Space Compaction of Test Responses Using Orthogonal Transmission Functions

Krishnendu Chakrabarty, *Senior Member, IEEE*, and Markus Seuring

Abstract—Space compaction of test responses provides parallel access to functional outputs and reduces testing time and test data volume. We present a new space compaction approach that only uses information about the fault-free responses for a precomputed test set T . It is, therefore, especially suitable for embedded cores. It does not make any assumption about an underlying fault model, and it does not make use of any structural information about the core. Advantages of this approach include complete error propagation for all errors and optimum (provably maximum) compaction ratio. Classical linear space compactors often require structural information and can only guarantee error propagation for a specific fault model. In contrast, the proposed compactor design is based on the use of orthogonal transmission functions, which allow all errors produced by T to be propagated through the space compactor. We illustrate the proposed method by presenting case studies on compactor synthesis for several large ISCAS benchmark circuits. We show that maximum compaction with low area overhead can be achieved for circuits for which compact test sets are available.

Index Terms—Embedded cores, error propagation, nonmodeled faults, one-step compaction, test data volume, two-step compaction, transparency.

I. INTRODUCTION

EMBEDDED CORES are now being increasingly used in large system-on-a-chip (SOC) designs [2]. These pre-designed functional blocks facilitate design reuse, allow greater on-chip functionality, and lead to shorter product development cycles. However, intellectual property (IP) cores pose major test challenges; for example, the system integrator is faced with the problem of propagating the test responses of an IP core to observable chip I/Os. If an IP core feeds some user-defined logic (UDL) or other IP cores, the errors in its test response may be masked before they can be observed. While wrapper cells can be placed on functional outputs to observe test responses [1], they increase testing time due to test data serialization. This is shown in Fig. 1(a). A different approach is to use space compaction to observe test responses of an IP core that feeds UDLs or other IP cores; see Fig. 1(b). Such parallel access to a core facilitates at-speed testing, and reduces test application time and test data volume.

Space compaction refers to the process of reducing a k -bit-wide data stream to a q -bit-wide signature stream, $q \ll k$.

This may however lead to error masking or aliasing, which occurs when a faulty response maps to the faulty signature. As shown experimentally in [5], [7], error masking can be a major problem in space compaction since it reduces fault coverage. Several space compaction methods have been proposed in the literature, e.g., [3]–[17]. These include linear compactors based on parity trees [7], [4], [9], [10], nonlinear tree compactors [6], [8], [17], and more general nonlinear compactors [11].

The increased use of core-based SOC designs has led to renewed interest in output space compaction to reduce test data volume [14]–[16]. When k outputs are compacted to q signature streams, a factor of k/q reduction is obtained in output data volume. Such output compaction techniques are intended to complement data compression methods that address large test sets for SOC designs [18]–[22].

All the above methods use an underlying fault model to minimize or eliminate error masking. For example, complete error propagation for single stuck-at faults is guaranteed in [8], [9], [11]. However, these techniques are closely related to the single stuck-line fault model, therefore their effectiveness for nonmodeled faults is unknown. Since a number of defects for today's ICs cannot be modeled by classical fault models, it is important to design space compactors that propagate the maximum amount of error information, irrespective of an underlying fault model. Furthermore, the above space compaction methods are inherently unsuitable for IP cores since they require structural knowledge about the circuit under test.

It has recently been shown that error-propagating space compaction for embedded IP cores can be achieved with $q = 1$ by connecting, in addition to the outputs, all the inputs of the core under test to the compactor [13]. While this method offers a number of advantages, the additional interconnect area required to connect the inputs of the core to the compactor may be unacceptable.

We present a general approach for designing error-propagating space compactors that relies only on the knowledge of a precomputed compact test set T (and fault-free test responses) provided by the designer. These compactors propagate *all* errors that are produced by T at the output of a faulty circuit. This is especially desirable since compact test sets for a classical fault model, such as single-stuck at faults, also detect many nonmodeled faults. The proposed compactor design method makes no restrictive assumption about an underlying fault model, and no information is required about the internal structure of the circuit. It can therefore be directly used for compacting the test responses at the functional outputs of an IP core, especially if the core is designed to be "P1500-ready" [2].

For any given test set T , we present a lower bound $q_{\min}$ on the number of compactor outputs necessary to propagate all errors

Manuscript received December 15, 2002; revised May 4, 2003. This work was supported in part by the National Science Foundation under Grant Number CCR-9875324.

K. Chakrabarty is with the Department of Electrical and Computer Engineering, Duke University, Durham, NC 27708 USA (e-mail: krish@ee.duke.edu).

M. Seuring is with the AMD Saxony Manufacturing GmbH, Dresden, Germany.

Digital Object Identifier 10.1109/TIM.2003.818542

Fig. 1. Accessing the test responses of an IP core. (a) Serial test access based on wrapper cells (the scan outputs may also be wrapped); (b) parallel access through space compaction for IP cores.

produced by T . The derivation of this lower bound also provides a procedure for designing error-propagating compactors with exactly $q_{\min}$ outputs. We then show that the compaction ratio k/q can be increased further by propagating all errors by applying T twice in two time-steps. The two-step compactor is designed using the concept of orthogonal transmission functions, which was first described in [24] for propagating precomputed test sets in hierarchical test generation. Two-step compaction allows us to trade off testing time with the compaction ratio. We show that orthogonal two-step compaction can often be achieved without applying the entire test set in the second time-step; in fact, the total testing time can be reduced from $2|T|$ cycles to $(1 + \alpha)|T|$ cycles, where $0 < \alpha \leq 1$.

While the proposed method is general and can be applied to any circuit with a precomputed test set T , it yields the best results (high compaction ratio and low area overhead) if T is compact and contains a small number of patterns. Recent advances in automatic test pattern generation have led to techniques and tools that provide such compact sets for large circuits [23], [28], [29]. If a long test sequence T' is applied to a circuit, the area overhead necessary for propagating all errors produced by T' can be high. This is admittedly a limitation of the proposed method; nevertheless, it can be mitigated as outlined below.

Often, long test sequences include a smaller, essential test set T . For example, T is often generated by an LFSR with additional logic yielding a large test set T' with $T \subset T'$. In this case, an error-propagating compactor designed using T as described below can be combined with a parity tree [3] that improves the propagation of errors produced by test patterns in $T' - T$. To augment the error-propagating compactor for T , we ideally need a space compactor that propagates the maximum amount of error information irrespective of the test set and the underlying fault model. If, in the absence of specific test and fault information, we assume that all errors are equally likely, then it can be shown that a parity tree has the highest probability of propagating arbitrary errors [24]. The parity tree space compactor requires one more time-step for applying the test set T' , but in this additional time-step, faults may be detected that are not tested by T .

The proposed method for one-step compaction is also area-optimal in the following sense: it provides a (symbolic) logic specification of an error-propagating compactor for subsequent synthesis. Any other compactor design that also propagates all

errors produced by T' and uses the same synthesis tools will therefore require at least as much silicon area. However, this is not true in general for two-step compaction since the orthogonal transmission functions for the two time-steps are not unique.

The proposed design can not only be used for combinational IP cores but also to access the test responses at the functional outputs of full-scan circuits. In order to reduce testing time, most current designs employ multiple scan chains. This often implies that the number of functional outputs is greater than the length of a single scan chain. In order to access the test responses at the functional outputs without increasing further the testing time, the proposed compactor design can be used to propagate errors to a test access mechanism as shown in Fig. 1(b).

If two-step compaction is used to access the test responses at the functional outputs of a full scan circuit then loading the scan chain twice with the same vector can be avoided as follows: After serially loading the scan cells and running the functional clock for one cycle, the values at the functional output must be held stable until both compaction steps have been carried out. The two compaction steps at the functional outputs can be interleaved with the scan-in and scan-out operation, which avoids an increase in the testing time compared to one-step compaction.

The main contributions of this paper are listed below.

- We derive an upper bound on the compaction ratio that can be achieved for an IP core with any given test set T . This provides a lower bound on the number of compactor outputs q . It is always achieved since the proof of this result also provides a logic specification for the compactor.
- We prove the following striking result: space compaction with complete error propagation is not possible with linear compactors. In other words, if only linear compactors are used, then complete error propagation can only be achieved if $q = k$, i.e., the compaction ratio can be at most 1. This is in contrast to space compaction with the single stuck-at fault model, for which complete error propagation with high compaction ratio can indeed be achieved with linear compactors [4], [9].
- In order to increase the compaction ratio, we relate space compaction to transmission functions [24], and utilize orthogonal transmission functions for two-step space compaction. We also derive a lower bound on q for two-step compaction.

- We carry out case studies by identifying a number of large ISCAS benchmark circuits to which the proposed method can be easily applied. We synthesize space compactors for these circuits and demonstrate that the area overhead is low to moderate in all cases.

The organization of the paper is as follows. In Section II, we describe one-step compaction, and prove that complete error propagation cannot be achieved with linear compactors. In Section III, we review transmission functions, investigate two-step compaction, and present a lower bound on q for two-step compaction. Finally, in Section IV, we present case studies on the synthesis of space compactors for several large benchmark circuits.

II. ONE-STEP COMPACTION

In this section, we examine error-propagating space compaction when the test set T is applied to the core exactly once; we refer to this as *one-step compaction*. First, we present the following lower bound on q .

Theorem 1: Let T be a test for a core under test C . Let N be the number of distinct fault-free responses of C to T . A lower bound on q for an error-propagating space compactor for (C, T) is given by

$$q \geq \log_2(N + 1).$$

Proof: The space compactor should detect all faults that cause the test response to change from a fault-free response R to some other response R' . Since R' may also belong to the set of fault-free responses (R' may be the fault-free response for some other test pattern), the signature should be distinct for all the fault-free responses. On the other hand, all the faulty responses can be mapped to the same signature. Therefore, the compactor must produce at least $N + 1$ distinct signatures, and at least $\lceil \log_2(N + 1) \rceil$ outputs are necessary to generate these signatures.

The proof of Theorem 1 is constructive, in that it also provides an algorithm for synthesizing an error-propagating space compactor with $q = \lceil \log_2(N + 1) \rceil$ outputs. The designer can map the fault-free responses to unique symbolic values, as well as all other (faulty) responses of the core to another distinct symbolic value. A logic synthesis tool can then be used for mapping the symbolic values to binary values and for optimizing the space compactor. In this way, we can design a minimum-area space compactor with the maximum possible compaction ratio. For our experiments (described in Section VI), we used the SIS synthesis package from the University of California, Berkeley [27].

For example, consider the c6288 ISCAS 85 benchmark circuit [25]. We use a compact test set T for this circuit consisting of 12 test patterns derived from the Mintest ATPG program [23]. These patterns yield 12 distinct fault-free responses, and the resulting specification for the compactor is shown in Fig. 2.

In general, the procedure for designing a one-step compactor with complete error propagation for (C, T) can be described as follows.

- 1) Determine the number of compactor outputs q using Theorem 1.

	Response value	Signature
R_1	11001110011010001010011011	value1
R_2	000000000000000000000000	value2
R_3	111101000010001001010010101	value3
R_4	1000011000011101111010101001	value4
R_5	100000001000101111110010111001	value5
R_6	0011000101111011110011101000100	value6
R_7	0100111010000110101010001110011	value7
R_8	01110100111111000011010111010110	value8
R_9	11010110110101100110010011011110	value9
R_{10}	10001100110001011101100011000010	value10
R_{11}	10101111101011000001010000101100	value11
R_{12}	0010011010100110101100101011111	value12
	<i>All other response values</i>	value13

Fig. 2. Error-propagating space compactor specification for the c6288 benchmark circuit.

Fig. 3. Generation of orthogonal transmission functions.

- 2) Assign unique signature values to the fault-free responses and a different signature to the set of faulty responses. This can be done using encoding programs such NOVA in the SIS package. For compact tests, the set of fault-free responses is far smaller than the set of all possible faulty responses, hence the latter should be mapped to the all-zero signature for logic minimization.
- 3) Synthesize the compactor using SIS. If the fault-free responses contain don't-care values then these are utilized to simplify the compaction function and to reduce further the area overhead of the resulting compactor.

The above design procedure will, in general, result in a non-linear compactor. Linear compactors, composed of exclusive-or gates, are often desirable since they are believed to possess excellent error propagation properties and it is relatively easier in these cases to detect faults within the compactor. However, we show below that linear compactors cannot provide any compaction with complete error propagation for IP cores.

Theorem 2: If a q -output linear compactor provides complete error propagation for (C, T) , then $q = k$, where k is the number of outputs in the core under test.

Proof: Suppose the q -output compactor consists of q linear functions. From the proof of Theorem 1, we know that there must be $N + 1$ signature values, where N is the number of distinct fault-free responses produced by T . Furthermore, each fault-free response must map to a unique signature. Since each of the q linear functions is 1 for exactly 2^{k-1} input combinations, we conclude that each of these functions divides the input space of the compactor into two equal sets. Furthermore, each successive linear function subdivides these sets into equal halves. Therefore, a total of $l \leq q$ functions will divide the input space of the compactor into a total of 2^l sets, each containing 2^{k-l} input combinations. In order to achieve complete error propagation, we must have at least N sets of

Fig. 4. Two-step space compaction using orthogonal transmission functions. (a) Conceptual idea; (b) practical implementation.

a single element each, which implies that $l = k$. Hence, we cannot achieve complete error propagation with $q < k$.

III. TWO-STEP COMPACTION

Since Theorem 1 provides a tight (realizable) lower bound on q , the number of outputs in the compactor, we must consider alternative compactor designs to reduce the value of q further. One such alternative approach is based on the use of orthogonally transparent *two-step compaction*, in which error information is propagated through the compactor in two time steps, and T is applied twice to the core under test. The compactor implements a different function for each time step, such that all errors that are not propagated in the first time step are guaranteed to be propagated in the second.

We first review some basic terminology and notation on transparency, as presented in [24]. A space compactor M is said to be *transparent* if any two different values R_i and R_j that appear on its input produce different values at its output. The function realized by M partitions the set of values that can appear at its output into blocks. This partition is formally represented by a *transmission function*, denoted as $\text{Tr} = \{(\alpha_i, \beta_i)\}$, where α_i is the set of input values in the i th block, and β_i is the corresponding output value produced by the module for all of α_i . For example, for the 2-input AND gate, $\text{Tr} = \{(0, 1, 2; 0), (3; 1)\}$ has two blocks.

The amount of information that can be propagated by M clearly depends on the structure of the blocks in M 's transmission function. If each block has only one element, then M is transparent, otherwise it is only partially transparent. M can be made transparent by propagating the input data in multiple time steps by applying a sequence of values on control inputs; in fact, such k -step transparency was used in [5], [16] for space compaction. However, the k -step transparency approach in [5] relies on structural information and is therefore unsuitable for IP cores. The two-step, error-propagating compactor proposed in [16] is independent of a test set, and it reduces $2k + 1$ outputs of the core under test to k signature streams for all errors. We are interested here in using the knowledge of fault-free responses for the synthesis of transparent space compactors with much higher compaction ratio. No other structural information of the core under test is used in the proposed approach.

A space compactor can be designed to be transparent in two time steps if the transmission functions Trans_1 and Trans_2 for the two time steps are orthogonal. A pair of transmission functions $(\text{Trans}_1, \text{Trans}_2)$ is defined to be orthogonal in [24] if no two elements that appear in the same

block of Trans_1 appear in a common block in Trans_2 . For example, $\text{Trans}_1 = \{(0, 1, 2; 0), (3; 1), (4, 5; 2), (6, 7; 3)\}$ and $\text{Trans}_2 = \{(0, 3, 4, 6; 0), (1, 5, 7; 1), (2; 3)\}$ are orthogonal to each other. In fact, as shown in Fig. 3, Trans_2 can always be directly constructed from Trans_1 .

A space compactor that performs two-step compaction implements two functions corresponding to the transmission functions Trans_1 and Trans_2 . A q -bit 2-to-1 multiplexer is used to select the appropriate compaction function [see Fig. 4(a)]. In order to reduce area (both gate and interconnect) overhead, we present an implementation in which Trans_1 is trivially implemented using a set of q feed-through wires, i.e., in the first time-step, q carefully-chosen outputs of the core under test are directly connected to the compactor outputs. This is illustrated in Fig. 4(b).

A procedure for selecting primary outputs that determine Trans_1 is shown in Fig. 5. In Fig. 6, we present the pseudocode description for a procedure to generate Trans_2 from Trans_1 .

Next, we present a lower bound on q for two-step compaction. This lower bound can be achieved through a careful choice of Trans_1 .

Theorem 3: Let T be a test for a core under test C . Let N be the number of distinct fault-free responses of C to T . A lower bound on q for a two-step error-propagating space compactor for (C, T) is given by

$$q \geq \log_2(1 + \sqrt{1 + 4N}) - 1.$$

Proof: Let l be the number of elements in the largest block in Trans_1 . Since Trans_2 is orthogonal to Trans_1 , we need to ensure that each of these elements is assigned to a different block of Trans_2 . For a q -output compactor, there are a total of 2^q blocks available in Trans_2 , out of which one block must be used for the responses that are not generated by the fault-free core under test. This implies that $l \leq 2^q - 1$. Furthermore, since there are 2^q blocks in Trans_1 , the number of elements in the largest block of Trans_1 is at least $N/2^q$. This imposes the condition $l \geq 2^q/N$, which implies that $2^q/N \leq l \leq 2^q - 1$. Therefore

$$\frac{2^q}{N} \leq 2^q - 1$$

which yields $2^q(2^q - 1) \geq N$. Let $p = 2^q$. We now get the quadratic inequality

$$p^2 - p - N \geq 0.$$

```

Procedure EXTRACT_LINES( $k, q, N$ ) /* Select a set of primary outputs that generate  $Trans_1$  */
  begin
    max_set_size :=  $\lceil N/2^q \rceil$ ;
    while (no_solution)
      #lines_extracted := 0;
      start_position := 1;
      SEARCH(#lines_extracted, start_position);
      max_set_size := max_set_size + 1;
      if (max_set_size =  $2^q$ )
         $q := q + 1$ ;
        max_set_size :=  $\lceil N/2^q \rceil$ ;
      end if
    end while
  end

Procedure SEARCH(#lines_extracted, start_position)
  begin
    #lines_extracted := #lines_extracted + 1;
    for next_line = start_position to  $k$ 
      size_of_largest_set := determine_size_of_largest_set();
      /* Determine set of blocks if next_line is selected */
      if (size_of_largest_set ≤ max_set_size) then
        return(solution);
      else
        if (size_of_largest_set ≤  $2^{q-\text{\#lines\_extracted}}$  · max_set_size)
          SEARCH(#lines_extracted, next_line+1);
          if (solution)
            return(solution);
          end if
        end if
      end for
    return(no_solution);
  end

```

Fig. 5. Procedures for selecting primary outputs that determine $Trans_1$.

This can be written as

$$\left(p - \frac{1 + \sqrt{1 + 4N}}{2}\right) \left(p - \frac{1 - \sqrt{1 + 4N}}{2}\right) \geq 0$$

which implies that $p \geq ((1 + \sqrt{1 + 4N})/2)$, and therefore $q \geq \log_2(1 + \sqrt{1 + 4N}) - 1$.

A corollary to Theorem 3 is that the number of outputs for a two-step compactor can never be less than half the number of outputs required by a one-step compactor. This result is also supported by intuition since all errors produced by the test set T are now being propagated in two time-steps.

Corollary 1: Let $q_1(q_2)$ be the number of outputs in a one-step (two-step) compactor for (C, T) . Then $q_2 \geq q_1/2$.

Proof: We need to show that $2q_2 \leq q_1$. We do this, as follows:

$$\begin{aligned}
 & 2\lceil \log_2(1 + \sqrt{1 + 4N}) - 1 \rceil \\
 & \geq \lceil 2(\log_2(1 + \sqrt{1 + 4N}) - 1) \rceil \\
 & = \left\lceil 2\log_2\left(\frac{1 + \sqrt{1 + 4N}}{2}\right) \right\rceil \\
 & = \left\lceil \log_2\left(\frac{1 + \sqrt{1 + 4N}}{2}\right)^2 \right\rceil \\
 & = \left\lceil \log_2\left(\frac{1 + 2\sqrt{1 + 4N} + 1 + 4N}{4}\right) \right\rceil
 \end{aligned}$$

$$\begin{aligned}
 & \geq \left\lceil \log_2\left(\frac{1 + 2 + 1 + 4N}{4}\right) \right\rceil \\
 & = \lceil \log_2(1 + N) \rceil.
 \end{aligned}$$

Therefore, $2q_2 \geq 2\lceil \log_2(1 + \sqrt{1 + 4N}) - 1 \rceil \geq \lceil \log_2(1 + N) \rceil = q_1$. This completes the proof.

Once we obtain a lower bound on q from Theorem 3, we attempt to find q outputs of the core under test (and pass them through the compactor) such that transmission function $Trans_1$ contains at most $2^q - 1$ elements in any block. This can be obtained through an $O\left(\binom{k}{q}\right)$ exhaustive search algorithm. Since Theorem 3 yields a small value of q , the exhaustive search algorithm is a small-degree polynomial in k . If it is not possible to find q outputs such that no block in $Trans_1$ has more than $2^q - 1$ elements, then the lower bound of Theorem 3 cannot be achieved for the implementation of Fig. 4(b) and higher values of q must be considered. Note that in general, a larger proportion of errors are propagated by $Trans_2$; nevertheless, $Trans_1$ is necessary to ensure complete error propagation.

As an example, we revisit the fault-free test responses for the c6288 circuit shown in Fig. 2. Since $N = 12$ for this example, $q \geq 2$. We next note that if the two rightmost outputs of the circuit are fed through in the first time-step, then the corresponding transmission function $Trans_1$ has four blocks of three elements each, and therefore the lower bound on q can indeed

```

Procedure GET_ORTHOGONAL_FUNCTION( $q, R, Trans_1$ ) /* Determine  $Trans_2$  from  $Trans_1$  */
begin
  for trans_2_output_value=0 to  $2^q - 1$ 
    for trans_1_output_value=0 to  $2^q - 1$ 
      for each response  $r$  of  $R$ 
        if ( $Trans_1(r) = trans\_1\_output\_value$ ) then
           $Trans_2(r) := trans\_2\_output\_value$ ;
          remove  $r$  in  $R$ ;
          break; /* of for each */
        end if
      end for
      if ( $R \leq 2^q - 1 - trans\_2\_output\_value$ ) then
        /* if number of remaining responses with no  $Trans_2$  output values assigned is */
        /* lower or equal to number of unused output values of  $Trans_2$  */
        for each response  $r$  of  $R$ 
          trans_2_output_value++;
           $Trans_2(r) := trans\_2\_output\_value$ ;
        end for
        break; /* of for trans_1_output_value... */
      end if
    end for
  end for
end

```

Fig. 6. Procedure for determining $Trans_2$ from $Trans_1$.

be achieved for this example. This results in the following structure for $Trans_1$

$$Trans_1 = \{(R_2, R_6, R_{11}; 0), (R_3, R_4, R_5; 1), (R_8, R_9, R_{10}; 2), (R_1, R_7, R_{12}; 3)\}.$$

From this we generate the orthogonal transmission function $Trans_2$

$$Trans_2 = \{(R_2, R_3, R_8, R_1; 1), (R_6, R_4, R_9, R_7; 2), (R_{11}, R_5, R_{10}, R_{12}; 3), (\{R_f\}; 0)\}$$

where $\{R_f\}$ is the set of all (faulty) responses that are not included in the given set of fault-free responses. The compactor specifications for the two steps are shown in Fig. 7.

The space compactor for the second time-step corresponding to $Trans_2$ can now be easily implemented using standard logic synthesis tools. In general, a procedure for implementing a two-step completely error-propagating compactor for (C, T) is given below.

- 1) Determine the minimum number of compactor outputs q using Theorem 3.
- 2) Determine q outputs of the core under test such that for the architecture of Fig. 4(b), at most $2^q - 1$ elements in any block of $Trans_1$. This is done through the recursive procedure described in Fig. 5.
- 3) Determine the orthogonal transmission function $Trans_2$ using the $O(N)$ algorithm outlined in Fig. 3. A pseudocode procedure for generating $Trans_2$ is given in Fig. 6.
- 4) Assign unique signature values to the $2^q - 1$ blocks of $Trans_2$ that contain fault-free responses, and a different signature to the block of faulty responses. This can once again be done using encoding programs such NOVA in the SIS package. For compact tests, the set of fault-free re-

	Response value	Signature
R_1	11001110011010001010010100011011	11
R_2	000000000000000000000000000000	00
R_3	11110100001000100101001101010101	01
R_4	10000110000111011110101101011001	01
R_5	10000000010001011111100101111001	01
R_6	00110001011110111100111010000100	00
R_7	01001110100000110101010001110011	10
R_8	01110100111111000011010111010110	10
R_9	11010110110101100110010011011110	10
R_{10}	10001100110001011101100011000010	10
R_{11}	10101111101011000001010000101100	00
R_{12}	00100110101001101011001010111111	11

(a)

	Response value	Signature
R_1	11001110011010001010010100011011	01
R_2	000000000000000000000000000000	01
R_3	11110100001000100101001101010101	01
R_4	10000110000111011110101101011001	10
R_5	10000000010001011111100101111001	11
R_6	00110001011110111100111010000100	10
R_7	01001110100000110101010001110011	10
R_8	01110100111111000011010111010110	01
R_9	11010110110101100110010011011110	10
R_{10}	10001100110001011101100011000010	10
R_{11}	10101111101011000001010000101100	10
R_{12}	00100110101001101011001010111111	10
	$\{R_f\}$, i.e. all other response values	00

(b)

Fig. 7. Specification for the two-step compactor for c6288. (a) $Trans_1$ (responses in R_f are mapped to signatures implicitly); (b) $Trans_2$ (responses in R_f are explicitly mapped to the all-zero pattern).

sponses is far smaller than the set of all possible faulty responses, hence the latter should be mapped to the all-zero signature for logic minimization.

TABLE I
RESULTS OF ONE-STEP COMPACTOR SYNTHESIS FOR PROPAGATING ALL ERRORS PRODUCED BY COMPACT TEST SETS

Circuit	Test set	No. of test patterns	No. of fault-free responses	No. of functional outputs	No. of compactor outputs q	Circuit literal count L_1	Compactor literal count L_2	Percentage overhead $(L_2/L_1) \times 100$
c499	[28]	52	3	32	2	616	64	10.39
c880	[23]	16	16	26	5	703	277	39.40
c880	[28]	17	17	26	5	703	214	30.44
c1355	[28]	84	14	32	4	1032	197	19.09
c3540	[23]	84	84	22	7	2934	864	29.45
c6288	[23]	12	12	32	4	4800	266	5.54
c6288	[28]	12	12	32	4	4800	201	4.19
s35932	[23]	12	12	320	40	35181	2414	6.86
s35932	[23]	12	12	320	32	35181	2309	6.56
s35932 ¹	[23]	12	12	320	4	35181	2542	7.23
s35932 ²	[23]	12	12	320	4	35181	2603	7.40
s38417	[23]	68	68	106	28	38790	3897	10.05
s38417	[23]	68	68	106	7	38790	4652	11.99

¹Two-stage design of Figure 9(a), ²Two-stage design of Figure 9(b)

5) Synthesize the compactor for Trans_2 using SIS. If the fault-free responses contain don't-care values then these are utilized to simplify the compaction function and to reduce further the area overhead of the resulting compactor.

Even though two-step compaction based on orthogonal transmission functions requires two time-steps, the testing time can be reduced from $2|T|$ cycles to $(1 + \alpha)|T|$ cycles, where $0 < \alpha \leq 1$ depends on q and the number of fault-free responses N . In order to achieve this reduction in testing time, we apply T and compact first the responses using Trans_2 —this allows us to propagate errors that change a fault-free response to an output which is not in the set of fault-free responses. In addition, depending on the structure of Trans_2 , a number of other errors are also propagated in this time-step. The second time-step compacting the test responses using Trans_1 is only performed for a subset of $|T|$. The following lemma characterizes which patterns of T are unnecessary for error propagation in this time-step.

Lemma 1: It is not necessary to compact the response r corresponding to a test pattern $t \in T$ in the second time-step (using Trans_1) if and only if the response r is the only element in its block in Trans_2 .

Proof: To proof sufficiency, suppose r is the only element in its block in Trans_2 . Since the complete test set T is applied to the core under test in the first time-step corresponding to Trans_2 , any error that changes r to some other response is propagated. Therefore, it is not necessary to propagate such errors in the next time-step using Trans_1 . The proof for necessity is similar.

We utilize Lemma 1 in the procedures described in Figs. 5 and 6 in order to increase the likelihood of having blocks in Trans_2 with single elements. We next use Lemma 1 to derive an upper bound on the number of patterns m that can be dropped in the second time-step when Trans_1 is used.

Theorem 4: Let N be the number of fault-free responses and let q be the number of outputs in the error-propagating compactor. An upper bound on the number of patterns m that can be dropped in the second time-step is given by

$$m \leq 2^q - \frac{N}{2^q - 1}.$$

Proof: If m patterns can be dropped when Trans_1 is used, then from Lemma 1, there can be at most m blocks in Trans_2 that contain one element. Furthermore, since two elements in the same block of T_2 must belong to different blocks of T_1 , each block of T_2 can have at most 2^q elements. Therefore, the number of blocks in Trans_2 containing more than one element is at most $(2^q - 1 - m)2^q$. This implies that

$$N - m \leq (2^q - 1 - m)2^q$$

which yields $m \leq 2^q - (N/2^q - 1)$.

For example, for the c6288 test set, since $N = 12$ and $q = 2$, $m = 0$ and no test pattern can be dropped. This shows that no reduction in testing time is possible if $2^q - (N/2^q - 1) < 1$. It also suggests that for a given test set, greater the compaction ratio, the smaller is the number of test patterns that can be eliminated. Therefore we can trade off testing time with the compaction ratio by choosing a value for q that is slightly higher than the lower bound. In fact, we can characterize the rate at which q must grow such that the fraction reduction in testing time, $m/|T|$, remains constant with increasing N .

Theorem 5: Let N be the number of fault-free responses and let q be the number of outputs in the completely error-propagating compactor. Let m be the number of patterns that are unnecessary for error propagation with Trans_1 . If the fraction reduction in testing time, $m/|T|$, is to remain constant with increasing N , then $q = O(\log_2 N)$.

Proof: We assume that $N = |T|$, since for most compact tests, the fault-free responses are all distinct. From Theorem 4, we know that $r = m/N \leq (2^q/N) - (1/2^q - 1)$. If we let $p = 2^q$, we get the following quadratic inequality

$$p^2 - p(1 + Nr) - N(1 - r) \leq 0$$

which yields

$$p \leq \frac{(1 + Nr) + \sqrt{(1 + Nr)^2 + 4N(1 - r)}}{2}$$

Treating r as a constant, we get $p = O(N)$ and $q = O(\log_2 N)$.

Fig. 8. Output partitioning for s35932. (a) Ten groups of 32 outputs each; (b) eight groups of 40 outputs each.

Fig. 9. One-step, two-stage maximal compactor designs for s35932.

IV. CASE STUDIES

In this section, we present experimental results on the synthesis of space compactors for several large ISCAS benchmark circuits [25], [26]. We considered the combinational benchmarks and the functional outputs of the full-scan versions of the sequential benchmarks. We used test sets from the Mintest ATPG program [23] as well as compact test sets derived using high-level functional models [28]. The test sets achieve 100% fault coverage for nonredundant stuck-at faults. For our experiments, we used the SIS package running on a Sun Ultra Sparc workstation with 256 MB of memory, and estimated the circuit area by using the literal counts provided by SIS. For the ISCAS 89 circuits, we included the area estimates of the scan flip-flops in our calculations. The experimental results for one-step compaction for propagating all errors are presented in Table I.

For the smaller benchmarks, the hardware overhead of the proposed method is high since the compactor area is comparable to the area of the circuits. Typical IP cores are considerably larger than these circuits. Nevertheless, we were able to design

a one-step compactor for c499 with only 10.6% overhead using the test set from [28]. There is a significant relationship between the area overhead of the compactor and the test set T . For example, if the test set from [23] is used for c1355, the overhead is over 100%; however, it reduces to 19% when the test set from [28] is used. The area overhead of the compactor for c6288 is very low for test sets from both [23] and [28]. This benchmark is a 16-bit multiplier circuit and is more representative of typical combinational cores.

Most cores however are sequential circuits with internal scan. Therefore, we also designed the space compactor for the functional outputs of two large ISCAS 89 circuits. We did not attempt to compact the scan outputs of these circuits. Typically, the number of functional outputs are far greater than the number of scan outputs. We assumed that during test application, the response patterns at the functional outputs change only when a new pattern is applied from the functional inputs and from the internal scan chain(s).

For the two large ISCAS 89 circuits that we considered, we partitioned their functional outputs such that SIS could synthe-

TABLE II
RESULTS OF TWO-STEP COMPACTOR SYNTHESIS FOR PROPAGATING ALL ERRORS PRODUCED BY COMPACT TEST SETS

Circuit	Test set	No. of test patterns	No. of fault-free responses	No. of functional outputs	No. of compactor outputs q	Circuit literal count L_1	Compactor literal count L_2	Percentage overhead $(L_2/L_1) \times 100$
c499	[28]	52	3	32	2	616	77	12.50
c880	[23]	16	16	26	3	703	259	36.84
c880	[28]	17	17	26	3	703	204	29.02
c1355	[28]	84	14	32	3	1032	200	19.38
c3540	[23]	84	84	22	4	2934	792	26.99
c6288	[23]	12	12	32	2	4800	254	5.29
c6288	[28]	12	12	32	2	4800	201	4.19
s35932	[23]	12	12	320	20	35181	2385	6.78
s35932	[23]	12	12	320	16	35181	2290	6.51
s35932 ¹	[23]	12	12	320	2	35181	2623	7.46
s35932 ²	[23]	12	12	320	2	35181	2478	7.04
s38417	[23]	68	68	106	16	38790	3666	9.45
s38417	[23]	68	68	106	4	38790	4553	11.74

¹Two-stage design, similar to Figure 9(a), ²Two-stage design, similar Figure 9(b)

size space compactors for them in reasonable time (at most a couple of hours of user time). The complexity of the synthesis procedure depends on the number of functional outputs k (the number of compactor inputs), and we found in our experiments that optimized compactors can be generated more easily if k is at most 32. This presents a tradeoff between compaction ratio and the ability to automatically synthesize space compactors. For example, for s35932, we first partitioned the 320 functional outputs into ten groups of 32 outputs each. We then synthesized a space compactor for each of these partitions. As shown in Fig. 8(a), this resulted in a total of 40 compactor outputs, yielding a compaction ratio of 8. The total overhead for this design was only 6.86%. Alternatively, as shown in Fig. 8(b), we can also partition the 320 outputs into 8 groups of 40 outputs each. The overhead for the latter design is only 6.56%.

In order to increase the compaction ratio, we used a second stage of compaction, in which the 40 outputs from the first stage are further compacted to 4 outputs (see Fig. 9). The overhead for these alternative designs, which yield the maximum possible compaction ratio, are only 7.23% and 7.40%, respectively. Even better results can be expected with a systematic partitioning strategy.

We partitioned the 106 functional outputs of s38417 in a similar way. We formed three groups of 27 outputs each, and we formed a fourth group containing 25 outputs. This yields a total of 28 compacted outputs, with a total area overhead of 10.05%. If we reduce the number of compactor outputs to seven (minimum possible), then the area overhead increases only slightly to 11.99%.

We next present experimental results on two-step compaction using orthogonal transmission functions. As noted before, two-step compaction yields a smaller value for q and it also often reduces the area overhead. While these benefits are accompanied by an increase in the testing time, for compact test sets, this increase does not add substantially to the test cost. Furthermore, we can use Lemma 1 to reduce the number of patterns that must be applied in the second time-step.

Table II shows the area overhead and the number of compactor outputs q for two-step compaction. The lower bound on q given by Theorem 3 is achieved for every case that we

TABLE III
NUMBER OF PATTERNS m THAT CAN BE DISCARDED DURING THE SECOND TIME-STEP. (UPPER BOUNDS ARE OBTAINED USING THEOREM 4)

Circuit	Test set	No. of test patterns	Upper bound on m	m	Percentage saving
c499	[28]	52	4	3	5.77
c880	[23]	16	5	5	31.25
c880	[28]	17	5	5	29.41
c1355	[28]	84	6	5	5.95
c3540	[23]	84	10	9	10.71
c6288	[23]	12	0	0	0
c6288	[28]	12	0	0	0
s35932 ¹	[23]	12	0	0	0
s35932 ²	[23]	12	0	0	0
s38417	[23]	68	11	11	16.17

¹Two-stage design of Figure 9(a), ²Two-stage design of Figure 9(b)

consider. The area overhead figures in Table II include the literal counts for the multiplexers to switch between the two transmission functions. In a number of cases, the area overhead for two-step compaction (including the multiplexer needed to switch between Trans_1 and Trans_2) is less than that for one-step compaction.

In Table III, we present experimental results on the number of test patterns that can be discarded in the second time-step. We observe that a significant fraction of patterns can be dropped in many cases, especially if q is not as low as the lower bound predicted by Theorem 3.

V. CONCLUSION

We have presented a new space compaction approach that only uses information about the fault-free responses for a precomputed test set T . It is particularly suitable for reducing output test data volume for embedded cores. The proposed approach does not make any assumption about an underlying error model; moreover, it does not make use of any structural information about the core. This approach has several advantages: complete propagation of all errors, optimum (provably maximum) compaction ratio, parallel access to the functional outputs of the core, and reduced testing time and test data volume. The compactor design is based on the use of orthogonal transmission functions, which allow all errors produced

by T to be propagated through the space compactor. We have demonstrated the feasibility of this approach by presenting case studies on compactor synthesis for several large ISCAS benchmark circuits. The proposed method is best suited to circuits for which compact test sets are available. This limitation can be mitigated by designing a completely error-propagating compactor for a high-quality test set T embedded within a larger test set T' , and using a parity tree to handle responses for patterns that are in $T' - T$.

REFERENCES

- [1] IEEE P1500 General Working Group Website [Online]. Available: <http://grouper.ieee.org/groups/1500>
- [2] Y. Zorian, E. J. Marinissen, and S. Dey, "Testing embedded-core based system chips," in *Proc. Int. Test Conf.*, 1998, pp. 130–143.
- [3] S. M. Reddy, K. K. Saluja, and M. G. Karpovsky, "A data compression technique for built-in self-test," *IEEE Trans. Comput.*, vol. 37, pp. 1151–1156, Sept. 1988.
- [4] J. Savir, "Shrinking wide compressors," *IEEE Trans. Comput.-Aided Design*, vol. 14, pp. 1379–1387, Nov. 1995.
- [5] K. Chakrabarty and J. P. Hayes, "Test response compaction using multiplexed parity trees," *IEEE Trans. Comput.-Aided Design*, vol. 15, pp. 1399–1408, Nov. 1996.
- [6] A. Ivanov, B. Tsuji, and Y. Zorian, "Programmable BIST space compactors," *IEEE Trans. Comput.*, vol. 45, pp. 1393–1404, Dec. 1996.
- [7] M. Seuring, M. Gössel, and E. Sogomonyan, "A structural approach for space compaction for concurrent checking and BIST," in *Proc. VLSI Test Symp.*, 1998, pp. 354–361.
- [8] B. Pouya and N. A. Toubia, "Synthesis of zero-aliasing elementary-tree space compactors," in *Proc. VLSI Test Symp.*, 1998, pp. 70–77.
- [9] K. Chakrabarty, "Zero-aliasing space compaction using linear compactors with bounded overhead," *IEEE Trans. Comput.-Aided Design*, vol. 17, pp. 452–457, May 1998.
- [10] K. Chakrabarty and J. P. Hayes, "Zero-aliasing space compaction using multiple parity signatures," *IEEE Trans. VLSI Syst.*, vol. 6, pp. 309–313, June 1998.
- [11] K. Chakrabarty, B. T. Murray, and J. P. Hayes, "Optimal zero-aliasing space compaction of test responses," *IEEE Trans. Comput.*, vol. 17, pp. 1171–1187, Nov. 1998.
- [12] M. Gössel, E. S. Sogomonyan, and A. Morosov, "A new totally error propagating compactor for arbitrary cores with digital interfaces," in *Proc. VLSI Test Symp.*, 1999, pp. 49–56.
- [13] B. B. Bhattacharya, A. Dmitriev, and M. Gössel, "Zero-aliasing space compaction using a single period output and its application to testing of embedding cores," in *Proc. Int. Conf. VLSI Design*, 2000, pp. 382–387.
- [14] O. Sinanoglu and A. Orailoglu, "Space and time compaction schemes for embedded cores," in *Proc. Int. Test Conf.*, 2001, pp. 521–529.
- [15] —, "Compaction schemes with minimum test application time," presented at the *Proc. Asian Test Symp.*, 2001.
- [16] A. Morosov, K. Chakrabarty, B. B. Bhattacharya, and M. Goessel, "Design of parameterizable error-propagating space compactors for response observation," *Proc. IEEE VLSI Test Symp.*, pp. 48–53, 2001.
- [17] S. R. Das, T. F. Barakat, E. M. Petriu, M. H. Assaf, and K. Chakrabarty, "Space compression revisited," *IEEE Trans. Instrumen. Meas.*, vol. 49, pp. 690–705, June 2000.
- [18] A. Chandra and K. Chakrabarty, "System-on-a-chip test data compression and decompression architectures based on Golomb codes," *IEEE Trans. Comput.-Aided Design of Integrated Circuits and Systems*, vol. 20, pp. 355–368, Mar. 2001.
- [19] —, "Test resource partitioning for system-on-a-chip based on test data compression and on-chip decompression," *IEEE Design Test Comput.*, vol. 18, pp. 80–91, Sept./Oct. 2001.
- [20] A. Jas and N. A. Toubia, "Test vector decompression via cyclical scan chains and its application to testing core-based designs," *Proc. IEEE Int. Test Conf.*, pp. 458–464, 1998.
- [21] P. T. Gonciari, B. Al-Hashimi, and N. Nicolici, "Improving compression ratio, area overhead, and test application time for system-on-a-chip test data compression/decompression," presented at the *Proc. Design, Automation and Test in Europe (DATE) Conf.*, 2002, to appear.
- [22] R. Dorsch and H.-J. Wunderlich, "Reusing scan chains for test pattern decompression," in *Proc. European Test Workshop*, 2001, pp. 124–132.
- [23] I. Hamzaoglu and J. H. Patel, "Test set compaction algorithms for combinational circuits," in *Proc. Int. Conf. Computer Aided Design*, 1998, pp. 283–289.
- [24] B. T. Murray, "Hierarchical Testing with Precomputed Tests," Dept. of Electrical Engineering and Computer Science, The Univ. of Michigan, Tech. Rep. CSE-TR-245-95, 1995.
- [25] F. Brglez and H. Fujiwara, "A neutral netlist of ten combinational benchmark circuits and a target simulator in Fortran," in *Proc. 1985 Int. Symp. Circuits and Systems*, 1985, pp. 695–698.
- [26] F. Brglez, D. Bryan, and K. Kozminski, "Combinational profiles of sequential benchmark circuits," in *Proc. Int. Symp. Circuits and Systems*, 1989, pp. 1929–1934.
- [27] E. M. Sentovich *et al.*, "SIS: A System for Sequential Circuit Synthesis," Electronic Research Laboratory, University of California, Berkeley, CA, Technical report UCB/ERL M92/41, 1992.
- [28] M. C. Hansen and J. P. Hayes, "High-level test generation using physically-induced faults," in *Proc. VLSI Test Symp.*, 1995, pp. 20–28.
- [29] S. Kajihara, I. Pomeranz, K. Kinoshita, and S. M. Reddy, "On compacting test sets by addition and removal of vectors," in *Proc. VLSI Test Symposium*, 1994, pp. 202–207.

Krishnendu Chakrabarty (S'92–M'96–SM'00) received the B.Tech. degree from the Indian Institute of Technology, Kharagpur, in 1990, and the M.S.E. and Ph.D. degrees from the University of Michigan, Ann Arbor, in 1992 and 1995, respectively, all in computer science and engineering.

He is currently Associate Professor of electrical and computer engineering at Duke University, Durham, NC. From 2000 to 2002, he was a Mercator Visiting Professor at University of Potsdam, Germany. His current research projects (supported by NSF, DARPA, ONR, Army Research Office, and industrial sponsors) are focused on system-on-a-chip test, embedded real-time systems, distributed sensor networks, and modeling, simulation, and optimization of microelectrofluidic systems. He is a coauthor of two books: *Microelectrofluidic Systems: Modeling and Simulation* (Boca Raton, FL: CRC, 2002) and *Test Resource Partitioning for System-on-a-Chip* (Norwell, MA: Kluwer, 2002), and the Editor of *SOC (System-on-a-Chip) Testing for Plug and Play Test Automation* (Norwell, MA: Kluwer, 2002). He is an Editor of the *Journal of Electronic Testing: Theory and Applications (JETTA)*. He was the Guest Editor of a special issue of *JETTA* on system-on-a-chip testing, published in August 2002. He was also a Guest Editor in 2001 of a special issue of *Journal of the Franklin Institute* on distributed sensor networks. He has published over 130 papers in archival journals and refereed conference proceedings, and he holds a U.S. patent in built-in self-test.

Dr. Chakrabarty is a member of ACM and ACM SIGDA, and a member of Sigma Xi. He serves as Vice Chair of Technical Activities in IEEE's Test Technology Technical Council, and is a member of the program committees of several IEEE/ACM conferences and workshops. He is an Associate Editor of the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS II: ANALOG AND DIGITAL SIGNAL PROCESSING. He is a recipient of the National Science Foundation Early Faculty (CAREER) Award and the Office of Naval Research Young Investigator Award. He is a recipient of a best paper award at the 2001 Design, Automation and Test in Europe (DATE) Conference. He is also a recipient of the Humboldt Research Fellowship from the Alexander von Humboldt Foundation, Germany.

Markus Seuring received the diploma degree in computer science from the Johann Wolfgang von Goethe University, Frankfurt, Germany, in 1996 and the Ph.D. degree from the University of Potsdam, Potsdam, Germany, in 2001.

Since the summer of 2001, he has been a Product and Test Engineer at the Dresden Design Center of AMD Saxony. His research interests include design for testability, test data compaction, test pattern generation, fault modeling, and fault simulation.