

マルチサイクルキャプチャテスト生成モデルにおけるテスト生成時間とテスト長のトレードオフ解析に関する研究

日大生産工(学部) ○小河 宏志 日大生産工(院) 大森 悠翔
日大生産工 細川 利典 明大 山崎 浩二

1. はじめに

近年, 大規模回路に対するテストコスト削減のため, ATPG(automatic test pattern generator)が利用されている. しかし順序回路に対しては, 高故障検出効率のテストパターンを生成するのは困難である.

そこでテスト容易化設計として, フルスキャン設計¹⁾が提案されている. フルスキャン設計は, 回路中の FF(フリップフロップ)を全てスキャン FF に置き換え, シフトレジスタ状に接続(スキャンチェーン)したものである. テスト時には, フルスキャン回路からスキャンチェーンを取り除いた回路(核回路)が組合せ回路となるので組合せテスト生成アルゴリズムが適用でき, 高故障検出効率を得ている. しかしそのテスト長は(スキャン FF 数) × (核回路のテストパターン数)に比例するため, テスト長は回路の大規模化により膨大になる.

そこで本稿では不必要なスキャン化を抑えるために k マルチサイクルキャプチャを行うことにより, スキャン FF の選択を行ない, シングルクロックパーシャルスキャン回路を生成する k の値の増加により, シフト回数の削減やテスト生成モデルの大規模化が予測されるので, ISCAS' 89 ベンチマーク回路を用いて, テスト長とテスト生成時間のトレードオフ解析を試みた.

2. k 時間展開によるスキャン FF 決定法

2.1 k 時間展開

(定義: k 時間展開)

時刻 0 でスキャン FF の出力を擬似外部入力(PPI)とし, 時刻 $k-1$ でスキャン FF のデータ入力を擬似外部出力(PPO)として k 時間分回路を展開することである.

図 1 は順序回路例である. 図 1 において CC1 ~CC4 は組み合わせ回路部分で, FF1~FF9 は FF である. 図 2 は図 1 に対して外部出力(PO)から外部入力(PI)方向へ 2 時間展開を行い, スキャン FF の決定を行う概念図である. これにより FF4, FF5, FF6, FF7 がスキャン FF として決定される.

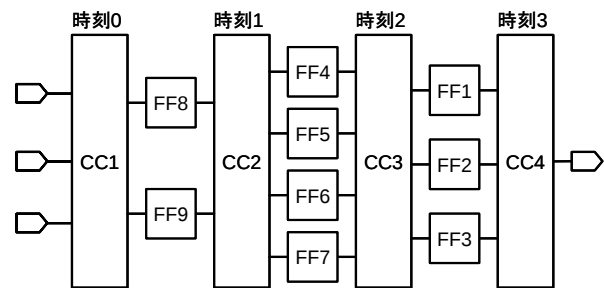


図 1. 順序回路例

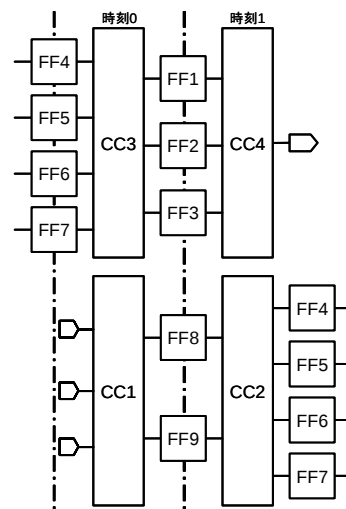


図 2. 2 時間展開によるスキャン FF の決定

Trade-off Analysis between Test Generation Time and Test Length for Multi-cycle Capture Test Generation Model

Hiroshi OGAWA, Yusho OMORI,
Toshinori HOSOKAWA and Koji YAMAZAKI

2.2 可制御用スキャン FF と可観測用スキャン FF

一般にスキャン FF は、制御(スキャンイン)と観測(スキャンアウト)を行う。

可制御性とは回路内部の信号線に対して論理値を外部からの入力で設定する容易性を示す。また、可観測性とは回路内部の信号線に対して、その論理値を外部で観測する容易性を示す。一般的に可観測性と可制御性が高いほど有効なテストパターンの生成が容易である。

スキャン FF は、外部入力と外部出力の役割を兼ね備えている。

今回はスキャン FF であっても制御用、観測用とする決定を行った。これにより観測用のスキャン FF には制御を行う必要がないため、シフト回数および、テスト長の削減が期待できる。

2.3 制御用と観測用のスキャン FF 決定

以下に、制御用と観測用のスキャン FF 決定について説明する。

制御用のスキャン FF 決定は、PO 又は観測用のスキャン FF から PI 方向へ k 時間展開を行い、時刻 0 に現れた FF を制御用スキャン FF とする。処理は全ての FF を探索した時点で終了する。

観測用のスキャン FF 決定は、制御用のスキャン FF の入力側の回路部分がまだ k 時間展開回路に現れていない場合に、その制御用のスキャン FF を観測用のスキャン FF とする。また、制御用スキャン FF から PO 方向に順序深度 k-1 離れた FF を観測用 FF とする。

以下に図 1 の順序回路例に対して 3 時間展開による制御用スキャン FF 決定と、観測用スキャン FF 決定を行った場合について説明する。

図 1 の順序回路例に対して、図 3 のように PO から PI 方向に 3 時間展開による制御用スキャン FF 決定を全ての FF を探索するまで行う。これにより、FF8 と FF9 が制御用のスキャン FF として決定される。

次に全 PI と制御用のスキャン FF である FF8 と FF9 から PO 方向に順序深度 2 離れた FF を観測用スキャン FF として決定する。ただし FF8 と FF9 から PO 方向に順序深度 2 離れると、PO に到達するので観測用となるスキャン FF は決定されない。

PI から PO 方向に順序深度 2 離れて観測用スキャン FF の決定を行った場合、観測用スキャン FF として FF1 と FF2、FF3 が決定される。

図 4 は図 1 の順序回路例に 3 時間展開によるス

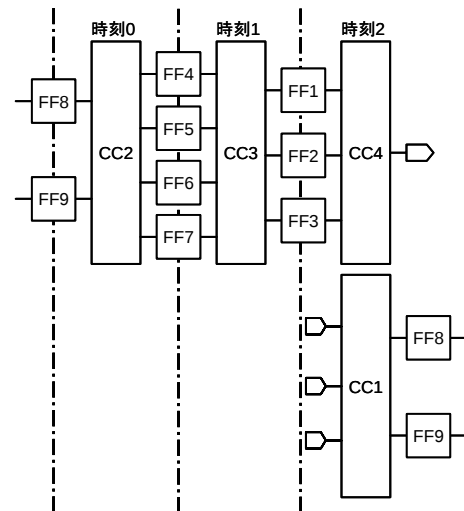


図 3. 3 時間展開による制御用スキャン FF 決定

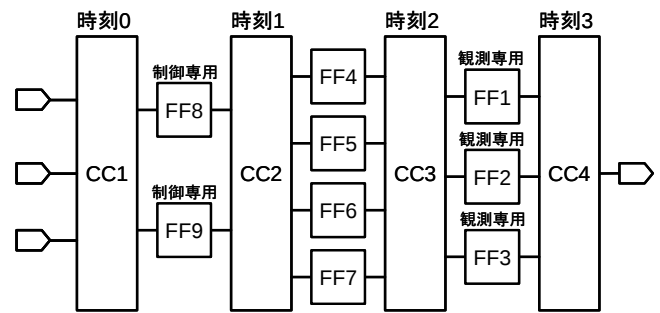


図 4. 制御用と観測用のスキャン FF 決定

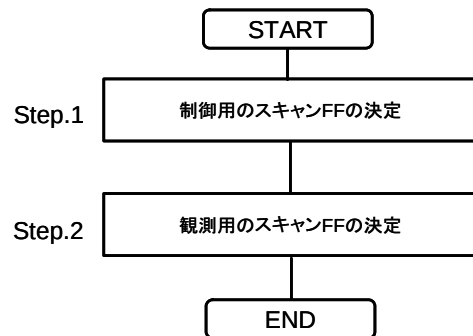


図 5. スキャン FF 決定法全体アルゴリズム

キャン FF 決定を行うことにより、制御用スキャン FF 決定と、全 PI と観測用スキャン FF から PO 方向に順序深度 2 離れた FF を観測用スキャン FF と決定した場合の順序回路図である。

2.4 スキャン FF 決定法全体アルゴリズム

図 5 はスキャン FF 決定法全体アルゴリズムのフローチャートである。

各 Step の処理は以下の通りである。

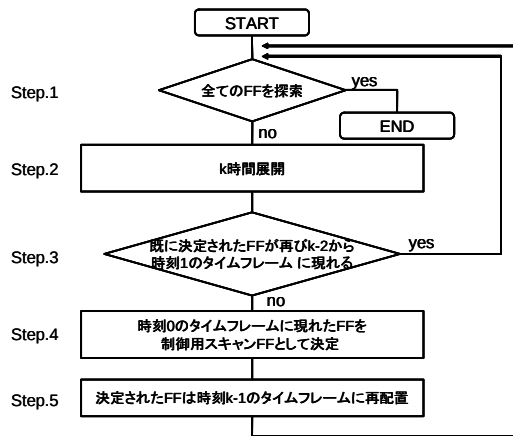


図 6. 制御用スキャン FF 決定アルゴリズム

(Step.1)

PO から PI 方向に k 時間展開による制御専用のスキャン FF の決定を，全ての FF を探索するまで行う。

(Step.2)

全 PI と制御用のスキャン FF から PO 方向に順序深度 $k-1$ 離れた FF を観測用のスキャン FF として決定する。

2.5 k 時間展開による制御用スキャン FF 決定アルゴリズム

k 時間展開による制御用スキャン FF 決定アルゴリズムについて説明する。図 6 は制御用スキャン FF 決定アルゴリズムのフローチャートである。

各 Step の処理は以下の通りである。

(Step.1)

全ての FF を探索するまで，Step.2～Step.5 を繰り返す。全ての FF を探索し終えたら処理を終了する。

(Step.2)

順序回路を k 時間展開する。ただし初めのみ PO から PI 方向へ k 時間展開を行う。これ以降は，時刻 $k-1$ のタイムフレームに再配置された FF より k 時間展開を行う。

(Step.3)

すでにスキャン化 FF として決定された FF が再び時刻 $k-2$ から時刻 1 のタイムフレームに現れた場合は Step.1 へ，そうでない場合は Step.4 へと移る。これにより一度処理を行った経路での処理を行わないため，処理時間の短縮となる。

(Step.4)

時刻 0 のタイムフレームに現れた FF を制御用

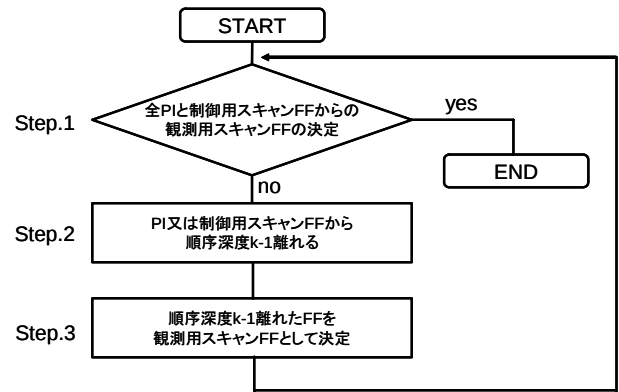


図 7. 観測用スキャン FF 決定アルゴリズム

スキャン FF として決定する。

(Step.5)

Step.4 で決定された制御用スキャン FF は時刻 $k-1$ のタイムフレームに再配置され，Step.1 に戻る。

2.6 k 時間展開による観測用スキャン FF 決定アルゴリズム

観測用スキャン FF 決定アルゴリズムについて説明する。図 7 は観測用のスキャン FF 決定アルゴリズムのフローチャートである。

各 Step の処理は以下の通りである。

(Step.1)

全 PI と制御用スキャン FF からの観測用スキャン FF 決定が終わるまで，Step.2～Step.3 を繰り返す。全 PI と制御用スキャン FF からの観測用スキャン FF 決定が終わったら処理を終了する。

(Step.2)

全 PI と制御専用スキャン FF から PO 方向へ順序深度 $k-1$ 離れる。

(Step.3)

順序深度 $k-1$ 離れた FF を観測用スキャン FF として決定する。ただし順序深度 $k-1$ 離れたときに，PO に到達した場合は観測用スキャン FF の決定は行われない。

3. 実験結果

k マルチサイクルキャプチャテスト生成でフルスキャン時と k 時間展開によるスキャン FF 決定時について，テスト生成時間とテスト長について ISCAS'89 ベンチマーク回路に対して実験を行った。

実験環境は，OS は RedHat，CPU は Pentium4 3.0GHz メインメモリは 2GB である。

表.1 の構成は，ISCAS'89 ベンチマーク回路の解析結果で回路名，全 FF 数，セルフループ数 (S_LOOP) となっている。S_LOOP での 0 内の数

字はセルフループ数の割合をパーセンテージで載せている。

表.2 はフルスキャン時の実験結果を載せている。表.3～6 は $k(k=2\sim5)$ マルチサイクルキャプチャテスト生成を行った時の実験結果を載せている。表.2～表.6 の構成は回路名、スキャン化率とテスト長、故障検出効率、テスト生成時間となっている。

表.1 ISCAS'89 ベンチマーク回路解析

回路名	全FF数	S LOOP
s5378	179	0(0)
s9234	228	150(65.79)
s13207	669	310(46.34)
s15850	597	438(73.37)
s35932	1728	288(16.67)
s38417	1636	1078(65.89)
s38584	1452	1098(75.62)

表.2 フルスキャンでの実験結果

回路名	フルスキャン			
	スキャン化率(%)	テスト長	故障検出効率(%)	CPU time(s)
s5378	100	777959	100	2.74
s9234	100	1414532	99.82	9.79
s13207	100	5924809	99.67	13.26
s15850	100	6170761	99.72	19.69
s35932	100	54731494	100	327.18
s38417	100	48465021	100	45.28
s38584	100	46543948	100	261.96

表.3 2 マルチサイクルキャプチャでの実験結果

回路名	2時刻			
	スキャン化率(%)	テスト長	故障検出効率(%)	CPU time(s)
s5378	98.32	718940	98.83	10.36
s9234	98.68	1300254	94.31	53.64
s13207	84.75	4427956	98.68	45.57
s15850	96.65	5754679	97.19	79.94
s35932	100	54763148	100	969.99
s38417	96.58	44961274	96.57	168.15
s38584	99.93	46189415	98.97	756.19

表.4 3 マルチサイクルキャプチャでの実験結果

回路名	3時刻			
	スキャン化率(%)	テスト長	故障検出効率(%)	CPU time(s)
s5378	94.41	689889	98.8	19.63
s9234	99.12	1283542	91.61	103.41
s13207	97.01	5208825	93.02	173.16
s15850	96.82	5760612	97.06	180.86
s35932	100	54794802	100	1778.46
s38417	98.59	46422829	97.35	413.81
s38584	100	46148232	98.56	1637.49

表.5 4 マルチサイクルキャプチャでの実験結果

回路名	4時刻			
	スキャン化率(%)	テスト長	故障検出効率(%)	CPU time(s)
s5378	100	717905	97.72	36.14
s9234	99.12	1141946	80.84	215.25
s13207	98.51	5327864	91.67	261.61
s15850	97.65	5830667	96.37	305.77
s35932	100	54826456	100	*
s38417	100	47676436	97.77	900.29
s38584	100	45036988	96.04	*

表.6 5 マルチサイクルキャプチャでの実験結果

回路名	5時刻			
	スキャン化率(%)	テスト長	故障検出効率(%)	CPU time(s)
s5378	100	699747	94.98	61.95
s9234	99.56	1095499	76.27	320.2
s13207	99.1	4905758	83.82	505.35
s15850	97.65	5597167	92.5	473.96
s35932	100	54858110	100	*
s38417	100	47564380	97.79	1704.3
s38584	100	44346704	94.64	667.82

実験結果の表.3～表.6 を見ると s5378 と s15850, s38417 では時間展開数を増やした場合も、故障検出効率の誤差が 1%以下となっている。ただしテスト長とテスト生成時間を 2～5 時間展開で比べた場合、2 時間展開が 3～5 時間展開よりテスト長もテスト生成時間も短くなっていることが分かる。また他の回路で故障検出効率の誤差を考えた場合も 3, 4, 5 時間展開では大きく故障検出効率下がっている、2 時間展開が一番良いと考えられる。

次にフルスキャン時のテスト長と k 時間展開でのテスト長の比較を行う。この比較では、2～5 時間展開時のほうがフルスキャン時に比べて、全ての回路でテスト長の削減が成功している。

4. おわりに

本稿では k 時間展開を用いたスキャン化率の調査と、 k 時間展開でのスキャン化率で ISCAS'89 ベンチマーク回路に対してテスト生成時間とテスト長のトレードオフ解析を行った。実験結果より、テスト生成時間とテスト長のトレードオフに関しては、展開時間数を延ばすことによりフルスキャン時よりテスト長の削減は成功しているが、逆にテスト生成時間が延びてしまうという結果となった。

今後の課題として、高検出効率を維持したまま、さらなるテスト長の削減が挙げられる。

参考文献

- 1) M. Abramovici, M.A. Breuer, and A.D. Friedman, "Digital systems testing and testable design" IEEE Press, 1955.
- 2) H. Fujiwara, "Logic Testing and Design for Testability" The MIT Press, 1985
- 3) Srimat T. Chakradhar, Arum Balakrishnak, Vishwaki D. Agrawal, "AN EXACT ALGORITHM FOR SELECTING PARTIAL SCAK FLIP-FLOPS"
- 4) International Technology Roadmap for Semiconductors 1999 Edition, Semiconductor Industry Association, 1999