A GA-Based Method for High-Quality *X*-Filling to Reduce Launch Switching Activity in At-Speed Scan Testing

Yuta Yamato[1], Xiaoqing Wen[1,2], Kohei Miyase[1,2], Hiroshi Furukawa[1], and Seiji Kajihara[1,2]

1: Kyushu Institute of Technology, Iizuka 820-8502, Japan
2: JST CREST

Abstract

Power-aware X-filling is a preferable approach to avoiding IR-drop-induced yield loss in at-speed scan testing. However, the quality of previous X-filling methods for reducing launch switching activity may be unsatisfactory, due to low effect (insufficient and global-only reduction) and/or low scalability (long CPU time). This paper addresses this quality problem with a novel, GA (Genetic Algorithm) based X-filling method, called GA-fill. Its goals are (1) to achieve both effectiveness and scalability in a more balanced manner, and (2) to make the reduction effect of launch switching activity more concentrated on critical areas that have higher impact on IR-drop-induced yield loss. Evaluation experiments are being conducted on benchmark and industrial circuits, and initial results have demonstrated the usefulness of GA-fill.

Keywords: *X*-filling, GA, Launch Switching Activity, IR-Drop, At-Speed Scan Testing.

1. Introduction

With shrinking feature sizes, growing clock frequency, and decreasing power supply voltage, modern integrated circuits are increasingly suffering from the impact of timing related defects, usually of small delays [1]. As a result, delay testing has become mandatory in order to reduce the defect level of deep submicron (DSM) circuits.

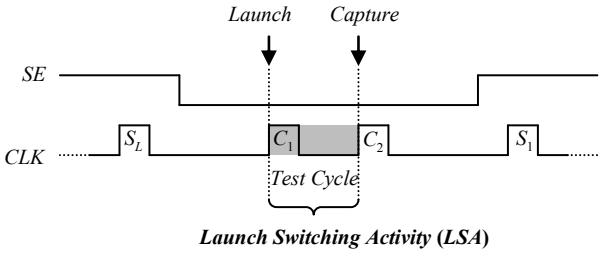

Fig. 1 Launch-on-Capture Scheme.

Since most of integrated circuits are based on scan design, ***at-speed scan testing*** is widely used for delay testing in practice [2]. Fig. 1 shows the timing diagram of at-speed scan testing based on the ***launch-on-capture*** (***LOC***) scheme [2]. In ***shift*** mode (*SE* = 1), a test vector is applied by operating scan flip-flops (FFs) as one or more shift registers over multiple shift clock pulses, with S_L as the last shift clock pulse. In ***capture*** mode (*SE* = 0), two capture pulses are applied: C_1 (***launch capture***) for launching a transition at the start-point of a path and C_2 (***response capture***) for capturing the circuit response to the launched transition at the end-point of the path. Note that the ***test cycle*** is equal to the functional clock cycle, which is very short for a high-speed design, in at-speed scan testing.

Switching activity due to the launch capture C_1 for a test vector, referred to as ***launch switching activity*** (***LSA***) in this paper, is usually higher (5x higher as reported in [3]) than that of a functional vector. This can lead to excessive ***IR-drop*** in the power distribution network, which may increase gate delays along paths in the test cycle to such an extent that timing violations may occur at the response capture C_2 only during at-speed scan testing. As a result, test-induced yield loss may occur, which is costly and unacceptable [4].

The IR-drop-induced yield loss problem is especially severe for high-speed circuits, whose test cycles are short. In addition, low-power circuits are also highly vulnerable to excessive launch switching activity due to narrowed noise margins. Therefore, there is a strong need to reduce launch switching activity during at-speed scan testing.

Previous techniques for reducing launch switching activity can be classified into two gross categories: ***circuit modification*** and ***data manipulation***. Circuit-modification-based techniques include *additional circuitry insertion* [5, 6], *scan chain segmentation* [7], *scan re-ordering* [8], and *partial capturing* [9]. On the other hand, data-manipulation-based techniques include *test vector reordering* [10], *test generation* [11, 12], and *X-filling* [13-18]. Generally, the data-manipulation approach is preferable, since it does not introduce any circuit overhead and performance degradation that are inevitable with the circuit-modification approach.

Especially, ***X-filling*** techniques are now widely used in practice for reducing launch switching activity, since they are compatible with any existing ATPG flow, and have no impact on test data volume, circuit overhead, and circuit timing [19]. *X*-filling is the process of properly assigning logic values to the *X*-bits in a partially-specified test cube for reducing launch switching activity in at-speed scan testing. This is

realized by minimizing either the Hamming distance between a test vector and its response at flip-flops [13-16] or the ***weighted switching activity*** (***WSA***) for all nodes in the whole circuit [17, 18].

However, previous *X*-filling methods for reducing launch switching activity in at-speed scan testing suffer from a severe quality problem, in terms of ***low effect*** (insufficient and global-only reduction) and ***low scalability*** (long CPU time), as described bellow:

(1) ***Low Effect***

This issue manifests itself in two ways: First, some *X*-filling methods cannot achieve sufficient effect in reducing launch switching activity. This is usually the case when high scalability of *X*-filling is required. Second, most of *X*-filling methods can only reduce launch switching activity globally in a gross manner. That is, the effect of *X*-filling cannot be concentrated on the critical areas that have higher impact on IR-drop-induced yield loss. Obviously, this issue of low effect severely limits the actual usefulness of *X*-filling in practice.

(2) ***Low Scalability***

Many *X*-filling methods are computation-extensive, and thus time-consuming. This is usually the case when high effectiveness of *X*-filling is required. Obviously, this issue of low scalability makes it impractical to apply such *X*-filling methods to large industrial circuits.

Clearly, there is a strong need to improve the quality of *X*-filling for reducing launch switching activity. This is important for effectively and efficiently avoiding test-induced yield loss in at-speed scan testing.

This paper addresses this quality issue of *X*-filling with a novel, GA (Genetic Algorithm) based *X*-filling method, called ***GA-fill***. Its goals are (1) to achieve both effectiveness and scalability in a more balanced manner, and (2) to make the reduction effect of launch switching activity more concentrated on critical areas that have higher impact on IR-drop-induced yield loss.

This paper is organized as follows: Section 2 describes some typical *X*-filling methods to highlight the quality problem in *X*-filling. Section 3 proposes the GA-fill as a solution to the problem. Section 4 shows experimental results, and Section 5 concludes the paper.

2. Background

X-filling for reducing launch switching activity in at-speed scan testing is conducted by assigning logic values to the *X*-bits in a partially-specified test cube, in such a manner that the Hamming distance between a test vector and its response at FFs [13-16], or the weighted switching activity for all nodes in the whole circuit [17, 18], are reduced. There are three basic approaches to *X*-filling, namely ***justification-based***, ***probability- based***, and ***hybrid***.

LCP (***Low-Capture-Power***) ***X-filling*** [14] is a typical justification-based technique. PI-PPO bit-pairs of the form <*X*, *val*> are processed first (*val* is a logic value), by assigning *val* to the PPI *X*-bit. After that, PPI-PPO bit-pairs of the form <*val*, *X*> are processed, one at a time, by justifying *val* to the PPO *X*-bit. Finally, PPI-PPO bit-pairs of the form <*X*, *X*> are processed, which may require two passes of justification for each such pair. That is, 1 is first assigned to the PPI *X*-bit and 1 is justified on the PPO *X*-bit. If this fails, 0 is then tried for the PPI and PPO *X*-bits. LCP *X*-filling is highly effective. However, it is less scalable due to long CPU time, since only one PPI-PPO bit-pair is processed at a time and justification is conducted.

Preferred Fill [15] and ***PWT*** (***Probability Weighted Capture Transition***) ***Fill*** [17] are typical probability-based techniques. The 0 and 1 probabilities of each node are first calculated by setting 0.50 as the 0 and 1 probabilities at input *X*-bits and conducting probability propagation. Then, the logic value for a PPI *X*-bit is determined by comparing the 0 and 1 probabilities of its corresponding PPO *X*-bit. In Preferred Fill [12], once 0 and 1 probabilities are calculated, all PPI *X*-bits in a test cube are assigned at one time in order to achieve high scalability. On the other hand, in PWT Fill [14], 0 and 1 probabilities are iteratively calculated after each new *X*-bit assignment so as to achieve high effectiveness. However, the effect of probability-based techniques may be damaged by inaccurate probability results due to approximation in probability calculation. In addition, if the difference between the 0 and 1 probabilities at a PPO *X*-bit is negligibly small, the effect of logic value determination for its corresponding PPI *X*-bit becomes highly unpredictable.

JP Fill [13] is a typical hybrid technique employing two basic operations, namely *limited justification* and *probability recalculation*. The first operation is conducted for PPI-PPO bit-pairs of the form <logic value, *X*>, but not for any PPI-PPO bit-pair of the form <*X*, *X*> for which time-consuming, multiple passes of justification are needed. This is to achieve higher scalability. The second operation is conducted for PPI-PPO bit-pairs of the form <*X*, *X*>, but in multiple passes. That is, the logic value for a PPI *X*-bit is determined only if its corresponding PPO *X*-bit has significantly different 0 and 1 probabilities; otherwise, signal probabilities are re-calculated in the next pass. This is to achieve higher effectiveness through improved accuracy in logic value determination.

Recently, *X*-filling methods have been proposed to reduce both shift and capture power [20, 21], and some can reduce both shift-in and shift-out power [20]. It should also be noted that test cubes in *X*-filling can be either obtained by disabling random-fill in ATPG or identified after ATPG from fully-specified test vectors by *test relaxation* or *X-identification* (*XID*) [22].

However, previous *X*-filling methods for reducing launch switching activity may suffer from a serious issue, i.e. *low quality*. The major reason is that these *X*-filling methods try to directly determine logic values for PPI *X*-bits, which may cause problems in terms of *low effect* and *low scalability*, as described bellow:

(1) Some *X*-filling methods try to improve the effectiveness by determining logic values for one PPI *X*-bit at a time, while others try to improve the scalability by determining logic values for multiple PPI *X*-bits simultaneously. This makes it difficult to achieve both effectiveness and scalability in a balanced manner. This problem manifests itself as either insufficient reduction effect or prohibitively long CPU time, leading to the low quality of *X*-filling.

(2) Furthermore, the direct effect of previous *X*-filling methods is the reduction of launch switching activity at FFs or in the whole circuit, instead of in any specific area. That is, their reduction effect is global and gross. However, it has been shown that high IR-drop around activated critical paths is the major reason for test-induced yield loss [23, 24]. This means that the LSA reduction effect achieved by previous *X*-filling methods may not be what is actually needed from the view-point of avoiding yield loss. Obviously, this problem also leads to the low quality of *X*-filling.

Note that the low quality problem of previous *X*-filling methods for reducing launch switching activity is caused by the fact that direct logic value determination is conducted for PPI *X*-bits. This motivates us to solve the problem by guiding *X*-filling in a manner that is not restricted by PPI *X*-bits. For this purpose, we choose to use the Genetic Algorithm (GA) approach [25], and propose a novel *X*-filling method, called ***GA-fill***. Its advantage is that a flexible evaluation function, instead of restrictive PPI *X*-bits, is used to guide *X*-filling.

Since an evaluation function is free from the restrictions caused by PPI *X*-bits, this makes it possible to achieve both effectiveness and scalability in a more balanced manner. In addition, an evaluation function can be made to represent the regional launch switching activity in a critical area that has higher impact on IR-drop-induced yield loss. This makes it possible to achieve better actual effect of launch switching activity reduction through such fine-grained *X*-filling.

3. GA-Based *X*-Filling

3.1 Basics

GA is a search technique that is used to find exact or approximate solutions to optimization and search problems [25, 26]. A GA-based search process starts with a set (***population***) of solutions (***individuals***), represented by ***chromosomes***. ***Fitness*** is calculated for each individual in the population. Individuals are taken and used to breed a new population according to their fitness, measured by a certain evaluation function. This process is repeated until a preset termination condition, e.g. the number of total populations or the amount of improvement of the best individual, is satisfied.

Obviously, the GA approach is ideal to solve the problem of finding optimal logic values assignments in *X*-filling for reducing launch switching activity in at-speed scan testing. This is because finding a good solution for *X*-filling needs to search through an extremely large solution space. In addition, this search process is guided by an evaluation function, which is free of restrictions caused by PPI *X*-bits and can qualitatively express any reduction goal, such as the launch switching activity in a critical area. As a result, the GA approach has the potential of improving the quality of *X*-filling for reducing launch switching activity in at-speed scan testing.

3.2 General Flow

The general flow of the proposed GA-based *X*-filling method, called ***GA-fill***, is shown in Fig. 2. It consists of 5 basic operations as described in the following.

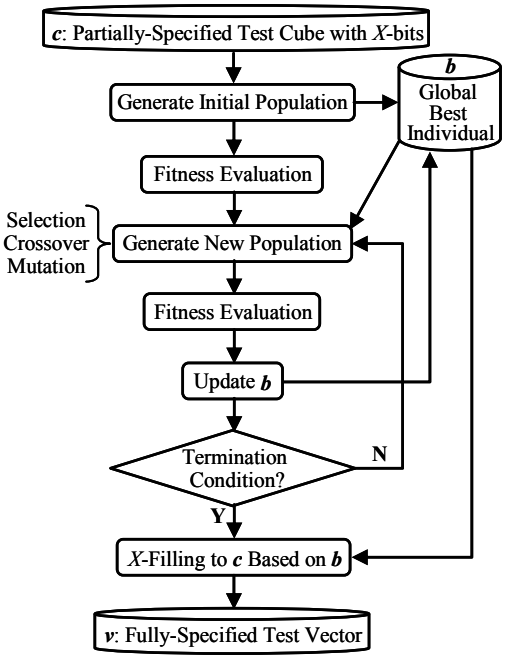

Fig. 2 General Flow of GA-Fill.

(1) ***Initialization***
At the beginning of GA-fill, a certain number of random assignments are made for all *X*-bits in a partially-specified test cube. This results in initial individuals that are used to form the initial population. Fig. 3 (a) shows an example, in which the test cube has 9 *X*-bits, and *n* initial individuals are created.

(2) ***Evaluation***
The fitness of each individual is evaluated in GA-fill. Here, pattern-parallel 2-valued logic simulation is used to collect information for multiple individuals

simultaneously in order to reduce computation time. Then, a proper evaluation function is calculated to quantify the fitness of each individual. Evaluation functions will be discussed in Subsection 3.3.

(3) ***Selection***
A new individual is created from two parent individuals, which are selected based on a certain criterion. Generally, fitter individuals are made more likely to be selected. This can be achieved by using such selection methods as the Roulette wheel selection. An example is shown in Fig. 3 (b), where an individual with higher fitness is assigned with a larger area, making it more likely to be selected.

(4) ***Crossover***
From two parent individuals, one child individual can be created by crossing-over parts of the chromosomes in the parent individuals. An example of two-point crossover is shown in Fig. 3 (c). Obviously, a child individual created this way typically shares many of the characteristics of its parent individuals.

(5) ***Mutation***
After a child individual is created, some of its chromosomes are flipped with low probability in order to promote the diversity of individuals. An example is shown in Fig. 3 (d).

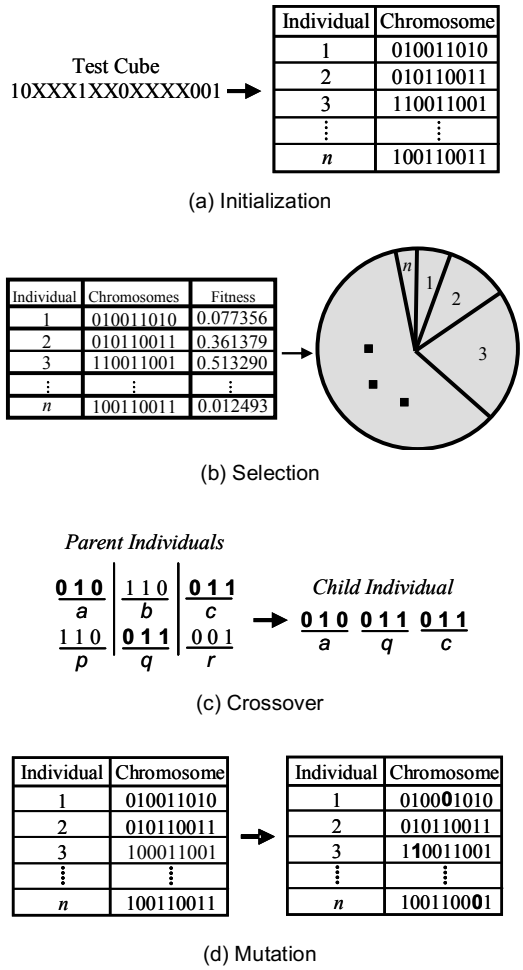

Fig. 3 Examples of Basic Operations in GA-Fill.

As shown in Fig. 2, GA-fill keeps generating new populations and evaluating new individuals to improve the quality of the global best individual. Note that an individual with the highest fitness throughout the whole population is kept. This process is repeated until a certain termination condition is satisfied. At this time, all *X*-bits in the test cube are assigned with the logic values corresponding to the global best individual.

3.3 Fitness Evaluation

In the following, we propose two sets of evaluation functions for GA-fill: one for achieving effectiveness and scalability in a balanced manner, and the other for achieving concentrated reduction effect of launch switching activity in a given critical area.

3.3.1 Balanced Effectiveness and Scalability

Basically, GA-fill uses the ***WSA*** (***Weighted Switching Activity***) metric [8, 15] to estimate the launch switching activity. Here, the weight of a node is set to be the number of its fanouts plus 1. If a node is fanout-free, its weight is set to 1.

In order to achieve balanced effectiveness and scalability, the following two evaluation functions are used selectively. Here, the fitness function of each individual i, $fitness(i)$, is defined as follows:

$$fitness(i) = 1 - \frac{WSA\_FF(i)}{WSA\_FF\_All} \quad \text{..... (i)}$$

if the *X*-bit ratio in the test cube under *X*-filling is over 60%; otherwise,

$$fitness(i) = 1 - \frac{WSA\_Node(i)}{WSA\_Node\_All} \quad \text{..... (ii)}$$

Here, $WSA\_FF(i)$ and WSA_FF_All in Expression (i) are the WSA value for all scan FFs with respect to the individual i and the WSA value for all scan FFs when all of them are switching, respectively. $WSA\_Node(i)$ and WSA_Node_All in Expression (ii) are the WSA value for all nodes (FFs and gates) with respect to the individual i and the WSA value for all nodes when all of them are switching, respectively.

The computation time of Expression (ii) is longer than that of Expression (i); however, Expression (ii) is more effective in reducing WSA for all nodes in a circuit. For this reason, GA-fill uses Expression (i) for a test cube with a large number of *X*-bits, and Expression (ii) for a test cube with a small number of *X*-bits. As a result, both high effectiveness and good scalability can be achieved in a balanced manner.

3.3.2 Critical Area Concentration

By using a proper fitness function, GA-fill can also achieve concentrated reduction effect of launch

switching activity in a given critical area. Generally, a ***critical area*** is a set of nodes (FFs and gates), for which IR-drop due to its launch switching activity has significant impact on timing of the circuit.

Critical area identification [23,24,27] can be conducted dynamically for each test vector or statically for the whole circuit, based on gate-level information or even layout information. An example is shown in Fig. 4, in which an activated critical path goes from G_1 to G_{12}. Since launch switching activity near this path has high impact on timing, it is reasonable to use all nodes on the path, G_1, G_5, G_8, and G_{12}, as well as nodes near the path, G_2, G_9 and G_{11}, to form a critical area [23].

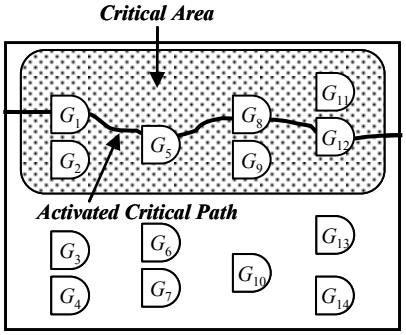

Fig. 4 Example of Critical Area Identification.

When the goal is to achieve concentrated reduction effect of launch switching activity in a given critical area, the following fitness function can be used:

$$fitness(i) = 1 - \frac{WSA\_CS\_Node(i)}{WSA\_CS\_Node\_All} \quad \text{..... (iii)}$$

Here, *WSA_CS_Node(i)* and *WSA_CS_Node_All* in Expression (iii) are the WSA value for all nodes in the given critical area with respect to the individual *i* and the WSA value for all nodes in the critical area when all of them are switching, respectively.

4. Experimental Results

GA-Fill was implemented in C, and experiments were conducted on five largest ISCAS'89 benchmark circuits. The computer used had a 3.0GHz CPU and 24GB memory. Fully-specified transition delay test vectors were first generated with an internal ATPG tool, and partially-specified test cubes were then obtained from them with an internal test relaxation tool [22]. The Roulette wheel selection and two-point crossover were employed for parent selection and crossover, respectively. Mutation probability was set to 0.1%. GA-fill was terminated when the number of generations reached 100 or the fitness of the global best individual became greater than 0.85.

First, we used the fitness function targeted for balanced effectiveness and scalability in GA-fill. For comparison, the justification-based LCP *X*-filling [14] with high effectiveness and the probability-based preferred fill [15] with high scalability were also implemented. The experimental results are summarized in Table 1. Here, the test set information of each circuit, including the number of test vectors ("*Test Vec. #*"), fault coverage ("*Fault Cov.* (%)"), and percentage of *X*-bits ("*X* (%)") identified by test relaxation, is shown. This table also shows the maximum reduction ratio of WSA for all nodes ("*Max. WSA Reduction Ratio* (%)") and the CPU time ("*CPU* (s)").

Table 1 Results for Balanced Effectiveness and Scalability

Circuit	Test Set			Reduction Ratio (%)						CPU (s)		
	Test Vec. #	Fault Cov. (%)	X (%)	Max. WSA-FF			Max. WSA-Node					
				[14]	[15]	GA	[14]	[15]	GA	[14]	[15]	GA
s13207	323	79.5	67.4	23.5	19.2	**23.2**	11.7	7.9	**13.2**	40	9	**28**
s15850	221	70.2	54.3	35.2	31.7	**36.7**	26.2	23.3	**27.6**	33	8	**29**
s35932	337	82.5	92.0	44.5	29.3	**41.4**	7.6	10.0	**10.4**	230	26	**152**
s38417	270	98.0	47.1	36.2	23.0	**35.9**	14.2	14.2	**20.1**	654	27	**296**
s38584	412	83.9	73.9	17.8	11.2	**17.7**	14.9	8.7	**15.0**	2086	42	**194**
Ave.	—	—	54.7	31.4	22.9	**31.0**	14.9	12.8	**17.3**	—	—	—

As shown in Table 1, for effectiveness in terms of the maximum reduction ratio of WSA for all nodes in a circuit, GA-fill outperformed other *X*-filling methods. In addition, for scalability in terms of CPU time, GA-fill was much closer to that of the best-performing probability-based preferred fill [15] than the justification-based LCP *X*-filling [14]. This clearly demonstrates that GA-fill can indeed achieve both effectiveness and scalability in a much more balanced manner than previous *X*-filling methods for reducing launch switching activity in at-speed scan testing. This can also be seen in Fig. 5 from the relation between scalability (measured by CPU time) and effectiveness (measured by maximum reduction ratio of WSA for all nodes) for each *X*-filling method on s38584.

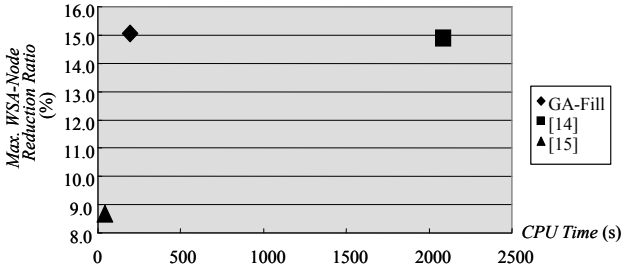

Fig. 5 Effectiveness vs. Scalability for s38584.

Next, we used the fitness function targeted for concentrated reduction effect of launch switching activity in a given critical area. 10% of nodes in each circuit were randomly selected to form a critical area. For comparison, GA-fill using the fitness function targeted for balanced effectiveness and scalability was also executed. The same test sets in Table 1 were used. The experimental results are shown in Table 2.

As shown in Table 2, GA-fill using the fitness function targeted for concentrated reduction effect of launch switching activity (GA-2: GA-Concentrated) did

outperform GA-fill (GA-1: GA-Balanced) using the fitness function targeted for balanced effectiveness and scalability for a given critical area. For some circuits, the difference is not significant. This is because (1) the circuits were small, (2) nodes in critical areas were selected randomly, and (3) only gate-level information was used. We are planning further evaluation experiments using large industrial circuits and physical layout information for critical area selection [27].

Table 2 Results for Critical Area Concentration

Circuit	Critical Area (# of Nodes)	Max WSA Reduction Ratio (%) for Critical Area		CPU (s)	
		GA-Balanced (GA-1)	GA-Concentrated (GA-2)	GA-1	GA-2
s13207	1401	11.5	**12.7**	30	**20**
s15850	1655	31.0	**31.6**	31	**22**
s35932	3771	10.4	**10.5**	149	**99**
s38417	4012	18.0	**18.0**	312	**98**
s38584	4018	16.9	**16.9**	205	**132**
Ave.	—	17.6	**18.0**	—	—

5. Conclusions

This paper proposed a novel *X*-filling method, *GA-fill*, for reducing launch switching activity in at-speed scan testing. GA-fill guides *X*-filling with flexible fitness functions, so as to achieve effectiveness and scalability in a more balanced manner, and to make the effect of launch switching activity reduction more concentrated on critical areas that have higher impact on IR-drop-induced yield loss. Initial experimental results clearly demonstrated the usefulness of GA-fill.

We are currently conducting additional experiments on large industrial circuits to further evaluate GA-fill, and the results will be used to fine-tune GA-fill parameters.

References

[1] Y. Sato, S. Hamada, T. Maeda, A. Takatori, Y. Nozuyama, and S. Kajihara, "Invisible Delay Quality - SDQM Model Lights Up What Could Not Be Seen," *Proc. of ITC*, Paper 47.1, 2005.

[2] L.-T. Wang, C.-W. Wu, and X. Wen, (Editors), *VLSI Test Principles and Architectures: Design for Testability*, San Francisco: Morgan Kaufmann, first edition, 2006.

[3] S. Sde-Paz and E. Salomon, "Frequency and Power Correlation between At-Speed Scan and Functional Tests," *Proc. of ITC*, Paper 13.3, 2008.

[4] J. Saxena, K. Butler, V. Jayaram, and S. Hundu, "A Case Study of IR-Drop in Structured At-Speed Testing," *Proc. of ITC*, pp. 1098-1104, 2003.

[5] R. Sankaralingam and N. A. Touba, "Inserting Test Points to Control Peak Power During Scan Testing", *Proc. of Intl. Symp. on DFT*, pp. 138-146, 2002.

[6] S. Gerstendoerfer and H. Wunderlich, "Minimized Power Consumption for Scan-Based BIST," *Proc. of ITC*, pp. 77-84, 1999.

[7] K. Lee, T. Huang, and J. Chen, "Peak-Power Reduction for Multiple-Scan Circuits during Test Application," *Proc. of ATS*, pp. 453-458, 2000.

[8] P. Girard, "Survey of Low-Power Testing of VLSI Circuits," *IEEE Design & Test of Computers*, Vol. 19, No. 3, pp. 82-92, Feb. 2002.

[9] S. Wang and W. Wei, "A Technique to Reduce Peak Current and Average Power Dissipation in Scan Designs by Limited Capture," *Proc. of ASP-DAC*, pp. 810-816, 2007.

[10] 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 CAD*, Vol. 17, No. 12, pp. 1325-1333, 1998.

[11] F. Corno, P. Prinetto, M. Redaudengo, and M. Reorda, "A Test Pattern Generation Methodology for Low Power Consumption," *Proc. of VTS*, pp. 35-40, 1998.

[12] N. Ahmed, M. Tehranipoor, and V. Jayaram, "Supply Voltage Noise Aware ATPG for Transition Delay Faults," *Proc. of VTS*, pp. 179-184. 2007.

[13] W. Li, S. M. Reddy, I. Pomeranz, "On Reducing Peak Current and Power during Test," *Proc. of ISVLSI*, pp. 156-161, 2005.

[14] X. Wen, Y. Yamashita, K. Kajihara, L.-T Wang, K. K. Saluja, and K. Kinoshita, "Low-Capture-Power Test Generation for Scan Testing," in *Proc. of VTS*, pp. 265-270, 2005.

[15] S. Remersaro, X. Lin, Z. Zhang, S. M. Reddy, I. Pomeranz, and J. Rajski, "Preferred Fill: A Scalable Method to Reduce Capture Power for Scan Based Designs," *Proc. of ITC*, Paper 32.2, 2006.

[16] X. Wen, K. Miyase, S. Kajihara, T. Suzuki, Y. Yamato, P. Girard, Y. Ohsumi, and L.-T. Wang, "A Novel Scheme to Reduce Power Supply Noise for High-Quality At-Speed Scan Testing," *Proc. of ITC*, Paper 25.1, 2007.

[17] 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. of ICCD*, pp. 251-258, 2006.

[18] J.-L. Yang and Q. Xu, "State-Sensitive *X*-Filling Scheme for Scan Capture Power Reduction," *IEEE Trans. on Computer-Aided Design of Integrated Circuits & Systems,* Vol. 27, No. 7, pp. 1338-1343, July 2008.

[19] P. Girard, X. Wen, and N. A. Touba, *Low-Power Testing* (Chapter 7) in *Advanced SOC Test Architectures - Towards Nanometer Designs*. San Francisco: Morgan Kaufmann, first edition, 2007.

[20] J. Li, Q. Xu, Y. Hu and X. Li, "iFill: An Impact-Oriented *X*-Filling Method for Shift- and Capture-Power Reduction in At-Speed Scan-Based Testing," *Proc. of DATE*, pp. 1184-1189, 2008.

[21] S. Remersaro, X. Lin, S. M. Reddy, I. Pomeranz, J. Rajski, "Low Shift and Capture Power Scan Tests," *Proc. of VLSI Design*, pp. 793-798, 2007.

[22] K. Miyase and S. Kajihara, "XID: Don't Care Identification of Test Patterns for Combinational Circuits," *IEEE Trans. on Computer-Aided Design*, Vol. 23, No. 2, pp. 321-326, Feb. 2004.

[23] X. Wen, K. Miyase, T. Suzuki, S. Kajihara, Y. Ohsumi and K. K. Saluja, "Critical-Path-Aware *X*-Filling for Effective IR-Drop Reduction in At-Speed Scan Testing," *Proc. of DAC*, pp. 527-532, 2007.

[24] A. K. Kokrady and C. P. Ravikumar, "Static Verification of Test Vectors for IR Drop Failure," *Proc. of ICCAD,* pp. 760-764, 2003.

[25] D. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.

[26] J. Holland, *Adaptation in Natural and Artificial Systems*, The University of Michigan, 1975., and MIT Press, 1992.

[27] N. Ahmed, M. Tehranipoor, and V. Jayaram, "Transition Delay Fault Test Pattern Generation Considering Supply Voltage Noise in a SOC Design," *Proc. of DAC*, pp. 533-538, 2007.

[28] M.-F. Wu, J.-L. Huang, X. Wen, and K. Miyase, "Reducing Power Supply Noise in Linear-Decompressor-

Based Test Data Compression Environment for At-Speed Scan Testing," *Proc. of ITC*, Paper 13.1, 2008.