

# Modeling and Synthesis of Molecular Memory

Ph. D. Thesis

**RENU KUMAWAT**  
(I.D. No. 2008REC503)



**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING**  
**MALAVIYA NATIONAL INSTITUTE OF TECHNOLOGY JAIPUR**

**November 2015**



# Modeling and Synthesis of Molecular Memory

*by*

RENU KUMAWAT

DEPARTMENT OF ELECTRONICS AND COMMUNICATION  
ENGINEERING

*Submitted*

*in fulfillment of the requirements*

*of*

*the degree of*

DOCTOR OF PHILOSOPHY

*to the*



MALAVIYA NATIONAL INSTITUTE OF TECHNOLOGY  
JAIPUR, INDIA

November 2015



*This thesis is dedicated to  
My mother-in-law,  
Mrs. Manju Lata Mamoria*



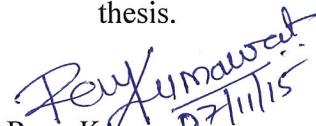
Malaviya National Institute of Technology  
Department of Electronics and Communication Engineering

Date: November 7, 2015

**Certificate**

It is certified that:

1. The thesis has not been submitted in part or full to any other University or Institute for the award of any degree.
2. I, Renu Kumawat (2008REC503) have fulfilled the requirements for submission of the thesis.


  
Renu Kumawat  
07/11/15


Institute ID: (2008REC503)

Date: November 7, 2015

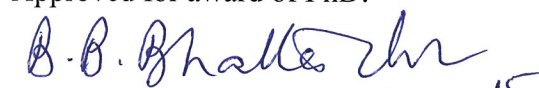
This is to certify that Ms. Renu Kumawat (2008REC503) has worked under our supervision for the award of degree of Philosophy in Electronics and Communication Engineering on the topic entitled "*Modeling and Synthesis of Molecular Memory*". She has successfully defended her thesis work in viva voce examination in front of ODC and satisfactorily incorporated the changes suggested by examiners in this revised version of the thesis.

Thesis was examined and has been recommended for award of degree of PhD.

  
(Vineet Sahula)  
Professor, ECE Department  
MNIT, Jaipur  
India  
(Supervisor: Internal Examiner)

  
(Manoj Singh Gaur)  
Professor, CSE Department  
MNIT, Jaipur  
India  
(Supervisor: Internal Examiner)

Approved for award of PhD.

  
(Bhargab B. Bhattacharya) 07-11-15  
Professor, ISI Kolkata

Date: November 7, 2015

Place: Jaipur, India





---

## Certificate

This is to certify that the dissertation entitled "*Modeling and Synthesis of Molecular Memory*" being submitted by **Renu Kumawat** to the Department of Electronics and Communication Engineering, Malaviya National Institute of Technology, Jaipur, for the award of the degree of Doctor of Philosophy, is a bona fide research work carried out by her under our supervision and guidance. The results obtained in this thesis have not been submitted to any other university or institute for the award of any other Degree.

**Dr. Vineet Sahula**

Professor

Dept. of Electronics & Comm. Engg.

MNIT, Jaipur

**Dr. Manoj Singh Gaur**

Professor

Dept. of Computer Engg.

MNIT, Jaipur



---

## Acknowledgment

First and foremost, I would like to express my profound gratitude to my supervisors, Dr. Vineet Sahula and Dr. Manoj Singh Gaur for giving me an opportunity to work under their noble guidance. My sincere thanks to them for their invaluable guidance, generous support, encouragement and deepest sense of motivation during the hard times of my work. Their constant support and compassion helped me complete my work fruitfully and successfully. Without their insights, advice and encouragement, this dissertation would not exist. I am grateful to them for all the things that they have taught me and for being very patient with me. They made my Ph.D. research an enjoyable learning experience indeed.

I would like to express my sincere thanks to Prof. V. Sinha, Prof. R. P. Yadav, Prof. Vijay Janyani, Prof. S. G. Modani, Prof. K. C. Jain, Prof. D. Boolchandani, Prof. K. K. Sharma, Dr. Lava Bhargava, Dr. Md. Salim, Dr. C. Periasamy, Dr. Md. Samar Ansari Dr. S. J. Nanda for their help, useful comments and encouragement while patiently listening to my proposals during all progress presentations. Dr. Vijay Laxmi deserves a special mention as she has always been a big help to me. I am thankful for her strong backing and unconditional support. My gratitude extends to Prof. D. Boolchandani for giving me moral support and encouragement during my tough times.

All my thanks to fellow research colleagues for their warm friendship. In particular, I am grateful to my colleagues Lokesh Garg, Rajesh A. Patil and Sapna Khandelwal, for useful discussions, constant moral support and valuable assistance provided to me in times of need. I owe my debt towards Pankaj Jha, Namita Sharma, Manjit, Mahesh Soni, A. M. Reddy, Rahul Sharma and Arjun Singh Chauhan for their useful feedbacks during our informal discussions. I am also thankful to staff of VLSI Lab of the ECE department, especially Lab Engineer, Mr. Alok Sharma. I also thank all my colleagues in the ECE department for their cooperation.

I am thankful to Prof. Vineet Sahula and Prof. D. Boolchandani for providing me financial assistance to attend conferences during my Ph.D. through the funded research project "Special Manpower Development Project for VLSI Design and related software", phase II, sponsored by Ministry of Communication and Information Technology, India.

---

I take this opportunity to sincerely thank Mrs. & Mr. Vineet Sahula, Mrs. & Mr. Manoj S. Gaur for their kind hospitality.

I feel very proud of my in-laws. My parents in-law, Mr. Rajendra Kumar Verma & Mrs. Manju Lata, have constantly encouraged me to pursue this research endeavor with the perseverance. I will always be grateful to them for the sacrifices they have made for me and for the faith and confidence they have always had in me. I am thankful to them for their boundless love and blessing they have bestowed upon me. My husband, Mr. Satendra Kumar Mamoria, daughter, Rishika and son, Avanish, deserves a special mention for the tremendous patience they have shown and the moral support they have provided to me during my research work.

Finally, I would like to express my heartfelt thanks to my beloved parents, Mr. Suwa Lal Kumawat & Mrs. Rekha Kumawat, for their blessings, support and persistence which has always been a source of inspiration that keeps me going to complete this venture. Thanks to all my other family members, whose name I could not specify here. They have always taken pride in my achievements and have always been a source of encouragement to me.

Above all, I thank God for providing me this life, and the opportunity and ability to pursue this research endeavor. I am thankful to my beloved *didi*, late Mrs. Sangeeta Mamoria, whom I lost during the course of Ph.D. She has always been and will always remain the pillars of strength for me to move forward in my life.

I thank all who have directly or indirectly helped me in my work. Apologies to anyone whose help is not acknowledged.

Renu Kumawat

## Abstract

Molecular electronics is emerging as one of the promising alternatives to the present CMOS technology. Such a device is fabricated by chemical self-assembly of mono or multiple layers of single or array of molecules, to manifest the behavior of a wire, a switch or a latch. Researchers have demonstrated the switching behavior of these molecules and have fabricated simple logic functions as well as memory by using such programmable molecules. However, the characteristics of single or few molecular devices are extremely sensitive to the external parameters such as contacts, nanogap, environment, etc. Such a sensitivity poses a serious design challenge to realize the reliable molecular devices. In contrast to other molecular devices, a nanocell consists of conducting metal nanoparticles connected via self assembled monolayer of molecules. Tour et *al.* [1] demonstrated that the nanocell device has an in-built defect tolerance, ultra high density, post fabrication programmability through mortal training and hence lack the need for precise molecular ordering. These features make the nanocell a good choice for future nano-scale devices.

One of the primary goals of this thesis is to develop modeling and post fabrication synthesis algorithms for a nanocell based molecular memory device. Also, in this thesis we have analytically proved that such a memory can withstand environmental uncertainties. A model has been developed on the basis of already proposed analytical framework for molecular devices. The model is based on circuit behavior of nitro-substituted Oligo (Phynylene Ethynylene) (OPE) molecule. This model is subsequently used to simulate crossbar molecular devices as well as nanocell based 1-bit molecular memory and verified using HSPICE. The concept is further augmented by post fabrication synthesis of 2-bit molecular memory using external

---

control signal voltages. Most suitable high and low voltage values for these control signals is a design space search problem. This search is handled by Genetic Algorithm such that some of the molecules switch to '*ON*' or to '*OFF*' state and the nanocell is programmed to behave as a 2-bit memory cell. Our results demonstrated that the proposed methodology is versatile enough to train nanocell for multi-bit storage functionality.

Further, a computational framework is proposed to compute the probability of retrieving the stored data bits correctly at the output terminal of the proposed nanocell. During exploration, this nanocell configuration is simulated by systematically varying the number of nanoparticles and molecular switches. It is observed that, the probability of existence of at least one path, from input to output, approaches close to unity with presence of at least 20 or more nanoparticles in the nanocell. During memory model validation, 1000 samples of 1-bit memory (consisting of 20 nanoparticles) are generated using Monte Carlo simulation and verified for read and write operations. It is observed that such a memory cell can successfully perform read and write operations for more than 99.5% of the untrained nanocell based 1-bit memory samples. Thus, it can be stated that the model verification results obtained for this memory cell closely matches to those obtained using analytical results from probabilistic graph model.

A novel extension over the continuous parameter birth-death model is also proposed to estimate the reliability of a nanocell, in presence of transient errors. On the basis of our model, an algorithm is developed and implemented in MATLAB, PERL and HSPICE, to generate a representation for a given nanocell. Theoretical results for reliability estimations are validated by simulating HSPICE model of nanocell in presence of varying defect rates. It is observed that the device reliability increases with increase in the number of nanoparticles and molecules, which is validated by theoretical formulation.

# Contents

|   |              |
|---|--------------|
| <b>Certificate</b>  | <b>i</b>     |
| <b>Acknowledgment</b>                                       | <b>iii</b>   |
| <b>Abstract</b>   | <b>v</b>     |
| <b>List of Figures</b>                                      | <b>xiii</b>  |
| <b>List of Tables</b>                                       | <b>xv</b>    |
| <b>List of algorithms</b>                                   | <b>xvi</b>   |
| <b>List of Abbreviations</b>                                | <b>xvii</b>  |
| <b>List of symbols</b>                                      | <b>xviii</b> |
| <b>1 Introduction</b>                                       | <b>1</b>     |
| 1.1 Taxonomy of Emerging Memory Devices . . . . .           | 3            |
| 1.2 Historical Perspective: Molecular Electronics . . . . . | 5            |
| 1.3 Motivation . . . . .                                    | 7            |
| 1.4 Scope of work . . . . .                                 | 9            |
| 1.5 Contributions of the thesis . . . . .                   | 11           |
| 1.6 Organization of thesis . . . . .                        | 13           |

---

|          |  |           |
|----------|--|-----------|
| <b>2</b> | <b>State of the Art</b>  | <b>14</b> |
| 2.1      | Organic Molecule: an active device . . . . .                           | 15        |
| 2.1.1    | Characteristics and Synthesis . . . . .                                | 16        |
| 2.1.2    | Modeling Approaches . . . . .  | 21        |
| 2.2      | Crossbar based Molecular Devices: design and modeling . . . . .        | 24        |
| 2.3      | Probabilistic Modeling Approaches for Nano-scale Sub-Systems . . . . . | 29        |
| 2.3.1    | Markov Random Field (MRF) . . . . .                                    | 29        |
| 2.3.2    | Proposed work for reliability analysis using MRF approach . . . . .    | 32        |
| 2.3.2.1  | Combinational circuit design . . . . .                                 | 33        |
| 2.3.2.2  | Sequential circuit design . . . . .                                    | 36        |
| 2.3.2.3  | Reliability enhancement using redundancy . . . . .                     | 36        |
| 2.3.3    | Other Probabilistic Modeling Techniques . . . . .                      | 40        |
| 2.3.3.1  | Bayesian Network (BN) . . . . .  | 40        |
| 2.3.3.2  | Probability Transfer Matrix (PTM) . . . . .                            | 41        |
| 2.3.3.3  | Probability Decision Diagram (PDD) . . . . .                           | 43        |
| 2.3.4    | Models Evaluation . . . . .  | 44        |
| 2.4      | Conclusions . . . . .  | 47        |
| <b>3</b> | <b>Proposed Modeling and Synthesis Approaches</b>                      | <b>50</b> |
| 3.1      | Nanocell Molecular Memory Design Approach . . . . .                    | 51        |
| 3.1.1    | Nanocell Molecular Devices . . . . .                                   | 51        |
| 3.1.2    | Design and Verification of 1-bit Molecular Memory Cell . . . . .       | 52        |
| 3.2      | Omnipotent Training of Molecular Memory . . . . .                      | 54        |
| 3.2.1    | Experimental Setup . . . . .   | 58        |
| 3.3      | Mortal Training of Molecular Memory . . . . .                          | 60        |
| 3.3.1    | Experimental Setup . . . . .   | 62        |
| 3.3.2    | Design space exploration . . . . .                                     | 63        |
| 3.4      | Conclusions . . . . .  | 67        |



|          |   |            |
|----------|---|------------|
| <b>4</b> | <b>Probabilistic Analysis of Nanocell Molecular Memory</b>                | <b>70</b>  |
| 4.1      | Reliability Analysis of Nanocell in Spatial Domain . . . . .              | 72         |
| 4.1.1    | Bounds on reliability of a Nanocell . . . . .                             | 76         |
| 4.1.2    | Experimental setup for Reliability Prediction in Spatial Domain . . . . . | 78         |
| 4.2      | Reliability Analysis of Nanocell in Temporal Domain . . . . .             | 84         |
| 4.2.1    | Probabilistic Modeling of Nanocell . . . . .                              | 84         |
| 4.2.1.1  | Steady state probability . . . . .  | 88         |
| 4.2.1.2  | Total probability being in a given sub-state . . . . .                    | 91         |
| 4.2.2    | Lifetime Analysis of Nanocell . . . . .                                   | 93         |
| 4.2.2.1  | Expected Lifetime . . . . .   | 93         |
| 4.2.2.2  | Availability . . . . .  | 97         |
| 4.2.3    | Experimental setup for Reliability Analysis in Time Domain . . . . .      | 98         |
| 4.3      | Conclusions . . . . .   | 107        |
| <b>5</b> | <b>Conclusions and Future Work</b>  | <b>109</b> |
| 5.1      | Future Work . . . . .   | 110        |
|          | <b>Bibliography</b>   | <b>113</b> |



# List of Figures

|     |  |    |
|-----|--|----|
| 1.1 | Evolution of Extended CMOS: relationship between "More Moore", "More-than-Moore" and "Beyond CMOS" - Reproduced from [2]   | 2  |
| 1.2 | Taxonomy for nanoscale emerging information processing devices - Reproduced from [2]   | 2  |
| 1.3 | The hierarchical classification of emerging memory devices - Reproduced from [2]   | 3  |
| 2.1 | Structures of (a) oligo(phenylene-ethynylene)s (b) alkanes (c) phenylenes with thiol end groups  | 15 |
| 2.2 | Structure of oligo(phenylene-ethynylene) (OPE) molecule with different functional groups (a) Dithiolated OPE (b) nitro OPE (c) amino-nitro OPE   | 17 |
| 2.3 | (a) Nitro substituted Oligo molecule (OPE) (b) I-V curve obtained using device model. The black dots represent the ' <i>ON</i> ' state of the molecule. In ' <i>OFF</i> ' state, approximately zero current flows. | 23 |
| 2.4 | Current Voltage Characteristics of an example molecule behaving as NDR: Simulation results obtained using equation (2.3) and (2.4). Here, $V_p = 3.5V$ , $I_p = 1.5\mu A$ , $I_s = 1e - 10A$ and $n = 2$ .         | 24 |

|      |   |    |
|------|---|----|
| 2.5  | A cartoon of crossbar molecular device. Here, a molecule is present between each cross-section of orthogonally placed nanowires. . . .  | 25 |
| 2.6  | Experimental setup for crossbar molecular (a) OR gate (b) AND gate  | 27 |
| 2.7  | Simulation results for crossbar molecular OR and AND gate . . . .   | 27 |
| 2.8  | A 2:1 multiplexer based logic gate design (a) AND gate (b) XOR gate   | 33 |
| 2.9  | A 1-bit Full Adder with TMR at each mux-based logic gate . . . . .  | 35 |
| 2.10 | Simulation results: Probability of getting correct output for a NOR based SR flip flop . . . . .  | 37 |
| 2.11 | Simulation results: Probability of next state $Q_{n+1}$ for a 4-bit Serial-In-Serial-Out right shift register when $p(S_{in} = 0) = 0.9$ . . . . .  | 37 |
| 2.12 | Simulation results: Probability of obtaining logic '1' on Sum(3) for a 4-bit ripple carry adder for varying defect rate at each simulation run. Total test cases are 50 and simulation is done for (a) No redundancy (b) TMR at each 1-bit adder (c) TMR at each logic gate . . . | 38 |
| 2.13 | Simulation results: Probability of logic '1' at $Sum(0)$ , $Sum(1)$ , $Sum(2)$ , $Sum(3)$ and $Cout$ with increasing fault rate and varying levels of modular redundancy. Plots are for cases: (i) no redundancy, (ii) TMR (iii) 5MR (iv) 7MR . . . . .                           | 39 |
| 3.1  | Nanocell based 1 - bit molecular memory cell (a) Schematic (b) Simulation results for Address (Ad), Write/Read (W/R), Data_In (In) and Data_Out (Out) terminals. Here, 2V and 0.5V represent logic '1' and logic '0', respectively. . . . .                                       | 53 |

|     |  |    |
|-----|--|----|
| 3.2 | Simulation results for 1000 samples of 1-bit memory cell. The left and right y-axes represents Data_Out voltage for Read 1 and Read 0, respectively . . . . .  | 54 |
| 3.3 | Nanocell based 2 - bit molecular memory cell: Simulation results for Address 1 (A1), Address 2 (A1), Write/Read (W/R), Data_In (In) and Data_Out (Out) terminals. Here, 2V and 0.5V represent logic '1' and logic '0', respectively. . . . .   | 59 |
| 3.4 | Schematic to design multi-bit memory by using multiple n-bit Memory Cells (represented by green square box). Each Memory Cell consists of $N$ nanoparticles (red circles) and $M$ molecular switches (black arrows). [Note: Not drawn to scale] . . . . .  | 60 |
| 3.5 | Schematic of proposed mortally trained 2-bit memory device consisting of 20 control signals [Note: Not drawn to scale] . . . . .   | 63 |
| 4.1 | A small network of nanoparticles and molecular switches (a) 3-particles, 3-switches, 2-paths between node $A$ to node $C$ (b) multiple particles and switches, multiple paths between node $A$ to node $F$ . . . . .   | 74 |
| 4.2 | Simulation results for samples generated using NRPA algorithm (a) Probability of existence of at least one minimal path in a nanocell ( $P_{path}$ ) (b) Number of molecules ( $M$ ) in a nanocell. Here, (i) the central mark is the median (ii) the edges of the box are 25 <sup>th</sup> and 75 <sup>th</sup> percentiles (iii) the whisker extend to the most data points which are not considered as outliers (iv) the unfilled dots represent the outliers . . . . . | 81 |

4.3 Simulation results for samples generated using NRPA algorithm.  
 (a) Upper bound on probability  $P_{path}$  (b) Lower bound on probability  $P_{path}$ . Here, (i) the central red mark is the median (ii) the edges of the box are 25<sup>th</sup> and 75<sup>th</sup> percentiles (iii) the whisker extend to the most data points which are not considered as outliers . . . . . 83

4.4 Continuous time birth - death model for Nanocell (a) abstract model with only super-states (b) detailed model with sub-states (c) Here, each circle represents a sub-state and a set of sub-states at each level combine to form a super-state, which is represented by an ellipse (d) bidirectional arrows between these states represents two unidirectional arrows, one for failure and another for repair. . . . . 85

4.5 An example nanocell consisting of four nanoparticles and five molecular switches. Here, the unfilled (white) circles represent the nanoparticles and the filled (black) circles with arrows represent the molecular switches. The direction of the arrows denotes the current flow. The nanoparticles A and D are the input and output nodes, respectively. . . . . 96

4.6 Upper bound on reliability ( $R_{UB}$ ), from  $t = 0$  to  $t = 10000$  units, for the example nanocell . . . . . 96

4.7 Representation of the example nanocell in the proposed model. Here, the circles represent the sub-states and the set  $\{i, j, \dots\}$  corresponding to each sub-state, represents the set of 'ON' molecules in that state. The filled sub-states are the *nanocell - failure - states* and remaining sub-states are the *nanocell - functioning - states*. The patterned filled circles represent one of the possible sequence in which the nanocell may make transition. . . . . 99

|      |   |     |
|------|---|-----|
| 4.8  | Simulation results for the sequence of sub-states of the example nanocell with constant $\lambda$ and $\mu$ in each case. The $p_f$ is exponentially distributed and it is explored for different values between 0 to 1. One thousand Monte Carlo simulations are done for each case. Here, (i) the central mark is the median (ii) the edges of the box are 25 <sup>th</sup> and 75 <sup>th</sup> percentiles (iii) the whisker extend to the most data points which are not considered as outliers (iv) the unfilled dots represent the outliers. . . . . | 101 |
| 4.9  | Simulation results for <i>success_ratio</i> for different values of failure probability ( $p_f$ ). . . . .  | 104 |
| 4.10 | Simulation results for example nanocell for 10 different cases. For each case, simulations are done for 10000 times, with varying $\lambda$ and $\mu$ . . . . .   | 106 |





# List of Tables

|     |   |    |
|-----|---|----|
| 2.1 | Comparison of properties and applications of (a) Alkanes and (b) OPEs . . . . .   | 16 |
| 2.2 | Summary of Model parameters . . . . .   | 22 |
| 2.3 | Logic compatibility table for 2:1 mux-based AND gate . . . . .  | 34 |
| 2.4 | Logic compatibility table for 2:1 mux-based XOR gate . . . . .  | 34 |
| 2.5 | Probabilities of getting logic '1' at sum output for mux-based and non-mux based 8-bit ripple carry adder with input $A = \text{"11101010"}$ and $B = \text{"11011111"}$ and $C_{in} = '1'$ . . . . . | 35 |
| 2.6 | Probability of sum and carry nodes of a half adder modeled by PDD   | 44 |
| 2.7 | Probability of <i>sum</i> of 8-bit ripple carry adder using MRF when inputs are $A = \text{"11101010"}$ and $B = \text{"11011111"}$ and $C_{in} = '0'$ . . . . .                                      | 45 |
| 2.8 | Probability of obtaining logic '1' ( $P_{11}$ ) on output sum of 8-bit ripple carry adder using PTM (assuming input probabilities = 0.5 and gate probability = 0.9) . . . . .                         | 45 |
| 2.9 | Probability Decision Diagram: results for example circuit in Figure 9 of [3] . . . . .  | 46 |

|      |   |     |
|------|---|-----|
| 2.10 | Comparison of probabilistic modeling approaches based on run time [3–5] . . . . .   | 46  |
| 2.11 | Comparison of probabilistic modeling approaches [6] . . . . .   | 48  |
| 3.1  | Number of successfully trained nanocell configuration for 2-bit memory read and write operation. Here, noise margin for Data read operation on output voltage node is considered as $0.4V$ . . . . .  | 66  |
| 4.1  | Simulation results for an example nanocell configuration when none or some of the molecules are missing . . . . .   | 74  |
| 4.2  | Simulation results for path probability ( $P_{path}$ ) and number of molecules ( $M$ ) w.r.t. number of nanoparticles ( $N$ ) in a nanocell . . . . .   | 82  |
| 4.3  | Simulation results for the example nanocell for different values of failure probability $p_f$ . The <i>success_ratio</i> represents the ratio of number of successful read '1' and '0' out of 10000 simulations. Upper and lower bounds on reliability are represented by $R_{UB}$ and $R_{LB}$ , respectively. . . . . | 105 |

# List of Algorithms

|   |   |     |
|---|---|-----|
| 1 | Omnipotent training of n-bit memory using Genetic Algorithm . . . .                                     | 56  |
| 2 | Mortal training of n-bit memory using Genetic Algorithm . . . . .                                       | 61  |
| 3 | Mortal training of n-bit molecular memory . . . . .   | 64  |
| 4 | Nanocell Reliability Prediction Algorithm (NRPA) . . . . .  | 79  |
| 5 | Extended Continuous Parameter Birth Death Model Generation for<br>Nanocell (Model_Generation) . . . . . | 102 |
| 6 | Reliability Evaluation Algorithm for Nanocell (REAN) . . . . .  | 103 |



# List of Abbreviations

|             |   |
|-------------|---|
| <i>BN</i>   | Bayesian Network                              |
| <i>CTMR</i> | Cascaded Triple Modular Redundancy            |
| <i>CVS</i>  | Control Voltage Signal                        |
| <i>GA</i>   | Genetic Algorithm                             |
| <i>GNPs</i> | Gold Nanoparticles                            |
| <i>GRF</i>  | Gibbs Random Field                            |
| <i>MRF</i>  | Markov Random Field                           |
| <i>NMR</i>  | N Modular Redundancy                          |
| <i>NRPA</i> | Nanocell Reliability Prediction Algorithm     |
| <i>OPE</i>  | Oligo(Phynylene Ethynylene) molecules         |
| <i>PDD</i>  | Probability Decision Diagram                  |
| <i>PTM</i>  | Probability Transfer Matrix                   |
| <i>REAN</i> | Reliability Evaluation Algorithm for Nanocell |
| <i>TMR</i>  | Triple Modular Redundancy                     |



# List of Symbols

|            |  |
|------------|--|
| $N$        | Number of nanoparticles in a nanocell  |
| $M$        | Number of molecular switches in a nanocell                                       |
| $IP$       | Input node of nanocell   |
| $OP$       | Output node of nanocell  |
| $Ad$       | Address bit  |
| $W/R$      | Write/Read bit   |
| $In$       | Data_In  |
| $Out$      | Data_Out   |
| $w_1(w_0)$ | Write '1' ('0') operation  |
| $r_1(r_0)$ | Read '1' ('0') operation   |
| $\mu$      | Mean of Gaussian Distribution  |
| $\sigma$   | Variance of Gaussian Distribution  |
| $X_i$      | Random variable $i$ , representing 'ON' or OFF state of a molecule               |
| $p_i$      | Probability that molecular switch $i$ is present in 'ON' state in a nanocell     |
| $R(p_i)$   | Reliability of molecule $i$ to be present in 'ON' state, at some instant of time |
| $m_i$      | molecule $i$   |
| $m$        | a subset of molecules out of $M$   |





---

|             |  |
|-------------|--|
| $P_{path}$  | Path probability or probability that at least one path is present between input and output of nanocell |
| $l_i$       | Length of path $i$   |
| $x_{ji}$    | Indicator variable that denotes state of molecule $j$ on path $i$                                      |
| $\phi(x)_i$ | Structure function for path $i$  |
| $\phi(X)$   | Structure function of complete nanocell  |
| $R(p)$      | Reliability of nanocell at some instant of time  |
| $path_i$    | Minimal path $i$   |
| $E_i$       | At least one molecular connection on path $path_i$ has failed  |
| $cut_i$     | Minimal cut set $i$  |
| $C_i$       | At least one molecular device in $cut_i$ is functioning  |
| $P_n$       | Probability of a random graph with $n$ nodes to be connected   |
| $\chi(t)$   | Number of 'ON' molecules at time $t$   |
| $S_j$       | Super-State, denoting $j$ molecules of nanocell are in 'ON' state                                      |
| $S_j^k$     | Sub-state of super-state $S_j$   |
| $I$         | State space of birth death process   |
| $T$         | Parameter space of birth death process   |
| $q_j(t)$    | Non-negative continuous function defined as rate of being in same state $j$                            |
| $q_{jk}(t)$ | Rate of transition from state $j$ to $k$   |
| $p_{jj}$    | Conditional probability of being in same state   |
| $p_{jk}$    | Conditional transition probability from state $j$ to $k$   |
| $o(h)$      | Probability that two or more transitions occur in time $(t, t + h]$                                    |
| $\lambda$   | Failure rate of molecules  |
| $\mu$       | Repair rate of molecules   |

---

|                    |  |
|--------------------|--|
| $P_j(t+h)$         | Total probability that the nanocell is in sub-state $S_j^k$ at time $(t+h]$  |
| $p_k$              | Steady-state probability that nanocell is in state $k$                       |
| $\rho$             | Ratio of failure and repair rate   |
| $E[\chi]$          | Mean number of molecules present in nanocell                                 |
| $VAR[\chi]$        | Variance of number of molecules in nanocell                                  |
| $F_i(t)$           | Lifetime distribution of molecule $mi$                                       |
| $P_0^i(t)$         | Probability that molecule $mi$ is in state $S_0^1$ by time $t$               |
| $R(\bar{F}(t))$    | Reliability of nanocell  |
| $E(Nanocell)$      | Expected lifetime of a nanocell  |
| $R_{UB}$           | Upper bound on reliability of nanocell                                       |
| $R_{LB}$           | Lower bound on reliability of nanocell                                       |
| $E_{UB}(Nanocell)$ | Upper bound on expected lifetime of nanocell                                 |
| $E_{LB}(Nanocell)$ | Lower bound on expected lifetime of nanocell                                 |
| $A(t)$             | Availability of nanocell or probability that nanocell is working at time $t$ |
| $A_i(t)$           | Probability that molecule $mi$ is 'ON' at time $t$                           |
| $A_{UB}(t)$        | Upper bound on availability of nanocell at time $t$                          |
| $A_{LB}(t)$        | Lower bound on availability of nanocell at time $t$                          |
| $A_{UB}$           | Upper bound on steady state availability of nanocell                         |
| $A_{LB}$           | Lower bound on steady state availability of nanocell                         |

# Chapter 1

## Introduction

Urge for high performance and more functionality in modern designs, drive the semiconductor technology to increasingly smaller dimensions. Continued dimensional and functional scaling of CMOS aims to build a smaller, cheaper as well as faster transistor that consumes less power and thus, results in increased chip density. Also, integrated circuits demand high speed information processing. For high processing speed, high density is required and it is achieved by reducing the device size. Eventually, dimensional scaling will approach the fundamental limits in near future. Henceforth, intense research is going on to replace the present CMOS devices by new devices for data storage and information processing. New emerging technologies are being explored for heterogeneous integration of multiple functions, which is termed as "More-than-Moore". Also, as shown in Figure 1.1 [2], the new emerging information processing devices as well as architectures are known as "Beyond CMOS" technologies. The heterogeneous integration of "Beyond CMOS" and "More-than-Moore" into "More Moore" will extend the CMOS platform functionality to form ultimate "Extended CMOS" [2]. As predicted by technical report on Emerging Research Devices (ERD 2013) [2], the two-dimensional scaling of SRAM and FLASH memories will reach definite limits within the next few years.

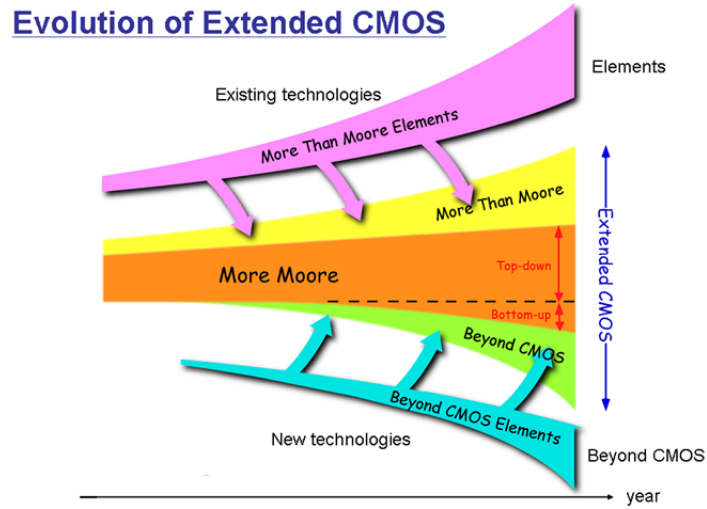


Figure 1.1: Evolution of Extended CMOS: relationship between "More Moore", "More-than-Moore" and "Beyond CMOS" - Reproduced from [2]

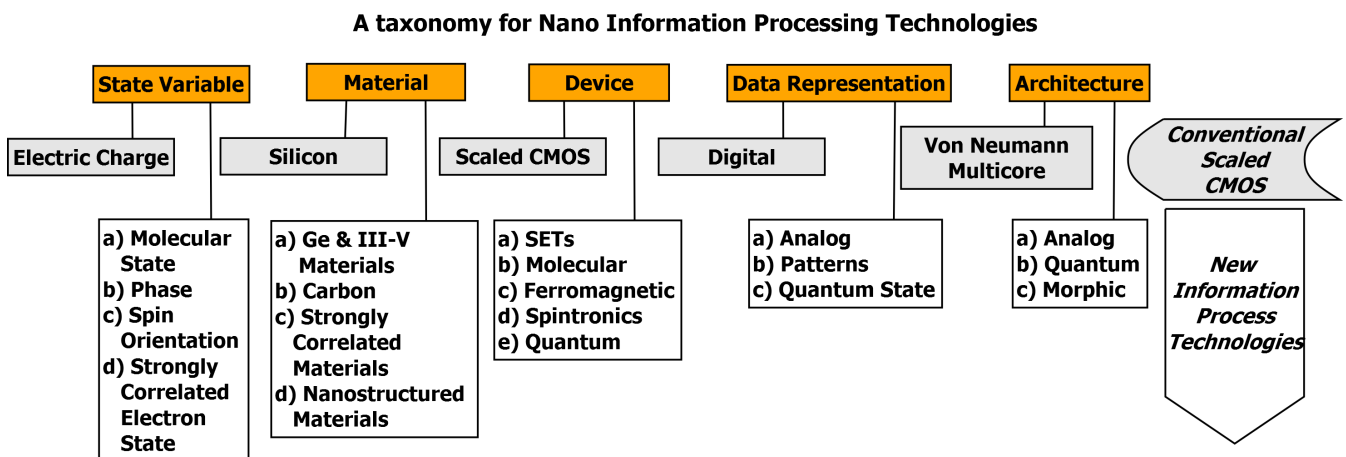


Figure 1.2: Taxonomy for nanoscale emerging information processing devices - Reproduced from [2]

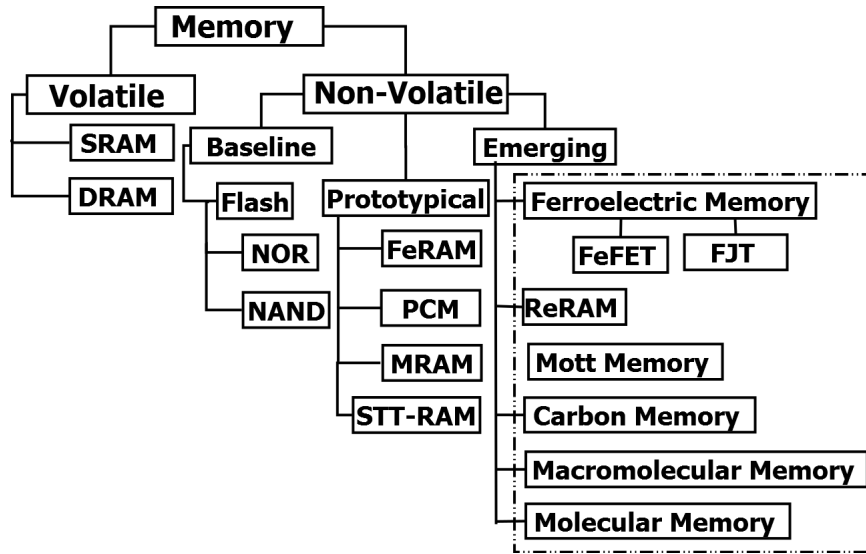


Figure 1.3: The hierarchical classification of emerging memory devices - Reproduced from [2]

These limits are compelling to expedite the development of new memory technologies as well as to replace CMOS based SRAM and FLASH memories in next few years. Henceforth, a technology breakthrough is required in order to identify the most promising alternative to present semiconductor memories. The future memory devices are expected to be electrically accessible with precision, consume low power, must have high density and speed, can be either volatile or non-volatile.

## 1.1 Taxonomy of Emerging Memory Devices

The Figure 1.2 [2] depicts the taxonomy of new nanoscale emerging information processing devices. Further, as shown in Figure 1.3 [2], Ferroelectric, Carbon, Mott, Macromolecular, Molecular Memories, etc. are some of the "Beyond CMOS" memory technologies which are being explored for future memory devices. Among these, Ferroelectric RAM (FeRAM) is a non-volatile, low power memory with fast write performance ( $< 3ns$ ) and large endurance ( $> 10^8$ ). However, FeRAM has

lower density and it is sensitive to contamination. When a field is applied to a crystal of Perovskites material in the required plane, the atom in its center will move in the direction of the plane. The position of the atom determines the state of the material. The operation of Nano-Electro-Mechanical (NEM) memory devices is based on the binary position of a nano-size mechanical beam to close or open an electrical contact, thereby completing or opening a circuit path [2]. Low energy dissipation, near zero leakage current very low subthreshold slope, etc. are some of the advantages of NEM memory switch. However, there are lots of obstacles in scaling FeRAM and NEM switch below  $16\text{nm}$ . Magnetic RAMs are non-volatile, high speed, ultra dense memories which can be easily integrated to backend CMOS process.

MRAMs make use of electron spin to store data. Data are written by small electrical currents in the magnetic write lines that create magnetic fields, which flips electron spins in the spin dependent tunnel junction storage layer, thus changing the junction resistance. Data is read by tunneling current or resistance through tunnel junction. MRAMs are non-volatile, high speed, ultra dense memories which can be easily integrated to backend CMOS process. However, they require 0.8-2.0 nm thick 10-12 different layers deposited by physical vapour deposition process and thus it is difficult to scale them beyond CMOS.

Operation mechanism of Macro-molecular or polymer memories is not known in detail. Memory effect originates from oxide while polymer acts as current limiter. These memories can be easily processable by printing, coating, etc., have flexible substrate and are disposable. Switching time is less than  $1\mu\text{s}$  and data retention time of more than 1 month has been reported.  $\text{Al}_2\text{O}_3$ -Polymer memory cell is approx. 100nm in size. But they suffer from low yield and high variability in operating conditions. Also, these are sensitive to oxygen and have low on/off ratio ( approx.

10).

In molecular memory, the one bit of information is stored into the molecule. When an external voltage is applied to the molecule, a conformational change occurs and it causes the molecule to transit from high conduction state to low or vice-versa. Data is read by measuring the change in resistance of the molecule. Thus, one bit of data can be stored in the space of a single molecule and thus, extremely high density memories can be obtained. The molecular switches of same type are expected to have identical device characteristics. This will reduce the component's variability problem. Emerging molecular crossbar technology offers high density, regular array-like and non-volatile memory structure [7–13]. These devices consume low power, offer low programming voltage and high switching speed. Non-volatility feature provided by these molecular devices, permits memory to be used as programmable elements within a logic device. The bottom-up approach is used for fabrication of nano-scale devices and it lacks precision in molecular device ordering. Thus, such crossbar molecular devices are inherently defective.

## **1.2 Historical Perspective: Molecular Electronics**

In late 1950s, Arthur R. von Hippel and his research group at MIT proposed to grow single molecules which may function as an electronic circuit or a component and thus miniaturize the electronic circuits [14]. Ease of fabrication, fast switching speed and smaller size were expected to be key advantages of molecular circuits over silicon integrated circuits. Thus, funded research projects began to develop solid state "molecular" circuits from doped inorganic crystals. Researchers at Westinghouse promised to grow a crystal of germanium that would behave as a complex circuit. However, failure to deliver such a device and continual success of Silicon ICs led to disappearance of the first wave of molecular electronics.

In 1974, Arieh Aviram and Mark Ratner proposed a single molecule rectifier [15,16] using organic charge transfer salts. The proposed molecular rectifier was similar to semiconductor PN-junction diode. This molecular rectifier consists of a donor and an acceptor group which is separated by a tunneling bridge. Once connected between similar metal leads, the difference in electron affinity between the two ends of the molecule was expected to result in asymmetric current transport. Forrest Carter, a chemist at U.S. Navel Research Laboratory took the idea and generated wide publicity for potential of molecular electronics. He brought a diverse set of chemists, physicists, electrical engineers, funding agencies, etc. under a single umbrella. However, all such efforts were in vain as no one managed to synthesize even a single molecular diode or a transistor. Carter died in 1987. Meanwhile, continuing his research, Aviram started organizing a series of conferences on molecular electronics.

In 1991, Mark Reed, a specialist in microfabrication from Yale met James M. Tour, a synthetic chemist from University of South California, at one of such conferences. Combining their ideas, Reed and Tour submitted a research proposal to Jan "Xan" Alexander, a grant officer at the Defense Advanced Research Projects Agency (DARPA). DARPA also funded similarly directed groups at Hawlett-Packard, IBM, Northwestern, Penn State, etc. Later, Reed and Tour with several other colleagues cofounded Molecular Electronics Corp. and even filed a patent for a "molecular computer" in 2000. They contributed several research publications [1, 17–22]. Reproducibility of similar current transport characteristic and synthesis of similar molecular circuits was the biggest challenge. Hence, many researchers faced criticism for their work. In this context, Paul Weiss [21], one of the Tour's collaborator at Penn State, told the journal *Science* that some of the Reed and Tour's earlier results on molecular electronic properties were not as solid as [17] had implied. R.



Stanely Williams of Hewlett-Packard as well as James Heath and Fraser Stoddart of University of California also faced criticism for their work on fabrication of Cross-bar Molecular Memories using Rotaxane molecules [9, 11–13]. Moreover, a series of discoveries in molecular electronics and claim of synthesis of single-molecule transistor, in particular, by Jan H. Schon proved to be false. Schon's fraud was disastrous to many working in this field and most of the fundings were stopped.

Today, bulk ensembles of molecular electronics have made their way into commercial displays. Recent break through of single molecule light emitting diodes and carbon nanotube transistors coupled to silicon in a monolithic integrated circuit have been reported [14].

### **1.3 Motivation**

Experts predict that high defect density of nano-scale devices is due to quantum physical defects, reduced noise margin, fabrication defects, environmental factors, thermal perturbation, etc. The defects can occur at fabrication time, called structural faults or can be introduced later due to alpha particles, cosmic rays, radiation, crosstalk, noise, thermal perturbation, etc., called transient errors. The unreliable nature of the device requires some fault tolerant approaches to be introduced, to make the system reliable. The present day semiconductor devices have lower defect rates, i.e., between  $10^{-7}$  to  $10^{-9}$ . Thus, the reliability can be achieved by introducing defect tolerance in the circuit. Modern defect tolerant techniques are reconfiguration, spatial and temporal redundancy, Built In Self Test (BIST), Built In Self Repair (BISR), etc. Since, defect rate for nano-scale devices is expected to be in the range  $10^{-3}$  to  $10^{-7}$ , such traditional fault tolerant approaches will eventually fail or might be impractical, in near future. Problems in traditional fault tolerance

approaches are:

1. Fixed modular redundancy is not effective in presence of transient faults, as defect rate will vary with time. N Modular Redundancy (NMR), Cascaded Triple Modular Redundancy (CTMR) and NAND Multiplexing are impractical for high defect rates of Nano-scale devices [23].
2. The majority voting circuit used for NMR is generally assumed to be fault free, but this circuit will be faulty in presence of high defect density.
3. While reconfiguration requires least amount of redundancy but it cannot handle transient faults or runtime defects.
4. Using Built-In Self Test (BIST) for defect map generation and fault recovery can be time consuming with such a high defect rate and ultra high device density.
5. Again, circuitry required for BIST is generally CMOS based, as in literature [24, 25]. In these hybrid CMOS/Nano fault tolerant architecture, majority of the device area will be occupied by the BIST circuitry. Hence, overall device density will be reduced.
6. Built-In Self Repair (BISR) used for embedded memories, is not applicable in case of nano-scale memories as spare rows and columns can also be defective.

In this context, a nanocell molecular device [1, 20] provides the in-built defect tolerance, high density, post-fabrication programmability through mortal training.

Since, a molecule is the smallest component whose electrical properties can be engineered, it is easy to conclude that the ultimate integrated circuit will be constructed at the molecular level. This fact has been the driving force behind molecular electronics research from decades. However, these molecular devices are still viewed as

a long term research goal. To use these molecules in future electronic devices, it is necessary to understand the electronic properties, dynamics, and interactions of the molecules in pure and mixed monolayers. It is required to develop implementations tailored to these properties. These molecular devices are constructed by bottom-up chemical self assembly and exploiting the electrical properties of one or more molecules. For design, synthesis, and development of next generation molecular materials, improved understanding of the required structural and electronic properties of the molecular material are needed [2].

## 1.4 Scope of work

We concentrate our efforts on probabilistic modeling, post fabrication synthesis and analysis of molecular memory. One of the primary goal of this thesis is to develop modeling and post fabrication synthesis algorithm for nanocell based molecular memory device. Also, to analytically explore if such a memory can withstand environmental uncertainties. These objectives can be elaborated as:

- To develop a model of OPE molecule and use it to propose a nanocell model.
- To propose an algorithm to train a nanocell as 1-bit memory while assuming that we can switch the state of individual molecules (omnipotent assumption). In other words, aim of this experiment is to find an optimal configuration of nanoparticles and molecules within a randomly assembled nanocell such that it behaves as a n-bit memory.
- To find proof of the concept of mortal training which to our understanding has not been tried yet. In this case, the nanocell is assumed to be a black box to the training algorithm and voltage signals are applied externally to switch the state of the molecules.

- To extend this concept of single bit storage to multi-bit storage.
- To theoretically compute spatial and temporal reliability of nanocell based molecular memory.

A brief survey of organic molecules, which exhibit non linear transport characteristics, is presented in the thesis. We also discuss their properties, synthesis process and modeling approaches. In the thesis, the device model of nitro-substituted Oligo (Phenylene Ethynylene) (OPE) molecule or 2'-amino-4,4'-di(ethynylphenyl)-5'-nitro-1-benzenethiol [17–19] has been described using Verilog-A. This molecular device model is used to model crossbar and nanocell based molecular devices described in HSPICE. In earlier works [1], a nanocell device was omnipotently trained to behave as simple logic devices. By omnipotence, we mean to say that, the internal topology of the nanocell is priori known to us and we can easily switch the individual molecules to '*ON*' or '*OFF*' state. However, due to extremely small size of the molecules and the nanoparticles, it would be impossible to switch the conduction state of an individual molecule. Thus, to realize the nanocell devices it is necessary to develop an alternate method such as mortal training of nanocell. Mortal training, implies that, the nanocell is assumed to be a black box to the training algorithm and voltage signals are applied externally to switch the state of the molecules. Hence, the focus here is on (i) omnipotent, as well as (ii) mortal training of the nanocell in order to make it behave as a memory device with read, write and erase capabilities. Further, the review of various probabilistic modeling approaches for nanoelectronics devices is presented in the thesis. The computational framework for Markov Random Field (MRF), Probabilistic Transfer Matrices (PTM) and Probabilistic Decision Diagrams (PDD) is developed using MATLAB. The HUGIN Lite tool [26] is used for Bayesian networks (BN) based circuit design and analysis. Under these frameworks, each logic variable has finite, but random probability of being logic '*0*'

or '1' and/or a certain error probability is associated with each logic gate. A design library has been developed for modeling this probabilistic behavior of nanodevices. A computational framework is proposed to compute the probability of retrieving the stored data bits correctly, at the output terminal of the nanocell buffer. Further, a novel extension over the continuous parameter birth-death model is proposed to evaluate the reliability of a nanocell, in presence of transient errors. For this mathematical framework, the steady state probability and probability of being in each sub-state is computed. The proposed approach is extended to compute the expected lifetime and availability of the nanocell using the birth-death model of molecules and their spatial connectivity.

## **1.5 Contributions of the thesis**

This thesis investigates the aspects of designing a nanocell based molecular memory. A model has been developed on the basis of already proposed analytical framework for molecular devices. The model is based on circuit behavior of nitro-substituted Oligo (Phynylene Ethynylene) (OPE) molecule. This model subsequently forms basis to simulate nanocell based 1-bit molecular memory and is verified using HSPICE. Due to hysteresis characteristics of OPE molecule, it is observed that even an untrained nanocell behaves as 1-bit memory cell. The simulation results of a 1-bit memory are compared against results obtained using analytical probabilistic modeling approach for the nanocell based memory devices. The proposed nanocell molecular memory demonstrates read, write and erase capability.

The concept is further augmented by post fabrication synthesis of 2-bit molecular memory using external control signal voltages [27]. The post fabrication synthesis of a nanocell is similar to that of a Field Programmable Gate Array (FPGA). As in case of FPGA, first a Hardware Descriptive Language (HDL) code for a

desired functionality is written, simulated and synthesized (technology mapped). After placement and routing, this HDL code is converted to Electronic Data Interchange Format (EDIF) and it is downloaded to program the FPGA. In this way, the FPGA is programmed for a particular functionality. Similarly, the nanocell is fabricated in a laboratory and later it can be adapted anywhere (in field) for a given functionality by applying appropriate external Control Voltage Signals (CVS).

Most suitable high and low voltage values for these control signals is a design space search problem. This search is handled by Genetic Algorithm such that some of the molecules turn to '*ON*' or '*OFF*' state and the nanocell is programmed to behave as a 2-bit memory cell. It is observed that to successfully train a 2-bit molecular memory, the number of control signals should be more than approximately one-fourth of total number of nanoparticles. Our results demonstrate that the proposed methodology is versatile enough to train nanocell for multi-bit storage functionality.

A computational framework is proposed to compute the probability of retrieving the stored data bits correctly at the output terminal of the proposed nanocell [27]. This graphical model for the nanocell is simulated by systematically varying number of nanoparticles and molecular switches. It is observed that, the probability of existence of at least one path, from input to output, approaches close to unity with presence of at least 20 or more nanoparticles in the nanocell. An algorithm is proposed to (i) generate an instance of nanocell consisting of  $N$  nanoparticles and (ii) compute the probability that at least one path is present between input and output node of nanocell (iii) bounds on reliability of the nanocell.

Further, a novel extension over the continuous parameter birth-death model is also proposed to estimate the reliability of a nanocell, in presence of transient errors [28]. In this computational framework, the steady state probability and probability of being in each sub-state is computed. The proposed approach is augmented to theo-

retically determine the expected lifetime and availability of the nanocell using the birth-death model of molecules and their spatial connectivity. The lower and upper bounds for nanocell reliability are calculated. An algorithm is proposed to automatically generate the proposed model representation for a given nanocell and use it to estimate the *success\_ratio* as well as the nanocell reliability, while considering the uncertainties. It is observed that as long as, molecular failure rate is less than its repair rate, the nanocell functions correctly. Also, the device reliability increases with increase in the number of nanoparticles and molecules which is validated by theoretical formulation.

## 1.6 Organization of thesis

In Chapter 2, first we review the characteristics of organic molecules which are used as an active device in molecular nanoelectronics. This is followed by the synthesis and modeling of nitro-substituted Oligo (Phynylene Ethynylene) (OPE) molecules and gold nanoparticles, as proposed in literatures. Further, the crossbar molecular devices and their modeling approaches are reviewed. In the end of this chapter, we present the review of major proposed approaches, available in the literature, related to probabilistic modeling and analysis of the nano-scale devices.

In Chapter 3, a detailed methodology for nanocell molecular memory modeling and synthesis is presented. Then, we discuss the probabilistic analysis of the nanocell in spatial and temporal domain in Chpter 4. Conclusions are presented in Chapter 5, where possible extensions to our work are also enumerated.





# Chapter 2

## State of the Art

This chapter is broadly classified into three parts. First, the literature review on organic molecules working as an active device, change in their characteristics with various functional groups and mechanism to switch to high/low conduction states with externally applied voltage, synthesis process, etc. is presented in Section 2.1. Besides, a brief description of the molecular device modeling techniques, in particular, the Universal Device Model (UDM), is discussed. We have developed this device model using VerilogA. Then, it is used to model the behavior of molecular diodes as well as negative differential resistors, and their simulation results are illustrated in the thesis.

In Section 2.2, a brief literature review on crossbar molecular nanoelectronic devices is discussed. This is followed by the experimental setup for crossbar logic devices. In Section 2.3, we present the literature review on probabilistic modeling approaches for nano-scale devices. These approaches are evaluated and simulation results are presented in this section. These modeling approaches are Markov Random Field, Bayesian Networks, Probabilistic Transfer Matrices, Probabilistic Decision Diagrams. Finally, the conclusions are summarized in Section 2.4.

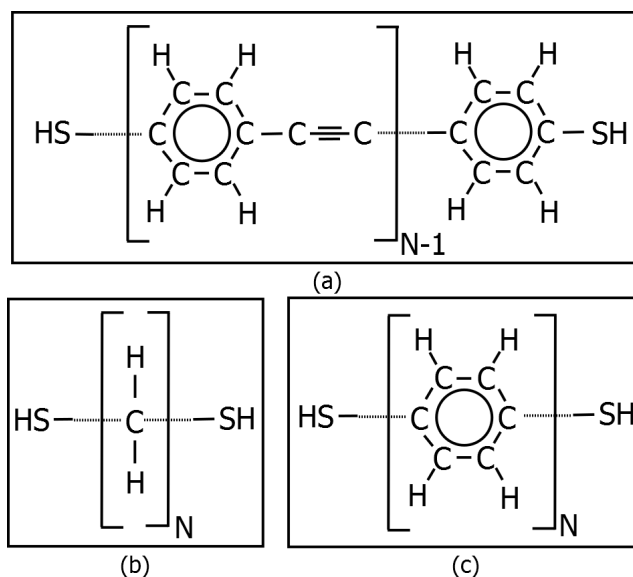


Figure 2.1: Structures of (a) oligo(phenylene-ethynylene)s (b) alkanes (c) phenylenes with thiol end groups

## 2.1 Organic Molecule: an active device

A molecule is the smallest component whose electrical properties can be engineered and thus it is expected that the ultimate integrated circuit will be constructed at the molecular level. This fact has been the driving force behind moletronics research [14]. In recent years, researchers have demonstrated the switching behavior of these molecules and have fabricated simple logic functions as well as memory by using such programmable molecules.

However, these molecular devices [29] are viewed as a long term research goal. It is required to properly understand the special properties of such molecules and develop implementations tailored to these properties. For design, synthesis, and development of next generation molecular materials, improved understanding of the required structural and electronic properties of the molecular material are needed [2]. In this context, characteristic properties of organic molecules functioning as an

active device are presented here.

### 2.1.1 Characteristics and Synthesis

Table 2.1: Comparison of properties and applications of (a) Alkanes and (b) OPEs

| Alkanes   | Oligo(Phenylene-Ethynylene)  |
|---|--|
| <b>Properties</b>   |  |
| Alkanes are $\sigma$ -bonded saturated molecules and are thus poor conductors with no obvious electronic device potential, other than as a tunneling barrier. | The extended $\pi$ -conjugation of OPEs make them theoretically much better conductors than alkanes.   |
| Comparatively less rigid as there is no triple bond in alkanes.   | The triple bond between the phenyl rings in OPE molecules enhances the coupling between them, thus making OPEs more conductive as well as rigid than oligo-phenyls of equivalent length. |
| <b>Applications</b>   |  |
| Alkanes can also be commercially obtained in a variety of lengths and suitable for electron transport measurements.   | The conductivity and structural rigidity makes OPE a potential molecular wire.   |
| The low conductivity of alkanes has been exploited as a gate dielectric in the fabrication of a hybrid (organic/Si) field-effect transistor.                  | OPEs have been reported to exhibit switching, negative differential resistance (NDR), and memory (hysteresis) effects.   |

The characteristic properties of the organic molecules depend on their chemical structure and functional groups attached. It is observed that saturated molecules like alkanes have low conductivity [30]. This is due to  $\sigma$ -bonding and hence these molecules can be used for gate dielectric in fabrication of a hybrid (organic/Si) field-effect transistor. The alkanethiolates (Fig. 2.1(b)) can be commercially synthesized in a variety of lengths. However, the extended  $\pi$ -conjugation of Oligo(Phenylene Ethynylene)s (OPEs) (Fig. 2.1(a)) make them theoretically much better conductors than alkanes. Table 2.1 gives a brief comparison of Alkanes and OPEs. Theoretical work shows that the triple bond between the phenyl rings in OPE molecules enhances the coupling between them. In this way, OPE molecules become more

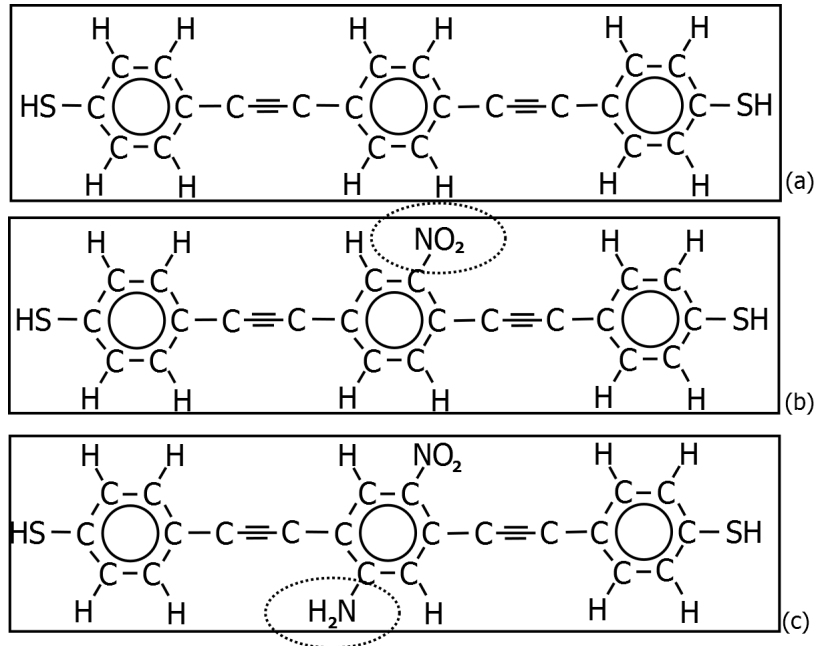


Figure 2.2: Structure of oligo(phenylene-ethynylene) (OPE) molecule with different functional groups (a) Dithiolated OPE (b) nitro OPE (c) amino-nitro OPE

conductive than oligo-phenyls (Fig. 2.1(c)) of equivalent length. Due to its structural rigidity, OPEs can be used as molecular wire. As shown in Fig. 2.2, the OPEs can be synthesized to possess donor groups, acceptor groups or heterocyclic interiors and thus modify their electronic structure. OPEs have been reported to exhibit switching, negative differential resistance (NDR), and memory (hysteresis) effects. This is due to varying chemical properties of these functional groups, for example:

- Nitro ( $NO_2$ ) functional group is electron withdrawing (Fig. 2.2(b)).
- Amino ( $NH_2$ ) functional group is electron donating (Fig. 2.2(c)).
- Thiol ( $SH$ ) group is binds the molecule to transition metals such as gold ( $Au$ ) or palladium ( $Pd$ ).

There are two types of metal-molecule contacts, namely, physisorbed contact and chemisorbed contact. Physisorbed contacts involve only electrostatic (e.g. van der

Waals) forces between the molecule and the electrode. However, it is comparatively a weak contact. On the other hand, chemisorbed contacts involve a chemical bond between the molecule and the electrode, as is the case with self-assembled monolayers (SAMs). Chemisorbed contacts can be further categorized by the type of electrode and molecular alligator clip. The exact nature of chemisorbed contacts can have a large influence on the measured properties of the molecular device. There are two types of alligator clips: (i) isonitrile, and (ii) thiol end groups. Among these, "thiol-Au" is the de-facto standard contact. However, experimental results confirm that Pd provides a better contact than Au for both alligator clips. It is also observed that by increasing the S-Au bond distance, an exponential decay in conductance occurs.

Wold et al. [30] investigated that self assembled monolayer of Oligo phenylene thiulates shows linear current voltage transport characteristics. It is observed that for saturated and unsaturated molecules, the resistance increase exponentially with increase in molecular length, according to the relation:

$$R = R_0 \exp\{\beta s\} \quad (2.1)$$

where,  $R_0$  is an effective contact resistance,  $\beta$  is structure dependent factor that depends on bonding and functional group patterns in the molecules,  $s$  is inter-electrode separation defined by molecular length. It is observed that  $\beta$  is almost double for saturated molecules. This provides an experimental proof for lower conductivity of alkanes.

Chen et al. [17,18] reported that the amino-nitro oligo molecule, i.e., Oligo(phenylene ethynylene) (OPE) molecule (Fig. 2.2(c)) shows negative differential resistance (NDR) characteristics. This shows that nitroamine redox center is liable for negative dif-

ferential resistance characteristics of these molecular devices. The nitro substituted oligo molecule (Fig. 2.2(b)) also show the NDR behavior while it was found to be absent in oligo molecule with only amino ( $NH_2$ ) group. Thus, we can conclude that the nitro ( $NO_2$ ) group is liable for this current-voltage characteristics of the molecule. Also, it is investigated that increase in temperature causes decrease in peak - to- valley (PVR) ratio of the molecule. It is observed that for OPE molecules, the PVR is 1030 : 1 at 60 K and it reduces to 1.5 : 1 at 300 K. However, such devices are highly unstable and it is difficult to reproduce the similar characteristics at a given temperature. It is expected that in near future, technology will be developed to overcome this problem. Kiehl *et al.* [31] showed that a self-assembled monolayer of 2'-amino, 5'-nitro oligo(phenylene ethynylene) (An-OPE) deposited on an Au electrode coupled to a Hg electrode covered with a tetradecane-thiolate leads to a well defined and stable NDR at room temperature and bias voltage of 0.6 V. Reed *et al.* [19] demonstrated the storage of charge in a self assembled layer of OPE molecules. Such a device is reported to be operated as a random access memory (RAM) with bit retention time of  $> 10$  min.

Both alkanes and oligo molecules can be made functional for self assembly and they form well packed, ordered monolayers. Majumdar *et al.* [22] studied the electrical behavior of nitro substituted OPE molecule in pure and mixed monolayer. In mixed monolayer, the nitro OPE molecules were separated by dodecanethiol molecules. The hysteresis NDR characteristics is found to be present in case of pure monolayer of nitro OPE molecules. It is observed in [22] that the electrical switching with memory behavior is a phenomenon that only occurs when a large group of molecules is present in a pure monolayer. Also, the magnitude of molecular conduction varies with environmental factors, e.g. temperature, type of monolayers, etc.

As reported in literatures [21,22,32], it is argued that the conductance in molecules is due to the conformational change in molecular state. Kim et al. [32] investigated the mechanism for molecular conformational changes. It is reported that such changes cause the sweep rate dependent hysteresis in the NDR. The transition between the high and low conduction states is due to interaction between electrical field and molecular dipole moment of the middle benzene ring in amino-nitro substituted oligo (An-OPE) molecule. The experimental results show that during the voltage sweep cycles, the kinetic effect of the molecular amino-nitro OPE SAM is responsible for the the NDR characteristics of these molecules. As reported in [32], the rotation of the second phenyl ring of the amino-nitro OPE molecules is influenced by the forward voltage sweep and it results in a planar to twisted phase transition. However, backward voltage sweep influences the twisted to planar phase transition. This rotation of the second phenyl ring restrains the  $\pi$  orbital overlaps between phenyl rings. This results in the lower electrical conductance of the twisted phase compared to the planar phase.

Chen et al. [17, 18] reported the detailed synthesis process for OPE molecule with different functional and end groups. The synthesis of OPE molecule is out of the scope of this thesis. But the method proposed in [17, 18] can be utilized for large scale synthesis of nitro-substituted OPE molecule. A lot of literatures are available for synthesis of capped and uncapped gold nanoparticles of  $< 100$  nm diameter [33–38]. For using the gold nanoparticles in nanocell architecture, these are required to be capped with  $-SH$  or  $-COOH$  group. Since, the focus of this thesis is to model and theoretically analyze the nanocell molecular memory in presence of transient errors, we next discuss an empirical device model for OPE molecule.

### 2.1.2 Modeling Approaches

Rose et al. [39–42] proposed a Universal Device Model (UDM) based on the equations (2.2), (2.3), (2.4) and few other equations. This UDM model can be used to capture the Current Voltage characteristics of a molecule behaving as a linear or a non-linear diode or negative differential resistance. We have used these three equations to model the electrical characteristics of an OPE molecule in Verilog-A. We have modified the parameters of these equations to model the behavior of the OPE molecule, appropriately. Table 2.2 describes the meaning of the parameters used in these equations.

$$I_R(V) = V/R \quad (2.2)$$

$$I_D(V) = I_s \cdot \left\{ \exp\left(\frac{V}{nV_T}\right) - 1 \right\} \quad (2.3)$$

$$I_T(V) = I_p \cdot \exp\left\{ \frac{-(V - V_p)^2}{2\sigma_p^2 f_- + 2\sigma_n^2 f_+} \right\} \quad (2.4)$$

Here,

$$f_-(V) = \frac{1}{1 + \exp\left[\frac{(V - V_p)}{S_f}\right]} \quad \text{and} \quad f_+(V) = \frac{1}{1 + \exp\left[\frac{-(V - V_p)}{S_f}\right]}$$

The positive differential resistance (PDR) region and negative differential resistance (NDR) region typically have different slopes, which lead to asymmetry in the curve. This asymmetry can be included in the model by having two width parameters: one for the PDR region and the another for the NDR region. Since each of the width parameters ( $f_-$  and  $f_+$ ) is only relevant to one of the two regions, step functions are



Table 2.2: Summary of Model parameters

|                      |  |
|----------------------|--|
| Resistor             |  |
| R                    | Resistance                             |
| Diode                |  |
| $I_s$                | Saturation current                     |
| n                    | emission coefficient                   |
| $V_T$                | KT/q, symbols have their usual meaning |
| NDR                  |  |
| $V_p$                | Peak Voltage                           |
| $I_p$                | Peak Current                           |
| $\sigma_p, \sigma_n$ | Width parameter                        |
| $S_f$                | Step rise                              |

used to filter out unwanted contributions to the other region. Two of the parameters of equation 2.4 are the peak current  $I_p$  and the peak voltage  $V_p$ , which specify the point at which the PDR region transitions to the NDR region. The width of this Gaussian can be specified by using the parameters  $\sigma_p$  and  $\sigma_n$ , for the PDR and NDR regions, respectively. In general, the slope parameter should be as small as possible so that the functions exhibit sharp steps thus keeping the width parameters, and thus the PDR and NDR regions, independent of one another.  $S_f$  is associated with the slope of the rising edges of the step functions used to separate the width of the PDR region from that of the NDR region.

The nanocell based memory has been designed using the device model of gold nanoparticles and OPE molecular switches. The verilogA model of OPE molecule gives the IV characteristics as shown in Figure 2.5(b) which almost matches to actual molecule curve as in fig. 3 of ref. [1]. The NDR model of the molecule proposed in [43] shows the IV curve in fig.6 of [43] which is similar to our curve. The empirical model used here is general and can be used to fit any curve. By varying the parameters of equation (2.2), (2.3) and (2.4) the behavior of an example molecule has been depicted in Figure 2.4. Here, the assumed parameter values are:  $V_p = 3.5V$ ,

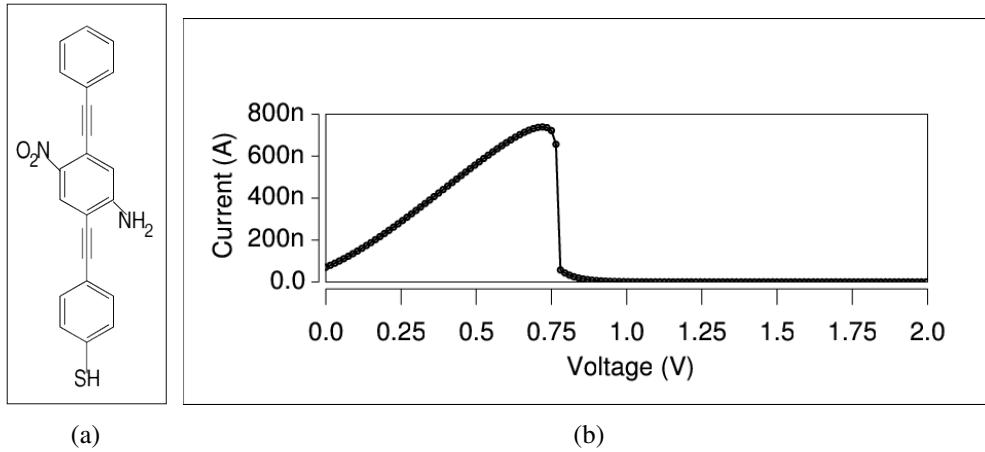


Figure 2.3: (a) Nitro substituted Oligo molecule (OPE) (b) I-V curve obtained using device model. The black dots represent the 'ON' state of the molecule. In 'OFF' state, approximately zero current flows.

$I_p = 1.5$ ,  $I_s = 1e - 10$  and  $n = 2$ . We can argue that by adjusting the parameter values, the desired electrical behavior of the molecule can be modeled.

Jha *et al.* [43] discusses omnipotent training of a nanocell to behave as logic gates. In their work, they have proposed a device model of OPE molecule in HSPICE. This model gets complicated when generalized for linear or non-linear electrical behavior of different organic molecules. Kim *et al.* [32] modeled the An-OPE molecule and validated this model with experimental results. The I-V performance of each conformation (Planar and Twisted) of the molecule is calculated by combining Green's function theory with DFT Hamiltonian that are determined from the SeqQuest calculation with Perdew-Burke-Ernzerhof (PBE) generalized gradient approximation exchange-correlation density functional. This quantum mechanical model explains how the such conformational changes are responsible for NDR behavior of OPE molecules. However, this type of molecular device model to be integrated with HSPICE model of nanocell. Barepour *et al.* [44] extended the UDM model to model the hysteresis effects. The proposed SPICE model is comprised of a differentiator circuit, two switches and two resistors/UDM modules followed by an integrator cir-

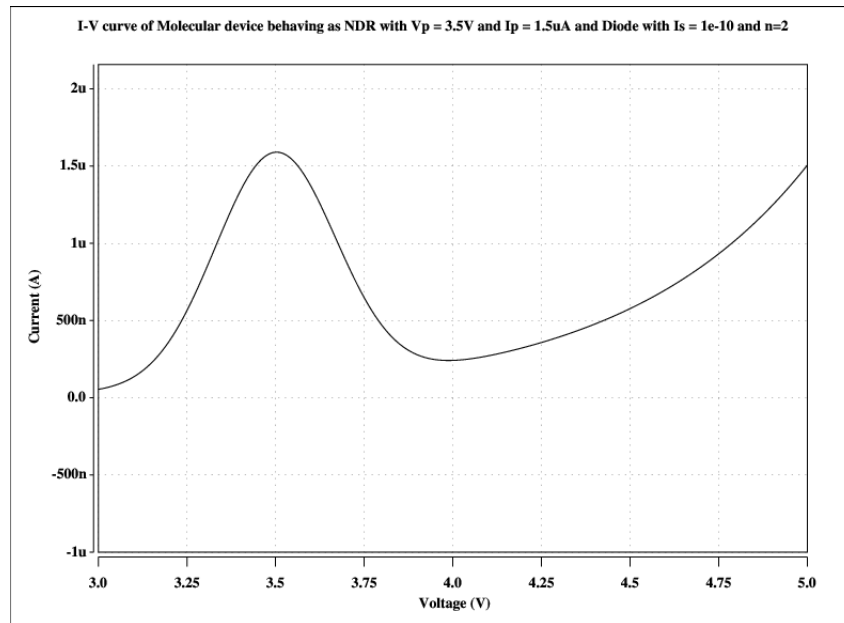


Figure 2.4: Current Voltage Characteristics of an example molecule behaving as NDR: Simulation results obtained using equation (2.3) and (2.4) . Here,  $V_p = 3.5V$ ,  $I_p = 1.5\mu A$ ,  $I_s = 1e - 10A$  and  $n = 2$ .

cuit.

## 2.2 Crossbar based Molecular Devices: design and modeling

In crossbar architecture [9,13,45–48], a self-assembled monolayer of bistable molecules is inserted between orthogonally placed nanowires, such that there is a molecule at cross-section of each of these nanowires. This is depicted from Figure 2.5. A Nanowire (NW) [7,8,49,50] is a long, thin semiconducting wire that can be used as both an active device as well as interconnect wires. Its diameter could be as small as few nanometer and length from tens to thousands of microns. Silicon, germanium, gold, gallium phosphide, gallium nitride, indium phosphide, etc. can be used to fabricate nanowires. The nanowires can be grown from hundred to thousands of

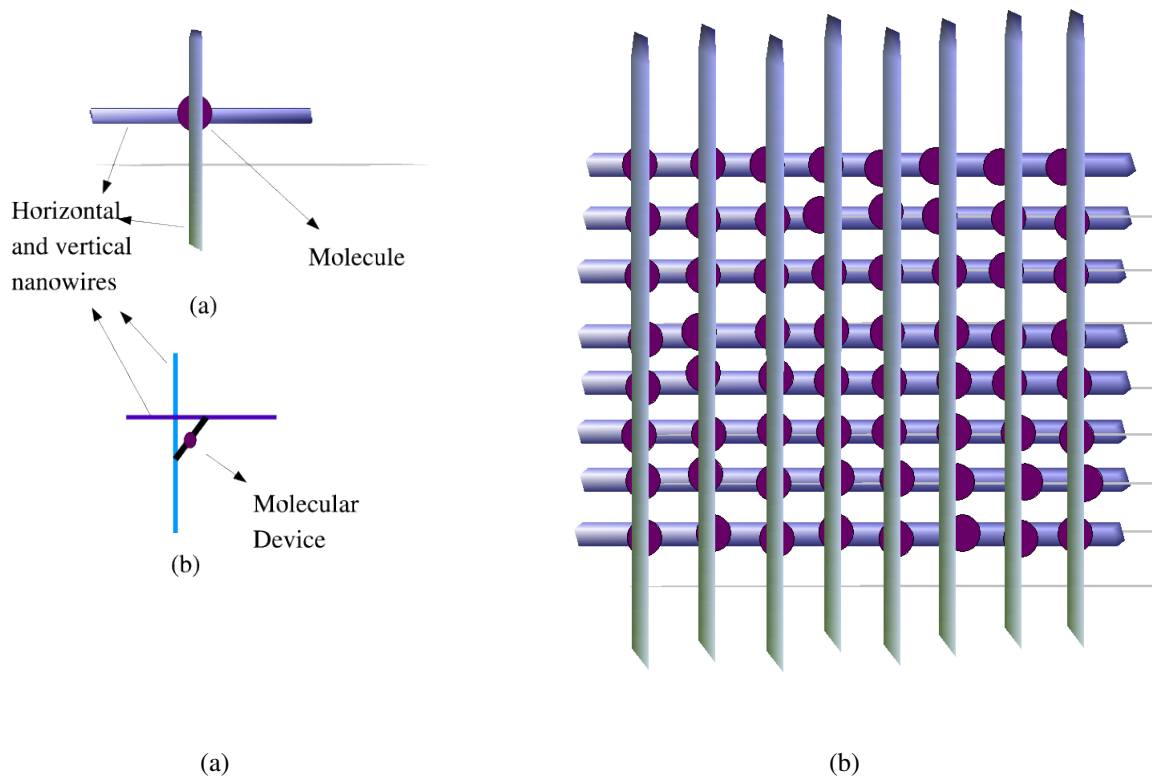


Figure 2.5: A cartoon of crossbar molecular device. Here, a molecule is present between each cross-section of orthogonally placed nanowires.

microns in length. As reported in [7, 8], the density greater than  $10^8 \text{ gateequiv}/\text{cm}^2$  can be achieved for crossbar devices. Recently, Guilera et. al. [51] have fabricated single-nanowire-based gas sensors, photodetectors, and field-effect transistors using metal-oxide nanowires.

Chen et al. [9] fabricated a molecular rewritable nonvolatile memory with density  $6.4 \text{ Gbits cm}^{-2}$ . Rotaxane molecules are sandwiched between bottom Ti(3 nm) / Pt(5 nm) and top Ti(11 nm) / Pt(5 nm) nanowires. Data retention time of 2months has been achieved in this case. The ‘ON’ resistance was roughly 500 KOhms and ‘OFF’ resistance 9 MOhms. Green et al. [12] have demonstrated a 160 kbit molecular memory offering 33 nm pitch with density  $10^{11} \text{ bits cm}^{-2}$ . This crossbar memory

array is comprised of a monolayer of Rotaxane molecule sandwiched between 400 x 400 Phosphorous doped Si bottom nanowires (16 nm) and Ti top nanowires (16 nm). Such crossbar molecular memories, based on Rotaxane molecules, offers cell size of approx.  $0.0011 \mu m^2$ . The best projected read time of  $< 10$  ns and write/erase time of  $< 40$  ns has been achieved [10]. Green however demonstrated write/erase time of 0.2 s. The voltage required for reading the data bit is 0.3 – 0.5 v [10, 11] and write operating voltage 1.5 v [11]. Such molecular memories are non-volatile, low power and ultra dense memories with high retention period [2]. We have simulated the crossbar devices in HSPICE, using the molecular device model.

Different read circuit are proposed in literatures [42, 46–48] for crossbar molecular memories. Here value of  $V_0$  varies, for eg. Chen et al. [9] used  $V_0 = V_{DD}/2$  while Csaba et al. [52] and Mustafa et al. [53] used  $V_0 = (V_{offThr} - V_{onThreshold})/4$  in their designs.

We have implemented the crossbar molecular logic gates in HSPICE using the device model of molecule (as discussed in previous section). The Figure 2.6 shows the experimental results for crossbar molecular OR gate and AND gate. This is similar to programmable logic arrays (PLAs). The simulation results for these logic gates are shown in Figure 2.7. Similar to the approach as proposed by [42], we have also tried to simulate the crossbar molecular memory. The ultra high density crossbar molecular memory modeling and synthesis is part of our future work.

A novel high functional density Graphene nanoribbon crossbar architecture is proposed and analytically simulated by [54, 55]. Similar to crossbar devices, the horizontal and vertical layers of this nanoribbon were assumed to have linear resistance while their cross-section were assumed to have NDR behavior. It is analytically demonstrated that the GNR-based crossbar circuits outperform conventional CMOS circuits in low power applications [55]. Vourkas et al. [56] demonstrated a

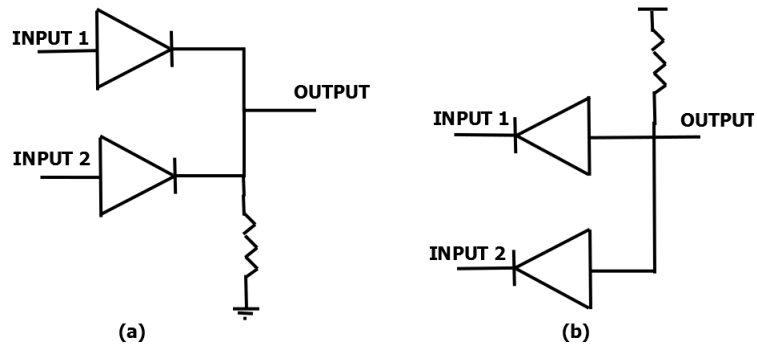


Figure 2.6: Experimental setup for crossbar molecular (a) OR gate (b) AND gate

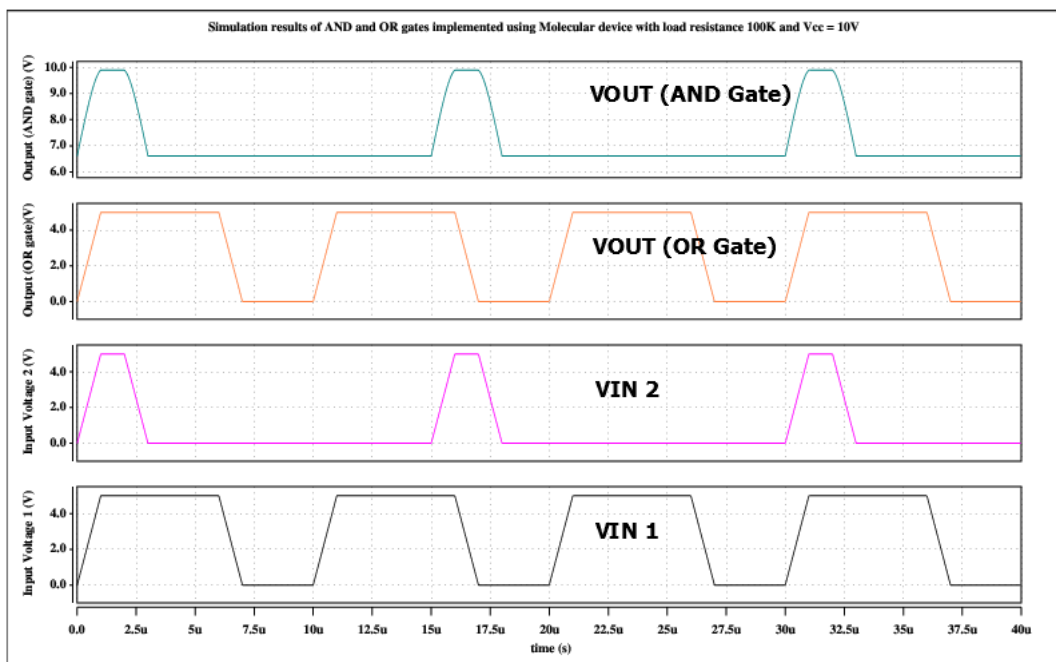


Figure 2.7: Simulation results for crossbar molecular OR and AND gate

memresistor (concatenation; of “memory resistor”) based crossbar framework with encoders and decoders as design example. Gholipour *et al.* [57] presents a brief review and comparison of different crossbar based architectures.

However, the crossbar memories suffer from high defects as it is difficult to chemically self-assemble the nanowires perfectly, with the available bottom-up approaches. Several redundancy based [23] and reconfiguration based defect tolerance techniques have been proposed to handle the soft and hard errors. However, at nanoscale, the defect density will be high and thus traditional fault tolerant techniques will fail. Intense research is going on to deal with defects in emerging technologies. Many defect mapping techniques have been proposed to identify the defective devices and replace them by spare devices. Dehon *et al.* [7] has proposed the similar approach for crossbar architectures. That is, to replace the defective rows or columns by spare rows and columns. Jeffery *et al.* [58] proposed error control coding techniques to provide defect tolerance in crossbar devices. However such techniques introduce large area overhead of defect tolerance circuitry. Recently Wang *et al.* [59] has proposed hybrid redundancy allocation technique for defect tolerant crossbar molecular memory systems. The proposed approach combines temporal and spatial redundancy. The simulation results demonstrate significant improvement in defect tolerance, efficiency and scalability of the proposed approach. Zamani *et al.* [60] has proposed variation and defect tolerant logic mapping on crossbar molecular architectures. The proposed ILP formulation can be used for both diode based and FET-based crossbars. A set of ILP formulations have been proposed for efficient logic mapping in order to minimize variation effects, tolerate defects, and increase reliability of a circuit implemented on crossbar nano-architectures. In short, the defect tolerance techniques are required to be improved or some other architecture is required to be proposed which does not need such precise molecular ordering.

One such architecture for molecular devices is proposed by Tour and it is named as nanocell. In this thesis, we concentrate on nanocell molecular memory modeling, synthesis and probabilistic analysis in presence of high defect rate. The nanocell modeling and synthesis is discussed in next chapter.

## 2.3 Probabilistic Modeling Approaches for Nano-scale Sub-Systems

### 2.3.1 Markov Random Field (MRF)

A Markov Random field (MRF) [61,62] is a Markov-type two dimensional process. It is defined as a set of random variables  $X = \{X_1, X_2, \dots, X_n\}$  satisfying Markov property on a rectangular lattice  $L$ . Given, the present state of a random variable and some of its past states, any stochastic random process has Markov property if conditional probability distribution of future states depends only on the present state and is independent of the past states. This definition of Markov property can be extended to MRF as a random variable  $X_i$  has Markov property if and only if it is conditionally dependent only on its neighbors.

Next, we define the Markov blanket (or neighborhood) of  $X_i$  as  $\eta_i$ , such that  $\eta_i \subset X$  and each  $X_k \in \eta_i$  is neighbor of  $X_i$ . The positivity condition for existence of MRF is stated in equation (2.5) and Markov property is given in equation (2.6).

$$P(X_i = x) > 0, \forall X_i \in X. \quad (2.5)$$

$$P(X_i = x | X - X_i) = P(X_i | \eta_i). \quad (2.6)$$

A random variable  $X_i \in X$  defined on  $\{L, \eta_i\}$  is called a Gibbs Random Field (GRF)



if and only if its joint probability mass function is of the form as in equation (2.7),

$$P(X_i = x) = \frac{1}{Z} e^{(-\frac{1}{K_B T} E(x))}, \quad (2.7)$$

where  $K_B$  is Boltzmann constant;  $T$  is temperature constant;  $Z$  is normalization constant, known as partition function;  $E(x) = \sum_{c \in C} U_c(x_i)$  is called clique energy function;  $c$  is a clique;  $C$  is set of all cliques of  $\{L, \eta\}$ ;  $U_c(x)$  is the potential associated with clique  $c$ . It is stated that  $X$  is a MRF with respect to  $\eta$  and  $P(X = x) > 0$  for all  $x$ , if and only if  $X$  is a GRF with respect to  $\eta$  and associated cliques. This is known as Hammersley and Clifford theorem, first stated and proved by Hammersley and Clifford in their unpublished work and later proved by Besag [63]. It is deduced from the equations (2.5), (2.6) and (2.7) that GRF implies MRF. As proved by Besag [63], the overall joint probability  $P(X = x)$  can be expressed in terms of conditional probabilities (or, local characteristics)  $P(X_i | \eta_i)$ , as given in equation (2.8).

$$P(X_1, X_2 \dots X_n) = \prod_{i=1 \dots n} P(X_i | \eta_i). \quad (2.8)$$

Markov Random Field based probabilistic computing for nanoelectronic devices is proposed by Bahar et al. [61]. Each logic variable of a digital circuit is represented as a node and statistical dependence between these variables is modeled as an edge of MRF Graph. A clique in an undirected graph  $G = (V, E)$  is a subset of the vertex set  $C \subseteq V$ , such that for every two vertices in  $C$ , there exists an edge connecting the two. This is equivalent to saying that the subgraph induced by  $C$  is complete. The clique energy expression  $E(x_0, x_1, x_2) = -\sum_c U_c(x_0, x_1, x_2)$ , where  $U_c = 1$ , is obtained by the negative sum over minterms from valid states in logic compatibility table, and these minterms are transformed using the Boolean ring rules. The energy

of invalid logic states is greater than that of valid states. For an  $n$ -input gate, the number of rows in logic compatibility table are  $2^{n+1}$  and number of columns are  $n + 2$ . Each variable has some random probability of occurrence of logic zero and logic one. At the primary output, joint probability is calculated using equation (2.8) and for each intermediate node marginal probability is calculated using the belief propagation algorithm [64, 65]. For example, for a two input XOR gate with input  $x_0 = '0'$ ,  $x_1 = '1'$  and output  $x_2 = '1'$  is called a Valid state. But, if output is  $x_2 = '0'$  for same inputs, then it is called an invalid state. The clique energy for two input XOR gate is computed as:

$$\begin{aligned}
 E(x) &= -(1-x_0)(1-x_1)(1-x_2) - (1-x_0)x_1x_2 - x_0(1-x_1)x_2 - x_0x_1(1-x_2) \\
 &= -1 + x_0 + x_1 + x_2 - 2x_0x_1 - 2x_1x_2 - 2x_2x_0 + 4x_0x_1x_2
 \end{aligned} \tag{2.9}$$

Bhaduri and Shukla [66, 67] designed NANOLAB toolbox in MATLAB, based on theoretical work proposed by Bahar *et al.* [61]. The NANOLAB can calculate node probability, clique energy and entropy in presence or absence of noise. It provides fault tolerant techniques like Triple Modular Redundancy (TMR), Cascaded Triple Modular Redundancy (CTMR), NAND multiplexing, etc. However, it lacks sequential circuit modeling. The NANOLAB tool is evaluated for structural faults and bridge faults in [68]. The results are compared with that of deterministic approach.

Nepal and Bahar [69–72] used the MRF approach to implement CMOS based devices at nanoscale. The MRF based logic gates at  $70 \text{ nm}$  are designed and device characteristics are compared with CMOS based logic gates, in presence of noise and soft errors in [69]. The reliability is thus achieved on the cost of size. For example, a four transistor NAND Gate implementation requires 60 transistors using

MRF approach, as suggested by Nepal [69]. Further, in [71, 72], the modified clique energy function is proposed and this new implementation of 2-input NAND Gate requires only 28 transistors. Also, as shown in Table 4 of [72], the power is greatly reduced (by 33%) for larger circuits using MRF mapping. Error correcting codes technique (Hamming code (6,3)) is introduced to enhance reliability at lower power with reduced device area as compared to previous works of Nepal et al. [69]. The feedback path in MRF, reinforces the logic values and the authors claim to fix all one-bit deviations from correct codeword by reinforcing principle of MRF circuit. However, the reconvergent fanouts induce delay to the circuit.

We have implemented Markov Random Field based approach using MATLAB for combinational and sequential circuits design. All 2-input logic gates are modeled using this approach. The NANOLAB toolbox is augmented with the flipflop library for designing sequential circuits [73]. In next Section, we discuss the experimental setup and simulation results for 2:1 multiplexer based combinational and sequential circuit design.

### **2.3.2 Proposed work for reliability analysis using MRF approach**

The NANOLAB tool is explored to design TMR and CTMR based logic gates. It is used to study the effects on device output in presence of noise. The NANOLAB tool is augmented to include 2:1 multiplexer as a basic building block. Thus, a design library is developed to model multiplexer based logic gates and for this necessary changes are made to the logic compatibility table. In order to enhance the system reliability, each multiplexer based logic gate is modeled by Triple Modular Redundancy (TMR). Various flip flops are modeled using logic gates and a design library is created. A 4-bit Serial-In-Serial-Out (SISO) right shift register is used as a vehicle to exemplify our approach. The combinational circuits designed using MRF based approach, are analyzed for fixed redundancy (TMR) at different levels of ab-

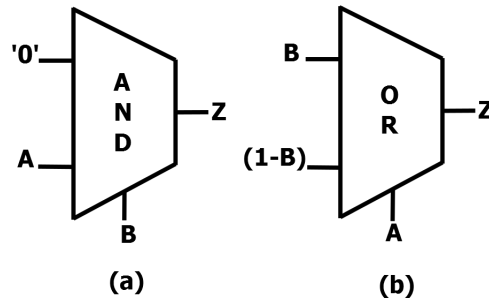


Figure 2.8: A 2:1 multiplexer based logic gate design (a) AND gate (b) XOR gate

straction while considering fixed defect rate. A 4-bit ripple carry adder having TMR at each Module ( $TMR_M$ ) and at each logic gate level ( $TMR_G$ ) are compared to an adder without redundancy. Secondly, with varying levels of redundancy ( $No\_MR$ ,  $TMR$ ,  $5MR$ ,  $7MR$ ,  $9MR$ ) the adder is analyzed for varying defect rates. In this case, hardware redundancy is applied to each full adder.

### 2.3.2.1 Combinational circuit design

We consider a design of an 8-bit ripple carry adder using mux-based logic gates. The TMR is provided at each logic gate level. As discussed earlier, necessary changes are made to logic compatibility table of gates to model them using a multiplexer. The logic compatibility table of 2:1 multiplexer based AND gate and XOR gate is given in Table 2.3 and Table 2.4, respectively. These multiplexer based logic gates are giving same equations for Gibbs Energy and same entropy values as in case of basic logic gates approach. All multiplexer based logic gates, except XOR gate, offer the advantage that one of their input lines can be directly connected to Vdd or Gnd. Hence, their logic compatibility table is reduced by half which decreases the search time for valid states during simulation. The 1-bit full adder, as shown in Figure 2.9, has triple modular redundancy at each logic gate level. Using this 1-bit full adder, an 8-bit ripple carry adder is designed and it is compared to an 8-bit ripple carry adder using simple logic gates. Both adders have TMR at each logic gate. The

Table 2.3: Logic compatibility table for 2:1 mux-based AND gate

| Input<br>0 | Input<br>A | Select<br>B | Output<br>Z | Valid/Invalid<br>(1/0) |
|------------|------------|-------------|-------------|------------------------|
| 0          | 0          | 0           | 0           | 1                      |
| 0          | 0          | 1           | 0           | 1                      |
| 0          | 1          | 0           | 0           | 1                      |
| 0          | 1          | 1           | 1           | 1                      |

Table 2.4: Logic compatibility table for 2:1 mux-based XOR gate

| Input<br>B | Input<br>1-B | Select<br>A | Output<br>Z | Valid/Invalid<br>(1/0) |
|------------|--------------|-------------|-------------|------------------------|
| 0          | 0            | 0           | 0           | 1                      |
| 0          | 0            | 1           | 0           | 0                      |
| 0          | 1            | 0           | 0           | 1                      |
| 0          | 1            | 1           | 1           | 0                      |
| 0          | 0            | 0           | 0           | 1                      |
| 0          | 0            | 1           | 0           | 0                      |
| 0          | 1            | 0           | 0           | 1                      |
| 0          | 1            | 1           | 1           | 0                      |

two adders are simulated for different input values. It is observed that for all input combinations, the probabilities of sum and carry values are same up to 2 places after decimal, for both the adders. As depicted from Table 2.5, the mux-based circuits show higher probabilities for logic '0'. When input bit-vector  $A = "11101010"$  and  $B = "11011111"$  with input carry  $C_{in} = '1'$ , the expected output sum is  $"11001001"$ . In such a case, the probabilities of sum for mux and non-mux based 8-bit ripple carry adders are given in Table 2.5. At input side, logic '1' is represented by probability vector  $[0.2 \ 0.8]$  and logic '0' by  $[0.8 \ 0.2]$ . The similar results can be obtained for

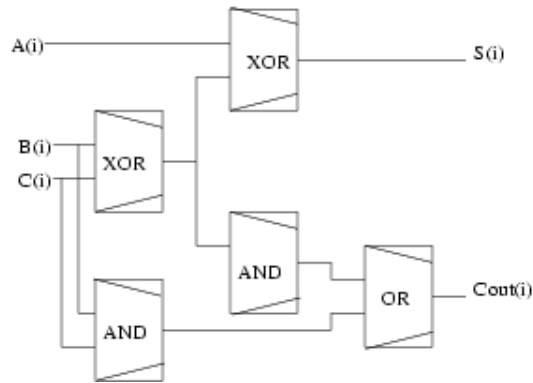


Figure 2.9: A 1-bit Full Adder with TMR at each mux-based logic gate

Table 2.5: Probabilities of getting logic '1' at sum output for mux-based and non-mux based 8-bit ripple carry adder with input  $A = "11101010"$  and  $B = "11011111"$  and  $C_{in} = '1'$

| Adder Output | Mux-based Adder | Non-Mux Based Adder | Logic value |
|--------------|-----------------|---------------------|-------------|
| s(7)         | 0.738954        | 0.739023            | 1           |
| s(6)         | 0.682778        | 0.682954            | 1           |
| s(5)         | 0.284839        | 0.284728            | 0           |
| s(4)         | 0.260738        | 0.260673            | 0           |
| s(3)         | 0.686478        | 0.686591            | 1           |
| s(2)         | 0.282181        | 0.282115            | 0           |
| s(1)         | 0.172751        | 0.172673            | 0           |
| s(0)         | 0.724985        | 0.724022            | 1           |

other circuits also.

### 2.3.2.2 Sequential circuit design

We designed a MATLAB library of all flip flops and latches based on MRF approach. The results of a NOR based SR flip flop are shown here as an example. The probabilities of Set and Reset are taken as an input and probability of next state output is generated. When  $p(\text{set} = 1) = 0.9$  and  $p(\text{reset} = 1) = 0.1$  the plot in Figure 2.10(a) is generated and next state probability for logic '1' is estimated as 0.8235 which can be accepted as logic '1'. Thus, flip flop is in set state. Similarly, Figure 2.10(b) shows reset conditions and next state probability for logic zero is 0.8353. When  $p(\text{set} = 1) = 0.1$  and  $p(\text{reset} = 1) = 0.1$  and present state probability for logic '0' as 0.9, the next state probability is calculated as 0.8353, as shown in Figure 2.10(c). This can be accepted as logic '0' and we get no change in state as desired by given input conditions. Now, these flip flops and latches can be used to design sequential circuits. We here discuss a 4-bit Serial-In-Serial-Out right shift register to exemplify our approach. A D flip flop is used as a memory element in the design of SISO. We assume that there is 90% probability of occurrence of logic '0' at the input  $S_{in}$ , then this data is shifted out serially and obtained at  $S_{out}$  after a delay of four flip flops. The plots generated at each flip flop from  $FF_0$  to  $FF_3$  are shown in Figure 2.11.

### 2.3.2.3 Reliability enhancement using redundancy

#### Fixed Modular Redundancy v/s Fixed Defect Rate

Three different MRF based 4-bit Ripple Carry Adders have been designed. These adders have TMR at varying levels of granularity. Although *adder1* has no redundancy, *adder2* and *adder3* have been designed with TMR at each 1-bit Full adder

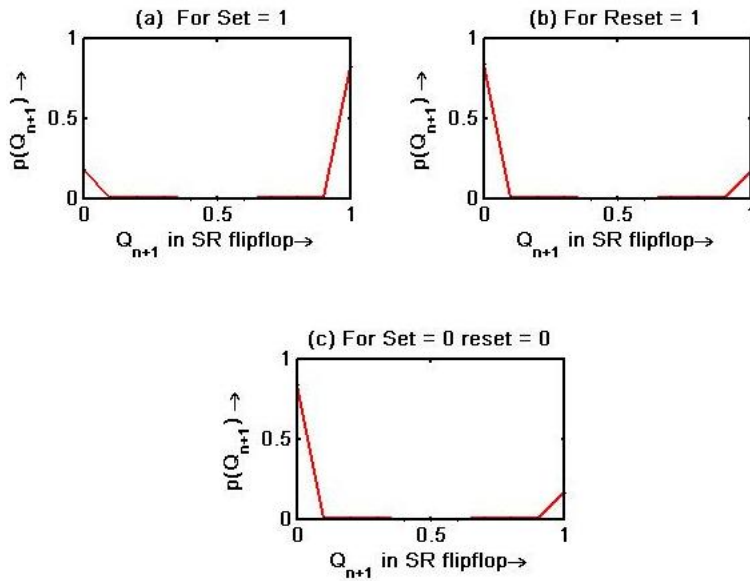


Figure 2.10: Simulation results: Probability of getting correct output for a NOR based SR flip flop

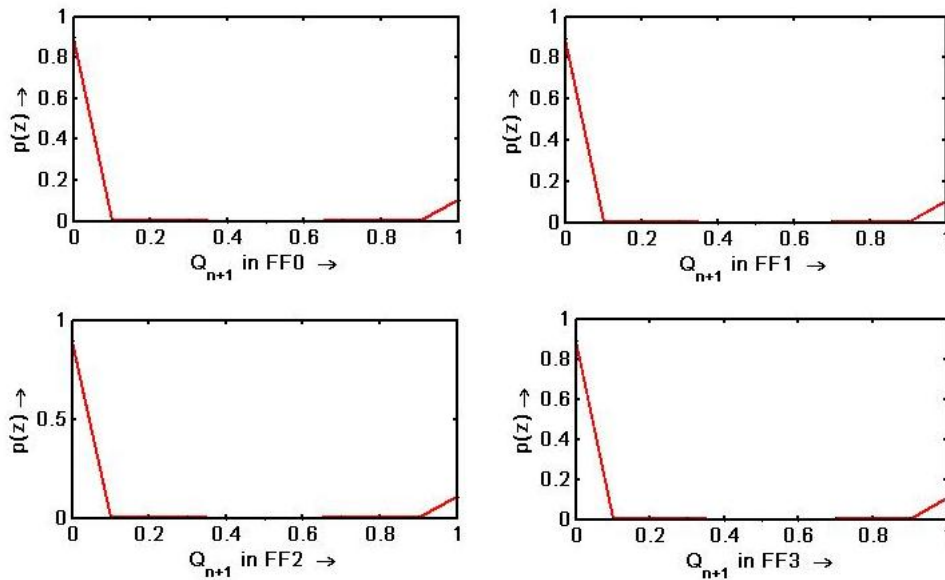


Figure 2.11: Simulation results: Probability of next state  $Q_{n+1}$  for a 4-bit Serial-In-Serial-Out right shift register when  $p(S_{in} = 0) = 0.9$



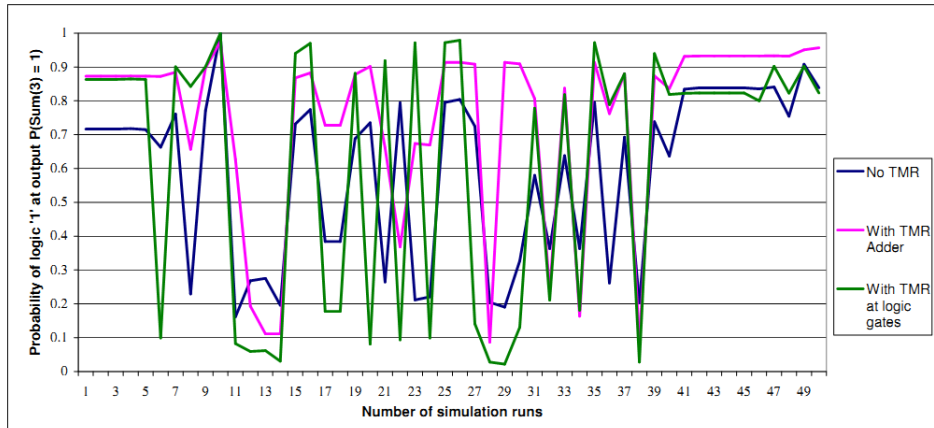


Figure 2.12: Simulation results: Probability of obtaining logic '1' on Sum(3) for a 4-bit ripple carry adder for varying defect rate at each simulation run. Total test cases are 50 and simulation is done for (a) No redundancy (b) TMR at each 1-bit adder (c) TMR at each logic gate

and TMR at each logic gate, respectively. The probability that correct input appears on primary input is assumed to be 0.7. The transient faults are injected randomly at any potential fault site. For fixed redundancy, various combinations of inputs are applied and faults are injected randomly. Total numbers of simulations are 50. It is observed that for some input combinations and fault sites,  $TMR_M$  is less reliable than  $TMR_G$  and vice versa. For varying redundancy, let us consider inputs to the adder be  $A = "0011"$  and  $B = "0110"$  with input carry  $C_{in} = '1'$ . The probability of logic '1' occurring on output ( $sum$ ) is plotted with increasing faults in Figure 2.13. As expected, modular redundancy increases the circuit's reliability. However, as depicted from Figure 2.12, for most of the input combinations and fault sites, TMR at module level (adder2) is less reliable than TMR at gate level (adder3) [74].

### Varying Modular Redundancy v/s Increasing Defect Rate

Here, four different MRF based 4-bit Ripple Carry Adders have been designed, namely  $adder_{0mr}$ ,  $adder_{3mr}$ ,  $adder_{5mr}$  and  $adder_{7mr}$ . The modular redundancy has been introduced at gate level. This means  $adder_{imr}$  has been designed with mod-

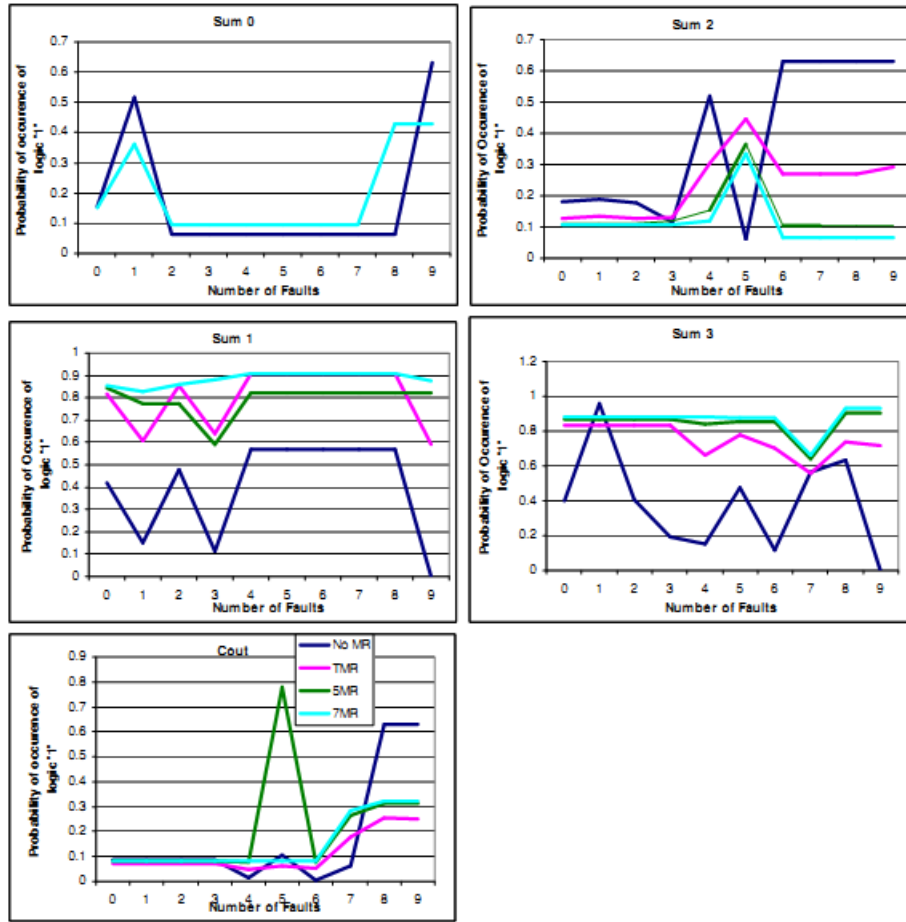


Figure 2.13: Simulation results: Probability of logic '1' at  $Sum(0)$ ,  $Sum(1)$ ,  $Sum(2)$ ,  $Sum(3)$  and  $Cout$  with increasing fault rate and varying levels of modular redundancy. Plots are for cases: (i) no redundancy, (ii)  $TMR$  (iii)  $5MR$  (iv)  $7MR$

ular redundancy  $i$  at each logic gate, for  $i = \{0,3,5,7\}$ . The defects have been introduced randomly and increased by one at each simulation run. Let us consider inputs to these adders be  $A = "0011"$  and  $B = "0110"$  with input carry  $C_{in} = '1'$ . The probability of logic '1' occurring on output has been plotted with increasing faults in Figure 2.13. It has been observed that, although the reliability increases with modular redundancy, sometimes we get wrong results with this approach.

### 2.3.3 Other Probabilistic Modeling Techniques

#### 2.3.3.1 Bayesian Network (BN)

Bayesian Networks (BNs) based probabilistic approach is proposed by Bhanja *et al.* [4, 75–79] for modeling nano-domain circuits. Bayesian Networks are graphical models representing the joint probability function over a set of random variables, using a Directed Acyclic Graph (DAG) structure [80]. The nodes of this DAG represent random variables and node to node arcs denote direct conditional dependencies. Both Bayesian Network and Markov Random Field makes use of belief propagation algorithm [64] to calculate joint and marginal probabilities of primary output and intermediate nodes. A conditional dependency is associated with nodes in both frameworks. Both Bayesian Network and Markov Random Field can generate Junction trees, but by using BN, we can arrive at smallest Junction Tree [79].

Bhanja *et al.* [75] modeled each logic gate using conditional probability table (CPT). If input signal states of a gate are given, then a CPT models the probability of gate output signal being at a logic state zero (or one). Signal probabilities in a Bayesian Network model are computed using local message passing. The inference problem becomes complicated if the underlying undirected graph has cycles. The network compilation process ensures generation of a loop free tree of cliques known as Junction Tree. The Junction Tree helps in local message passing at the cost of increased complexity. A Bayesian network is converted to moral graph followed by triangulation to generate a Junction Tree.

The probabilistic modeling of nano-scale circuits as a Bayesian Network finds its application in computing output error probability and switching activity estimation. The Bayesian network preserves dependencies and can be applied to Networks having casual flow, for example Nano-CMOS, CNT, RTD, etc. The average case complexity of Bayesian Networks is least among most of the probabilistic models. The

worst case space complexity of BN is linear, i.e.,  $O(nF_{max})$  where  $n$  is number of nodes and  $F_{max}$  is maximum fan-in. The time complexity is also linear, i.e.,  $O(nN)$  where  $N$  is the number of samples for stochastic inference scheme [4, 75–79]. Time and space complexities of Bayesian Network are independent of gate error probabilities. Hence, we can conclude that the Bayesian network model is faster than other probabilistic models.

The tools used in [4, 75–79] for inferencing are SMILE (Structural Modeling, Inference, and Learning Engine), GeNIe, HUGIN Lite, SAMIM (Sensitivity Analysis, Modeling, Inference and More), etc.

We have used Hugin Lite software for probabilistic modeling of combinational circuits using the methodology proposed by Bhanja et al.. Under this framework, all logic gates are modeled and used for combinational circuit design and reliability analysis in presence of soft transient errors.

### 2.3.3.2 Probability Transfer Matrix (PTM)

To estimate the effects of soft transient errors on logic circuits, another computational framework based on Probabilistic Transfer Matrices is proposed in literatures [5, 81]. This transfer matrix formulation represents parallel composition of logic gates with tensor products. Instead of probabilistic input signals as in MRF, the PTM approach assumes gate error probabilities. With this methodology, we can compute probability of output signal value and overall probability of correctness for a nano-domain circuit [5]. The Probability Transfer Matrix and Ideal Transfer Matrix model the fault prone and fault free behavior of logic gates, respectively. A probabilistic transfer matrix  $P$  represents probability of four states of a logic signal, i.e.,  $P_{00} = \text{correct}_0$ ,  $P_{01} = \text{incorrect}_0$ ,  $P_{10} = \text{incorrect}_1$  and  $P_{11} = \text{correct}_1$ . A matrix representing all possible input probabilities can be computed by Kronecker product

of the input signal probabilities [5]. The PTMs for logic circuits can be calculated by multiplying the PTMs of gates connected in series and tensoring the PTMs of gates connected in parallel. The gate PTMs are combined by considering signal dependencies between gates. These signal dependencies use the joint and conditional probabilities, within the circuit. As reported in [5], the Probabilistic Transfer Matrices gives accurate information about output error probability, reliability and signal observability of the circuit. Under this framework, all input combinations are computed simultaneously and exact error probabilities are calculated.

The extremely large memory usage limits the performance and usage of PTM approach to smaller circuits. As reported in [5], PTM representation requires  $O(2^{n+m})$  memory space for a gate in  $n$  inputs and  $m$  outputs. Such performance bottlenecks can be handled by using some matrix compression technique like Algebraic Decision Diagrams or by applying some heuristic techniques like Dynamic Weighted Averaging Algorithm and Multi-pass approach [81] for approximate analysis. Still the memory usage is large or results are obtained at the cost of approximations.

Franco et al. proposed an SPR tool which computes the signal reliability of combinational circuit based on SPR model proposed in [81]. The SPR tool works with standard cell generated logic. Here, the intrinsic signal reliability of each cell is supposed to be known a priori. The circuit description generated by the synthesis tool is evaluated for computing the reconvergent fanout signals, cell logical ordering and signals that will be considered for the reliability analysis. In brief, the SPR tool computes signal reliability of complete circuit and calculates the theoretical results which can be used for a reliability driven design process.

We have designed the Probabilistic Transfer Matrices using MATLAB for probabilistic analysis of nano-scale circuits. The basic logic gates, modeled with Probabilistic Transfer Matrices approach can be used for combinational circuit design

and reliability analysis. An 8-bit ripple carry adder is designed this methodology. The results are provided in Section 2.3.4

### 2.3.3.3 Probability Decision Diagram (PDD)

Abdollahi et al. [3] proposed a computational framework based on Binary Decision Diagrams (BDD) for representing probabilistic behavior of circuits with faulty gates. This is known as Probabilistic Decision Diagrams (PDD). In this methodology [3], a probabilistic inverter is attached at output of each logic gate to model its probabilistic behavior. Thus, output function  $f$  of this gate changes to  $f'$  with probability  $p$  of probabilistic inverter. Abdollahi et al. [3] describes how to construct Probability Decision Diagrams of different logic gates and circuits and extracting output probabilities and other information from a PDD. The Probability Decision Diagram is used to encode probabilities into the decision diagrams using weights of the edges. A PDD is a weighted graph with directed edges and it has a single terminal node which represents constant '0'. The terminal node has no outgoing edges. All other nodes have two outgoing edges and a decision variable  $x$ . In contrast to other models, both nodes and edges of a Probabilistic Decision Diagram represents a function. The function  $f'$  of an edge is determined by  $*$  operation on weight of edge  $p$  and function  $f$  of its end node. The  $*$  operation on two probability values  $a$  and  $b$  is given by equation (2.10). Table 2.6 gives sum and carry probabilities for different input combinations of a half adder modeled by PDD approach. Here,  $p$  denotes the gate probability.

$$a * b = (1 - a)b + a(1 - b). \quad (2.10)$$

Abdollahi et al. [3] explains the construction of a PDD for a probabilistic circuit and extracting output error probabilities and other information from it. The time

Table 2.6: Probability of sum and carry nodes of a half adder modeled by PDD

| Input A | Input B | $f_{sum}(A, B)$     | $f_{carry}(A, B)$   |
|---------|---------|---------------------|---------------------|
| 0       | 0       | $p * 0 * 0 = p$     | $p * 0 = p$         |
| 0       | 1       | $p * 0 * 1 = 1 - p$ | $p * 0 = p$         |
| 1       | 0       | $p * 1 * 0 = 1 - p$ | $p * 0 * 0 = p$     |
| 1       | 1       | $p * 1 * 1 = p$     | $p * 0 * 1 = 1 - p$ |

complexity is comparatively less in PDD model. However, reconvergent fanouts in the circuit increase the runtime.

We have modeled the Probabilistic Decision Diagram methodology in MATLAB. The Probability Decision Diagrams for all logic gates are designed and simulated. These logic gate PDDs are used to make combinational circuits.

### 2.3.4 Models Evaluation

The probabilistic framework based on Markov Random Field, Probabilistic Transfer Matrices and Probabilistic Decision Diagrams is developed using MATLAB [6, 82]. The two-input logic gate library is developed following these three methodologies, one at a time. These logic gates are used for combinational circuit design and reliability analysis in presence of soft transient errors.

Lets consider a 1-bit full adder. Suppose, the probability of logic '1' and logic '0' on primary inputs of this adder to be 0.5. When such an adder is modeled by Markov Random Field, the probability of sum is obtained as 0.4908. For modeling this adder using Probabilistic Transfer Matrix, the gate probability is assumed to be 0.9. Therefore, Probability Transfer Matrix for sum and carry is computed as:

$$\text{sum} = \begin{bmatrix} 0.41 & 0.09 \\ 0.09 & 0.41 \end{bmatrix} \quad \text{carry} = \begin{bmatrix} 0.2921 & 0.0829 \\ 0.0729 & 0.5521 \end{bmatrix}$$

Table 2.7: Probability of  $sum$  of 8-bit ripple carry adder using MRF when inputs are  $A = "11101010"$  and  $B = "11011111"$  and  $C_{in} = '0'$ 

| $i$ | $P_{MRF}(sum(i) = '1')$ | $Logic$ |
|-----|-------------------------|---------|
| 0   | 0.724022                | 1       |
| 1   | 0.172673                | 0       |
| 2   | 0.282115                | 0       |
| 3   | 0.686591                | 1       |
| 4   | 0.260673                | 0       |
| 5   | 0.284728                | 0       |
| 6   | 0.682954                | 1       |
| 7   | 0.739023                | 1       |

Table 2.8: Probability of obtaining logic '1' ( $P_{11}$ ) on output sum of 8-bit ripple carry adder using PTM (assuming input probabilities = 0.5 and gate probability = 0.9)

|          | $P(correct_0)$ | $P(incorrect_0)$ | $P(incorrect_1)$ | $P(correct_1)$ |
|----------|----------------|------------------|------------------|----------------|
| $sum(0)$ | 0.4100         | 0.0900           | 0.0900           | 0.4100         |
| $sum(1)$ | 0.3739         | 0.1886           | 0.1886           | 0.3739         |
| $sum(2)$ | 0.3706         | 0.2154           | 0.2154           | 0.3706         |
| $sum(3)$ | 0.3716         | 0.2232           | 0.2232           | 0.3716         |
| $sum(4)$ | 0.3724         | 0.2256           | 0.2256           | 0.3724         |
| $sum(5)$ | 0.3729         | 0.2264           | 0.2264           | 0.3729         |
| $sum(6)$ | 0.3731         | 0.2267           | 0.2267           | 0.3731         |
| $sum(7)$ | 0.3731         | 0.2268           | 0.2268           | 0.3731         |

Here, probability of obtaining logic '1' ( $P_{11}$ ) on sum and carry are computed as 0.41 and 0.5521, respectively.

An 8-bit ripple carry adder is modeled using Markov Random Field and Probabilistic Transfer Matrices approaches. Table 2.7 shows the signal probabilities computed by Markov Random Field for this adder. In this case, we assume that probability of getting correct logic value at primary inputs is 0.9. Table 2.7 gives the probability of getting logic '1' on  $sum$  when inputs are  $A = "11101010"$  and  $B = "11011111"$  and  $C_{in} = '0'$ . For implementing this adder using PTM approach, the probability of correct logic on primary inputs is taken as 0.5 and gate probability is assumed to



be 0.9. Table 2.8 shows the output probabilities of obtaining logic '1' ( $P_{11}$ ) on sum. With same input probabilities, the probability of sum is obtained as 0.4908 for all 8 bits, using MRF model. However, it is inferred from Table 2.8, that error propagated through the circuit is also modeled with PTM approach. Hence, as compared to approaches discussed in this manuscript, the PTM model is more accurate.

Table 2.9: Probability Decision Diagram: results for example circuit in Figure 9 of [3]

| Node                    | $F$   | $X$  | $Y$ |
|-------------------------|-------|------|-----|
| Left child probability  | 0.05  | 0.15 | 0   |
| Right child probability | 0.068 | 0.15 | 1   |

Table 2.10: Comparison of probabilistic modeling approaches based on run time [3–5]

| Circuits | BN time(s) | PDD time(s) | PTM time(s) |
|----------|------------|-------------|-------------|
| C17      | 0.0        | 0.001       | 0.076       |
| pcl      | 0.07       | 0.002       | 74.9        |
| decod    | 0.14       | –           | 56.9        |
| cu       | 0.56       | –           | 93.87       |
| count    | 1.14       | 0.010       | –           |
| alu4     | 1.87       | 1.049       | –           |
| 9symml   | –          | 0.220       | 1758        |

Further, all logic gates are also designed using Probabilistic Decision Diagram using MATLAB. The example circuit discussed in Figure 9 of [3] is implemented and results are included in Table 2.9. The edge weights of left and right child's of parent node  $F$  are taken as  $p = 0.1$  and  $q = 0.2$ . The weight associated with parent node is  $r = 0.05$ .

In Table 2.10 the Bayesian Network, Probabilistic Transfer Matrices and Probabilistic Decision Diagrams based approaches are compared for benchmark circuits. The data in this table is obtained from [3–5].

## 2.4 Conclusions

In this chapter, we have presented the state of the art work for molecular devices modeling and analysis. We have presented the literature review on the characteristic properties of saturated and unsaturated organic molecules being used as electronic device and the plausible mechanism for current transport in such devices. It is concluded that Oligo(Phynylene Ethynylene) molecule can be a good candidate for being used as a molecular switch. It shows NDR as well as hysteresis characteristics. The methods to synthesis this OPE molecule and Gold nanoparticles of ( $< 100$  nm diameter) are available in the literatures. An empirical device model for this molecule is developed in VerilogA. It is used for modeling crossbar and nanocell molecular devices. The high density crossbar molecular memories have been fabricated. However, these memories suffer from large defects as precise molecular ordering of nanowires and molecules is not achieved with available bottom-up technology. The crossbar architecture based basic logic gates and 4x4 memory devices are modeled and their simulation results are presented.

Then we have briefly reviewed some of the probabilistic modeling techniques for nanoscale devices. The computational framework for Markov Random Field, Probabilistic Transfer Matrices and Probabilistic Decision Diagrams is developed using MATLAB. Under these frameworks, each logic variable has finite, but random probability of being logic '0' or '1' and/or a certain error probability is associated with each logic gate. A design library is developed for modeling this probabilistic behavior of nanodevices. An 8-bit ripple carry adder is taken as a design example. The NANOLAB tool has been used for probabilistic modeling of combinational circuits using MRF. Both hard errors and soft transient errors, with time varying defect probability, have been assumed to be present during simulation. The NANOLAB tool is augmented to include multiplexer based logic gate and flip flop library. These

design entities can be directly used for combinational and sequential circuit design. Instead of 2:1 multiplexer, 4:1 multiplexer or bigger ones can be used as basic building block. The fault tolerant approach, N-Modular Redundancy (NMR) has been compared at different levels of granularity and for varying levels of redundancy. It is observed that NMR fails to make the device fault tolerant when defect rate is much higher than threshold value. A proposal for augmenting Markov Random Field based modeling approach is evaluated for designing sequential circuits.

Table 2.11: Comparison of probabilistic modeling approaches [6]

| Approach | Time & Space Complexity | Accuracy | Scalability    |
|----------|-------------------------|----------|----------------|
| PDD      | least                   | better   | more difficult |
| BN       | less                    | better   | very easy      |
| MRF      | high                    | better   | difficult      |
| PTM      | highest                 | best     | most difficult |

The HUGIN Lite tool is used for Bayesian networks based circuit design and analysis. It has been observed that Bayesian Network and Probabilistic Decision Diagrams have least time complexity among these approaches. The Probabilistic Transfer Matrices require a lot of memory for data storage and long simulation time. However, PTM approach is comparatively accurate. It is observed that:

1. time complexity of PDD and BN is less than that of MRF and PTM.
2. PTM has highest memory requirement as compared to other approaches.
3. in contrast to other approaches, time and space complexity of PTM model are dependent on gate error probabilities.
4. PTM approach is difficult to scale because of memory limitations.
5. time complexity of PDD is better than Bayesian Network.

6. PTM, BN and PDD assumes gate error probabilities while MRF considers probabilistic inputs only.
7. MRF logic can be implemented in modified CMOS-logic based circuitry.

We thus conclude that, the traditional fault tolerant techniques fail to provide reliability under high defect rates, at nano-scale. Although nano-scale devices possess ultra high density, so hardware is not an issue. Self reliable and self resilient circuits are required to be designed to withstand the high defect rate at nano-scale.

## Chapter 3

# Proposed Modeling and Synthesis Approaches

Emerging molecular crossbar technology offers high density, regular array-like and non-volatile memory structure [7–12]. These devices consume low power, offer low programming voltage and high switching speed. Non-volatility feature provided by these molecular devices, permits memory to be used as programmable elements within a logic device. However, the bottom-up approach employed for device fabrication at nano-scale, lacks precision in molecular device ordering and hence such crossbar molecular devices are inherently defective. The programmable nanocell based approach [1] circumvents this problem. In a nanocell architecture, the molecular switches need not be assembled deterministically but they are randomly oriented and interconnected via gold nanoparticles. In this chapter, we present the modeling and synthesis approach for nanocell molecular memory. Similar to Field Programmable Gate Arrays (FPGAs), a nanocell can be trained for a given functionality after fabrication. This postfabrication training of the nanocell, considers nanocell to be a black box. The nanocell is trained via external control voltage signals. An algorithm is proposed in this chapter for postfabrication synthesis or training of the nanocell.

The outline of this chapter is as follows. We first present in brief the literature review of the nanocell in Section 3.1. This is followed by design and verification of 1-bit molecular memory. In Section 3.2, we illustrate the omnipotent training algorithm for molecular memory. In Section 3.3, we illustrate a noval mortal training algorithm for synthesizing the nanocell molecular memory via external control voltage signals. The experimental setup and simulation results are also reported for both the training algorithms. Then, we conclude the chapter in Section 3.4.

## **3.1 Nanocell Molecular Memory Design Approach**

### **3.1.1 Nanocell Molecular Devices**

In contrast to molecular crossbar devices, a nanocell [1, 20, 43, 83] consists of conducting nanoparticles connected via randomly placed molecules and addressed by relatively small number of leads located at the edges. These molecules exhibit re-programmable negative differential resistance (NDR) characteristics. The spacing between nanoparticles and insertion of molecules between nanoparticles is controlled by chemical self assembly. A monolayer of alkanethiols that coats each nanoparticle, prevents them from coalescing into a multi-particle array. The electrical contacts between adjacent nanoparticles and between nanoparticles and I/O leads is established via molecule-metal chemical bonding. A typical nanocell would contain 250-1000 nanoparticles and 750-10,000 molecular switches approximately. Tour et *al.* used gold nanoparticles of diameter  $60nm$  and spacing of  $3nm$ . The size of a typical nanocell is approximately  $1\mu m^2$ . The post fabrication training of a nanocell (omnipotently or mortally [1]) can be formulated as an optimization problem. The optimization process would require as an input the nanocell to be trained, the target logic in the form of a truth table and the I-V characteristics of the '*ON*' and '*OFF*' states of the molecular switch. Search space for omnipotent training would

include all possible combinations of 'ON' and 'OFF' molecules and all possible assignments of input and output pins. Tour et al. [1] provides quantitative proof in support of the probabilistic feasibility of extracting logic from the random arrangement of nanoparticles and issues pertaining to reliability of nanocells. The in-built defect tolerance, small size, post fabrication programmability through mortal training and lack of requirement of precise molecular ordering features makes it a good choice for future nanoscale devices.

### 3.1.2 Design and Verification of 1-bit Molecular Memory Cell

As an initial attempt, a 1-bit molecular memory cell model has been designed using nanocell based approach. Figure 3.1(a) shows schematic of a nanocell based memory, which has an Address ( $Ad$ ), Write/Read ( $W/R$ ), Data\_In ( $In$ ) and Data\_Out ( $Out$ ) ports. As explained earlier, the nanoparticles are connected to the molecules via metal-molecule chemical bonding. When Address bit is set to high voltage ( $2V$ ) and the memory is in write mode ( $Ad = 2V$  and  $W/R = 0.5V$ ), the data on the Data\_In port is stored to the memory. Now, when the memory is switched to read mode ( $Ad = 2V$  and  $W/R = 2V$ ), the data stored in the memory can be read out from Data\_Out port. Initially, the nanocell is designed in HSPICE using VerilogA model of the Oligo (Phynylene Ethynylene) molecule, as discussed in Section 2.1.2. The molecular connections in this nanocell are done using the directed acyclic graph generated by Nanocell Reliability Prediction Algorithm(NRPA) (to be explained in Section 4.1.2, Algorithm 4). For molecular connections, the following assumptions are made:

1. A certain/fixed number of nanoparticles are present inside the nanocell.
2. Molecules are spatially distributed following the Gaussian Distribution within the nanocell.

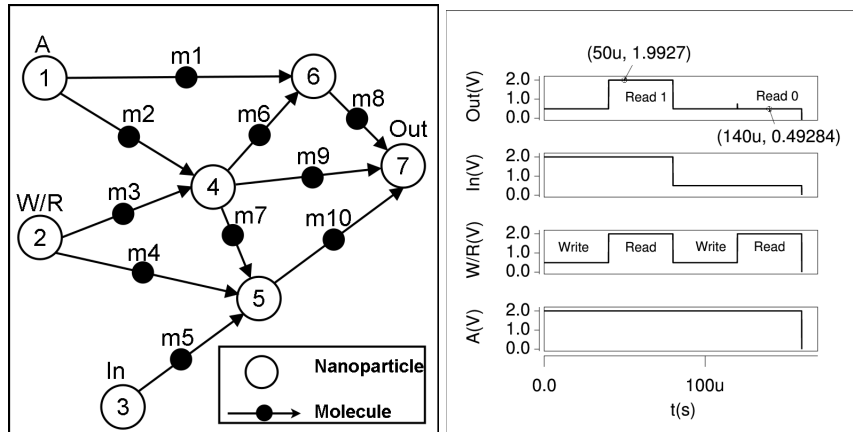


Figure 3.1: Nanocell based 1-bit molecular memory cell (a) Schematic (b) Simulation results for Address (Ad), Write/Read (W/R), Data\_In (In) and Data\_Out (Out) terminals. Here, 2V and 0.5V represent logic '1' and logic '0', respectively.

3. Between two nanoparticles  $N_i$  and  $N_j$ , only single molecular switch  $M_{ij}$  is present.
4. When voltage ( $\geq V_{threshold}$ ) is applied on  $N_i$ , the conformational changes occur in the molecule  $M_{ij}$ . This changes its resistance value and current flows through  $M_{ij}$  to the nanoparticle  $N_j$ , (as shown in Figure 1(b) of [19]). Thus, we say that, the molecules in the nanocell are unidirectional i.e. directed in the direction of current flow (from the set of input nodes to the output node).

Considering these assumptions and using the VerilogA model of the OPE molecule, we have modeled a nanocell molecular memory in HSPICE. Figure 3.1(b) depicts the simulation results for a 1-bit molecular memory cell, which consists of 7 nanoparticles and 10 molecular switches. The nanocell molecular memory instance is simulated for  $\{(w_1, r_1), (w_0, r_0)\}$  operations. The high and low input voltage levels are defined as 2.0V and 0.5V, respectively. Further, one thousand samples of proposed 1-bit molecular memory cell with 20 nanoparticles are generated using Monte Carlo simulation. The input signals for  $\{(w_1, r_1), (w_0, r_0)\}$  operations are given on the input terminals and output is observed on the Data\_Out terminal. Also,



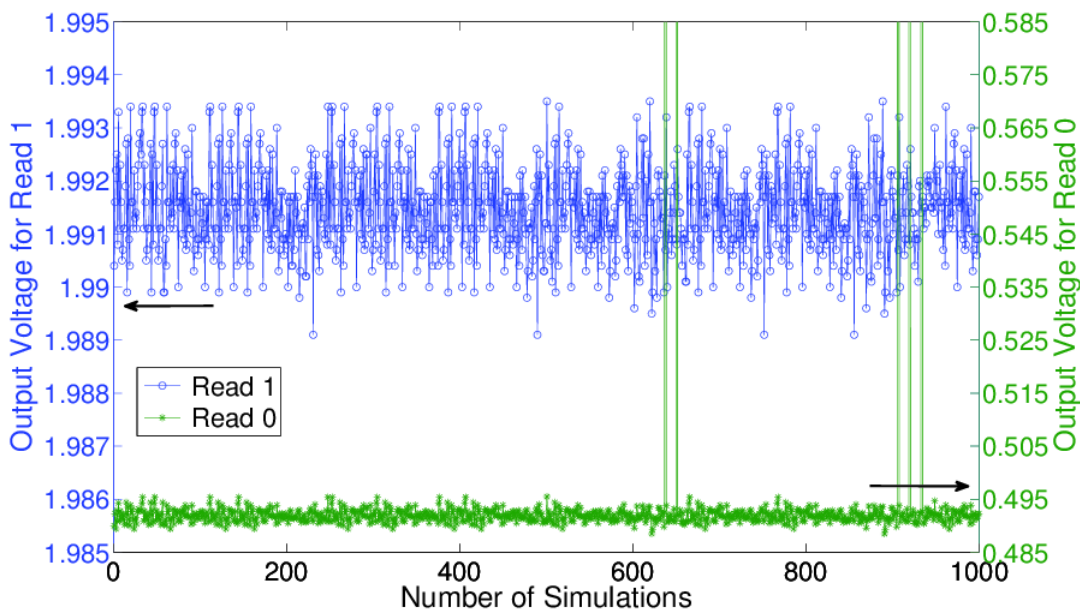


Figure 3.2: Simulation results for 1000 samples of 1-bit memory cell. The left and right y-axes represents Data\_Out voltage for Read 1 and Read 0, respectively

the simulation results for Read 1 ( $r_1$ ) and Read 0 ( $r_0$ ) are plotted in Figure 3.2. The output voltage range for  $r_1$  operation falls between 1.989 V and 1.994 V. It is inferred that even for an untrained memory cell, bit 1 is always read correctly and bit 0 is read correctly for more than 99.5% cases in the range of 0.490 V to 0.495 V. Thus, we can conclude that even an untrained nanocell, consisting of 20 or more nanoparticles and a monolayer of OPE molecules, behaves as a reliable memory device. This observation is theoretically proved in the Chapter 4. However, in order to obtain the behavior of multi-bit storage, the nanocell is required to be trained. The algorithms for training the nanocell molecular memory are discussed Section 3.2 and Section 3.3.

## 3.2 Omnipotent Training of Molecular Memory

In this Section, we will discuss the omnipotent training of a nanocell to behave as

a multi-bit memory. By omnipotence, we mean to say that, the internal topology of the nanocell is priori known to us and individual molecules can be switched to 'ON' or 'OFF' state. Most suitable 'ON' and 'OFF' states of these molecules is a design search problem. This search is handled by Genetic Algorithm (GA). The GA is a method for solving both constrained and unconstrained optimization problems that is based on natural selection, the process that is derived emulating the biological evolution. The GA repeatedly modifies a population of individual solutions. At each step, the genetic algorithm selects individuals at random from the current population to be parents and uses them to produce the children for the next generation. Over successive generations, the population "evolves" toward an optimal solution.

The goal in training a nanocell is to find configurations within a randomly assembled nanocell that will perform as a memory. The set of molecular switches and their locations within the nanocell is passed as initial population to the algorithm. The algorithm switches these molecules to 'ON' or 'OFF' state, to find the optimal set of 'ON' molecular switches for which the nanocell behaves as n-bit memory.

The Algorithm 1 discusses the omnipotent training of nanocell memory using Genetic Algorithm. In this algorithm, the initial population is a set of sequences of '1' and '0' bits and these sequences are randomly generated. Each sequence constitutes a 'chromosome' or an 'individual', representing the state of molecular switches connected between the nanoparticles within the nanocell. The length of each chromosome is equal to number of molecules in the nanocell instance. A '1' value at the  $i^{th}$  position in this population represents that the  $i^{th}$  molecule is present in 'ON' state in the nanocell. Similarly, '0' represents that the molecule is in 'OFF' state. Multiple sequences of  $M$  bit are generated randomly, to represent the multiple nanocell instances. The nanocell is then simulated using HSPICE. The values of the output voltage are then used to calculate the fitness scores of each of the individuals. The

---

**ALGORITHM 1:** Omnipotent training of n-bit memory using Genetic Algorithm

---

```
1: inputs
2:   untrained_MEM:= an untrained sample of a nanocell,
3:   initial_P:= initial population of states of M molecular switches in the nanocell,
4:   /* It defines the desired output voltage levels for Write/Read modes of n-bit
   memory */
5:   num_GEN:= number of generations,
6:   max_GEN:= maximum number of generations
7: outputs
8:   trained_MEM:= A trained n-bit memory device
9: do
10:  Initialize Genetic Algorithm (GA) with population initial_P,
11:  for num_GEN = 1 to max_GEN do
12:    (i): Modify the states of molecular switches in untrained_MEM,
13:    (ii): Simulate the modified untrained_MEM using HSPICE,
14:    (iii): Store output voltage (Out) values for Read and Write operations for all n
        bits in voltage_VAL,
15:    (iv): The GA evaluates fitness function fitness_MEM() using voltage_VAL,
16:    (v): It quantifies the fitness of each individual and generates new population
17:  end for
18:  The GA converges and provides as output:
19:    (i): the optimal set of states of M molecular switches
20:    (ii): the trained_MEM by setting the molecules of the nanocell to ‘ON’ and
        ‘OFF’ state, as obtained above
21:  return (trained_MEM)
```

---

methodology of computing the fitness scores is based on the calculation of the magnitude of deviation from the expected/desired response as described in [43]. The fitness function defines the desired output voltage levels for Write/Read modes of n-bit memory. The genetic algorithm quantifies the fitness of each individual and generates new population. These steps repeat with new population and process stops after *max\_GEN* generations. This algorithm is implemented in MATLAB, HSPICE and PERL.

To exemplify the approach, consider a nanocell which contains 20 nanoparticles interconnected by 242 molecules. For omnipotent training, the size of each individual is equal to number of molecules (i.e., 242) and 200 instances of this individual are randomly generated. The individual is specified in 'bitstring' format, where molecule in 'ON' state is denoted by '1' and 'OFF' state is denoted by '0'. So, in MATLAB simulations, the initial population for GA is 200x242. In MATLAB script for nanocell training, the options used for 'ga' function are:

```
options = gaoptimset('InitialPopulation',y,'TolFun',1.0000e-020,'Generations',  
10,'PlotFcns',@gaplotscores;@gaplotbestf,'PopulationSize',20,  
'SelectionFcn',@selectiontournament,'Display','diagnose','StallGenLimit',  
Inf,'StallTimeLimit',Inf,'PopulationType','bitstring');
```

As we are providing the initial population 'y' to GA, so we have not used creation function ('*CreationFcn*') here. Creation function specifies the function that creates the initial population for 'ga' and selection function ('*SelectionFcn*') specify how the genetic algorithm chooses parents for the next generation. We have used tournament selection ('@*selectiontournament*'). Tournament selection chooses each parent by choosing tournament size players at random and then choosing the best

individual out of that set to be a parent. Tournament size must be at least two and its default value is four. We have used two plot functions, namely (i) '@gaplotscores' and (ii) '@gaplotbestf'. The '@gaplotscores' plots the scores of the individuals at each generation and '@gaplotbestf' plots the best function value versus generation. The algorithm stops if

1. the average relative change in the best fitness function value over Stall generations ('StallGenLimit') is less than or equal to Function tolerance ('TolFun', 1.0000e-020), or,
2. there is no improvement in the best fitness value for an interval of time in seconds specified by Stall time limit ('StallTimeLimit'), as measured by CPU time.

Information is displayed at each iteration because we have chosen 'diagnose' option for 'Display'. These options along with fitness function are given as input to the 'ga' function which computes an optimal set of 'ON' molecules as output with which the nanocell behaves as memory. A fitness of zero indicates that the individual successfully functions as the target memory device.

### 3.2.1 Experimental Setup

Consider a nanocell consisting of 20 nanoparticles and 242 molecules. Assume an initial set of connection of these molecules to the nanoparticles. Let this nanocell is to be trained to behave as a 1-bit memory using omnipotent training. Then, the algorithm stated above can be applied to compute the set of 'ON' molecules (out of 242) such that the nanocell behave as 1-bit memory cell. The simulation results of trained memory cell are similar to the one shown in Figure 3.1. In order to train such a nanocell as n-bit memory, only the fitness function is required to be changed.

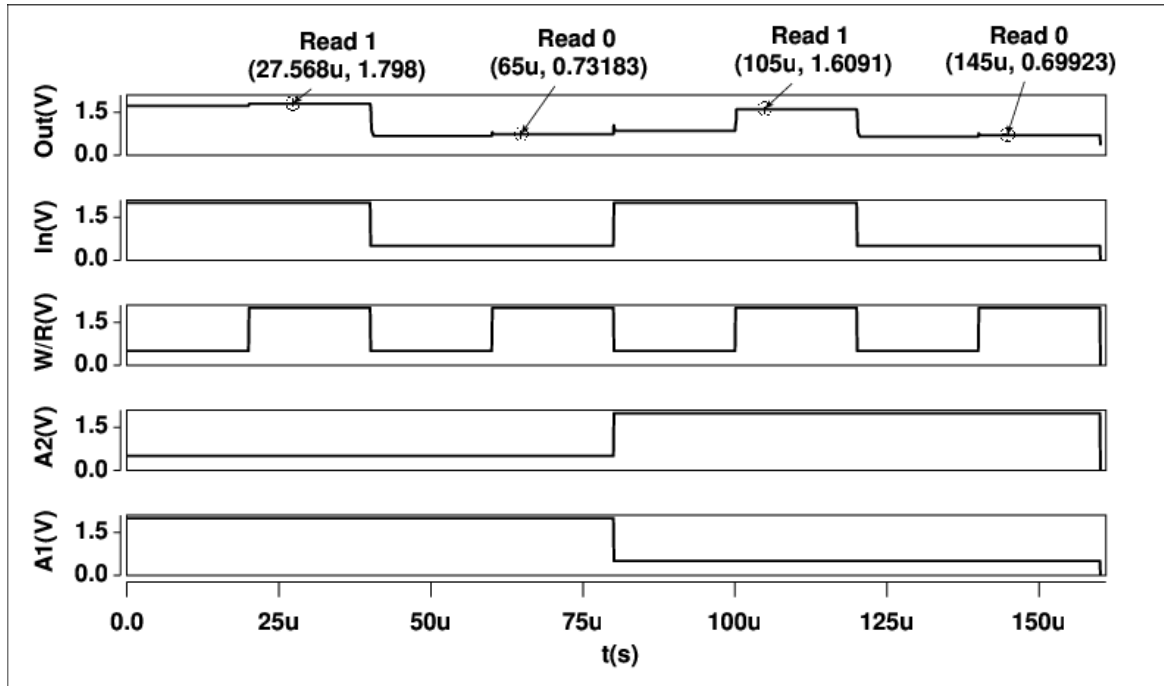


Figure 3.3: Nanocell based 2 - bit molecular memory cell: Simulation results for Address 1 (A1), Address 2 (A2), Write/Read (W/R), Data\_In (In) and Data\_Out (Out) terminals. Here, 2V and 0.5V represent logic '1' and logic '0', respectively.

A  $n$ -bit memory device consists of  $n$  Address lines ( $A_i, \forall i = 1$  to  $n$ ), a Write/Read ( $W/R$ ), a Data\_In ( $In$ ) and a Data\_Out ( $Out$ ) ports. Whenever  $i^{th}$  Address line ( $A_i = 2V$ ) is high, the Write ( $w_a^i$ ) or Read ( $r_a^i$ ) operation is performed for the  $i^{th}$  bit ( $a = 0$  or  $1$ ). The fitness function is modified thus accordingly to depict such behavior. The Figure 3.3 shows the simulation results of an omnipotently trained two-bit memory device. The output of  $Out$  port for  $\{(w_1^1, r_1^1), (w_0^1, r_0^1), (w_1^2, r_1^2), (w_0^2, r_0^2)\}$  operations on bit 1 ( $A_1 = 2V$  and  $A_2 = 0.5V$ ) and then on bit 2 ( $A_1 = 0.5V$  and  $A_2 = 2V$ ) have been plotted. The noise margin high and low are assumed to be  $NM_H = NM_L = 0.4$ . As shown in Figure 3.3, the voltage value on  $Out$  port for Read 1 ( $r_1^i$ ) and Read 0 ( $r_0^i$ ) is within this noise margin for both the bits ( $i = 1$  and  $2$ ). Applying this methodology, we have successfully trained up to 4-bit memory.

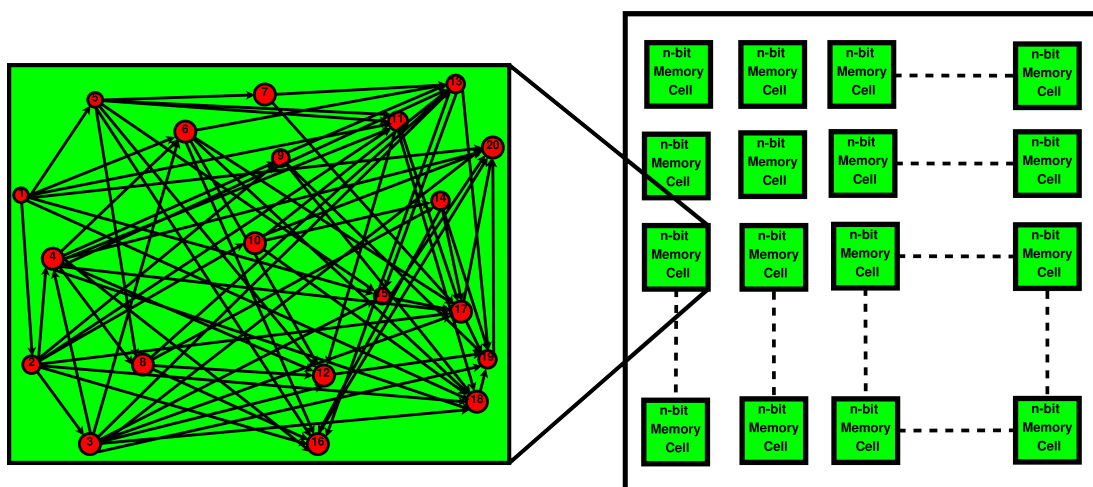


Figure 3.4: Schematic to design multi-bit memory by using multiple n-bit Memory Cells (represented by green square box). Each Memory Cell consists of  $N$  nanoparticles (red circles) and  $M$  molecular switches (black arrows). [Note: Not drawn to scale]

Figure 3.4 illustrates how, in general, n-bit memory cells could be used in mesh formation to construct multi-bit memory. In this figure, each block (named as n-bit Memory Cell) represents a nanocell having 20 nanoparticles and it has been trained as 2-bit memory device. We have trained a 64-bit memory by using thirty two such 2-bit memory cells, as illustrated in Figure 3.4. The higher order memory devices can also be trained using similar concepts.

### 3.3 Mortal Training of Molecular Memory

As stated earlier, for On-Chip training or postfabrication of the nanocell, it would be impossible to switch the individual molecules to 'ON' or 'OFF' state. This is due to extremely small size of the molecules and the nanoparticles. Hence, the nanocell must be trained by mortal assumption. By mortal training, we mean to say that, the nanocell is assumed to be a black box to the training algorithm and voltage signals

---

**ALGORITHM 2:** Mortal training of n-bit memory using Genetic Algorithm

---

```
1: inputs
2:   untrained_MEM: an untrained sample of a nanocell,
3:   initial_C: initial population of randomly generated k Control Voltage Signals
   (CVS),
4:   /* These CVS are denoted as  $C_i, \forall i = 1 \text{ to } k$ 
5:   /* Each  $C_i$  value is either low ( $V_{low}$ ) or high ( $V_{high}$ ) */
6:   num_GEN: number of generations,
7:   max_GEN: maximum number of generations
8: outputs
9:   trained_MEM: A trained n-bit memory device
10: do
11:   Initialize Genetic Algorithm (GA) with population initial_C
12:   for num_GEN = 1 to max_GEN do
13:     (i): Modify the values of Control Voltage signals in untrained_MEM
14:     (ii): Simulate the modified untrained_MEM using HSPICE
15:     (iii) Store output voltage (Out) values for Read and Write operations for all n bits
        in voltage_VAL
16:     (iv): The GA evaluates fitness function fitness_MEM() using voltage_VAL
17:     (v): It quantifies the fitness of each individual and generates new population
18:   end for
19:   The GA converges and provides as output:
20:     (i): the optimal set of control voltage ( $C_i$ ) values
21:     (ii): the trained_MEM by using these control voltage values
22:   return (trained_MEM)
```

---



are applied externally to switch the state of the molecules. This methodology for mortal training of the nanocell using Genetic Algorithm (GA) is proposed in this Section.

The Algorithm 2 discusses the Genetic Algorithm (GA) based mortal training of the molecular memory. Initially, the set of nanoparticles (out of 50) which are to be connected to the CVS ports are chosen randomly. For this, the Gaussian distribution with  $\mu = 25$  and  $\sigma = 20$  is used. A DC voltage (i)  $V_{low} = 0.5V$ , or (ii)  $V_{high} = 2.0V$ , is applied to all these Control Voltage Signals ( $C_i, \forall i = 1 \text{ to } 20$ ). The number of individuals are kept constant throughout all generations (i.e. number of CVS =  $k = 20$ ). Other options are same as we have chosen for omnipotent training (as discussed in previous Section). The GA trains the nanocell for a maximum of  $max\_GEN$  generations. The fitness function  $fitness\_MEM()$  computes whether the *Out* voltage for read operation is obtained in acceptable noise margin or not. In the end, the GA converges to optimal set of Control Voltage ( $C_i$ ) values (high or low) for which the nanocell behaves as a n-bit memory. By this we mean to say that, a DC voltage value (i)  $V_{low} = 0.5V$ , or (ii)  $V_{high} = 2.0V$ , is assigned to each  $C_i$ , such that the nanocell, *trained\\_MEM*, behaves as a n-bit memory.

### 3.3.1 Experimental Setup

Consider a nanocell consisting of 50 nanoparticles and 644 molecular switches. Suppose this nanocell is to be trained mortally to exhibit the behavior of a 2-bit memory device. The schematic of a 2-bit memory is shown in Figure 3.5. As depicted from this figure, the internal molecular connections of the nanocell are assumed to be unknown for mortal training. The Address 1 ( $A_1$ ), Address 2 ( $A_2$ ), Write/Read ( $W/R$ ), and Data\_In ( $In$ ) are the input ports and Data\_Out ( $Out$ ) is the output port. There are twenty Control Voltage Signals (CVS), denoted as  $C_i, \forall i = 1 \text{ to } 20$  in Figure 3.5. A set of high or low voltage signals are applied to these

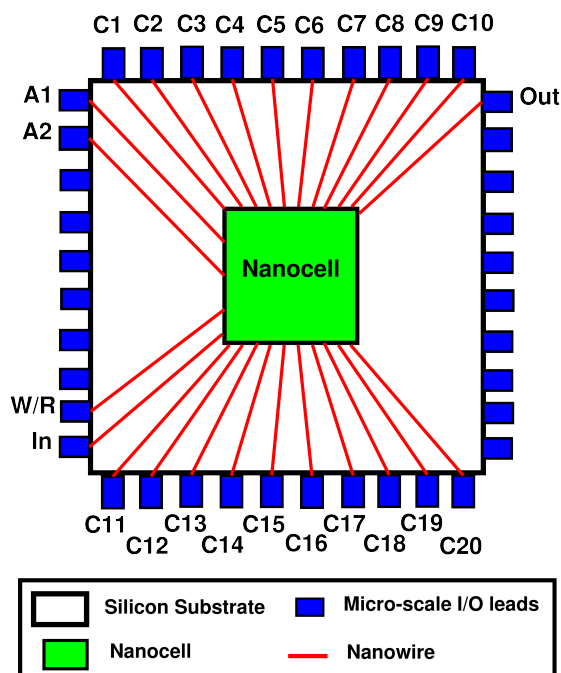


Figure 3.5: Schematic of proposed mortally trained 2-bit memory device consisting of 20 control signals [Note: Not drawn to scale]

CVS ports which are used to externally switch the state of molecules to '*ON*' or '*OFF*'. The set of CVS values for which the nanocell behaves as a 2-bit memory is computed by utilizing the Algorithm 2. In this way, the nanocell is mortally trained to behave as a 2-bit memory. The simulation results of 2-bit memory device for read and write operations are similar to those obtained by omnipotent training, as shown in Figure 3.3.

### 3.3.2 Design space exploration

As a proof of concept, an experiment is performed to analyze the mortal training approach in complete design space of a nanocell, consisting of  $N$  nanoparticles and  $M$  molecular switches. Suppose such a nanocell is to be trained for storing 2-bits of data. Then, as depicted from Figure 3.5, there will be five input-output nodes and five or more Control Voltage Signals (CVS) that can be chosen from

---

**ALGORITHM 3:** Mortal training of n-bit molecular memory

---

```

1: inputs
2:   untrained_MEM:= an untrained sample of a nanocell consisting of  $N$  nanoparticles,
3:   num_C:= number of Control Voltage Signals,
4:   num_C_LOC:= number of nanoparticles to which Control Voltage Signals (CVS)
   can be applied
5:   /* num_C_LOC  $\subset N$  */
6:   /* num_C_LOC  $\leq (N - IO)$ , where IO denotes number of input and output nodes
   other than CVS*/
7:   /* num_C_LOC  $\geq num_C$  */
8: outputs
9:   trained_MEM: set of all successfully trained n-bit memory devices,
10:  num_MEM: number of trained n-bit memory samples in the set trained_MEM
11: do
12:   Generate_C(num_C)
13:   for  $i = 1$  to num_C do
14:     Generate a set  $S$  which consists of all bipartitions of num_C Control Voltage
     Signals,  $C_i$ .
15:     for  $j = 1$  to sizeof( $S$ ) do
16:       Assign  $\{V_{low} \text{ or } V_{high}\}$  to each  $C_i^j$ 
17:     end for
18:     Save these in CVS_VALUES.
19:   end for
20:   Generate_C_Loc(num_C_LOC, num_C)
21:     (i):Generate all possible combinations of num_C_LOC nanoparticles to which
     num_C control voltage signals can be connected.
22:     (ii):Save these in LOCATIONS.
23:   Generate_Mem(untrained_MEM, LOCATIONS)
24:     (i): Modify untrained_MEM by using Control Voltage Signal location specified
     in LOCATIONS.
25:     (ii): Save all memory samples in MEM_MODIFIED.
26:   Generate_Spfiles(MEM_MODIFIED, CVS_VALUES)
27:     (i): For each memory sample in MEM_MODIFIED, assign Control Voltage
     values from CVS_VALUES.
28:     (ii): Save all newly modified memory samples in SP_FILES.
29:   Simulate_Mem(SP_FILES)
30:     (i): Simulate all memory samples in SP_FILES using HSPICE.
31:     (ii): Save output in OUT_FILES.
32:   Fetch_Mem2bit(OUT_FILES)
33:     (i): Select all those memory samples, for which n-bit memory read and write is
     done in acceptable noise margin, using the OUT_FILES.
34:     (ii): Copy all selected memory samples to trained_MEM.
35:   return (trained_MEM, num_MEM)

```

---

remaining  $N - 5$  nanoparticles. Let  $num\_C$  and  $num\_C\_LOC$  denote the number of CVS and the number of nanoparticles to which CVS can be assigned, respectively. Again,  $num\_C\_LOC \leq (N - 5)$ . So, a total of  $C_{num\_C}^{num\_C\_LOC}$  combinations in which  $num\_C$  Control Voltage Signals can be applied to the untrained memory (*untrained\_MEM*). Thus, all possible combinations of  $num\_C\_LOC$  nanoparticles to which  $num\_C$  Control Voltage Signals can be connected are generated using the sub-routine: **Generate\_C\_Loc**( $num\_C\_LOC, num\_C$ ), as shown in Algorithm 3. Further, on these  $num\_C$  Control Voltage Signals, a DC voltage (i)  $V_{low} = 0.5V$ , or (ii)  $V_{high} = 2.0V$  is to be applied. The sub-routine **Generate\_C**( $num\_C$ ) is used to find all possible bipartitions of  $num\_C$  control signals. Total number of unique bipartitions for  $num\_C$  control signals are  $2^{num\_C-1} - 1$ . Here we have excluded all low voltage and all high voltage. These bipartitions are stored in *LOCATIONS*. Then, *CVS\_VALUES* and *LOCATIONS*, obtained in step 12 and 20 of this algorithm, are used to modify the *untrained\_MEM* and generate the memory samples, as explained in steps 23 and 26. Further in step 29, these memory samples are simulated in HSPICE and results are saved in *OUT\_FILES*. Finally in step 32, all spice files which successfully perform read and write operation on both bits are selected. These are the set of trained memory files for a given noise margin. In Algorithm 3, all directory names are written in capital, to distinguish them from variables. The functions (or sub-routines) are written in bold with a brief description beneath them. These functions are implemented in MATLAB, HSPICE and PERL.

To exemplify this approach, consider a nanocell consisting of twenty nanoparticles and six CVS which can be connected to any of the twelve nanoparticles. Then, total combinations in which six CVS signals can be connected is  $C_6^{12} = 924$ , calculated using sub-routine **Generate\_C\_Loc**( $num\_C\_LOC, num\_C$ ). Also, the sub-routine **Generate\_C**( $num\_C$ ) computes  $2^{6-1} - 1 = 31$  bipartitions in which high or

Table 3.1: Number of successfully trained nanocell configuration for 2-bit memory read and write operation. Here, noise margin for Data read operation on output voltage node is considered as  $0.4V$

| No. of nano-particles per Nanocell ( $N$ ) | No. of molecules ( $M$ ) | No. of successfully trained nanocell instances with $i$ CVS |           |          |          |
|--|--------------------------|---|-----------|----------|----------|
|  |                          | 6 CVS   | 8 CVS     | 10 CVS   | 12 CVS   |
| 20   | 97                       | 2   | 0         | 0        | 0        |
| 20   | 104                      | 1   | 0         | 0        | 0        |
| 30   | 223                      | 27  | 13        | 5        | 0        |
| 30   | 214                      | 33  | 19        | 4        | 0        |
| 40   | 399                      | 103   | 55        | 13       | 2        |
| 40   | 392                      | 98  | 43        | 10       | 1        |
| Total                                      |                          | 6x31x924  | 6x127x495 | 6x511x66 | 6x2047x1 |

low voltage is applied to six CVSs. Using this, a total of  $(31 \times 924)$  modified memory samples are simulated for  $\{(w_1^i, r_1^i), (w_0^i, r_0^i)\}$ , for both the bits ( $i = 1$  and  $2$ ). The process is repeated by (i) varying the number of CVS (6, 8, 10, 12), and (ii) increasing the number of nanoparticles (20, 30, 40). That is, consider three nanocells, each consisting 20, 30 and 40 nanoparticles, respectively. It is observed from Figure 4.2(a) that the probability of at least one path from input to output is close to unity for most of the cases, when  $N = 20$  or more. Hence, for mortal training, we have chosen  $N=20, 30$  and  $40$ . Any other value for  $N (> 20)$ , can also be chosen. Then, for each value of  $N$ , two samples of nanocell are generated using NRPA algorithm (Algorithm 4) (to be explained in Section 4.1.2). So, two nanocell samples with  $M = 97$  and  $M = 104$ , respectively, are generated. It is deduced from Table 4.2, that if  $N = 20$ , the number of molecules in nanocell sample may vary from 67 to 123, with mean 95. So, we have generated a total of six memory samples with varying number of nanoparticles. Now, for each of these memory samples, generate four memory instances by varying number of CVS, namely, 6, 8, 10 and 12 CVS respectively. So, a total of 24 memory samples are generated. However, any other value

of CVS can also be chosen for design space exploration. Then, for each case apply Algorithm 3 and obtain number of successfully trained memory samples. The total number of successfully trained 2-bit memory samples are listed in the Table 3.1. The last row of this table depicts the total number of memory samples (*SP\_FILES*) simulated for 6, 8, 10 and 12 CVS, respectively. It is inferred from this table that to successfully train a 2-bit memory, the number of CVS must be approximately one-fourth of the total number of nanoparticles. In this way, we can conclude that, if a nanocell contains  $N$  nanoparticles and  $IO$  input/output ports, then upper limit on CVS is  $(N - IO)$  and lower limit is  $(N/4)$ . Also, it is observed that as we increase the number of CVS signals, nanocell's overall functionality is decreased.

### 3.4 Conclusions

In chapter 3, the nanocell molecular memory modeling and synthesis approach is presented. The device model of this OPE molecule [17–19] is described in VerilogA and used for modeling the nanocell molecular memory in HSPICE. Due to the hysteresis property of the OPE molecule, it is observed that even an untrained nanocell behaves as a buffer or a latch, provided that at least one path is present between input and output node. The 1-bit molecular memory cell is verified for 1000 random configurations generated using Monte Carlo Simulation. It is observed that such a memory cell can successfully perform read and write operation for more than 99.5% of the samples.

The concept is further extended and applied for post fabrication synthesis of the nanocell by (i) omnipotent as well as (ii) mortal training. The Genetic Algorithm (GA) is used for training the nanocell. In the proposed approach for mortal training, the external Control Voltage Signals (CVS) are applied to some of the nanoparticles in the nanocell. All possible combinations of high and low voltage values are ap-

plied to these (CVS). Finally, a set of CVS voltage values are obtained for which the nanocell behaves as a n-bit memory. Since, the search space exponentially increases with the increase in number of nanoparticles and number of CVS, the proposed Genetic Algorithm based mortal training algorithm can be applied to successfully train a nanocell in polynomial time. However, a considerable amount of noise margin is present in the trained 2-bit memory device. Our methodology is flexible and easily scalable to synthesize the nanocell for multi-bit storage functionality.

However, there are certain limitations associated with our theoretical modeling approach. These are listed below:

- The molecular device model used in this thesis is based on empirical equations as proposed by [39–42]. In practical, the change in conduction states of the molecule depends on its conformational changes as discussed by [32]. Thus, it is a dire necessity to propose an enhanced as well as efficient molecular device model that captures these conformational changes. Also, it should be flexible enough to be used in nanocell or crossbar device model. Hence, it is expected that the updated molecular memory model, will be close to realistic behavior of molecule.
- We have assumed that between any two nanoparticles only one molecule is present. However, when a self assembled monolayer of molecules is inserted between randomly placed gold nanoparticles, it will be quiet impossible to have single molecular connection between two nanoparticles. Thus, in order to make a robust nanocell model, such an assumption must be removed. It is required to analyze the effects of multiple edges and self loops in a nanocell.
- We have not focused on optimally tuning the parameters of Genetic Algorithm (GA). As explained in Section 3.3.2, the search space is very large and it takes

a couple of days to find a near optimal solution. Hence, we used GA to find the best optimal solution in polynomial time for both the training algorithms. However, it is observed that the Genetic Algorithm consumes large amount of time for convergence, even for training a small nanocell of 50 nanoparticles. It is required to explore adaptive learning algorithms to reduce the training time.

- The proposed mortal training needs to be applied for training large nanocell molecular memory.



## Chapter 4

# Probabilistic Analysis of Nanocell Molecular Memory

In nano-scale devices, the soft transient errors play a vital role [2]. The defect rate can be in the range  $10^{-3}$  to  $10^{-7}$  in nano-scale devices [7, 8]. These defects can occur due to hard errors or soft errors. The hard errors or structural defects are caused during the device fabrication and later due to aging effects. Noise, thermal perturbation, cosmic rays, etc. are the environmental factors that may cause a soft transient error to occur. As compared to other nano-scale devices, a nanocell is defect tolerant even in presence of high defect rate [1, 20]. This is because of multiple redundant conducting paths present between input and output node of a nanocell. Thus, even if some of the molecules fail, the device functions correctly until no such path is present. As we know, effect of transient errors is for a short while, so the failed molecules may get repaired after sometime. Throughout this thesis, by molecular failure, we mean to say that it has turned '*OFF*'. This can happen because of (i) broken chemical bond (connection) between nanoparticle and molecule, or (ii) the transient errors cause a conformational change to the molecule and it switches to a low conducting state. Similarly, repairing of a molecule denotes that it turns '*ON*' and now it is functioning properly. The external electric field applied to the

molecule is responsible for the transition between high and low conductance states. The electric field changes the molecular dipole moment of middle phenylene ring in OPE molecule and thus the transition of the states takes place [22, 32]. Henceforth, this failure and repair behavior of the molecules within the nanocell can be modeled as a continuous parameter birth-death process [84, 85]. A birth-death process is a Markov chain with states  $\{0, 1, \dots\}$  for which transitions from state  $S_n$  may go only to either state  $S_{n-1}$  or state  $S_{n+1}$ . That is, only one step transitions are allowed in a birth-death process.

In this chapter, we present the probabilistic modeling and analysis of nanocell. In section 4.1, a computational framework is proposed to compute the probability of retrieving the stored data bits correctly at the output terminal of the nanocell buffer. Also, the bounds on reliability of nanocell at any instant of time are computed. The necessary and sufficient conditions for correct functioning of a nanocell are derived. An algorithm is proposed and evaluated to automatically generate a nanocell instance and compute the probability that at least one path is present between input and output. This algorithm is implemented in MATLAB, HSPICE and PERL. Moreover, the experimental setup for the nanocell's reliability estimation is presented and results are discussed.

Further, an augmented continuous parameter birth-death model is proposed in section 4.2 and it is used for reliability evaluation of a nanocell in presence of transient errors. For this mathematical framework, the steady state probability and probability of being in each sub-state is computed. The proposed approach is extended to compute the expected lifetime and availability of the nanocell using the birth-death model of molecules and their spatial connectivity. An algorithm is proposed and implemented in MATLAB, PERL and HSPICE, to automatically generate the proposed model representation for a given nanocell and use it to estimate the

*success\_ratio* as well as the nanocell reliability, while considering the uncertainties. Conclusions along with the limitation of theoretical modeling are presented in section 4.3

## 4.1 Reliability Analysis of Nanocell in Spatial Domain

A nanocell may have multiple paths connecting input to output ports. At least one of the minimal path must be present between these ports. This is a necessary condition for correct functioning of the nanocell based device. It is sufficient to have multiple paths from inputs to outputs and some of which may or may not intersect. The presence of multiple paths introduce redundancy and increase probability of getting correct output.

Let us model the nanocell as a planar graph  $G(V, E)$ , where nanoparticles are the nodes and molecular switches in '*ON*' state are the edges. The graph  $G(V, E)$  is assumed to be a directional graph such that all the molecules are oriented in same direction, i.e. from input to the output (as explained in Chapter 3). The primary input port is the root node and the primary output port is the leaf vertex of the graph  $G(V, E)$ . Consider a nanocell with '*N*' nanoparticles and '*M*' molecular switches. Assume that these nanoparticles are always present and molecular switches are distributed by Gaussian distribution within the nanocell. A molecular switch  $i$  is present in '*ON*' state with probability  $p_i$ .

$$P\{X_i = 1\} = p_i, \quad \forall i = 1 \text{ to } M \quad (4.1)$$

This probability that  $i^{\text{th}}$  molecule is present in '*ON*' state is called reliability of that molecule at that instant of time, denoted as  $R(p_i)$ . So, the probability that there is no edge (or connection) between two nodes (or nanoparticles) is  $(1 - p_i)$ . Again,

the probability that at least one edge is present between two nodes is given by:

$$P\{X_i \geq 1\} = 1 - (1 - p_i), \quad \forall i = 1 \text{ to } M \quad (4.2)$$

To exemplify the proposed approach, a small nanocell is considered. Let us suppose that it consists of three molecular switches ( $m_1$ ,  $m_2$  and  $m_3$ ) and three nanoparticles ( $A$ ,  $B$  and  $C$ ), as shown in Figure 4.1(a). Here, input voltage is applied on  $A$  and received on  $C$ . This can be done via two minimal paths: (i)  $\overrightarrow{ABC} = \{A - m_1 - B - m_2 - C\}$ , or (ii)  $\overrightarrow{AC} = \{A - m_3 - C\}$ . Correct output will be received on  $C$  via path  $\overrightarrow{ABC}$  if both molecules  $m_1$  and  $m_2$  are present. Similarly, data will be correctly received on  $C$  via path  $\overrightarrow{AC}$  if the molecule  $m_3$  is in 'ON' state. These are the two redundant paths and at least one of them should be working correctly to obtain correct output. Thus, the probability of receiving correct data on node  $C$  is given by:

$$\begin{aligned} P_{path} &= (\text{Probability that both } m_1 \text{ and } m_2 \text{ are present in 'ON' state} \\ &\quad \text{on path } \overrightarrow{ABC}) \text{ or (Probability that } m_3 \text{ is present in 'ON' state} \\ &\quad \text{on path } \overrightarrow{AC}) \\ &= 1 - (1 - p_{m_1}p_{m_2})(1 - p_{m_3}) \end{aligned} \quad (4.3)$$

$$= p_{m_3} + p_{m_1}p_{m_2} - p_{m_1}p_{m_2}p_{m_3} \quad (4.4)$$

In this example, the nanocell is working as a buffer. The low and high voltage applied on input node  $A$  are 0.5 V and 2.0 V, respectively. Table 4.1 shows the low and high voltage values received on node  $C$  when either none or some of the molecules are missing. Out of eight test cases, correct output is received for five cases only. In other words,  $5/8 = 0.625$  or there are 62.5% chances of getting

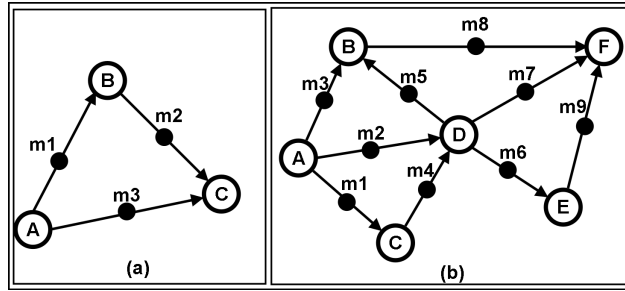


Figure 4.1: A small network of nanoparticles and molecular switches (a) 3-particles, 3-switches, 2-paths between node A to node C (b) multiple particles and switches, multiple paths between node A to node F

Table 4.1: Simulation results for an example nanocell configuration when none or some of the molecules are missing

| Missing Molecules | V(low) | V(High) | $P_{path}$ |
|-------------------|--------|---------|------------|
| None              | 0.4935 | 1.9935  | 0.625      |
| m1                | 0.4951 | 1.9951  | 0.500      |
| m2                | 0.4951 | 1.9951  | 0.500      |
| m3                | 0.4902 | 1.9902  | 0.250      |
| m1, m2            | 0.4951 | 1.9951  | 0.500      |
| m2, m3            | 0      | 0       | 0.000      |
| m1, m3            | 0      | 0       | 0.000      |
| m1, m2, m3        | 0      | 0       | 0.000      |

correct output voltage. Theoretically, on substituting  $p_{mi} = 0.5$ ,  $\forall i = 1, 2, 3$  in the above equation, we get  $P_{path} = 0.625$ . Thus, our theoretical and experimental results are matching. The last three cases in the Table 4.1 denotes the minimal cut sets for this nanocell, denoted by,  $C_1 = \{m2, m3\}$ ,  $C_2 = \{m1, m3\}$  and  $C_3 = \{m1, m2, m3\}$ . As depicted from Table 4.1, output voltage is not received for these cases.

Further, to evaluate the reliability of a trained nanocell, we assume that there are  $k$  redundant paths from input to output. Each of these paths may vary in length. The length of any path  $i$  can be represented by variable  $l_i$ ,  $\forall i = 1$  to  $k$ . That is, each path consists of  $l_i$  molecules connected in series and such  $k$  paths are working in parallel. For correct functioning of the system, at least one of the  $i$  paths must

function correctly. Consider an indicator variable  $x_{ji}$  which denotes the state of molecular switch  $j$  on path  $i$ , such that:

$$x_{ji} = \begin{cases} '1' & \text{if molecule } j \text{ on path } i \text{ is 'ON'}, \\ '0' & \text{otherwise.} \end{cases} \quad (4.5)$$

We define a structure function  $\phi(x)_i$  for path  $i$  as:

$$\phi(x)_i = \prod_{j=1}^{l_i} x_{ji} = \phi_i, \quad \forall i = 1 \text{ to } k \quad (4.6)$$

Then, structure function of the whole system can be given as:

$$\phi(X) = \max(\phi_1, \phi_2, \dots, \phi_k) \quad (4.7)$$

Hence, the reliability  $R(p)$  of the whole system at any instant of time is given as:

$$R(p) = P\{\phi(X) = 1\} \quad (4.8)$$

$$= 1 - \prod_{i=1}^k (1 - \prod_{j \in l_i} p_j) \quad (4.9)$$

Let's consider another example having multiple paths from input to output, as shown in Figure 4.1(b). The node  $A$  is input and node  $F$  is output. The structure function for all minimal paths from  $A$  to  $F$  are  $\{x_{31}x_{81}, x_{22}x_{52}x_{82}, x_{23}x_{63}x_{93}, x_{24}x_{74}, x_{15}x_{45}x_{75}, x_{16}x_{46}x_{66}x_{96}, x_{17}x_{47}x_{57}x_{87}\}$ . We can say that, at least one of the minimal path from  $A$  to  $F$  must be present to receive correct output at  $F$ . This can be defined as necessary condition for a workable nanocell. It is sufficient to have more than one path from

A to F. The structure function for this system is given as:

$$\begin{aligned} \phi(X) = \max(x_{31}x_{81}, x_{22}x_{52}x_{82}, x_{23}x_{63}x_{93}, x_{24}x_{74}, x_{15}x_{45}x_{75}, \\ x_{16}x_{46}x_{66}x_{96}, x_{17}x_{47}x_{57}x_{87}) \end{aligned} \quad (4.10)$$

and, the reliability is computed as:

$$\begin{aligned} R(p) &= P(\phi(X) = 1) \\ &= 1 - (1 - p_3p_8)(1 - p_2p_5p_8)(1 - p_2p_6p_9)(1 - p_2p_7) \\ &\quad (1 - p_1p_4p_7)(1 - p_1p_4p_6p_9)(1 - p_1p_4p_5p_8) \end{aligned} \quad (4.11)$$

#### 4.1.1 Bounds on reliability of a Nanocell

Let  $\{path_1, path_2, \dots, path_s\}$  denote minimal path sets connecting input node to the output node and we define  $E_i, \forall i = 1 \dots s$  as  $E_i = \{\text{at least one molecular connection on path } path_i \text{ has failed}\}$ .

By failing of the molecular connection, we mean to say that, molecule is in 'OFF' state. If at least one of the molecules in the minimal path set has failed, the system will fail eventually. Mathematically, it is denoted as:

$$1 - R(p) = P(E_1E_2 \dots E_s) \quad (4.12)$$

$$= P(E_1)P(E_2|E_1) \dots P(E_s|E_1E_2 \dots E_{s-1}) \quad (4.13)$$

Henceforth, it can be easily derived that failure of at least one molecule in minimal path  $path_i$  can increase the probability of failure of at least one molecule in path  $path_j$ . This would be the case if both paths  $path_i$  and  $path_j$  overlap. So, we can

write:

$$P(E_j|E_i) \geq P(E_j) \quad (4.14)$$

Similarly, we can say:

$$P(E_i|E_1E_2 \dots E_{i-1}) \geq P(E_i) \quad (4.15)$$

Substituting in equation (4.13), we get:

$$1 - R(p) \geq \prod_{i=1}^s P(E_i) \quad (4.16)$$

On simplifying above equation, we obtain:

$$R(p) \leq 1 - \prod_{i=1}^s \left[ 1 - \prod_{j \in \text{path}_i} p_j \right] \quad (4.17)$$

Again, let  $\{cut_1, cut_2, \dots, cut_r\}$  denote the minimal cut sets. We define the events  $C_1, C_2, \dots, C_r$  by  $C_i = \{\text{at least one molecular device in } cut_i \text{ is functioning}\}$ . Since, the nanocell will function iff all of the events  $C_i$  occur, we say:

$$R(p) = P(C_1, C_2 \dots C_r) \quad (4.18)$$

$$= P(C_1)P(C_2|C_1) \dots P(C_r|C_1 \dots C_{r-1}) \quad (4.19)$$

$$\geq \prod_{i=1}^r P(C_i) \quad (4.20)$$

Hence,

$$R(p) \geq \prod_{i=1}^r \left[ 1 - \prod_{j \in cut_i} (1 - p_j) \right] \quad (4.21)$$



So, bounds on reliability function are given as:

$$\prod_{i=1}^s \left[ 1 - \prod_{j \in \text{cut}_i} (1 - p_j) \right] \leq R(p) \leq 1 - \prod_{i=1}^r \left[ 1 - \prod_{j \in \text{path}_i} p_j \right] \quad (4.22)$$

Considering the same example as shown in Figure 4.1(a), the reliability bounds are expressed as:

$$\begin{aligned} & (1 - (1 - p_{m2})(1 - p_{m3}))(1 - (1 - p_{m1})(1 - p_{m3})) \\ & (1 - (1 - p_{m1})(1 - p_{m2})(1 - p_{m3})) \\ & \leq R(p) \leq 1 - (1 - p_{m1}p_{m2})(1 - p_{m3}) \end{aligned} \quad (4.23)$$

Substituting  $p_{mi} = 0.5 \ \forall i = 1, 2, 3$  we get:

$$0.4922 \leq R(p) \leq 0.6250 \quad (4.24)$$

These reliability bounds match the values computed in column 4 of Table 4.1. This example can be generalized for nanocells consisting of more than three nanoparticles and molecules and similar results can be obtained.

## 4.1.2 Experimental setup for Reliability Prediction in Spatial Domain

We have proposed a Nanocell Reliability Predication Algorithm (NRPA) in this subsection. This Algorithm 4 computes set of all minimal paths connecting the input node to the output node of a nanocell and return as an output, the probability  $P_{\text{path}}$  of at least one path being present between input and output node. The NRPA algorithm is based on probabilistic analysis of nanocell in spatial domain and it is im-

---

**ALGORITHM 4:** Nanocell Reliability Prediction Algorithm (NRPA)

---

```

1: inputs
2:    $N$  := Number of Nanoparticles,
3:    $IP$  := Primary Input Node ,
4:    $OP$  := Primary Output Node,
5:    $P_{mol}$  := Probability of a molecule being present between two nanoparticles and
      found in 'ON' state
6: outputs
7:    $P_{path}$  := Probability that at least one path exists between  $IP$  and  $OP$ ,
8: do
9:   Initialization.
10:  Generate a  $N \times N$  random array (Adjacency Matrix) with Gaussian Distribution
      ( $\mu = 1, \sigma = 0.9$ ); Adj_Matrix = Generate_Matrix( $N, N, \mu, \sigma$ ).
12:  Convert this Adjacency Matrix to Edge Matrix;
13:   Edge_Matrix = Convert_to_EdgeMatrix(Adj_Matrix).
14:  Remove self loops and backward pointing edges from this matrix;
15:   Fwd_pointing_array = Generate_Dag(Edge_Matrix).
16:  Calculate Number of Molecules, denoted as ' $M$ ';
17:    $M = \text{size}(\text{Fwd\_pointing\_array}, 1)$ .
18:  Add weight to each connected edge =  $P_{mol}$  and unconnected edge =  $\infty$  of
      Fwd_pointing_array;
19:   Nanocell_Mat[ $N$ ][ $N$ ] = Weighted_Matrix(Fwd_pointing_array,  $P_{mol}$ ).
20:  Calculate  $k$  shortest paths from  $IP$  to  $OP$  using modified Dijkstra algorithm;
21:   shortest_path = Dijkstra_k(Nanocell_Mat,  $IP, OP, k$ ).
22:  Assign probabilities to each molecule;
23:    $P_M = \text{Assign_Prob}(P_{mol})$ .
24:  Calculate probability  $P_{path}$ , using equation (4.9);
25:    $P_{path} = \text{Compute_Prob}(\text{shortest\_path}, P_M)$ .
26:  return ( $P_{path}$ )

```

---

plemented in MATLAB. The parameters such as number of nanoparticles ( $N$ ), input ( $IP$ ) and output ( $OP$ ) terminals of the nanocell and the probability of presence of a molecule and subsequently of being found in ' $ON$ ' state between two nanoparticles ( $P_{mol}$ ) are given as an input to the algorithm. First, a graph  $G(V,E)$  for the nanocell is generated. The nodes of the graph are the nanoparticles and directed edges are the molecules, which points from the input to the output node. It is assumed that the molecules are distributed with Gaussian Distribution within the nanocell. As we assume that, between two nanoparticles only one molecular connection is present and these molecules are directed from input to the output. Thus, all the multiple edges and self loops are removed from the graph to obtain a directed acyclic graph in the step 4 of the NRPA algorithm. Thereafter, the probability  $P_{mol} = 0.5$  is assigned to each molecule. Further, the Dijkstra's shortest path algorithm is modified and utilized to find set of all paths from the input node to the output node in step 7. This NRPA algorithm finally computes the probability that at least a path is present between input and output node of the nanocell device, denoted as  $P_{path}$ , using equation (4.9).

A single input single output memory element is considered as a test case for exemplifying the algorithm. The number of nanoparticles are increased from 1 to 20. For each case, 10000 samples are generated using NRPA algorithm. It has been observed that  $P_{path}$  approaches to unity with increase in number of connected paths between input and output nodes. The number of connected paths depend on  $N$  and  $M$ . With 20 nanoparticles,  $P_{path}$  is close to one, for majority of the samples. The simulation results for nanoparticles from 5 to 20 are summarized as boxplots in Figure 4.2. Further, Table 4.2 depicts the simulation results for 1 to 20 nanoparticles. Theoretically, it can be proved from (9.10) of [85] that the probability of a random

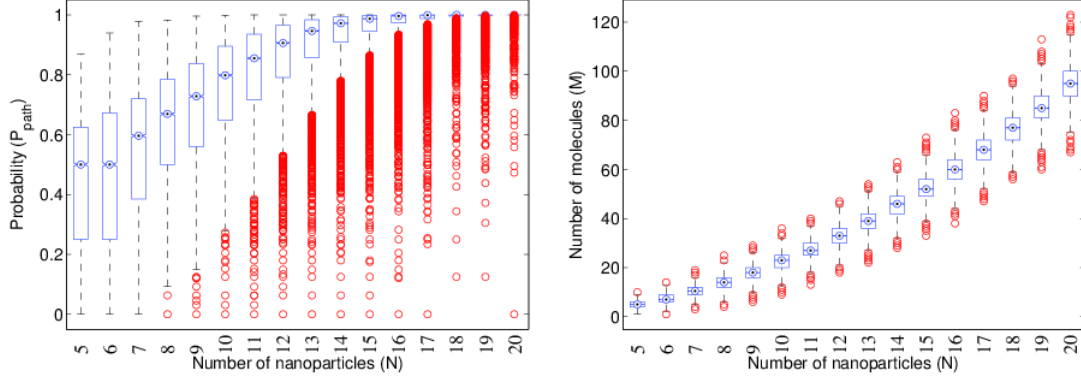


Figure 4.2: Simulation results for samples generated using NRPA algorithm (a) Probability of existence of at least one minimal path in a nanocell ( $P_{path}$ ) (b) Number of molecules ( $M$ ) in a nanocell. Here, (i) the central mark is the median (ii) the edges of the box are 25<sup>th</sup> and 75<sup>th</sup> percentiles (iii) the whisker extend to the most data points which are not considered as outliers (iv) the unfilled dots represent the outliers

graph with  $n$  nodes to be connected is given by:

$$1 - P_n \approx nq^{n-1} \text{ as } n \rightarrow \infty$$

For  $n = 20$  and  $q = 0.5$ , the probability  $P_n$  is computed as 0.99998.

Further, the NRPA algorithm is augmented to compute the upper and lower bounds on reliability using equation (4.22). The number of nanoparticles are varied from 7 to 15 and using modified NRPA algorithm 5000 samples of nanocell, for each value of  $N$ , are generated. For these samples upper and lower bounds on probability the nanocell reliability are computed. The boxplots for the reliability bounds are shown in Figure 4.3. It is observed that as the number of nanoparticles increase, the upper bound on reliability approaches to one and for most of the nanocell samples, the lower bound on reliability is close to 0.25. Here, we have assumed that  $P_{mol} = 0.5$ . However these bonds on reliability will vary with the probability of the molecule, that is,  $P_{mol}$ .

Table 4.2: Simulation results for path probability ( $P_{path}$ ) and number of molecules ( $M$ ) w.r.t. number of nanoparticles ( $N$ ) in a nanocell

| No. of nano-particles per Nanocell ( $N$ ) | Probability that at least a path exists ( $P_{path}$ ) |        |        | No. of molecules per Nanocell ( $M$ ) |      |      |
|--|--|--------|--------|---------------------------------------|------|------|
|  | min.   | avg.   | max.   | min.                                  | avg. | max. |
| 1  | 0.0000   | 0.0000 | 0.0000 | 0                                     | 0    | 0    |
| 2  | 0.0000   | 0.5000 | 0.5000 | 0                                     | 1    | 1    |
| 3  | 0.0000   | 0.3434 | 0.6250 | 1                                     | 2    | 3    |
| 4  | 0.0000   | 0.3654 | 0.7542 | 1                                     | 3    | 6    |
| 5  | 0.0000   | 0.4111 | 0.8679 | 1                                     | 5    | 10   |
| 6  | 0.0000   | 0.4714 | 0.9390 | 1                                     | 7    | 14   |
| 7  | 0.0000   | 0.5412 | 0.9781 | 3                                     | 10   | 19   |
| 8  | 0.0000   | 0.6100 | 0.9832 | 4                                     | 14   | 25   |
| 9  | 0.0000   | 0.6712 | 0.9961 | 6                                     | 18   | 29   |
| 10   | 0.0000   | 0.7434 | 0.9999 | 9                                     | 22   | 36   |
| 11   | 0.0000   | 0.7969 | 1.0000 | 13                                    | 27   | 40   |
| 12   | 0.0000   | 0.8473 | 1.0000 | 18                                    | 32   | 47   |
| 13   | 0.0000   | 0.8909 | 1.0000 | 22                                    | 40   | 54   |
| 14   | 0.0000   | 0.9234 | 1.0000 | 28                                    | 45   | 63   |
| 15   | 0.0000   | 0.9472 | 1.0000 | 33                                    | 52   | 73   |
| 16   | 0.0000   | 0.9669 | 1.0000 | 38                                    | 60   | 83   |
| 17   | 0.0000   | 0.9783 | 1.0000 | 47                                    | 70   | 90   |
| 18   | 0.1250   | 0.9871 | 1.0000 | 56                                    | 76   | 97   |
| 19   | 0.1250   | 0.9924 | 1.0000 | 60                                    | 85   | 113  |
| 20   | 0.0000   | 0.9952 | 1.0000 | 67                                    | 95   | 123  |

In this section, we have discussed the probabilistic modeling and analysis of nanocell in spatial domain. It is observed that the presence of at least one minimal path from input to output is a necessary condition for correct functioning of the device. Again, it is sufficient to have multiple minimal paths to obtain the desired output. These multiple paths add redundancy to the nanocell device and thus make it defect tolerant. It is observed that the probability of the existence of at least one path from input to output approaches close to unity with presence of 20 or more nanoparticles in a nanocell. Also, the bounds on reliability are computed. In this section, we have assumed that a molecule is present between two nanoparticles with a constant prob-

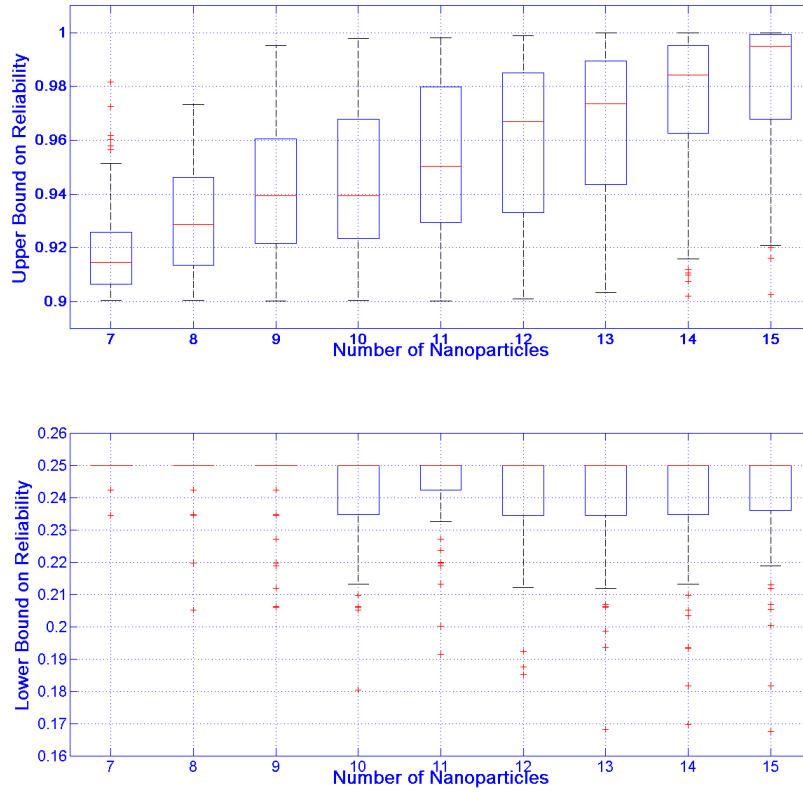


Figure 4.3: Simulation results for samples generated using NRPA algorithm. (a) Upper bound on probability  $P_{path}$  (b) Lower bound on probability  $P_{path}$ . Here, (i) the central red mark is the median (ii) the edges of the box are 25<sup>th</sup> and 75<sup>th</sup> percentiles (iii) the whisker extend to the most data points which are not considered as outliers

ability of 0.5. But, in reality, this probability ( $P_{mol}$ ) will follow some distribution and it will depend on time as well as environmental uncertainties. Therefore, in next section, we discuss the probabilistic modeling and analysis of nanocell in temporal domain, while considering the transient errors.

## 4.2 Reliability Analysis of Nanocell in Temporal Domain

### 4.2.1 Probabilistic Modeling of Nanocell

Consider a nanocell having  $N$  nanoparticles connected via  $M$  molecular switches. Let  $\chi(t)$  denote the number of 'ON' molecules at time  $t$ . The functional behavior of the nanocell depends on  $\chi(t)$  as well as on the combinations in which these molecules are connected. This is also called spatial connectivity of the molecules. Let at any time  $t$ ,  $\chi(t) = m$ , where  $m \leq M$ . Then, there are  ${}^M C_m$  ways in which  $m$  out of  $M$  molecules are 'ON' at time  $t$ .

Further, assume that at any time  $t$ , the nanocell is in super-state  $S_j$  if  $(M - j)$  molecules are 'ON' (or  $j$  molecules are 'OFF') and  $S_j^k, \forall k = 1, 2, \dots, a_j$  are a set of sub-states of this super-state. In other words, the sub-states of super-state  $S_j$  denote the combinations in which these  $(M - j)$  molecules are 'ON' and there can be  $a_j = {}^M C_j = {}^M C_{M-j}$  possible combinations. The transient errors can occur at any instant of time, because of which the system can transit from a given state to another state. In this way, the nanocell can be modeled as a continuous time birth - death process. Let the state space of this process be  $I = \{0, 1, 2, \dots, M\}$  and  $T = \{t | 0 \leq t < \infty\}$  be its parameter space. Thus, as shown in Fig. 4.4(b), at any time  $t$ , nanocell in one of the sub-states  $S_j^k, k = 1$  to  $a_j$  of a super-state  $S_j$ , can make:

1. a *down* transition to one of the  $(M - j)$  out of  ${}^M C_{j+1}$  sub-states of  $S_{j+1}$  super-state, with failure rate  $\lambda_j$ .
2. an *up* transition to one of the  $j$  out of  ${}^M C_{j-1}$  sub-states of  $S_{j-1}$  super-state, with repair rate  $\mu_j$ .

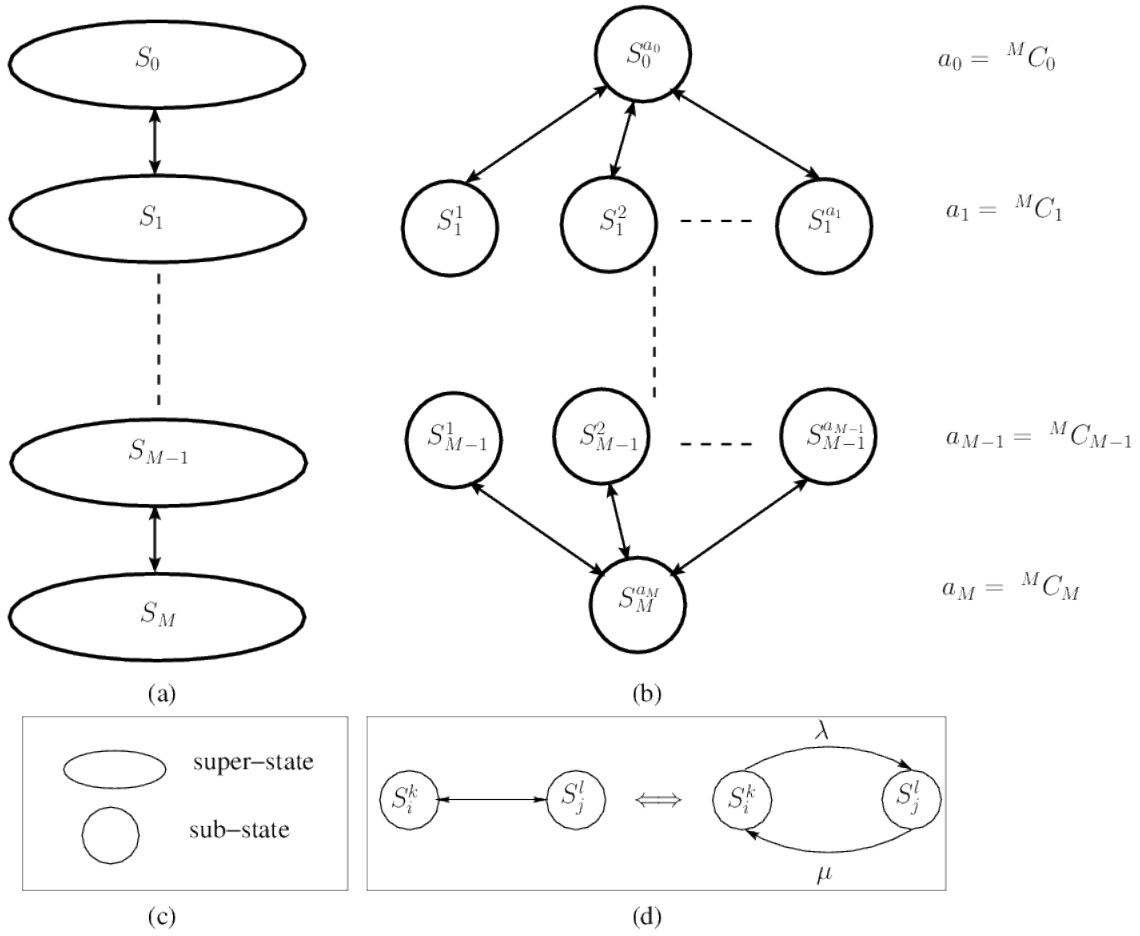


Figure 4.4: Continuous time birth - death model for Nanocell (a) abstract model with only super-states (b) detailed model with sub-states (c) Here, each circle represents a sub-state and a set of sub-states at each level combine to form a super-state, which is represented by an ellipse (d) bidirectional arrows between these states represents two unidirectional arrows, one for failure and another for repair.

So,  $(M - j) + (j) = M$  edges are connected to each sub-state. As shown in Fig. 4.4(d), each bidirectional arrow between sub-states represents two unidirectional arrows, one for failure and another for repair. Hence, the total in-degree and out-degree for each sub-state is  $2 \times M$ . Also, each sub-state of  $S_j$  has  $j$  parent sub-states and  $(M - j)$  child sub-states. The total number of super-states are  $M + 1$  and sub-states are  $\sum_{j=0}^M {}^M C_j = 2^M$ . The Fig. 4.7 in sub section 4.2.3 shows the state transition model for an example nanocell consisting of 5 molecules and Fig. 4.5 depicts this



example nanocell.

Let the repair rate for all the states be equal and be denoted by  $\mu$ . Also, let the failure rate for all states be equal and be denoted by  $\lambda$ . We here define a non-negative continuous function  $q_j(t)$  defined by:

$$\begin{aligned}
 q_j(t) &= -\frac{\partial}{\partial t} p_{jj}(v, t)|_{v=t} \\
 &= \lim_{h \rightarrow 0} \frac{p_{jj}(t, t) - p_{jj}(t, t+h)}{h} \\
 &= \lim_{h \rightarrow 0} \frac{1 - p_{jj}(t, t+h)}{h}, \quad \text{since } p_{jj}(t, t) = 1 \quad (4.25)
 \end{aligned}$$

Here,  $p_{jj}$  is defined as the conditional probability of being in same state  $j$ . Similarly, for each  $j$ , ( $j \neq k$ ) there is a non-negative continuous function  $q_{jk}(t)$  defined by:

$$\begin{aligned}
 q_{jk}(t) &= \frac{\partial}{\partial t} p_{jk}(v, t)|_{v=t} \\
 &= \lim_{h \rightarrow 0} \frac{p_{jk}(t, t+h) - p_{jk}(t, t)}{h} \\
 &= \lim_{h \rightarrow 0} \frac{p_{jk}(t, t+h)}{h}, \quad \text{since } p_{jk}(t, t) = 0 \quad (4.26)
 \end{aligned}$$

The  $q_{jk}(t)$  is also known as transition rate and  $p_{jk}$  denote the conditional transition probability from state  $j$  to  $k$ . Then, the transition rate and the transition probabilities are related by:

$$p_{jk}(t, t+h) = q_{jk}(t) \cdot h + o(h), \quad j \neq k; \quad (4.27)$$

$$p_{jj}(t, t+h) = 1 - q_j(t) \cdot h + o(h), \quad j = k; \quad (4.28)$$

Here,  $o(h)$  denote the probability that two or more transitions occur in time  $(t, t+h]$ . Also, for this birth-death model, the transition rates for each sub-state are related to

their failure and repair rates by following equations:

$$q_j = \lambda + \mu \quad (4.29)$$

$$q_{j,j-1} = \mu \quad (4.30)$$

$$q_{j,j+1} = \lambda \quad (4.31)$$

Now, consider that at time  $t$ , the system is in sub-state  $k$  of super-state  $j$  (i.e.,  $S_j^k$ ). Then, using equations (4.28) and (4.29) conditional probability that the system will remain in same sub-state at time  $(t+h]$  is:

$$p_{jj}(t, t+h) = 1 - (j\mu + (M-j)\lambda) \cdot h + o(h) \quad (4.32)$$

Similarly, using equations (4.27), (4.30) and (4.31), the conditional probability that system makes up or down transitions to  $S_j^k$  in the interval  $(t+h]$  is given by:

$$p_{j+1,j}(t, t+h) = (M-j)\mu \cdot h + o(h) \quad (4.33)$$

$$p_{j-1,j}(t, t+h) = j\lambda \cdot h + o(h) \quad (4.34)$$

Then, using equations (4.32), (4.33) and (4.34), the total probability that the nanocell is in sub-state  $S_j^k$  at time  $(t+h)$  is computed as:

$$\begin{aligned}
 P(\chi(t+h) = j) &= P_j(t+h) \\
 &= P_j(t)p_{jj}(t, t+h) + P_{j-1}(t)p_{j-1,j}(t+h) + \\
 &\quad P_{j+1}(t)p_{j+1,j}(t+h) + o(h) \\
 &= P_j(t)[1 - (j\mu + (M-j)\lambda).h + o(h)] + \\
 &\quad P_{j-1}(t)[j\lambda.h + o(h)] + P_{j+1}(t)[(M-j)\mu \\
 &\quad + o(h)] + o(h)
 \end{aligned} \tag{4.35}$$

Dividing both sides of equation (4.35) by  $h$  and taking  $\lim_{h \rightarrow 0}$ , we obtain:

$$\begin{aligned}
 \frac{d}{dt}P_j(t+h) &= -(j\mu + (M-j)\lambda)P_j(t) + j\lambda P_{j-1}(t) \\
 &\quad + (M-j)\mu P_{j+1}(t)
 \end{aligned} \tag{4.36}$$

Thus, we can write differential equation for initial and final state as:

$$\frac{d}{dt}P_0(t) = -M\lambda P_0(t) + M\mu P_1(t) \tag{4.37}$$

$$\frac{d}{dt}P_M(t) = -M\mu P_M(t) + M\lambda P_{M-1}(t) \tag{4.38}$$

#### 4.2.1.1 Steady state probability

Let the derivative  $dP_j(t)/dt = 0$ , then the steady-state probability that the system is in state  $k$ , is denoted by  $p_k$ , where

$$p_k = \lim_{t \rightarrow \infty} P_k(t) \tag{4.39}$$

Substituting this steady-state probability in differential-difference equation (4.37), we obtain:

$$0 = -M\lambda p_0 + M\mu p_1 \quad (4.40)$$

$$p_1 = \frac{\lambda}{\mu} p_0 \quad (4.41)$$

Similarly, from equation (4.38),

$$p_M = \frac{\lambda}{\mu} p_{M-1} \quad (4.42)$$

Then, the steady-state probability expression obtained from equation (4.36) can be rewritten as:

$$\begin{aligned} (M-j)\lambda p_j - (M-j)\mu p_{j+1} &= j\lambda p_{j-1} - j\mu p_j \\ \dots &= \lambda p_0 - \mu p_1 \end{aligned}$$

As we know,  $\lambda p_0 - \mu p_1 = 0$  (from equation (4.41)), we can write:

$$\begin{aligned} j\lambda p_{j-1} &= j\mu p_j \\ p_j &= \frac{\lambda}{\mu} p_{j-1} \\ p_j &= \left(\frac{\lambda}{\mu}\right) \left(\frac{\lambda}{\mu}\right) p_{j-2} \\ p_j &= \left(\frac{\lambda}{\mu}\right) \left(\frac{\lambda}{\mu}\right) \left(\frac{\lambda}{\mu}\right) p_{j-3} \end{aligned}$$

Thus, steady state or limiting probability of super-state  $S_j$  is given as:

$$p_j = \left(\frac{\lambda}{\mu}\right)^j p_0 \quad (4.43)$$

Since  $\sum_{k \geq 0}^M p_k = 1$ ,

$$\begin{aligned} \left(\frac{\lambda}{\mu}\right)^0 p_0 + \left(\frac{\lambda}{\mu}\right)^1 p_0 + \left(\frac{\lambda}{\mu}\right)^2 p_0 + \dots + \left(\frac{\lambda}{\mu}\right)^M p_0 &= 1 \\ \left[ \sum_{j \geq 0}^M \left(\frac{\lambda}{\mu}\right)^j p_0 \right] &= 1 \\ p_0 &= \frac{1}{\sum_{j \geq 0}^M \left(\frac{\lambda}{\mu}\right)^j} = \frac{1 - \rho}{1 - \rho^{M+1}} \end{aligned} \quad (4.44)$$

where  $\rho = \frac{\lambda}{\mu}$  and  $\rho \neq 1$ . If  $\rho = 1$ , then  $p_0 = \frac{1}{M+1}$ . Hence, we have derived the expression for steady state probability.

Using the properties of modified geometric distribution, mean and variance of number of molecules present in the nanocell is given as [84, 85]:

$$E[\chi] = \frac{\rho}{1 - \rho} \quad (4.45)$$

$$VAR[\chi] = \frac{\rho}{(1 - \rho)^2} \quad (4.46)$$

For system to be stable,  $\rho < 1$ , that is, or mean time to repair should be less than mean time to failure ( $\lambda < \mu$ ).

#### 4.2.1.2 Total probability being in a given sub-state

Let initial state of the system is  $S_0$ , then  $P_0(0) = 1$  and  $P_k(0) = 0$  for  $k \geq 1$ . Taking Laplace Transform of equations (4.37) and (4.38), [84] we get:

$$\begin{aligned}
 s\bar{P}_0(s) - P_0(0) &= -M\lambda\bar{P}_0(s) + M\mu\bar{P}_1(s) \\
 (s + M\lambda)\bar{P}_0(s) &= 1 + M\mu\bar{P}_1(s) \\
 \bar{P}_0(s) &= \frac{1}{s + M\lambda} + \frac{M\mu}{s + M\lambda}\bar{P}_1(s)
 \end{aligned} \tag{4.47}$$

Similarly,

$$\begin{aligned}
 s\bar{P}_M(s) - P_M(0) &= -M\mu\bar{P}_M(s) + M\lambda\bar{P}_{M-1}(s) \\
 (s + M\mu)\bar{P}_M(s) &= M\lambda\bar{P}_{M-1}(s) \\
 \bar{P}_M(s) &= \frac{M\lambda}{s + M\mu}\bar{P}_{M-1}(s)
 \end{aligned} \tag{4.48}$$

In this way, by taking Laplace Transform of equation (4.36), we get the generalized equation as:

$$\bar{P}_j(s) = \frac{j\lambda}{s + j\mu + (M - j)\lambda}\bar{P}_{j-1}(s) + \frac{(M - j)\mu}{s + j\mu + (M - j)\lambda}\bar{P}_{j+1}(s) \tag{4.49}$$

This equation can be solved to determine the time domain expression for probability. For example, consider that in a nanocell only two nanoparticles ( $N = 2$ ) are present connected by one molecule ( $M = 1$ ). So, there are only two super-states and  $2^1 = 2$  sub-states, namely  $S_0^1$  and  $S_1^1$ . As defined in Section 4.2.1,  $S_0^1$  and  $S_1^1$  denote that molecules is in '*ON*' and '*OFF*' states, respectively. The state diagram for this case can be represented by Fig. 4.4(d), by substituting  $j = k = l = 1$ ,  $i = 0$ . Substituting

$M = 1$  in equations (4.47), (4.48) and (4.49), we can derive the following expression:

$$\bar{P}_0(s) = \left( \frac{\mu}{\lambda + \mu} \right) \frac{1}{s} + \left( \frac{\lambda}{\lambda + \mu} \right) \frac{1}{s + \lambda + \mu} \quad (4.50)$$

Then, on inverting the transform to the time domain, we obtain:

$$P_0(t) = \left( \frac{\mu}{\lambda + \mu} \right) + \left( \frac{\lambda}{\lambda + \mu} \right) e^{-(\lambda + \mu)t} \quad (4.51)$$

since  $P_0 + P_1 = 1$ , we get

$$P_1(t) = \left( \frac{\lambda}{\lambda + \mu} \right) + \left( \frac{\lambda}{\lambda + \mu} \right) e^{-(\lambda + \mu)t} \quad (4.52)$$

In this way, we can derive the expression for total probability of a nanocell, being in a given state, for any value of  $M$ . We know that, the nanocell will function with reliability as long as at least one path is present between input and output nodes. For this, we have to find the sub-states at which at least one path is available and nanocell's probability of being in these states at time  $t$ . Then, compute their joint probability to derive the expression for reliability at time  $t$ . In contrast to this, we can also compute the probabilities of being '*ON*' and '*OFF*' of a single molecule (computed by equation (4.51) and (4.52)) and based on their spatial connectivity, we can derive the expression for upper and lower bounds on reliability, expected nanocell lifetime and nanocell availability. The mathematical framework for this approach is discussed in the next subsection.

## 4.2.2 Lifetime Analysis of Nanocell

### 4.2.2.1 Expected Lifetime

The system lifetime is defined as the time up to which the system is correctly functional [84, 85]. A system is comprised of one or more components and its lifetime is dependent on the lifetime of each of these individual components as well as on their spatial connectivity. For example, in our case, the nanocell will function correctly as long as one of the conducting path is present between its input and output node. Thus, it depends on (i) the lifetime of the individual molecule and (ii) their spatial arrangement. Let  $F_i(t)$ ,  $i = 1, 2, \dots, M$  denote the lifetime distribution of the molecule  $m_i$  and  $\bar{F}_i(t) = 1 - F_i(t)$ . Here, individual molecule can be modeled as a two-state continuous time birth-death model as shown in Fig. 4.4(d), by substituting  $j = k = l = 1$ ,  $i = 0$ . The lifetime of the molecule  $m_i$  is the duration up to which it is in 'ON' state, i.e.,  $S_0^1$ . We represent this for molecule  $m_i$  by probability  $P_0^i$ . Then, from equation (4.51), we write:

$$F_i(t) = P_0^i(t) = \left( \frac{\mu}{\lambda + \mu} \right) + \left( \frac{\lambda}{\lambda + \mu} \right) e^{-(\lambda + \mu)t} \quad (4.53)$$

Then, the reliability of the nanocell can be expressed by its molecule's lifetime, as:

$$R(\bar{F}(t)) = P(\text{Nanocell} > t) \quad (4.54)$$

where  $\bar{F}(t) = \{\bar{F}_1(t), \bar{F}_2(t), \dots, \bar{F}_M(t)\}$ . Then, by definition, the expected lifetime of a nanocell is:

$$E(\text{Nanocell}) = \int_0^{\infty} R(\bar{F}(t)) dt \quad (4.55)$$



Let  $\{path_1, path_2, \dots, path_s\}$  denote minimal path sets connecting input node to the output node and we define  $E_i, \forall i = 1 \dots s$  as  $E_i = \{\text{at least one molecular connection on path } path_i \text{ has failed}\}$ . If at least one of the molecules in the minimal path set has failed, the system will fail eventually. Mathematically, it is denoted as:

$$\begin{aligned} 1 - R(\bar{F}(t)) &= P(E_1 E_2 \dots E_s) \\ &= P(E_1) P(E_2 | E_1) \dots P(E_s | E_1 E_2 \dots E_{s-1}) \end{aligned}$$

Henceforth, it can be easily derived that failure of at least one molecule in the minimal path  $path_i$  can increase the probability of failure of at least one molecule in the path  $path_j$ . This would be the case if both the paths  $path_i$  and  $path_j$  overlap. So,

$$P(E_j | E_i) \geq P(E_j)$$

Similarly,

$$P(E_i | E_1 E_2 \dots E_{i-1}) \geq P(E_i)$$

Substituting in equations stated above we get,

$$1 - R(\bar{F}(t)) \geq \prod_i P(E_i)$$

or,

$$R(\bar{F}(t)) \leq R_{UB}(\bar{F}(t)) = 1 - \prod_i \left[ 1 - \prod_{j \in path_i} P_0^j \right] \quad (4.56)$$

In this way, we have derived the expression of upper bound on reliability of nanocell.

Let  $\{cut_1, cut_2, \dots, cut_r\}$  denote the minimal cut sets. We define the events  $C_1, C_2, \dots, C_r$

by  $C_i = \{\text{at least one molecular device in } cut_i \text{ is functioning}\}$ . Since, the nanocell will function iff all of the events  $C_i$  occur, we say,

$$\begin{aligned} R(\bar{F}(t)) &= P(C_1, C_2 \dots C_r) \\ &= P(C_1)P(C_2|C_1) \dots P(C_r|C_1 \dots C_{r-1}) \\ &\geq \prod_i P(C_i) \end{aligned}$$

Hence,

$$R(\bar{F}(t)) \geq R_{LB}(\bar{F}(t)) = \prod_i \left[ 1 - \prod_{j \in cut_i} (1 - P_0^j) \right] \quad (4.57)$$

As we know, the nanocell will work as long as at least one of the conducting paths is present between input and output node. Thus, we can rewrite the expression of expected lifetime of nanocell as:

$$E_{UB}(\text{Nanocell}) = \int_0^\infty \left[ 1 - \prod_i \left\{ 1 - \prod_{j \in path_i} F_j(t) \right\} \right] dt \quad (4.58)$$

$$E_{LB}(\text{Nanocell}) = \int_0^\infty \prod_i \left[ 1 - \prod_{j \in cut_i} (1 - F_j(t)) \right] dt \quad (4.59)$$

Thus, we get the lower and upper bounds on expected system lifetime for an example nanocell, as shown in Figure 4.5, we get

$$E_{UB}(\text{N1}) = \int_0^\infty \left[ 1 - \{1 - P_0^1(t)P_0^4(t)\} \{1 - P_0^3(t)\} \{1 - P_0^2(t)P_0^5(t)\} \right] dt \quad (4.60)$$

$$\begin{aligned} E_{LB}(\text{N1}) &= \int_0^\infty \left[ 1 - \{1 - P_0^1(t)\} \{1 - P_0^3(t)\} \{1 - P_0^2(t)\} \right] \left[ 1 - \{1 - P_0^1(t)\} \right. \\ &\quad \left. \{1 - P_0^3(t)\} \{1 - P_0^5(t)\} \right] \left[ 1 - \{1 - P_0^4(t)\} \{1 - P_0^3(t)\} \{1 - P_0^5(t)\} \right] \\ &\quad \left[ 1 - \{1 - P_0^4(t)\} \{1 - P_0^3(t)\} \{1 - P_0^2(t)\} \right] dt \end{aligned} \quad (4.61)$$

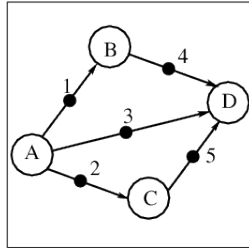


Figure 4.5: An example nanocell consisting of four nanoparticles and five molecular switches. Here, the unfilled (white) circles represent the nanoparticles and the filled (black) circles with arrows represent the molecular switches. The direction of the arrows denotes the current flow. The nanoparticles A and D are the input and output nodes, respectively.

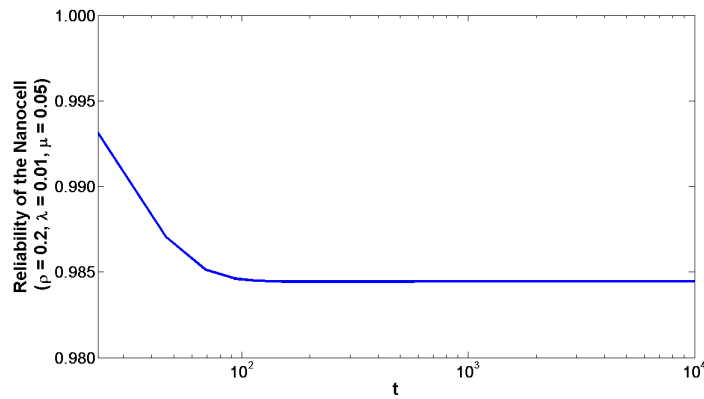


Figure 4.6: Upper bound on reliability ( $R_{UB}$ ), from  $t = 0$  to  $t = 10000$  units, for the example nanocell

Equation (4.56) denotes the upper bound on Reliability of a nanocell. On substituting  $P_0^i$  by 0.5 ( $\forall i = 1 \dots 5$ ) and  $\rho = 0.2$  in equation (4.53), we have obtained the probability that molecule  $m_i$  is in 'ON' state. Then, these probability values are used in equation (4.56) to plot the upper bound on reliability for the example nanocell, as shown in Figure 4.5. This  $R_{UB}$  is obtained by simulation (MATLAB) and Figure 4.6 is plotted for time  $t$  varying from 0 to 10000 units. It is observed from the plot that the reliability of the nanocell attains a constant value ( $= 0.984$ ) after  $t = 100$  units. In this example, the failure rate  $\lambda$  ( $= 0.01$ ) is less than the repair rate  $\mu$  ( $= 0.05$ ), thus the nanocell has very high reliability of 0.984 and maintains this value. How-

ever, this may not be the scenario when  $\lambda > \mu$ . Hence, another necessary condition for nanocell reliability is that the failure rate must be less than repair rate.

#### 4.2.2.2 Availability

The nanocell availability, at any time  $t$ , can be defined as the probability that it is functioning properly at  $t$  and it is denoted as:

$$A(t) = P\{\text{nanocell is working at } t\} \quad (4.62)$$

Since, all molecules act independently,  $A(t)$  can be expressed in terms of reliability function as follows:

$$A(t) = R(A_1(t), \dots, A_n(t)) \quad (4.63)$$

where

$$A_i(t) = P\{\text{molecule } i \text{ is 'ON' at } t\} \quad (4.64)$$

$$= \left( \frac{\mu_i}{\lambda_i + \mu_i} \right) + \left( \frac{\lambda_i}{\lambda_i + \mu_i} \right) e^{-(\lambda_i + \mu_i)t} \quad (4.65)$$

The equation (4.65) is obtained from equation (4.51). Assuming constant failure and repair rates for all molecules, i.e.,  $\lambda_i = \lambda$  and  $\mu_i = \mu$  for  $i = 1, 2, 3, \dots$ . Thus,

$$A_i(t) = \frac{\mu}{\lambda + \mu} + \frac{\lambda}{\lambda + \mu} e^{-(\lambda + \mu)t} \quad (4.66)$$

Then, limiting or steady-state availability for each molecule is

$$A_i = \lim_{t \rightarrow \infty} A_i(t) = r \left( \frac{\mu}{\lambda + \mu} \right) \quad (4.67)$$

Using the equations (4.56) and (4.57), we can determine the upper and lower bounds on nanocell availability as

$$\begin{aligned}
 A_{UB}(t) &= 1 - \prod_i \left[ 1 - \prod_{j \in path_i} P_0^j(t) \right] \\
 &= 1 - \prod_i \left[ 1 - \prod_{j \in path_i} \left\{ \frac{\mu_j}{\lambda_j + \mu_j} + \frac{\lambda_j}{\lambda_j + \mu_j} e^{-(\lambda_j + \mu_j)t} \right\} \right] \quad (4.68)
 \end{aligned}$$

and

$$\begin{aligned}
 A_{LB}(t) &= \prod_i \left[ 1 - \prod_{j \in cut_i} (1 - P_0^j(t)) \right] \\
 &= \prod_i \left[ 1 - \prod_{j \in cut_i} \left\{ \frac{\lambda_j}{\lambda_j + \mu_j} (1 - e^{-(\lambda_j + \mu_j)t}) \right\} \right] \quad (4.69)
 \end{aligned}$$

The bounds on the steady state availability of the Nanocell are given as

$$A_{UB} = 1 - \prod_i \left[ 1 - \prod_{j \in path_i} \left\{ \frac{\mu_j}{\lambda_j + \mu_j} \right\} \right] \quad (4.70)$$

$$A_{LB} = \prod_i \left[ 1 - \prod_{j \in cut_i} \left\{ \frac{\lambda_j}{\lambda_j + \mu_j} \right\} \right] \quad (4.71)$$

### 4.2.3 Experimental setup for Reliability Analysis in Time Domain

To exemplify the proposed model, consider a small nanocell which contains four nanoparticles ( $A, B, C, D$ ) connected via five molecular switches (1, 2, 3, 4, 5), as shown in Figure 4.5. There are three conducting paths which connects input node  $A$  to output node  $D$ , namely (i)  $\overrightarrow{ABD} = \{A - 1 - B - 4 - D\}$ , (ii)  $\overrightarrow{ACD} = \{A - 2 - C - 5 - D\}$ , and (iii)  $\overrightarrow{AD} = \{A - 3 - D\}$ . Let at  $t = 0$ , all five molecules are functioning.

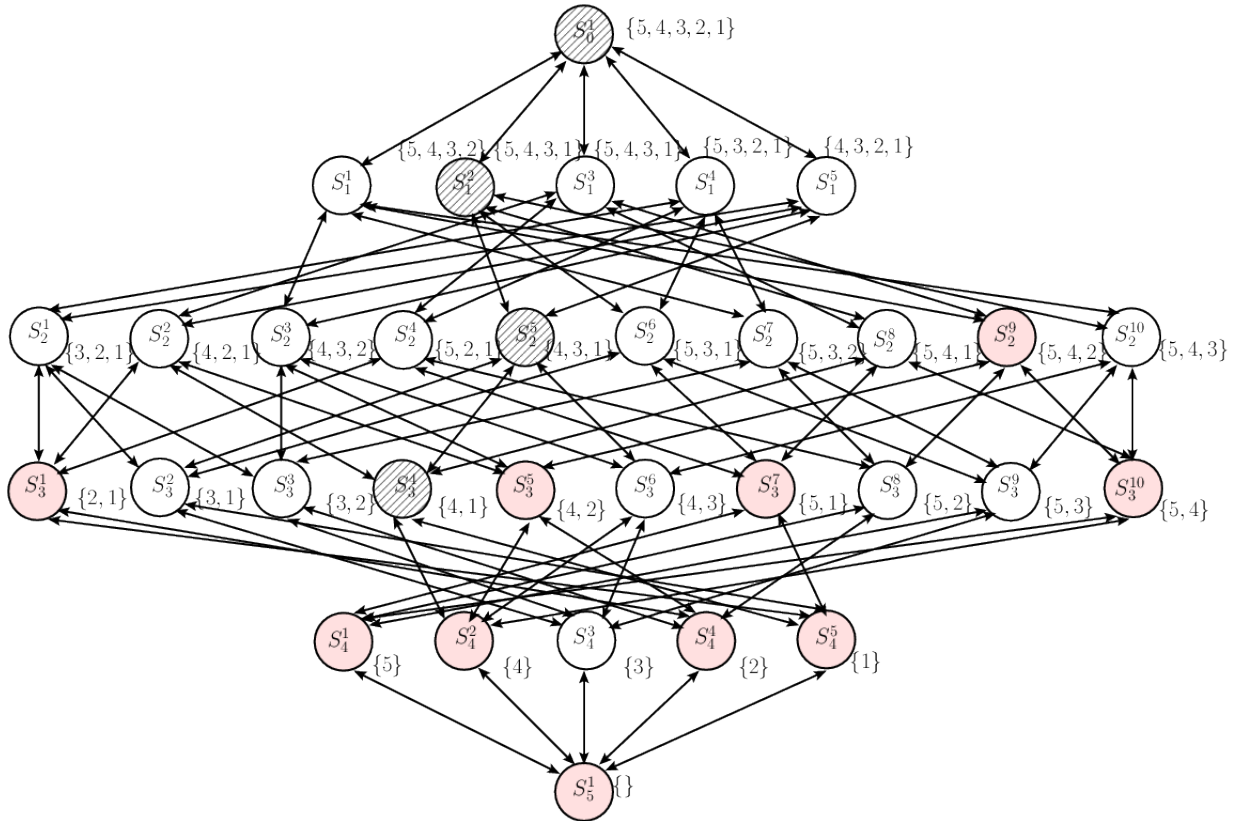


Figure 4.7: Representation of the example nanocell in the proposed model. Here, the circles represent the sub-states and the set  $\{i, j, \dots\}$  corresponding to each sub-state, represents the set of 'ON' molecules in that state. The filled sub-states are the *nanocell – failure – states* and remaining sub-states are the *nanocell – functioning – states*. The patterned filled circles represent one of the possible sequence in which the nanocell may make transition.

This means, the nanocell is in the initial state of the proposed model and this state is denoted by  $S_0^1$ . At next level, one of the five molecules may fail, due to some soft transient error. As stated earlier, the rate of failure is denoted by  $\lambda$  and rate of repair by  $\mu$ . When nanocell is in super-state  $S_1$ , it is actually in one of the  ${}^5C_1 = 5$  different sub-states. Similarly, at  $S_3$ ,  ${}^5C_2 = 10$  sub-states are possible, and so on. In all, there are six super-states and  $\sum_{j=0}^5 {}^5C_j = 2^5 = 32$  sub-states. Each of the super-states can make one-step up or down transition, but one-step transition among the sub-states at same level is not possible. However, initial and final states can make only down and up transitions, respectively. Thus, in general, with every sub-state, a set of parent and child sub-states are associated, and it can transit to only these states. This scenario for the example nanocell is depicted from the Figure 4.7. As shown in this figure, there are nine sub-states for which no path is present between input and output node of the nanocell. The light filled circles represent these sub-states and we will call these states as the *nanocell – failure – states*. The remaining 23 states are called *nanocell – functioning – states*. The steady state probability that the nanocell is in one of the super-state can be calculated by equations (4.43) and (4.44) derived in previous subsection.

First, this nanocell is modeled and it is simulated in HSPICE. The high ( $V_{high} = 2.0V$ ) and low ( $V_{low} = 0.5V$ ) voltages are applied to the input node  $A$  and received at the output node  $B$  with acceptable noise margin. That is, output voltage for read '1' is 1.9927 V and for read '0' is 0.4927 V. Thirty-two instances of this nanocell are generated, each depicting the behavior of one of these 32 sub-states. On simulating the nanocell instances corresponding to the nine *nanocell – failure – states* sub-states, zero voltage value is received at the output node.

Let at  $t_0$ , the nanocell is in  $S_0^1$  sub-state, i.e., all molecules are 'ON' initially. Then, corresponding nanocell instance is simulated in HSPICE. After some time  $t_1$ , due

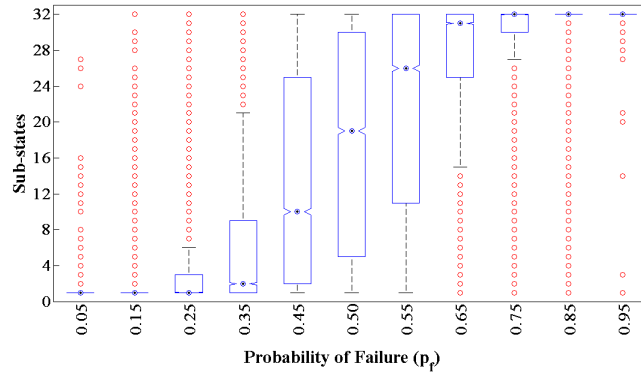


Figure 4.8: Simulation results for the sequence of sub-states of the example nanocell with constant  $\lambda$  and  $\mu$  in each case. The  $p_f$  is exponentially distributed and it is explored for different values between 0 to 1. One thousand Monte Carlo simulations are done for each case. Here, (i) the central mark is the median (ii) the edges of the box are 25<sup>th</sup> and 75<sup>th</sup> percentiles (iii) the whisker extend to the most data points which are not considered as outliers (iv) the unfilled dots represent the outliers.

to soft errors it may move to one of its child sub-state, say  $S_1^2$ , with probability of failure  $p_f$ . Again at  $t_2$ , it may move to either its parent state with repair rate  $\mu$  or to one of its child state (say  $S_3^4$ ) with failure rate  $\lambda$ . This transition path is represented by pattern filled circles in the Figure 4.7. The repair probability is represented by  $p_r$  and  $p_s = 1 - (p_f + p_r)$ , where  $p_s$  is probability that it will continue to be in same state. Here,  $p_f$  and  $p_r$  are exponentially distributed with mean  $\lambda$  and  $\mu$ , respectively and let  $p_s = 0$ . The  $p_f$  is explored for eleven different values between 0 to 1 and 10000 Monte Carlo Simulations are done in each case. The sequence of states and corresponding output voltages for read '1' and '0' are saved for each run. It is observed that, if  $p_f < p_r$  or  $\lambda < \mu$ , then the nanocell is in *nanocell – functioning – states* for most of the time. For these states, the acceptable output voltage is always received. We here define *success\_ratio* as the ratio of correct read '1' and '0' to the number of simulation. Hence, for  $\rho < 1$ , *success\_ratio* is close to unity. This is depicted from the Figure 4.8 and Figure 4.9.

This simulation procedure is discussed in Reliability Evaluation Algorithm for Nanocell



---

**ALGORITHM 5:** Extended Continuous Parameter Birth Death Model Generation for Nanocell (Model\_Generation)

---

```

1: inputs
2:    $N$  := Number of Nanoparticles in the nanocell,
3:    $M$  := Number of molecules in the nanocell,
4:    $IP$  := Primary Input Node,
5:    $OP$  := Primary Output Node,
6: outputs
7:    $NANOCELL[]$  := A set of Nanocell instances corresponding to each  $SUBSTATE$  ,
8:    $SUBSTATE[]$  := A set of substates in the nanocell model
9:    $PARENT[]$  := A set of parents corresponding to each  $SUBSTATE$  in the model,
10:   $CHILD[]$  :=A set of children corresponding to each  $SUBSTATE$  in the model,
11: do
12:   Call the subroutine NRPA( $N, IP, OP, 1$ );
13:   /* Generate an instance of a nanocell ( $NANOCELL$ ) consisting of  $N$  nanoparticles
      connected by  $M$  molecular switches using NRPA algorithm described in
      Algorithm 4 */
14:    $num = 0$ ;
15:   for  $i = 0$  to  $M$  do
16:     for  $j = 1$  to  ${}^M C_i$  do
17:        $SUBSTATE(num)$  = Generate a sub-state when  $(M - i)$  out of  $M$  molecules
          are present in 'ON' state;
18:        $NANOCELL(num)$  = Create an instance of nanocell corresponding to
           $SUBSTATE(i)$ ;
19:        $num = num + 1$ ;
20:     end for;
21:   end for;
22:   for  $i = 1$  to  $2^M$  do
23:      $PARENT(i)$  = Compute set of parents for  $SUBSTATE(i)$ ;
24:      $CHILD(i)$  = Compute set of children for  $SUBSTATE(i)$ ;
25:   end for;
26:   return ( $NANOCELL, SUBSTATE, PARENT, CHILD$ );

```

---

---

**ALGORITHM 6:** Reliability Evaluation Algorithm for Nanocell (REAN)

---

```

1: inputs
2:    $N$  := Number of Nanoparticles in the nanocell,
3:    $M$  := Number of molecules in the nanocell,
4:    $IP$  := Primary Input Node,
5:    $OP$  := Primary Output Node,
6:    $MAX\_RUNS$  := Maximum number of simulation runs;
7: outputs
8:    $success\_ratio$  := Number of times correct output is received for  $Max\_Num$ 
   simulations;
9:    $STATE\_SEQUENCE$  := Array of sequence of sub-states visited by nanocell in
    $MAX\_RUNS$  simulations;
10: do
11:   [ $NANOCELL$ ,  $SUBSTATE$ ,  $PARENT$ ,  $CHILD$ ] = Model_Generation( $N$ ,  $M$ ,  $IP$ ,
    $OP$ );
12:   Initialization;
13:   /* In the nanocell, (i) all  $M$  molecules are 'ON', (ii)  $STATE\_SEQUENCE(1)$  =
    $SUBSTATE(1)$ , (iii)  $present\_state=1$  */
14:   for  $k = 1$  to  $MAX\_RUNS$  do
15:     Simulate  $NANOCELL(SUBSTATE(present\_state))$ ;
16:      $DATAOUT(k)$  = output voltage of  $NANOCELL(SUBSTATE(present\_state))$ ;
17:      $STATE\_SEQUENCE(k)$  =  $present\_state$ ;
18:     Generate the value of  $p_f$  and  $p_r$  by exponential distribution using the Inverse
     Transform Method;
19:     Compute  $p_s = 1 - p_f - p_r$ ;
20:     if  $p_f + p_r \leq 1$  then
21:       if  $p_s > p_r$  and  $p_s > p_f$  then
22:         Stay in same state;
23:       else if  $p_r > p_f$  then
24:          $present\_state = PARENT(present\_state)$ , that is make an UP transition to
         any one of the parent sub-state;
25:       else
26:          $present\_state = CHILD(present\_state)$ , that is make an DOWN transition
         to any one of the child sub-state;
27:       end if
28:     else
29:       GOTO step 18;
30:     end if
31:   end for
32:    $success\_ratio$  = Compute number of times correct output is received using array
    $DATAOUT(1 : MAX\_RUNS)$ ;
33:   return ( $success\_ratio$ ,  $STATE\_SEQUENCE$ )

```

---

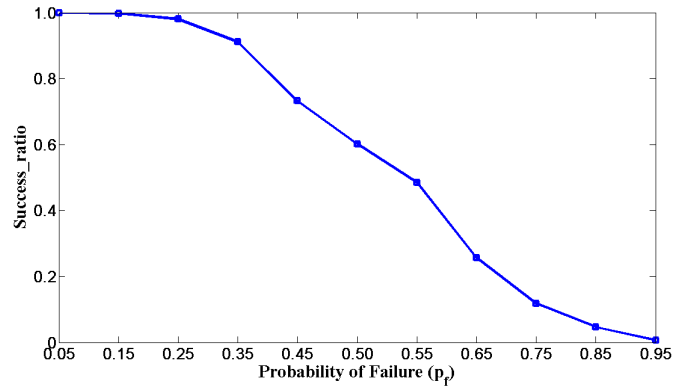


Figure 4.9: Simulation results for *success\_ratio* for different values of failure probability ( $p_f$ ).

(REAN) as shown in Algorithm 6. The Algorithm 6 first calls the Algorithm 5 to generate the birth death model for a nanocell having  $N$  nanoparticles. This *Model\_Generation* subroutine calls the *NRPA* algorithm 4 to generate a nanocell instance and outputs the DAG with  $N$  nanoparticles and  $M$  molecules. For this nanocell instance, Algorithm 5 computes the total number of sub-states as vector *SUBSTATE* and a nanocell instance corresponding to each of these sub-states as a vector *NANOCELL*. Also, a set of parents and children for each sub-state is calculated and saved as vectors *PARENT* and *CHILD*. Then, in REAN Algorithm 6, the nanocell instance corresponding to initial state is simulated and output voltage is stored in vector *DATAOUT*. The probabilities  $p_f$  and  $p_r$  are generated by exponential distribution using the Inverse Transform Method [85]. Then, if  $p_f > p_r$ , then we say, one of the molecule fail and nanocell instance corresponding to one of the child state is simulated in HSPICE. Otherwise, one of the molecule is assumed to be repaired and nanocell instance corresponding to one of the parent state is simulated. The vectors *STATE\_SEQUENCE* and *DATAOUT* are updated. The process is repeated to *MAX\_RUNS* times. In the end, we compute the *success\_ratio* which is defined as the number of times the correct output is received and for this we use the

Table 4.3: Simulation results for the example nanocell for different values of failure probability  $p_f$ . The *success\_ratio* represents the ratio of number of successful read '1' and '0' out of 10000 simulations. Upper and lower bounds on reliability are represented by  $R_{UB}$  and  $R_{LB}$ , respectively.

| $p_f$ | $\rho = \frac{\lambda}{\mu}$ | <i>success_ratio</i> | $R_{UB}$ | $R_{LB}$ |
|-------|------------------------------|----------------------|----------|----------|
| 0.05  | 0.02                         | 0.9998               | 0.9999   | 0.9999   |
| 0.15  | 0.09                         | 0.9973               | 0.9979   | 0.9978   |
| 0.25  | 0.21                         | 0.9826               | 0.9843   | 0.9816   |
| 0.35  | 0.41                         | 0.9109               | 0.9315   | 0.9099   |
| 0.45  | 0.75                         | 0.7318               | 0.8051   | 0.7204   |
| 0.55  | 1.34                         | 0.4851               | 0.6153   | 0.4352   |
| 0.65  | 2.44                         | 0.2565               | 0.4052   | 0.1711   |
| 0.75  | 4.82                         | 0.1178               | 0.2159   | 0.0348   |
| 0.85  | 11.67                        | 0.0464               | 0.0891   | 0.0023   |
| 0.95  | 58.40                        | 0.0064               | 0.0152   | 0.00001  |

array *DATAOUT*(1 : *MAX\_RUNS*). The *success\_ratio* and *STATE\_SEQUENCE* are returned as output. These algorithms are implemented in MATLAB, HSPICE and PERL.

Table 4.3 shows the *success\_ratio* obtained for each value of  $p_f$ . Also using the equations (4.56) and (4.57), the upper and lower bounds on reliability are computed in MATLAB, for  $t = 0$  to  $t = 10000$  units. It is observed that, *success\_ratio* obtained by using the proposed model lies within the range  $[R_{UB}, R_{LB}]$ . So, our proposed model computes the nanocell reliability efficiently. Also, any of the two methods, discussed here, can be used for estimating the nanocell lifetime.

Again, consider the same example nanocell. Assume that  $p_f$  and  $p_r$  are distributed exponentially and for each simulation run the values of  $\lambda$  and  $\mu$  change. Also, let the probability  $p_s = 1 - p_f - p_r$  and  $\rho < 1$ . Then, starting from the initial state, we perform simulations ten different times, each for 10000 runs. Then, as shown in the Figure 4.10, for each of the 10 cases, the median lies in the range (12,20).

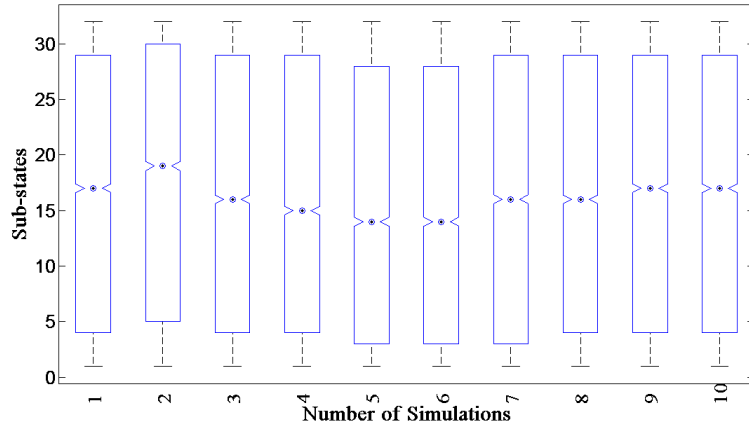


Figure 4.10: Simulation results for example nanocell for 10 different cases. For each case, simulations are done for 10000 times, with varying  $\lambda$  and  $\mu$ .

The sub-states for this range are *nanocell – functioning – states*. Further, as the number of nanoparticles ( $N$ ) and molecules ( $M$ ) will increase, the number of connected paths between input and output node will increase exponentially. Also, total number of sub-states and number of *nanocell – functioning – states* sub-states will increase. This will result in large state-space for the nanocell. Thus, with increase in number of molecules, the probability of transiting to *nanocell – failure – state* will decrease. For example if  $M = 20$ , total number of sub-states are  $2^{20} = 1048576$ , which are extremely large number of states. For such a system, the probability that at least one path is present between input to output is close to unity (pg. 592 of [85]). So, the number of successful read '1' and '0' increases exponentially. Hence, the nanocell reliability increases with increase in number of molecules. Hence, we can conclude that, under high variability and uncertainties, the nanocell will remain reliable and defect tolerant as long as  $\rho < 1$ .

In this section, the probabilistic modeling and analysis of Nanocell in temporal domain is presented. While considering the transient errors, the nanocell is modeled as an extended continuous parameter birth death model. The mathematical expression

for expected nanocell lifetime and its availability, in presence of transient errors is computed. On the basis of our model, an algorithm is developed and implemented in MATLAB, PERL and HSPICE, to automatically generate the proposed model representation for a given nanocell. It is used to estimate the success ratio as well as the nanocell reliability, while considering the uncertainties induced by transient errors. The theoretical results for reliability are validated by simulating HSPICE model of nanocell in presence of varying defect rates. It is observed that the device reliability increases with increase in the number of nanoparticles and molecules. A lower and upper bounds for nanocell reliability are calculated in theory which is validated in simulations.

### 4.3 Conclusions

In this chapter, a novel extended continuous parameter birth-death model is proposed to evaluate the reliability of a nanocell, in presence of transient errors. For this mathematical framework, the steady state probability and probability of being in each sub-state is computed. The proposed approach is extended to compute the expected lifetime and availability of the nanocell using the birth-death model of molecules and their spatial connectivity. On the basis of our model, an algorithm is developed and implemented in MATLAB, PERL and HSPICE, to automatically generate the proposed model representation for a given nanocell. It is used to estimate the *success\_ratio* as well as the nanocell reliability, while considering the uncertainties induced by transient errors. It is observed that as long as, molecular failure rate is less than its repair rate, the nanocell functions correctly. Also, with increase in number of molecules, the nanocell reliability increases. Thus, we can conclude that, a nanocell device remains defect tolerant and it works reliably in presence of transient errors as long as (i) at least one path is present between in-

put and output node, (ii) the failure rate of molecules is less than their repair rate. To satisfy the first condition, the number of nanoparticles must be greater than 20. Hence, we can argue that, at nano-scale, the nanocell device can function reliably and can withstand high defect rates due to transient errors.

In this Chapter, we have assumed that a molecule is present between two nanoparticles with a constant probability of 0.5. Also, the repair rate and failure rate are assumed to be constant for all molecules. But, in reality, these parameters will follow some distribution and their values will depend on time as well as environmental uncertainties. Also, more than one molecule may be present between two nanoparticles. Thus, lower and upper bounds on reliability will slightly deviate from our results.





## Chapter 5

### Conclusions and Future Work

The future memory devices are expected to be electrically accessible, higher density, higher speed, lower power, volatile and/or non-volatile RAM. Since, a molecule is the smallest component whose electrical properties can be engineered, it can be argued that the ultimate integrated circuit will be constructed at the molecular level. This fact has been the driving force behind molecular electronics research of recent times. In principle, one bit of information can be stored in the space of a single molecule and thus, extremely high density memories can be obtained. As all molecules of one type are identical, the molecular switches should have identical characteristics. This will reduce the problem of variability of components. Emerging molecular crossbar technology offers high density, regular array-like and non-volatile memory structure [7–12]. These devices consume extremely low power, offer low programming voltage and high switching speed. Non-volatility feature provided by these molecular devices, permits memory to be used as programmable elements within a logic device with high density.

In our work, we have developed the HSPICE as well as probabilistic models for nanocell molecular memory. An attempt has been made to develop a CAD tool for synthesis of such molecular memories which are posing interesting and promising

research challenges at futuristic cutting edge of technology spectrum. During exploration, it has been observed that the probability of existence of at least one path from input to output, approaches close to unity with presence of 20 or more nanoparticles in the nanocell. Due to hysteresis property of the OPE molecule, it has been observed during memory model validation that even an untrained nanocell depicts the behavior of 1-bit memory cell. Further, to train 2-bit molecular memory, number of control voltage signals must be more than one-fourth of total number of nanoparticles. The proposed methodology is versatile enough to train nanocell for multiple bit storage functionality. Extended Continuous Parameter Birth-Death model has been proposed to estimate the reliability of nanocell, in presence of transient errors.

## 5.1 Future Work

The molecular memory design and synthesis is viewed as a long term research goal. Although, the problem of nanocell molecular memory modeling, synthesis and analysis has been thoroughly explored in this thesis, there are still some potential improvements that should be explored.

A model is a conceptual notion that describes the system behavior. The molecular device model used in this thesis is based on empirical equations as proposed by [39–42]. In practical, the change in conduction states of the molecule depends on its conformational changes as discussed by [22,32]. Thus, it is a dire necessity to propose an enhanced as well as efficient molecular device model that captures these conformational changes. Also, it should be flexible enough to be used in nanocell or crossbar device model. Hence, the updated molecular memory model, will be close to realistic behavior of molecule.

Some of the assumptions made for nanocell modeling in this thesis are perhaps impractical. For example, we have assumed that between any two nanoparticles only

one molecule is present. However, when a self assembled monolayer of molecules is inserted between randomly placed gold nanoparticles, it will be quite impossible to have single molecular connection between two nanoparticles. Thus, in order to make a robust nanocell model, such an assumption must be removed. To analyze the effects of multiple edges and self loops in a nanocell has been left as a future work.

The aim of our experiment is to find proof of the concept of mortal training which to our understanding has not been tried yet. Hence, we have not focused on optimally tuning the parameters of Genetic Algorithm (GA). As explained in Section 3.3.2, the search space is very large and it takes number of days to find a near optimal solution. Hence, we used GA to find the best optimal solution in polynomial time for both the training algorithms. However, it is observed that the Genetic Algorithm consumes large amount of time for convergence, even for training a small nanocell of 50 nanoparticles. As a future work, we will explore adaptive learning algorithms to reduce the training time. The proposed mortal training needs to be applied for training large nanocell molecular memory.

To realize a nanocell device, several issues related to connection of macroscopic input-output pads to the 50 nm gold nanoparticles are required to be addressed. Instead of directly connecting them, we have proposed to use nanowires as shown in Figure 3.5 on page 51. The proposed hybrid CMOS-NANOCELL architecture needs to be realized. Again the issues related to connecting the two nanocells together must be addressed in near future. Although, the present CMOS read/write circuitry can be used for the proposed nanocell molecular memory, the nano-scale read/write circuitry must be proposed and realized.

We had tried to use Markov Random Field (MRF) based powerful probabilistic modeling technique for nanocell molecular memories. However, this modeling ap-

proach could not offer spatial as well as temporal domain analysis of naocell memory. Hence, we did not explore it further. The Markov Random Process in time and space is to be explored and remains part of our future work. Further, the extended continuous birth death model proposed in this thesis must be augmented to include effects of the aging and fabrication defects on the nanocell molecular memory reliability. Such a model can be used for estimating the data retention time of the nanocell memory. Also, the mathematical framework proposed in this thesis, needs to be validated by fabricated nanocell in presence of environmental uncertainties and aging effects.

# Bibliography

- [1] J. Tour, W. Van Zandt, C. Husband, S. Husband, L. Wilson, P. Franzon, and D. Nackashi, “Nanocell logic gates for molecular computing,” *IEEE Transactions on Nanotechnology*, vol. 1, pp. 100 – 109, jun 2002.
- [2] “International technology roadmap for semiconductors,” 2013.
- [3] A. Abdollahi, “Probabilistic decision diagrams for exact probabilistic analysis,” in *IEEE/ACM International Conference on Computer-Aided Design, (ICCAD 2007)*, pp. 266 –272, 4-8 2007.
- [4] T. Rejimon and S. Bhanja, “Scalable probabilistic computing models using Bayesian networks,” in *48th Midwest Symposium on Circuits and Systems*, pp. 712 –715 Vol. 1, 7-10 2005.
- [5] S. Krishnaswamy, G. F. Viamontes, I. L. Markov, and J. P. Hayes, “Accurate reliability evaluation and enhancement via probabilistic transfer matrices,” in *Design, Automation and Test in Europe Conference and Exposition (DATE 2005), 7-11 March 2005, Munich, Germany*, pp. 282–287, IEEE Computer Society, 2005.
- [6] R. Kumawat, V. Sahula, and M. Gaur, “Probabilistic modeling approaches for nanoscale devices,” *IEEE International Conference on Circuit, Power and Computing Technologies (ICCPCT 2013)*, pp. 720–724, March 2013.
- [7] A. Dehon, “Nanowire-based programmable architectures,” *J. Emerg. Technol. Comput. Syst.*, vol. 1, no. 2, pp. 109–162, 2005.
- [8] M. Stan, P. Franzon, S. Goldstein, J. Lach, and M. Ziegler, “Molecular electronics: from devices and interconnect to circuits and architecture,” *Proceedings of the IEEE*, vol. 91, pp. 1940–1957, Nov 2003.

## References

---

- [9] Y. Chen, D. A. A. Jung, G. and Ohlberg, X. Li, D. R. Stewart, J. O. Jeppesen, K. A. Nielsen, J. Fraser Stoddart, and R. S. Williams, “Nanoscale molecular-switch crossbar circuits,” *Nanotechnology*, vol. 14, no. 4, pp. 462–468.
- [10] A. DeHon, S. Goldstein, P. Kuekes, and P. Lincoln, “Nonphotolithographic nanoscale memory density prospects,” *IEEE Transactions on Nanotechnology*, vol. 4, no. 2, pp. 215 – 228, 2005.
- [11] W. Wu, G.-Y. Jung, D. Olynick, J. Straznicki, Z. Li, X. Li, D. Ohlberg, Y. Chen, S.-Y. Wang, J. Liddle, W. Tong, and R. S. Williams, “One-kilobit cross-bar molecular memory circuits at 30-nm half-pitch fabricated by nanoimprint lithography,” *Applied Physics A: Materials Science & Processing*, vol. 80, pp. 1173–1178, 2005. 10.1007/s00339-004-3176-y.
- [12] J. E. Green, J. Wook Choi, A. Boukai, Y. Bunimovich, E. Johnston-Halperin, E. DeIonno, Y. Luo, B. A. Sheriff, K. Xu, Y. Shik Shin, H.-R. Tseng, J. F. Stoddart, and J. R. Heath, “A 160-kilobit molecular electronic memory patterned at  $10^{11}$  bits per square centimeter,” *Nature*, vol. 445, pp. 414–417, 2007/01/25/print.
- [13] Z. Li, M. Pickett, D. Stewart, D. A. A. Ohlberg, X. Li, W. Wu, W. Robinett, and R. Williams, “Experimental demonstration of a defect - tolerant nanocrossbar demultiplexer,” *Nanotechnology*, vol. 19, no. 16, 2008.
- [14] K. Kelly and C. Mody, “The booms and busts of molecular electronics,” *Spectrum, IEEE*, vol. 52, pp. 52–60, October 2015.
- [15] A. Aviram and M. A. Ratner, “Molecular rectifiers,” *Chemical Physics Letters*, vol. 29, no. 2, pp. 277 – 283, 1974.
- [16] A. Aviram, “Molecules for memory, logic, and amplification,” *Journal of the American Chemical Society*, vol. 110, no. 17, pp. 5687–5692, 1988.
- [17] J. Chen, M. A. Reed, A. M. Rawlett, and J. M. Tour, “Large on-off ratios and negative differential resistance in a molecular electronic device,” *Science*, vol. 286, no. 5444, pp. 1550–1552, 1999.
- [18] J. Chen, W. Wang, M. A. Reed, A. M. Rawlett, D. W. Price, and J. M. Tour, “Room-temperature negative differential resistance in nanoscale molecular junctions,” *Applied Physics Letters*, vol. 77, no. 8, 2000.

## References

---

- [19] M. A. Reed, J. Chen, A. M. Rawlett, D. W. Price, and J. M. Tour, "Molecular random access memory cell," *Applied Physics Letters*, vol. 78, no. 23, 2001.
- [20] C. Husband, S. Husband, J. Daniels, and J. Tour, "Logic and memory with nanocell circuits," *IEEE Transactions on Electronic Devices*, vol. 50, pp. 1865 – 1875, sep 2003.
- [21] A. M. Moore, B. A. Mantooth, Z. J. Donhauser, F. Maya, D. W. Price, Y. Yao, J. M. Tour, and P. S. Weiss, "Cross-step place-exchange of oligo(phenylene ethynylene) molecules," *Nano Letters*, vol. 5, no. 11, pp. 2292–2297, 2005. PMID: 16277470.
- [22] N. Majumdar, N. Gergel-Hackett, J. Bean, L. Harriott, G. Pattanaik, G. Zangari, Y. Yao, and J. Tour, "The electrical behavior of nitro oligo(phenylene ethynylene)'s in pure and mixed monolayers," *Journal of Electronic Materials*, vol. 35, no. 1, pp. 140–146, 2006.
- [23] J. Han, J. Gao, Y. Qi, P. Jonker, and J. A. B. Fortes, "Toward hardware-redundant, fault-tolerant logic for nanoelectronics," *IEEE Design and Test*, vol. 22, no. 4, pp. 328–339, 2005.
- [24] J. Brown and R. Blanton, "CAEN-BIST: testing the nanofabric," in *Test Conference, 2004. Proceedings. ITC 2004. International*, pp. 462–471, Oct. 2004.
- [25] Z. Wang and K. Chakrabarty, "Built-in self-test and defect tolerance in molecular electronics-based nanofabrics," *J. Electron. Test.*, vol. 23, no. 2-3, pp. 145–161, 2007.
- [26] "<http://www.hugin.com/productservices/demo/hugin-lite>,"
- [27] R. Kumawat, V. Sahula, and M. S. Gaur, "Probabilistic modeling and analysis of molecular memory," *J. Emerg. Technol. Comput. Syst.*, vol. 11, pp. 6:1–6:16, Oct. 2014.
- [28] R. Kumawat, V. Sahula, and M. S. Gaur, "A probabilistic model for nanocell reliability evaluation in presence of transient errors," *IET Computers Digital Techniques*, vol. 9, pp. 213–220, July 2015.
- [29] M. Tsutsui and M. Taniguchi, "Single molecule electronics and devices," *Sensors*, vol. 12, pp. 7259–7298, May 2012.

## References

---

- [30] D. J. Wold, R. Haag, M. A. Rampi, and C. D. Frisbie, "Distance dependence of electron tunneling through self-assembled monolayers measured by conducting probe atomic force microscopy: Unsaturated versus saturated molecular junctions," *Journal of Physical Chemistry B*, vol. 106, pp. 2813–2816, march 2002.
- [31] R. A. Kiehl, "Information processing in nanoscale arrays: DNA assembly, molecular devices, nano-array architectures," in *2006 International Conference on Computer-Aided Design, ICCAD 2006, San Jose, CA, USA, November 5-9, 2006*, pp. 828–829, 2006.
- [32] H. Kim, S. S. Jang, R. A. Kiehl, and W. A. Goddard, "Negative differential resistance of oligo(phenylene ethynylene) self-assembled monolayer systems: The electric-field-induced conformational change mechanism," *The Journal of Physical Chemistry C*, vol. 115, no. 9, pp. 3722–3730, 2011.
- [33] C. Li, D. Li, G. Wan, J. Xu, and W. Hou, "Facile synthesis of concentrated gold nanoparticles with low size-distribution in water: temperature and ph controls," *Nanoscale Research Letters*, vol. 6, no. 1, 2011.
- [34] L. Polavarapu and Q.-H. Xu, "A simple method for large scale synthesis of highly monodisperse gold nanoparticles at room temperature and their electron relaxation properties," *Nanotechnology*, vol. 20, no. 18, p. 185606, 2009.
- [35] H. Chu, J. Wang, L. Ding, D. Yuan, Y. Zhang, J. Liu, and Y. Li, "Decoration of gold nanoparticles on surface-grown single-walled carbon nanotubes for detection of every nanotube by surface-enhanced Raman spectroscopy," *Journal of the American Chemical Society*, vol. 131, no. 40, pp. 14310–14316, 2009. PMID: 19764748.
- [36] S. Bhat and U. Maitra, "Facially amphiphilic thiol capped gold and silver nanoparticles," *Journal of Chemical Sciences*, vol. 120, no. 6, pp. 507–513, 2008.
- [37] A. Das, S. Das, and A. Raychaudhuri, "Growth of two-dimensional arrays of uncapped gold nanoparticles on silicon substrates," *Bulletin of Materials Science*, vol. 31, no. 3, pp. 277–282, 2008.



## References

---

- [38] J. Kimling, M. Maier, B. Okenve, V. Kotaidis, H. Ballot, and A. Plech, “Turkevich method for gold nanoparticle synthesis revisited,” *The Journal of Physical Chemistry B*, vol. 110, no. 32, pp. 15700–15707, 2006.
- [39] M. M. Ziegler and M. R. Stan, “A case for CMOS/nano co-design,” in *IEEE/ACM International Conference on Computer-Aided Design, (ICCAD 2002)*, pp. 348–352, 2002.
- [40] M. Ziegler and M. Stan, “CMOS/nano co-design for crossbar-based molecular electronic systems,” *IEEE Transactions on Nanotechnology*, vol. 2, pp. 217 – 230, dec. 2003.
- [41] G. Rose, M. Ziegler, and M. Stan, “Large-signal two-terminal device model for nanoelectronic circuit analysis,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 12, pp. 1201 –1208, nov. 2004.
- [42] G. Rose, Y. Yuxing, J. M. T., C. C. Adam, G. Nadine, M. Nabanita, C. B. John, R. H. Lloyd, and R. S. Mircea, “Designing CMOS/molecular memories while considering device parameter variations,” *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 3, no. 1, 2007.
- [43] P. Jha and V. Sahula, “Omnipotent and mortal training of a nanocell model to emulate the functionality of a logic gate,” in *Annual IEEE India Conference (INDICON), 2010*, pp. 1 –6, dec. 2010.
- [44] D. Bahrepour and M. J. Sharifi, “A novel SPICE compatible behavioral model for molecular electronics having hysteresis effects,” *Scientific Research and Essays*, vol. 9, no. 6, pp. 128–135, 2014.
- [45] Z. Zhong, D. Wang, Y. Cui, M. Bockrath, and C. M. Lieber, “Nanowire crossbar arrays as address decoders for integrated nanosystems,” *Science*, vol. 302, pp. 1377–1379, November 2003.
- [46] M. Ziegler and M. Stan, “CMOS/nano co-design for crossbar-based molecular electronic systems,” *IEEE Transactions on Nanotechnology*, vol. 2, pp. 217 – 230, dec. 2003.
- [47] A. Coker, V. Taylor, D. Bhaduri, S. Shukla, A. Raychowdhury, and K. Roy, “Multijunction fault-tolerance architecture for nanoscale crossbar memories,” *IEEE Transactions on Nanotechnology*, vol. 7, pp. 202 –208, march 2008.

## References

---

- [48] G. Csaba and P. Lugli, “Read-out design rules for molecular crossbar architectures,” *IEEE Transactions on Nanotechnology*, vol. 8, pp. 369–374, may 2009.
- [49] X. Duan, Y. Huang, Y. Cui, J. Wang, and C. M. Lieber, “Indium phosphide nanowires as building blocks for nanoscale electronic and optoelectronic devices,” *Nature*, vol. 409, pp. 66–69, 2001/01/04/print.
- [50] D. Wang, Q. Wang, A. Javey, R. Tu, H. Dai, H. Kim, P. C. McIntyre, T. Krishnamohan, and K. C. Saraswat, “Germanium nanowire field-effect transistors with  $\text{SiO}_2$  and high-k  $\text{HfO}_2$  gate dielectrics,” *Applied Physics Letters*, vol. 83, pp. 2432–2434, sep 2003.
- [51] J. Guilera, C. Fabrega, O. Casals, F. Hernandez-Ramirez, S. Wang, S. Mathur, F. Udrea, A. D. Luca, S. Z. Ali, A. Romano-Rodriguez, J. D. Prades, and J. R. Morante, “Facile integration of ordered nanowires in functional devices,” *Sensors and Actuators B: Chemical*, vol. 221, pp. 104–112, 2015.
- [52] G. Csaba and P. Lugli, “Read-out design rules for molecular crossbar architectures,” *IEEE Transactions on Nanotechnology*, vol. 8, pp. 369–374, may 2009.
- [53] J. Mustafa and R. Waser, “A novel reference scheme for reading passive resistive crossbar memories,” *IEEE Transactions on Nanotechnology*, vol. 5, pp. 687–691, November 2006.
- [54] K. M. M. Habib, A. Khitun, A. A. Balandin, and R. K. Lake, “Graphene nanoribbon crossbar nanomesh,” in *Proceedings of the 2011 IEEE/ACM International Symposium on Nanoscale Architectures*, NANOARCH ’11, (Washington, DC, USA), pp. 86–90, IEEE Computer Society, 2011.
- [55] M. Gholipour and N. Masoumi, “Graphene nanoribbon crossbar architecture for low power and dense circuit implementations,” *Microelectronics Journal*, vol. 45, no. 11, pp. 1533–1541, 2014.
- [56] I. Vourkas and GeorgiosCh.Sirakoulis, “Memristor-based combinational circuits: A design methodology for encoders/decoders,” *Microelectronics Journal*, vol. 45, pp. 59–70, 2014.

## References

---

- [57] M. Gholipour and N. Masoumi, “Design investigation of nanoelectronic circuits using crossbar-based nanoarchitectures,” *Microelectron. J.*, vol. 44, pp. 190–200, Mar. 2013.
- [58] C. M. Jeffery and R. J. Figueiredo, “Hierarchical fault tolerance for nanoscale memories,” *IEEE Trans. Nanotechnol.*, vol. 5, pp. 407–414, July 2006.
- [59] S. WANG, J. DAI, and L. WANG, “Hybrid redundancy for defect tolerance in molecular crossbar memory,” *J. Emerg. Technol. Comput. Syst.*, vol. 9, pp. 7:1–7:18, Oct. 2014.
- [60] M. Zamani, H. Mirzaei, and M. B. Tahoori, “ILP formulations for variation/defect-tolerant logic mapping on crossbar nano-architectures,” *J. Emerg. Technol. Comput. Syst.*, vol. 9, pp. 21:1–21:21, Oct. 2013.
- [61] R. I. Bahar, J. Mundy, and J. Chen, “A probabilistic-based design methodology for nanoscale computation,” in *International Conference on Computer-Aided Design*, pp. 480–486, 2003.
- [62] H. Derin and P. Kelly, “Discrete-index Markov-type random processes,” *Proceedings of the IEEE*, vol. 77, pp. 1485–1510, Oct 1989.
- [63] J. Besag, “Spatial interaction and the statistical analysis of lattice systems,” *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 36, no. 2, pp. 192–236, 1974.
- [64] J. S. Yedidia, W. T. Freeman, and Y. Weiss, “Understanding belief propagation and its generalizations,” in *Exploring artificial intelligence in the new millennium*, (San Francisco, CA, USA), pp. 239–269, Morgan Kaufmann Publishers Inc., 2003.
- [65] J. S. Yedidia, W. T. Freeman, and Y. Weiss, “Constructing free-energy approximations and generalized belief propagation algorithms,” *IEEE Transactions on Information Theory*, vol. 51, no. 7, pp. 2282–2312, 2005.
- [66] D. S. Ha, B. R. Member, D. Bhaduri, and D. Bhaduri, “Tools and techniques for evaluating reliability trade-offs for nano-architectures,” in *Nano, Quantum and Molecular Computing: Implications to High Level Design and Validation*, Kluwer Academic Publishers, 2004.

## References

---

- [67] D. Bhaduri and S. Shukla, “Nanolab: A tool for evaluating reliability of defect-tolerant nano architectures,” in *IEEE Computer Society Annual Symposium on VLSI*, pp. 03–09, Press, 2004.
- [68] M. S. Gaur, R. Narasimhan, V. Laxmi, and U. Kumar, “Structural fault modelling in nano devices,” in *NanoNet*, pp. 6–10, 2008.
- [69] K. Nepal, R. I. Bahar, J. Mundy, W. R. Patterson, and A. Zaslavsky, “Designing logic circuits for probabilistic computation in the presence of noise,” in *DAC '05: Proceedings of the 42nd annual Design Automation Conference*, (New York, NY, USA), pp. 485–490, ACM, 2005.
- [70] K. Nepal, R. I. Bahar, J. Mundy, W. R. Patterson, and A. Zaslavsky, “Optimizing noise-immune nanoscale circuits using principles of Markov Random Fields,” in *GLSVLSI '06: Proceedings of the 16th ACM Great Lakes symposium on VLSI*, (New York, NY, USA), pp. 149–152, ACM, 2006.
- [71] K. Nepal, R. I. Bahar, J. Mundy, W. R. Patterson, and A. Zaslavsky, “Designing MRF based error correcting circuits for memory elements,” in *DATE '06: Proceedings of the conference on Design, automation and test in Europe*, (3001 Leuven, Belgium, Belgium), pp. 792–793, European Design and Automation Association, 2006.
- [72] K. Nepal, R. I. Bahar, J. Mundy, W. R. Patterson, and A. Zaslavsky, “Designing nanoscale logic circuits based on Markov Random Fields,” *J. Electron. Test.*, vol. 23, no. 2-3, pp. 255–266, 2007.
- [73] R. Kumawat, V. Sahula, M. Gaur, and V. Laxmi, “Modeling and reliability evaluation of logic circuits at nanoscale,” *IEEE International Workshop on Reliability Aware System Design and Test (RASDAT 2010)*, Jan. 2010.
- [74] M. Soni, R. Kumawat, V. Sahula, and M. Gaur, “Reliability evaluation of redundancy based fault tolerant techniques at nanoscale,” *IEEE International Workshop on Reliability Aware System Design and Test (RASDAT 2012)*, Jan. 2012.
- [75] S. Bhanja and N. Ranganathan, “Switching activity estimation of VLSI circuits using Bayesian networks,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 11, pp. 558 – 567, aug. 2003.

## References

---

- [76] S. Bhanja and N. Ranganathan, “Cascaded Bayesian inferencing for switching activity estimation with correlated inputs,” *IEEE Transaction on VLSI System*, vol. 12, no. 12, pp. 1360–1370, 2004.
- [77] T. Rejimon and S. Bhanja, “An accurate probalistic model for error detection,” in *VLSID '05: Proceedings of the 18th International Conference on VLSI Design held jointly with 4th International Conference on Embedded Systems Design*, (Washington, DC, USA), pp. 717–722, IEEE Computer Society, 2005.
- [78] K. Lingasubramanian and S. Bhanja, “Probabilistic maximum error modeling for unreliable logic circuits,” in *GLSVLSI '07: Proceedings of the 17th ACM Great Lakes symposium on VLSI*, (New York, NY, USA), pp. 223–226, ACM, 2007.
- [79] T. Rejimon, K. Lingasubramanian, and S. Bhanja, “Probabilistic error modeling for nano-domain logic circuits,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 17, pp. 55 – 65, jan. 2009.
- [80] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., 1988.
- [81] D. T. Franco, M. C. Vasconcelos, L. Naviner, and J.-F. Naviner, “Signal probability for reliability evaluation of logic circuits,” *Microelectronics reliability*, vol. 48, no. 8-9, pp. 1586 – 1591, 2008.
- [82] R. Kumawat, V. Sahula, and M. Gaur, “Reliable circuit analysis and design using nanoscale devices,” *Proc. SPIE , International Conference on Communication and Electronics System Design (ICCESD 2013)*, vol. 8760, pp. 242–247, Jan. 2013.
- [83] J. Skoldberg, C. Onnheim, and G. Wendin, “Nanocell devices and architecture for configurable computing with molecular electronics,” *IEEE Transactions on Circuits and Systems*, vol. 54, pp. 2461 –2471, november 2007.
- [84] K. S. Trivedi, *Probability and Statistics with Reliability, Queuing and Computer Science Applications*. Prentice-Hall, twelfth ed., 2001.
- [85] S. M. Ross, *Introduction to Probability Models*. Academic Press, ninth ed., 2007.

## References

---

- [86] S. Ko, G. Han, and J. K. Lee, “Surface organic chemistry for application to organic electronics,” *Tetrahedron Letters*, vol. 56, no. 24, pp. 3721 – 3731, 2015.

## List of publications out of this thesis work

### In Peer-reviewed International Journals

- R. Kumawat, V. Sahula, and M. S. Gaur, "Probabilistic Model for Nanocell Reliability Evaluation in presence of Transient Errors", *IET Computers & Digital Techniques, Special Issue on "Designing with Uncertainty - Opportunities & Challenges"*, Vol. 9, No. 4, pp. 213-220, July 2015, ISSN 1751-8601, doi:10.1049/iet-cdt.2014.0124
- R. Kumawat, V. Sahula, and M.S. Gaur, "Probabilistic Modeling and Analysis of Molecular Memory", *ACM Journal on Emerging Technologies in Computing Systems(JETC)*, vol. 11, no. 1, pp. 6:1-6:16, September 2014, doi: <http://dx.doi.org/10.1145/2629533>

### In Peer-reviewed International Conferences

- Renu Kumawat, Vineet Sahula and Manoj Singh Gaur, "Modeling and Synthesis of Molecular Memory", *VLSI Design and TEST (VDAT) 2015, 19<sup>th</sup> International Symposium on*, pp. 1-2, June 26-29, 2015, doi: 10.1109/ISV-DAT.2015.7208081
- R. Kumawat, V. Sahula, and M. S. Gaur, "Probabilistic Modeling Approaches for Nanoscale Devices", *IEEE International Conference on Circuit, Power and Computing Technologies (ICCPCT-2013)*, vol., no., pp. 720-724, 20-21 March 2013, doi: 10.1109/ICCPCT.2013.6528997
- R. Kumawat, V. Sahula, and M. S. Gaur, "Reliable circuit analysis and design using nanoscale devices" *Proc. SPIE , International Conference on Commu-*

## References

---

- nication and Electronics System Design (ICCESD-2013)*, vol. 8760, no.1, pp. 242-247 , Jan. 2013, doi: 10.1117/12.2020905.
- M. Soni, R. Kumawat, V. Sahula, and M. S. Gaur, "Reliability Evaluation of Redundancy based Fault Tolerant Techniques at Nanoscale", *IEEE International Workshop on Reliability Aware System Design and Test (RASDAT-2012)*, vol. 3, no., pp. 37- 42, Jan. 2012.
  - R. Kumawat, V. Sahula, M. S. Gaur, and Vijay Laxmi, "Modeling and Reliability Evaluation of Logic Circuits at Nanoscale", *IEEE International Workshop on Reliability Aware System Design and Test (RASDAT-2010)*, vol. 1, no., pp. 67-72, Jan. 2010.



## **Bio-Data**

R. Kumawat has nine years of teaching experience and is currently working as an Assistant Professor I, in the Department of Electronics and Communication Engineering at Manipal University Jaipur, India. She has served as Assistant Professor (On Contract) under the "Special ManPower Development Project in the Field of VLSI Design (SMDP-VLSI-II)" sponsored by Ministry of Electronics & IT, Government of India, from Sept 2006 to March 2013, in the ECE department of Malaviya National Institute of Technology Jaipur. She obtained Bachelor's degree in Computer Engineering from Government Engineering College Ajmer, Rajasthan University and Master's degree in VLSI Design from Banasthali Vidyapeeth, Niwai in 2004 and 2006 respectively. She is currently pursuing Ph.D. at Malaviya National Institute of Technology Jaipur, India, in the field of Molecular Memory Design and Modeling. She has guided more than 6 Master's thesis and several undergraduate projects. She is currently student member of IEEE, USA and life member of IETE, India.

