Cellular Automata: The computing machine in VLSI era

by

Biplab K Sikdar

Bengal Engineering and Science University
West Bengal, India - 711103
biplab@cs.becs.ac.in  bksikdar@hotmail.com
Bengal Engineering and Science University

• Formerly Bengal Engineering College
• Second oldest engineering institute in Asia
• Celebrating its 150th anniversary
• Running 10 engineering & 4 science departments
• Conducting research in emerging fields
• Encouraging Collaborative research projects
• CA research in VLSI era receives recognition
Cellular Automata Research

Two functional profiles

- Development of sound theoretical basis of Cellular Automata (CA)
- Demonstrating its effectiveness in VLSI application domains
Motivation

• Machine Computer

  Demands high speed computation \( \sim 10^{15} \)

• Information Processor
  - Small size, high speed, low cost, low power
  - New technology, new architecture, efficient soft support
Motivation

• Technology

Vacuum tube → IC → MSI → LSI → VLSI → Nano-Tech → QCA

• Requirement

- $10^{14}$ transistor to replace a human brain

<table>
<thead>
<tr>
<th>BK Sikdar</th>
<th>Green Party</th>
</tr>
</thead>
<tbody>
<tr>
<td>Made in Taiwan</td>
<td>Made in India</td>
</tr>
<tr>
<td>2020</td>
<td>2030</td>
</tr>
</tbody>
</table>
CA Technology for target applications

- Modeling of Physical Systems
- Pattern Recognition
- Classification
- Data Compression
- Cryptosystem
- Authentication
- Error Correcting Codes
- VLSI Circuit Testing
- Cellular Mobile Network
- Sensor Network
The Coverage

• Cellular Automata
• Hierarchical Cellular Automata (HCA)
• HCA Applications in VLSI Domain
• Non-Linear CA
• Concluding Remarks
Cellular Automata

- Autonomous Finite state machine
- A cell is updated at every clock cycle
- The state of a cell is dictated by its immediate neighbors
- Major structures proposed in
  - 50’s - J Von Neumann
  - 80’s - Wolfram
  - Late 80’s - Work at IIT
  - EAR 90’s - Work at BECDU

Objective is to simulate complex computation
Concentration on CA states  Next state logic  Mathematical formalism
Cellular Automata

A typical CA cell

- Autonomous machine
- Next state logic (combinational logic) is of importance

4-cell GF(2) CA

Null Boundary CA
**Cellular Automata**

A 2-state 3-neighborhood CA cell

- **Next state logic/CA Rules**
  - RMT/lsr: 111 110 101 100 011 010 001 000
  - Next state: 1 0 0 1 0 1 1 0 - rule 150
  - 0 1 0 0 1 0 1 1 - rule 75

256 CA rules

- For 150, next state=r ⊕ s ⊕ r
- 150 is linear 75 non-linear rule

- • **Linear CA characterized by a T matrix** – essentially dependency matrix
  
  \[
  T = \begin{bmatrix}
  1 & 1 & 0 & 0 \\
  1 & 0 & 1 & 0 \\
  0 & 1 & 1 & 1 \\
  0 & 0 & 1 & 0 
  \end{bmatrix}
  \]

  \[\text{NS} = T \times \text{PS}\]

- • The elements are in GF(2) – GF(2) CA

- 4-cell GF(2) CA

- Null Boundary CA
Cellular Automata

State transitions

Next state logic/CA Rules

RMT/ lsr: 111 110 101 100 011 010 001 000

Next state: 1 0 0 1 0 1 1 0 - rule 150
0 1 0 1 1 0 1 0 - rule 90
GF(2) Cellular Automata

State Transition Behavior

Group CA

All states lie on some Cycle

NON MAXIMUM LENGTH GROUP CA

MAXIMUM LENGTH GROUP CA

Employed for VLSI circuit testing
GF(2) Cellular Automata

State Transition Behavior

Complemented CA

\[ <153,153,153,153> \]

![State Transition Diagram](image)

Employed in CA based data encryption

Non Group CA

cyclic/non-cyclic/non reachable states

![State Transition Diagram](image)

CA Based Classifier

Zero tree 1- basin = (1,4,11,14)

Non-reachable states = 2,3,4,5,10,11,12,13  Attractors = 0,1,8,9
Hierarchical Cellular Automata

- GF(2) CA – 0 level of hierarchy – a CA cell contains a FF

- GF(2^p) CA – single level of hierarchy – a CA cell contains p FFs
Hierarchical Cellular Automata

• Provides Abstraction.
• Provides hierarchy to look at Problem from higher levels

• A bit
• A set of $p$ bits (pixel)
• A set $p \times q$ bits (pixel block)

0/1

1 2 p
0 to $2^p - 1$

1 2 k q

1 2 p
0 to $2^{pq} - 1$

$\rightarrow$ GF(2)

$\rightarrow$ GF($2^p$)

$\rightarrow$ GF($2^p$)$^q$
T_{ij} = \begin{cases} w_{ij}, & \text{if next state of } i^{\text{th}} \text{ cell depends on present state of } j^{\text{th}} \text{ cell} \\ 0, & \text{otherwise} \end{cases}

Next state \quad X_{next} = T \times X_{current}

Elements of T & X are \{ 0, \alpha, \alpha^2, \ldots, \alpha^{2^{p-1}} \} \in \text{Galois Extension Field } \text{GF}(2^p)

\alpha - is the generator of GF(2^p)

Utility?  - Abstraction & Hierarchy  Transistor - Gate Level - RTL Level
Extension Field  \( \mathbb{GF}(2^p) \)

Elements of \( \mathbb{GF}(2^p) \)

\( \alpha \)-matrix \( [M]_{pxp} \)
- char poly of \( M \) is generator poly
\( \alpha_j \) matrix = \( M^j \)

Vector representation a column vector of \( \alpha_j \) matrix

Operations On \( \mathbb{GF}(2^p) \) Elements

Addition  Multiplication

follow the generator polynomial

Compute \( NS = T \times PS \)
HCA Characterization

Group HCA  All states lie on some Cycle  $\det[T] \neq 0$

Non Group HCA: cyclic/non-cyclic/non reachable states

Group HCA for HCABIST/Encryption  Non group for classification /diagnosis
Special Class of Non Group GF(2^p) CA

Non reachable states

Attractors

MACA

PEF specifies the attractor
Extension of Extension Field

• GF(2^{pq}) can be viewed as extension of GF(2^p) - that is GF(2^{pq})
  Similarly, GF(2^{pq...}) \rightarrow GF((2^p)^q) \cdots GF(2^6) as GF(2^3) or GF(2^3^2)

Elements of GF(2^{pq}) are \{ 0, \beta, \beta^2, \ldots, \beta^{2^{pq}-1} \}

Elements of \beta are \{ 0, \alpha, \alpha^2, \ldots, \alpha^{2^p-1} \} \quad \alpha - is the generator in GF(2^p)

• In GF(2^3) generator \beta is a 3 x 3 matrix with elements \in GF(2^2) field

• Generator \alpha in GF(2^2) represented by a 2 x 2 binary matrix
The HCA

Each $GF(2^p q^r)$ CA cell contains $r$ no. of sub-cells
A $GF(2^p q^q)$ cell consists of $q$ no. of sub-cells each having $p$ FFs

$GF(2)$ CA : 0-level of hierarchy
$GF(2^p)$ CA : single-level of hierarchy
$GF(2^p q^q)$ CA : two-level of hierarchy
A 3 cell $\text{GF}(2^3)^3$ Hierarchical CA

A $\text{GF}(2^6)$ CA cell can be viewed as $\text{GF}(2^{3^2})$ or $\text{GF}(2^{2^3})$ CA cell

Field elements are partitioned

- Generator: $\alpha_3 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$
  
  - Generator polynomial: $x^6 + x^3 + x^1 + x^3 + 1$

  a) An element of $\text{GF}(2^6)$ with $p=3, q=2$

- Generator: $\beta_1 = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$
  
  - Generator polynomial for $\beta_1$: $x^3 + 2x^2 + 2x + 2$

  b) Structure of an element in $\text{GF}(2^3)^3$ with $p=3, q=2$

- Generator: $\beta_2 = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$
  
  - Generator polynomial for $\beta_2$: $x^3 + 3x + 6$

  - Generator polynomial for $\alpha_3$: $x^3 + x^2 + 1$

  c) Structure of an element in $\text{GF}(2^3)^3$ with $p=3, q=2$

Effective for engineering application – VLSI testing
Reduced Computation Overhead

- Multiplication and Addition are avoided by Star and Plus tables - Substituted by simple table lookup
- Size of the matrices is reduced by a factor of $p^2 \text{ GF}(2^p)$
- Overall execution time decreased by the order of $p^2$
- Overhead to store the Star and Plus tables
- Overhead to implement in hardware
Application of Hierarchical Cellular Automata in VLSI Circuit Testing

Objective

• Design of HCA based BIST structure
• Efficient fault diagnosis in VLSI circuits
• Handling Prohibited Patterns in BIST design
VLSI Circuit Test Solutions

• Requirements
  - High fault coverage, Low overhead, At-speed test

• Solution with BIST Structure

• Conventional TPG  - Built around LFSR/CA
  - No sound methodology
  - Ad-hoc, Independent of CUT

• Solution? - Hierarchical CA for testing VLSI Circuits
  - Theory of extension field  Sound methodology
  - Tuned to a CUT    Reasonable overhead
GF(2) CA Based BIST

- **GF(2) CATPG**
  - Based on 1-D CA (maximum length)
- **CARE**
  - maximum length 1-D CA

Test Solutions with HCA

HCABIST

CUT with hierarchical description

An Illustration

36 PI

Selected value of

pq = 9

p=3 and q=3 for all cells

value of n = 4

HCATPG designed with 4-cell GF(2^3) CA

We employ group HCA
Experimental Results

Randomness test results

Fault efficiency
Overhead Reduction
Folding of Test Structure

$k$ input CUT is tested by $n'–cell$ GF(2, $B$) CATPG, $n'x$ < $k$

multi-input PL clusters (A & B) single-input PL clusters (C, D, & E)

Maximum no. of single-input PL clusters in any module = 2 (block 3)

$n' = 4$

Before Folding

After Folding
Assume #POs = 4

If unique faulty signature set for
module 1 is \{11,22,33\}
module 2 is \{01,10,23\}
module 3 is \{02,20,31,13\}

And signatures \{03,12,30\} are common to more than one block

The D1-MACA can act as a classifier to identify faulty module in L
MACA as a Classifier

- Generic Classifier

\[
< s_0 > < s_1 > \ldots < s_i > \ldots < s_m >
\]

\[ s_{i1}, s_{i2}, \ldots s_{ir} \]

\[ s_{ij} = j^{th} \text{ symbol string in the } i^{th} \text{ cluster} \]

- CA Based Classifier

CA effectively provides implicit storage mechanism
Multi-class Classifier using 2-class

Faulty Signature


MACA tree

Fault in partition

1  2  3  4  5
Design TPG Without PPS

Problem definitions

• Pseudo-random test pattern generation

• Patterns declared prohibited for a CUT (PPS)
  Set of patterns which may place the system into an undesirable state
  Example: Toggle State of a flip flop

• Cellular automata (CA) based solution
  - CA based TPG without PPS ensures desired randomness quality
  - Requires no additional cost

GF(2) CA with 0-level of hierarchy are considered for the design
Design TPG Without PPS

- Problem Definition
  PRPG \rightarrow WITH MAXLENGTH CA

- Solution:
  Non Maximum Length CA

Our TPG Design is based on this type of CA
The Design

• Non-maximal length group CA to design TPG without PPS

Given PPS
0000110
0000010
0001001
0000111
PPS = 0001111
0010100
1101101
1011001
0100100
0010001

• PPS are made to fall on RCs
The Design

- All members of PPS don’t fall in RCs
  - 80% of PPS fall on RCs
- Sacrifice a small part of TC
- PPS fall on larger cycle (TC) are avoided by computing $D_{\text{max}}$

Target cycle (TC) is used for generation of test patterns
Pattern Recognition

Attractors and Transient States    Behaves as an associative memory
The design implements lossy/lossless compression

CA based design of codebook for specific domain
CA based search to reduce encoding time
Same structure can be tuned to any level between lossy to lossless
Experimental Results

- Girl
- PSNR 34.91 db
- Compression ratio 98.73 %
Experimental Results

Original Image

Decompressed Image

- Proj18
- PSNR 31.40 dB
- Compression ratio 98.83 %
Non-Linear Cellular Automata

Scalable Test Structure

Non-Linear CA based Design

*Group CA/PRPG for any* $n$ *can be synthesized*

*Synthesis of PRPG takes* $O(n)$ *time*

Scalable structure

*From an n-cell group CA, n+1-cell group CA synthesis takes one time step*
PRPG Design

Requirements:

1. Design a CA with large cycle
   - Non-group CA (x)
   - Group CA (✓)

2. Randomness of the patterns/ cycle length should be high
   - Group CA any rules (x)
   - Group CA with specified rules (✓)

Demands efficient group CA synthesis scheme satisfying the requirements
**PRPG Design**

Group CA Synthesis: $O(n)$ time
Based on characterization of CA rule (RMT)

CA Rule

Group rule  Non-group rule

Presence of a *non group rule* makes the CA as non group
All other rules are *group rules*

$<105, 177, 170, 75>$ is a group CA  all are group rules
$<105, 177, 171, 75>$ is a non-group CA  171 is non-group rule

*A particular sequence of group rules forms a group CA*
PRPG Design

Group CA Synthesis: $O(n)$ time
Based on characterization of CA rule (RMT)

**Unbalanced rules are non-group**

177 (10110001) balanced  171 (10101011) unbalanced

62 group rules out of 70 balanced rules

A particular sequence of 62 group rules forms a group CA

$<105, 177, 170, 75>$ group CA

$<105, 170, 177, 75>$ non-group CA

177 does not belong to the rule class of 3rd cell

Task is to find the rule class of CA cells
PRPG Design

6 classes of group rules - a rule may belong to different classes
Class of \((i+1)^{th}\) cell - determined from \(i^{th}\) rule and class of \(i^{th}\) cell
Class of first cell - determined only from the first rule set

\[
\begin{array}{c|c|c|c|c}
R_i & Cl of R_i & Cl of R_{i+1} \\
\hline
105,54,.. & III & \\
\hline
I & 102,150,.. & III \\
II & 45,75,.. & I \\
III & 177,216,.. & V \\
& 89,101,.. & VI \\
V & 170,85 & II \\
\end{array}
\]

\(<105, 177, 170, 75>\) group CA

\((n+1)-cell\) group CA synthesis from \(n\)-cell group CA

\[
\begin{array}{c|c|c|c|c}
R_i & Cl of R_i & Cl of R_{i+1} \\
\hline
105,177,170,75 & \\
\hline
105,177,170,75 & III & \\
\hline
105,177,170,75,150 & III \\
\end{array}
\]

5-cell group CA
PRPG Design

Randomness in CA rules

RMT: 111 110 101 100 011 010 001 000
Next state: 0 1 1 0 1 0 0 1 - rule 105

For better randomness each rule of group CA should obey

Property 1: The RMT pairs (0 & 1), (2 & 3), (4 & 5) and (6 & 7) should have different values.

Property 2: The RMT pairs (0 & 4), (1 & 5), (2 & 6) and (3 & 7) should have different values
Experimental Observations

Study of randomness property
Platform used is DiehardC
Testing Multiple Cores

- Scalable PRPG Design

\[ \text{Concatenation} \]

\[ \square-\square-\square-\square + \square-\square-\square-\square-\square \]

n1-cell group CA + n2-cell group CA

\[ \text{State space} \]

\[ \circ \circ \circ \circ \circ \circ \circ \]

Results

\[ \square\square\square\square\square\square\square\square\square\square\]

n1+ n2-cell group CA

\[ \text{State space} \]

\[ \circ \circ \circ \circ \]

Employed to devise test logic for VLSI chip with multiple cores
Testing Multiple Cores

- Scalable TPG
Concluding Remarks

- CA act as an effective tool for modeling physical system
- Hierarchical CA is the better choice
- CA applications can further be extended to
  - Modeling Physical Systems
  - VLSI Design & Testing
    - Protein folding - Genetic Code
  - Data Compression
  - Data security - Bio-metric authentication
  - Cellular mobile network — query processing - location management
Thanks for listening