IEEE the 12th Asian Test Symposium (ATS'03), pp.300-303, Nov. 2003.

Non-Scan Design for Testability for Mixed RTL Circuits with Both Data Paths and Controller via Conflict Analysis

Dong Xiang Shan Gu
School of Software
Tsinghua University
Beijing, 100084
China

Hideo Fujiwara
Grad. School of Inform. Sci.
Nara Institute of Sci. and Techn.
Ikoma, Nara 630-0101
Japan

Abstract

A non-scan design for testability method for RTL circuits based on conflict analysis is proposed. Conflict analysis is presented based on a new 5-valued system to estimate testability of data paths. New test point structures for RTL circuit design for testability are introduced. Non-scan design for testability is proposed based on conflict resolution. Unlike most of the previous methods, our method considers testability of data paths and the controller simultaneously. Different classes of test points can be inserted into data paths and the controller. Intensive techniques are presented to connect extra inputs of test points with PI ports, which avoids generating reconvergent fanouts that cause new conflicts.

1 Introduction

Non-scan design for testability has attracted extensive studies. It is essential to present a good testability measure which can reflect the actual testability of an RTL circuit in the process of test generation. Dey and Potkonjak [2] introduced a new testability measure called $k-$level controllability/observability to break cycles of the EXU $s-$graph for RTL circuits. Test multiplexers were inserted to avoid equal weight reconvergent fanouts. Ghosh, Raghunathan, and Jha [3] proposed a non-scan design for testability of RTL circuits using a testability measure independent of data path widths. Complete or near complete test efficiency was obtained for almost all circuits with very low area, and power overheads. Ohtake *et. al.* [6] proposed new concepts called strong testability and fixed-control testability to handle non-scan DFT for RTL circuits, which achieved complete test efficiency for RTL circuits. Makris and Orailoglu [4] analysed correlation between blocks for test justification and propagation, which was further utilized to guide design for testability.

We shall propose a conflict-analysis-based measure for RTL circuits. A couple of techniques are utilized to estimate testability of the RTL circuit in order to emulate the actual testability of a sequential circuit during test generation. Some definitions and notation of the paper should be introduced first. An *assignment* is a 2-tuple (A, v), which means a data path A is assigned a value v, where $v \in \{1, 0,$ $a1$, *even*, *odd*$\}$. Let $v, v^{'} \in \{1, 0, a1, even, odd\}$.

Definition 1 *Let a data path l be assigned value v in the previous process of testgeneration, and l needs to be assigned value v' at the same clock cycle. If intersection of v and v' produces a new covered value, the line l is assigned $v \cap v'$; otherwise, a conflict occurs on l.*

Definition 2 *A mandatory assignment is defined as the easiest assignment of a data path in order to meet a specific signal requirement.*

When all assignments are mandatory, a conflict indicates the fault under consideration is redundant; otherwise, it can be resolved by backtracking. The main cause of conflicts is still reconvergent fanouts with uniform delay.

Definition 3 *Sequential depth for testability $seq_v(l,s)$ $(v \in \{0,1\})$ in an RTL circuit from a fanout stem s to a data path l is defined as a set of $3-$tuples (p, d, v) to justify an assignment (l, v) at the data path l to the fanout stem d in the easiest way, where p, d, and v stand for a path from l to s, the number of registers in the path, and the required value at s according to the conflict-analysis-based mandatory assignments.*

Definition 4 *Controllability measure $C_l(v)$ of a data path l is defined as the number of conflicts or the number of clock cycles required to assign a value $v \in \{0, a1, 1, even, odd\}$ on l. Observability $O_l(v)$ is defined as the number of conflicts or the number of clock cycles to propagate the fault effect $v \in \{1, a1, even, odd\}$ of data path l to a primary output.*

This work is in part supported by JSPS under grant L03540, and in part by the national science foundation of China under grant 69773030, and in part by the 985 fundamental foundation of national education ministry.

2 A 5-valued System and Symbolic Testability Estimation

In our method, a modified value system is adopted. The system contains 5 different values: 1, 0, $a1$, *even*, and *odd*. The value g in [3] is split into *odd* and *even*, which present more direct information for data paths. As for a multiplier, fault effects of an input can be propagated if another input is assigned value *odd* consider data widths for inputs and output of the multiplier are the same. Testability of data paths is estimated based on conflict analysis [7,8].

2.1 Conflicts Generated by Assignment Justification

It should be noted that only assignments at outputs of arithmetic and logic units may cause conflicts. We consider potential conflicts at the outputs of functional units. Let a and b be inputs of an adder with output l.

$$\begin{aligned}
C_l(0) &= C_a(0) + C_b(0) + p,\\
C_l(1) &= min(C_a(0) + C_b(1) + p, C_a(1) + C_b(0) + p),\\
C_l(a1) &= min(C_a(0) + C_b(a1) + p, C_a(a1) + C_b(0) + p),\\
C_l(o) &= min(C_a(0) + C_b(o) + p, C_a(o) + C_b(0) + p,\\
&\quad C_a(e) + C_b(1) + p, C_a(1) + C_b(e) + p,\\
&\quad C_a(o) + C_b(e) + p, C_a(e) + C_b(o) + p),\\
C_l(e) &= min(C_a(0) + C_b(e) + p, C_a(e) + C_b(0) + p,\\
&\quad C_a(o) + C_b(1) + p, C_a(1) + C_b(o) + p,\\
&\quad C_a(o) + C_b(o) + p).
\end{aligned}$$

where p indicates the number of potential conflicts to meet the corresponding assignment. The outputs of an adder can be assigned 0 only when both inputs are assigned 0. The output l can be assigned 1 when one input is 1 and the other is 0. Similarly, the output l is assigned $a1$ when one input is set as $a1$ and the other is 0. However, there exist a couple of assignments that set the the adder's output to $a1$. We present a pessimistic calculation. Let a and b be inputs of a multiplier with the output l.

$$\begin{aligned}
C_l(0) &= min(C_a(0), C_b(0)),\\
C_l(1) &= C_a(1) + C_b(1) + p,\\
C_l(a1) &= min(C_a(a1) + C_b(1) + p, C_a(1) + C_b(a1) + p),\\
C_l(o) &= min(C_a(1) + C_b(o) + p, C_a(o) + C_b(1) + p,\\
&\quad C_a(o) + C_b(o) + p, C_a(a1) + C_b(o) + p,\\
&\quad C_a(o) + C_b(a1) + p),\\
C_l(e) &= min(C_a(1) + C_b(e) + p, C_a(e) + C_b(1) + p,\\
&\quad C_a(o) + C_b(e) + p, C_a(e) + C_b(o) + p,\\
&\quad C_a(a1) + C_b(e) + p, C_a(e) + C_b(a1) + p,\\
&\quad C_a(e) + C_b(e) + p).
\end{aligned}$$

The output of a multiplier can be assigned value 0 when one of its inputs is 0. It can be 1 when both inputs are set as 1. The multiplier's output can be assigned value $a1$ when one of its inputs is 1 and the other is set as $a1$. Calculations of other arithmetic and logic units are similar.

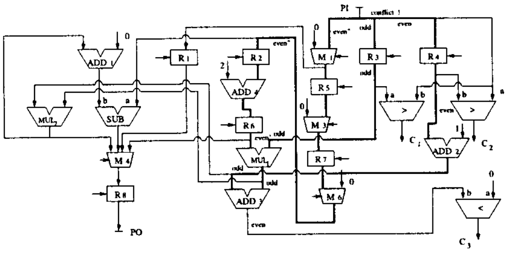

Figure 1: Conflicts caused by mandatory assignments.

We would like to illustrate how to obtain the penalty using an example. The circuit presented in Fig. 1 is a variation of the b04 circuit. Consider an assignment (add_3, odd), we have two equivalent assignments $(R_4, even)$ and (MUL_1, odd). Assignment (MUL_1, odd) has two equivalent mandatory assignments $(R_6, even^{'})$ and (R_3, odd). Justification of assignment $(R_6, even^{'})$ finally generates an assignment $even^{''}$ on the corresponding fanout branch on the PI port. The mandatory assignments are presented in the bold-faced paths as shown in Fig. 1. A conflict generates because paths ADD_3-R_4-PI and ADD_3-MUL_1-R_3-PI have uniform delays and different assignments on PI. Mandatory assignment corresponds to the path ADD_3-MUL_1-R_6-ADD_4-R_2-R_7-R_5 never causes any conflict with the above two paths because of different delays.

2.2 Conflicts Caused by Fault Effect Propagation

Observability measures are defined with respect to different fault effects. Let a and b be inputs of an adder with output l, $v \in \{1, a1, even, \ odd\}$. We have,

$$O_l(v) = min(C_b(0), C_b(1), C_b(a1), C_b(o), C_b(e)) + O_l(v)).$$

Fault effect on an input of a multiplier can be propagated to its output if the other input of the multiplier is assigned 1, *odd*, and $a1$, respectively. The corresponding fault effects on the multiplier's output should be 1, *odd*, and $a1$, respectively. A $a1$ fault effect on an input of a multiplier can be propagated to its output if the other input is assigned 1, *odd*, and $a1$. The corresponding fault effects on its output should be $a1$, *odd*, and *odd*, respectively. An odd fault effect on an input of a multiplier can be propagated to its output if the other input is assigned one of 1, *odd*, and $a1$. The corresponding fault effect on its output should be *odd* in three cases. An *even* fault effect on an input of a multiplier generates an *even* fault effect in all cases when the other input is assigned value 1, *odd*, and

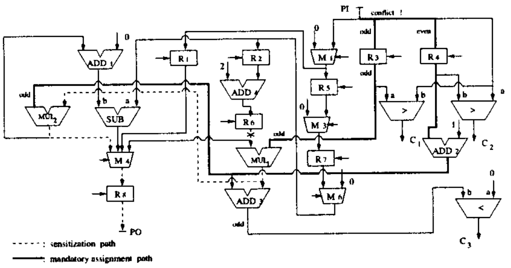

Figure 2: Conflicts caused by mandatory assignments for fault effect propagation.

$a1$, respectively. Let a and b be inputs of a multiplier, and l be the output, we have,

$$\begin{aligned}
O_a(1) &= min(C_b(1) + O_l(1) + p, C_b(o) + O_l(o) + p, \\
&\quad C_b(a1) + O_l(a1) + p), \\
O_a(a1) &= min(C_b(1) + O_l(a1) + p, C_b(o) + O_l(o) + p, \\
&\quad C_b(a1) + O_l(o) + p), \\
O_a(o) &= min(C_b(1) + O_l(o) + p, C_b(o) + O_l(o) + p, \\
&\quad C_b(a1) + O_l(o) + p), \\
O_a(e) &= min(C_b(1) + O_l(e) + p, C_b(o) + O_l(e) + p, \\
&\quad C_b(a1) + O_l(e) + p).
\end{aligned}$$

Fault effect propagation may also generates conflicts. Let us propagate a fault effect from the output of R_6 to the primary output port. The dashed lines in Fig. 2 are the fault effect sensitization paths, and the bold-faced paths are mandatory assignment paths. An *odd* is required on the sensitization path R_3 when propagating the fault effect on data path R_6. An *odd* is necessary on data path ADD_2 in order to sensitize the fault effect through MUL_2. Justifications of the above assignments definitely cause a conflict at the PI port. It should be noted that sequential delays of two paths MUL_2-ADD_2-R_4-PI and MUL_1-R_3-PI are the same and no register exists in the path between MUL_1 and MUL_2.

3 New Testability Structures and Testability Insertion

Most of the current non-scan design for testability methods insert test multiplexers into the circuit directly [1,2,3,4,6]. These methods need an additional register to control the selecting inputs of the test multiplexers. A succinct design for testability architecture is proposed, which needs only one extra pin to control the whole DFT system. New test point structures are proposed for RTL circuits, which are economical in delay, area, pin overheads, and test application cost.

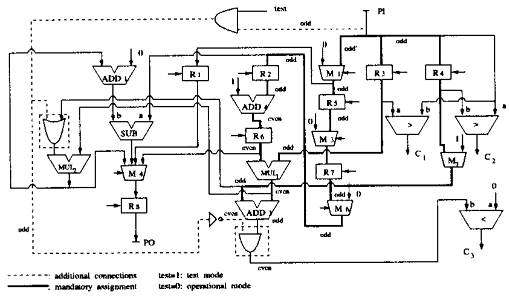

Figure 3: PI port sharing.

A new testability structure is introduced in this paper. The structure does not insert the multiplexers into the functional paths directly unlike [1,2,3,4,6], which causes only one gate delay as shown in Fig. 3. Only one extra pin *test* is utilized to control testing of the system. The DFT circuit is set to test mode when *test* = 1, and operational mode when *test* = 0. When *test* = 1, no signal is blocked in the test circuit any more. Four different classes of control test points are inserted into the circuit because the synthesized RTL circuit is a mixed one with both logic gates and RTL units. The controller contains only logic gates and flip-flops. The control test points are logic 1−control, logic 0−control, vector AND, and vector OR test points. It is clear that a vector AND test point can improve *even*−controllability and 0−controllability of a data path, and a vector OR test point can improve *odd*−controllability and $a1$−controllability of a data path.

As shown in Fig. 3, a vector OR control test point and a vector AND control test point are inserted into the output of ADD_3 and the input of MUL_2. The dashed lines present the the additional connections, and dashed boxes shown the inserted test points. The bold-faced lines demonstrate mandatory assignments with respect to the *even* assignment at the data path b. The same *odd* number can be assigned for the mandatory assignment paths ADD_3-M_2-R_4 and ADD_3-MUL_1-R_3 although both paths have the same delay. The assignments on both data paths generate no conflict with additional connection path and ADD_3-MUL_1-R_6-R_2-R_7-R_5 because of different delays. Similarly, all other assignments on data path b also generate no conflict. The inserted vector OR test point also causes no conflict.

More than one test point can share the same PI port. It is better for both data paths inserted vector test points not to converge, where no conflict occurs. Otherwise, our algorithm checks each reconvergent point of two data paths. The algorithm generates all mandatory assignments for each assignment at the convergent data path. Both test points cannot share the same PI port if two of the mandatory assignments cause conflicts. As shown in Fig. 3, both control test points share the same PI port because MUL_2 and data path b do not converge in the original data path.

Table 1: Comparison of *rtldft* with *lcdft* [8].

circuits	#PIs	rtldft					lcdft [7]				
		FC (%)	TE (%)	vec	cpu	ao (%)	FC (%)	TE (%)	vec	cpu	ao (%)
b01	2	100	100	108	0.02	3.92	100	100	108	0.001	3.92
b02	1	100	100	82	0.001	5.88	100	100	86	0.02	8.82
b03	4	98.75	99.0	243	12.34	1.94	98.75	99.0	217	5.75	1.94
b04	11	98.87	99.37	421	22.08	1.72	97.37	98.65	317	642	1.72
b05	1	86.93	92.95	610	419.47	5.16	85.81	90.21	636	602	5.16
b06	2	100	100	81	0.001	4.00	100	100	63	0.017	5.33
b07	1	88.24	90.33	622	323.7	1.54	74.56	76.57	248	609	1.54
b08	65	97.40	99.57	577	16.08	2.68	97.40	98.92	356	11.7	2.68
b09	1	98.33	100	669	17.05	2.25	99.08	99.56	836	22.1	1.80
b10	11	97.52	99.24	436	12.18	2.60	97.31	99.81	479	10.35	2.60
b11	7	91.55	94.42	715	261.23	2.05	74.52	77.10	247	789	2.05
b13	10	91.23	99.34	612	23.73	7.13	0.00	0.00	–	–	7.13
b14	32	90.23	90.40	5538	21672	4.53	89.74	89.77	5382	15864	4.53
b17	37	84.54	85.28	6168	54 h	2.24	84.38	85.13	6460	99.6 h	2.24

Both the RTL conflict-based testability measure and the gate-level testability measure *conflict* [7] are adopted to select test points because the synthesized circuit is a mixed one. It is not so easy to identify the controller from the circuit. Selection of test points is still based on the selective tracing procedure in [7]. The selective tracing technique updates only the active part of the circuit after a vector test point has been inserted. The method can select both gate level test points and RTL test points. A test point may make both gate level units and RTL units more testable. Therefore, a weighted gain function is utilized to select test points. The following weighted gain function is adopted,

$$TG = \sum_i w \cdot \Delta T_1(i) + \sum_i \Delta T_2(i),$$

where the first part represents testability improvement of RTL units, the second part testability improvement of gate level units, and w is a weight (suggested value is the data path width). The selective tracing technique is adopted to calculate testability gain or updates testability after a testability structure has been inserted.

4 Experimental Results

We have implemented the proposed method. All benchmark circuits are synthesized with the *Synopsys* system. Table 1 presents comparison of the proposed method called *rtldft* and a recent gate level non-scan DFT tool called *lcdft* [8]. The *lcdft* selects test points based on the *conflict* testability measure [7], which tries to resolve as many as possible ATPG conflicts. As shown in Table 1, #PIs is the number of primary inputs of the gate level circuits. *FC, TE, vec, cpu*, and *ao* represents fault coverage, test efficiency, the number of vectors, CPU time required, and area overhead, respectively. Compared with *lcdft*, the proposed method obtains better fault coverage than *lcdft* except circuit b09. Both methods do not work well on several controller circuits very small number of primary inputs, such as b05 and b07. Both methods get the same fault coverages for circuits b01, b02, b03, b06, and b08. The proposed method *rtldft* obtains much better fault coverage than *lcdft* on circuits b07, b11, and b13.

5 Conclusions

Non-scan design for testability for RTL circuits is proposed based on conflict analysis. A new testability measure for RTL circuits was introduced based on a 5-valued system, which is a refinement of the multiple valued system in [3]. A new definition called mandatory assignment is presented to analyse conflicts for RTL circuits. New test point structures for RTL circuits are introduced, which are economical in pin, delay, and area overheads.

References

[1] V. Chickermane, E. M. Rudnick, P. Banerjee, and J. H. Patel, "Non-scan design for testability techniques for sequential circuits," Proc. of *ACM/IEEE Design Automation Conf.*, pp. 236-241, 1993.

[2] S. Dey and M. Potkonjak, "Non-scan design for testability techniques using RT-Level design information," *IEEE Trans. CAD of ICAS*, vol. 16, no. 12, pp. 1488-1506, 1997.

[3] I. Ghosh, A. Raghunathan, and N. K. Jha, "Design for hierarchical testability of RTL circuits obtained by behavioral synthesis," *IEEE Trans. CAD of ICAS*, vol. 16, no. 9, pp. 1001-1014, 1997.

[4] Y. Makris and A. Orailoglu, "DFT guidance through RTL test justification and propagation analysis," Proc. of *IEEE Int. Test Conference*, pp. 668-677, 1998.

[5] P. C. Maxwell, R. C. Aitken, V. Johansen, and I. Chiang, "The effect of different test sets on quality level prediction: When is 80% better than 90% ?" Proc. of *IEEE Int. Test Conf.*, pp. 358-364, 1991.

[6] S. Ohtake, S. Nagai, H. Wada, and H. Fujiwara, "A DFT method for RTL circuits to achieve complete fault efficiency based on fixed control testability," Proc. of *IEEE Asian and South Pacific Design Automation Conf.*, pp. 331-334, 2001.

[7] D. Xiang, Y. Xu, and H. Fujiwara, "Nonscan design for testability for synchronous sequential circuits based on conflict resolution," Accepted to appear in *IEEE Trans. on computers*, Aug., 2003.

[8] D. Xiang and H. Fujiwara, "Handling the pin overhead problem of DFTs for high quality and at-speed test," *IEEE Trans. on Computer-Aided Design of ICAS*, vol. 21, no. 9, pp. 1105-1113, 2002.