LSI 歩留まり向上のための誤テスト回避型テスト方式に関する研究

17500039

平成17年度～平成18年度　科学研究補助金

（基盤研究（C））　研究成果報告書

平成19年5月

研究代表者　温　暁青

九州工業大学大学院情報工学研究科教授

はしがき

LSIテストとは、設計データに基づいて製造されたLSI回路に対して、テスト入力を印加しそれに対するテスト応答を期待値と比較して良品・不良品の判定を行う重要な作業である。近年、LSI回路の超大規模化・超微細化に伴い、正常なLSI回路を不良品として誤って判定してしまうという誤テストが多発するようになってきている。これは、LSI回路の歩留まり低下を招き、半導体産業に大きな打撃を与える危険性がある。

テスト応答が回路内の記憶素子に取り込まれるときに、同時に状態変化する記憶素子が多すぎると電源電圧が一時降下するため、記憶素子が誤動作して誤ったテスト応答を取り込んでしまい、誤テストが発生する。明らかに、誤テストを回避するためには、同時に状態変化する記憶素子の数を少なくする必要がある。本研究では、テスト時の状態変化数を削減することによる誤テスト回避を試みた。

本報告書は、平成17年度～平成18年度に科学研究補助金を受けて行なった研究の成果をまとめたものである。現在も半導体集積回路の誤テストによる歩留まり低下問題が深刻化しつつある。本研究で開発した誤テスト回避テスト手法で完結するものではなく、更なる研究が今後も必要である。本研究はその礎ともなりうるので、本研究の研究成果を本報告書で整理し報告する。

研究組織

研究代表者：　温 暁青（九州工業大学大学院情報工学研究科教授）
（研究協力者：　梶原 誠司）

交付決定額（配分額）

（金額単位：円）

	直接経費	間接経費	合　計
平成 17 年	1,400,000	0	1,400,000
平成 18 年	700,000	0	700,000
総　　計	2,100,000	0	2,100,000

研究発表

(1) 学会誌等

[1] 温暁青, 梶原誠司, 玉本英夫, K. K. Saluja, 樹下行三 : 中間故障電圧値を扱う故障シミュレーションの高速化について、電子情報通信学会論文誌 D-I, Vol. J88-D-I, No. 4, pp. 906-907 (2005).

[2] X. Wen, T. Suzuki, S. Kajihara, K. Miyase, Y. Minamoto, L.-T. Wang, and K. K. Saluja : Efficient Test Set Modification for Capture Power Reduction、J. of Low Power Electronics, Issue 3, pp. 319-330 (2005).

[3] X. Wen, Y. Yamashita, S. Kajiihara, L.-T. Wang, K. K. Saluja, and K. Kinoshita : A New Method for Low-Capture-Power Test Generation for Scan Testing、IEICE Trans. Inf. & Syst., Vol. E89-D, No. 5, pp. 1679-1686 (2006).

(2) 口頭発表

・ 国際会議

[1] X. Wen, Y. Yamashita, S. Kajiihara, L.-T. Wang, K. K. Saluja, and K. Kinoshita : On Low-Capture-Power Test Generation for Scan Testing、Prof. IEEE VLSI Test Symp., pp. 265-270 (2005).

[2] K. Miyase, S. Nagayama, S. Kajihara, X. Wen, and S. M. Reddy : On the Extraction of a Minimum Cube to Justify Signal Line Values、Informal Digest of IEEE European Test Symp., pp. 79-84 (2005).

[3] X. Wen, Y. Yamashita, S. Morishima, S. Kajiihara, L.-T. Wang, K. K. Saluja, and K. Kinoshita : A Method for Low-Capture-Power At-Speed Test Generation、Proc. IEEE Workshop on RTL and High Level Testing, pp. 40-49 (2005).

[4] L.-T. Wang, X. Wen, B. Hsu, S. Wu, and J. Guo : A Flexible Logic BIST Scheme for Multiple-Clock Circuits、Proc. IEEE Int'l Conf. on Computer Design, pp. 475-478 (2005).

[5] X. Wen, Y. Yamashita, S. Morishima, S. Kajiihara, L.-T. Wang, K. K. Saluja, and K. Kinoshita : Low-Capture-Power Test Generation for At-Speed Scan Testing、Proc. IEEE Int'l Test Conf., Paper. 39.2 (2005).

[6] K. Miyase, K. Terashima, S. Kajihara, X. Wen, and S. M. Reddy : On Improving Defect Coverage of Stuck-at Fault Tests、Proc. IEEE Asian Test Symp., pp. 216-221 (2005).

[7] X. Wen, S. Kajihara, K. Miyase, T. Suzuki, K. K. Saluja, L.-T. Wang, K. S. Abdel-Hafez, and K. Kinoshita : A New ATPG Method for Efficient Capture Power Reduction During Scan Testing、Proc. IEEE VLSI Test Symp., pp. 58-63 (2006).

[8] K. Taniguchi, H. Fujii, S. Kajihara, and X. Wen : Hybrid Fault Simulation with Compiled and Event-Driven Methods、Proc. IEEE Int'l Conf. on Design & Test of Integrated Systems in Nanoscale Technology (DTIS), pp. 240-243 (2006).

[9] X. Wen, K. Miyase, T. Suzuki, Y. Yamato, S. Kajihara, L.-T. Wang, and K. K. Saluja : A Highly-Guided X-Filling Method for Effective Low-Capture-Power Scan Test Generation、Proc. IEEE Int'l Conf. on Computer Design, pp. 251-258 (2006).

[10] H. Furukawa, X. Wen, L.-T. Wang, B. Sheu, Z. Jiang, and S. Wu : A Novel and Practical Control Scheme for Inter-Clock At-Speed Testing、Proc. IEEE Int'l Test Conf., Paper 17.2 (2006).

- **国内学会**

[1] 鈴木達也、温暁青、梶原誠司、宮瀬紘平、皆本義弘： スキャンテストにおけるキャプチャ時の低消費電力化に効果的なテスト集合変更について、電子情報通信学会技術研究報告, Vol. 105, No. 443, 446, pp. 1-6 (2005).

[2] 谷口謙二郎、宮瀬紘平、梶原誠司、温暁青：ブロードサイドテストにおけるN回検出用テストパターンに対するX判定、電子情報通信学会技術研究報告 Vol. 106, No. 387, pp. 35-40 (2006).

[3] 藤井秀雄, 谷口謙二郎, 梶原誠司, 温暁青： 順序回路用故障シミュレーションにおけるコンパイル方式の適用と効果について、電子情報通信学会技術研究報告, Vol. 106, No. 528, pp. 13-18 (2007).

[4] 塔ノ上義章, 温暁青, 梶原誠司, 宮瀬紘平, 鈴木達也, 大和勇太： 低消費電力テストのための制約付テスト生成手法について、電子情報通信学会技術研究報告, Vol. 106, No. 467, pp. 109-114 (2007).

- **国内研究会**

[1] 藤井秀雄, 谷口謙二郎, 梶原誠司, 温暁青: コンパイル方式とイベント駆動方式を用いた故障シミュレーションの高速化について、第55回 FTC 研究会資料集 (2006) .

[2] 山本真裕, 森島翔平, 梶原誠司, 温暁青, 福永昌勉, 畠山一実, 相京隆: 遅延テスト品質の正確な評価法とテスト生成への応用、第56回 FTC 研究会資料集 (2007) .

(3) 研究成果による工業所有権の出願・取得状況

[1] 温暁青、梶原誠司：半導体論理回路装置のテスト方法、装置及び半導体論理回路装置のテストプログラムを記憶した記憶媒体
特許出願 2005-130806、出願日：2005年4月28日

[2] 温暁青、梶原誠司：半導体論理回路装置のテストベクトル生成方法、装置及び半導体論理回路装置のテストベクトル生成プログラムを記憶した記憶媒体
特許出願 2005-215214、出願日：2005年7月26日

研究成果

目次

1. 総論

LSIテストは、回路の品質と信頼性を決定付ける重要な作業である。しかし近年では、LSI回路の超大規模化・超微細化に伴い、正常なLSI回路を不良品として誤って判定してしまうという誤テストが多発するようになってきている。これは、LSI回路の歩留まり低下を招き、半導体産業に大きな打撃を与える危険性がある。

テスト応答が回路内の記憶素子に取り込まれるときに、同時に状態変化する素子が多すぎると電源電圧が降下するため、記憶素子が誤動作して誤ったテスト応答を取り込んでしまい、誤テストが発生する。本研究では、テスト時の回路内部信号変化数を削減することによる誤テスト回避を試みた。その研究成果を以下のような構成で報告する。

仮定した故障を与えられたテスト入力で検出できるか否かを調べる故障シミュレーション(**Fault Simulation**)は、先端テスト技術を開発する上で重要な基本技術である。特に、近年の超微細化と超大規模化によって、従来の縮退故障に基づく故障シミュレーションは精度と実行時間に関しては不十分になってきている。そこで本研究では、超微細化に伴う中間故障電圧現象、および、超大規模化に伴う実行時間の膨大化といった問題の解決方法を開発した。その成果は第２章にまとめられている。

最近の**SoC** (**System-on-Chip**) は同期回路であっても複数のクロックドメインを有することが多く、高テスト品質の実現するためには、すべてのクロックドメインの中の故障、および、クロックドメイン間の故障に対して、実速度テストを行なう必要がある。そこで本研究では、コンパクトで実現容易なテスト制御方式と回路設計を開発した。その成果は第３章にまとめられている。

テストベクトルの中間形態であるキューブの中に、故障検出率と無関係なビット（Xビット）が高い割合で存在することが示されている。しかし、テスト生成の段階からこのようなXビットを残す場合、最終テスト集合が膨大になる欠点がある。そのため、テスト生成の段階においてXビットにランダムな論理値を割り当てることによる動的圧縮を実施し、コンパクトな初期テスト集合を作る必要がある。そこで本研究では、このような論理値のみで構成される初期テスト集合から再び故障検出率と無関係なXビットを特定する手法を開発した。その成果は第４章にまとめられている。

テストベクトルの中間形態であるキューブの中に、故障検出率と無関係なビット（Xビット）が高い割合で存在することが示されている。これらのXビットに対して、テスト消費電力が低下するように最適な論理値を決定する研究開発が求められている。このような、Xビット埋め込みによる低テスト消費電力の実現は、回路変更に伴う面積増大・性能低下などの欠点がないため、高い実用性を有している。そこで本研究では、様々なXビット埋め込み手法を開発した。その成果は第５章にまとめられている。

テスト時の消費電力を削減するために、Xビットへの論理値埋め込みという後処理のみではなく、最初から最適な論理値を生成することも必要である。そこで本研究では、様々な低消費電力テスト生成手法を開発した。その成果は第５章にまとめられている。

2. 基本技術

仮定した故障を与えられたテスト入力で検出できるか否かを調べる故障シミュレーション(Fault Simulation)は、先端テスト技術を開発する上で重要な基本技術である。特に、近年の超微細化と超大規模化によって、従来の縮退故障に基づく故障シミュレーションは精度と実行時間に関しては不十分になってきている。従って、超微細化に伴う中間故障電圧現象、および、超大規模化に伴うCPU時間の膨大化といった問題の解決方法が求められている。

超微細化の集積回路において中間故障電圧値が多発するので、故障シミュレーションにおいてこの現象を扱う必要がある。そこで本研究では、トランジスタレベルの構造情報を用いて，トランジスタ短絡故障が引き起こす中間故障電圧値を扱う故障シミュレーションを高速化する手法を開発し、その有効性を実験で示した。その研究成果を３.１で紹介する。

故障シミュレーションの主な方式として、コンパイル方式とイベントドリブン方式がある。コンパイル方式では、すべての対象素子に対して論理値の評価を行なうが、プロセス制御にかかる時間が短い。一方、イベントドリブン方式では、論理値変化可能な素子のみに対して論理値の評価を行なうため処理時間が短いが、プロセス制御にかかる時間が長い。そこで本研究では、対象論理回路をFFR (Fanout-Free Region) に分割し、FFR内ではコンパイル方式、また、FFR間ではイベントドリブン方式を活用することによって故障シミュレーションの高速化手法を開発し、その有効性を実験で示した。その研究成果を３.２で紹介する。

2．1．中間故障電圧値を扱う故障シミュレーションの高速化手法

超微細化の集積回路において中間故障電圧値が多発するので、故障シミュレーションにおいてこの現象を扱う必要がある。トランジスタレベルの構造情報を用いて，トランジスタ短絡故障が引き起こす中間故障電圧値を扱う故障シミュレーションの高速化手法を本研究で開発し、その成果を次の学会誌で発表した。ここではその内容を紹介する。

温暁青，梶原誠司，玉本英夫，K. K. Saluja，樹下行三：
中間故障電圧値を扱う故障シミュレーションの高速化について
電子情報通信学会論文誌D-I
Vol. J88-D-I, No. 4, pp. 906-907（2005）

2. 2. コンパイル・エベントドリブン混在型故障シミュレーション手法

故障シミュレーションの主な方式として、コンパイル方式とイベントドリブン方式があるが、それぞれ長所と短所がある。対象論理回路をFFR（Fanout-Free Region）に分割し、FFR内ではコンパイル方式、また、FFR間ではイベントドリブン方式を活用することによって故障シミュレーションを高速化する手法を本研究で開発し、その成果を次の国際会議で発表した。ここではその内容を紹介する。

K. Taniguchi, H. Fujii, S. Kajihara X. Wen:
Hybrid Fault Simulation with Compiled and Event-Driven Methods
Proc. Int'l Conf. on Design & Test of Integrated Systems in Nanoscale Technology
pp. 240-243 (2006)

Hybrid fault simulation with compiled and event-driven methods

Kenjiro Taniguchi, Hideo Fujii, Seiji Kajihara, and Xiaoqing Wen
Department of Computer Science and Electronics
Kyushu Institute of Technology
680-4 Kawazu, Iizuka 820-8502 Japan
{taniguchi, fujii, kajihara, wen}@aries30.cse.kyutech.ac.jp

1. Introduction

Fault simulation for a logic circuit, that calculates the behavior of a faulty circuit for given test patterns, plays an important role in VLSI design processes [1,2]. The objectives of fault simulation is fault grading, test pattern generation, or fault diagnosis. Run time of fault simulation increases with the circuit size and the number of test patterns. While the reduction of test costs is a critical issue for logic circuit testing, it is required to develop a hi-speed fault simulator.

There are some methods to accelerate fault simulation. Parallel fault simulation is a simple and well-known method that assigns one test pattern or one faulty circuit to each bit of a word. If one word of a machine is 32 bits, 32 patterns or 32 faulty circuits are simulated in parallel. Concurrent fault simulation [3] and deductive fault simulation [4] are the other acceleration methods. These methods calculate faults detectable at each line of the circuit and propagate the faults to outputs of the circuits.

Since the procedure of fault simulation consists of fault injection and logic simulation, keys for acceleration of fault simulation are shown as follows:

(1) To employ a fast logic simulation method.

(2) To avoid waste simulation such that faulty behavior is the same as the fault-free behavior.

As a fast logic simulation method, compiled simulation is well-known, which predetermines the order of lines to be evaluated and implements it in the assembly program. In fault simulation, however, it is difficult to use compiled simulation efficiently, because we don't have to compute the circuit behavior if a faulty circuit behaves the fault-free circuit. Event-driven simulation is more adequate for fault simulation because only difference between the fault-free circuit and faulty circuits can be simulated.

In this paper, we propose a method to speed-up fault simulation. The proposed method takes a hybrid approach with compiled simulation and event-driven simulation. Compiled simulation is applied for fan-out free regions (FFRs). FFRs to be simulated are selected with the event-driven manner. Since the event-driven simulation contributes to avoidance of waste simulation and the compiled simulation contributes to reduction of memory access, the proposed method can reduce the simulation time effectively. Note that this work targets on combinational circuits or a full-scan sequential circuit, and the single stuck-at fault model is assumed. Experimental results for benchmark circuits show that the proposed method could reduce runtime in half compared with concurrent (event-driven) fault simulation.

This paper is organized as follows. In Section 2, we define an FFR, and show the overview of the proposed simulation method. In Section 3, we give the proposed fault simulation method. In Section 4 we show experimental results and in Section 5 we conclude this paper.

2. Preliminary

2.1 FFR (Fanout Free Region)

At first, we define an FFR (Fanout Free Region). FFR is a subcircuit in which every gate has only one output. An input of an FFR is a primary input or a fanout branch, and an output of an FFR is a primary output or a fanout stem. An example is given in Fig. 1. A circuit in Fig. 1(a) consists of two FFRs as shown in Fig. 1(b).

Suppose that two faults *fa* and *fb* exist in a same FFR. If the effects of *fa* and *fb* are propagated to the output of the FFR, we can treat two faults as one fault on the output of the FFR. Therefore, simulation of each fault can be divided into two parts: one is from the fault site to the output of its FFR, and another is from the FFR output to primary outputs.

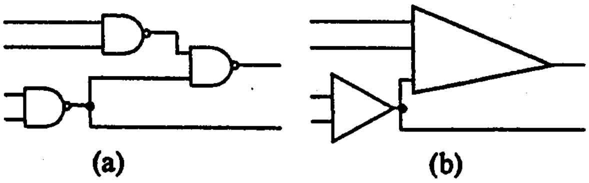


Fig. 1: Example of FFRs

2.2 Overview of the proposed simulation method

In this work we employ event-driven simulation, compiled simulation and parallel simulation. In fault simulation, critical path tracing is also used to check the possibility of fault propagation from a fault site to its FFR output. Faults are explicitly injected only at outputs of

0-7803-9726-6/06/$20.00©2006 IEEE

FFRs which are a fanout stem or a primary output. When a fault is injected at a fanout stem, the fault-free value of the fanout stem is inverted. It means that an event appears at fanout branches from the fanout stem. Once an event appears at an input of an FFR, a logic value of the output of the FFR is calculated with compiled fault simulation. If the logic value of the FFR output is different from the fault-free value, it is treated as a new event. But if the logic value of the FFR output is the same as the fault-free value, no event is created. This process is repeated as long as any event exists at a fanout.

3. Details of the proposed method

3.1 Compiled Simulation

Primary inputs or fanout brances are input lines of FFRs, and primary outputs or fanout-stems are output lines of FFRs. The other lines are internal lines of FFRs. To achieve the speed-up of fault simulation, compiled simulation are applied inside FFRs. To realize this, we make logical formula for each FFR as one function. For example, we consider an FFR of Fig. 2. The input lines of the FFR are *a*, *b* and *c*. The output line of the FFR is *e*. Using un-compiled simulation, the value of line *d* is derived after accessing a fanin list of the AND gate and then the value of line *d* is calculated from the values of line *a* and line *b*. And the value of line *e* is derived from the values of line *d* and line *c* after accessing a fanin list of the OR gate.

On the other hand, in the compiled method a function (or a subprogram), where expression e=($a \wedge b$)$\vee c$ is described, is prepared for the value of line *e*. It means that we don't have to access the fanin lists of the AND gate and the OR gate to calculate a value of line *e* during simulation. Hence the simulation speed goes up.

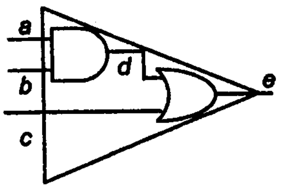


Fig. 2: Example of an FFR

A disadvantage of compiled simulation for fault simulation was that fault injection is difficult. In the above example in Fig. 2, fault injection to line *d* is impossible because a value of line *d* is calculated in the predetermined function. Since the proposed method injects faults at outputs of FFRs, such a problem can be avoided.

3.2 Function for calculating an FFR value

We explain a method of creating a function from a FFR. The basic rules are described in the following.

- Scan all lines of the FFR from the output to inputs with a depth-first manner.
- When we meet an input of a gate first, we output "(".
- When we go back to another input of the gate, we output the symbol of the gate(e.g. ∧, ∨).
- When we go back to the output of the gate, we output ")".

We give an example for the FFR in Fig.2 as follows.

1. Start from line *e* and output the output line"*e*=".
2. Go to input line *d* of OR gate and output "(".
3. Go to input line *a* of AND gate and output "(".
4. Output "*a*".
5. Go to another input line *b* of the AND gate and output the symbol "∧".
6. Output "*b*".
7. Go back to output line *d* of the AND gate, and output ")".
8. Go to another input line *c* of the OR gate, and output the symbol "∨".
9. Output "*c*".
10. Go back to output line *e* of the OR gate, and output ")".

As a result, we can derive the function "e=(($a \wedge b$)$\vee c$)".

3.3 Fault simulation

Creating a function of each FFR is done at a preprocessing phase. A fault list is created at the preprocessing phase too. The main phase of fault simulation is given in the following.

1. Logic simulation for a given test pattern is performed to calculate fault-free values of each line.
2. If there is a fault in a FFR which can be sensitized and propagate to the output of the FFR, set an event to the output of the FFR. If not, stop this procedure. Note that when more than one FFR has an event, choose one that is the closest to primary inputs.
3. If an event exists at a primary output, mark the fault as "detected", and remove the event. If an event exists at a fanout stem, then call a function of the FFRs which include a fanout branch of the fanout stem and calculate the output value of the FFRs.
4. If the output value is different from the fault-free value, set a new event to the output of the FFR.
5. Return to 2.

We illustrate the overall procedure of fault simulation in Fig. 3.

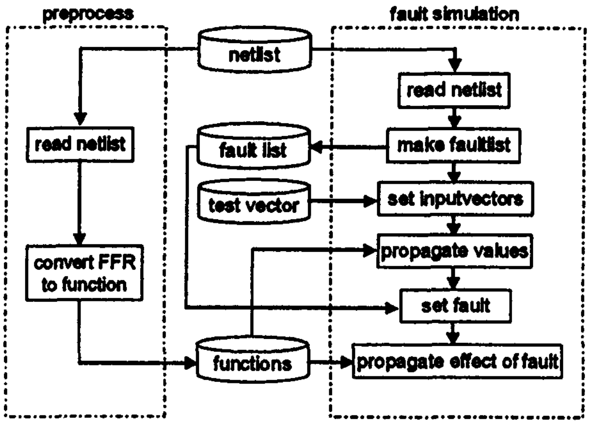


Fig. 3: Procedure of fault simulation

4. Experimental results

We implemented the proposed method using C programming language on a PC (OS: FreeBSD 4.11 Release, CPU: Pentium4 530J (3.0GHz), memory: 512MB), and applied to combinational parts of ITC'99 benchmark circuits. We first made an experiment of logic simulation by the proposed method and a traditional method that calculate logic values by accessing the net list for each gate. About input vectors, we used 10,000 random patterns. We show experimental results in Table1. The first three columns in Table1 show the circuit name, the number of lines and the number of gates. The following two columns show CPU time in seconds, about the proposed method and the traditional method. The last two columns show the reduction time and the percentage of reduction by the proposed method. By introducing compiled simulation, the run time of logic simulation was reduced to a few percents of the traditional method.

We also give experimental results of fault simulation by the proposed method and the traditional method. The method of fault simulation is based on PPSFP (parallel pattern single fault propagation). About input vectors, we used 1,000 random patterns. And during fault simulation, no fault dropping was done, we show results in Table2. The first three columns in Table2 show the circuit name, the number of lines and the number of gates. The following two columns show CPU time in seconds, about the proposed method and the traditional method. The last two columns show the reduction time and the percentage of reduction by the proposed method.

5. Conclusion

We proposed a speed-up method of fault simulation based on a compiled approach and an event-driven approach. Though it had been difficult for fault simulation to use the compiled approach, the proposed method allowed it by partitioning the functions of compiled codes into FFRs. Experimental results for ITC'99 benchmark circuits showed that fault simulation time was reduced in half compared with a traditional PPSFP method. Now this fault simulation can be applied only for combinational circuits. So we are extending this fault simulation to one for sequential circuits.

References

[1] M. Abramovici, M. A. Breuer, A. D. Friedman, *Digital Systems Testing and Testable Design*, Piscataway, New Jersey: IEEE Press, 1990.

[2] M. L. Bushnell, and V. D. Agrawal, *Essentials of Electronic Testing for Digital, Memory & Mixed-Signal VLSI Circuits*, Kluwer Academic Publishers, 2000.

[3] E. G. Ulrich, T. Baker, "The concurrent simulation of nearly identical digital networks," *Proc. of Design Automation Workshop*, vol. 6, pp. 145-150, 1973.

[4] D.B. Armstrong, "A deductive method for simulating faults in logic circuits," *IEEE Trans. on Computer*, vol. C-21, No. 5, pp. 464-471, 1972.

Table1: Results of logic simulation

circuit information			simulation time		evaluation	
circuits	# of lines	# of gates	traditional	proposed	reduction time	%
b04s	1373	512	0.27	0.06	0.21	77.78
b07s	1015	362	0.21	0.04	0.17	80.95
b11s	1190	437	0.23	0.03	0.20	86.96
b12	2517	904	0.52	0.08	0.44	84.62
b14s	11645	4444	10.84	0.26	10.58	97.60
b15s	21804	8338	22.54	0.45	22.09	98.00
b17s	60253	22645	69.41	1.31	68.10	98.11
b20s	23045	8875	24.14	0.52	23.62	97.85
b21s	24053	9259	25.48	0.53	24.95	97.92
b22s	36707	14282	40.95	0.78	40.17	98.10

Table2: Results of parallel pattern simulation

circuit information			simulation time		evaluation	
circuits	# of lines	# of gates	traditional	proposed	reduction time	%
b04s	1373	512	1.79	1.12	0.67	37.43
b07s	1015	362	1.38	0.96	0.42	30.43
b11s	1190	437	2.60	1.57	1.03	39.62
b12	2517	904	2.60	1.73	0.87	33.46
b14s	11645	4444	132.30	58.59	73.71	55.71
b15s	21804	8338	253.86	103.19	150.67	59.35
b17s	60253	22645	761.21	338.96	422.25	55.47
b20s	23045	8875	286.36	137.02	149.34	52.15
b21s	24053	9259	318.04	156.67	161.37	50.74
b22s	36707	14282	489.14	231.27	257.87	52.72

3. 実速度テスト技術

最近のSoC（System-on-Chip）は同期回路であっても複数のクロックドメインを有することが多く、高テスト品質の実現するためには、すべてのクロックドメインの中の故障、および、クロックドメイン間の故障に対して、実速度テストを行なう必要がある。このため、コンパクトで実現容易なテスト制御方式と回路設計に関する研究開発が求められている。

複数のクロックドメインを有する回路の場合、まずクロックドメイン間の相互依存関係を調べるためのクロック解析を行ない、すべてのクロックドメインを独立グループに分ける必要がある。各クロックドメイングループに対しては同時にキャプチャを行ない、また、異なるクロックドメイングループに対してはキャプチャタイミングをずらすことを行なうことによって、タイミング問題を回避することができる。そこで本研究では、ダブルキャプチャ方式に基づいた実速度テスト制御手法を開発し、その有効性を実験で示した。その研究成果を３．１で紹介する。

複数のクロックドメインを有する回路の場合、クロックドメインの間に故障が存在することが多い。これらの故障はタイミングに大きな影響を与えるため、それらを検出できなければ、テスト品質が著しく低下することが避けられない。そこで本研究では、クロックドメイン間に存在する故障を実速度でテストするためのテスト制御手法を開発し、その有効性を実験で示した。その研究成果を３．２で紹介する。

３．１．多重クロックを有する回路に適用可能なロジックBIST方式

最近のSoC（System-on-Chip）は同期回路であっても複数のクロックドメインを有することが多く、高テスト品質の実現するためにすべてのクロックドメインにおいて実速度テストを行なう必要がある。ダブルキャプチャ方式に基づいた実速度テスト制御手法を本研究で開発し、その成果を次の国際会議で発表した。ここではその内容を紹介する。

L.-T. Wang, X. Wen, B. Hsu, S. Wu, J. Guo:
A Flexible Logic BIST Scheme for Multiple-Clock Circuits
Proc. IEEE Int'l Conf. on Computer Design, pp. 475-478 (2005)

3. 2. クロックドメイン間における故障の実速度テスト方式

最近のSoC（System-on-Chip）は同期回路であっても複数のクロックドメインを有することが多い。クロックドメイン間に存在する故障を実速度でテストするためのテスト制御手法を本研究で開発し、その成果を次の国際会議で発表した。ここではその内容を紹介する。

H. Furukawa, X. Wen, L.-T. Wang, B. Sheu, Z. Jiang, S. Wu:
A Novel and Practical Control Scheme for Inter-Clock At-Speed Testing
Proc. IEEE Int'l Test Conf., Paper 17.2 (2006)

A Novel and Practical Control Scheme for Inter-Clock At-Speed Testing

Hiroshi Furukawa, Xiaoqing Wen
Dept. of Computer Science and Electronics, Kyushu Institute of Technology, Iizuka 820-8502, Japan
Laung-Terng Wang, Boryau Sheu, Zhigang Jiang, Shianling Wu
SynTest Technologies, Inc., 505 S. Pastoria Ave., Suite 101, Sunnyvale, CA 94086, USA

Abstract

The quality of at-speed testing is being severely challenged by the problem that an inter-clock logic block existing between two synchronous clocks is not efficiently tested or totally ignored due to complex test control. This paper addresses the problem with a novel inter-clock at-speed test control scheme, featuring a compact and robust on-chip inter-clock enable generator design. The new scheme can generate inter-clock at-speed test clocks from PLLs, and is feasible for both ATE-based scan testing and logic BIST. Successful applications to industrial circuits have proven its effectiveness in improving the quality of at-speed testing.

1. Introduction

As process features shrink into the deep submicron (DSM) range and circuit speeds accelerate into the GHz domain, timing-related physical defects (often modeled as transition or path delay faults) are rapidly becoming the dominant cause of failing integrated circuits [1]. Nowadays it is mandatory to conduct *at-speed testing* in order to screen out chips that cannot operate at rated frequencies [2].

Different from conventional slow-speed structural testing, at-speed testing presents many new challenges to the test community, in terms of *test data volume*, *test application cost*, *test quality*, and *test control*. These issues, if not properly addressed, will make it impossible to achieve effective and efficient at-speed testing, as described bellow:

Test Data Volume: This issue is mainly because two test vectors are needed to detect one delay fault in at-speed testing. In addition, more stringent constraints need to be followed in at-speed test vector generation. These factors make it difficult to detect many delay faults with a single pair of test vectors, resulting in larger test data volume. Recently, various test compaction and compression techniques have been proposed to address this issue [3-6].

Test Application Cost: This issue can be very significant in at-speed testing if not properly handled. The major reason is the need for using functional clock frequencies to capture test responses during at-speed testing. As more and more circuits, including even low-price consumer chips, are designed to run at high frequencies, the conventional approach of using high-speed automatic test equipment (ATE) for at-speed testing is not sustainable from the test cost point of view. A more practical test application approach is to use on-chip clock sources for at-speed testing, in one of two forms. In the *partial* form, high-speed test clocks, e.g. capture clocks in scan testing, are provided from modified on-chip phase-locked loops (PLLs), while low-speed test clocks, e.g. shift clocks in scan testing, are provided from an external ATE [7]. In the *complete* form, all test clocks are provided internally from an on-chip test controller, as in logic built-in self-test (BIST) [8].

Test Quality: This is a critical issue that needs special attention. The fact is that the quality of at-speed testing cannot be measured by merely relying on one simple parameter, usually fault coverage, as often used in conventional slow-speed testing [9]. This is mainly due to three reasons: (1) The transition delay fault model has weak correlation with distributed small-delay defects. As a result, test vectors generated by conventional transition delay test generation methods often fail to detect small-delay defects. (2) The path delay fault model can better reflect the effect of defective delay increase in a circuit. However, this fault model suffers from severe challenges in terms of path selection quality and test generation efficiency. (3) Accurate evaluation of a delay test set is difficult, especially in the presence of noise and process variation. Recently, improving the quality of at-speed testing has been the focus of many studies [10-14].

Test Control: Test control, whose purpose is to issue all necessary test signals, such as clocks and scan enables, is another critical issue in at-speed testing. At-speed test control significantly differs from conventional slow-speed test control for the following reasons: (1) At-speed test control requires more complex waveforms than slow-speed testing. This is because, in order to conduct at-speed testing, a value transition for activating a delay fault should be launched at the start-point of a path and its response should be captured at the end-point at the rated clock speed, which can be very fast. (2) Timing relations among inter-dependent test control signals should be strictly followed, especially for a multi-clock circuit. As a result, many previous at-speed test control schemes, though logically sound, are not feasible for physical implementation because of high complexity, use of many timing-critical signals, or costly circuit modification for handling clock skews [15, 16]. Although there are some easy-to-implement test control schemes, they can only provide test control for *intra-clock logic blocks*, while totally ignoring *inter-clock logic blocks* [17-19]. Obviously, this results in incomplete at-speed testing, which significantly lowers test quality.

1-4244-0292-1/06/$20.00 © 2006 IEEE

Generally, the combinational portion of a clock domain in a multi-clock circuit consists of two types of logic blocks: An *intra-clock logic block* is surrounded by flip-flops (FFs) driven by the same clock, while an *inter-clock logic* block exists between FFs driven by two different but synchronous clocks. Growing circuit sizes, shrinking process features, and accelerating circuit speeds are all making inter-clock logic blocks more and more significant. Obviously, ignoring inter-clock logic blocks in at-speed testing presents a tremendous risk for chip quality, which is not acceptable. Therefore, there is a strong and urgent need to establish an easy-to-implement test control scheme for inter-clock at-speed testing. This is the focus of this paper.

Previous test control schemes for inter-clock at-speed testing are too complex or too difficult for physical implementation [15, 16], mainly because they mix inter-clock test control with intra-clock test control and use the launch-on-shift approach that requires timing-critical scan enable signals. This observation leads to the key idea of this paper: *separating inter-clock test control from intra-clock test control and using the launch-on-capture approach that does not require any timing-critical scan enable signal.*

Based on the above key idea, this paper proposes a novel and practical inter-clock at-speed test control scheme, with the following characteristics:

- **Novel Clock Enable Generator**: A compact and robust on-chip inter-clock enable generator design is proposed, which creates two clock enable signals from two free-running synchronous functional clocks. The clock enable signals, when gated with the synchronous functional clocks, accurately generate all capture clock pulses required for the at-speed testing of the inter-clock logic block existing between the two functional clocks.
- **Easy Physical Implementation**: Inter-clock at-speed testing is conducted with the launch-on-capture approach. As a result, a single and non-timing-critical scan enable signal is sufficient for the scan testing of the entire circuit, significantly simplifying physical implementation.
- **Low Application Cost**: At-speed capture clock pulses are generated by on-chip circuitry through clock-gating with functional clocks from on-chip PLLs. There is no need to provide any high-speed test clock from ATE.
- **Easy Integration**: The new inter-clock at-speed test control scheme can be easily integrated with any existing easy-to-implement intra-clock at-speed test control scheme [17-19] at the top-level to achieve complete at-speed testing. This greatly improves test quality.
- **High Flexibility**: The new inter-clock at-speed test control scheme can be readily applied in both ATE-based scan testing and logic BIST.

The new inter-clock at-speed test control scheme has been successfully applied to large industrial circuits. As proven in layout and on tester, this scheme is easy to implement and can improve the at-speed test quality of multi-clock circuits required by today's system-on-a-chip (SoC) designs.

The rest of the paper is organized as follows: Section 2 describes the background. Section 3 presents the new inter-clock at-speed test control scheme. This section also discusses the integration of inter-clock and intra-clock at-speed test control schemes at the top-level for complete at-speed testing. Section 4 shows application results on industrial circuits, and Section 5 concludes the paper.

2. Background

2.1 Intra-Clock Logic and Inter-Clock Logic

Modern SoC circuits commonly use multiple clocks, often provided internally from on-chip PPLs. This can be due to the need for interfacing with different external systems, such as PCI and USB. It may also be due to the chip size being too large for a single fast clock to be effectively distributed over the entire circuit. Such a circuit usually consists of multiple clock domains, as defined bellow [20]:

Definition 1: A *clock domain* is the part of a circuit driven by either a single clock or multiple clocks that have constant phase relationships.

For example, a clock and its inverted clock or its derived divide-by-two clocks are considered as one clock domain. On the other hand, domains that have clocks with variable phase and time relationships are considered as different clock domains.

The clock skew within a clock domain is strictly managed, typically through clock tree synthesis (CTS). As a result, FFs in the same clock domain receive clock edges almost at the same time. On the other hand, a clock domain often has multiple synchronous internal clocks. That is, different FFs in a clock domain may operate at different but synchronous frequencies. An example is shown in Fig. 1, where the clock domian contains two synchronous clocks: a fast clock *FCK* and a slow clock *SCK*, running at 660MHz and 330MHz, respecitvely.

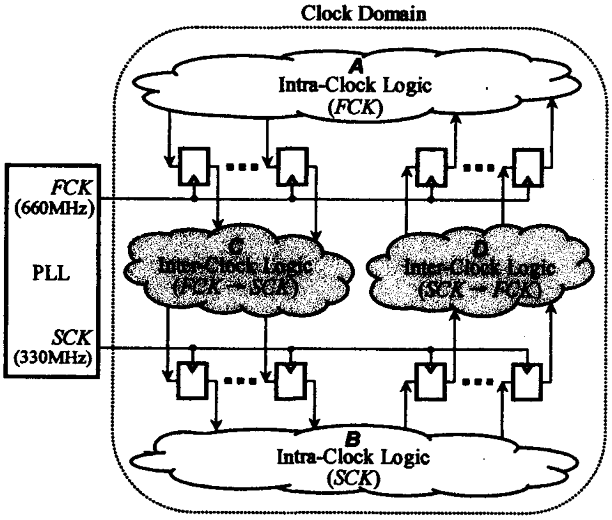


Fig. 1 Intra-Clock Logic and Inter-Clock Logic.

Therefore, a clock domain can be seen to consist of (1) FFs driven by a number of synchronous internal clocks and (2) combinational logic blocks surrounding those FFs. The combinational logic blocks can be further divided into two types as defined bellow:

Definition 2: *Intra-clock logic block* in a clock domain is the combinational logic portion existing between FFs driven by the same internal clock.

Definition 3: *Inter-clock logic block* in a clock domain is the combinational logic portion existing between FFs driven by two synchronous internal clocks.

For example, the clock domain shown in Fig. 1 consists of four combinational logic blocks: *A*, *B*, *C*, and *D*. Obviously, *A* and *B* are intra-clock logic blocks for internal clocks *FCK* and *SCK*, respectively. On the other hand, *C* and *D* are inter-clock logic blocks, with *C* being from *FCK* to *SCK* and *D* being from *SCK* to *FCK*.

Note that data transfer between two asynchronous clock domains needs to be made through synchronizers or dual-port random access memories (RAMs). Due to glitch concerns, functional logic is usually not placed between two clock domains with FF-based synchronizers.

From the discussions above, it is clear that the combinational portion in a circuit consists of intra-clock logic blocks and inter-clock logic blocks in all clock domains. Therefore, all intra-clock logic blocks as well as all inter-clock logic blocks need to be tested in order to achieve complete at-speed testing for the circuit.

2.2 Test Control for At-Speed Testing

2.2.1 General Concept

Logic testing, both slow-speed and at-speed, are usually based on full-scan design, in which all functional FFs in a circuit are replaced with scan FFs. A full-scan circuit operates in either *shift* or *capture* mode, selectable by *scan enable* (*SE*) signals. In shift mode (*SE* = 1), scan FFs form scan chains that operate as shift registers, through which a new test vector is shifted-in and a test response is shifted-out. In capture mode (*SE* = 0), scan FFs operate as functional FFs and catch the test response corresponding to the applied test vector. Such scan-based testing is the foundation of both ATE-based testing and logic BIST.

The basic concept of test control for scan-based testing is illustrated in Fig. 2. Basically, scan enable (*SE*) signals and clock (*CK*) pulses should be provided in proper order and with right timing relations in both shift and capture modes, so as to guarantee correct shift and capture operations.

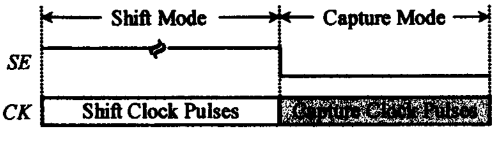


Fig. 2 Test Control for Scan-Based Testing.

Generally, test control for slow-speed testing targeted at structural defects can be easily conducted for both intra-clock and inter-clock logic blocks. However, test control for at-speed testing targeted at timing-related defects is significantly more difficult, especially for multi-clock SoC circuits. As a result, test control has become a major challenge in pursuing high-quality at-speed testing.

2.2.2 Control Tasks for At-Speed Testing

Test control for scan-based testing is realized by controlling scan enable and clock signals to conduct a series of control tasks, which are repeated for each test vector. For at-speed testing, there are three types of control tasks, ***shift***, ***launch***, and ***capture***, as illustrated in Fig. 3, whose circuit is the full-scan version of the circuit shown in Fig. 1.

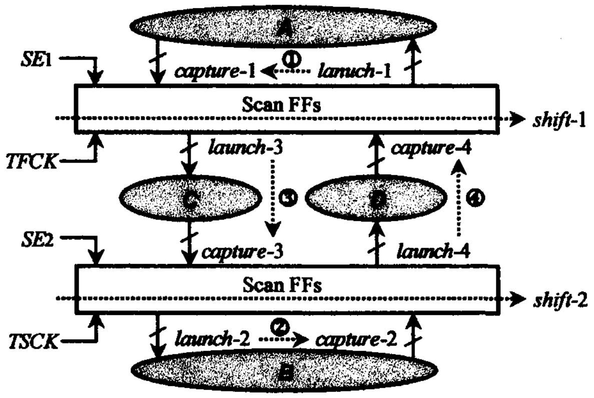


Fig. 3 Control Tasks for At-Speed Testing.

In Fig. 3, there are two shift tasks, *shift*-1 and *shift*-2, each for one scan chain. A shift task is to operate a scan chain as a shift register to load a test vector or unload a test response, by setting the corresponding *SE* signal to 1 and issuing the same number of shift clock pulses as the length of the corresponding scan chain. Since shift tasks are independent from each other, they can be conducted simultaneously without considering their inter-relations. In addition, a shift task can be conducted at any speed, usually a slow speed for test power reduction, even for at-speed testing. As a result, test control for shift tasks is easy, and will not be further considered in this paper.

After shift tasks are finished, a pair of launch and capture tasks should be conducted in order to apply at-speed testing to each logic block. There are two types of launch-capture pairs as defined bellow:

Definition 4: An *intra-clock launch-capture pair* is used for the at-speed testing of an intra-clock logic block, while an *inter-clock launch-capture pair* is used for the at-speed testing of an inter-clock logic block.

In Fig. 3, there are two intra-clock launch-capture pairs, ① and ②, as well as two inter-clock launch-capture pairs, ③ and ④. For example, ① (<*launch*-1, *capture*-1>) is the intra-clock launch-capture pair for *A*, where *launch*-1 is to create value transitions at the inputs of *A* and *capture*-1 is to capture the test response from the outputs of *A* at the rated speed of the clock *FCK* in order to achieve intra-clock at-speed testing. On the other hand, ③ (<*launch*-3, *capture*-3>) is the inter-clock launch-capture pair for *C*, where *launch*-3 is to create value transitions at the inputs of *C* and *capture*-3 is to capture the test response from the outputs of *C* following the exact inter-clock relation from *FCK* to *SCK* in order to achieve inter-clock at-speed testing.

Note that the free-running functional clocks *FCK* and *SCK* shown in Fig. 1 are replaced with test clocks *TFCK* and *TSCK*, respectively, as shown in Fig. 3. *TFCK* and *TSCK*, as well as scan enable signals *SE*1 and *SE*2, should be properly provided from a test control scheme in order to accomplish all test control tasks. In this sense, *FCK* and *SCK* can be seen as inputs to a test control scheme, while *TFCK* and *TSCK* are outputs from the test control scheme.

2.3 Previous At-Speed Test Control Schemes

There are two approaches to test control for scan-based at-speed testing: *launch-on-shift* and *launch-on-capture* [21, 22]. The major difference is how a launch task is conducted, i.e. how value transitions are created at the inputs of a logic block for delay testing. Launch-on-shift creates value transitions by the difference between the next-to-last and the last shifted-in values in shift mode, while launch-on-capture creates value transitions by the difference between the last shifted-in values in shift mode and the first-captured values in capture mode.

In the following, we briefly describe three test control schemes, focusing on how launch-capture pairs are conducted. The *launch aligned skewed-load* scheme and the *capture aligned skewed-load* scheme are based on the launch-on-shift approach, while the *double-capture* scheme is based on the launch-on-capture approach.

Launch Aligned Skewed-Load Scheme: This scheme is illustrated in Fig. 4, corresponding to the circuit shown in Fig. 3. This scheme aligns the last shift clock pulse *S*1 of *TFCK* with the last shift clock pulse *S*2 of *TSCK* to create input value transitions simultaneously for both *TFCK* and *TSCK*, and then issues at-speed capture clock pulses *C*1 and *C*2 to capture corresponding test responses at the rated speeds of *FCK* and *SCK*, respectively [15].

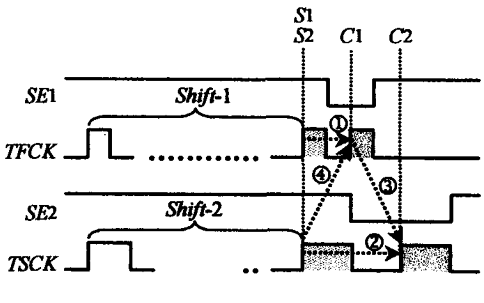


Fig. 4 Launch Aligned Skewed-Load Scheme.

Although this scheme can accomplish all launch-capture pairs ① ~ ④ to achieve complete at-speed testing, it requires two *SE* signals, *SE*1 and *SE*2, which may be timing-critical since both of them need to settle from 1 to 0 during the system clock periods of *FCK* and *SCK*, respectively. This makes already-hard physical implementation even more difficult. In addition, the complexity of this scheme increases rapidly with the number of clocks.

Capture Aligned Skewed-Load Scheme: This scheme is illustrated in Fig. 5, corresponding to the circuit shown in Fig. 3. This scheme aligns the at-speed capture clock pulses *C*1 and *C*2 following the last shift clock pulses *S*1 and *S*2 so as to capture corresponding test responses simultaneously for the clocks *TFCK* and *TSCK*, respectively [16].

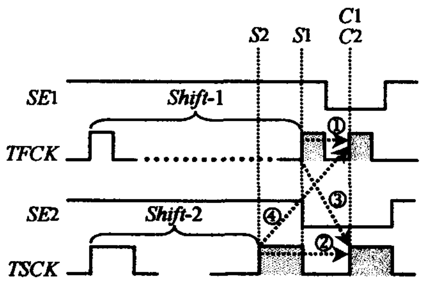


Fig. 5 Capture Aligned Skewed-Load Scheme.

This scheme can also accomplish all launch-capture pairs ① ~ ④ to achieve complete at-speed testing, but also suffers from increased difficulty in physical implementation as in the launch aligned skewed-load scheme. In addition, frequency pre-scaling may have to be conducted in some cases, making it hard to achieve real at-speed testing. Moreover, costly circuit modification may be needed in order to prevent clock skews from disturbing test responses.

Double-Capture Scheme: This scheme is illustrated by the example of Fig. 6, corresponding to the circuit shown in Fig. 3. This scheme uses two at-speed capture clock pulses for each clock: *C*1/*C*2 for *TFCK* and *C*3/*C*4 for *TSCK* [19]. The input value transitions for *TFCK* (*TSCK*) are launched by the difference between the values loaded by the last-shift pulse *S*1 (*S*2) and the values captured by the first-capture pulse *C*1 (*C*3). Note that capture operations are conducted for both clocks in the same capture window (*SE* = 0).

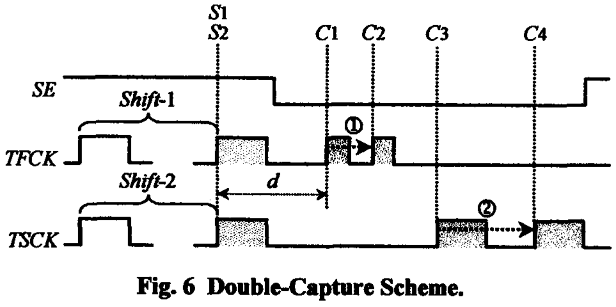


Fig. 6 Double-Capture Scheme.

The biggest advantage of this scheme is its easy physical implementation. This is because only one scan enable signal *SE* is used, which is not timing-critical if *d* is made long enough. Some other schemes can achieve similar effects [17, 18]. However, the biggest disadvantage of these schemes is that they can only achieve intra-clock at-speed testing. As shown in Fig. 6, this is because only intra-clock launch-capture pairs, ① and ②, are accomplished, while inter-clock launch-capture pairs, ③ and ④, are totally ignored. That is, these schemes cannot provide test control for inter-clock at-speed testing. As a result, complete at-speed testing cannot be achieved for a multi-clock circuit.

Therefore, there is a strong need for an easy-to-implement inter-clock at-speed test control scheme. Combining it with any easy-to-implement intra-clock at-speed test control scheme, such as the double-capture scheme [17-19], forms an integrated test control scheme to achieve complete at-speed testing for all logic blocks in a circuit. As a result, the at-speed test quality can be significantly improved.

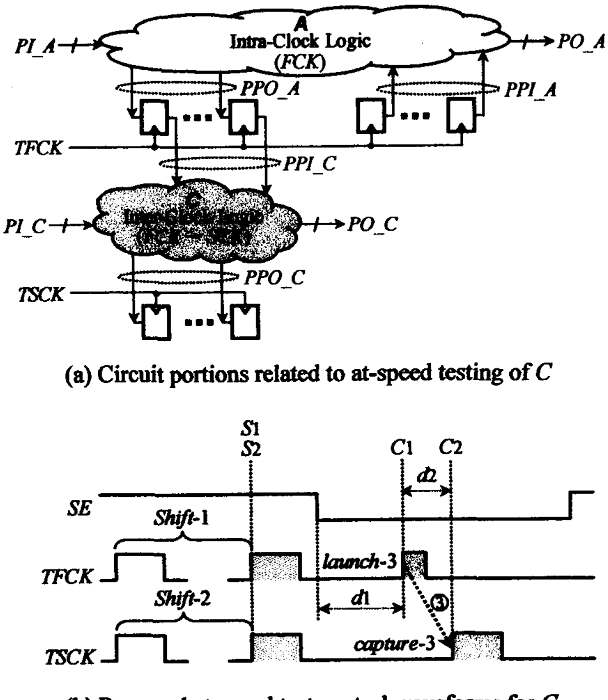
3. Inter-Clock At-Speed Test Control Scheme

3.1 Basic Idea

As described in 2.3, previous test control schemes for complete at-speed testing suffer from high complexity and difficult physical implementation. This is mainly because they mix inter-clock test control with intra-clock test control and use timing-critical scan enable signals due to the launch-on-shift approach. This observation leads to the basic idea of this paper as follows:

- Separate inter-clock test control from intra-clock test control in order to simplify test control operations.
- Use the launch-on-capture approach in order to avoid the use of any timing-critical scan enable signal.

The basic idea is illustrated in Fig. 7, using the example of at-speed testing for the inter-clock logic block *C* in Fig. 1.



(a) Circuit portions related to at-speed testing of *C*

(b) Proposed at-speed test control waveforms for *C*

Fig. 7 Basic Idea of Inter-Clock At-Speed Test Control.

The two shift tasks in Fig. 7 (b), *shift*-1 and *shift*-2, are conducted with the same shift clock in this example. However, it is allowed to use any shift control method with any number of shift clocks running at any frequencies. After shift tasks are finished, the launch-capture pair ③ (<*launch*-3, *capture*-3>) in Fig. 3 should be performed in order to conduct at-speed testing for the inter-clock logic block *C* in Fig. 7 (a). Based on the basic idea described above, this paper proposes a new inter-clock at-speed test control scheme that uses the test control waveforms of Fig. 7 (b) to accomplish the launch-capture pair ③ as follows:

Launch Control: *launch*-3 is conducted by the capture clock pulse *C*1 of *TFCK*, as shown in Fig. 7 (b). That is, value transitions at the inputs *PPI_C* are created by the difference between the values loaded by the last-shift pulse *S*1 of *TFCK* and the values captured by the capture pulse *C*1 of *TFCK*. Since the launch-on-capture approach is used, *d*1 can be made long enough to allow the use of a single and non-timing-critical scan enable signal *SE* for all scan chains. This significantly simplifies physical implementation.

Capture Control: *capture*-3 is conducted by the capture clock pulse *C*2 of *TSCK*, as shown in Fig. 7 (b). For at-speed testing of the inter-clock logic block *C*, it is required that *d*2 be set by following the exact clock-triggering relation from *FCK* to *SCK* for transferring data as in functional operations. As an example, Fig. 7 (b) shows the positive-to-positive single-cycle at-speed test requirement.

In the following, details on the new inter-clock at-speed test control scheme are described by using the example shown in Fig. 7. The focus is on how capture pulses *C*1 and *C*2 for test clocks *TFCK* and *TSCK* are generated from free-running functional clocks *FCK* and *SCK* (PLL clocks as shown in Fig. 1), respectively, by using new on-chip circuitry so that the requirements on *d*1 and *d*2 are satisfied.

3.2 General Architecture

Fig. 8 shows the general architecture of the new inter-clock at-speed test control scheme for the example of Fig. 7.

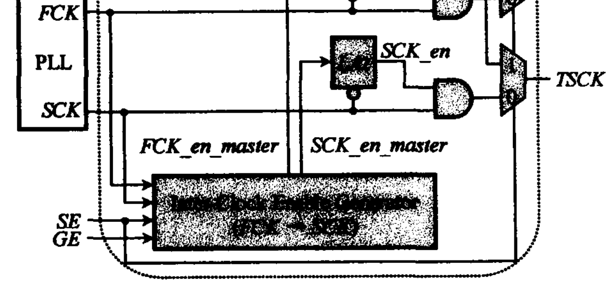


Fig. 8 Architecture of Inter-Clock Test Control Scheme.

In shift mode (*SE* = 1), *shift_CK*, either generated internally or provided externally, is used as the shift clock for both *TFCK* and *TSCK*. Note that *shift_CK* can be a slow clock for reducing power dissipation during test application.

In capture mode (*SE* = 0), the inter-clock enable generator creates *master* clock enable signals *FCK_en_master* and *SCK_en_master*, the two latches generate *final* clock enable signals *FCK_en* and *SCK_en*, and the two AND gates conduct clock-gating to generate test clocks *TFCK* and *TSCK* from functional PLL clocks *FCK* and *SCK*, respectively. *SE* is a normal scan enable signal as shown in Fig. 7 (b), while *GE* is a generator enable signal used to indicate if the inter-clock enable generator is active or not. *GE* = 1 means that *FCK_en_master* and *SCK_en_master* are active clock enable signals, while *GE* = 0 will inactivate them by holding them at 0.

Note that circuitry for switching between test and normal modes is not shown in Fig. 8 for clarity. As far as test concerned, only one multiplexor and one AND gate are added to a functional clock line, while the rest of the circuitry in the new inter-clock at-speed test control scheme, especially the inter-clock enable generator, does not cause any performance degradation. In addition, this new scheme can be used in both ATE-based testing and logic BIST.

Obviously, the key part of the new inter-clock at-speed test control scheme is the inter-clock enable generator for creating master clock enable signals. Detailed information on this part is described in the following section.

3.3 Inter-Clock Enable Generator Design

Fig. 9 shows the schematic of the proposed inter-clock enable generator design used in Fig. 8.

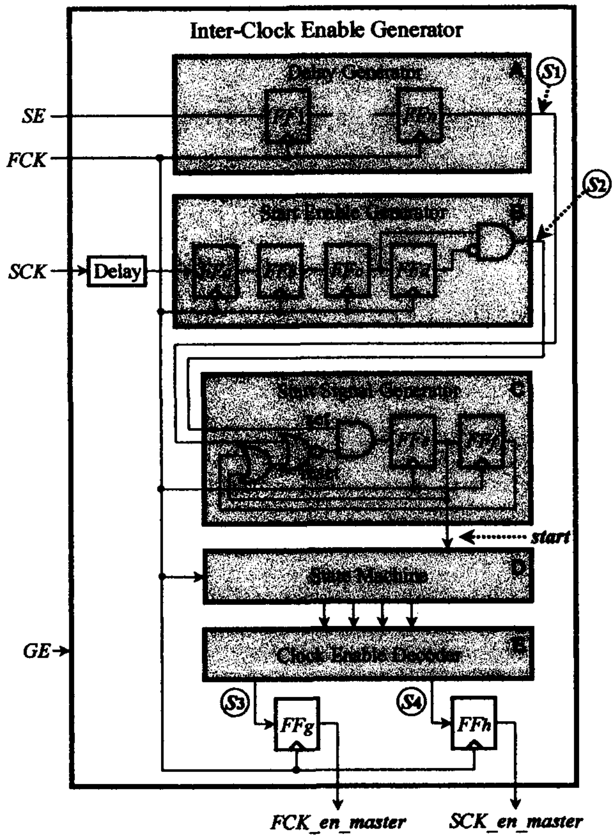


Fig. 9 Schematic of Inter-Clock Enable Generator.

The inter-clock enable generator consists of a delay generator (A), a start enable generator (B), a start signal generator (C), a state machine (D), and a clock enable decoder (E), in the operational order. Its inputs are two free-running synchronous clocks *FCK* (fast clock) and *SCK* (slow clock) as well as two enable signals *SE* and *GE*, while its outputs are two master clock enable signals *FCK_en_master* and *SCK_en_master*.

The inter-clock enable generator is a compact synchronous circuit driven by the fast clock *FCK*. Note that slow clock *SCK* is used as data input for the start enable generator. The operation of this circuit is shown in Fig. 10. As to be described, many design techniques are used to make this circuit functionally accurate and operational robust.

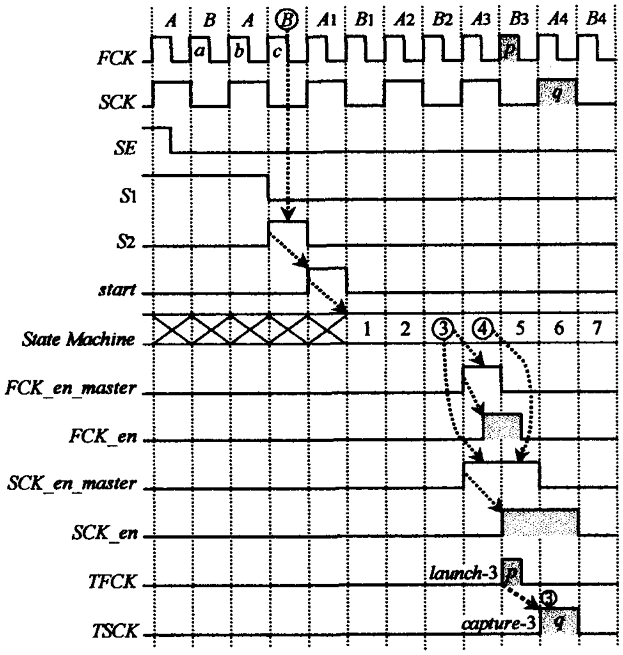


Fig. 10 Waveforms of Inter-Clock Enable Generator.

A. Delay Generator

The delay generator is basically an *n*-stage shift register driven by the fast clock *FCK*. Its purpose is to delay the effective arrival time of the falling edge of the scan enable signal *SE* at least by the time of (*n*-1) × T_{FCK}, where T_{FCK} is the period of *FCK*. This delay is used to make d_1 in Fig. 7 (b) long enough so that *SE* becomes non-timing-critical. This will significantly ease physical implementation.

For example, Fig. 10 shows the case where *n* is set to 3. That is, the *SE*→0 event takes a little more than 2 *FCK* clock periods to bring the internal signal S_1 to 0. It is this S_1→0 event that marks the actual start of a series of capture operations. This way, d_1 can be made as long as required.

Generally, d_1 needs to be made longer than the maximum delay of *SE* when *SE* is designed as a data signal, not as a clock signal requiring CTS. This maximum delay is then used to figure out the necessary value for *n*. For example, *n* was set to 8 for a 533MHz industrial circuit with 11.1M gates and 404.9K FFs. This created at least 13ns for d_1 in Fig. 7 (b), long enough to make *SE* non-timing-critical.

B. Start Enable Generator

The start enable generator consists of a 4-stage shift register, composed of FF_a ~ FF_d, as shown in Fig. 9. Its purposes are (1) to avoid *metastability* and (2) to provide a *re-timing* mechanism so that the state machine can be started properly.

Metastability may occur if the slow clock *SCK* is applied directly to the data input of an FF driven by the fast clock *FCK*. Two measures are taken against metastability: one is to slightly delay *SCK* before sending it to the shift register driven by *FCK*, and the other is to add a synchronizer composed of two FFs, FF_a and FF_b, on the input side of the shift register, as shown in Fig. 9.

The re-timing mechanism is realized by using a 2-bit shift register composed of FF_c and FF_d, as shown in Fig. 9. The outputs of the two FFs are decoded by the AND gate to generate the start enable signal S_2 so that it becomes 1 at the falling-edge of *SCK*, as shown in Fig. 10. This signal is used as a timing base to guarantee that the *start* signal for the state machine becomes 1 at the rising-edge of *SCK*.

C. Start Signal Generator

The start signal generator is designed as shown in Fig. 9. When S_1 is 1, *start* is 0 and the state machine does not operate. S_1→0 indicates the start of capture operations by releasing the reset input of the AND gate. After S_1 has become 0, if the start enable signal S_2 becomes 1, *start* will become 1. The feedbacks from the two FFs make *start* stay at 1 only for one *TCK* period, as shown in Fig. 10. This *start* signal is used to start the state machine.

D. State Machine

The state machine is basically a 4-bit counter, whose operation is shown in Fig. 11. It resets to state 0, and starts to count up from state 1 after the *start* signal becomes 1. After all states are traversed, it stops at state 0. The purpose of the state machine is to provide a means to accurately identify the phase and time relations of the clock pulses between the two input functional clocks *FCK* and *SCK*.

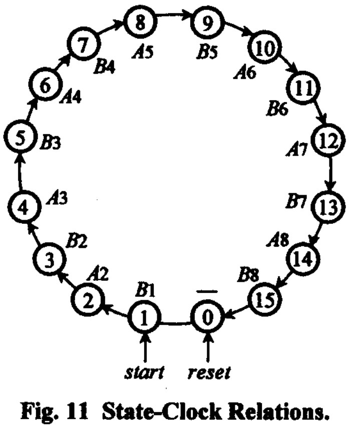


Fig. 11 State-Clock Relations.

In the example shown in Fig. 10, the state machine starts to count at the first rising-edge of *TCK* after *start* becomes 1. As a result, the dynamic relations between the fast clock *FCK* and the slow clock *SCK* can be fully represented by the current state of the state machine.

As shown in Fig. 11, states 1, 2, 3, 4, ..., 15 correspond to the clock relations B_1, A_2, B_2, A_3, ..., B_8, respectively. As shown in Fig. 10, *A* corresponds to the relation that *FCK* has a full-period and *SCK* has a half-period of 1, while *B* corresponds to the relation that *FCK* has a full-period and *SCK* has a half-period of 0. Obviously, by selectively decoding these states, master clock enable signals can be created at required timing and for required duration.

E. Clock Enable Decoder

The clock enable decoder creates two master clock enable signals, *FCK_en_master* and *SCK_en_master*, as shown in Fig. 9. The start time and duration of each master clock enable signal are determined by decoding the states of the state machine.

In the example shown in Fig. 10, the clock enable decoder reacts to the state 3 to create a pulse (not shown in Fig. 10) with the duration of one *TCK* period on its S_3 output. This pulse, delayed by FF_g for one *TCK* period, becomes the master clock enable signal *FCK_en_master*. Similarly, the clock enable decoder reacts to the state 3 and the state 4 to create a pulse (not shown in Fig. 10) with the duration of two *TCK* periods on its S_4 output. This pulse, delayed by FF_h for one *TCK* period, becomes the master clock enable signal *SCK_en_master*.

This way, two master clock enable signals, *FCK_en_master* and *SCK_en_master*, are accurately generated by the inter-clock enable generator of Fig. 9. Then, as shown in Fig. 8, the final clock enable signal *FCK_en* is obtained by delaying *FCK_en_master* by half period of *FCK* with the latch LA_1, while the final clock enable signal *SCK_en* is obtained by delaying *SCK_en_master* by half period of *SCK* with the latch LA_2. This is also shown in Fig. 10.

As shown in Fig. 10, the clock enable pulse on *FCK_en* can accurately pick out the *FCK* clock pulse *p* for the test clock *TFCK*, and the clock enable pulse on *SCK_en* can accurately pick out the *SCK* clock pulse *q* for the test clock *TSCK*. As a result, two capture pulses are generated for the test clocks *TFCK* and *TSCK* so that the inter-clock at-speed test requirements shown in Fig. 7 (b) are satisfied. Therefore, the at-speed testing of the inter-clock logic block *C* (*FCK*→*SCK*) can be successfully achieved.

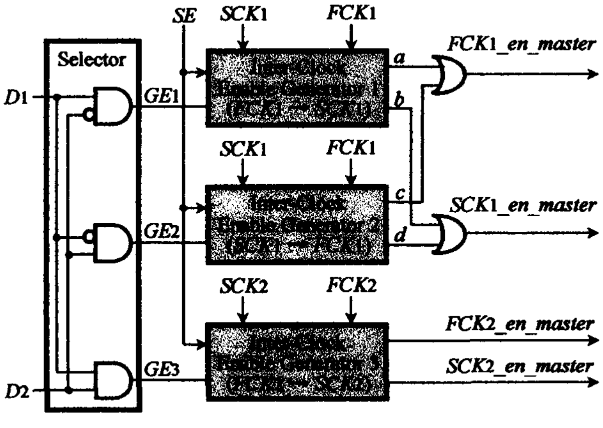
The characteristics of the inter-clock enable generator are summarized as follows:

- **High Flexibility**: Although different inter-clock logic blocks with different inter-clock relations need to use different inter-clock enable generators, only the clock enable decoder part needs to be slightly changed. That is, it is only necessary to make a clock enable decoder to decode the necessary states so that a master clock enable signal becomes 1 at the time and for the duration that are required by a specific inter-clock relation. In addition, multi-cycle paths can also be easily handled by slightly changing the decoding logic to increase the duration of a master clock enable pulse.

- **Wide Application**: The inter-clock enable generator design can be used in both ATE-based testing and logic BIST. Since only on-chip PLLs are needed to generate fast test clocks, there is no need to use a high-speed tester for at-speed testing. This helps reduce test costs.
- **Robust Operation**: Subtle issues such as metastability and clock skews are all carefully considered and measures against them are properly taken in logic design. This guarantees the robust operation of the inter-clock enable generator, even in the case of large and complicated industrial circuits.
- **Easy Implementation**: Since subtle issues such as metastability and clock skews are all handled by logic design techniques, no extra burden is placed on layout design in order to make the inter-clock enable generator work properly. This simplifies physical implementation.
- **Low Overhead**: The inter-clock enable generator design is compact. It contains about 20 FFs and only a small number of logic gates. Its area overhead is roughly 124 equivalent 2-input NAND gates for a typical configuration with an 8-stage delay generator.

3.4 Use of Multiple Inter-Clock Enable Generators

A multi-clock circuit usually contains multiple inter-clock logic blocks that should be targeted in at-speed testing. Basically, the at-speed testing of one inter-clock logic block needs one inter-clock enable generator. An example is shown in Fig. 12, in which there are 3 inter-clock enable generators, corresponding to three synchronous clock pairs: *FCK*1→*SCK*1, *SCK*1→*FCK*1, and *FCK*2→*SCK*2.


SE
SCK1
FCK1
Selector
D1
GE1
Inter-Clock Enable Generator 1 (FCK1→SCK1)
a
b
FCK1_en_master
SCK1
FCK1
GE2
Inter-Clock Enable Generator 2 (SCK1→FCK1)
c
d
SCK1_en_master
SCK2
FCK2
D2
GE3
Inter-Clock Enable Generator 3 (FCK2→SCK2)
FCK2_en_master
SCK2_en_master

a: *FCK*1*_en_master_*1
b: *SCK*1*_en_master_*1
c: *FCK*1*_en_master_*2
d: *SCK*1*_en_master_*2

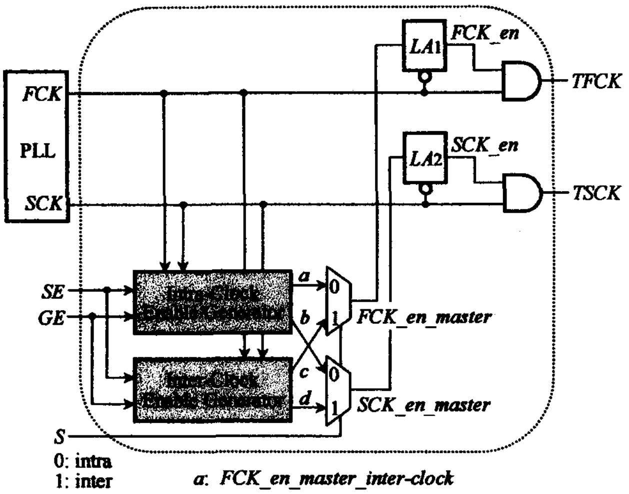
Fig. 12 Clock Enable Generator Selection.

Although it is possible to share some part of inter-clock enable generation circuitry among different inter-clock enable generators, this often increases control complexity and introduces more delay on functional clock lines. Given the fact that an inter-clock enable generator is small, it is advisable to use multiple inter-clock enable generators, one for each inter-clock logic block. In this case, a selection mechanism needs to be provided to activate the inter-clock enable generators one by one during at-speed testing.

In the example shown in Fig. 12, the selector is used for this purpose. *D*1 and *D*2 are decoded into 3 generator enable signals, *GE*1, *GE*2, and *GE*3, such that only one of them is 1 at any time. The activated inter-clock enable generator is used for test control, and the at-speed testing of the corresponding inter-clock logic block is achieved.

3.5 Top-Level Integration

The basic idea of this paper is to separate inter-clock test control from intra-clock test control in order to simplify test control operations. Therefore, in order to achieve complete at-speed testing, it is necessary to combine the new inter-clock at-speed test control scheme with an easy-to-implement intra-clock at-speed test control scheme, such as the double-capture scheme [17-19] as described in 2.3. The task of combining inter-clock and intra-clock test control schemes is conducted in top-level integration. An example is shown in Fig. 13, in which there are one inter-clock enable generator and one intra-clock enable generator, and the selection between them is made by the signal *S*.



a: *FCK_en_master_inter-clock*
b: *SCK_en_master_inter-clock*
c: *FCK_en_master_intra-clock*
d: *SCK_en_master_intra-clock*

Fig. 13 Top-Level Integration.

Note that, different from inter-clock at-speed test control, it is possible to use only one intra-clock enable generator for at-speed testing of all intra-clock logic blocks, based on the multi-capture technique [19] that generates at-speed capture pulses for all intra-clock logic blocks in a serial manner in the same capture window (*SE* = 1). Some other methods can achieve similar effects [17, 18].

3.6 Design Flow

The design flow for complete at-speed testing is shown in Fig. 14, which consists of three sub-flows: a netlist flow, an ATPG flow for ATE-based scan testing, and a fault simulation flow for logic BIST. The netlist flow is the basic flow that needs to be conducted for both ATE-based scan testing and logic BIST.

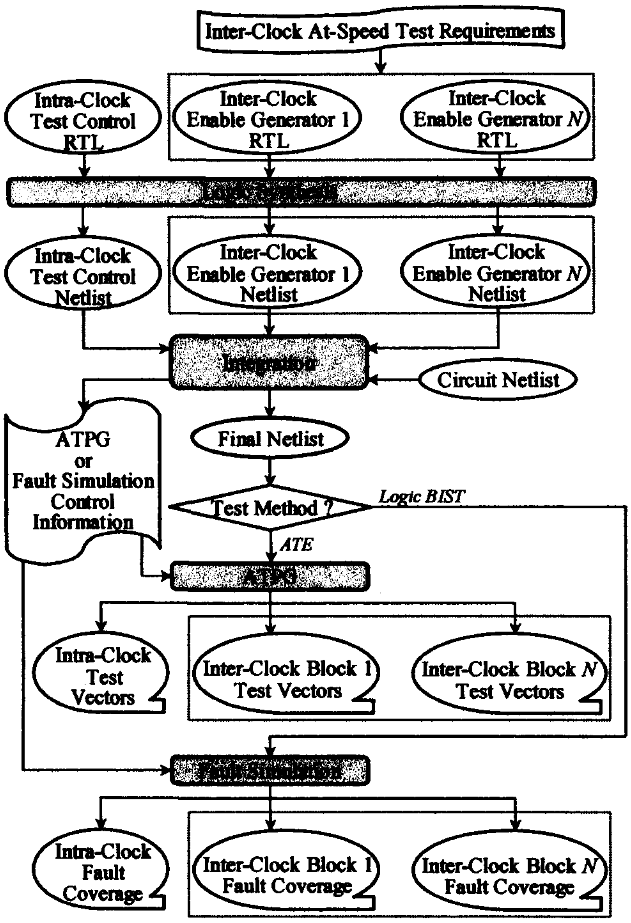

Inter-Clock At-Speed Test Requirements
Intra-Clock Test Control RTL
Inter-Clock Enable Generator 1 RTL
Inter-Clock Enable Generator N RTL
Logic Synthesis
Intra-Clock Test Control Netlist
Inter-Clock Enable Generator 1 Netlist
Inter-Clock Enable Generator N Netlist
Integration
Circuit Netlist
ATPG or Fault Simulation Control Information
Final Netlist
Test Method ?
Logic BIST
ATE
ATPG
Intra-Clock Test Vectors
Inter-Clock Block 1 Test Vectors
Inter-Clock Block N Test Vectors
Fault Simulation
Intra-Clock Fault Coverage
Inter-Clock Block 1 Fault Coverage
Inter-Clock Block N Fault Coverage

Fig. 14 Design Flow.

In the netlist flow, in addition to the intra-clock test control RTL design, RTL designs for all necessary inter-clock enable generators need to be prepared according to inter-clock at-speed test requirements. These RTL designs are synthesized into separate netlists. Then, at the integration stage, these test-oriented netlists are combined with the circuit netlist into the final netlist. Additional circuitry needed in inter-clock and intra-clock at-speed test control schemes are also added at the integration stage. In addition, control information necessary for the ATPG flow or the fault simulation flow is also generated at this stage.

In the ATPG flow, test generation is conducted for ATE-based scan testing. For all intra-clock logic blocks, it is possible to use only one ATPG run. However, in some cases, PLL-based multi-run ATPG is conducted, with one ATPG run for all intra-clock logic blocks related to one PLL. On the other hand, each inter-clock logic block requires its own ATPG run, and test vector sets for different inter-clock logic blocks are generated separately.

The fault simulation flow is similar to the ATPG flow in that each inter-clock logic block needs its own fault simulation run. The fault simulation flow is used for logic BIST, where test vectors are generated by a pseudo-random pattern generator (PRPG), and it is only necessary to fault-grade the test vectors.

3.7 Test Generation and Fault Simulation

As described above, in addition to the new inter-clock at-speed test control scheme, deterministic test vectors need to be generated for ATE-based scan testing, and random test vectors need to be fault-graded for logic BIST, in order to conduct inter-clock at-speed testing.

Fig. 15 shows the circuit model for test generation and fault simulation, corresponding to the at-speed testing of the inter-clock logic block *C* as shown in Fig. 7 (a). Obviously, only combinational logic blocks *A* and *C* need to be included for test generation and fault simulation of *C*.

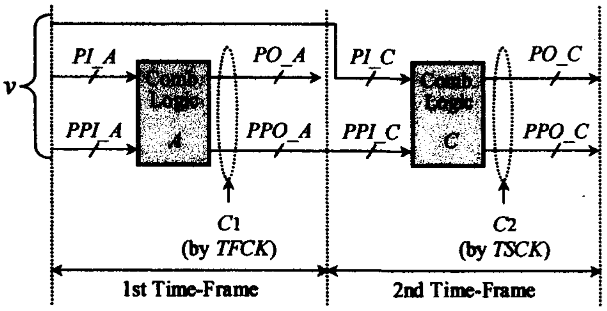


Fig. 15 Circuit Model for Test Generation & Fault Simulation.

Note that the circuit model shown in Fig. 15 is independent of the inter-clock at-speed test control scheme used. That is, the results of test generation or fault simulation are valid as long as at-speed capture pulses *C*1 and *C*2 are properly provided as shown in Fig. 7 (b). This separation of test generation and fault simulation from the underlying inter-clock at-speed test control scheme results in more flexibility in a test design flow.

4. Application Results

The new inter-clock at-speed test control scheme has been applied to a number of large-scale industrial designs, all with successful tape-outs. The inter-clock enable generator design and other circuitry necessary for the proposed test control scheme have been verified in logic design and layout implementation. Test vectors generated for inter-clock at-speed testing have also been verified on testers.

Due to page limitation, we only report the application results on one of the large industrial designs in the following. This design is in production now, and its circuit statistics are shown in Table 1.

Table 1 Circuit Statistics

# of Gates	11.1M
# of FFs	404.9K
# of Scan Chains	32
Max. Chain Length	13598
No. of Clock Domains	11
Min. Frequency	66MHz
Max. Frequency	533MHz

Intra-clock logic blocks were tested by logic BIST with top-up ATPG vectors. The transition delay fault coverage for all intra-clock logic blocks was 96.9% with 64K random vectors in logic BIST and 8183 vectors in top-up ATPG.

Inter-clock logic blocks were tested by the new inter-clock at-speed test control scheme with ATPG vectors, and the results are shown in Table 2. In this case, 6 inter-clock logic blocks, *A* ~ *F*, were targeted. For example, *A* is a logic block from a 100MHz clock to a 300MHz clock, and contains 36858 transition delay faults. Table 2 also shows the information on the number of ATPG vectors, fault coverage, and CPU time. Furthermore, the area overhead in this application was mainly due to 6 inter-clock enable generators, each containing 20 FFs and consuming an area of roughly 124 equivalent 2-input NAND gates.

Table 2 Application Results

Inter-Clock Logic Blocks	From (MHz)	To (MHz)	# of Faults	ATPG		
				# of Vec.	Fault Cov.	CPU (h:m)
A	100	300	36858	232	86.4	4:30
B	133	533	8350	32	100	0:15
C	133	266	4942	36	100	0:14
D	533	133	1940	9	100	0:10
E	266	533	732	9	100	0:10
F	266	133	64	3	100	0:09

Table 2 shows that some inter-clock logic blocks, such as *A*, was quite large, and ignoring them in at-speed testing would have been a tremendous risk to chip quality. This was why inter-clock at-speed testing was made mandatory by the user company of this design. The new inter-clock at-speed test control scheme successfully provided a novel and practical way to improve the overall chip quality by making inter-clock at-speed testing easy and efficient.

5. Conclusions

The paper addressed the test control problem in at-speed testing with a novel inter-clock at-speed test control scheme, featuring a compact, robust, and easy-to-implement inter-clock enable generator design. By combining this scheme with any existing intra-clock at-speed test control scheme, complete at-speed testing of the entire circuit can be efficiently achieved, which significantly improves the quality of at-speed testing. The effectiveness of the new inter-clock at-speed test control scheme has been proven by successful applications to large industrial circuits.

References

[1] T. M. Mak, A. Krstic, K. Cheng, and L. Wang, "New Challenges in Delay Testing of Nanometer, Multigigahertz Designs," *IEEE Design and Test of Computers*, Vol. 21, No. 3, pp. 241-247, 2004.

[2] G. Aldrich and B. Cory, "Improving Test Quality and Reducing Escapes," *Proc. Fabless Forum*, pp. 34-35, 2003.

[3] L.-T. Wang, X. Wen, H. Furukawa, F. Hsu, S. Lin, S. Tsai, K. S. Abdel-Hafez, and S. Wu, "VirtualScan: A New Compressed Scan Technology for Test Cost Reduction," *Proc. Int'l Test Conf.*, pp. 916-925, 2004.

[4] C. Barnhart, V. Brunkhorst, F. Distler, O. Farnsworth, B. L. Keller, B. Könemann, and A. Ferko, "The OPMISR: Foundation for Compressed ATPG Vectors," *Proc. Int'l Test Conf.*, pp. 748-757, 2001.

[5] J. Rajski, M. Kassab, N. Mukherjee, N. Tamarapalli, J. Tyszer, and J. Qian, "Embedded Deterministic Test for Low-Cost Manufacturing," *IEEE Design & Test of Computers*, Vol. 20, No. 5, pp. 58-66, 2003.

[6] P. Wohl, J. Waicukauski, S. Patel, and M. Amin, "Efficient Compression and Application of Deterministic Patterns in a Logic BIST Architecture," *Proc. Design Automation Conf.*, pp. 934-939, 2003.

[7] M. Beck, O. Barondeau, M. Kaibel, F. Poehl, X. Lin, R. Press, "Logic Design for On-Chip Test Clock Generation - Implementation Details and Impact on Delay Test Quality," *Proc. Proc. Design, Automation and Test in Europe*, pp. 56-61, 2005.

[8] J. Savir and P. Bardell, "Built-in Self-Test: Milestones and Challenges," *Gordon and Breach Science Publishers, VLSI Design*, Vol. 1, No. 1, pp. 23-44, 1993.

[9] J. Rearick, "Too Much Delay Fault Coverage Is a Bad Thing," *Proc. Int'l Test Conf.*, pp. 624-633, 2001.

[10] S. Mitra, E. Volkerink, E. McCluskey, and S. Eichenberger, "Delay Defect Screening Using Process Monitor Structures," *Proc. VLSI Test Symp.*, pp. 43-52, 2004.

[11] Y. Sato, S. Hamada, T. Maeda, A. Takatori, Y. Nozuyama, and S. Kajihara "Invisiable Delay Quality – SDQM Model Lights Up What Could Not Be Seen," *Proc. Int'l Test Conf.*, Paper 47.1, 2005.

[12] P. Gupta and M. S. Hsiao, "High Quality ATPG for Delay Defects," *Proc. Int'l Test Conf.*, pp. 584-591, 2003.

[13] L. Lee, L. Wang, P. Parvathala, and T. M. Mak, "On Silicon-Based Speed Path Identification," *Proc. VLSI Test Symp.*, pp. 35-41, 2005.

[14] S. Kajihara, M. Fukunaga, X. Wen, T. Maeda, S. Hamada, and Y. Sato, "Path Delay Test Compaction with Process Variation Tolerance," *Prof. Design Automation Conf.*, pp. 845-850, 2005.

[15] G. Hetherington, T. Fryars, N. Tamarapalli, M. Kassab, A. Hassan, and J. Rajski, "Logic BIST for Large Industrial Designs: Real Issues and Case Studies", *Proc. Intl. Test Conf.*, pp. 358-367, 1999.

[16] B. Nadeau-Dostie, D. Burek, and A. Hassan, "ScanBist: A Multifrequency Scan-Based BIST Method," *IEEE Design & Test of Computers*, Vol. 11, No. 1, pp. 7-17, 1994.

[17] X. Lin, R. Press, J. Rajski, P. Reuter, T. Rinderknecht, B. Swanson, and N. Tamarapalli, "High-Frequency, At-Speed Scan Testing," *IEEE Design & Test of Computers*, pp. 17-25, Vol. 20, No. 5, 2003.

[18] M. Beck, O. Barondeau, F. Poehl, and X. Lin, "Measures to Improve Delay Fault Testing on Low-Cost Testers – A Case Study," *Proc. VLSI Test Symp.*, pp. 223-228, 2005.

[19] L.-T. Wang, X. Wen, B.-C. Hsu, S. Wu, and J. Guo, "A Flexible Logic BIST Scheme for Multiple-Clock Circuits," *Proc. Intl. Conf. on Computer Design*, pp. 475-478, 2005.

[20] "Clock Domain Crossing," *Cadence Design Systems, Inc.*

[21] S. Savir and S. Patil, "On Broad-Side Delay Test," *Proc. VLSI Test Symp.*, pp. 284-290, 1994.

[22] L.-T. Wang, C.-W. Wu, and X. Wen, (Editors), *VLSI Test Principles and Architectures: Design for Testability*, Elsevier Science, July 2006.

4. Xビット判定技術

テストベクトルの中間形態であるキューブの中に、故障検出率と無関係なビット（Xビット）が高い割合で存在することが示されている。しかし、テスト生成の段階からこのようなXビットを残す場合、最終テスト集合が膨大になる欠点がある。そのため、テスト生成の段階においてXビットにランダムな論理値を割り当てることによる動的圧縮を実施し、コンパクトな初期テスト集合を作る必要がある。このような論理値のみで構成される初期テスト集合から再び故障検出率と無関係なXビットを特定する研究開発が求められている。

与えられたテスト集合が論理値（0と1）しか含まない場合でも、その故障検出率を下げずに多くのビットをドントケア（Xビット）にすることができる。また、このように判定されたXビットに再度最適な論理値を埋め込むことによって、テスト品質向上、テスト消費電力削減など、様々な効果を得ることできる。そこで本研究では、Xビットをなるべく多く判定する手法を開発し、その有効性を実験で示した。その研究成果を4.1で紹介する。

与えられたテスト集合から特定するXビットの数が多ければ多いほど、低消費電力の削減効果が大きい。このため、故障検出に必要な論理値ビット数をなるべく抑える必要がある。そこで本研究では、テスト生成時においてなるべく多くのXビットを残す手法を開発し、その有効性を実験で示した。その研究成果を4.2で紹介する。

テスト集合内のXビット位置はそのテスト時消費電力の削減効果を左右することが多い。つまり、テスト品質を向上させるためのXビット分布は必ずしもテスト時消費電力の削減にとって有効ではない。そこで本研究では、消費電力削減に最も効果的なビット位置を特定し、それらを優先にXビットに判定する手法を開発し、その有効性を実験で示した。その研究成果を4.3で紹介する。

4. 1. テスト品質向上のためのXビットの判定手法

与えられたテスト集合が論理値しか含まない場合でも、その故障検出率を下げずに多くのビットをドントケア（Xビット）にすることができる。また、これらのXビットはテスト品質向上、テスト消費電力削減など、様々な応用分野に適用することができる。Xビットをなるべく多く判定する手法を本研究で開発し、その成果を次の国際会議で発表した。ここではその内容を紹介する。

K. Miyase, K. Terashima, S. Kajihara, X. Wen S. M. Reddy:
On Improving Defect Coverage of Stuck-at Fault Tests
Proc. IEEE Asian Test Symp., pp. 216-221 (2005)

4. 2. 信号線正当化のための最小テストキューブの導出手法

与えられたテスト集合から特定するＸビットの数が多ければ多いほど、低消費電力の削減効果が大きい。テスト生成時においてなるべく多くのＸビットを残す手法を本研究で開発し、その成果を次の国際会議で発表した。ここではその内容を紹介する。

K. Miyase, S. Nagayama, S. Kajihara, X. Wen S. M. Reddy:
On the Extraction of a Minimum Cube to Justify Signal Line Values
Informal Digest of IEEE European Test Symp., pp. 79-84 (2005)

4. 3. 低消費電力テスト生成のためのXビットの判定手法

テスト集合内のＸビット位置はそのテスト時消費電力の削減効果を左右することが多い。テスト時消費電力の削減に最も効果的なビット位置を特定し、それらを優先にＸビットに判定する手法を本研究で開発し、その成果を次の学会誌で発表した。ここではその内容を紹介する。

X. Wen, T. Suzuki, S. Kajihara, K. Miyase, Y. Minamoto, L.-T. Wang, K. K. Saluja:
Efficient Test Set Modification for Capture Power Reduction
J. of Low Power Electronics, Issue 3, pp. 319-330 (2005)

5. Xビット埋め込み技術

テストベクトルの中間形態であるキューブの中に、故障検出率と無関係なビット（Xビット）が高い割合で存在することが示されている。これらのXビットに対して、テスト消費電力が低下するように最適な論理値を決定する研究開発が求められている。このような、Xビット埋め込みによる低消費電力の実現は、回路変更に伴う面積増大、性能低下などの欠点がないため、高い実用性を有している。

単一キャプチャ環境において低消費電力を実現するために、故障検出率を落とさず記憶素子（フリップ・フロップ）における状態数を最大限に削減する必要がある。そのため、高い精度でXビットへ割り当てる論理値を決定することが重要である。そこで本研究では、テストパターン自動生成（ATPG）において多用される正当化操作（Justification）や含意操作（Implication）を利用してフリップ・フロップの入出力値の差を削減する手法を開発し、その有効性を実験で示した。その研究成果を５.１で紹介する。

実速度テストにおいては、物理実現の容易さから２重キャプチャというクロック方式が多用されている。テスト時消費電力削減の観点から、２つのキャプチャにおいてフリップ・フロップの状態変化数を削減する必要がある。そこで本研究では、一番目のキャプチャ（C1）と二番目のキャプチャ（C2）におけるフリップ・フロップの状態変化数をバランスよく最小化する手法を開発し、その有効性を実験で示した。その研究成果を５.２で紹介する。

論理回路内の状態遷移量は、記憶素子（フリップ・フロップ）における状態数のみでは精確に評価することができない場合がある。このため、フリップ・フロップの状態変化数に基づくXビット埋め込み手法では、十分なテスト消費電力削減効果が得られないこともある。そこで本研究では、フリップ・フロップのみでもなく組合せ回路部の論理ゲートのみでもなく、両方における状態変化数を最小化することによって、キャプチャ時消費電力を削減する手法を開発し、その有効性を実験で示した。その研究成果を５.３で紹介する。

5. 1. 単一キャプチャテストにおけるXビット埋め込み手法

単一キャプチャ環境において低消費電力を実現するために、記憶素子（フリップ・フロップ）における状態数を削減する必要がある。テストパターン自動生成（ATPG）において多用される正当化操作（Justification）や含意操作（Implication）を利用してフリップ・フロップの入出力値の差を削減する手法を本研究で開発し、その成果を次の国際会議と学会誌で発表した。ここではその内容を紹介する。

X. Wen, Y. Yamashita, S. Kajiihara, L.-T. Wang, K. K. Saluja, K. Kinoshita:
A New Method for Low-Capture-Power Test Generation for Scan Testing
IEICE Trans. Inf. & Syst., Vol. E89-D, No. 5, pp. 1679-1686 (2006)

5. 2. 2重キャプチャテストにおけるXビット埋め込み手法

２重キャプチャは実速度テストを実現するための重要なクロック方式である。一番目のキャプチャ（C1）と二番目のキャプチャ（C2）におけるフリップ・フロップの状態変化数をバランスよく最小化する手法を本研究で開発し、その成果を次の国際会議で発表した。ここではその内容を紹介する。

X. Wen, Y. Yamashita, S. Morishima, S. Kajiihara, L. Wang, K. Saluja, K. Kinoshita:
Low-Capture-Power Test Generation for At-Speed Scan Testing
Proc. IEEE Int'l Test Conf., Paper. 39.2 (2005)

Low-Capture-Power Test Generation for Scan-Based At-Speed Testing

Xiaoqing Wen [1], Yoshiyuki Yamashita [1], Shohei Morishima [1], Seiji Kajihara [1], Laung-Terng Wang [2], Kewal K. Saluja [3], and Kozo Kinoshita [4]

[1] Dept. of CSE, Kyushu Institute of Technology, Iizuka 820-8502, Japan
[2] SynTest Technologies, Inc., 505 S. Pastoria Ave., Suite 101, Sunnyvale, CA 94086, USA
[3] Dept. of ECE, 1415 Engineering Drive, University of Wisconsin - Madison, Madison, WI 53706, USA
[4] Faculty of Informatics, Osaka Gakuin University, Suita 564-8511, Japan

Abstract

Scan-based at-speed testing is a key technology to guarantee timing-related test quality in the deep submicron era. However, its applicability is being severely challenged since significant yield loss may occur from circuit malfunction due to excessive IR drop caused by high power dissipation when a test response is captured. This paper addresses this critical problem with a novel low-capture-power X-filling method of assigning 0's and 1's to unspecified (X) bits in a test cube obtained during ATPG. This method reduces the circuit switching activity in capture mode and can be easily incorporated into any test generation flow to achieve capture power reduction without any area, timing, or fault coverage impact. Test vectors generated with this practical method greatly improve the applicability of scan-based at-speed testing by reducing the risk of test yield loss.

1. Introduction

Scan-based testing, carried out by a tester on a full-scan circuit with deterministic test vectors obtained through automatic test pattern generation (ATPG), is the most widely adopted test strategy to achieve required test quality for an integrated logic circuit at acceptable costs.

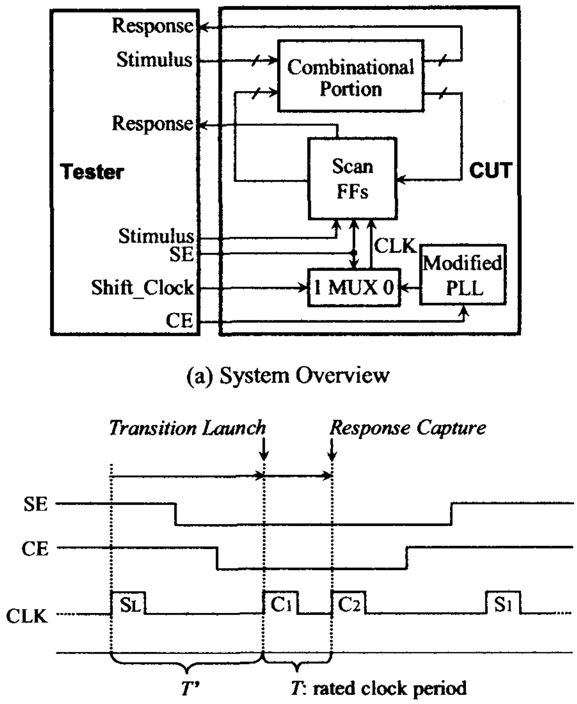
In a full-scan sequential circuit, scan flip-flops (FFs) replace all functional FFs and operate in two modes: *shift* and *capture*. In shift mode, scan FFs form scan chains, through which a test vector is applied during shift-in or a test response is observed during shift-out, for the combinational portion of the circuit. In capture mode, scan FFs operate as functional FFs and load the test response of the combinational portion for a test vector into themselves, getting ready for shift-out later in shift mode. Thus, the problems of testing a full-scan sequential circuit is reduced to that of testing its combinational portion, in that now it is sufficient to generate test vectors only for the combinational portion with combinational ATPG.

In scan-based testing, after a test vector is applied in shift mode, its test response is loaded into FFs in capture mode after a waiting period either *greater than* or *equal to* the rated clock period. The former is called *low-speed testing*, and the latter is called *at-speed testing* [1, 2]. Low-speed testing checks for unexpected logic values based on such fault models as stuck-at and bridging; while at-speed testing checks for unexpected excessive delays based on such fault models as transition delay and path delay.

As transistor feature sizes shrink, more chips fail because of timing-related defects [3]. I_{DDQ} testing [4] was widely used for screening out such defective chips, but is now losing its effectiveness due to elevated normal quiescent current. Therefore, at-speed testing through options such as logic built-in self-test (BIST) or external scan-based testing needs to be considered.

Compared to at-speed logic BIST, which is difficult to implement and usually has low fault coverage because of random pattern usage, scan-based at-speed testing with ATPG and an external tester has the advantages of low circuit overhead, low application cost, and high test quality [2]. As a result, scan-based at-speed testing, especially when conducted by using on-chip phase-locked loops (PLLs) [5], has emerged as a key technology for guaranteeing test quality in the deep submicron (DSM) era.

Fig. 1 shows an example scan-based at-speed testing system with on-chip PLL and the broadside clocking scheme.



(a) System Overview

(b) Broadside Clocking Scheme

Fig. 1 Scan-Based At-Speed Testing System.

0-7803-9039-3/$20.00 © 2005 IEEE

In Fig. 1, SE and CE are the normal scan enable signal and a newly added capture enable signal, respectively. As shown in Fig. 1 (a), a test vector is applied in shift mode (SE = 1) via a series of shift clock pulses with S_L being the last one. In capture mode (SE = 0), the modified PLL responds to the falling edge of the CE signal to provide two pulses C_1 and C_2 at the rated clock interval of T as shown in Fig. 1 (b). C_1 launches a transition with respect to S_L while C_2 captures the circuit response to the transition at-speed. As a result, timing-related defects can be detected. This is the so-called *broadside clocking scheme* or *launch-off-capture clocking scheme* [1, 2].

The scan-based at-speed testing system shown in Fig. 1 has the following advantages: (1) *Low Tester Requirement*: A low-speed tester can be used to provide shift clock pulses at a lower frequency than the rated clock frequency, while only the at-speed capture clock pulse, e.g. C2 in Fig. 1 (b), needs to be generated by the on-chip PLL that is also used in functional mode. (2) *Easy Physical Implementation*: The broadside clocking scheme only needs a non-timing-critical, thus easy-to-implement, SE signal since T' can be much larger than the rated clock period of T in Fig. 1 (b). This makes it easier for the broadside clocking scheme to be physically implemented than other clocking schemes, such as *skewed-load* or *launch-off-shift* [2]. (3) *High Test Quality*: The broadside clocking scheme generally activates fewer false paths since logic value transitions are generated by the difference between a shifted-in vector and the functional response to the vector. This generally results in less "over-testing" than the skewed-load scheme.

The above advantages make scan-based at-speed testing with on-chip PLL and the broadside clocking scheme highly preferable for screening out chips with timing-related defects in production testing. However, the adoption of this testing technology is being severely hindered by four problems: (1) *test data volume*, (2) *test application time*, (3) *test heat dissipation*, and (4) *test yield loss*.

The test data volume and test application time problems are caused by larger gate/FF counts, longer scan chains, and the use of complex delay fault models, all inevitable in the DSM era. Several approaches, such as test compaction, multi-capture clocking, decompression-compression, encoding, are available for addressing these problems.

The test heat dissipation and test yield loss problems are both related to test power dissipation during scan testing, which is much higher than during normal operation [6].

Test heat is caused by the *accumulated* effect of test power dissipation, mostly in shift mode for a large number of cycles. Excessive heat may cause permanent damage to the chip-under-test, increasing package costs, or reducing circuit reliability due to accelerated electromigration [7].

Previous methods for test heat reduction are based on four major approaches: *scheduling*, *test vector manipulation*, *circuit modification*, and *scan chain modification*, to reduce the switching activity in shift mode. Test scheduling [8, 9] takes the power budget into consideration when selecting modules to be tested simultaneously. Test vector manipulation includes power-aware ATPG [10, 11], static compaction [12], test vector modification [13], test vector reordering [14], test vector compression [15], and coding [16]. Circuit modification includes transition blocking [17], clock gating [18], and the use of multiple clock duties [19]. Scan chain modification includes scan chain reordering [15, 20], scan chain partitioning [21], and scan chain modification [22]. Methods tailored for BIST applications, such as toggle suppression [23] and low-power test pattern generation [24], have also been proposed.

Test yield loss is caused by excessive *instantaneous* test power dissipation in both shift and capture mode, because FFs and/or PLL may malfunction due to power supply voltage drop and ground bounce [19, 25, 28]. This problem is worsening as feature sizes shrink below 0.18 micron.

Most of the previous methods [8-24] for test heat reduction in shift mode also reduce instantaneous test power dissipation in shift mode, thus lowering the risk of test yield loss in shift mode. Among them, the multi-duty scan method [19] is especially effective, which changes clock duties so that fewer FFs operate simultaneously.

There are a few methods for reducing test yield loss in capture mode. One method [26] uses an interleaving scheme to reduce the number of FFs that are clocked simultaneously in capture mode, at the cost of increased control complexity. Another method [27] uses an *X*-filling technique in static compaction to reduce the number of capture transitions at FFs. Yet another method [28], called *single-capture low-capture-power* (*SC-LCP*) *X*-filling, conducts algorithmic *X*-Filling in dynamic or static compaction so as to reduce circuit switching activity in capture mode. These methods, however, only work for low-speed testing and at-speed testing based on the skewed-load clocking scheme, both featuring a single capture pulse.

The impact of IR-drop for capture mode in scan-based at-speed testing has been analyzed in [25] for the broadside clocking scheme, where two capture pulses are used. Quiet test vectors, which result in low switching activity in capture mode, were shown to be beneficial. However, it was also shown that existing ATPG programs failed to generate such "hot" test vectors when the straightforward approach of placing additional constraints was used. This is a serious problem since "cool" test vectors may result in significant test yield loss, thus severely challenging the applicability of scan-based at-speed testing that is considered indispensable in the DSM era.

This paper proposes a unique and novel ATPG approach to reducing instantaneous test power dissipation in capture mode for scan-based at-speed testing with the broadside clocking scheme. The basic idea is to make use of *test cubes*, i.e., test vectors with unspecified bits (*X-bits*), which exist during ATPG. We develop a novel technique, called *double-capture low-capture-power* (*DC-LCP*) *X*-filling, for algorithmically assigning 0's and 1's to the *X*-bits in test cubes so as to reduce the circuit switching activity caused by two capture pulses in the broadside clocking scheme.

The DC-LCP *X*-filling method can be easily incorporated into dynamic compaction of any test generation flow, and the resulting "cool" test vectors can achieve capture power reduction without any area, timing, or fault coverage impact. As a result, test yield loss in capture mode can be efficiently lowered, thus greatly improving the applicability of scan-based at-speed testing with the broadside clocking scheme.

The rest of the paper is organized as follows: Section 2 describes the research background. Section 3 presents the DC-LCP *X*-filling method. Section 4 shows experimental results, and Section 5 concludes the paper.

2. Background

As shown in Fig. 2, an integrated circuit can be seen as a network of interconnected transistors existing between a VDD (power grid) and a VSS (ground grid). These transistors form functional cells, i.e. logic gates and FFs. Cells switch their output values dynamically to perform various required functionality.

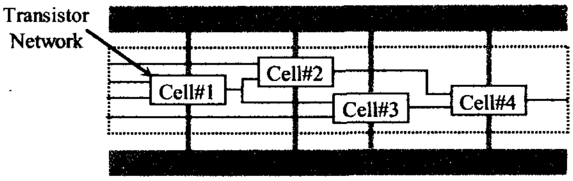
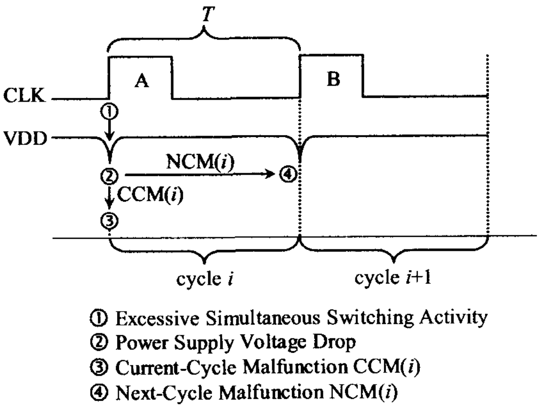


Fig. 2 Example Integrated Circuit with Power/Ground Grids.

Whenever a cell switches its output, a dynamic current path will be established between VDD and VSS. If a large amount of cells switch their outputs simultaneously, i.e. if instantaneous power is too large, a significant power supply voltage drop will occur. The major reason is the IR effect since a dynamic current (I) flows through the resistance (R) of the VDD grid, the VSS grid, and the transistor network. In addition, parasitic or capacitive effects also contribute to power supply voltage drop. Generally, the amount of power supply voltage drop depends on the number of simultaneous switching cells, their types, their locations, etc.

Normally, the power supply/ground pins and distribution system of a circuit is designed only for handling the peak power that occurs during normal operation. Thus, it may not be able to handle the large instantaneous power that could occur during scan testing since test power is much higher than normal power. Unfortunately, IR drop analysis for test mode is almost never conducted in most design flows, making a circuit vulnerable to test yield loss.

For example, as reported in [19], power supply voltage even dropped to 17% of its normal value in a 0.18micron industrial design during scan testing. Such power supply voltage drop in test mode may cause circuit malfunction, resulting in test yield loss [19, 25]. Its mechanism is illustrated in Fig. 3.



① Excessive Simultaneous Switching Activity
② Power Supply Voltage Drop
③ Current-Cycle Malfunction CCM(*i*)
④ Next-Cycle Malfunction NCM(*i*)

Fig. 3 Malfunctions due to Power Supply Voltage Drop.

In Fig. 3, simultaneous switching activity (①) increases dynamic power dissipation, which in turn causes power supply voltage drop (②). The direct result is nonlinear performance degradation of transistors, especially in a DSM circuit of very fine geometries. For a FF consisting of degraded transistors, the degradation can translate into direct malfunction, i.e. loading of a wrong value into the FF in the same cycle where simultaneous switching activity occurs. This is called the *current-cycle malfunction* (*CCM*) (③). In addition, for a combinational logic gate consisting of degraded transistors, the degradation often translates into increased gate delay, which in turn increases path delays in a circuit. Generally, a 10% drop in power supply voltage can increase path delay by 30%. The increased path delay may violate required timings at some FFs in the next cycle, also resulting in circuit malfunction. This is called the *next-cycle malfunction* (*NCM*) (④). Moreover, supply power drop may also cause PLL malfunction. Obviously, all these factors may result in potential test yield loss.

Note that the clock pulse A in Fig. 3 can be a shift pulse or a capture pulse. This means that test yield loss may occur in shift mode or capture mode or both. Several techniques exist for reducing switching activity in shift mode [8-24] and in capture mode [25-28].

Also note that scan-based at-speed testing is especially vulnerable to the power supply voltage drop problem. The reason is that, in at-speed testing, there must be one clock interval equal to the related clock period for each test vector. Suppose *T* in Fig. 3 is such an interval. Since *T* is very short for a high-speed circuit, the risk of voltage-drop-induced delays causing next-cycle malfunction is high, thus increasing the total risk of test yield loss.

For scan-based at-speed testing with the broadside clocking scheme, which features two capture pulses as shown in Fig. 1 (b), its detailed mechanism for voltage-drop-induced malfunction is illustrated in Fig. 4.

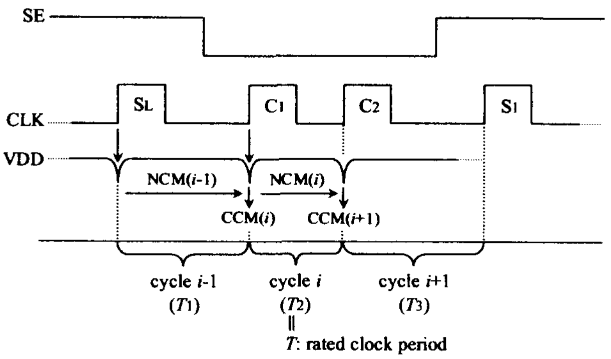


Fig. 4 Possible Malfunctions in Broadside Clocking Scheme.

As shown in Fig. 4, possible malfunctions related to test yield loss in capture mode for the broadside clocking scheme are NCM(*i*-1), CCM(*i*), NCM(*i*), and CCM(*i*+1), as described below:

(1) NCM(*i*-1) is the next-cycle malfunction for cycle *i*-1, which is the last shift cycle. Its risk, however, is low since *T*1 can be made as large as necessary, allowing enough timing margin for absorbing any voltage-drop-induced delay. Thus, NCM(*i*-1) can be ignored.

(2) CCM(*i*) and CCM(*i*+1) are current-cycle malfunctions for the two capture pulses, i.e. cycle *i* and cycle *i*+1, respectively. Both of them may cause test yield loss and their risks need to be contained by reducing the circuit switching activity at C1 and C2.

(3) NCM(*i*) is the next-cycle malfunction for cycle *i*, and it may cause malfunction at C2. This is because *T*2 is equal to the rated clock period, which is very short for a high-speed circuit, leaving less space in timing margin for absorbing voltage-drop-induced delay. Its risk needs to be contained by reducing the circuit switching activity at C1.

Thus, to reduce the test yield loss in capture mode for scan-based at-speed testing with the broadside clocking scheme, it is necessary to reduce the risks of CCM(*i*), NCM(*i*), and CCM(*i*+1) by reducing the circuit switching activity at C1 and C2. The next section presents an innovative method to achieve this goal.

3. Low-Capture-Power Test Generation

3.1 Test Generation Flow

In ATPG, a primary target fault is selected from undetected faults and a test *v* is generated for it. At this stage, *v* usually contains unspecified bits (*X*-bits), and it is called a *test cube*. Next, a conventional dynamic compaction as shown in Fig. 5 is conducted for *v* to detect more faults.

In Fig. 5, the function *promising*(*v*) decides whether *v* is a good candidate for dynamic compaction. If *v* is promising, *X*-bits in *v* will be explored algorithmically to see whether a secondary target fault can be detected. If *v* is not promising, random *X*-filling is conducted to all remaining *X*-bits in *v*.

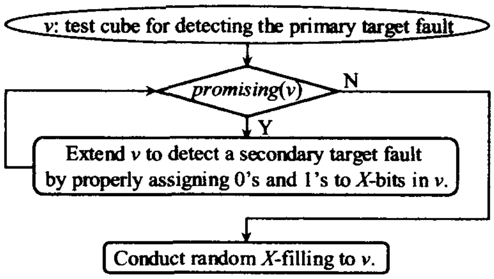

v: test cube for detecting the primary target fault
promising(v)
N
Y
Extend v to detect a secondary target fault by properly assigning 0's and 1's to X-bits in v.
Conduct random X-filling to v.

Fig. 5 Conventional Dynamic Compaction Flow.

Random *X*-filling may help in reducing the number of total test vectors since it increases the chances of detecting additional faults. These additionally detected faults can be identified by fault simulation after random *X*-filling. However, random *X*-filling usually adversely affects test power dissipation [12].

The basic idea of low-capture-power test generation is to algorithmically, instead of randomly, assign 0's and 1's to *X*-bits in the *X*-filling stage, so that capture power dissipation is reduced. Fig. 6 shows the proposed dynamic compaction flow for low-capture-power test generation.

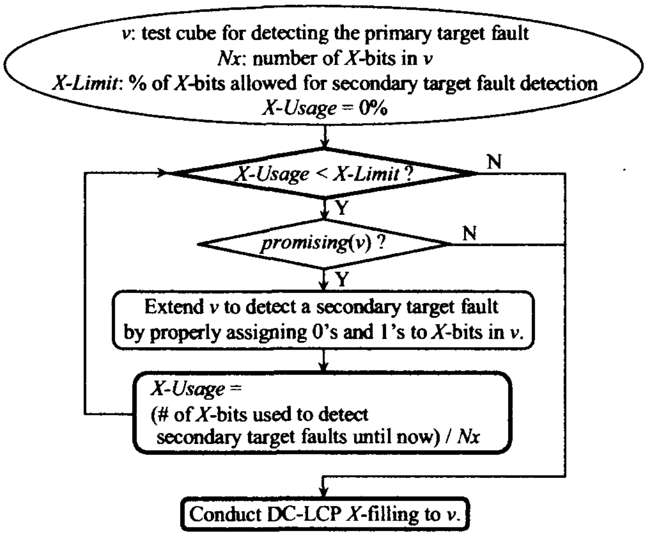

v: test cube for detecting the primary target fault
Nx: number of X-bits in v
X-Limit: % of X-bits allowed for secondary target fault detection
X-Usage = 0%
X-Usage < X-Limit ?
N
Y
promising(v) ?
N
Y
Extend v to detect a secondary target fault by properly assigning 0's and 1's to X-bits in v.
X-Usage = (# of X-bits used to detect secondary target faults until now) / Nx
Conduct DC-LCP X-filling to v.

Fig. 6 Proposed Dynamic Compaction Flow.

In Fig. 6, *X*-filling is conduced by a new method called *double-capture low-capture-power* (*DC-LCP*), instead of random *X*-filling. That is, different from the conventional dynamic compaction flow as shown in Fig. 5, *X*-bits in the proposed dynamic compaction are used not only with the objective of reducing the number of total test vectors but also with the objective of reducing capture power dissipation. Obviously, these two objectives can be conflicting.

In order to balance the conflicting objectives, a new concept, called *X-usage control*, is introduced. As shown in Fig. 6, *X-Limit* is a user-specified threshold that defines the percentage of original *X*-bits to be allowed for detecting secondary target faults. A measure, *X-Usage*, is updated each time a secondary target fault is detected. When *X-Usage* becomes greater than *X-Limit,* test cube extension for the objective of fault-detection is terminated and DC-LCP *X*-filling is invoked for the remaining *X*-bits to achieve the objective of reducing capture power dissipation.

The details of the DC-LCP *X*-filling method are presented in the following subsections 3.2 through 3.4.

3.2 Circuit Model

For the convenience of presentation in the following, a signal scan chain in a single clock domain, as shown in Fig. 7, is assumed. The DC-LCP *X*-filling method, however, can be readily extended for any full-scan circuit with multiple scan chains in multiple clock domains.

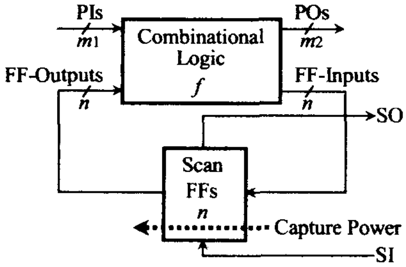


Fig. 7 General Full-Scan Circuit.

Fig. 8 shows the circuit model for low-capture-power test vector generation in the broadside clocking scheme shown in Fig. 1 for the general full-scan circuit shown in Fig. 7. Note that, the combinational logic in Fig. 7 is assumed to implement the logic function *f*.

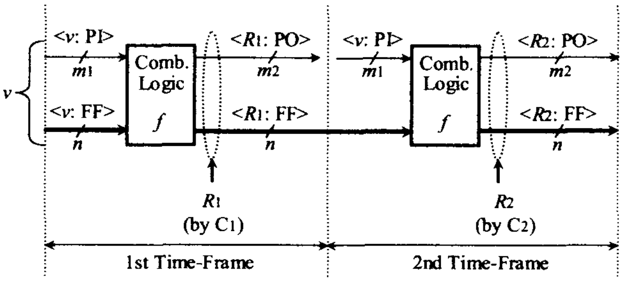


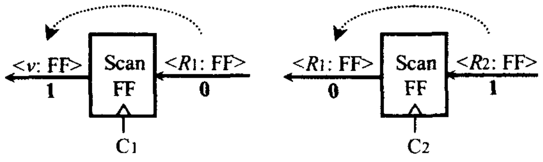
Fig. 8 Circuit Model for the Broadside Clocking Scheme.

In Fig. 8, *v* is the input vector in the first time-frame, which is provided from primary inputs and the scan FFs of a scan chain. That is, *v* consists of two parts: primary input bits denoted by <*v*: PI> and the FF-output bits denoted by <*v*: FF>. The functional response of the combinational logic to *v* is *f*(*v*), denoted by *R*1. For *R*1, its bits related to primary outputs are denoted by <*R*1: PO> and its bits related to FFs are denoted by <*R*1: FF>. When the first capture C1 is conducted, <*R*1: FF> is loaded into all FFs to replace <*v*: FF>, and <<*v*: PI>, <*R*1: FF>> becomes the input vector in the second time-frame. The functional response of the combinational logic to this input vector is *f*(<<*v*: PI>, <*R*1: FF>>), denoted by *R*2. For *R*2, its bits related to primary outputs are denoted by <*R*2: PO> and its bits related to FFs are denoted by <*R*2: FF>. When the second capture C2 is conducted, <*R*2: FF> is loaded into all FFs to replace <*R*1: FF>. Note that both *R*2 and *R*2 can be readily obtained through logic simulation.

Also note that the values for the primary inputs remain the same in both time-frames. This assumption is made since it is usually difficult and costly to change primary input values during the rated clock period between the first and second capture pulses in the broadside clocking scheme for a high-speed design.

3.3 DC-LCP *X*-Filling Problem Formalization

As shown in Fig. 8, <*v*: FF> is replaced by <*R*1: FF> when the first capture C1 is conducted, and <*R*1: FF> is replaced by <*R*2: FF> when the second capture C2 is conducted. Obviously, if <*v*: FF> is different from <*R*1: FF> at some scan FFs, *capture transitions* will occur at the outputs of these scan FFs for C1 as shown in Fig. 9 (a). Similarly, if <*R*1: FF> is different from <*R*2: FF> at some scan FFs, *capture transitions* will occur at the outputs of these scan FFs for C2 as shown in Fig. 9 (b).



(a) Capture C1 (b) Capture C2

Fig. 9 Capture Transitions at a Scan FF.

Capture transitions at FFs has a strong correlation with circuit switching activity [12], and thus capture test power dissipation. Therefore, capture power reduction can be achieved by reducing the number of capture transitions. Note that not all FFs carry the same weight regarding to power dissipation in practice. That is, capture transitions at some FFs may cause more power dissipation than other FFs. In this paper, it is assumed that all FFs have the same weight. The case where FFs have different weights will be considered in the future.

From Fig. 8 and Fig. 9, it is clear that capture transitions for C1 and C2 are caused by the difference between <*v*: FF> and <*R*1: FF> and the difference between <*R*1: FF> and <*R*2: FF>, respectively. Therefore, capture transitions for C1 and C2 can be reduced by making <*v*: FF> similar to <*R*1: FF> and <*R*1: FF> similar to <*R*2: FF> as much as possible. DC-LCP *X*-filling is used to achieve this goal by properly assigning 0's and 1's to the *X*-bits in *v*, which is a test cube with at least one *X*-bit.

The obvious requirement for DC-LCP *X*-filling is to reduce capture transitions for C1 and C2 as much as possible. In addition, the fact that two captures are involved makes it necessary to conduct capture transition reduction in a balanced manner with respect to C1 and C2. Therefore, the DC-LCP *X*-filling problem can be formalized as follows:

DC-LCP X-Filling Problem: Given a test cube v for a full-scan circuit with respect to the broadside clocking scheme as shown in Fig. 8, assign 0's and 1's to all X-bits in v so that $(N_1 + N_2)$ and $|N_1 - N_2|$ are both minimized, where N_1 and N_2 are the numbers of capture transitions for the first capture C_1 and the second capture C_2, respectively.

3.4 DC-LCP *X*-Filling Algorithm

In Fig. 8, suppose that x is one bit in <v: FF> with respect to a FF. Then, there must be one bit y in <R_1: FF> and one bit z in <R_2: FF>, both with respect to the same FF as x. <x, y, z> is called a *3-bit-tuple* in this paper. The circuit model in Fig. 8 has n 3-bit-tuples since there are n FFs in the full-scan circuit.

In addition, if v is a test cube with at least one X-bit, there must be some X-bits in 3-bit-tuples for the circuit. Depending on how X-bits appear, 3-bit-tuples can be classified into 8 *X-types* as summarized in Table 1.

Table 1 *X*-Types

Type	# of X's	<v: FF>	<R_1: FF>	<R_2: FF>	Target Capture
1	0	b_1	b_2	b_3	—
2	1	X	b_2	b_3	C_1
3		b_1	X	b_3	C_1, C_2
4		b_1	b_2	X	C_2
5	2	b_1	X	X	C_1, C_2
6		X	b_2	X	C_1, C_2
7		X	X	b_3	C_1, C_2
8	3	X	X	X	C_1, C_2

(b_1, b_2, b_3: 0 or 1)

Obviously, there is no need to consider any 3-bit-tuple of Type-1 in DC-LCP X-filling. A 3-bit-tuple of Type-2 through Type 8 has at least one X-bit and it can be used for capture transition reduction in DC-LCP X-filling.

Note that 3-bit-tuples of different types may reduce capture transitions for different captures. For example, a 3-bit-tuple of Type-2 in the form of <X, b_2, b_3> can only reduce capture transitions for the first capture C_1, and this is achieved if the X-bit can take logic value b_2. On the other hand, consider a 3-bit-tuple <b_1, X, b_3> of Type-3 with $b_1 \neq b_3$. This 3-bit-tuple can be used to reduce capture transitions for C_1 if X-bit takes logic value b_1 or for C_2 if X-bit takes logic value b_3. The type of information on what X-type can reduce capture transitions for what capture is also shown in the column "Target Capture" of Table 1.

In the following, the details of the DC-LCP X-filling algorithm are described, starting with the general procedure and an example in 3.4.1, followed by detailed discussions of the three key operations in 3.4.2 through 3.4.4.

3.4.1 General Procedure

Fig. 10 shows the general DC-LCP X-filling procedure.

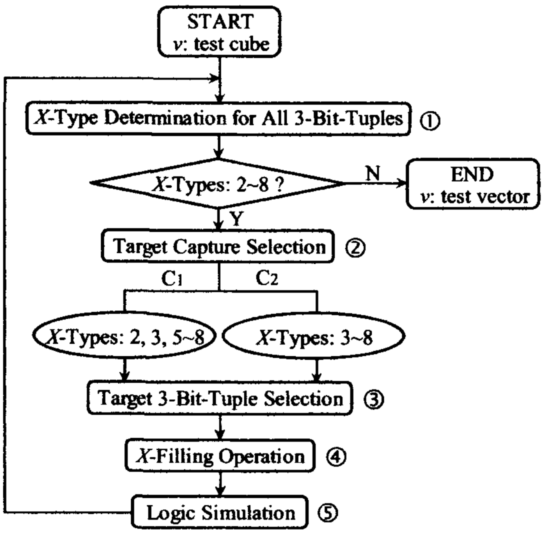


Fig. 10 DC-LCP *X*-Filling Procedure.

The input to the DC-LCP X-filling procedure is a test cube v with at least one X-bit, and the output is a fully-specified test vector. The procedure consists of the following steps:

(1) *X-Type Determination* is conducted to determine the X-types of all 3-bit-tuples. If only 3-bit-tuples of Type-1 exist, v is already a fully-specified test vector and the procedure ends.

(2) *Target Capture Selection* is conducted to determine which capture, C_1 or C_2, should be targeted in the current iteration of capture transition reduction. This is to guarantee that capture transitions for C_1 and C_2 are reduced in a balanced manner.

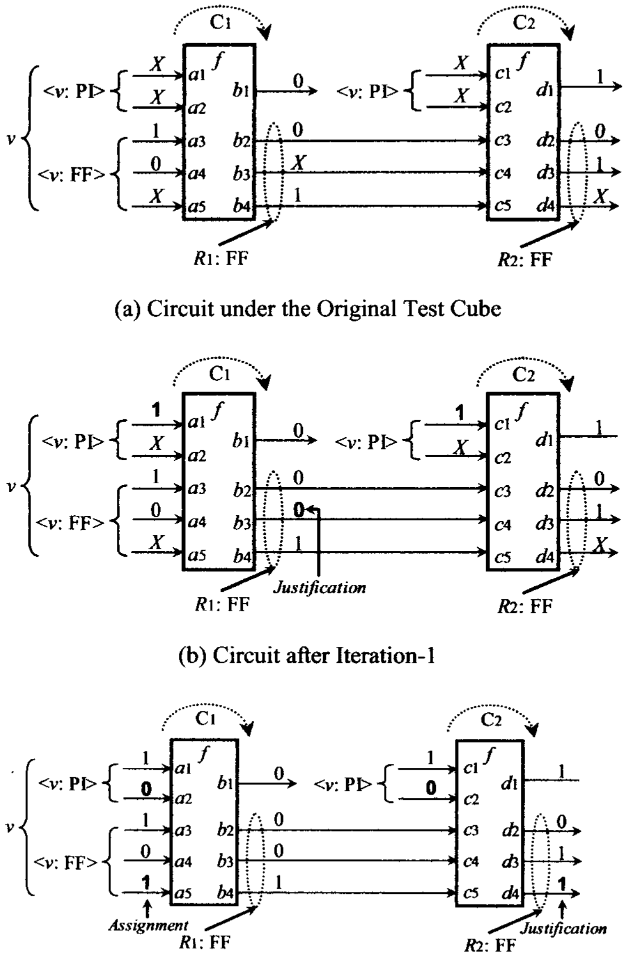
(3) *Target 3-Bit-Tuple Selection* is conducted to pick up one 3-bit-tuple that has at least one X-bit and that has the highest possibility of successfully reducing capture transitions for the capture determined at Step-1.

(4) *X-Filling Operation* uses assignment and justification techniques to find proper logic values for the X-bits in the test cube v so that necessary logic value(s) can appear at the X-bit(s) in the 3-bit-tuple selected at Step-2 in order to reduce capture transitions for the capture selected at Step-1.

(5) *Logic Simulation* is conducted to spread the effect of the newly determined logic values at X-bits in v to the whole circuit. Obviously, the X-types of some 3-bit-tuples may change because of this.

Clearly, the DC-LCP X-filling procedure shown in Fig. 10 handles one 3-bit-tuple in each iteration, and each iteration consists of the above 5 steps. For a circuit of n FFs as shown in Fig. 8, there are n 3-bit-tuples. That is, at most n iterations are needed in order to complete DC-LCP X-filling. Since each iteration mainly consists of justification and logic simulation operations, the run time of DC-LCP X-filling is strictly under control, making it feasible to be used in the proposed dynamic compaction procedure shown in Fig. 6 for large circuits.

An example of DC-LCP *X*-filling is shown in Fig. 11. The circuit under the original test cube *v* <*X*, *X*, 1, 0, *X*> is shown in Fig. 11 (a). For this test cube, there are three 3-bit-tuples: <1, 0, 0>, <0, *X*, 1>, and <*X*, 1, *X*>.



(a) Circuit under the Original Test Cube

(b) Circuit after Iteration-1

(c) Circuit after Iteration-2

Fig. 11 Example of DC-LCP *X*-Filling.

***Iteration*-1**:

As shown in Fig. 11 (a), there is one capture transition for C_1 but no capture transition for C_2 with respect to <1, 0, 0>. Capture transition information with respect to <0, *X*, 1> and <*X*, 1, *X*> is unclear since *X*-bits are involved. As a result, in order to achieve balanced capture transition reduction at this stage, it is necessary to reduce capture transitions for C_1. Although both <0, *X*, 1> and <*X*, 1, *X*> may serve this purpose, <0, *X*, 1> is selected since it involves only one *X*-bit, making it easier to bring 0 to the *X*-bit to reduce capture transitions for C_1.

Proper logic values needed for *X*-bits in *v* in order to bring logic 0 to the *X*-bit in <0, *X*, 1> are found by justifying 0 on b_3. The result is logic 1 for the *X*-bit on a_1 and c_1. As shown in Fig. 11 (b), the 3-bit-tuple <0, *X*, 1> becomes <0, 0, 1>, which causes no capture transition for C_1. □

***Iteration*-2**:

As shown in Fig. 11 (b), there is one capture transition for C_1 with respect to <1, 0, 0> and one capture transition for C_2 with respect to <0, 0, 1>. Capture transition information with respect to <*X*, 1, *X*> is unclear since *X*-bits are involved. As a result, it is necessary to reduce capture transitions for both C_1 and C_2. Here, <*X*, 1, *X*> is the only 3-bit-tuple to serve this purpose, requiring logic 1 to appear on both *X*-bits in <*X*, 1, *X*>.

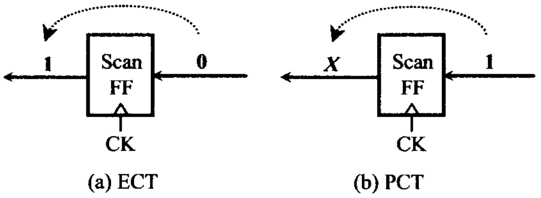
Proper logic values needed for *X*-bits in *v* in order to bring logic 1 to both *X*-bits in <*X*, 1, *X*> are found by assigning 1 to the *X*-bit on a_5 and justifying 1 on d_4. The result of justification is logic 0 for the *X*-bit on a_2 and c_2. As shown in Fig. 11 (c), the 3-bit-tuple <*X*, 1, *X*> becomes <1, 1, 1>, which causes no capture transition for both C_1 and C_2. □

As shown in Fig. 11, after two iterations of DC-LCP *X*-filling, the original test cube *v* <*X*, *X*, 1, 0, *X*> becomes a fully-specified test vector <**1**, **0**, 1, 0, **1**>.

In the following, details of three key operations in DC-LCP *X*-filling: *target capture selection*, *target 3-bit-tuple selection*, and *X-filling*, are described in 3.4.2 through 3.4.4.

3.4.2 Target Capture Selection

The DC-LCP *X*-filling method dynamically selects a target capture in order to achieve a balanced reduction of capture transitions for the first capture C_1 and for the second capture C_2. The target capture selection heuristic is based on the *total estimated capture transition activity* (*TECTA*), which is calculated from *existing capture transitions* (*ECTs*) and *potential capture transitions* (*PCTs*) as illustrated in Fig. 12.



(a) ECT

(b) PCT

Fig. 12 Existing and Potential Capture Transitions.

An ECT is a capture transition in the case where a logic value is loaded into a scan FF to replace a different logic value. An example of ECT is shown in Fig. 12 (a). On the other hand, a PCT is a capture transition in the case where a value V_2 is loaded into a scan FF to replace another value V_1, where either V_1 or V_2 or both are *X*-bits. An example of PCT is shown in Fig. 12 (b).

The probability of an ECT to occur is 100%; while the probability of a PCT to actually cause a real capture transition is 50% if it is simply assumed that all related *X*-bits in the PCT could take any logic value with equal probability. Based on this observation, *TECTA* for capture C_i (i = 1, 2), denoted by $TECTA_i$, can be calculated as follows:

$TECTA_i$ = (# of ECTs for C_i) + (0.5 × (# of PCTs for C_i))

Generally, the capture with a higher *TECTA* is selected as the target capture, since the number of capture transitions for this capture is likely to be greater than that for the other capture, and hence it needs to be reduced first. An example is shown in Fig. 13, which has four 3-bit-tuples. In this case, C_1 is selected since $TECTA_1 > TECTA_2$.

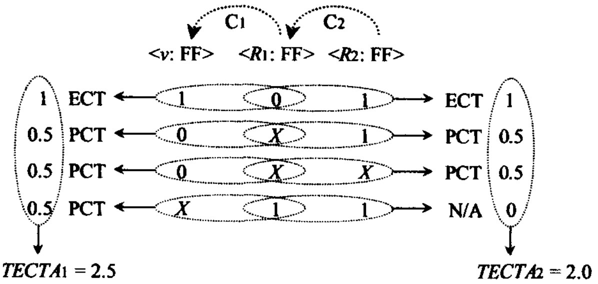


Fig. 13 Example of Target Capture Selection.

3.4.3 Target 3-Bit-Tuple Selection

Once a target capture is selected, it is necessary to further select a target 3-bit-tuple that has at least one *X*-bit and that has the highest possibility of successfully reducing capture transitions for the selected target capture.

As shown in the example of DC-LCP *X*-filling in Fig. 11, assignment and justification are used to determine logic values for *X*-bits in a test cube *v* to make required logic values appear at the *X*-bits in a 3-bit-tuple so that capture transitions are reduced. ***Assignment*** is to set a logic value to an *X*-bit in <*v*: FF> directly. Since any logic value can be loaded to any scan FF in shift mode for <*v*: FF>, assignment is simple and always successful. On the other hand, ***justification*** is to identify proper logic values for *X*-bits in *v* to make required logic values appear at the *X*-bits in <R_1: FF> or <R_2: FF>. Obviously, justification can be difficult and there is no guarantee that this operation is always successful.

As a result, in target 3-bit-tuple selection, we first select a 3-bit-tuple that only needs assignment in *X*-filling. Only when there is no such 3-bit-tuple, we select from 3-bit-tuples that need justification in *X*-filling, based on a heuristic measure. An example is shown in Fig. 14.

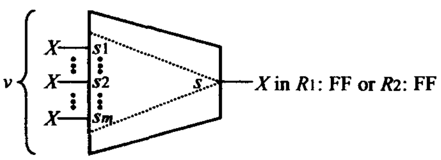


Fig. 14 *X*-Bit Justification.

In Fig. 14, there is one *X*-bit on signal line *s* on which justification is needed. Suppose that the level of *s* is *Ls*. Also suppose that *s* can reach *m X*-bit signal lines $s_1, s_2, \ldots, s_m$ corresponding to a test cube *v*, and that the levels of these signal lines are Ls_1, Ls_2, ..., Ls_m. Here, levels are assigned from the output side to the input side, and the highest level is denoted by *L*.

Conceptually, it is evident that if more *X*-bit signal lines are reachable from *s* and closer they are to *s*, then easier they are to justify a logic value on *s*. Based on this observation, the ***justification easiness*** (*JE*) of *s*, denoted by *JE*(*s*), is calculated as follows:

$$JE(s) = \sum_{i}^{m} \frac{(L - |Ls - Ls_i|)}{L}$$

Obviously, the larger the value of *JE*(*s*), the easier the justification of a logic value on *s*.

When it is necessary to select a 3-bit-tuple that needs justification, we first select from 3-bit-tuples with one *X*-bit in <R_1: FF> or <R_2: FF>. The *JE* value for the signal line with the *X*-bit is calculated, and the 3-bit-tuple of the largest *JE* value is selected. If there are only 3-bit-tuples that have two *X*-bits in <R_1: FF> and <R_2: FF>, the sum of the *JE* values for the signal lines with the *X*-bits is calculated, and the 3-bit-tuple with the largest sum of *JE* values is selected.

3.4.4 X-Filling Operation

After a target capture and a target 3-bit-tuple are selected, assignment and/or justification are conducted to determine logic values for *X*-bits in a test cube *v* in order to make required logic values appear at the *X*-bits in a 3-bit-tuple so that capture transitions are reduced.

Note that justification may fail. For example, for 3-bit-tuple <1, *X*, *X*>, the best choice is to make logic 1 appear at both the *X*-bits. This choice is tried first by justification. If it fails, we then try the next-to-best choice of making logic 1 appear at the first *X*-bit and logic 0 at the second *X*-bit. If this justification also fails, we then try to make logic 0 appear at both *X*-bits. If this justification also fails, the last choice is to make logic 0 appear at the first *X*-bit and logic 1 at the second *X*-bit.

3.5 Practical Issues

3.5.1 Handling of X-Sources

In practice, a circuit may contain such *X*-sources as analog blocks, memories, uninitialized FFs, multiple clock domains, floating bus, inaccurate simulation models, etc. These *X*-sources, as well as *X*-bits in a test cube, may result in some *X*-bits in the corresponding test response at the inputs of FFs.

Different from *X*-bits existing in a test cube, above-mentioned *X*-sources are uncontrollable in that it is impossible to set an *X*-source to any required logic value. As a result, in the DC-LCP *X*-filling procedure, if justifying a logic value at an *X*-bit in a test response ends up needing to set a specific value at an *X*-source as the only choice, the justification is considered unsuccessful.

3.5.2 Application to Unconventional Scan Schemes

Conventional scan scheme uses one external scan input pin and one external scan output pin for each internal scan chain. Recently, some unconventional scan schemes, such as OPMISR, VirtualScan, EDT, SoCBIST, etc., have been proposed for reducing test data volume and test application time. These scan schemes can be divided into two groups: *X-independent* (OPMISR and VirtualScan) and *X-dependent* (EDT and SoCBIST) according to if its fault detection capability depends on the use of *X*-bits in a test cube. Obviously, the DC LCP *X*-filling method readily works with any *X*-independent scan scheme.

As for *X*-dependent scan schemes, an interactive approach may be needed. That is, *X*-bits are first utilized to guarantee the minimum fault detection capability. The remaining *X*-bits are then used for detecting more faults or reducing capture test power with the DC LCP *X*-filling method, as long as the resulting test cube can be successfully compressed. Test power analysis may also need to be conducted in order to determine which type of reduction should be targeted with *X*-bits: test size or test power.

4. Experimental Results

X-filling experiments were conducted on ISCAS'89 circuits with an internally developed ATPG program for transition delay faults. Table 2 shows the circuit statistics and *X*-bit information in initial test cubes. An initial test cube is generated for detecting a primary target fault, and its *X*-bits are used in dynamic compaction for detecting secondary target faults and reducing capture power dissipation.

Table 2 Circuit Statistics and *X*-Bit Information

Circuit	# of PIs	# of FFs	# of Faults	*X*-Bits in initial test cubes	
				Ave. *X*-Bits	Max. *X*-Bits
s1423	17	74	2290	63 (69.7%)	87 (95.6%)
s5378	35	179	5980	173 (80.6%)	201 (93.9%)
s9234	19	228	10712	214 (86.4%)	244 (98.8%)
s13207	31	669	15440	658 (93.1%)	699 (99.9%)
s15850	14	597	18324	523 (85.6%)	605 (99.0%)
s35932	35	1728	54044	1485 (84.2%)	1758 (99.7%)
s38417	28	1636	49342	1420 (85.4%)	1648 (99.0%)
s38584	12	1452	55706	1294 (88.4%)	1455 (99.4%)

Table 3 shows the results of random *X*-filling in conventional dynamic compaction as shown in Fig. 8. The number of test vectors and fault coverage are shown under "*# of Vec.*" and "*Fault Cov.*". In addition, the average number of capture transitions per test vector and the maximum number of capture transitions for each circuit for the first and second captures are shown under "*# of Vec.*", "*Fault Cov.*", "*Ave. Trans.*", and "*Max. Trans.*", respectively.

In order to conduct DC-LCP *X*-filling in the proposed dynamic compaction as shown in Fig. 9, it is necessary to set an *X-Limit* for defining the percentage of original *X*-bits in an initial test cube to be allowed for detecting secondary target faults. When this *X-Limit* is reached, dynamic compaction switches immediately to DC-LCP *X*-filling for the remaining *X*-bits to reduce capture power dissipation. Generally, the smaller the *X-Limit*, the more test vectors will be generated since fewer secondary target faults can be detected. However, the smaller the *X-Limit*, the higher the capture transition reduction effect of DC-LCP *X*-filling will be since more *X*-bits are available for this purpose.

Table 3 Results for Random *X*-Filling

Circuit	# of Vec.	Fault Cov. (%)	1st Capture		2nd Capture	
			Ave. Trans.	Max. Trans.	Ave. Trans.	Max. Trans.
s1423	112	86.1	24	50	15	32
s5378	214	88.9	89	106	49	72
s9234	342	81.5	73	107	44	76
s13207	330	79.8	264	324	196	281
s15850	187	69.9	171	251	126	182
s35932	45	84.7	897	991	809	964
s38417	221	98.0	504	668	407	506
s38584	410	82.6	437	813	331	766

Extensive experiments on ISCAS'89 circuits have revealed an interesting fact that the number of test vectors will not grow too much if *X-Limit* is greater than a certain value, which can be as small as 20%. Fig. 17 shows partial experimental results on three largest ISCAS'89. This fact is very useful in achieving a good capture transition reduction effect while keeping a test vector set compact.

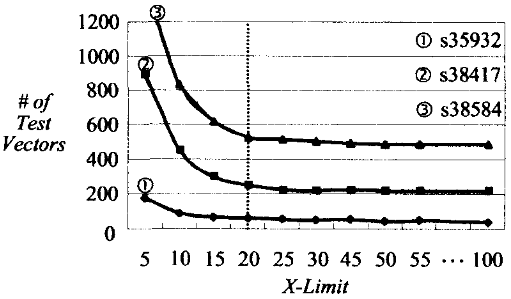


Fig. 17 Impact of *X-Limit*.

Table 4 shows the results of DC-LCP *X*-filling in the proposed dynamic compaction flow (*X*-Limit = 20%) as shown in Fig. 9. The meanings of the columns in Table 4 are the same as Table 3, except that Table 4 also shows CPU time.

Comparing the experimental results for random *X*-filling in Table 3 and for DC-LCP *X*-filling in Table 4, it can be seen that on average, DC-LCP *X*-filling achieved **52.4%** and **41.5%** reduction on average and maximum capture transition for the first capture, respectively, and **39.7%** and **24.6%** reduction on average and maximum capture transition for the second capture, respectively, in a balanced manner and without any fault coverage degradation. The cost was a slightly larger test vector set. It was possible to keep the number of test vectors unchanged by increasing *X-Limit*, which led to roughly 1/4 lower reduction effect.

Table 4 Results for DC-LCP *X*-Filling

Circuit	# of Vec.	Fault Cov. (%)	1st Capture		2nd Capture		CPU Time (sec.)
			Ave. Trans.	Max. Trans.	Ave. Trans.	Max. Trans.	
s1423	135	86.1	17	32	9	27	1
s5378	248	78.8	42	63	32	57	7
s9234	350	81.5	38	71	37	74	88
s13207	356	79.8	138	202	144	250	66
s15850	220	69.9	56	119	63	116	236
s35932	72	84.7	295	714	297	716	146
s38417	227	98.0	273	421	263	404	149
s38584	444	82.6	165	269	151	274	1427

5. Conclusions

This paper addressed a new test power reduction problem, i.e. reducing capture power dissipation to lower the risk of yield loss caused by faulted test responses in capture mode for scan-based at-speed testing with the broadside clocking scheme. A novel double-capture low-capture-power (DC-LCP) *X*-filling method has been proposed for algorithmically assigning 0's and 1's to *X*-bits in a test cube in order to reduce the switching activity at FFs for the resulting fully-specified test vector. Experimental results have shown the effectiveness of the method, which can be easily incorporated into any test generation flow to achieve capture power reduction without any area, timing, or fault coverage impact in reasonably short CPU time.

Further evaluation is in progress to assess the effect of DC-LCP *X*-filling directly through power consumption instead of switching activity at FFs although it is evident that they have a strong correlation. Research for an algorithmic method to determine a proper value for *X-Limit* in order to balance test set size reduction and capture power reduction in dynamic compaction is also under way.

References

[1] S. Savir and S. Patil, "On Broad-Side Delay Test," *Proc. VLSI Test Symp.*, pp. 284-290, 1994.

[2] X. Lin, R. Press, J. Rajski, P. Reuter, T. Rinderknecht, B. Swanson, and N. Tamarapalli, "High-Frequency, At-Speed Scan Testing," *IEEE Design & Test of Computers*, pp. 17-25, September-October, 2003.

[3] J. Gatej, L. Song, C. Pyron, R. Raina, and T. Munns, "Evaluating ATE Features in Terms of Test Escape Rates and Other Cost of Test Culprits," *Proc. Intl. Test Conf.*, pp. 1040-1048, 2002.

[4] P. Nigh, D. Vallett, A. Patel, J. Wright, F. Motika, D. Forlenza, R. Kurtulik, and W. Chong, "Failure Analysis of Timing and IDDq-only Failures from the SEMATECH Test Methods Experiment ," *Proc. Intl. Test Conf.*, pp. 43-52, 1998.

[5] N. Tendolkar, R. Raina, R. Woltenberg, X. Lin, B. Swanson, and G. Aldrich, "Novel Techniques for Achieving High At-Speed Transition Fault Test Coverage for Motorola's Microprocessors Based on PowerPC™ Instruction Set Architecture," *Proc. VLSI Test Symp.*, pp. 3-8, 2002.

[6] P. Girard, "Survey of Low-Power Testing of VLSI Circuits," *IEEE Design & Test of Computers*, vol. 19, no. 3, pp. 82-92, May/June 2002.

[7] N. Nicolici and B. Al-Hashimi, *Power-Constrained Testing of VLSI Circuits*, Kluwer Academic Publishers, 2003.

[8] Y. Zorian, "A Distributed BIST Control Scheme for Complex VLSI Devices," *Proc. VLSI Test Symp.*, pp. 4-9, 1993.

[9] R. Chou, K. Saluja, and V. Agrawal, "Scheduling Tests for VLSI Systems under Power Constraints," *IEEE Trans. on VLSI Systems*, vol. 5, no. 6, pp. 175-185, 1997.

[10] S. Wang and S. Gupta, "ATPG for Heat Dissipation Minimization during Test Application," *IEEE Trans. on Computers*, vol. 47, no. 2, pp. 256-262, 1998.

[11] F. Corno, P. Prinetto, M. Redaudengo, and M. Reorda, "A Test Pattern Generation Methodology for Low Power Consumption," *Proc. VLSI Test Symp.*, pp. 35-40, 2000.

[12] R. Sankaralingam, R. Oruganti, and N. Touba, "Static Compaction Techniques to Control Scan Vector Power Dissipation," *Proc. VLSI Test Symp.*, pp. 35-40, 2000.

[13] S. Kajihara, K. Ishida, and K. Miyase, "Test Vector Modification for Power Reduction during Scan Testing," *Proc. VLSI Test Symp.*, pp. 160-165, 2002.

[14] V. Dabholkar, S. Chakravarty, I. Pomeranz, and S. Reddy, "Techniques for Minimizing Power Dissipation in Scan and Combinational Circuits during Test Application," *IEEE Trans. on Computer-Aided Design*, vol. 17, no. 12, pp. 1325-1333, 1998.

[15] A. Chandra and K. Chakrabarty, "Combining Low Power Scan testing and Test Data Compression for System-on-a-Chip," *Proc. Design Automation Conf.*, pp. 166-169, 2001.

[16] A. Chandra and K. Chakrabarty, "Reduction of SoC Test Data Volume, Scan Power and Testing Time Using Alternating Run-Length Codes," *Proc. Design Automation Conf.*, pp. 673-678, 2002.

[17] A. Hertwig and H. Wunderlich, "Low Power Serial Built-In Self-Test," *Proc. European Test Workshop*, pp. 49-53, 1998.

[18] R. Sankaralingam, R. Oruganti, and N. Touba, "Reducing Power Dissipation during Test Using Scan Chain Disable," *Proc. VLSI Test Symp.*, pp. 319-324, 2001.

[19] T. Yoshida and M. Watari, "MD-Scan Method for Low Power Scan Testing," *Proc. Intl. Test Conf.*, pp. 480-487, 2003.

[20] Y. Bonhomme, P. Girard, C. Landrault, and S. Pravossoudovitch "Power Driven Chaining of Flip-Flops in Scan Architectures," *Proc. Intl. Test Conf.*, pp. 796-803, 2002.

[21] J. Saxena, K. Butler, and L. Whetsel, "A Scheme to Reduce Power Consumption during Scan Testing," *Proc. Intl. Test Conf.*, pp. 670-677, 2001.

[22] O. Sinanoglu and A. Orailoglu, "Scan Power Minimization through Stimulus and Response Transformations," *Proc. Design, Automation and Test in Europe*, pp. 404-409, 2004.

[23] S. Gerstendoerfer and H. Wunderlich, "Minimized Power Consumption for Scan-Based BIST," *Proc. Intl. Test Conf.*, pp. 77-84, 1999.

[24] S. Wang, "Generation of Low-Power-Dissipation and High-Fault Coverage Patterns for Scan-Based BIST," *Proc. Intl. Test Conf.*, pp. 834-843, 2002.

[25] J. Saxena, K. M. Butler, V. B. Jayaram, and S. Kundu, "A Case Study of IR-Drop in Structured At-Speed Testing," *Proc. Intl. Test Conf.*, pp. 1098-1104, 2003.

[26] K. Lee, T. Huang, and J. Chen, "Peak-Power Reduction for Multiple-Scan Circuits during Test Application," *Proc. Asian Test Symp.*, pp. 435-440, 2000.

[27] R. Sankaralingam and N. A. Touba, "Controlling Peak Power During Scan Testing," *Proc. VLSI Test Symp.*, pp. 153-159, 2002.

[28] X. Wen, H. Yamashita, S. Kajihara, L.-T. Wang, K. Saluja, and K. Kinoshita, "On Low-Capture-Power Test Generation for Scan Testing," *Proc. VLSI Test Symp.*, pp. 265-270, 2005.

5. 3. 全素子における状態遷移を考慮したXビット埋め込み手法

論理回路内の状態遷移量は、記憶素子（フリップ・フロップ）における状態数のみでは精確に評価することができない場合がある。フリップ・フロップのみでもなく組合せ回路部の論理ゲートのみでもなく、両方における状態変化数を最小化することによって、キャプチャ時消費電力を削減する手法を本研究で開発し、その成果を次の国際会議で発表した。ここではその内容を紹介する。

X. Wen, K. Miyase, T. Suzuki, Y. Yamato, S. Kajihara, L.-T. Wang, K. K. Saluja:
A Highly-Guided X-Filling Method
for Effective Low-Capture-Power Scan Test Generation
Proc. IEEE Int'l Conf. on Computer Design, pp. 251-258 (2006)

6. 低消費電力テスト生成技術

テスト時の消費電力を削減するために、Xビットへの論理値埋め込みという後処理のみではなく、最初から最適な論理値を生成することも必要である。このため、低消費電力テスト生成に関する研究開発が求められている。

テストキューブは通常論理値ビットとXビットによって構成される。Xビットに埋め込む論理値はもちろん、最初から存在する論理値も最終テストベクトルの電力消費量に大きい影響を与える。このため、テスト消費電力を考慮してテストキューブの中の論理値を生成する必要がある。そこで本研究では、検出矛盾（Detection Conflict）の他にキャプチャ（Capture Conflict）を導入し、フリップ・フロップにおける状態変化数が最小になるような論理値を決定する手法を開発した。また、この手法とXビット埋め込み手法の併用によって高いテスト消費電力削減効果が得られることを実験で示した。その研究成果を６.１で紹介する。

一般に、与えられたテスト集合の中で、高い消費電力を有するテストは一部に過ぎず、これのテストに対してテスト消費電力を考慮してテスト再生成を行なう必要がある。そのため、このよう高消費電力テストでしか検出できない故障（必須故障）を特定し、すべての必須故障を検出するように新しいテストパターンを生成する必要があるが、テストベクトル数が増えるという問題点がある。そこで本研究では、必須故障の処理順序を制約し、テストベクトルの増加を抑える手法を開発し、その有効性を実験で示した。その研究成果を６.２で紹介する。

6．1．低消費電力テスト生成のためのATPG手法

テストキューブ中の論理値は最終テストベクトルの電力消費量に大きい影響を与える。検出矛盾（Detection Conflict）の他にキャプチャ（Capture Conflict）を導入し、フリップ・フロップにおける状態変化数が最小になるような論理値を決定する手法を本研究で開発し、その成果を次の国際会議で発表した。ここではその内容を紹介する。

X. Wen, S. Kajihara, K. Miyase, T. Suzuki, K. K. Saluja, L.-T. Wang,
K. S. Abdel-Hafez, and K. Kinoshita:
A New ATPG Method for Efficient Capture Power Reduction During Scan Testing
Proc. IEEE VLSI Test Symp., pp. 58-63（2006）

6. 2. 低消費電力テストのための制約付きテスト生成手法

与えられたテスト集合の中の高い消費電力を有するテストに対して、そのテストでしか検出できない故障（必須故障）を対象にテスト再生成を行なう必要がある。必須故障の処理順序を制約し、テストベクトルの増加を抑える手法を本研究で開発し、その成果を次の研究報告で発表した。ここではその内容を紹介する。

塔ノ上義章，温暁青，梶原誠司，宮瀬紘平，鈴木達也，大和勇太：
低消費電力テストのための制約付テスト生成手法について
電子情報通信学会技術研究報告，Vol.106，No.467，pp.109-114（2007）

7. 結論

本研究では、テスト時消費電力増大による誤テストを回避するために必要な技術群を開発した。基本技術としては、超微細化に伴う中間故障電圧現象、および、超大規模化に伴う処理時間の膨大化といった問題の解決方法を提案した。応用技術としてはまず、すべてのクロックドメインの中の故障、および、クロックドメイン間の故障を検出する実速度テスト用の制御方式と回路設計を開発した。また、論理値のみで構成される初期テスト集合から再び故障検出率と無関係なＸビットを特定する手法を開発した。更に、様々なＸビット埋め込み手法と低消費電力テスト生成手法を開発した。

現在、LSI回路の超大規模化・超微細化に伴い、歩留まりは低下する一方である。特に、テスト時の消費電力が増大する傾向にあるため、テスト中のLSI回路を誤動作させるという誤テストによる歩留まり低下問題が年々深刻になってきている。このため、本研究で開発された誤テスト回避型テスト技術は、LSIの技術的・産業的な発展のみならず、LSIを基礎とする情報化社会にとっても極めて重要である。

今後は、以下の研究課題について研究を行なう予定である。

（１）クリチカルパスを考慮した誤テスト回避型テスト技術

（２）微小遅延検出能力を保持したＸビット判定技術

（３）Ｘビット埋め込み処理の高速化技法

（４）信号品質を考慮したテストパターン生成技術

（５）圧縮スキャンテスト環境における低消費電力テスト生成技術

（６）消費電力を意識したテスト生成フローの確立

謝辞

本研究を進めるにあたり、九州工業大学情報工学部　梶原誠司教授、宮瀬紘平助教には多大なご協力を頂いた。また、SynTest Technologies, Inc. L.-T. Wang氏には貴重なコメントを頂いた。さらに本研究成果となっている各論文の共著者の方々には並々ならぬ協力を頂いた。ここに深く謝意を表す。