IEICE TRANS. INF. & SYST., VOL.E88–D, NO.8 AUGUST 2005

PAPER

Power-Constrained Test Synthesis and Scheduling Algorithms for Non-Scan BIST-able RTL Data Paths

Zhiqiang YOU[†a)], ***Student Member***, **Ken'ichi YAMAGUCHI**[††b)], **Michiko INOUE**[†c)], ***Members***,
Jacob SAVIR[†††d)], ***Nonmember***, ***and*** **Hideo FUJIWARA**[†e)], ***Fellow***

SUMMARY This paper proposes two power-constrained test synthesis schemes and scheduling algorithms, under non-scan BIST, for RTL data paths. The first scheme uses boundary non-scan BIST, and can achieve low hardware overheads. The second scheme uses generic non-scan BIST, and can offer some tradeoffs between hardware overhead, test application time and power dissipation. A designer can easily select an appropriate design parameter based on the desired tradeoff. Experimental results confirm the good performance and practicality of our new approaches.

key words: *design for testability, RTL data path, built-in self-test, low power testing, test scheduling*

1. Introduction

Non-scan built-in self-test (BIST) is a promising approach that can realize at-speed testing with a short application time. However, existing BIST schemes have excessive hardware overheads. Moreover, the excessive power dissipation during these BIST schemes constitutes a considerable problem in some applications.

The techniques in [1], [2] propose a test synthesis and scheduling algorithm under power constraints for BISTed register-transfer level (RTL) data paths. These proposed techniques, which use adjacent non-scan BIST [3], may exhibit high hardware overheads due to the use of an excessive number of reconfigured registers.

Masuzawa et al. [4] propose a BIST methodology for RTL data paths that uses a boundary non-scan BIST scheme. The approaches in [5], [6] improve the method in [4] by introducing concurrent testing, exploiting time division between existing test pattern generators (TPGs), so that two different input ports of a module can share the same TPG. However, these previous works did not consider the problem of power dissipation during test. More specifically, when these methods try to excite a single module under test (MUT), and observe its test response, multiple modules and registers, that are not adjacent to the data path of the MUT, also dissipate power. As a result, the accumulated power dissipation is quite high. For some applications, this high power dissipation is unacceptable. Furthermore, hardware overheads in these methods are way too high. As we show in this paper, lower hardware overheads are achievable, while still limiting the power dissipation during test. In [3] TPGs and response analyzers (RAs) are placed not only at the chip boundary, but also inside the data path itself. We will continue to utilize this approach in this paper as well.

In this paper, we introduce two power-constrained DFT algorithms. The first uses a boundary non-scan BIST scheme that focuses on achieving a low hardware overhead (referred to in the paper as "problem 1"). This scheme, therefore, is more efficient in reducing the hardware overhead than previously described methods. The second algorithm is based upon a general non-scan BIST scheme that explores possible trade-offs between hardware overhead and test application time under power constraints (referred to in this paper as "problem 2"), rather than consider only one such factor, as previous published power-constrained methods do.

This paper is organized as follows. Section 2 introduces some basic concepts, such as the data path digraph, and outlines the problems to be solved. Section 3 addresses the power constraints for problem 1 and shows algorithms for performing the test while meeting the given constraints. Section 4 addresses the same issues for problem 2. Section 5 reports on some experimental results using our proposed schemes. Section 6 concludes with a brief summary.

2. Preliminaries

2.1 The Data Path Digraph

A data path [4] consists of hardware elements and lines. Hardware elements, in this context, include primary inputs (PIs), primary outputs (POs), registers (Rs), multiplexers (MUXes), and functional modules (Ms) that have any number of input ports and one output port. Since the multiplexing function can be embedded within an M, we will use the term M in this wider sense of its capability (including multiplexing). Input patterns enter the circuit through the PIs, and exit through the POs. Input values enter into a hardware element through its input ports, and exit through its output port.

Manuscript received September 22, 2004.
Manuscript revised January 27, 2005.

[†]The authors are with the Graduate School of Information Science, Nara Institute of Science and Technology (NAIST), Ikomashi, 630–0192 Japan.
[††]The author is with Nara National College of Technology, Yamatokoriyama-shi, 639–1080 Japan.
[†††]The author is with New Jersey Institute of Technology (NJIT), USA.
a) E-mail: you-z@is.naist.jp
b) E-mail: yamaguti@info.nara-k.ac.jp
c) E-mail: kounoe@is.naist.jp
d) E-mail: savir@njit.edu
e) E-mail: fujiwara@is.naist.jp
DOI: 10.1093/ietisy/e88–d.8.1940

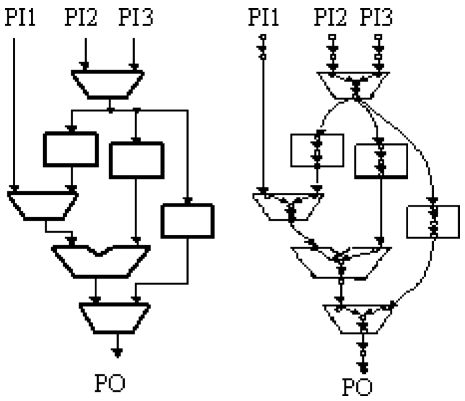


Fig. 1 A data path and its associated digraph.

For any given data path, we assume that every non-constant input port of any M has at least one path from some PI, and every output port of any M has at least one path to some PO.

Similar to the definition in [4], we define a data path digraph $G = (V, A)$ as follows.

- $V = V_H \cup V_{IN} \cup V_{OUT}$, where
  - V_H is the set of nodes that correspond to all hardware elements in the data path. Let $V_H = V_M \cup V_R \cup V_{OTH}$, where, V_M, V_R and V_{OTH} are the set of nodes which represent modules, registers and other hardware elements respectively.
  - V_{IN} is the set of nodes which correspond to all input ports in the data path, and
  - V_{OUT} is the set of nodes which correspond to all output ports in the data path.
- $A = A_1 \cup A_2 \cup A_3$, where
  - A_1={$(x, y) \in V_{OUT} \times V_{IN}$ | output port x is connected to input port y by a line } ,
  - A_2={$(y, u) \in V_{IN} \times V_H$ | y is an input port of u } , and
  - A_3={$(u, x) \in V_H \times V_{OUT}$ | x is an output port of u }.

Note that in a digraph, each PI or PO corresponds to a pair of nodes, and not to a single node. For example, Fig. 1 shows a data path fragment with its associated digraph.

An input port $i_j \in V_{IN}$ is an input port of a node $u_M \in V_M$, such that they are connected together by an arc in A_2. We denote the arc outgoing from node u_M by e_M; and the head node of an arc e by h_e. The sequential depth of a path is the number of register elements along the path.

2.2 Definitions

We define the following two concepts.

Definition 1: A data path is *boundary non-scan BIST-able* if each module M in the data path can be tested as follows.

There exists a TPG for each input port of M, and an RA (response analyzer) for the output port of M such that

(I-i). TPGs and RAs are placed only at PIs and POs respectively.

(I-ii). There are paths that propagate test patterns generated by the TPGs to the input ports of M, and test responses of M to the corresponding input ports of the RA, concurrently, without any conflict of control signals.

(I-iii). For any two input ports of any M, test patterns can either be propagated to these from two different TPGs, or from the same TPG, provided it has different sequential depths leading to these two ports. ■

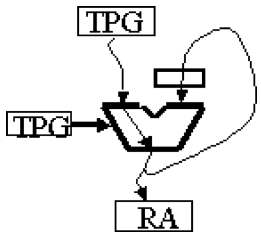


Fig. 2 A module with a type 3 path.

Notice that we allow test patterns to be propagated through a module M using its *thru input function*, if such a function exists. Thus, a module with a thru input can be operated in a transparent mode to pass test patterns generated upstream to other components downstream.

In Definition 1, the control signals include select signals for MUXes; hold inputs for registers, and thru inputs for functional modules.

Definition 2: A data path is *non-scan BIST-able* if each module M in the data path can be tested as follows.

There exists a TPG for each input port of M, and an RA for the output port of M, such that properties (II-i), (I-ii), and (I-iii) in Definition 1&2 hold.

(II-i). TPGs and RAs can be placed at PIs and POs respectively, and any register inside the data path can be a candidate for augmentation into a TPG or an RA. ■

In boundary non-scan BIST, and non-scan BIST schemes, we categorize the different types of *control paths* that propagate test patterns from TPGs to the inputs of a module under test. We distinguish, therefore, between the following cases:

Type 1: A control pattern can be chosen such that no two input ports of M share a TPG.
Type 2: Some input ports share a TPG with paths of different sequential depths.
Type 3: Some input ports share a TPG, and the control path for one of its input ports passes through another input and output ports of this same module (See Fig. 2).

An *observation path* propagates test responses from the output port of a module to an RA. In the sequel, we will refer to both control paths and observation paths simply as test paths.

2.3 Problem Description

Two problems have been formulated in [3] and are repeated here. Let f_H(HOH,TAT) be a test overhead cost function, such that $f_H(h_1, t_1) < f_H(h_2, t_2)$ if $h_1 < h_2$ or ($h_1 = h_2$ and $t_1 < t_2$). The "hardware" argument reflects hardware overhead (HOH), and the "time" argument of the function reflects test application time (TAT).

Problem 1: Minimize the hardware overhead of a given data path under a boundary non-scan BIST, and a test scheduling algorithm, subject to a given power constraint. Stating it more formally,

Given:

- **Input:** a data path and peak power dissipation limit P_{max}.

Task:

- **Output:** a boundary non-scan BIST-able data path, a test schedule that satisfies P_{max}, and that achieves the
- **Objective:** minimization of f_H(HOH,TAT), i.e. minimize hardware overhead. ■

In order to achieve this task we are allowed to add DFT elements, such as linear feedback shift registers (LFSRs), multiple-input signature registers (MISRs), test MUXes (T_MUXes), hold functions for registers, and thru-functions for functional modules.

Problem 2: Given a design parameter α, design a non-scan BIST-able data path, and a test scheduling algorithm, under a given power constraint. More formally,

Given:

- **Input:** a data path, co-optimization ratio α ($0 \leq \alpha \leq 1$), and a peak power dissipation limit P_{max}.

Task:

- **Output:** a non-scan BIST-able data path, a test-schedule satisfies P_{max}, and that achieves the
- **Objective:** minimization of α·HOH+$k(1-\alpha)$·TAT[†]. ■

In order to achieve this task, we are allowed to add DFT elements, such as Built-In Logic-Block Observations (BILBOs) [7], concurrent BILBOs (CBILBOs) [8], LFSRs, MISRs, T_MUXs, hold functions for registers, and thru-functions for functional modules.

3. Power Constrained DFT Algorithm for Problem 1

3.1 Algorithm Description

This algorithm consists of the following three phases.

Phase 1. Convert the given data path to a boundary non-scan BIST-able one utilizing the following steps:

1. Eliminate *critical arcs* for modules.
2. Add thru-functions for functional modules whenever necessary.

Phase 2. Determine the test paths for each module. If the power constraint is violated, consider adding minimum number of T_MUXes to bypass some paths to reduce power. Determine the test paths again until the modules can be tested one by one, while satisfying the power constraint.

Phase 3. Schedule the test.

3.2 Critical Arc Elimination

Definition 3: For a data path digraph G and an arc e, let G_e be a digraph obtained from G by deleting e. An arc e is critical for a node $u_M \in V_M$ if one of the following three cases holds (for the sake of simplicity we state the conditions for modules with two ports only):

Case 1: None of the input ports of u_M is reachable from any PI in G_e, and the sequential depth of any path from h_e to the two ports is identical.

Case 2: None of the input ports of u_M is reachable from any PI in G_e; the sequential depths of any path from h_e to the two ports are different, and no PO is reachable from u_M in G_e.

Case 3: Let u_{M_1}, u_{M_2} and u_{M_3}, be members of V_M, and let e_{M_1} and e_{M_2} be the outgoing arcs from nodes u_{M_1} and u_{M_2} respectively. Arcs e_{M_1} and e_{M_2} are critical for u_{M_3} if no PO is reachable from u_{M_3} in $G_{e_{M_1}}$ or $G_{e_{M_2}}$, and input port i_j of u_{M_3} is unreachable from any PI in $G_{e_{M_j}}$, for j=1,2, respectively.

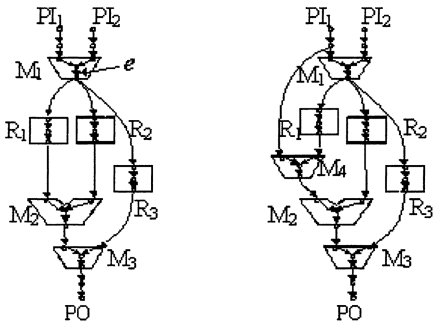
If e is a critical arc of u_M, we say u_M is dominated by e. ■

The hardware area of a T_MUX is usually higher than that of a module-embedded thru-function. There are, however, instances where only T_MUXes can be used to establish the desired testability. These instances occur when there is a need to eliminate critical arcs. We, therefore, consider adding a minimum number of T_MUXes into the data path only when it is necessary.

Theorem 1: If all modules have thru-functions for their input ports, a data path is boundary non-scan BIST-able if and only if (iff) there does not exist a critical arc in its associated digraph. ■

If more than one module are dominated by a critical arc, the order by which we handle these modules plays a key role in reducing the overall hardware overhead. To determine this order, we introduce notions that reflect the relationship between two dominated modules, called a *down-stream module* (DSM), and an *up-stream module* (USM).

[†]k is a unit conversion constant with value $|k|$=1.



(a)Before adding a T_MUX (b)After adding a T_MUX for e

Fig. 3 Example of adding a T_MUX to eliminate a critical arc.

For a dominated node $u_M \in V_M$ of a data path digraph G, let $E(u_M)$ be the set of critical arcs of u_M.

Definition 4: For two dominated nodes u_M and u'_M, we say that u_M is the *up-stream module*, iff u_M is a predecessor of u'_M in the digraph G', where $G'.V = G.V$, $G'A = G.A - E(u_M) - E(u'_M)$, or conversely, we say that u'_M is the *down-stream module* (DSM) iff u'_M is a successor of u_M in the digraph G', provided the dominating critical arc does not meet the condition stated in case 1 of definition 3. ■

From the above definition, the following theorem follows.

Theorem 2: If M is the USM of M', the critical arcs of both M and M'can be eliminated by introducing a T_MUX to add a path from one PI to some other input port of M. Similarly, if M' is a DSM of M, the critical arcs of both M and M' can be eliminated by introducing a T_MUX to add a path from the output port of M' to some PO. ■

Figure 3 illustrates how to eliminate a critical arc. From Definition 3, and the original data path digraph (Fig. 3 (a)), we find that both modules, M_2 and M_3, have one critical arc e in Fig. 3 (a). M_2 is the predecessor of M_3, in other words, M_2 is the USM of M_3. Therefore, according to Theorem 2, addition of a T_MUX (M_4, in Fig. 3 (b)) to establish a path from PI_1 to one input port of M_2, eliminates the critical arc e for both modules. The data path digraph after adding the T_MUX for e is shown in Fig. 3 (b).

The problem of adding a minimum number of T_MUXes to eliminate critical arcs is equivalent to the minimum prime-implicant covering problem, which is known to be NP-hard. We, therefore, use a greedy algorithm, where we select a dominated module that can eliminate critical arc(s) of the maximum number of dominated modules, by adding an extra path to that module. We repeat this algorithm until we eliminate all the critical edges in the system.

3.3 Thru-Function Addition

After adding the necessary T_MUXes, we consider adding a minimum number of thru-functions, whose hardware overhead is usually lower than that of a T_MUX, in order to achieve boundary non-scan BIST-ability. First, we add some necessary thru-functions as described in the following theorem.

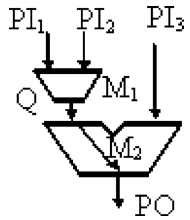


Fig. 4 Need for adding a thru-function.

Theorem 3: If there exists a module M, that is an immediate successor of another functional module M', then an addition of a thru-function to M' is needed to test M. ■

After adding the necessary thru-functions, it may still be possible that the data path in question is not boundary non-scan BIST-able. We, therefore, may need to add some more thru-functions. In Fig. 4 there is no critical arc. However, a thru function from Q to PO needs to be added in order to facilitate vector propagation through module M_2.

3.4 Control Paths and Observation Paths Determination

After the thru-function addition, the data path is boundary non-scan BIST-able. We now determine the control paths and observation path for each module using the shortest, power-weighted, path.

3.5 Bypassing Overly Power Consuming Paths

In a boundary non-scan BIST scheme, TPGs and RAs are placed only at PI and PO sites respectively. Therefore, some modules may end up having long test paths, thus dissipating an extended amount of power. If some modules have long test paths, which dissipate more power than P_{max}, we try to bypass some of them by inserting T_MUXes. In this case, if two or more modules share a portion of their test paths (sub-paths), these modules might be able to share the added bypass as well. In this stage, we search for a minimum number of common sub-paths, so that when being bypassed, the underlying modules satisfy the given power constraints. This problem is also equivalent to the minimum prime-implicant covering problem. We, therefore, use a greedy algorithm, where we always select the common sub-path such that, if bypassed, it reduces the maximum sum-of-powers for the modules involved. Finally, we add the needed T_MUXes to bypass these sub-paths so identified.

3.6 Test Scheduling

We proceed to obtain the test incompatibility graph defined similarly to that given in [9].

Definition 5: Two modules M_1 and M_2 are test incompatible, if one of the following conditions holds.

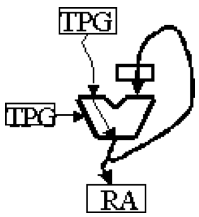


Fig. 5 Essential power for a module with type 3 path.

i. The observation path of M_1 is joined with the test path of M_2.
ii. The control section associated with M_1 is of type 3, and joins the test path of M_2. ■

Since modules can share TPGs and parts of control paths, the power dissipated in these LFSRs and parts of these control paths, need not be accounted for repeatedly, when considering all modules under test. We, therefore, introduce the following concept.

Definition 6: *Essential power dissipation* is:

i. the power consumed by the module itself and its associated observation path, if the test path of the module is either of type 1 or of type 2.
ii. the power dissipated in the tested module, its associated observation path, and its feed-around portion of the control path, if the test path of the module is of type 3. ■

For example, the hardware elements on the bold lines of Fig. 5 (line feeding the RA and the feedback line) dissipate essential power for the module and its type 3 path.

After bypassing the overly power-consuming subpaths, we create the incompatibility graph. In this graph, the nodes are the tested modules, and edges only exist between incompatible modules. We extend the scheduling algorithm from [10] for concurrent testing of multiple modules. In [10] the power is evaluated as the sum of the powers consumed by the individual logic blocks. In our extended algorithm, presented here, two important features come to light:

a. By sharing control paths of different tested modules, we decrease the total consumed power.
b. If it so happens that two modules activate secondary paths off their main test paths, and the paths reach different ports of the same MUX, and since we cannot stop the activity at the MUX, the total power consumed is larger than the sum of the powers of their individual stand-alone paths.

The approach in [10] schedules blocks based on the "necessary" power dissipation. Here we consider "unnecessary" power dissipation, as well as essential power dissipation.

4. Power Constrained DFT Algorithm (Tabu Search-Based) for Problem 2

Figure 6 summarizes the tabu search-based algorithm [11].

```
Algorithm: Power constrained test synthesis and
  scheduling algorithm for Problem 2 (PCTSP2)
1. Generate an initial solution;
2. S_current ← S_init;
3. repeat {
4.   for every register and functional module {
5.     for every possible move that is not in tabu list{
6.       Obtain data path D_i;
7.       if D_i is non-scan BIST-able{
8.         Schedule the test, get S_i;
9.         If it meets power constraints, compute TAT (T_i)
             and HOH (H_i);
10.      }
11.    }
12.  }
13.  Find S_k for which α · H_i+(1 − α) · T_i is minimum;
14.  S_current ← S_k;
15.  Record the move into tabu list;
16.  If this solution is the best so far, then
17.  set S_best ← S_k;
18.}
19.until #iterations>Min{N_itr1, N_itr2}
```

Fig. 6 PCTSP2 algorithm.

Line 1 starts with an initial solution, taken as the solution for Problem 1. Lines 3-19 are the heart of the optimization process. For every register and functional module, we try every possible move[†] , which is not in the tabu list (lines 4-5). After a move, if the data path D_i is non-scan BIST-able, proceed to schedule the test (S_i). If it meets the power constraints, compute the test application time (T_i), and hardware overhead (H_i), (lines 6-9). Here, we treat the internal test registers as either PIs or POs, depending on whether they are used to generate values, or capture responses. We, then, search for a solution[††] S_k that minimizes the value of the cost function $\alpha \cdot H_i + (1-\alpha) \cdot T_i$, and set $S_{current} = S_k$,. This move is then recorded in the tabu list (line 15). If this solution turns out to be the best one so far, we set $S_{best} = S_k$. The algorithm ends when either the maximum number of iterations is reached (N_{itr1}), or the maximum number of iterations since the last obtained best solution exceeds some predetermined value (N_{itr2}).

5. Experimental Results

We have conducted experiments on the data paths of LWF [4], Paulin [12] and Tseng [13]. Table 1 shows the characteristics of these data paths. Columns #Pi, #Po, #R, #Mux, #M, denote the number of PIs, POs, registers, MUXes and functional modules, respectively. Columns "Bit" and "Area" denote bit-width, and the equivalent area as synthesized and reported by the Synopsys Design Compiler.

We first treat modules of type 1 test paths. Let T_M be

[†]A move is a general term for adding/removing thru functions in a module; reconfiguring a register into a BILBO, or CBILBO, adding a hold function to a register, or removing of some previously added hardware.

[††]A solution is a complete test scheduling with established values for TAT, HOH, and the resulting power.

Table 1 Circuit characteristics.

Ckt	#Pi	#Po	Bit	#R	#Mux	#M	Area
LWF	2	2	32	5	5	3	6714
Paulin	2	2	32	7	11	4	36114
Tseng	3	2	32	6	7	7	23234

Table 2 Experimental results for data path of LWF.

Method	α	P_{max} (P_u)	Pow (P_u)	HOH (%)	TAT (T_u)
PCTSP2	0	60	59	32.4	15.5
		65	65	33.4	12.0
		70	65	33.4	12.0
	0.5	60	58	14.3	23.5
		65	58	12.4	23.5
		70	68	9.1	23.5
	1	60	58	14.3	23.5
		65	58	12.4	23.5
		70	68	9.1	23.5
PCTSP1		60	60	21.0	22.5
		65	64	15.7	24.0
		70	68	9.1	23.5
PTCSC		—	69	14.3	15.0

Table 3 Experimental results for data path of Tseng.

Method	α	P_{max} (P_u)	Pow (P_u)	HOH (%)	TAT (T_u)
PCTSP2	0	72	70	27.1	65.5
		82	82	29.5	44.0
		92	92	25.1	41.0
	0.5	72	70	15.4	78.0
		82	81	9.6	65.0
		92	92	8.7	59.0
	1	72	70	12.1	78.0
		82	81	9.6	65.0
		92	86	7.3	93.5
PCTSP1		72	72	12.1	76.5
		82	81	10.2	65.0
		92	92	9.3	59.0
PTCSC		—	77	11.8	103.0

the test application time for a MUX, $T_M = T_u$, where T_u is an integer unit. We assume that the test application time of an adder (T_+), subtractor (T_-), multiplier (T_*), constant-input multiplier ($T_{*'}$), AND gate ($T_\&$), and OR gate ($T_|$) are $T_+=T_-=5T_u$, $T_*=20T_u$, $T_{*'}=3T_u$ and $T_\&=T_|=4T_u$, respectively. The test application time of a module with test path of either type 2, or type 3, are assumed to be $T_{type2}=1.5T_{type1}$, and $T_{type3}=2T_{type1}$, respectively. Let P_u be a standard unit of power. Using the technique in [14], we further assume that the power dissipations for MUX (P_M), AND gate ($P_\&$), OR gate ($P_|$), register (P_{Reg}), adder (P_+), subtractor (P_-), multiplier (P_*), constant-input multiplier ($P_{*'}$), BILBO (P_{BIL}), and CBILBO (P_{CBIL}), are $P_M=P_\&=P_|=P_u$, $P_{Reg}=P_+=P_-=5P_u$, $P_*=20P_u$, $P_{*'}=P_{BIL}=P_{CBIL}=10P_u$, respectively. The hardware overhead, in our experiments, has been determined from the Synopsys Design Compiler for DFT elements.

Tables 2-4 display the experimental results of the Power-Constrained Test Synthesis and Scheduling algorithm for Problem 1 (PCTSP1), Problem 2 (PCTSP2), and the power-driven optimization TCSC (PTCSC) methods. TCSC is our previous methodology [6]. We have extended it here mainly in order to save power by assigning fixed values to unused control signals. Columns α, P_{max}, Pow, HOH and TAT are the co-optimization ratio, peak power dissipation limit, actual peak power dissipation, hardware overhead, and test application time, respectively. Notice that for a fixed P_{max}, the hardware overhead decreases with the increase of α. By the same token, the test application time increases with the increase of α. There is, therefore, a tradeoff between HOH and TAT. Notice that when P_{max} is increasing, the hardware overhead and test application time are both decreasing due to a potentially higher test activity. If we relax the peak power dissipation limit, we can use this relaxation in power to schedule more modules in a given test session, or, equivalently may need less hardware to test the modules in a given test session.

Table 4 Experimental results for data path of Paulin.

Method	α	P_{max} (P_u)	Pow (P_u)	HOH (%)	TAT (T_u)
PCTSP2	0	60	60	25.3	53.5
		100	99	25.1	31.0
		140	137	19.8	28.0
	0.5	60	58	7.0	72.5
		100	87	5.8	61.5
		140	114	3.1	71.5
	1	60	60	6.4	91.5
		100	100	4.9	91.5
		140	114	3.1	71.5
PCTSP1		60	58	7.9	89.0
		100	99	4.9	91.5
		140	114	3.1	71.5
PTCSC		—	112	3.4	82.0

In Table 4, for the case of α=1 and P_{max}=60, notice that PCTSP2 enjoys lesser hardware overhead than PCTSP1. This is because in the non-scan BIST scheme we can add more kinds of DFT elements that will make the approach more hardware-efficient. For cases other than α=1, the results are pretty much the same.

In Tables 2-4, when P_{max} is large enough, the hardware overheads of PCTSP1 and PCTSP2 (for α=1) are lower than that of PTCSC. This shows that our methodology is more efficient, even when there are no power constraints.

6. Conclusions

This paper proposed two power constrained DFT algorithms for two non-scan BIST schemes for RTL data-paths. The first proposed algorithm is for a boundary non-scan BIST scheme. Experimental results have shown that this method is efficient in achieving a low hardware overhead. The second algorithm is for a generic non-scan BIST scheme. We use a Tabu search algorithm to explore the solution space. Experimental results presented here show that it can co-optimize the hardware overhead, test application time, and the power dissipation. A chip designer may utilize these tradeoffs to prioritize one such parameter over the rest.

Acknowledgments

The authors wish to thank Drs. Satoshi Ohtake and Tomokazu Yoneda of Nara Institute of Science and Technology, Tsuyoshi Iwagaki of Japan Advanced Institute of Science and Technology, for their valuable comments. Thanks are also due to the members of Fujiwara laboratory. This work was supported in part by Japan Society for the Promotion of Science (JSPS) under Grants-in-Aid for Scientific Research B(2) (No. 15300018).

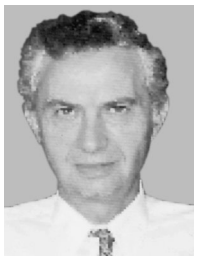
References

[1] N. Nicolici and B.M. Al-Hashimi, Power-Constrained Testing of VLSI Circuits, Kluwer Academic Publishers, Boston, MA, 2003.
[2] N. Nicolici and B. Al-Hashimi, "Power conscious test synthesis and scheduling for BIST RTL data paths," Proc. Int'l test Conf. (ITC2000), pp.662–671, 2000.
[3] Z. You, M. Inoue, and H. Fujiwara, "On the non-scan BIST schemes under power constraints for RTL data paths," Digest of papers 4th Int'l Workshop on RTL and High Level Testing, (WRTLT2003), pp.14–21, 2003.
[4] T. Masuzawa, M. Izutsu, H. Wada, and H. Fujiwara, "Single-control testability of RTL data paths for BIST," Proc. 9th Asian Test Symposium, (ATS2000), pp.210–215, 2000.
[5] K. Yamaguchi, H. Wada, T. Masuzawa, and H. Fujiwara, "A BIST method based on concurrent single-control testability of RTL data paths," Proc. 10th Asian Test Symposium, (ATS2001), pp.313–318, 2001.
[6] K. Yamaguchi, M. Inoue, and H. Fujiwara, "Hierarchical BIST: Test-per-clock BIST with low overhead," Digest of papers 3rd Int'l Workshop on RTL and High Level Testing, (WRTLT2002), pp.42–47, 2002.
[7] B. Koenemann, J. Mucha, and G. Zwiehoff, "Built-in logic block observation techniques," Proc. Int'l Test Conf. (ITC79), pp.37–41, Oct. 1979.
[8] L.T. Wang and E.J. McCluskey, "Concurrent built-in logic block observer (CBILBO)," Proc. The Int. symp. on Circuits and systems, pp.1054–1057, 1986.
[9] R. Chou, K. Saluja, and V. Agrawal, "Scheduling tests for VLSI systems under power constraints," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol.5, no.2, pp.175–185, June 1997.
[10] V. Muresan, X. Wang, V. Muresan, and M. Vladutiu, "The left edge algorithm and the tree growing technique in block-test scheduling under power constraints," Proc. 18th IEEE VLSI Test Symposium (VTS2000), pp.417–422, 2000.
[11] F. Glover and M. Laguna, "Tabu search," in Modern Heuristic Techniques for Combinatorial Problems, ed. C.R. Reeves, pp.70–150, McGraw-Hill Book Company, 1995.
[12] P.G. Paulin and J.P. Knight, "Force-directed scheduling for the behavioral synthesis of ASICs," IEEE Trans. Comput.-Aided Des., Integr. Circuits Syst., vol.8, no.6, pp.661–679, June 1989.
[13] C. Tseng and D.P. Siewiorek, "Automated synthesis of data paths in digital systems," IEEE Trans. Comput.-Aided Des., Integr. Circuits Syst., vol.5, no.3, pp.379–395, July 1986.
[14] E. Macii, M. Pedram, and F. Somenzi, "High level power modeling, estimation, and optimization," IEEE Trans. Comput.-Aided Des., Integr. Circuits Syst., vol.17, no.11, pp.1061–1079, Nov. 1998.

Zhiqiang You received the BS and ME degrees from Hunan University, China, in 1995 and 2002, respectively. He is a PhD candidate in Nara Institute of Science and Technology, Japan. His research interests include low power testing, and Boolean process.

Ken'ichi Yamaguchi received the B.E. degrees from Nara National College of Technology, Advanced Course of Electronics and Information Engineering in 1999. He received the M.E. and Ph.D. in Nara Institute of Science and Technology in 2001 and 2003, respectively. He currently works for Research Associate at Department of Information Engineering, Nara National College of Technology. His research interests include design for testability, built-in-self testing, VLSI design, and educational technology.

Michiko Inoue received her B.E., M.E, and Ph.D degrees in computer science from Osaka University in 1987, 1989, and 1995 respectively. She worked at Fujitsu laboratories Ltd. from 1989 to 1991. She is an associate professor of Graduate School of Information Science, Nara institute of Science and Technology (NAIST). Her research interests include distributed algorithms, parallel algorithms, graph theory and design and test of digital systems. She is a member of IEEE, the Information Processing Society of Japan, and Japanese Society for Artificial Intelligence.



Jacob Savir holds a B.Sc. and an M.Sc. degree in Electrical Engineering from the Technion, Israel Institute of Technology, and an M.S. in Statistics and a Ph.D. in Electrical Engineering from Stanford University. He is currently a Distinguished Professor at New Jersey Institute of Technology (NJIT). During the summer of 2004 he was a visiting professor at the Nara Institute of Science and Technology in Japan. During 2002–2003 he was a Visiting Professor at the Nanyang Technological University in Singapore. Before that, he was the Director of Computer Engineering at NJIT (1996–2000), and Newark College of Engineering Associate Dean for research (1999–2000). Previously with IBM, Dr. Savir was a Senior Engineer/Scientist at the IBM PowerPC Development Center in Austin, TX; at IBM Micro electronics Division in Hudson Valley Research Park; at IBM Enterprise Systems in Poughkeepsie, NY, and a Research Staff Member at the IBM T.J. Watson Research Center, Yorktown Heights, N.Y. He was also an Adjunct Professor of Computer Science and Information Systems at Pace University, N.Y, and SUNY Purchase, N.Y. Dr. Savir's research interests lie primarily in the testing field, where he has published numerous papers and coauthor-ed the text "Built-In Test for VLSI: Pseudorandom Techniques" (Wiley, 1987). Other research interests include design automation, design verification, design for testability, statistical methods in design and test, fault simulation, fault diagnosis, and manufacturing quality. Dr. Savir was awarded the Teruhiko Yamada Memorial Award in 2001. He also received four IBM Invention Achievement Awards, six IBM Publication Achievement Awards, and four IBM Patent Application Awards. He is an associate editor of the Journal of Computer Science and Technology. Dr. Savir is a member of Sigma Xi, and a fellow of the Institute of Electrical and Electronics Engineers.

Hideo Fujiwara received the B.E., M.E., and Ph.D. degrees in electronic engineering from Osaka University, Osaka, Japan, in 1969, 1971, and 1974, respectively. He was with Osaka University from 1974 to 1985 and Meiji University from 1985 to 1993, and joined Nara Institute of Science and Technology in 1993. In 1981 he was a Visiting Research Assistant Professor at the University of Waterloo, and in 1984 he was a Visiting Associate Professor at McGill University, Canada. Presently he is a Professor at the Graduate School of Information Science, Nara Institute of Science and Technology, Nara, Japan. His research interests are logic design, digital systems design and test, VLSI CAD and fault tolerant computing, including high-level/logic synthesis for testability, test synthesis, design for testability, built-in self-test, test pattern generation, parallel processing, and computational complexity. He is the author of Logic Testing and Design for Testability (MIT Press, 1985). He received the IECE Young Engineer Award in 1977, IEEE Computer Society Certificate of Appreciation Award in 1991, 2000 and 2001, Okawa Prize for Publication in 1994, IEEE Computer Society Meritorious Service Award in 1996, and IEEE Computer Society Outstanding Contribution Award in 2001. He is an advisory member of IEICE Trans. on Information and Systems and an editor of IEEE Trans. on Computers, J. Electronic Testing, J. Circuits, Systems and Computers, J. VLSI Design and others. Dr. Fujiwara is a fellow of the IEEE, a Golden Core member of the IEEE Computer Society, and a fellow of the Information Processing Society of Japan.