In IEEE/ACM International Workshop on Logic Synthesis (IWLS), pages 375-386, June 1998.

# Power Reduction in BIST by Exploiting Don't Cares in Test Patterns

José C. Costa, Paulo F. Flores, Horácio C. Neto, José C. Monteiro and João P. Marques Silva

Instituto Superior Técnico
Cadence European Laboratories/INESC
R. Alves Redol, 9, 1
1000 Lisboa, Portugal
{jccc,pff,hcn,jcm,jpms}@algos.inesc.pt

## Abstract

*For a significant number of electronic systems used in safety-critical applications circuit testing is performed periodically. For these systems, power dissipation due to Built-In Self Test (BIST) can represent a significant percentage of the overall power dissipation. One possible solution to address this problem consists of test pattern reordering with the purpose of reducing the amount of power dissipated during circuit testing. By reordering test patterns one is able to find test sequences for which power dissipation is minimized. Moreover, a key observation is that test patterns are in general expected to exhibit don't cares, which can naturally be exploited during test pattern reordering. In this paper we describe efficient algorithms for test pattern reordering in the presence of don't cares. Preliminary experimental results amply confirm that the power savings due to test pattern reordering using don't cares can be significant.*

# 1 Introduction

Many circuits today include on-chip structures that enable circuit self-testing, known as built-in self-test (BIST) [1]. Initially designed to make the testing of the circuits out of the fabrication line easier, they allow for the periodic testing of the circuit. This can be especially important for circuits used in safety-critical or mobile devices. Clearly the penalty to pay is the extra circuitry required for BIST. One approach to reduce this overhead is to use a simple linear feedback shift register (LFSR) to generate a pseudo-random input sequence, which is run until a given fault coverage has been achieved [1, 5]. The disadvantage is that for high fault coverages the run time may become too long. A different approach is to use an automated test-pattern generator (ATPG) tool to obtain a (ideally minimum) set of test patterns necessary for the desired fault coverage. Then, the BIST structure reduces to a counter-type finite state machine (FSM) that generates each of these patterns sequentially [5]. Even though this latter solution in general requires larger area, it is also clear that it provides shorter test sequences, thus being the option of choice for specific applications [1, 5]. Moreover, the increased use of periodic testing in safety-critical devices raises concerns about the power that is consumed during this process. Consequently, techniques for reducing the power dissipation during testing are particularly relevant for these devices.

In this paper we address the problem of power reduction during testing. Even though solutions for solving this problem consist of reordering sequences of completely specified test patterns [6], one might expect the potential existence of don't cares in test patterns to help further reducing power dissipation during testing. The main purpose of this paper is to propose solutions for this problem and provide comprehensive empirical evidence that the existence of don't cares in test patterns can in fact play a significant role in reducing power dissipation during testing.

The paper is organized as follows. We start in Section 2 by introducing a formal model for reducing power dissipation during testing whenever test patterns are completely specified. Afterwards, we generalize the model to handle incompletely specified test patterns, and in Section 4 we describe power reduction algorithms for both models. Section 5 presents experimental results validating the proposed power reduction approach. Section 6 concludes by outlining ideas for future research work.

# 2 Power Reduction with Completely Specified Test Patterns

## 2.1 Power Dissipation Model

The main sources of power dissipation in CMOS devices are summarized by the following expression [17, p. 236]:

$$P = \frac{1}{2} \cdot C \cdot V_{DD}^2 \cdot f \cdot N + Q_{SC} \cdot V_{DD} \cdot f \cdot N + I_{leak} \cdot V_{DD} \tag{1}$$

where $P$ denotes the total power, $V_{DD}$ is the supply voltage, and $f$ is the frequency of operation.

The first term in (1) corresponds to the power involved in charging and discharging circuit nodes. $C$ represents the node capacitances and $N$ is the switching activity, i.e. the number of gate output transitions per clock cycle (also known as *transition density* [14]). $\frac{1}{2} \cdot C \cdot V_{DD}^2$ is the energy involved in charging or discharging a circuit node with

capacitance $C$ and $f \cdot N$ is the average number of times per second that the nodes switches.

The second term in (1) represents the power dissipation due to current flowing directly from the supply to ground during the (hopefully small) period that the pull-up and pull-down networks of the CMOS gate are both conducting when the output switches. This current is often called *short-circuit current*. The factor $Q_{SC}$ represents the quantity of charge carried by the short-circuit current per transition.

The third term in (1) is related to the static power dissipation due to leakage current $I_{leak}$. The transistor source and drain diffusions in a MOS device form parasitic diodes with bulk regions. Reverse bias currents in these diodes dissipate power. Subthreshold transistor currents also dissipate power. $I_{leak}$ accounts for both these small currents.

These three factors for power dissipation are often referred to as *dynamic* power, *short-circuit* power and *leakage current* power respectively.

It has been shown [7] that during normal operation of well designed CMOS circuits the switching activity power accounts for over 90% of the total power dissipation. Thus power optimization techniques at different levels of abstraction target minimal switching activity power. The model for power dissipation for a gate *i* in a logic circuit is simplified to:

$$\frac{1}{2} \cdot C_i \cdot V_{DD}^2 \cdot f \cdot N_i \tag{2}$$

Both simulation-based (e.g., [4]) and probabilistic (e.g., [8]) techniques have been proposed for the computation of $N_i$. Simulation-based techniques use a logic or timing simulator. The circuit is simulated with a *sufficiently large* number of randomly generated input vectors to obtain an average transition count at every gate in the circuit.

Simulation-based techniques can be very efficient for loose accuracy bounds. Increasing the accuracy may require a prohibitively high number of simulation vectors. Given some statistical information of the inputs, probabilistic methods propagate this information through the logic circuit obtaining statistics about the switching activity at each node in the circuit. Only one pass through the circuit is needed, thus making these methods potentially very efficient. Still, modeling issues like correlation between signals can make these methods computationally expensive.

## 2.2 Model for Completely Specified Test Patterns

For the testing of the circuit, the only requirement is that all the test-patterns generated by the ATPG are applied to the circuit. Thus, one degree of freedom that can be explored is the *order* by which these patterns are applied.

Let $\{T_1, T_2, \ldots, T_m\}$ be a given sequence of *completely specified* test patterns. The problem of power reduction during testing can be formulated as the identification of a permutation $\langle i_1, \ldots, i_m \rangle$ such that the overall power consumption is minimized. This problem can be naturally reduced to the (euclidean) traveling salesperson problem (TSP) [10]. Let each test pattern be a vertex in a graph and let the weight $w_{kl}$ of the edge between vertices $v_k$ and $v_l$ be the power that is consumed due to the sequence of input vectors $T_k$ and $T_l$. The cycle $\langle v_{i_1}, v_{i_2}, \ldots, v_{i_m}, v_{i_1} \rangle$ that visits every vertex in the graph with minimum sum of the edge weights is the optimum solution to the power minimi-

zation problem[1].

Moreover, the power consumption between every possible input-vector pair $(T_k, T_l)$ can be heuristically approximated by the Hamming distance between the input vectors. The argument is that by minimizing the switching activity at the inputs we will also be minimizing the switching activity on internal nodes in the circuit. Although this is not always true (one transition in a given input may cause many transitions in internal nodes, whereas several inputs changing may cause fewer transitions), it is a good approximation for generic circuits as confirmed by the results presented in Section 5.

Finally, we note that even though the euclidean traveling salesman problems is NP-hard, several efficient polynomial-time approximation algorithms exist [10]. In Section 4 we modify one of these approximation algorithms to obtain an efficient power reduction algorithm in the presence of don't care conditions.

# 3 Power Reduction Using Incompletely Specified Test Patterns

In this section we consider the changes to the power reduction model described in the previous section whenever test patterns are allowed to be incompletely specified. In general, ATPG algorithms attempt to generate test patterns with a maximal number of don't cares, so that compaction of test patterns becomes facilitated. Hence, power reduction techniques for circuit testing should address the potential advantages of exploiting the don't cares in the test set. Even though most ATPG tools are in general able to generate test patterns with don't cares, we start this section by briefly outlining a formal model for the minimization of test patterns (or equivalently for the maximization of don't cares in test patterns). Afterwards, we propose a power reduction modeling approach for handling don't cares in test patterns.

## 3.1 Generating Incompletely Specified Test Patterns

The identification of test patterns which minimize the number of test patterns can be formally modeled as an integer linear program (ILP) formulation [9]. The main steps for constructing the ILP model are the following:

1. The first step is to represent circuits and fault detection problems using Conjunctive Normal Form (CNF) formulas and formulate the fault detection problem as an instance of the Propositional Satisfiability (SAT) problem. For the results included in this paper, the model of [15] is assumed, but the models of [12, 16] could also be used.
2. The next step is to develop a CNF model in which variables can have unspecified assignments. Notice that solving SAT requires all variables to be specified. Consequently, a dedicated formal model needs to be developed. (The full description of this model is beyond the scope of the paper. More details can be found in [9].)
3. Afterwards, we apply the resulting CNF model to the representation of circuits and fault detection problems.
4. We can then map the resulting CNF into an ILP model. This step is straightforward, since clauses can also be viewed as algebraic inequalities [2].

1. Even though other formulations not based on a graph tour could also be used, the proposed formulation allows a simpler modeling approach.

5. Finally, we specify the cost function of the resulting integer optimization model so that the total number of specified assignments is minimized.

It can be shown that the proposed ILP model is indeed correct [9]. A prototype tool based on this model, MTP, is used in Section 5 to obtain test patterns containing an optimal number of unspecified assignments. Besides MTP, we also use the most recent version of ATALANTA [13], which allows computing and simulating test patterns with don't cares.

### 3.2 Power Reduction Using Don't Cares

It can readily be concluded that if test patterns contain don't cares, then the straightforward mapping of the power reduction problem to the TSP is no longer valid. Indeed, the existence of test pattern with don't cares implies that the Hamming distances between test patterns become conditional, and depend on the final assignments to the unspecified bits.

Let us consider the following test set $\{T_1 = \langle 1, X, 0, 0\rangle; T_2 = \langle X, 1, 0, 1\rangle; T_3 = \langle 0, X, 1, 1\rangle\}$. Depending on the values specified to the don't care bits, the Hamming distance from $T_1$ to $T_2$ can range from 1 to 3. Moreover, if the distance from $T_1$ to $T_2$ is known to be 1, then the distance from $T_2$ to $T_3$ is at least 2.

In order to address the problem of power reduction in the presence of don't cares, we propose to use the TSP formulation of Section 2, but where edge weights are now conditional numbers, and modify existing TSP approximation algorithms in order to handle conditional edge weights. These algorithms are described in the next section.

## 4 Power Reduction Algorithms

As described in Section 2.2, for completely specified test patterns, the straightforward representation of the power reduction problem as an instance of the TSP problem immediately yields a wealth of approximation algorithms [10]. On the other hand, and as illustrated earlier, exploiting the existence of don't cares in test patterns either requires using dedicated models and algorithms or adapting existing approximation algorithms for the TSP. In this paper we choose to adapt the 2-opt [10] local search approximation algorithm for the TSP, that is described below. The resulting power reduction algorithm is organized as follows:

1. Use a dedicated algorithm for computing a test set where each test pattern contains don't cares. Either MTP [9] or ATALANTA [13] can be used.
2. Apply a heuristic procedure for identifying an initial tour. Several different heuristics are described below.
3. Use a modified 2-opt local-search approximation algorithm for the TSP to reorder the test patterns. Repeat this step while the tour cost can be reduced.

The following initial ordering heuristics have been implemented (which will henceforth be referred to as **H1** through **H5**):

1. Randomly order the test patterns.

2. Order test patterns by decreasing order of don't cares in each test pattern. By choosing for the first test patterns those with more don't cares one can expect that the distances between the first test patterns be the lowest possible.
3. This heuristic starts by applying heuristic 2. Afterwards, greedily select the next test pattern as one that minimizes the distance from the current test pattern. This heuristic goes one step further in minimizing the distances between the first test patterns by choosing the second best test pattern, and then the third best, and so on.
4. In this heuristic for each bit position the don't care bits are set to the bit that occurs more often. By using this approach the test patterns are expected to become more similar between each other. Next an ordering is made that approximates Gray coding. This approach attempts to order the test patterns in such a way that the average distances between test patterns is minimized.
5. The last heuristic sets the don't cares in the same manner in the heuristic 4. Afterwards, with all the test patterns specified, the Christofides TSP approximation algorithm is used for defining the initial tour. This heuristic permits using a TSP approximation algorithm in a tour where the test patterns are expected to be similar to each other.

After having the initial tour of the test patterns the following modified 2-opt [10] is applied:

1. Evaluate the tour cost by specifying the don't care bits which minimize the distance between consecutive test patterns.
2. Reverse the action taken in Step 1 by getting the test patterns with don't cares.
3. For every pair of test patterns ($T_i$ and $T_j$), cut the link between those test patterns and the next ones ($T_{i+1}$ and $T_{j+1}$), and link $T_i$ with $T_{j+1}$ and $T_j$ with $T_{i+1}$. For this new ordering obtain the tour cost as in Step 1.
4. If the *lowest* tour cost found in Step 3 is lower than the initial tour cost then keep the order for that lowest tour cost and repeat Step 1 for that ordering. Otherwise the algorithm terminates.

Finally, the test sequence considered, and which is used for the experimental results by the power estimator tool, is given by the result of the modified 2-opt with the don't care bits specified in such a way that the Hamming distance between consecutive test patterns is minimized.

# 5 Experimental Results

This section includes results of applying the algorithm described in the previous section to the IWLS'89 [11] benchmark circuits and to the ISCAS'85 [3] benchmark circuits. The ATPG tools ATALANTA [13] and MTP [9] were used on all the experiments. ATALANTA was used to generate both completely and incompletely specified test patterns. (Observe that ATALANTA allows for the generation of test patterns with don't cares.) MTP was used to generate test patterns with a maximum number of incompletely specified assignments. Finally, we note that the results on power saving subsequently shown correspond to measured power dissipation obtained by actually simulating the test sequences

| Benchmark | Completely specified (ordered vs. unordered) | | Incompletely specified versus ordered completely specified | | | | | |
|---|---|---|---|---|---|---|---|---|
| | # TP | % power reduction | # TP | % power reduction | | | | |
| | | | | H1 | H2 | H3 | H4 | H5 |
| 9symml | 78 | 43.9 | 80 | 3.8 | 7.2 | 15.3 | 11.1 | 17.1 |
| alu4 | 100 | 29.0 | 128 | 12.2 | 11.7 | 18.3 | 13.7 | 20.4 |
| cht | 17 | 5.6 | 10 | 25.3 | 27.9 | 24.7 | 17.9 | 23.0 |
| cm138a | 12 | 36.3 | 12 | 20.9 | 20.7 | 11.0 | 16.2 | 17.5 |
| cm150a | 34 | 16.5 | 39 | 38.6 | 46.2 | 53.6 | 42.4 | 45.3 |
| cm163a | 15 | 15.6 | 14 | 31.2 | 34.7 | 40.0 | 35.4 | 42.7 |
| cmb | 30 | 43.2 | 27 | 16.9 | 17.7 | 21.9 | 13.8 | 15.1 |
| comp | 56 | 27.1 | 60 | 57.0 | 56.9 | 60.6 | 58.7 | 57.7 |
| comp16 | 72 | 38.9 | 99 | 40.6 | 47.6 | 44.2 | 46.6 | 42.3 |
| cordic | 43 | 36.6 | 47 | 49.3 | 56.2 | 61.5 | 54.0 | 59.4 |
| cu | 27 | 35.6 | 26 | 23.4 | 34.3 | 36.4 | 13.1 | 30.3 |
| majority | 11 | 36.1 | 11 | 9.6 | 15.9 | 5.1 | 10.9 | 4.6 |
| misex1 | 18 | 14.3 | 17 | 36.2 | 26.5 | 24.9 | 26.7 | 33.0 |
| misex2 | 47 | 20.5 | 37 | 41.7 | 48.7 | 52.6 | 52.5 | 51.6 |
| misex3 | 154 | 38.3 | 178 | 21.0 | 24.6 | 27.9 | 19.1 | 24.5 |
| mux | 35 | 20.5 | 38 | 28.7 | 31.2 | 36.9 | 41.4 | 32.0 |
| pcle | 17 | 16.6 | 20 | 28.1 | 37.1 | 47.2 | 31.2 | 42.7 |
| pcler8 | 19 | 20.4 | 21 | 35.4 | 23.7 | 43.2 | 27.5 | 37.8 |
| term1 | 43 | 14.4 | 43 | 43.6 | 49.5 | 47.2 | 39.6 | 46.9 |
| too_large | 103 | 32.1 | 146 | 36.1 | 41.2 | 49.9 | 42.1 | 46.3 |
| unreg | 15 | 15.8 | 10 | 12.4 | 12.5 | 12.0 | 11.2 | 17.2 |

Table 1: Power reduction results for the IWLS'89 benchmarks obtained with ATALANTA

## 5.1 IWLS Benchmarks

The results for power reduction in test sequences for the IWLS'89 benchmark circuits are shown in Table 1. The columns labeled ***completely specified*** indicate the percentage power savings that result from ordering a sequence of completely specified test patterns, and the number of computed test patterns (**#TP**). For this experiment the Christofides approximation algorithm [10] was used. The columns labeled ***incompletely specified*** indicate the power savings from exploiting the don't cares in incompletely specified test patterns over an *already ordered* sequence of completely specified test patterns. For this experiment the algorithm described in Section 4 was used, and the different

| Benchmark | Completely specified (ordered vs. unordered) | | Incompletely specified versus ordered completely specified | | | | | |
|---|---|---|---|---|---|---|---|---|
| | # TP | % power reduction | # TP | % power reduction | | | | |
| | | | | H1 | H2 | H3 | H4 | H5 |
| c432 | 58 | 16.9 | 75 | 36.7 | 48.6 | 43.5 | 41.3 | 43.0 |
| c499 | 60 | 28.9 | 61 | 18.3 | 14.8 | 17.4 | 18.5 | 15.9 |
| c880 | 51 | 10.4 | 79 | 52.1 | 46.8 | 44.9 | 44.9 | 54.2 |
| c1355 | 94 | 30.5 | 96 | 9.7 | 7.8 | 10.8 | 11.4 | 10.7 |
| c1908 | 128 | 27.4 | 175 | 44.8 | 45.2 | 50.3 | 44.7 | 50.8 |
| c2670 | 117 | 14.2 | 156 | 68.0 | 68.5 | 68.7 | 66.2 | 69.4 |
| c3540 | 159 | 18.7 | 253 | 30.7 | 33.6 | 44.2 | 33.9 | 47.4 |
| c5315 | 116 | 11.0 | 158 | 50.1 | 49.9 | 52.9 | 50.7 | 53.6 |
| c6288 | 25 | 18.6 | 54 | 55.6 | 57.7 | 57.5 | 57.1 | 53.9 |

Table 4: Power reduction results for the ISCAS85 benchmarks

initial ordering heuristics (**H1** through **H5**) were considered.

As can be readily concluded, large power savings ranging from 30% to 60% are achieved in most cases. This is particularly significant since these results measure the percentage power savings over the *already ordered* sequence of test patterns. Finally, we note that the number of test patterns (**#TP**) does not change significantly (especially for usage in BIST) from completely specified to incompletely specified test patterns.

It is interesting to note that if the test patterns are not compacted, then we can achieve even higher power savings. Indeed, by allowing a larger number of test patterns, which for BIST may be perfectly acceptable, the degrees of freedom of the optimization algorithm are larger and larger power savings can be obtained. As shown in Table 2 for ATALANTA, with non-compacted test patterns and for most benchmark examples, we are in general able to obtain larger power savings than with compacted test patterns. By using MTP instead, and as shown in Table 3, we are able to obtain even better results in most cases. With non-compacted test patterns, the results of MTP tend to be better than those of ATALANTA. This is to be expected, since MTP produces test patterns with an maximum number of don't cares, which can then be exploited by the reordering algorithms.

## 5.2 ISCAS Benchmarks

The results in the previous section validate the proposed approach for reducing power dissipation for medium-size circuits. In this section we use ATALANTA to generate test patterns with and without don't cares and apply the power reduction procedure to the ISCAS'85 benchmark circuits. The results are shown in Table 4. As can be concluded, once again, large power savings can be obtained by generating test pattern with don't cares, reordering the test sets and specifying the unassigned bits so that the dissipated power is minimized. From Table 4, we can conclude that

| Benchmark | Completely specified (ordered vs. unordered) | | Incompletely specified versus ordered completely specified | | | | | |
|---|---|---|---|---|---|---|---|---|
| | # TP | % power reduction | # TP | % power reduction | | | | |
| | | | | H1 | H2 | H3 | H4 | H5 |
| 9symml | 78 | 43.9 | 80 | 3.8 | 7.2 | 15.3 | 11.1 | 17.1 |
| alu4 | 100 | 29.0 | 187 | 26.9 | 33.4 | 40.7 | 29.8 | 37.8 |
| cht | 17 | 5.6 | 181 | 85.0 | 82.4 | 86.6 | 84.1 | 88.9 |
| cm138a | 12 | 36.3 | 12 | 20.9 | 20.7 | 11.0 | 16.2 | 17.5 |
| cm150a | 34 | 16.5 | 51 | 44.9 | 56.0 | 54.5 | 53.1 | 60.9 |
| cm163a | 15 | 15.6 | 35 | 59.2 | 73.4 | 71.6 | 68.5 | 69.4 |
| cmb | 30 | 43.2 | 30 | 23.5 | 26.9 | 28.6 | 22.8 | 21.0 |
| comp | 56 | 27.1 | 63 | 55.7 | 56.7 | 58.5 | 61.2 | 55.6 |
| comp16 | 72 | 38.9 | 105 | 44.0 | 45.4 | 47.2 | 44.0 | 46.5 |
| cordic | 43 | 36.6 | 48 | 49.7 | 59.2 | 63.3 | 58.3 | 55.3 |
| cu | 27 | 35.6 | 35 | 43.0 | 53.8 | 50.6 | 32.5 | 50.5 |
| majority | 11 | 36.1 | 11 | 9.6 | 15.9 | 5.1 | 10.9 | 4.6 |
| misex1 | 18 | 14.3 | 22 | 35.2 | 36.3 | 45.5 | 41.5 | 45.1 |
| misex2 | 47 | 20.5 | 67 | 62.6 | 67.6 | 69.6 | 67.8 | 72.0 |
| misex3 | 154 | 38.3 | 198 | 27.0 | 29.1 | 36.4 | 26.9 | 37.5 |
| mux | 35 | 20.5 | 49 | 38.8 | 41.5 | 50.2 | 41.8 | 52.7 |
| pcle | 17 | 16.6 | 55 | 65.2 | 76.5 | 78.6 | 74.3 | 76.6 |
| pcler8 | 19 | 20.4 | 78 | 62.4 | 77.4 | 80.4 | 82.4 | 75.3 |
| term1 | 43 | 14.4 | 103 | 71.1 | 70.5 | 77.4 | 73.1 | 74.7 |
| too_large | 103 | 32.1 | 197 | 44.2 | 49.1 | 59.6 | 47.2 | 58.7 |
| unreg | 15 | 15.8 | 128 | 73.7 | 80.9 | 85.3 | 77.1 | 81.5 |

Table 2: Power reduction results for non-compacted test patterns obtained with ATALANTA

with specified test patterns, the power savings from reordering the test patterns range from 10% to 30%. In addition, after generating test patterns with don't cares we obtain, over the already ordered (but completely specified) test sequence, power savings that range from 10% to 70%. Moreover, for the majority of benchmarks the power savings are between 40% and 60%. Consequently, we can conclude that test pattern reordering in the presence of don't cares leads to large power savings over already ordered test sequences.

Furthermore, we noticed that the percentage power savings in general increases as the size of the circuit and number of test patterns increases. Hence, for large circuits we expect the proposed power reduction algorithm to lead to

| Benchmark | Completely specified (ordered vs. unordered) | | Incompletely specified versus ordered completely specified | | | | | |
|---|---|---|---|---|---|---|---|---|
| | # TP | % power reduction | # TP | % power reduction | | | | |
| | | | | H1 | H2 | H3 | H4 | H5 |
| 9symml | 78 | 43.9 | 83 | 4.8 | 8.0 | 13.5 | 2.6 | 11.2 |
| alu4 | 100 | 29.0 | 217 | 33.2 | 33.0 | 52.7 | 38.1 | 46.1 |
| cht | 17 | 5.6 | 184 | 83.0 | 89.8 | 90.7 | 90.0 | 91.4 |
| cm138a | 12 | 36.3 | 12 | 0.5 | 0.5 | 13.5 | 17.8 | 8.1 |
| cm150a | 34 | 16.5 | 43 | 36.9 | 49.6 | 49.7 | 34.7 | 59.9 |
| cm163a | 15 | 15.6 | 35 | 70.3 | 73.0 | 76.4 | 73.4 | 71.2 |
| cmb | 30 | 43.2 | 30 | 24.7 | 22.7 | 26.5 | 30.2 | 21.9 |
| comp | 56 | 27.1 | 70 | 53.7 | 54.7 | 57.2 | 52.5 | 58.4 |
| comp16 | 72 | 38.9 | 138 | 55.2 | 56.6 | 56.1 | 48.8 | 54.1 |
| cordic | 43 | 36.6 | 48 | 60.0 | 55.6 | 65.8 | 60.1 | 60.5 |
| cu | 27 | 35.6 | 36 | 44.2 | 47.9 | 46.6 | 42.8 | 46.7 |
| majority | 11 | 36.1 | 11 | 10.1 | -13.9 | -3.0 | 9.6 | 6.3 |
| misex1 | 18 | 14.3 | 21 | 39.8 | 29.1 | 45.1 | 41.7 | 44.6 |
| misex2 | 47 | 20.5 | 69 | 66.7 | 68.5 | 66.9 | 69.0 | 70.1 |
| misex3 | 154 | 38.3 | 247 | 36.3 | 34.9 | 45.0 | 30.5 | 39.7 |
| mux | 35 | 20.5 | 42 | 40.0 | 48.9 | 47.5 | 38.2 | 37.7 |
| pcle | 17 | 16.6 | 46 | 67.3 | 76.2 | 77.2 | 73.5 | 73.0 |
| pcler8 | 19 | 20.4 | 65 | 74.1 | 80.8 | 77.2 | 75.1 | 78.2 |
| term1 | 43 | 14.4 | 109 | 69.9 | 76.0 | 74.0 | 74.2 | 73.1 |
| too_large | 103 | 32.1 | 194 | 52.5 | 53.0 | 64.4 | 53.8 | 60.4 |
| unreg | 15 | 15.8 | 116 | 78.4 | 86.2 | 88.3 | 81.2 | 87.6 |

Table 3: Power reduction results for non-compacted test patterns obtained with MTP

similar or greater power savings. Regarding the heuristics proposed in Section 4 for constructing the initial tour, the results do not identify a clear best heuristic, even though the greedy heuristic **H3** performs better in most cases. Finally, these experimental results clearly indicate that exploiting don't cares in sequences of test patterns may prove extremely useful whenever power reduction is the main objective.

For the ISCAS benchmarks we noticed that the proposed (and non-optimized) 2-opt algorithm would take a couple of hours of CPU time for the examples with a larger number of test patterns, and would take more than 24 of CPU time for C7552. As a result, we modified the 2-opt algorithm so that only 500 links were examined at each iteration of

| Benchmark | Completely specified (ordered vs. unordered) | | Incompletely specified versus ordered completely specified | | | | | |
|---|---|---|---|---|---|---|---|---|
| | # TP | % power reduction | # TP | % power reduction | | | | |
| | | | | H1 | H2 | H3 | H4 | H5 |
| c432 | 58 | 10.4 | 75 | 48.4 | 45.9 | 49.0 | 45.1 | 51.3 |
| c499 | 60 | 30.5 | 61 | 5.3 | 5.9 | 7.8 | 5.7 | 5.1 |
| c880 | 51 | 27.4 | 79 | 33.5 | 38.6 | 49.6 | 35.5 | 48.5 |
| c1355 | 94 | 14.2 | 96 | 58.1 | 64.5 | 68.6 | 63.2 | 66.9 |
| c1908 | 128 | 18.7 | 175 | 20.3 | 21.7 | 42.4 | 30.8 | 43.4 |
| c2670 | 117 | 16.9 | 156 | 37.2 | 40.7 | 44.9 | 35.2 | 45.6 |
| c3540 | 159 | 28.9 | 253 | 13.5 | 8.1 | 18.8 | 14.2 | 20.1 |
| c5315 | 116 | 11.0 | 158 | 47.5 | 45.5 | 51.2 | 44.8 | 51.5 |
| c6288 | 25 | 18.6 | 54 | 54.0 | 54.9 | 55.6 | 56.3 | 50.9 |
| c7552 | 217 | 15.1 | 347 | 39.7 | 49.6 | 64.1 | 52.7 | 67.8 |

Table 5: Power reduction results with modified 2-opt algorithm

the algorithm (instead of a number that is quadratic in the number of test patterns). The results obtained with this modified 2-opt algorithm are shown in Table 5. As can be concluded, for the benchmarks with a larger number of test patterns, the new results are significantly worse, thus indicating a tradeoff between the computational effort spent with the reordering algorithm and the attained power savings.

# 6 Conclusions

In this paper we describe a procedure for reducing power dissipation during testing by exploiting don't cares in test sequences. We provide experimental evidence that exploiting don't cares in test sequences can lead to very significant savings in dissipated power. In designs where periodic testing is required, these power reduction techniques may play a key role in the design of BIST hardware.

Future research work entails developing a more flexible configuration of the local search (2-opt) optimization algorithm, that will permit controlling the number of iterations implemented by the 2-opt algorithm and, consequently, the execution time of the algorithm and the amount of power savings. Additional research work involves introducing a more detailed model, intended to accurately predict power dissipated in signal transitions between pairs of test patterns, and studying alternative algorithmic solutions. The impact of reordered test sequences in the resulting area of the BIST logic must also be evaluated, even though power and not area is known to be the key metric for some applications.

## References


[1] M. Abramovici, M. A. Breuer and A. D. Friedman, *Digital Systems Testing and Testable Design*, Computer Science Press, 1990.

[2] P. Barth, "A Davis-Putnam Based Enumeration Algorithm for Linear Pseudo-Boolean Optimization," Technical Report MPI-I-95-2-003, Max-Planck-Institut für Informatik, January 1995.

[3] F. Brglez and H. Fujiwara, "A Neutral List of 10 Combinational Benchmark Circuits and a Target Translator in FORTRAN," in *Proceedings of the International Symposium on Circuits and Systems(ISCAS)*, 1985. (ISCAS'85 Benchmark Suite available from http://www.cbl.ncsu.edu/pub/Benchmark_dirs/ISCAS85/.)

[4] R. Burch, F. Najm, P. Yang and T. Trick, "McPOWER: A Monte Carlo Approach to Power Estimation," in *Proceedings of the International Conference on Computer-Aided Design*, November 1992.

[5] K. Chakrabarty, B. T. Murray, J. Liu and M. Zhu, "Test Width Compression for Built-In Self Testing", in *Proceedings of the International Test Conference*, October 1997.

[6] S. Chakravarty and V. Dabholkar, "Minimizing Power Dissipation in Scan Circuits During Test Application," in *International Workshop on Low-Power Design*, April 1994.

[7] A. Chandrakasan, T. Sheng and R. Brodersen, "Low Power CMOS Digital Design," in *IEEE Journal of Solid State Circuits*, no. 4, vol. 27, pp. 473-484, April 1992.

[8] J. C. Costa, J. C. Monteiro and S. Devadas, "Switching Activity Estimation using Limited Depth Reconvergent Path Analysis", in *Proceedings of the International Symposium on Low Power Design*, 1997.

[9] P. F. Flores, H. C. Neto and J. P. Marques Silva, "An Exact Solution to the Minimum-Size Test Pattern Problem," in *IEEE/ACM International Workshop on Logic Synthesis*, 1998.

[10] D. S. Johnson and L. A. McGeoch, "The Traveling Salesman Problem: A Case Study in Local Optimization," in *Local Search in Combinatorial Optimization*, E. H. L. Aarts and J. K. Lenstra (eds.), John Wiley and Sons, 1996.

[11] IWLS'89 Benchmark Suite, available from http://www.cbl.ncsu.edu/pub/Benchmark_dirs/LGSynth89/.

[12] T. Larrabee, "Test Pattern Generation Using Boolean Satisfiability," *IEEE Transactions on Computer-Aided Design*, vol. 11, no. 1, pp. 4-15, January 1992.

[13] H. K. Lee and D. S. Ha, "On the Generation of Test Patterns for Combinational Circuits," Technical Report No. 12_93, Department of Electrical Engineering, Virginia Polytechnic Institute and State University, 1993.

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

[15] J. P. M. Silva and K. A. Sakallah, "Robust Search Algorithms for Test Pattern Generation," in *Proceedings of the Fault-Tolerant Computing Symposium*, June 1997.

[16] P. R. Stephan, R. K. Brayton and A. L. Sangiovanni-Vincentelli, "Combinational Test Generation Using Satisfiability," *IEEE Transactions on Computer-Aided Design*, vol. 15, no. 9, pp. 1167-1176, September 1996.

[17] N. Weste and K. Eshraghian, *Principles of CMOS VLSI Design*, Addison-Wesley, 2nd Edition, 1994.