Broadcasting Test Patterns to Multiple Circuits

Kuen-Jong Lee, Jih-Jeen Chen, and Cheng-Hua Huang

***Abstract*—Scan designs can alleviate test difficulties of sequential circuits by replacing the memory elements with scannable registers. However, scan operations usually result in long test application time. Most classical methods to solving this problem either perform test compaction to obtain fewer test vectors or use multiple scan chain design to reduce the scan time. For a large system, test vector compaction is a time-consuming process, while multiple scan chains either require extra pin overhead or need the sharing of normal I/O and scan I/O pins. In this paper, we present a novel test methodology that not only substantially reduces the total test pattern number for multiple circuits but also allows a single input data line to support multiple scan chains. Our main idea is to explore the "sharing" property of test patterns among all circuits under test (CUT's). By appropriately connecting the inputs of all CUT's during automatic test-pattern generation process such that the generated test patterns can be broadcast to all scan chains when the actual testing operation is executed, the above-mentioned problems can be solved effectively. Our method also provides a low-cost and high-performance method to integrate the boundary scan and scan architectures. Experimental results show that 157 test patterns are enough to detect all detectable faults in the ten ISCAS'85 combinational circuits, while 280 are enough for the ten largest ISCAS'89 scan-based sequential circuits.**

***Index Terms*— Design for testability, scan-based design, test compaction and boundary scan (IEEE 1149.1 Std.), test generation.**

I. Introduction

SCAN-based design is a structural design for test (DFT) technique that has been widely accepted in industry [1]. The basic idea of this method is to convert all or part of the internal registers of the circuit under test (CUT) into scan registers such that the controllability and observability of the CUT can be enhanced and the test generation complexity can be greatly reduced. However, it is well known that the test application time of a scan system is proportional to both the number of test patterns and the number of flip-flops in the scan chain(s). In a modern VLSI circuit, the number of internal flip-flops can be in the range of thousands or even higher. Hence, how to reduce test application time has become an important issue when a scan-based design is used.

Multiple scan chain techniques have been developed to alleviate the long test application time problem [2]–[6]. By dividing a single serial scan chain into a number of shorter scan chains, test patterns (results) can then be shifted into (out of) all scan chains in parallel to reduce the test application time. This method, however, will require a large number of extra I/O pins if a pair of I/O pins is used for each scan chain. One may share a normal I/O pin with a scan I/O pin such that the pin can be used for test or normal operation depending on the required functional mode. This, however, will result in extra loading on the I/O of the CUT and hence may introduce further performance degradation that already exists in a scan-based system.

Moreover, currently boundary scan has become a standard for board-level testing [7], [8]. Unfortunately, the boundary scan architecture allows only one pin for test data input and another one for data output, and hence cannot efficiently support multiple scan chains. One may use a demultiplexer to distribute the test patterns from the single input line to multiple scan chains. Clearly, this will give up the most important advantage of the multiple scan chain technique, i.e., reducing test application time via parallel loading of test patterns.

In this paper, we propose a novel method that allows one single data input to support multiple scan chains for multiple circuits while the test application time is still very close to that for the conventional multiple scan method. The key idea is to consider all the circuits driven by all scan chains as a single circuit when executing the automatic test-pattern generation (ATPG) process. A novel circuit model called the *virtual circuit* is developed for ATPG to generate common test vectors that are effective for all CUT's. The generated test patterns can then be "broadcast" to all circuits simultaneously, and hence the test application time can be greatly reduced. It is shown that there exists an exponential number of different virtual circuits, and hence heuristic methods are required to construct a "good" virtual circuit with which the size of the generated test set can be nearly minimized. In this paper, we shall propose four heuristic methods for combinational circuits and two for sequential circuits. For combinational circuits, we assume that there exists one flip-flop for each input of each circuit so that all the flip-flops to each circuit can be chained together to form a scan chain. This is generally true for any embedded combinational circuit inside a chip, and is certainly true for a chip with the boundary scan architecture. For sequential circuits we may have two cases: either only the internal registers are scanned or both primary inputs and internal registers are scanned. For the former case, we only need an input line for all the scan registers. For the latter case, our method can either use one input line to support all scan chains or use one line to provide test data to all inputs and another for the internal registers.

To verify the effectiveness of the proposed techniques, experiments on the ISCAS'85 combinational benchmark circuits [9] and the ISCAS'89 sequential benchmark circuits [10] have been made. The results show that for combinational

Manuscript received June 14, 1999. This work was supported by the National Science Council of Taiwan, R.O.C., under Contract NSC-87-2215-E-006-011. This paper was recommended by Associate Editor K.-T. Cheng.

The authors are with the Department of Electrical Engineering, National Cheng Kung University, Tainan, Taiwan 70101 R.O.C.

Publisher Item Identifier S 0278-0070(99)09918-2.

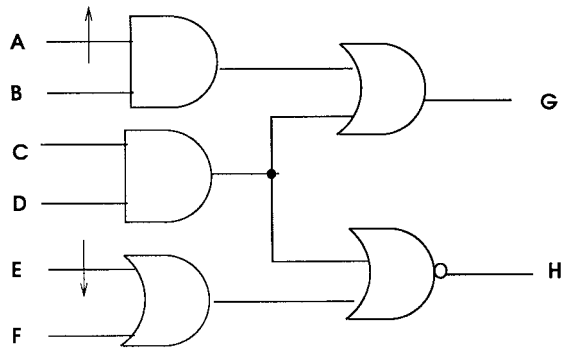


Fig. 1. Assigning ABCDEF $= 010x10$ can detect both A SA-1 and E SA-0 faults.

circuits even very simple heuristic will result in great test application time reduction, while more involving methods will give even better results. It is found that we only need 157 test patterns to detect all detectable faults in all ten ISCAS'85 combinational circuits. For the sequential circuits, even more surprising results are obtained. We show that with only some simple heuristic, 280 test patterns are enough for the ten largest ISCAS'89 scan-based sequential circuits, which contain more than 100 000 gates in total.

The small size of test sets clearly leads to significant reduction in both test application time and memory space required to store the test patterns. In addition, we shall show that the hardware implementation is quite easy and that the hardware overhead is about the same as that of a single scan design. Furthermore, since the ATPG is carried out for all circuits simultaneously, the test generation time is usually shorter than the sum of those for all circuits once the virtual circuit is constructed. The time required to construct a virtual circuit depends on the heuristic used. For the simple heuristic methods, this time is negligible. Yet another major contribution of this paper is its capability to allow boundary scan to efficiently support multiple scan test structure, which clearly will significantly reduce the difficulty of the system-on-a-chip (SOC) testing problem.

The remainder of this paper is organized as follows. In the next section, we describe the basic concepts of test set compaction and broadcasting. In Section III, we present the virtual circuit model for ATPG process. The heuristic methods to construct virtual circuits are described in this section as well. Section IV discusses the hardware implementation for the proposed techniques. Experimental results are given and discussed in Section V. Section VI concludes this paper.

II. Basic Concepts of Test-Pattern Broadcasting

It is well known that when generating a pattern for a specific fault in a CUT using a PODEM-like algorithm [11], usually only a subset of the primary inputs need to be specified. The *don't care* bits that appear in the pattern can be further assigned some specific values to detect more faults. For examples, the fault A stuck at one in Fig. 1 can be detected by ABCDEF $= 010xxx$. If another fault E stuck at zero is to be detected, then we can further assign ABCDEF $= 010x10$. This "test compaction" concept has been used in some test-generation algorithms, and it is shown that both test-generation time and test-pattern size can be reduced [12], [13].

Test compaction can also be done after a set of test patterns has been generated. The basic idea here is to explore the compatibility among the generated test patterns and try to replace them with a new set of test patterns that has smaller size but still covers all faults that are detected by the original test set [14]. Previous test compaction techniques, however, only focus on generating a small test set for a single circuit. In this paper, we extend this concept to multiple circuits that have their own inputs and scan registers. We shall show that significant reduction of test set size can be achieved by considering not only the compatibility of test patterns but also the feature of an ATPG process in generating both random and deterministic patterns.

We use two circuits CUT(1) and CUT(2) to illustrate the basic concept of multiple circuit test compaction and generation. Assume that these two circuits have their own test sets $T_1 = \langle t_{11}, t_{12}, \ldots, t_{1k} \rangle$ and $T_2 = \langle t_{21}, t_{22}, \ldots, t_{2l} \rangle$, respectively. In the beginning of an ATPG process, usually random patterns are used until a specified fault coverage, say, 85%, is reached. If the same random pattern generation process is used when generating T_1 and T_2, then it is likely that $t_{11} = t_{21}$, $t_{12} = t_{22}, \ldots, t_{1i} = t_{2i}$ up to some i (for simplicity, assume that the two circuits have the same number of inputs). Since most faults have been detected by the first i patterns, each of the remaining test patterns generated by deterministic ATPG may be needed only for a small number of faults. Therefore, it is likely that these patterns may have many don't care bits. For example, when generating $t_{1(i+1)}$, many don't care bits may still exist when no more faults in CUT(1) can be detected. If we use the test pattern with bits assigned so far for faults in CUT(2), then in addition to those faults that can be detected by the current pattern, we can further assign specific values to the don't care bits in the pattern to detect more faults. We can expect that if CUT(1) and CUT(2) are independent, then the don't care bits that cannot be further assigned to detect more faults in CUT(1) now may be assigned some values to detect more faults in CUT(2).

With the concept presented above, we know that either random patterns or deterministic patterns can be shared by both CUT(1) and CUT(2). This gives us a totally new idea to deal with the multiple scan chain problem as explained below. If we take into account the requirement of these two circuits driven by all scan chains simultaneously when generating tests, then the same test patterns will be effective for both CUT(1) and CUT(2). Therefore, when the actual test application procedure is carried out, we can simply "broadcast" the same patterns to them. This also allows us to use one single data line to provide test patterns to both circuits. Clearly, the same concept can be extended to deal with multiple circuits.

III. Virtual Circuits Construction for ATPG

To implement the concept presented in Section II, we have to reconfigure all the circuits under test into a single circuit before the ATPG process. We call the resulting circuit the "virtual circuit." It should be pointed out that physically the virtual circuit does not exist. It is just for the ATPG process to

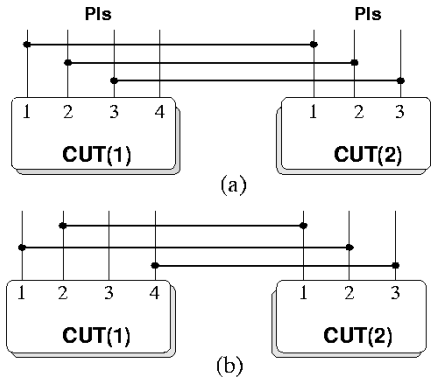


Fig. 2. Examples of two virtual circuits: (a) i-to-i connection and (b) random connection.

generate common test vectors that are effective for all test circuits such that complete stuck-at fault coverage can be achieved. The actual hardware configuration will be presented later in Section IV.

A. General Structure

In this paper, we mainly consider test compaction among multiple circuits that are independent, i.e., each circuit has its own inputs and any interconnect among these circuits, if it exists, can be broken by scan registers. We first show how to construct a virtual circuit from two combinational CUT's, CUT(1) and CUT(2). Assume that the numbers of primary inputs of CUT(1) and CUT(2) are N_1 and N_2, respectively. Without loss of generality, assume $N_1 \geq N_2$. We select N_2 pins from the N_1 pins of CUT(1) and connect them to the N_2 pins of CUT(2). The pin connection is in a 1-1 mapping manner, i.e., different pins in CUT(2) are connected to different pins in CUT(1). It is easy to verify that there exist $N_1!/(N_1-N_2)!$ different configurations based on CUT(1) and CUT(2). We call each configuration a virtual circuit of CUT(1) and CUT(2). Fig. 2 shows two possible virtual circuits of two CUT's with four and three inputs, respectively. In Fig. 2(a), the ith pins of CUT(1) and CUT(2) are connected, $i = 1, 2, 3$, while in Fig. 2(b), the connection is more random.

Similar construction can be extended to more circuits. Let CUT(1), CUT(2), ..., CUT(k) be k ($k \geq 2$) circuits with N_1, $N_2, \ldots, N_k$ inputs, respectively, where $N_1 = \max_{i=1,\ldots,k} N_i$. Then we can connect the N_2 pins of CUT(2) to some N_2 pins of CUT(1), the N_3 pins of CUT(3) to some N_3 pins of CUT(1), and so on. There are totally $(N_1!/(N_1-N_2)!) \times (N_1!/(N_1-N_3)!) \times \cdots \times (N_1!/(N_1-N_k)!)$ different connections among the k circuits.

After a virtual circuit is constructed, we can then apply an ATPG process to the circuit, and the resulting test patterns should be effective for all original circuits. Since only the inputs of all circuits are connected together and the outputs of all circuits are not changed, the test patterns generated can detect all detectable faults in all circuits.

B. Heuristic Methods

As mentioned before, there are totally $(N_1!/(N_1-N_2)!) \times (N_1!/(N_1-N_3)!) \times \cdots \times (N_1!/(N_1-N_k)!)$ virtual circuits for k CUT's with $N_1, N_2, \ldots, N_k$ inputs, respectively. In the current LSI/VLSI technology, the numbers of system primary inputs can be quite large. Hence how to select a virtual circuit such that the number of generated test patterns is minimum becomes the most important problem to be addressed. This is clearly an NP-complete problem because even the test compaction problem for a single combinational circuit is NP-complete [15]. Therefore, heuristic methods must be employed. In this paper, we shall present four selection methods for combinational circuits and two for sequential circuits. The first two for combinational circuits are quite simple and straightforward, the third one requires the knowledge of signal probability (i.e., the probability of being zero or one) on each input, and the fourth one is more or less an exhaustive method. The two methods for sequential circuits can be considered as the extensions of the first two methods for combinational circuits.

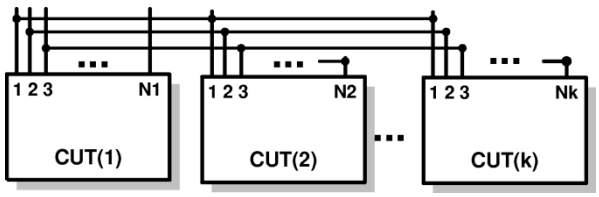


Fig. 3. The first ATPG connection method.

1) Method$_1$—1-1 In-Order Mapping Method: The first virtual circuit construction method for combinational circuits is shown in Fig. 3, which simply connects the N_i inputs, $i = 2, \ldots, k$ of each CUT(i) circuit to the first N_i pins of N_1 such that all the first pins of each circuit are connected together, all the second pins of each circuit are connected together, and so on. This method is very simple and easy to implement. However, it has the disadvantage that the "burden" of the pins with small indexes is greater than that of the pins with large indexes because the first $N_{\min}$ bits of the generated test pattern bits must be applied to all circuits, while those for the last $N_1 - N_{\max}$ pins are only applied to CUT(1), where $N_{\min} = \min_{i=2,\ldots,k} N_i$ and $N_{\max} = \max_{i=2,\ldots,k} N_i$.

2) Method$_2$—Even Distribution Method: The second selection method tries to "evenly distribute" the connection among the N_1 pins of CUT(1). We connect the N_2 pins of CUT(2) to the first N_2 pins of CUT(1). Then, we search the remaining circuits to see whether any one can fit the last $N_1 - N_2$ pins of CUT(1). If one exists, say, CUT(i), then we connect the N_i pins of CUT(i) to the $N_2+1, \ldots, N_2+N_i$ pins of CUT(1). This process continues until no circuits can fit the remaining pins of CUT(1). Then for the remaining circuits, we resume the above connection method from the first pin of CUT(1). Fig. 4 shows an example in which the inputs of eight CUT's are configurated into four "scan chain"-like structures.

3) Method$_3$—Nearest Signal Probability Matching Method: In this heuristic method, we employ the one-state or zero-state probability of each bit in the test set of each individual circuit to guide the construction of the virtual circuit. The basic idea is to connect the pins that have relatively closest probabilities of logic one or zero such that any two pins that will share the same test data should have similar one or zero

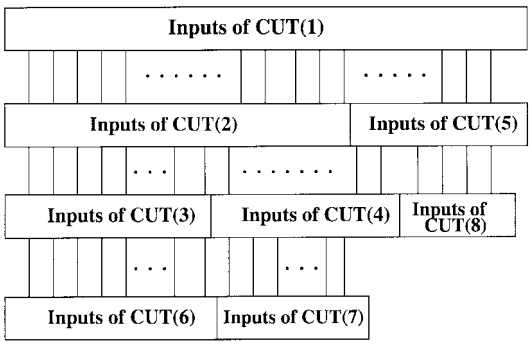


Fig. 4. The second ATPG connection method.

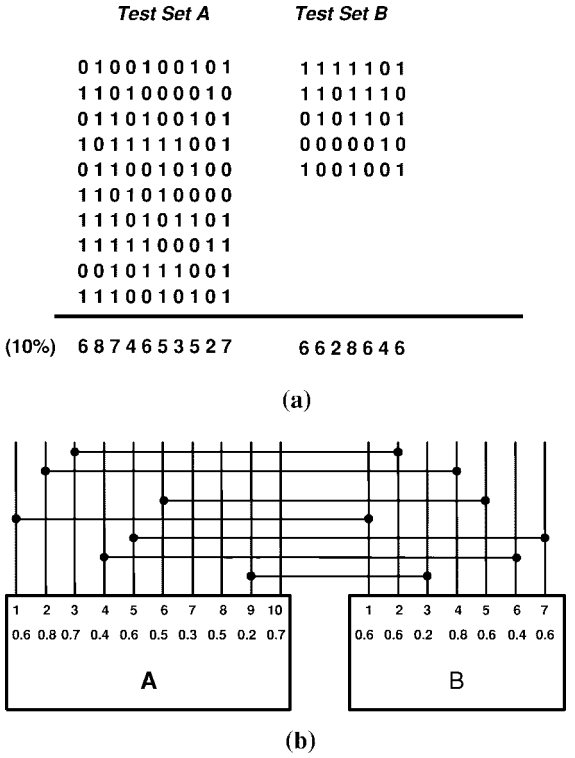


Fig. 5. (a) Test sets of circuits A and B and (b) the third ATPG connection method.

probability. In Fig. 5(a), we show an example to construct the virtual circuit of two circuits A and B by this method. Assume that we have found the test sets for circuits A and B, as shown in Fig. 5(a), where test sets A and B consists of ten and five vectors, respectively. We can compute the one-probability of each pin for A and B, as shown in the bottom of Fig. 5(a). Then we simply connect two pins between A and B that have closest one-probability. For example, pin 1 of A is connected to pin 1 of B, pin 2 of A is connected to pin 4 of B, and so on. Clearly, this method can also be applied to k $(k>2)$ circuits. With this method, it is likely that when the ATPG process is applied to the constructed virtual circuit, the number of required test patterns is smaller than those obtained from the first two heuristic methods.

4) Method$_4$—In-Order Pseudoexhaustive Method: In the above-mentioned three methods, only one virtual circuit is constructed for each method. Therefore the ATPG process

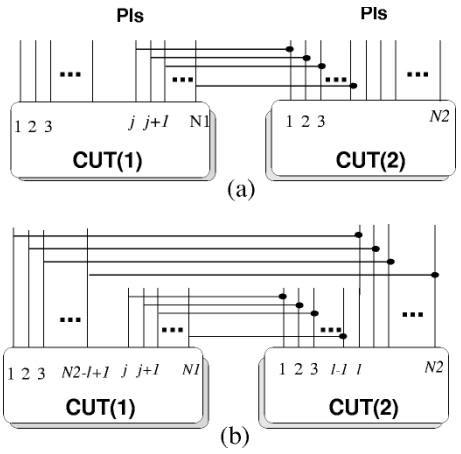


Fig. 6. The fourth ATPG connection method: (a) no circular connection and (b) circular connection.

can be efficiently carried out. However, not too much "optimization" has been done to identify a test set that has the minimum size. In the fourth method, we attempt to use a "pseudoexhaustive" method to find the minimum test set. We call it pseudoexhaustive because it is exhaustive only in some aspects as explained below.

The basic idea of this method is to extend the possible construction of *Method*$_1$ such that when a circuit CUT(i) is connected to the circuit CUT(1), the connection can be started with any pin of CUT(1) rather than just the first pin of CUT(1). In other words, we will construct N_1 virtual circuits when dealing with any CUT(i). In the first such virtual circuit, the N_i pins of CUT(i) are connected to the first N_i pins of CUT(1), just like *Method*$_1$. In the second such virtual circuit, the N_i pins of CUT(i) are connected to the 2, 3, ..., N_i+1 pins of CUT(1). In the general case, the jth virtual circuit is constructed by connecting the N_i pins of CUT(i) to the $j, j+1, \ldots, N_i+j-1$ pins of CUT(1). Note that when $N_i+j-1>N_1$, some pins of CUT(i) will exceed the pin range of CUT(1). In this case either the connection is considered as invalid or the connection can be in a "circular" form as shown in Fig. 6, where pins $l, \ldots, N_2$ of CUT(2) are connected to pins 1, 2, ..., N_2-l+1. In this paper, we shall use both methods in the experiments, but for simplicity in the following discussion we will only describe the case for the later method.

The virtual circuit is constructed in an "incremental" manner. We first construct a virtual circuit containing CUT(1) and CUT(2), which is then used as the new CUT(1) when CUT(3) is considered. Then CUT(4) is combined with the virtual circuit containing CUT(1), CUT(2), and CUT(3). This process continues until CUT(k) is included. Among the N_1 virtual circuits when processing CUT(i), we will apply ATPG to each of them and select the one that leads to the minimum test set. That means we have to carry out N_1 ATPG processes for each $i, i = 2, \ldots, k$. Therefore, we have to carry out $N_1 \times (k-1)$ ATPG processes in total. The final ATPG process for the virtual circuit that contains all individual circuits will give the required test patterns for all circuits.

The above method clearly will take much preprocess time before the final virtual circuit can be identified. However, since

this is a one-time work, i.e., once the patterns are identified, they can be used again and again, it may be worthwhile if the final test set size can be greatly reduced.

5) Sequential Circuits: The construction of virtual circuits for sequential circuits is similar to that for combinational circuits. The only difference is that for sequential circuits, we may have to consider the primary inputs and the internal flip-flops separately. For a circuit whose primary inputs are directly connected to the input pins of the chip containing the circuit, only the internal flip-flops needed to be considered, while for a circuit whose primary inputs are also driven by flip-flops, we may assume that these flip-flops can be configured into a scan chain.

Once all flip-flops to be scanned for each circuit are identified, we can use the same methods as for combinational circuits to construct the virtual circuit. Then the ATPG process can be executed on the virtual circuit to obtain the test patterns that are effective for all circuits.

In this paper, we shall show that by just using a simple heuristic (*Method*$_1$ and *Method*$_2$), extremely good results can be obtained for the ISCAS'89 benchmark circuits.

IV. Hardware Configuration

As mentioned in the last section, the virtual circuit is just for the ATPG process. Now we describe the hardware implementation of the proposed techniques. Again, we will describe the implementation for combinational circuits first, followed by that for sequential circuits.

A. Scan Architecture for Combinational Circuits

Since all circuits will receive the same patterns, we can use one single data line to broadcast test patterns to all circuits. But the configuration of the scan chains will be dependent on the selection of virtual circuits. In Fig. 7(a), a general scan configuration based on the first method for combinational circuits presented in Section III-B1 is given. In this figure, all circuits under test will receive the same test patterns through scan input. For each pattern, after N_1 shift operation, the entire pattern will appear in the scan chain of CUT(1), or SC_1, and the first N_i bits of the pattern (i.e., the last N_i bits that are scanned) will appear in the scan chain of CUT(i), or SC_i for $i = 2, \ldots, k$. Hence each CUT will receive its required pattern.

Up to now we have only dealt with how to broadcast test patterns to all circuits to save the test time. The examination of the test results also needs to be efficiently carried out. Otherwise, if the test results were to be shifted out just like the conventional scan methods, then our proposed technique would be meaningless. In this work, we solve this problem by compressing the test results using a multiple input signature register (MISR) as shown in Fig. 7(a). With this method, our architecture then becomes complete, and we are able to achieve the goal of using one single input line to support multiple scan chains for multiple circuits.

It should be pointed out that there may be some variations on the result compression scheme. Fig. 7(b) shows an example that uses a number of smaller and individual MISR's instead of a single large MISR. With this architecture, each CUT has his own signature analyzer to compress the test response and

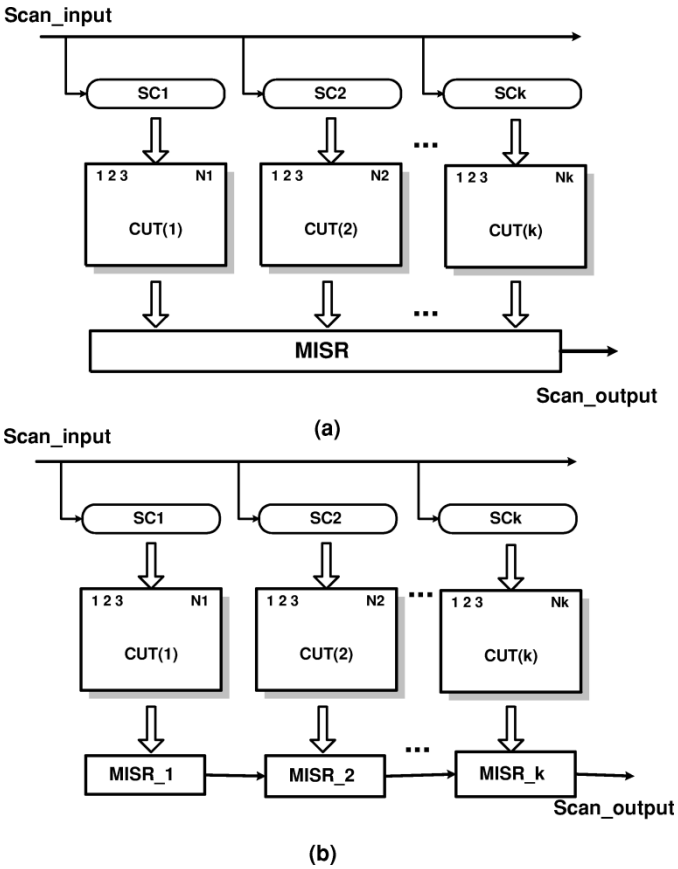


Fig. 7. Scan configuration with first selection method. (a) A common MISR and (b) individual and multiple MISR's.

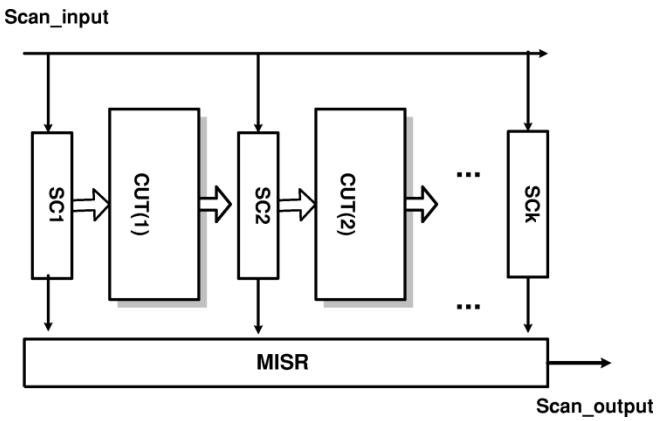


Fig. 8. Broadcasting test patterns using a STUMP-like structure.

hence the global feedback delay due to the MISR is smaller. In addition, each MISR can be placed next to its corresponding CUT so as to save the routing area.

In the case where the outputs of some circuits are on the scan chains (e.g., a pipeline structure), we may use a test structure similar to that used in the STUMPS structure [16], [17]. As shown in Fig. 8, the scan paths are driven in parallel by the scan input, and the signature is generated in parallel from each scan path using a MISR. With this architecture, the required number of flip-flops in the MISR can also be reduced.

Fig. 9 shows the scan chain architecture corresponding to the virtual circuit given in Fig. 4 using our second virtual

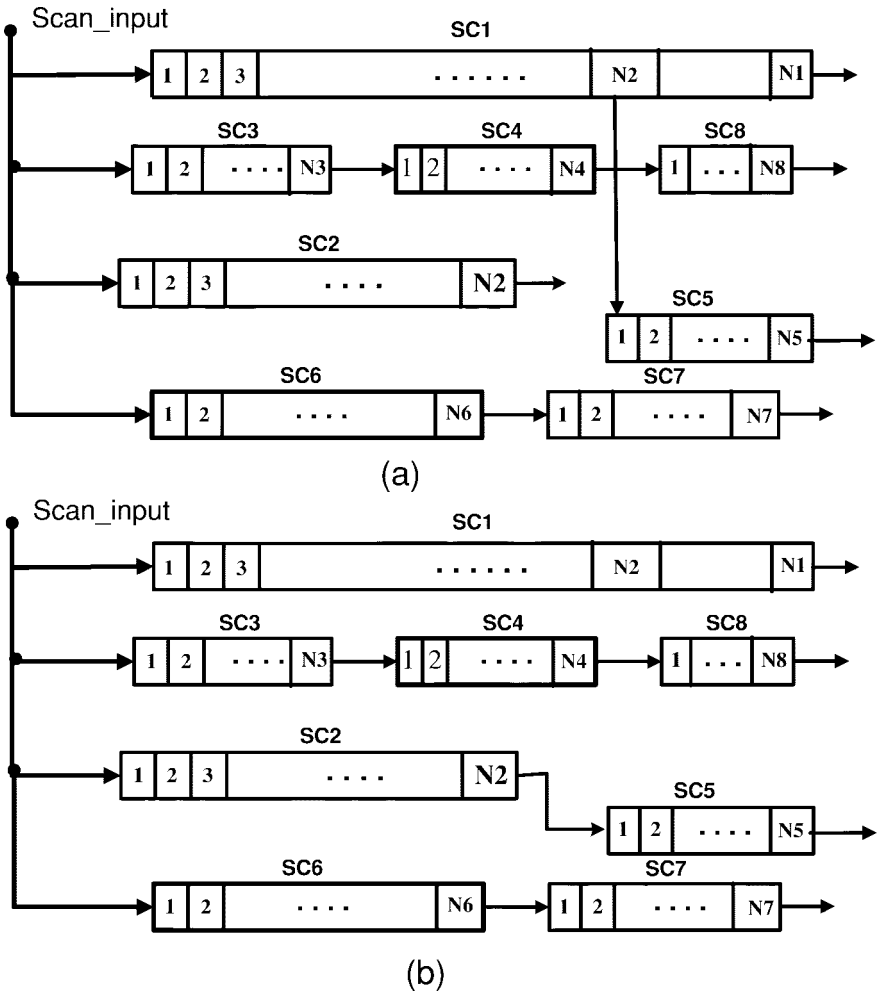


Fig. 9. The scan architecture of the second virtual circuit with different connection position. (a) Longer and (b) shorter routing length.

circuit selection method. For brevity, only the scan chains are shown. Note that depending on the actual layout, the connection among all scan registers can be adjusted to optimize the routing area. For example, assume that the first pin of CUT(5) is connected to the N_2+1 pin of CUT(1) in the virtual circuit. In the actual implementation, if the layout distance between CUT(5) and CUT(1) is longer than that between CUT(5) and CUT(2), then the input to the first scan cell of SC_5 can be connected to the output of the last scan cell of SC_2, as shown in Fig. 9(b).

The hardware configuration for the third method *Method*$_3$ can be similar to that for *Method*$_1$ except that the flip-flops in the scan registers must be reordered so that each bit of the test patterns can arrive at the right place simultaneously for each CUT. This configuration, however, may be difficult to implement for some data-path circuits such as a multiplier. In Fig. 10, we show another possible implementation, which actually can save large hardware overhead. Here we only use one scan register for all CUT's. Each output of the scan register is then fanout to all required inputs of CUT's. With this structure, we only need N_1 scan flip-flops, at the expense of routing area and a multiplexer for each input pin (to provide the normal or test mode). Clearly, this method can also be used for *Method*$_1$ and *Method*$_2$.

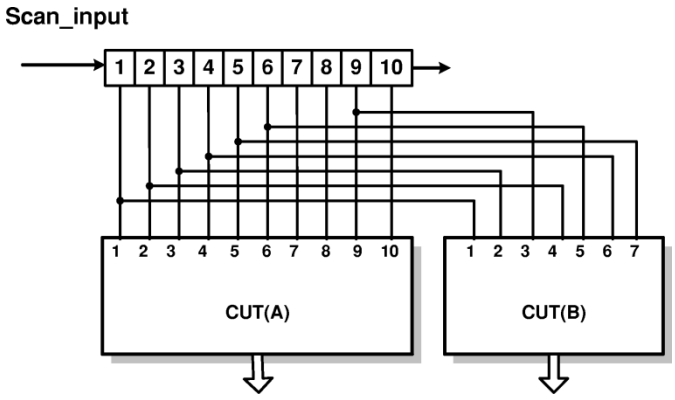


Fig. 10. The scan chain structure of the Fig. 5.

Although the construction procedure of the virtual circuit for *Method*$_4$ is quite complex, the hardware implementation for this method is quite simple. Fig. 11 shows the scan structure for two CUT's, CUT(1) and CUT(i). Assume that the first pin of CUT(i) is connected to the jth pin of CUT(1) in the virtual circuit. Then the input of the first flip-flop of SC_i should be connected to the output of the $(j-1)$th flip-flop of SC_1. Note that it is possible that $N_i-j+1>N_1$. In this case, the input of

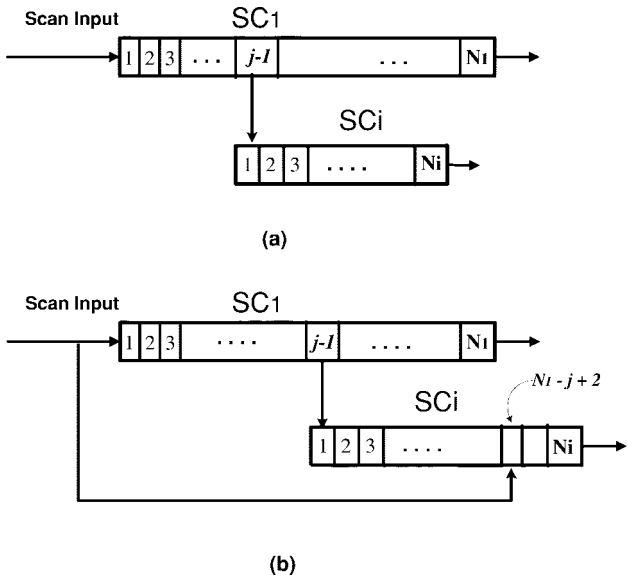


Fig. 11. Scan architectures of the fourth method: (a) $N_i - j + 1 \leq N_1$ and (b) $N_i - j + 1 > N_1$.

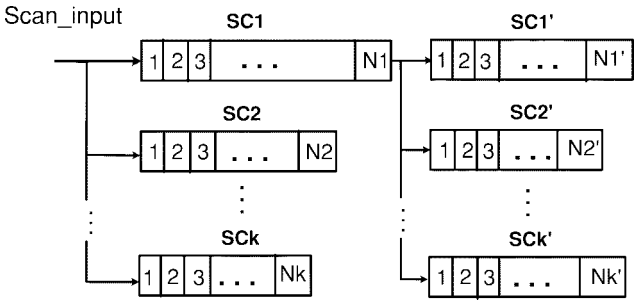


Fig. 12. A scan architecture for sequential circuits using one scan input.

the $(N_1 - j + 2)$th flip-flop of SC_i should be directly connected to the scan input, as shown in Fig. 11(b).

B. Scan Architecture for Sequential Circuits

As described in the beginning, we may have two cases for scan-based design depending on whether the primary inputs are scanned or not. When the primary inputs are directly connected to the input pins of the chip containing the circuit, only the internal scan chains needed to be constructed. In this situation, the internal scan chain architectures for sequential circuits are similar to those for combinational circuits, and all CUT's will receive the same test patterns through only one input line. In the case where the primary inputs of the CUT's are also driven by flip-flops, we can either use two data lines to broadcast test patterns to the input scan chains and the internal scan chains of all circuits, respectively, or use only one input scan to broadcast patterns to all scan chains simultaneously. Clearly, all the scan architectures used for the combinational circuits can also apply to sequential circuits.

Fig. 12 shows yet another scan chain architecture for sequential circuits that will be used in our experiments, where SC_i and SC_i' are input and internal scan chains, respectively. By connecting the common input line of internal scan chains to the scan output of SC_1, where SC_1 has the largest primary inputs number (N_1) for all CUT's, it is easy to achieve the goal of using one single input line to support all scan registers. The test patterns are applied after shifting in the vector by clocking the serial scan chains for $N_1 + N_1'$ cycles, where N_1' is the length of the longest internal scan chain among all CUT's.

V. Experimental Results and Discussion

We use the ten ISCAS'85 combinational circuits and the ten largest circuits of ISCAS'89 (S1488 is not included because it is the same as S1494) in our experiments. A commercial ATPG tool is used to generate test patterns, and all detectable single stuck-at faults are targeted. All experiments are performed on a SUN UltraSparc workstation.

Table I shows the test-generation results for each individual circuit in the ISCAS'85 benchmarks. We find that totally 738 test patterns are required to detect all 31 835 (32 342 − 507) detectable faults in the ten ISCAS'85 benchmark circuits. The total test generation time is 163.2 s. If all of these circuits are put into the same chip and all their inputs are connected into a single scan chain, then we will need about $834 \times 130 = 108\,420$ clock cycles to apply all patterns to all circuits, where 834 is the total numbers of inputs of all circuits and 130 is the largest number of required test patterns among all circuits (C3540). On the other hand, if multiple scan chains are used, then the test application time will be $206 \times 130 = 26\,780$ clock cycles (assume that only one test session is used [4]), where 206 is the largest number of inputs (C7552) and 130 is the largest number of test patterns for all circuits. The above two results are given in the second and third columns of Table II.

The remaining columns of Table II show the results of the four heuristic methods presented in Section III-B. There are two cases for *Method*$_4$ as described in Section III-B4. Totally 195, 177, 167, 158, and 157 test patterns are required to detect all faults in the ten ISCAS'85 circuits using the five methods, respectively. Clearly, the number of required test patterns is significantly smaller than the total number of patterns required for all circuits (738). The test generation times of *Method*$_1$ and *Method*$_2$ (122.2 and 130.3 s) are also less than the sum of those for all individual circuits (163.2 s). The *Method*$_3$ and *Method*$_4$ have to spend more preprocesses time in calculating the signal probability and in applying ATPG for different virtual circuits, respectively. For comparison purpose, the test-generation times shown in Table II for *Method*$_3$ and *Method*$_4$ are just for the final virtual circuits that contain all circuits. The test application times are 40 170 (206 × 195), 36 462 (206 × 177), 34 402 (206 × 167), 32 548 (206 × 158), and 32 342 (206 × 157) cycles for these methods, which are about 37.0%, 33.6%, 31.7%, 30.0%, and 29.8% of the single scan chain method and 150.0%, 136.1%, 128.4%, 121.5%, and 120.7% of the multiple scan method, respectively. The required test pattern sizes can be calculated by the test vector width times the number of test patterns in the test set, which are 40 170 (206 × 195), 36 462 (206 × 177), 34 402 (206 × 167), 32 548 (206 × 158), and 32 342 (206 × 157) bits, respectively. They are 61.2%, 55.6%, 52.5%, 49.6%, and 49.3% of the sum of all individual circuits (65 534 bits), respectively.

TABLE I
INDIVIDUAL TEST GENERATION RESULTS OF ISCAS'85 CIRCUITS

Circuits	# PI/PO	# Faults	# Gates	# RF	# TP	TSS (bits)	TG Time (sec)	FC (%)	TE (%)
C432	36/7	524	160	4	43	1548	2.4	99.24	100
C499	41/32	758	202	8	54	2214	2.5	98.94	100
C880	60/26	942	383	0	35	2100	2.7	100	100
C1355	41/32	1574	546	8	86	3526	8.3	99.49	100
C1908	33/25	1879	880	9	117	3861	18.2	99.52	100
C2670	157/64	2595	1193	117	71	11147	12.2	95.49	100
C3540	50/22	3428	1669	137	130	6500	39.2	96	100
C5315	178/123	5350	2307	59	76	13528	17.1	98.90	100
C6288	32/32	7744	2416	34	29	928	31.8	99.56	100
C7552	206/107	7548	3512	131	97	19982	28.8	98.26	100
Total	834/470	32342	13268	507	738	65334	163.2	97.71	100

PI/PO: Primary Input/Output, RF: Redundant Faults, TP: Test Patterns, TSS: Test Sets Sizes, TG Time: Test Generation Time, FC: Fault Coverage, TE: Test Efficiency.

TABLE II
EXPERIMENTAL RESULTS FOR ISCAS'85 CIRCUITS

	Single	Multiple	$Method_1$	$Method_2$	$Method_3$	$Method_4$	
						noncircular	circular
Test Efficiency(%)	100%	100%	100%	100%	100%	100%	100%
# Test Patterns	738*	738*	195	177	167	158	157
Scan Chain Length	834	206	206	206	206	206	206
TG Time (sec)	163.2	163.2	122.2	130.3	127.7**	114.7**	112.4**
TA Cycles	108420	26780	40170	36462	34402	32548	32342
Test Set Sizes (bits)	65534	65534	40170	36462	34402	32548	32342
Normalized TA Cycles	4.048	1	1.500	1.361	1.284	1.215	1.207
	1	0.247	0.370	0.336	0.317	0.300	0.298
Normalized Test Set Size	1	1	0.612	0.556	0.525	0.496	0.493

TG Time: Test Generation Time, TA Cycles: Test Application Cycles.
* The total number of test patterns for all circuits.
** Not including preprocess time.

TABLE III
INDIVIDUAL TEST GENERATION RESULTS OF ISCAS'89 CIRCUITS

Circuits	# PI/PO	#Gates	# FF	# Faults	# RF	# TP	TG Time (sec)	FC (%)	TE (%)
S1238	14/14	598	18	1355	69	149	9.9	94.91	100
S1423	17/5	906	74	1515	14	43	7.4	99.08	100
S1494	8/19	735	6	1506	12	117	4.9	99.20	100
S5378	35/49	3400	179	4603	40	134	29.7	99.13	100
S9234	19/22	6326	228	6927	452	159	88.2	93.47	100
S13207	31/121	10167	669	9815	153	281	200.4	98.44	100
S15850	14/87	11739	597	11727	391	158	239.1	96.67	100
S35932	35/320	21903	1728	39094	3984	28	164.1	89.81	100
S38417	28/106	27379	1636	31180	165	113	224.9	99.47	100
S38584	12/278	24173	1452	36305	1506	174	324.7	95.85	100
Total	213/1021	107326	6587	144027	6786	1356	1293.3	95.28	100

PI/PO: Primary Input/Output, FF: Flip-Flops, RF: Redundant Faults, TP: Test Patterns, TG Time: Test Generation Time, FC: Fault Coverage, TE: Test Efficiency.

For the sequential benchmark circuits, we have done two experiments. In the first experiment only the flip-flops of the circuits are chained together, and in the second all flip-flops and all primary inputs are chained together. The results for individual circuit process are given in Table III, where we find that totally 137 241 (144 027 − 6786) faults can be detected by 1356 patterns, and the total test-generation time is 1293.3 CPU s. The results of our methods and their comparison with single and multiple scan schemes are given in Table IV. The meanings of the second and third columns are the same as those in Table II, where we can see that the total test application times are 1 850 947 and 485 568 cycles for single and multiple scan chains, respectively. The data are obtained by assuming that the primary inputs of all CUT's can directly receive test patterns from the outside tester without shifting operations.

The fourth and fifth columns respectively show the results of $Method_1$ without and with PI's connected together. For the latter case, only one input line for all scan registers is used, hence the shifting time per test vector (i.e., scan chain length) is longer than the former case. Similarly, the sixth and seventh columns are for $Method_2$. We can see that the number

TABLE IV
EXPERIMENTAL RESULTS FOR ISCAS'89 CIRCUITS

	Single	Multiple	$Method_1$		$Method_2$	
			FFs	FFs & PIs	FFs	FFs & PIs
Test Efficiency(%)	100	100	100	100	100	100
# Test Patterns	1356*	1356*	287	294	280	285
Scan Chain Length	6587	1728	1728	35+1728	1728	35+1728
TG Time (sec)	1293.9	1293.9	1802.0	1820.1	1893.7	1869.2
TA Cycles	1850947	485568	495936	518322	483840	502455
Normalized	3.812	1	1.021	1.067	0.996	1.034
TA Cycles	1	0.262	0.268	0.280	0.261	0.271

TG Time: Test Generation Time, TA Cycles: Test Application Cycles.
* The total number of test patterns for all circuits.

of test patterns for the case that PI's are not connected is also smaller than the case that PI's are connected for both methods. This is expected because in our method, when a pin is connected to another pin, the freedom to assign any logic value to it will reduce. An interesting result is that the number of test patterns using $Method_2$ with PI's not scanned is even smaller than the largest number of patterns required for a single circuit S13207, and hence the test application time using this method is even shorter than that using multiple scan chains. This can be explained as follows. Since the number of test patterns required for S13207 is much larger than the maximum of those required for other circuits (281 versus 174), and the number of FF's of S13207 is much smaller than the maximum number of FF's of other circuits (669 versus 1728), it is likely that the number of patterns required for S13207 will dominate the final total number of patterns for all circuits. Also since in general test compaction is NP-complete and can only be dealt with using heuristic methods, it is possible that the final total number of test patterns for all circuits is slightly smaller than that required for the single "dominating" circuit. For the combinational ISCAS'85 circuits, the difference of the numbers of test patterns between the "dominating" circuit (C3540) and other circuits is not as great as that for ISCAS'89. Hence the number of final total number of test patterns for all circuits (157) is large compared to the largest number of test patterns (130) required for a single circuit (C3540). The experiments on the ISCAS'89 circuits also suggest that it may be unnecessary to use multiple scan chains when there exists a circuit whose test set size is much larger than those of other circuits; using a single line to broadcast test patterns to all flip-flops may achieve the same fault coverage with comparable test application time.

Unlike the case of ISCAS'85 circuits, the test-generation time in our methods for ISCAS'89 circuits is longer than the sum of that for each individual circuit. We have found that this is because when the virtual circuit is very large (107 326 gates in our case), memory swapping activity will consume a great amount of CPU time during the ATPG process.

VI. CONCLUSIONS

A novel DFT strategy based on the concept of pattern sharing has been presented in this paper that can substantially reduce the test application time for scan-based circuits. The key idea is to combine all circuits under test into a virtual circuit and execute the ATPG process on this circuit. With this method, the generated test patterns are able to cope with the requirements of each CUT simultaneously and can be shared by all CUT's. The test quality will not be lowered, and the complexities of fault simulation and test generation will not be increased. Some heuristic methods for combinational circuits and sequential circuits are developed in this paper to select a virtual circuit such that the number of generated test patterns can be nearly minimized. With this method, the test patterns can then be broadcast to all CUT's through a single scan input line. Since all test signals can be propagated throughout each CUT using one single line, the routing area of our scheme should be close to that required for a single scan chain scheme. Another major contribution of our design is it provides a low-cost and high-performance method to integrate the boundary scan (IEEE 1149.1 Std.) and multiple scan test structure. In today's SOC environment, where a design can consist of tens or even hundreds of independent cores, our method should be able to substantially reduce the test vector volume and hence greatly decrease the test cost.

One important issue that has not been addressed in this paper is the power management problem. In real industry, it has been found that the number of vector sets that can be applied simultaneously is limited by "test power consumption" because usually test power is much higher than normal functional power. Since in this paper we have been trying to compress test vectors for many different cores, it may be necessary to set some power limit when generating test patterns. This, however, requires further research and should be a good direction of future work.

ACKNOWLEDGMENT

The authors wish to acknowledge the valuable suggestions from Prof. W.-B. Jone and the constructive comments from the anonymous reviewers.

REFERENCES

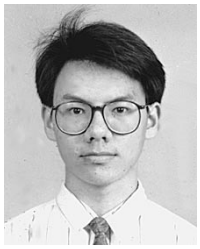
[1] R. S. Fetherston, I. P. Shaik, and S. C. Ma, "Testability Features of $AMD-K6^{TM}$ Microprocessor," in *Proc. Int. Test Conf.*, 1997, pp. 406–413.
[2] Zhang and R. D. McLeod, "An efficient multiple scan chain testing scheme," in *Proc. 6th Great Lakes Symp. VLSI*, Mar. 1996, pp. 294–297.
[3] S. Narayanan and M. A. Breuer, "Asynchronous multiple scan chains," in *Proc. VLSI Test Symp.*, May 1995, pp. 270–276.
[4] R. Gupta, S. Narayanan, and M. A. Breuer, "Optimal configuring of multiple scan chains," *IEEE Trans. Comput.*, vol. 42, pp. 1121–1131, Sept. 1993.
[5] S. Lee and K. G. Shin, "Design for test using partial parallel scan," *IEEE Trans. Computer-Aided Design*, vol. 9, pp. 203–211, Feb. 1990.
[6] S. M. Reddy and R. Dandapani, "Scan design using standard flip-flops," *IEEE Design Test Comput. Mag.*, pp. 70–75, Feb. 1987.
[7] *Supplement to IEEE Std. 1149.1-1990. IEEE Standard Test Access Port and Boundary Scan Architecture.* New York: McGraw-Hill, 1994.
[8] S. J. Jou, C. Su, and Y. T. Ting, "Decentralized BIST for 1149.1 and 1149.5 based interconnects," in *Proc. ED&TC*, Mar. 1996, pp. 120–125.
[9] F. Brglez and H. Fujiwara, "A neutral netlist of 10 combinational benchmark designs and a special translator in Fortran," in *Proc. Int. Symp. Circuits and Systems*, June 1985, pp. 663–698.
[10] F. Brglez, D. Bryan, and K. Kozminski, "Combinational profiles of sequential benchmark circuits," in *Proc. Int. Symp. Circuits and Systems*, May 1989, pp. 1929–1934.
[11] P. Goel, "An implicit enumeration algorithm to generate tests for combinational logic circuit," *IEEE Trans. Comput.*, vol. C30, pp. 215–222, Mar. 1981.
[12] P. Goel and B. C. Rosales, "Test generation & dynamic compaction of tests," in *Dig. Test Conf.*, Oct. 1979, pp. 189–192.

[13] M. Abramovici and J. J. Kulikowski, "Smart and fast: Test generation for VLSI scan-design circuits," *IEEE Design Test Comput. Mag.*, pp. 43–54, Aug. 1986.
[14] J. S. Chang and C. S. Lin, "Test set compaction for combinational circuits," *IEEE Trans. Computer-Aided Design*, vol. 14, pp. 1370–1378, Nov. 1995.
[15] B. Krishnamurthy and S. B. Akers, "On the complexity of estimating the size of a test set," *IEEE Trans. Comput.*, vol. C-33, pp. 750–753, Aug. 1984.
[16] P. H. Bardell and W. H. McAnney, "Self-testing of multichip logic modules," in *Proc. Int. Test Conf.*, Nov. 1982, pp. 200–204.
[17] ———, "Parallel pseudorandom sequences for built-in test," in *Proc. Int. Test Conf.*, Oct. 1984, pp. 302–308.

Kuen-Jong Lee received the B.S. degree in electrical engineering from the National Taiwan University, Taiwan, R.O.C., in 1981, the M.S. degree in electrical and computer engineering from the University of Iowa, Iowa City, in 1986, and the Ph.D. degree in electrical engineering from the University of Southern California, Los Angeles, in 1991.

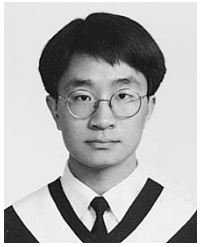
He joined the Department of Electrical Engineering, National Cheng Kung University, Tainan, Taiwan, in 1991, where he is a Professor. Currently, his research focuses on the design and testing of memory, CPU, and mixed-mode circuits. He has served on the technical program committees of a number of conferences and symposiums and will be a Technical Program Cochair for the 2000 Asian Test Symposium.

Dr. Lee's work on testable filter design received the Best Paper Award in the 1994 Hi-Media Technology and Application Workshop.



Jih-Jeen Chen received the B.S. degree in electronic engineering from Chung Yuan Christian University, Chung-Li, Taiwan, R.O.C., in 1995 and the M.S. degree in electrical engineering from Chung-Hua Polytechnic Institute, Hsin-Chu, Taiwan, in 1997. He is currently working toward the Ph.D. degree at National Cheng Kung University, Tainan, Taiwan.

His research interests include the fields of system design of digital signal-processing applications, scan-based architecture, and testable design.



Cheng-Hua Huang received the B.S. and M.S. degrees in electrical engineering from National Cheng Kung University, Tainan, Taiwan, R.O.C., in 1996 and 1998, respectively.

He is currently serving his compulsory military service in Taiwan.