VLSI Design
1993, Vol. 1, No. 1, pp. 23–44
Reprints available directly from the publisher
Photocopying permitted by license only

Built-In Self-Test: Milestones and Challenges

JACOB SAVIR
PAUL H. BARDELL
IBM, Data Systems Division, Poughkeepsie, New York, USA

This paper describes the progress in built-in self-test (BIST) since its inception, and the important problems that still need to be solved to make the technique widely acceptable. The paper includes a reference list and an extensive bibliography on the subject matter.

Key Words: *BIST; Test generation; Fault simulation; Signature analysis; Exhaustive test; Signal probability; Detection probability; Random test; Weighted random test*

The introduction of very large scale integrated circuits (VLSI) in electronic technology has brought about a tremendous testing problem. Test generation and fault simulation of VLSI packages are very costly and inefficient. Since the cost of test generation increases in a nonlinear fashion with respect to the size of the device [1], often this task is prematurely terminated yielding relatively low fault coverage figures. Moreover, since the quality of the end product is directly related to the amount of test performed on it [2], more and more defective parts would have been shipped due to insufficient test. It then became evident that new testing techniques are necessary to cope with the increasing complexity of VLSI.

The idea of having the circuit test itself has emerged as a possible solution to the VLSI test problem. The basic idea is to add some small amount of test hardware to the chip in order to facilitate a self-test. This solution requires additional controls to switch the circuit into test mode, and a mechanism to compress the test responses into a manageable size, so that its expected responses can be stored on chip for a pass/fail indication. Since the test is performed on chip, one would also have the added benefit of testing the device at its own speed. Another added benefit would be not having to rely on expensive testers that, besides their price tag, also occupy large amounts of floor space.

The earliest detailed study of BIST structure for random logic chips is described in Benowitz et al. [3]. The idea of BIST received a boost when Hewlett-Packard (HP) 5004A signature analyzer had been introduced [4]. Although not intended for BIST applications, HP's idea to compress test responses into a short *signature,* and use these signatures as test reference, had proven to be a cost effective way of performing a test with a relatively small amount of hardware. The signature collected by HP's 5004A signature analyzer was the contents of a linear feedback shift register (LFSR) that was built into the equipment, and whose width was 16 bits. Experience with HP's 5004A showed that the *aliasing probability,* namely, the probability of having a good signature indication for a faulty product, is quite low. It was also discovered during the use of the 5004A that in sequential products the signature measured was not necessarily unique. This phenomenon was due to nonuniform initial states that existed in the tested products. Today this situation is referred to by the generic term X-states (unknown states). Since it was impossible to initialize some products to a unique state, different (but correct) final test signatures were measured. Most test engineers were not ready to cope with this at the time, and some of them abandoned this new technique. This problem is being overcome today by either acknowledging multiple correct signatures, or by assuring that all sequential elements are initializable to the same state at the start of the test. Despite this early drawback, the

experience with HP 5004A led the way to the placement of signature analyzers on chip to compress test responses with little or no loss of test quality.

There are other data compression methods, besides signature analysis, that researchers have investigated. Methods like *one's counting* and *transition counting* have been discussed quite extensively in the literature [5–7]. One's counting captures the one's count at the output of the circuit and displays it as the test signature. Transition count captures the number of times the circuit output undergoes a 0 to 1, or a 1 to 0 change, and displays this count as the final test signature. Although some attempts have been made to implement these data compression techniques for BIST, they did not receive wide enough acceptance to become a popular technique, like signature analysis.

Another issue that divides BIST techniques is the test stimuli supplied to the logic under test. The most common test stimulus is the application of random patterns via an LFSR. This *random pattern BIST* is further divided into *exhaustive* and *nonexhaustive*. Exhaustive random pattern BIST applies all possible patterns to the circuit under test (CUT). Since an LFSR is incapable of generating the all-zero pattern, it may be necessary to modify it [8], to add this pattern to make its space fully exhaustive. Obviously, the applicability of exhaustive BIST is limited to CUTs with a relatively small number of inputs. Thus, testing CUTs with more than 30 primary inputs by an exhaustive BIST takes too long, even with the fastest technology in existence today. Nonexhaustive BIST only uses a small fraction of the entire input space as test patterns. Thus, the number of primary inputs in a CUT is usually not a limiting factor for nonexhaustive BIST.

Input patterns can also be generated by a counter. Counter-based exhaustive BIST is quite common in testing random access memories (RAMs). On the other hand, counter-generated nonexhaustive BIST for combinational logic is quite uncommon. The reason is that the most significant bit(s) of the counter will almost stay fixed during the nonexhaustive test, and, therefore, will fail to exercise many faults in the CUT.

This paper is primarily concerned with nonexhaustive random pattern BIST. It is assumed that the circuit has some kind of scan structure, such as level sensitive scan design (LSSD) [9]. Some comments on other BIST techniques are scattered throughout the paper. The paper covers numerous BIST areas: subjects like fundamental BIST structures, random pattern generators, signature registers, fault coverage analysis (probabilistic and simulation-based), estimation of the random pattern test length required to achieve a given test quality, the weighted random pattern approach to reduce the test length, logic modification to enhance random pattern testability, testing of random access memory (RAM)-based products, fault diagnosis in a BIST environment, AC test, and the self-test tester. A list of open problems in BIST and their importance is provided in a separate section. The paper concludes with a brief summary.

BASIC STRUCTURES

Figure 1 shows a schematic diagram of a BIST structure. The source in Figure 1 supplies random patterns to the CUT. The source also has a Start/Stop control line which feeds the data compressor and the comparator. The Start signal is issued with the generation of the first random pattern. This Start signal tells the data compressor to start compressing output responses. The compression will go on until a Stop signal is issued. The Stop signal is issued by the source after the application of the last random pattern. A Stop signal to the data compressor also constitutes a Ready signal for the comparator. This Ready signal informs the comparator that the signature stored in the data compressor is ready for comparison. For the scheme to work, a reference

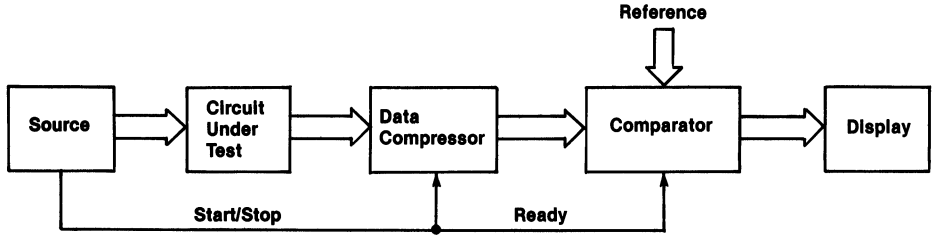

FIGURE 1 General BIST Structure.

signature must be supplied. This reference signature is the expected signature from the fault-free circuit, and is usually computed beforehand by performing a good machine simulation. At the comparison time, the measured signature is compared to the reference signature. The circuit passes the test if the signatures are identical.

BILBO

Many chip level BIST structures follow in principle, at least, one or more of the BILBO modes described in Koenemann et al. [10–12]. Figure 2 shows the Built-in Logic Block Observer (BILBO) register, which is a multifunction register.

The BILBO register constitutes a set of latches (for simplicity assume D-latches) where the inverted output of each latch (the $\overline{Q}$ output) is connected via a NOR and an EXCLUSIVE OR (XOR) gate to the data input of the succeeding latch. A multiplexer (MUX) allows either a serial input to the latches in one mode of operation ($B_1 = 0$), or a connection of the XOR feedback ($B_1 = 1$) in another mode of operation.

The BILBO register can be operated in a number of different modes. The mode of operation is determined by the value of the control inputs B_1 and B_2.

When $B_1 = B_2 = 1$, the BILBO acts as a regular register with parallel inputs Z_i, and parallel outputs $\{Q_i, \overline{Q}_i\}$ (not shown).

When $B_1 = B_2 = 0$, the BILBO acts as a shift register with scan input SI and scan output SO. In this mode of operation the XOR feedback is disconnected from the first latch by the multiplexer. This mode of operation may serve as a scan path if it is necessary to input external vectors to the device. Notice that the vector stored in the latches is the bitwise complement of the vector applied at the SI port.

When $B_1 = 1$ and $B_2 = 0$ the BILBO acts as a multiple input signature register (MISR), or LFSR. In this case the XOR feedback is active. If the Z_is are outputs of the circuit test, then the BILBO acts as a MISR compressing the circuit responses. If, however, $Z_i = 0$ for all i, then, provided there is a nonzero seed in the BILBO register, it acts as an LFSR generating pseudorandom stimuli. Notice that in this mode of operation the feedback function may be either an EXCLUSIVE OR or EQUIVALENCE (the complement of EXCLUSIVE OR) depending on the number of stages being fed back.

When $B_1 = 0$ and $B_2 = 1$ the BILBO register is reset to the all zero state.

STUMPS

BIST structure for higher levels of packaging requires a different test architecture. A scan path structure called Self-Test Using a MISR and a Parallel Shift Register Sequence Generator (STUMPS), is a built-in test architecture for an LSSD-based multiple chip system, in which each field replaceable unit (FRU) has a large number of logic chips [13, 14]. Consider the test of this replaceable unit. Each unit contains an added STUMPS test chip. The special test chip contains a parallel pseudorandom pattern source (a slight modification of an LFSR) called a parallel shift register sequence generator (SRSG),

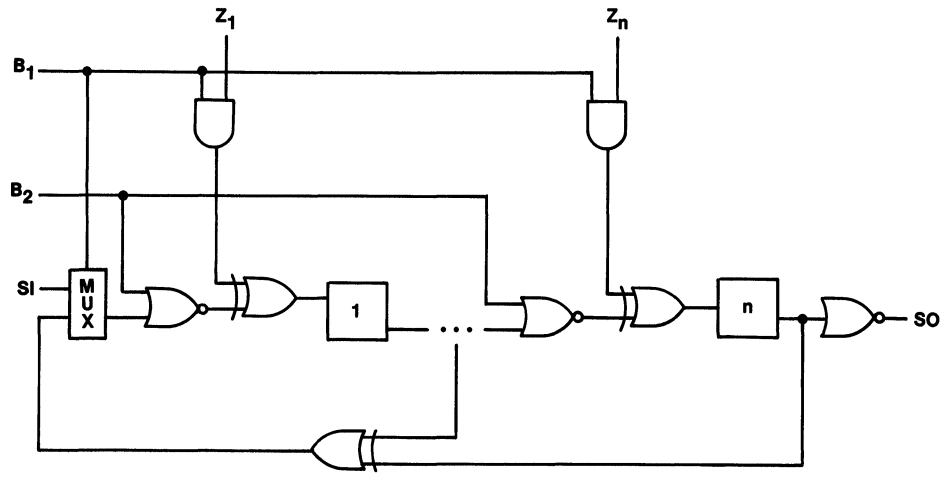

FIGURE 2 The BILBO Register.

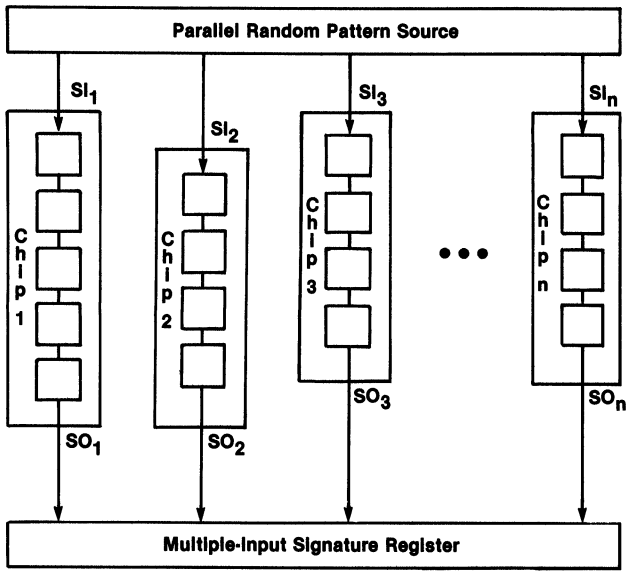

FIGURE 3 STUMPS Architecture.

and a MISR. The test chip is wired only into the scan path of the CUT, and thus can be overlayed on an already completed scan path design.

An illustration of the global structure of STUMPS is given in Figure 3.

Assume that the CUT has n logic chips, and that each logic chip has a scan path containing a number of shift register latches (SRLs). The A/B shifting clocks of the SRLs and the shift clocks of both the random pattern source and the MISR are tied together. In the built-in testing mode, the scan-in port (SI) of each of the n logic chips is connected to a stage of the SRSG, the pseudorandom pattern source, and the scan-out port (SO) of each logic chip is wired to an individual stage of the MISR.

Using the A/B shift clocks, the SRLs are loaded with pseudorandom patterns from the source. The number of A/B clock cycles required is equal to the number of SRLs in the longest string on any logic chip. This will cause the shorter shift register (SR) strings to overflow into the MISR, but will not affect the correctness of the final MISR signature.

After loading random patterns into the SRLs, the system clocks are cycled to capture the test results back into the SRLs. These results are then scanned out into the MISR, simultaneously loading the next random pattern set from the source. A test pass or fail indication is obtained after the last test by comparing the signature remaining in the MISR with the expected signature.

The number of SRLs on each logic chip of the CUT will vary and some logic chips may not have any. (A logic chip with no SRLs must contain pure combinational logic and is tested by stimuli provided by its surrounding chips.) There is a test time advantage to be had by structuring the parallel SRL "channels" so that they each contain roughly the same number of SRLs. This may require that several logic chips are connected in series in one or more channels.

There is a considerable amount of global wiring required to connect the test chip with each of the logic chips. For 100 logic chips on the CUT, there are 200 global wiring nets connecting the chips to the STUMPS test chip. This is a disadvantage only if the CUT package is severely limited in wiring capacity.

STUMPS requires only one additional I/O pin, the Test Mode signal. Since the test chip is inserted into the scan path of the CUT, there is no adverse effect on critical paths of the system.

TX1

Toshiba [15, 16] has reported on their success in manufacturing the TX1, a 32-bit microprocessor. The TX1 embeds three testable designs: scan test, self-test, and macroblock test. It uses scan test for the sequential logic that includes over 1660 flip flops. These 1660 flip flops are divided into 29 scan chains, the longest being of size 64. The TX1 uses self-test to detect faults in regular structures such as Read Only Memories (ROMs), RAMs, Programmable Logic Arrays (PLAs), and data flow blocks, like Adders, Arithmetic Logic Units (ALUs), etc. The self-test in TX1 is based on signature analysis. The macroblock test is a unique testable design specifically oriented towards design verification. The total area overhead used for the three testable designs is 4.6%.

A discussion of other BIST structures can be found in Bardell et al. and McCluskey [17, 18].

INPUT GENERATORS

Most BIST applications use pseudorandom input patterns. A *pseudorandom* input sequence is an "approximation" to a random sequence, and is generated by an LFSR. An n-stage LFSR should, preferably, be of *maximum length* to be able to generate all the $2^n - 1$ different patterns. Such an LFSR is said to implement a *primitive polynomial* [19]. To define an LFSR one only needs to specify which stages feed back to the first stage via an XOR circuit.

Some feedback connections may produce maximal length sequences when seeded with a nonzero initial pattern, and some may not. A list of primitive polynomials that are desirable in BIST applications may be found in Bardell et al., and Peterson and Weldon [17, 20].

There are some potential problems associated with the use of LFSRs as pseudorandom pattern generators (PRPGs). It is important that the test engineer be aware of them so that he can avoid unnecessary low fault coverages when he actually builds his BIST structure. Some of these properties are discussed below.

When an LFSR is used to feed pseudorandom patterns into scan strings (as in STUMPS) some SRLs in the scan chain may visit all possible patterns, while others may see only a shortened sequence. This phenomenon happens when the shift advance into the scan is not relatively prime to the sequence length. This phenomenon is called *decimation*. In STUMPS operation a new test is applied only after all scan chains are filled with pseudorandom bits. This shift advance may not be relatively prime to the generator sequence length. If this happens, some patterns will be excluded from the scan chains, adversely affecting the final fault coverage. To avoid loss of fault coverage due to decimation one has to choose a generator and scan chain lengths so that the shift advance needed to fill them up will be relatively prime to the generator's sequence length.

Another inherent property of PRPGs is the existence of *linear dependencies*. Linear dependencies occur between selected bits of a scan chain that is fed serially from a given stage of an LFSR (as in STUMPS). To see this consider the LFSR and the scan chain of Figure 4.

Figure 4 shows a four-stage LFSR that feeds a five-stage scan chain. The LFSR is of maximal length; namely, if seeded with a nonzero pattern, and continuously clocked, it will generate all 15 possible patterns. We denote the sequence of bits entering the scan chain by $\{a_n\}$. From the feedback connections of the LFSR it is evident that the sequence $\{a_n\}$ has the bit-dependency

$$a_n = a_{n-4} \oplus a_{n-1}.$$

As a result of this dependency the values entering the scan string are dependent. In particular, stages a_1, a_4, a_5 are related according to

$$a_5 = a_1 \oplus a_4$$

This linear dependency will prevent the bits in stages a_1, a_4, a_5 from visiting the state $a_1 = 1$, $a_4 = 0$, $a_5 = 0$. If, for example, stages a_1, a_4, a_5 were to feed a 3-input OR gate, none of the stuck-at-0 faults at the inputs of the OR gate will be detected during test. Moreover, the number of different patterns that can appear at stages a_1, a_4, a_5 is only 4 (compare this to a total of 8 if the generated sequence were truly random).

Fault coverage degradation due to linear dependencies can only occur when the span of the circuit connected to the scan string exceeds the width of the LFSR generating the pseudorandom patterns [17]. The *span* of a circuit fed from a scan string is defined as the distance (measured in scan string stages) between the two circuit inputs that are connected farthest apart on the chain. In the previous example, the OR gate was fed from stages 1, 4 and 5 in the scan chain. The span of this OR gate is 5, which exceeds the LFSR width, which is 4.

Linear dependencies cannot be eliminated. They will always exist as long as the generated patterns are not truly random (which, unfortunately, is always the case). The best one can hope for is to make intelligent choices in terms of circuit connections in order to minimize their effect.

Another source of concern to scan-based BIST is the existence of *structural dependencies*. When adjacent scan paths are fed from contiguous LFSR stages (as in STUMPS), the same values appear in the adjacent scan paths offset by one shift [17, 21, 22]. In particular, if a 2×2 window is placed anywhere on the array of latches formed by the scan paths, only one half of the possible 2×2 binary arrays will appear as the LFSR is shifted through its cycle. Larger windows have even smaller coverages. As in the case of linear dependencies, this a priori exclusion of array bits from appearing during testing can adversely affect the quality of the shipped product.

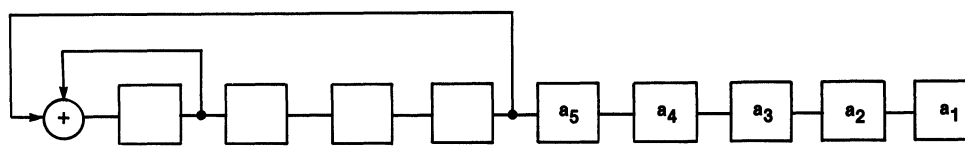

FIGURE 4 Illustration of Linear Dependencies.

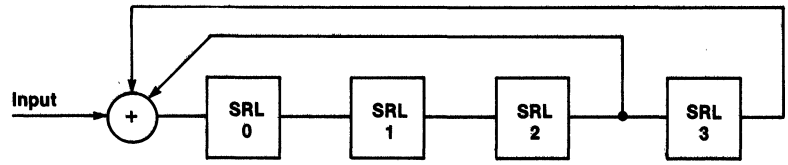

FIGURE 5 An LFSR as a Single Input Signature Register.

One way of reducing the effect of structural dependencies is to insert a phaseshift network between the LFSR and the circuit being driven (for example, between the SRSG and the scan chains in STUMPS) [17, 22]. The phaseshift network is made out of XOR gates, and it produces pseudorandom sequences that appear to be shifted in time by more than one bit. As a matter of fact, large phaseshifts may be accomplished between consecutive outputs of the phaseshift network. These arbitrary large phaseshifts between sequences entering the scan chain will increase the number of different bit sequences that may occur in a given window, and could, therefore, perform better during test.

A new type of generator has received some interest recently. This generator, called *cellular automata register* (CAR), is a cascade of special finite state machines, where the next state of a particular storage cell is determined by a linear combination of the present states of the cell and its two adjacent neighboring cells [23–25]. Two linear combinations have been found to be useful for CARs in BIST applications. The first linear combination (called rule 90) is where the next state of a cell is a XOR of the present states of its two neighbors. The second linear combination (called rule 150) is where the next state of a cell is a XOR of its own present state, and the present state of its two adjacent neighbors.

Wolfram [24, 25] has shown that CARs formed from cells with a single rule generally exhibit a sequence length much shorter than the maximum possible. Pries et al. [26] have noticed that some cascades of rule 90 and rule 150 will exhibit maximum length sequences. Serra et al. [27] have pointed out that CARs are isomorphic to LFSRs. Bardell [23] has analyzed the linear dependency problem in CAR-based BIST and has found it to be similar to those existing in LFSRs. On the other hand, structural dependencies seem to be different in CARs, although they still do exist.

SIGNATURE REGISTERS

Signature registers are used to collect the final signature after test. They, generally, perform a cyclic code compression on the incoming bit stream [17, 19]. The LFSR of Figure 5 can serve as a single input signature register (SISR).

The external input in Figure 5 is fed from a single output circuit (not shown). The LFSR is continuously clocked to capture the circuit responses to the input patterns. At the end of the test the contents of the LFSR is read out as the circuit test signature. Notice that in the LFSR implementation of Figure 5, the XOR gate feeds only the first stage.

A different implementation of a SISR is possible, where the feedback signal is distributed to a number of XOR gates that exist between stages. Such an implementation is shown in Figure 6.

The SISR in Figure 6 has, in principle, the same test characteristics as that of Figure 5. That is to say, provided the two implement the same primitive polynomial, they will have similar aliasing characteristics. One obvious difference between the two implementations is that the SISR of Figure 5 may be faster than that of Figure 6 because it does not have interstage XOR gates. On the other hand, the SISR of Figure 6 only has 2-input XOR gates, while the SISR of Figure 5 needs, in principle, a multiple-input XOR gate. In case a multiple-input XOR gate is not available, the XOR gate of Figure 5 needs to be implemented by a tree of lower fan-in XOR gates that may slow down its performance.

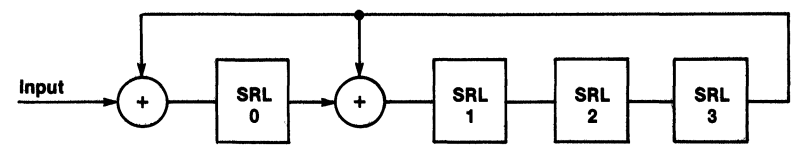

FIGURE 6 An Alternative Implementation of a Single Input Signature Register.

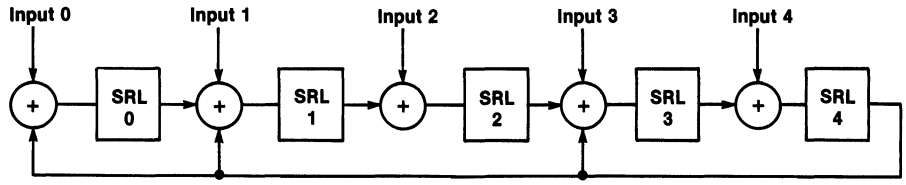

FIGURE 7 A Multiple Input Signature Register.

For multiple output circuits a MISR is used to collect the test signature. Usually a MISR has as many stages as there are circuit outputs. An XOR gate is placed between each of the MISR stages, where one of its inputs is driven from one of the circuit outputs. The feedback function also feeds the interstage XORs to implement the desired polynomial (see Figure 7).

The problem of computing the aliasing probability in SISRs and MISRs has been considered by many researchers [4, 28, 29, 30]. Different assumptions were used by different researchers regarding the behavior of errors entering the signature registers. Regardless of the assumptions made, most researchers agree that in most cases the aliasing probability converges to 2^{-m} as the test proceeds, where m is the number of stages in the register. Thus, for all practical purposes the aliasing probability can be kept under control, provided the signature register width is at least 16.

MISRs exhibit a different kind of masking, called *error cancellation*. This type of aliasing is unique to MISRs, and does not appear in SISRs. Error cancellation in MISRs happens when two errors occur on two adjacent input streams such that an error appears at input channel i, as a response to test t, and at input channel $i + 1$, as a response to test $t + 1$. In this case the first error will be captured in stage i at time t, and it will then be shifted to stage $i + 1$ at time $t + 1$, cancelling the new error entering this channel at that point in time. This type of aliasing is independent of the feedback polynomial, and it can occur at many different combinations of input streams and test cycles. For instance, two error bits distanced 2 cycles in time, entering two stages distanced two away from each other, will, also, cancel out. Error cancellation, however, is much less likely than the primary aliasing discussed earlier. If T is the number of test patterns applied to the logic, and if m represents the width of the MISR, the probability of encountering an error cancellation is approximately 2^{1-m-T} [17].

Counting techniques also exhibit aliasing. The aliasing probability in both one's counting and transition counting have the same asymptotic behavior. If T is the test length, the aliasing probability asymptotically approaches $1/\sqrt{\pi T}$ [31]. Thus, for test lengths that exceed 1 million patterns, this aliasing probability is also negligible.

The problem of X-states with regard to HP 5004A has been referred to earlier. This same problem will exist in BIST designs unless special precautions are taken to avoid it. In custom-made products the easiest solution is to avoid the X-states altogether by providing a unique initial state. When a product includes vendor chips with uninitializable states it is important to recognize all the possible good signatures that might occur during test.

FAULT COVERAGE ANALYSIS: PROBABILISTIC APPROACH

Probabilistic methods of estimating the fault coverage of a pseudorandom test exploit the circuit topology. By using the circuit topology it is possible to compute the detection probability of an arbitrary fault. The *detection probability* of a fault in a combinational circuit is defined as the probability that a randomly selected input will detect the fault in question. Thus, given the circuit structure, it is possible to compute the detection probabilities of all faults of interest (single stuck-at faults, for example). Once the list of fault detection probabilities is generated, it is possible to estimate how many of the faults in question will be detected by a given random pattern test length. This estimate allows us to make statements such as "BIST having a test length T will detect q percent of the faults with c percent confidence." For example, a statement such as "a pseudorandom test of length 100,000 will detect 92% of all single stuck-at faults with 98% confidence," means that if somebody were to repeatedly test the logic with 100,000 pseudorandom patterns (each time with a different set of 100,000 pseudorandom patterns), and record the fault coverage of each set, then at least 98% of the time this fault coverage will reach a figure of at least 92%.

Exact computation of probabilistic expressions in logic was proposed by Parker and McCluskey (PM)

[32]. The main thrust of their work was to compute signal probabilities in a combinational network. A *signal probability* of a line in a combinational circuit is defined as the probability that this line will assume a value 1 in response to a random input. Savir et al. [33, 34] have shown how to compute detection probabilities of faults by signal probability manipulations. The computation of exact detection probability of a fault, according to PM, requires analyzing a structure whose size is a little more than twice the size of the original circuit. Basically, the circuit structure is duplicated where one copy is the fault-free version of the circuit, and the other copy is the circuit with the fault in question being injected. An additional transformation circuit is added to the outputs of the above two copies to yield a value 1 whenever the corresponding outputs of the two copies differ. Thus, the signal probability at the output of the transformation circuit is equal to the detection probability of the fault in question. This analysis needs to be repeated for every fault. Details of the method appear in Bardell et al. [17].

Computation of exact detection probabilities is known to be a very hard problem. The PM algorithm described earlier, for example, may have an exponential complexity in the worst case. It was necessary, therefore, to come up with less complex algorithms to cope with the ever-increasing densities of VLSIs. This could not have been achieved without sacrificing the accuracy of the results.

Savir et al. [33, 34] have proposed to compute lower bounds on detection probabilities, rather than exact values. The computational complexity of those lower bounds will, then, drop from exponential to polynomial. The question is "are lower bounds of detection probabilities sufficient to assess the testability of a BIST design?" The answer to that is yes, provided a conservative approach to test length is acceptable. The conservativeness results from using lower bounds of detection probabilities to compute a test length that will detect the faults in question with a given degree of confidence. The use of lower bounds of detection probabilities, in lieu of exact values, will increase the necessary test length to detect the target faults. The use of this longer test length in BIST will yield a better quality test than what was initially intended. For example, if in reality a pseudorandom test of length 1 million will detect 98% of the faults, and if the conservative test length happened to be 4 million tests, then by actually applying 4 million tests the fault coverage that is actually achieved may be in excess of 99%, which is better than the 98% it was targeted for.

Savir et al. [33, 34] have proposed the cutting algorithm (CA) (later to be improved by Savir [35] and Markowsky [36]) as a means to compute lower bounds of detection probabilities. The essence of the CA is cutting sufficient number of fanout branches to turn the circuit into a tree, where detection probability computation is a relatively simple task. This so-called "cutting" is only a software modification of the circuit topology used solely for the purpose of computing the detection probabilities. By performing cuts on the circuit, exact signal probabilities need to be replaced by bounds. These bounds are a by-product of the cutting, and are necessary in order to preserve the correctness of the final results. The end result of using the CA is that a list of lower bounds of fault detection probabilities is computed. This list of lower bounds are then used to compute the desired test length necessary to meet a given quality objective (see section on test length calculation).

Other probabilistic algorithms for estimating fault detection probabilities have also been proposed. Statistical fault analysis (STAFAN) [37] uses controllability information, as well as statistics of sensitization frequency of lines, to deduce fault detection probabilities. Controllability observability program (COP) [38] also uses controllability and observability calculations to estimate fault detection probabilities. COP completely ignores signal dependencies in the circuit during computation. Probabilistic estimation of digital circuit testability (PREDICT) [39] is in principle identical to STAFAN. The difference between PREDICT and STAFAN lies in the way the basic data is collected. While STAFAN collects its data from simulation runs, PREDICT generates them analytically. All three algorithms (STAFAN, COP and PREDICT) do not generate "one-sided" results; namely, none of these algorithms will consistently compute a lower bound (upper bound) of fault detection probabilities. These algorithms produce results that include cases where the probabilities are underestimated, as well as cases where the probabilities are overestimated. Thus, these estimates cannot be used to compute a conservative test length as shown earlier with the CA. Results from STAFAN, COP, and PREDICT can only be used as crude estimates.

FAULT COVERAGE ANALYSIS: SIMULATION APPROACH

The detection probabilities of faults in the circuit can be estimated by fault simulating a large number of randomly generated vectors. Let f be a fault in the

circuit whose detection probability is to be estimated. Let T be the number of random patterns that are used to estimate the detection probability of f. For each random pattern the logic is simulated in order to determine whether or not the fault is detected. Let N be the number of times the fault f has been detected, $N \leq T$. The detection probability, q, of the fault f can be estimated to be

$$q = \frac{N}{T}$$

This estimate, however, includes a random error that will approach zero as T approaches infinity. For large T the sample mean given in the above equation is approximately normally distributed. The 95% confidence interval for this point estimate lies between $q - 2e$ and $q + 2e$, where

$$e = \sqrt{\frac{N}{T^2}\left(1 - \frac{N}{T}\right)}$$

Fault simulation to determine random pattern testability is an expensive process in most cases. To see this let τ be the simulation time needed to propagate values from an input to a gate to the output of the same gate. Let G be the number of gates in the circuit, and let T be the number of random patterns that have to be simulated. Let ω be the number of faults whose testability has to be determined. Using this notation the time it takes to simulate one random pattern on the entire circuit is $\alpha\tau G$, where the factor α depends on the simulation method and indicates what fraction of the logic is simulated, on the average, in order to determine whether or not a given vector detects a given fault. If for every new pattern the entire logic is simulated $\alpha = 1$. On the other hand, if the method of simulation is such that only the gates which experience a change in the values applied to their inputs (due to the propagation of errors from the fault site) are simulated, then $\alpha < 1$. The coefficient α is called the *simulation reduction factor,* and it may range between 0.5 and 0.9 depending on the number of levels in the circuit, and on whether or not the logic has disjoint partitions. The faults of interest are usually all the single stuck-at faults and, therefore, $\omega = k_1 G$. If no fault reduction is being made, then k_1 is approximately twice the average number of the gate fan-in plus fan-out. With fault reduction this coefficient may range between 1 and 3. Notice that if the fault reduction includes a process of identifying fault equivalence classes the coefficient k_1 can even be less than 1. This process of fully identifying the fault equivalence classes is, however, very expensive, and it is hardly worth it in an automatic environment. The number of patterns, T, is usually proportional to the number of gates in the circuit, and in many cases is quite high because it takes many random patterns to reach an acceptable fault coverage. Thus $T = k_2 G$, where k_2 is this coefficient of proportionality. It is important to note that a reduction in simulation time may be achieved if more than one pattern is simulated against a given fault for each simulation pass. This is possible since a 32-bit machine can conceptually simulate 32 patterns in parallel. It is also possible to take advantage of the extended length attribute that may exist in some instruction sets (IBM 370 instruction set supports length attributes of up to 256 bytes). If this is done it will decrease the coefficient k_2 significantly [40].

Notice that it is much more advantageous to perform the fault simulation so that many patterns are simulated in parallel against a single fault in each pass, rather than performing it on many faults in parallel and a single pattern per pass. The reason for that is that when many patterns are simulated in parallel against a single fault, it is possible to take advantage of the fact that the fault affects a relatively small number of gates yielding a relatively small α. This localized effect of the fault can be utilized in the simulation process resulting in a lower CPU time. If, on the other hand, the simulator simulates many faults in parallel for each pattern, a great portion of the network has to be simulated each simulation pass since the faults are generally spread all over the network. The method of simulating multiple patterns against a single fault is called parallel pattern single fault propagation (PPSFP).

Combining all the above arguments together yields a formula for the running time, R_t:

$$R_t = \alpha k_1 k_2 G^3 \tau = KG^3$$

The constant $K = \alpha k_1 k_2 \tau$.

TEST LENGTH CALCULATION

One of the key parameters that need to be decided for BIST is the pseudorandom test length. This parameter directly affects the quality of the test.

When simulation techniques are used the fault coverage achieved by the test is known exactly. Maintaining this fault coverage in practice will require running the test with exactly the same patterns as

used during simulation. If the patterns are generated by an LFSR, for example, this will require performing the test with the same initial seed as the one chosen during simulation. Similarly, the same signature analyzer needs to be placed in the BIST structure as the one used during the simulation runs. Any discrepancy between the simulation parameters and BIST parameters will render the fault coverage evaluated by simulation only an approximation.

When analytical techniques are used to determine the testability of BIST no definite fault coverage figures can be asserted. Fault coverages, in this case, can only be stated statistically, as was described in the previous sections. Statistical coverages, however, are not so sensitive to test parameters. Statistical fault coverages are, for instance, independent of the actual patterns used during test, provided they are generated pseudorandomly. Similarly, fault coverage figures are relatively insensitive to LFSR initial seed, or MISR primitive polynomial. The most important factor influencing the quality of the test, in this case, is its length. It is, therefore, very important to be able to quantify the value of the test length in order to statistically guarantee the quality of the test.

Let $q_{\min}$ be the smallest detection probability of any fault in the circuit. Let s be the number of faults in the circuit whose detection probability lies between $q_{\min}$ and $2q_{\min}$. Let c be the confidence level required for the test. Given these parameters, the BIST test length necessary to achieve 100% fault coverage is given by

$$T \simeq \frac{\ln\left(\frac{1-c}{s}\right)}{\ln(1-q_{\min})}$$

If, for example, a circuit has a detection probability profile, such that $q_{\min} = 10^{-5}$, and $s = 100$, the pseudorandom test length necessary to achieve 100% fault coverage with 99% confidence is $T \simeq 9.2 \times 10^5$.

For cases where only lower bounds on detection probabilities are known, the same formula may be used to determine the conservative test length. The main factor determining how conservative the test length is going to be is the extent to which $q_{\min}$ has been underestimated. An underestimation of $q_{\min}$ by a factor of 10 will approximately drive the test length higher by a factor of 2.3.

Details about test length calculations can be found in Bardell et al., and Savir and Bardell [17, 41, 42]. Some interesting comments on the difference between test length results obtained under the assumption that the test patterns are truly random, and those obtained under the assumption that the test patterns are only pseudorandom may be found in Wagner et al. [43].

WEIGHTED RANDOM PATTERNS

The flat (unbiased) random pattern test length can be very large in a real design. One of the methods to reduce this large test length is to use *weighted random patterns,* where the input vectors are not equally likely. To see the merit of this approach, consider a 6-input AND gate. If all inputs are equally likely, then the probability of detecting a stuck-at 0 fault on one of the input lines is $2^{-6} \simeq 0.015$. If the patterns, however, are generated with a signal probability 0.9, then this detection probability becomes $0.9^6 \simeq 0.53$. It is evident that, in this case, biasing the input signal probabilities so that more 1s are generated than 0s reduces the length of the test.

Weighted pseudorandom patterns can be generated by nonlinear operations combining outputs from various stages of an LFSR. Notice that the probability that two statistically independent signals are both at a value 1 is the product of the probabilities of each being at 1. Thus, when two such signals are fed from an LFSR into an AND gate, the signal probability at the output of the gate is 0.25. A circuit operating on this principle is shown in Figure 8.

The weight, or signal probability, of the output can be controlled by the word in the control register of Figure 8. The AND gate whose output is labelled 2 has inputs from two stages of the LFSR. Consequently, when position 3 of the control register contains a 1, AND gate 2 will feed a 1 to the OR gate 1/4 of the time. When position 3 of the control register contained a 0, AND gate 2 is disabled and contributes nothing to the signal probability of the output. The outputs of the three AND gates, which contribute 0.5, 0.25, or 0.125, respectively, are selected by the value of the control register. Combinations of these weights can also be selected by having control words with more than a single 1 in them. The XOR gate driven by position 1 of the control register serves as an inverter, allowing the output to display weights larger than 0.5 (generating more 1s than 0s).

A different circuit that can generate weighted random patterns was proposed in Gloster [44]. A latch specifically designed to accommodate weighted random patterns was proposed in McAnney and Savir [45].

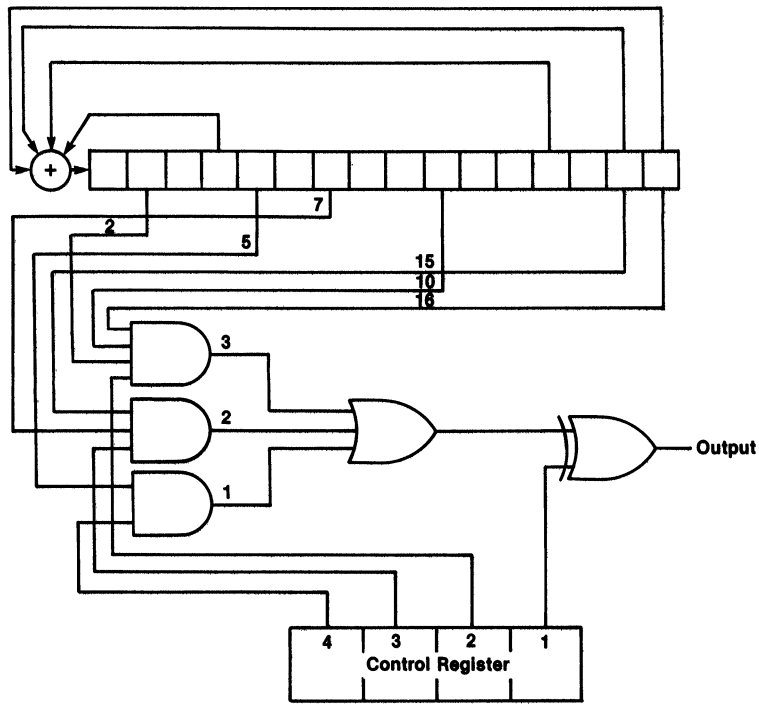

FIGURE 8 Programmable Weighted Random Pattern Generator.

The question as to how to bias the input signal probabilities so that the random test length will be optimally reduced was considered in Bardell et al., and Waicukauski and Lindbloom [17, 46]. These algorithms compute a nearly optimal global set of weights that will minimize the necessary test length. Both algorithms add few additional sets of weights targeted at detecting the faults left over after using the global set of weights. These additional sets of weights are computed by first generating tests for those uncovered faults, and then choosing the weights so that these tests will be relatively easily generated by the weighting scheme. An interesting way of computing weights based upon clustering of test patterns was described in Gloster [44].

LOGIC MODIFICATION

A different approach to enhance the random pattern testability of logic is to modify the logic to upgrade the detection probabilities of the hard-to-detect faults.

Since logic modification to enhance random pattern testability may adversely affect the circuit performance it is bound to face resistance from the logic design community. Thus it is of utmost importance to have a logic modification algorithm that has little or no effect on the circuit speed. The algorithm must be such that it will not require the designer's intervention. It should be implementable on a computer as a design-aid tool, and should have the capability of suggesting logic modification actions that will enhance the detection probability of the hard-to-detect faults. The user of such a design-aid tool would then be able to weigh one possible logic modification proposal against another and make a decision as to which approach is better.

Controllability and observability are important factors in enhancing the random pattern testability of a design. Although, in general, designs which have high values of controllability and observability tend also to have high degrees of testability, this is not always the case [47].

It is possible to enhance the observability of a line by inserting test points in the logic. The simplest way of doing this is by adding an extra output pin for every test point. This solution is quite costly since pins are very often scarce. If LSSD is used in the circuit, one can make the test points observable at a shift register latch (SRL). In this way the need for extra pins is avoided, and is traded off by using extra hardware and longer shift register chains. If the number of test points needed to enhance the observability of a design is small this solution is generally accept-

able (provided you have a control over the circuit implementation). If, however, the number of test points is large, it is possible to connect the test points to a XOR tree whose output feeds an SRL. In this way one SRL is shared by many test points. The expense of this approach is, therefore, the addition of the XOR tree. The XOR tree has the property that any odd number of faults propagated through it will yield an output error. It will, however, mask the appearance of an even number of faults. Thus, it may be advisable to have a number of XOR trees with independent signals feeding each one of them. This requires clustering independent test points in one group and connecting them to a XOR tree.

Controllability enhancement may be achieved in a number of ways. One can add control inputs to already existing gates. One can add additional gates, not existing in the original design, controlled by additional inputs. One can, also, form wired logic connections with additional inputs whenever the technology allows. Usually controllability enhancement operations affect the circuit speed more than observability enhancement operations. It is, therefore, necessary to weigh the potential benefits that such an operation may offer against the potential loss in circuit performance.

There are few observations to be made regarding the effects of logic modification. We describe some of them here.

The first observation is that observability enhancement is usually a benign operation. No degradation of the detection probability of faults will occur as a result of enhancing observability. Thus, the observability enhancement operation may increase the observability of some lines in the circuit and, at the same time, it will not decrease the detection probability of the rest of the faults for which this operation was not intended.

This is certainly not true when controllability enhancement is employed. An insertion of an extra input to enhance the controllability of a line may decrease the detection probability of a fault by a factor of 2 [17]. In other words, controllability enhancement operation is not, in general, a benign operation. The reason for that is that when an additional line is introduced, there may exist a fault in the logic whose detection depends upon this newly inserted line being at a specific value. This new requirement exacerbates the detection probability of the fault. It is possible, therefore, to perform a controllability enhancement operation making some faults easier to detect, while at the same time worsening the detection probability of others.

More on logic modification may be found in Bardell et al., and Iyengar and Brand [17, 48].

RAM TEST

BIST of a RAM-based product requires some departure from random patterns. The reason for this departure is due to the need to test for possible faults in all addresses. An effective way of guaranteeing the visit of all possible addresses during test is stepping through the address space with a counter. The alternative of using random patterns to step through the address space is very expensive in terms of test length. Thus, the address space is covered either by a counter, or by an LFSR that is modified to generate the all zero pattern as well. The data ports, on the other hand, can still be stimulated pseudorandomly. The same holds true for the read/write control.

Stand-alone RAMs (namely RAMs with no logic surrounding them) can be tested quite effectively against stuck-at faults. By walking through the address space twenty times, and writing random data followed by a read of the data, all single stuck-at faults can be detected with 99% confidence [49]. During the read operation the data is collected in a MISR. The final MISR signature, after twenty such cycles, is used for a pass/fail screening of the RAM.

Testing faults in embedded RAMs and its surrounding logic is by far more complicated than testing stand-alone RAMs. Some additional hardware may be necessary to aid the testing of such a product. This additional hardware may consist of boundary scan SRLs at both the inputs and the outputs of the embedded RAM. These boundary SRLs may partition the testing problem into two separate tasks: the testing of the RAMs themselves, and the testing of the surrounding logic. When the additional hardware to facilitate RAM test is too costly, one may perform the test by initializing the embedded RAM with random data, and then testing the surrounding logic by applying pseudorandom patterns at the primary inputs, while at the same time performing a read from a random RAM address. The test may continue in this mode for a prespecified amount of time before a new set of random data is written into the embedded RAM. After this is done, the test is repeated again. To make this approach effective, it may be necessary to perform the test with many sets of random data in the RAM. Notice that during the logic test no write operations are performed on the embedded RAM.

An important failure mode in RAMs is the so-called *coupled-cell faults*. These coupled-cell faults are pattern-sensitive, and usually involve more than one cell.

An example of such a coupled-cell fault is one where a 0 to 1 change in cell i forces cell j to undergo a transition from 0 to 1 as well. For this fault to be active cell j must be in a 0 state, and cell i must be written from a 0 to a 1. If all these conditions exist, cell j will flip from a 0 to 1. Obviously, detection of coupled-cell faults in RAMs is much more difficult than stuck-at faults. Testing these faults effectively requires many more random read/write cycles through the address space. Testing the above coupled-cell fault, for example, requires at least 90 such read/write cycles through the address space [50].

Analysis of the detection of faults in embedded RAMs and its surrounding logic was considered in Carter et al. [51]. Fault propagation through embedded RAMs was treated in McAnney et al. and Savir et al. [52–54].

FAULT DIAGNOSIS

In many cases diagnosis to a repair action is needed. In the introduction of a new part into production, the problem of zero yield at first silicon requires the ability to diagnose faults within the VLSI chip to a precision of a few circuits. By isolating the failure to a very small area the trouble spot can be identified and rectified. At higher levels of packaging (card, module, subsystem) diagnosis is necessary to identify a failing FRU. Usually a unit that is removed in the field is sent to a repair depot. At this stage the FRU is tested to determine the exact cause of the failure.

Unless careful thought is given to diagnostics, BIST products may be a hard nut to crack. The reason for that is that the only information available to the test engineer is a failing final signature. Working backwards from a final faulty signature to identify the failing component may be quite difficult. One possible solution to the problem is to create ahead of time a dictionary of all possible faults and their faulty signatures. Besides being a tremendously difficult task, it may also require a huge storage to hold this dictionary.

A more manageable approach to BIST diagnostics is to partition the entire test into a number of *test windows,* and compute the good machine signature for every such window. The cost of this approach is the addition of multiple window signatures. The diagnostics, however, can be greatly simplified in the presence of multiple test windows. The test is performed as it would normally be, stopping at the end of each window to compare the signature at hand with its fault-free version. If the two match, the test proceeds to the next window. Once the window final signature does not match the precomputed one, the test rolls back to the beginning of that window repeating the test, pattern by pattern, till the failing pattern is uncovered. From here on traditional diagnosis practices are being used to isolate the faulty component. If the test window is relatively small, it is possible to keep the circuit responses to the last test window in hardware, thus avoiding rerunning the test when a failing signature is observed. This additional storage should be designed to operate in scan mode, so that whenever the test proceeds to the next window, the new test responses overlay those of the previous one.

As in traditional fault diagnosis, there is a need to compute the smallest possible set of faults that might explain a given faulty response. The goodness of any diagnostics scheme may be measured by how effectively it can narrow down the list of potential causes of the failure. Ideally we would like the diagnostics scheme to always point to a single cause. Unfortunately, this is not possible because many faults may induce identical faulty signatures. This is the *fault-signature equivalence class*. Isolating the actual cause of the observed failure beyond its fault-signature equivalence class may require a rerun of the test to collect partial signatures at some selected points. This may be impossible because, in many cases, the package is sealed and one cannot probe into it.

Some theoretical work was done as to how to ease the diagnostic burden in BIST. Two such diagnostic aids were proposed in McAnney and Savir, and Savir and McAnney [55, 56]. These diagnostic aids will, under certain assumptions, compute the failing test from its failing signature. Thus, the use of these diagnostic aids may supplement the multiple window signatures method to identify the failing test.

AC TEST

Ascertaining correct operation of digital logic circuits requires verification of functional behavior as well as correct operation at desired clock rates. Failures causing logic circuits to malfunction at desired clock rates are called *delay faults,* or *AC faults*. These delay faults are normally due to random variation in pro-

cess parameters that often cause propagation delays to exceed their limits. Detection of an AC fault normally requires the application of a two-pattern test: the first pattern applies an initialization value at the site of the suspected fault, and the second pattern provokes the transition at the site of the fault and propagates its effect to a primary output or latch. The two-pattern test will provoke a 0 to 1 transition at the site of the fault to test for a slow-to-rise, and a 1 to 0 transition to test for a slow-to-fall fault. By measuring the output at the desired time interval one can ascertain whether or not a delay fault exists in the circuit.

Two models have been proposed for AC faults: the gate delay fault [40, 57–59], and the path delay fault [60–64]. According to the gate delay fault model any input to a gate can be subjected to a delay fault that may cause the response of the gate to be slow compared to its specification. The path delay fault model focuses on the aggregate delay along the path and not on the individual delays of the gates comprising the path. Thus, according to the path delay fault model, a path has an AC fault if the time it takes to propagate a signal along the path exceeds some specified value. Note that according to the path delay fault model, a path may be AC fault-free while one or more of the gates along the path may exceed their specified propagation delay. This phenomenon may happen since the response of slow gates may be compensated by the response of faster gates along the path, so that the total input to output delay happens to be within an acceptable range.

There are three kinds of AC tests: hazard-free, robust and nonrobust tests. A hazard is created at an output of a gate when two or more inputs change values simultaneously, and the change in one input has a reverse polarity compared to the change in the other input. Consider a two-input AND gate which undergoes a change from 01 to 10. If the time it takes the gate to switch is zero, then the output of the AND gate should be 0 both before and after the input change. Since no device can switch in zero time, it is possible that the 0-to-1 change occurs faster than the 1-to-0 change. In that case, the AND gate will experience a momentary output of 1 before it changes back to 0. This spike at the output of the gate is called a *hazard*.

The three types of AC tests mentioned earlier (hazard-free, robust, and nonrobust) differ on the basic requirements needed to detect the AC fault. Hazard-free test imposes the most severe detection requirement, while robust and nonrobust relax some of these conditions. A description of these three types of AC tests can be found in Pramanick and Reddy [60, 61].

Random patterns are very often used to detect AC faults. Effectiveness of random patterns to delay testing has been discussed by Savir and McAnney [62, 63]. The authors compute upper bounds on the test length needed to detect delay faults. In this analysis credit is only taken when the delay fault is detected by a hazard-free pair. Since both the generation of robust and nonrobust test pairs is possible during the random test the computed figure only constitutes an upper bound according to these other two test strategies.

Lin and Reddy [65, 66] estimate the probability of detecting a path delay fault by randomly generated vectors. The authors also estimate the necessary test length to guarantee the detection of a given path fault.

The effectiveness of weighted random patterns to detect transition faults was reported by Waicukausky and Lindbloom [46]. In this approach the random test pattern generator is modified so that it is able to supply unequal distribution of 1s and 0s. The authors claim to have achieved a dramatic reduction in the number of test patterns needed to detect transition faults as compared to other deterministic tests.

In order to facilitate the execution of an effective AC test in a built-in self-test environment it is necessary to design the generator in such a way that it will be able to apply test patterns at machine speed. Moreover, the generator should be able to supply an arbitrary test pattern-pair at the inputs of the tested logic. Generators that have these two properties have been suggested in McAnney and Savir, and McAnney [67–69].

McAnney and Savir [67] propose a three-latch SRL for AC test. This SRL adds an L3 latch to the normal LSSD latch. This L3 latch is driven by a separate clock, and facilitates a storage of two independent bits in the L1 and L2 latches. This new SRL can be used to perform a latch-to-latch AC test at machine speed.

McAnney [68] also uses a three-latch SRL to allow for AC test without scan operations (that normally slow down the test). This SRL also has an added L3 latch and a multiplexor controlled by a test mode signal and two extra clocks. In order to avoid scanning during the response capture, a XOR gate has been added that allows compression in conjunction with signature analysis.

In order to perform a fast AC test in scan-based designs, the second vector of the pair is sometimes generated by performing a one-bit shift of the first vector. This form of AC test, called *skewed-load,* [70–71] has the disadvantage of being unable to generate all the possible AC test pairs. This reduced AC test vector space may lead to a relatively low AC test coverage. A possible way to overcome this severe problem

is connect the logic inputs that are fed from the scan, so that no two inputs that feed a given output are connected to adjacent latches on the scan path. This testable design is called *input separation.* Input separation [72–73] restores the low AC test coverage to its maximum potential by eliminating the shift dependency between the vectors that compose the AC test pair.

Still a different form of AC test in scan-based designs is to have a protocol where the first vector of the pair is independently generated, and the second vector of the pair is the combinational logic response to the first vector. This is called a *broad-side delay test.* Broad-side delay test is quite convenient in scan designs with multiple scan chains and combinational logic embedded inbetween. Normally such designs have multiple system clocks. By permuting the order in which the system clocks are applied during test, the broad-side delay test can cover different sections in the design.

THE SELF-TEST TESTER

Even when the self-test is fully built-in, it may still require some interaction from the outside world. This may merely be power and "switch to test mode" in some cases. In more complex situations, initialization sequences and special clocking may be required. The means by which these outside support and control functions are provided to the CUT is given the generic term "tester." The tester may be used to invoke various levels of BIST, some which rely mainly on the test structure built into the product. The extended function of the tester prompts one to think of it as a system including its computer and control programs.

The function of the self-test tester may differ depending on the total test strategy. We list here some of the functions that the tester may support: BIST diagnosis, including boundary scan, probe-based, and simulation-based techniques; AC test, including special clocking sequences; power-on reset; initialize storage elements, like RAMs; ability to send commands to the CUT, including select test mode, doing a single pass test, repeat test, stop on test X; ability to monitor the CUT by collecting and comparing signatures; ability to apply an external test pattern. Obviously, the tester has to have all the software support for these functions.

CHALLENGES FOR THE FUTURE

An attempt to automate the design of BIST systems was reported in Jones and Baker [74]. Along these lines, there is an urgent need for a computer-aided design (CAD) tool specifically oriented for BIST. This CAD tool should be able to take a design, propose some options on the placement of LFSRs and MISRs, compute good machine signatures, and assess random pattern testability. The CAD tool should include a logic modification program to aid the designer in improving its random pattern testability. The use of this logic modification program will help the designer choose the enhancement that mostly suits his purpose. The CAD tool should be flexible in selecting a weighting scheme if the designer so chooses. It should be able to compute the best set of weights to test the product.

There are a number of theoretical problems that look for an answer. The whole issue of assessing random pattern testability in nonscan design has not been given much attention. The question of how to generalize notions like signal probability and detection probability in an unrestricted sequential design has been neglected. A research into this area may provide some insight on how testable an unrestricted sequential design may be in a BIST environment. Another topic that needs some attention is the random pattern testability assessment in vendor logic. This is even a harder problem because the details of the implementation are not normally known.

More work is needed in the area of embedded RAMs. All the known solutions to isolate the failing component in RAM-based products are quite expensive and not very efficient.

More ideas are needed to improve the performance of fault simulators for BIST. Since chips are expected to increase in density at approximately the same pace (or even faster) as in the past, more powerful simulators are needed to cope with the testing problem that will emanate from this.

Another important problem that needs an immediate solution is the lack of test education among the designer community. There is an urgent need to educate the designers about the importance of test in general, and BIST in particular. Designers should be aware of the benefit of BIST and its cost. They should have some knowledge of what may create a testing bottleneck, so that they can avoid it during the design. Once they learn about its benefits, designers may be more willing to adopt BIST and make it an integral feature of future products.

CONCLUSIONS

This paper has described the BIST development since its inception. Attention was primarily given to

nonexhaustive random pattern-based BIST. The paper dealt with most of the important issues of BIST, and provided a list of problems that still need a solution in order to make the method more widely used in practice.

Bibliography

Abadir M.S., and M.A. Breuer "Test Schedules for VLSI Circuits Having Built-In Test Hardware," Comput. Math. Appl., Vol. 13, No. 5–6, 1987. pp. 519–536.

Aboulhamid, M.E., and E. Cerny "A Class of Test Generators for Built-In Testing," IEEE Trans. Comput., Vol. C-32, No. 10, Oct. 1983, pp. 957–959.

Agarwal, V.K., and E. Cerny "Store and Generate Built-In Testing Approach," 11th Annual Fault-Tolerant Computing Symp., Portland, ME, June 1981, pp. 35–40.

Agrawal, V.D., and M.R. Mercer "Testability Measures—What Do They Tell Us?," 1982 International Test Conference, Philadelphia, PA, Nov. 1982, pp. 391–396.

Agarwal, V.K. "Increasing Effectiveness of Built-In Testing by Output Data Modification," 13th Annual Fault-Tolerant Computing Symp., Milano Italy, June 1983, pp. 227–233.

Ahmad, A., and N.K. Nanda "Effectiveness of Multiple Compressions of Multiple Signatures," Int. J. Electron., Vol. 66, No. 5, May 1989, pp. 775–787.

Aizenbud, Y. et al, "AC Test Quality: Beyond Transition Fault Coverage," Proc. 1992 Int. Test Conf., Baltimore, MD, Sept. 20–24, 1992, pp. 568–577.

Akers, S.B. "Test Set Embedding in a Built-In Self-Test Environment," Proc. 1989 Int. Test Conf., Washington DC, Aug. 1989, pp. 257–263.

Ambler, A.P. et al "Economically Viable Automatic Insertion of Self-Test Features for Custom VLSI," Proc. Int. Test Conf., Washington DC, Sept. 1986, pp. 232–243.

Anderson, F., R. Brzozwy, and S. Metzgar "6K Gate Array with Self-Test and Maintenance," Proc. 1987 Int. Solid-State Circuits Conf., pp. 150–151.

Ando, H. "Testing VLSI with Random Access Scan," Digest of Papers, COMPCON 80, Feb. 1980, pp. 50–52.

Arzoumanian, Y., and J. Waicukauski "Fault Diagnosis in an LSSD Environment," 1981 International Test Conference, Philadelphia PA, Oct. 1981, pp. 86–88.

Baanen, P. "Testing Word Oriented Embedded RAMs Using Built-In Self Test," Proc. COMPEURO 1988, pp. 196–202.

Bardell, P.H., and W.H. McAnney "Self-Test of Random-Access Memories," IBM Tech. Discl. Bull., Vol. 26, No. 1, June 1983, pp. 336–340.

Bardell, P.H., and T.H. Spencer "A Class of Shift-Register Sequence Generators: Hurd Generators applied to Built-In Test," IBM Corporation Technical Report TR 00.3300, Sept. 20, 1984, IBM, Poughkeepsie, NY, 12602.

Bardell, P.H., and W.H. McAnney "Simultaneous Self-Testing System," U.S. Patent 4,513,418, Apr. 23, 1985.

Bardell, P.H. and M. Lapointe, "Production Experience with Built-In Self-Test in IBM ES/9000 System," Proc. 1991 Int. Test Conf., Nashville, TN, Oct. 26–30, 1991, pp. 28–36.

Barzilai, Z., J. Savir, G. Markowsky, and M.G. Smith "The Weighted Syndrome Sums Approach to VLSI Testing," IEEE Trans. Comput., Vol. C-30, No. 12, Dec. 1981, pp. 996–1000. of

Barzilai, Z., D. Coppersmith, and A.L. Rosenberg "Exhaustive Generation of Bit Patterns with Applications to VLSI Self-Testing," IEEE Trans. Comput., Vol. C-32, No. 2, Feb. 1983, pp. 190–194.

Beauchamp, K.G. *Walsh Functions and Their Applications,* Academic Press, NY, 1975.

Becker, B., and H. Soukup "CMOS Stuck-Open Self-Test for an Optimal-Time VLSI-Multiplier," Microprocess. Microprogram., Vol. 20, No. 1–3, Apr. 1987, Short Notes, Euromicro '86, Venice, Italy, pp. 153–157.

Bennetts, R.G., C.M. Maunder, and G.D. Robinson "CAMELOT: A Computer-Aided Measure for Logic Testability," IEE Proc., Vol. 128, Part E, 1981, pp. 177–189.

Berg, W.C., and R.D. Hess "COMET: A Testability Analysis and Design Modification Package," 1982 International Test Conference, Philadelphia PA, Nov. 1982, pp. 364–378.

Bhavsar, D.K., and R.W. Heckelman "Self-Testing by Polynomial Division," 1981 International Test Conference, Philadelphia PA, Oct. 1981, pp. 208–216.

Bhavsar, D.K., and B. Krishnamurthy "Can We Eliminate Fault Escape in Self Testing by Polynomial Division (Signature Analysis)?" 1984 International Test Conference, Philadelphia PA, Oct. 1984, pp. 134–139.

Bhavsar, D.K. "Concatenable Polydividers: Bit-sliced LFSR Chips for Board Self-test," 1985 International Test Conference, Philadelphia PA, Nov. 1985, pp. 88–93.

Bilton, J.M. "Survey of Self-Test and Bite Program Generation," Euromicro J., Vol. 6, No. 3, May 1980, pp. 168–174.

Bobeh, A. and J. Savir, "Statistical Resistance to Detection," IEEE Trans. Comput., V. 41, No. 1, Jan. 1992, pp. 123–126.

Bottorff, P.S., S. DasGupta, R.G. Walther, and T.W. Williams "Self-Testing Scheme using Shift Register Latches," IBM Tech. Disclosure Bull., Vol. 25, No. 10, Mar. 1983, pp. 4958–4960.

Bozorgui-Nesbat, S., and E.J. McCluskey "Structured Design for Testability to Eliminate Test Pattern Generation," 10th Annual Fault-Tolerant Computing Symp., Kyoto Japan, Oct. 1980, pp. 158–163.

Brynestad, O., E. Aas, and A. Vallestad, "State Transition Graph Analysis as a Key to BIST Fault Coverage," Proc. 1990 Int. Test Conf., Washington DC, Sept. 10–14, 1990, pp. 537–543.

Buehler, M.G., and M.W. Sievers "Off-Line, Built-In Test Techniques for VLSI Circuits," Computer, Vol. 15, No. 6, June 1982, pp. 69–82.

Buswell, S.M., and F.P. Graves "Self-Test of Timing and Control Circuitry," Proc. 1988 Reliab. and Maintain. Symp., pp. 108–111.

Calvin, D.A. "Design for Self-Test SRL Implementation," IBM Tech. Discl. Bull., Vol. 26, No. 3B, Aug. 1983, pp. 1604–1606.

Carter, J.L. "The Theory of Signature Testing for VLSI," 14th ACM Symp. Theory of Computing, San Francisco CA, May 1982, pp. 66–76.

Carter, W.C. "The Ubiquitous Parity Bit," 12th Annual Fault-Tolerant Computing Symp., Santa Monica CA, June 1982, pp. 289–296.

Carter, W.C. "Signature Testing with Guaranteed Bounds for Fault Coverage," 1982 International Test Conference, Philadelphia PA, Nov. 1982, pp. 75–82.

Caspi, P., J. Piotrowski, and R. Velazco, "An A Priori Approach to the Evaluation of Signature Analysis Efficiency," IEEE Trans. Comput., V. 40, No. 9, Sept. 1991, pp. 1068–1071.

Catthoor, F. and H. De Man, "Optimized BIST Strategies for Programmable Data Paths Based on Cellular Automata," Proc. 1992 Int. Test Conf., Baltimore, MD, Sept. 20–24, 1992, pp. 110–119.

Cerny, E., E.M. Aboulhamid, G. Bois, and J. Cloutier "Built-In Self-Test of a CMOS ALU," IEEE Design & Test, Vol. 5, No. 4, Aug. 1988, pp. 38–48.

Chan, J. and J. Abraham, "A Study of Faulty Signatures Using a Matrix Formulation," Proc. 1990 Int. Test Conf., Washington DC, Sept. 10–14, 1990, pp. 553–561.

Chen, C.L. "Linear Dependencies in Linear Feedback Shift Registers," IEEE Trans. Comput., Vol. C-35, No. 12, Dec. 1986, pp. 1086–1088.

Chen, T.-H., and M.A. Breuer "Automatic Design for Testability Via Testability Measures," IEEE Trans. Computer-Aided Design, Vol. CAD-4, No. 1, Jan. 1985, pp. 3–11.

Cheng, K., "Transition Fault Simulation for Sequential Circuits," Proc. 1992 Int. Test Conf., Baltimore, MD, Sept. 20–24, 1992, pp. 723–731.

Craig, G.L., C.R. Kime, and K.K. Saluja "Test Scheduling and

Control for VLSI Built-In Self-Test," IEEE Trans. Comput., Vol. 37, No. 9, Sept. 1988, pp. 1099–1109.

Daehn, W., and J. Mucha, "Hardware Test Pattern Generation for Built-In Testing," 1981 International Test Conference, Philadelphia PA, Oct. 1981, pp. 110–113.

Daehn, W., and J. Mucha "A Hardware Approach to Self-Testing of Large Programmable Logic Arrays," IEEE Trans. Comput., Vol. C-30, No. 11, Nov. 1981, pp. 829–833.

Damiani, M., P. Olivo, and B. Ricco, "Analysis and Design of Linear Finite state Machines for Signature Analysis Testing," IEEE Trans. Comput., V. 40, No. 9, Sept. 1991, pp. 1034–1045.

Dandapani, R., R.K. Gulati, and D.K. Goel "Built-In Self-Test for Large Embedded CMOS Folded PLAs," IEEE Int. Conf. on Comput. Aided Design, ICCAD-88, pp. 236–239.

Daniels, R.G., and W.C. Bruce "Built-In Self-Test Trends in Motorola Microprocessors," IEEE Design & Test, Vol. 2, No. 2, Apr. 1985, pp. 64–71.

Das, A. K. and P. Pal Chaudhuri, "Vector Space Theoretic Analysis od Additive Cellular Automata and its Application for Pseudoexhaustive Test Pattern Generation," IEEE Trans. Comput., V. 42, No. 3, Mar. 1993, pp. 340–352.

David, R. "Feedback Shift Register Testing," 8th Annual Fault-Tolerant Computing Symp., Toulouse, France, June 1978, pp. 103–107.

David, R. "Testing by Feedback Shift Register," IEEE Trans. Comput., Vol. C-29, No. 7, July 1980, pp. 668–673.

David, R. "Signature Analysis of Multi-Output Circuits," 14th Annual Fault-Tolerant Computing Symp., Kissimmee FL, June 1984, pp. 366–371.

Dekker, R., F. Beenker, and L. Thijssen "Realistic Self-Test Machine for Static Random Access Memories," Proc. Int. Test Conf., Washington, DC, Sept. 1988, pp. 353–361.

Dekker, R., F. Beenker, and L. Thijssen "Realistic Built-In Self-Test for Static RAMs," IEEE Design & Test, Vol. 6, No. 1, Feb. 1989, pp. 26–34.

Dervisoglu, B.I. "VLSI Self-Testing using Exhaustive Bit Patterns," Technical Report 685, MIT Lincoln Laboratory, Lexington, MA, May 31, 1984 (ESD-TR-84-019).

Dervisoglu, B.I. "Scan-Path Architecture for Pseudorandom Testing," IEEE Design & Test, Vol. 6, No. 4, Aug. 1989, pp. 32–48.

Eichelberger, E.B., T.W. Williams, E.J. Muehldorf, and R.G. Walther "A Logic Design Structure for Testing Internal Arrays," 3rd USA-JAPAN Computer Conf., San Francisco CA, Oct. 1978, pp. 266–272.

Eichelberger, E.B., and E. Lindbloom "Random-Pattern Coverage Enhancement and Diagnosis for LSSD Logic Self-Test," IBM J. Research and Development, Vol. 27, No. 3, May 1983, pp. 265–272.

Erdal, A.C., and P.A. Uszynski "Global Chip Test Implementation Including Built-In Self-Test," Proc. Int. Test Conf., Washington DC, Sept. 1988, pp. 446–449.

Fasang, P.P. "Built-In Self Test Technique for Digital Circuits," Siemens Forsch Entwicklungsber, Vol. 11, No. 2, 1982, pp. 65–68.

Fink, F., K. Fuchs, and M. H. Schulz, "Robust and Nonrobust Path Delay Fault Simulation by Parallel Processing of Patterns," IEEE Trans. Comput., V. 41, No. 12, Dec. 1992, pp. 1527–1536.

Franklin, M., K.K. Saluja, and K. Kinoshita "Row/Column Pattern Sensitive Fault Detection in RAMs via Built-In Self-Test," Proc. 19th Int. Symp. on Fault-Tolerant Comput., FTCS-19, June 1989, pp. 36–43.

Fujiwara, H. "Design for Testability and Built-In Self-Test for VLSI Circuits," Microprocess. Microsyst., Vol. 10, No. 3, Apr. 1986, pp. 139–147.

Furuya K. and E. J. McCluskey, "Two-Pattern Test Capabilities of Autonomous TPG Circuits," Proc. 1991 Int. Test Conf., Nashville, TN, Oct. 26–30, 1991, pp. 704–711.

Gartner, J., et al, "Weighted Random Test Program Generation for a Per-Pin Tester," Proc. 1990 Int. Test Conf., Washington DC, Sept. 10–14, 1990, pp. 1000–1005.

Gelsinger, P.P. "Built In Self Test of the 80386," Proc. IEEE Int. Conf. on Computer Design, ICCD-86, pp. 169–173.

Gelsinger, P.P. "Design and Test of the 80386," IEEE Design & Test, Vol. 4, No. 3, June 1987, pp. 42–50.

Gillis, P.S., and L.V.K. Smith "Self Test for Arrays Embedded in Logic: Motivation and Methodology," IBM Tech. Rpt., TR-19.90454, June 1989.

Gloster, C., and F. Brglz "Boundary Scan with Cellular-Based Built-In Self-Test," Proc. Int. Test Conf., Washington DC, Sept. 1988, pp. 138–145.

Gloster, C., and F. Brglz "Boundary Scan with Built-In Self-Test," IEEE Design & Test, Vol. 6, No. 1, Feb. 1989, pp. 36–44.

Goldstein, L.H. "Controllability/Observability Analysis of Digital Circuits," IEEE Trans. Circuits and Systems, Vol. CAS-26, No. 9, Sept. 1979, pp. 685–693.

Goldstein, L.H., and E.L. Thigpen "SCOAP: Sandia Controllability/Observability Analysis Program," 17th ACM/IEEE Design Automation Conf., Minneapolis MN, June 1980, pp. 190–196.

Grason, J. "TMEAS, A Testability Measurement Program," 16th Design Automation Conf., San Diego CA, June 1979, pp. 156–161.

Grassl, G., and H.J. Pfleiderer "Function-Independent Self-Test for Large Programmable Logic Arrays," Integr. VLSI J., Vol. 1, No. 1, Apr. 1983, pp. 71–80.

Guha, A. and L.L. Kinney, "Relating the Cyclic Behavior of Linear and Intrainverted Feedback Shift Registers," IEEE Trans. Comput., V. 41, No. 9, Sept. 1992, pp. 1088–1100.

Gupta, S. and D. Pradhan, "Can Concurrent Checkers Help BIST?," Proc. 1992 Int. Test Conf., Baltimore, MD, Sept. 20–24, 1992, pp. 140–150.

Gutfreund, K. "Integrating the Approaches to Structured Design for Testability," VLSI Design, Vol. 4, No. 6, Oct. 1983, pp. 34–37, 40–42.

Ha, D.S., and S.M. Reddy "On the Design of Pseudoexhaustive Testable PLAs," IEEE Trans. Comput., Vol. 37, No. 4, Apr. 1988, pp. 468–472.

Haberl, O.F., and H.J. Wunderlich "Synthesis of Self-Test Control Logic," Proc. COMPEURO 1989, pp. 134–136.

Hack, G.E. "Memory Self Test," US Patent 4903266, Feb. 1990.

Haedtke, J.E., and W.R. Olson "Multilevel Self-Test for the Factory and Field," Proc. 1987 Reliab. and Maintain. Symp., pp. 274–279.

Hassan, S.Z., D.J. Lu, and E.J. McCluskey "Parallel Signature Analyzers—Detection Capability and Extensions," 26th IEEE Comp. Soc. International Conf., COMPCON Spring '83, San Francisco CA, Feb.–Mar. 1983, pp. 440–445.

Hassan, S.Z., and E.J. McCluskey "Increased Fault Coverage through Multiple Signatures," 14th Annual Fault-Tolerant Computing Symp., Kissimmee FL, June 1984, pp. 354–359.

Hassan, S.Z. "Efficient Self-Test Structure for Sequential Machines," Proc. Int. Test Conf., Washington DC, Sept. 1986, pp. 12–17.

Hatayama, K. et al "Enhanced Delay Test Generator for High Speed Logic LSIs," Proc. Int. Test Conf., Washington DC, Aug. 1989, pp. 161–165.

Hayes, J.P., and A.D. Friedman "Test Point Placement to Simplify Fault Detection," IEEE Trans. Comput., Vol. C-23, No. 7, July 1974, pp. 727–735.

Hayes, J.P. "Detection of Pattern-Sensitive Faults in Random-Access Memories," IEEE Trans. Comput., Vol. C-24, No. 2, Feb. 1975, pp. 150–157.

Hellebrand, S., H. Wunderlich, and O. Haberl, "Generating Pseudo-Exhaustive Vectors for External Testing," Proc. 1990 Int. Test Conf., Washington DC, Sept. 10–14, 1990, pp. 670–679.

Hellebrand, S. et al, "Generation of Vector Patterns Through Reseeding of Multiple-Polynomial Linear Feedback Shift Registers," Proc. 1992 Int. Test Conf., Baltimore, MD, Sept. 20–24, 1992, pp. 120–129.

Hilla, S., "Boundary Scan Testing for Multichip Modules," Proc. 1992 Int. Test Conf., Baltimore, MD, Sept. 20–24, 1992, pp. 224–231.

Hess, R.D., W.C. Berg, and G.B. Hoffman "Design-for-Test Techniques Suit Diverse Applications," EDN, Vol. 32, No. 8, Apr. 15, 1987, pp. 189–191, 194–195.

Hlawiczka, A., "Parallel Signature Analyzers Using Hybrid Design of Their Linear Feedbacks," IEEE Trans. Comput., V. 41, No. 12, Dec. 1992, pp. 1562–1571.

Hortensius, P.D., R.D. McLeod, W. Pries, M.D. Miller, and H.C. Card "Cellular Automata-Based Pseudorandom Number Generators for Built-In Self-Test," IEEE Trans. Comput. Aided Design, Vol. 8, No. 8, Aug. 1989, pp. 842–859.

Hortensius, P.D., R.D. McLeod, and B.W. Podaima "Cellular Automata Circuits for Built-In Self-Test," IBM J. of Research and Development, Vol. 34, No. 2/3, March/May 1990, pp. 389–405.

Hsiao, M.Y. "Generating PN Sequences in Parallel," 3rd Annual Princeton Conf. on Inform. Sciences and Systems, Mar. 1969, pp. 397–401.

Hsiao, T.-C., and S.C. Seth "An Analysis of the Use of Rademacher–Walsh Spectrum in Compact Testing," IEEE Trans. Comput., Vol. C-33, No. 10, Oct. 1984, pp. 934–937.

Hudson, C.L. Jr., and G.D. Peterson "Parallel Self-Test with Pseudo-Random Test Patterns," Proc. Int. Test Conf., Washington DC, Sept. 1987, pp. 954–963.

Hurd, W.J. "Efficient Generation of Statistically Good Pseudonoise by Linearly Interconnected Shift Registers," IEEE Trans. Comput., Vol. C-23, No. 2, Feb. 1974, pp. 146–152.

Ibarra, O.H., and S.K. Sahni "Polynomially Complete Fault Detection Problems," IEEE Trans. Comput., Vol. C-24, No. 3, Mar. 1975, pp. 242–249.

Illman, R.J. "Self-Tested Data Flow Logic: A New Approach," IEEE Design & Test of Comp., Vol. 2, No. 2, Apr. 1985, pp. 50–58.

Illman, R.J., and S. Clarke "Built In Self Test of the Macrolan Chip," Proc. 1989 Int. Test Conf., Washington DC, Aug. 1989, pp. 735–744.

Illman, R. et al, "Built-In Self-Test of the VLSI Content Addressable Filestore," Proc. 1991 Int. Test Conf., Nashville, TN, Oct. 26–30, 1991, pp. 37–46.

Iwasaki, K. and N. Yamaguchi, "Design of Signature Circuits Based on Weight Distribution of Error-Correcting Codes," Proc. 1990 Int. Test Conf., Washington DC, Sept. 10–14, 1990, pp. 779–785.

Jain, S.K., and C.E. Stroud "Built-In Self Testing of Embedded Memories," IEEE Design & Test, Vol. 3, No. 5, Oct. 1986, pp. 27–37.

Jain, S.K. "Introduction to Built-In Self-Test Concepts and Applications," Proc. National Communications Forum, 1988, pp. 1162–1168.

Joersz, R., and C.R. Kime "Distributed Hardware Approach to Built-In Self Test," Proc. Int. Test Conf., Washington DC, Sept. 1987, pp. 972–980.

Johnston, C.E. "Cut Cost of Ownership with Optimized Self-Test and Calibration," Electro 1979 Conf. Rec., New York, NY, Apr. 1979, paper 22.1.

Kameda, T., S. Pilarski, and A. Ivanov, "Notes on Multiple Input Signature Analysis," IEEE Trans. Comput., V. 42, No. 2, Feb. 1993, pp. 228–234.

Kapur, R. and M.R. Mercer, "Bounding Signal Probabilities for Testability Measurement Using Conditional Syndromes," IEEE Trans. Comput., V. 41, No. 12, Dec. 1992, pp. 1580–1588.

Karpovsky, M., S. Gupta, and D. Pradhan, "Aliasing and Diagnosis Probability in MISR and STUMPS Using a General Error Model," Proc. 1991 Int. Test Conf., Nashville, TN, Oct. 26–30, 1991, pp. 828–839.

Kazi, A.M., and R.F. Miller "Large-Scale Integration Self-Test Site," IBM Tech. Discl. Bull., Vol. 26, No. 7A, Dec. 1983, pp. 3314–3316.

Keller, B.L., and T.J. Snethen "Built-In Self-Test Support in the IBM Engineering Design System," IBM J. of Research and Development, Vol. 34, No. 2/3, March/May 1990, pp. 406–415.

Kjeldsen, K., and E. Andresen "Some Randomness Properties of Cascaded Sequences," IEEE Trans. Inform. Theory, Vol. IT-26, No. 2, Mar. 1980, pp. 227–232.

Koenemann, B. et al, "Delay Test: The Next Frontier for LSSD Test Systems," Proc. 1992 Int. Test Conf., Baltimore, MD, Sept. 20–24, 1992, pp. 578–587.

Koike, H., T. Takeshima, and M. Takada, "A BIST Scheme Using Microprogram ROM for Large Capacity Memories," Proc. 1990 Int. Test Conf., Washington DC, Sept. 10–14, 1990, pp. 815–822.

Komonytsky, D. "Synthesis of Techniques Creates Complete System Self-Test," Electronics, Vol. 56, No. 5, Mar. 10, 1983, pp. 110–115.

Koren, I. "Analysis of the Signal Reliability Measure and an Evaluation Procedure," IEEE Trans. Comput., Vol. C-28, No. 3, Mar. 1979, pp. 244–249.

Krasniewski, A. "Knowledge-Based Selection of Self-Test Techniques in Computer-Aided Designing of Self-Testing VLSI Circuits," Proc. Fourth Int. IEEE Multilevel Interconnection Conf., 1987, pp. 440–443.

Krasniewski, A., and S. Pilarski "Circular Self-Test Path: A Low-Cost BIST Technique," Proc. 24th Design Automation Conf., 1987, pp. 407–415.

Krasniewski, A., and S. Pilarski "Cellular Self-Test Path: A Low-Cost BIST Technique for VLSI Currents," IEEE Trans. Comput. Aided Design, Vol. 8, No. 1, Jan. 1989, pp. 46–55.

Lake, R. "A Fast 20K Gate Array with On-Chip Test System," VLSI System Design, Vol. 7, No. 6, June 1986, pp. 46–47, 50–51, 54–55.

Lambidonis, D.A. Ivanov, and V. Agrawal, "Fast Signature Computation for Linear Compactors," Proc. 1991 Int. Test Conf., Nashville, TN, Oct. 26–30, 1991, pp. 808–817.

Lambrecht, R.I. "ATE Hardware Diagnostics, Fault Detection, and Fault Isolation Tool (Self Test Adapter)," Proc. Futuretest Symp., AUTOTESCON 1988, pp. 125–128.

Landis, D.L. "Self-Test System Architecture for Reconfigurable WSI," Proc. 1989 Int. Test Conf., Washington DC, Aug. 1989, pp. 275–282.

LeBlanc, J.J. "LOCST: A Built-In Self-Test Technique," IEEE Design & Test, Vol. 1, No. 4, Nov. 1984, pp. 45–52.

LeBlanc, J.J. "LSSD On-Chip Self Test (LOCST)," Proc. National Communications Forum, 1988, p. 1169–1175.

Lempel, A., and M. Cohn "Design of Universal Test Sequences for VLSI," IEEE Trans. Inform. Theory, Vol. IT-31, No. 1, Jan. 1985, pp. 10–17.

Lewis, W.J. "Implementing Exhaustive Self Test Using Minimal Cycle Count and Reduced Area Overhead," Proc. 1st European Test Conf., pp. 403–408.

Ligthart, M.M., P. Baltus, and M. Freeman "RPST: A ROM Based Pseudo-Exhaustive Self Test Approach," Proc. Int. Test Conf., Washington DC, Sept. 1987, pp. 915–922.

Liu, C.Y., K.K. Saluja, and J.S. Upadhyaya "BIST-PLA: A Built-In Self-Test Design of Large Programmable Logic Arrays," Proc. 24th Design Automation Conf., 1987, pp. 385–391.

Liu, D.L., and E.J. McCluskey "High Fault Coverage Self-Test Structures for CMOS ICs," Proc. 1987 Custom Integrated Circuits Conf., pp. 68–71.

Liu, D.L., and E.J. McCluskey "Design of Large Embedded CMOS PLAs for Built-In Self-Test," Proc. 1987 Int. Conf. on Coput. Design, (ICCD-87), pp. 678–681.

Liu, D.L., and E.J. McCluskey "CMOS PLA Design for Built-In Self-Test," Proc. 1987 Symp. on Circuits and Systems, pp. 859–862.

Liu, D.L., and E.J. McCluskey "Design of Large Embedded CMOS PLAs for Built-In Self-Test," IEEE Trans. Comput. Aided Design, Vol. 7, No. 1, Jan. 1988, pp. 50–59.

Losq, J. "Efficiency of Random Compact Testing," IEEE Trans. Comput., Vol. C-27, No. 6, June 1978, pp. 516–525.

MacWilliams, F.J., and N.J.A. Sloane "Pseudo-Random Se-

quences and Arrays," Proc. IEEE, Vol. 64, No. 12, Dec. 1976, pp. 1715–1729.
Mao, W. and M. Ciletti, "Robustness Enhancement and Detection Threshold Reduction in ATPG for Gate Delay Faults," Proc. 1992 Int. Test Conf., Baltimore, MD, Sept. 20–24, 1992, pp. 588–597.
Marinos, P.N. "Non-Linear Feedback Shift-Register as a Built-In Self-Test (BIST)," Proc. Int. Test Conf., Washington DC, Sept. 1988, p. 998.
Markowsky, G. "Bounding Signal Probabilities in Combinational Circuits," IEEE Trans. Comput., Vol. C-36, No. 10, Oct. 1987, pp. 1247–1251.
McAnney, W.H. "Diagnosis of Self-Test Failures," IBM Tech. Discl. Bull., Vol. 26, No. 9, Feb. 1984, pp. 4483–4488.
McAnney, W.H., and J. Savir "Built-In Checking of the Correct Self-Test Signature," 1986 International Test Conference, Washington DC, Sept. 1986, pp. 54–58.
McAnney, W.H., and J. Savir "Built-In Checking of the Correct Self-Test Signature," IEEE Trans. Comput., Vol. 37, No. 9, Sept. 1988, pp. 1142–1145.
McLeod, G., "BIST Techniques for ASIC Design," Proc. 1992 Int. Test Conf., Baltimore, MD, Sept. 20–24, 1992, pp. 496–505.
McCluskey, E.J., and S.B. Nesbat "Design for Autonomous Test," IEEE Trans. Comput., Vol. C-30, No. 11, Nov. 1981, pp. 866–875.
McCluskey, E.J. "Verification Testing," Digest, 19th Ann. Design Automation Conf., Las Vegas, Nev., June 1982, pp. 495–500.
McCluskey, E.J. "Built-In Verification Test," Proc. Int. Test Cont., Philadelphia, PA, Nov. 1982, pp. 183–190.
McCluskey, E.J. "Built-In Self-Test Techniques," IEEE Design & Test, Vol. 2, No. 2, Apr. 1985, pp. 21–28.
Miczo, A. "Self-Test Hardwired Control Section," IEEE Trans. Comput., Vol. C-32, No. 7, July 1983, pp. 695–696.
Min, Y., Y.K. Malaya, and B. Jin, "Analysis of Detection Capability of Parallel Signature Analyzers," IEEE Trans. Comput., V. 40, No. 9, Sept. 1991, pp. 1075–1081.
Mucha, J.P., W. Daehn, and J. Gross "Self Test in a Standard Cell Environment," IEEE Design & Test, Vol. 3, No. 6, Dec. 1986, pp. 35–41.
Muradali, F.V. Agarwal, and B. Nadeau-Dostie, "A New Procedure for Weighted Random Built-In Self-Test," Proc. 1990 Int. Test Conf., Washington DC, Sept. 10–14, 1990, pp. 660–669.
Murray, A.F., and P.B. Denyer "Testability and Self-Test in NMOS and CMOS VLSI Signal Processors," IEE Proc. Part G, Vol. 132, No. 3, June 1985, pp. 93–104.
Nadeau-Dostie, B., D. Burek, and A. Hassan, "ScanBIST: A Multi-frequency Scan-Based BIST Method," Proc. 1992 Int. Test Conf., Baltimore, MD, Sept. 20–24, 1992, pp. 506–513.
Nagle, H.T., S.C. Roy, C.F. Hawkins, M.G. McNamer, and R.R. Fritzemeier "Design for Testability and Built-In Self Test: a Review," IEEE Trans. Ind. Electron., 1989, pp. 129–140.
Nanda, N.K., A. Ahmad, and V.C. Gaindhar "Shift Register Modification for Multipurpose Use in Combinational Circuit Testing," Int. J. Electron., Vol. 66, No. 6, June 1989, pp. 875–878.
Nicolaidis, M. "Unified Built In Self-Test Scheme: (UBIST)," Proc. 12th European Solid-State Circuits Conf., ESSCIRC 1986, pp. 173–175.
Nicolaidis, M. "Transparent BIST for RAMs," Proc. 1992 Int. Test Conf., Baltimore, MD, Sept. 20–24, 1992, pp. 598–607.
Nuez, J.P., and J.M. Proust "Self-Test on Read Only Memory," IBM Tech. Discl. Bull., Vol. 27, No. 9, Feb. 1985, p. 5338.
Oakland, S.F. et al "Self-Testing the 16-Mbps Adapter Chip for the IBM Token Ring Local Area Network," IBM J. of Research and Development, Vol. 34, No. 2/3, March/May 1990, pp. 416–427.
Oomman, B., P. Kongara, and C. Mallipeddi, "Amdahl Corporation Board Delay Test System," Proc. 1992 Int. Test Conf., Baltimore, MD, Sept. 20–24, 1992, pp. 558–567.
Oshawa, T. et al "60ns 4 MB CMOS DRAM with Built-In Self Test," Proc. 1987 Int. Solid-State Circuits Conf., pp. 286–287, 430.
Oshawa, T. et al "60ns 4-Mbit CMOS DRAM with Built-In Self-Test Function," IEEE J. Solid State Circuits, Vol. SC-22, No. 5, Oct. 1987, pp. 663–668.
Palchaudhuri, P. "Design of Testable VLSI Circuits with Minimum Area Overhead," IEEE Trans. Comput., Vol. 38, No. 10, Oct. 1989, pp. 1460–1462.
Paraskeva, M., W.L. Knight, and D.F. Burrows "New Test Structure for VLSI Self-Test: The Structured Test Register (STR)," Electron. Lett., Vol. 21, No. 19, Sept. 12, 1985, pp. 856–857.
Park, B and P. Menon, "Robustly Scan-Testable CMOS Sequential Circuits," Proc. 1991 Int. Test Conf., Nashville, TN, Oct. 26–30, 1991, pp. 263–272.
Park, S. and S. Akers, "Parity Bit Calculation and Test Signal Compaction for BIST Applications," Proc. 1991 Int. Test Conf., Nashville, TN, Oct. 26–30, 1991, pp. 1016–1023.
Park, E.S., M.R. Mercer, and T.W. Williams, "The Total Delay Fault Model and Statistical Delay Coverage," IEEE Trans. Comput., V. 41, No. 6, June 1992, pp. 688–698.
Parker, K.P. "Compact Testing: Testing with Compressed Data," 6th Annual Fault-Tolerant Computing Symp., Pittsburgh PA, June 1976, pp. 93–98.
Pataricza, A. "Improved Signature Analysis Based CPU Self Test," Microprocess. Microprog., Vol. 23, No. 1, March 1988, pp. 167–172.
Pateras, S. and J. Rajski, "Cube-Contained Random Patterns and Their Application to the Complete Testing of Synthesized Multi-Level Circuits," Proc. 1991 Int. Test Conf., Nashville, TN, Oct. 26–30, 1991, pp. 473–482.
Pomeranz, I and S. Reddy, "Achieving Complete Delay Fault Testability by Extra Inputs," Proc. 1991 Int. Test Conf., Nashville, TN, Oct. 26–30, 1991, pp. 273–282.
Porter, E.H. "Testability Considerations in a VLSI Design Automation System," 1980 Test Conference, Philadelphia PA, Nov. 1980, pp. 26–28.
Pradhan, D.K. and S.K. Gupta, "A New Framework for Designing and Analyzing BIST Techniques and Zero Aliasing Compression," IEEE Trans. Comput. V. 40, No. 6, June 1991, pp. 743–763.
Ratiu, I.M., A. Sangiovanni-Vincentelli, and D.O. Pederson "VICTOR: A Fast VLSI Testability Analysis Program," 1982 International Test Conference, Philadelphia PA, Nov. 1982, pp. 397–401.
Ratiu, I.M., and H.B. Bakoglu "Pseudorandom Built-In Self-Test Methodology and Implementation for the IBM RISC System/6000 Processor," IBM J. of Research and Development, Vol. 34, No. 1, Jan. 1990, pp. 78–84.
Reddy, S.M., K.K. Saluja, and M.G. Karpovsky "Data Compression Technique for Built-In Self-Test," IEEE Trans. Comput., Vol. 37, No. 9, Sept. 1988, pp. 1151–1156.
Resnick, D.R. "Testability and Maintainability with a New 6K Gate Array," VLSI Design, Vol. 4, No. 2, Mar.–Apr. 1983, pp. 34–38.
Robinson, J.P., and N.R. Saxena "Simultaneous Signature and Syndrome Compression," IEEE Trans. Comput. Aided Design, Vol. 7, No. 5, May 1988, pp. 584–589.
Robinson, J.P., "Aliasing Probabilities for Feedback Signature Compression of Test Data," IEEE Trans. Comput., V. 40, No. 7, July 1991, pp. 867–873.
Sakashita, N. et al, "Built-In Self-Test in a 24bit Floating Point Digital Signal Processor," Proc. 1990 Int. Test Conf., Washington DC, Sept. 10–14, 1990, pp. 880–885.
Saluja, K.K., H.S. Sng, and K. Kinoshita "Built-In Self Testing RAM: A Practical Alternative," IEEE Design & Test, Vol. 4, No. 1, Feb. 1987, pp. 42–51.
Sarma, S., "Built-In Self-Test Considerations in a High-Performance, General-Purpose Processor," Proc. 1991 Int. Test Conf., Nashville, TN, Oct. 26–30, 1991, pp. 21–27.
Savir, J. "Syndrome-Testable Design of Combinational Circuits," IEEE Trans. Comput., Vol. C-29, No. 6, June 1980, pp. 442–

451. Also see corrections, ibid, Vol. C-29, No. 11, Nov. 1980, pp. 1012–1013.
Savir, J. "A New Empirical Test for the Quality of Random Integer Generators," IEEE Trans. Comput., vol. C-32, No. 10, Oct. 1983, pp. 960–961.
Savir, J. "The Bidirectional Double Latch (BDDL)," IEEE Trans. Comput., Vol. C-35, No. 1, Jan. 1986, pp. 65–66.
Savir, J., and W.H. McAnney "Double-Parity Signature Analysis for LSSD Networks," Proc. Int. Test Conf., Washington DC, Sept. 1987, pp. 510–516.
Savir, J. and W.H. McAnney, "A Multiple Seed Linear Feedback Shift Register," Proc. 1990 Int. Test Conf., Washington DC, Sept. 10–14, 1990, pp. 657–659.
Savir, J., W.H. McAnney, and S.R. Vecchio, "Testing for Coupled Cells in Random-Access Memories," IEEE Trans. Comput., V. 40, No. 10, Oct. 1991, pp. 1177–1180.
Savir, J. and W.H. McAnney, "A Multiple Seed Linear Feedback Shift Register," IEEE Trans. Comput., V. 41, No. 2, Feb. 1992, pp. 250–252.
Savir, J., "Developments in Delay Testing," Proc. 1992 IEEE VLSI Test Symp., Apr. 7–9, 1992, Atlantic City, NJ, pp. 247–253.
Saxena, N.S. Gupta, and E.J. McCluskey, "Refined Bounds on Signature Analysis Aliasing for Random Testing," Proc. 1991 Int. Test Conf., Nashville, TN, Oct. 26–30, 1991, pp. 818–827.
Saxena, N.R., P. Franco, and E.J. McCluskey, "Simple Bounds on Serial Signature Analysis Aliasing for Random Testing," IEEE Trans. Comput., V. 41, No. 5, May 1992, pp. 638–645.
Schwair, T. and H. Ritter, "Complete Self-Test Architecture for a Coprocessor," Proc. 1990 Int. Test Conf., Washington DC, Sept. 10–14, 1990, pp. 886–890.
Segers, M.T.M. "A Self-Test Method for Digital Circuits," 1981 International Test Conference, Philadelphia PA, Oct. 1981, pp. 79–85.
Shanks, J.L. "Computation of the Fast Walsh-Fourier Transform," IEEE Trans. Comput., Vol. C-18, No. 5, May 1969, pp. 457–459.
Siavoshi, F. "WTPGA: A Novel Weighted Test-Pattern Generation Approach for VLSI Built-In Self Test," Proc. Int. Test Conf., Washington DC, Sept. 1988, pp. 256–262.
Silberman, G.M. and I. Spillinger, "Functional Fault Simulation as a Guide for Biased-Random Test Pattern Generation," IEEE Trans. Comput. V. 40, No. 1, Jan. 1991, pp. 66–79.
Smith, J.E. "Measures of the Effectiveness of Fault Signature Analysis," IEEE Trans. Comput., Vol. C-29, No. 6, June 1980, pp. 510–514.
Spearman, R.W. Jr., and F.D. Patch "Quality and Control Self-Test," Proc. 1980 Test Conf., Nov. 1980, Philadelphia PA, pp. 257–260.
Sridhar, T., D.S. Ho, T.J. Powell, and S.M. Thatte "Analysis and Simulation of Parallel Signature Analyzers," 1982 International Test Conference, Philadelphia PA, Nov. 1982, pp. 656–661.
Stahnke, W. "Primitive Binary Polynomials," Mathematics of Computation, Vol. 27, No. 124, Oct. 1973.
Starke, C.W. "Efficient Self-Test Strategy for Testing VLSI Chips," Proc. COMPEURO 1989, pp. 116–119.
Stephenson, J.E., and J. Grason "A Testability Measure for Register Transfer Level Digital Circuits," 6th Annual Fault-Tolerant Computing Symp., Pittsburgh PA, June 1976, pp. 101–107.
Stroele, A. and H. Wunderlich, "Error Masking in Self-Testable Circuits," Proc. 1990 Int. Test Conf., Washington DC, Sept. 10–14, 1990, pp. 544–552.
Stroele, A., "Self-Test Scheduling with Bounded Test Execution Time," Proc. 1992 Int. Test Conf., Baltimore, MD, Sept. 20–24, 1992, pp. 130–139.
Stroud, C., "Built-In Self-Test for High-Speed Data-Path Circuitry," Proc. 1991 Int. Test Conf., Nashville, TN, Oct. 26–30, 1991, pp. 47–56.
Su, C. and C. Kime, "Computer-Aided Design of Pseudoexhaustive BIST for Semiregular Circuits," Proc. 1990 Int. Test Conf., Washington DC, Sept. 10–14, 1990, pp. 680–689.
Susskind, A.K. "Testing by Verifying Walsh Coefficients," IEEE Trans. Comput., Vol. C-32, No. 2, Feb. 1983, pp. 198–201.
Tang, D.T., and C.L. Chen "Logic Test Pattern Generation using Linear Codes," 13th Annual Fault-Tolerant Computing Symp., Milano, Italy, June 1983, pp. 222–226.
Tasar, V. "Analysis of Fault Detection Coverage of a Self-Test Software Program," Proc. 8th Int. Conf. Fault Tolerant Comput., Toulouse, France, June 1978, pp. 65–71.
Tasar, V., and H.L. Ohlef "Method for Determining the Statistically Weighted Percent Fault Detection Coverage of a Self-Test Program," Proc. Ann. Reliab. Maintain. Symp., Washington DC, Jan. 1979, pp. 39–43.
Totton, K., and S. Shaw "Self-Test: the Solution to the VLSI Test Problem?," IEE Proc. Part E, Vol. 135, No. 4, July 1988, pp. 190–195.
Trempel, C.A. "Self-Test Scheme for Embedded Memories Using the Coupling Fault Model," 1987 Symp. on VLSI Circuits, pp. 11–12.
Treuer, R., H. Fujiwara, and V.K. Agarwal, "Implementing a Built-In Self-Test PLA Design," IEEE Design & Test, Vol. 2, No. 2, Apr. 1985, pp. 37–48.
Treuer, R., V.K. Agarwal, and H. Fujiwara "New Built-In Self-Test Design for PLAs with High Fault Coverage and Low Overhead," IEEE Trans. Comput., Vol. C-36, No. 3, March 1987, pp. 369–373.
van Sas, J., F. Catthoor, and H. De Man, "Cellular Automata Based Self-Test for Programmable Data Paths," Proc. 1990 Int. Test Conf., Washington DC, Sept. 10–14, 1990, pp. 769–778.
Wadsack, R.L. "Fault Modeling and Logic Simulation of CMOS and MOS Integrated Circuits," Bell System Tech. J., Vol. 57, No. 5, May–June 1978, pp. 1449–1474.
Wang, L.T., and E.J. McCluskey "Complete Feedback Shift Register Design for Built-In Self-Test," Proc. Int. Conf. on Computer-Aided Design, ICCAD-86, pp. 56–59.
Wang, L.T., and E.J. McCluskey "Built-In Self-Test for Sequential Machines," Proc. Int. Test Conf., Washington DC, Sept. 1987, pp. 334–341.
Wang, L.T., and E.J. McCluskey "Linear Feedback Shift Register Design Using Cyclic Codes," IEEE Trans. Comput. Vol. 37, No. 10, Oct. 1988, pp. 1302–1306.
Wang, L.T., and S. Mourad "SST: Scan Self-Test for Sequential Machines," IEE Proc. Part E, Vol. 136, No. 9, Nov. 1989, pp. 569–574.
Wang, L.T., M. Marhoefer, and E.J. McCluskey "Self-Test and Self-Diagnosis Architecture for Boards Using Boundary Scans," Proc. 1st European Test Conf., pp. 119–126.
Writer, P.L. "Design for Testability," IEEE Automatic Support System Symposium for Advanced Maintainability, Westbury NY, Oct. 1975, pp. 84–87.
Wu, D.M., and J. Waicukauski "Built-In Self-Test Using Perturbed Deterministic Patterns," Proc. 1st European Test Conf., pp. 398–402.
Wunderlich, H.J. "Self-Test Using Unequiprobable Random Patterns," Proc. 17th Fault-Tolerant Comput. Symp., FTCS-17, pp. 258–263.
Yount, L.J., and G.W. Winkler "Digital Computer Self-Test Confidence Level Validation Methodology," Digital Avionics Syst. Conf., Fort Worth, Tex., Nov. 1979, pp. 335–341.
Zorian, Y., and V.K. Agarwal "Higher Certainty of Error Coverage by Output Data Modification," 1984 International Test Conference, Philadelphia PA, Oct. 1984, pp. 140–147.
Zorian, Y., and V.K. Agarwal "Optimizing Error Masking in BIST by Output Data Modification," J. of Elect. Testing, Vol. 1, No. 1, Feb. 1990, pp. 59–71.
Zorian, Y. and A. Ivanov, "EEODM: An Effective BIST Scheme for ROMS," Proc. 1990 Int. Test Conf., Washington DC, Sept. 10–14, 1990, pp. 871–879.
Zorian, Y. and A. Ivanov, "An Effective BIST Scheme for ROMs," IEEE Trans. Comput., V. 41, No. 5, May 1992, pp. 646–653.

References

[1] P. Goel, "Test Generation Costs Analysis and Projections," *17th Annual IEEE Design Automation Conf.*, 77–84, June 1980.

[2] T.W. Williams, "Test Length in a Self-Testing Environment," *Design & Test,* 2(2), 59–63, April 1985.

[3] N. Benowitz et al., "An Advanced Fault Isolation System for Digital Logic," *IEEE Trans. Comput.*, C-24(5), 489–497, May 1975.

[4] R.A. Frohwerk, "Signature Analysis: A New Digital Field Service Method," *Hewlett-Packard J.*, 28(9), 2–8, May 1977.

[5] J.P. Hayes, "Testing Logic Circuits by Transition Counting," *5th Annual Fault-Tolerant Computing Symp.*, Paris, France, 215–219, June 1975.

[6] J.P. Hayes, "Transition Count Testing of Combinational Logic Circuits," *IEEE Trans. Comput.*, C-25(6), 613–620, June 1976.

[7] J.P. Hayes, "Check Sum Methods for Test Data Compression," *J. Design Automation and Fault Tolerant Computing,* 3–17, June 1976.

[8] N.G. de Bruijn, "A Combinatorial Problem," *Nederl. Akad. Wetensch. Proc.*, Ser. A, 49(2), 758–764, 1946. Also, *Indag. Math.*, 8, 461–467, 1946.

[9] E.B. Eichelberger and T.W. Williams, "A Logic Design Structure for LSI Testability," *J. Design Automation and Fault Tolerant Computing,* 2(2), 165–178, May 1978.

[10] B. Koenemann, J. Mucha, and G. Zwiehoff, "Built-In Logic Block Observation Techniques," *Test Conference,* Cherry Hill, NJ, 37–41, October 1979.

[11] B. Koenemann, J. Mucha, and G. Zwiehoff, "Built-In Test for Complex Digital Integrated Circuits," *IEEE J. Solid-State Circuits,* SC-15(3), 315–318, June 1980.

[12] B. Koenemann, J. Mucha, and G. Zwiehoff, "Logic Module for Integrated Digital Circuits," U.S. Patent 4,377,757, March 22, 1983.

[13] P.H. Bardell and W.H. McAnney, "Self-Testing of Multichip Logic Modules," *International Test Conference,* Philadelphia, PA, 200–204, November 1982.

[14] W.H. McAnney, "Parallel Path Self-Testing System," U.S. Patent 4,503,537, March 5, 1985.

[15] A. Nishimura, Y. Nozuyama, and J. Iwamura, "Design for Testability of a 32-Bit TRON Microprocessor," *Microprocess. Microsyst.*, 13(1), 17–27, January/February 1989.

[16] Y. Nozuyama, A. Nishimura, and J. Iwamura, "Implementation and Evaluation of Microinstruction Controlled Self-Test Using a Masked Microinstruction Scheme," *Proc. Int. Test Conf.*, Washington, DC, 624–632, August 1989.

[17] P.H. Bardell, W.H. McAnney, and J. Savir, *Built-In Test for VLSI: Pseudorandom Techniques*. New York: John Wiley, 1987.

[18] E.J. McCluskey, "Built-In Self-Test Structures," *IEEE Design & Test,* 2(2), 29–36, April 1985.

[19] S.W. Golomb, *Shift Register Sequences*. Laguna Hills, CA: Aegean Park Press, Revised Ed., 1982.

[20] W.W. Peterson and E.J. Weldon, Jr., *Error-Correcting Codes*. Cambridge, MA: The MIT Press, Second Ed., 1972.

[21] P.H. Bardell and W.H. McAnney, "Pseudorandom Arrays for Built-In Tests," *IEEE Trans. Comput.*, C-35(7), 653–658, July 1986.

[22] P.H. Bardell, "Design Consideration for Parallel Pseudorandom Pattern Generators," *J. of Elect. Testing,* 1(1), 73–87, February 1990.

[23] P.H. Bardell, "Analysis of Cellular Automata Used as Pseudorandom Pattern Generators," *Proc. Int. Test Conf.*, Washington DC, 762–768, Sept. 1990.

[24] S. Wolfram, "Statistical Mechanics of Cellular Automata," *Rev. Modern Physics,* 55(3), 601–644, July 1983.

[25] S. Wolfram, "Random Sequence Generation by Cellular Automata," *Adv. in Appl. Math.*, 7, 127–169, 1986.

[26] W. Pries, A. Thanailakis, and H.C. Card, "Group Properties of Cellular Automata and VLSI Applications," *IEEE Trans. Comput.*, C-35(12), 1013–1024, December 1986.

[27] M. Serra, D.M. Miller, and J.C. Muzio, "Linear Cellular Automata and LFSRs Are Isomorphic," *Proc. 3rd Tech. Workshop on New Dir. for IC Testing,* Halifax, N.S., 213–223, October 1988.

[28] W. Daehn, T.W. Williams, and K.D. Wagner, "Aliasing Errors in Linear Automata Used as Multiple-Input Signature Analyzers," *IBM J. of Research and Development,* 34(2/3), 363–380, March/May 1990.

[29] A. Ivanov and V.K. Agarwal, "Analysis of the Probabilistic Behavior of Linear Feedback Signature Registers," *IEEE Trans. Comput.-Aided Design,* 8(10), 1074–1088, October 1989.

[30] T.W. Williams et al., "Aliasing Errors in Signature Analysis Registers," *IEEE Design & Test,* 4, 39–45, April 1987.

[31] J. Savir and W.H. McAnney, "On the Masking Probability with One's Count and Transition Count," *Proc. Int. Conf. on Computer-Aided Design (ICCAD-85),* 111–113, November 1985.

[32] K.P. Parker and E.J. McCluskey, "Probabilistic Treatment of General Combinational Networks," *IEEE Trans. Comput.*, C-24(6), 668–670, June 1975.

[33] J. Savir, G. Ditlow, and P.H. Bardell, "Random Pattern Testability," *Proc. Fault-Tolerant Comput. Symp. (FTCS-13),* 80–89, June 1983.

[34] J. Savir, G.S. Ditlow, and P.H. Bardell, "Random Pattern Testability," *IEEE Trans. Comput.*, C-33(1), 79–90, January 1984.

[35] J. Savir, "Improved Cutting Algorithm," *IBM J. of Research and Development,* 34(2/3), 381–388, March/May 1990.

[36] G. Markowsky, "Bounding Fault Detection Probabilities in Combinational Circuits," *J. of Elect. Testing: Theory and Appl.* (JETTA), 2(4), 315–323, Nov. 1991.

[37] S.K. Jain and V.D. Agrawal, "Statistical Fault Analysis," *IEEE Design & Test of Comp.*, 2(1), 38–44, February 1985.

[38] F. Brglez, P. Pownall, and R. Hum, "Applications of Testability Analysis: From ATPG to Critical Delay Path Tracing," *International Test Conference,* Philadephia PA, 705–712, October 1984.

[39] S.C. Seth, L. Pan, and V.D. Agrawal, "PREDICT—Probabilistic Estimation of Digital Circuit Testability," *15th Annual Fault-Tolerant Computing Symp.*, Ann Arbor, MI, 220–225, June 1985.

[40] J.A. Waicukauski et al., "Transition Fault Simulation by Parallel Pattern Single Fault Propagation," *International Test Conference,* Washington, DC, 542–549, September 1986.

[41] J. Savir and P.H. Bardell, "On Random Pattern Test Length," *Proc. Int. Test Conf.*, Philadelphia, PA, 95–106, October 1983.

[42] J. Savir and P.H. Bardell, "On Random Pattern Test Length," *IEEE Trans. Comput.*, C-33(6), 467–474, June 1984.

[43] K.D. Wagner, C.K. Chin, and E.J. McCluskey, "Pseudorandom Testing," *IEEE Trans. Comput.*, C-36(3), 332–343, March 1987.

[44] C. Gloster, "Synthesis for BIST with PREWARP," *Third Ann. OASIS Research Review,* Research Triangle Park, NC, May 8, 1990.

[45] W.H. McAnney and J. Savir, "Distributed Generation of Non-Uniform Patterns for Circuit Testing," *IBM Tech. Discl. Bull.*, 31(5), 113–116, October 1988.

[46] J.A. Waicukauski and E. Lindbloom, "Fault Detection Effectiveness of Weighted Random Patterns," *Proc. Int. Test Conf.*, Washington, DC, 245–255, September 1988.

[47] J. Savir, "Good Controllability and Observability Do Not Guarantee Good Testability," *IEEE Trans. Comput.*, C-32(12), 1198–1200, December 1983.

[48] V.S. Iyengar and D. Brand, "Synthesis of Pseudo-Random

Pattern Testable Designs," *Proc. Int. Test Conf.*, Washington, DC, 501–508, August 1989.
[49] W.H. McAnney, P.H. Bardell, and V.P. Gupta, "Random Testing for Stuck-At Storage Cells in an Embedded Memory," *International Test Conference,* Philadelphia, PA, 157–166, October 1984.
[50] J. Savir, W.H. McAnney, and S. Vecchio, "Testing for Coupled Cells in Random Access Memories," *Proc. Int. Test Conf.*, Washington, DC, 439–451, August 1989.
[51] J.L. Carter, L.M. Huisman, and T.W. Williams, "Determination of Testability of Combined Logic and Memory by Ignoring Memory," US Patent 4,726,023.
[52] W.H. McAnney, J. Savir, and S. Vecchio, "Random Pattern Testing for Data-Line Faults in an Embedded Multiport Memory," *Proc. Int. Test Conf.*, Philadelphia, PA, 100–105, November 1985.
[53] J. Savir, W.H. McAnney, and S. Vecchio, "Random Pattern Testing for Address-Line Faults in an Embedded Multiport Memory," *Proc. Int. Test Conf.*, Philadelphia, PA, 106–114, November 1985.
[54] J. Savir, W.H. McAnney, and S. Vecchio, "Fault Propagation Through Embedded Multiport Memories," *IEEE Trans. Comput.*, C-36(5), 592–602, May 1987.
[55] W.H. McAnney and J. Savir, "There Is Information in Faulty Signatures," *Proc. Int. Test Conf.*, Washington, DC, 630–636, September 1987.
[56] J. Savir and W.H. McAnney, "Identification of Failing Tests with Cycling Registers," *Proc. Int. Test Conf.*, Washington, DC, 322–328, September 1988.
[57] V.S. Iyengar, B.K. Rosen, and I. Spillinger, "Delay Test Generation I. Concepts and Coverage Metrics," *Proc. Int. Test Conf.*, Washington, DC, 857–866, September 1988.
[58] V.S. Iyengar, B.K. Rosen, and I. Spillinger, "Delay Test Generation II. Algebra and Algorithms," *Proc. Int. Test Conf.*, Washington, DC, 867–876, September 1988.
[59] J.A. Waicukauski et al., "Transition Fault Simulation," *IEEE Design & Test,* 32–38, April 1987.
[60] A.K. Pramanick and S.M. Reddy, "On the Detection of Delay Faults," *Proc. Int. Test Conf.*, Washington, DC, 845–856, September 1988.
[61] A.K. Pramanick and S.M. Reddy, "On the Design of Path Delay Fault Testable Combinational Circuits," *Proc. 20th Fault-Tolerant Comput. Symp. (FTCS-20),* Newcastle Upon Tyne, UK, 374–381, June 1990.
[62] J. Savir and W.H. McAnney, "Random Pattern Testability of Delay Faults," *International Test Conference,* Washington, DC, 263–273, September 1986.
[63] J. Savir and W.H. McAnney, "Random Pattern Testability of Delay Faults," *IEEE Trans. Comput.*, 37(3), 291–300, March 1988.
[64] G.L. Smith, "Model for Delay Faults Based upon Paths," *International Test Conference,* Philadelphia, PA, 342–349, November 1985.
[65] C.J. Lin and S.M. Reddy, "On Delay Fault Testing in Logic Circuits," *Proc. Int. Conf. on Computer-Aided Design (ICCAD-86),* 148–151, 1986.
[66] C.J. Lin and S.M. Reddy, "On Delay Fault Testing in Logic Circuits," *IEEE Trans. Comput. Aided Design,* 694–703, September 1987.
[67] W.H. McAnney and J. Savir, "Latch to Latch Delay Testing in LSSD Using a Three-Latch SRL," *IBM Tech. Discl. Bull.*, 89–95, October 1988.
[68] W.H. McAnney and J. Savir, "Parallel Port Shift Register Latch for Built-In Testing Using Data Compression," *IBM Tech. Discl. Bull.*, 109–111, October 1988.
[69] W.H. McAnney, "Shift Register Latch for Delay Testing," *IBM Tech. Discl. Bull.*, 32(4A), 231–232, September 1989.
[70] Savir, J. "Skewed-Load Transition Test: Part I, Calculus," Proc. 1992 Int. Test Conf., Sept. 20–24, Baltimore, MD, pp. 705–713.
[71] Patil, S. and J. Savir, "Skewed-Load Transition Test: Part II, Coverage," Proc. 1992 Int. Test Conf., Sept. 20–24, Baltimore, MD, pp. 714–722.
[72] Savir, J. and R. Berry, "At-Speed Test is not Necessarily an AC Test," Proc. 1991 Int. Test Conf., Oct. 26–30, Nashville, TN, pp. 722–728.
[73] Savir, J. and R. Berry, "AC Strength of a Pattern Generator," J. Elec. Test.: Theory and Appl. (JETTA), V. 3, No. 2, May 1992, pp. 119–125.
[74] N.A. Jones and K. Baker, "Knowledge-Based System Tool for High-Level BIST Design," *Microprocess. Microsyst.*, 11(1), 35–48, January/February, 1987.

Biographies

DR. SAVIR holds a B.Sc. and an M.Sc. degree in electrical engineering from the Technion, Israel Institute of Technology, and an M.S. in Statistics and a Ph.D. in electrical engineering from Stanford University. He is currently a Senior Engineer in the IBM Technology Products in East Fishkill, N.Y., and was previously a Senior Engineer at IBM Enterprise Systems in Poughkeepsie, NY, and a Research Staff Member at the IBM T.J. Watson Research Center, Yorktown Heights, N.Y. He is also an Adjunct Professor of Computer Science and Information Systems at Pace University, N.Y.

Dr. Savir's research interests lie primarily in the testing field, where he has published numerous papers and coauthored the text "Built-In Test for VLSI: Pseudorandom Techniques" (Wiley, 1987). Other research interests include design automation, design verification, design for testability, statistical methods in design and test, fault simulation, fault diagnosis, and manufacturing quality.

Dr. Savir has received three IBM Invention Achievement Awards, four IBM Publication Achievement Awards, and three IBM Patent Application Awards. He is a member of Sigma Xi, and a fellow of the Institute of Electrical and Electronics Engineers.

PAUL H. BARDELL received a B.S.E.E. from the University of Colorado and an M.S.E.E. and Ph.D. from Stanford University. Until his retirement in March of 1993, he was employed by IBM as an IBM Fellow and a member of the IBM Academy of Technology. He is now consulting in the field of design for testability and built-in self-test. He has written numerous papers on test and related topics. He is the co-author along with Dr. Savir of the text, Built-In Self-Test for VLSI, published by John Wiley and Sons. He has nine U.S. Patents and one pending. He is a fellow of the IEEE, a member of the American Physical Society and the New York Academy of Sciences.