Embedded Test Decompressor to Reduce the Required Channels and Vector Memory of Tester for Complex Processor Circuit

Yinhe Han, *Member, IEEE*, Yu Hu, *Member, IEEE*, Xiaowei Li, *Senior Member, IEEE*, Huawei Li, *Member, IEEE*, and Anshuman Chandra, *Member, IEEE*

***Abstract*—An embedded test stimulus decompressor is presented for the test patterns decompression, which can reduce the required channels and vector memory of automatic test equipment (ATE) for complex processor circuit. The proposed decompressor mainly consists of a periodically alterable MUX network which has multiple configurations to decode the input information flexibly and efficiently. In order to reduce the number of test patterns and configurations, a test patterns compaction algorithm, using CI-Graph merging, is proposed. With the proposed periodically alterable MUX network and the patterns compaction algorithm, smaller test data volume and required external pins can be achieved as compared to previous techniques.**

***Index Terms*—Automatic test equipment (ATE), Godson processor, MUX network, test stimulus decompression.**

I. Introduction

As MOORE's law predicated that the number of transistors on a chip doubles every couple of years, today's VLSI circuits have great integration density. As the amount of logic on the chip has increased, the mount of test patterns required for such large designs is also growing rapidly. Moreover, the current generation testers have limited speed, memory and input/output (I/O) channels, and upgrading to more advanced testers can be very costly. Hence, conventional external testing approaches where test data is stored on the tester and transferred to and from circuit-under-test is becoming increasingly difficult. The main bottleneck is the limited vector memory and I/O bandwidth (product of the number of channels and tester frequency) [1]. Test compression can mitigate the requirement of the expensive automatic test equipments (ATEs) which have high-volume vector memory and a large number of channels.

The ability of test compression scheme to compress test data in previous works, stems from the high X ratio in test patterns.

TABLE I
High X Ratio in Godson and PowerPC Microprocessors

Circuits	ATPG Tool	Fault Coverage(Stuck-At)	X ratio
Godson2-C	TetraMAX	97.40%	98.53%
PowerPC1	TetraMAX	99.80%	97.82%
PowerPC2	TetraMAX	99.76%	95.73%

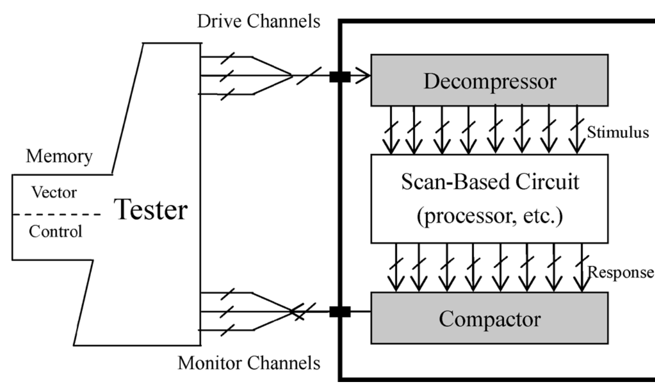


Fig. 1. Conceptual test architecture with embedded test resource.

The cases are investigated by some processor circuits. Table I lists the X ratio in test patterns of: one ICT microprocessor Godson2-C [2] and two IBM PowerPC microprocessor circuits. These patterns are generated by the Synopsys TetraMax ATPG tool and only consider stuck-at faults. In test pattern generation, the static compaction is applied without random X filling.

Test compression is implemented by some embedded test resources. The conceptual test compression architecture is shown in Fig. 1. A stimulus decompressor and a response compactor (such as the compactor based on advanced convolutional code [3]) are inserted between the ATEs and scan-based circuit-under-test (CUT). The compressed stimulus is transferred from the vector memory of ATE to the on-chip decompressor through a few drive channels. The decompressor decodes the compressed stimulus into original test patterns and applies them to the CUT. The test responses are output to a compactor and then the compacted data is transferred to ATE to compare. Thus, with the decompressor and compactor, only a small number of test channels and smaller vector memory are required in the ATE.

Due to the efficiency, the test stimulus decompressor is considered as an important test resource. It is a well researched topic and a number of techniques have been presented. Golomb [4], FDR [5], Selective Huffman Code [7], and Nine-coded [6] are

Manuscript received July 29, 2006; revised November 15, 2006. This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 60576031, Grant 60633060, Grant 60606008, and Grant 90607010, and by the National Basic Research Program of China (973) under Grant 2005CB321604 and Grant 2005CB321605. Y. Han's work was supported by the fund of Chinese Academy of Sciences due to the President Scholarship. A preliminary version of this paper appeared in Proceedings of the IEEE/ACM International Symposium on Quality Electronic Design (ISQED), San Jose, CA, March 23–24, 2005.

Y. Han, Y. Hu, X. Li, and H. Li are with the Key Laboratory of Computer System and Architecture, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100080, China (e-mail: yinhes@ict.ac.cn).

A. Chandra is with the R&D Department, Synopsys, Inc., Mountain View, CA 94043 USA.

Digital Object Identifier 10.1109/TVLSI.2007.893652

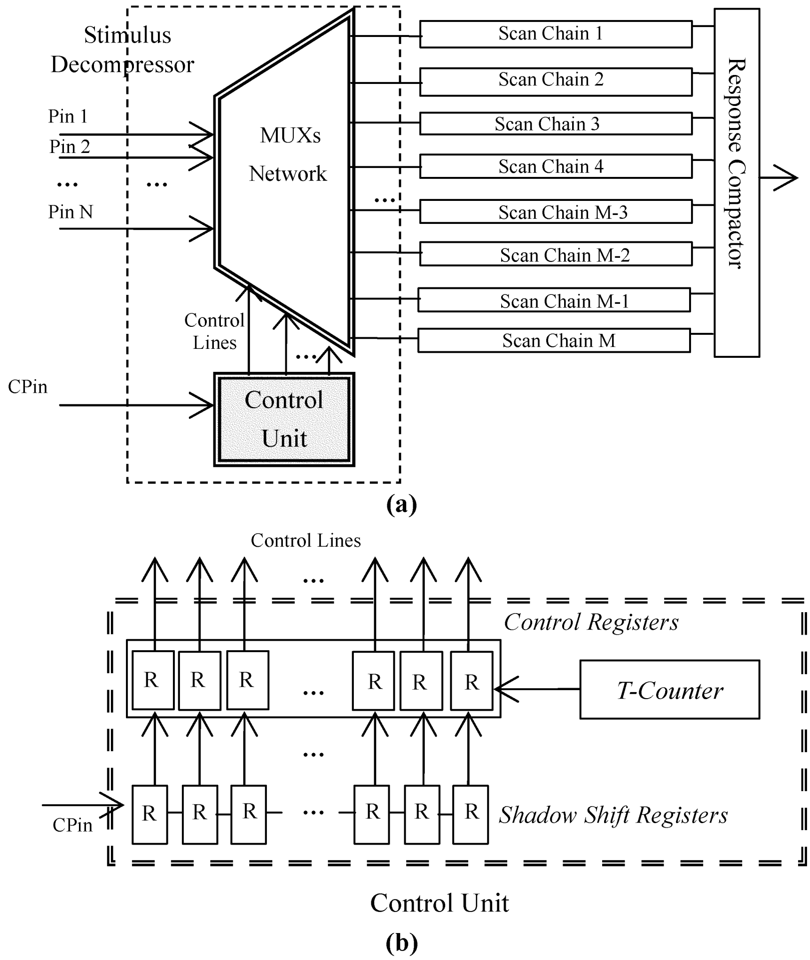


Fig. 2. (a) Proposed periodically alterable MUX network decompressor. (b) Detailed design of control unit.

some coding-based techniques to compress the test stimulus. Reda [9] presented a mutation decoder to compress test patterns into bit stream that indicates which bits need to be flipped in current test slice to obtain the subsequent one. Linear feedback shift registers (LFSRs) were applied in [10] to expand the input information. A large LFSR driven by few external pins was presented. It is composed of a large number of smaller LFSRs, each of which feeds a scan chain. The mutation version of LFSR: ring generator was presented in [11]. Some other LFSRs-based techniques were presented in [12] and [13]. The XORs network can be used as a decompressor [14]–[17]. These methods used a network composed of XOR gates to expand the input information. They are more efficient when combined with the custom patterns compaction algorithm.

In this paper, we present an embedded decompressor based on the periodically alterable MUX network. Previous works based on MUX network were presented in [18]–[20]. The proposed scheme is different from the previous techniques in the following ways.

1) Only a serial load input is required to change the configurations, which saves the pins count required. Further, multiple configurations can guarantee to flexibly decode the input information and provide the high fault coverage.
2) A graph merging-based test compaction algorithm is proposed which cannot only reduce the number of required configurations in MUX network, but also the number of test patterns.

The rest of this paper is organized into five sections. The proposed decompression scheme is introduced in Section II. The static compaction algorithm is presented in the Section III. In Section IV and V, we analyze the experimental results, and draw some conclusions in Section VI.

II. Proposed Embedded Decompressor Architecture

A. Architecture of Proposed Decompressor

The proposed embedded decompressor architecture based on MUX network is shown in Fig. 2. There are N external scan-in pins that feeds M internal scan chains through the MUX network, where $M \gg N$. The proposed scan architecture is similar to the Illinois scan architecture [21], where multiple scan chains are fed through a single scan-in pin. However, the decompression architecture presented in this paper is more efficient than [21] as the connection relations between the external scan-in pins and the internal scan chains can be altered periodically, which results in generating a small number of more effective patterns. The periodic reconfiguration of the mapping between the scan-in pins and the internal scan chains are done by changing the control signals of the MUX network. The control signals (control lines in Fig. 2) are generated by the *control* *unit*. The detailed architecture is shown in Fig. 2(b). The control signals come from *control registers* and the T-*counter* that runs for T cycles. The mapping relation between scan-in pins and the internal scan chains is loaded into the *shadow shift registers* through a separate serial load pin (CPin) as shown in Fig. 2(b). The contents of *shadow shift registers* are loaded into the *control registers* after every T cycles. For example, if T is equal to 3, the *control registers* will be updated from the *shadow shift registers* and the mapping would change after every three cycles.

The MUX network in the proposed decompressor is mainly constructed by MUX gates. Some other kinds of gates are also included, which transfer the control signals coming from the *control unit* to the select lines of MUX gates. The MUX network contains several configurations. Each "configuration" means a connection relation between the inputs and the outputs of MUX network. The configurations can be selected by changing the data in control lines. This procedure is also called as "reconfigure." The reconfiguration period is equal to the number of stages in *control registers*, which is also equal to the number of select lines of MUX network. Therefore, if the number of configurations of MUX network is C, then the reconfiguration period of MUX network T should be constrained as

$$T \geq \lceil \log_2(C) \rceil .$$

As an example, the MUX network that has two configurations is shown in the following.

Example: In a circuit with scan chains, four external input pins will drive eight scan chains. The MUX network has two configurations. The first one is: $1 \longrightarrow \{2,3,6\}$, $2 \longrightarrow \{7\}$, $3 \longrightarrow \{5,8\}$, $4 \longrightarrow \{1,4\}$, where $A \longrightarrow \{B_1, B_2, \ldots, B_n\}$ means the A^{th} external input pin drives the $B_1^{\text{th}}, B_2^{\text{th}}, \ldots, B_n^{\text{th}}$ scan chain. Another configuration is: $1 \longrightarrow \{1,6\}$, $2 \longrightarrow \{2,4\}$, $3 \longrightarrow \{3,5,7,8\}$.

The block diagram of the MUX network of the previous example is shown in Fig. 3. It consists of five two-input MUX gates. Since there are two configurations, only one control line of MUX network is needed. If the control line is "0," the first configuration is selected. Otherwise, the second configuration is selected.

B. Synthesis of the Proposed MUX Network

If the length of the longest scan chain is L and the number of patterns is P, then the entire test set will contain $L \times P$ scan slices. These slices can be partitioned into $B = \lceil L \times P/T \rceil$ blocks. A block can be represented by a scan chain incompatible graph CI-Graph: $G(V,E)$.[1] In this graph, each node in V represents a scan chain. If the value of two nodes: V_i and V_j are *incompatible*, there will be an edge between them. The following definition describes the "*incompatible*" property.

Definition 1: To a scan chain S_i, $S_i[q]$ is the value of q^{th} scan cell in S_i. Given two scan chains S_i, S_j, they are *incompatible*, if $\exists q(1 \leq q \leq L)$, then $(S_i[q] = 0, S_j[q] = 1)$ or $(S_i[q] = 1, S_j[q] = 0)$.

Thus, $S_i[q]$ and $S_j[q]$ are *compatible* if they are equal or at least one of them is don't care (X) bit. Two examples

[1]A similiar graph model is also defined in [31].

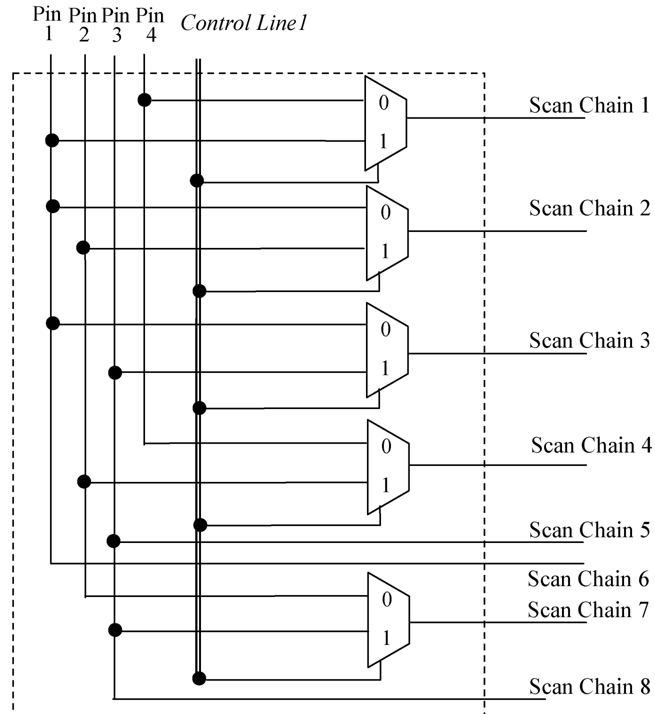


Fig. 3. Internal structure of MUX network.

of CI-Graphs are shown in Fig. 4(b) and (c). The corresponding test pattern is listed in Fig. 4(a). The reconfiguration period is assumed as five cycles and each fragment of scan vector contains 10 bits (two periods). The block consists of eight scan chains, which corresponds to eight nodes in CI-Graph. We note the configuration of the first period. Considering the scan chain 1, it is incompatible with chain 5, chain 7 in the first cell and with chain 2, chain 7 in the third cell. Therefore, node 1 is connected with nodes 2, 5, and 7.

If two scan chains are compatible, they can be driven by the same pin. In the CI-Graph, if we consider compatibility, the nodes can be partitioned into independent sets. In each independent set, any pair of nodes is compatible. Thus, the scan chains whose corresponding nodes are included in an independent set, can be assigned to one external input. Hence, the following property can be obtained.

Property 1: If a CI-Graph of a period is partitioned into independent sets, these independent sets will correspond to a configuration. The scan chains within an independent set can be driven by one external input pin.

Note the first period in Fig. 4 again. One of possible partitioned independent sets are: {2,3,6}, {7}, {5,8}, {1,4}. They are shown in Fig. 4(b) through filling the different independent sets with different backgrounds. Each independent set needs to be driven by an external pin. The detailed architecture of the connection relation between external pins and scan chains of this example is shown in Fig. 3 when *control line* 1 is selected as 0.

In order to get the minimum number of external pins, the minimum number of independent sets have to be determined first. This problem of partitioning into independent sets is equivalent to the graph coloring problem. Thus, the minimum number of independent sets is equivalent to obtaining the

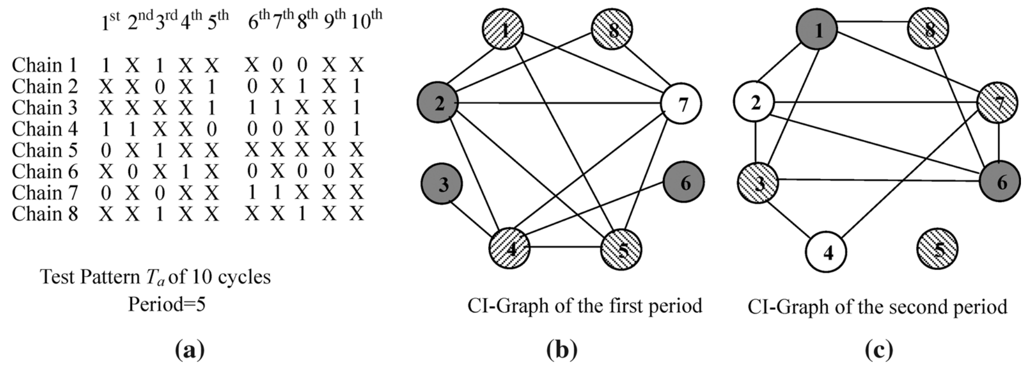


Fig. 4. Example of CI-Graphs.

minimum number of distinct colors to color all nodes in a CI-Graph. If we define the compression ratio (CR) as

$$((\text{Regular Scan Test Data}) - (\text{Periodically Alterable MUX Network Test Data})) / (\text{Regular Scan Test Data}) \times 100\%.$$

The following corollary provides an upper bound of the CR.

Corollary 1: If the minimum number of distinct colors of each CI-Graph in the total pattern is: $G_1, G_2, \ldots, G_B$ and $N = MAX(G_1, G_2, \ldots, G_B)$, the upper bound of CR of MUX network is obtained as

$$\text{MAX(CR)} = (M - N - 1)/M \times 100\%$$

where M is the number of scan chains.

Proof: The test data volume of regular scan architecture is

$$V_{\text{regular}} = M \times L \times P.$$

Based on Property 1, if a MUX network is inserted in the front of regular scan chains, the MUX network will have N inputs at least. The length of scan chains and number of patterns are not varied. Thus, the MUX network test data volume is

$$V_{\text{MUX}} = (N + 1) \times L \times P$$

at least ($N + 1$ means N data inputs and one serial load input). So the upper bound of CR of MUX network is calculated as Corollary 1.

The minimum number of distinct colors is known as the chromatic number. The chromatic number problem is a well-known NP-complete problem in graph theory. Many algorithms have been proposed to obtain approximate coloring in reasonable time [24].

Parameters Choice: The reconfiguration period T is an important parameter to determine compression ratio, area overhead, and even test quality. In the proposed scheme, the parameter T is predetermined based on many tries. After the determination of T, the number of inputs of periodically alterable MUX network N can be obtained by Property 1. Generally, a long period will lead to a small number of configurations (area overhead), but high conflict probability, which can lead that a fault cannot be encoded. In test practices, test quality often has a higher priority than compression or others. Hence, the short period is selected in this paper. We search the period T from 3 to 10, and select the best results to report.

C. Configurations Reduction Using Graph Merging

Corresponding to $\lceil L \times P/T \rceil$ blocks, there are $\lceil L \times P/T \rceil$ original CI-Graphs. However, this does not imply that the MUX decompressor requires so many configurations. Some configurations can be merged and reused. If we limit the number of external pins as N, the definition "N-*Mergeable*" can be used to formulate this mergence.

Definition 2: Graph G_1 is defined as *subgraph* of graph G_2, if the following conditions are satisfied: $\forall\ node\ n$, where $n \in V(G_1)$, then $n \in V(G_2)$ and $\forall\ edge\ e$, where $e \in E(G_1)$, then $e \in E(G_2)$.

Definition 3: Two CI-Graphs G_1 and G_2, are defined as N-*mergeable* if they satisfy one of two following conditions.

1) G_1 is a subgraph of G_2 or G_2 is a subgraph of G_1.
2) After merging G_1 and G_2 to a new graph G', the chromatic number in G' is not greater than N. The set of nodes in new G is $V(G') = \{v | v \in V(G_1)\ or\ v \in V(G_2)\}$ and the set of edges is $E(G') = \{e | e \in E(G_1)\ or\ e \in E(G_2)\}$).

If two CI-Graphs are not N-*mergeable*, they are N-*conflicting*. Based on the definition of N-*mergeable*, the following property about the number of configurations of MUX decompressor can be obtained:

Property 2: If all CI-Graphs are grouped into N-*mergeable sets* and in each set any pair of CI-Graphs is N-*mergeable*, then the minimum number of configurations in MUX decompressor is equal to the minimum number of N-*mergeable sets*.

As we computed the minimum number of external inputs, the minimum number of configurations can be reduced to a graph coloring problem. It is also NP-complete. Many heuristic algorithms can be applied to determine the minimum number of configurations. However, in this paper, a simple heuristic approach as shown in the following, was used to merge CI-Graphs.

CI-Graphs Merging Algorithm

---

1) Given test patterns: the number of patterns is P and the length of scan chain is L;
2) Partition the patterns into $\lceil L \times P/T \rceil$ blocks;
3) Construct the CI-Graph for every block. GL is the CI-Graph list;

4) Select the first CI-Graph in GL as G;
5) While GL is not empty
   a) Select a new CI-Graph in GL as G';
   b) If G and G' are N-*Mergeable*, then merge G' into G, delete G' from GL.

In step 5.b, in order to determine the N-*mergeable*, a graph-coloring program is embedded to calculate the chromatic number of the merged graph.

III. Test Patterns Compaction Algorithm Using CI-Graph Merging

From the synthesis flow of the periodically alterable MUX network, it is seen that the internal structure of MUX network is determined by test patterns. If the test patterns are predetermined, the maximum compression ratio of proposed MUX network is also determined. So, if we are able to custom the test patterns based on the structure of decompressor, the higher compression result can be expected.

The ability of the fixed rate compression scheme that is proposed in this work to compress test data stems from the high X ratio. In order to complete fault coverage and high test compression result, the test generation needs to be incorporated into the compaction algorithm. While a regular compaction algorithm only checks for compatibility of two test cubes before merging them, this work needs to check for N-*mergeable* of two patterns. The proposed technique necessitates only slight and well- contained modifications to current ATPG methods, thus retaining the significant investment of software in the IC industry. The algorithm in [17] is selected to modified. The main difference with [17] is that the N-*mergeable* is used to merge patterns instead of the compressible merging. Furthermore, a redundant patterns reduction algorithm is augmented to remove the redundant patterns.

Test Patterns Compaction Algorithm Using CI-Graph Merging

(1.) Patterns Generation(similar with [17]):
  (1.1) While(The targeted fault converge is not reached)
    1.1.1) Select a fault from undetected fault list (UFL), generate the test pattern;
    1.1.2) Fault Dropping. Fault simulation with X-bit and delete the detected faults from UFL.
(2.) Patterns Compaction:
  (2.1) Run the Compatible Patterns Merging Algorithm;
  (2.2) Run the Redundant Patterns Reduction (RPR) Algorithm.

In the Compatible Patterns Merging Algorithm of phase (2.1), two conditions determine whether two patterns should be merged. They are: 1) two patterns are compatible and 2) the CI-Graphs of the two patterns are N-*mergeable*. If both are satisfied, two patterns can be merged into one pattern and compacted patterns are obtained.

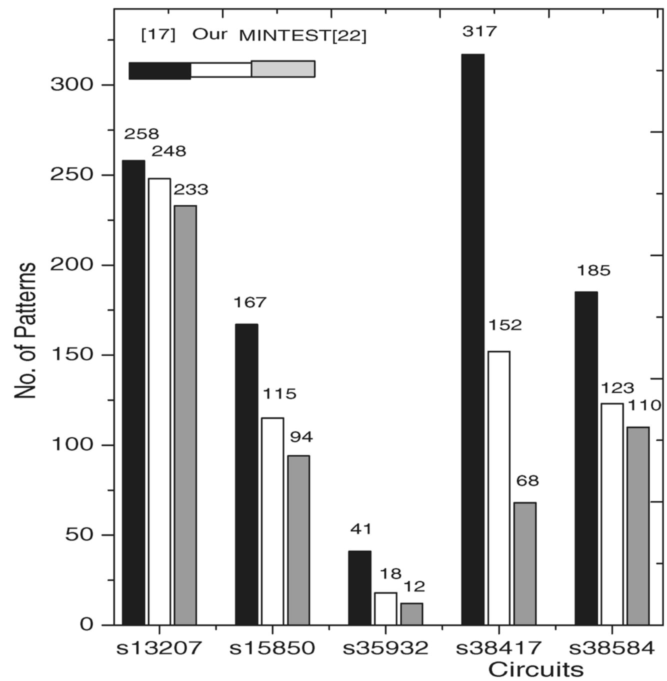


Fig. 5. Comparisons of the numbers of test patterns.

Compared to the compaction algorithm presented in [17], an RPR algorithm is performed. The algorithm in [17] is forward-compacting completely, so some faults detected by earlier patterns may also be accidentally detected by the test patterns later. Thus, some of the patterns generated earlier may become redundant. The RPR algorithm identifies these patterns and removes them. In our implementation, a counter was setup for each fault to record the number of times a fault was detected so as to determine which patterns should be removed. Through the RPR, the compaction is improved further. Experimental results listed in Fig. 5 shows this compaction improvement. Our compaction algorithm can get better results than [17] and closer to the MINTEST [22]. For example, for s38417, 165 patterns were reduced when the RPR algorithm was run compared to [17].

The efficiency of the compaction algorithm can be further improved by maximizing the number of unspecified bits in a test cube without losing the ability to test the target fault [22]. The number of test patterns may be further reduced with no significant importance on the compression ratio through incorporation of more efficient dynamic test compaction algorithms and fault ordering schemes.

IV. Experimental Results

The proposed compaction algorithm is built on top of the ATPG tool: *ATLANTA* and simulation tool: [29] *HOPE* [30]. The chromatic numbers of coloring graph are obtained by the J. Culberson's [24] program. We partition the scan chains and get the information about the compatible relation of scan slices and block first. Then, the compatible information is recorded as *DIMACS* standard graph format file. At last, *Iterated GREEDY* program is called to get the independent sets. In the implementation of *Iterated GREEDY* program, the selection of greedy type is the "simple type" and the "largest first type" with probability 100/250 each. The "random" is selected with probability

TABLE II
REQUIRED TESTER'S CHANNELS AND VECTOR MEMORY REDUCTION OF ISCAS'89 CIRCUITS ONLY WITH PROPOSED DECOMPRESSOR

Circuits	No. of Patterns	X Ratio	Vol. of T_o	No. of S. C.	No. of Ext. Pins	No. of Confs.	No. of MUX	Period	Vol. of T_c	Channels Saved	Vector Memory Saved
s13207	236	82.30%	165,200	32	8	22	30	5	41,536	75.00%	74.86%
				64	8	18	54	5	20,768	87.50%	87.43%
				100	9	18	79	5	14,868	91.00%	91.00%
s15850	126	93.20%	76,986	32	8	13	26	4	20,160	75.00%	73.81%
				64	8	11	58	4	10,080	87.50%	86.91%
				100	10	10	104	4	8,820	90.00%	88.54%
s35932	16	83.70%	28,208	32	8	26	37	5	7,168	75.00%	74.59%
				64	8	20	67	5	3,584	87.50%	87.29%
				100	8	21	102	5	2,304	92.00%	91.83%
s38417	99	38.60%	164,736	32	16	46	58	5	82,368	50.00%	50.00%
				64	21	38	60	5	54,054	67.19%	67.19%
				100	22	52	213	5	37,026	78.00%	77.52%
s38584	136	68.10%	199,104	32	11	19	31	5	68,816	65.63%	65.44%
				64	11	17	48	5	34,408	82.81%	82.72%
				100	13	33	167	5	26,520	87.00%	86.68%

50/250. These programs are run on a PC workstation with *Intel Pentium IV1.6-G* processor and 768-M memory. The OS platform is *RedHat Enterprise AS 3 Linux*. The programs are debugged under $GNU\ GCC$ environment.

To evaluate the proposed decompressor and patterns compaction algorithm, two sets of experiments are conducted first. In the first set of experiments, we will only evaluate the efficiency of proposed decompressor and the compaction algorithm is not applied. Table II lists the experimental results. The predetermined MINTEST test sets are used for these circuits targeting 100% fault coverage. One point should be noted that MINTEST test sets used here come from Duke University [4], [5] and are different with the test sets in Fig. 5 which are presented in [22]. To distinguish these two test sets, we label the MINTEST test sets coming from Duke University [4], [5] as "MINTEST(D)" and [22] as "MINTEST(U)." MINTEST(D) test sets have more patterns since they contain more X bits to achieve the higher compression ratio. The experiments are carried out to each circuit with 32 scan chains, 64 scan chains and 100 scan chains. In Table II, the volume of original test patterns (*Vol. Of* T_o) is calculated by the product of the number of patterns (*No. of Patterns*) and number of scan cells (*No. of S. C.*). The volume of compressed test data (*Vol. Of* T_c) is calculated by the product of the number of inputs of decompressor (*No. of Ext. Pins*), the length of scan chain and the number of patterns. "*No. of Confs.*" column shows the number of configurations and "*No. of MUX*" column shows the number of two-input MUX gates in the MUX network. "*Channels Saved*" and "*Vector Memory Saved*" columns show that our technique outperforms the method directly using the ATPG in every case, which are computed as

$$\begin{aligned}&\text{Channels Saved\%}\\&\quad=(\text{No. of Scan Chains}-\text{No. of Ext. Pins})/\\&\qquad\text{No. of Scan Chains}\times\ 100\%\\&\text{Vector Memory Saved\%}\\&\quad=(\text{Vol. of }T_o-\text{Vol. of }T_c)/\text{Vol. of }T_o\times\ 100\%.\end{aligned}$$

Next, we conduct the second set of experiments to apply the proposed decompressor and the compaction algorithm simultaneously. Test patterns are generated by the compaction algorithm targeting 100% fault coverage. Table III lists experimental results. Comparing Tables II and III, we find even though for some circuits ($s13207$, $s35932$ $s38417$), the numbers of test patterns of proposed compaction algorithm are worse than Duke'MINTEST test sets, the better compression ratios are always obtained because the graph-merging is considered during compaction, compaction algorithm outperforms 5.00% and 4.97% of required vector memory and channels to directly using MINTEST(D) test sets.

In order to demonstrate the effectiveness of the proposed compression technique for complex processor circuit, we present the experimental results on two PowerPC circuits from IBM: PowerPC_CKT 1 and PowerPC_CKT 2. These circuits are the same as the PowerPC_CKT 1 and PowerPC_CKT 2 presented in [31]. PowerPC_CKT 1 is a logic core consisting of 51 082 gates and its test set provides 99.80% fault coverage. It has 12 256 scan cells and primary inputs. PowerPC_CKT 2 is a logic core consisting of 94 340 gates and its test set provides 99.76% fault coverage. It has 22 216 scan cells and primary inputs. The scan cells in these two production circuits are partitioned into 128, 200, 400, 600, 800, and 1000 scan chains. Because of absence of the internal circuit information, the proposed decompressor is only applied. The experimental results are listed in Table IV. It is shown that our proposed decompressor is more efficient than the input reduction technique presented in [31]. The volume of test data compressed by our technique is only 1/2.5 and 1/3.5 of the results in [31].

To evaluate the impacts of the test quality and execution time of proposed scheme to the large circuit designs, the experiments are carried on two large circuits which are assembled by some larger ISCAS'89 circuits. The ICT_CKT 1 is made up of 5 $\times$ $s38417$, 7 $\times$ $s35932$, 7 $\times$ $s38584$, 10 $\times$ $s15850$, and contains about 0.43-M gates, 31-K DFFs. The ICT_CKT 2 is made up of $15\times s38417$, $20\times s35932$, $20\times s38584$, $30\times s15850$, and 1.3-M gates, 102-K DFFs. Tables V and VI give the detailed experimental results. The column "Regular Scan + ATPG" in the tables shows the results of test patterns generated by a MINTEST-like ATPG program and the regular full scan design. In this scenario, the numbers of scan chains is also 200. The column "decompressor+compaction algorithm" lists the results that proposed

TABLE III
REQUIRED TESTER'S CHANNELS AND VECTOR MEMORY REDUCTION OF ISCAS'89 CIRCUITS WITH PROPOSED DECOMPRESSOR + COMPACTION ALGORITHM

Circuits	No. of Patterns	X Ratio	Vol. of T_o	No. of S. C.	No. of Ext. Pins	No. of Confs.	No. of MUX	Period	Vol. of T_c	Channels Saved	Vector Memory Saved
s13207	248	87.21%	173,600	32	6	14	29	5	32,736	81.25%	81.25%
				64	6	15	51	4	16,368	90.63%	90.62%
				100	7	16	76	4	12,152	93.00%	93.00%
s15850	115	89.33%	70,265	32	8	24	30	5	17,280	75.00%	73.81%
				64	9	22	55	5	9,720	85.94%	85.27%
				100	11	10	75	4	8,316	89.00%	87.40%
s35932	18	95.20%	31,734	32	6	12	27	4	6,048	81.25%	80.94%
				64	7	18	63	5	3,528	89.06%	8.88%
				100	7	19	95	4	2,268	93.00%	92.85%
s38417	152	83.30%	252,928	32	9	23	35	5	71,136	1.88%	771.88%
				64	9	26	100	5	35,568	85.94%	85.94%
				100	11	34	177	5	28,424	89.00%	88.76%
s38584	123	82.50%	180,072	32	9	13	31	4	50,922	71.88%	71.72%
				64	10	24	48	5	28,290	84.38%	84.29%
				100	10	44	152	4	18,450	90.00%	89.75%

TABLE IV
COMPARISON TO THE INPUT REDUCTION TECHNIQUE [31] FOR IBM POWERPC MICROPROCESSOR CORE ONLY WITH PROPOSED DECOMPRESSOR

Circuits	Vol. of T_o	Fault Coverage	X Ratio	No. of Scan Chains	Len. of Scan Chains	Proposed approach			[31]	
						No. of required drive channels	Size of required vector memory	No. of Confs.	No. of required drive channels	Size of required vector memory
PowerPC_CKT1	46,180,608	99.80%	97.82%	128	96	11	3,979,008	26	16	6,374,400
				200	62	13	3,037,008	46	19	8,944,554
				400	31	20	2,336,160	32	26	6,560,840
				600	21	28	2,215,584	27	32	4,906,272
PowerPC_CKT2	58,561,376	99.76%	95.73%	128	174	8	3,669,312	23	-	-
				200	112	10	2,952,320	23	-	-
				400	56	13	1,919,008	35	16	7,705,600
				600	38	22	2,203,696	31	20	7,505,760
				800	28	22	1,623,776	29	23	5,314,288
				1000	23	25	1,515,700	40	27	5,132,565

TABLE V
APPLICATION RESULTS OF ICT_CKT 1

ICT_CKT 1	Regular Scan+ ATPG	Proposed Decompressor+ Compaction Algorithm 0.01% fault coverage loss
Fault Coverage	99.82%	99.81%
No. of Patterns	689	805
No. of External Pins	200	8
Vol. of Test Data	23.37M	1.13M
No. of Confs.	N/A	27
Run Times	1.2hrs	1.7hrs

TABLE VI
APPLICATION RESULTS OF ICT_CKT 2

ICT_CKT 2	Regular Scan+ ATPG	Proposed Decompressor+ Compaction Algorithm 0.1% fault coverage loss
Fault Coverage	99.23%	99.13%
No. of Patterns	763	869
No. of External Pins	200	9
Vol. of Test Data	81.93M	4.44M
No. of Confs.	N/A	26
Run Times	2.4hrs	3.6 hrs

the compaction algorithm and MUX Network are applied to two circuits, respectively. In this scenario, the number of internal scan chains is 200. To ICT_CKT 1, when the proposed compaction algorithm and MUX network are applied, the required drive channels and vector memory are reduced from 200 and 23.37 M to 10 and 1.13 M, only with 0.01% fault coverage loss. Because the graph coloring program is executed during generating the test patterns, the proposed compaction algorithm costs about two times run time than direct ATPG. To ICT_CKT 2, when proposed compaction algorithm and MUX network are applied, the required drive channels and vector memory are reduced from 200 and 81.93 M to 9 and 4.44 M, only with 0.1% fault coverage loss.

Furthermore, we provide two comparsions of the results to previously published data, as shown in Tables VII and VIII. Table VII presents a comparsion with approaches that use the same MINTEST(D) test sets. The results are taken from the five recent compression schemes: FDR [5], Mutation code [9], 9C [6], Selective Huffman [7], and three-stage [32]. The data volume in terms of the number of bits required to represent the test vectors (without the expected outputs) is provided in the table. The test volumes in bold denote the best test volume for each circuit hitherto achieved among the five previously proposed schemes. If test patterns compaction algorithm is applied, the compression ratio will rise further. The compression results with decompressor and compaction algorithm are listed in

TABLE VII
COMPARISON TO THE PREVIOUS COMPRESSION SCHEMES UNDER THE SAME MINTEST(D) TEST PATTERNS

Circuits	FDR [5]	9C [6]	Selective Huffman [7]	Mutation Code[9]	Three-stage [32]	Proposed
s13207	30,880	29,224	19,608	74,433	**15,221**	14,868
s15850	26,000	25,883	**12,024**	26,221	N/A	8,820
s35932	22,744	N/A	**2,508**	7,222	3,308	2,304
s38417	**43,466**	64,857	54,207	45,003	72,312	37,036
s38584	77,812	68,631	**28,120**	73,464	56,301	26,520

TABLE VIII
COMPARISON TO THE PREVIOUS SCAN CHAINS HIDING TECHNIQUES

Circuits	XOR-based			LFSR-based				Proposed
	[14]	[15]	[17]	[10]	[11]	[12]	[13]	
s13207	25,344	14,145	19,608	121,788,824	**10,585**	11,285	11,320	12,152
s15850	22,784	13,919	12,024	76,020	**9,805**	12,438	11,584	8,316
s35932	7,128	4,492	**2,508**	N/A	N/A	N/A	N/A	2,268
s38417	89,856	52,793	54,207	308,508	31,458	34,767	**30,560**	28,424
s38584	38,796	26,644	28,120	202,370	**18,568**	29,397	27,248	18,450

TABLE IX
AREA AND TIMING OVERHEAD OF PROPOSED DECOMPRESSOR

		ICT_CKT 1 with 200 scan chains	ICT_CKT 2 with 200 scan chains	PowerPC_CKT 1 with 600 scan chains	PowerPC_CKT 2 with 1000 scan chains
Area of CUT(μm^2)		6312660.14	18403675.2	1206749.6	2205207.763
Maximum Slack Time of CUT(ns)		23.36	32.37	-	
MUX Network	Area(μm^2)	23466.56	35923.43	52363.78	97993.61
	Maximum Slack Time (ns)	0.25	0.25	0.31	0.39
Control Unit	Area(μm^2)	1254.59	1254.59	1254.59	1417.45
	Maximum Slack Time (ns)	0.26	0.26	0.26	0.28
Area Overhead%		0.39%	0.20%	4.44%	4.51%
Timing Overhead%		1.07%	0.77%	-	-

Table VIII and compared to the previous compression methods. The previous methods are mainly classified as: XOR-based and LFSR-based techniques. For most of the benchmark circuits, the proposed method provides better results. Different with Table VII, the different patterns are used in these compression scheme to gain higher results. The EDT [11] is an efficient compression scheme since its experimental results are far better than other schemes. However, our architecture provides better results compared to EDT for three out of four large benchmark circuits.

V. DISCUSSION ON THE PENALTIES OF THE PROPOSED DECOMPRESSOR

Although the test community has been enthusiastic about adopting the embedded compressor, the design community still has several concerns to apply this scheme: 1) area overhead and 2) performance degradation during functional operations. The proposed decompressor, as shown in Fig. 2, consists of two major parts: the periodically alterable MUX network and control unit. It is necessary to estimate the area and performance degradation of these parts.

We use a C program to automatically generate the Verilog behavior model of the MUX network and the control unit in a SUN Blade1000 EDA server. It is debugged under $GNU\ CC$ environment. The behavior model is synthesized into gate netlist by the Synopsys Design Complier and the Chartered 0.13-μm (csm13hp) process library. The standard cells are provided by Aritsan Components, Inc. We set the wire load model as "ForQA," in which the normalized area for a unit length of wire is 1 μm^2. Table VIII lists the results. Because of absence of the internal structural information of the PowerPC_CKT 1 and PowerPC_CKT 2, the areas of them are roughly estimated by the number of gates and number of scan cells. Each logic gate is optimistically mapped to 10.18 μm^2 which is the area of two NAND2X1 gates and each scan cell is mapped to 56.01 μm^2 which is the area of scan cell in Aritsan standard cell library. The interconnect net area of these two circuits are assumed as 10% of logic area. 10% is the approximate ratio obtained from the ICT_CKT1 and ICT_CKT2.

Compared to the regular full scan design, our method does not augment any additional area overhead to scan cells and circuit-under-test. The main area overhead of our method is the MUX network and control unit. Observed from Table IX, the area overhead of proposed hardware is lower than 0.4% to ICT_CKT 1, ICT_CKT 2 and 4.5% to PowerPC_CKT 1, PowerPC_CKT2. A possible reason of the high area overhead ratio of IBM PowerPC circuit is that we only know the number of gates in these circuits and do not know what kinds of gates, thus we have no method to estimate the area precisely. As the technology feature size plunged to the nanometer range, the concern about the area overhead has been eased. Currently, allocating 10–20-k gates to the DFT logic has become acceptable. The impact on performance along the critical path is still challenging because of the high-frequency of function unit. The critical path of circuits in our experiment is evaluated using the "maximum slack time" which includes the data required time and data arrival time. Because the control unit is not in the main data path during testing, we only use the maximum slack time of MUX network and CUT to compute the "timing overhead" ratios which are listed in the last row of table. Observed from the table, the maximum slack time of MUX network is far smaller than CUT and the ratio is only about 1% for ICT_CKT 1 and ICT_CKT 2.

VI. Conclusion

A scheme for test pattern compression is proposed. The proposed decompressor can be easily placed on the root of scan chains. By using a periodically alterable MUX network, the number of visible scan chains is reduced, and tester channels are saved. The storage requirements of ATE are significantly reduced due to the high compression levels attained in this paper. It is viable and efficient for overcoming the storage gap between external tester and internal complex circuit.

The proposed test patterns compaction algorithm can be incorporated seamlessly with current DFT flow. In it, compaction procedure is independent with test generation. Thus, some commercial ATPG tools can be employed as test pattern generation engine and do the fault simulation. They can call each other through the interfaces programmed by some script languages.

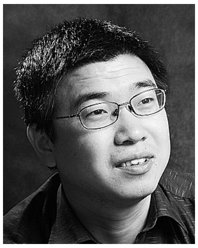
The proposed schemes do not require modifications to the test validation program. From the tester point of view, it behaves in a matter identical to regular scan applications. Reduction in required channels and storage enables utilization of low cost testers. Consequently, significant cost reduction in manufacturing tests can be achieved by incorporating the proposed compression schemes into the current design, test, and tester flows.

Acknowledgment

The authors would like to thank S. Swaminathan for providing two IBM PowerPC test data. They would also like to thank Dr. Q. Xu of the Chinese University of Hong Kong for his discussion during his visit to their laboratory. They would also like to thank Dr. G. Zhang and J. Dong of Institue of Computing Technology for their help in conducting the experiments using commercial EDA tools.

References

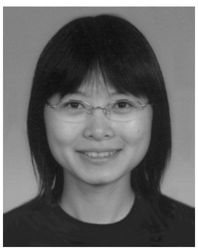
[1] ITRS, USA, "ITRS 2001 Edition," (2007). [Online]. Available: http://public.itrs.net
[2] W. Hu, F. Zhang, and Z. Li, "Microarchitecture of the Godson-2 processor," *J. Comput. Sci. Technol.*, vol. 20, no. 2, pp. 243–249, 2005.
[3] Y. Han, X. Li, H. Li, and A. Chandra, "Embedded test resource for SoC to reduce required tester channels based on advanced convolutional codes," *IEEE Trans. Instrum. Meas.*, vol. 55, no. 2, pp. 389–399, Apr. 2006.
[4] A. Chandra and K. Chakrabarty, "System-on-a-chip test data compression and decompression architectures based on Golomb codes," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 20, no. 3, pp. 113–120, Mar. 2001.
[5] A. Chandra and K. Chakrabarty, "Test data compression and test resources partitioning for system-on-a-chip using frequency-directed run-length (FDR) codes," *IEEE Trans. Comput.*, vol. 52, no. 8, pp. 1076–1088, Aug. 2003.
[6] M. Tehranipour, M. Nourani, and K. Chakrabarty, "Nine-coded compression technique for testing embedded cores in SoCs," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 13, no. 6, pp. 719–731, Jun. 2005.
[7] A. Jas, J. Gosh-Dastidar, M. Ng, and N. Touba, "An efficient test vector compression scheme using selective Huffman coding," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 22, no. 6, pp. 797–806, Jun. 2003.
[8] S. Chun, Y. Kim, J. Im, and S. Kang, "MICRO: A new hybrid test data compression/decompression scheme," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 14, no. 6, pp. 649–654, Jun. 2005.
[9] S. Reda and A. Orailoglu, "Reducing test application time through test data mutation encoding," in *Proc. DATE*, 2002, pp. 387–393.
[10] A. Jas, B. Pouya, and N. A. Touba, "Virtual scan chains: A means for reducing scan length in cores," in *Proc. VTS*, 2000, pp. 73–78.
[11] J. Rajski, J. Tyszer, M. Kassab, and N. Mukherjee, "Embedded deterministic test," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 23, no. 5, pp. 776–792, May 2004.
[12] C. V. Krishna, A. Jas, and N. A. Touba, "Reducing test data volume using LFSR reseeding with seed compression," in *Proc. ITC*, 2002, pp. 321–330.
[13] C. V. Krishna and N. A. Touba, "3-stage variable length continuous-flow scan vector decompression scheme," in *Proc. VTS*, 2004, pp. 79–86.
[14] I. Bayraktaroglu and A. Orailoglu, "Test volume and application time reduction through scan chains concelment," in *Proc. DAC*, 2001, pp. 151–155.
[15] W. Rao, I. Bayraktaroglu, and A. Orailoglu, "Test application time and volume compression through seed overlapping," in *Proc. DAC*, 2003, pp. 732–737.
[16] S. Mitra and K. S. Kim, "XPAND: An efficient test stimulus compression technique," *IEEE Trans. Comput.*, vol. 55, no. 2, pp. 163–173, Feb. 2006.
[17] I. Bayraktaroglu and A. Orailoglu, "Decompression hardware determination for test volume and time reduction through unified test pattern compaction and compression," in *Proc. VTS*, 2003, pp. 113–118.
[18] L. 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," in *Proc. ITC*, 2004, pp. 916–925.
[19] L. Li and K. Chakrabarty, "Test set embedding for deterministic BIST using a reconfigurable interconnection network," *IEEE Trans. Comput.-Aided Design Circuits Syst.*, vol. 23, no. 9, pp. 1289–1305, Sep. 2004.
[20] N. Sitchinava, S. Samaranayake, R. Kapur, E. Gizdarski, F. Neuveux, and T. W. Williams, "Changing the scan enable during shift," in *Proc. VTS*, 2004, pp. 73–78.
[21] I. Hamzaoglu and J. H. Patel, "Reducing test application time for full scan embedded cores," in *Proc. FTCS*, 1999, pp. 260–267.
[22] I. Hamzaoglu and . H. Patel, "Test set compaction algorithms for combinational circuits," in *Proc. ICCAD*, 1998, pp. 283–289.
[23] M. Sharma, J. H. Patel, and J. Rearick, "Test data compression and test time reduction of longest-path-per-gate test based on Illinois scan architecture," in *Proc. VTS*, 2003, pp. 15–21.
[24] J. C. Culberson, "Iterated greedy graph coloring and the difficulty landscape," Univ. Alberta, Edmonton, AL, Canada, Tech. Rep. TR-92-07, (1992). [Online]. Available: http://www.cs.ualberta.ca/ joe/Coloring/
[25] B. Bollobás, "The chromatic number of random," *Combinatorica 8*, vol. 1, pp. 49–56, 1988.
[26] B. Bollobás, *Random Graphs*, 2nd ed. Cambridge, U.K.: Cambridge Univ. Press, 2001.
[27] H. Kobayashi and L. R. Bahl, "Image data compression by predictive coding, part I: Prediction algorithm," *IBM J. Research Development*, vol. 18, p. 164, 1974.
[28] J. A Storer, *Data Compression: Method and Theory*. Rockville, MD: Comput. Sci. Press, 1988.
[29] H. K. Lee and D. S. Ha, "On the generation of test patterns for combinational circuits," Dept. Electr. Eng., Virginia Polytech. Inst. State Univ., Tech. Rep. 12.93, 1993.
[30] H. K. Lee and D. S. Ha, "An efficient parallel fault simulator," in *Proc. DAC*, 1992, pp. 336–340.
[31] L. Li, K. Chakrabarty, S. Kajihara, and S. Swaminathan, "Efficient space/time compression to reduce test data volume and testing time for IP cores," in *Proc. VLSI Design*, 2005, pp. 53–58.
[32] L. Li, K. Chakrabarty, S. Kajihara, and S. Swaminathan, "Three-stage compression approach to reduce test data volume and testing time for IP cores in SOCs," *Proc. Inst. Elect. Eng.—Comput. Digit. Tech.*, vol. 152, no. 6, pp. 704–712, 2005.



Yinhe Han (M'06) received the B.Eng. degree from Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 1997, and the M.Eng. and Ph.D. degrees in computer science from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2003 and 2006, respectively.

His research interests include VLSI/nanoelectronic/MEMs test, reliability design and development of DSP-based advanced digital instrumentation.

Dr. Han was a recipient of the Test Technology Technical Council Best Paper Award at the Asian Test Symposium 2003. He is a member of IEICE.



Yu Hu (M'06) received the B.S., M.S., and Ph.D. degrees, all in electrical engineering, from the University of Electronic Sciences and Technology, Chengdu, China, in 1997, 1999, and 2003, respectively.

She is currently an Associate Professor at the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China. Her research interests include design-for-reliability, design-for-testability, and development of EDA tools.

Dr. Hu is a member of IEICE.

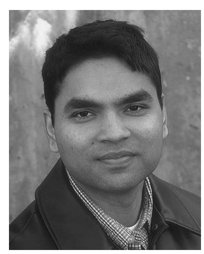
Xiaowei Li (M'00–SM'04) received the B.Eng. and M.Eng. degrees in computer science from Hefei University of Technology, Hefei, China, in 1985 and 1988, respectively, and the Ph.D. degree in computer science from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 1991.

He joined the Department of Computer Science and Technology, Peking University, Beijing, China, as a Postdoctoral Research Associate in 1991, and was promoted to Associate Professor in 1993. From 1997 to 1998, he was a Visiting Research Fellow in the Department of Electrical and Electronic Engineering, University of Hong Kong, Hong Kong. In 1999 and 2000, he was a Visiting Professor in the Graduate School of Information Science, Nara Institute of Science and Technology, Nara, Japan. He joined the Institute of Computing Technology, Chinese Academy of Sciences as a Professor in 2000. His research interests include VLSI/SoC design verification and test generation, design for testability, low-power design, and dependable computing.

Dr. Li was a recipient of the Natural Science Award from the Chinese Academy of Sciences in 1992 and the Certificate of Appreciation from IEEE Computer Society in 2001. He is an area editor of the *Journal of Computer Science and Technology* and an Associate Editor-In-Chief of the *Journal of Computer-Aided Design and Computer Graphics* (in Chinese).

Huawei Li (M'00) received the B.S. degree in computer science from Xiangtan University, Hunan, China, in 1996, and the M.S. and Ph.D. degrees from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 1999 and 2001, respectively.

She is currently an Associate Professor at the Institute of Computing Technology, Chinese Academy of Sciences. Her research interests include VLSI/SoC design verification and test generation, delay test, and dependable computing.



Anshuman Chandra (S'97–M'01) received the B.Eng. degree in electrical engineering from the University of Roorkee, Roorkee, India, and the M.S. and Ph.D. degrees in electrical and computer engineering from Duke University, Durham, NC, respectively.

Currently, he works as a Senior R&D Engineer at Synopsys, Inc., Mountain View, CA. His research interests include VLSI design, digital testing, and computer architecture.

Dr. Chandra was a recipient of the Test Technology Technical Council James Beausang Student Paper Award for a paper in PROCEEDINGS OF THE IEEE VLSI TEST SYMPOSIUM 2000 and a Best Paper Award from DATE 2001. He is a member of the ACM SIGDA.