Proceedings of the 5th WSEAS Int. Conf. on Instrumentation, Measurement, Circuits and Systems, Hangzhou, China, April 16-18, 2006 (pp39-42)

# Efficient Built-In Self-Test Techniques for Sequential Fault Testing of Iterative Logic Arrays

*SHYUE-KUNG LU* and *MAO-YANG DONG*
Department of Electronic Engineering
Fu-Jen Catholic University, Taipei, Taiwan, R.O.C.

*Abstract:-* In today's nanometer technology era, more sophicated defect mechanisms might exist in the manufactured integrated circuits which are not covered by traditional fault models. In order to ascertain the quality of shipped chips, more realistic fault models should be addressed. In this paper, we propose built-in self-test (BIST) techniques for iterative logic arrays (ILAs) based on *realistic sequential cell fault model (RS-CFM)*. According to the proposed testability conditions and the adopted fault model, exhaustive *SIC (single input change)* pairs for a cell are applied to each cell in the ILA. The outputs can be propagated to the primary outputs and then observed. The *SIC component generator* and *output response analyzer* are also designed as the BIST circuitry. Due to the regularity of ILAs, the hardware overhead of the BIST circuitry is almost negligible. In order to illustrate our approach, a pipelined array multiplier is used as an example. The number of test pairs for completely testing of the array is only 72. Moreover, the BIST overhead to make it delay fault testable is about 1.67%.

## 1 Introduction

Iterative logic arrays are widely used in designing application-specific integrated circuits (ASICs) due to their regularity and modularity. The most popular applications include the datapath parts of digital circuits and digital signal processors. It is well known that the complexity of the general logic testing problem is *NP-complete* [1]. Fortunately, many novel approaches [2, 3] were proposed to ease the testing problem for ILAs and the most widely adopted fault model is the *single cell fault model (SCFM).* Owing to the rapid growth in VLSI technology, more sophicated failure mechanisms might exist in the manufactured circuit. This makes the traditional SCFM insufficient to ascertain the quality of shipped products.

Therefore, more comprehensive fault models should be established. These include the *transistor stuck-open*, the *gate delay*, and the *path delay* fault models. These fault models are also defined as *sequential fault models* [4]. To detect sequential faults, two-pattern tests are usually required. Several works have been proposed for test generation of sequential faults [5, 6]. However, combinational ILAs are addressed.

In [4], a novel sequential fault model, called *Realistic Sequential Cell Fault Model* (RS-CFM) was proposed. It is a more comprehensive, cell-level and implementation independent model suitable for ILA testing. The problem of robust two-pattern sequential fault test generation is addressed. Therefore, the important problem of *test invalidation* in sequential fault testing is solved. The adopted RS-CFM is defined as follows [4]:

- At most one ILA cell can be faulty at a time.
- The test set of RS-CFM contains all the possible *Single Input Change (SIC)* pairs, i.e., pairs of vector $<v_1, v_2>$ with Hamming distance 1, that generate at least one change at the cell outputs.
- The test set described above must be applied to every ILA cell (*test application*). A fault may change the horizontal outputs (and/or vertical outputs) of the cell.
- Faulty cell outputs must be propagated to primary outputs of the ILA *(fault propagation).*

According to this fault model, SIC pairs must be applied to every cell in the array. It is shown that SIC pairs can achieve very high sequential fault coverage and contribute to robustness [7]. The C-testability conditions for one-dimensional and two-dimensional ILAs can be found in our previous work [8]. However, we still lack of built-in self-test techniques for sequential fault testing of such ILAs. Therefore, BIST techniques are proposed in this paper. Both the *SIC component generator* and *output response analyzer* are designed as the BIST circuitry. The hardware overhead of the BIST circuitry is almost negligible. In order to illustrate our approach, a pipelined array multiplier is used as an example. The number of test pairs for completely testing of the array is only 72. Moreover, the hardware overhead to make it delay fault testable is about 1.67%.

The organization of this paper is as follows. Section 2 reviews C-testability conditions for 2-D ILAs. Section 3 proposes the BIST architecture. An array multiplier is used as an example in Section 4. Finally, some conclusions are given in Section 5.

## 2 Review of C-Testability Conditions for 2-D Iterative Logic Arrays

Proceedings of the 5th WSEAS Int. Conf. on Instrumentation, Measurement, Circuits and Systems, Hangzhou, China, April 16-18, 2006 (pp39-42)

In order to ease our discussion, some definitions are defined first. They are also used in [2, 8].

**Definition:** A *cell* in an ILA with function $f$ is a combinational machine $(\Sigma, \Delta, f)$, where $f: \Sigma \rightarrow \Delta$ is the cell function, and $\Sigma = \Delta = \{0, 1\}^w$, $w$ denotes the word length of a cell. An ILA is an array of cells.
**Definition: A** *Single Input Change* pair is a pair of vectors $<v_1, v_2>$, where the Hamming distance between $v_1$ and $v_2$ is 1, $v_1, v_2 \in \{t_1, t_2, t_3, ..... t_n\}$, $n=2^w$.
**Definition:** We say that the function $f$ of a cell is *bijective* when $\forall \theta_1 \neq \theta_2, f(\theta_1) \neq f(\theta_2)$, $\theta_1, \theta_2 \in \Sigma$.
**Definition:** A *complete SIC sequence* ($SIC_{com}$) is a two-pattern sequence that contains exhaustive SIC pairs of a cell.

The cell behaviors of a pipelined ILA can be viewed as a *finite state machine (FSM)*. The state transition graph $G_t(V, E)$ of the cell function $f$ is a directed graph, where $V = \Sigma$ and $E = \{(v, f(v))| \ v \in V\}$ [8]. Since each sequential fault requires a two-pattern test, therefore, we should define the state transition graph of $f$ with respect to a 2-pattern sequence. This transition graph is denoted as

$$G_t^2 = (V_t^2, E_t^2),$$

where

$$V_t^2 = \{< v_1, v_2 > | \ v_1, v_2 \in \Sigma\},$$
$$E_t^2 = \{(v, f(v)) \mid v \in V_t^2\}.$$

The characteristics of the transition graph $G_t^2$ determine the controllability and observability of the whole array. Three conditions are proposed in [8]. For simplicity, we only consider the case that the cell function is bijective. DFT techniques can be used to make the cell function bijective [8]. For this case, $G_t^2(\Sigma_t^2, E_t^2)$ will only consists of *components.* If a component of $G_t^2$ contains at least one SIC pair, then this component is called a *SIC component (*$SIC_{comp}$*)*. We assume that there are $n$ SIC components in $G_t^2$. Then, the test set $S$ which contains all SIC pairs of a basic cell is represented as

$$S = \{t \mid t \in \bigcup SIC_{compi}, \quad 1 \leq i \leq n\}.$$

It is evident that $S$ is a complete SIC sequence. Let $L_i$ denote the number of vertices in $SIC_{compi}$, then the length of the test set S can be expressed as:

$$| S | = \sum_{i=1}^{n} L_i.$$

Let $t_{ij}$ denote the $i^{th}$ two-pattern test in $SIC_{compj}$, $1 \leq i \leq L_j$, $1 \leq j \leq n$. If we apply $t_{ij}$ to a cell, we obtain the output sequence $f(t_{ij})$. Since $t_{ij}$ is a vertex of $SIC_{compj}$ in $G_t^2$, $f^{L_j}(t_{ij}) = t_{ij}$. We define $T_j$ as

$$T_j = \{t_{1j}, t_{2j,...,} t_{L_j j}\}.$$

**Theorem:** A mesh-connected ILA as shown in Fig. 1 is C-testable under RS-CFM if the cell function is bijective.

*Proof:* If we apply all two-pattern tests $T_1 = (I_1, J_1)$ to $cell_{11}$, and design the adaptive vertical and horizontal input sequences as shown in Fig. 1. $cell_{rs}$ and $cell_{pq}$ will receive the same input sequence if $r + s = p + q$ (indicated by the dashed lines in Fig. 1). Fault propagation in this case is straightforward due to the component property of $G_t^2$. Therefore, the testing process is a parallel one. Similarly, we can apply $T_j$, $1 \leq j \leq n$, to $cell_{11}$ sequentially. Then, the complete SIC sequence $S$ can be sent to each cell in the array. Fault propagation can be achieved due to the bijective cell function. Therefore, the whole array is said to be *C-testable* under RS-CFM since the number of tests is constant regardless of the array size. ❒

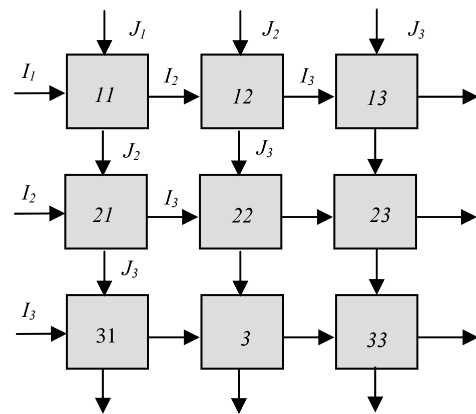


Fig. 1: Test tessellaton of a mesh-connected iterative logic array.

## 3 Built-In Self-Test Techniques

To discuss the built-in self-test schemes, we first assume that the cell function is bijective such that the theorem described above can be applied directly. The general BIST architecture consists of the following:

1. the original ILA under test,
2. the SIC component generator, which generates all the two-pattern test pairs of the test set S,
3. the mechanism for routing the generated test set S to each cell in the ILA,
4. the mechanism for routing the responses to the output response analyzer,
5. the output response analyzer, and
6. the test controller.

The control circuitry is simply a switch, which decides the operation mode of the ILA (*test mode* or *normal mode*). The remaining items (2)-(5) are the critical parts for cost-effective implementation of the BIST architectures. Based on the test pattern tessellation shown in Fig. 1, a typical BIST structure containing items (1)-(5) is shown in Fig. 2. In the figure, extra circuit elements and wires are highlighted. The SIC component generator is marked *SICCG*, which is simply a logic circuit generating all the component patterns contained in $G_t^2$. Since the

SICCG is a bit-level implementation, which is very small as compared to the whole ILA.

The multiplexers are marked *M,* which are controlled by a mode-selection signal that indicates whether the circuit is in test mode or normal mode. Consider the ILA shown in Fig. 2, if it is in test mode, all multiplexers take the inputs which are drawn in thick line segments. Therefore, if *SICCG* generates the test set *S* for $cell_{11}$, all other cells subsequently receive their own complete SIC sequences according to the theorem described above. When it is in normal mode, all multiplexers take the inputs which are drawn in thin line segments, i.e., the test paths are disabled, and the array returns to normal operation. The performance penalty is almost negligible since the multiplexers can be implemented with simple pass transistors or transmission gates.

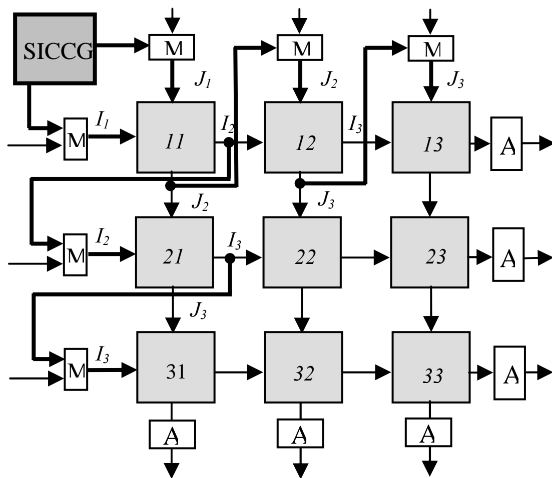


Fig. 2: The proposed BIST structure, including SIC component generator, output response analyzer (A), and routing mechanisms.

In Fig. 2, the output response analyzers are marked *A*. According to the special features of the test patterns—a complete SIC sequence for each cell—there are a variety of approaches for output response analysis. The first approach is the *check-sum* approach. By using this approach, each *A* module in Fig. 2 is replaced with an accumulator, i.e., a binary adder. Since the complete SIC sequence is fixed, therefore, the accumulated output is also fixed. Judging from the accumulated output, we can identify whether the output sequence is faulty or not. By using this approach, the resulted fault coverage is high. Aliasing will occur for multi-bit errors corresponding to different permutations of the complete SIC sequence.

We can also use parity checkers to check the parity of each output bit. A complete SIC sequence *S* indicates that the parity for each output bit position must be fixed. Therefore, we can check the parity of each output bit to determine whether the output bit is faulty or not. We can also have other approaches to perform output response analysis, e.g., transition detectors and comparators. Due to the tessellation property of the test patterns, all cells whose indices sum to the same number, i.e., those lie in the same 45° line generate the same output sequence. Therefore, comparators can be used to compare the primary outputs with the outputs lie in the same 45° line.

## 4 Testable Design of Array Multiplier

An array multiplier performs the operation $x \times r$. A 4 × 3 array multiplier [8, 9] and its cell structure is shown in Fig. 3. The 4-bit *x* multiplicand propagates downward. The 3-bit multiplier *r* can be preloaded and stored in the A-reg's. The 7-bit product can be received from the cells in the rightmost column. Each cell performs the functions given as follows:

$$x_o = x_i$$
$$r_o = r_i$$
$$s_o = s_i \oplus c_i \oplus r_i x_i$$
$$c_o = s_i c_i + c_i r_i x_i + s_i r_i x_i$$

where $x_i$ and $r_i$ represent multiplier and multiplicand bits, $s_i$ is the summand bit, and $c_i$ is the carry bit. Their respective output bits are denoted as $x_o$, $r_o$, $s_o$, and $c_o$, which are propagated to the next stage. When a cell contains a number *i*, it is the index of an *r* bit. These numbers thus range from 0 to 2. The cell's register A holds the *r* bit indicated. A cell that has no index contains a 0 bit. Each cell is essentially a latched 1-bit full adder.

From [8], we know that the cell function is not bijective. Therefore, DFT techniques can be applied. The resulted SIC components in $G_t^2$ for $A= 0$ are shown in Fig. 5. From the SIC components we can see that all SIC pairs are included in these components. Therefore, the previous theorem can be applied directly. Moreover, the BIST structure shown in Fig. 2 is also suitable for the array multiplier. Our next problem is to design the SIC component generator for the array multiplier. The proposed structure is shown in Fig. 6.

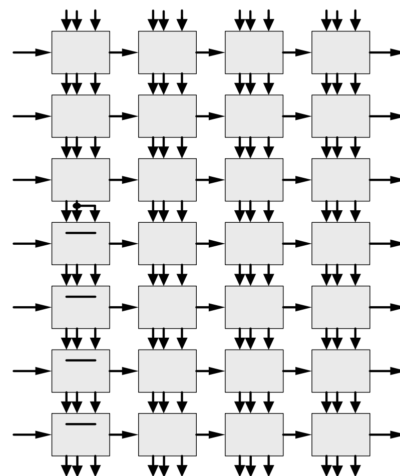

Fig. 3: A 4 × 3 array multiplier.

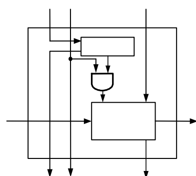

Fig. 4: The multiplier cell.

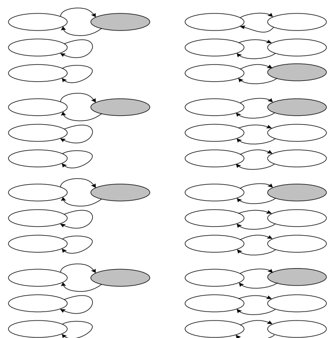

Fig. 5: SIC Components for A = 0.

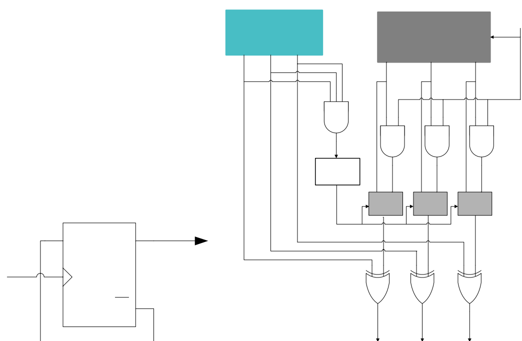

Fig. 6: The SIC Component generator for array multiplier.

In Fig. 6, the SIC component generator contains a 3-bit binary counter which count from 000 to 111, a 3-bit barrel shifter, and some logic gates. The clock cycle of the counter is six times that of the barrel shifter. The content of the barrel shifter ($X_2X_1X_0$) is initialized to 001 and the output of the control module is 0. The barrel shifter is shifted right for each BIST clock cycle. We have to notice that the BIST clock has the half frequency of the multiplier clock. The BIST mode includes two phases. During *phase1* the generator produces all SIC pairs. Alternatively, during *phase2,* it will produce a sequence of two-pattern tests with Hamming distance 2. The value of X during *phase1* can be computed as:

$$BIST_X_i = (X_i \bullet Bist_clk) \oplus C_i,\ 0 \le i \le 2$$

Similarly, $X$ can be computed as follows during phase2.

$$BIST_X_i = X_i \oplus C_i,\ 0 \le i \le 2$$

An implementation of the control module is presented in Fig. 6(b). Initially, the flip-flop is set to ‘0’. When the flip-flop is enabled by the ‘*n*’ signal, it indicates that the counter has reached the value 111 and the output of the control module is ‘1’. Therefore, the generator enters into *phase2*. The simulated results of the generator are shown in Fig. 7.

(a)

(b)

Fig. 7: (a) Test patterns of the SIC component generator during Phase 1, and (b) Test patterns during Phase 2.

## 5 Conclusions

Built-in self-test (BIST) techniques for iterative logic arrays (ILAs) based on *realistic sequential cell fault model (RS-CFM)* are proposed in this paper. A transition graph model is used to model the behaviors of a cell. Exhaustive *SIC (single input change)* pairs for a cell can be applied to each cell in the ILA. The *SIC component generator* is designed and used as the test patten generator. Due to the regularity of ILAs, the hardware overhead of the BIST circuitry is almost negligible. In order to illustrate our approach, a pipelined array multiplier is used as an example. The number of test pairs for completely testing of the array is only 72. Moreover, the BIST overhead to make it delay fault testable is about 1.67%.

## Reference


[1] H. Fujiwara and S. Toida, “The complexity of fault detection problems for combinational logic circuits,” *IEEE Trans. Computers,* vol. C-31, no. 4, pp. 555-560, June 1982.

[2] C. W. Wu and P. R. Cappello, “Easily Testable Iterative Logic Arrays,” *IEEE Trans. Comput.,* vol. 39, no. 5, pp. 640-652, May 1990.

[3] C. Y. Su and C. W. Wu, “Testing iterative logic arrays for sequential faults with a constant number of patterns,” *IEEE Trans. Comput.,* vol. 43, no. 4, pp. 495-501, Apr. 1994.

[4] M. Psarakis, D. Gizopoulos, A. Paschalis, and Y. Zorian, “Sequential fault modeling and test pattern generation for CMOS iterative logic arrays,” *IEEE Trans. Computers*, vol. 49, no. 10, pp. 1083-1099, Oct. 2000.

[5] D. Gizopoulos, D. Nikolos, and A. Paschalis, “Testing CMOS Combinational Iterative Logic Arrays for Realistic faults,” Integration: *The VLSI J.,* vol. 21, pp. 209-228, 1996.

[6] P. Girard, C. Landrault, V. Moreda and S. Pravossoudovitch, “An Optimized BIST Test Pattern Genrator for Delay Testing,” in *Proc. VLSI Test Symp.,* pp. 94-99, April 1997.

[7] G. L. Smith, “Model for Delay Faults Based upon Path,” in *Proc. IEEE International Test Conf.,* pp. 342-349, 1985.

[8] S. K. Lu, “Delay Fault Testing for CMOS Iterative Logic Arrays with a Constant Number of Patterns,” *IEICE Trans. Info. and Syst.,* vol. E86-D, no. 12, pp. 2659-2665, Dec. 2003.

[9] J. V. McCanny and J. G. McWhirter, “Completely iterative pipelined multiplier array suitable for VLSI design,” *IEE Proc.*, vol. 129, no. 2, pp. 40-46, Apr. 1982.

[10] M. Psarakis, D. Gizopoulos and A. Paschalis, “Built-in sequential fault self-testing of array multipliers,” *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems,* vol. 24, no. 3, pp. 449-460, Mar. 2005.

[11] M. Psarakis, D. Gizopoulos, A. Paschalis and Y. Zorian, “An Effective BIST Architecture for Sequential Fault Testing in Array Multipliers,” *in Proc. 17th IEEE VLSI Test Symp.*, 1999, pp.252-258.