ANALYSIS OF LOGIC CIRCUITS WITH FAULTS USING INPUT SIGNAL PROBABILITIES*

KENNETH P. PARKER and EDWARD J. McCLUSKEY
DIGITAL SYSTEMS LABORATORY
Departments of Electrical Engineering and Computer Science
Stanford University
Stanford, California

ABSTRACT

In this paper a method is given for calculating the probability that the output of a general combinational network is 1 given the probabilities for each input being 1. We define the notions of the probability of a signal and signal independence. Then several proofs are given to show the relationship between boolean operations and algebraic operation upon probabilities. As a result of these a simple algorithm is presented for calculating output probabilities. Simplifications that arise when sets of input probabilities are set to identical values are noted in a process called bundling. Then a series of examples illustrate various techniques for handling problems currently relevant to fault tolerant computing.

INTRODUCTION

In this paper we present an algorithm for general combinational logic circuits which allows the calculation of an output probability given a set of input probabilities. An output signal probability is defined as the probability that the output attains a specified value, and similarly, an input signal probability is the probability that a given input has the specified value. The circuit function (which is independent of circuit structure) effectively provides a transform from input to output probabilities. Here we emphasize that we are treating the signals in a probabilistic sense and not the circuit components or function, and that we consider input signals to be independent binary random variables. We then motivate these results with a series of examples to illustrate the application of the probabilistic approach.

ASSIGNING A PROBABILITY TO A LOGIC SIGNAL

First, we arbitrarily choose a "reference value" about which to define probabilities. We choose "1" and then define the probability of a logic signal as follows:

Definition 1: The probability of a (logic) signal, expressed as

$$a = P(A=1)$$

for signal A, is a real number on the interval [0,1] which expresses the probability that signal A equals 1. [We use the convention here that upper case letters correspond to signal names (boolean variables) and lower case letters represent the corresponding probabilities.] Since boolean algebra is based on two-valued variables, the probability that signal A = 0 is given as

$$P(A=0) = 1 - P(A=1) = 1 - a.$$

OPERATIONS ON SIGNAL PROBABILITIES

The following statements relate boolean operations to corresponding operations on probabilities.

Lemma 1: Boolean negation (NOT) such as in the expression $B = \bar{A}$ corresponds to the probability expression:

$$b = 1 - a$$

Proof: $b = P(B=1) = P(\bar{A}=1) = P(A=0) = 1 - P(A=1) = 1 - a.$ □

* This work was supported in part by the National Science Foundation under Grant GJ-40286.

Before continuing, we must clarify the notion of independence.

The concept of independence is that given a set of events, these events are individually determined by properties that are in no way interrelated. That is, event A (or subset of events A) in no way affects the outcome of event B (or subset of events B). The mathematical equivalent of the statement "A is independent of B" is

$$P(AB) = P(A)P(B).$$

Independence is a family property; i.e., if events $X_1, X_2, \ldots X_n$ are independent, $[P(X_1X_2\ldots X_n) = P(X_1)P(X_2)\ldots P(X_n)]$ then all possible subsets of events are independent. We continue.

Lemma 2: Boolean conjunction (AND) of two independent signals A,B in the expression C = AB correspond to the probability expression c = ab.

Proof: C=1 if A=1 AND B=1. It follows from the independence property that:

$$c=P(C=1) = P(A=1 \text{ AND } B=1) = P((A=1)(B=1)) = P(A=1)P(B=1) = ab.$$ □

Corollary: $Z = X_1X_2X_3\ldots X_n$ corresponds to the probability expression $z = x_1x_2x_3\ldots x_n$ provided all X_i are independent.

Lemma 3: Boolean disjunction (OR) of two independent signals A,B in the expression C = A ∨ B corresponds to the probability expression c = a + b - ab.

Proof: $C = A \vee B = \overline{\bar{A}\bar{B}}$

$$c = P(C=1) = P(\overline{\bar{A}\bar{B}}=1) = 1 - P(\bar{A}\bar{B}=1) = 1 - P(\bar{A}=1)P(\bar{B}=1) = 1 - (1 - a)(1 - b) = a + b - ab.$$ □

Corollary: $Z = X_1 \vee X_2 \vee \ldots \vee X_n$ corresponds to

$$z = 1 - \prod_{i=1}^{n} (1 - x_i)$$ provided all X_i are independent.

Lemma 4: Boolean disjunction (OR) in the expression $C = A \vee B$ with the restriction that the inputs A,B are not simultaneously 1 yields the following expression:

$$c = a + b$$

Proof: Since only 1 input A or B equals 1 at a time, their union is disjoint (i.e., AB = 0 and P(AB=1) = 0). A basic axiom [1] of probability is that the probability of the disjoint union of events is the sum of the individual probabilities. Hence the lemma follows directly from the axiom. □

Corollary: $Z = \bigcup_{i=1}^{n} X_i$ where $X_iX_j = 0$ for all $i \neq j$

corresponds to $z = \sum_{i=1}^{n} x_i$.

Notice that Lemma 3 can be shown from Lemma 4. Let C = A ∨ B with A,B independent. Then $C = AB \vee \bar{A}B \vee A\bar{B}$. But $A\bar{A}B=0$ and $AB\bar{A}=0$ and $\bar{A}B\bar{A}=0$ so Lemma 4 produces

$$c = P(AB=1) + P(\bar{A}B=1) + P(A\bar{B}=1) = ab + (1 - a)b + a(1 - b) = ab + a - ab + b - ab = a + b - ab,$$ which is the result of Lemma 3.

Reprinted from *FTCS 4* 1974, pp.1.8-1.12, © IEEE
0-8186-7150-5/96 $5.00 © 1996 IEEE
Proceedings of FTCS-25, Volume III

AN ALGORITHM FOR PRODUCING PROBABILITY EXPRESSIONS FROM GENERAL NETWORKS

We base our algorithm on the following facts:

1) Any general combinational network function can be expressed as a canonical sum [2] of fundamental products Π_i.

2) In a canonical sum of Π_i, at most one Π_i=1 for any input, i.e., $\Pi_i \Pi_j$=0 for all $i \neq j$.

ALGORITHM 1:

Step 1) From the circuit description obtain the canonical sum expression.* $Z = \bigcup_{i=1}^{k} \Pi_i$

Step 2) For each Π_i in the canonical sum, produce the appropriate π_i product, where $\pi_i = P(\Pi_i = 1)$.

Step 3) The probability expression is given as

$$z = \sum_{i=1}^{k} \pi_i$$

which may be simplified algebraically.

Since each boolean function has a unique canonical sum, there is a unique probability expression for each boolean function.

Example 1: Given the circuit in Fig. 1, find the probability expression.

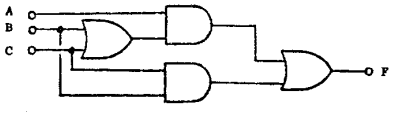

Figure 1. Circuit realizing $F = A(B \vee C) \vee BC$.

Step 1) $F = A(B \vee C) \vee BC = A\bar{B}C \vee \bar{A}BC \vee AB\bar{C} \vee ABC$

Step 2) $P(ABC = 1) = abc$
$P(\bar{A}BC = 1) = (1 - a)bc = bc - abc$
$P(A\bar{B}C = 1) = a(1 - b)c = ac - abc$
$P(AB\bar{C} = 1) = ab(1 - c) = ab - abc$

Step 3) $f = abc + ab - abc + ac - abc + bc - abc$
$= ab + ac + bc - 2abc$

EQUAL INPUT PROBABILITIES

In a number of instances (as in examples 2, 4, 5, and 6) it is convenient to let some input probabilities be equal. For example, if we let $a = b = c = x$ in the probability expression derived in example 1, then $f = 3x^2 - 2x^3$. Though this new expression lacks some of the degrees of freedom of the old expression it is simpler and in some cases the extra freedom of the more complex expression is not useful. We shall call the process of assigning equal probabilities to a set of inputs bundling, i.e., each signal in a bundle is constrained to have the same input probability as the others. Since it is the probabilities that are equal among bundled inputs, we have not introduced any dependencies among the inputs. We shall say that in analyzing a circuit where we are considering bundled inputs, we are operating within a bundling restraint.

As just illustrated, we can analyze a circuit with a bundling restraint by first deriving the probability expression using algorithm 1 and then making the appropriate variable substitutions. However, if it is known beforehand that certain inputs should be bundled, the resultant expression can be derived directly by modifying algorithm 1.

Algorithm 1 produces a probability expression by summing all the subexpressions related to each fundamental product that exists for a given function. Since each input X_i in a fundamental product is either complemented or uncomplemented, the corresponding component in the probability expression is either $(1 - x_i)$ or x_i. If we consider a fundamental product in which m of the X_i are bundled (i.e., each with probability x), then the expression for that fundamental product will contain a term of the form $x^j(1 - x)^{m-j}$ where j is the number of X_i in the bundle that were uncomplemented. There are $\binom{m}{j}$ possible fundamental products that have j uncomplemented variables in a bundle of m inputs (though not all or any may be present in a given function). Since each of these $\binom{m}{j}$ products will give rise to the same $x^j(1 - x)^{m-j}$ component in the probability expression we see the following:

GIVEN 1) a function of n inputs $Z(X_1, X_2, \ldots, X_n)$ expressed as a sum of fundamental products Π_ℓ,

2) a set of b disjoint non-empty bundles Y_i with associated probabilities y_i,

3) m_i equal to the number of inputs in bundle Y_i,

THEN 1) we can assign each Π_ℓ to a class $[\Pi_\ell]$ of fundamental products with same number of uncomplemented variables in each bundle. That is, for each class there are integers $j_1, j_2, \ldots, j_b$ such that each fundamental product in $[\Pi_\ell]$ has exactly j_i uncomplemented variables in bundle Y_i.

2) we can write down a probability subexpression for each class $[\Pi_\ell]$ as

$$s_\ell = y_1^{j_1}(1 - y_1)^{m_1 - j_1} y_2^{j_2}(1 - y_2)^{m_2 - j_2} \cdots y_b^{j_b}(1 - y_b)^{m_b - j_b}.$$

3) we can get the final probability expression for z in terms of the variables $y_1, y_2, \ldots, y_b$ by summing the subexpressions s_ℓ for each class weighted by the number of Π_ℓ in each class.

The original result of algorithm 1 was an expression for z derived by summing the subexpressions for each Π_ℓ;

$$z = \sum_{\Pi_\ell} s_\ell.$$

With bundling restraints allowed this generalizes to summing weighted subexpressions for each class $[\Pi_\ell]$;

$$z = \sum_{[\Pi_\ell]} \left|[\Pi_\ell]\right| s_\ell.$$

Example 4 will illustrate this modified algorithm.

We now motivate this work with a variety of examples which use the previous results and provide new approaches to problems in fault tolerant computing.

*Or form a cover of disjoint implicants.

MOTIVATION

Until very recently the concepts of probability were not used in discussions of circuit testing and verification. In [3] it is noted that the longer a circuit under test successfully executes a chain of tests, the greater the probability that the circuit is fault free, which is useful for circuits where exhaustive testing is not practical and a complete test set is not available. In [4] a number of empirical results are presented showing that controlling the statistical properties of a series of "randomly" generated tests will greatly affect the fault detection capacity of these tests. Example 2 illustrates why this is so.

Example 2: On random testing: consider the circuit $Z = \overline{ABCDE} \vee FGHI$ shown in Fig. 2.

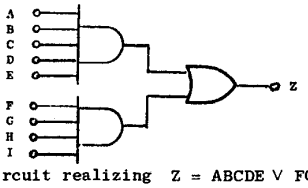

Figure 2. Circuit realizing $Z = \overline{ABCDE} \vee FGHI$.

We see that in order to detect the failure Z stuck-at-0 we need to provide a Z = 1 output. Thus the probability of detecting Z stuck-at-0 is simply P(Z=1) when we apply randomly chosen inputs.

Now $z = abcde + fghi - abcdefghi$.

Let the input test vectors be chosen such that P(A=1) = P(B=1) = ... = P(I=1) = x, which can be the case when tests are created from a set of random numbers, a popular method. Then $z = x^4 + x^5 - x^9$ which is graphed in Fig. 3.

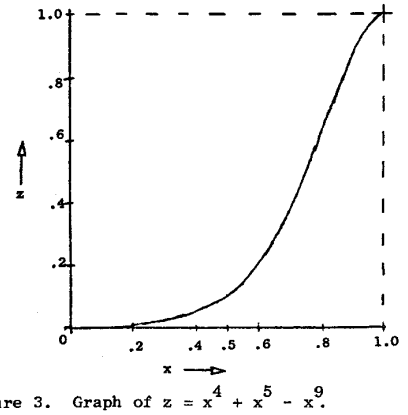

Figure 3. Graph of $z = x^4 + x^5 - x^9$.

Though not always immediately obvious from the circuit structure, the probability of detecting the fault is highly dependent upon the value of x. It can be shown that many methods of producing tests from a list of random numbers inherently fixes x at the value of 0.5 and for this example, when x = 0.5, z = 0.09, and we see in this instance why purely random test generation can be inefficient. As reported in [4], fair increases in testing efficiency can be obtained by controlling the value of x.

A further implication of Fig. 3 is that a given fault may remain desensitized for a period of time related to the input distribution. We call this time the latency period. If we compute the average latency periods for a set of faults, we arrive at a value for the mean-time-to-detection (MTTD) for the circuit as a function of input distribution. Such a measurement would be useful in evaluating test routines, the extent of coverage in a hybrid redundant circuit, the effectiveness of self testing circuits, etc. Research is proceeding in these areas.

Example 3: The Probability of Detecting a Fault, Two Methods.

(The first method.) Consider a function of a set of inputs X_i, $Z(X_i)$. For any single stuck-at type fault we can add another input F and the appropriate gating with the circuit so that when F = 0 the circuit behaves properly and when F = 1 the circuit appears to contain the specific fault. This yields a new function $\Phi(X_i, F)$. We can then use the functions Z and Φ to drive an exclusive-OR gate which produces a 1 whenever the application of an input and the presence of a fault produces an output discrepancy as shown in Fig. 4.

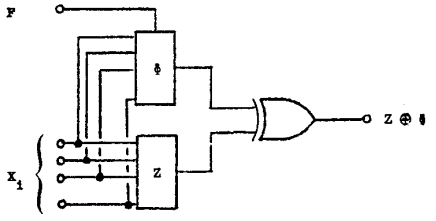

Figure 4. Circuit for producing an output discrepancy.

Then $P(Z \oplus \Phi = 1)$ is the probability that the fault is detected with respect to the input probabilities and, if desired, with respect to P(F = 1), the probability that the fault occurs. An interesting application of this model arrises when we let $0 < P(F = 1) < 1$, and consider this to be an intermittent fault. For example, let P(F = 1) = .01. Then 1 percent of the time the fault exists and yet it is still occurring randomly in time. In this way we can examine the ratio of intermittent faults to false output signals as a function of input distribution. As observed in [8] this ratio can be quite small.

(The second method.) If we continue this example setting P(F = 1) = 1, then our probability expressions for $P(Z \oplus \Phi = 1)$ can be derived from Φ by using the boolean difference [5,6]. Letting $\Delta = \partial/\partial F[\Phi(X_i, F)]$ we find that Δ is a function of the inputs X_i only. We can then derive $P(\Delta = 1) = \delta$ using the probability lemmas. Then δ is a function of the x_i values that expresses the probability of detecting the fault.

As an illustration consider the circuit of Fig. 1 with an input F added so as to produce the effect of input A stuck-at-1. The function $\Phi(A,B,C,F) = (A \vee F)(B \vee C) \vee BC$ produces the faulty function when F = 1. Now

$$\partial/\partial F[\Phi(A,B,C,F)] = \partial/\partial F[F(B \vee C)\overline{(AB \vee AC \vee BC)}]$$
$$= (B \vee C)\overline{(AB)}\,\overline{(AC)}\,\overline{(BC)} = \overline{ABC} \vee ABC = \Delta$$

and $P(\Delta = 1) = (1 - a)b(1 - c) + (1 - a)(1 - b)c$ or letting $a = b = c = x$, $P(\Delta = 1) = 2x - 2x^3$ (which is not maximized at x = .5).

Example 4: Signal Reliability.

We execute a very simple example in this section in order to emphasize the procedure and interpretation for the calculation of the signal reliability of an output of a circuit operating in the presence of faults. The concept of signal reliability is defined and explored in greater detail in [7] using some of the techniques shown in this paper. We define the signal reliability of an output as the probability that the

output of a function is still correct in the presence of some input and fault condition. For the example we shall consider the circuits of Figs. 5a and 5b which are a normal AND gate and the same AND gate with additional circuitry to affect the 3 possible stuck-at-1 faults within the gate. Lines F_1, F_2, and F_3 control the presence of the faults.

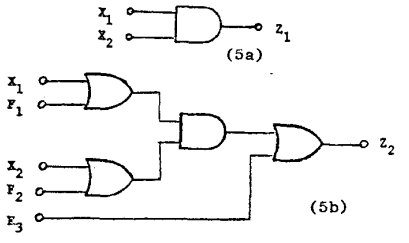

Figure 5. An AND gate and an AND gate with 3 stuck-at-1 faults.

$Z_1 = X_1X_2$ and $Z_2 = F_3 \vee (X_1 \vee F_1)(X_2 \vee F_2)$. We let function $R(X_i,F_j) = \overline{Z_1 \oplus Z_2}$ and call it the signal reliability of an AND gate with respect to the inputs X_i and faults F_j. Simplified, $R(X_i,F_j)$ is

$$X_1X_2 \vee \bar{X}_1\bar{F}_1\bar{F}_3 \vee \bar{X}_2\bar{F}_2\bar{F}_3.$$

In Table 1 is the list of inputs to $R(X_i,F_j)$ that produce 1 outputs and each is labeled as to which class it belongs to under the following bundling restraint:

$$X_1 = \{ x_1 \}, \qquad X_2 = \{ x_2 \},$$

$$\mathcal{F} = \{ F_1, F_2, F_3 \}.$$

In Table 2 the non-empty classes are listed with their weights, the number of fundamental products in each. For each class the appropriate probability expression is listed. Finally, $r(x_1,x_2,f) = P(R(X_i,F_j) = 1)$ is the sum of these weighted expressions.

Table 1.

	$X_1X_2F_1F_2F_3$	Class Name
X_1X_2	1 1 0 0 0	1 1 0
	1 1 0 0 1	1 1 1
	1 1 0 1 0	1 1 1
	1 1 0 1 1	1 1 2
	1 1 1 0 0	1 1 1
	1 1 1 0 1	1 1 2
	1 1 1 1 0	1 1 2
	1 1 1 1 1	1 1 3
$X_1F_1F_3$	0 0 0 0 0	0 0 0
	0 0 0 1 0	0 0 1
	0 1 0 0 0	0 1 0
	0 1 0 1 0	0 1 1
$X_2F_2F_3$	0 0 0 0 0	counted
	0 0 1 0 0	0 0 1
	1 0 0 0 0	1 0 0
	1 0 1 0 0	1 0 1

Table 2.

Class Name	Weight	Probability subexpression
0 0 0	1	$(1 - x_1)(1 - x_2)(1 - f)^3$
0 0 1	2	$(1 - x_1)(1 - x_2)f(1 - f)^2$
0 1 0	1	$(1 - x_1)x_2(1 - f)^3$
0 1 1	1	$(1 - x_1)x_2f(1 - f)^2$
1 0 0	1	$x_1(1 - x_2)(1 - f)^3$
1 0 1	1	$x_1(1 - x_2)f(1 - f)^2$
1 1 0	1	$x_1x_2(1 - f)^3$
1 1 1	3	$x_1x_2f(1 - f)^2$
1 1 2	3	$x_1x_2f^2(1 - f)$
1 1 3	1	$x_1x_2f^3$

$$\begin{aligned} \text{Thus } r(x_1,x_2,f) = {} & (1 - x_1)(1 - x_2)((1 - f)^3 + 2f(1 - f)^2) \\ & + (1 - x_1)x_2 \quad ((1 - f)^3 + f(1 - f)^2) \\ & + x_1(1 - x_2) \quad ((1 - f)^3 + f(1 - f)^2) \\ & + x_1x_2 \quad (1), \end{aligned}$$

when the x terms are factored left and the f terms factored right. Let us examine the behavior of $r(x_1,x_2,f)$ under various conditions.

Let $f = 0$, meaning that no faults exist in the circuit. Then $r(x_1,x_2,f) = (1 - x_1)(1 - x_2) + (1 - x_1)x_2 + x_1(1 - x_2) + x_1x_2 = 1$. This is consistant; no faults implies total reliability.

Let $f = 1$, meaning all faults have occured. Then $r(x_1,x_2,f) = x_1x_2$, and not zero. This is because the input $X_1 = X_2 = 1$ does not detect the fault.

There are four input vectors possible for this circuit; $X_1X_2 = \{00,01,10,11\}$. An interesting result arises when we assign a probability of occurrence to each. Choosing $\{ .25, .25, .25, .25 \}$* we can say that the output signal reliability of the circuit with respect to all input vectors is

$$\begin{aligned} r_t = {} & 1/4 \ ((1 - f)^3 + 2f(1 - f)^2) \\ & + 1/4 \ ((1 - f)^3 + f(1 - f)^2) \\ & + 1/4 \ ((1 - f)^3 + f(1 - f)^2) \\ & + 1/4 \ (1) \\ = {} & 1/4 + 3/4 (1 - f)^3 + f(1 - f)^2. \end{aligned}$$

Again, when $f = 0$, $r_t = 1$. When $f = 1$, then $r_t = 1/4$ which means the circuit produces the correct output 25% of the time even though all faults have occured. For any f between 0 and 1 we have the suitable expression for predicting the probability of correct output.

Since any distribution of probabilities for the input signals could have been used we have the flexibility to predict the signal reliability of the output when some input vectors are more likely than others. Finally, the value of f in r_t could be replaced by a time function such as $f = 1 - e^{-\alpha t}$ where α is the decay parameter. Thus, at $t = 0$ the value of r_t is one and decays with time thereafter.

The amount of calculation in this simple example would seem discouraging but it is shown in [7] that the amount of work for all types of faults in a TMR system is roughly the same. We point out that the use of bundling in this example had the effect of reducing the complexity of our expressions by eliminating some variables. Essentially, the assumption was made that all faults were equally likely. However, we need not

*Care must be taken that some assignment of $P(X_1=1)$ and $P(X_2=1)$ will produce the desired probabilities of input vectors.

use this restriction if we desired to give each fault an independent probability of occurrence. With this option and the freedom to choose our own distributions for the probabilities of inputs, we see we have a powerful analysis technique.

CIRCUITS WITH SPECIAL PROPERTIES

Some circuits have properties in their implicant composition that make the application of the probability lemmas quite easy. The next two examples are such circuits. The final example deals with iterative networks.

Example 5: Threshold Functions.

Consider a 3-out-of-5 threshold function or voter. Bundling all 5 inputs into bundle X (with probability x) we note the regular pattern in the fundamental products. All ten fundamental products (each with the same probability expression) that recognize 3 ones in 5, all five fundamental products recognizing 4 ones in 5 and the one fundamental product recognizing 5 ones in 5 are summed so the probability expression is $z = 10x^3(1 - x)^2 + 5x^4(1 - x)^1 + x^5$. Note that a compact notation for this equation would be a linear vector of 6 elements containing the weights of each class of fundamental product. In this case the vector would be $< 0,0,0,10,5,1 >$.

Example 6: A Large Parity Generator.

Consider a 9 bit odd-parity generator. Using a 9 signal bundle X (with probability x) we can represent the probability expression with the same vector notation used in Example 5 as:

$$< 1,0,36,0,126,0,84,0,9,0 >.$$

The 5th entry is 126 because there would be 126 fundamental products in the table of combinations where 4 out of 9 inputs are 1; $126 = \binom{9}{4}$; however, the 6th term is zero since for no input having 5 out of 9 ones does the circuit produce a 1-output. In this example bundling allowed us to easily treat a circuit with 9 inputs.

Example 7: Unilateral Iterative Networks.

Consider a unilateral iterative cell with output $C_{i+1} = G(X_i, C_i)$ within a network of cells with a fixed (boundary) value C_0. Since each X_i is independent, the probability expression for the 0th cell is $c_1 = g(x_0, c_0)$ where c_0 is the boundary probability. All subsequent cell outputs can be expressed by the recursion formula $c_{i+1} = g(x_i, c_i)$. In the case of a full adder, $C_{i+1} = A_iB_i \vee A_iB_iC_i \vee A_iB_iC_i$ and $c_{i+1} = a_ib_i + c_i(a_i + b_i - 2a_ib_i)$. Hence using an appropriate bundling restraint we can treat an arbitrarily large parallel adder with comparable ease. Since it is well known that many sequential circuits can be modeled with unilateral iterative networks, the probabilistic approach may become useful for sequential analysis.

CONCLUSION

We have demonstrated a method for producing probability expressions for general combinational networks which relates the output probability to the set of input probabilities. The formulation is consistent with the requirements of probability theory and is shown to be immediately applicable to many areas of fault tolerant computing. In the examples we have shown that the probabilistic approach is general and has great latitude of interpretation. We have illustrated the treatment of large non-trivial circuits with the simplification technique we call bundling. Finally we note that all formula derivations are executed with simple and straightforward steps, i.e., there are no quasi-heuristic bottlenecks in any of the formulation procedures.

Further work in this area is currently under way. An examination of algorithm 1 shows that the amount of memory needed by a suitable computer program can increase exponentially with the number of inputs to the circuit. Recent work has revealed a new process for calculating probability expressions that proceeds from inputs to outputs in gate by gate fashion, in an amount of time roughly linear in the number of gates. Indeed, one circuit having 18 inputs was analyzed rather easily using only paper and pencil. With the ability to handle larger circuits, more ambitious questions can be approached regarding fault latencies, intermittent faults, signal reliabilities, etc.

ACKNOWLEDGMENT

Helpful comments were garnered during active discussions with J. Abraham, E. Fregni, S. Kolupaev, J. Losq, R. Ogus, J. Shedletsky and J. Wakerly.

REFERENCES

[1] Papoulis, A., Probability, Random Variables, and Stochastic Processes, McGraw-Hill Book Co., New York, 1965.

[2] McCluskey, E. J., Introduction to the Theory of Switching Circuits, McGraw-Hill Book Co., New York, 1965.

[3] Bastin, D., E. Girard, J. C. Rault, and R. Tullioue, "Probabilistic Test Generation Techniques," Digest of Papers 1973 International Symposium on Fault Tolerant Computing, p. 171.

[4] Parker, K. P., "Probabilistic Test Generation," Technical Note 18, Digital Systems Laboratory, Stanford University, Stanford, California, August 1973.

[5] Akers, S. B, Jr., "On a Theory of Boolean Functions," J. Soc. Indust. Appl. Math., Vol. 7, No. 4, December 1959.

[6] Sellars, F., M. Hsiao, and L. Bearnson, "Analysing Errors with the Boolean Difference," IEEE Trans. on Comp., July 1968.

[7] Ogus, R. C., "The Probability of a Correct Output from a Combinational Circuit," presented to the 4th Sumposium on Fault Tolerant Computing, June 1974.

[8] Ball, M. and F. Hardie, "Effects and Detection of Intermittent Failures in Digital Systems," Fall Joint Computer Conference, 1969, p. 329-335.