Published in IET Computers & Digital Techniques
Received on 15th May 2007
Revised on 17th December 2007
doi: 10.1049/iet-cdt:20070078

Selected Papers from NORCHIP '06

ISSN 1751-8601

Architecture for integrated test data compression and abort-on-fail testing in a multi-site environment

E. Larsson

Embedded Systems Laboratory, Department of Computer and Information Science, Linköpings Universitet, Sweden
E-mail: erila@ida.liu.se

Abstract: The semiconductor technology development makes it possible to fabricate increasingly advanced integrated circuits (ICs). However, because of imperfections at manufacturing, each individual IC must be tested. A major problem at IC manufacturing test is the increasing test data volume as it leads to high automatic test equipment (ATE) memory requirement, long test application time and low throughput. In contrast with existing approaches, which address either test data compression for ATE memory reduction or abort-on-fail testing for test time minimisation, an architecture that supports both test data compression and abort-on-fail testing at clock-cycle granularity is proposed, and hence both ATE memory reduction and test application time minimisation are addressed. Further, the proposed architecture efficiently tackles low throughput as the architecture allows multi-site testing at a constant ATE memory requirement, which is independent of the number of tested ICs. Advantages of the architecture, compared with test compression architecutre, are that diagnostic capabilities are not reduced and there is no need for special handling of unknowns (X) in the produced test responses (PR). Experiments on ISCAS benchmark circuits and an industrial circuit have been performed.

1 Introduction

Semiconductor technology development makes it possible to fabricate integrated circuits (ICs) where transistor count, number of transistors per area unit, number of wiring layers, die size, clock rates and power consumption increase whereas feature size and voltage decrease.

IC fabrication is, however, far from perfect and therefore every manufactured IC is tested. The regular approach to test is as follows. The test stimuli (TS), stored in the automatic test equipment (ATE), are applied to the IC, the device-under-test (DUT), and the produced test responses (PR) are by the ATE collected and compared with the expected test responses (ER). At the diagnostic phase when the yield is low, the produced test responses are, in the case of defects, a mismatch between produced test responses and expected test responses, analysed in detail in order to find the root cause of the defect. At volume production when the yield is high, the produced test responses are only compared with the ER in a pass/fail manner. That is, fault-free ICs are shipped whereas faulty ICs are discharged.

As ICs are becoming highly complex and advanced, increasing test data volumes are needed. The high test data volumes lead to long test application times and high ATE memory requirements. Low throughput is a direct consequence of high test data volumes. Approaches such as multi-site testing, test scheduling based on abort-on-fail testing and test data compression have been proposed to address problems related to high test data volumes.

There are, however, disadvantages with multi-site testing, test scheduling based on abort-on-fail testing

and test data compression. The disadvantage with multi-site testing, where several devices are tested in parallel in order to increase throughput, is the increased need of ATE memory. The disadvantage with abort-on-fail testing, where the testing is terminated as soon as a fault is detected in order to spend less time on faulty ICs, is that it does not address' the ATE memory requirements.

Test data compression addresses the ATE memory requirement. In test data compression, the test architecture is modified. The compressed test stimuli (CTS) are stored in the ATE, and at test application, the compressed stimuli are applied to an added decompression unit that expands the CTS and applies it to the DUT. The produced test responses are compacted by added on-chip compactors such as multiple-input signature registers (MISRs). At the end of the testing, the MISR signature is shifted out and compared with the expected signature.

The advantage with test data compression is that it is possible to reach a high compression ratio and therefore the need of ATE memory is reduced. Disadvantages with current test data compression schemes are that:

1. The added logic for TS decompression and test response compaction can be test dependant; hence, if the test data for some reason have to be changed, the logic in the IC has to be changed. It is better to ensure that the logic added for test data compression is test independent.

2. Diagnostic capabilities are reduced as there is an information loss in test response compaction. The information loss because of a faulty signature makes it difficult to determine where in the IC the fault is. It is obviously possible to load the uncompressed test data volume in the ATE and bypass the decompression unit and the test response compaction. However, such a scheme is time consuming as the full test data volume may not fit the ATE in a single load. Instead, the test data has, in order to fit the memory, be partitioned in several parts and applied in a sequence.

3. At volume production where pass/fail testing is applied, it is not possible to terminate as soon as a defect is detected. If MISRs are used for test response compaction, the test result is only known at the end of the testing when the produced MISR signature is compared with the expected MISR signature. Hence, the usage of time-saving abort-on-fail testing is limited.

4. When the unspecified bits (do not care bits) in the TS are defined, the expected test responses are defined through simulation. Unfortunately, simulation cannot always determine all bits in the test responses to 0 or 1. In these cases the response bits are simply unknowns (X). A fault-free design may produce a 0 or 1 at such an X-position where both are correct. The problem is that unknowns (X) corrupt the signature in the MISR. Several X-masking approaches have been proposed; however, they often become test dependent and cannot guarantee the handling and masking of any number of X at any time.

In this paper, an architecture is proposed that enables the usage of both test data compression and abort-on-fail testing [1–3]. The basic idea is to compress both TS and expected test responses, and store CTS and compressed ER in the ATE, and perform the test evaluation on-chip.

The advantages with the architecture are as follows. Time-saving abort-on-fail testing can be used. The test data volume can be compressed. The diagnostic capabilities are good as compaction of the produced test responses is not made use of. There is no longer any problem with unkowns (X) in the test responses. Any number of unknowns can be handled by the architecture. The ATE memory requirement is constant regardless of the degree of multi-site testing (number of DUTs tested concurrently). Further, it is straight forward to modify (increase/decrease) the tests as the added evaluation logic (EL) is test independent.

To verify the architecture, the facsimile [4] the fixed-length indices [5] compression algorithms have been implemented and ISCAS designs and an industrial design have been experimented upon.

The rest of the paper is organised as follows. Background and prior work is detailed in Section 2 and the proposed architecture is described in Section 3. The experimental results are in Section 4 and the paper is concluded with Section 5.

2 Background and related work

Fig. 1 shows a typical setup for testing ICs. Prior to test application, the test data volume, that is, the TS and the ER, is loaded in the memory of the ATE. At test application, TS are loaded into the scan-chains of the

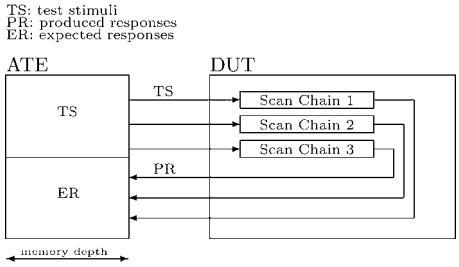

Figure 1 *DUT at test application*

doi: 10.1049/iet-cdt:20070078

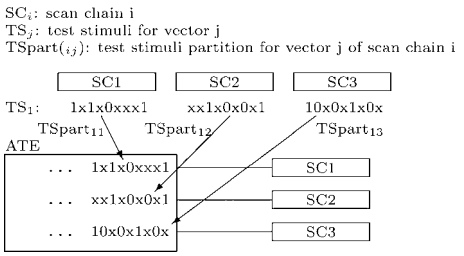

Figure 2 *ATE arrangement for TS$_1$*

DUT and produced test responses are after the capture cycle shifted out of the scan-chains and the ATE compares produced test responses with ER. The TS have to be arranged in the ATE to correspond to the scan-chains in the DUT. Fig. 2 shows the arrangment of TS$_1$ (the first test stimulus) in the ATE for the DUT in Fig. 1. The scan-chains and their length along with test data are given prior to test application, see Fig. 3.

The advantage with the architecture in Fig. 1 is that the produced test responses are available without any information loss. This is important for diagnosis as the produced test responses can be analysed in order to find the root cause of eventual defects. For volume production, where pass/fail testing is used, information loss is acceptable. However, it is important that pass/fail information is given at clock-cycle granularity. Hence, test time can be saved by not wasting test time on applying all test vectors for faulty ICs.

As ICs are becoming increasingly complex, the need of test data increases. It leads to higher costs because of long test application times high ATE memory requirement and low throughput.

In single-site testing as in Fig. 1, a single DUT is tested at a time. To increase throughput, several DUTs can be tested concurrently. However, in order to apply multi-site testing, the ATE memory requirement increases. If TS and ER are duplicated for each n degree of multi-site test, the memory

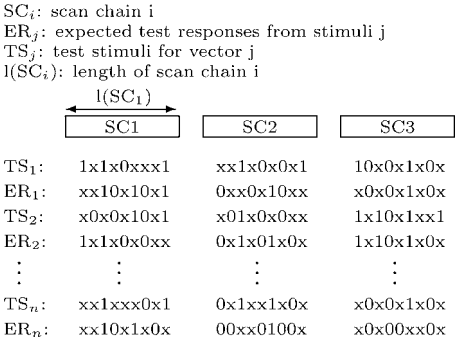

Figure 3 *Scan chains and their test data*

requirement will be $n \times (\mathrm{TS} + \mathrm{ER})$. If TS is broadcasted to n devices, the ATE memory requirement is $\mathrm{TS} + n \times \mathrm{ER}$. The memory requirement increases linearly with the degree of multi-site test (n).

As ICs are becoming highly complex and advanced, core-based design is increasingly used in order to be able to design and fabricate ICs in a timely manner [6]. Modular designs can be tested in a modular fashion [7, 8]. The advantage by making each core a testable unit is the possibility to order the testing of the cores such that the test application time is minimised. Marinissen *et al.* discuss the problem of isolating cores during testing [8] and Iyengar *et al.* showed that the test time for each wrapped core decreases in a nonlinear way when the number of wrapper-chains increases [9, 10]. A number of optimisation techniques have been proposed to order the testing of the cores such that the test application time is minimised [8–14].

To further lower the test application time, test scheduling based on abort-on-fail, where the testing is terminated as soon as a defect is detected, can be applied [15–17]. Larsson *et al.* defined for System-on-Chip (SoC) testing a scheme to take defect probabilities into account during test scheduling [15, 17]. The idea is to spend less time on faulty circuits. Ingelsson *et al.* showed that by using abort-on-fail at clock-cycle granularity the savings in test time becomes significant [16].

The main objective for test scheduling approaches is to lower the test application time. It will also lower the ATE memory requirement however not as significant as test data compression schemes.

In test data compression, the architecture, shown in Fig. 1, is modified. At the inputs, decompressor units are added. Decompressor units can be hardware or software running on a processor. The decompressor unit takes CTS as inputs and expands it to uncompressed TS. At the outputs of the DUT, compactors are added. A compactor takes produced tests responses as inputs and produces compacted test responses.

Test data compression schemes take advantage of the high number of do not care bits and fill them such that high test data compression is reached [18–21]. The common approach to test data compression is to fill the unspecified bits (x) in the TS such that a high compression ratio is reached. Once the unspecified bits in the TS are defined, the unspecified bits in ER are determined through simulation and then the on-chip response compactor, such as an MISR, can be designed for the defined ER.

doi: 10.1049/iet-cdt:20070078

Several approaches have been proposed for test data volume compression. Chandra and Chakrabarty propose the use of frequency-directed run-length (FDR) code [18] and Gonciari and Al-Hashimi [19] propose a Huffman-algorithm. Balakrishnan and Touba [20] use matrix operations to compress the test data, and Jas and Touba [21] present an approach where an embedded processor is used for decompression.

These compression approaches rely on on-chip test response compaction. In many cases MISRs are used for test response compaction. There are, however, disadvantages with MISRs. The usage of time-saving abort-on-fail testing is limited as the test result is known only at the end of the testing when the signature is produced. Diagnostic capabilities are reduced as it is difficult to determine where the fault is in the DUT based on a faulty signature. And when the unspecified bits in the TS are defined, the expected test responses are defined after simulation. However, simulation cannot always determine all bits in the test responses to 0 or 1. In these cases the responses are simply unknowns (X). This will corrupt the signature in the MISR. Several masking approaches have been proposed. Wu and Ivanov propose a scheme with multiple signatures for MISR [22] and Mitra *et al.* [23] propose a scheme to address X-bit handling. The main drawbacks are that X-handling techniques often become test dependent and, further, they cannot guarantee masking all Xs.

The idea to send TS and expected test responses to the DUT is not new. Nahvi and Ivanov[24] proposed for example, a scheme where the basic idea is to send both TS and expected test responses to the SoC, and the produced test responses are compared with the expected test responses on-chip and not at the ATE. However, there is no test data compression in the approach and abort-on-fail testing is not discussed.

Poehl *et al.* recently proposed an architecture where CTS and compacted expected responses are sent to the DUT [25]. The idea is to compare the compacted produced test responses and the compacted expected test responses onchip. As in our approach, test evaluation is moved from the ATE. In the scheme by Poehl *et al.*, unspecified bits in the TS are filled such that high test data compression is reached, and once the bits in the TS are defined, simulation is used to define the expected test responses and then the expected test responses can be compacted and stored in the ATE. At test application, the compacted expected test responses are onchip compared with the compacted expected test responses.

A major difference between the work by Poehl *et al.* and this approach is that here the expected test responses are not compacted. Instead, the expected test responses are compressed and stores do not in the ATE. The advantage is that simulation is not required to figure out expected test responses once the do not care bits in the TS are defined. Further, our scheme can handle any number of unknowns (Xs) in the produced test responses.

Ollivierre *et al.* [26] propose an architecture that explore sending TS and ER to the system. A masking register is used to unmask the responses. The focus is to find a minimal number of re-loads of the masking register. In current proposal, a dedicated mask is used for each response.

Based on the discussion above, the conclusion is that there is a need for a test architecture that allows test data compression and abort-on-fail testing, has high diagnostic resolution and does not have problem with X-handling.

3 Proposed test architecture

The proposed architecture allows, in contrast with prior architectures, integrated test data compression and abort-on-fail testing. The architecture handles any number of unknowns (Xs), does not limit diagnosis and has a constant ATE memory requirement that is independent of the degree of multi-site testing.

The architecture, which is outlined in Fig. 4, will be detailed in this section. The basic idea is to compress TS, ER and an introduced mask (M) into CTS, compressed expected responses (CER) and compressed mask (CM). CTS, CER and CM, all initially stored in the ATE and at test application fed to the decompressor (in Fig. 4 the decompressor is a CPU). The on-chip evaluation logic (EL) makes test evaluation by comparing produced test responses with expected test responses.

3.1 Test preparations

The preparation prior to test application is outlined in Fig. 5 using the example in Fig. 3. First the test data, TS and ER, are arranged for the ATE such that the scan-chains match (Fig. 5*a*). The unspecified bits in the TS are filled such that high compression is reached

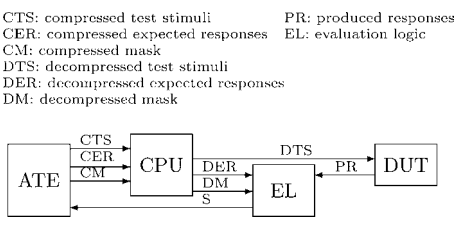

Figure 4 *Illustration of the scheme at test application where the DUT is as in Fig. 1*

doi: 10.1049/iet-cdt:20070078

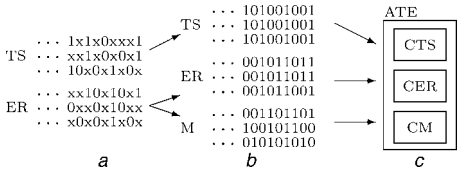

Figure 5 *Test data handling*

a Arranging test data for ATE
b Unspecified bit filling, mask definition
c Compressed test data in ATE

(Fig. 5*b*). The way TS is handled is similar to previous compression schemes.

Different from previous approaches is our handling of ER. Instead of compacting the PR on-chip, ER is compressed and stored off-chip in the ATE. To compress ER highly, arbitrary filling of the do not care bits in the ER is a necessity. To achieve that a mask (M) is introduced, see Fig. 5*b*.

Now M is explained. For each bit in ER, there is a corresponding bit in M (see Fig. 5*b*). Initially all bits in M are set to 0. For any specified bit (0 or 1) in ER, the corresponding bit in M is set to 1. The result is that a bit in M that is set to 1 indicates that the corresponding bit in ER is a care bit whereas if a bit in M is set to 0, the corresponding bit in ER is a do not care bit. It means that the mask keeps track of which of the bits in ER are care bits and which of the bits are do not care bits. Given the mask (M), the do not care bits in ER can be filled in an arbitrary manner, that results in a high compression ratio of ER.

Finally, TS, ER and M are compressed into CTS, CER and CM. CTS, CER and CM are all stored in the ATE (Fig. 5*c*).

Note, no simulation is needed in order to define unspecified bits in ER and all unspecified bits in ER will be masked away during test application. Hence, only care bits will be used for test response analysis and therefore there will be no problem with unknowns (Xs) in the test responses.

3.2 Test application

The scheme at test application is outlined in Fig. 4. Given is an ATE where CTS, CER and CM are stored. At test application, CTS, CER and CM are decompressed. In this example, the decompression is performed by a test program running on a processor.

The processor can be an on-chip processor or a processor placed on the ATE loadboard. The processor is used for decompression and must be tested prior to testing the DUT. In the case of an on-chip processor it is tested first, and in the case of a processor on the loadboard it is tested once for the batch of DUTs (all dies) that are to be tested.

In the case of hardware for decompression, it is tested prior to applying the test of the system.

The decompressor (software or hardware) takes CTS, CER and CM as inputs and produces decompressed test stimuli (DTS) decompressed expected responses (DER) and decompressed mask (DM). The DTS are sent and applied to the DUT and the produced responses (PR) are received by added EL. The EL hardware receives as input the PR, DER and DM and produces a signature (S) at every clock cycle that indicates if a fault is detected or not (0 if fault-free, 1 if fault). In the fault-free case, the test process proceeds; however, in the case of a detected fault, the testing can be aborted, and if desirable, the response bits can be shifted out for diagnostics.

The test EL in Fig. 4 is detailed in Fig. 6. For each produced test response bit, PR_i, the corresponding bit DM_i, and decompressed expected response bit DER_i defines a result bit R_i. R_i is stored in an added scan flip-flop (SFF) and S_i is used to produce the signature.

Table 1 contains the truth table for the evaluation of a response bit where the comments are:

A: $DM_i = 0$ and PR is masked
B: PR = ER—no fault
C: PR ≠ ER—a fault is detected

At each clock cycle *n* evaluation bits are combined by an or-operation into a single bit signature that indicates pass or fail. In our example with three scan-chains, there are three bits produced at each clock cycle.Assume at a given clock cycle the PR corresponding to the initial ER 'x1x' are '010'. The initial ER is filled such that when decompressed it is '110'. Even though ER '110' and PR '010' are different, the mask, which for ER 'x1x' is '010' will ensure that only the middle bit is used for test evaluation, and as the middle bits in ER and

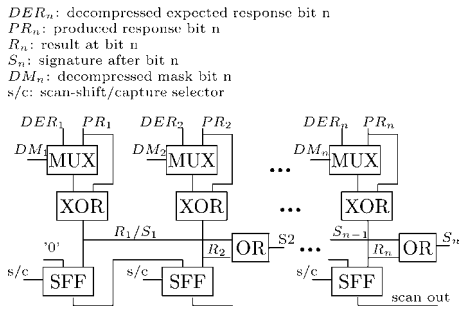

Figure 6 *Evaluation logic*

doi: 10.1049/iet-cdt:20070078

Table 1 Truth table for DER_i, PR_i and DM_i at test access mechanism (TAM) wire i

DER_i	PR_i	DM_i	S_i	Comment
—	—	0	0	comment A
0	0	1	0	comment B
1	1	1	0	comment B
0	1	1	1	comment C
1	0	1	1	comment C

PR are equal, there is no fault. The mask will ensure that only specified bits in ER are compared with PR.

In the case of a fault, it is possible to terminate the testing (abort-on-fail) at clock-cycle granularity or terminate the testing for diagnosis.

In the case of diagnosis, the position of a fault is stored in the added SFFs, and by shifting out the response bits stored in EL, it is possible to identify at which scan-chain the fault appeared.

Note, as the mask is used to identify the initially specified bits in ER, the problem with X-handling, where the simulation cannot define if an output should be 0 or 1, is solved. Only specified bits in the ER are used for test evaluation as the EL masks away unspecified bits using the mask data.

The overhead is for each bit an added multiplexer (MUX), an XOR, an OR (but for the first bit) and a SFF. If n bits are produced as test responses, n MUXes, n XORs, $n-1$ ORs and n SFFs are needed. The hardware cost is linear to n (number of produced response bits per clock cycle). Note, that the EL is test independent, and hence the order of tests and the tests themselves can be modified at any time.

Finally, as test evaluation is performed on-chip, the test data stored in the ATE, CTS, CER and CM, can be broadcasted to n parallel DUTs; hence the memory requirement is constant in the case of multi-site testing.

4 Experimental results

An architecture that addresses the high test data volumes that are needed for the testing of ICs has been proposed. The architecture allows simultaneous test data compression and abort-fail-testing. It also allows simultanous test data compression and diagnosis. The architecture has no problems with so-called unknowns(X).

Different from current test data compression architectures that compress TS and compact test responses, TS, ER and a mask are compressed here. As the proposed architecture compresses expected test responses and introduces a mask, the objective of the experiment is to demonstrate that high compression can be achieved when stimuli and ER including the mask are compressed. ATE memory saving with the proposed scheme at multi-site testing is also illustrated.

For the experiments, the ISCAS benchmark circuits and one industrial design have been utilised, and to compute the efficiency of the test data compression ratio, the following expression is used

$$\text{Compression}(\%) = \frac{\text{Original Bits} - \text{Compressed Bits}}{\text{Original Bits}} \times 100$$

The proposed architecture is not fixed to a certain decompression scheme. One software-based compression algorithm, namely the facsimile compression algorithm(facsimile coding standard is the ITU-T Group 3 standard) [4], and one hardware-based algorithm, the compression scheme propsed by Li and Chakrabarty [5] have been selected.

The basic idea in the facsimile compression algorithm is that a line on a printed paper is similar to the line just above and therefore only the difference is sent. A dot on the paper is coded to be either white or black, also known as Bi-level images. As the codewords in the facsimile standard are based on paper copy characteristics, an analysis on the test data has been made to find the most suitable coding words. Further, as the order in which the test vectors are applied does not impact the test result, the test data volume is ordered to achieve better compression. The size of the decompression program is 88 assembly instructions only.

First, the TS compression ratios are compared. The compression scheme propsed by Li and Chakrabarty [5] has been validated for TS compression. However, facsimile compression has not, and therefore it is validated against the methods by Chandra and Chakrabarty [18], Balakrishnan and Touba [20] and Balakrishnan and Touba [27]. The results collected in Table 2 show that facsimile produces results that are in a similar range as the approaches by Chandra and Chakrabarty [18] and Balakrishnan and Touba [20]. The approach by Balakrishnan and Touba [27] performs better than any other compression technique. From the results, it can be concluded that the facsimile compression as it is used here is an average compression method.

Secondly, the compression ratios of TS compression and expected test response compression are compared. The objective is to demonstrate that a similar compression ratio can be achieved for the

doi: 10.1049/iet-cdt:20070078

Table 2 Comparing TS compression

Circuit	FDR [18]		Matrix [20]		Linear [27]		Our architecture with Facsimile [24]	
	Comp. bits	% Comp.	Comp. bits	% Comp.	Comp. bits	% Comp.	Comp. bits	% Comp.
s13207	30880	81.30	33470	79.99	9920	94.44	25204	84.68
s15850	26000	66.22	23552	67.88	11168	87.65	19832	66.54
s38417	93466	43.26	69556	56.00	30432	82.58	66428	54.11
s38584	77812	60.91	66838	65.15	30208	84.25	72103	56.80
s5378	12346	48.02	10390	59.20	5696	81.39	11005	48.57
s9234	22152	43.59	16888	53.49	9280	74.27	14209	48.17

Table 3 Separate TS compression and expected test responses compression using facsimile compression

Circuit	Size, bits (stimuli only)	Stimuli only		Size (bits) (responses only)	Responses only	
		Comp. bits	% Comp.		Comp. bits	% Comp.
s838	5092	2218	56.44	2584	789	69.47
s9234	27417	14209	48.17	27750	12815	53.82
s38584	166896	72103	56.80	166896	72095	56.80
s13207	164500	25204	84.68	185650	28936	84.41
s15850	59267	19832	66.54	66348	23676	64.32
s5378	21400	11005	48.57	22800	10860	52.37
s35932	21156	3502	83.45	24576	3553	85.55
s38417	144768	66428	54.11	151554	71640	52.73

compression of expected test responses as for the compression of TS. Experiments where TS and expected test responses are compressed separately were performed and the result can be found in Table 3 for facsimile coding and in Table 4 using the compression proposed by Li and Chakrabarty [5]. In Table 3, column two gives the original size of the test set, columns three and four give the compressed size and the percentage. Columns five through seven show the same for the responses. Table 4 shows the compression ratio for TS and expected test responses for the ISCAS benchmark circuits. The results in Tables 3 and 4 confirm that compression of expected test responses reaches a similar compression ratio as the TS compression.

Thirdly, the compression of TS, expected test responses and the mask are compared. The results are collected in Table 5. The second column shows the original test data volume (TS and expected test responses). For each design the compressed data size and the compression ratio (percentage) in relation to

Table 4 Comparing TS, ER and mask compression with the compression technique proposed by Li and Chakrabarty [5]

Circuit	Compression ratio (%)		
	TS	Test responses	Mask
c7552	36.6	26.5	8.4
s838	59.0	57.7	55.5
s9234	55.9	56.5	29.6
s38584	64.0	62.8	34.4
s13207	80.2	76.5	56.0
s15850	60.5	60.0	39.4
s5378	60.6	58.7	39.4
s35932	46.6	89.8	86.0
s38417	62.4	58.9	6.7

doi: 10.1049/iet-cdt:20070078

Table 5 Combined compression of test stimuli and test responses with facsimile compression

Circuit	Original test data volume (stimuli and ER), (bits)	Compressed test data volume (stimuli, ER and mask), (bits)	Compression ratio, %
s838	7676	3979	48.16
s9234	55167	40754	26.13
s38584	333792	238922	28.42
s13207	350150	117236	65.52
s15850	125615	72680	42.14
s5378	44200	36556	17.29
s35932	45732	10434	77.18
Industrial design	14056068	4398738	68.71

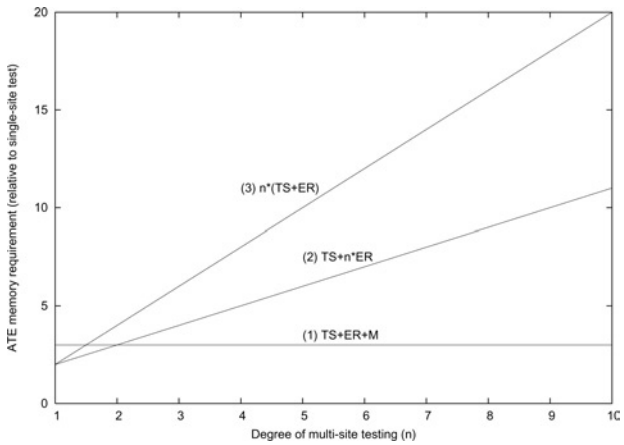

Figure 7 *Relative ATE memory requirement for TS, ER, and mask data (M) at multi-site (n) for the approaches (1) proposed scheme (TS + ER +M), (2) stimuli broadcast (TS + n × ER) and (3) standard (n × (TS + ER))*

the initial test data volume are shown. Note that the original test data volume includes only the original TS and expected test responses, and not the mask data, whereas the compressed test data includes TS, test responses and mask data. The results show that high compression can be achieved when the mask is included.

Fourthly, the ATE memory requirement at n-degree multi-site testing at no compression is shown. The following are compound:

1. A standard approach that duplicates TS and ER based on the degree of multi-site testing (n). The memory requirement then depends on the size of TS and ES as: ($n \times$ (TS + ER)),

2. A broadcast approach that duplicates ER only, and hence the ATE memory requirement depends on TS and ES as TS + $n \times$ ER and

3. The proposed approach, which has a constant memory requirement depending on the size of TS, ER and the mask M: TS + ER + M, and hence it is independent on the degree of multi-site testing

From Fig. 7 the ATE memory saving becomes clear using the proposed approach. The proposed architecture has a constant ATE memory requirement that is independent of the degree of multi-site testing.

5 Conclusions

Rapid semiconductor technology development makes it possible to fabricate increasingly complex and advanced ICs. IC fabrication is far from perfect and therefore each individual IC must be tested. As ICs are becoming more complex and advanced, increasing test data volumes are needed. The problem is that high test data volume requirement leads to a high ATE memory requirement, long test application time and low throughput.

Multi-site testing improves throughput; however, the ATE memory requirement increases. The long test application time can be lowered by making use of abort-on-fail testing where the testing is terminated as soon as a fault is detected, and hence less time is spent on faulty circuits. Test data compression can be used to lower ATE memory requirements. However, a major problem is that current test data compression architectures makes it difficult to employ multi-site testing, abort-on-fail testing and test data compression in an effective way. Further, current architectures reduce the diagnostic capabilities because of information loss in produced test response compaction and require special care to handle so-called Unknowns (X) in the produced test responses.

In this paper, an architecture that enables both test data compression and abort-on-fail testing is proposed. The architecture makes use of a novel way to handle test

doi: 10.1049/iet-cdt:20070078

data. All test data, TS and expected test responses are compressed and sent to the DUT. Test evaluation is performed on-chip. A scheme is proposed to allow arbitrary filling of the unspecified bits in expected test responses. The scheme ensures the possibility of reaching high compression of the expected test responses.

The advantages with the proposed architecture are the following. The architecture enables both test data compression and abort-on-fail testing. It is therefore possible to save test time and ATE memory in volume production. Further, the ATE memory requirement for architecture is constant ATE memory, independent of the number of DUTs that are tested concurrently. The architecture does not require test response compaction, hence, there is no information loss in the case of diagnosis and there are no problems with unknowns (X). Any number of unknowns, at any time, can be handled, and the added on-chip test EL is test independent.

A set of experiments on ISCAS benchmark circuits and one industrial design have been performed. The results show that the proposed scheme performs well compared with previous compression schemes in terms of the compression ratio.

6 Acknowledgment

The research is partially supported by the Swedish STRINGENT project.

7 References

[1] LARSSON E.: 'Combined test data compression and abort-on-fail testing', *Proc. Norchip*, 2006, pp. 137–140

[2] LARSSON E., PERSSON J.: 'An architecture for combined test data compression and abort-on-fail test'. Proc. Asia and South Pacific Design Automation Conf. (ASP-DAC), 2007, pp. 726–731

[3] LARSSON E., GILIANI I.: 'A test data compression architecture with abort-on fail capability'. Proc. IEEE Workshop on RTL and High Level Testing, 2005, pp. 86–91

[4] KAHLID S.: 'Introduction to data compression' (Morgan Kaufmann Publishers, 2000, 2nd edn.)

[5] LI L., CHAKRABARTY K.: 'Test data compression using dictionaries with fixed-length indices', *Proc. VLSI Test Symp. (VTS)*, 2003, pp. 219–224

[6] GUPTA R.K., ZORIAN Y.: 'Introducing core-based system design', *IEEE Des. Test Comput.*, 1997, **14**, (4), pp. 15–25

[7] ZORIAN Y., MARINISSEN E.J., DEY S.: 'Testing embedded-core based system chips'. Test Conf., 1998, October 1998, pp. 130–143

[8] MARINISSEN E.J., GOEL S.K., LOUSBERG M.: 'Wrapper design for embedded core test'. Proc. IEEE Int. Test Conf., 2000, pp. 911–920

[9] IYENGAR V., CHAKRABARTY K., MARINISSEN E.J.: 'Co-optimization of test wrapper and test access architecture for embedded cores', *J. Electron. Test., Theory Appl. (JETTA)*, 2002, **18**, (2), pp. 213–230

[10] IYENGAR V., CHAKRABARTY K., MARINISSEN E.J.: 'Test wrapper and test access mechanism co-optimization for system-on-chip'. Proc. IEEE Int. Test Conf., 2001, pp. 1023–1032

[11] GOEL S.K., MARINISSEN E.J.: 'SOC test architecture design for efficient utilization of test bandwidth', *ACM Trans. Des. Autom. Electron. Syst.*, 2003, **8**, (4), pp. 399–429

[12] IYENGAR V., CHAKRABARTY K., MARINISSEN E.J.: 'Test access mechanism optimization, test scheduling, and tester data volume reductio n for system-on-chip', *IEEE Trans. Comput.*, 2003, **52**, (12), pp. 1619–1632

[13] LARSSON E., ARVIDSSON K., FUJIWARA H., *ET AL.*: 'Efficient test solutions for core-based designs', *Trans. Comput.-Aided Des. Integr. Circuits Syst.*, 2004, **23**, (5), pp. 758–775

[14] POUGET J., LARSSON E., PENG Z.: 'SOC test time minimization under multiple constraints'. Proc. Asian Test Symp. (ATS), 2003, pp. 312–317

[15] LARSSON E., PENG Z.: 'Abort-on-fail based test scheduling', *J. Electron. Test., Theory Appl. (JETTA)*, 2005, **21**, (6), pp. 651–658

[16] INGELSSON U., GOEL S., LARSSON E., *ET AL.*: 'Test scheduling for modular SOCs in an abort-on-fail environment'. European Test Symp, 2005, pp. 8–13

[17] LARSSON E., POUGET J., PENG Z.: 'Defect-aware SOC test scheduling'. VLSI Test Symp, 2004, pp. 359–364

[18] CHANDRA A., CHAKRABARTY K.: 'Test data compression and test resource partitioning for system-on-a-chip using frequency-directed run-length (FDR) codes', *Trans. Comput.*, **52**, (8), pp. 1076–1088

[19] GONCIARI P.T., AL-HASHIMI B.M., NICOLICI N.: 'Improving compression ratio, area overhead, and test application time for system-on-a-chip test data compression/decompression'. Design, Automation and Test in Europe, March 2002, pp. 604–611

[20] BALAKRISHNAN K.J., TOUBA N.A.: 'Matrix-based test vector decompression using an embedded processor'. Defect and Fault Tolerance in VLSI Systems, November 2002, pp. 159–165

[21] JAS A., TOUBA N.A.: 'Deterministic test vector compression/decompression for systems-on-a-chip using an embedded processor', *J. Electron. Test., Theory Appl. (JETTA)*, 2002, **18**, (4/5), pp. 503–513

[22] WU Y., IVANOV A.: 'Single-reference multiple intermediate signature (SREMIS) Analysis for BIST', *IEEE Trans. Comput.*, June 1995, **44**, (6), pp. 817–825

[23] MITRA S., MITZENMACHER M., LUMETTA S., *ET AL.*: 'X-tolerant test response compaction', *IEEE Des. Test Comput.*, 2005, **22**, (6), pp. 566–574

[24] NAHVI M., IVANOV A.: 'Indirect test architecture for SoC testing', *Trans. Comput.-Aided Des. Integr. Circuits Syst*, 2004, **23**, pp. 1128–1142

[25] POEHL F., RZEHA J., BECK M., *ET AL.*: 'On-chip evaluation, compensation, and storage of scan diagnosis data - a test time efficient scan diagnosis architecture'. Proc. European Test Symp. (ETS), May 2006, pp. 239–246

[26] OLLIVIERRE S., KINSMAN A.B., NICOLICI N.: 'Compressed embedded diagnosis of logic cores'. Proc. Int. Conf. Computer Design, 2004, pp. 534–539

[27] BALAKRISHNAN K.J., TOUBA N.A.: 'Deterministic test vector decompression in software using linear operations'. VLSI Test Symp, April–May 2003, pp. 225–231

[28] PERSSON J.: 'Deterministic test vector compression/decompression using an embedded processor and facsimile coding', Master's Thesis, Linkopings Universitet, 2005 Sweden

doi: 10.1049/iet-cdt:20070078