2019 32nd International Conference on VLSI Design and 2019 18th International Conference on Embedded Systems (VLSID)

A Binary Decision Diagram Approach to On-line Testing of Asynchronous Circuits

Pradeep Kumar Biswal
Department of Computer Science & Engineering
IIIT Bhagalpur, Bihar, India
biswalpradeep28@gmail.com

Santosh Biswas
Department of Electrical Engineering and Computer Science
IIT Bhilai, India
santoshbiswas402@yahoo.com

***Abstract*—The rapid increase in complexity of digital VLSI circuits with the advent of Deep Sub-Micron (DSM) technology causes development of faults during their normal operation. Some non-modeled faults may not always be detectable by off-line test or Built-In-Self-Test (BIST), or a number of critical faults may require detection or alerts at the functional mode during run-time. On-line Testing (OLT) provides a solution to both problems, and can be implemented using appropriate Design-for-Testability (DFT) techniques. There are few OLT techniques for asynchronous circuits have been proposed as compared to synchronous circuits. The existing OLT techniques for asynchronous circuits have the issues of protocol dependency, high area overhead and scalability. In this work, we have proposed a partial replication based OLT technique for asynchronous circuits using Binary Decision Diagram (BDD). The proposed scheme works for all circuits irrespective of their design protocols and achieves comparatively low area overhead. Further, use of BDD enables the scheme to handle fairly large circuits.**

***Index Terms*—On-line Testing (OLT), Asynchronous circuit, Binary Decision Diagram (BDD), Area Overhead (AO), Fault Coverage (FC), Signal Transition Graph (STG)**

I. Introduction

The complexity of digital VLSI circuits in recent years has increased in a very impressive manner. The sophistication of VLSI technology has reached a point where an effort is made to put a large number of devices on a single chip by decreasing the dimensions of the transistors and interconnection wires, from micrometers to nanometers. As the fabrication technology moves to lower sub-micron processes and engineers keep increasing the design complexity, testing encounters greater challenges [9]. Testing of digital VLSI circuits can be classified into two important classes; Off-line testing and On-line Testing (OLT). The off-line testing strategies (Automatic Test Equipment (ATE) based testing and Built-In-Self-Test (BIST) [4], [11]) cannot detect faults that develop on-the-fly during operation of the circuit. It has been observed that the probability of occurrence of such faults in the present day VLSI circuits designed using deep sub-micron technology is high [14], [21]. Therefore, OLT is becoming an indispensable part of testing. OLT can be defined as the procedure to enable integrated circuits to verify the correctness of their functionality during normal operation by checking whether the response of the circuit conforms to its desired dynamic behavior. In OLT, it requires an on-chip Design For Testability (DFT) circuity to test the CUT for all the input patterns that would appear during normal operation [7], [18], [22].

Since last two decades, a number of OLT techniques have been proposed for digital circuits, which can be broadly classified as−signature monitoring in Finite State Machines (FSMs) [10], [25], self-checking design [13], [18], partial replication [6], [8], [12] and on-line BIST [2], [3], [27]. The OLT techniques namely, signature monitoring and self checking design, require some special properties in the circuit structure, which lead to a change in the original structure of the circuit. So, these two OLT techniques are intrusive in nature. Since change in the original structure of the circuit is not desirable, so these techniques have limited applicability. Also, the OLT technique based on on-line BIST utilizes the idle times of the various parts of the circuit during operation to perform testing. Therefore, the efficiency of on-line BIST mainly depends on the amount of idle times available in the circuit modules. The present day circuits target to achieve pipelining and parallelism, which reduce the idle times of their modules (i.e, high utilization of their modules). So, on-line BIST cannot be considered as an efficient technique for OLT. In the case of partial replication technique, a minimized version of the Circuit Under Test (CUT) is designed and OLT is performed by cross-checking for similarity of output responses of the CUT and the replicated circuit. The partial replication technique is widely used in OLT because of the advantages such as simplicity in design, non-intrusiveness (minimal changes in original structure of the CUT), flexibility in terms of trade-offs between area and power overheads of the on-line tester versus fault coverage and detection latency, etc [6], [12], [20].

Recently, VLSI community has grown interest in asynchronous circuits because they have no clock skew problem, have potentially lower power consumption, can be designed for average case performances rather than the worst case performances, and have higher degree of modularity. Testing of asynchronous circuits as compared to synchronous circuits is considered difficult due to the absence of the global clock [15]. Also, OLT of such circuits is one of the challenging tasks. It has been found that most of the OLT schemes are designed for synchronous circuits compared to asynchronous circuits. There are few works that have been proposed on OLT of asynchronous circuits [5], [19], [23], [24], [26]. Now we

2380-6923/19/$31.00 ©2019 IEEE
DOI 10.1109/VLSID.2019.00035

elaborate these works as follows.

OLT of asynchronous circuit discussed in [26] by Verdel et al. is based on simple double redundancy method, where two copies of the same circuit run in parallel and the on-line tester checks if they generate the same output responses. The main drawback of this scheme is that it has more than 100% area and power overheads. Further, both being the same circuit, they are susceptible to similar nature of failures. The OLT schemes for asynchronous circuits presented in papers [23], [24] work by checking if the output of the circuit satisfies a predefined protocol. The protocol is maintained in such a way that there is no late occurrence or premature transitions. The on-line tester circuit (called checker) is designed using Mutual Exclusion (Mutex) elements, David Cells (DC), C-elements [17] and logic gates. Basically, the checker operates in two modes; normal test mode and self-test mode. In normal test mode, the checker detects if there is any contravene in the protocol being executed by the Circuit Under Test (CUT). In self-test mode, the checker detects if there are any faults occurring within it. The main drawback of these schemes is that the area overhead is much higher than that of the CUT even the redundancy based methods. This is because the on-line tester is designed using Mutex elements and area overhead of the Mutex elements is quite high. Further these schemes only check the protocol, thus there is no guarantee to fault coverage and detection latency. The OLT schemes reported in papers [5], [19] are non-intrusive in nature and have low area overhead, however they are not scalable because of state explosion problem in Discrete Event System model. From the above discussion we aim at developing a Binary Decision Diagram (BDD) based OLT scheme for asynchronous circuits which is protocol independent, scalable and incurs low area overhead.

The rest of the paper is organized as follows. Circuit modeling under normal and faulty conditions and generation of fault detecting transitions are illustrated in Section II. We discuss the procedure of generation of fault detecting transitions using BDD in Section III. Design of on-line tester circuit from fault detecting transitions is explained in Section IV. We present experimental results regarding fault coverage and area overhead in Section V and conclude in Section VI.

II. Circuit modeling under normal and faulty conditions and generation of fault detecting transitions

In this section, we start with modeling of asynchronous circuit using Signal Transition Graphs (STGs) under normal and faulty conditions, then convert these STGs into State Graphs (SGs) and generate Fault Detecting transitions (FD-transitions) from the state graphs. In order to discuss our proposed scheme, an example of asynchronous circuit shown in Figure 1 (taken from [16]) is considered as the CUT. The circuit consists of dynamic C-elements (act as storage elements) and logic gates. When both the inputs of C-element are low (high) then it's output is low (high), otherwise it keeps it's previous logic value. Thus, the Boolean expression of C-element is $C = AB + AC' + BC'$, where C and C' are the next and old state values, respectively. Transistor level diagram of C-element is shown in Figure 2.

The STG represents the behavior of CUT (Figure 1) as shown in Figure 3. There are two types of signals; rising (indicated by $+$) and falling (indicated by $-$). The dark circles along the edges are called tokens. Presence of token enables the signal to fire. When there are tokens at all the input edges of a signal then it is enabled to fire. For example, when both the signals Rin and $Rout$ become low (denoted by $Rin-$ and $Rout-$ respectively), then only $Aout-$ can fire. After firing of $Aout-$, a token is placed on each of its outgoing edges, thus enabling $Rin+$ to fire.

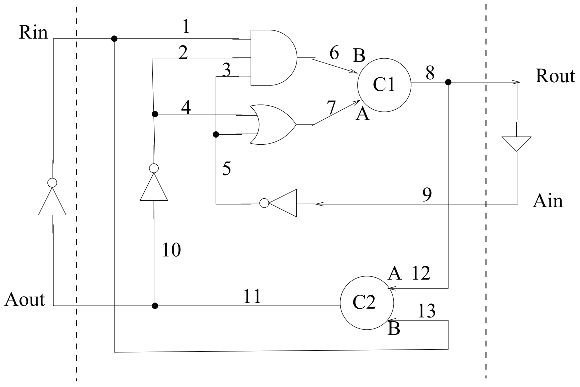

Fig. 1. Example of an asynchronous circuit (i.e. CUT).

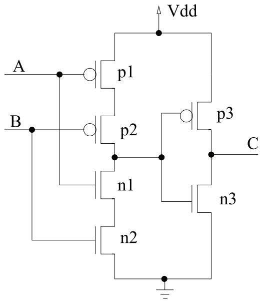

Fig. 2. Transistors of dynamic C-element.

To illustrate fault modeling in case of asynchronous circuits, we have applied stuck-on and stuck-off faults for each transistors of C-elements and stuck-at faults at different lines of the CUT. Consider stuck-on fault at transistor $p1$ of C-element $C2$ in the CUT (shown in Figure 1). Inputs of $C2$ are Rin and $Rout$, and output of $C2$ is $Aout$. Stuck-on fault at transistor $p1$ (Say F_1) waits transistor $p2$ to be enabled to generate the output (referring to the transistor level diagram of C-element shown in Figure 2). When $p2$ turns on then a path to V_{dd} is established via $p1$ and $p2$. Now transistor $n3$ turns on, transistor $p3$ turns off and output becomes low. Thus, the effect of F_1 is that output of $C2$ becomes low (i.e., $Aout-$) when $p2$ turns on (i.e., $Rin-$). That means $Aout-$ can fire just after firing of $Rin-$ and does not wait for firing of $Rout-$ but

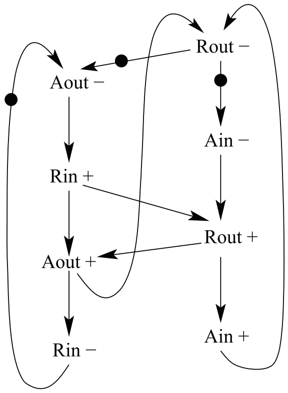

Fig. 3. STG representation of CUT (Figure 1).

in normal case $Aout-$ can fire only after firing of $Rin-$ and $Rout-$. Thus, this fault results premature firing of $Aout-$. The STG representation of the CUT under F_1 is shown in Figure 4. We denote "1" on the edge of $Rout-$ indicating that $Aout-$ can fire as soon as $Rin-$ arrives, without waiting for $Rout-$.

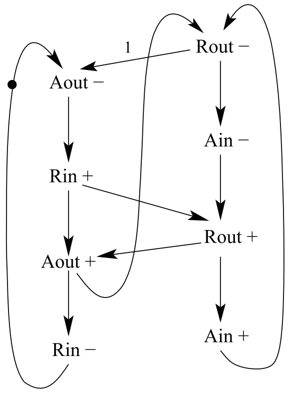

Fig. 4. STG representation of CUT under F_1.

Now we convert the STGs representing CUT under normal and faulty conditions into separate State Graphs (SGs). Since the CUT has 4 input and output signals (Rin, $Rout$, Ain, $Aout$), so in corresponding SG has 4 state variables and each variable assumes values from the set $\{0, 1\}$. Figure 5 shows the SG for the STG representing the CUT under normal condition. The SG shows the state mappings and transitions, e.g., the state ($\langle Rin, Rout, Ain, Aout\rangle$) mappings of $X0$ and $X1$ are $\langle 1, 1, 0, 0\rangle$ and $\langle 1, 1, 0, 1\rangle$, respectively. The transition from $X0$ (present state) to $X1$ (next state) changes $Aout$ from 0 to 1, which is indicated by transition $Aout+$. Similarly, the whole SG is constructed. Figure 6 shows the SG for the STG representing the CUT under F_1. If we compare the normal and faulty SGs, it can be found that there are two transitions ($\langle X4, X14\rangle$ and $\langle X5, X15\rangle$) in the faulty SG which are differ from that of normal SG. These transitions are appeared due to occurrence of fault F_1 in the CUT. This indicates premature firing of $Aout-$. We call these transitions as Fault Detecting transitions (FD-transitions). The FD-transitions for all targeted faults in the CUT are generated and on-line tester circuit is designed using such transitions. In OLT, the on-line tester circuit runs parallel with the CUT and detect fault by means of occurrence of FD-transitions.

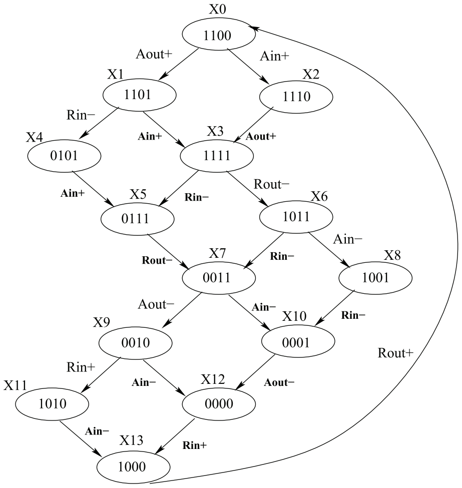

Fig. 5. State graph under normal condition.

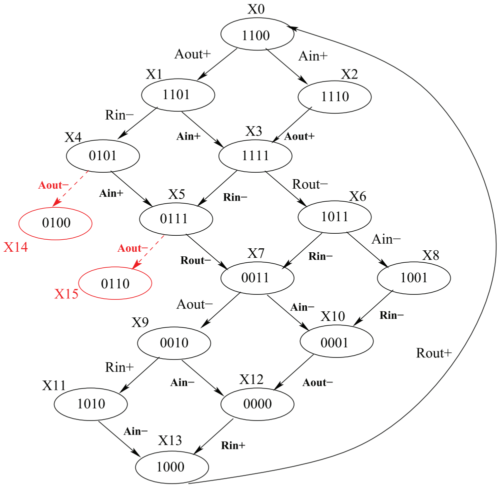

Fig. 6. State graph under fault F_1.

This process of generating FD-transitions from the SG framework becomes complex for large circuits because the number of states in the SG grows exponentially with the variables/ signals of the circuit. Further, comparing each transition under normal and faulty SGs in order to generate FD-transitions is a time taking task. In order to solve these problems we have devised a procedure which is capable of generating FD-transitions directly from the circuit description using Ordered Binary Decision Diagrams (OBDDs), without need of the explicit SG models.

III. Generation of FD-transitions using OBDD

In this section, we demonstrate the procedure of generation of FD-transitions using OBDD by considering the running example of CUT and fault F_1 ($p1$ stuck-on fault of $c2$), following that the detailed procedure is discussed. The STG representation of CUT under F_1 is shown in Figure 4, where $\mathit{Aout}-$ is fired as soon as $\mathit{Rin}-$ fires, without waiting for $\mathit{Rout}-$. In other words, we can say that under this fault $\mathit{Rout}-$ is always true, (i.e., $\mathit{Rout} = 0$) and whenever $\mathit{Rin}-$ occurs in the circuit then $\mathit{Aout}-$ occurs. Since Aout is the output of $C2$, thus the Boolean expression for Aout under normal condition is $\mathit{RinRout} + \mathit{RoutAout}' + \mathit{RinAout}'$ (say Aout_N). Similarly, the Boolean expression for Aout under F_1 is $\mathit{RinAout}'$ (say Aout_{F_1}), which is obtained by substituting $\mathit{Rout} = 0$ in Aout_N. The OBDD representations of Aout_N and Aout_{F_1} are shown in Figure 7 and Figure 8, respectively. Figure 9 represents the XORed OBDD corresponding to $\mathit{Aout}_N \oplus \mathit{Aout}_{F_1}$ which is obtained by XORing the normal and faulty OBDDs shown in Figure 7 and Figure 8, respectively. The set of next states under F_1 which are different compared to the normal condition is obtained by applying "satisfy-all-1" operation on the XORed OBDD. In case of F_1, the such set of next states is $\{\langle \mathit{Rin} = 0, \mathit{Rout} = 1, \mathit{Ain} = 0, \mathit{Aout} = 0\rangle, \langle \mathit{Rin} = 0, \mathit{Rout} = 1, \mathit{Ain} = 1, \mathit{Aout} = 0\rangle, \langle \mathit{Rin} = 1, \mathit{Rout} = 1, \mathit{Ain} = 0, \mathit{Aout} = 1\rangle, \langle \mathit{Rin} = 1, \mathit{Rout} = 1, \mathit{Ain} = 1, \mathit{Aout} = 1\rangle\}$. Since the effect of fault F_1 is premature firing of $\mathit{Aout}-$, so the value of Aout in the next states must be equal to 0. Therefore, we have considered the next states where $\mathit{Aout} = 0$. Now the set of next state becomes as $\{\langle \mathit{Rin} = 0, \mathit{Rout} = 1, \mathit{Ain} = 0, \mathit{Aout} = 0\rangle, \langle \mathit{Rin} = 0, \mathit{Rout} = 1, \mathit{Ain} = 1, \mathit{Aout} = 0\rangle\}$. Corresponding to the next state $\langle \mathit{Rin} = 0, \mathit{Rout} = 1, \mathit{Ain} = 0, \mathit{Aout} = 0\rangle$, the present state is $\langle \mathit{Rin} = 0, \mathit{Rout} = 1, \mathit{Ain} = 0, \mathit{Aout} = 1\rangle$ and the FD-transition is $\langle 0101, 0100\rangle$. Similarly, corresponding to the next state $\langle \mathit{Rin} = 0, \mathit{Rout} = 1, \mathit{Ain} = 1, \mathit{Aout} = 0\rangle$, the FD-transition is $\langle 0111, 0110\rangle$. The same was found in the faulty SG shown in Figure 6.

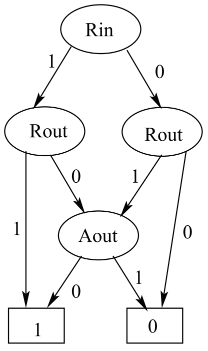

Fig. 7. Normal OBDD

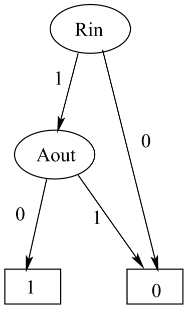

Fig. 8. Faulty OBDD

Next we discuss the procedure for applying OBDDs for generation of FD-transitions for a given fault (say F_i). Initially, the effect of the F_i is found from the STG model of the fault. Let signal v_k be one of the affected signals for the fault F_i which fires prematurely. The following steps are used to generate FD-transitions to detect the premature fire of v_k due to the fault F_i.

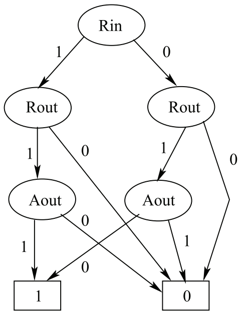

Fig. 9. XORed OBDD.

1) Find the Boolean expression for v_k from the circuit description under normal condition (say v_{k_N}). Generate OBDD for v_{k_N}. Let this OBDD be termed as "normal OBDD".
2) Find the Boolean expression for v_k from the circuit description under fault F_i (say $v_{k_{F_i}}$). Generate OBDD for $v_{k_{F_i}}$. Let this OBDD be termed as "faulty OBDD".
3) The two OBDDs (normal and faulty) are XORed and "satisfy-all-1" operation is applied on the resulting XORed OBDD. This operation on the XORed OBDD generates the set of next states that are attained by the CUT on firing of v_k (rising or falling) signal under fault F_i. Let NS_{v_k} be the set of such next states.
4) If F_i affects the rising signal of v_k, i.e., v_k+, then remove the states from NS_{v_k} where the value of v_k is 0. Let NS_{v_k+} be the set of such next states.
5) If F_i affects the falling signal of v_k, i.e., v_k-, then remove the next states from NS_{v_k} where the value of v_k is 1. Let NS_{v_k-} be the set of such next states.
6) Once the required set of next states (either NS_{v_k+} or NS_{v_k-}) is determined, then for each state $ns \in NS_{v_k+}$ (NS_{v_k-}), its corresponding present state (say, ps) is obtained by changing only the value of v_k in ns from 1 (0) to 0 (1) and other signal values remain unchanged. Finally, for each state $ns \in NS_{v_k+}$ (NS_{v_k-}), an FD-transition is generated which includes both present state and next state as $\langle ps, ns\rangle$, which detects the occurrence fault F_i in the CUT.

IV. Design of On-line tester using FD-transitions

The On-line tester is designed using FD-transitions and is executed parallelly with the CUT. It detects whenever the CUT traverses through any FD-transitions. Since the CUT is an asynchronous circuit, so the on-line tester must be an asynchronous circuit. We illustrate the design of on-line tester circuit using the FD-transition $\langle 0101, 0100\rangle$ (say FD_1). We have designed the tester circuit using CAD tool Petrify [1]. In case of FD_1, the signal Aout is changed from 1 to 0 (i.e., $\mathit{Aout}-$) when $\mathit{Rin} = 0$ and $\mathit{Rout} = 1$, whereas in normal case $Aout-$ fires when $Rin = 0$ and $Rout = 0$ (states $X7$ and $X10$ in Figure 5). So to detect whether FD_1 has occurred, the on-line tester taps lines $Rout$ and $Aout$ of the CUT and determines whether $Aout-$ has fired with $Rout = 1$; if so, a status output line (S) is became high. It can be noted that the line Ain is not required to tap because the value of Ain remains same under normal and faulty cases.

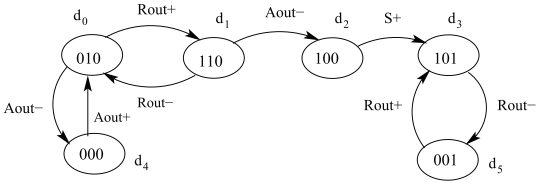

Fig. 10. SG for on-line tester to detect FD_1.

Figure 10 demonstrates on-line tester in the form of state graph model to detect FD_1. The state encoding ($\langle Rout, Aout, S\rangle$) of initial state d_0 is 010, where first two bits complement of $Rout = 1$ and $Aout = 0$, and third bit represents status output (S) which is 0 until FD_1 is detected. Once $Rout+$ occurs in the CUT, the on-line tester moves to state d_1. In d_1, if $Aout-$ fires (i.e., $Aout-$ fires when $Rout = 1$) then FD-transition has occurred in the CUT and hence fault is detected. This is accomplished by the on-line tester moving to state d_2 having encoding $\langle 100\rangle$. Following that, the tester moves to state d_3 by making status output (S) high, indicating fault has occurred in the CUT. In d_1, if $Rout-$ fires then the on-line tester moves to state d_0. The tester moves to state d_4 from state d_0 and back from d_4 to d_0 by the transitions $Aout-$ and $Aout+$, respectively. Similarly, the tester moves to state d_5 from state d_3 and back from state d_5 to d_3 by the transitions $Rout-$ and $Rout+$, respectively. We have added states d_4 and d_5 in the state graph in order to satisfy the proprieties of Complete State Encoding (CSE) and no liveness issues, which are mandatory in asynchronous circuit design. The state graph description of on-line tester is given as input to the Petrify and the output of Petrify is set of equations, from which on-line tester is designed. Figure 11 shows the on-line tester circuit for the state graph shown in Figure 10 to detect FD_1.

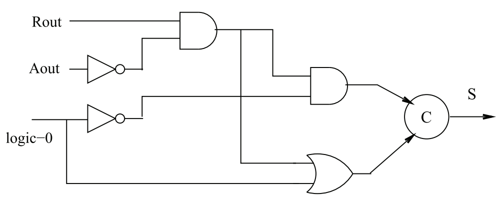

Fig. 11. On-line tester to detect FD_1.

V. Experimental evaluation

We have taken different asynchronous benchmark circuits to show the efficiency of the proposed scheme. In this work, we have studied two important parameters in OLT; Fault Coverage (FC) and Area Overhead (AO). We define fault coverage and area overhead as follows. A fault is said to be covered if at least one FD-transition of that fault is considered in on-line tester design.

FC = ((Total number of faults covered)/(Total number of faults in CUT)) $\times$ *100 %*

AO = (Area of on-line tester)/(Area of the CUT)

It can be noted that for a fault if there doesn't exist any FD-transition to detect it then such fault is called a redundant fault. Table I shows the details of fault coverage and area overhead achieved by the proposed scheme for different benchmark circuits and comparison of area overhead with the scheme reported in [23]. Column 1 provides the name of the circuit. Column 2 and Column 3 illustrate total number of gates and total number of faults of the circuit, respectively. In Column 4 and Column 5, we report the percentage of fault coverage and area overhead of the proposed scheme, respectively. Column 6 shows the area overhead of the scheme proposed in [23]. The following points may be noted.

- On an average 95% of faults are covered by the proposed scheme. However, remaining 5% of faults are not covered because in such cases there doesn't exist any FD-transitions (i.e., these faults are redundant faults).
- We cannot compare fault coverage of proposed scheme with [23] because the scheme reported in [23] based on checking of a predefined protocol, so fault coverage cannot be guaranteed.
- It can be seen in Table I that the area overhead of the proposed scheme is less than that of [23]. The reason is, the existing scheme ([23]) designed on-line tester circuit using Mutex elements and the area of the Mutex elements is high.
- It can be observed in the table that the area overhead of the proposed scheme decreases with increase in circuit size. This is because of common FD-transitions which can detect more than one faults simultaneously. The common FD-transitions are selected first for design of on-line tester circuit, hence area overhead is decreased.
- To the best of our knowledge, the size of the benchmark circuits for which the on-line tester could be designed by the proposed scheme are much higher than the ones considered in the OLT literature for asynchronous circuits. This scalability could be achieved due to the application of BBDs.

VI. Conclusion

In this paper, we have proposed a Binary Decision Diagram (BDD) based on-line testing scheme for asynchronous circuits. The proposed scheme works by representing circuit under normal and faulty conditions using separate Signal Transition Graphs (STGs), then corresponding Ordered Binary Decision Diagrams (OBDDs), normal and faulty OBDDs, are constructed from the STGs. The normal and faulty OBDDs are XORed and FD-transitions are generated from the XORed OBDD. Finally, on-line tester is designed using these FD-transitions. Experimentally, it has seen that on an average 95% of faults are covered and area overhead of the proposed scheme

TABLE I
Fault Coverage, Area overhead of the proposed scheme and comparison with existing scheme [23]

Circuit	Number of gates	Number of Faults	Fault Coverage (%)	Area Overhead ratio		CPU time ⋆
				Proposed scheme	Scheme reported in [23]	(seconds)
2 David cells	6	44	94	2.9	3.3	24
4 David cells	12	88	93	1.8	4.9	38
chu172	9	47	96	0.78	N/A	37
alloc-outbound	15	130	98	0.56	N/A	46
sbuf-read-ctl	19	152	95	0.46	N/A	53
sbuf-send-ctl	18	140	94	0.40	N/A	51
ram-read-sbuf	23	197	93	0.27	N/A	54

⋆ Intel Core i5-7200U CPU with 8 GB RAM in Linux OS.

is less than that of existing scheme. The proposed scheme can be easily applicable to any asynchronous circuit irrespective of it's design protocol. The proposed scheme selects FD-transition randomly from the set of FD-transitions for a fault in design of on-line tester circuit. However, the selection of FD-transitions can be better accomplished by solving a multi-criterion optimization problem using area and power of the tester, fault coverage, fault detection latency, importance of FD-transitions in functionally of the circuit, number of tappings in the critical paths, etc., as optimization parameters. Further, it is assumed that the on-line tester is free of faults. To make the OLT scheme more robust, testing of the on-line tester is required. This can be accomplished by designing another tester circuit for the on-line tester. Testing of the on-line tester can be performed either on-line or off-line. Off-line testing will incur less design issues, low area overhead, etc. However, in fault tolerant systems OLT of the on-line tester is necessary. Thus, further research is required to solve these issues.

References

[1] Petrify: a tool for synthesis of petri nets and asynchronous circuits. http://www.cs.upc.edu/~jordicf/petrify/, 1999.

[2] Hussain Al-Asaad. Efficient techniques for reducing error latency in on-line periodic built-in self-test. *IEEE instrumentation and measurement magazine*, 13(4):28–32, 2010.

[3] Y. Balasubrahamanyam, G. L. Chowdary, and T. J. V. S. Subrahmanyam. A novel low power pattern generation technique for concurrent BIST architecture. *International Journal Computer Technology and Applications*, 3(2):561–565, 2012.

[4] Anitha Balasubramanian, BL Bhuva, LW Massengill, B Narasimham, RL Shuler, TD Loveless, and W Timothy Holman. A built-in self-test (BIST) technique for single-event testing in digital circuits. *IEEE Transactions on Nuclear Science*, 6(55):3130–3135, 2008.

[5] P. K. Biswal and S. Biswas. Timed discrete event system approach to online testing of asynchronous circuits. In *Mediterranean Conference on Control and Automation (MED)*, pages 341–348, 2015.

[6] P.K. Biswal, H.P. Sambho, and S. Biswas. A discrete event system approach to on-line testing of digital circuits with measurement limitation. *Engineering Science and Technology, an International Journal, Elsevier*, 19(3):1473 – 1487, 2016.

[7] Pradeep Kumar Biswal and Santosh Biswas. A binary decision diagram based on-line testing of digital vlsi circuits for feedback bridging faults. *Microelectronics Journal, Elsevier*, 46(7):598 – 616, 2015.

[8] Santosh Biswas, Siddhartha Mukhopadhyay, and Amit Patra. A formal approach to on-line monitoring of digital VLSI circuits: theory, design and implementation. *Journal of Electronic Testing, Springer*, 21(5):503–537, 2005.

[9] M Bushnell and Vishwani Agrawal. *Essentials of Electronic Testing for Digital, Memory and Mixed-Signal VLSI Circuits*, volume 17. Springer Science and Business Media, 2006.

[10] Yung-Yuan Chen. Concurrent detection of control flow errors by hybrid signature monitoring. *IEEE transactions on Computers*, 54(10):1298–1313, 2005.

[11] Nachiketa Das, Pranab Roy, and Hafizur Rahaman. Built-in-self-test technique for diagnosis of delay faults in cluster-based field programmable gate arrays. *IET Computers and Digital Techniques*, 7(5):210–220, 2013.

[12] P. Drineas and Y. Makris. SPaRre: Selective partial replication for concurrent fault-detection in FSMs. *IEEE Transactions on Instrumentation and Measurement*, 52(6):1729–1737, 2003.

[13] Petr Fišer, Pavel Kubalík, and Hana Kubátová. An efficient multiple-parity generator design for on-line testing on FPGA. In *Euromicro Conference on Digital System Design Architectures, Methods and Tools*, pages 96–99, 2008.

[14] David A Hodges et al. *Analysis and Design of Digital Integrated Circuits, In Deep Submicron Technology (special Indian Edition)*. Tata McGraw-Hill Education, 2005.

[15] Henrik Hulgaard, Steven M Burns, and Gaetano Borriello. Testing asynchronous circuits: A survey. *Integration, the VLSI journal, Elsevier*, 19(3):111–131, 1995.

[16] D. Lu and C.Q. Tong. High level fault modeling of asynchronous circuits. In *Proceedings on VLSI Test Symposium*, pages 190–195, 1995.

[17] Chris J Myers. *Asynchronous circuit design*. John Wiley and Sons, 2004.

[18] Michael Nicolaidis and Yervant Zorian. On-line testing for VLSI– a compendium of approaches. *Journal of Electronic Testing, Springer*, 12(1-2):7–20, 1998.

[19] S. Biswas P. K. Biswal, K. Mishra and H. K. Kapoor. A discrete event system approach to online testing of speed independent circuits. *VLSI Design*, vol. 2015 (Article ID 651785):16 pages, 2015.

[20] Gopal Paul, Santosh Biswas, Chittaranjan Manda, and Bhargab B Bhattacharya. A BDD-based approach to design power-aware on-line detectors for digital circuits. In *International SOC Conference (SOCC)*, pages 343–346. IEEE, 2010.

[21] Philippe Perdu. Life issues, robustness consequences and reliability challenges for very deep sub micron technologies. In *Asia-Pacific International Symposium on Electromagnetic Compatibility*, pages 1014–1019, 2010.

[22] Bibhash Sen, Jyotirmoy Das, and Biplab K Sikdar. A DFT methodology targeting online testing of reversible circuit. In *International Conference on Devices, Circuits and Systems*, pages 689–693, 2012.

[23] D. Shang, A. Bystrov, A. Yakovlev, and D. Koppad. On-line testing of globally asynchronous circuits. In *International On-Line Testing Symposium*, pages 135–140, 2005.

[24] D. Shang, A. Yakovlev, F.P. Burnsand, F. Xia, and A. Bystrov. Low-cost online testing of asynchronous handshakes. In *European Test Symposium*, pages 225–232, 2006.

[25] Lanfang Tan, Ying Tan, and Jianjun Xu. CFEDR: Control-flow error detection and recovery using encoded signatures monitoring. In *International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems*, pages 25–32, 2013.

[26] T. Verdel and M. Yiorgos. Duplication-based concurrent error detection in asynchronous circuits: Shortcomings and remedies. In *International Symposium on Defect and Fault-Tolerance in VLSI Systems*, pages 345–353, 2002.

[27] I. Voyiatzis, A. Paschalis, D. Gizopoulos, C. Halatsis, F. S. Makri, and M. Hatzimihail. An input vector monitoring concurrent BIST architecture based on a precomputed test set. *IEEE Transactions on Computers*, 57(8):1012–1022, 2008.