A NEW PROBABILISTIC APPROACH TO VLSI CIRCUITS TESTING*

M.C. Karpovsky and L.B. Levitin
College of Engineering
Boston University
Boston, MA, USA

1. Introduction

It is well known that the problem of test generation becomes intractable (NP-hard) for complex circuits (with large numbers of inputs, gates and connecting lines), if one tries to find an optimal specific test for any given circuit.[1,2] Even to construct a test which is not optimal, but provides a good fault coverage (say, > 95%) is quite difficult. It results usually in an excessively large size of a test (in some cases the tests include tens of thousands of test patterns). Test generation and application become more and more time - and money - consuming and threaten to turn into a real bottleneck of the computer industry. The time and the cost of testing increases tremendously (in general, exponentially) with the complexity of the devices to be tested. The cost of testing is already, in many cases, higher than the cost of development and manufacturing. 'The majority of direct labor time required to

*This work was partially supported by the National Science Foundation under Grant MCS8021262.

build an Apple III is straight test time.....Testing requires 35% of the direct labor time involved in building a system.....If the test and the rework time is added together, 72% of the process time is accounted for.'[3] There is a common understanding among specialists that efforts must be concentrated to fight this trend: 'It has become evident to designers and test engineers that to combine development of VLSI, new techniques tackling the problems of testing parts are needed over and above mere extensions.'[4]

The following three approaches have been traditionally used for testing, namely: gate-level testing, functional testing and random testing. For the gate-level testing the input data for test generation consist of a gate-level description of a device under test and a gate-level description of a class of possible faults. This approach has been proved to be very efficient for SSI and MSI circuits, [5,6,7]. The classical D-algorithm [6] has been widely used for gate-level testing during many years. With the transition to VLSI technology the gate-level description of a device under test becomes too complicated and in many cases (in particular, for the user) is not available at all. Even if the gate-level description is available to a test designer the cost of test generation in many cases begins to be prohibitively high for VLSI devices [8].

These considerations stimulated the development of functional testing approach, especially for microprocessors testing. In the case of functional testing, the input data for testing is represented by a functional description of a device under test and of a class of faults [9,10,11,12,13,14,15] . In [9,10] the functional testing approach has been used for testing of microprocessors in a user's environment. This procedure was based on a general graph-theoretical model of a microprocessor at the register-transfer level. The functional fault model was developed using the register-transfer level description and the instruction set.

Binary decision diagrams have been used for functional testing in [11], where this approach has been applied to testing of ALU. For functional testing the cost of test generation for VLSI devices is still very high, especially when a broad spectrum of devices has to be tested. On the other hand, it is very difficult, in general, to estimate fault coverage in the case of functional testing.

For random testing test patterns are generated randomly [16,17,18,19,20,21,22,23,24,25]. For generation of pseudorandom test patterns linear feedback shift registers has been used [26]. Random testing has been used both at the gate level and at the functional level. In the latter case a random sequence of instructions has been used for testing (see,e.g.,[24]). The major problem in random testing is to estimate the test length and the probability of fault detection [23,24]. For random testing the cost of test generation is not high but a number of test patterns (testing time) may be very high for VLSI devices.

There is one more approach for testing, when one does not try to develop a test for a specific device, but constructs a deterministic standard test which can be used for a large set of devices. One example of this approach is presented in [27] where exhaustive testing and simple data compression schemes have been used for testing. In [28,29,30] standard tests have been developed for combinational devices such that every output depends only on a small subset of inputs.

2. The Concept of Universal Testing

Fortunately, the same reasons that make the problem of generation of individual test for various VLSI circuits so difficult - namely, the increasing complexity and large diversity of the circuits - open a way to a different approach to test generation. This approach has been recently developed in [31,32,33,34], and called <u>universal testing</u>. To explain the basic idea of this new approach we need to turn from the deterministic concept of testing to the probabilistic one. It is somewhat similar in 'ideology' to the transition from mechanics to statistical mechanics. Indeed, it is well known that only very simple mechanical systems can be treated in a deterministic way (the famous 'three-body problem' of celestial mechanics has no general solution). On the other hand, when the number of particles (or degrees of freedom) of a system becomes very large, the methods of statistical mechanics can be applied. Statistical mechanics describes the behavior of a large <u>ensemble</u> of systems of a given kind, rather than the details of the trajectory of an individual system. The remarkable fact is that the larger is the system the more exactly the vast majority of all such systems follows the same typical pattern of behavior, so that the probability that an individual system will deviate considerably is vanishingly small. Thus, the laws of statistical mechanics (thermodynamics), though being probabilistic in nature, become fully deterministic for all practical purposes. The major breakthrough in communication engineering owing to Shannon's informations theory was based on very similar ideas [35,36,37,38]. The same philosophy was successfully applied in some other areas, such as collective behavior of automata [39,40], neural networks [41], etc.

This way of reasoning brings us to the idea of _universal tests_ - the tests that are able to detect _all the faults_ of a given class in _almost all circuits_ of a given ensemble of circuits. The fraction of the circuits in which a universal test detects all the faults of a given class approaches one when the circuit complexity increases. The rigorous definition of universal sequence of tests is given in Sec. 3. Preliminarily, the concept of universal testing can be explained in the following way. Let $C_i = \{c_{ij}\}$ be a finite set of circuits c_{ij} of a given complexity, the complexity increasing with i. Suppose that a circuit c_{ij} can be chosen randomly from the set C_i with probability p_{ij} ($\sum_j p_{ij} = 1$). Denote by F_i a set of faults of given type F which can occur in any c_{ij}. Denote by $P(T_i, F_i, C_i)$ the probability that the test T_i detects all the errors from F_i

in a circuit c_{ij} randomly chosen from C_i with probability p_{ij}.

Def. 1. A sequence of tests (T_i) is called universal for detection of all faults of type F if

$$\lim_{i\to\infty} P(T_i, F_i, C_i) = 1 \quad . \tag{1}$$

To avoid misunderstanding, we would like to stress that the probability P is the fraction of the circuits for which the universal test provides complete (100%) fault coverage (and not the measure of fault coverage for a given circuit).

One should distinguish between our use of the term universal testing on one hand, and universal tests, on the other as the latter term is used in [42, 43]. Our approach is essentially probabilistic, and the universality is understood in the asymptotic meaning, as expressed in Def. 1. No use of additional hardware and no special design is assumed to improve testability of the circuits. The authors of [42, 43] suggest tests which can detect some classes of faults in all PLA with given numbers of inputs and product terms (in a deterministic way), but they can do it only by the expense of adding extra logic and designing the PLA's in a special way. Thus the interesting approach adopted in [42, 43], belongs, in essence, to the 'design for testability' area, and differs in principle from our approach.

One should not confuse universal testing with another probabilistic approach - random testing. The comparison of gate-level testing, functional testing, random testing and universal testing is given in Sec. 5 of this paper. Universal tests are somewhat similar to standard tests and aimed to fill the gap between the functional and random testing, combining the advantages of both of them. For universal testing the input data for test generation consists of parameters of a device under test (e.g., numbers of input and output lines, numbers of flip-flop, etc.) and description of a class of possible faults (e.g., stuck-at faults of a given multiplicity). Universal tests may be very efficient for a testing of a broad spectrum of devices or as a first step in a testing procedure.

The use of the concept of 'almost all devices' needs some justification. Indeed, it implies that universal testing methods divide a given set of circuits into two subsets: circuits in which all the faults of a given class are detectable by the test (let us call them 'sheeps'), and those which are not completely testable ('goats'). It might happen that, though the latter ones constitute only an infinitesimally small fraction of the total set, practically we are dealing just with these 'exceptional', 'non-typical' circuits. In fact, a similar situation is not at all unusual in coding theory, logic design, etc. (One may call it 'Shannon paradox', since it appears, in particular, in information theory). To illustrate this point, let us consider a relevant example from the theory of complexity of Boolean functions.

It is well known that for almost all Boolean functions of m arguments, the minimal number G(m) of two-input gates in network implementing these functions is $G(m) \sim \frac{r \cdot 2^m}{m}$ (r is a constant) [44], but not even one class of Boolean functions which require for their implementation more than a polynomial number of gates is known yet.

The second example relates to Shannon's basic coding theorem which states that almost all block codes of a given length provide data transmission rate arbitrarily close to the channel capacity with probability of error tending to zero when the length increases. But in spite of the fact that all but infinitesimal small fraction of codes are 'good', not a single class of such codes was known until quite recently [45].

Fortunately, this is not the case in universal testing. A number of examples considered below will demonstrate applicability of universal testing to some practically important classes of circuits. Though examples of 'goats' can be indicated among practical circuits, they seem to be relatively rare.

Universal tests for terminal stuck-at and bridging faults in combinational circuits have been developed in [31-34]. Consider the standard combinational circuits regarding their testability by these universal tests. A circuit is called a 'sheep' (s), if universal tests provide 100% coverage of faults of a given class, and it is called a 'goat' (g) otherwise. The classification is given in Table 1. One can see that 'sheeps' with respect to one class of faults may be 'goats' with respect to another class.

Table 1. Testability of standard computer components by universal tests ($m \rightarrow \infty$). s - 'sheeps', g - 'goats'

Name of the Circuit	CLASS OF FAULTS	
	Input Stuck-at	Input Bridgings
AND, OR, NAND, NOR gates	g	g
Parity checker	s	s
Majority gate (m=2n)	g	s
Shifter (with 2 control inputs)	s	s
Code convertors (binary to BCD, binary to Grey code, etc.)	s	s
Counter with parallel load	s	s
Up-and-down counter	s	s
Adder	s	s
Subtractor	s	s
Multiplier	s	s
Decoder	s	s
Multiplexer	g	g

It is remarkable that while elementary AND, OR, NAND and NOR are 'goats', more complex circuits become 'sheeps'.

Since universal testing of combinational circuits was the object of [31-34], in the present paper we shall consider mostly results related to sequential circuits.

3. Definitions and Notations.

An arbitrary digital binary circuit with memory may be represented by a block-diagram of Fig. 1.

Here m is the number of input lines, k is the number of output lines, and s is the number of feedback lines (the number of binary memory cells). Such a device will be referred to as an (m,k,s) - circuit so that an (m,k,o) - circuit is a combinational circuit.

Denote also:

$\underset{\sim}{x} = (x_1,\ldots,x_m)$ is an input binary vector,

$\underset{\sim}{y} = (y_1,\ldots,y_k)$ is an output binary vector,

$\underset{\sim}{z} = (z_1,\ldots,z_s)$ is a feedback binary vector.

For a given circuit c at discrete time τ:

$$\begin{aligned} y_j(\tau)&=y_j^c(\underset{\sim}{x}(\tau),\ \underset{\sim}{z}(\tau-1)),\ j=1,\ldots,k, \\ z_r(\tau)&=z_r^c(\underset{\sim}{x}(\tau),\ \underset{\sim}{z}(\tau-1)),\ r=1,\ldots,s, \end{aligned} \qquad (2)$$

where y_j^c and z_r^c are Boolean functions of (m+s) variables, which are specified by the structure of the circuit c. A test $T^{(m,N)} = (\underset{\sim}{t}^{(1)},\ldots,\underset{\sim}{t}^{(N)})$ is a sequence of N binary m - dimensional vectors $\underset{\sim}{t}^{(g)} = (t_1^{(g)},\ldots,t_m^{(g)})$, g=1,...,N, which are applied to the input lines of the circuit. In the absence of input faults, obviously, $\underset{\sim}{x}^{(g)}=\underset{\sim}{t}^{(g)}$.

We denote by $Y^{(k,N)}=(\underset{\sim}{y}^{(1)},\ldots,\underset{\sim}{y}^{(N)})$ the sequence of output vectors produced by application to the circuit a sequence of test patterns $T^{(m,N)} = (\underset{\sim}{t}^{(1)},\ldots,\underset{\sim}{t}^{(N)})$, provided that the memory was cleared before testing. For a fault-free device $Y^{(k,N)}$ is

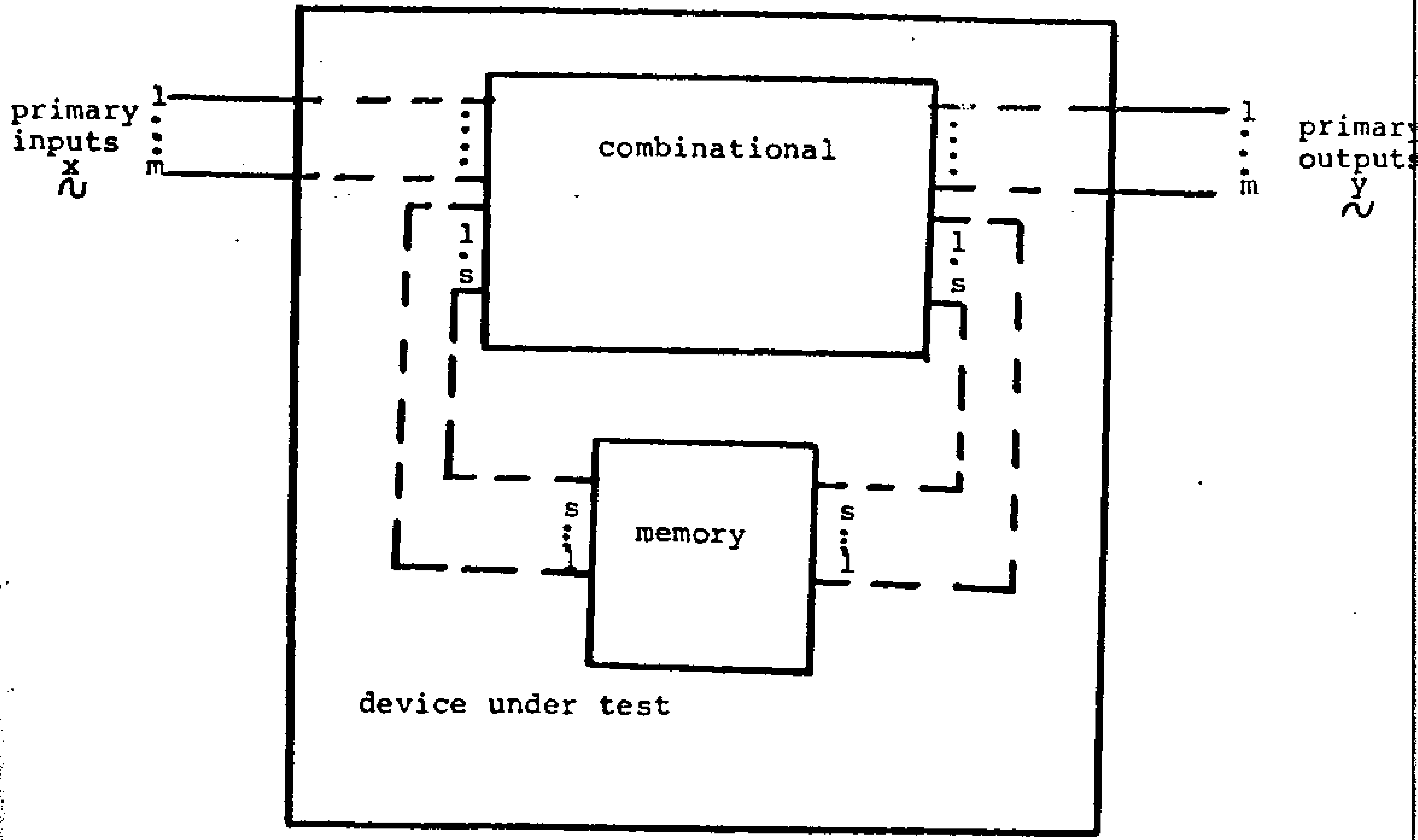

Fig. 1. Block-diagram of a device with memory.

uniquely determined by $T^{(m,N)}$. However, if a fault f occurs in the circuit, $Y^{(k,N)}$ may depend on the fault: $Y^{(k,N)}=Y^{(k,N)}(f)$. Denote by f_o the situation when circuit c is fault-free.

<u>Definition 2.</u> It is said that test $T=T^{(m,N)}$ detects a fault f in a given circuit c, if for this circuit

$$Y^{(k,N)}(f) \neq Y^{(k,N)}(f_o). \qquad (3)$$

Consider now a set of faults $F=\{f_w\}$, $w=0,1,\ldots, W$, $W=|F|-1$, which may occur in the circuit c.

<u>Definition 3.</u> It is said that test $T=T^{(m,N)}$ detects all the faults from set F in the circuit c if for any $f_w \varepsilon F$, $w \neq o$,

$$Y^{(k,N)}(f_w) \neq Y^{(k,N)}(f_o) \qquad (4)$$

<u>Definition 4.</u> It is said that test $T=T^{(m,N)}$ identifies all the faults from set F in the circuit c if for any f_v, $f_w \varepsilon F$, $v \neq w$,

$$Y^{(k,N)}(f_v) \neq Y^{(k,N)}(f_w) \qquad (5)$$

Now let us consider a set of circuits $c=\{c_n\}$, $n=1,\ldots,M$, and a probability space (E,B,P), where E is the space of elementary events $E=\{e_n\}$, the elementary event e_n is the application of a given test T to the circuit c_n. B is a Borel field on E, and P is a probability measure on E, defined by the probabilities of elementary events $p(e_n)$. For a given set of faults F there exists an element $B_{det}(F)$ (respectively $B_{id}(F)$) of the Borel field B which is the set of all the elementary events e_n such that (4) (respectively, (5)) is satisfied for the circuit c_n.

Then the corresponding probabilities

$$P\{B_{det}(F)\} = \sum_{e_n \varepsilon B_{det}(F)} p(e_n), \qquad (6)$$

$$P\{B_{id}(F)\} = \sum_{e_n \varepsilon B_{id}(F)} p(e_n), \qquad (7)$$

have a meaning of probabilities that the test T detects (respectively, identifies) all the faults from set F being applied to a circuit chosen randomly from c according to probability distribution $p(e_n)$. Henceforth we shall assume that $c = C_{seq}$ is the set of all possible (m,k,s) - circuits, i.e., each circuit $c_n \varepsilon C_{seq}$ is a relaization of (k+s) (not necessarily distinct) Boolean functions of (m+s) input variables, and exactly one circuit corresponds to any possible ordered (k+s)-tuple of Boolean functions. Thus, $|C_{seq}| = M = 2^{(k+s)2^{m+s}}$. We assume also that the probability distribution is uniform: $p(e_n) = \frac{1}{M}$. Then $P\{B_{det}(F)\}=P_{det}(T,F,m,k,s,)$ and $P\{B_{id}(F)\}=P_{id}(T,F,m,k,s)$ are equal to the fractions of all the (m,k,s) - circuits in which test T detects (respectivley, identifies) all the faults from set F.

Now let m be a (natural) variable, and k and s be functions of m (in particular, constants). Consider a sequence of sets $(C_{seq}=C(m,k,s))$ where m takes on increasing natural values, and a sequence of corresponding tests $(T^{(m,N)})$, where N is a function of m.

<u>Definition 5</u>. A sequence of tests $(T^{(m,N)})$ is called universal

for detection (respectively, for identification) of all faults from set F = F(m,k,s), if

$$\lim_{m\to\infty} P_{det}(T,F,m,k,s)=1 \qquad (8)$$

(For the sake of simplicity, we shall say also "universal tests" instead of "universal sequence of tests"). It is seen from Definition 5, that the performance of universal tests becomes the better, the larger are VLSI devices under test. This asymptotic property of universal tests makes this approach especially relevant for complex VLSI circuits.

The following notations will be used throughout the paper:

ℓ - multiplicity of faults.

$\hat{T}$ - test matrix formed by test patterns from T as rows.

$N_{det}(F,m,k,s)$ ($N_{id}(F,m,k,s)$) - asymptotic minimum number of test patterns in universal tests for detection (respectively, identification) of all fault from F.

$\varepsilon(a)$ - an arbitrary function such that $\varepsilon(a)\to\infty$ when $a\to\infty$.

Two different constructions of universal sequence of tests will be used. In the first sequence the test of size N has the following structure:

$$t_i^{(1)} = 0\,,\ i = 1,\dots,m\ ;$$

$$t_h^{(2h+1)} = 1,\quad t_i^{(2h+1)}=0,\ i\neq h,\ i=1,\dots,m,\ h=1,\dots,\tfrac{N}{2}-1\ ; \qquad (10)$$

$$\underset{\sim}{t}^{(2h)} = \overline{\underset{\sim}{t}^{(2h-1)}},\ h=1,\dots,\tfrac{N}{2}\quad .$$

Here $\bar{\underset{\sim}{t}}$ is a binary vector which is the complement (negation)of $\underset{\sim}{t}$. Note that the order of test patterns (indicated by the upper index) is essential in the case of networks with memory.

The second construction makes use of Hadamard matrices. Let n be the minimum number such that $n \geq m$ and there exists a binary Hadamard matrix $\hat{A}_n$ of order n [44]. We use the conjecture (proved for all $n<268$) that a Hadamard matrix exists for all n which are multiples of 4. Consider an $(n \times m)$-matrix $\hat{A}$ whose columns coincide with the first m columns of $\hat{A}_n$. Let a_h, $h=1,2,\ldots,n$ be the h-th row of the matrix $\hat{A}$. Design a test $T=T^{(m,N)}$ as follows: $\underset{\sim}{t}^{(1)}=\underset{\sim}{0}$, $\underset{\sim}{t}^{(2)}=\overline{\underset{\sim}{t}^{(1)}}$, $\underset{\sim}{t}^{(2g+1)}=a_g$, $\underset{\sim}{t}^{(2g+2)}=\overline{\underset{\sim}{t}^{(2g+1)}}$ $(g=1,2,\ldots,\frac{N}{2}-1)$, where $N=|T| \leq 2n$ and $\underset{\sim}{0}$ denotes all zeros vector.

4. Applications of universal tests to sequential circuiits.

4.1. Detection of input stuck-at faults.

Universal testing of terminal faults in sequential circuits is a natural generalization of the approach applied to the terminal faults in combinational circuits in [31-34]. Yet it poses a number of quite non-trivial new problems, since the temporal behavior of sequential circuits is much more complex. In particular, the time order of test patern becomes essential in this case. We present here results related to single input stuck-at faults in (m,k,s) - circuits.

Denote by $F_1^{(st)}$ the set of all single input stuck-at faults.

Theorem 1. (i). The probability that a test $T=T(m,N)$ defined by (10) detects all the faults from $F_1^{(st)}$ in a circuit chosen by random from C_{seq} satisfies the inequality:

$$P(T,F_1^{(st)},C_{seq}) \geq [1-2^{\frac{-kN}{2}}(2^{-s}+(1-2^{-s})2^{-k})^{\frac{N}{2}}]2m. \qquad (12)$$

(ii). The tests defined by (10) form a universal sequence of tests for detection of all the faults from $F_1^{(st)}$, if

$$N = 2\left\lceil \frac{\log_2 m+\varepsilon(m)}{k-\log_2(2^{-s}+(1-2^{-s})2^{-k})} \right\rceil \qquad (13)$$

(iii). The minimum number of test patterns in a universal sequence of test which detects all the faults from $F_{1(st)}$ in almost all (m,k,s) - circuits is

$$N(F_1^{(st)},m,k,s) \sim 2\left\lceil \frac{\log_2 m}{k-\log_2(2^{-s}+(1-2^{-s})2^{-k})} \right\rceil. \qquad (14)$$

Theorem 1 is a generalization of Theorem 1 and Theorem 2 in [34] for the case when the set of faults is $F_1^{(st)}$. It can be proved by a similar technique, taking into account that for a test $T(m,N)$ defined by (10) either $\underset{\sim}{t}^{(2g-1)}(f)\neq\underset{\sim}{t}^{(2g-1)}$, or $\underset{\sim}{t}^{(g)}(f)\neq\underset{\sim}{t}^{(g)}$ for all $g=1,2,\ldots,\frac{N}{2}$. (Here $t(f)$ is the external input vector in the presence of fault f.)

In partcular,

$$N(F_1^{(st)},m,k,s) \sim \begin{cases} 2\left\lceil \frac{\log_2 m}{k} \right\rceil, & \text{if } \frac{S}{K} \longrightarrow 0; \\ 2\left\lceil \frac{\log_2 m}{2k} \right\rceil, & \text{otherwise.} \end{cases} \qquad (15)$$

Comparing (13) with the corresponding results for combinational circuits [31, 33] one can see that for a small number of memory cells numbers of test patterns for combinational and sequential circuits are symptomatically equal, but if $s \gtrsim k$, then the size of the test can be reduced by half.

Example 1. Consider detection of stuck-at faults in an adder accumulator with m input lines. In this case $k=s>m$, and the test $T^{(m,2)}$ with test patterns $\underset{\sim}{t}_1 = \underset{\sim}{0} = (0,\dots,0)$, $\underset{\sim}{t}_2 = \underset{\sim}{1} = (1,\dots,1)$

detects all the faults, which agrees with Theorem 1.

The probability $P(T,F_1^{(st)}, C_{seq})$ converges to 1 very fast with the increase of the number of test patterns. For example, if $m=s=16$, $k=1$, then $P(T,F_1^{(st)},C_{seq}) \gtrsim 1-2^{-5}$ for N=10 and $P(T,F_{1(st)},C_{seq}) \gtrsim 1-2^{-15}$ for N=20. Some numerical values of $P(T,F_{1(st)},C_{seq})$ are plotted in Figure 2, for $k=s=1$.

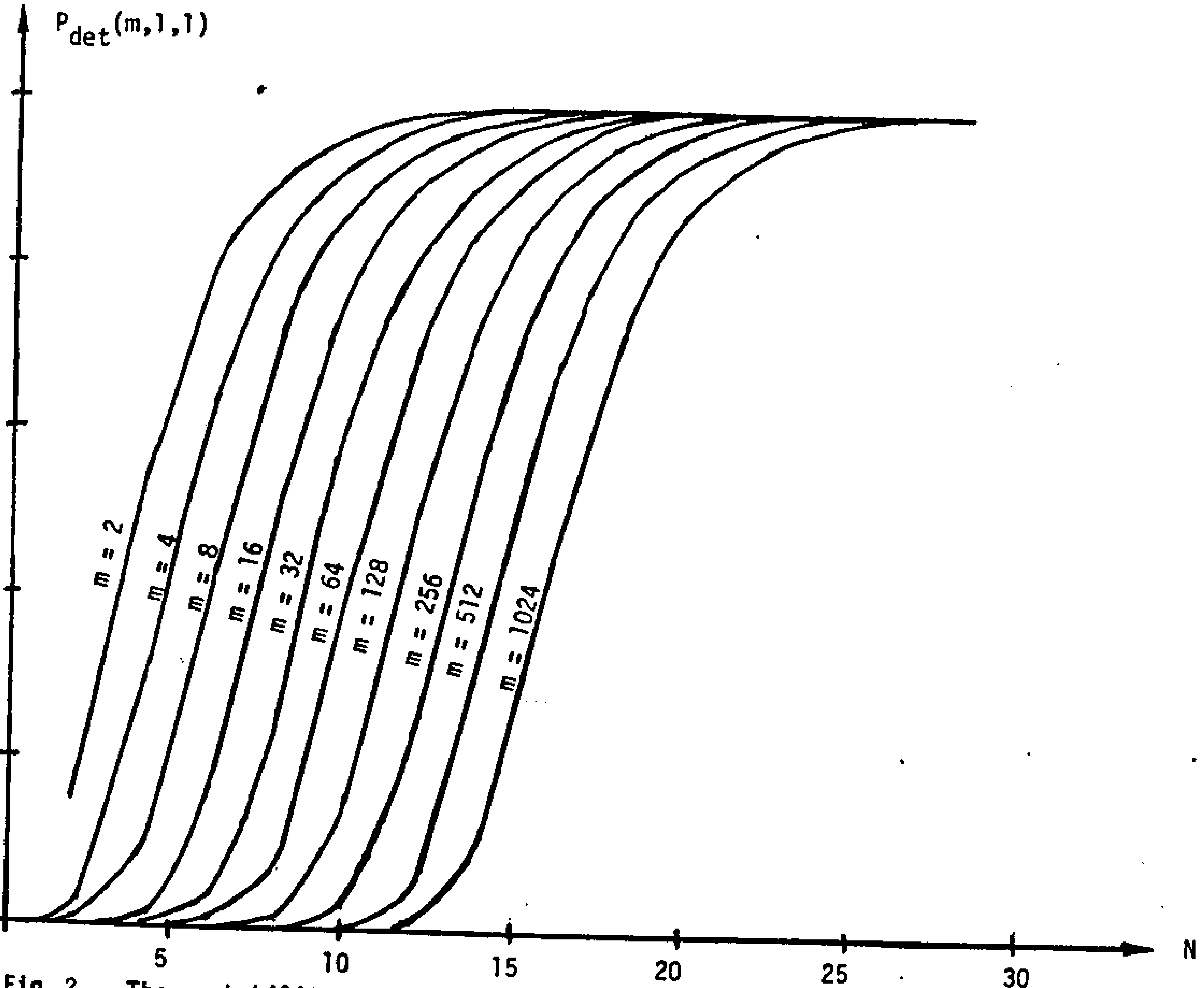

Fig. 2. The probability of detection of single input stuck-at faults in sequential (m,1,1)- circuits as a function of the number of test patterns.

4.2 Detection of stuck-at faults in memory.

Universal testing of internal faults is a complicated problem because of its generality and because the concepts of controllability and observability come into play in a more involved manner than for terminal faults. (Terminal faults can be considered as a very special case, when we have either complete controllability (input faults), or complete observability (output faults)).

We shall illustrate how universal tests can be applied to detection of internal faults by the example of the faults in memory cells in (m,k,s)-circuits. This class of faults, $F_{\ell}^{(m)}$ (ℓ is the largest fault multiplicity), deserves special analysis, since in many practical situations memory cells are less reliable than gates. Tests defined by (11) appear to be convenient for detection of these faults. The results obtained for this class of faults are summarized in the following theorem.

<u>Theorem</u> <u>2</u>. (i) The probability that a test defined by (11) detects all the faults from $F_{\ell}^{(m)}$ in an (m,k,s)-circuit chosen randomly from C_{seq}, is bounded ℓ by:

$$P(T,F_{\ell}^{(m)},C_{seq}) \geq \left(1-\left(\frac{1+2^{-k}}{2}\right)^{N}\right)^{\sum_{i=1}^{\ell} 2^{i}\binom{s}{i}} \qquad (16)$$

(ii) The tests defined by (11) form a universal sequence of tests for detection of all the faults from $F_{\ell}^{(m)}$, if

$$N = \left\lceil \frac{\log_2 \sum_{i=0}^{\ell} 2^{i}\binom{s}{i} + s(s)}{1-\log_2(1+2^{-k})} \right\rceil \qquad (17)$$

(iii) The minimum size of tests in a universal sequence of tests for detection of all faults from $F_{\ell(m)}$ is

$$N(F_{\ell}^{(m)}, m, k, s) \leq \frac{\log_2 \sum_{i=1}^{\ell} 2^{i}\binom{s}{i}}{1-\log_2(1+2^{-k})} \qquad (18)$$

To prove (16) we note that a fault in memory cells is not detected either if the memory output is not distorted by the fault, or if the memory output is distorted by the fault, but is "masked" by the combinational part of the circuit. The total probability of non-detection of a given fault by N test patterns does not exceed $(1+2^{-K})^{N}\, 2^{-N}$, which leads to (16). Then (17) and (18) can be obtained by use of the definition of universality: (17) is the necessary and sufficient condition for the convergence of $P(T, F_{\ell}^{(m)}, C_{seq})$ to 1.

In particular, for detection of single stuck-at faults in memory

$$P(T, F_1^{(m)}, C_{seq}) \geq [1-(1+2^{-k})^N\, 2^{-N}]^{2s} \quad , \qquad (19)$$

and for detection of all possible memory stuck-at faults (ℓ=s)

$$P(T, F_s^{(m)}, C_{seq}) \geq [1-(1+2^{-k})^N\, 2^{-N}]^{3^s-1} \quad . \qquad (20)$$

For example, if k=s=16, $P(T, F_1^{(m)}, C_{seq}) \geq 1-2^{-15}$ for N=20, and $P(T, F_s^{(m)}, C_{seq}) \geq 1-2^{-15}$ for N=40.

The number of test patterns for ℓ=s

$$N(F_s^{(m)}, m, k, s) \leq s(1-\log_2(1+2^{-k}))^{-1} \log_2 3 \quad . \qquad (21)$$

Thus, the number of test patterns grows at most linearly with the number of memory cells.

It can be shown that identification of input stuck-at faults and memory faults in (m,k,s) - circuits requires universal tests of double size as compared with universal tests for detection of those faults (i.e. $N_{id}(F,m,k,s) = 2N(F,m,k,s)$).

5. Advantages and Limitations of Universal Testing Methods. General remarks.

As any other method of testing, universal tests have their specific advantages and limitations. In this section we discuss briefly some general properties of universal tests, and then we compare this approach with other known methods.

5.1. Advantages.

1. Simple test generation. Since universal tests are developed for a wide class of circuits, it eliminates the necessity of generating tests individually for any particular device - the problem that becomes practically unsolvable for complex circuits, if we want to test them on the gate level.

2. Guaranteed full coverage for almost all devices. The estimation of fault coverage poses sometimes a difficult problem, for instance, in functional testing. Universal tests are constructed in such a way as to provide a 100% coverage of faults for almost all devices from a broad set. It should be taken into account, however, that this coverage is provided for a specific class of faults, for which this particular universal test was designed.

3. Applicability to complex circuits. The crucial question in universal testing is, of course, how large is the fraction of 'goats' - the circuits which are not completely testable by a given universal test. Perhaps, the most attractive feature of universal tests is that they are asymptotically efficient, i.e., they work the better the more complex is the circuit under test (e.g., the larger is the number of input and/or output faults). Therefore universal testing seems to have a good potential for VLSI testing.

For example, consider universal testing of input stuck-at faults of multiplicity at most ℓ (set of faults $F_\ell^{(st)}$) in combinational (m,k,o)- circuits. This problem has been solved in

[33,34]. The results show that if the size of the test is chosen in such a way that $N = N(F_{\ell}^{(st)}, m, k)(1+\delta)$ and $\lim_{m\to\infty} \frac{\ell}{m} = 0$, then the fraction of "goats" approaches zero as $m^{-\ell\delta}$ when $m \longrightarrow \infty$. Notice also that it works the better the larger is the multiplicity of faults ℓ. (Of course, the corresponding test size is also larger). For single stuck-at faults (the worst case) and $\delta = 5$, $m = 15 \div 20$, the fraction of "goats" does not exceed 1%.

4. Easy extension. Sometimes we may be interested to extend the universal test in order to further diminish the risk to encounter a "goat". The universal tests designed above can be easily extended (up to the limit $N = 2m$) without any change in construction. As it was shown above, with the increase of the number of test patterns, the probability that the device under test is a "goat" decreases exponentially.

5. Good coverage for "goats". One should not think that the "goats" are tested poorly by universal tests. The situation is just the opposite: in fact, though the coverage of faults in "goats" is not complete, almost all "goats" are covered almost completely. For example, if total fraction of "goats" is 1%, then only in 0.5%

of them (i.e., in a 5.10^{-5} fraction of all the circuits) two or more faults remain undetected, and only in a 1.7×10^{-7} fraction of all the devices more than two faults are undetectable by the test.

Let us consider now the general case when the required coverage of faults is not 1, but $1-\beta<1$. It has been recently

shown ([34], Theorem 7) that in order to detect a fraction of $(1-\beta)\cdot W$ faults (out of the total number W) in almost all circuits the minimum number of test patterns in a universal sequence of tests obeys inequalities:

$$N^{(\beta)}(F_{\ell}^{(st)}, m, k) \leq 2\left(\frac{1-\log_2\beta}{K}\right) \qquad (22)$$

for input stuck-at faults of any multiplicity ℓ, when $m \rightarrow \infty$, and

$$N^{(\beta)}(F_{\ell}^{(st)}, m, k) \leq n\left(m, \left\lfloor\frac{1-\log_2\beta}{K}\right\rfloor\right) \qquad (23)$$

for any input bridgings, $m \rightarrow \infty$, where $n(m,d)$ is the length of a shortest linear binary code with the distance d and the number of codewords at least m. ($\lfloor a \rfloor$ means the greatest integer, which is not greater than a). It is seen that, in contrast with the detection of all stuck-at faults, the detection of any fraction $1-\beta$ of the total number of faults (even if this fraction is arbitrarily close to 1, but remains constant) requires asymptotically a number of test patterns which depends on β only, but does not depend on m and ℓ.

<u>Example 2.</u> Let us estimate the minimum number of test patterns sufficient for detecting 99% of input stuck-at faults in almost all devices. Then $\beta = 0.01$ and, for $k = 1$, (22) yields: $N(F_{\ell}^{(st)}, m, k) \leq 14$. (If $k \geq 7$, then two test patterns are sufficient.)

6. Another advantage of the proposed approach - this time, one of the methodological nature - can be mentioned. Since from

the viewpoint of universal testing classes of circuit are described probabilistically, this opens a way for applications of powerful mathematical techniques based on probability theory and theory of random processes (for sequential circuits) to testing problems. In particular, close relation between testing theory and information theory (including coding) can be already seen. (For example, universal tests for input bridging faults are based on optimal error - correcting codes - see [34]).

5.2. Limitations.

1. The major limitation of universal testing is that it becomes efficient only in the case when a broad spectrum of circuits is to be tested. This argument is applicable both for manufacturer and user testing. Indeed, if many copies of a few types of devices only are to be tested, there is a reason to invest into development of specific tests for each of the types, since a specific test may be shorter and may provide a sufficient coverage just for this particular type. Seemingly, the most justifiable application of universal tests is in the case of a user who has to test a large variety of devices which come in small numbers of copies. Moreover, there is a good reason to use universal tests as a first step in testing procedure, since with probability close to 1 they will detect faulty devices, thereby eliminating the necessity of further testing.

2. Though universal tests are good for almost all devices, there is always a danger that a particular device under test is a "goat". Since universal testing is based on probabilistic reasoning, it provides good results only in statistical sense and

cannot guarantee against individual failures.

3. The size of universal tests may appear to be substantially larger than that of specific tests. Naturally, that is the price for their universality. However, for many typical circuits (as we have seen it in the example of terminal faults in adders, subtractors, multipliers, decoders) the universal tests are not longer than specific ones.

Universal testing, by its nature, combines some feature of functional and random testing, filling the gap between them. Therefore, we shall compare in more details these two approaches with universal testing.

5.3. Universal vs. functional testing.

Functional testing requires complete knowledge of the functional description of the device under test, while universal testing based on general information only about the class of devices (such as number of input and output variables, some general features of circuit topology, etc.). Functional testing, being device-oriented, can provide shorter tests than the universal one, but has greater test generation complexity. It is usually very hard to estimate the fault coverage for functional tests. The fault coverage for universal tests is known in terms of probability that in a device chosen from a given class all (or a given fraction) of faults are detectable.

Let us illustrate the situation for the case of multiple input stuck-at faults. The problem of functional testing for this class of faults has been considered in [46,47]. It has been shown in [46] that, if the number of inputs $m>5$, a specific test can be constructed for any such circuit which detects all input stuck-at faults and consist of not more than 2m-4 test patterns. There exists an example of a function which requires at least 2(m-r) tests patterns, where $r \geq \log_2(m-r)$. It follows from the results described in [33,34] that a universal test which consists of $N = 2m(\log_2 3 + \delta)$ test patterns will detect all the input stuck-at faults in all possible devices, except for a fraction of $2^{-m\delta}$. On the other hand, even for this simple class of faults functional test generation is much more complex than that for universal tests.

5.4. Universal vs. random testing.

At the first glance universal testing looks very similar to random testing. Indeed, both of them ignore specific features of the device under test and the performance of both types of tests can be characterized in terms of probability. However, this similarity is misleading, since there are essential differences between these two approaches which lead to different results as regards to the size of tests and their fault-detecting capability.

1. Universal tests are usually designed in a deterministic manner: where not only the test patterns, but also their order (in case of sequential circuits) may be essential. On the other hand, for random testing test patterns and their order are chosen randomly [5,17,18,19,20,21,22,23,24,25,48].

2. In contrast with random testing, universal tests are designed on the base of information about a class of circuits.

The class can be characterized by the numbers of input and output lines, feedback lines, memory cells, by the number of gates and/or internal lines, by characterization of the class inplemented functions, e.g., self-dual or symmetric functions, by features of circuit topology, such as path complexity, number of gate levels (e.g., PLA's), etc.

3. Universal tests are fault-oriented: they are designed for a specific class of faults and depend on this class, as we have seen in it on the example of single and multiple input stuck-at faults and input bridging faults at input lines. It was shown, for instance in [34], that one needs a universal test with two patterns only to detect all single input stuck-at faults in such devices as adders, subtractors, decoders, counters with parallel load while a random test would require more than $\log_2 m$ test patterns. For the class of all input multiple bridging faults we need [34] $N \sim \log_2 m$ for universal tests (based on error-correcting codes with the distance depending on the multiplicity of faults), while a random test would require $N \geq 2\log_2 m$ to detect all single input bridging faults. Thus, as a rule, universal tests are substantailly shorter than random ones. However, in case of output faults universal test degenerate to random tests. As a matter of fact, the extent of universality is rather flexible and depends on the width of the class of circuits and the class of faults involved. If the classes are not specified, we come back to random testing, while a complete description of a device on functional or on gate level will lead to functional or gate-level testing, respectively. Thus, both

device-specific and random testing may be considered as special limit cases of universal testing. The comparison of various types of testing is summarized in Table 2.

Table 2.

Comparative characterization of various approaches to testing.

Type of testing	Input data for test generation	Test generation complexity	Test size necessary for a given fault coverage	Fault coverage for a given test size
1. Gate-level	Gate-level description of the device under test: class of fault	Maximal (Optimal test generation is NP-hard)	Minimal	Maximal
2. Functional	Functional description of the device under test: class of faults	Large	Larger than for a gate-level test	Lower than for a gate-level (often, difficult to estimate)
3. Universal	Description of a class of devices: class of faults	Small (the same test for the whole class of devices, for a given set of faults)	Larger than for the first two types. (Depends on the class of devices and the set of faults)	High for almost all devices from a given class
4. Random	Set of possible test patterns only	Minimal	Maximal	Minimal

Finally, we may conclude that universal testing suggests a viable alternative to other types of testing and expands the arsenal of tools for a testing engineer.

REFERENCES

[1] Ibarra, A.K. and Sahni, S., 'Polynomially Complete Fault Detection Problems', IEEE Trans. on Computers, C-24, March 1975, pp. 242-248.

[2] Sahni, S. and Bhatt, A., 'The Complexity of Design Automation Problems', Proc. 17th Design Automation Conference, Minneapolis, MN, May 1980, pp. 402-411.

[3] Broussard, M.E., 'Higher Yields, Lower Costs', Ibid., pp. 6-11.

[4] Chalkley, M.J., 'Trends in VLSI Testing', 1979 Test Conference, Oct. 23-25, 1979, pp. 3-6.

[5] Breuer, M.A., and Friedman, A.D., 'Diagnosis and Reliable Design of Digital Systems', Computer Science Press, 1976.

[6] Roth, J.P., 'Diagnosis of Automata Failure: A Calculus and a Method', IBM Journal of Research and Development, Vol. 10, July 1966, pp. 278-291.

[7] Sellers, F.F., Hsiao, M.Y., and Bearnson, C.L., 'Analyzing Errors with the Boolean Difference', IEEE Trans. on Computers, C-17, July 1968, pp. 676-683.

[8] Goel, P., 'Test Generation Cost Analysis and Projections', 17th Design Automation Conference Proceedings, Minneapolis, MN, June 1980, pp. 77-84

[9] Thatte, S.M., and Abraham, J.A., 'Test Generation for Microprocessors', IEEE Trans. On Computers, Vol. C-29,N6, June 1980, pp. 423-441.

[10] Abraham, J.A., and Parker, K.P., 'Practical Microprocessor Testing: Open Loop and Closed Loop Approaches', FTCS 1981.

[11] Akers, Sheldon B., 'Functional Testing with Binary Decision Diagrams', The 8th Annual International Conference on Fault Tolerant Computing, Toulouse, France, June 1978.

[12] Karpovsky, M.G., 'Error Detection in Digital Devices and Computer Programs with the Aid of Linear Recurrent Equations over Finite Commutative Groups', IEEE Trans. on Computers, C-26, N3, 1377, pp. 208-219.

[13] Karpovsky, M.G., and Trachtenberg, E.A., 'Linear Checking Equations and Error-Correcting Capability of Computaton Channels', Proc. IFIP Congress, North Holland, pp. 619-624, 1977.

[14] Goel, N., and Karpovsky, M.G., 'Functional Testing of Computer Hardware Based on Minimizaton of Magnitude of Undetected Errors', IEEE Proc., Vol. 129, Sept. 1982, pp. 169-181.

[15] Akers, S.B., 'Functional Testing with Binary Decision Diagrams', The 8th Int. Symposium on Fault-Tolerant Computing, Toulouse, France 1978, pp. 75-82.

[16] Shedletsky, J.J., 'Random Testing: Practicality vs. verified effectivness', The 7th Annual Intern. Conference on Fault-Tolerant Computing, Los Angeles, CA, June 1977, pp. 175-179.

[17] Rault, J.C., 'A Graph Theoretical and Probabilistic Approach to the Fault Detection of Digital Circuits', International Symposium on Fault-Tolerant Computing, March 1971, pp. 16-29.

[18] David, R., and Thevenod-Fosse, P., 'Minimal Detecting Transition Sequences: Application to Random Testing', IEEE Trans. on Computers, C-29, N6, June 1980, pp. 514-518.

[19] Agrawal, V.D., and Agrawal, P., 'An Automatic Test Generation System for ILLIAC-IV Logic Boards', IEEE Trans. on Computers, Vol. C-21, pp. 1015-1017, Sept. 1972.

[20] Agrawal P., and Agrawal, V.D., 'Probabilistic Analysis of Randon Test Generation Method for Irredundant Logic Networks', IEEE Trans. on Computers, Vol. C-24, pp. 681-685, June 1975.

[21] Agrawal, P., and Agrawal V.D., 'On Improving the Efficiency of Monte Carlo Test Generation', Int. Symposium on Fault-Tolerant Computing, Paris, France, pp. 205-208, June 1975.

[22] Bostin, D., Girard, E., Rault, J.C., and Tullone, R., 'Probabilistic Test Generation Methods', Int. Symposium on Fault-Tolerant Computing, Palo Alto, CA, p. 171, June 1973.

[23] Savir, J., Ditlow, G., and Bardell P.H., 'Random Pattern Testability', International Symposium on Fault-Tolerant Computing, June 1983, Milano, Italy.

[24] Thevenod, P., and David, R., 'Random Testing of Control Section of a Microprocessor', International Symposium on Fault-Tolerant Computing, June 1983, Milano, Italy.

[25] Thevenod-Fosse, P., and, David, R., 'A Method to Analyze Random Testing of Sequential Circuits', Digital Processes, Vol. 4, pp. 313-332, 1978.

[26] McCluskey, E.J., Bozorgui-Nesbat, S., 'Design for Autonomous Test', IEEE Trans. on Computers, V. C-30, pp. 866-875.

[27] Savir, J., 'Syndrome-Testable Design of Combinational Circuits', IEEE Trans. on Computers, C-29, No 6, June 1980, pp. 442-450.

[28] Barzilai, Z., Coppersmith, D., and Rosenberg, A., 'Exhaustive Bit Pattern Generation in Discontiguous Positions, with Applications to VLSI Self-testing', IBM Research Report, RC-8750, March 1981.

[29] Tang, T.D., Chen, C.L., 'Logic Test Pattern Generation using Linear Codes', Proc. of 13th International Fault-Tolerant Computing Symposium, Milano, Italy, June 1983, pp. 222-226.

[30] Tang, T.D. and Woo, L.S., 'Exhaustive Test Pattern Generation with Constant Weight Vectors', IBM Research Report, RC-9442, June 1982.

[31] Karpovsky, M.G., 'Universal Tests for Detection of Input/Output Stuck-at and Bridging Faults', IEEE Trans. on Computers, Nov. 1983.

[32] Karpovsky, M.G., 'Universal Tests Detecting Input/Output Faults in Almost All Devices', Proc. of 1982 Int. Test Conference, Cherry Hill, NJ, Nov. 1982.

[33] Karpovsky, M., and Levitin, L., 'Error Detection in Combinational Networks by Use of Codes Based on Hadamard Matrices', 1983 IEEE Inter. Symp. on Information Theory, St. Jovite, Quebec, Canada, 1983.

[34] Karpovsky, M.G., and Levitin, L., 'Detection and Identification of Input/Output Stuck-at and Bridging Faults in Combinational and Sequential VLSI Networks by Universal Tests', to appear in Integration, The VLSI Journal, 1983.

[35] Shannon, C.E., and Weaver, W., 'The Mathematical Theory of Communication', Univ. of Illinois Press, Urbana, IL, 1949.

[36] Levitin, L. 'Information Content of an Image, Applications of Holography and Optical Data Processing', Pergamon Press, 1977.

[37] Levitin, L., 'A Thermodynamic Characterization of Ideal Physical Information Channels', Journal of Information and Optimization Sciences, V. 2, No. 3, September 1981.

[38] Levitin, L., 'Physical Limitations of Rate, Depth and Minimum Energy in Information Processing', International Journal of Theoretical Physics, V. 21, No. 2/3, 1982.

[39] Shmukler, Yu., 'Unidimensional and Bidimensional Gur Games', Automation and Remote Control, No. 10, 1970, p.1634.

[40] Shmukler, Yu., 'Ballot Problem with a Random Error', DAN SSSR, v. 196, No. 4, 1971, p. 789.

[41] Rosonoer, L.I., 'Random Logical Nets, I, II, and III', Automation and Remote Control, No. 5, pp. 773-781, No. 6, pp. 922-931, No. 7, pp.1130-1139, 1969.

[42] Fujiwara, H., Kinoshita, K., and Ozaki, H., 'Universal Test Sets for Programmable Logic Arrays', in Dig. 10th Inter. Symp. on Fault-tolerant Computing, Kyoto, Japan, June 1980, pp. 137-142.

[43] Fujiwara, H., Kinoshita, K., 'A Design of Programmable Logic Arrays with Universal Tests', IEEE Trans. on Circuits and Systems, V. CAS-28, No. 11, pp. 1027-1032, Nov. 1981.

[44] Lupanov, O.B., 'On the Possibilities of Synthesis of Networks from Different Types of Elements', Dokl. Akad. Nauk, USSR, 103, (1955), pp. 561-563.

[45] MacWillilams, F.J. and Sloane, N.J.A., The Theory of Error-Correcting Codes, North Holland, 1977.

[46] Weiss, C.D., 'Bounds on the Length of Terminal Stuck-Fault Tests', IEEE Trans on Computers, C-21, 1972, pp.305-309.

[47] Kuhl T.G., and Reddy, S.M., 'On Detection of Terminal Stuck-Faults', IEEE Trans. on Computers, C-27, May 1978, pp. 467-469.

[48] Huary, H., and Breuer, M.A., 'Analysis of Detectability of Faults by Random Patterns in a Special Class of NAND Networks', Computer and Elect. Engr., Vol. 1, pp. 77-84, March 1975.