_u = a vertex selected from V' at random;
24.   ST = T_j(max_u);
25.   for(each u ∈ c_i \ max_u) {
26.     ST = T_j(u) is compacted for ST with skew (max_t − t(u));
27.   }
28.   return ST;
29. }
```

図 4. テスト動作制御・状態信号系列圧縮アルゴリズム

の要素数が 2 以上の場合，上記の繰り返し操作を終了する．繰り返し操作の終了後，ST を新たな圧縮テスト動作制御・状態信号系列として返す(行 17)．SC の要素が 1 つより多ければ, SC を新しいテスト動作制御・状態信号系列とする．

ST を生成する詳細なアルゴリズムを以下に示す．クリークc_iを入力とする．出力は部分的に圧縮したテスト動作制御･状態信号系列である．初めに, $t(u)(u \in c_i)$の最大値を max_tに格納する(行 21). 頂点集合V'は $t(u)$と max_t が等しい u を格納する(行 22). V'からランダムに頂点を選択し，その頂点を max_u とする(行 23)．ST に$T_{j(\max_u)}$を格納する(行 24)．ST の$T_{j(\max_u)}$を除いたそれぞれのテスト動作制御・状態信号系列$T_{j(u)}$をスキュー(max_t-$t(u)$)で ST と圧縮し，ST を更新する(行 25-27)．最後に ST を返す(行 28)．

4. 実験結果

本論文では，本圧縮手法の有効性を示すために，動作合成ベンチマーク回路[4]を用いた実験結果を示す．本実験の故障モデルは単一縮退故障であり，データパス及びコントローラ内の全故障を評価対象とする．本実験では，対象回路に対して文献[3]の DFT 手法適用後，本圧縮手法を適用する．対象回路のビット幅は 32bit である．また，本圧縮手法を適用せず文献[3]の DFT 手法のみを適用した回路とフルスキャン設計を適用した回路，オリジナル回路を比較対象とした．本圧縮手法におけるパーシャルスキャン設計とフルスキャン設計のスキャンパス数は 1 本とする．動作合成には内製の動作合成システム PICTHY[8]を用い，論理合成には Synopsys 社の DesignCompiler を用いた．また，テスト生成には Synopsys 社の TetraMAX を用い，TetraMAX のバックトラックリミットは 10000 とした．

表 4 に回路情報を示す．比較対象である文献[3]の DFT 手法のみを適用した手法の面積オーバヘッドは約 14%~16%である．それに対して，文献[3]の DFT 手法と本圧縮手法をともに適用した手法の面積オーバヘッドは約 10%~13%である．文献[3]のみと文献[3]+提案手法の面積オーバヘッドの削減率を比較した結果，約 3%~4%削減できた。

5. おわりに

本論文では，k サイクルキャプチャテストにおける無効テスト状態の圧縮法を提案した．動作合成ベンチマーク回路を用いた実験では，文献[3]の DFT 手法に加えて提案する圧縮手法を適用することで，文献[3]の DFT 手法のみを適用する場合と比べて面積オーバヘッドを約 3%~4%削減することができた．今後の課題として，状態遷移の圧縮をコントローラの有効状態にも適用することにより，さらに無効テスト状態数を削減し，面積オーバヘッドを削減することが挙げられる．

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] 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.

[3] 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," 17th Forum on Information Technology (FIT'18).

[4] 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.

[5] L.M.FLottes, B.Rouzeyre, L.Volpe,"A Controller Resynthesis Based Methods for Improving Datapath Testabilty," IEEE International Symposium on Circuits and Systems, pp. 347 -350, May 2000.

[6] T. Hosokawa, H. Date, and M. Muraoka, "Two Test Generation Methods Using a Compacted Test Table and a Compacted Test Plan Table for RTL Data Path Circuits," IEICE Trans. Inf & Syst. , Vol. E85-D, No. 10, pp.1474-1482, Oct. 2002.

[7] J. Lee and J.H. Patel, "Hierarchical test generation under architectural level functional constraints," IEEE Trans. on CAD, vol.15, no.9, pp.1144–1151, Sept. 1996.

[8] 石井 英明, 細川 利典, "テスト容易化のためのインターフェースを設けた動作合成システム PICTHY の開発," 第 62 回 FTC 研究会, Jan. 2010.

表 4. 回路情報

回路名	テスト容易化設計手法		スキャンFF/総FF数	増加状態レジスタビット幅	追加状態遷移数	回路面積	面積オーバヘッド（%）
Sehwa	オリジナル		0 / 262	0	0	4848	0.00
	フルスキャン		262 / 262	0	0	5910	21.91
	[3]	圧縮前	10 / 266	3	227	5637	14.21
		圧縮後	9 / 265	2	45	5353	10.42
Maha	オリジナル		0 / 198	0	0	4177	0.00
	フルスキャン		198 / 198	0	0	4983	19.30
	[3]	圧縮前	10 / 202	3	134	4845	15.99
		圧縮後	9 / 201	2	73	4665	13.31
Kim	オリジナル		0 / 198	0	0	4776	0.00
	フルスキャン		198 / 198	0	0	5581	16.86
	[3]	圧縮前	10 / 202	3	216	5537	15.93
		圧縮後	9 / 201	2	59	5341	11.83