故障検出確率に基づくテストパターン数削減のためのテストポイント挿入

日大生産工(学部)　○齊藤　善洋　　日大生産工　細川　利典

1. はじめに

近年の半導体集積技術の進展に伴いLSIの回路規模が増大し，テスト設計の工数が増大しており，その自動化技術が重要になってきている．一般の順序回路の自動テストパターン生成（ＡＴＰＧ）は困難な問題であり，高い故障検出効率を得るテストパターンを生成するには，フルスキャン[1)]，[2)]に代表されるテスト容易化設計（DFT）が必要である．

フルスキャン設計方法におけるテスタでのテスト実行時間は，テスト長とテスタ動作のクロック周期の積から計算できる．テスト長は，スキャンパスを取り除いた核回路(組合せ回路)のテストパターン数(ATPGパターン数)と最大スキャンパス長の積に比例する．

LSIの大規模化が進んでいることから，回路内のゲート数が激増している．ITRS[3)]のロードマップによると，ゲート数の増加に伴い外部ピンの数も増加しているが，外部ピンの増加率はゲート数の増加率に比べて鈍いと報告されている．FF数はゲート数に比例し，またスキャンパス数は外部ピン数に比例することから，フルスキャン設計においては，一つのスキャンパス上のFF数（スキャンパス長）が増加してしまう．さらに，一般的にゲート数が増加すれば，ATPGパターン数も増加する．このことより，フルスキャン設計方法では今後テスト実行時間が大幅に増大することが予測される．

本稿では，ATPGパターン数を削減することによってテスト実行時間の削減を図る．ATPGパターン数を削減するための回路構造に基づいたテストポイント挿入アルゴリズムを提案する．

テストポイント挿入技術は，これまで主に組み込み自己テスト(BIST)[4)]などのランダムテスト環境での故障検出率の向上のために用いられていた．しかし，本稿ではATPGパターン数を削減することを目的とし，テストポイント挿入技術を用いる．本研究では，テストポイント挿入位置を決定するための指標として，故障検出確率を提案し，与えられたテストポイン数で出来るだけ故障検出率を低下されるためのテストポイント挿入アルゴリズムを提案する．

2. テストポイント

テストポイントとはある信号線の可制御性と可観測性[5)]を高めるために付加する論理回路のことである．テスト対象回路の内部信号線にテストポイントを挿入することによって，その信号線はテスト時に擬似的に外部入力，外部出力とみなすことができる．テストポイント挿入の影響でいくつかの信号線の可制御性，可観測性が改善される．図1は信号線 l にテストポイントを挿入したことにより可制御性，可観測性改善される範囲を示している．

テストポイント挿入の目的は二つある．一つは，擬似ランダムなテストパターンで回路をテストする際に，検出困難となる故障を減少させることである．もう一つは，ATPGによるテストパターン数を削減することである．

本稿では，指定されたテストポイント数でより多くのATPGパターン数を削減することを目的とする．

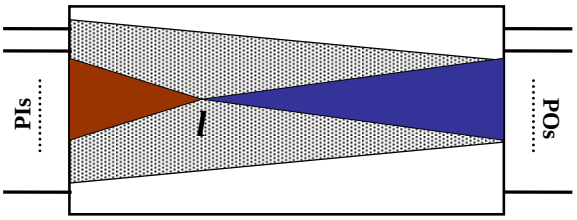

図1：信号線l にテストポイント挿入時の回路への影響

3. 故障検出確率

テストパターン圧縮の効果を表す尺度として故障検出確率を定義する．回路中の信号線数をL，信号線をl_i（i＝1，2，．．．，L），外部出力数をM，外部出力をPO_j（j＝1，2，．．．，M），外部入力数をN，外部入力をPI_k（k＝1，2，．．．，N），とする．また，信号線に対して，0縮退故障と1縮退故障を区別しない一つの故障を仮定する．

A Test Point Insertion Method Based on Fault Detection Probability to Reduce the Number of ATPG Patterns

Yoshihiro SAITO,　and Toshinori HOSOKAWA,

3．1　故障検出確率の定義

外部出力PO_jの故障検出確率を$FDP(PO_j)$とすると，$FDP(PO_j)$は式(1)に示される．ただしR（a, b）は式(2)に示す関係式とする．

$$FDP\ (PO_j) = \frac{1}{L}\sum_{p=1}^{L}\frac{R(l_p, PO_j)}{\sum_{q=1}^{M} R(l_p, PO_q)}. \tag{1}$$

$$R(a,b) = \begin{cases} 0(\text{a is not reachable to b}) \\ 1(\text{a is reachable to b}) \end{cases} \tag{2}$$

故障検出確率は外部出力PO_jに定義され，全信号線数Ｌに対する外部出力PO_jから入力側に到達可能な信号線数の割合と定義する．到達可能な信号線l_iの故障はp本の外部出力で検出できる可能性があり，これらp本の外部出力で等しい確率で検出できると仮定する．

また式(2)のa ， bは信号線であり，信号線a から出力側に探索したときの信号線bへの到達可能性を示す関数である．信号線aから出力側に探索し，信号線 b に到達不可能である場合，式(2)は0となり，到達可能である場合は，式(2)は1となる関数である．

[例1] 故障検出確率

ここでは故障検出確率の説明を行う．図1の回路では，PO_1は信号線1～5，7～10，12から到達可能である．同様にPO_2信号線2，3，4，7，9，11，13から到達可能であり，PO_3は信号線2，6から到達可能である．また信号線2はPO_1，PO_2，PO_3へそれぞれ到達可能であり，信号線3，4，7，9は， PO_1，PO_2へそれぞれ到達可能である．よって式(1)に基づきPO_1の故障確率は式(3)のように$FDP(PO_1)=56.4(\%)$と求められる．同様にPO_2の故障検出確率も式(4)のように$FDP(PO_2)=33.3(\%)$，PO_3の故障検出確率も，式(5)のように$FDP(PO_3)=10.3(\%)$と求められる．

3．2　故障検出確率の正当性

3.1章で定義した故障検出確率が高いということは，ある外部出力に故障の検出が集中している．その外部出力から到達可能な外部入力に0, 1の割り当てが集中する可能性が高くなる．外部入力に値の割り当て要求が集中すると，その外部入力がネックとなりテストパターン数が増加してしまう．そこで，ある外部出力が値の割り

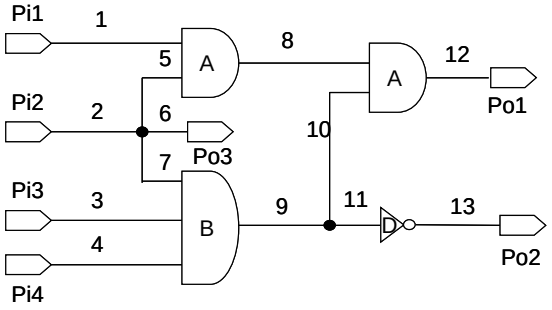

図:2　組合せ回路

$$FDP\ (PO_1) = \frac{5 + \frac{4}{2} + \frac{1}{3}}{13} = 56.4(\%) \tag{3}$$

$$FDP\ (PO_2) = \frac{2 + \frac{4}{2} + \frac{1}{3}}{13} = 33.3(\%) \tag{4}$$

$$FDP\ (PO_2) = \frac{1 + \frac{1}{3}}{13} = 10.3(\%) \tag{5}$$

当てが集中しないようにするために，故障検出確率が高い外部出力をターゲットとして，その故障検出確率を減らすようにテストポイントを挿入する．その結果テストパターン数の削減に効果があると考えられる．

4．故障検出確率を削減するためのテストポイント挿入箇所探索アルゴリズム

故障検出確率を削減するテストポイント挿入アルゴリズムを提案する．アルゴリズムを図2に示す．このテストポイント挿入アルゴリズムは，一つずつ故障検出確率のもっとも高い外部出力の入力側を探索し，各信号線に対してテストポイントを仮に挿入し，故障検出確率の最大値がもっとも小さくなる信号線を一つ求め，その信号線をテストポイント挿入箇所と決定する．上記の処理を指定されたテストポイント数回繰り返すアルゴリズムである．

関数Reduce_FDP () は，組み合わせ回路(CC)とテストポイント挿入数tを入力とし，テストポイント挿入信号線の集合(TP)を出力とする関数である．まず，TPを空集合に初期化し(3行目)，t回テストポイントの挿入信号線の探索(4行目から25行目)を繰り返す．

挿入信号線の探索部分では，まず，すべての外部出力の中でもっとも故障検出確率の高い外部出力(MAX_PO)を探索する(6行目)．故障検出確率の最大値(MAX_FDP)をMAX_POの故障検出確率とする．(7行目)．テストポイント挿

入箇所探索の対象となる信号線の集合(SEARCH_BR) を，MAX_POの入力側の信号線とする．(8行目)SAERCH_BRより一つ信号線*l*を取り出し，SEARCH_BRが空になるまで，信号線*l*に対して, 9行目から22行目までの処理を繰り返す．

まず，今回探索する信号線*l*をSEARCH_BRから削除し(11行目)，信号線*l*に仮にテストポイントを挿入した場合の故障検出確率(FDP)を算出する．(12行目)．次に算出した故障検出率の値に基づき，信号線*l*とテストポイント挿入を行ったことで新しく更新されたMAX_POの最大値を求め(13行目)，もし従来の挿入箇所よりも故障検出確率の最大値が小さい場合，仮テストポイン挿入箇所信号線BRを*l*に更新(15行目)．
仮テストポイント挿入後の故障検出確率の最大値で，MAX_FDPを更新する(16行目)．更に，MAX_POの故障検出確率が信号線*l*の故障検出確率の2倍以下であれば, *l*の入力側にテストポイント挿入箇所信号線の探索を行う．(18行目)．*l*の入力側に探索を行う場合, SEARCH_BRに信号線*l*の入力側のゲートに接続する全ての信号線を加える．(20行目)SEARCH_BRが空になり， i回目のテストポイント挿入箇所信号線の探索が終了したとき，BRをi回目のテストポイント挿入箇所とし，BRにテストポイントを挿入し，故障検出確率を再計算し，(23行目)BRをTPに加える，(24行目) t回テストポイント挿入箇所を探索し終えたとき，テストポイント挿入箇所の集合であるTPが求まる．(26行目)

```
1:   Reduce_FDP (CC,t)
2: {
3:   Initial TP to Φ.
4:   for(i = 1 ; i ≦ t ; i++)
5:   {
6:     Find MAX_PO with maximum FDP
7:     MAX_FDP = FDP of MAX_PO
8:     SEARCH_BR = single lines which are all inputs of MAX_PO
9:     while(SEACH_BR ≠  Φ)
10:       {
11:     Delete a signal a line l in SEARCH_BR
12:     Assuming that a test point is inserted into l, calclate FDP
13:     if(max (FDP of MAX_PO,FDP of l,<MAX_FDP))
14:     {
15:       BR = l
16:       MAX_FDP
        =max (FDP of MAX_PO,FDP of l
17:     }
18:     if(FDP of MAXPO < (2 × FDP of l))
19:     {
20:       Add signal lines which are all inputs of l to SEARCH_BR
21:     }
22:    }
23:     Insert test point into BR and calculate FDP
24:     Add of BR to TP
25:   }
26:   return (TP)
27: }
```

図 1　テストポイント挿入アルゴリズム

5. 実験結果

4章で提案したアルゴリズムを実装し，ISCAS’85ベンチーマーク回路に対して実験を行った．本実験では，テストポイントを回路全体の信号線の1%を挿入した．また，実験にはCeleron(R) 2．40GHz のCPUをもつ計算機を用い，ATPGテストパターン生成ツールとしてTetraMAXを用いた．

故障検出確率に基づくテストポイント挿入によってATPGパターン数がどの程度減少したかを調べた．比較対象として，同じ数のテストポイント数をランダムに挿入したものを用いた．二つのアルゴリズムと，テストポイントを挿入せずに実験を行った結果を表2に示す．TP数はテストポイント挿入数，ATPG数はATPGテストパターン数，TCPUはテストポイント探索に要したCPU時間(秒)，ACPUは，ATPGに要したCPU時間(秒)，検出率は故障検出率，削減率は，テストポイント挿入によってATPGパターン数がどの程度削減されたのかを示す割合である．削減率は式(6)に表す．式(6)のRRは削減率，#CTはテストポイント挿入前のATPGパターン数，#ACTはテストポイント挿入後のATPGパターン数を示している．

$$RR = \frac{\#CT - \#ACT}{\#CT} \qquad (6)$$

表1は4章で提案したアルゴリズムを用いたときの故障検出率の最大値の変化を示す．表1ののBFDPは挿入前の故障検出確率, AFDPは挿入前の故障検出確率を表す. 表2に示すとおり，テスト回路に対して，4章で提案したアルゴリズムを用いてテストポイントを挿入することによって，最大でATPGパターン数を72%削減することができた. テストポイント挿入位置の探索に要する時間も，ほぼ20秒以内で探索することができた.

表１　故障検出確率

回路名	BFDP(%)	AFDP(%)
C432	29.5	18.9
C499	3.1	3.1
C880	10.1	7.4
C1355	3.1	3.1
C1908	28.4	3.9
C2670	14.2	2.8
C3540	14.4	2,3
C5315	8.2	1.2
C6288	6.5	1.7
C7552	11.2	1.1

今回の実験では，c7552，c2670，c5315のように40%近くパターン数を削減できたものがあった．その3つの回路は，表1に示すように，故障検出確率もかなり減少している．ただ，c6288のように，故障検出確率が改善させているにもかかわらず，思うようにテストパターン数が削減されず，ランダムにテストパターンを入れた回路のほうが，よりいい結果が得られたものもあった．c1355のように，故障検出確率の最大値が改善されていないのにもかかわらず，テストパターン数がかなり削減されるものもあった．これは故障検出確率の定義のあいまいさがあるために起こっていると考えられる．

5. おわりに

今回故障検出確率を定義し，それを指標としたテストポイン挿入アルゴリズムを実装し，テスト生成を行い，テストパターン数の削減結果を示した．表2の結果より，より効果的な位置のテストポイントを挿入するための指標が昼用であることがわかる． 今後は，もっとATPGツールの故障伝搬を考慮するように，故障検出率を改善し，テストパターン数のさらなる減少を考えている．

表2　テストパターン生成の結果

回路名	アルゴリズム	TP数	ATPG数	検出率(%)	削減率(%)	冗長故障数	TCPU(sec)	ACPU(sec)
C432	Not DFT	0	50	100	1	13	0	0.03
	FDP	4	47	100	6	1	0	0.01
	Randam	4	45	100	10	1	0	0.03
C499	Not DFT	0	80	99.85	1	8	0	0.11
	FDP	4	79	99.93	1	4	0	0.06
	Randam	4	88	99.96	-10	10	0	0.04
C880	Not DFT	0	34	100	1	0	0	0.03
	FDP	8	29	100	15	0	0	0.03
	Randam	8	34	100	1	0	0	0.05
C1355	Not DFT	0	96	100	1	8	0	0.18
	FDP	13	64	100	33	20	0	0.14
	Randam	13	98	100	-2	8	0	0.13
C1908	Not DFT	0	126	100	1	13	0	019
	FDP	19	118	100	6	4	0	0.16
	Randam	19	130	100	-3	15	0	0.13
C2670	Not DFT	0	72	100	1	253	0	0.23
	FDP	26	34	100	53	116	3	0.13
	Randam	26	75	100	4	196	0	0.16
C3540	Not DFT	0	130	100	1	349	0	0.48
	FDP	35	116	100	11	256	2	0.39
	Randam	35	116	100	11	312	0	0.31
C5315	Not DFT	0	96	100	1	63	0	0.36
	FDP	53	59	100	39	59	13	0.25
	Randam	53	86	100	10	64	0	0.34
C6288	Not DFT	0	38	100	1	85	0	1.370
	FDP	56	32	99.97	16	87	21	1.08
	Randam	56	26	100	31	84	0	0.42
c7552	Not DFT	0	114	100	1	303	0	0.65
	FDP	75	32	99.97	72	87	7	1.09
	Randam	75	100	100	12	191	0	0.54

［参考文献］

1) H.Fujiwara, “Logic Testing and Design for Testability”, The MIT Press, 1985.

2) M Abramovici, M.A.Breuer, and A.D.Frindman, “Digital Systems Testing and Testable Design,” Computer Science Press, 1990.

3) International Technology Rodemap for Semiconductors 1999 Edition, Semiconductor Industry Association, 1999.

4)Huaguo Liang, ”A New Technique for Deterministic Scan-Based Built-In Self-Test(BIST)”, SHAKER VERLAG, 2003

5)J. Geuzebroek, “Test Point Insertion to improve BIST performance, and to reduce ATPG test time and data volume”, DUP Science, ２００３