PAPER

Test Compression for Robust Testable Path Delay Fault Testing Using Interleaving and Statistical Coding*

Kazuteru NAMBA[†a)], ***Member and*** **Hideo ITO**[†], ***Fellow***

SUMMARY This paper proposes a method providing efficient test compression. The proposed method is for robust testable path delay fault testing with scan design facilitating two-pattern testing. In the proposed method, test data are interleaved before test compression using statistical coding. This paper also presents test architecture for two-pattern testing using the proposed method. The proposed method is experimentally evaluated from several viewpoints such as compression rates, test application time and area overhead. For robust testable path delay fault testing on 11 out of 20 ISCAS89 benchmark circuits, the proposed method provides better compression rates than the existing methods such as Huffman coding, run-length coding, Golomb coding, frequency-directed run-length (FDR) coding and variable-length input Huffman coding (VIHC).

key words: *test compression, statistical coding, run-length coding, delay fault testing, two-pattern testing, scan testing*

1. Introduction

In recent years, the size and density of VLSIs have increased, resulting in large volume of test data. The large volume requires automatic test equipment (ATE) with a large memory, which mean high test cost. In order to reduce test volume, test compression techniques are required [1]. There are two classes of non-intrusive approaches for test compression; those using and expander and those using coding. Approaches categorized into the former class, e.g. Smart-BIST [2] and Embedded deterministic test (EDT) [3], provide very efficient compression. The approaches can, however, manipulate such a limited test cube that they require dedicated test generation and compaction algorithms like described those in [3]. In other words, test cubes which have already been generated and compacted are not available for the approaches. Therefore, the approaches are not available to IP cores if core vendors provide system integrators with only test cubes and the system integrators do not know the structural information about the IP cores. Test compression using the following coding can be categorized into the latter class: statistical coding such as Huffman coding and selective Huffman coding [4], run-length coding [5], Golomb coding [6], FDR coding [7] and VIHC [8]. Although the approaches do not provide as efficient compression as the approaches categorized into the former class, they can be applied to IP cores as long as system integrators know test cubes for the IP cores. This paper focuses upon the latter class.

The recent increases in VLSI density have also resulted in increasing delay faults. Two-pattern testing is indispensable to delay fault testing. In two-pattern testing, two vectors, an initial (first) vector and a transition (final, second) vector, are applied to devices under test (DUTs) in two consecutive clock cycles. There are two classes of approaches for two-pattern testing using scan design: testing on DUTs employing standard scan design, e.g. broadside testing (functional justification) and skewed-load testing [9]; and testing using scan design facilitating two-pattern testing, e.g. enhanced scan design [9] and scan design proposed by the authors in [10]. This paper calls the scan design [10] *Chiba scan design*. Approaches categorized into the former class permit us to apply only strongly limited pair of initial and transition vectors to DUTs. Therefore, they may not be able to detect all robust testable path delay faults. Meanwhile, any robust testable path delay faults can be detected by enhanced and Chiba scan testing. This paper focuses upon the latter class. Since enhanced scan design requires too large area overhead to use in actual systems while Chiba scan design requires only a small area overhead as standard scan design, this paper discusses Chiba scan testing and not enhanced scan testing.

From those, test compressions for two-pattern testing using coding and Chiba scan design are important and are required. Polian, et al. have pointed out that test data for enhanced scan testing can be compressed by the traditional coding as long as the test data are compressed in the order in which they are applied to DUTs, i.e. in order of an initial vector and the corresponding transition vector [11]. Obviously, a similar discussion can be held for test data for Chiba scan design. Test data can, however, be compressed in another order, and reordering of test data may make test compression for delay fault testing more efficient.

This paper presents a reordering method providing efficient test compression using statistical coding. The proposed method targets robust testable path delay fault testing using Chiba scan design. In the proposed method, test vectors are interleaved before test compression.

This paper is organized as follows: Section 2 is preliminary. Section 3 presents the proposed method to make test compression more efficient. Section 4 shows test architecture in SoC for two-pattern testing using the proposed method. Sections 5 and 6 are evaluation and conclusion, respectively.

Manuscript received July 8, 2008.
Manuscript revised August 27, 2008.

[†]The authors are with the Graduate School of Advanced Integration Science, Chiba University, Chiba-shi, 263–8522 Japan.

*The earlier version of this paper was presented at the 15th Asian Test Symposium (ATS '06).

a) E-mail: namba@ieee.org

DOI: 10.1587/transinf.E92.D.269

2. Preliminary

2.1 Terminology Related to Delay Fault Testing

(1) CV (Controlling Value) and NCV (Non Controlling Value)

An input value of a gate is called a *controlling value* (CV) if it determines the output value of the gate regardless of the other gate input values. An input value that does not qualify as a CV is called a *non-controlling value* (NCV). For example, in an AND gate, the logic 0 is a CV while the logic 1 is an NCV.

(2) On-input and Off-input

Suppose a gate G is on a path P. If an input line of G is on P, the input line is called an *on-input* of P, or else it is called an *off-input* of P.

(3) Robust Off-input

Whether a path delay fault occurring on P can be observed on the output of P depends upon values of all of the off-inputs of P. An off-input is called a *robust off-input* if the off-input has values that enable the observation of delay faults on P regardless of delay faults occurring outside P.

An off-input is a robust off-input in the following cases [9]: 1) there is a transition CV $\rightarrow$ NCV or a stable NCV on the off-input if the on-input of G has a transition CV $\rightarrow$ NCV, or 2) there is a stable NCV on the off-input if the on-input of G has a transition NCV $\rightarrow$ CV.

(4) Robust testable path delay fault

For a path delay fault, if there is at least one pair of initial and transition vectors such that the pair makes all of the off-inputs of P robust off-inputs, the fault is called a *robust testable path delay fault*.

(5) Two-pattern testing

When pairs of test vectors $V = (V_1, V_2)$ are applied to DUTs in two consecutive clock cycles in a testing, the testing is called *two-pattern testing*, where $V_1 = (v_{1,0}, v_{1,1}, v_{1,2}, \cdots, v_{1,(n-1)})$ is an initial vector, $V_2 = (v_{2,0}, v_{2,1}, v_{2,2}, \cdots, v_{2,(n-1)})$ is a transition vector and n is the length of each test vector. Two-pattern testing is indispensable to delay fault testing.

(6) Nine-valued logic system

Nine-valued logic system [12] consists of nine symbols; S0, S1, T0, T1, H0, H1, U0, U1 and XX. The symbols represent a pair of initial and transition values of a signal, i.e. a pair of a bit of an initial vector $v_{1,i}$ ($0 \leqq i < n$) and the corresponding bit of the transition vector $v_{2,i}$. The symbols S0 and S1 represent stable values 0 and 1, i.e. pairs $(v_{1,i}, v_{2,i}) = (0, 0)$ and (1,1), respectively. The symbols T0 and T1 represent failing and rising transitions, i.e. pairs (1,0) and (0,1), respectively. The symbols H0 and H1 indicate values which may include hazards. In general, they do not occur in test data, and they do not appear in this paper. The symbols U0 and U1 represent pairs (X,0) and (X,1), respectively, where X is an unspecified value. The symbol U0 covers S0 and T0 while U1 covers S1 and T1. The symbol XX represents a pair (X,X) and covers eight other symbols.

2.2 Chiba Scan Design

Chiba scan design [10] is a class of scan design facilitating two-pattern testing. Chiba scan design imposes the following restriction on two-pattern test data: for any pair of initial and transition vectors, the values of either even-numbered or odd-numbered bits of the initial vector have to be the same as those of the corresponding bits of the transition vector. Note that, even if the values of even-(odd-)numbered bits in a pair of initial and transition vectors are made the same, the values of even-(odd-)numbered bits in another pair V may differ as long as the values of odd-(even-)numbered bits in V are the same. Without loss of generality, we assume that the values of odd-numbered bits are the same in later explanation.

To make two-pattern testing in Chiba scan design, the even-numbered bits of an initial vector, the odd-numbered bits of the initial/transition vector, and the even-numbered bits of the transition vector have to be applied to a DUT in that order as the examples shown in the following and Fig. 1:

$$[v_{1,0},\ v_{1,2},\ v_{1,4}, \cdots, v_{1,(n-2)},$$
$$v_{1,1},\ v_{1,3},\ v_{1,5}, \cdots, v_{1,(n-1)},$$
$$v_{2,0},\ v_{2,2},\ v_{2,4}, \cdots, v_{2,(n-2)}].$$

Because of the restriction, Chiba scan design does not permit to apply arbitrary two-pattern test data. However, it permits to apply two-pattern test data detecting all robust testable path delay faults because it permits to apply at least arbitrary single input change two-pattern test data. Moreover, the volume of robust testable path delay faults test data on Chiba scan design is not much larger than that on enhanced scan design. The authors' previous paper [10] presents detailed description of Chiba scan design and comparison between testing on Chiba scan design and other two-pattern testing such as enhanced scan, broadside and skewed-load testing.

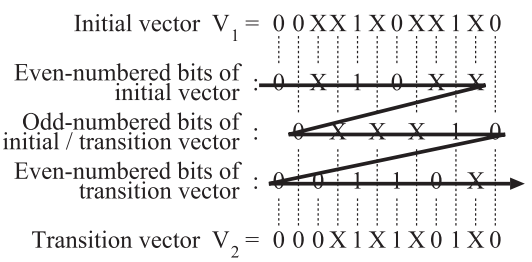


Fig. 1 Test data with original order on Chiba scan design.

2.3 Test Compression

(1) Statistical coding

In statistical coding such as Huffman coding and selective Huffman coding [4], original test data are divided into blocks of fix-length, and every block is represented by a codeword of variable length. The lengths of codewords for common blocks are shorter than those for less common blocks. Effectiveness of test compression using statistical coding strongly depends upon the frequency of occurrence of blocks. On one hand, if some blocks occur much more frequently than other blocks, the test data are effectively compressed. On the other hand, if codewords occur with uniform frequency, the test data are not compressed at all.

(2) Run-length coding

In run-length coding, runs of original data, i.e. sequences where the same value occurs in consecutive bits of original data, are represented by single codewords. If original data comprise many long runs, the test data are effectively compressed by run-length coding.

3. Interleaving of Test Data

3.1 Interleaving before Compression with Statistical Coding

By reordering test data before compression, we may compress test data more efficient. This section shows a reordering method, namely interleaving, to make test compression for robust testable path delay fault testing on Chiba scan design more efficient. In Chiba scan design, the following three are applied into a DUT: the even-numbered bits of an initial vector, the odd-numbered bits of the initial/transition vector, and the even-numbered bits of the transition vector.

In the proposed method, before compression, the above three are interleaved as the following and the example in Fig. 2:

$$[v_{1,0},\ v_{1,1},\ v_{2,0},\ v_{1,2},\ v_{1,3},\ v_{2,2},\ v_{1,4},\ v_{1,5},\ v_{2,4},\ \cdots,\ v_{1,(n-2)},\ v_{1,(n-1)}, v_{2,(n-2)}].$$

The next explains why the interleaving affords efficient compression. It is attributed to a feature of robust testable path delay fault test data in which the number of stable values is larger than that of transitions. First, the reason why robust testable path delay fault test data have the feature is explained, and next, the reason why the feature makes the interleaving bring efficient compression is explained.

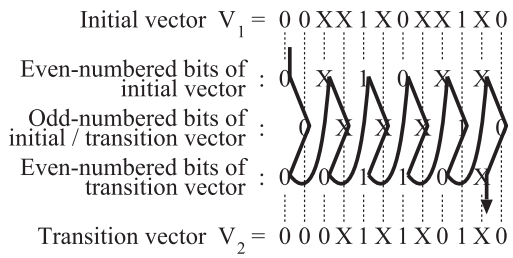


Fig. 2 Interleaved test data.

To test a path, every off-input of the path has to be robust off-input while every on-input of the path should have a transition. Each on-input of the path has either a transition NCV → CV or CV → NCV. If the on-input of a gate has NCV → CV, the off-inputs of the gate should have a stable NCV to make the off-input robust off-input. In order to supply stable values to the off-inputs corresponding to on-inputs with NCV → CV, stable values have to be supplied to some primary inputs and scan-FFs that are connected to the off-inputs directly or through some gates. In sum, in order to test the path, stable values have to be supplied to some (usually, multiple) primary inputs and scan-FFs while a transition has to supply to only the input of the path.

Figure 3 shows an example of robust testing. The circuit is a sub-circuit of s27 of ISCAS89. A rising transition T1 is supplied to the primary input G1 and the path G1-G12-G15-G9-G11-G10 is sensitized. The on-inputs of the gates G12, G9 and G11 have a transition NCV → CV while those of G15 and G10 have CV → NCV. The outputs of scan-FFs G7 and G5 are directly connected to the off-inputs of G12 and G11, respectively, and stable values S0s are supplied to G7 and G5. The primary input G3 is connected to the off-input of G9 through an OR-gate G16, and a stable value S1 is supplied to G3. In sum, stable values are supplied to G7, G5 and G3 while a transition is supplied to only G1.

Next, the reason why the large numbers of stable values make the interleaving bring efficient compression is explained. As discussed in Sect. 2, test data have to be divided into blocks before the test data are compressed by statistical coding. When interleaved test data are divided into b-bit (b is multiples of three) blocks, the j-th block appears as ($v_{1,2jb/3}$, $v_{1,2jb/3+1}$, $v_{2,2jb/3}$, $\cdots$, $v_{1,2(j+1)b/3-2}$, $v_{1,2(j+1)b/3-1}$, $v_{2,2(j+1)b/3-2}$). Consequently, the i-th and $(i + 2)$-th bits (i is multiples of three) of every block correspond to the identical even-numbered scan flip-flop. As shown above, the number of stable values is larger than that of transitions in robust testable path delay fault test data. In other words, each bit of every initial vector usually has the same value as the corresponding bit of the transition vector. Therefore, blocks where the i-th and $(i + 2)$-th bits are the same occur more frequently than blocks where the i-th bit differs from the $(i + 2)$-th bit. Since skewed frequency of occurrence of blocks brings efficient compression, test data are often compressed more efficiently by interleaving test vectors before compression.

3.2 Combination with Run-Length Coding

The statistical coding can be used in combination with run-length coding and the combination frequently provides efficient compression. The proposed interleaving may also provide efficient test compression using the combination of

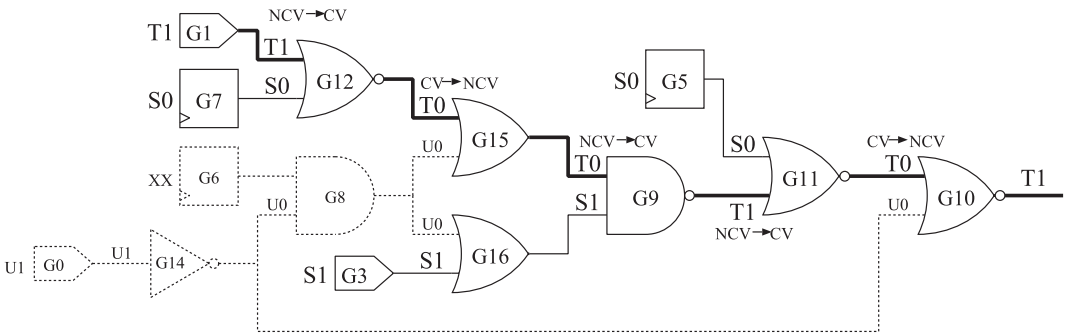


Fig. 3 Robust testing for transition on path G1-G12-G15-G9-G11-G10.

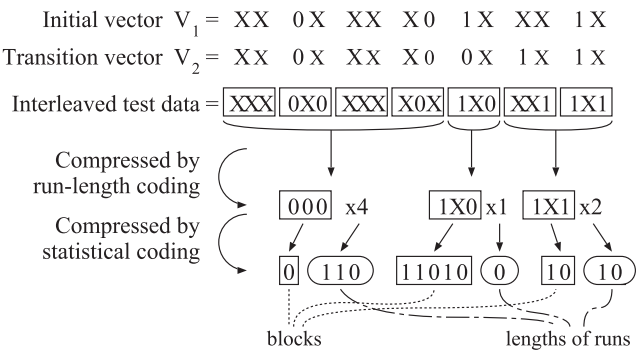


Fig. 4 Test data encoded by combination with statistical coding and run-length coding.

both the codings. The encoding algorithm using the interleaving and both the codings is shown in the following and illustrated in Fig. 4:

1. Interleave test data and divide the interleaved test data into blocks of fixed-length as discussed in the previous section.
2. Fill don't care bits so that the test data comprise many long runs of blocks and compress the test data with run-length coding. Here, the compressed data contains pairs of blocks and corresponding lengths of runs as shown in Fig. 4.
3. Compress the blocks and the lengths of runs with statistical coding.

4. Test Architecture

4.1 Outline of Test Architecture

This section presents test architecture for two-pattern testing using the proposed interleaving. Figure 5 illustrates a conceptual diagram of the test architecture. The architecture mainly comprises an external ATE, a decompressor, a DUT with Chiba scan design and a compactor. The decompressor not only receives compressed test data but also sends and receives synchronous signal to and from the external ATE. The decompressor also sends not only decompressed test data but synchronous signal to the DUT. It is just like the existing decompressors for variable-length coding such as statistical coding [13]. The conventional ATE test programs for test data compressed with variable-length coding are available to ATE test programs for the proposed method. The compactor is capable of compacting test responses. This paper focuses on only test compression and does not discuss test response compaction. Conventional test response compactors such as [14] are available in the test architecture.

4.2 Decompressor for Statistical Coding with Proposed Interleaving

Figure 6 illustrates a decompressor for statistical coding using the proposed interleaving. The decompressor consists of a decompressor for the statistical coding, a de-interleaver and a controller. The decompressor for the statistical coding can be constructed by the conventional construction such as [13], [15]. Decompressors for statistical coding always contain a frequency table, and the area of the frequency table often occupies a large percentage of the area of the decompressor for statistical coding. The de-interleaver comprises two FIFO memories and a multiplexer.

Next, the process of two-pattern testing on the proposed test architecture using the decompressor shown in Fig. 6 is explained. The testing is made in the following four phases.

Phase 1: The external ATE sends the decompressor for statistical coding a compressed block composed of the following three: even-numbered bits of an initial vector, odd-numbered bits of the initial/transition vector, and even-numbered bits of the transition vector. After the decompressor decompresses the block, the even-numbered bits of the initial vector are sent to the DUT. At the same time, the odd-numbered bits of the initial/transition vector are sent to a FIFO memory (FIFO memory 1), and the even-numbered bits of the transition vector are sent to the other FIFO memory (FIFO memory 2).

Phase 2: The odd-numbered bits of the initial/transition vector are sent from the FIFO memory 1 to the DUT.

Phase 3: The even-numbered bits of the transition vector are sent from the FIFO memory 2 to the DUT.

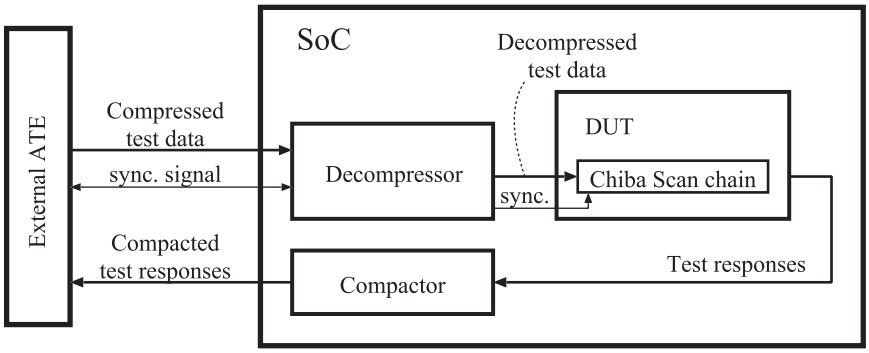


Fig. 5 Test architecture.

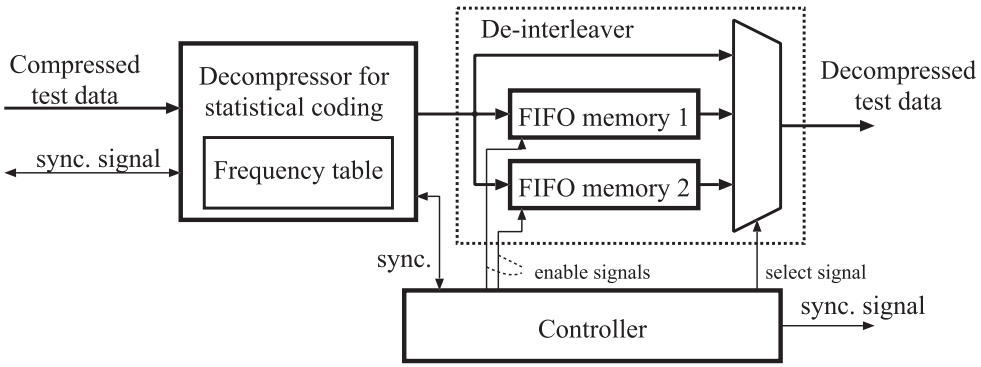


Fig. 6 Decompressor for statistical coding using proposed interleaving.

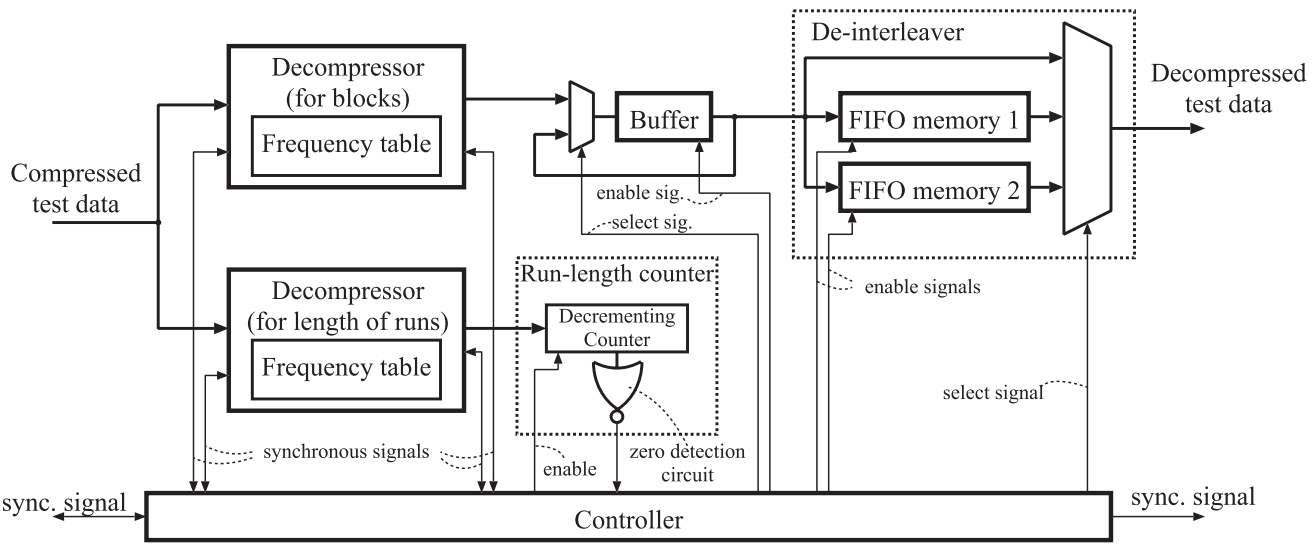


Fig. 7 Decompressor for combination of statistical and run-length coding using proposed interleaving.

Phase 4: The test response is captured, compacted by the compactor, and observed in the ATE.

The four phases are performed sequentially and not in parallel. The phase 4 for a test vector pair and the phase 1 for the next test vector pair can be made in parallel. It is noted that no data are read out from the FIFO memories in the phase 1. In addition, no data are written into the FIFO memories in the phases 2 and 3. No data are read out from and written into the FIFO memories in the phase 4.

4.3 Decompressor for Combination of Statistical and Run-Length Coding with Proposed Interleaving

Figure 7 illustrates a decompressor for the combination of statistical and run-length coding using the proposed interleaving. The decompressor consists of two decompres-

sors for the statistical coding, a de-interleaver, a run-length counter, a buffer, and a controller. As shown in Sect. 3.2 and Fig. 4, compressed test data consist of two types of informations, namely blocks and lengths of runs, and the two decompressors correspond to the respective types. The construction of the de-interleaver is the same as that shown in Fig. 6. The run-length counter consists of a decrementing counter and a zero detection circuit, i.e. an NOR gate.

Next, the process of two-pattern testing using the decompressor shown in Fig. 7 is explained. The process is almost the same as that for the decompressor shown in Fig. 6. The phase 1 differs from each other while the phases 2, 3, 4 are the same. The phase 1 is explained in the following. First, the external ATE sends a compressed block to the decompressor for blocks. The decompressor decompresses it and sends it to the buffer. Next, the external ATE sends a compressed length of runs to the decompressor for lengths of runs. The decompressor decompresses it and sends it to the decrementing counter in the run-length counter. The block stored in the buffer is sent to the de-interleaver and the value in the decrementing counter is decremented. If the value in the counter is not zero, the block is sent and the value in the counter is decremented again. It is repeated until the value in the counter becomes zero. In sum, the decompressed block is sent to the de-interleaver l times, where l is the length of runs. In the de-interleaver, the even-numbered bits of the initial vector are sent to the DUT while the odd-numbered bits of the initial/transition vector and the even-numbered bits of the transition vector are stored in the FIFO memories just like the test process for the decompressor shown in Fig. 6.

4.4 De-Interleaver Using Embedded RAM

The de-interleavers contain two FIFO memories as shown in Figs. 6 and 7. The size of each FIFO memory is half as large as the length of the scan chain in the DUT, and thus the area of the FIFO memories is too large to be used in actual systems as quantitatively shown in Sect. 5. Recent SoCs usually contain embedded RAMs [16], and it is desirable that the RAMs are used as the FIFO memories in order to reduce area overhead.

Figure 8 illustrates construction of a de-interleaver using an embedded RAM. Figure 8 (a) shows outline of construction. For ease of illustration, the data path that test data pass is simplified in the figure. The de-interleaver consists of the embedded RAM and some multiplexers. The controller in the decompressor is designed to control the RAM as FIFO memories. During testing, in order to make the RAM work as FIFO memories, interleaved test data are input to the data input of the RAM and control signals which the controller outputs are input to the address and enable inputs of the RAM. Meanwhile, in normal operation, the outputs of functional circuits, i.e. the original inputs of the embedded RAM, are selected as the input of the RAM and the RAM is used for the original purpose. The outputs of the RAM are used as the outputs of FIFO memories in testing operation while they are used as the inputs of functional circuits, i.e. the original outputs of the embedded RAM, in normal operation.

Figure 8 (b) illustrates the detail of the data path on the de-interleaver. The de-interleaver can receive the following three bits in parallel: an even-numbered bit of an initial vector, the odd-numbered bit of the initial/transition vector and the even-numbered bit of the transition vector. Even-numbered bits of initial vectors can be output through the multiplexer at the output of the de-interleaver immediately. The others can be input to the RAM as the first and zeroth bits in input data, respectively. The stored data can be output as the first and zeroth bits of output data of the RAM and output from the de-interleaver through the multiplexer. Note that, if the decompressor for the statistical coding in Fig. 6 and the buffer in Fig. 7 are not send the three bits to the de-interleaver in parallel, it must be adjusted such that the de-interleaver receives the three bits in parallel. Circuits adjusting it are not illustrated in the figures.

Next, the operations in the de-interleaver are explained. As discussed above, the testing is made in the four phases. In the phase 1, the de-interleaver receives the three bits in parallel. The even-numbered bit of the initial vector is sent to the DUT. At the same time, the other two bits are written in the RAM. Note that they are written at the same address. In the phase 2, the odd-numbered bits of the initial/transition vectors are output from the RAM as the first bit of output data, and sent to DUT. Likewise, in the phase 3, the even-numbered bits of the transition vectors are output from the RAM as the zeroth bit of output data, and sent to DUT. The de-interleaver is not used in the phase 4.

As shown in Fig. 8, on every functional path from the functional circuits to the RAM, there is at most one extra multiplexer. It is required that system integrators construct SoCs with the delay for the extra multiplexer in mind in case that they use the proposed de-interleavers. As mentioned above, in the test process for the proposed method, reading and writing FIFO memories are not made simultaneously. So, we can use ordinary RAMs where reading and writing cannot be made simultaneously.

4.5 Consideration

Test data may contain some one-pattern test vectors, such as stuck-at fault test vectors, as well as delay fault testing two-pattern test vectors. To apply one-pattern test vectors into DUT on the test architecture, it is advisable that one-pattern test vectors are compressed without the interleaving. When the decompressor decompresses the one-pattern test vectors, the input of the de-interleaver is chosen as the output of the de-interleaver and decompressed test data bypasses the de-interleaver.

Decompression can be made on external ATEs and not on-chip decompressors. In that case, test architecture contains no on-chip decompressors and ATE programs emulate the operation of the on-chip decompressors discussed above.

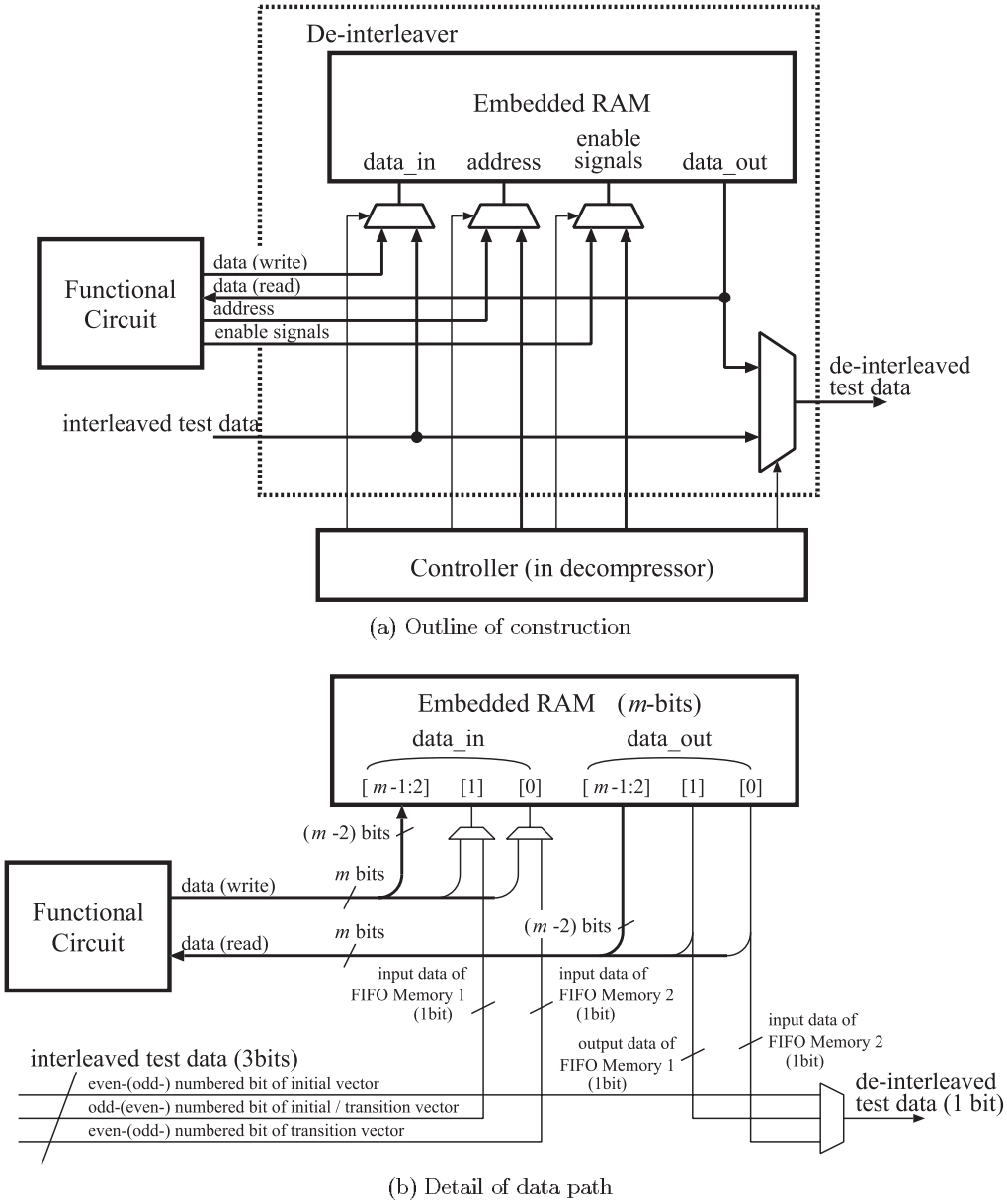


Fig. 8 Construction of de-interleaver using embedded RAM.

5. Evaluation

Table 1 summarizes notations used in the tables showing the evaluation results.

5.1 Benchmark Circuits and Test Data

Table 2 shows the data on ISCAS89 benchmarks and test data for robust testable path delay fault testing used in the evaluation. The test data are obtained by the robust testable path delay fault test generation algorithm shown in [10]. The test data are dynamically compacted by the test generation algorithm. The fault coverage of every test data except those for s9234, s13207 and s38584 is 100%. The test data for the three circuits can not detect some testable faults because of timeout on ATPG. The ST, TR and X columns show the number ratio of stable values S0 and S1 to all pairs of values in test data, that of transitions T0 and T1, and that of unspecified values U0, U1 and XX, respectively. Note that values of either even-numbered or odd-numbered bits of test vectors have to be stable in Chiba scan design, and the stable values are not included in the three columns. The average ratio of stable values is 26.5%. That of transitions is 10.9% and smaller than that of stable values.

5.2 Test Data Compression Rates

In the evaluation, compression rates are calculated by the following formula:

Table 1 Notations used in evaluation.

H	result for Huffman coding
H+R	result for the combination of Huffman coding and run-length coding
b	block size for Huffman coding
w.I	result for Huffman coding with the proposed interleaving
w/o	result for Huffman coding without the interleaving
w.D	result for decompressor using de-interleaver with FIFO memories dedicated to testing
w.E	result for decompressor using de-interleaver with embedded RAMs
R.L.	result for run-length coding [5]
G.	result for Golomb coding [6]
FDR	result for Frequency-directed run-length (FDR) coding [7]
VIHC	result for variable-length input Huffman coding [8]

Table 2 ISCAS89 benchmarks and robust testable path delay fault test data.

Name	#S	#G	#PDF	#TP	#TB	ST	TR	X
s298	14	119	343	102	2,142	26.2	25.1	48.7
s344	15	160	611	193	4,364	39.4	20.5	40.1
s349	15	161	611	181	4,091	40.4	20.7	38.9
s382	21	158	667	181	5,698	27.8	19.6	52.6
s400	21	162	665	172	5,415	28.3	19.0	52.7
s420.1	16	218	948	518	12,432	57.0	14.7	28.3
s444	21	181	586	156	4,912	26.4	22.2	51.4
s526n	21	194	695	190	5,989	33.5	21.8	44.7
s641	19	379	1,979	280	7,919	24.9	8.8	66.3
s713	19	393	1,184	166	4,693	22.9	8.8	68.4
s838.1	32	446	3,428	1,812	86,976	60.9	7.9	31.3
s953	29	395	2,302	723	31,339	13.5	6.6	79.9
s1196	18	529	3,581	698	18,846	9.7	1.8	88.5
s1238	18	508	3,589	682	18,414	9.8	2.0	88.2
s1423	74	657	28,696	5,921	657,231	33.2	4.6	62.1
s5378	179	2,779	18,618	1,119	300,935	32.0	4.0	64.0
s9234	228	5,597	21,335*	2,682	917,244	16.1	1.7	82.2
s13207	669	7,951	27,088*	3,745	3,758,042	3.0	0.5	96.5
s35932	1,728	16,065	21,207	154	399,168	19.7	6.8	73.5
s38584	1,452	19,253	91,815*	3,215	7,002,270	4.8	0.8	94.4

#S, number of scan cells; #G, number of gates
#PDF, number of robust testable path delay faults
* Some testable faults are not included because of timeout on ATPG.
#TP, number of pairs of test vectors
#TB, number of bits in original test data
ST, number ratio of S0 and S1 to all pairs of values
TP, number ratio of T0 and T1 to all pairs of values
XX, number ratio of U0, U1 and XX to all pairs of values

$$1 - \frac{\text{(the number of bits in compressed test data)}}{\text{(the number of bits in original test data)}}.$$

Table 3 shows a comparison of compression rates between compressions with the proposed interleaving and ones without it. In the experiment, unspecified bits are filled by the algorithm shown in [4]. For 45 out of 60 cases with Huffman coding and 46 out of 60 cases with the combination of both the coding, compression rates for test compression with the interleaving are higher than those without the interleaving. Especially, in all of the cases where the number of pairs of test vectors is larger than 1,000, compression rates for test compression with the interleaving are higher than those without the interleaving.

Figure 9 shows relation between the number ratio of unspecified values U0, U1 and XX and the differences between compression rates of the proposed two methods, namely Huffman coding with the interleaving and the combination of Huffman and run-length coding with the interleaving. When the number ratio of unspecified values is large, the combination of both the coding brings better result than Huffman coding. It is because the large numbers of unspecified values make many long runs.

Table 4 shows a comparison between the compression rates for the proposed methods and those for the existing test compressions, namely Huffman coding and the combination of Huffman and run-length coding not using the interleaving, run-length coding [5], Golomb coding [6], Frequency-directed run-length (FDR) coding [7], and variable-length input Huffman coding (VIHC) [8]. The block size for Huffman coding b is set to nine. For 11 out of 20 circuits, the proposed interleaving affords the highest compression rates.

Figure 10 shows relation between the number ratio of one-pattern test vectors and the differences between compression rates of Huffman coding with interleaving and that without interleaving. Test data used in the evaluation consist of a part of stuck-at fault testing one-pattern test data and a part of robust testable path delay fault testing two-pattern test data. The ratio of unspecified value in the parts of the

Table 3 Compression rates (%) of robust testable path delay fault test data for ISCAS89 benchmarks.

		H						H+R					
		$b = 3$		$b = 6$		$b = 9$		$b = 3$		$b = 6$		$b = 9$	
Name	#TP	w.I	w/o	w.I	w/o	w.I	w/o	w.I	w/o	w.I	w/o	w.I	w/o
s298	102	**24.3**	13.2	**29.7**	24.1	42.0	**43.2**	21.6	**22.4**	**31.1**	27.0	**44.1**	34.3
s344	193	**17.8**	15.9	**30.2**	20.4	35.1	**37.2**	**15.4**	9.5	**25.2**	17.0	**31.0**	30.3
s349	181	**18.7**	14.7	**31.0**	22.5	35.6	**38.4**	**17.9**	11.7	**25.9**	20.4	31.9	**32.2**
s382	181	**26.1**	25.7	31.3	**37.8**	**43.2**	32.2	**25.1**	24.0	32.7	**33.1**	**43.3**	36.7
s400	172	**27.0**	26.4	33.5	**39.0**	**44.4**	33.8	25.6	**25.8**	**35.5**	34.7	**44.2**	39.7
s420.1	518	**34.4**	29.0	**44.7**	34.5	**47.1**	45.7	**34.2**	25.7	**41.1**	35.0	43.2	**44.9**
s444	156	**22.9**	22.7	**34.5**	29.7	**42.0**	30.7	**23.6**	23.0	**36.0**	32.4	**43.2**	39.5
s526n	190	**25.1**	17.4	**28.8**	27.1	**37.7**	29.9	**17.7**	14.3	**28.0**	27.5	**37.3**	33.6
s641	280	**39.0**	37.4	48.5	**48.9**	51.1	**51.9**	**41.4**	33.7	**48.8**	47.1	51.0	**58.6**
s713	166	40.4	**41.1**	**52.4**	51.4	**56.5**	55.4	**43.8**	36.8	**52.7**	51.0	56.7	**61.4**
s838.1	1,812	**45.6**	40.6	**60.5**	56.8	**63.0**	60.0	**57.2**	47.8	**59.8**	54.2	**62.4**	58.8
s953	723	53.4	**53.6**	66.9	**67.4**	**75.7**	74.4	74.7	**78.6**	76.7	**78.4**	**84.5**	78.4
s1196	698	**60.9**	59.7	**75.3**	71.1	80.4	**81.4**	78.3	**78.5**	81.0	**83.6**	84.3	**87.5**
s1238	682	**61.2**	59.9	**75.3**	71.2	81.5	**81.9**	**78.7**	77.6	81.1	**82.4**	84.3	**86.6**
s1423	5,921	**40.7**	33.6	**49.7**	44.2	**53.0**	48.0	**41.9**	29.7	**46.9**	42.9	**53.2**	46.9
s5378	1,119	**43.5**	33.3	**50.7**	42.5	**54.4**	46.5	**46.0**	33.5	**51.0**	43.0	**54.0**	47.0
s9234	2,682	**57.2**	54.0	**70.4**	65.5	**72.9**	68.5	**69.5**	62.4	**73.6**	68.4	**74.9**	69.3
s13207	3,745	**64.8**	64.5	**80.7**	80.6	**86.2**	85.7	**94.0**	92.2	**94.6**	93.6	**94.8**	93.3
s35932	154	57.0	**62.1**	77.4	**79.4**	82.6	**84.2**	**93.9**	89.7	**93.6**	90.7	**91.4**	90.5
s38584	3,215	**63.3**	63.1	**78.7**	78.3	**83.7**	83.1	**88.8**	86.5	**89.9**	88.3	**90.6**	89.1
Average		**41.2**	38.4	**52.5**	49.6	**58.4**	55.6	**49.5**	45.2	**55.2**	52.5	**60.0**	57.9

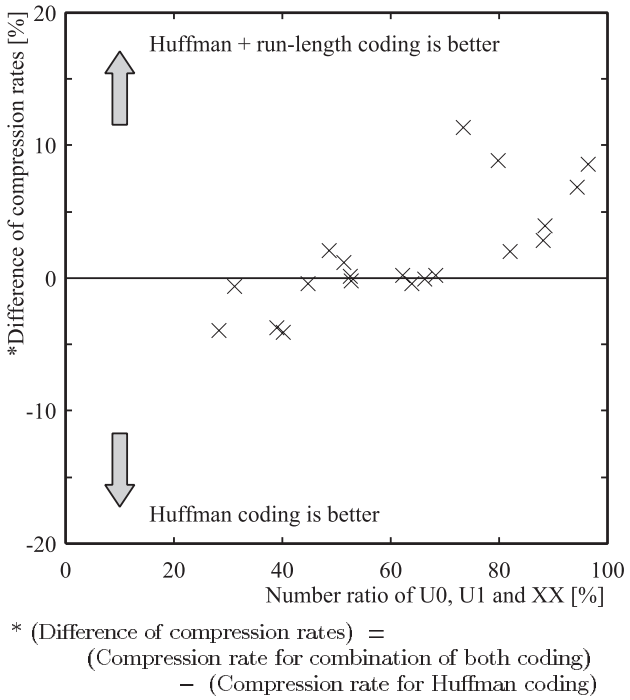


Fig. 9 Number ratio of unspecified values vs. Difference between compression rates of proposed two methods.

test data is made to be about the same as that in the original test data. The result indicates that if test data include many one-pattern test vectors, the effect of the interleaving upon test compression rates is low.

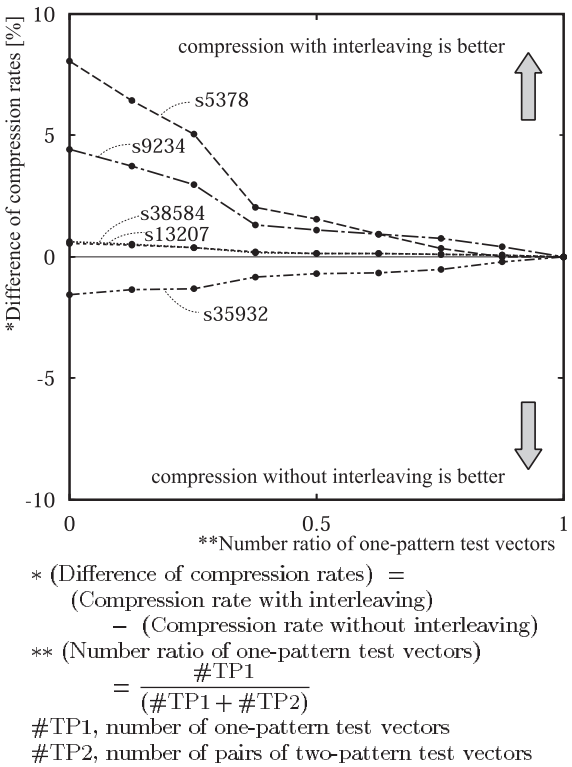
5.3 Test Application Time

Table 5 shows a comparison between the test application times with the proposed interleaving and ones without it. In the evaluation, operations are pipelined as much as possible. On-chip decompressors can usually work faster than external ATEs, and the table presents test application time under several ratios of the operation speed (clock frequency) of decompressors to that of ATEs. The "CLK" column shows the clock ratio of decompressors to ATEs. The w.I and w/o columns show the ratio of the test application time for compressed test data to that for uncompressed test data. The test application time with the proposed interleaving is an average of 6.5% longer than that without the proposed interleaving.

The reason why the use of the proposed interleaving increases the application time though it reduces the number of bits in test data is as follows. In decompression without the proposed method, sending of compressed test data from the external ATE to the decompressor, decompression in the decompressor and scan-shift operation can be pipelined, and made at the same time. As discussed in Sect. 4, decompression with the proposed method is made in the four phases. In the phase 1, they can be pipelined like decompression without the proposed method, except as follows. In the decompression without the proposed interleaving, the following three are scanned into scan chains at the same time as the sending and the decompression: an even-(odd-)numbered bit of an initial vector, the odd-(even-)numbered bit of the initial/transition vector, and the even-(odd-)numbered bit of the transition vector. Meanwhile, in the phase 1 of the decompression with the proposed interleaving, only an even-(odd-)numbered bit of an initial vector is scanned into scan chain at the same time as the sending and the decompression. The remaining two are scanned into in the phases 2 and 3, i.e. at different time from the sending and the decompression. To make it at different time causes longer test application time for the proposed method.

Table 4 Comparison of compression rates (%) with existing test compressions.

Name	#TP	w.I		w/o		R.L.	G.	FDR	VI-HC
		H	H+R	H	H+R				
s298	102	42.0	**44.1**	43.2	34.3	18.1	15.7	11.0	22.9
s344	193	35.1	31.0	**37.2**	30.3	15.8	18.4	10.7	20.3
s349	181	35.6	31.9	**38.4**	32.2	14.4	16.5	8.5	17.7
s382	181	43.2	**43.3**	32.2	36.7	22.5	24.4	22.1	29.2
s400	172	**44.4**	44.2	33.8	39.7	23.6	26.4	23.8	30.5
s420.1	518	47.1	43.2	45.7	44.9	41.6	42.7	32.8	**50.5**
s444	156	42.0	**43.2**	30.7	39.5	21.1	21.6	20.4	26.4
s526n	190	**37.7**	37.3	29.9	33.6	14.0	13.5	10.8	18.4
s641	280	51.1	51.0	51.9	**58.6**	38.5	40.9	38.2	43.9
s713	166	56.5	56.7	55.4	**61.4**	39.2	40.9	37.5	44.1
s838.1	1,812	63.0	62.4	60.0	58.8	62.2	62.4	54.5	**69.6**
s953	723	75.7	**84.5**	74.4	78.4	62.4	60.9	56.1	74.0
s1196	698	80.4	84.3	81.4	**87.5**	78.8	80.9	77.8	82.8
s1238	682	81.5	84.3	81.9	**86.6**	77.4	79.7	76.6	81.7
s1423	5,921	53.0	**53.2**	48.0	46.9	40.1	40.4	40.9	45.6
s5378	1,119	**54.4**	54.0	46.5	47.0	38.0	41.2	44.9	46.8
s9234	2,682	72.9	74.9	68.5	69.3	71.4	73.1	74.0	**76.6**
s13207	3,745	86.2	**94.8**	85.7	93.3	92.5	92.6	92.0	94.6
s35932	154	82.6	**91.4**	84.2	90.5	65.3	58.9	78.3	86.0
s38584	3,215	83.7	**90.6**	83.1	89.1	84.2	86.1	87.7	88.9
Average		58.4	**60.0**	55.6	57.9	46.1	46.9	44.9	52.5



$*$ (Difference of compression rates) $=$
(Compression rate with interleaving)
$-$ (Compression rate without interleaving)

$**$ (Number ratio of one-pattern test vectors)
$= \dfrac{\#\mathrm{TP1}}{(\#\mathrm{TP1}+\#\mathrm{TP2})}$

#TP1, number of one-pattern test vectors
#TP2, number of pairs of two-pattern test vectors

Fig. 10 Ratio of one-pattern test vectors in test data vs. Difference between compression rates for Huffman coding with and without interleaving.

5.4 Area Overhead of Decompressors

Table 6 shows area of the on-chip decompressors for statistical coding with and without the proposed interleaving. The area is obtained by applying decompressors described in Verilog-HDL to Synopsis Design Compiler. The frequency tables in the Huffman decompressors are constructed from smaller of either combinational circuits or ROMs. The "Area" sub-column in the "DUT" column shows the area ratio of the DUT to an inverter. The values outside parentheses in the "Area of decompressor" columns indicate the area ratio of the decompressor to an inverter, and those in parentheses indicate the area ratio (%) of the decompressor to the DUT. The "Total" sub-column shows the total area of the decompressor. The "FIFO" sub-column shows the area of the FIFO memories. The "Freq. table" column shows the area of the frequency table in the Huffman decompressors. The decompressors for the combination of Huffman and run-length coding comprise two Huffman decompressors, and the column shows the sum of areas of the two frequency tables in the decompressors. The "other" sub-column shows the total area of all parts consisting of the decompressor other than the FIFO memories and the frequency tables.

For decompressors with de-interleavers using embedded RAMs and those for compression without the proposed interleaving, increase in the size of DUTs makes little impact on the area of the decompressors and reduces the area ratio to the DUTs. Meanwhile, for decompressors with dedicated FIFO memories, the area ratio of the decompressor to the DUT is large, regardless of the size of the DUT because the increase in the size of DUTs does not reduce the area ratio of FIFO memories to the DUTs. The area of decompressors using embedded RAMs is, on average, 39% larger than that of decompressors for compression without the proposed interleaving.

The logic values in the frequency tables for decompressors using dedicated FIFO memories are the same as those for decompressors using embedded RAMs. However, the areas of both frequency tables are not always the same because the tables are sometimes constructed from combina-

Table 5 Test application time.

Name	CLK	H						H+R					
		$b = 3$		$b = 6$		$b = 9$		$b = 3$		$b = 6$		$b = 9$	
		w.I	w/o	w.I	w/o	w.I	w/o	w.I	w/o	w.I	w/o	w.I	w/o
s5378	1	1.22	1.12	1.24	1.05	1.21	1.03	1.28	1.15	1.21	1.07	1.16	1.04
	2	0.884	0.852	0.814	0.705	0.828	0.671	0.878	0.801	0.819	0.687	0.787	0.658
	4	0.720	0.635	0.650	0.623	0.610	0.548	0.692	0.681	0.642	0.576	0.612	0.538
	8	0.638	0.635	0.568	0.541	0.528	0.512	0.606	0.650	0.557	0.555	0.528	0.517
s9234	1	1.09	1.03	1.08	1.02	1.09	1.04	1.11	1.04	1.06	1.03	1.04	1.05
	2	0.754	0.715	0.618	0.575	0.664	0.618	0.668	0.605	0.623	0.561	0.613	0.577
	4	0.589	0.445	0.453	0.435	0.424	0.439	0.467	0.421	0.425	0.370	0.413	0.374
	8	0.507	0.445	0.370	0.320	0.342	0.354	0.374	0.359	0.336	0.310	0.322	0.303
s38584	1	1.03	1.01	1.03	1.01	1.03	1.01	1.03	1.02	1.02	1.01	1.01	1.01
	2	0.697	0.678	0.540	0.519	0.589	0.571	0.550	0.533	0.537	0.521	0.533	0.521
	4	0.530	0.363	0.374	0.358	0.324	0.360	0.317	0.303	0.305	0.291	0.300	0.287
	8	0.447	0.363	0.290	0.208	0.240	0.255	0.204	0.198	0.193	0.184	0.188	0.179

CLK, ratio of clock frequency of decompressor to that of external ATE

Table 6 Area of decompressor.

(a) Area of decompressor for w.I, w.D

DUT		Coding		Area of decompressor (for w.I, w.D)							
Name	Area		b	Total		FIFO		Freq. table		other	
s5378	5,902	H	3	2,142	(36.3)	1,727	(29.3)	46	(0.8)	369	(6.3)
			6	2,864	(48.5)	1,727	(29.3)	604	(10.2)	532	(9.0)
			9	4,327	(73.3)	1,727	(29.3)	1,924	(32.6)	676	(11.5)
		H+R	3	3,277	(55.5)	1,727	(29.3)	812	(13.8)	738	(12.5)
			6	3,633	(61.6)	1,727	(29.3)	1,091	(18.5)	815	(13.8)
			9	4,903	(83.1)	1,727	(29.3)	2,204	(37.3)	971	(16.5)
s9234	9,887	H	3	2,607	(26.4)	2,188	(22.1)	48	(0.5)	371	(3.8)
			6	3,321	(33.6)	2,188	(22.1)	597	(6.0)	537	(5.4)
			9	4,784	(48.4)	2,188	(22.1)	1,924	(19.5)	673	(6.8)
		H+R	3	3,910	(39.5)	2,188	(22.1)	981	(9.9)	742	(7.5)
			6	4,217	(42.7)	2,188	(22.1)	1,214	(12.3)	816	(8.2)
			9	5,447	(55.1)	2,188	(22.1)	2,288	(23.1)	971	(9.8)
s38584	48,801	H	3	14,754	(30.5)	14,456	(29.9)	48	(0.1)	250	(0.5)
			6	15,512	(32.0)	14,456	(29.9)	645	(1.3)	412	(0.9)
			9	17,108	(35.3)	14,456	(29.9)	1,924	(4.0)	728	(1.5)
		H+R	3	18,655	(38.5)	14,456	(29.9)	3,241	(6.7)	957	(2.0)
			6	17,325	(35.8)	14,456	(29.9)	1,911	(3.9)	958	(2.0)
			9	18,730	(38.7)	14,456	(29.9)	3,191	(6.6)	1,083	(2.2)

(b) Area of decompressor for w.I, w.E; for w/o

DUT	Coding		Area of decompressor (for w.I, w.E)						Area of decompressor (for w/o)					
Name		b	Total		Freq. table		other		Total		Freq. table		other	
s5378	H	3	449	(7.6)	46	(0.8)	403	(6.8)	275	(4.7)	56	(1.0)	219	(3.7)
		6	1,211	(20.5)	647	(11.0)	564	(9.6)	981	(16.6)	627	(10.6)	354	(6.0)
		9	2,607	(44.2)	1,924	(32.6)	683	(11.6)	2,412	(40.9)	1,924	(32.6)	487	(8.3)
	H+R	3	1,570	(26.6)	790	(13.4)	780	(13.2)	1,049	(17.8)	502	(8.5)	546	(9.3)
		6	1,911	(32.4)	1,061	(18.0)	850	(14.4)	1,495	(25.3)	860	(14.6)	635	(10.8)
		9	3,198	(54.2)	2,203	(37.3)	994	(16.8)	2,872	(48.7)	2,108	(35.7)	764	(12.9)
s9234	H	3	449	(4.5)	47	(0.5)	402	(4.1)	263	(2.7)	44	(0.4)	219	(2.2)
		6	1,145	(11.6)	578	(5.8)	567	(5.7)	1,009	(10.2)	653	(6.6)	355	(3.6)
		9	2,612	(26.4)	1,924	(19.5)	688	(7.0)	2,414	(24.4)	1,924	(19.5)	489	(4.9)
	H+R	3	1,760	(17.8)	981	(9.9)	780	(7.9)	459	(4.6)	87	(0.9)	371	(3.8)
		6	2,050	(20.7)	1,204	(12.2)	847	(8.6)	1,172	(11.9)	639	(6.5)	533	(5.4)
		9	3,269	(33.1)	2,268	(22.9)	1,001	(10.1)	2,655	(26.9)	1,961	(19.8)	695	(7.0)
s38584	H	3	524	(1.1)	51	(0.1)	473	(1.0)	249	(0.5)	30	(0.1)	219	(0.5)
		6	1,303	(2.7)	663	(1.4)	640	(1.3)	1,003	(2.1)	650	(1.3)	353	(0.7)
		9	2,681	(5.5)	1,924	(4.0)	757	(1.6)	2,410	(5.0)	1,924	(4.0)	485	(1.0)
	H+R	3	4,242	(8.8)	3,241	(6.7)	1,001	(2.1)	3,931	(8.1)	3,232	(6.7)	699	(1.4)
		6	2,896	(6.0)	1,891	(3.9)	1,005	(2.1)	3,343	(6.9)	2,590	(5.4)	753	(1.6)
		9	4,335	(9.0)	3,191	(6.6)	1,143	(2.4)	4,052	(8.4)	3,191	(6.6)	861	(1.8)

Freq. table, frequency table in Huffman decoder

Area (outside parenthesis), Area ratio to an inverter; Area (in parenthesis), Area ratio to DUT (%)

Table 7 Comparison of area of decompressor (area ratio of decompressor to DUT (%)).

	w.I†		w/o					
Name	H	H+R	H	H+R	R.L.†	G.†	FDR†	VIHC†
s5378	44.2	54.2	40.9	48.7	7.3	10.6	15.6	36.7
s9234	26.4	33.1	24.4	26.9	5.2	6.5	9.3	26.1
s38584	5.5	9.0	5.0	8.4	1.4	1.7	2.4	25.6

† area ratio of decompressors using an embedded RAMs

Table 8 Line number in source files for decompressor.

H†			H+R†						
w.I			w.I						
w.D	w.E	w/o	w.D	w.E	w/o	R.L.	G.	FDR	VIHC†
175	192	146	240	257	216	140	141	155	136

† frequency tables in Huffman decompressors are not included

tional circuits and the Design Compiler sometimes generates different net-lists from the same source file considering several factors. The frequency tables in the decompressors for the compression with the proposed interleaving differ from those for the compression without the proposed interleaving.

Table 7 shows the area ratio of decompressors to DUTs for the compression with the proposed interleaving and the existing test compressions. In the evaluation, embedded RAMs and not dedicated FIFO memories are used for the proposed interleaving. The block size b for Huffman coding is set to nine. Just like the proposed interleaving, the four existing test compressions, run-length coding, Golomb coding, FDR coding, and VIHC, require large FIFO memories in the decompressors. For fair comparison, embedded RAMs are used as the FIFO memories by a similar way to the proposed interleaving. The area ratio of the decompressors for the proposed method is larger than that for run-length, Golomb and FDR coding.

5.5 Discussion on Design Time for Decompressor

For estimating design time for the decompressors, Table 8 shows the line numbers in the Verilog source files for the proposed and existing decompressors used in the above evaluations. As the frequency tables in the Huffman decompressors are automatically generated by the Huffman compression tool, the line numbers for the frequency tables are not included in the results. The results are independent of the targeted benchmark circuits. The line number for Huffman coding (the combination of Huffman and run-length coding) with the interleaving in case that an embedded RAM are used is 46 (41) lines larger than that without the interleaving and 17 (17) lines larger than that using dedicated FIFO memories in the de-interleavers.

5.6 Consideration

Here, from the above results, it is discussed when system integrators should use the statistical codings (Huffman coding/the combination of Huffman and run-length coding) with the proposed interleaving and not the conventional codings, namely statistical codings without the proposed interleaving, run-length coding, Golomb coding, FDR coding, and VIHC.

For any codings other than the statistical codings without the proposed interleaving, large FIFO memories are required in the decompressors. So, if the target SoC includes no embedded RAMs or it is hard to use embedded RAMs as the FIFO memories, the codings except the statistical codings without the proposed interleaving are not practically available. In later paragraphs, it is assumed that there are embedded RAMs usable as the FIFO memories in SoCs.

As shown in Sect. 5.4, for not large DUTs such as s5378, the area ratio of the decompressors for the statistical codings are large, and thus the statistical codings are not practically available regardless of the use of the proposed interleaving. The statistical codings are available for only large DUTs such as s38584 or larger. In later paragraphs, it is assumed that the area of DUTs is large enough to practically use the statistical codings.

The run-length, Golomb and FDR codings require low area overhead for decompressors. However, they achieve only low compression rates. Meanwhile the combination of Huffman and run-length coding achieves higher compression rates, but it requires higher area overhead than the three codings regardless of the use of the proposed interleaving. System integrators may decide to use either the three conventional codings or the combination of Huffman and run-length coding by considering the trade-off between compression rates and area overhead. Note that, for large DUTs, the compression rates for Huffman coding are not high as compared with run-length, Golomb and FDR codings though the decompressor for Huffman coding is larger than that for the three codings. The area ratio of decompressors for VIHC is large and not practically usable regardless of the size of DUTs. So, it is unwise for system integrators to select Huffman coding or VIHC.

If system integrators decide to use the combination of Huffman and run-length coding, they have to choose whether or not to use the proposed interleaving. As mentioned in Sect. 5.2, the use of the proposed interleaving provides higher compression rates. For example, for the three largest benchmarks, the bit numbers in the compressed data using the proposed interleaving are, on average, 18% smaller than those not using the proposed inter-

leaving. However, the use increases the area of decompressors and path delays on functional path, especially between the embedded RAM and functional circuits. Moreover it slows down design time for decompressors. For example, for s38584, the use increases the area of decompressors by 0.6% of that of DUTs. As discussed in Sect. 4, for every functional path from the functional circuits to the RAM, an extra multiplexer is inserted and the path delays increase. Moreover, the line number in the Verilog source files increases by 41. Test application time also increases by 6.5%, on average. System integrators should decide to use the proposed interleaving by considering the trade-off.

6. Conclusion

This paper has proposed a method providing efficient test compression. The proposed compression is for delay fault testing using a scan design facilitating two-pattern testing called Chiba scan design. In the proposed method, test data are interleaved before test compression using statistical coding. This paper has also shown the combination of the interleaving, statistical coding and run-length coding. In this paper, test architecture for two-pattern testing using the proposed method has also been shown. The proposed method has been experimentally evaluated from several viewpoints such as test data compression rates, test application time and area overhead. The experimental results give evidence that the proposed method often makes test compression more efficient. For 11 out of 20 ISCAS89 benchmark circuits, the compression rates for the proposed method are better than those for the existing test compression such as Huffman coding, run-length coding, Golomb coding, FDR coding and VIHC.

While the proposed interleaving reduces the bits in compressed data and required memory size in external ATE, test application time increases as long as the presented test architecture are used. Development of test architecture reducing test application time is an important future work.

Acknowledgment

The authors gratefully thank a graduate of our laboratory Hayahiko Koizumi for his valuable comments and suggestions on earlier versions of this paper. The authors also gratefully thank the anonymous reviewers whose comments have improved the quality of this paper. This work is supported by VLSI Design and Education Center (VDEC), the University of Tokyo in collaboration with Synopsis, Inc. and Cadence Design Systems, Inc. The authors' researches were partially supported by grants from Kurata Memorial Hitachi Science and Technology Foundation and CASIO Science Promotion Foundation.

References

[1] J. Rajski, "DFT for high-quality low cost manufacturing test," Proc. IEEE Asian Test Symp., pp.3–8, 2001.

[2] B. Koenemann, C. Barnhart, B. Keller, T. Snethen, O. Farnsworth, and D. Wheater, "A SmartBIST variant with guaranteed encoding," Proc. IEEE Asian Test Symp., pp.325–330, 2001.

[3] J. Rajski, J. Tyszer, M. Kassab, and N. Mukherjee, "Embedded deterministic test," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol.23, no.5, pp.776–792, May 2004.

[4] A. Jas, J. Ghosh-Dastidar, N. Mom-Eng, 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.

[5] A. Jas and N.A. Touba, "Test vector decompression via cyclical scan chains and its application to testing core-based designs," Proc. IEEE Int'l Test Conf., pp.458–464, 1998.

[6] A. Chandra and K. Chakrabarty, "Test data compression for system-on-a-chip using Golomb codes," Proc. IEEE VLSI Test Symp., pp.113–120, 2000.

[7] A. Chandra and K. Chakrabarty, "Frequency-directed run-length (FDR) codes with application to system-on-a-chip test data compression," Proc. IEEE VLSI Test Symp., pp.42–47, 2001.

[8] P.T. Gonciari, B.M. Al-Hashimi, and N. Nicolici, "Variable-length input Huffman coding for system-on-a-chip test," IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., vol.22, no.6, pp.783–796, June 2003.

[9] A. Krstic and K.-T. Cheng, Delay fault testing for VLSI circuits, Kluwer Academic Publishers, 1998.

[10] K. Namba and H. Ito, "Scan design for two-pattern test without extra latches," IEICE Trans. Inf. & Syst., vol.E88-D, no.12, pp.2777–2785, Dec. 2005.

[11] I. Polian, B. Becker, and A. Czutro, "Compression methods for path delay fault test pair sets: A comparative study," Proc. IEEE European Test Symp., p.4.2, 2004.

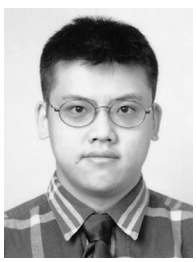
[12] A.K. Pramanick and S.M. Reddy, "On the detection of delay faults," Proc. IEEE Int'l Test Conf., pp.845–856, 1988.

[13] G. Zeng and H. Ito, "Efficient test data decompression for system-on-a-chip using an embedded FPGA core," Proc. IEEE Int'l Symp. Defect Fault Tolerance VLSI Syst., pp.503–510, 2003.

[14] C. Barnhart, V. Brunkhorst, F. Distler, O. Farnsworth, A. Ferko, B. Keller, D. Scott, B. Koenemann, and T. Onodera, "Extending OPMISR beyond 10× scan test efficiency," IEEE Des. Test Comput., vol.19, no.5, pp.65–73, Sept.–Oct. 2002.

[15] A. Jas, J. Ghosh-Dastidar, and N.A. Touba, "Scan vector compression/decompression using statistical coding," Proc. IEEE VLSI Test Symp., pp.114–120, 1999.

[16] Y. Zorian and S. Shoukourian, "Embedded-memory test and repair: Infrastructure IP for SoC yield," IEEE Des. Test Comput., vol.20, no.3, pp.58–66, May–June 2003.



Kazuteru Namba received B.E., M.E. and Ph.D. from Tokyo Institute of Technology in 1997, 1999 and 2002, respectively. He joined Chiba University in 2002. He is currently an Assistant Professor of Graduate School of Advanced Integration Science, Chiba University. His current research interests include dependable computing. He is a member of the IEEE and the IPSJ.

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 Graduate School of Advanced Integration Science. His research interests include easily testable VLSI design, defect-tolerant VLSI design, VLSI architecture, fault-tolerant computing, and dependable computing. He is a member of the IEEE and the IPSJ.