Testability of AND-EXOR Based Iterative Logic Arrays

Avik Chakraborty
avik@bhelhyd.co.in

Abstract— Iterative Logic Arrays (ILAs) are ideal as VLSI sub-systems because of their regular structure and it’s close resemblance with FPGAs (Field Programmable Gate Arrays). AND-EXOR based circuits are of interest in the design of very low power circuits where energy loss implied by high frequency switching is of much consideration. This paper examines the testability of AND-EXOR based Iterative Logic Arrays (ILAs). For certain ILAs it is possible to find a test set whose size remains constant irrespective of the size of the ILA, while for others it varies with array size. Former type of ILAs is known as Constant-Testable, i.e. C-Testable. It has been shown that AND-EXOR based Logic Arrays are C-Testable and size of test set is equal to number of entries in cell’s truth table. The test generation problem has been shown to be related to certain properties of cycles in a set of graphs derived from cell truth table. By careful analysis of these cycles an efficient test generation technique that can be easily converted to an ATPG program has been presented for both 1D and 2D ILAs. How this property of ILAs can be used for testing FPGAs has also been discussed.

Index Terms—C-Testable, Iterative Logic Arrays, Bijective, Test Generation, ATPG

I. INTRODUCTION

In this paper we examine testability of Iterative Logic Arrays (ILA’s) constructed from a library of AND-EXOR gates called controlled-NOT (CNOT) gates. A k-CNOT gate has k + 1 input wires and k + 1 output wires. It transmits the first k input signals unchanged, and inverts the last input signal iff the first k inputs are all 1; clearly this input-output mapping is bijective. Figure 1 shows examples of k-CNOTs and the standard graphic symbols used for them. If k = 0, a k-CNOT is a simple NOT gate or inverter. Since a 2-CNOT gate can implement the NAND function, any Boolean function can be implemented by a k-CNOT circuit A k-CNOT gate can be implemented using 1 k-input AND gate and an EXOR gate as shown in Fig 2. Frequency of switching in a k-CNOT gate is very less, i.e. 1/2^k. Figure 3 shows k-CNOT implementation of a single ILA cell.

The testing of ILAs has been widely studied in the past and even more so in recent times due to advances in VLSI which have made these structures attractive to the circuit designer. Most test generation techniques use the regular cell interconnection structure of the ILAs in one way or the other [1, 2, 3, 4, 5].

The paper is organized as follows. In section 2 the assumed Single Cell Fault (SCF) model has been discussed. In section 3 C-Testability of ILAs has been proven and it has been shown that number of tests required is equal to number of entries in the cell truth table. In section 4 we generate tests for 1D ILAs. In section 5 we present test generation technique for 2D ILAs which can be actually extended for any n-dimensional ILAs. Section 6 concludes the paper with a brief discussion on possible application of this test technique in testing FPGAs.

Figure 1: (a) NOT (b) CNOT and (c) general k-CNOT gate

II. THE FAULT MODEL

We assume that the truth table of a single ILA cell can be altered in any manner in presence of a fault as long as it remains combinational. In such a scenario all possible inputs must be applied to each of the ILA cells and we must ensure any such single cell fault (SCF) must propagate to the observable outputs of the ILA (for ex. in case of 2D ILA, vertical outputs of the last row and horizontal outputs of the last column).
The complexity of the TS in the faulty cell. It maps every distinct row to that faulty cell. This implies 2^{th + v} tests suffice to detect all SCFs in 2D ILAs.

Consider 1D and 2D ILAs constructed from the cell shown in Fig. 3. Both the ILAs can be tested for SCFs using only 2^3 = 8 tests. Any ILA constructed from (2, 1)-cells can not be tested for SCFs by less than 8 tests. This testability is known as Optimal-Testability (i.e., O-Testability), where number of tests is independent of size of the ILA and equal to number of entries in the cell truth table. The truth table for the cell has been shown in Fig. 4. It can be proven that 2^{th + v} -testability implies following properties:

1. The x-transition diagram (transitions on h wires) consists of disjoint Euler tours. See Fig. 5.
2. The y-transition diagram (transitions on v wires) consists of disjoint Euler tours. See Fig. 6.
3. The state transition diagram (transitions on h \mid v, where \mid is concatenation operator) comprises of edge-disjoint Euler tours. See Fig. 7. The corresponding x and y transitions also form edge-disjoint Euler tours in x and y-transition diagram respectively.
4. The state transition (concatenation of transitions of all dimensions) of an n-dimensional ILA can be decomposed into disjoint Euler tours.
5. The transition diagram for each dimension corresponding to decomposition of state table also consists of edge-disjoint Euler tours.

III. C-TESTABILITY OF ILAS

ILA consists of identical ILA cells arranged in a geometrically regular interconnection pattern. A cell which receives h horizontal inputs \( h_{\text{in}} \), v vertical inputs \( v_{\text{in}} \) and produces h horizontal outputs \( h_{\text{out}} \), v vertical outputs \( v_{\text{out}} \) is referred to as \((h, v)\)-cell. See Fig. 3. The \( k_{\text{max}} \) of the k-CNOT gates that implement the internal circuitry of the ILA cell is bounded by: 

\[ k_{\text{max}} \leq (h + v - 1) \]

Due to bijectivity it maps every distinct input \((h_{\text{in}}, v_{\text{in}})\) to \((h_{\text{out}}, v_{\text{out}})\). We denote the cell function realized by \((h, v)\) cell as \( f \) where \( f: \{0,1\}^h \times \{0,1\}^v \rightarrow \{0,1\}^h \times \{0,1\}^v \). \( f \) is clearly bijective. In ILA of one dimension the cells are connected in a line. In 2D ILA cells are interconnected in a rectangular structure.

**Theorem 1:** All SCFs in one dimensional ILA of \( p \) \((h, v)\)-cells can be detected by \( 2^{(h + v)} \) tests independent of \( p \).

**Proof:** As the cell function \( f \) realized by \((h, v)\) cell is bijective, any single cell fault propagates to either \( h_{\text{out}} \) or \( v_{\text{out}} \) of the cell in which the failure occurs. As all other cells are bijective, this ensures that any SCF propagates to an observable output \( h_{\text{out}} \) or \( v_{\text{out}} \) of the rightmost cell or \( v_{\text{out}} \) of all \( p \) cells) for any vertical input \( v_{\text{in}} \) to that faulty cell. This implies \( 2^{(h + v)} \) tests suffice to detect all SCFs.

**Theorem 2:** All SCFs in two dimensional ILA of \( p \) rows and \( q \) columns constructed from \((h, v)\)-cells can be detected by \( 2^{(h + v)} \) tests independent of \( p \) and \( q \).

IV. ONE-DIMENSIONAL ILAS

We use property 1 to generate tests for 1D ILAs. Given, cell truth table we construct x-transition diagram and decompose it into disjoint Euler tours. Please note that x-transition diagram can be decomposed in more than one ways, and each of the decomposition gives a solution. We illustrate the test generation for 1D ILA constructed from cells shown in Fig. 3. If we decompose the x-transition diagram of Fig. 5 as follows:

\[ \{(q_0, 0), (q_0, 1), (q_1, 0), (q_2, 1), (q_2, 1) \rightarrow (q_1, 1), (q_3, 0) \rightarrow (q_2, 0)\} \],

then we generate following 8 tests: TS\( ^{\text{ID}_1} \) applies \( (q_0, 0) \) in all the cells; similarly TS\( ^{\text{ID}_2} \), TS\( ^{\text{ID}_3} \), TS\( ^{\text{ID}_4} \) apply \( (q_0, 1), (q_1, 0), (q_3, 1) \) respectively in all the cells; TS\( ^{\text{ID}_5} \) applies \( (q_1, 1) \) in odd cells and \( (q_3, 1) \) in even cells; TS\( ^{\text{ID}_6} \) applies \( (q_1, 1) \) in odd cells and \( (q_1, 1) \) in even cells; similarly TS\( ^{\text{ID}_7} \) and TS\( ^{\text{ID}_8} \) alternate \( (q_0, 0) \) and \( (q_3, 0) \) along the row. The complexity of the algorithm is \( O(E + V) \) (\( E = \) no. of edges = \( 2^{(h + v)} \) and \( V = \) no. of vertices = \( 2^{(h + v)} \)) = \( O(1) \).

V. TWO-DIMENSIONAL ILAS

2D ILAs constructed from \((h, v)\) cells can be tested for SCFs by using \( 2^{(h + v)} \) tests. For the ILA cell of Fig. 3, 2D ILA built from this cell can be tested by 8 tests only. In the first approach, we use property 1 in conjunction with prop. 2 to generate tests for 2D ILA. We first generate tests...
for 1D ILA (we ignore vertical dimension and treat it as vertical input/output to the cell); we decide upon the decomposition of the x-transition diagram. Then for that decomposition we check whether it satisfies certain property in the y-transition diagram. If it satisfies the property we generate tests for 2D ILA using tests of 1D ILA.

**Corollary 1:** For a 2D ILA, if x-transition diagram can be decomposed into disjoint Euler tours in more than one ways, then there is at least one such decomposition for which the corresponding transitions form a closed walk in the y-transition diagram.

Consider the decomposition used in the previous section to derive tests for 1D ILA constructed from the same cell. The transitions \((q_0, 0), (q_0, 1)\) form a closed loop in the y-transition graph. This implies that we can have tests TS\(^{2D}\)\(_1\), which applies TS\(^{1D}\)\(_1\) in the odd rows and TS\(^{1D}\)\(_2\) in the even rows; TS\(^{2D}\)\(_2\) which applies TS\(^{1D}\)\(_2\) in the odd rows and TS\(^{1D}\)\(_1\) in the even rows. Similarly transitions \((q_1, 0), (q_2, 1)\) form a closed walk in the y-transition graph. But the problem arises with the loop \((q_1, 1) \rightarrow (q_3, 1)\). The corresponding transitions in the y-transition diagram are \((v_1, q_1)\) and \((v_1, q_3)\). They clearly don’t form a closed walk; hence, we can not derive tests for 2D ILA using this decomposition. The implication of this is illustrated with the help of Fig. 8. The cell with bold outline should have been \(q_3\), and then only we could have derived 8 tests for the 2D ILA using this decomposition.

Instead, if the x-transition diagram is decomposed as follows: \{\((q_0, 0), (q_0, 1), (q_2, 0), (q_2, 1)\), \((q_1, 1) \rightarrow (q_3, 0) \rightarrow (q_2, 0) \rightarrow (q_1, 1)\}\), then the corresponding transitions for the last cycle (first two and second two transitions form closed walks in the y-transition diagram as before) in the y-transition diagram are:

\[000, 010, 011, 111, 100, 101\]
In fact, all the problem lies in obtaining all the BUTs (Block Under Test) while others should be made as helper cell and each PLB with a helper PLB constitutes a cell in the 2D ILA. In the next session the helper PLBs are made

Corollary 2: If state transition diagram can be decomposed into disjoint Euler tours in more than one ways, then for each of such decompositions the corresponding transitions in x and y transition diagrams also form closed walks.

Consider the 3 cycles in the state transition diagram in Fig. 7. These cycles correspond to following transitions:

\((q_0, 0) \rightarrow (q_0, 1) \) and \((v_0, q_0) \rightarrow (v_1, q_0) \)
\((q_1, 0) \rightarrow (q_1, 1) \rightarrow (q_3, 1) \) & \((v_0, q_1) \rightarrow (v_1, q_1) \rightarrow (v_3, 1) \)
\((q_2, 0) \rightarrow (q_2, 1) \) & \((v_2, q_0) \rightarrow (v_2, q_1) \rightarrow (v_4, 1) \)

Please note that all of them (two transition pairs from (1) and four transition triplets from (2) & (3)) form closed walks.

Let’s derive tests for 1D ILA: ATPG\(^{1D}\) applies \((q_0, 0)\) in all the cells. ATPG\(^{1D}\) applies \((q_0, 1)\) in all the cells. ATPG\(^{1D}\), ATPG\(^{1D}\) alternate \((q_1, 0) \rightarrow (q_1, 1) \) in every 3 cells. Let’s derive tests for 2D ILA: ATPG\(^{2D}\) alternate ATPG\(^{1D}\), ATPG\(^{2D}\) in every 2 rows. ATPG\(^{2D}\), ATPG\(^{2D}\) alternate ATPG\(^{1D}\), ATPG\(^{2D}\) in every 3 rows. Similarly ATPG\(^{2D}\), ATPG\(^{2D}\) alternate ATPG\(^{1D}\), ATPG\(^{2D}\) in every 4 rows. 

VI. CONCLUSION

In this paper, we characterize the testability properties of both 1D, 2D and nD ILAs. The ATPG method of test derivation for 2D ILAs can be extended to any n-dimensional ILAs. Lots of research works have been reported in literature on iterative logic arrays primarily because of its regular structure and close resemblance with FPGAs. FPGAs can be reconfigured as iterative logic arrays and the idea of ILA testing can be applied in testing FPGAs. The only problem lies in the fact that no. of outputs is normally far less than no. of inputs in each PLB. (i.e. Programmable Logic Block; FPGA is a 2D array of PLBs.) The solution is that some of the PLBs should be made BUTs (Block Under Test) while others should be made as helper cell and each PLB with a helper PLB constitutes a cell.

**REFERENCES**

[1] S. M. Reddy, “Easily Testable Realizations for Logic Functions,” IEEE Trans. on Computers, vol. C-22, no. 12, Dec 1973.
[2] A. D. Friedman, “A functional Approach to Efficient Fault Detection in Iterative Logic Arrays,” IEEE Trans. on Computers, vol. 39, no. 5, May 1990.
[3] C. W. Wu, P. R. Cappello, “Easily Testable Iterative Logic Arrays,” IEEE Trans. on Computers, vol. 39, no. 5, May 1990.
[4] A. D. Friedman, “Easily Testable Iterative Systems”, IEEE Trans. on Computers, vol. C-22, no. 12, pp. 1041–1064, Dec 1973.
[5] A. Chatterjee, J. A. Abraham, “Test Generation for Iterative Logic Arrays based on an N-cube of cell state model”, IEEE Trans. on Computers, vol. 40, no. 10, pp. 1133-1148, Oct 1991.
[6] C. Stroud, E. Lee, S. Konala, M. Abramovici, “Using ILA Testing for BIST in FPGAs”, Proc. International Test Conf., Oct. 1996, pp. 68-75.