PAPER

Concurrent Core Testing for SOC Using Merged Test Set and Scan Tree

Gang ZENG[†], *Nonmember and* **Hideo ITO**[††a)], *Fellow*

SUMMARY A novel concurrent core test approach is proposed to reduce the test cost of SOC. Prior to test, the test sets corresponding to cores under test (CUT) are merged by using the proposed merging algorithm to obtain a minimum merged test set. During test, the proposed scan tree architecture is employed to support the concurrent core test using the merged test set. The approach achieves concurrent core test with one scan input and low hardware overhead. Moreover, the approach does not need any additional test generation, and it can be used in conjunction with general compression/decompression techniques to further reduce test cost. Experimental results for ISCAS 89 benchmarks have proven the efficiency of the proposed approach.

key words: *concurrent testing, SOC testing, test cost reduction, test data compression*

1. Introduction

A system-on-a-chip (SOC) typically contains various predesigned and prevalidated embedded intellectual property (IP) cores provided by different core vendors. While the design methodology of SOC with IP reuse can meet shorter time to market, the test cost of SOC has increased greatly. This is because the test data volume (TDV) and test application time (TAT) associated with SOC test cost tend to increase more than linearly with gate count, as the complexity of SOC continuously increases [1]. On the other hand, the number of tester channels, and data memory of conventional automatic test equipment (ATE) are limited, thus the high-cost ATE with a greater number of test channels and larger-sized data memory is necessary for SOC testing. In addition, the limitation of traditional ATE requires a close to sequential test of all embedded cores [2], resulting in long TAT for entire SOC testing. All these reasons thus lead to high-cost SOC testing. In order to reduce test cost, research has been done in two ways. One way is to test a complex SOC chip with a low-cost ATE by employing appropriate design-for-testability (DFT) techniques [3], [4], and another way is to enable higher degrees of parallel testing [2], [5]. As far as the effectiveness of the two ways is concerned, it has been shown quantitatively in [6] that parallel test reduces test cost more effectively than low-cost ATE.

For low-cost ATE, several test techniques have been proposed such as scan tree techniques [7], [8], and test data compression/decompression techniques [9]–[11]. While scan tree can achieve significant test cost reduction at the minimum hardware overhead, it typically requires reordering the scan cells according to their compatible relations, which is very expensive because it requires changing the place and route information of the scan cells and may change the timing of the circuit [15]. As reported in [7], while the optimal scan tree with reordering scan cells can achieve 70% reduction in test cost, the constrained scan tree with reordering scan cells under the layout constraint can only achieve 32% reduction in test cost. For test data compression techniques [9]–[11], they mainly focus on core-level testing. That is, the on-chip decoder is typically designed for testing one core by using one tester channel. In the case that multiple cores need to be tested, on one hand, multiple decoders will be required, resulting in high hardware overhead. On the other hand, as far as test application time is concerned, if only one tester channel can be used, multiple cores have to be tested sequentially, thus resulting in long TAT. Otherwise, to shorten TAT multiple tester channels can be employed for parallel test, leading to high pin-count testing.

To support concurrent test and overcome the above problems, a "virtual circuit" method was proposed in [12] for testing multiple independent circuits simultaneously. While the method can effectively reduce test application time and the number of required tester channels, it relied on running a special ATPG procedure to obtain the shared test set for the virtual circuit. Moreover, it is difficult for this method to find the optimal virtual circuit and corresponding scan chain architecture to achieve the largest reduction in test cost because there are too many possibilities for constructing the virtual circuit [12]. More recently, an "overlapped vector" method was proposed in [13] where the overlapped vectors were broadcasted to multiple cores using an input pin. While this method achieves significant reduction in test cost without use of any additional ATPG, it suffers from the complex hardware structure in which an FIFO queue and an additional ATE channel have to be used to retrieve the original test vectors from the overlapped vectors. In addition, it did not present the design of control circuit for scan-enable signal when capturing test responses, which will make the hardware more complex.

In this paper, we will focus on system-level test of SOC, and propose an approach to support concurrent core test for the reduction in SOC test cost. First, the precom-

Manuscript received July 7, 2005.
Manuscript revised September 20, 2005.

[†]The author is with the Graduate School of Science and Technology, Chiba University, Chiba-shi, 263–8522 Japan.
[††]The author is with the Faculty of Engineering, Chiba University, Chiba-shi, 263–8522 Japan.
a) E-mail: h.ito@faculty.chiba-u.jp

DOI: 10.1093/ietisy/e89–d.3.1157

puted test sets from core vendors are merged into a shared test set by using the proposed merging algorithm. Second, the scan chains of different cores in one SOC are utilized to construct a scan tree based on the merged test set. Note that the scan tree is constructed without the requirement for reordering scan cells, which is different from the previous scan tree methods [7], [8]. Finally, all cores in the scan tree can be tested simultaneously using only one test input. In comparison with previous work in [13], the advantages of the proposed approach are that the scan tree is easy to construct with low hardware overhead and it does not require additional ATE channel. Furthermore, the approach does not need to run any special test generation and can be used in conjunction with general compression/decompression techniques to further reduce test cost.

The rest of the paper is organized as follows. Section 2 gives an overview of the proposed test approach. Section 3 presents the proposed algorithm for merging test sets. Section 4 describes the possible test application strategies with the proposed approach. In Sect. 5, experimental results are given. Finally, Sect. 6 summarizes the paper.

2. Overview of the Proposed Approach

The main idea behind this technique is to share one test set for multiple cores so that the shared test set can be used for simultaneously testing multiple cores. The shared test set is obtained by applying the proposed merging algorithm that tries to merge test sets of different cores in such a way that the merged test set has the minimum size. This method takes advantage of the large number of don't care bits in the test set generated by ATPG. As reported in [3] and [14], for large industry circuits the percentage of care bits in test sets is typically in the range of 1% to 5%. Therefore, there is sufficient probability to achieve successful merging. The basic idea for merging test sets is similar to that of static compaction that is employed in the procedure of ATPG. While static compaction tries to merge different test vectors from one test set into one test vector, the approach tries to merge different test sets from different cores into one test set. After the shared test set is obtained, another key point is how to retrieve original test vectors from the merged test set during test. We propose scan tree architecture to support multiple cores concurrent testing. The advantage of this architecture is the low hardware overhead for its implementation. Furthermore, the construction of scan tree is straightforward simply based on the merged bit positions for different test sets.

In [7], scan tree has been proposed for single-core testing to reduce TDV and TAT. Figure 1 shows an example to illustrate how to construct scan tree based on the compatible relation between flip-flops. Two flip-flops are compatible if any test data in one test set corresponding to the two flip-flops are same or at least one of them is a don't care bit (X). A compatible group is a set of flip-flops that are pairwise compatible. In this example, there are three groups of compatible scan flip-flops, i.e., {FF1, FF4, FF6}, {FF2, FF3} and {FF5}. As the compatible scan flip-flops should receive the same test data, the single scan chain in Fig. 1 (a) can be reconstructed into a scan tree given in Fig. 1 (b).

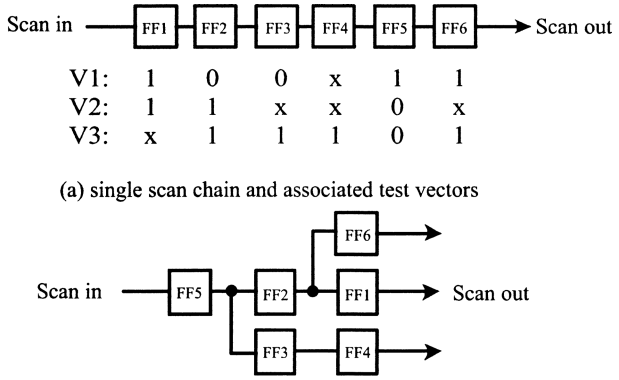


Fig. 1 Construction of scan tree for single-core testing.

In this paper, we will explore the potential of scan tree in supporting multiple cores concurrent testing so as to reduce the cost for entire SOC testing. The method is to construct a scan tree based on the compatible relation between test sets such that multiple cores can be tested simultaneously using the scan tree. Unlike the compatible relation in the same test set, the compatible relation between different test sets has more flexibility because the number and the length of test vectors for different test sets may be different. This flexibility thus results in simple construction of scan tree for multiple cores without the requirement of reordering the scan chain.

As an example, Fig. 2 shows a possible application with the proposed approach. As can be seen from Fig. 2 (a), (b), and (c), there are three individual test sets corresponding to three different cores, respectively. Because there are compatible relations among three test sets, they can be merged into one test set with the minimum size. In this example, to obtain a minimum size of merged test set, test set 1 and test set 3 should be merged with test set 2 from the second bit position and the third bit position, respectively. After merging operation, a merged test set can be obtained as shown in Fig. 2 (d) where the dashed blocks and the black blocks represent the original test vectors in test set 1 and test set 3, respectively. According to the beginning bit positions of merging operation, i.e., the second and the third bit of test set 2, the corresponding scan tree is easy to construct as shown in Fig. 2 (e). Note that if the required interconnection between cores is impossible for some IP protection cores, an alternative method is to use clock disable technique as done in [16]. By employing the scan tree architecture, three cores therefore can be tested simultaneously using the merged test set. The test procedure with the proposed approach is the same as the normal scan testing, except replacing the original test set with a merged test set.

As far as output response analyzer (ORA) is concerned, as done in [12], [13], a time-based compactor, e.g., MISR

	FF1	FF2	FF3
V11:	1	1	1
V12:	0	1	0

(a) Test set 1 for core 1

	FF1	FF2	FF3	FF4	FF5	FF6
V21:	1	0	1	1	x	0
V22:	0	1	1	1	0	1
V23:	1	1	0	0	1	x

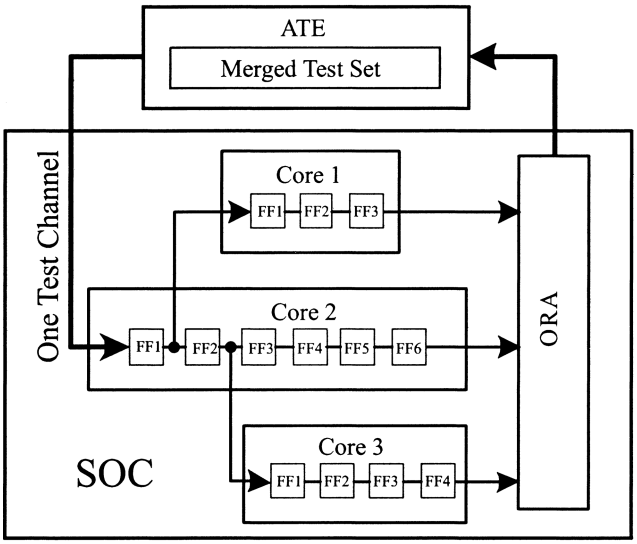
(b) Test set 2 for core 2

	FF1	FF2	FF3	FF4
V31:	0	0	1	0
V32:	1	0	0	x
V33:	x	1	0	0

(c) Test set 3 for core 3

	FF1	FF2	FF3	FF4	FF5	FF6
V1:	1	0	1	1	0	0
V2:	0	1	1	1	0	1
V3:	1	1	0	0	1	0
V4:	x	0	1	0	0	x

(d) Merged test set



(e) Test application using merged test set and scan tree

Fig. 2 Concurrent core test using merged test set and scan tree.

can be used to compact test responses and eventually delivers the signature to ATE using only one output pin. Alternatively, a space-based compactor with X-tolerant capability [17] can also be used.

3. Proposed Algorithm for Merging Operation

Without loss of generality, we assume that the test sets of CUTs provided by core vendors have been generated with dynamic and static compaction during ATPG procedure. That is, each test set has the minimum size and the test vectors in the same test set cannot be further merged with each other. In this section, the proposed normal and partition merging algorithm and the corresponding scan tree architecture will be presented, respectively.

3.1 Algorithm for Normal Merging Operation

It is known that the test compaction problem for a single test set is NP-complete [18]. Therefore, we propose a heuristic algorithm for merging multiple test sets to obtain a minimum merged test set. The outline of the proposed algorithm is presented as follows:

(1) The test set with the longest vector is selected as target test set and the remaining test sets are selected as candidate test sets.
(2) Determine the order of merging candidate test sets according to the number of test vectors and the length of test vector in each candidate test set. The principle is to first merge the test set with more difficulty for merging. The detailed criteria are described later.
(3) Select one test set from the candidate test sets according to the order of merging operation determined in the Step 2.
(4) Sort the test vectors in both the target test set and the selected candidate test set, respectively, as the descending order of the number of care bits in each vector.
(5) Explore all possible merging positions to determine the best merging position for this selected candidate test set. The best merging position is defined as the bit position at which the most number of test vectors in the candidate test set can be merged with target test set, i.e., the merged test set has the minimum size.
(6) Select one test vector from the selected candidate test set and try to merge it with the target test set at the best merging position. If the merging is not successful, a full-X (don't care bit) vector is appended to the end of the target test set. Then the selected vector is merged with the appended vector at the best merging position.
(7) Repeat the Step 6 until all vectors in this selected candidate test set have been merged with target test set.
(8) Replace the target test set with the merged test set obtained in Step 7, and repeat the steps from 3 to 8 until all candidate test sets have been merged with the target test set.

In Step 2, the order of merging candidate test sets is determined as follows. Assume a candidate test set containing C_n C_l-bit vectors will be merged with a target test set containing T_n T_l-bit vectors, where C_n C_l and T_n T_l denote the number of vectors, the length of vector in the candidate test set and in the target test set, respectively. If $C_n > T_n$, the candidate test set with larger C_n is prioritized to merge with the target test set. Otherwise, the candidate test set with larger C_l is merged with target test set first. In Step 5, generally, when merging a candidate test set with C_l-bit vector with a target test set with T_l-bit vector, where $T_l > C_l$, the possible merging position is from the first bit to the $T_l - C_l + 1$ bit position of the target test set. The algorithm will explore all these possible merging positions and give the corresponding merging results for each possible merging. Based on the results, the best merging position for each candidate test set is selected so that the merged test set has the minimum size.

We illustrate the proposed algorithm using the same example as shown in Fig. 2. First, as described in Step 1, test set 2 is selected as target test set since it has the longest vector; and then the remaining test set 1 and 3 are selected as candidate test sets. Second, test set 3 is selected to merge with the target test set first because it has longer vector than test set 1. As shown in Fig. 2 (d), because at the third bit

position of the target vector, merging can achieve the better results than merging at any other bit positions, this bit position is referred to as the best merging position. Note that for the merging of V32, since it cannot be merged with any target vectors, a full-X test vector V4 is appended to the end of test set 2, and then V32 is merged with it. Finally, the remaining candidate test set 1 is selected to merge with the target test set at the second bit position. As a result, the finally merged test set in Fig. 2 (d) can be shared for testing all three cores by using the corresponding scan tree as shown in Fig. 2 (e).

It is important to point out that for all test vectors in one candidate test set, the beginning merging position is fixed to the best merging position. This is an important principle for merging operation to simplify the corresponding hardware. On one hand, this principle guarantees that each CUT can receive its expected test vector from the merged test set at its corresponding best merging position, thus making the scan tree applicable to multiple cores testing. On the other hand, this principle guarantees that all CUTs can share one capture cycle for one applied test vector, thus greatly simplifying the design of control logic for generating the capture sequence. The employed method of one capture for one merged test vector is distinct from the method of [13] in which multiple vectors from one test set can be overlapped with one target vector at different bit positions, which will complicate the design of control logic.

3.2 Algorithm for Partition Merging Operation

When the normal merging procedure cannot achieve completely merging as occurred in the above Step 6, i.e., there are some vectors in the candidate test set that cannot be merged with any vectors in the target test set, we propose a novel partition merging operation to improve the results of normal merging. The basic idea of partition merging is based on the fact that the distribution of care bits in one test set is not uniform according to the bit positions. Figure 3 illustrates this distribution for the s5378 benchmark circuit and the similar property can also be observed for the test sets of the other circuits. As shown in the figure, in some positions such as from 106 to 114 bit positions, almost every bit in the test set is care bit. In contrast to this, the nearby positions have much less distribution of care bits. Therefore, when merging this candidate test set with other target test set, if the target test set also has high distribution of care bits corresponding to this high distribution of care bits in the candidate test set, the merging results may be incomplete. The method for solving this problem is to partition the candidate test set in such a way that the high distribution of care bits will be merged with low distribution of care bits or vice versa. To this end, we propose the following heuristic procedure to perform partition merging when the normal merging cannot achieve completely merging.

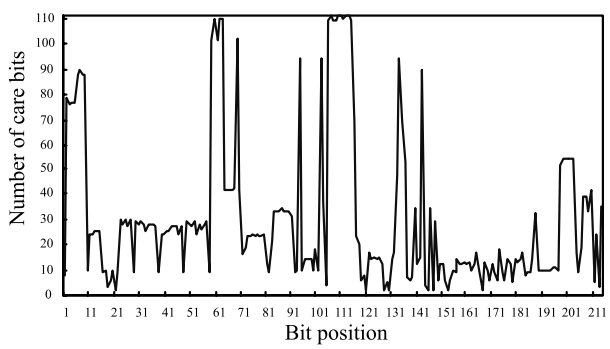


Fig. 3 Distribution of care bits in the test set of the ISCAS benchmark circuit s5378.

(1) Perform normal merging operation. If the merging results are incomplete, record the best merging position.
(2) Estimate the probability of merging conflict at each bit position for both the candidate and target test set by calculating the total number of care bits at each bit position as done in Fig. 3. The bit positions in which there are most number of care bits will be used for partition first.
(3) Partition the candidate test set into two test set segments at the bit position obtained in step 2, and insert one don't care bit between the two segments.
(4) Perform the normal merging operation again at the recorded best merging position.
(5) Check the merged results. If the results have not been improved, then repeat Step 3, 4, 5, and partition the candidate test set into more test set segments by inserting don't care bits into the candidate test set at different bit positions until the number of remaining test vectors in the candidate test set that cannot be merged with target test set has been decreased to the expected number or the partitioned candidate vector has the same length as that of target vector.

We use an example to illustrate the proposed partition algorithm. As shown in Fig. 4 (a), the candidate and target test set contain two vectors respectively that cannot be merged with each other from the first bit position using normal merging operation. It can be observed after the Step 2 of partition merging that the fourth and the sixth bit position of the target test set have the most number of care bits than any other bit positions. Actually, since there are two merging conflicts at these bit positions denoted with underline, these test vectors cannot be merged directly. Therefore, we need to partition the candidate test set to avoid merging conflict at these bit positions. In Step 3, one don't care bit is first inserted at the fourth bit position to perform the partition (a) as shown in Fig. 4 (b). Because the two candidate vectors still cannot be merged with target test set completely, another don't care bit is inserted at the sixth bit position to further perform the partition (b). As can be seen in Fig. 4 (c), the two test sets can be merged successfully at the first bit position after the partition (b) operation. The corresponding scan tree to support this partition merging is given in Fig. 4 (d) where the T_m and C_m represent the m scan cell in target and candidate scan chain, respectively.

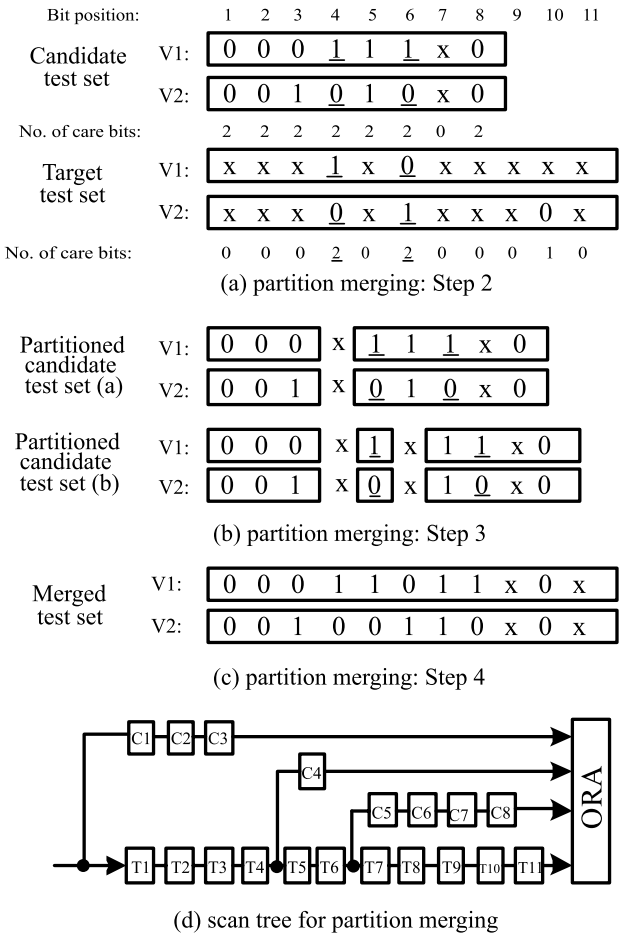


Fig. 4 An example of partition merging operation.

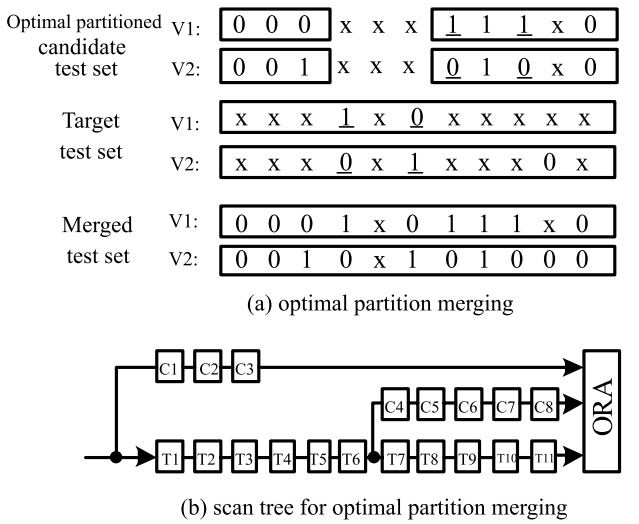


Fig. 5 An example of optimal partition merging operation.

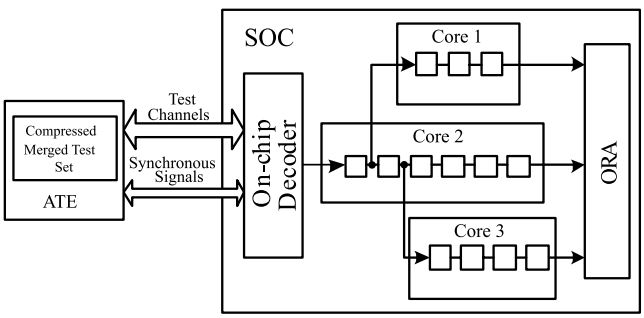


Fig. 6 Test application using ATE and an on-chip decoder.

As shown in Fig. 4 (d), the original candidate scan chain has been broken into three scan chain segments after partition merging. Generally, one additional scan chain segment will require one additional flip-flop for the MISR to implement the ORA, which means more hardware overhead. Consequently, the objective of partition merging is to achieve improved merging with the minimum number of partitions. This optimal partition merging can be achieved by inserting additional don't care bits to combine separate and close don't care bits into don't care bit segments so that the number of partitioned scan chain segments is reduced. As for the above example, the corresponding optimal partition is given in Fig. 5 where two separate don't care bits are combined into one 3-bit segment by inserting a new don't care bit such that the number of partitioned scan chain segments is reduced to two. When comparing the optimal scan tree in Fig. 5 (b) with the normal scan tree in Fig. 4 (d), it is clear that the optimal partition merging needs less hardware resource than normal partition merging.

4. Test Applications with the Approach

The feasible test applications with the proposed approach can be classified as the following two strategies.

Strategy one : Concurrent test using ATE as shown in Fig. 2 (e). The significant advantage of this application is the simple implementation with the minimum hardware overhead and no additional requirement for synchronous signal between ATE and SOC.

Strategy two : Concurrent test using ATE and an on-chip decoder as shown in Fig. 6. In this application, the merged test set first is compressed to a smaller one using general test compression technique, and then the compressed test set is stored in ATE. During test, the compressed test set is decompressed by an on-chip decoder, and then the restored test vectors are applied to all cores through the predetermined scan tree architecture. This strategy can achieve better compression than strategy one at the expense of a small hardware overhead for the decoder.

The detailed evaluation of the two strategies will be given in the next section.

5. Experimental Results

We implement the proposed merging algorithm in the C language on a PC with a 2.80 GHz Pentium 4 processor and 512 MB memory. The algorithm is used to merge test sets

of the large ISCAS 89 benchmark full-scan circuits. In this experiment, the benchmark circuits are treated as cores belonging to one SOC, and the purpose is to test all these cores in parallel by using a merged test set. The corresponding test sets for these benchmark circuits are the dynamically compacted test sets generated by MINTEST [19] with 100% stuck-at fault coverage. Note that the MINTEST sets have the minimum size and any vector in one test set cannot be further merged with each other, which is consistent with the assumption in Sect. 3.

Table 1 lists the characteristics of MINTEST sets for the large ISCAS 89 benchmark circuits. As can be seen, the s38417 has the longest scan chain. We therefore select the test set of s38417 as the target test set and try to merge the other test sets with it. The definition of percentage compression used in the experiment is given as follows:

$$\frac{(\text{Size of the test set} - \text{Size of merged test set})}{\text{Size of the test set}} \times 100$$

5.1 Experimental Results of Strategy One

The results using the proposed merging algorithm are listed in Table 2. Based on the assumption in Sect. 3, the theoretical bound for the minimum size of merged test set can be calculated as follows:

$$\text{Size of minimum merged test set} = L_{max} \times N_{max}$$

where L_{max} and N_{max} represent the length of the longest vector, and the number of vectors of the largest test set, respectively, in all test sets used for merging. With regard to this experiment, the theoretical minimum merged test set is composed of 236 1664-bit vectors. As can be seen from the results, the proposed method can merge six test sets into one test set with only 3 more number of test vectors than the theoretical optimum results. The bit map of the merged test set is illustrated in Fig. 7. Note that except for special denotation, the inserted number of don't care bits between two adjacent segments is one. From this bit map, it is straightforward to construct the corresponding scan tree.

Table 3 shows the experimental results of normal merging and partition merging. As can be seen, the partition merging can achieve better compression than the normal merging. The last column of Table 3 denotes the required minimum number of flip-flops for the MISR to implement the ORA. As discussed in Sect. 3.2, the more segments we partition the test set into, the more flip-flops the MISR will require. Therefore, Table 3 offers a trade-off between hardware overhead and achieved compression.

5.2 Experimental Results of Strategy Two

As shown in Table 2, we know the fact that even the density of care bits has been increased after merging operation,

Table 1 Characteristics of MINTEST sets for ISCAS 89 benchmark circuits.

Circuit name	No. of scan cells	No. of test vectors	No. of bits in test set	Density of don't care bit (%)
S5378	214	111	23,754	73
S9234	247	159	39,273	73
S15850	611	126	76,986	84
S13207	700	236	165,200	93
S38417	1664	99	164,736	68
S38584	1464	136	199,104	82
Total bits			669,053	

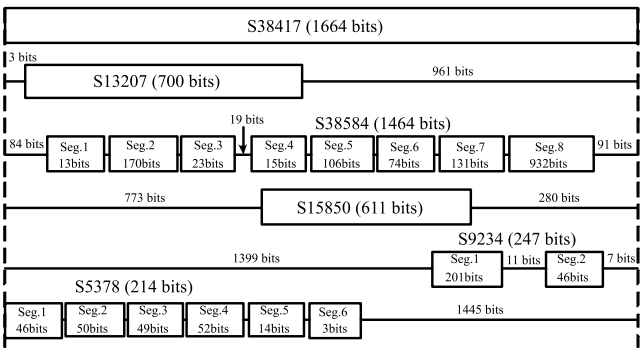


Fig. 7 The bit map for the merged test set.

Table 3 Comparison of merged results using different merging methods.

Merging method	Length of vector	No. of vectors	No. of bits	Comp.	No. of FFs
Normal	1664	258	429,312	35.8%	6
Partition	1664	239	397,696	40.6%	19

Table 2 Merged results for ISCAS 89 benchmark circuits.

Order for merging	Candidate test set	Target test sets	No. of vectors in candidate	No. of vectors in target	No. of merged vectors	No. of partitions	Best merging position (bits)	density of care bits	CPU time (seconds)
1	s13207	s38417	236	99	96	0	4	16.0%	57
2	s38584	{ s13207, s38417 }	136	239	136	8	85	24.8%	5
3	s15850	{ s38584, s13207, s38417 }	126	239	126	0	774	27.9%	37
4	s9234	{ s15850, s38584, s13207, s38417 }	159	239	159	2	1400	30.5%	74
5	s5378	{ s9234, s15850 s38584, s13207, s38417 }	111	239	111	6	1	32.1%	54

the merged test set still has a large number of unspecified bits. This redundancy indicates that the merged test set can be further compressed by using general lossless compression methods. Since the statistical codes is more effective in compressing the test set with higher density of care bit than run length based codes such as FDR codes or Golomb codes, we employ the selective Huffman codes [9] to compress the merged test set. As suggested in [9], in order to minimize the hardware of the decoder, the merged test set is divided into 8-bit block and only 10 patterns with the highest frequency of occurrence are selected to perform Huffman coding. As can be seen from the compressed results listed in Table 4, selective Huffman codes can further compress the merged test set by 47.6% that is indicated as comp.1 in Table 4. Eventually, the proposed method can achieve total 68.9% compression by combining the proposed merging method and the selective Huffman codes, which is indicated as comp.2 in Table 4.

As discussed in Sect. 1, for multiple cores test, the test compression techniques [9]–[11] require high hardware overhead and either long TAT or high pin-count testing. Table 5 shows the comparison of concurrent test using general test compression and strategy two. Note that this comparison is made under the condition of parallel test of 6 cores. In this case, as can be seen from the table, it is obvious that combining the test compression technique with the proposed method can achieve not only low hardware overhead and reduced pin-count testing but also more reduction in test data volume. Note that if under the condition of one tester channel, the 6 cores should be tested sequentially for general test compression technique. In this case, total 6 test sets with 270372-bit data are required to transfer from the ATE to the on-chip decoder. In contrast, only one test set with 208325-bit data is required to transfer for the proposed strategy two, which indicates shorter TAT. The detailed analysis of TAT with selective Huffman codes can be found in [9].

Table 4 Compression results of merging method and merging method combined with selective Huffman codes (8 bits coding with 10 selective patterns).

Size of original test set (bits)	Merged test set		Selective Huffman codes using merged test set		
	No. of bits	Comp.	No. of bits	Comp.1	Comp.2
669,053	397,696	40.6%	208,325	47.6%	68.9%

Table 5 Comparison of concurrent test using selective Huffman codes for different test set.

Method of concurrent test	No. of CUTs	No. of decoders	No. of ATE channels	Comp.
Test comp. for original test sets	6	6	6	59.6%
Test comp. for merged test set	6	1	1	68.9%

Table 6 shows the comparison of test data compression with previously published results. Note that it is difficult to compare with [12] and [13] directly, because they used different test sets for experiments. To validate the effectiveness of the proposed approach, we therefore select test compression techniques [9]–[11] for fair comparison since all these methods use the same MINTEST sets. The compression results of previous methods are calculated by accumulating the 6 individual compressed test sets, which are obtained by compressing the original 6 test sets, separately. Note that the results of selective Huffman code are obtained by using 8-bit block coding. As can be seen from the results, the proposed approach can achieve better compression than previous test compression methods.

6. Conclusion

This paper has proposed a novel concurrent core test approach for SOC to reduce test cost. The approach achieves concurrent testing by using the proposed merging algorithm and scan tree architecture. Prior to test, the test sets corresponding to cores under test are merged by using the proposed merging algorithm to obtain a minimum merged test set. During test, the predetermined scan tree architecture is used to retrieve the original test vectors from the merged test set. In addition to the benefit of reduced test costs, the approach achieves concurrent core test with only one scan input and low hardware overhead. Moreover, the approach does not need any additional test generation and it can be used in conjunction with general compression/decompression techniques to further reduce test cost. Experimental results for ISCAS 89 benchmarks show that the proposed merging method can achieve 40.6% reduction in TDV and TAT. When combining with selective Huffman codes, the proposed method can achieve total 68.9% compression in TDV.

In future work, we need to find a solution to apply the proposed approach to the multi-clock SOC, which is a similar problem associated with logic BIST when applying logic BIST to multi-clock designs [20].

Acknowledgments

The work of the first author is supported by a research grant from the Marubun Research Promotion Foundation

Table 6 Comparing test data compression with previously published results.

Size of original test set (bits)	Golomb codes [10]		FDR codes [11]		Selective Huffman codes [9]		Selective Huffman codes using merged test set	
	No. of bits	Comp.	No. of bits	Comp.	No. of bits	Comp.	No. of bits	Comp.
669,053	314,875	52.9%	262,656	60.7%	270,372	59.6%	208,325	68.9%

of Japan. The authors would like to thank Lei Li and Dr. Krishnendu Chakrabarty from Duke University for providing the MINTEST sets.

References

[1] C. Barnhart, V. Brunkhorst, F. Distler, O. Farnsworth, B. Keller, and B. Koenemann, "OPMISR: The foundation for compressed ATPG vectors," Proc. Int. Test Conf., pp.748–757, 2001.
[2] R. Dorsch, R.H. Rivera, H.J. Wunderlich, and M. Fischer, "Adapting an SOC to ATE concurrent test capabilities," Proc. Int. Test Conf., pp.1169–1175, 2002.
[3] J. Bedsole, R. Raina, A. Crouch, and M. Abadir, "Very low cost testers: Opportunities and challenges," IEEE Design Test Comput., pp.60–69, Sept./Oct. 2001.
[4] E.J. McCluskey, D. Burek, B. Koenemann, S. Mitra, J. Patel, J. Rajski, and J. Waicukauski, "Roundtable: Test data compression," IEEE Design Test Comput., pp.76–87, March/April 2003.
[5] E. Volkerink, A. Khoche, J. Rivoir, and K. Hilliges, "Test economics for multi-site test with modern cost reduction techniques," Proc. VLSI Test Symposium, pp.411–416, 2002.
[6] J. Rivoir, "Lowering cost of test: Parallel test or low-cost ATE," Proc. Asian Test Symposium, pp.360–364, 2003.
[7] K. Miyase and S. Kajihara, "Optimal scan tree construction with test vector modificatioin for test compression," Proc. Asian Test Symposium, pp.136–141, 2003.
[8] H. Yotsuyanagi, T. Kuchii, S. Nishikawa, M. Hashizume, and K. Kinoshita, "Reducing scan shifts using folding scan trees," Proc. Asian Test Symposium, pp.6–11, 2003.
[9] A. Jas, J.G. Dastidar, M.E. Ng, and N.A. Touba, "An efficient test vector compression scheme using selective Huffman coding," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol.22, no.6, pp.797–806, June 2003.
[10] A. Chandra and K. Chakrabarty, "System-on-a-chip test data compression and decompression architectures based on Golomb codes," IEEE Trans. Computer-Aided Des. Integr. Circuits Syst., vol.20, no.3, pp.355–368, March 2001.
[11] A. Chandra and K. Chakrabarty, "Test data compression and test resource 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.
[12] K. Lee, J. Chen, and C. Huang, "Using a single input to support multiple scan chains," Proc. Int. Conf. Computer-Aided Design, pp.74–78, 1998.
[13] T. Shinogi, Y. Yamada, T. Hayashi, T. Yoshikawa, and S. Tsuruoka, "Between-core vector overlapping for test cost reduction in core testing," Proc. Asian Test Symposium, pp.268–273, 2003.
[14] H. Vranken, F. Hapke, S. Rogge, D. Chindamo, and E. Volkerink, "ATPG padding and ATE vector repeat per port for reducing test data volume," Proc. Int. Test Conf., pp.1069–1078, 2003.
[15] S. Makar, "A layout-based approach for ordering scan chain flip-flops," Proc. Int. Test Conf., pp.341–347, 1998.
[16] G. Zeng and H. Ito, "Sharing test patterns for multiple cores using scan chain disable," IEICE Technical Report, FIIS 2004-145, 2004.
[17] S. Mitra and K.S. Kim, "X-Compact an efficient response compaction technique for test cost reduction," Proc. Int. Test Conf., pp.311–320, 2002.
[18] B. Krishnamurthy and S.B. Akers, "On the complexity of estimating the size of a test set," IEEE Trans. Comput., pp.750–753, Aug. 1984.
[19] I. Hamzaoglu and J.H. Patel, "Test set compaction algorithms for combinational circuits," Proc. Int. Conf. Computer-Aided Design, pp.283–289, 1998.
[20] G. Hetherington, T. Fryars, N. Tamarapalli, M. Kassab, A. Hassan, and J. Rajski, "Logic BIST for large industrial designs: Real issues and case studies," Proc. Int. Test Conf., pp.358–367, 1999.

Gang Zeng graduated from Hunan University, Changsha, Hunan Province, China, with B.E., M.S. degrees in 1993, 2001, respectively. From 1993 to 1998, he joined Hunan University where he was a lecturer in the Institute of Electric and Information Engineering. From 2001 to 2002, he joined ZTE Corporation where he was a senior engineer. Since April 2003, he has been with the graduate school of science and technology, Chiba University in Japan as a Ph.D. candidate. His research interests include design for testability, built-in self-test and system-on-a-chip testing etc. He is a student member of the IEEE.

Hideo Ito was born in Chiba, Japan, on June 1, 1946. He received the B.E. degree from Chiba University in 1969 and the D.E. degree from Tokyo Institute of Technology in 1984. He joined Nippon Electric Co. Ltd. in 1969 and Kisarazu Technical College in 1971. Since 1973, he has been a member of Chiba University. He is currently a Professor of Department of Information and Image Sciences. His research interests include easily testable design, test generation, VLSI architecture, fault-tolerant design of parallel & distributed systems, defect-tolerant VLSI design, and home network. He is a member of the IEEE and the IPSJ.