POWERTEST: A Tool for Energy Conscious Weighted Random Pattern Testing

Xiaodong Zhang, Kaushik Roy, and Sudipta Bhawmik†
EE Building, Purdue University, West Lafayette, IN 47907
†Lucent Technologies, Princeton, New Jersey

Abstract— **Due to the increasing use of portable computing and wireless communications systems, energy consumption is of major concern in today's VLSI circuits. With that in mind we present an energy conscious weighted random pattern testing technique for Built-In-Self-Test (BIST) applications. Energy consumption during BIST operation can be minimized while achieving high fault coverage. Simple measures of observability and controllability of circuit nodes are proposed based on primary input signal probability (probability that a signal is logic ONE). Such measures help determine the testability of a circuit. We developed a tool, POWERTEST, which uses a genetic algorithm based search to determine optimal weight sets (signal probabilities or input signal distribution) at primary inputs to minimize energy dissipations. The inputs conforming to the primary input weight set can be generated using cellular automata or LFSR (Linear Feedback Shift Register). We observed that a single input distribution (or weights) may not be sufficient for some random-pattern resistant circuits, while multiple distributions consume larger area. As a trade-off, two distributions have been used in our analysis. Results on ISCAS benchmark circuits show that energy reduction of up to 97.82% can be achieved (compared to equi-probable random-pattern testing with identical fault coverage) while achieving high fault coverage.**

Keywords— **Testing, Random Testing, Weighted Random Pattern Testing, Energy Conscious Testing.**

I. Introduction

With the increasing use of portable computing and wireless communications, energy dissipation has been a major concern in today's VLSI. For example, the cost constraints of consumer electronic products typically require plastic package, which imposes a strong limit on the energy dissipation. The market acceptance of mobile applications and notebook computers also depends on the operation time per battery pack. Therefore, the reduction of the electrical energy consumption has become one of the most rapidly growing topics of interest in the electronics industry and one of the most challenging areas of research in this domain. In order to meet the energy and reliability constraints of these products, an energy conscious Built-In-Self-Test(BIST) technique is necessary. Since BIST is generally used during regular operation of a circuit or during power on, low-power BIST would enhance battery life.

CMOS technologies have very low power consumption when signals are not switching. The majority of the energy dissipation in the current day technology is due to charging and discharging of load capacitance of logic gates, which occurs when logic gates undergo signal transitions. Hence in order to reduce power consumptions in BIST, it is desirable to reduce the switched capacitance during the test mode.

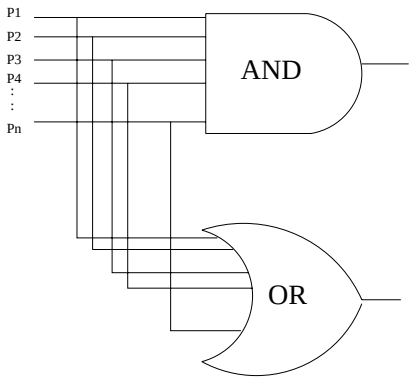

Fig. 1. n-input AND gate and n-input OR gate

Random pattern testing is very attractive in BIST because it does not require large memory overhead while achieving high fault coverage with a limited number of input vectors [5], [6], [1], [7], [8]. Energy consumption during test is proportional to the product of the test length and the average dynamic power. Therefore for random-pattern resistant circuits, energy issues are especially important. For such circuits, to assure high fault coverage, the required test sequences can be long. To handle such problems, usually multiple sets of weights (signal probabilities or input signal distributions) are used [1]. That requires extra silicon area for the generation of such weights. Hence, it is desirable to use a single weight set by globally optimizing the weight set for acceptable performance [5]. Let us consider the circuit shown in Figure 1, where an n-input AND gate and an n-input OR gate are connected to the same set of primary inputs. The optimal weight set with a single distribution is the equiprobable set [1] (since AND and NOR gates have conflicting testability requirements), and it requires prohibitively large numbers of equiprobable patterns to achieve high fault coverage. In this paper, we use two separate weight sets to trade-off memory requirement versus testability and energy dissipation.

Energy consumption during test is proportional to the product of test length and average dynamic power.There are two approaches to reduce energy: the first one is to reduce the test length by finding the optimal signal probabilities such that the test length is minimum; the other one is to reduce the dynamic power by reducing the signal activities (probabilities of signal switching). While the test length depends on signal probabilities and activities, our experience shows that it is not very sensitive to signal

This research is supported in part by DARPA and by Lucent Technologies

activities. Fault simulations show that the test length is weakly sensitive (for same fault coverage) to the signal activities in combinational circuits if we keep the same signal probabilities. In this paper, we determine signal probabilities and optimal activities of primary inputs to achieve lowest energy during the self test mode. Such probabilities and activities are then used to generate weighted random patterns which can achieve high fault coverage while being energy efficient. The corresponding weighted random pattern generator can be implemented using a one-dimensional cellular automata or LFSR (Linear Feedback Shift Register).

The rest of the paper is organized as follows: Section II introduces the formal definitions of weight set, signal probability, signal activity and other related definitions. Section III proposes a simple and efficient technique to generate and evaluate the weight sets at the primary inputs. Section IV describes a genetic algorithm based technique to obtain weights optimized for high testability and low energy dissipation. Experimental results for a number of benchmark circuits are presented in Section VI. Finally, Section VII gives the conclusions.

II. Preliminaries and Definitions

Definition 1 (**Primary Input Weight Set**) For a circuit with n primary inputs, a weight set S_P assigned to its inputs is a vector $(p(1), p(2), ..., p(n))$, where $p(i) \in [0,1]$ is the probability that input i is a logic ONE. $p(i)$ is also called the weight assigned to primary input i.

Definition 2 (**Signal Probability**) Signal probability $P(k)$ of a node k of a circuit is the probability that node k is logical ONE if the circuit is stimulated by a random sequence implementing weight set S_P.

Definition 3 (**Signal Activity**) The *activity* $A(x)$ of a signal $x(t)$ is defined as $\lim_{\tau\to\infty} \frac{n_x(\tau)}{\tau}$, and equals to the expected value of $\frac{n_x(\tau)}{\tau}$. The variable n_x is the number of switching of $x(t)$ in the time interval $(-\tau/2, \tau/2]$. The *normalized activity* $a(x)$ is defined as $A(x)$ divided by the clock frequency f, and is the probability for the signal to switch within a clock period, and is given by:

$$a(x) = P(x(t-T)\bar{x}(t) + \bar{x}(t-T)x(t))$$

where $x(t-T)\bar{x}(t)$ denotes a switching transition from 0 to 1 while $\bar{x}(t-T)x(t)$ denotes a switching transition from 1 to 0 (T is the clock period). We assume, without loss of generality, that the signal switches at the rising edge of the clock cycle. Therefore, we have, $1 - a(t)$ as the probability that the signal does not switch, and is given by:

$$1 - a(x) = P(\bar{x}(t-T)\bar{x}(t) + x(t-T)x(t))$$

In this paper, we assume the probability and activity of signal $x(t)$ do not change with time, hence, $P(x(t))$ and $a(x)$ are constants with respect to time.

The *activity* $a(y_j)$ at the output node y_j of the module is given by

$$a(y_j) = \sum_{i=1}^{n} P(\frac{\partial y_j}{\partial x_i}) a(x_i) \qquad (1)$$

where $\partial y/\partial x$ is the Boolean difference of function y with respect to x and is defined by

$$\frac{\partial y}{\partial x} = y\mid_{x=1} \oplus y\mid_{x=0} = y(x) \oplus y(\bar{x}) \qquad (2)$$

Given the primary input probabilities and activities, we can use equation 1 to recursively calculate the switching activities at the internal nodes [2], [17], [19], [18]. It should be noted that for testing of combinational circuits, temporal correlations are not important. However, they do play a role when we trade-off test time versus energy dissipation.

Definition 4 (**Activity Set**) For a circuit with n primary inputs, an activity set S_A assigned to its inputs is a vector $(a(1), a(2), ..., a(n))$, were $a(i)$ is the activity of i-th primary input.

the average power for a CMOS circuit can be approximated by

$$Power_{avg} = \frac{1}{2} V_{dd}^2 \sum_{j \in all\ nodes} C(j)\ A(j) \qquad (3)$$

where V_{dd} is the supply voltage, $C(j)$ is the node capacitance, and $A(j)$ is the activity at node j and is proportional to the *normalized activity* $a(j)$. The node capacitance $C(j)$ is approximately proportional to the fan-outs at node j. So we can define the *normalized power dissipation measure* Φ as:

Definition 5 (**normalized power dissipation measure**)

$$\Phi = \sum_{j \in all\ nodes} f_{anout}(j)\ a(j) \qquad (4)$$

where $f_{anout}(j)$ is the number of fan-outs at node j.

Definition 6 (**Power Sensitivity**) To measure the change in average power due to the changes in the distributions of primary inputs, we define *power sensitivity to primary input activity* $S_{a(x_i)}$ as follows:

$$S_{a(x_i)} = \lim_{\Delta a(x_i)\to 0} \frac{\Delta Power_{avg}}{\Delta a(x_i)} = \frac{\partial Power_{avg}}{\partial a(x_i)} \qquad (5)$$

where $a(x_i)$ is the activity of primary input x_i.

Definition 7 (**Normalized Power Sensitivity**) Since $Power_{avg}$ is proportional to *normalized power dissipation measure* Φ, we can define *normalized power sensitivity to primary input activity* $\zeta_{a(x_i)}$ in terms of Φ as follows:

$$\zeta_{a(x_i)} = \frac{\partial \Phi}{\partial a(x_i)} = \sum_{j \in all\ nodes} f_{anout}(j)\ \frac{\partial a(j)}{\partial a(x_i)} \qquad (6)$$

where $a(j)$ is the activity of each internal node or primary output. The term $\partial a(j)/\partial a(x_i)$ can be referred as activity sensitivity.

III. Determination of Weight Set

A. Estimation of the signal probabilities and activities

In this paper, we use the general algorithm proposed in [2] [3] and adopt a data structure similar to [4] to estimate

signal probabilities and activities at the internal nodes of the circuit.

It has been shown in [3] that the signal probability $P(f)$, can be expressed as a sum of *primary input signal probability product terms* $\sum_{i=1}^{p} \alpha_i (\prod_{k=1}^{n} P^{m_{i,k}}(x_k))$, where n is the number of the independent inputs to the circuit, and α_i is some integer. The exponent $m_{i,k}$ is either 1 or 0. The sum has p product terms. This form will be referred to as the *sum of probability products* of f, which is a form of *symbolic signal probability*. Also for convenience, $\bar{P}(x_i)$ is defined as $P(\bar{x_i})$ and equals $1 - P(x_i)$. Therefore, $P(f)$ can be expressed as $\sum_{i=1}^{p} \alpha_i (\prod_{k=1}^{n} P^{m_{i,k}}(x_k) \bar{P}^{l_{i,k}}(x_k))$, where $m_{i,k}$ and $l_{i,k}$ are either 1 or 0 but both cannot be 1 simultaneously. Since

$$P(x_k)\bar{P}(x_k) = P(x_k)(1 - P(x_k)) = P(x_k) - P^2(x_k)$$

and equals 0 after exponent suppression [3], [4], the product term $P^{m_{i,k}}(x_k)\bar{P}^{l_{i,k}}(x_k)$ will be eliminated from $P(f)$ if $m_{i,k} = l_{i,k} = 1$. We first find the independent inputs of each Boolean function f. Then we can use the above approach to calculate the symbolic probability. [2].

Signal activity estimation is shown in Section II and is the same as described in [2].

B. Estimation of the Power Sensitivity

We use symbolic technique to obtain power sensitivities, and the basic idea is to express the activity of each internal node or primary output in terms of the probabilities and activities of primary inputs. After we obtain the exact expression for signal activity, we can easily compute power sensitivities by equations 6.

Note that accurate calculation of power sensitivity depends on whether we can accurately express the probability and activity of each internal node and primary output in terms of independent inputs. However, the size of symbolic probability expression and activity expression grow exponentially with respect to the number of independent inputs. Consequently, we resort to circuit partitioning to trade-off accuracy versus computation time. The circuit partitioning technique is similar to the one given in [21].

C. Measurement of Controllability and Observability

Our primary aim is to determine the weights (signal probabilities) at primary inputs which can achieve a high fault coverage while being energy efficient. To define stuck-at fault testability of a node, we define controllability and observability. The controllability measure of a node is defined as the relative difficulty of setting a node to logic ONE or ZERO. For a node k, if $P(k)$ is around 1.0 or 0.0, then the node is either very difficult to set to ZERO or to ONE, and the controllability of that node is very low, while $P(k) = 0.5$ implies that node k can be set to ZERO or ONE with equal ease (hence, high controllability). The observability measure of a node is defined as the relative difficulty of propagating an error from the node to a primary output [11]. It is apparent that the observability and

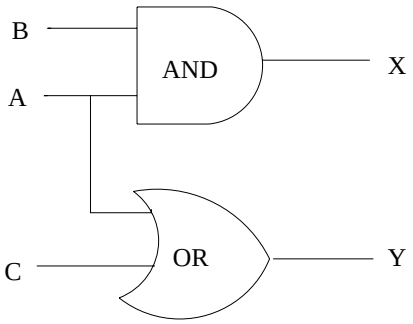

Fig. 2. observability of a node

controllability measures of a node determine if the node is easily testable for stuck-at faults. For a node A, let us denote the difficulty of setting A to ONE or ZERO as $C_1(A)$ or $C_0(A)$, respectively. Let the observability of node A be $O(A)$. Then we have (considering Figure 2),

$$O(B) = C_1(A) + O(X)$$

$$O(C) = C_0(A) + O(Y)$$

If $P(A)$ is near 0.0, then not only the controllability of A is low, but also the observability of B is low, since B is very difficult to be observed at X. If $P(A)$ is near 1.0, then both the controllability of A and observability of C are low. On the other hand, if $P(A) = 0.5$, then the observabilities of both B and C are average, while the controllability of A is high. Since we want to optimize the controllability and observability of the entire circuit, a signal probability of 0.5 is desirable for each node of the circuit from testability point of view.

Now let us consider stuck-at fault testability using random patterns. We need an optimization procedure which determine an input weight set such that the internal node signal probabilities are close to 0.5 to achieve high controllability and observability. We define a cost function for node k such that:

- The cost function is proportional to the difficulty of controlling and observing node k.
- The cost function is maximum when $P(k) = 0$ or $P(k) = 1$, since the controllability of the node is lowest when $P(k) = 0$ or $P(k) = 1$.
- The cost function should distinguish the relative difficulty between two possible probabilities of node k. For example, if $P_1(k) = 0.01$ and $P_2(k) = 0.001$, or if $P_3(k) = 0.99$ and $P_4(k) = 0.999$, the absolute differences between $P_1(k)$ and $P_2(k)$ or between $P_3(k)$ and $P_4(k)$ are small, but the controllability of node k is much worse when $P_2(k) = 0.001$ or $P_4(k) = 0.999$ than when $P_1(k) = 0.01$ or $P_3(k) = 0.99$. A possible solution to this problem is to take logarithm of the probability: $-log(P_1(k))$ and $-log(1.0 - P_3(k))$ are equal to 2, and they are much smaller than $-log(P_2(k))$ or $-log(1.0 - P_4(k))$, which are equal to 3. Hence by taking logarithm, the cost function can distinguish the

relative difficulty between two probabilities with absolute value near 0.0 or 1.0.

In order to meet the above objectives, we define the cost function for testability as follows:

Definition 8 (**Cost Function for Testability**)

$$cost(k) = \begin{cases} -log_{10}P_{tl} & : \quad P(k) < Pt_l \\ -log_{10}P(k) & : \quad P_{t_l} < P(k) < P_{t_1} \\ 0 & : \quad P_{t_1} < P(k) < P_{t_2} \\ -log_{10}(1-P(k)) & : \quad P_{t_2} < P(k) < P_{t_h} \\ -log_{10}(1-P_{th}) & : \quad P_{t_h} < P(k) \end{cases}$$

$$COST_{test} = \sum_{k \in allnodes} cost(k)$$

In the cost function, P_{t_1}, P_{t_2}, Pt_l and P_{t_h} are defined as testability thresholds, and will be explained in detail in this section. It should be noted that the cost is a single number and can mask the conflicting nature of the internal nodes during optimization. Two nodes A and B can be conflicting in the sense that an optimization motivated by this cost function improves the testability of A while degrading B. There are instances where the testability of a circuit is hindered by a relatively small number of hard-to-detect faults. If B is a random-pattern resistant node, while A is easily detectable, then B can dominate A for determining the required test length. Node B may not be optimized in the presence of node A if they are conflicting. Hence, a lower weightage can be given to those stuck-at faults which are "easy-to-detect". Our cost function neglects all easy-to-detect faults, and enhances the probability of generating suitable patterns to expose the pseudo-random-resistant faults, thus making it possible to achieve high fault coverage with shorter test length.

Definition 9 (**Testability Threshold**) The testability thresholds P_{t_1}, P_{t_2}, P_{t_l} and P_{t_h} of an internal node are defined as follows. For any internal node A, if

$$P_{t_1} < P(A) < P_{t_2}$$

we say any stuck-at fault in node A is easily detectable and can be neglected in the cost function. If $P(A) < P_{t_l}$ or $P(A) > P_{t_h}$, we say the stuck-at faults in node A is hard to detect, and the cost function for node A is heuristically set to constants, namely, $-log_{10}P_{t_l}$ and $-log_{10}(1-P_{t_h})$, respectively.

For a test sequence containing several hundred vectors, if a node has a signal probability between 0.1 and 0.9, then the node will be driven to logic ONE or ZERO several hundred times, and both s-a-0 or s-a-1 faults in the node have very high chance of being detected. This node can be categorized as an easy-to-detect node, and it must be neglected in the cost function. Typical values for P_{t_1} and P_{t_2} are set to 0.1 and 0.9, respectively, when we test the circuit with several hundred vectors.

Due to the complex nature of the optimization problem, the testability of some random-pattern resistant faults may not be maximum even when the global optimal point of the cost function is obtained. There are situations where these extremely random-resistant faults may interfere with the testability of other nodes during the optimization process. This can be better understood by considering a simple example. Since the optimization algorithm will minimize the cost function, it is possible that the signal probability of a random resistant node A changes from 0.0000001(say 10^{-7}) to 0.00001(say 10^{-5}) during optimization. Then the contribution of node A to the cost function will decrease rapidly from 7 to 5, but the stuck-at-0 fault at node A may remain undetectable if we test the circuit with several hundred vectors. However, this rapid decrease of cost(A) may sacrifice the testability of some other nodes. This situation should be avoided since it actually reduces the testability of the circuit. Testability thresholds P_{t_l} and P_{t_h} help mitigate the above problem. P_{t_l} and P_{t_h} in the cost function are defined as follows: for a node A, if $P(A) < P_{t_l}$ or $P(A) > P_{t_h}$, we say the stuck-at faults at node A is hard to detect, and the cost function of node A is heuristically set to constants, namely, $-log_{10}P_{t_l}$ and $-log_{10}(1-P_{t_h})$, respectively. For example, if we test the circuit with several hundred vectors, the typical values of P_{t_l} and P_{t_h} can be $P_{t_l} = 0.001$ and $P_{t_h} = 0.999$. Hence, if $P(A)$ changes from $0.0000001(10^{-7})$ to $0.00001(10^{-5})$, the testability measure of node A will remain unchanged during the optimization, thereby this extremely-hard-to-detect fault will not be able to interfere with the testability of other nodes.

Our testability measure is much simpler and more straightforward than those based on fault-detection probabilities [6], and can be efficiently calculated from signal probabilities.

It is worth noting that this cost function is only related to the signal probabilities, and temporal correlations among signals are not important for the testability of combinational circuit. We also notice that the test length is weakly dependent on signal activity (temporal correlation among signal). Therefore, the optimization of testability and power can be two separate processes: probabilities can be optimized to achieve certain fault coverage with shortest test length, while temporal correlations can be optimized to achieve lowest energy with the same test length.

D. Estimation of Total Switched Capacitance

Since we want to optimize the activities for low power, we need to estimate the dynamic power which is proportional to the total switched capacitance. If the signal activity of node k is $a(k)$, then the average switched capacitance at that node is equal to $a(k)C_k$, where C_k is the capacitance associated with node k. C_k is approximately proportional to the fanout at node k.

The summation of the switched capacitance over all nodes of the circuit is proportional to the average dynamic power dissipation and is given by:

$$\sum_{k \in allnodes} a(k) fanout(k) \tag{7}$$

The energy dissipation during the test mode is proportional to the product of the average switched capacitance and test length, since the average switched capacitance

is proportional to the dynamic power consumption during each clock cycle, while test length indicates how many clock cycles the test lasts.

When we optimize the probabilities of primary inputs for testability, we assume temporal independence, then for node k, the probability of signal switching equals to $2p(k)(1-p(k))C_k$. Let us denote this activity (without temporal correlations) as $\alpha_0(k)$, When we optimize the activities of primary inputs separately to achieve minimum energy, we should take the signal correlation into consideration, and the activity of the internal nodes can be estimated by our power estimation technique described in Section II

Equation 7 which gives the total switched capacitance should be modified when two weight sets are applied to the primary inputs of the circuit. Two weight sets are necessary when the circuit is random pattern resistant. Let us denote the vectors generated by the first weight set as U and the vector generated by the second weight set as V. We merge the vectors of these two weight sets by using the first weight set to generate the first n vectors, then the second one for the second n vectors, and so on :

$$U_1U_2....U_n \qquad V_1V_2....V_n \qquad ....$$

Given two weight sets, let us denote the activities of node k as $a_1(k)$ and $a_2(k)$, respectively. The average switched capacitance is then given as:

$$\sum_{k \in allnodes} [a_1(k) + a_2(k)] fanout(k)/2$$

Power consumption is a function of primary input activities.

Given the dynamic power and requred test length, the energy consumption can be determined as the product of power and test length, To determine the energy consumption during test mode, for each set of primary input activities, we use a fault simulator to determine the required test length, and employ the power estimation tools described in [2] to accurately calculate the corresponding dynamic power. Minimum energy consumption can be achieved by searching the best activities.

IV. Optimization of Probabilities

In order to optimize the primary input signal probabilities to achieve high fault coverage, we need to globally minimize the cost for testability. The temporal correlations of primary inputs are neglected here because temporal correlations are not important for the testability of combinational circuits. The cost function for testability is a function of signal probabilities, but it is independent of temporal correlations. Genetic Algorithm (GA) [13] is employed to search the best primary input probabilities such that the testability cost function is minimum. GA is well-suited to this problem because of the complex nature of the cost function.

For a circuit with n primary inputs, the optimization problem requires a search for the n-tuple $(p(1), p(2), ..., p(n))$. This n-tuple is encoded as single binary string of length nk, such that each $p(i)$ is encoded as a segment of length k. The population size is typically set to 100. Each individual has an associated fitness, which is the testability cost function. This will drive the genetic algorithm to optimize the controllability and observability of the circuit.

The population is first initialized with random strings. The fitness function is then calculated based on the signal probability of internal nodes. The evolutionary processes include selection, 1-point crossover, 2-point crossover, uniform crossover, multi-point crossover, mutations and inversions. Mutation probability is set to 0.01, and crossover probability is set to 0.25 [14].

V. Optimization of Activities

After the probabilities of primary inputs have been optimized for testability, we fix them to their optimal values. The activities can be optimized subsequently to achieve minimum energy for the same fault coverage. In this case, the n-tuple to be optimized is $(a_1, a_2, ..., a_n)$, where a_i is the activity of the i-th primary input $(i = 1, 2, ..., n)$. The optimization of activities is complex if we search each a_i individually. There are two approaches to simplify this optimization problem: the first one is to reduce the activities of each primary input uniformly, and the second one reduces the activity of each primary input according to its power sensitivity.

A. Uniformly Reducing Activities

Let us denote the activities without temporal correlations as $(\alpha_{0_1}, \alpha_{0_2}, ..., \alpha_{0_n})$. We multiply each activity by a factor f_{act} such that,

$$\begin{aligned} a_1 &= f_{act}\alpha_{0_1} \\ a_2 &= f_{act}\alpha_{0_2} \\ &..... \\ a_n &= f_{act}\alpha_{0_n} \end{aligned} \tag{8}$$

Hence each value of f_{act} defines an activity set. We can determine the optimal activity set by searching the optimal value of f_{act} which gives lowest energy consumption while achieving the same stuck-at fault coverage. For each value of f_{act} (or activity set), we use a fault simulator to determine the required test length for the same fault coverage, and employ power estimation tools described in [2] to accurately calculate the corresponding dynamic power. Since energy is proportional to the product of dynamic power and test length, it is a function of f_{act}. Consequently, an optimal value can be determined for f_{act}.

B. Reducing Activities According to Power Sensitivities

We can also reduce the primary input activities according to the power sensitivities. This optimization process can be better understood with the help of power surface.

Power surface can be constructed by plotting the relationship between average power and primary input activities(the primary input probabilities are fixed to their optimal values), and it is intrinsically $(n+1)-$dimensional, since n dimensions are required to represent the activities of n primary inputs.

In the power surface plot, let us use A to denote the point which represents the case without temporal correlations (with primary input activities $\alpha_{0_1}, \alpha_{0_2}, ..., \alpha_{0_n}$). Let O represent the origin. Our optimization process starts from point A. If we draw a line AO between points A and O, then uniformly reducing activities implies that we search for the minimum energy along the line AO. This optimization process, however, can be improved by using the information of power sensitivities, which gives the direction of the largest gradient on the power surface. Reducing activities according to power sensitivities means that we search for the minimum energy on the power surface along the direction of the largest gradient.

VI. Experimental Results

The tool **POWERTEST** to determine weight sets has been implemented in C under the Berkeley SIS environment[15], [16]. We adopt a data structure similar to [4] to estimate signal probabilities at the internal nodes of the circuit. **POWERTEST** takes a circuit description file and a parameter $num_{generation}$ as inputs, where the circuit description file is of Berkeley BLIF format. The parameter $num_{generation}$ will determine how many generations the genetic algorithm will evolve. The outputs of **POWERTEST** are the two optimal weight sets.

After we find the optimal weight sets by **POWERTEST**, we generate the random patterns conforming to the optimal weight sets. Then we apply those random patterns to a fault simulator. The fault simulator we used is HITEC/PROOFS package from UIUC, which supports gate types of AND, NAND, OR, NOR, INVERTER and BUFFER [12], [20]. Consequently, all the benchmark circuits are technology mapped to circuits consisting of only AND, NAND, OR, NOR, INVERTER and BUFFER.

The fault coverage and required test length are different if we optimize the weight set with different testability cost function. Table I shows the effect of the choice of cost functions on the test length for ISCAS circuit C432. The test length in the table is the required vector length to detect 472 faults in C432. If the cost function contains both easy-to-detect nodes and hard-to-detect nodes, then the easy nodes become easier to detect, while the hard-to-detect nodes become even more difficult. The required test length of the circuit increases, since it is determined by the relatively small number of random-resistant nodes. The second column of Table I is the test length when primary inputs (PI) are equiprobable, the third column is the test length for weighted random patterns with the cost function containing easy-to-detect nodes, the fourth column is for the case when the cost function neglects those easy-to-detect nodes while only considering hard-to-detect nodes, the fifth column represents the test length with optimized activities for low energy consumption. We can see that the test length can be larger if we include the easy-to-detect nodes in the cost function.

After the optimal primary input probabilities have been determined by **POWERTEST**, we can optimize the signal activities of each primary input while keeping the signal probabilities unchanged. The first method to simplify this optimization problem is to reduce each activity uniformly by a multiplying factor f_{act} and the second method is to search on the power surface along the direction of the largest gradient. We can search for lowest energy consumption while fixing the required fault coverage to a certain value. The energy dissipation is proportional to the product of dynamic power and test length, where test length is determined by the fault simulator, and the power is calculated by the power estimation tools. The value of primary input activities are optimal when energy reaches minimum.

The summary of energy reduction of ISCAS benchmark circuits is given in Table II. The second column of Table II refers to the number of faults we want to detect by either equiprobable or weighted random sequences, the third and fourth columns show the results for equi-probable case. Column five through seven present the results of reducing activities according to power sensitivities, and column eight through eleven present the results of uniformly reducing primary input activities. The column "length" and "power" represents required test length and dynamic power, respectively, and column "energy reduction" shows the energy reduction compared to the equi-probable case,

VII. Conclusions

Low power BIST is important for battery operated systems. In this paper we have shown that proper selection of weights to the primary inputs of a combinational logic can be effective in reducing the energy consumption while achieving high fault coverage. The weights can be generated using LFSR (Linear Feedback Shift Register) or one dimensional cellular automata. Our results on ISCAS benchmark circuits show that an energy reduction of up to 97.82% can be achieved, compared to equiprobable random pattern testing.

Acknowledgments

This research was supported in part by Lucent(670-1285-4016) and DARPA (F33615-95-C-1625).

References

[1] H. Wunderlich, " Multiple Distributions for Biased Random Test Patterns" *IEEE Transactions on Computer-Aided Design*, Vol9, No. 6, June 1990, pp. 584-593.

[2] T.-L. Chou and K. Roy, *Estimation of Activity for Static and Domino CMOS Circuits Considering Signal Correlations and Simultaneous Switching* , IEEE Transactions on Computer-Aided Design of Integrated Circuits, October 1996, pp. 1257-1265.

[3] K. P. Parker and E. J. McCluskey, "Probabilistic Treatment of General Combinatorial Networks," *IEEE Trans. on Computers*, vol. C-24, Jun. 1975, pp. 668-670.

[4] K. Roy and S. Prasad, "Circuit Activity Based Logic Synthesis for Low Power Reliable Operations," *IEEE Trans. on VLSI Systems*, Dec. 1993, pp. 503-513.

TABLE I

EFFECT OF COST FUNCTION (C432)

	equiprobable	all nodes	easy nodes neglected	easy nodes neglected(activities optimized)
length	588	1998	508	521

TABLE II

Cir--cuit	# of faults detected /(total#)	equi-probable		by power sensitivity			Uniformly		
		length	power	length	power	energy reduction	length	power	energy reduction
C432	472/(512)	586	145.3	537	52.4	66.83%	521	62.8	61.5%
C499	1248/(1262)	1185	369.7	988	173.8	60.8%	1246	161.9	53.5%
C880	896/(912)	1555	264.3	485	51.1	94.0%	702	54.0	90.77%
C1908	1435/(1575)	978	447.64	493	116.3	86.9%	723	99.1	83.6%
C1355	1580/(1606)	1001	398.2	863	121.0	73.8%	1145	117.3	66.3%
C2670	1855/(2280)	145	722.41	168	265.7	57.6%	210	222.9	55.31%
C3540	2696/(2959)	8685	813.78	496	297.6	97.92%	521	295.29	97.82%
C5315	4723/(4826)	472	1657.9	778	398.8	60.3%	836	412.5	56%
C6288	7616/(7618)	130	2074.21	93	1658.7	42.8%	92	1809.3	38.3%
C7552	6070/(6744)	162	2207.66	1983	52.7	70.8%	2064	51.1	70.5%

[5] M. A. Miranda, Carlos A. Lopez-Barrio, " Generation of Optimized Single Distributions of Weights for Random Built-In Self-Test " *International Test Conference 1993*, Dec. 1993, pp. 1023.

[6] H. Wunderlich, "Protest: A Tool for Probabilistic Testability Analysis" *22nd Design Automation Conference* PP. 204-211.

[7] A. Majumdar, " WRAPTure: A Tool for Evaluation and Optimization of Weights for Weighted Random Pattern Testing", *Proceedings - IEEE international Conference on Computer Design: VLSI in Computers and Processors 1994* pp. 288-291.

[8] J.A. Waicukauski, etc. " A Method for Generating Weighted random Test Patterns" *IBM Journal Research Develop.* Vol 33, No.2, March 1989, pp. 149-160.

[9] F.N. Najm, "Transition Density: A New Measure of Activity in Digital Circuits," *IEEE Trans. on Computer-Aided Design*, vol. 12, no. 2, Feb. 1993, pp. 310-323.

[10] A. Ghosh, S. Devadas, K. Keutzer, and J. White, "Estimation of Average Switching Activity in Combinational and Sequential Circuits," *ACM/IEEE Design Automation Conf.*, 1992, pp. 253-259.

[11] M. Abramovici,"Digital Systems Testing and Testable Design", 1990 Computer Science Press, pp214.

[12] T. M. Niermann, W. Cheng, and J. H. Patel, "PROOFS: A fast, memory efficient sequential circuit fault simulator", Proceedings of the ACM/IEEE Design Automa pp. 535-540, June 1990.

[13] D.E. Goldberg, "Genetic Algorithms in search, Optimization, and Machine Learning, Reading", 1989 Addison-Wesley.

[14] "Genetic algorithms in engineering and computer science", Wiley, c1995.

[15] R.K. Brayton, G.D. Rudell, A. Sangiovanni-Vincentelli and A.R. Wang, "MIS: A Multiple-Level Logic Optimization System", IEEE Transactions on Computer-Aided Design, CAD-6(6):1062-1081, November 1987.

[16] E.M. Sentovich, K.J. Singh, etc. "SIS: A system for Sequential Circuit Synthesis", Memorandum No. UCB/ERL M92/41.

[17] F.N.Najm,"A survey of power estimation techniques in VLSI circuits", IEEE Trans. VLSI Systems, Vol.2, No.4, pp.446-455, Dec.1994.

[18] T.Uchino, F.Minami, T.Mitsuhashi and N.Goto, "Switching activity analysis using boolean approximation method",Proc. Int. Conf. Computer-Aided Design, pp.20-25,1995.

[19] S.Devadas, K. Keutzer, J. White, "Estimation of power dissipation in CMOS combinational circuits using boolean function manipulation", IEEE Trans. Computer-Aided Design, pp.373-383, March 1992.

[20] W. Cheng and J. H. Patel, "PROOFS: A super fast fault simulator for sequential circuits", Proceedings of the IEEE European Conference on Design Automation, pp. 475-479, March 1990.

[21] T.-L. Chou, K.Roy, and S. Prasad, "Estimation of Circuit Activity Considering Signal Correlations and Simultaneous Switching," IEEE Intl. Conf. on Computer-Aided-Design, pp.300-303, 1994.