Compressed Embedded Diagnosis of Logic Cores

Scott Ollivierre, Adam B. Kinsman and Nicola Nicolici
Department of Electrical and Computer Engineering
McMaster University, Hamilton, ON L8S 4K1, Canada
Email: ollivis@mcmaster.ca, kinsmaab@mcmaster.ca, nicola@ece.mcmaster.ca

Abstract

This paper introduces a new method for deterministic diagnosis of logic cores. The proposed method is based on on-chip decompression and comparison of incompletely specified test patterns and test responses. Using experimental data, the trade-offs between the number of tester channels, on-chip area and scan time are discussed.

Keywords: Built-In Diagnosis, Test Data Compression.

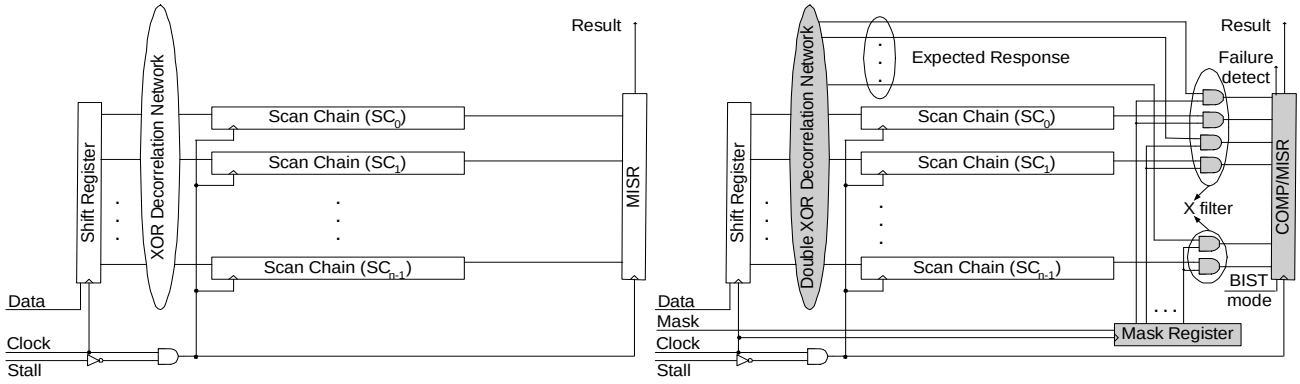
1 Introduction

As process technologies continue to shrink, designers are able to integrate most of the functional components found in a traditional system-on-a-board onto a single silicon die, called system-on-a-chip (SOC). This is achieved by incorporating pre-designed components, called intellectual property (IP) cores (e.g., processors, controllers, memories) into a single chip. While SOCs benefit designers in many aspects, their heterogeneous nature presents unique technical challenges for design validation and manufacturing test.

Fabrication anomalies in the integrated circuits (IC) manufacturing process may cause some devices to behave erroneously. *Manufacturing test* helps to detect physical defects (e.g., shorts or opens) prior to delivering the packaged circuits to end-users and it is an essential step during the SOC production that screens out the defective chips. Embedded deterministic test [5, 9, 14], which combines the benefits of low-cost automatic test equipment (ATE) and built-in self-test (BIST), is establishing itself as an enabling technology that can cost-effectively tackle the manufacturing test problem. Once a defective chip has been detected, comprehensive defect screening through *failure analysis* is required to adjust the manufacturing process and accelerate the transition from the yield learning phase to the volume production phase of a new manufacturing technology. Since the traditional physical methods used to identify failures are facing problems due to shrinking device sizes, the increased number of interconnect layers and flip-chip packaging technologies, automated logic *fault diagnosis* has already been established as an essential vehicle used to isolate the sources of the physical defects [13].

Embedded memory diagnosis is a well-documented area and reliable solutions are available based on the interaction between low cost ATEs and memory BIST technology [8]. What makes memory BIST particularly suitable for diagnosis is the functional nature and the regularity of memory tests, which facilitate easy on-chip generation of the expected response and its comparison against the computed memory output. To make a full reuse of the available test resources for failure analysis, BIST-based diagnosis of logic blocks (cores) has been researched over the last decade [1, 3, 4, 7, 10, 11, 15]. Since BIST signatures are observed at the end of each pseudorandom test session, first of all the failing pattern and the failing scan cells must be identified within the BIST session and only then the focus changes to finding the failing circuit node(s). To achieve this, additional on-chip hardware and test session partitioning schemes have been investigated. Motivated by the observation that defects affect only a small number of scan cells [10], which was confirmed also by our preliminary experiments, the question we aim to answer is how can the test data compression concepts, employed in state-of-the-art *deterministic* test solutions [5, 9, 14], be reused for cost-effective diagnosis?

The objective of this paper is to understand how controllability/observability of inspected faults can be achieved by leveraging the decompression hardware available for embedded deterministic manufacturing test. It is shown how the basic principles for diagnosis of logic cores can be reduced to the ones employed for memory cores, i.e., a low-bandwidth ATE is used in conjuction with on-chip generation and comparison hardware. A novel solution, based on *deterministic* masking of the scan cells that do not capture the targeted fault effects, is proposed and new encoding problems are formulated. Finally, experimental results show that, by using test data compression concepts and by solving the new encoding problems, we can use only a few tester channels to identify the failing scan cells and, at the same time, reduce the scan time and volume of diagnosis data, at the expense of limited area overhead, when compared to the available embedded test hardware.



(a) Embedded test architecture [5]

(b) Compressed embedded diagnosis architecture

Figure 1. Proposed extensions to an embedded test architecture for compressed diagnosis

2 Compressed Deterministic Diagnosis

Test data compression technology [5, 9, 14] exploits the don't cares (Xs) available in test patterns. For example, the architecture described in [5], and shown in Figure 1(a), employs a shift register to assemble a test data stream coming from a low-bandwidth ATE and then an XOR network (or phase shifter) is used to decorrelate the value stored in the register and apply it to a high number of internal scan chains. A Gaussian elimination solver operating in Galois Field(2) can be used to compute the input data stream based on a set of equations determined by the care bits in each clock cycle in every scan chain, the value stored in the shift register and the phase shifter's XOR functions. Whenever the system of equations cannot be solved, the stall signal will gate the clock used for the scan shifting process and new test data (equation variables) will be fed into the shift register until the pattern lockout is overcome.

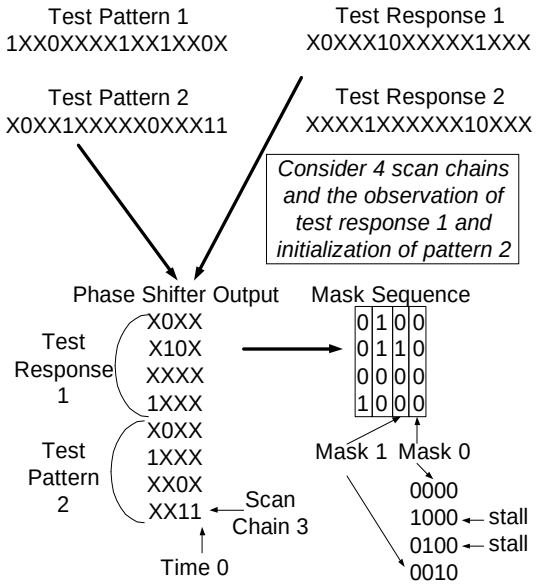


Figure 2. Dynamic mask loading.

Figure 1(b) shows how to exploit Xs in test patterns and test responses for compressed diagnosis. A masking mechanism, similar to the one described in [9, 10], is used to filter out the Xs from the test responses. The fundamental difference lies in the fact that the mask changes dynamically (i.e., from one shift to another) and that not only the outputs of the scan chains are masked, but also *the expected responses*, which are decompressed at the same time when the computed response is shifted out. The expected response decompression is achieved through an extended phase shifter that drives a number of channels equal to twice the number of physical scan chains. Figure 2 illustrates the decompression process, mask extraction and dynamic mask loading specific to the proposed architecture. Given two 16 bit patterns and 4 scan chains, we illustrate the scan cycle when the response 1/pattern 2 is shifted out/in. We consider that the rightmost 4 bits from the test pattern/response are part of the last scan chain and that the rightmost phase shifter column illustrates the values observed first at the scan outputs. In the bottom right part of the figure we show the masks, which need to be updated for every shift. The mask is taken into account only when the shifting process is active (i.e., no stall occurs). This ensures that all the flip-flops that do not capture the fault effects (Xs in test response) are filtered out and *only* the care bits of the expected and computed responses are fed into an integrated comparator/MISR (see Figure 3), which can work as a comparator in the diagnosis mode and a MISR in the BIST (or embedded test) mode.

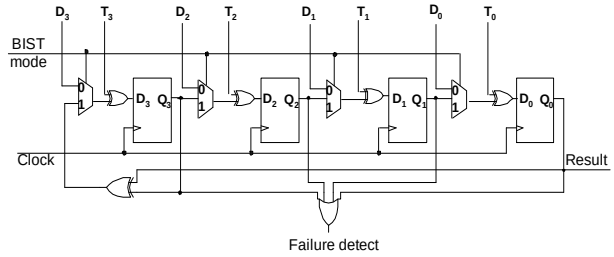
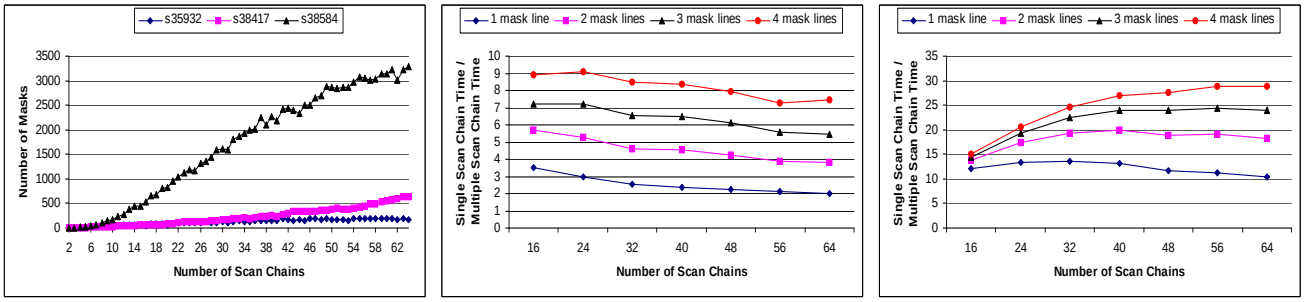


Figure 3. Integrated comparator and MISR.



(a) Masks vs. scan chains (b) Multiple mask lines for s38584 (c) Multiple mask lines for s35932

Figure 4. Number of masks for diagnosis test sets and impact of multiple mask lines on scan time

As it can be seen in Figure 1, the proposed architecture emulates the basic principle used in memory testing: the expected response of pattern i is decompressed on-chip concurrently with the scan input of pattern $i+1$ and compared on-chip against the computed response of pattern i. The on-chip comparison is performed using the proposed *Comparator/MISR unit*. In the BIST mode, this unit works as a MISR, while in the diagnosis mode the filtered expected responses, i.e., only the expected values in the flip-flops that capture the targeted fault (marked as D_3 to D_0 in Figure 3), are compared against the computed test response, i.e., the filtered scan chain outputs (marked as T_3 to T_0 in Figure 3). The *BIST mode* signal is a static signal driven by the SOC-level test controller. Whenever a mismatch is observed the *Failure detect* signal will be asserted and the *Result* can be shifted out using a memory diagnosis protocol (e.g., fail data streaming protocol [8]). It is important to stress that the scan cells, which are observed for each pattern, are selected in a deterministic fashion, unlike [10] where they are chosen pseudo-randomly. Therefore, the proposed architecture is particularly suitable for effect-cause analysis since the failing flip-flops are identified with the same accuracy as ATE-based approaches that require either larger ATE bandwidth or longer scan time. Whenever *selective application/observation* is sought, i.e., only a subset of input flip-flops needs to be set and a subset of output flip-flops needs to be observed (and all the remaining behavior is masked out), the proposed architecture can be employed.

A natural question is how will the increased number of channels driven by the phase shifter impact the decompression hardware size and scan time. When targeting a single fault, it was empirically observed that both the test patterns and the test responses are filled with many Xs. In addition, there are *implicit stall cycles* required to *dynamically* update the *mask register which filters Xs on the output*. By exploiting all the new variables fed into the input shift register while the mask register is updated, the input test pattern lockout occurs rarely. It is interesting to note that the same principle can be applied to the existing methods that eliminate *only a few Xs* (using static mask loading) during embedded deterministic manufacturing test (e.g, [9]).

3 Encoding Problems in Logic Diagnosis

Having introduced the basic principles of compressed embedded diagnosis of logic cores, there are a few questions which need to be answered. How many masks are necessary for a diagnosis test set that targets each fault individually? For a given test set, how does the number of masks vary with the number of scan chains? If, for an increased number of scan chains, the number of stall cycles needed to shift in a new mask will become larger, how can we overcome the excessive scan time penalty?

To answer the above questions, we have used ATALANTA [6] on three largest ISCAS89 benchmark circuits [2]. The tool has been used in the automatic test pattern generation (ATPG) mode and one pattern for each fault has been derived (all the unspecified positions on the input part were left as Xs, which, in turn, imply a large number of Xs on the output). The number of masks is upper bounded by the minimum value between the test sequence length (i.e., the number of test patterns multiplied by the sum of the largest scan chain length and the number of capture cycles) and two powered by the number of scan chains. Although in theory this may be a very large number, especially for long diagnosis test sets or many internal scan chains, it was found that the number of masks for s38417, s38584 and s35932 was within reasonable limits, as illustrated in Figure 4(a). In the following only the two extreme cases (s35932 and s38584) are used for scan time analysis illustrated in Figures 4(b) and 4(c). If no X filtering is required, and hence no stall cycles for mask loading are needed, the ratio between the scan time for the single scan chain and for multiple scan chains is equal to the number of scan chains. However, as clearly shown in Figure 4, when the number of scan chains increases, because of the excessive number of stall cycles required to feed in larger masks, the benefits in scan time reduction, due to the use of multiple scan chains, are diminished. By increasing the number of mask lines that drive the mask register (see Figures 4(b) and 4(c)), the scan time will decrease, however the volume of diagnosis data will grow since the number of input tester channels becomes larger as well. This ultimately defeats the very purpose of this work, which aims to find a cost-effective solution.

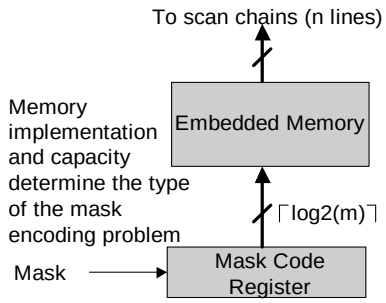


Figure 5. Encoding for stall time reduction.

The penalties in scan time, shown in Figures 4(b) and 4(c), are justified by the growing size of the mask register (when the number of scan chains increases), as well as the very low correlation between masks needed to filter out Xs in two consecutive shifts. For example, for an n bit mask register, if in time frame i there are no flip-flops that capture the effect of the fault, i.e., only Xs are shifted out and the mask value is all 0, and if in time frame $i+1$ there is a care bit only in the top scan chain (i.e., a 1 only in the rightmost position of the mask register) then it will take $n-1$ stall cycles between two consecutive shifts. To address the previously described problems, a new solution is required to reduce the number of stall cycles and, at the same time, keep the volume of diagnosis data low. The proposed solution analyzes the mask sequence, it encodes the masks on $\lceil \log_2 m \rceil$ bits, where m is the number of masks, and it uses an embedded memory, placed in between the mask code register and the scan chain outputs, to store the mask values (see Figure 5). The mask register from Figure 1(b) is replaced by the mask code register from Figure 5, whose outputs are used as address lines for the embedded memory that will drive the gray shaded AND gates from Figure 1(b). It is important to note that the mask code register length is independent of the number of scan chains n and it is dependent only on the number of distinct masks m. This approach is motivated by the observation that even for very long scan sequences the *number of masks is low*. The reason we are using an embedded memory and not dedicated hardware is that it provides re-programmability from one set of masks to another and, if it is already available on-chip for functional purposes, it can be reused without incurring any additional area overhead. Based on the above observations, mask encoding problems are introduced in the following.

The first problem addresses the case when the number of locations in the embedded memory is equal to the number of masks. There is a single code per mask (SCM) and the problem can be formulated as follows.

Problem P_{SCM}: Given the number of scan chains n, the number of masks m, the mask sequence over the entire diagnosis process, determine the code for each mask such that: (i) there is a *single* code for each mask; (ii) the number of stall cycles is minimized.

For the SCM problem, the embedded memory has $\lceil \log_2 m \rceil$ address lines, the number of data lines is n and the memory usage is m x n. Since the total number of binary values represented on $\lceil \log_2 m \rceil$ bits is $2^{\lceil \log_2 m \rceil}$, there are $\binom{2^{\lceil \log_2 m \rceil}}{m}$ x $m!$ distinct code assignments to be explored. This is an intractable problem and to introduce a minimal impact on the overall implementation time, we have developed fast greedy heuristics to solve it. The intuition behind our algorithms lies in analyzing how frequently two masks need to be applied one after each other during the entire diagnosis process. If the frequency is high then the assigned codes should lead to a low number of stall cycles.

A way to further reduce the number of stall cycles when the available embedded memory has a full address space, is to assign multiple codes to some masks. This can be done because there are $2^{\lceil \log_2 m \rceil}$ available address locations, however only m distinct masks. The multiple codes per mask (MCM) problem is stated as follows.

Problem P_{MCM}: Given the number of scan chains n, the number of masks m, the mask sequence over the entire diagnosis process, determine the *set of codes* for each mask such that: (i) there is *at least* one code for each mask; (ii) the number of stall cycles is minimized; (iii) the intersection of any two sets of codes is void and the cardinality of the union of all the sets of codes is $2^{\lceil \log_2 m \rceil}$.

When a certain mask appears in a test set, more than one code will result in the same mask appearing on the output of the memory. Thus whichever code results in the lowest shift time from the preceding code and to the next code will be used. Assigning multiple codes poses some problems, since the algorithms to assign codes to masks will become increasingly complex as to be really efficient they must take into consideration not only the number of transitions from mask to mask but which of codes are being used at the time.

An additional problem arises when the available memory to store the masks is constrained. If the available memory is 2^k x n bits, then the mask code size is limited to k. In this case, the given test set will be partitioned such that the union of all the masks of all the patterns in one partition (session) does not exceed 2^k. The memory will be reprogrammed after each session. To obtain the code assignments for each session, the algorithm for MCM can be applied iteratively. First, the memory constrained-MCM (MC-MCM) problem is formulated and then it is explained using Figure 6.

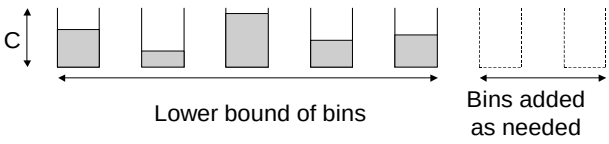


Figure 6. Test set partitioning under the embedded memory constraints.

Problem P_{MC-MCM}: Given the number of scan chains n, the number of masks m, the number of test patterns r and the mask sequence in every test response, and the maximum number of codes $C = 2^k$ for each diagnosis session, find a partition of the test set and determine the *set of codes* for each mask in every session such that: (i) there is *at least* one code for each mask in every session; (ii) *the number of sessions* and the number of stall cycles are minimized; (iii) the intersection of any two sets of codes in every session is void and the cardinality of the union of all the sets of codes in each session is $C = 2^k$.

It is very important to note that, because after each session a new set of masks is loaded in the embedded memory, the same mask code can be used *for different masks in different sessions*. The above problem can also be viewed as splitting all the test patterns into l bins such that the number of distinct masks m_l in each bin is less than or equal to the memory constraint C, and l is minimized, which guarantees that the number of times the memory needs to be reprogrammed is reduced. The lower bound for the number of bins (i.e., diagnosis sessions) is $\lceil (m-1)/(C-1) \rceil$ (we have -1 because the all 0 mask is guaranteed to appear for each test response in the capture cycle). However, this lower bound is highly unlikely to occur in practice, since it assumes that the masks of the test responses from one bin do not intersect the masks of the test responses from the other bins. A heuristic algorithm starts by initializing $\lceil (m-1)/(C-1) \rceil$ bins with patterns whose mask sets are mutually exclusive (see Figure 6). The algorithm then proceeds by assigning a pattern P_i to bin B_j such that the intersection of the masks from P_i and B_j is maximized and new bins will have to be added as existing ones reach capacity.

4 Experimental Results

This section discusses the cost trade-offs (area overhead, scan time and volume of diagnosis data) when using compressed embedded logic diagnosis for the tree largest ISCAS89 benchmarks [2]. The ATALANTA [6] diagnosis test sets, described in Section 3, are used in our experiments. Before analyzing scan time and diagnosis data volume, we discuss the impact of diagnosis hardware on area overhead.

The test and diagnosis architectures shown in Figures 1(a) and 1(b) have been synthesized for 16, 24, 32, 40, 48, 56 and 64 scan chains and technology mapped onto a standard cell library in 0.18μ CMOS technology [12]. The logic area is independent of the circuit under consideration, since it depends only on the number of scan chains. The results are shown in Figure 7. As it can be seen in the figure, the added test area to an embedded test (or BIST) only architecture is low. Its main contributing factors are the extended phase shifter, the mask register, AND gates used for X filtering and MISR extension for comparison in the diagnosis mode (highlighted by the shaded boxes in Figure 1(b)). Note, the memory area was not accounted for in Figure 7.

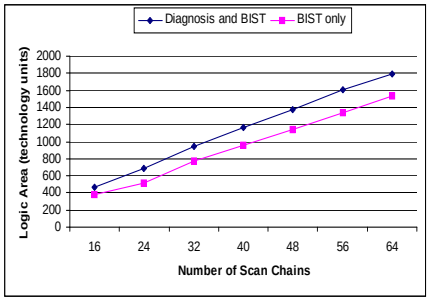
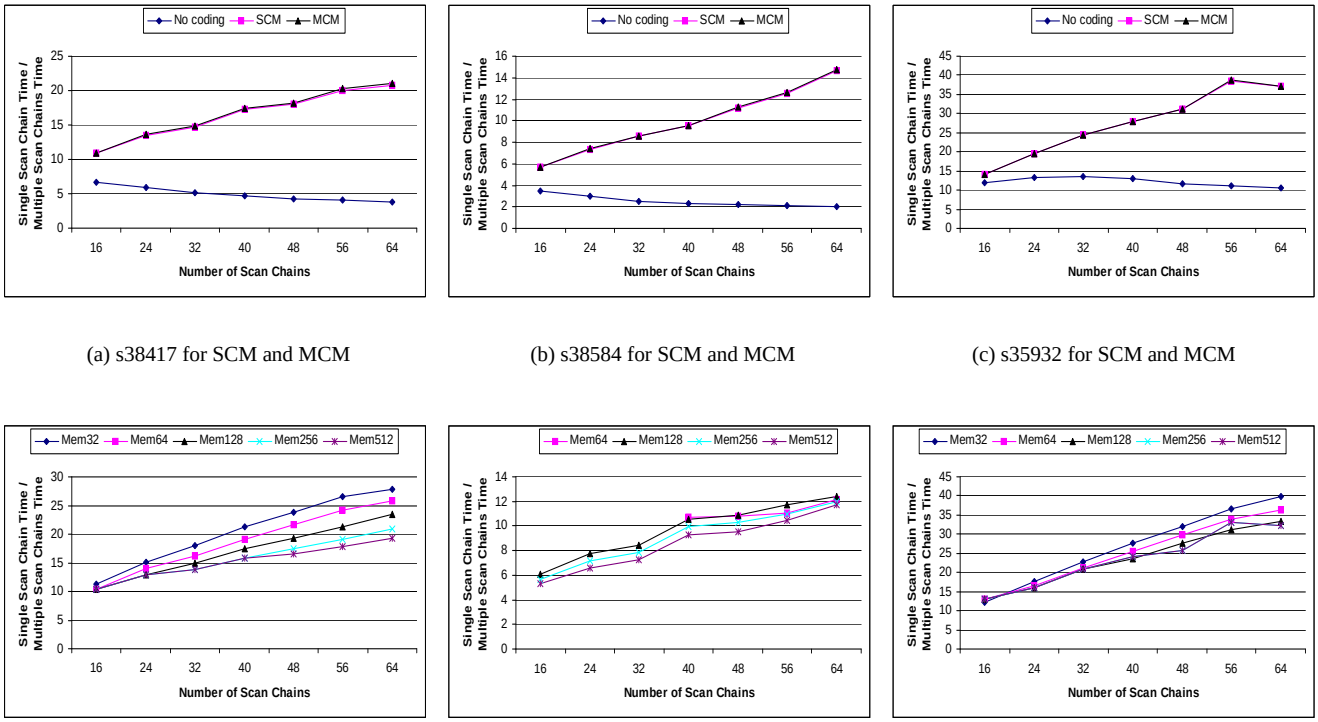


Figure 7. Area overhead comparison.

Figure 8 shows how mask encoding can maintain the scan time benefits of multiple scan chains. If no mask encoding is used the scan time may decrease when the number of scan chains increases and, therefore, additional hardware is not used in a cost-effective way. When mask encoding is employed, it can be observed that results obtained by solving both the SCM and the MCM problems are within the same range (Figures 8(a)-8(c)). This can be explained by the fact that the additional codes used by MCM are not sufficient to bring in visible benefits, as well as by our greedy implementation of these two intractable problems. If an embedded memory is not available on-chip, in order to keep the area overhead low, we need to solve the memory constrained problem (MC-MCM). From the results given in Figures 8(d)-8(f), we can see that using a small embedded memory (the number of locations varies from 32 to 512) can further decrease the scan time for s38417 and s35932. This is not the case for s38584 where a memory of 32 locations will not lead to feasible solution (there are single patterns with more than 32 codes). By varying the number of memory locations from 64 to 512, due to a high number of diagnosis sessions (bins in Figure 6), the overall time for diagnosis with MC-MCM will slightly increase when compared to SCM or MCM. This is because the scan time accounts also for the time needed to reload the memory after each session. Nevertheless, Figures 8(a)-8(f) have clearly shown that mask encoding reduces the scan time for all the three circuits, which, also influences the savings in volume of diagnosis data, as explained in next.

The proposed architecture needs only four data channels. As illustrated in Figure 1(b), three input channels (*Data*, *Mask* and *Stall*) carry test information and, whenever a flip-flop captures a failure, one output channel (*Result*) is used to stream out the failing scan position(s) using a memory test and diagnosis engine, such as the fail data streaming protocol [8]. The number of channels used for diagnosis needs to be multiplied by the overall scan time (including the stall cycles) to give us the total volume of diagnosis data. Since the volume of test data used by an ATE-based approach is given by the scan time for a single scan chain multiplied by two (both test data and response must be stored on the tester) the reduction in volume of diagnosis data is the reduction in scan time shown in Figure 8 divided by two.



(a) s38417 for SCM and MCM

(b) s38584 for SCM and MCM

(c) s35932 for SCM and MCM

(d) s38417 for MC-MCM

(e) s38584 for MC-MCM

(f) s35932 for MC-MCM

Figure 8. Scan time reduction when using mask encoding

5 Conclusion

This paper has described our experience with compressing diagnosis data by extending the existing knowledge from embedded deterministic manufacturing test and memory diagnosis. When both diagnosis patterns and responses are filled with Xs, a selective application/observation of patterns/responses for suspect faults can be employed and a low number of tester channels can be used with simultaneous reduction in scan time and volume of diagnosis data. The added logic area for diagnosis purposes is negligent and we believe that the potential benefits of the proposed solution outweigh its overhead. Finally, if the silicon debug process can be improved by selective application/observation, the very same infrastructure, based on on-chip comparison proposed in this paper, can be employed.

References

[1] I. Bayraktaroglu and A. Orailoglu. Cost-Effective Deterministic Partitioning for Rapid Diagnosis in Scan-Based BIST. *IEEE Design & Test of Computers*, 19(1):42–53, January-February 2002.

[2] F. Brglez, D. Bryan, and K. Kozminski. Combinational Profi les of Sequential Benchmark Circuits. In *Proc. International Symposium on Circuits and Systems*, pages 1929–1934, 1989.

[3] J. Ghosh-Dastidar and N. A. Touba. Fault Diagnosis in Scan-based BIST Using Both Time and Space Information. In *Proc. IEEE International Test Conference*, pages 95 –102, 1999.

[4] J. Ghosh-Dastidar and N. A. Touba. A Rapid and Scalable Diagnosis Scheme for BIST Environments With a Large Number of Scan Chains. In *Proc. IEEE VLSI Test Symposium*, pages 79–85, 2000.

[5] B. Koenemann, C. Barnhart, B. Keller, O. Farnsworth, and D. Wheater. A SmartBIST Variant with Guaranteed Encoding. In *Proc. IEEE Asian Test Symposium*, pages 325–330, 2001.

[6] H. K. Lee and D. S. Ha. On the Generation of Test Patterns for Combinational Circuits. Technical Report No. 12-93, Department of Electrical Engineering, Virginia Polytechnic Institute and State University, 1991.

[7] C. Liu and K. Chakrabarty. Failing Vector Identifi cation Based on Overlapping Intervals of Test Vectors in a Scan-BIST Environment. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 22(5):593–604, May 2003.

[8] S. Pateras. IP for Embedded Diagnosis. *IEEE Design & Test of Computers*, 19(3):46–55, May-June 2002.

[9] J. Rajski, M. Kassab, N. Mukherjee, N. Tamarapalli, J. Tyszer, and J. Qian. Embedded Deterministic Test for Low-Cost Manufacturing. *IEEE Design & Test of Computers*, 20(5):58–66, September-October 2003.

[10] J. Rajski and J. Tyszer. Diagnosis of Scan Cells in BIST Environments. *IEEE Transactions on Computers*, 48(7):724–731, July 1998.

[11] J. Savir. Salvaging Test Windows in BIST Diagnostics. *IEEE Transactions on Computers*, 47(4):486 –491, April 1998.

[12] Taiwan Semiconductor Manufacturing Corporation. TSMC 0.18μ CMOS technology. http://www.tsmc.com.

[13] S. Venkataraman and S. B. Drummonds. Poirot: Applications of a Logic Fault Diagnosis Tool. *IEEE Design & Test of Computers*, 18(1):19–30, January-February 2001.

[14] P. Wohl, J. A. Waicukauski, S. Patel, and M. B. Amin. Effi cient Compression and Application of Deterministic Patterns in a Logic BIST Architecture. In *Proc. IEEE/ACM Design Automation Conference*, pages 566–569, 2003.

[15] Y. Wu and S. M. I. Adham. Scan-based BIST Fault Diagnosis. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 18(2):203–211, February 1999.