JES
Journal of Engineering Sciences

Vol 13, Issue 03, MARCH /2022

ISSN NO:0377-9254

IMPLEMENTATION OF BUILT-IN SELF-TEST FOR MEMORY CELLS USING LFSR TEST PATTERN

[1]KAMARTHI NEERAJA, [2]C BHARGAV
[1]M.Tech Student, [2]Assistant Professor
DEPARTMENT OF ECE
St.Johns College Of Engineering and Technology, YEMMIGANUR.

Abstract— Design for testability (DFT) help in simplifying the 'manufacturing tests' used to detect post fabrication manufacturing defects in an integrated circuit (IC). The embedded memory tests in an integrated circuits utilize Memory Built In Self Test (MBIST) strategy. In this paper we have shown BIST technique and several algorithms used in BIST to test embedded memory. Such memory BIST technique comprises of address generator, controller, comparator and memory. The work presents the three different algorithms for implementing controller used in the memory BIST.

I.INTRODUCTION

The electronics industry has achieved a phenomenal growth over the last two decades, mainly due to the rapid advances in integration technologies, large-scale systems design - in short, due to the advent of VLSI. The number of applications of integrated circuits in high-performance computing, telecommunications, and consumer electronics has been rising steadily, and at a very fast pace. Typically, the required computational power (or, in other words, the intelligence) of these applications is the driving force for the fast development of this field. Gives an over view of the prominent trends in information technologies over the next few decades. The current leading-edge technologies (such as low bit-rate video and cellular communications) already provide the end-users a certain amount of processing power and portability. This trend is expected to continue, with very important implications on VLSI and systems design. One of the most important characteristics of information services is their increasing need for very high processing power and bandwidth (in order to handle real-time video, for example). The other important characteristic is that the information services tend to become more and more personalized (as opposed to collective services such as broadcasting), which means that the devices must be more intelligent to answer individual demands, and at the same time they must be portable to allow more flexibility/mobility.

Very-large-scale integration (VLSI) is the process of creating integrated circuits by combining thousands of transistor-based circuits into a single chip. VLSI began in the 1970s when complex semiconductor and communication technologies were being developed. The microprocessor is a VLSI device. The term is no longer as common as it once was, as chips have increased in complexity into the hundreds of millions of transistors. VLSI stands for "Very Large Scale Integration". This is the field which involves packing more and more logic devices into smaller and smaller areas. Thanks to VLSI, circuits that would have taken board full of space can now be put into a small space few millimeters across! This has opened up a big opportunity to do things that were not possible before.

VLSI has been around for a long time, there is nothing new about it, but as a side effect of advances in the world of computers, there has been a dramatic proliferation of tools that can be used to design VLSI circuits. Alongside, obeying Moore's law, the capability of an IC has increased exponentially over the years, in terms

of computation power, utilization of available area, yield. The combined effect of these two advances is that people can now put diverse functionality into the IC's, opening up new frontiers. Examples are embedded systems, where intelligent devices are put inside everyday objects, and ubiquitous computing where small computing devices proliferate to such an extent that even the shoes you wear may actually do something useful like monitoring your heartbeats.

Verilog was developed at a time when designers were looking for tools to combine different levels of simulation. In the early 1980s, there were switch-level simulators, gate-level simulators, functional simulators (often written ad-hoc in software) and no simple means to combine them. Further, the more-widespread, traditional programming languages themselves were/are essentially sequential and thus "semantically challenged" when modeling the concurrency of digital circuitry.Verilog was created by Phil Moore in 1983-4 at Gateway Design Automation and the first simulator was written a year later. It borrowed much from the existing languages of the time: the concurrency aspects may be seen in both Modula and (earlier) Simulate; the syntax is deliberately close to that of C; and the methods for combining different levels of abstraction owe much to Hilo.

In 1989, Gateway Design Automation (and rights to Verilog) was purchased by Cadence who put Verilog in the public domain in the following year. This move did much to promote the use of Verilog since other companies were able to develop alternatives tools to those of Cadence which, in turn, allowed users to adopt Verilog without dependency on a single (primarily workstation-tool) supplier. In 1992, work began to create an IEEE standard (IEEE-1364) and in December 1995 the final draft was approved. Thus Verilog has become an n international standard - which will further increase its commercial development and use. At present, there is standards activity to extend Verilog beyond purely digital circuits. This includes Verilog-MS for "mixed signal specification" and Verilog- A for "analog" design; the latter was recently approved (June 1996) by the board of Open Verilog International and is now under consideration by the IEEE. In addition, work is underway to automate the proof of "equivalence [between] behavioral and synthesizable specifications" (see the Cambridge web site below) to which Verilog readily lends itself.

While Verilog emerged from developments within private companies, its main rival came from the American Department of Defense (DOD). In 1981, the DOD sponsored a workshop on hardware description languages as part of its Very High Speed Integrated Circuits (VHSIC) program, and the outcome formed a specification for the VHSIC hardware description language (VERILOG) in 1983. There is, of course, the question as to which language is better. And this, of course, is a hard question to answer without causing excitement and rebuttals from the marketing departments of the less preferred language. However, the following points featured in a recent debate in the VERILOG and Verilog news groups.

The main factor is the language syntax - since Verilog is based on C and VERILOG is based on ADA. Verilog is easier to learn since C is a far simpler language. It also produces more compact code: easier both to write and to read. Furthermore, the large number of engineers who already know C (compared to those who know ADA) makes learning and training easier. VERILOG is very strongly typed, and allows programmer to define their own types although, in practice, the main types used are either the basic types of the language itself, or those defined by the IEEE. The benefit is that type

checking is performed by the compiler which can reduce errors; the disadvantage is that changing types must be done explicitly. Verilog has two clear advantages over VERILOG It allows switch-level modeling which some designers find useful for exploring new circuits. It ensures that all signals are initialized to "unknown" which ensure s that all designers will produce the necessary logic to initialize their de sign - the base types in VERILOG initialize to zero and the "hasty" designer may omit a global reset.

OBJECTIVE

THE aggressive scaling of microelectronic technology is enabling the fabrication of increasingly complex ICs. Together with several benefits (improved performance, decreased cost per function, etc.), this poses serious challenges in terms of test and reliability [1]–[7]. In particular, during at-speed test of high-performance microprocessors, the IC activity factor (AF) induced by the applied test vectors is significantly higher than that experienced during in field operation [5], [8], [9], [11], [13]–[15]. Consequently, excessive power droop (PD) may be generated, which will slow down the circuit under test (CUT) signal transitions. This phenomenon is likely to be erroneously recognized as due to delay faults

As a result, a false test fail will be generated, with consequent increase in yield loss [9], [13], [16]. At-speed test of logic blocks is nowadays frequently performed using MBIST (MBIST) [2], [8]–[10], which can take the form of either combinational MBIST or scan-based MBIST, depending on whether the CUT is a combinational circuit or a sequential one with scan [8], [12].

In case of scan-based MBIST, two basic capture-clocking schemes exist [8], [12]:

1) the launch-on-shift (LOS) scheme and

2) the launch-on-capture (LOC) scheme. In LOS schemes, test vectors are applied to the CUT at the last clock (CK) of the shift phase, and the CUT response is sampled on the scan chains at the following capture CK. In the LOC scheme, instead, test vectors are first loaded into the scan-chains during the shift phase; then, in a following capture phase, they are first applied to the CUT at a launch CK, and the CUT response is captured on the scan chains in a following capture CK [8].

In this paper, we consider the case of sequential CUTs with scan-based MBIST adopting an LOC scheme, which is frequently adopted for high-performance microprocessors. They suffer from the PD problems discussed above, especially during the capture phase, due to the high AF of the CUT induced by the applied test patterns. Solutions allowing designers to reduce PD during the capture phase in scan-based MBIST are therefore needed. While several approaches have been proposed to reduce PD for combinational MBIST (see [8], [11], [13]), only a few solutions exist for scan-based MBIST [2], [9], [17]–[21]. In [2], PD is reduced by a multicycle BIST scheme with partial observation. This approach does not impact on fault coverage (FC) (actually, it presents a slight FC increase of 5% compared with conventional scan-based-MBIST), but enables reduction of PD by 33% only, compared with conventional scan-based-MBIST.

In [9], PD can be reduced by more than 50% by alternately disabling groups of scan chains during test. However, this approach implies an increase of more than 90% in the number of test vectors required to achieve a target FC, with consequent increase in test time (TT), compared with conventional scanbased MBIST. In [21], a test pattern generator with a preselected toggling level is presented. It enables more than 50% reduction in the AF of the scan chains by preselecting the number of shift cycles during which the scan chains are loaded with constant logic values. However, it requires more than

60% increase in the number of test vectors (thus TT) to achieve the same FC as with conventional scan-based MBIST.

The solution in [17] and [18] relies on inserting an additional phase, namely a burst phase, between each shift and capture phase. Such a burst phase aims at increasing the current drawn from the power supply, up to a value similar to that absorbed by the CUT during capture phases. This way, the inductive component of PD occurs during the burst phase, and vanishes before the following capture phase. This solution causes an increase in both the total power consumed during test and TT. Omaña et al. [19], [20] recently proposed alternative approaches to reduce PD during scan-based MBIST, for the LOS scheme. They enable reduction of PD (up to 50% in [19], and up to 87% in [20]) by increasing the correlation between adjacent bits of the scan chains. However, these approaches do not increase the correlation between test vectors applied at the following capture cycles, so that they are not effective in reducing PD during scan-based MBIST adopting the LOC scheme. In this paper, we propose a novel, scalable approach to reduce PD during capture phases of scan-based MBIST, thus reducing the probability to generate false test fails during test. Similar to the solutions in [8] and [11], our approach reduces the AF of the CUT compared with conventional scan-based MBIST, by properly modifying the test vectors generated by the Linear Feedback Shift Register (LFSR). Our approach is somehow similar to reseeding techniques (e.g., that in [22]), to the extent that the sequence of test vectors is properly modified in order to fulfill a given requirement that, however, is not to increase FC (as it is usually the case for reseeding), but to reduce PD. The basic idea behind our approach (in its non scalable version) was introduced in [23].

In our proposed scalable approach, one (or more) test vector(s) to be applied to the CUT according to conventional scanbased MBIST is (are) replaced by new, proper test vector(s), hereinafter referred to as substitute test (ST) vector(s). The ST vector(s) is (are) generated based on the test vectors to be applied at previous and future capture phases in order to reduce the maximum number of transitions between any two following test vectors. This way, the CUT AF and PD are reduced compared with the original test sequence [11]. We consider the presence of a phase shifter (PS), which is usually adopted in scan-based MBIST to reduce the correlation among the test vectors applied to adjacent scan-chains [10]. As shown in [2], all test vectors to be applied at previous and future capture phases to any scan-chain are usually given at proper outputs of the PS, or the PS can be easily modified to provide them. In our approach, this property is exploited to enable its low-cost hardware implementation. However, our approach can also be adopted if the PS does not provide the previous and future test vectors for all scan-chains or if the scan-based MBIST does not present a PS. Indeed, as shown in Section IV, the previous and future test vectors of scan-chains can be obtained as a linear combination of proper LFSR outputs. Our approach is scalable in the achievable PD reduction.

Therefore, test engineers could choose the proper AF in order to avoid the following:

1) faulty chips being tested as good (due to an induced too low AF, lower than that experienced during normal operation);

2) good chips being tested as faulty (due to an induced excessive AF, higher than that experienced during normal operation). PD scalability is obtained by scaling the number of ST vectors to be applied between original test vectors. We will prove that our approach can reduce the maximum AF between the following capture phases from 50% (one ST vector only) to 89% (10 ST vectors) compared with

conventional scanbased MBIST. This is achieved without increasing the number of test vectors (thus TT) over conventional scan-based MBIST, for a given target FC. Moreover, our approach requires a very limited area overhead (AO) compared with conventional scan-based MBIST, which ranges from approximately 1.5% (1 ST vector) to approximately 14% (10 ST vectors). In addition, our solution requires substantially less test vectors (thus TT) to achieve a target FC compared with the alternative solutions in [9] and [21].

2. MEMORY BUILT-IN SELF-TEST

The functionality of electronics equipments and gadgets has achieved a phenomenal growth over the last two decades while their physical sizes and weights have come down drastically. The major reason is due to the rapid advances in integration technologies, which enables fabrication of many millions of transistors in a single integrated circuit (IC) or chip. Every IC in the industry follows Moore's law. According to Moore's law, number of transistors (transistor density) in an IC doubles in every 1.5 years. With the recent advances in the technology, device shrinks to nanometer scale, but density and complexity of the ICs keep on increasing.

This may result in many manufacturing faults and device failure. To accommodate more number of transistors, the device feature size is reduced. Reduction in the feature sizes results in increasing the manufacturing faults and fault detection becomes very difficult. VLSI testing is becoming more and more important and challenging to verify whether a device functions properly or not. Conventional automatic test equipment (ATE) based testing method is no longer able to handle the ever-growing test challenges. Logic built-in self-test (MBIST) is widely being adopted as the testing technique for most current day scan based designs. MBIST does not alter the scan structure of the designs permitting them to have both ATE based testing and also MBIST. The nature of vectors in MBIST are usually pseudo random and so even for a moderately sized design, several thousands of patterns are to be generated in the MBIST compared to a few hundreds of deterministic test patterns in ATPG to achieve adequate fault coverage. So, methods to improve the fault coverage of MBIST by increasing the pattern efficiency are constantly explored. MBIST found its use mainly in safety-critical (automotive, medical, military), mission-critical (deep-space, aviation) and high-availability (telecom) applications. However, process technologies plunging below 22nm, MBIST will become compulsory for applicationspecific integrated circuits (ASICs), application specific standard products (ASSPs) and complex commercial ICs (Nan Li, et al., 2015). Any electronic system employed in safety critical applications is expected to have a periodical self testing scheme for sustained error free operation.

For example, medical electronic devices need to test themselves to assure continued safety of the patients. Another example is automotive electronics. With the explosion in the growth of the automotive semiconductors industry comes an associated and intense focus on high silicon quality and reliability. The last thing anyone wants is a brake system failure due to a latent silicon defect, and concerns over reliability are driving changes in the testing requirements for these chips. The electronics must meet certain safety standards to accommodate the fast growing technological revolution.

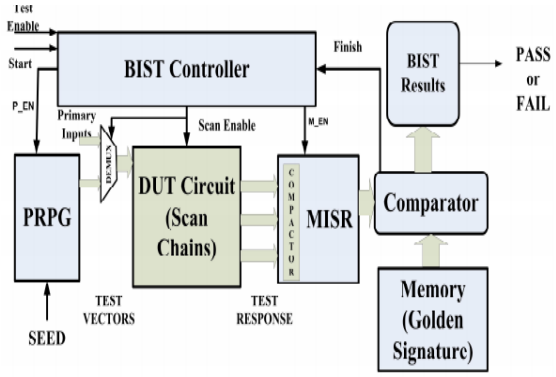


Figure 1. MBIST Architecture

Seed selection algorithm compresses the exhaustive test pattern generated by MBIST. It provides high compression ratio, as a result the number of test vectors applied to DUT are reduced, which in turn reduces the test application time and power as well. In BitFixing in Pseudorandom Sequences, deterministic test cubes that detect the random-patternresistant faults are embedded in a pseudorandom sequence of bits generated by a linear feedback shift register (LFSR). The proposed method uses STUMPS (Self-Test Using a MISR and Parallel Shift register) architecture which is widely used in practice (Agarwal, 2002).

3. PROPOSED METHOD

The design technique, technique, BIST, is used to test the circuit itself using some parts of the same circuit. With properly designed BIST architecture the additional testing hardware expense will be very much balanced by the advantages in terms of enhanced reliability and minimized maintenance expense. BIST minimizes total cost by reducing test pattern creation attempts at all levels, minimizing testing work at chip, improving system level maintenance and enhancing component repair [1][2].Moreover it can test numerous circuits simultaneously as well as minimized testing time is provided at operational real time clock. It reduces test-cycle duration [3].Every household appliances, computer, laptops uses memory chips. So BIST enabled memory ultimately helps everybody to serve better [4]. The proposed work concentrates on the standard RAM chip design with BIST enabled architecture using linear feedback shift register (LFSR) with the help of Very High Speed Integrated Circuit Hardware Description Language (VHDL) [5].

GENERAL BIST CONSTRUCTION

The fundamental BIST construction needs the inclusion of additional three h/w building blocks to a digital circuit with Circuit under test (CUT). They are i) A test pattern generator ii) A response analyzer and iii) A test controller as shown in Fig 1.

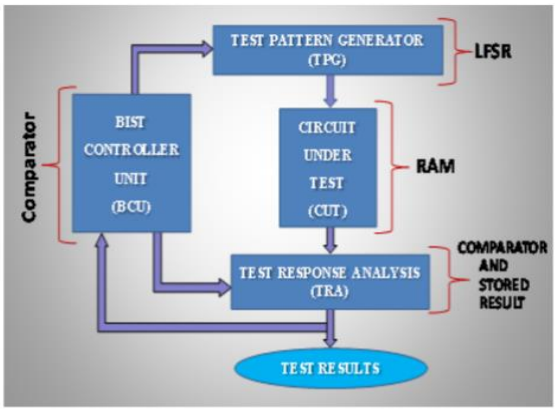


Fig 1: General BIST Construction

The test pattern generator creates the test patterns for the CUT. In standard way of exercise, the CUT accepts its inputs from other building blocks and execute the task for which it was planned. During test mode, pseudo random test patterns are given to the CUT, and the output test results are evaluated by comparator. For the proposed work CUT is standard RAM, Test Pattern Generator (TPG) is implemented by simple LFSR, BIST Controller Unit (BCU) and Test Response Analyzer (TRA) is implemented by comparator.

CIRCUIT UNDER TEST(RAM) The main building block of the proposed work is standard RAM as shown in Fig. 2.

Journal of Engineering Sciences

Vol 13, Issue 03, MARCH /2022

ISSN NO:0377-9254

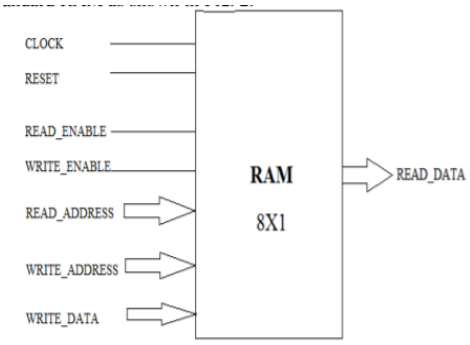


Fig. 2: RAM architecture block diagram

Table 1: Port Name and purpose of standard RAM

PORT NAME	TYPE	DESCRIPTION
CLOCK	INPUT	Provides clock signal to the block.
RESET	INPUT	Provides reset signal to the block
READ_ENABLE	INPUT	Enable reading operation of a ram.
WRITE_ ENABLE	INPUT	Enable reading operation of a ram.
READ_ADDRESS	INPUT	It shows that address from where reading operation will be done.
WRITE_ADDRESS	INPUT	It shows that address where we have to write our data.
WRITE_ DATA	INPUT	It is that data which should be written on given write address.
READ_DATA	OUTPUT	It will show the data placed at given read address after reading operation

LFSR (BIT PATTERN GENERATOR)

LFSR (Linear Feedback Shift Register) is a combination of shift register and XOR gate. LFSR generally produces pseudorandom sequences. A pseudo-random binary sequence is basically an array of strings consisting of '0's and '1's which seems to be random in the appearance, but actually the sequence is repeated after some clock pulse according to design. LFSR is mainly a combination of a D-flip flop and a XOR gate. Here three LFSRs are used for three different working purposes. They are used for generating random sequences for write address, read address and write data. The block diagrams of the LFSRs and the pseudo random sequences are shown below. LFSR1: It is generating pseudo random sequences for read address that is going into ram when test mode is on.The architecture and simulation are shown in Fig. 4 and Fig. 5.

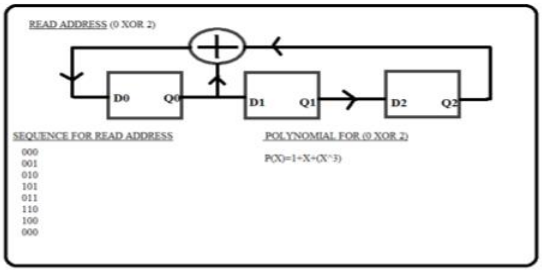


Fig. 4: Architecture of LFSR1 (generate Read Address)

LFSR2: It is used for generating pseudo random sequences for write address that is going into ram when test mode is on. The architecture and simulation result are shown in Fig. 6 and Fig. 7.

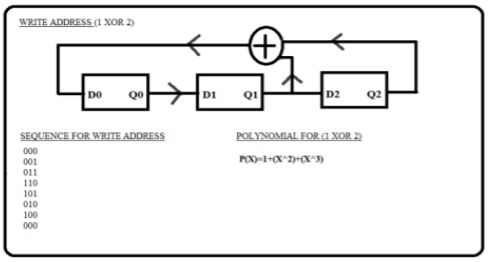


Fig. 6: Architecture of LFSR2 (generate Write Address)

BIST ENABLED RAM (FINAL PRODUCT)

As discussed earlier BIST architecture need some extra pins to test the CUT. In this proposed work single extra pin named as check_pin is used to enable the testability of the CUT as shown in Fig. 10.Check_pin=0 indicates normal mode and check_pin=1 indicates test mode. In normal mode the whole architecture will work as a normal RAM and READ_DATA output will not be compared with the stored output. In test mode RAM will accept the data from all three LFSRs and READ_DATA output will be compared with the expected stored output by comparator and BIST_OUTPUT will indicate the status of the RAM. The comparator compares the stored expected output with the read_data and asserts the error signal (BIST_OUTPUT) when test mode is on. In normal mode comparison is not needed so the

error signal will be disabled. Test mode and normal mode is differentiated with the help of three 2:1mux as shown in Fig. 11

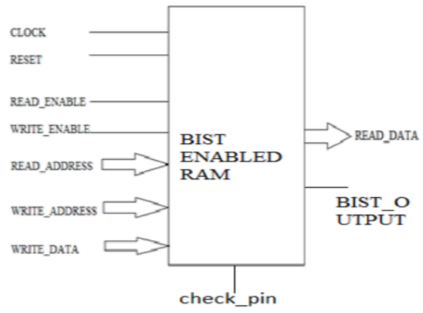


Fig. 10: Entity of Final Product

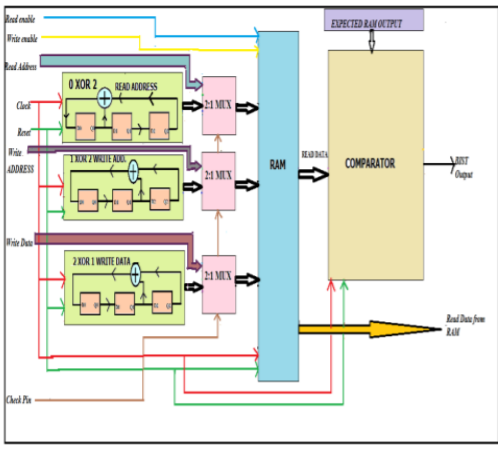


Fig. 11: Internal Diagram of Final Product

The complete working principle is shown with the help of a flow chart in fig. 12.

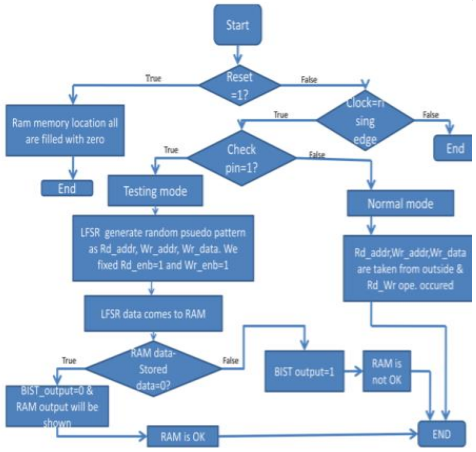


Fig. 12: Flow Chart of the final product

4. EXTENSION METHOD CONSIDERED SCENARIO

We consider the conventional scan-based MBIST (ConvMBIST) architecture shown in Fig. 1 [8], [10]–[12], [14]. The state flip-flops (FFs) of the CUT are scan FFs, arranged into many scan chains (s scan chains in Fig. 1). The pseudorandom pattern generator is implemented by an LFSR [10], [12], [14]. The PS, which reduces the correlation among the test vectors applied to adjacent scan-chains [10], is composed of an XOR network expanding the number of outputs of the LFSR to match the number of scan chains s [10]. As discussed more in

detail in Section IV, the PS gives to its output the current LFSR output configuration, together with future/past configurations at each shift CK.

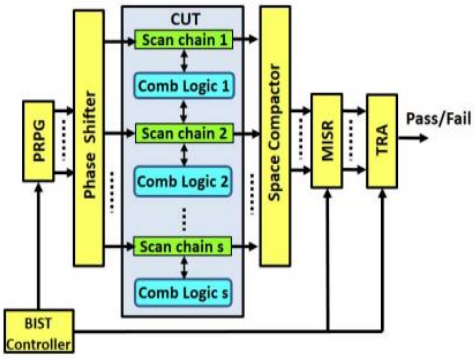


Fig. 1. Schematic of the considered scan-based MBIST architecture.

The Space Compactor compacts the outputs of the s scan chains to match the number of inputs of the Multiple-Input Signature Register (MISR). The MISR, the test response analyzer, and the BIST Controller are the same as in combinational scan-based MBIST [8], [12]. As for the scan FFs, our approach requires that, during shift phases, they maintain the last test vector applied to the CUT at their outputs. This is guaranteed by the scan-FF in [24], which is frequently employed in microprocessors [24], and considered here as a significant example.

However, this can also be achieved with other different scan FFs. The internal structure of this FF is shown in Fig. 2. It consists of two sub-

blocks, namely, the scan portion and the system portion, each consisting of a master-slave FF composed of two latches (Latches LA and LB for the scan portion, and latches PH2 and PH1 for the system portion) [24]. The latches have two clocks, and sample one out of two input data lines, depending on which clock is active [24].

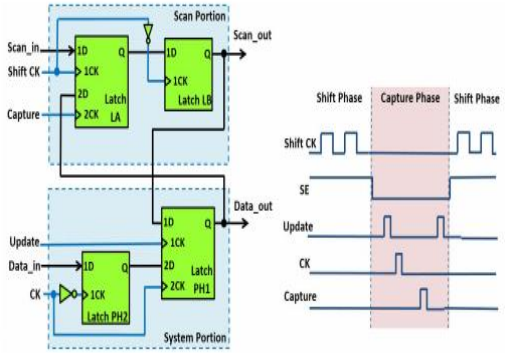


Fig. 2. Considered scan FF in [24] and signals' timing.

The clocking scheme adopted to implement an LOC strategy is also reported in Fig. 2. It consists of a shift phase [scan enable (SE = 1)] and a capture phase (SE = 0). During the shift phase, a new test vector is loaded in the scan chains after n shift CKs, where n is the number of scan FFs of the longest scan chain. At each shift CK, a new bit of the test vector present at the scan_in of latch LA is shifted to the scan_out of latch LB. We refer hereinafter to T m i as the part of the test vector that is loaded in the mth scan chain (m = 1 ...s) and is applied to the CUT at the ith capture phase. During the capture phase, a pulse is first applied on the Update clock (launch CK) to load the test vector (loaded on LB latches at the previous shift phase) on the PH1 latches. Thus, after the pulse on Update, the test vector T m i (m = 1 ...s) is applied to the CUT. Then, a pulse is applied on CK (capture CK) to sample the CUT response on PH1 latches. Then, the CUT response is copied to the latches LA (to enable shifting it out at the following shift phase) by applying a pulse on the Capture clock (Fig. 2). Finally, another pulse is applied on Update to load T m i again on the latches PH1. Since Update is not asserted during the following shift phase, T m i is maintained at the inputs of the CUT until the following test vector T m i+1 is applied. Therefore, in the LOC clocking scheme in Fig. 2, the CUT AF (thus PD) that occurs between the launch and capture CKs (i.e., between Update and CK pulses) and that may generate false test fails during testing, is proportional to the number of transitions between the following test vectors T m i and T m i+1.

SCALABLE APPROACH

As we introduced in Section I, the goal of our approach is to reduce the PD that may generate false test fails during atspeed test with scan-based MBIST. Such a PD occurs after the application of a new test vector to the CUT. This occurs at the launch CK (Update pulse in Fig. 2) within capture phases. The generated PD is proportional to the CUT AF induced by the application of a new test vector, which in turn depends on the AF of the scan FFs' outputs [8]. For the considered scan FFs (Fig. 2), such an AF depends on the number of FFs' outputs switching when the new test vector is applied. Therefore, the target of our approach is to reduce the number of FFs' outputs transitions occurring after the application of a new test vector to the CUT. In order to derive a mathematical description of our proposed solution, we make the following simplifying assumptions for Conv-MBIST.

1) All scan chains have the same number of scan FFs.

2) The maximum AF between two following test vectors T m i and T m i+1 is the same for all scan chains (m = 1 ...s). However, by logic-level simulations performed by the Synopsys Design Compiler tool, we have verified that our approach can achieve the same AF reduction also if such simplifying hypotheses are not satisfied.

Approach With 1 Substitute

Test Vector For each scan chain m (m = 1 ...s), one ST vector STm i replaces the original test vector T m i to be applied to the CUT at the ith capture phase according to Conv-MBIST (Fig. 3). It will be shown that this enables a 50% AF reduction compared with Conv-MBIST. In our approach, the ST vector STm i to be charged in the Scan-Chain (SC) m and applied to the CUT at the ith capture phase is constructed based on the structure of test vectors T m i−1 and T m i+1 to be applied at the (i−1)th and (i+1)th capture phases. Assuming the presence of a generic PS, our solution exploits the fact that, during the shift phase preceding the ith capture phase, test vectors T m i−1 and T m i+1 are given at proper outputs of the PS. Should some test vectors not be produced at the PS outputs, the PS could be easily modified to generate them.Denoting by STm i (j), T m i−1(j), and T m i+1(j) the logic value of the jth bit in test vectors STm i , T m i−1, and T m i+1, respectively, STm i (j) is chosen as follows:

$$\mathrm{ST}_i^m(j) = \begin{cases} T_{i-1}^m(j), & \text{if } T_{i-1}^m(j) = T_{i+1}^m(j) \\ R, & \text{if } T_{i-1}^m(j) \neq T_{i+1}^m(j) \end{cases}$$

where R denotes a random bit. Therefore, in all bit positions j in which test vectors T m i−1 and T m i+1 present the same logic value, STm i maintains the same logic value as in the previous test vector T m i−1. Instead, in the bit positions j in which test vectors T m i−1 and T m i+1 differ, STm i assumes a random logic value R. The bit R can simply come from one of the outputs of the LFSR, as suggested in [8].

Starting from the (i − 1)th capture phase (Fig. 3), the new test vector sequence in each scan chain m will be as follows:

$$T_{i-1}^m - \mathrm{ST}_i^m - T_{i+1}^m - \mathrm{ST}_{i+2}^m - T_{i+3}^m \cdots$$

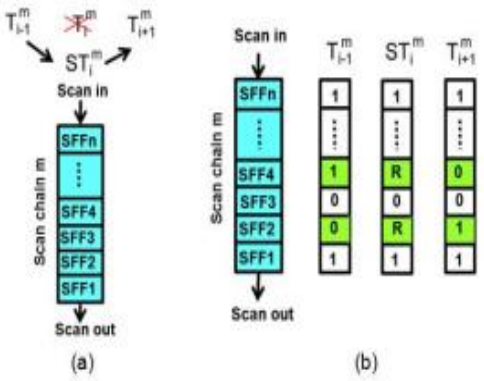


Fig. 3. Schematic of (a) sequence of test vectors filling each scan chain m, (b) bits in the ST vector STm i and in the test vectors applied/to be applied at the previous/following capture phase (T m i−1/T m i+1).

Therefore, the number of bits changing logic value between the following test vectors with the new sequence T m i−1−STm i − T m i+1 will be equal to, or smaller than, those with the original test sequence T m i−1 − T m i − T m i+1 of Conv-MBIST. In this regard, it is to be noted that the considered scanFFs (Fig. 2) update their outputs only at capture phases, while maintaining them constant during the shift phases. Therefore, the AF between successive test vectors will determine the AF of the CUT at each capture cycle. The presence of a random bit R in STm i in the bit positions where T m i−1 and T m i+1 differ allows the new sequence T m i−1 − STm i − T m i+1 to preserve the randomness of the original sequence [8]. Therefore, as shown in Section V, the number of test vectors required to achieve a target FC does not increase compared with the application of the original test sequence. The maximum AF between the following test vectors loaded in each SC in Conv-MBIST (AFsc con) is reduced to a half (AFsc con/2) by our approach. Consequently, denoting by AFtot 1ST the maximum AF between any two successive test vectors applied to the CUT at successive capture phases, for our approach with 1 ST vector, it is

$$AF_{1ST}^{tot} = AF_{con}^{tot}/2$$

where AFtot con is the max AF obtained with Conv-MBIST.

2.4 Approach With N Substitute Test Vectors

In order to reduce further the AF during capture phases of scan-based MBIST, a higher number N of ST vectors, STm i , STm i+1 ... STm i+N−1, with STm i+1 = ⋯ = STm i+N−1 = STm i , can be used to replace, for each scan chain m, N original test vectors T m i up to T m i+N−1. Similar to the case of 1 ST vector, the ST vectors STm i ... STm i+N−1to be applied at the ith ...(i + N − 1)th capture phases are constructed based on the test vector T m i−1 to be applied at the (i − 1)th capture phase, and the test vector T m i+N to be applied at the (i + N)th capture phase. Denoting by STm i (j), T m i−1(j) and T m i+N (j) the logic value of the jth bit in test vectors STm i , T m i−1 and T m i+N , respectively, STm i (j) is determined as follows:

$$ST_i^m(j) = \begin{cases} T_{i-1}^m(j), & \text{if } T_{i-1}^m(j) = T_{i+1}^m(j) \\ R, & \text{if } T_{i-1}^m(j) \neq T_{i+1}^m(j) \end{cases}$$

where, as before, R denotes a random bit. The number of bits changing logic value between successive test vectors with the new sequence T m i−1 − STm i − ⋯ − STm i+N−1 − T m i+N (Fig. 4) will be equal to, or smaller than in the original test sequence T m i−1 −T m i −⋯−T m i+N−1 −T m i+N of Conv-MBIST. The presence of a random bit R in STm i in the bit positions where T m i−1 and T m i+N differ allows the new sequence T m i−1 −STm i −⋯−STm i+N−1 − T m i+N to preserve the randomness of the original sequence in these bit positions [8]. As a result, as shown in Section V, the number of test vectors required by our approach to achieve a target FC is approximately the same as that in Conv-MBIST, even for the case of N = 10 ST vectors. As represented in Fig. 5, we interleave the insertion of the N ST vectors, so that they are applied at different capture phases for the different SCs. Thus, between any two successive capture phases, the same ST vector is loaded in (N − 1)-out-of -(N + 1) scan chains, which consequently exhibit AFsc = 0. Instead, 2-out-of-(N+1) scan chains present a transition between an original test vector and an ST vector, thus presenting an AFsc = AFsc con/2. If the number of scan chains s is a multiple of N + 1, the total AF between any two following test vectors is:

$$AF_{NST}^{tot} = \sum_{m=1}^{s} AF^{sc\_m} = \frac{s}{N+1}\left(\frac{AF_{con}^{sc}}{2} + \frac{AF_{con}^{sc}}{2} + 0\right) = \frac{AF_{con}^{tot}}{N+1} \quad (1)$$

where, as before, AFtot con = sAFsc con. We have verified that, even if s is not a multiple of N + 1, and s N (e.g., s > 10N), that is if the number of SCs s is much higher than the number of ST vectors N, (1) gives a good approximation of the AFtot NST. From (1) we can also derive that, with our approach, it is AFtot = AFtot con/3 for N = 2, AFtot = AFtot con/4 for N = 3, AFtot = AFtot con/5 for N = 4, and so on. As will be shown later, such reductions are achieved at no increase in the number of test vector (TVs) needed to reach a target FC, and with a limited cost in terms of AO.

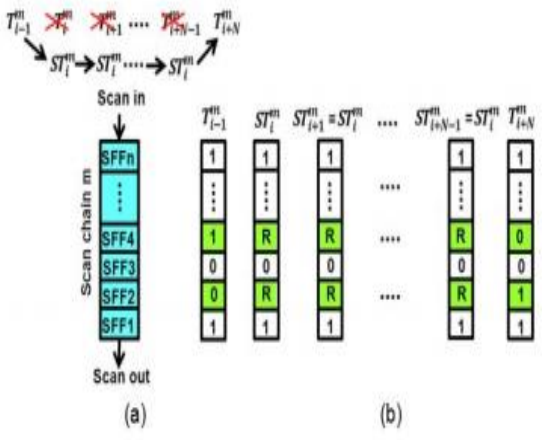


Fig. 4. Test vectors' selection of our approach with N ST vectors. (a) Sequence of test vectors

filling each scan chain. (b) ST vectors STm i ... STm i+N−1.

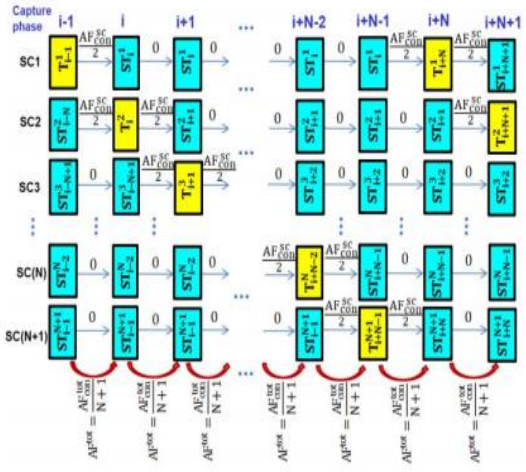


Fig. 5. Interleaved test vectors' application to the CUT and AF values for each capture phase for the case with N ST vectors.

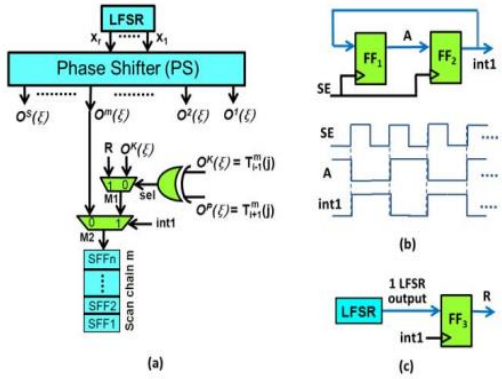


Fig. 6. Schematic of (a) possible implementation of our approach, (b) possible scheme to generate signal int1, and (c) strategy to generate the random bit R.

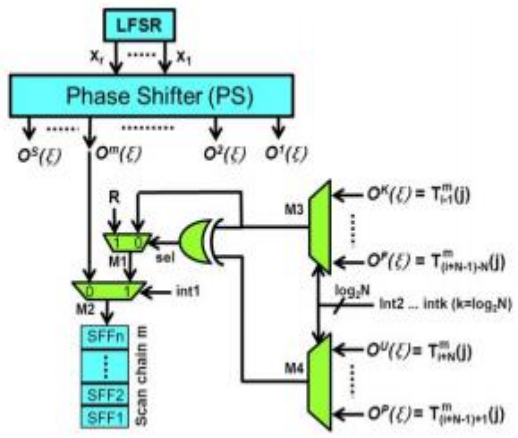


Fig. 7. Schematic of a possible implementation of our approach with N ST vectors

We compare our approach with Conv-MBIST [12] and the solutions in [9] and [21]. We consider the PD reduction and the number of test vectors required to achieve a target FC as metrics for comparison. We consider the FC for tuck-at faults as in [9], and we evaluate the AO equired by our approach over Conv-MBIST. Our approach has been validated by logiclevel simulations, as the alternative solutions in [9] and [21] that we consider for comparison purposes.

Comparison With Conv-Scan-Based MBIST

As for the effectiveness in reducing PD during scan-based MBIST, we have evaluated the maximum AF between any two following test vectors (to be applied at following capture phases), which is proportional to the CUT AF, thus also to its PD. Our approach has been implemented with up to 10 ST vectors. For each CUT, we have considered the maximum stuck-at FC achievable with Conv-MBIST as target stuck-at FC. The number of test vectors required to achieve such an FC has been evaluated by means of the Synopsys TetraMAX tool. Finally, the AO required by our approach over Conv-MBIST has been evaluated by the Synopsys Design Compiler tool. It should be noted that our approach requires no hardware modification of the considered scan FFs.

For comparison purposes, our approach has been applied to the largest four ISCAS'89 benchmarks considered in [9] (s38584, s38417, s13207, and s15850). For all circuits, we have used a 20-bit LFSR, with characteristic polynomial p(x) = x 20 + x 17 + 1 [25]. As for the PS, it has been implemented in order to minimize AO, according to the rules in [10]. Fig. 8(a)–(d) shows the AF of Conv-MBIST for the four considered benchmarks, as well as the AF and AO of our approach, as a function of the number of ST vectors. We can see that, for all benchmarks, our approach allows us to reduce noticeably the AF, thus also PD, with respect to Conv-MBIST. In this regard, it is worth noticing that the AF achieved by our solution reduces quickly as the number of ST increases for a small number of ST vectors (i.e., from 1ST to 5 ST), while it tends to saturate for more than 6 ST vectors. Particularly, with respect to the AF of Conv-MBIST, the AF achievable by our approach becomes approximately 50% with 1 ST vector, 33% with 2 ST vectors, 25% with 3 ST vectors, and 9% with 10 ST vectors. Fig. 8(a)–(d) also reports the relative AF reduction allowed by our approach over Conv-MBIST [AF = 100∗(AFOUR − AFConv−MBIST)/AFConv−MBIST] as a function of the number of ST vectors. We can observe that, for a number of ST vectors higher than 4, our approach enables an AF reduction higher than 80%. Moreover, we can note that, for all benchmarks, the AO of our approach over Conv-MBIST increases linearly with the number of ST vectors. A minimum of approximately 1.5% AO is achieved with 1 ST for the s38584 benchmark, and a maximum of approximately 14% AO is reached with 10 ST vectors for the s13207 benchmark.

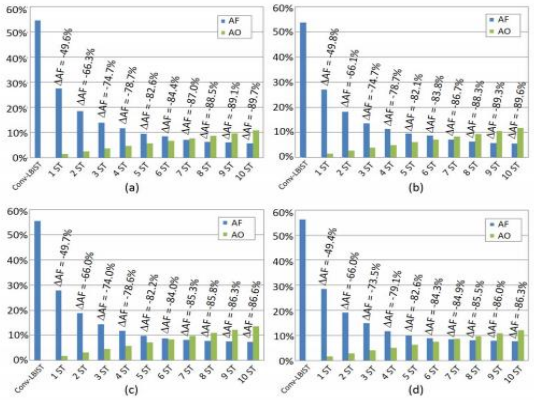
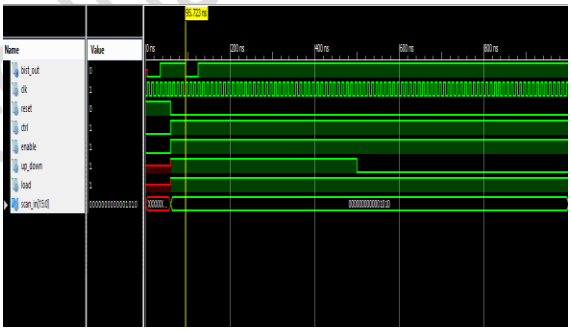


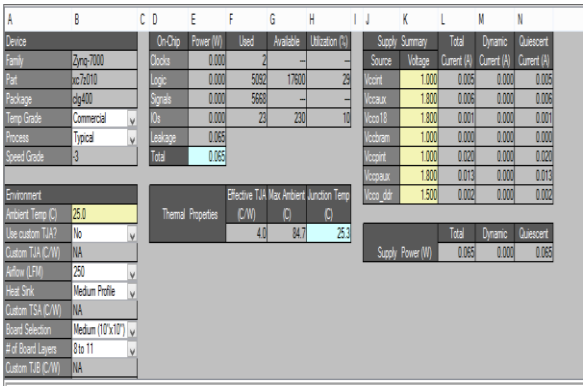
Fig. 8. Simulation results showing the AF of Conv-MBIST, as well as the AF and AO of our approach with up to 10 ST vectors for the considered benchmark circuits. (a) s38584. (b) s38417. (c) s13207. (d) s15850.

5. SIMULATION RESULTS

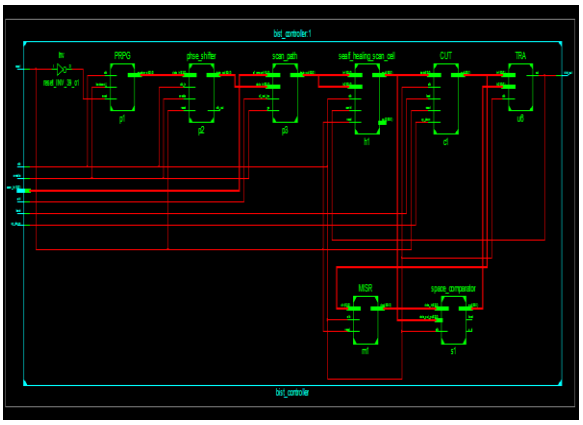
Proposed RESULTS



simulation



power



proposed schematic

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Registers	513	607200	0%
Number of Slice LUTs	12142	303600	3%
Number of fully used LUT-FF pairs	348	12307	2%
Number of bonded IOBs	23	700	3%
Number of BUFG/BUFGCTRL/BUFHCEs	2	200	1%
Number of DSP48E1s	1	2800	0%

Design summary

```
================================================================================
Timing constraint: Default OFFSET OUT AFTER for Clock 'clk'
  Total number of paths / destination ports: 1 / 1
--------------------------------------------------------------------------------
Offset:              0.511ns (Levels of Logic = 1)
  Source:            u6/out (FF)
  Destination:       bist_out (PAD)
  Source Clock:      clk rising

  Data Path: u6/out to bist_out
                              Gate     Net
    Cell:in->out      fanout  Delay    Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     FD:C->Q               1   0.232   0.279  u6/out (u6/out)
     OBUF:I->O                 0.000          bist_out_OBUF (bist_out)
    ----------------------------------------
    Total                      0.511ns (0.232ns logic, 0.279ns route)
                                       (45.4% logic, 54.6% route)

================================================================================
```

Time summary

6. CONCLUSION

In this paper we have presented memory BIST architecture and discussed about various blocks present in memory BIST. The controller algorithm for 'MARCH C-', 'MARCH A' and 'MARCH Y' and their Verilog implementation modelling and synthesis are compared. From the Algorithm description and synthesis results it is evident that 'MARCH A' algorithm is having more number of states and therefore controller implemented using 'MARCH A' occupies more area, consume more average power and still slower, while the controller implementation using 'MARCH Y' require least number of states and therefore it is occupies least area, consumes least power and yet bit faster. MARCH C-' covers additional faults as compared to the 'MARCH Y' algorithm at the cost of little increase in the area, average power consumption and delay.

REFERENCES

[1] J. Rajski, J. Tyszer, G. Mrugalski, and B. Nadeau-Dostie, "Test generator with preselected toggling for low power built-in self-test," in Proc. Eur. Test Symp., May 2012, pp. 1–6.

[2] Y. Sato, S. Wang, T. Kato, K. Miyase, and S. Kajihara, "Low power BIST for scan-shift and capture power," in Proc. IEEE 21st Asian Test Symp., Nov. 2012, pp. 173–178.

[3] E. K. Moghaddam, J. Rajski, M. Kassab, and S. M. Reddy, "At-speed scan test with low switching activity," in Proc. IEEE VLSI Test Symp., Apr. 2010, pp. 177–182.

[4] S. Balatsouka, V. Tenentes, X. Kavousianos, and K. Chakrabarty, "Defect aware X-filling for low-power scan testing," in Proc. Design, Autom. Test Eur. Conf. Exhibit., Mar. 2010, pp. 873–878.

[5] I. Polian, A. Czutro, S. Kundu, and B. Becker, "Power droop testing," IEEE Design Test Comput., vol. 24, no. 3, pp. 276–284, May/Jun. 2007.

[6] X. Wen et al., "On pinpoint capture power management in at-speed scan test generation," in Proc. IEEE Int. Test Conf., Nov. 2012, pp. 1–10.

[7] S. Kiamehr, F. Firouzi, and M. B. Tahoori, "A layout-aware X-filling approach for dynamic power supply noise reduction in at-speed scan testing," in Proc. IEEE Eur. Test Symp., May 2013, pp. 1–6.

[8] M. Nourani, M. Tehranipoor, and N. Ahmed, "Low-transition test pattern generation for BIST-based applications," IEEE Trans.

Vol 13, Issue 03, MARCH /2022

ISSN NO:0377-9254

Comput., vol. 57, no. 3, pp. 303–315, Mar. 2008.

[9] N. Z. Basturkmen, S. M. Reddy, and I. Pomeranz, "A low power pseudorandom BIST technique," in Proc. 8th IEEE Int. On-Line Test. Workshop, Jul. 2002, pp. 140–144.

[10] J. Rajski, N. Tamarapalli, and J. Tyszer, "Automated synthesis of large phase shifters for built-in self-test," in Proc. Int. Test Conf., Oct. 1998, pp. 1047–1056.

[11] P. Girard, L. Guiller, C. Landrault, S. Pravossoudovitch, and H. J. Wunderlich, "A modified clock scheme for a low power BIST test pattern generator," in Proc. IEEE VLSI Test Symp., Apr./May 2001, pp. 306–311.