A Wiring-Aware Approach to Minimizing Built-in Self-Test Overhead

Abdil Rashid Mohamed, Zebo Peng and Petru Eles
Department of Computer and Information Science,
Linköping University, S-581 83, Sweden
{abdmo, zebpe, petel} @ida.liu.se

Abstract

This paper describes a built-in self-test hardware overhead minimization technique used during a BIST synthesis process. The technique inserts a minimal amount of BIST resources into a digital system to make it fully testable. The BIST resource insertion is guided by the results of symbolic testability analysis. It considers both BIST register cost and wiring overhead in order to obtain the minimal area designs. A Simulated Annealing algorithm is used to solve the overhead minimization problem. Experiments show that considering wiring area during BIST synthesis results in smaller final designs as compared to the cases when the wiring impact is ignored.

1. Introduction

Work on BIST insertion at the behavioral and RT-levels can be classified in three different types:

i. BIST insertion at RT-level after high-level synthesis (HLS) is performed without considering the geometrical information (floor-planning and wiring area).
ii. BIST insertion during HLS without taking into account floor-planning and wiring area.
iii. BIST insertion which takes into account floor-planning and wiring area.

The first category of work [1], [2] inserts BIST such that each RT module can receive test patterns from on chip test pattern generators (TPG) and its responses can reach on chip multiple input signature registers (MISR).

The second category of work [3] inserts BIST during HLS such that the processes of scheduling, allocation and testability enhancements are simultaneously carried out. These approaches explore a bigger design space in search for the optimal solutions, and potentially give better designs with respect to area, testability and performance.

The approaches of these two categories ignore impact of placement of the functional and BIST modules in the final design. This usually leads to designs which are optimal in terms of the numbers of functional and BIST resources, but take more silicon area to implement since interconnects take large silicon space in deep sub-micron VLSI designs.

In order to take wiring information of the final designs into account during the BIST insertion process, it is needed to predict the wiring area and lengths at higher levels. Alvandpour [6] developed a heuristic to estimate wiring lengths at RT level. His approach was deployed in [7] to predict area increase due to wiring in a HLS system. Goel et al. [8] have proposed a model for wiring length computation for core based system-on-chip testing, where they assume the layout of the modules is known beforehand.

This work belongs to the third category. The problem of optimizing BIST insertion at behavioral and RT levels while taking into account geometrical information of the design is addressed and solved. The problem is: given a design represented as a scheduled data flow graph (SDFG) along with allocation and binding information, insert BIST modules into the design such that all functional modules are self-testable and the total design area is minimized.

2. Our BIST Synthesis Approach

Simulated Annealing [10] has been used to optimize the BIST structure in order to minimize the total design area. Details of our approach are discussed in Section 2.4.

The approach is assisted by a symbolic testability analysis (STA) methodology [4], [5] which identifies hard to test operations. STA is performed at the behavioral level on the SDFG representation of the design. Testability enhancement is performed on the RT-level representation of the design. The optimization process uses testability design transformations (section 2.1) to explore a design space in search of the smallest self-testable design. Since the objective is to minimize the total area, maximal BIST register sharing is also achieved.

2.1 Testability Design Transformations

A number of BIST design transformations have been defined. They provide the optimization heuristic with a mechanism for performing neighborhood search so as to converge towards low area, self-testable designs. Some transformations enhance testability while others reduce it. Those, which reduce testability, enable the optimization strategy to create intermediate solutions, which later, as a result of further transformations, converge towards minimal area designs while guaranteeing testability. Therefore, for each type of testability enhancement transformation (*TET)* a reverse *TET* (*RTET),* to cancel its effect is also defined. For example, if *TET*(r_k,TPG_k) transforms a functional register r_k, to a TPG_k, then its reverse transformation converts TPG_k to register r_k. It eliminates the effect of *TET*(r_k,TPG_k) and brings the design back to its original state.

Each testability transformation has advantages (reduced area, reduced wiring, and/or improved testability) or disadvantages (additional area, additional wiring, and/or reduced testability).

A conversion for controllability transformation changes controllability by converting an existing functional register to a TPG, which generates on-chip test patterns or converts one type of test pattern generator to another or back to a functional register. The conversion changes design area by $\pm\delta A$, ($\delta A = A_{TPG} - A_{reg}$). The TPG can be converted to a BILBO (Built-in Logic Block Observer), which generates test patterns and compresses test responses at different times. The BILBO is, however, larger than the TPG.

A conversion for observability transformation changes observability by converting an existing functional register to a MISR or converting one type of signature register to another or back to a functional register. A MISR can only compress test responses on chip to a single signature. The BILBO can enhance the observability, but it also improves controllability. Thus, converting the MISR to the BILBO improves both controllability and observability.

A connection for controllability is a transformation whereby an existing TPG or BILBO is connected directly to an input of the functional module to improve its controllability. The transformation necessitates addition of a test multiplexer in front of the functional module. Thus, wiring and multiplexer cost are incurred ($\delta A = A_{wire} + A_{mux}$). Reverse transformation that disconnects the TPG or the BILBO from the module results in reduced testability, but the test multiplexer and wiring can be removed.

A connection for observability is an observability enhancement transformation that connects the output of a module directly to an existing MISR or BILBO. Though the transformation adds wiring and a test multiplexer, it can enhance the observability of any chosen module. Reverse transformation disconnects the MISR or BILBO from the module resulting in reduced observability and a saving of test multiplexer and wiring.

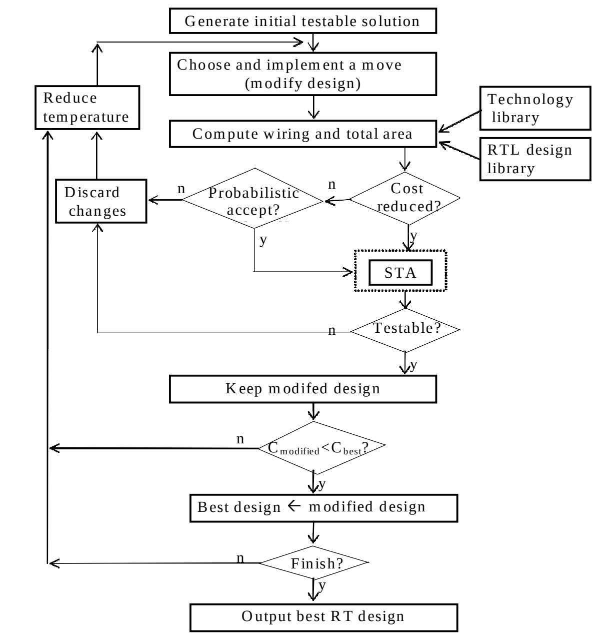

Figure 1. Framework for BIST analysis and optimization

2.2 Wiring Area Estimation

Wiring area, A_w, is estimated using the heuristics in [6] and [7] (see eq.1). They estimate the placement of modules on the chip based on their interconnection relationship and sizes, and computes the lengths of all interconnections.

$$A_w = \left(\left(\sum_i l_i \times w_i \times W_{av} \right) - k_{or} \times A_{nodes} \right) / N_{metal} \qquad \text{eq. 1}$$

The sum over i is for all interconnections. l_i, and w_i are the length and width (in bits) of interconnection i. W_{av} is the average width of wires (including the space between them), k_{or} is the over-route factor (fraction of the area above the datapath nodes and controller units in the metal layers that can be used for routing), A_{nodes} is the area of the datapath nodes, and N_{metal} is the number of metal layers.

2.3 Cost and Objective Functions

The objective is to minimize the total design area, which is the sum of the areas of *functional modules and registers, functional and test multiplexers, BIST registers, and total wiring area* as depicted in eq. 2.

$$A_{total} = A_{freg} + A_{f\,\mathrm{mod}} + A_{fMUX} + A_{BIST} + A_{tMUX} + A_{wire} \qquad \text{Eq. 2}$$

Eq. 3 gives the cost function used to drive the optimization heuristic, where k is a constant and N_{untest} is the number of untestable modules. If a transformation guarantees 100% testability, the total design area is used as cost (eq. 3a). Otherwise either the transformation is rejected or a penalty is paid eq. 3ab). The penalty is a function of the number of untestable modules and discourages transformations that do not guarantee testability. Testability constraints must be satisfied in the final solution generated.

$$Cost = A_{total} + \begin{cases} 0 & \text{eq. 3a} \\ k \times N_{untest} & \text{eq. 3b} \end{cases}$$

2.4 BIST Optimization Heuristic

Our Simulated Annealing (S.A.) based optimization framework is depicted in Figure 1. A fully testable initial solution, X_0, is an input to the heuristic. To get X_0, all constant nodes and primary input (PI) registers are converted into TPGs, all primary output (PO) registers to MISRs and all internal registers to BILBOs.

Testability transformations, which modify the design to produce neighboring solutions, are randomly applied. Although the aim is to converge towards minimal area, in order to escape from local minima, S.A. can probabilistically accept expensive transformations hoping that subsequent transformations may enable the optimization strategy to converge towards a global optimum. As an optimization parameter, called temperature, decreases, the probability of accepting expensive transformations decreases [10]. At low temperatures only the cheap solutions are accepted and the annealing converges towards minimum area designs. After each transformation, the testability status (testable, not testable) is checked by STA.

3. Experimental Results

Technology dependent parameters used are: average width of a 1-bit wire including spacing between the wires is $0.8\mu m$. The number of metal layers is 2 and over routing factor is 0.5. Sizes of the functional registers and modules are extracted from [9]. The areas of the 16-bit modules in μm^2 are: Multiplier-250000, adder-50000, subtractor-50000, register-15000, TPG-20000, MISR-30000, BILBO-40000, CBILBO is 50000 and multiplexer is 1000 + number_of_inputs * 500.

Tables 1 through 3 summarize the experimental results. Columns *P*, *M* and *B* represent the number of TPGs, MISRs and BILBOs respectively. A_{nodes} is the sum of the node areas, A_{wires} is the sum of all wiring areas, N_{mux} is the number of test multiplexers, A_{mux} is the area of test multiplexers and $\%A_w/A_t$ is the percentage of wiring area with respect to total design area. A_{total} represents the total area of the design after synthesis including wiring area. In Table 3, $A_{t\_with\_wire}$ and $A_{t\_no\_wire}$ are the respective total design areas with and without considering wiring during BIST optimization. $\%A_{loss}$ is the percentage of area that is lost if wiring is not considered as compared to the case when wiring is considered.

The BIST synthesis process was run in two different ways: In the first case we have ignored wiring area ($A_w=0$) during BIST synthesis optimization. The minimized cost is the total node area (data-path nodes, BIST registers and test multiplexers). The results are shown in Table 1. In the second case, we have taken into account wiring area during the BIST synthesis. The minimized cost is the sum of total node area and *wiring area*. Table 2 summarizes the results.

When wiring area is considered during BIST synthesis, we get smaller total design area (see Table 3), but the BIST registers occupy larger area than what they occupy if wiring is not considered. Since the designs obtained when wiring area is considered occupy smaller total area, this shows that ignoring wiring cost produces less efficient designs.

Table 1. Wiring area ignored during optimization

Design	P	M	B	$A_{nodes}(\mu m^2)$	$A_{wire}(\mu m^2)$	$A_{total}(\mu m^2)$	$\%A_w/A_t$	N_{mux}	A_{mux}
Paulin	2	1	0	1,971,000	558,067	2,529,067	22.07	9	21,000
Real	2	1	0	**2,051,500**	550,125	2,601,625	21.15	10	21,500
Ovenctrl	2	1	0	**1,054,500**	338,128	1,392,628	24.00	9	19,500
EX2	2	1	0	**1,586,000**	300,040	1,886,040	15.91	8	16,000
EWF	2	2	1	4,088,000	1,816,345	5,904,345	30.76	36	78,000

Table 2. Wiring area considered during optimization

Design	P	M	B	$A_{nodes}(\mu m^2)$	$A_{wire}(\mu m^2)$	$A_{total}(\mu m^2)$	$\%A_w/A_t$	N_{mux}	A_{mux}
Paulin	5	1	0	1,968,000	328,569	2,296,569	14.03	1	3,000
Real	6	1	1	**2,081,000**	340,214	2,421,214	14.05	3	6,000
Ovenctrl	6	1	0	**1,058,500**	236,124	1,294,624	18.24	1	3,500
EX2	5	1	0	**1,589,000**	215,166	1,804,166	11.93	2	4,000
EWF	10	2	0	4,034,500	971,507	5,006,007	19.41	4	9,500

Table 3. Area loss if wiring is not considered

Design	$A_{t\_no\_wire}(\mu m^2)$	$A_{t\_with\_wire}(\mu m^2)$	$\%A_{loss}$
Paulin	2,529,067	2,296,569	10.12
Real	2,601,625	2,421,214	7.45
Ovenctrl	1,392,628	1,294,624	7.57
EX2	1,886,040	1,804,166	4.54
EWF	5,904,345	5,006,007	17.95

4. Conclusion

We have developed an approach for BIST resources optimization that takes into account wiring area. The heuristic uses Simulated Annealing to optimize designs towards minimum area. Experimental results indicate that when wiring area is considered during the optimization process the total area of the resulting designs is smaller compared to cases when wiring is not considered. The number of BIST registers can be smaller in the latter case.

5. References

[1] "A built-in self-testing approach for minimizing hardware overhead", Chiu, S.S.K., Papachristou, C.A., Computer Design: VLSI in Computers and Processors. ICCD '91. Proc., 1991 IEEE International Conference on, 14-16 Oct 1991 Page(s): 282 -285.

[2] "BIST hardware synthesis for RTL data paths based on test compatibility classes Nicolici", N.; Al-Hashimi, B.M.; Brown, A.D.; Williams, A.C.; Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on, Volume: 19 Issue: 11, Nov 2000 Page(s): 1375 -1385

[3] L. Avra, "Allocation and assignment in high-level synthesis for self-testable data paths", in Proc. Int. Test Conf., p.463-472, 1991.

[4] Ghosh, I.; Jha, N.K.; Bhawmik, S, "A BIST Scheme for RTL Circuits Based on Symbolic Testability Analysis", IEEE Transactions on CAD of Integrated Circuits and Systems, Volume: 19 Issue: 1, Jan. 2000 Page(s): 111 –128.

[5] Ghosh, I.; Jha, N.; Bhawmik, S., "A BIST Scheme for RTL Controller-Data Paths Based on Symbolic Testability Analysis", ACM, 1998.

[6] A.Alvandpour and C.Svensson, "A Wire Capacitance Estimation Technique for Power Consuming Interconnects at High Levels of Abstraction", Workshop on Power And Timing Modeling, Optimization and Simulation (PATMOS97), Louvain-la-Neuve, Belgium, 1997.

[7] Jonas Hallberg, Zebo Peng, "Estimation and Consideration of Interconnection Delays during High Level Synthesis", 24th EUROMICRO Conf., 1998.

[8] Goel, S.K.; Marinissen, E. J., "Layout-Driven SoC Test Architecture Design for Test Time and Wire Length Minimization", Design Automation & Test in Europe, 2003.

[9] Vasily G. Moshyaga, Keikichi Tumaru, "A Placement Driven Methodology for High-Level Synthesis of Sub-Micron ASICS", International Symposium on Circuits and Systems, May 1996, Page(s) 572-575.

[10] Reeves, C. "Modern Heuristic Techniques for Combinatorial Problems", Blackwell Scientific Publications, 1993.