Microelectronics Journal 52 (2016) 91–103

Contents lists available at ScienceDirect

Microelectronics Journal

journal homepage: www.elsevier.com/locate/mejo

# A cellular automata based highly accurate memory test hardware realizing March C$^-$

Mousumi Saha [a,*], Mamata Dalui [b], Biplab K Sikdar [c]

[a] Department of Computer Applications, National Institute of Technology, Durgapur, India
[b] Department of Computer Science and Engineering, National Institute of Technology, Durgapur, India
[c] Department of Computer Science and Technology, Indian Institute of Engineering Science and Technology, India

ARTICLE INFO

*Article history:*
Received 9 December 2014
Received in revised form
19 February 2016
Accepted 14 March 2016
Available online 6 April 2016

*Keywords:*
March test
Cellular automata
Single length cycle cellular automata
SSLCA
TSLCA

ABSTRACT

This work reports a highly accurate test structure for high speed memories. The theoretical bases of the design are the March algorithm and cellular automata (CA) proposed by von Neumann in 1950s. Theory of 3 and 5-neighborhood CA, employed for the current application, has been developed to enhance the self-testability of memory test logic. The special class of single length cycle attractor cellular automata, introduced in this work, accepts status of each memory word and evaluates it to decide on the faults in the memory. The extension of CA neighborhood to 5 enables propagation of the effect of faults in memory or in the test logic to the error line of the test structure. This overcomes the inability of classical memory test hardware designed with the *ex*−*or* and *or* logic.



## 1. Introduction

The exponential growth of compaction density in memories compels the designers to devise expensive test logic. The traditional memory testing algorithms [1,2] are the Zero-One (Scan Test or MSCAN [3]), Checkerboard, Walking 1/0 [4], GALPAT, Sliding Diagonal [5], and many others. Zero-one and Checkerboard tests require linear time but achieve poor fault coverage. GALPAT and Walking 1/0 tests are of $O(n^2)$ and $O(n^{\frac{3}{2}})$ complexity, where $n$ is the size of memory. However, the March test [6] is found to be the most powerful as well as efficient than the classical pattern based test techniques. It is extensively used for functional testing of semiconductor memories and provides a linear time solution with high fault coverage.

Comparatively recent approach, introduced to improve the testability, is the redesign and augmentation of peripheral circuits in and around the RAM. This is popularly referred to as the design for testability (DFT). The memory BIST (built-in self-test) approaches [7–9] add even more extra hardware with the target to reduce the test time of a memory device. ABIST (Array BIST) [10], used for embedded memories, is an effective form of MBIST (memory BIST). However, the advantages of ABIST are often offset by its overhead requirements.

March test is also an MBIST algorithm. A large variations of March test algorithms are reported in the literature [6,11]. The main goal of these approaches is to reduce the memory testing time. The authors in [12] introduced March C with time complexity of the order of $11n$ ($n$, number of memory bits) to target SAFs (stuck-at faults), TFs (transition faults), and all CFs (coupling faults). This is further improved in March C$^-$ by [5] to achieve the complexity of $10n$. The other notable work on March algorithms are the [13–15].

In 1980s, Wolfram [16] studied a family of simple 1-dimensional cellular automata (CA) that could simulate complex system behaviors. A special class of Wolfram's 3-neighborhood CA, called the linear/additive CA had been employed for developing effective methodologies for VLSI design [17]. The CA had also been found effective for efficient design of fault detection and diagnosis schemes in VLSI circuits [18–20]. This motivates us to address the issue of high speed fault detection in memories and to devise an efficient test logic around CA that can reduce test time simultaneously satisfying the requirement of insignificant test overhead.

The test design, proposed in this work, is developed around a special class of 1-dimensional cellular automata referred to as the single length cycle attractor CA. The CA is configured for realizing the March test [11]. It runs on the data (memory word) read from a memory location and computes the status of the memory cells (faulty or non-faulty). The properties of single length cycle attractor CA are then exploited to memorize the status of the word. The final state, at the end of test run, indicates faults (if any) in memory.

* Corresponding author. Tel.: +919434788194.
*E-mail addresses:* msaha.nitd@gmail.com (M. Saha), mamata.06@gmail.com (M. Dalui), biplab@cs.iiests.ac.in (B. Sikdar).

http://dx.doi.org/10.1016/j.mejo.2016.03.009
0026-2692/

The CA based proposed test structure computes the status of memory words in parallel with the memory read/write operation of a March element that effectively reduces the number of comparisons required in the conventional test designs. The CA neighborhood is extended to 5 to enable identification of faults (if any) in a memory word even if the test logic is defective. The regular and modular structure of CA, enabling segmentation of the test hardware, realizes the memory testing in reduced test time as well as a scalable test design for high speed memories. It better suits for low cost VLSI implementation of the test hardware for a memory device that is inherently regular in structure. The CA based test design is introduced in Section 4 following a brief on March test in Section 3. The details of the design developed around the single length cycle cellular automata are reported in Section 5. Section 6 reports performance in terms of test accuracy of the proposed structure. The test design around 5-neighborhood CA with enhanced test accuracy is detailed out in Section 7. The next section introduces the preliminaries of CA that have relevance for the current work.

## 2. CA preliminaries

A cellular automaton (*CA*) evolves in discrete space and time, and can be viewed as an autonomous finite state machine (*FSM*). Each *CA* cell stores a discrete variable at time $t$ that refers to the present state (PS). The next state (NS) of the cell at $(t+1)$ is affected by its state and the states of its *neighbors* at $t$. In this work, we consider the 1-dimensional CA, where a cell is having two states – 0 or 1. The next state of $i$th *CA* cell in 3-neighborhood is

$$S_i^{t+1} = f_i(S_{i-1}^t, S_i^t, S_{i+1}^t)$$

$S_{i-1}^t$, $S_i^t$ and $S_{i+1}^t$ are the present states of the left neighbor, self and right neighbor of the $i$th cell at time $t$ and $f_i$ is the next state function. The $S^t = (S_1^t, S_2^t, \ldots, S_n^t)$ at $t$ defines the present state of the *CA*. Therefore, the next state of an $n-$cell *CA* is

$$S^{t+1} = (f_1(S_0^t, S_1^t, S_2^t), f_2(S_1^t, S_2^t, S_3^t), \ldots, f_n(S_{n-1}^t, S_n^t, S_{n+1}^t)) \quad (1)$$

The next state function of the $i$th CA cell can be expressed in the form of a truth table (Table 1). The decimal equivalent of the 8 outputs (NSs) is called 'rule' $R_i$. In a 2-state 3-neighborhood CA, there can be $2^8$ (256) rules. Three such rules 51, 254 and 255 are illustrated in Table 1. The first row lists the combinations of PSs of $(i-1)$th, $i$th and $(i+1)$th cells at $t$. The last three rows indicate the NSs of $i$th cell at $(t+1)$. The logic functions that are realized by the rules 254 and 255 are shown in Fig. 1.

**Table 1**
*RMTs* of the *CA* $< 51, 254, 255 >$.

| PS | 111 | 110 | 101 | 100 | 011 | 010 | 001 | 000 | *Rule* |
|---|---|---|---|---|---|---|---|---|---|
| RMT | (7) | (6) | (5) | (4) | (3) | (2) | (1) | (0) | |
| NS | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 51 |
| NS | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 254 |
| NS | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 255 |

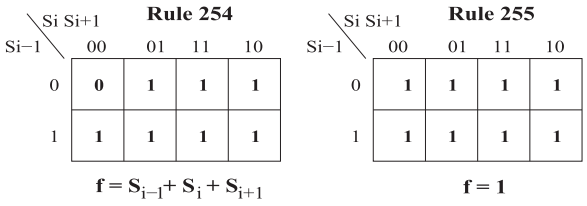


**Fig. 1.** Next state functions for rules 254 and 255.

**Definition 1.** The set $R = < R_1, R_2, \ldots, R_i, \ldots, R_n >$ of rules that configures the cells of a CA is called the rule vector of the CA. If all the $R_i$s are same, then the CA is a uniform CA; otherwise it is a non-uniform/hybrid CA.

**Definition 2.** A CA is a null boundary CA if the left (right) neighbor of the leftmost (rightmost) terminal cell is permanently fixed to 0-state. That is, $S_0^t = S_{n+1}^t = 0$ in Eq. (1).

**Definition 3.** A set of states forms loop (cycle) in the state transition diagram of a CA and is referred to as the *attractor*. An attractor forms a basin with the states that lead to the attractor. The maximum distance traversed to reach an attractor from any other state is the depth of the CA.

Fig. 2(a) is an $n$-cell 3-neighborhood non-uniform (hybrid) null boundary CA. Fig. 2(b) shows the state transition of a 4-cell CA $< 1, 236, 165, 69 >$. The cycles with states $7 \to 7$ and $9 \to 1 \to 9$ form the attractors. The set of states 0,2,4,5,6,7,10,11,12,13,14,15 forms the 7-basin and the depth of the CA is 5 (distance from state 2 to 7). The CA with single length cycle attractor ($7 \to 7$ of Fig. 2(b)) is of our current interest.

**Definition 4.** A combination of the present states $S_{i-1}^t, S_i^t, S_{i+1}^t$ (1st row of Table 1) is referred to as the Rule Min Term (RMT) in 3-neighborhood.

The column 011 of Table 1 is the 3rd RMT. The next states corresponding to this RMT are 0 for rule 51, and 1 for 254 and 255.

## 3. March test

The March test [5] is found to be the most powerful scheme for fault detection in memories. It is extensively used for functional testing and provides a linear time solution with high fault coverage. In the current design, we consider effective realization of March $C^-$ for determining correctness of memory function. However, any March algorithm, that is considered to be efficient in terms of fault coverage or any other parameter, can be realizable in the framework of proposed cellular automata (CA) based test hardware.

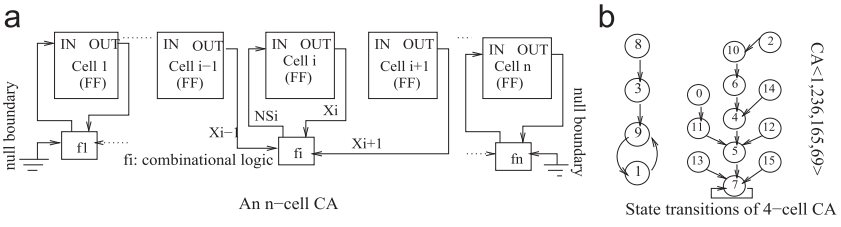


**Fig. 2.** An $n$-cell null boundary CA.