A Method of Unsensitizable Path Identification using High Level Design Information

Satoshi Ohtake Naotsugu Ikeda[†] Michiko Inoue Hideo Fujiwara
Graduate School of Information Science, Nara Institute of Science and Technology
8916-5 Takayama, Ikoma, Nara 630-0192, Japan
E-mail: {ohtake, kounoe, fujiwara}@is.naist.jp

***Abstract*—This paper proposes a method of unsensitizable path identification. By dealing with untestable paths in sequential circuits at register-transfer level (RTL) and utilizing design information obtained from high-level synthesis, the method can identify sequentially unsensitizable paths effectively and efficiently. Information about identified unsensitizable paths can not only facilitate process of test generation but also mitigate futileness of testing. In this work, we employ non-robust unsensitizability, which is widely supported in current ATPG systems, for identifying unsensitizable paths. We show the effectiveness of use of high-level design information through our experiments using RTL circuits synthesized from behavioral benchmark circuits.**

I. Introduction

Unsensitizable path information available prior to test generation is very valuable for reducing cost and improving accuracy and quality of delay testing of digital circuits. Since there exists no test pattern that activate delay faults on an unsensitizable path along the path, prior identification of them can greatly reduce test generation time. Furthermore, although some originally unsensitizable paths can become sensitizable due to application of design for testability (DFT) and result in over-testing, this can be alleviated by using the information. Several untestable path delay fault (PDF) identification methods at gate level for combinational circuits [1], [2], [3] and for sequential circuits [4], [5] have been proposed. Although handling untestability of PDFs at gate level is quite natural, it is difficult to get satisfactory results by these identification methods for large circuits containing a tremendous number of paths at gate level. To avoid this difficulty, some methods utilizing register transfer level (RTL) design information, instead of gate level, have been proposed [6], [7]. These methods can identify untestable paths based on non-robust testability and functionally unsensitizability of PDFs, respectively, in a reasonable amount of time by handling a number of gate level paths as an RTL path. However, for a circuit in RTL description, all the possible combinations of state transitions of the controller are needed to be examined and paths that go through status signal lines from the data path to the controller in the circuit are missed in the identification processes in those methods.

For a circuit designed at behavioral level, an RTL design of the circuit is obtained by high-level synthesis (HLS). It is known that, by analyzing behavioral level description of a circuit, unusable/unnecessary data transfers and/or control flows can be extracted (e.g., the method in [8], [9], [10]). In this work, we focus on such knowledge obtained from HLS and propose a method to identify unsensitizable paths by utilizing the knowledge in addition to RTL design information for further improvement from the previous method. In this work, (1) all the RTL paths in a circuit including paths that go through status signal lines, which are missed in [6], are dealt with, (2) transitions on control signal lines from the controller to the data path are considered when paths that go through the control signal lines are targeted, and (3) time required for identification is improved from the previous methods. We show the effectiveness of using information obtained from HLS through our experiments on RTL circuits synthesized from behavioral benchmark circuits.

The rest of this paper is organized as follows. Section II gives preliminaries of the work. In Section III, our RTL unsensitizable path identification method is presented and its correctness is shown. Experiments are provided in Section IV and the paper concludes with Section V.

II. Preliminary

A. RTL circuit model

Our target RTL design is structural RTL circuit model. A structural RTL description of a circuit consists of a controller and a data path represented by RTL modules and RTL signal lines, where an RTL module is a combinational operation module or a multiplexer (MUX) or a register and an RTL signal line has bit width and connects between two RTL modules. They are connected with each other by control signal lines and status signal lines. The controllers control control inputs of RTL modules (e.g., a combinational functions complex, a register and a MUX) in the data path. On the other hand, status outputs of RTL modules in the data path are connected to the controllers. An example of RTL description of a circuit is shown in Figure 1.

Note that since circuits are assumed to be designed at behavioral level and synthesized by a high-level synthesis tool (e.g. Explorations Tool (Y Explorations, Inc.) [11]), the synthesized RTL circuits are described structurally and their structural information can easily be made available.

[†]Currently, he is with Panasonic Corporation.

978-1-4244-6340-4/10/$26.00 ©2010 IEEE

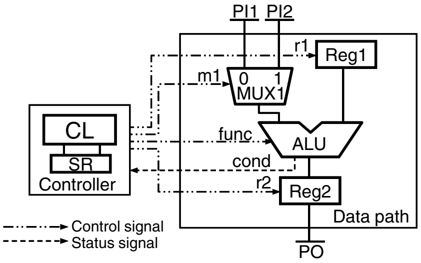


Fig. 1. An example of RTL description of a circuit.

B. Gate level unsensitizable paths

A path at gate-level is a sequence of circuit elements $\{g_0, g_1, \ldots, g_n\}$, where g_0, g_n and $g_i(0 < i < n)$ are a primary input or an FF, a primary output or an FF, and a gate, respectively, and the output of $g_i(0 \leq i < n)$ is an input of g_{i+1}. Given a sensitization criterion, a gate-level path p in a combinational circuit is said to be unsensitizable under the sensitization criterion if any transition at the start of p is not propagated to the end of p with the specified condition for off-inputs of gates on p. In this paper, the *non-robust sensitization criterion* [12], which is most widely used in ATPG systems, is employed and a path is referred to as an unsensitizable path if the rising and falling PDFs on the path are non-robust untestable. Under the non-robust sensitization criterion, p is sensitized by a pair of patterns (v_1, v_2) such that if (v_1, v_2) launches a transition at the start of p and v_2 sets non-controlling values to off-inputs of gates on p. Since, practically, the number of gate level paths in a circuit is extremely large, identification of unsensitizable paths in the circuit at gate level is intractable.

C. RTL unsensitizable paths

When we consider the circuit at RTL, a bundle of a large number of gate-level paths are dealt with as an *RTL path*. In this paper, we use the concept of RTL paths proposed in [13]. An RTL path is a sequence of RTL modules $\{m_0, m_1, \ldots, m_n\}$, where m_0, m_n and $m_i(0 < i < n)$ are a primary input or a register, a primary output or a register, and a combinational module, respectively, and the output of $m_i(0 \leq i < n)$ is an input of m_{i+1}. The number of RTL paths in an RTL circuit is much smaller than that of gate-level paths in its gate-level circuit. Note that, to uniquely distinguish a path, fanout branches are dealt with combinational modules if necessary.

Unsensitizability of RTL paths is defined as follows.

Definition 1 (RTL unsensitizable path): Given a sensitization criterion, an RTL path p in an RTL circuit is *RTL unsensitizable under the sensitization criterion* if any gate-level path corresponding to p in its gate-level circuit is gate-level unsensitizable under the criterion for any logic synthesis.

D. RTL path with transitions

In general, since an RTL path has bit width, transitions on the path cannot be used to examine unsensitizability of the path. However, if the RTL path has single-bit signal lines, some specific transition on each single-bit signal line can be dealt with. The previous work in [6] only handle the one-bit signal lines at the starts of paths. In this paper, to improve identification quality further, transitions on intermediate one-bit signal lines on RTL paths are also handled.

Definition 2 (RTL path with transitions): Suppose that an RTL path $P_r = \{R_0, M_1, \ldots, M_{n-1}, R_n\}$ includes control signals $(M_{j_{k-1}}, M_{j_k})(1 \leq k \leq q)$, where q is the total number of control signal lines on P_r. Let $(M_{j_{k-1}}, M_{j_k})[b_k]$ be the b_k-th bit of a control signal $(M_{j_{k-1}}, M_{j_k})$. A path, which goes through control signals $(M_{j_{k-1}}, M_{j_k})[b_k]$ and the control signal has transitions d_k, on P_r is said to be an *RTL path with transitions* and denoted as $P_r : \{(M_{j_{k-1}}, M_{j_k})[b_k]d_k | 1 \leq k \leq q\}$, were d_k is a rising ($\uparrow$) or falling ($\downarrow$) transition.

E. High level synthesis information

In HLS systems, a circuit is generally represented by a *control data flow graph (CDFG)* [14] at behavioral level. During HLS, each operation is assigned to a control step (*scheduling*) and operations and variables are mapped to operational modules and registers, respectively (*binding*). In this paper, we assume that, for a circuit, a *scheduled CDFG (S-CDFG)* and *binding information* are given in addition to RTL structural information. These are referred to as *high-level synthesis (HLS) information*. An example of HLS information of a circuit is shown in Figure 2 (a) and (b). In Figure 2 (a), the triangular nodes and the edges between them denote conditional branches or merges and control flows, respectively. The circular nodes and the edges between them in rectangles denote operations and variables, respectively. The horizontal dotted lines denote boundaries between control steps. In Figure 2 (b), for each line, the left of a colon denotes an operational module or a register and the right denotes operations or variables bound to.

In this work, we assume HLS and logic synthesis to be the following.

1) Every register allocated during HLS has a load enable control and is controlled so that it does not load if its input has no valid data.
2) For any combinational RTL module, no logic optimization beyond its boundary is done during logic synthesis.

The first assumption guarantees that, for every control step in which no variable bound to the register appear, load is disabled to prevent unexpected behavior which is not specified in the CDFG. The second assumption stands for guaranteeing existence of correspondence between RTL paths to gate-level paths. The same restriction on logic synthesis is employed in [6]. The assumption can also be removed by establishing such correspondence by the method of mapping from RTL paths to gate-level paths proposed in [15].

III. Proposed Method

A. Overview

For a circuit at RTL, existence of an RTL path does not always mean that the path can be used to propagate meaningful

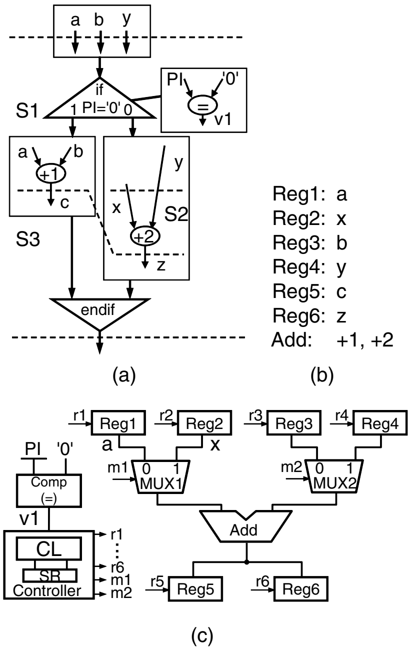


Fig. 2. An S-CDFG (a), binding information (b), and a synthesized RTL circuit (c).

data. For example, in Figure 2 (a) and (b), RTL module "Add" is shared by two operations "+1" and "+2". Figure 2 (c) shows an RTL circuit synthesized from the S-CDFG of (a) with the binding information of (b). There are eight RTL paths which pass through "Add" but only four paths of them are actually used path. Paths that start at "Reg2" or "Reg4" and end at "Reg5" never propagate meaningful data because any operation corresponding to such paths is not specified at behavior level.

In order to deal with RTL structural information and HLS information for a circuit efficiently, we construct *module input output dependence graphs (IODGs)* defined in the next subsection. An IODG represents relation between inputs (resp. an output) of an RTL module and variables on the inputs (resp. output) of operations bound to the RTL module and dependence of input and output variables. For an RTL path, our unsensitizable path identification method employs IODGs for modules on the path to examine whether some meaningful data flow from the start register to the end register along the path can be made or not.

B. Module input output dependence graph

The IODG of an RTL module M is a directed graph $G = (V, E)$, where V is the set of nodes representing input and output variables of operations corresponding to M, and $E \subset V \times V$ is the set of edges representing input/output (IO) dependence between a couple of input and output variables. IO dependence between two variables (u, v) is defined as a relation as follows.

1) u and v are an input variable and an output variable, respectively, of an operation.
2) The edge corresponding to the input variable u does not cross boundary of control steps more than once in the interested S-CDFG.

Note that 2) is a necessary condition for sensitizing the path to test delay fault along the path within consecutive two cycles.

RTL paths can start and/or end at a state register and also can contain control signals and/or status signals. For an RTL module M_C with a control input, in addition to the IODG of M_C, a partial IODG is created for each control value of the control input. The partial IODG is used for identifying RTL unsensitizable paths, which go through the control signals connecting with transitions on the control signals. Examples of partial IODGs are shown later.

Since operations corresponding to MUXs in a given RTL schematic are not described explicitly in the corresponding S-CDFG, we cannot extract IO dependences for variables corresponding to MUXs. To deal with the MUXs, *thru* operations corresponding to MUXs are added to the S-CDFG. Figure 3 shows the modified HLS information and RTL schematic of Figure 2. In this example, thru operations $t1, t2, t3, t4$ are added to the S-CDFG.

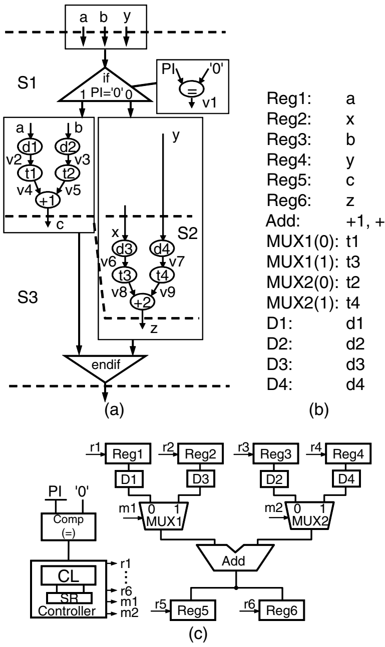
Furthermore, it is conceivable that there exist IO dependences for variables corresponding to the control logic (CL) in the controller. However they cannot be derived because operations corresponding to the CL do not appear in the S-CDFG. The CL can have the following four types of variables.

1) The variable for the state register.
2) Variables for data registers.
3) Control dependent variables corresponding to data inputs of combinational modules which have control inputs.
4) Variables corresponding to outputs of conditional branches.

To deal with the IO dependence of these IO variables of the CL, *dummy* operations to explicitly represent the variables are added to the S-CDFG and dummy modules corresponding to the dummy operations are added to the RTL schematic. For example, in Figure 3 (a), variables $v2$, $v3$, $v6$ and $v7$ represent variables on the inputs of the MUXs, "MUX1" and "MUX2" in the RTL schematic of Figure 3 (c) and are created by adding dummy operations $d1, d2, d3, d4$. According to addition of the dummy operations, dummy modules $D1, D2, D3, D4$ are added to the RTL schematic.

From the modified HLS information and RTL schematic, we can create IODGs for the RTL modules including the MUXs and CL. Examples of IODGs for modules in the RTL circuit of Figure 2(c) is shown in Figure 4. In the IODG for the CL (Figure 4(e)), a special variable S is used to represent the state register.

An example of partial IODGs for a MUX is shown in Figure 4(c). The control value is referred when handling RTL paths going through the control input of the MUX with a transition. If rising (resp. falling) transition is considered, the partial IODG with control value 1 (resp. 0) is referred.



Reg1:	a
Reg2:	x
Reg3:	b
Reg4:	y
Reg5:	c
Reg6:	z
Add:	+1, +2
MUX1(0):	t1
MUX1(1):	t3
MUX2(0):	t2
MUX2(1):	t4
D1:	d1
D2:	d2
D3:	d3
D4:	d4

Fig. 3. An augmented S-CDFG (a), augmented binding information (b), and its corresponding RTL circuit (c).

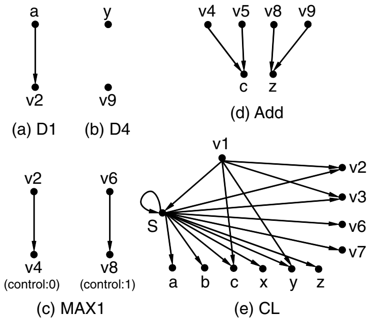


Fig. 4. IODGs for modules in the circuit of Figure 3 (c).

C. Identification of RTL unsensitizable paths

Given an RTL circuit and HLS information. For an RTL path $P = \{M_0, M_1, \ldots, M_{n-1}, M_n\}$, if there exists a data flow corresponding to P and the value of M_0 changes when a state transition occurs, the effect is propagated along P and is captured at M_n when the next state transition occurs. Thus there exist variables $v_0, v_1, \ldots, v_{n-1}$ and operations $o_1, o_2, \ldots, o_{n-1}$ which satisfy the following property.

1) v_0 and v_{n-1} are assigned to M_0 and M_n, respectively.
2) The edge corresponding to v_0 does not cross boundaries of control steps more than once on the S-CDFG.
3) For each $i(1 \leq i \leq n-1)$, o_i is assigned to M_i and v_{i-1} and v_i are the input variable and output variable of o_i, respectively.

From the definition of IODG, for each $M_i(1 \leq i \leq n-1)$, there exists an edge (v_{i-1}, v_i) of the IODG of M_i. This means that there exists a path $v_0, v_1, \ldots, v_{n-1}$ referred to as an *IODG path* on the union of IODGs. In the proposed method of unsensitizable path identification, an IODG path corresponding to a given RTL path is sought and the RTL path is judged to be unsensitizable if there exist no IODG path.

For example, consider an RTL path $\{Reg1, D1, MUX1, Add, Reg5\}$ in Figure 3. For the path, there exists the IODG path $\{a, v2, v4, c\}$ on the union of IODGs in Figure 4.

D. Identification of RTL unsensitizable paths with transitions

If an RTL path tried to be identified has control signals, identification is performed for its RTL paths with transitions. Let P_r and P_r^t be an RTL path $\{R_0, M_1, \ldots, M_{n-1}, R_n\}$ and $P_r : \{(M_{j_k-1}, M_{j_k})[b_k]d_k | 1 \leq k \leq q\}$, respectively. Suppose that, for a state S_0, the value of the variable corresponding to R_0 changes, then the induced transitions propagated along P_r and captured into R_n at a next state S_1. For sensitizing P_r^t, the value of control signal $(M_{j_k-1}, M_{j_k})[b_k]$ is necessary to be 0 (resp. 1) if $d_k = \downarrow$ (resp. $\uparrow$). Without loss of generality, M_{j_k} executes the operation corresponding to $(M_{j_k-1}, M_{j_k})[b_k] = 0$ if $d_k = \downarrow$. On the IODG path $v_0, v_1, \ldots, v_{j_k-1}, v_{j_k}, \ldots, v_{n-1}$ corresponding to P_r, the proposed method seeks partial IODGs corresponding to $(M_{j_k-1}, M_{j_k})[b_k] = 0$, of M_{j_k} whether the directed edge (v_{J_k-1}, v_{j_k}) exists or not. P_r^t is judged to be unsensitizable if there exist no IODG path corresponding to P_r^t.

For example, consider an RTL path with a transition $\{SR, CL, MUX1, Add, Reg6\} : \{(CL, MUX1)[1] \uparrow\}$ in Figure 3. For the path, there exists the IODG path $\{S, v6, v8, z\}$ on the union of IODGs with the partial IODG for $(CL, MUX1)[1] = 1$ of $MUX1$ in Figure 4.

E. Algorithm for identification

For an RTL circuit, the proposed method traverses RTL paths from each primary input or register until reaching a primary output or a register with depth-first fashion and checks every RTL path if it is unsensitizable or not. Suppose that, during the depth-first traversal, the algorithm detect that there exits no IODG path corresponding to a sub RTL path from a primary input or a register to currently visiting module. In such a case, all the paths that contains the sub RTL path as a prefix are identified as unsensitizable. In the proposed method, identification of both RTL unsensitizable paths and ones with transitions are simultaneously processed. Since the previous method of [6] tries all the possible state transitions exhaustively for every RTL path, the proposed algorithm is obviously faster than the previous method.

F. Correctness of the method

We prove the correctness of the proposed method for identifying RTL unsensitizable paths. In the previous work [6], sufficient condition of RTL unsensitizable paths has shown using load enable signals of registers and selection signal of MUXs. Although the following theorem is originally provided by [6], it is slightly modified for fitting the discussion in this work.

Theorem 1: An RTL path P_r is RTL unsensitizable if at least one of the following three conditions is satisfied for every state transition from any time t to $t+1$.

1) No transition is launched at the start register of P_r at t.
2) Control signals of MUXs on P_r do not select P_r at t.
3) The end register of P_r loads no transition propagated along P_r at $t+1$.

In [6], MUXs are considered to be the modules which can prevent propagation of transitions. In this work, we also consider other modules which have control input than MUXs. To support such modules, Theorem 1 is augmented into the following Lemma 1.

Lemma 1: An RTL path P_r is RTL unsensitizable if at least one of the following three conditions is satisfied for every state transition from any time t to $t+1$.

1) No transition is launched at the start register of P_r at t.
2) The output value of some combinational module on P_r does not depend on its input on P_r at t.
3) The end register of P_r loads no transition propagated along P_r at $t+1$.

Since the proof can be derived as the similar way as Theorem 1 shown in [6], it is omitted here due to the limitation of the space. A sufficient condition of an RTL unsensitizable path, which is employed in the identification method, is shown in Theorem 2.

Theorem 2: An RTL path $P_r = \{R_0, M_1, \ldots, M_{n-1}, R_n\}$ is RTL unsensitizable if there exist no sequence of variables, $v_0, v_1, \ldots, v_{n-1}$, satisfying the following conditions.

1) v_0 and v_n are the variables corresponding to R_0 and R_n, respectively.
2) For every $i(1 \leq i \leq n-1)$, (v_{i-1}, v_i) is an edge on the IODG of M_i.

Proof: From the assumption that each register has a load enable control and the control is enabled only if valid data is available on its input, there exists an IODG path starting from a primary input or a register to R_n if R_n is enabled to load mode. Suppose that R_0 has a transition at a time t. There exists an IODG path if the transition propagated along P_r at t and R_n loads the transition at $t+1$. Thus, there exist variables that satisfy the conditions 1) and 2). Therefore, one of the following cases satisfies at any time t if there do not.

i) The start, R_0, of P_r has no transition at t.
ii) The output value of some M_i on P_r does not depend on its input on P_r at t.
iii) The end, R_n, of P_r loads no transition at $t+1$.

Thus, from Lemma 1, P_r is RTL unsensitizable. □

A sufficient condition of an RTL unsensitizable path with transitions, which is employed in the identification method, is shown in Theorem 3.

Theorem 3: An RTL path with transitions $P_r^t = \{R_0, M_1, \ldots, M_{n-1}, R_n\} : \{(M_{j_k-1}, M_{j_k})[b_k]d_k | 1 \leq k \leq q\}$ is an RTL unsensitizable path with transitions if there exist no sequence of variables, $v_0, v_1, \ldots, v_{n-1}$, satisfying the following conditions.

1) v_0 and v_n are the variables corresponding to R_0 and R_n, respectively.
2) For every $i((1 \leq i \leq n-1), i \neq j_k(1 \leq k \leq q))$, (v_{i-1}, v_i) is an edge on the IODG of M_i.
3) For every $j_k(1 \leq k \leq q)$, (v_{j_k-1}, v_{j_k}) is an edge on the partial IODG of M_{j_k} that have the control signal value $(M_{j_k-1}, M_{j_k})[b_k] = c_k$, where $c_k = 0$ if $d_k = \downarrow$. Otherwise $c_k = 1$.

Proof: In case that there exist no sequence of variables that satisfy the condition 1) or 2) of Theorem 3, P_r^t is an RTL unsensitizable path with transitions because P_r is RTL false from Theorem 2. Consider the case that there exist no sequence of variables that satisfy the condition 3) for some k even if there exists a sequence of variables that satisfies the condition 1) or 2) at a time t. In between the start FF and end FF of any gate level path P_g in the set of gate level paths corresponding to P_r^t, no transition at the start FF at t cannot be propagated along P_g at t to the end FF because $(M_{j_k-1}, M_{j_k})[b_k] = c_k$ does not hold true, i.e., $(M_{j_k-1}, M_{j_k})[b_k]$ cannot have transition d_k at t. Therefore, the gate level paths in P_g cannot be sensitized using the transition d_k on $(M_{j_k-1}, M_{j_k})[b_k]$ at t. Thus P_r^t is an RTL unsensitizable path with transitions. □

IV. Experimental Results

In the experiments, we evaluate the effectiveness of the method. We used RTL circuits synthesized from behavioral level benchmark circuits, 3rd Lattice Wave Filter (LWF), Tseng[16], Paulin[17] 4th Jaumann Wave Filter (JWF) and GCD[18], by the HLS method of [19]. The circuit characteristics are shown in TABLE I. Columns “Bit width”, “#PIs”, “#POs”, “#Regs.”, “#Paths” and “#Paths w/ Trans.” denotes bit width of the data path and the numbers of primary inputs, primary outputs, registers, RTL paths and RTL paths with transitions, respectively. The identification results for these circuits are shown in TABLE II. We run our algorithm on SunFire V490 (UltraSPARC IV+ 1.8GHz with 32GB memory, Sun Microsystems Inc.). In the table and tables discussed below, columns “#Total” show the total number of paths in the circuits, columns “#US” denote the number of unsensitizable paths and columns “#USwt” denote the number of unsensitizable paths with transitions on control signals. Columns under “Through no control signal” and “Through control signal” correspond to paths that exclude and include control signal lines, respectively. The time required for the identification is less than 1 second for every circuit.

Note that the handling paths of RTL circuits is different from the previous work [6], we cannot compare the identification results directly. However, since our method can deal with paths with transitions on control signal lines and paths going through status signal lines, it is conceivable that our method can identify different paths than the previous work. Practically, both of the identification methods can be applied because the processing times is very short.

Since GCD is the only circuit which has status signals, we examined in detail. TABLE III and IV shows the break down for GCD at RTL and the classification of corresponding

TABLE I
THE CIRCUIT CHARACTERISTICS OF BENCHMARK CIRCUITS.

Circuit	Bit width	#PIs	#POs	#Regs.	#Paths	#Paths w/ Trans.
GCD	8	2	2	4	62	48
LWF	8	3	2	6	49	36
Tseng	8	4	3	7	48	28
Paulin	8	3	2	8	77	64
JWF	8	6	5	15	427	488

TABLE II
THE NUMBER OF RTL UNSENSITIZABLE PATHS IDENTIFIED BY OUR METHOD.

Circuit	Through no control signal		Through control signals		
	#US	#Total	#US	#USwt	#Total
JWF	124	155	78	144	516
LWF	8	21	0	12	46
Tseng	6	22	0	0	40
Paulin	13	31	2	12	78
GCD	0	10	4	9	70

gate-level paths, respectively. TABLE III gives classification of the RTL paths in GCD w.r.t. going through status signals. Although the method did not find any unsensitizable paths from the paths excluding control signals and that include control signals without status as shown in the first seven columns, 4 unsensitizable RTL path and 9 unsensitizable RTL paths with transitions were found from the paths that include both control and status signals. For this circuit, we synthesized and evaluated the number of the gate level paths corresponding to RTL paths. We used DesignCompiler and PrimeTime (Synopsys Inc.) as a logic synthesis tool and a tool for extracting gate level paths, respectively. In TABLE IV, the number of gate level paths are shown. The 9 unsensitizable RTL paths with transition corresponded to 800 unsensitizable gate-level paths with transitions (1/6 of the total number of paths with transitions), which cannot be identified by the previous method.

V. CONCLUSIONS

This paper proposed a method for identifying unsensitizable paths based on non-robust sensitization criterion in sequential circuits synthesized through high-level synthesis (HLS) and logic-synthesis. The method uses RTL structural and HLS information of the circuits. Since a bunch of gate-level paths, called an RTL path, is dealt with at RTL, a large number of unsensitizable gate-level paths can be efficiently identified. By using HLS information, the method can efficiently handle paths going through status signals which cannot be dealt with in the previous work. Our experimental results proved that the proposed method can improve the quantity of identified unsensitizable paths compared to the previous method.

ACKNOWLEDGMENT

This work was supported in part by Semiconductor Technology Academic Research Center (STARC) under the Research Project, and in part by Japan Society for Scientific Research (B) (No.20300018).

TABLE III
CLASSIFICATION OF RTL PATHS IN GCD W.R.T. GOING THROUGH STATUS SIGNALS.

Through no control signal				Through control signals					
No status		Through status		No status			Through status		
#US	#Total	#US	#Total	#US	#USwt	#Total	#US	#USwt	#Total
0	12	0	8	0	0	22	4	9	44

TABLE IV
CLASSIFICATION OF GATE-LEVEL PATHS IN GCD.

#Structural paths in the circuit			
Through no control	Through control		Total
	Register load	MUX select	
280	744	1400	2424

# US	
Without transition	With transition
0	800

REFERENCES

[1] K. T. Cheng and H. C. Chen, "Classification and identification of nonrobust untestable path delay faults," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 15, no. 8, pp. 854–853, Aug 1996.

[2] S. Kajihara, K. Kinoshita, I. Pomeranz, and S. Reddy, "A method for identifying robust dependent and functionally unsensitizable paths," in *Int'l Conf. on VLSI Design*, 1997, pp. 82–87.

[3] S. M. Reddy, S. Kajihara, and I. Pomeranz, "An efficient method to identify untestable path delay faults," in *Asian Test Symp.*, 2001, pp. 233–238.

[4] A. Krstić and K.-T. T. Cheng, "Resynthesis of combinational circuits for path count reduction and for path delay fault testability," in *European Design and Test Conf.*, 1996, pp. 486–490.

[5] R. Tekumalla and P. Menon, "Identifying redundant path delay faults in sequential circuits," in *Int'l Conf. VLSI Design*, 1996, pp. 406–411.

[6] Y. Yoshikawa, S. Ohtake, and H. Fujiwara, "False path identification using rtl information and its application to over-testing reduction for delay faults," in *Asian Test Symp.*, 2007, pp. 65–68.

[7] Y. Yoshikawa, S. Ohtake, T. Inoue, and H. Fujiwara, "Fast false path identification based on functional unsensitizability using RTL information," in *Asia and South Pacific Design Automation Conf.*, Jan. 2009, pp. 660–665.

[8] R. Bergamaschi, "The effects of false paths in high-level synthesis," in *Int'l Conf. on Computer-Aided Design*, 1991, pp. 80–83.

[9] A. Su, T.-Y. Liu, Y.-C. Hsu, and M. T.-C. Lee, "Eliminating false loops caused by sharing in control path," in *Int'l Symp. on System Synthesis*, 1996, pp. 39–44.

[10] L. Stok, "False loops through resource sharing," in *Int'l Conf. on Computer-Aided Design*, 1992, pp. 345–348.

[11] *eXplorations Environment (XE) Users Manual*, Y Explorations , Inc., 2003.

[12] A. Krstić and K.-T. T. Cheng, *Delay fault testing for VLSI circuits*. Kluwer Academic Publishers, 1998.

[13] M. A. Amin, S. Ohtake, and H. Fujiwara, "Design for hierarchical two-pattern testability of data paths," *IEICE Trans. on Information and Systems*, vol. E85-D, no. 6, pp. 975–984, Jun 2002.

[14] D. D. Gajski, *High-Level Synthesis: Introduction to Chip and System Design*. Kluwer Academic Publishers, 1992.

[15] H. Iwata, S. Ohtake, and H. Fujiwara, "Enabling false path identification from RTL for reducing design and test futileness," in *Int'l Symp. on Electronic Design, Test & Applications*, 2010, pp. 20–25.

[16] C. J. Tseng and D. P. Siewiorek, "Automated synthesis of data paths in digital systems," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 5, no. 3, pp. 379–395, 1986.

[17] P. G. Paulin and J. P. Knight, "Foce directed scheduling for the behavioral synthesis of ASIC's," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 8, no. 6, pp. 661–679, 1988.

[18] I. Ghosh, A. Raghunath, and N. K. Jha, "A design-for-testability technique for register-transfer level circuits using control/data flow extraction," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 17, no. 8, pp. 706–723, Aug 1998.

[19] T. Takasaki, T. Inoue, and H. Fujiwara, "A high-level synthesis approach to partial scan design," in *Asian Test Symp.*, 1999, pp. 309–314.