Selecting ATE Frequencies for Power Constrained Test Time Reduction Using Aperiodic Clock

Sindhu Gunasekar (Master’s Student Presenter) and Vishwani D. Agrawal
Department of Electrical and Computer Engineering
Auburn University, Auburn, AL 36849
Email: szg0034@tigermail.auburn.edu, vagrawal@eng.auburn.edu

Abstract—An aperiodic test clock methodology to reduce test time of wafer sort has been recently proposed. In practice, however, an automatic test equipment (ATE) allows only a small number of clock periods and finding those is a mathematically complex problem. This paper proposes an algorithm for optimal selection of any given number of tester clock periods.

Keywords-Aperiodic clock, Test time reduction, Automatic Test Equipment, Scan test, Adaptive clocking

I. INTRODUCTION

The scan based test is the most popular technique of testing sequential circuits. In this method, flip-flops functionally form one or more shift registers. Faults in the sequential circuit can be tested by shifting test patterns in and out of the shift register. An ATE is used to test integrated circuits for faults after fabrication. The ATE has high initial and recurring costs but it provides high-volume testing [5]. The testing cost of using an ATE increases directly with the time spent in testing the chip and adds to the final cost of the chip. In the present day, where complex integrated circuits may be constructed using a billion gates, long test times are a concern. In order to minimize the extensive time spent on the tester, Venkataramani and Agrawal [8] propose a methodology that reduces test time on the ATE using aperiodic clocking. Ideally, for a test that is n cycles long, the test time is minimum when the CUT is tested with n different frequencies. In practice, the ATE allows only a limited number of frequencies and determining an optimum set of frequencies is a combinatorial optimization problem.

II. PRIOR WORK

A recent approach to reduce test time of wafer sort test on an automatic test equipment (ATE) has been proposed by Venkataramani and Agrawal [8]. Power dissipation during testing is typically higher than the functional power dissipation because of high signal activity. To keep the test power dissipation within limits, test cycles are run with periods larger than the functional clock period, which in turn results in long test times. Since not all test cycles consume the same amount of power, the clock interval of each test cycle can be different from others, while still keeping the power dissipation under control.

Recent work [8] [9] proposes a methodology where the clock interval of each cycle is varied in proportion to the energy consumed in each cycle, constrained by the maximum allowable rate of power dissipation and the critical path delay. The minimum clock interval for each test cycle as proposed in [8] and [9] is given by,

$$T_{test(i)} = \max \left\{ T_{structure}, \frac{E_i}{P_{MAX(rated)}} \right\}$$ (1)

where $T_{test(i)}$ is the minimum clock interval for the $i$th clock cycle, $T_{structure}$ is the structure constrained clock period, $E_i$ is the total energy consumed during the $i$th cycle and $P_{MAX(rated)}$ is the maximum rated power of the circuit, given by the specification of the circuit. The average power consumed in such an aperiodic methodology is higher than the conventional method, but test time is reduced significantly.

Since energy consumed in each cycle is different, this aperiodic test methodology to reduce test time may, in the worst case require up to $n$ different frequencies for a circuit that has a test comprising of $n$ vectors. The ATE cannot be configured for testing using as many as $n$ different frequencies. Nevertheless, we can see in the next section that considerable reduction in test time may be obtained with fewer than $n$ frequencies. Only a selected subset of clock periods $k_i$ of the $n$ different clock intervals can be implemented on the ATE and finding the optimum subset of clock periods for minimum test time is a discrete optimization problem. We propose an algorithm for selection of the optimum subset of ATE frequencies/clock periods.
where, the test time, given by

$$x = \text{clock interval}$$

computation time can be significantly reduced by calculating
time. Instead of computing the total test time at each iteration,
to see which one gives maximum reduction in the total test
one iteration over all the remaining

$$k = \text{element}.$$ At each stage, another clock period

Initially, one optimum clock period

over all the
different clock intervals arranged in descending order. Initially,
one optimum clock period $$k_1$$ is selected and this is
the largest clock interval $$t_1$$ of the set $$t_i.$$ The optimum set $$k_i$$
now has one element. In the first stage, another clock period

$$k_2$$ is picked from $$t_i$$ and this is determined by performing
one iteration over all the remaining $$n - 1$$ clock intervals
to see which one gives maximum reduction in the total test
time. Instead of computing the total test time at each iteration,
computation time can be significantly reduced by calculating
the amount of test time saved when each clock interval of the
set $$t_i$$ is added to $$k_i.$$ At each stage, $$k_i$$ is determined by that
clock interval $$x \in t_i, \forall i$$ which gives a maximum saving in the
test time, given by

$$S_x = (k_j - x) \times R$$

where,

$$k_j = \text{clock period in the already existing set } k_i \text{ which is just greater than } x$$

$$k_h = \text{clock period in the already existing set } k_i \text{ which is just lower than } x$$

$$R = \text{number of test vectors in } t_i \text{ with clock intervals less than } x \text{ but greater than } k_h$$

At each stage, the set of optimal clock periods $$k_i$$ found in
the previous stages are unchanged and one optimum value is
appended to $$k_i.$$ This is repeated until $$k$$ optimal clock periods
are obtained. The computation time of the greedy algorithm
increases linearly with the number of allowed ATE frequencies
$$k$$ and the number of test cycles $$n$$ for testing the circuit and
the time complexity is given by $$O(nk).$$

B. Local Search

If the search space is small, there are algorithms that will
systematically search the space to determine the solution. If
the search space is too big for systematic search, it fails to
find any meaningful solutions in a reasonable time, because
systematic search fails to consider enough of the search
space [6]. The problem of finding $$k$$ optimum clock-periods
has a $$k$$-dimensional symmetrical search space and this is quite
large even for smaller circuits. The local search method does
not systematically search the whole search space but for the
problem at hand it helps find a local minimum quickly.

The method starts with an initial assignment of optimum
clock periods $$k_i$$ by assigning a value to all the variables.
This initial assignment is chosen heuristically. We used two
different initial assignments: the results of the greedy search
algorithm and equidistant clock periods. The algorithm then
searches a set of neighbors of the current assignment and se-
lects one to be the next current assignment $$k_i.$$ The assignment
is then iteratively improved till it converges [4]. This local
search is repeated by restarted with the next heuristic initial
assignment. The neighbors of the current assignment are those
assignments that differ in the assignment of a single variable.
In the problem at hand, the neighborhood is where $$k - 1$$
clock periods in the assignment $$k_i$$ are unchanged and the $$k$$th
clock period takes all the values in between the unchanged
values on its either side. The value in the neighborhood
that minimizes the objective function is then replaced in the
current assignment $$k_i.$$ It is analogous to making the $$k - 1$$
dimensions of the search space unchanged and finding the
optimum value in the $$k$$th dimension that would minimize the
objective function, not forgetting the symmetry of the objective
function in the $$k$$-dimensional space. These iterations are done

III. ALGORITHM

Suppose $$t_i$$ is the set of the $$n$$ clock intervals for each
of the $$n$$ test cycles of the scan based circuit obtained from
simulation. If the ATE allows only $$k$$ different frequencies to
be configured, finding a subset $$k_i$$ of $$t_i$$, where $$k_i$$ is the set
of $$k$$ optimum clock periods which should be implemented on
the ATE for minimum test time is a combinatorial optimization
problem in the discrete domain. This problem, like many other
optimization problems, is NP-complete. Using metaheuristics,
this NP-complete problem is solved in polynomial-time to find
near-optimal solutions. A greedy algorithm and a local search
algorithm are proposed. The greedy algorithm is quicker and is
used initially to find the number of optimum clocks $$k$$ required
to find the necessary reduction in test time, followed by the
local search algorithm to find a more optimal solution. The
proposed metaheuristics are verified by simulating the ISCAS
‘89 sequential benchmark circuits and experimentally on the
Advantest T2000GS ATE at Auburn University.

A. Greedy Algorithm

A greedy algorithm makes the locally optimal choice at each
stage in the hope of finding the global optimum. The greedy
heuristic, in general, does not produce an optimal solution,
but nonetheless it may yield locally optimal solutions in a
reasonable time that approximate a global optimal solution. In
the greedy algorithm for the optimization problem at hand, at
each stage, one optimum clock period is found by one iteration
over all the $$n$$ different clock intervals. Suppose $$t_i$$ is the set of
all the $$n$$ different clock intervals arranged in descending order.
Initially, one optimum clock period $$k_1$$ is selected and this is
the largest clock interval $$t_1$$ of the set $$t_i.$$ The optimum set $$k_i$$
now has one element. In the first stage, another clock period
$$k_2$$ is picked from $$t_i$$ and this is determined by performing
one iteration over all the remaining $$n - 1$$ clock intervals
to see which one gives maximum reduction in the total test
time. Instead of computing the total test time at each iteration,
computation time can be significantly reduced by calculating
the amount of test time saved when each clock interval of the
set $$t_i$$ is added to $$k_i.$$ At each stage, $$k_i$$ is determined by that
clock interval $$x \in t_i, \forall i$$ which gives a maximum saving in the
test time, given by

$$S_x = (k_j - x) \times R$$

where,
repeatedly varying \( k_1 \) to \( k_k \), one by one, until the assignments no longer change. The values obtained at this stage are most probably local minima of the objective function. The best of the two local searches is chosen to be implemented in the experiment.

Computing the value of the objective function (test time) at each assignment to find the optimum \( k \)th clock period is time-consuming. Instead of computing the total test time at each iteration, computation time can be significantly reduced by calculating the amount of test time saved by the addition of the \( k \)th clock when the \( k-1 \) clocks are unchanged given by (2). The computation time of local search increases linearly with the number of test cycles \( n \) and the time complexity is experimentally found to be \( O(nk^2/5) \).

### IV. Simulation and Experiment

We implemented the above algorithms and applied to ISCAS ‘89 sequential benchmark circuits. The experimental procedure is similar to [8]. Table 1 shows the number of test cycles \( n \) required for testing the scan design of the benchmark circuits we used. We synthesized the behavioral model of these benchmark circuits using Mentor Graphics Leonardo Spectrum [3] with TSMC 180nm technology. Leonardo Spectrum gives the critical path delay through Static Timing Analysis of the circuit. Using Mentor Graphics DFT Advisor, all the flip-flops in the circuit were daisy chained to form a full scan chain. We generated a set of ATPG test vector patterns for stuck-at faults using Mentor Graphics Tessent Fastscan [2]. The transistor level description of the netlist was generated using Mentor graphics Design Architect. We imported the SPICE file into Synopsys Nanosim [1] and performed a transistor level simulation at a nominal voltage of 1.8V to measure the energy dissipated per each cycle of the test. Based on these simulations, the minimum clock interval for each cycle was determined as proposed in [8] and [9] given by (1).

In the absence of any available data on maximum rated power for the ISCAS ‘89 benchmark circuits, we simulated each circuit in functional mode for 1,000 random vectors. The average power dissipated during these simulations was measured and was used as the maximum rated power. The clock interval of each cycle is constrained both by structure and by maximum rated power. The set of the clock intervals \( t_i \) for each of the \( n \) test cycles of the scan based circuit were obtained from this simulation. The proposed algorithm in Section III was used to find the the set of optimum clock periods \( k \) from \( t_i \). Figure 1 shows the normalized test time of s1238 circuit by simulation of the Greedy and Local Search algorithm for \( k \) optimum frequencies.

### Table 1

<table>
<thead>
<tr>
<th>Benchmark circuit</th>
<th>Total scan test clock cycles ( n )</th>
<th>Test time reduction (%) with ( k ) clock frequencies</th>
<th>Lower bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>s298</td>
<td>540</td>
<td>40.17, 44.83, 47.33</td>
<td></td>
</tr>
<tr>
<td>s713</td>
<td>773</td>
<td>30.20, 33.95, 36.09</td>
<td></td>
</tr>
<tr>
<td>s400</td>
<td>1076</td>
<td>25.66, 29, 31.25</td>
<td></td>
</tr>
<tr>
<td>s1238</td>
<td>3361</td>
<td>52.4, 55.65, 57.65</td>
<td></td>
</tr>
<tr>
<td>s1423</td>
<td>6975</td>
<td>23.5, 26.22, 27.89</td>
<td></td>
</tr>
<tr>
<td>s13207</td>
<td>62237</td>
<td>13.94, 14.98, 15.6</td>
<td></td>
</tr>
<tr>
<td>s15850</td>
<td>101707</td>
<td>24.33, 26.67, 27.98</td>
<td></td>
</tr>
<tr>
<td>s38584</td>
<td>224112</td>
<td>15.76, 16.59, 17.05</td>
<td></td>
</tr>
</tbody>
</table>

![Fig. 1. Normalized test time and CPU time of s1238 ISCAS ‘89 benchmark circuit by simulation of the Greedy and Local Search algorithm for \( k \) optimum frequencies.](image-url)
a Xilinx Spartan 3 FPGA XC3S50 soldered on a printed circuit board. The s713 benchmark circuit with full scan design was configured on the FPGA. The FPGA was configured on the run by the ATE using the configuration file generated by the Xilinx ISE tool [7]. The testplan was programmed and the test was performed on the ATE [8] accommodating the delay overhead caused due to the analog measurement module.

V. RESULTS

Table I shows the reduction in test times of the ISCAS ’89 benchmark circuits obtained from the proposed algorithm, for 4, 10 and \( n \) optimum frequencies. It can be emphasized that as few as four frequencies can help achieve up to 52% reduction in test time in some circuits. The s1238 circuit gives a reduction of 52%, 55% and 57% with four, ten and 3,361 frequencies, respectively. When the number of optimum frequencies \( k \) is increased from 10 to 3361, the test time only reduces by an additional 2%. CPU times for the algorithm were in the order of \( ms \) for the smaller circuits and up to 11 seconds for the largest circuit s38584 with 224,112 clock cycles on a computer with Intel Core i3 CPU, 2.27GHz clock and 4GB RAM. The CPU times for computing the greedy algorithm and the local search algorithm are not very different, as seen in Figure 1. The results of the local search algorithm are slightly better than the greedy algorithm for the s1238, with not much increase in CPU time.

VI. FUTURE WORK

The high costs of automatic test equipment (ATE) and the growing number of clock frequencies bring about the need to study on-chip clock generation circuitry for generating aperiodic clocks for testing.

ACKNOWLEDGMENT

This research is supported in part by the National Science Foundation Grant CCF-1116213.

REFERENCES