Design, implementation, and test of next generation FPGAs using quantum-dot cellular automata technology

Tejas Raviraj

The University of Toledo

Follow this and additional works at: http://utdr.utoledo.edu/theses-dissertations

Recommended Citation
http://utdr.utoledo.edu/theses-dissertations/690

This Thesis is brought to you for free and open access by The University of Toledo Digital Repository. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of The University of Toledo Digital Repository. For more information, please see the repository’s About page.
A Thesis

entitled

Design, Implementation, and Test of Next Generation FPGAs Using Quantum-Dot Cellular Automata Technology

by

Tejas Raviraj

Submitted to the Graduate Faculty as partial fulfillment of the requirements for the Master of Science Degree in Electrical Engineering

Dr. Mohammed Y. Niamat, Committee Chair

Dr. Mansoor Alam, Committee Member

Dr. Ezzatollah Salari, Committee Member

Dr. Patricia R. Komuniecki, Dean
College of Graduate Studies

The University of Toledo

May 2011
Moore’s Law states that the number of transistors on a unit area doubles every 18 months. Till date, CMOS technology has been successfully keeping pace with Moore’s Law. However, it faces serious fundamental challenges in the future, and will soon reach a limit where the quantum effects will begin to dominate the device performance that make further scaling difficult. The International Technology Roadmap for Semiconductors (ITRS) predicts the size limit for CMOS technology to be limited in the range of 5 nm to 10 nm and believes this limit will be reached by 2017[1]. As the devices are exponentially scaled down various factors including power dissipation, gate leakage current, interconnection noise (introduction of crosstalk and hot electron effect) and stray capacitances will become potential bottlenecks to circuit performance. Thus, there is a pressing need for new technologies which can overcome these limitations more effectively. Researchers are investigating alternative technologies at the nano scale to replace CMOS technology in the future. Amongst the many technologies being
investigated, Quantum-Dot Cellular Automata (QCA) is one promising transistor-less technology.

This research investigates the design, implementation, and simulation of a nano Field Programmable Gate Array (FPGA) using the Quantum-Dot Cellular Automata. The first phase of the research focuses on modeling, implementation, and simulation of a Configurable Logic Block (CLB) slice for a nano quantum FPGA. The proposed design is compared with various nano FPGA based architectures and optimized with respect to area and latency. The second phase of the research focuses on implementing a novel Built-In Self Test model at the quantum level for testing the CLB. New fabrication faults in the QCA components of the CLB are modeled and tested. Finally, the configurable logic block slices are tested by incorporating BIST in its design. The design is modeled using standard QCA cells with multiple layers for reliable interconnect crossovers. The design of the configurable logic block is implemented and simulated using the QCA Designer software tool.
This thesis is dedicated to my loving family members, Mr. & Mrs. Raviraj, my sister Nikitha and my loving fiancé Rashmi.
Acknowledgements

I would like to thank Dr. Niamat for giving me an opportunity to work under his leadership and guide me with his valuable advice during the most difficult phases of my research. I would like to thank Mr. Dan Solarek and other faculty members of the Engineering Technology for supporting me financially during the completion of my Masters. I would like to thank Dr. Vemuru for his valuable suggestions during the completion of my research. I would also like to extend my regards to Dr. Alam and Dr. Salari for serving as committee members.

I wholeheartedly thank my Dad and Mom who were instrumental in me pursuing my dream of obtaining a Masters degree. They were the pillars of my strength, motivation and the persons whom I always looked up to for valuable advice. I would also like to thank my sister Nikitha and my fiancé Rashmi without whom this work would have been impossible. There never was a day in my life that passed without thinking of them.

I would like to thank Swetha Pappala who has been a great friend, motivator and was always available to make discussions on my research. I would also thank all my roommates who stood by me all the time and made my stay in US a pleasant and a memorable experience.
## Contents

Abstract ................................................................................................................................. III

Acknowledgements ................................................................................................................ VI

Table of Contents ................................................................................................................ VI

List of Tables ......................................................................................................................... XI

List of Figures ....................................................................................................................... XII

1 Introduction ......................................................................................................................... 1

1.1 Motivation ......................................................................................................................... 1

1.2 Research Objectives ......................................................................................................... 2

2 Quantum-dot Cellular Automata ....................................................................................... 6

2.1 Introduction ....................................................................................................................... 6

2.2 Quantum Cells .................................................................................................................. 6

2.3 Clocking in QCA ............................................................................................................... 8

2.4 QCA Logic Devices .......................................................................................................... 12

2.4.1 Binary Wires .................................................................................................................. 13

2.4.2 Majority Voter .............................................................................................................. 14

2.4.3 Inverter ......................................................................................................................... 17
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.5 Cross-Wiring in QCA</td>
<td>20</td>
</tr>
<tr>
<td>2.5.1 Coplanar Crossover</td>
<td>21</td>
</tr>
<tr>
<td>2.5.2 Multi-Layer Crossover</td>
<td>23</td>
</tr>
<tr>
<td>2.6 QCA Implementation</td>
<td>25</td>
</tr>
<tr>
<td>2.6.1 Metal QCA</td>
<td>25</td>
</tr>
<tr>
<td>2.6.2 Molecular QCA</td>
<td>28</td>
</tr>
<tr>
<td>2.6.3 Magnetic QCA</td>
<td>31</td>
</tr>
<tr>
<td>3 Evaluation of QCA CAD Tools</td>
<td>36</td>
</tr>
<tr>
<td>3.1 Introduction</td>
<td>36</td>
</tr>
<tr>
<td>3.2 QCA-LG Tool</td>
<td>36</td>
</tr>
<tr>
<td>3.2.2 Circuit Expansion</td>
<td>37</td>
</tr>
<tr>
<td>3.2.3 Gate Placement</td>
<td>38</td>
</tr>
<tr>
<td>3.2.4 Gate Shaping</td>
<td>38</td>
</tr>
<tr>
<td>3.2.5 Input Signals Distribution</td>
<td>39</td>
</tr>
<tr>
<td>3.2.6 Output Layout</td>
<td>39</td>
</tr>
<tr>
<td>3.3 Design Comparisons</td>
<td>40</td>
</tr>
<tr>
<td>3.4 Conclusion</td>
<td>46</td>
</tr>
<tr>
<td>4 Design, Modeling, Implementation and Simulation of Configurable Logic Block in Quantum-Dot Cellular Automata</td>
<td>47</td>
</tr>
<tr>
<td>4.1 Introduction</td>
<td>47</td>
</tr>
<tr>
<td>4.2 Design, Modeling, Implementation and Simulation of a Nano Quantum-Dot Cellular Automata Configurable Logic Block</td>
<td>49</td>
</tr>
</tbody>
</table>
5 Design of Built-In Self Test for Quantum-Dot Cellular Array Nano FPGAs

5.1 Introduction

5.2 Built-In Self Test

5.2.1 Logic BIST (LBIST)

5.2.2 Memory BIST (MBIST)

5.3 Classification of BIST

5.3.1 Test Pattern Generator

5.3.2 Circuit Under Test

5.3.3 Output Response Analyzer

5.4 QCA Implementation of BIST

5.5 Modeling Faults in QCA

5.5.1 Cell Displacement

5.5.2 Cell Omission

5.5.3 Stuck at Polarization

5.6 Implementation of the Proposed BIST Architecture in QCA
5.7 Design Parameters and Analysis of Simulation Results ....................................... 102

6 Summary and Conclusions 104

6.1 Summary and Conclusions .................................................................................. 104
6.2 Contributions ..................................................................................................... 105
6.3 Future Work ....................................................................................................... 106

References 107

A QCA Designer Manual 114
List of Tables

2-1 Summary of Alignments of the Magnetic Dipoles ........................................ 35
3-1 Metrics of QCA Physical Layout Generated Using the QCA-LG Tool and the Proposed Technique ........................................................................................................ 42
3-2 Metrics of QCA Physical Layout Generated Using the QCA-LG Tool and the Proposed Technique ........................................................................................................ 45
4-1 Truth Table of D-Latch .................................................................................. 55
4-2 Truth Table of RS Flip-Flop ......................................................................... 60
4-3 Truth Table of the Equation Implemented in LUT ....................................... 66
4-4 Metrics of Various Components of the CLB .............................................. 71
4-5 Latency Comparison ..................................................................................... 72
4-6 Area Comparison .......................................................................................... 73
5-1 Truth Table of the Faulty Decoder ............................................................... 90
5-2 Design Parameters ....................................................................................... 102
5-3 Design Parameters of the Components in BIST ......................................... 103
List of Figures

2-1 QCA Cell Polarizations and Representations: (a) Binary 1. (b) Binary 0. ........ 7
2-2 Geometry of a Standard QCA Cell. ................................................................. 7
2-3 Modified Schematic Model to Represent Clocked QCA Array. ....................... 10
2-4 Information Transfer in a Clocked QCA Binary Wire. .................................... 11
2-5 Phases of a QCA Clock. ...................................................................................... 12
2-6 (a) Binary Wire Representation. (b) QCA Layout of Binary Wire. ............... 13
2-7 Binary Wire QCA Simulation. ............................................................................. 14
2-8 (a) Inverter Chain Representation. (b) QCA Layout of Inverter Chain......... 14
2-9 Inverter Chain QCA Simulation. ........................................................................ 14
2-10 (a) Majority Voter Representation. (b) QCA Layout of Majority Voter......... 15
2-11 QCA Simulation of Majority Voter. ................................................................. 15
2-12 (a) AND Gate Representation. (b) QCA Layout of AND Gate.................. 16
2-13 QCA Simulation of AND Gate. .......................................................................... 16
2-14 (a) OR Gate Representation. (b) QCA Layout of OR Gate....................... 17
2-15 QCA Simulation of OR Gate. ............................................................................ 17
2-16 (a) Inverter Representation. (b) QCA Layout of the Inverter....................... 18
2-17 QCA Simulation of the Inverter Using Standard QCA Cells. ....................... 18
2-18 Representation and QCA Layout of the Inverter Using Rotated QCA Cells.. 18
<table>
<thead>
<tr>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-19</td>
<td>QCA Simulation of the Inverter Using Rotated QCA Cells</td>
</tr>
<tr>
<td>2-20</td>
<td>(a) Representation of the Inverter. (b) QCA Layout of the Inverter</td>
</tr>
<tr>
<td>2-21</td>
<td>QCA Simulation of the Inverter</td>
</tr>
<tr>
<td>2-22</td>
<td>Relation Between Distance D and Angle θ</td>
</tr>
<tr>
<td>2-23</td>
<td>QCA Layout of Coplanar Crossover</td>
</tr>
<tr>
<td>2-24</td>
<td>QCA Simulation of Coplanar Crossover</td>
</tr>
<tr>
<td>2-25</td>
<td>QCA Layout of Multilayer Crossover</td>
</tr>
<tr>
<td>2-26</td>
<td>Cross-Sectional View of Various Layers in MultiLayer Crossover</td>
</tr>
<tr>
<td>2-27</td>
<td>Schematic Representation of a Four-Dot Metal QCA Cell</td>
</tr>
<tr>
<td>2-28</td>
<td>Majority Gate Designed Using Metal Dot QCA Implementation</td>
</tr>
<tr>
<td>2-29</td>
<td>Majority Gate Converted into Metal Dot Majority Gate</td>
</tr>
<tr>
<td>2-30</td>
<td>Schematic and Micrograph View of Clocked Shift Register</td>
</tr>
<tr>
<td>2-31</td>
<td>Molecular Implementation of a QCA Cell</td>
</tr>
<tr>
<td>2-32</td>
<td>The Allyl End-Groups which Act as Dots for QCA</td>
</tr>
<tr>
<td>2-33</td>
<td>Molecular Form of QCA cell</td>
</tr>
<tr>
<td>2-34</td>
<td>Magnetic QCA (a) Representation of Logic ‘1’ and Logic ‘0’. (b) Ground State and Meta Stable State of Coupled Pairs. (c) QCA Wire using Nano Magnets</td>
</tr>
<tr>
<td>2-35</td>
<td>Alignment of Magnetic Dipoles to Demonstrate the Correct Functionality of Majority Voter</td>
</tr>
<tr>
<td>3-1</td>
<td>Functional Block Diagram of QCA-LG Tool</td>
</tr>
<tr>
<td>3-2</td>
<td>QCA Layout of a 2x1 Mux Generated by the QCA-LG Tool</td>
</tr>
<tr>
<td>3-3</td>
<td>QCA Simulation for the Physical Layout of a 2x1 Mux Generated Using the QCA-LG Tool</td>
</tr>
</tbody>
</table>
Mux 2x1: (a) Gate Level Diagram. (b) QCA Layout Obtained by One-To-One Mapping ................................................................. 41

QCA Simulation 2x1 Mux .................................................................................................................................................. 42

QCA Layout of 1-Bit Adder Generated by the QCA-LG Tool ........................................ 43

QCA Simulation in QCA Designer of 1-Bit Adder Generated by the QCA-LG Tool ................................................................. 43

1-Bit Adder: (a) Gate Level Diagram. (b) QCA Layout Obtained by One-To-One Mapping. ................................................................. 44

QCA Simulation of 1-Bit Adder ........................................................................................................................................ 45

Block Diagram of a Single Slice of Configurable Logic Block of Virtex 5 FPGA .................................................................................................................. 48

Block Diagram of the Proposed Slice of Configurable Logic Block in QCA.. 48

Block Diagram of 5-to-32 Decoder ...................................................................................................................................... 50

Schematic Representation of Majority Voter Logic of the First Step of 2-to-4 Decoder. .................................................................................................................................. 50

Schematic Representation of Majority Voter Logic for 2-to-4 Decoder .... 51

QCA Layout of the 2-to-4 Decoder ...................................................................................................................................... 52

QCA Simulation of 2-to-4 Decoder when Enable = ‘1’ ................................................................. 52

QCA Simulation of the 2-to-4 Decoder when Enable = ‘0’ ................................................................. 53

QCA Layout of the 5-to-32 Decoder ...................................................................................................................................... 53

MV Logic of D-Latch .................................................................................................................................................. 55

QCA Layout of D-Latch .................................................................................................................................................. 55

QCA Simulation of the D-Latch ........................................................................................................................................ 56
5-4 QCA Simulation of the Test Pattern Generator................................. 81
5-5 QCA Layout of the Output Response Analyzer..................................... 82
5-6 QCA Simulation of the Output Response Analyzer................................. 82
5-7 (a) QCA Layout of a Fault-Free Majority Voter. (b) QCA Simulation of the
Majority Voter...................................................................................... 84
5-8 (a) QCA Layout of a Majority Voter with a Cell Displaced by 42nm. (b) QCA
Simulation of the Faulty Circuit............................................................ 85
5-9 (a) QCA Layout of a Majority Voter with a Cell Displaced by 62nm. (b) QCA
Simulation of the Faulty Circuit............................................................ 85
5-10 (a) QCA Layout of a Majority Voter with a Cell Displaced by 82nm. (b) QCA
Simulation of the Faulty Circuit............................................................ 86
5-11 (a) QCA Layout of Majority Voter with a Cell Omitted. (b) QCA Simulation of
the Faulty Circuit................................................................................ 87
5-12 (a) QCA Layout of the Majority Voter with Multiple Cells Omitted. (b) QCA
Simulation of the Faulty Circuit............................................................ 87
5-13 (a) QCA Layout of a Binary Wire with Missing Cell (b) QCA Simulation of the
Faulty Wire. ......................................................................................... 87
5-14 MV Logic of a 2-to-4 Decoder with an s@1 Fault in the Input Line B......... 88
5-15 QCA Layout of a 2-to-4 Decoder with Induced s@1 Fault at the Input Line B.
............................................................................................................. 89
5-16 (a) QCA Simulation of an Ideal 2-to-4 Decoder. (b) QCA Simulation of a 2-to-
4 Decoder with Stuck at Polarization Fault .......................................... 89
5-17 MV Logic of a 2-to-4 Decoder with an s@-1 Fault in the Output Line........ 91
5-18 (a) QCA Layout of a 2-to-4 Decoder with Induced s@-1Fault at the Output Line. (b) QCA Simulation of the Faulty Decoder ........................................ 91
5-19 Block Diagram of the Proposed BIST Model in QCA .......................... 92
5-20 Fault-Free Model of BIST. ............................................................... 93
5-21 QCA Simulation of a Fault–Free Model in BIST. ............................... 94
5-22 Fault Modeling for BIST Architecture ............................................ 95
5-23 QCA Layout of Faulty Model 1 with BIST ........................................ 96
5-24 Simulation of Faulty Model 1 with BIST ......................................... 97
5-25 QCA Layout of Faulty Model 2 with BIST ........................................ 98
5-26 QCA Simulation of Faulty Model 2 with BIST ................................. 99
5-27 QCA Layout of Faulty Model 3 with BIST ........................................ 100
5-28 QCA Simulation of Faulty Model 3 with BIST ................................. 101
A-1 Dialog Box for Managing Bus Layers .............................................. 116
A-2 Cell Selection for Clock Assignment .............................................. 117
A-3 Clock Assignment for Selected Cells .............................................. 117
A-4 Cells with Clock Assigned ............................................................... 117
A-5 Toolbar Selection to Create a Block ................................................. 118
A-6 Managing Layers in Multi-Layer Crossover ..................................... 119
A-7 Listing of Layers ............................................................................ 120
A-8 Ordering of Layers .......................................................................... 121
A-9 Dialog Box to Create New Layer .................................................... 121
A-10 Toolbar Selection for Rotating Cells .............................................. 123
A-11 Multi-Layer Crossover .................................................................... 125
<table>
<thead>
<tr>
<th>Page</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-12</td>
<td>Dialog Box for Manipulating Cell Function ........................................ 126</td>
</tr>
<tr>
<td>A-13</td>
<td>Dialog Box to Change Substrate Properties ........................................ 127</td>
</tr>
<tr>
<td>A-14</td>
<td>Dialog Box to Change Label Properties ............................................ 128</td>
</tr>
<tr>
<td>A-15</td>
<td>Exhaustive Verification Selection .................................................... 128</td>
</tr>
<tr>
<td>A-16</td>
<td>Vector Table .......................................................................................... 129</td>
</tr>
<tr>
<td>A-17</td>
<td>Adding Vectors to the Table ................................................................. 130</td>
</tr>
<tr>
<td>A-18</td>
<td>Editing the Vector Values in the Table .................................................. 130</td>
</tr>
<tr>
<td>A-19</td>
<td>Bistable Engine Options ........................................................................ 131</td>
</tr>
<tr>
<td>A-20</td>
<td>Radius of Effect ...................................................................................... 133</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

1.1 Motivation

The International Technology Roadmap for Semiconductors (ITRS) predicts the size limit for CMOS technology to reach 5 nm to 10 nm by 2017 [1]. This trend of down scaling in transistor size has long obeyed Moore’s law, which states that the number of transistors integrated on a chip of unit area will double approximately every 18 months. To handle this exponential growth curve of device density and increased processing power, there has been a pressing need for rapid improvements in all aspects of integrated circuit (IC) fabrication. At gate lengths below 0.1 mm, FETs will begin to encounter fundamental problems where the quantum effects begin to dominate the device performance that make further scaling difficult due to the following major reasons:

- As the channel length decreases, the thickness of the oxide layer decreases which results in an increase in the gate leakage current caused due to electron tunneling. This quantum mechanical phenomenon causes an increase in the power consumption as the channel length scales down.

- Interconnects scale linearly with channel length which results in electron-migration and other failure mechanisms.
As spacing between interconnects decreases, capacitive coupling increases. This causes an increase in the delay for the signal to traverse through interconnects which degrades the system performance.

Thus, there is need for new technologies that can overcome these limitations. Researchers are investigating for alternatives to CMOS technology at nano scale like carbon nanotube field effect transistors, nanowires, single electron transistors, quantum cellular automata and spin transistors. Amongst the alternatives being considered, quantum-dot cellular automata (QCA) technology has emerged as one promising transistor-less technology which may replace CMOS circuits in the future [2]. QCA circuits, comprising of quantum cells, transfer information from one cell to the other due to the coulombic interactions between the electrons [3] [4]. Since there is no actual current flow, QCA architectures can achieve low power dissipation of 100W/cm² with greater device density (10¹² devices/cm²) at Terra Hz operating speed [2]. The advantage of QCA lies in its extremely high packing density possible due to the small size of the dots, simplified interconnections, and extremely low power consumption.

1.2 Research Objectives

FPGAs present an attractive application of QCA as they are well suited for fabrication due to their homogenous structure [5]. Standard QCA cells are used to model the configurable logic block (CLB) of the FPGA. The CLB has more than one look-up table (LUT). A 5-input LUT is designed which emulates a Virtex-5 Xilinx FPGA architecture [6]. The CLB of Virtex-5 architecture has two slices each containing four LUTs.
The proposed research is divided into the following two phases:

Phase 1: Design, modeling, implementation and simulation of various FPGA components in QCA technology.

The CLB components that were designed, modeled and simulated in QCA are:

- Decoder
- Memory Array
- Multiplexer
- Flip Flop

The first three components listed above are integrated to form a LUT. The LUTs are integrated with Flip Flops (FF) to form a complete slice of CLB of an FPGA. Standard QCA cells are used for generating the physical layouts of the components. All the physical layouts of the components are simulated using the QCA Designer tool. The CLBs are tested by implementing boolean equations and verifying the simulations with their expected outputs. A detailed analysis of the proposed architecture has been discussed.

Phase 2: Design, modeling, implementation and simulation of testing strategies and fault analysis in QCA.

A built-in self test (BIST) model is designed, modeled and implemented at the quantum level. The components of the BIST model are:

- Test Pattern Generator
- Output Response Analyzer
The simulations of the components are verified with the expected outputs. The designed BIST model is incorporated in the CLB tested for various faults [7].

The QCA Designer tool is used to simulate and analyze the physical layouts of the digital circuits.

The thesis is organized as follows.

Chapter 1: This chapter gives the overview of the research.

Chapter 2: This chapter discusses the basic fundamentals of QCA. It gives a brief insight into the concepts of QCA cells, clocking scheme, simple QCA devices, interconnects and their implementations.

Chapter 3: This chapter discusses the different methods of creating a physical layout in QCA followed by analysis of methods with respect to parameters like area, number of QCA cells used and latency. The physical layouts are simulated using the QCA Designer tool.

Chapter 4: This chapter illustrates the design process, implementation and the simulations of the components of the CLB. The components are integrated to realize LUTs which are the core components of the CLB. It also discusses the evaluation of the proposed CLB architecture with the existing CLB architectures with respect to various design parameters.

Chapter 5: This chapter explains the fundamental concepts of BIST. It gives a brief description of various kinds of BIST architectures. Further, it describes the importance of each component in the BIST architecture. It also explains the integration of the CLB with the proposed BIST model. Finally, the model is used in
fault analysis of the CLB followed by a brief analysis about various parameters like area, number of QCA cells used to implement the model and latency.

Chapter 6: This chapter gives a summary of the research work done in this thesis followed by the intellectual contribution. It is also outlines how the research can be further extended in the future.
Chapter 2

Quantum-Dot Cellular Automata

2.1 Introduction

QCA represents a new technology at the nanotechnology level. It was first introduced by Lent et al. which offered a new potential paradigm shift in computing [3]. Unlike conventional technologies which use voltage or current to represent the binary values, QCA uses the position of the electrons in a cell to represent binary values [8]. QCA technology has inherent features like high device density (10^{12} devices/cm²), higher operating speed (high clock frequency, usually in the range of Tera Hz) and lower power consumption (100W/cm²) [2].

2.2 Quantum Cells

The standard QCA cells have four quantum dots and two electrons [3]. There are various kinds of QCA cells proposed which include a six-dot QCA cell and an eight-dot QCA cell. The six-dot and eight-dot are still in their developing stages therefore, the standard four-dot QCA cells are used for generating physical layouts throughout this research. Figure 2-1 illustrates the standard QCA cell with four dots and two electrons. The two electrons in each cell are free to tunnel within the cell but they are bounded by the cell boundary. The two electrons in the cell repel each other to occupy diagonally
opposite corners of the cell resulting in two stable states which represent the binary logic values. Logic ‘0’ is represented when the two electrons occupy the upper-left and lower-right dots. Logic ‘1’ is represented when the two electrons occupy the upper-right dot and lower-left dot [4].

![Figure 2-1 QCA Cell Polarizations and Representations: (a) Binary 1. (b) Binary 0.](image)

QCA is an array of nano-electronic device cells where the information transmission and processing is purely due to the coulombic mechanism unlike other conventional technologies where information is transferred by electric current [8]. Figure 2-2 shows the cell geometry and it also indicates the tunneling energy between the ground state and any one of the stable state designated by ‘t’.

![Figure 2-2 Geometry of a Standard QCA Cell.](image)
A quantity which measures the extent to which the charge distribution is aligned along one of the two diagonals is defined as polarization. If ‘\( \rho_i \)’ denotes the electron charge at dot ‘i’ where \( i = 0, 1, 2, 3 \) and 4 shown in Figure 2-2. The polarization ‘\( P \)’ is given by Equation 2-1.

\[
P = \frac{(\rho_1 + \rho_3) - (\rho_2 + \rho_4)}{\rho_0 + \rho_1 + \rho_2 + \rho_3 + \rho_4}
\] (2-1)

Quantum dots are small semi-conductors or metal islands with a diameter small enough to make their charging energy greater than ‘kB.T’ (where ‘kB’ is Boltzmann's constant and ‘T’ is the operating temperature) [3]. Exactly two mobile electrons are loaded in the cell which can move to different diagonal quantum dots within the QCA cell by means of electron tunneling. Electron tunneling is assumed to be controlled by potential barriers (that would exist underneath the cell) that can be raised and lowered between adjacent QCA cells by means of capacitive plates. Apart from the two states discussed above, there is another ‘unpolarized’ state in which the cell has little or no polarization. In such a state, the inter-dot potential barriers are lowered which reduces the confinement of electrons in the quantum dots [9].

Thus QCA cells perform the computation by coulombic interactions with its neighboring cells to influence each other’s polarization. In the following section, the clocking scheme in QCA is reviewed before proceeding to some simple, yet essential, QCA logic devices: QCA wires, majority gate and inverters.

2.3 Clocking in QCA

Timing, in general, is controlled through a reference signal (i.e., a clock) and is mostly required for sequential circuits. Timing in QCA is accomplished by clocking in
four distinct and periodic phases [10] that are needed for both combinational and sequential circuits. Clocking not only provides the control for information flow but also controls the true power gain in QCA [8]. The signal energy lost to the environment is also restored by the clock. ‘Clocking’ is a very important parameter in QCA design.

Abrupt switching and adiabatic switching are the two types of switching methods in the operation of QCA:

- **Abrupt Switching:** When the inputs to the QCA circuit change suddenly, the circuit would be in some randomly excited state which is unpredictable. Therefore, the QCA circuit has to be relaxed to ground state by dissipating energy. This inelastic relaxation is uncontrolled and the QCA circuit might enter a meta-stable state that may be determined by a ground state.

- **Adiabatic Switching:** In this kind of switching, the system is always kept in its instantaneous ground state. A clock signal is introduced to ensure adiabatic switching. This is the preferred method of switching.

For QCA, the clock signals are generated through an electric field, which is applied to the cells to either raise or lower the tunneling barrier between dots within a QCA cell. This electric field can be supplied by CMOS wires, or CNTs [11] buried under the QCA circuitry. Depending upon the strength of the electric field applied to a QCA cell, the QCA cell has two distinct polarizations. One of the methods for clocking was proposed by K. Hennessy et. al. where conducting wires were used beneath the plane of the QCA molecules to generate a suitable electric field [10]. The schematic representation of this idea is shown in Figure 2-3 [12]. The QCA cells are spread across the XY plane and the Y plane consists of a series of conducting wires just below the
plane of the QCA cells. The conducting wires are excited to produce electric fields by applying suitable voltage. A conductor placed above the QCA cells is grounded so that it draws the electric field in the Z direction. The state of the QCA cell is affected only by the \( \hat{z} \) component of the electric field. The \( \hat{x} \) component does not affect the state of the QCA cell and the \( \hat{y} \) component is zero due to symmetry. Time varying voltages are applied to the conducting wires in such a way that two adjacent wires have a phase shift of \( \pi/2 \) radians between them. This configuration ensures that every fourth wire will have the same applied signal. When the barrier is low, the cells are in a non-polarized state; when the barrier is high, the cells are not allowed to change state. Adiabatic switching is achieved by initially lowering the barrier, removing the previous input, applying the current input and then raising the barrier [8]. If transitions are gradual, the QCA system will remain close to the ground state.

Figure 2-3 Modified Schematic Model to Represent Clocked QCA Array.
In a QCA circuit, information is transferred and processed in a pipelined fashion [13] [14] and a multi-bit information transfer for QCA is allowed through signal latching. All cells within the same zone are allowed to switch simultaneously, while cells in different zones are isolated. This pipelined information transfer is illustrated in Figure 2-4. QCA has four different clocks and each clock is represented in a different color shown in Figure 2-4. Clock 0, 1, 2 and 3 are indicated by the colors green, pink, blue and white, respectively. The assignment of clock should be in the same order i.e. clock 0, clock 1, clock 2 and clock 3. When the clock is applied to the QCA circuit, all the cells in that particular clock zone attains a stable state depending on the driver and all the remaining QCA cells in other clock zones are in a ground state. This is indicated in Figure 2-4, where the electrons in stable state are highlighted and the ground state are indicated by the presence of dots only.

![Figure 2-4 Information Transfer in a Clocked QCA Binary Wire.](image)
QCA clock has four phases (switch, hold, release, and relax) and each phase acts as a potential that modules the inter-dot barriers of all the cells in that phase [10]. For effective information flow, array of QCA cells can be divided into sub-arrays of different clock phases. Figure 2-5 illustrates the polarizations in the QCA cell during the four phases of the clock. During the ‘switch’ phase of the clock, the unpolarized QCA cells polarize in accordance with the driver cell, the cells maintain their polarization for the duration of the ‘hold’ phase. The ‘release’ phase of the clock releases the inter-dot barriers and the cells tend to gradually lose their polarization, and continue to stay in an unpolarized state in the ‘relax’ phase.

![Figure 2-5 Phases of a QCA Clock.](image)

2.4 QCA Logic Devices

A driver of a QCA cell could be an input device such as a nanotube, a very thin wire or a tip of a scanning tunneling microscope (STM). In semiconductor QCA, a standard technique called “plunger electrode” has been used to alter the electron occupancy of the input cell [15] [16] [17]. Reading the output state of a QCA cell is difficult, because the required measurement process should not alter the charge of the output cell. Electrometers made from ballistic point-contacts [18] [19], the STM method
[20], and SET electrometer have been used to read the output. There are two configurations of fabricating the QCA cells: Standard QCA cells and 45° Rotated QCA cells. As the information in QCA is transferred due to coulombic interactions between the two corresponding QCA cells, the state of one cell influences the state of the other corresponding cell. The various kinds of QCA logic devices are:

- Binary Wires
- Majority Voter
- Inverter.

2.4.1 Binary Wires

The QCA cells tend to align to the polarization of its neighbors. Therefore, a linear arrangement of cells can be used as a wire to transmit information. An example of QCA wire is shown in Figure 2-6 (a), the QCA layout in Figure 2-6 (b) and its simulation is shown in Figure 2-7. It can be observed that the binary wire is divided into various clock zones. This is done to ensure that the signal strength carried by the wire is not degraded, as the signal strength tends to deteriorate with a long chain of QCA cells in the same clock zone.

![Binary Wire Representation](image1)

![QCA Layout of Binary Wire](image2)

(a) 
(b) Figure 2-6 (a) Binary Wire Representation. (b) QCA Layout of Binary Wire.
The other kind of QCA wire is termed as the “Inverter chain”. In the inverter chain, the standard QCA cells are rotated by $45^0$ in vertical or horizontal orientation. The polarizations in such QCA cells tend to align opposite to the corresponding QCA cell. Figure 2-8 (a) shows the inverter chain representation and Figure 2-8 (b) shows the QCA layout. The simulation is shown in Figure 2-9.

2.4.2 Majority Voter

Majority Voter (MV) is one of the logic gates in QCA. The logic function implemented by the MV is $f (A, B, C) = A.B + B.C + C.A$. The logic of the MV is
implemented by the five QCA cells indicated in Figure 2-10 (a). The QCA layout is shown in Figure 2-10 (b). The QCA cell present at the centre of the gate is called the decision making cell which tends to polarize to the majority of the inputs. The QCA simulation of the MV is shown in Figure 2-11.

\[ \text{Equation 2-2} \]

\[ \text{Equation 2-3} \]

By fixing the value of one of the three inputs to logic ‘0’ or logic ‘1’, the MV can be programmed as an AND gate or an OR gate, respectively. The logical functions of AND gate and OR gate are shown in Equation 2-2 and Equation 2-3.
\[ MV(A,B,C) = A \cdot B \text{ when } C=0 \]  
(2-2)

\[ MV(A,B,C) = A + B \text{ when } C=1 \]  
(2-3)

Figure 2-12 (a) and Figure 2-14 (a) show the representation of AND gate and OR gate, respectively. The QCA implementation of AND gate and OR gate are shown in Figure 2-12 (b) and Figure 2-14 (b) respectively. The simulations of AND and OR gate are shown in Figure 2-13 and Figure 2-15, respectively.

(a)

Figure 2-12 (a) AND Gate Representation. (b) QCA Layout of AND Gate.

(b)

Figure 2-13 QCA Simulation of AND Gate.
2.4.3 Inverter

Analogous to NOT gate, the basic functionality of the Inverter is to output an inverted value of the input. There are many ways of implementing an inverter in QCA. The standard cells in a diagonal orientation tend to align to the opposite polarization due to the electron repulsion shown in Figure 2-16 (a). The QCA layout is shown in Figure 2-16. The simulation is shown in Figure 2-17.

The representation of rotated cells shown in Figure 2-18 (a) can also be used as an inverter when there are ‘2n’ QCA cells including the input and output cells. The QCA
layout is shown in Figure 2-18 (b). The simulation for the rotated cells is shown in Figure 2-19.

(a) Inverter Representation. (b) QCA Layout of the Inverter.

Figure 2-17 QCA Simulation of the Inverter Using Standard QCA Cells.

(a) Representation and QCA Layout of the Inverter Using Rotated QCA Cells.
Figure 2-19 QCA Simulation of the Inverter Using Rotated QCA Cells.

The most commonly used inverter design is shown in Figure 2-20 (a). The QCA layout is shown in Figure 2-20 (b). The simulation for the inverter is shown in Figure 2-21.

Figure 2-20 (a) Representation of the Inverter. (b) QCA Layout of the Inverter.

Figure 2-21 QCA Simulation of the Inverter.
The signal comes in from the left on a binary wire and splits into two parallel wires as illustrated in the figure. The two corner cells on the parallel wires polarize the cell diagonal (at the right end) to them in the opposite direction. This causes the signal to be inverted. This anti-aligning behavior of standard cells in diagonal orientation can be used to “tap” the signal passing through an inverter chain. Placing a standard cell and aligning it halfway between an even and odd numbered rotated cell will produce an inverted signal and placing it halfway between an odd and even numbered cell will give a buffered value.

The tapping of signal is useful while implementing large circuits where crossover of wires in the same plane is essential.

2.5 Cross-Wiring in QCA

Coulombic interaction between two cells influences the data flow in the QCA circuits. The kink energy \( E_{k,i,j} \) represents the energy cost of cells ‘i’ and ‘j’ having opposite polarization. In other words, the kink energy can be thought of as the essential energy required for a successful switching. The parameters that govern this kink energy in QCA are the distance between the corresponding cells ‘D’ and the angle ‘θ’ as shown in Figure 2-22.

The relationship between kink energy θ and D is given by Equation 2-4.

\[
E_{\text{kink}}(D,\theta) \approx D^{-5}.\cos(4\theta) \quad (2-4)
\]
The cells interact through a quadrupole-quadrupole interaction which decays inversely to the fifth power of the distance between the cells. Therefore, the kink energy will decay rapidly with distance which indicates that every cell has an effective neighborhood and its capacity to polarize those in its neighborhood. This is termed as radius of effect.

There are two types of crossovers in QCA:

1. Coplanar Crossover
2. Multi-Layer Crossover

2.5.1 Coplanar Crossover

In QCA, two binary wires carrying information can cross over each other without interacting. This unique feature of QCA is used in coplanar crossover. The main principle is that the standard QCA cells do not get influenced by $45^0$ rotated cells in the horizontal or the vertical axis. This is because the energy between the standard QCA cells and the rotated QCA cells cancels out to have zero kink energy. As discussed earlier, the kink energy of a cell influences the polarization of the corresponding cell, $E_{\text{kink}} = 0$ J means that there is no influence of one QCA cell on the other QCA cell. Figure 2-23 shows a successful crossover of two binary wires having different polarization and Figure 2-24 shows its simulation.
The disadvantages of coplanar crossover are:

- The fabrication process becomes more complex and the cost increases due to the presence of different orientation of QCA cells.
- The cells have to be properly aligned. Any misalignment of the cells leads to a non-zero value of kink energy resulting in the interaction of the cells in the region of crossover.

Figure 2-23 QCA Layout of Coplanar Crossover.

Figure 2-24 QCA Simulation of Coplanar Crossover.
2.5.2 Multi-Layer Crossover

Multi-Layer crossover is considered to be a reliable method for crossover. QCA circuits can cross signals effectively by passing them in more than one layer using a vertical interconnect [21]. Multi-layer crossover is illustrated in Figure 2-25 [22]. The figure shows a wire polarized to logic ‘0’, call it ‘A’. Another wire is polarized to logic ‘1’, call it ‘B’. If wire ‘B’ has to crossover wire ‘A’, it can be sent to another layer and transmitted horizontally. Vertical interconnect cells are stacked on top of each other and the wire B is sent to the second layer. In the second layer, B can cross over any number of wires lying in the first layer. Wire B can be brought back to the first layer by using the same vertical interconnect cells one below the other shown in Figure 2-26.

The vertical separation between the cells is tuned to match the interaction energy $E_{\text{kink}}$ of the horizontal cells to reduce crosstalk. Several intermediate layers called ‘vias’ can be introduced to reduce interaction among the cells between the first and second layers.

The advantages of Multi-Layer crossover over coplanar crossover are:

- It eliminates the need to rotate the cells in any design, thereby reducing the overhead of fabricating two different orientations of QCA cells.
- Also, the extra layers created during the crossover can be used to implement circuits, providing better integration.
In this research, multilayer crossover technique is used in the design of circuits wherever crossover is required.
2.6 QCA Implementation

There are several methods used to implement a QCA device. Some of the popular ones are the following:

- Metal QCA
- Molecular QCA
- Magnetic QCA

2.6.1 Metal QCA

QCA cells can be built from metallic tunnel junctions and very small capacitors. In this implementation, the device consists of four aluminum islands (dots) connected with aluminum oxide tunnel junctions and capacitors. The major difference between semiconductor and metal QCA is the use of capacitive coupled metal islands instead of the conventional coulombically coupled quantum dots and the presence of many conduction band electrons in the metal island unlike the quantum dot.

In case of metal QCA, a combination of two series dots can either be coupled to other islands through a bias power supply and ground or can be left floating. In the former case, current may flow during the switching event of adding or removing an electron whereas in the later case, the dots connected by tunnel junction may only exchange electrons keeping the total number of electrons to be a constant [23] [24]. The capacitance of the island is determined by the area of the tunnel junction. This determines the operating temperature of the device. The device area is approximately $60\times60 \text{ nm}^2$ and is mounted on a surface at 10mK temperature.

The electrostatic energy of a configuration can be expressed in terms of the voltage and charges on gate electrodes and metal islands indicated by Equation 2-5. [25]
where \( C \) is the capacitance matrix for the islands and electrodes, \( v \) is a column vector of voltage on the gate electrodes, \( q \) and \( q' \) are the column vectors of the island charges and the lead charges respectively. The first term in the above expression calculates the electrostatic energy stored in the capacitors and tunnel junctions. The second term calculates the work done to transfer the charge from the source to the leads.

One of the methods to fabricate the device is by using electron beam lithography (EBL) and dual shadow evaporation on an oxidized silicon wafer. In Figure 2-27, the aluminum dots are located from ‘D1’ through ‘D4’, coupled by tunnel junctions [26]. The two dots (E1 and E2) are single electron tunneling (SET) electrometers for sensing the output. The operating temperature of these devices is approximately 70mK [27].

A majority voter designed using this implementation is shown in Figure 2-28 reproduced from [28]. Differential signals A (between gates 1 and 3), B (between gates 1 and 2), and C (between gates 2 and 4) constitute the inputs to the central cell. The negative (positive) bias on a gate, \( \phi^- (\phi^+) \), mimics the presence (absence) of an electron.
in the input dots, as shown by the shaded regions in Figure 2-29 [29]. The amplitudes of \( \phi^+ \) and \( \phi^- \) are carefully chosen to mimic the potentials due to the polarization of an input cell while they remain small enough not to change the number of excess electrons in the cell.

![Figure 2-28 Majority Gate Designed Using Metal Dot QCA Implementation.](image)

![Figure 2-29 Majority Gate Converted into Metal Dot Majority Gate.](image)

In Figure 2-29, as dots D₁ and D₂ are coupled to only one gate electrode each, voltages corresponding to inputs A and B on gate 1, and inputs B and C on gate 2, are added in order to mimic the effect of two input dots [29]. For instance, the input configuration (ABC = 111) is achieved by setting \( V_1 = 2 \phi^- \), \( V_2 = 2 \phi^+ \), \( V_3 = \phi^+ \), and \( V_4 = \phi^- \) [26]. With inputs A, B, and C traced as a function of time, the differential
potential between dots D_4 and D_3, φ_{D4} – φ_{D3}, is measured using the electrometers E_1 and E_2. The transient characteristics are determined by the time constant of the electrometer circuitry.

As another example, the metal dot QCA implementation of a shift register is shown in Figure 2-30 which is reproduced from [30].

![Figure 2-30 Schematic and Micrograph View of Clocked Shift Register.](image)

Metal QCA devices have certain advantages that motivate the use of such devices. Firstly, such devices are easy to fabricate. Secondly, these devices are easier to analyze and model. On the other hand, metal dot devices have several downsides that need to be considered. The metal dots are of the order of 1µm in dimension hence the devices need to be cooled to low temperatures (4K at most) for the observation of electron switching. Additionally, there could be fabrication defects for metal dot devices which may require special additional circuitry to balance the defects [31].

2.6.2 Molecular QCA

A simpler representation of QCA logic gates can be implemented using molecular QCA which not only yields greater performance but can also operate at room temperatures. In molecular QCA, cells are structurally homogeneous down to the atomic
level. The molecular QCA cells are implemented using a class of compounds called ‘mixed – valence compounds’ which exhibit a unique property of having multiple redox centers in different oxidation states. Each molecule functions as a QCA cell and the redox centers function as quantum dots where the information is encoded with charge configuration and the tunneling junction provided by bridging ligands shown in Figure 2-31. The transfer of electronic charge from one molecule to its neighboring molecule is through quadrupole to quadrupole interactions. The couloumbic energies should be in the range of 0.2 – 0.5 eV for the model to operate at room temperature [32].

To find the response functions of a model, the quadrupole moment of a single molecule should be interpolated between the states ‘0’ and ‘1’. The Schrödinger’s equation is used to evaluate the value of the output molecule.

![Figure 2-31 Molecular Implementation of a QCA Cell.](image)

The logic states can be defined by using the allyl group shown in Figure 2-32 [33]. The state with a positive dipole moment is represented as logic ‘1’ and the state with a negative dipole moment is represented as logic ‘0’. The alignment for logic ‘1’ is such that an unpaired electron is on the bottom allyl group and a positive charge is on the top allyl group. On the other hand, the alignment for logic ‘0’ is such that the bottom allyl group has a positive charge and the top allyl group has an electron.
Molecular QCA devices offer certain advantages over the other implementation models with respect to area, power dissipation and information transfer rates. Implementation of molecular QCA devices reduces the cell size to approximately 1nm x 1nm. Experiments conducted by Lent and Timler [34] [35] have proved that, if clocking is considerably slowed down, the reversible processes can be implemented with ultra low power dissipation. For the information to be transferred without any loss of data, the molecular QCA devices should be able to switch between ‘0’ and ‘1’ fast enough to cope with the clock. It has been reported that the transfer rates of the mixed valence compounds used in molecular QCA have switching times between 10 s and 13 s.

Molecular QCA presents unique challenges such as bonding of the array surface which requires treating the existent strongly bound, chemically robust and mixed valence complexes in chemistry with the use of complex stereoscopic and electro-chemical techniques.

Mobile chargers are created by chemical oxidation or reduction shown in Figure 2-33. Molecules are suitable for the implementation because they act as natural and
uniformly small quantum dots with high density that are suitable for room temperature operations.

![Molecular Form of QCA cell](image)

**Figure 2-33** Molecular Form of QCA cell.

### 2.6.3 Magnetic QCA

This concept was introduced by Cowburn and Welland [36], who demonstrated the operations of magnetic QCA (MQCA) using an array of disk shaped particles, with a diameter of 110nm, that exhibit collaborative behavior. In magnetic QCA, magneto static interactions between nano particles ensure that the system is bi-stable. The moments of nano-particles are either parallel or anti parallel to the axis of the chain. The information is propagated via magnetic interactions as opposed to the electrostatic interactions in metal and molecular interactions.

In this system of implementation, the representation of the binary information as well as the information propagation (magnetization reversal) is primarily determined by the coupling-induced magnetic anisotropy in the chain. The quantum mechanical interactions in an MQCA network are due to the interactions between the spins within a single dot which form a single strong classical spin.

The logic ‘1’ is signaled when the dots magnetization vector points in the upward direction and the logic ‘0’ is signaled when the vector points in the downward direction.
Figure 2-34(a) and (b) show how two magnets can be assigned logic states ‘1’ or ‘0’, and can couple either in a ground state, or higher-energy metastable state.

Most magnetic thin films, such as perm alloy or cobalt, display in-plane magnetization. That is, the preferred domain orientation is parallel to the plane of the film. Other films, such as carefully constructed CoPt multi-layers, can exhibit out-of-plane magnetization. Coupling of magnets in either of these configurations can, in principle, be used for QCA.

The result of the magnetization pattern for a chain of nano-magnets depends on their alignment. The magnets might be aligned to enter two different states depending on their axis of alignment. A collinear alignment of the magnets along their long axis reinforces the magnetization in the same direction. This state is called the ferromagnetically ordered state. On the other hand, placing the magnets side-by-side in parallel will result in a line that favors anti-parallel alignment of the electric dipoles. This state is called the anti-ferromagnetically ordered state [37]. In MQCA, these ordering phenomenon drive the computation of the patterns. It is not hard to see how a chain of narrowly spaced nano-magnets shown in Figure 2-34(c) could be used as a QCA ‘wire’.

A ‘NAND’ or ‘NOR’ gate can be designed using the simplest arrangement of five nano-magnets (a central magnet surrounded by four other magnets) and an inverter. The top, left and bottom magnets can be used as inputs driven by additional driver magnets oriented in the x-direction. The fourth magnet in the center acts as the output driver. Eight different logic combinations can be achieved by varying the states of the driver nano-magnets to demonstrate the majority voter is shown in Figure 2-35. Table 2-1 summarizes the states of the central magnet and the output magnet.
Figure 2-34 Magnetic QCA a) Representation of Logic ‘1’ and Logic ‘0’. b) Ground State and Meta Stable State of Coupled Pairs. c) QCA Wire using Nano Magnets.

Figure 2-35 (a – b) Alignment of Magnetic Dipoles to Demonstrate the Correct Functionality of Majority Voter.
Figure 2-35 (c – h) Alignment of Magnetic Dipoles to Demonstrate the Correct Functionality of Majority Voter.
Table 2-1 Summary of Alignments of the Magnetic Dipoles.

<table>
<thead>
<tr>
<th>Logic State of Input Magnets</th>
<th>Logic State of Central Magnet</th>
<th>Logic State of Output Magnet</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>001</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>010</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>011</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>100</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>101</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>110</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>111</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

The three input majority gate discussed above can be programmed as a two input NAND gate or NOR gate depending upon the state of any one of the three input magnets and the inversion at the output magnet. Thus any Boolean logic function can be regenerated using a network of arrangements.

With improvements in the current fabrication techniques, MQCA has the potential to replace CMOS technology in the coming years. The device packing of MQCA is intense as compared to CMOS technology. Going by Moore’s law the CMOS technology faces serious physical limitations for further downsizing but on the other hand MQCA seems to successfully continue abiding by this law. According to [38], integration density of 5500 million cm\(^{-2}\) is comparable with 6.6 million cm\(^{-2}\) in today’s CMOS technology.
Chapter 3

Evaluation of QCA CAD Tools

3.1 Introduction

This chapter reviews the CAD tools available for the automatic generation of physical layouts of various digital circuits at the quantum level which uses the standard QCA cells. A detailed description about various blocks of the QCA-Layout Generator (QCA-LG) is presented along with its merits and demerits. The proposed technique for generating the physical layouts is compared and evaluated with respect to area and latency.

3.2 QCA-LG Tool

The QCA-LG tool is a joint product of Tiago Teodosio and Leonel Sousa [39]. The tool automatically generates a file compatible with the QCA Designer. The generated file opened with the QCA Designer gives the QCA layout of the digital circuit. The operations carried out by the QCA-LG tool is shown in Figure 3-1 which is reproduced from the original work [39]. Various functional blocks of the QCA-LG tool are discussed in the following subsections.

3.2.1 Read Input Logic Circuit

The main goal of this stage is to import the combinational circuit to the tool to
generate the netlist based on certain predefined components in the libraries. These combinational circuits can be imported in only two supported formats. They are:

- LSI
- Gate

LSI format can be generated by using synopsys tools. Gate, specific netlist format, is generated by using MVSIS which is a SIS successor.

![Functional Block Diagram of QCA-LG Tool](image)

Figure 3-1 Functional Block Diagram of QCA-LG Tool.

3.2.2 Circuit Expansion

This stage focuses on expanding the shared nodes by duplicating them. This step makes the routing path easier by eliminating the cross wiring but in the process may increase the circuit area. Sometimes this may lead to an increase in the length of the wires which thereby leads to an increase in the circuit area. Expanding the shared nodes is a recursive process which uses the breath first search algorithm. The search begins from each primary output node heading towards the input nodes. A node is duplicated every time it is revisited. This step results in independent sub circuits that have one of the primary outputs as their root.
3.2.3 Gate Placement

For transforming a circuit to a physical level from the gate level, the following coordinates are considered:

1. Gate level coordinates

2. Cell level coordinates

3. Physical coordinates

Gate Level Coordinates – The relative positions of the gates are defined by the set of gate coordinates. The y-axis (vertical axis), defines the gate level. In a multi-layer crossover, there are always more than three layers namely main layer, vias and lower layer(s). So the main layer has a ‘y’ coordinate of ‘0’ followed by the vias, ‘1’ and then by the lower layer(s) ‘2’, and so on. The x-axis (horizontal axis) defines the width of the gate or wire. It usually defines the number of QCA cells.

Cell Level Coordinates – These set of coordinates represent a single QCA cell in x-axis and y-axis.

Physical Coordinates – A set of physical coordinates maps the coordinates of the cell to the coordinates used in the QCA Designer. The default values set in the QCA Designer will be used.

3.2.4 Gate Shaping

The transformation from the gate level to the cell level is done during the gate shaping process defined by the equations:

\[ x_{\text{cell}} = (x_{\text{gate}} + x_{\text{gate of f_set}}) \times \text{factor} \]  

(3-1)
\[ y_{\text{cell}} = (y_{\text{gate}}) \cdot y_{\text{factor}} + y_{\text{cell of fset}} \] (3-2)

where ‘\(x_{\text{cell of fset}}\)’ separates the various circuits along the layout and ‘\(y_{\text{cell of fset}}\)’ is the parameter used to vary the gate position when required.

3.2.5 Input Signals Distribution

In a logic circuit, a primary input may be used as an input for several other gates. So it should be ensured that the signal is distributed from a unique source and also care should be taken to make sure that the signal arrives at the destination in a synchronized way. An iterative method is used to determine the unique source from which the signal will derive its secondary distribution. For a successful distribution of the signal, the initial conditions are very critical.

To ensure the synchronization of the signal at the destination in a timely manner, additional clock zones are used at the beginning of the wire.

3.2.6 Output Layout

The layout generated is stored as QCA layout block which is compatible with QCA Designer. The QCA-LG tool translates the logic description into a physical layout. The tool is designed to operate on a Unix/Linux environment. Procedure for generating the QCA layout using this tool is as follows:

- A netlist should be obtained for the logical circuit either in Gate or LSI format.

- The file obtained from the first step is fed as standard input (stdin) and a standard output (stdout) file is obtained. The resulting file is the physical layout of the digital circuit.
3.3 Design Comparisons

The proposed design technique is based on minimizing the boolean equation using Karnaugh Maps and then translating the given equation into its gate level. Finally, the gate level is transformed to the physical level using the QCA Designer by manually placing the QCA cells. The translation from the gate level to the physical level is based on one-to-one mapping. The following section shows the comparisons of the physical layouts between the QCA-LG and the proposed design technique in terms of area occupied, number of QCA cells used to realize the logic circuit, and latency.

The QCA layout and simulation of a 2x1 Multiplexer obtained using the QCA-LG tool is shown in Figure 3-2 and Figure 3-3, respectively.

![Figure 3-2 QCA Layout of a 2 x 1 Mux Generated by the QCA-LG Tool.](image)

![Figure 3-3 QCA Simulation for the Physical Layout of a 2x1 Mux Generated Using the QCA-LG Tool.](image)
The boolean equation for 2x1 Mux is:

\[ Y = S'.A + S.B \]  \hspace{1cm} (3-3)

where ‘A’ and ‘B’ are the inputs to the multiplexer and ‘S’ is a select line. Finally, a one-to-one mapping from the gate level is done to obtain the physical level in the QCA Designer shown in Figure 3-4 (a) and (b) respectively. The arrows in the figure point to the QCA layouts of various gates. The simulation is shown in Figure 3-5.

The design parameters of the layouts generated using the QCA-LG tool and the proposed technique are tabulated in Table 3-1.

Figure 3-4 Mux 2x1: (a) Gate Level Diagram. (b) QCA Layout Obtained by One-To-One Mapping.
Figure 3-5 QCA Simulation of a 2x1 Mux.

Table 3-1 Metrics of QCA Physical Layout Generated Using the QCA-LG Tool and the Proposed Technique.

<table>
<thead>
<tr>
<th>Number of QCA Cells used.</th>
<th>Area Occupied in μm²</th>
<th>Latency in Clock Cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Design Simulated using QCA-LG Tool</strong></td>
<td>146</td>
<td>0.28</td>
</tr>
<tr>
<td><strong>Design Simulated using Proposed Technique</strong></td>
<td>77</td>
<td>0.13</td>
</tr>
</tbody>
</table>

It is found from the table that the layout generated using the proposed technique uses almost half the number of QCA cells used in the physical layout generated using the QCA-LG tool. This latency is also found to be optimal in the circuit generated using the proposed technique.
As another example, the QCA layout and simulation of a 1-bit Adder obtained using the QCA-LG tool are shown in Figure 3-6 and Figure 3-7, respectively.

![Figure 3-6 QCA Layout of a 1-Bit Adder Generated by the QCA-LG Tool.](image)

![Figure 3-7 QCA Simulation in QCA Designer of a 1-Bit Adder Generated by the QCA-LG Tool.](image)

The boolean equation for a 1-bit Adder is:

\[ F = A \cdot B \cdot C_{in} + \overline{A} \cdot B \cdot \overline{C_{in}} + \overline{A} \cdot \overline{B} \cdot C_{in} + A \cdot \overline{B} \cdot \overline{C_{in}} \]
\[ F = A \oplus B \oplus C_{in} \]  \hspace{1cm} (3-4)

The gate level diagram representing the above equation is shown in Figure 3-8 (a). The physical layout is derived from one-to-one mapping in the QCA Designer shown in Figure 3-8 (b). The arrows in the figure point to the QCA layouts of various gates.

Figure 3-8 1-Bit Adder: (a) Gate Level Diagram. (b) QCA Layout Obtained by One-To-One Mapping.

The simulation of the physical layout is shown in Figure 3-9. Finally, the design parameters from both the physical layouts are tabulated for comparison in Table 3-2.
As observed from the table, the area and the latency of the physical layout generated using the proposed technique resulted in an optimal circuit as compared to the one generated using the QCA-LG tool.
3.4 Conclusion

From the results tabulated in the previous section, it can be concluded that the area occupied by the physical layout generated using the QCA-LG tool is large as compared to the area occupied by the physical layout generated using the proposed technique. The large area is due to unnecessarily long wires which results in longer delays at the output. This can be observed from the simulations obtained using the QCA Designer.

The QCA-LG tool is not fully developed as certain critical issues are not addressed in the tool. The following are the known problems:

- The ‘vias’ cells are manually placed to obtain the correct simulation when simulated in the QCA designer. This becomes tedious work for complex circuits.

- Even for simple logical circuits, the physical layouts generated using the QCA-LG tool are significantly large. Optimization in the gate placement algorithm is necessary.

- When certain inputs are needed only at one place like a common clocking signal, input bus etc., unnecessary wire length is needed which can be avoided.

So, it can be concluded that the proposed technique for generating the physical layouts in QCAD Designer results in optimized layouts compared to the ones generated using the QCA-LG tool. Therefore, the proposed technique is used along with the QCA Designer tool to generate the physical layouts of all the circuits in this research. Further details on the operation of the QCA Designer tool are explained in detail in Appendix I.
Chapter 4

Design, Modeling, Implementation and Simulation of Configurable Logic Block in Quantum-Dot Cellular Automata

4.1 Introduction

FPGAs have become invaluable components in digital design with their applications ranging from simple logic to highly complex computing systems. Their numerous applications can be attributed to the fact that FPGA technology includes both hardware development and software programming. Configurable Logic Blocks, I/O pads and interconnects form integral parts of an FPGA. FPGAs have programmable hardware elements, which can be programmed to implement any boolean function. Such flexibility makes them ideal for implementation in nano technology like QCA. QCA FPGAs will present a platform to experiment with many applications at the nano level.

The basic block diagram of a single slice of the Xilinx FPGA architecture is shown in Figure 4-1. The LUTs of the Xilinx FPGA can be implemented either as 6-input LUT with one output or 5-input LUTs with two possible outputs [6]. This chapter discusses the CLB implementation of an FPGA at the quantum level. The basic block diagram of the proposed CLB is shown in Figure 4-2. The proposed CLB architecture closely imitates the architecture of Xilinx Virtex 5 FPGA. In this research, a CLB model with 5-input LUT with one output is implemented.
Figure 4-1 Block Diagram of a Single Slice of Configurable Logic Block of Virtex 5 FPGA.

Figure 4-2 Block Diagram of the Proposed Slice of Configurable Logic Block in QCA.
4.2 Design, Modeling, Implementation and Simulation of a Nano Quantum-Dot Cellular Automata Configurable Logic Block

The CLBs are the core of the FPGA architecture. They are the main logic resources for implementing sequential as well as combinatorial circuits. The number of CLBs in an FPGA varies from one device family to the other. The CLBs mainly consist of LUTs, multiplexers and flip-flops. The basic logic that is processed by the CLB resides in the Look-up Table (LUT). The design, modeling, and implementation of the various CLB components in QCA are described in the following subsections.

4.2.1 Design, Modeling and Simulation of Address Decoder

The address decoder is used to enable one of the memory locations specified by the addresses given at its inputs. In the proposed research, thirty two memory locations have to be addressed by the decoder as it is a 5-input LUT. Therefore, a decoder with 5 inputs and 32 outputs is needed. In Figure 4-3, A, B, C, D and E are the input lines and “Y₀ - Y₃₁” form the output lines of the decoder. The high output signal line is used to select one of the 32 memory elements.

The decoder is modeled using a hierarchical structure comprising of three stages. As shown in Figure 4-3, stage one consists of a 1-to-2 decoder, stage two consists of two 2-to-4 decoders and the final stage consists of eight 2-to-4 decoders. This hierarchical design exploits the advantage of modular design. The 2-to-4 decoders are designed in two steps. As shown in Figure 4-4, the first step uses four MV gates configured as four AND gates. The signals ‘I₀’ and ‘I₁’ are given as the inputs to the gates and all the possible input configurations are realized using inverters wherever necessary.
Figure 4-3 Block Diagram of 5-to-32 Decoder.

Figure 4-4 Schematic Representation of Majority Voter Logic of the First Step of 2-to-4 Decoder.
As shown in Figure 4-5, in the second step of the design of 2-to-4 decoder, there are four MVs configured as four AND gates. The outputs from the previous stage acts as one of the inputs to the AND gates and the other input to the AND gates is a common enable signal. This enable signal is used to enable or disable the decoder. As seen from the figure, the outputs of the decoder are pulled down to logic ‘0’ when the decoder is disabled due to the presence of AND gates at the outputs. When the decoder is enabled, the output depends on the combination of the input signals.

The physical layout of the 2-to-4 decoder in QCA generated with the QCA Designer CAD tool is shown in Figure 4-6. The simulation is shown in Figure 4-7. Care has been taken to ensure that the signals arrive at the gates in a synchronized manner so as to obtain the desired output. This is achieved by carefully configuring the circuit in appropriate clock zones.
Figure 4-7 QCA Simulation of 2-to-4 Decoder when Enable = ‘1’.
(Inputs: A and B, Outputs: Zero, One, Two and Three)

It can be seen from Figure 4-7 that there is a delay of two clock cycles between the inputs and outputs as indicated by the yellow dotted circle. In the above simulation, the enable line is continuously held high. In Figure 4-8, the enable line is held low and it can be observed that the outputs are pulled down to logic ‘0’ after a delay of two clock cycles.
Figure 4-8 QCA Simulation of the 2-to-4 Decoder when Enable = ‘0’.
(Inputs: A and B, Outputs: Zero, One, Two and Three)

The 1-to-2 decoder and the other 2-to-4 decoders are integrated as shown in the block diagram in Figure 4-3. The final physical layout of the 5-to-32 decoder is shown in Figure 4-9. Different stages in the hierarchical structure are indicated on the physical layout shown in this figure.

Figure 4-9 QCA Layout of the 5-to-32 Decoder.
During the design phase, care has been taken to avoid long wires which result in longer delays by using multi-layer crossover wherever possible. Care has also been taken to ensure that all the signals arrive at the inputs of various components in a synchronized way to avoid unreliable switching which is undesirable. For this reason, the QCA cells in the circuit are divided into several clock zones consisting of optimal number of QCA cells. It should also be noted that as the number of clock zones increases, the delay also increases. Therefore, a tradeoff between the number of clock zones and the length of each clock cycle is made to get an optimal result.

4.2.2 Design, Modeling and Simulation of Memory Array

The storage elements in an LUT are used to store the expected values of the boolean functions which describe the system. Also, LUTs can be configured as distributed SRAM modules which enable the FPGA to store large amount of data. The SRAM modules are synchronous resources (synchronous read and synchronous write). The decoder output has 32 unique addresses and each of these addresses is capable of addressing a memory location. The memory element should be efficient enough to have faster access time so as to reduce the latency at the output of the LUT.

In this work, a D-Latch with a data line, an enable signal line and an output line is designed and implemented in QCA. The truth table of the D-Latch is shown in Table 4-1. The schematic of the latch using MV logic is illustrated in Figure 4-10. The physical layout and the QCA simulation of a 1-bit D-latch are shown in Figure 4-11 and Figure 4-12, respectively. From the simulation, it can be observed that there is a latency of one clock cycle between the input and the output which is indicated by the yellow dotted circle.
Table 4-1 Truth Table of D-Latch.

<table>
<thead>
<tr>
<th>Enable</th>
<th>Data</th>
<th>Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>X</td>
<td>Latch</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Figure 4-10 MV Logic of D-Latch.

Figure 4-11 QCA Layout of D-Latch.
4.2.3 Design, Modeling and Simulation of the Multiplexer Circuit

The output circuit is used to output one of the 32 values stored in the memory array. The output circuit can be implemented in many ways. One way would be to combine the outputs using a serial chain of OR gates. But this configuration would result in a long latency [40] [41] [42]. Therefore, a multiplexer has been designed and implemented in QCA to function as an output circuit. The output circuit and the address decoder share a common address bus. Care has been taken to avoid the usage of long wires which causes delay at the output by using multilayer crossover wherever possible.

The 4x1 multiplexer is built using three 2x1 multiplexers. As shown in Figure 4-13, ‘S₀’ and ‘S₁’ are the select lines to the multiplexer. The inputs to the multiplexer are randomly assigned. The QCA physical layout and the simulation of a 4x1 multiplexer are shown in Figure 4-13 and Figure 4-14, respectively. The output has a delay of three clock cycles indicated by the yellow dotted circle.
The 16x1 multiplexer is designed by integrating five 4x1 multiplexer and a 2x1 multiplexer. Fixed values of alternate 1’s and 0’s were used as inputs to test the functionality of the logic circuit. The QCA physical layout of the 16x1 multiplexer is shown in Figure 4-15 and its simulation is shown in Figure 4-16.
Figure 4-15 QCA Layout of the 16x1 Multiplexer.

Figure 4-16 QCA Simulation of the 16x1 Multiplexer.

There is a latency of nine cycles at the output indicated by the yellow dotted circle in Figure 4-16. In Figure 4-17, 32x1 multiplexer is built using two 16x1 multiplexers combined with a 2x1 multiplexer. Figure 4-17 and Figure 4-18 show the QCA physical layout and its simulation, respectively. There is a delay of ten clock cycles at the output.
indicated by the dotted yellow circle.

![Figure 4-17 QCA Layout of the 32x1 Multiplexer.](image1)

**Figure 4-17 QCA Layout of the 32x1 Multiplexer.**

![Figure 4-18 QCA Simulation of the 32x1 Multiplexer.](image2)

**Figure 4-18 QCA Simulation of the 32x1 Multiplexer.**

4.2.4 Design, Modeling and Simulation of Flip-Flop

The main purpose of the flip flop in the CLB is to store the output of the LUT and provide a clocked output. The truth table of the flip-flop is shown in Table 4-2. Figure 4-19 shows the QCA layout of the RS Flip-Flop.
Table 4-2 Truth Table of RS Flip-Flop.

<table>
<thead>
<tr>
<th>E</th>
<th>S</th>
<th>R</th>
<th>D</th>
<th>Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>X</td>
<td>Latch</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>X</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>X</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>X</td>
<td>Latch</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>X</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>X</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>D</td>
<td>D</td>
</tr>
</tbody>
</table>

Figure 4-19 QCA Layout of RS Flip-Flop.

Figure 4-20 to Figure 4-23 show the simulation results of RS flip-Flop for various combinations of ‘R’ and ‘S’ when enable is high. The simulation results show a latency of two clock cycles at the output indicated by the dotted yellow circle.
Figure 4-20 QCA Simulation of RS Flip-Flop: RS= “11”.

Figure 4-21 QCA Simulation of RS Flip-Flop: RS= “00”.

Figure 4-22 QCA Simulation of RS Flip-Flop: RS= “01”.

61
4.3 Design and Implementation of Look-Up Table in QCA

In this research, the QCA function generators are implemented using 5-input LUTs. One single LUT can implement any five literal boolean equation. LUTs form one of the most fundamental elements of the CLB. To implement any five input equation, the truth table is programmed into the LUT. Therefore, a LUT requires an address decoder, an array of memory to store the values of the truth table and an output circuitry to output these values [43]. Decoder designed in QCA in the previous section, functions as an address decoder, an array of ‘D’ latches represents the storage elements required to store the values of the truth table and multiplexer functions as an output circuit. The basic block diagram of the LUT is shown in Figure 4-24. The following subsections emphasizes on modeling, implementing and generating the physical layout of an LUT in QCA [44].
4.3.1 Design, Modeling and Simulation of Look-Up Table

Different components designed in QCA in the previous section are integrated into an LUT shown in Figure 4-24. The decoder and the multiplexer have a common address bus and the memory array has 32 1-bit memory elements.

![Figure 4-24 Block Diagram of LUT.](image)

An arbitrarily selected function described by Equation 4-1 is implemented in the LUT to evaluate its functionality.

\[
F (A, B, C, D, E) = (A' \cdot C' \cdot D) + (A' \cdot C \cdot E) + (A \cdot C' \cdot D) + (A \cdot C \cdot E)
\] (4-1)

The QCA layout of the proposed 5-input LUT is shown in Figure 4-25. The red rectangular boxes indicate the various components of the LUT. It can be observed from the diagram that the multiplexer, which functions as an output circuit occupies larger area as compared to the area occupied by other components. Also, the multiplexer largely contributes to the latency of the LUT. The simulation is shown in Figure 4-26.
Figure 4-25 QCA Layout of the Proposed 5-Input LUT.
The functionality of the LUT is verified by comparing the simulated output with the expected values. From the simulation, a delay of 18 clock cycles at the output is observed as indicated by the yellow dotted circle in Figure 4-26.
The truth table for the function described by Equation 4-1 is shown in Table 4-3.

Table 4-3 Truth Table of the Equation Implemented in LUT.

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F(A,B,C,D,E)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

4.4 QCA Implementation of Configurable Logic Block

The CLB mainly consists of LUTs and Flip-Flops. The FFs and LUTs implemented in QCA in Section 4.2 and Section 4.3 are integrated to form a CLB slice.
The QCA physical layout of the CLB is shown in Figure 4-27. The layouts of the LUTs and FFs are enclosed within red rectangular boxes in this figure.

![Figure 4-27 QCA Layout of the CLB.](image)

Each LUT is programmed to implement a five variable boolean equation. Depending on the value of set and reset inputs of the flip flops, the simulations are checked with the expected values. The boolean equations implemented in the LUTs are the following:

\[
F (A, B, C, D, E) = (A'. C'.D) + (A'.C.E) + (A.C'.D) + (A.C.E) \quad (4-2)
\]

\[
G(A, B, C, D, E) = (A'.C'.D') + (A'.C.E') + (A.B.C.E') + (A.C'.D') + (A.B'.C.E') \quad (4-3)
\]

The CLB slice was simulated for various combinations of Set (S) and Reset (R) inputs of the flip flops, while keeping the Enable (E) high. The simulation result for RS =
’01’ is shown in Figure 4-28. The outputs ‘O₁’ and ‘O₂’ are high after one clock cycle verifying the Set operation.
Figure 4-29 was simulated for RS = ‘10’ and a low on both the outputs verifies the Reset operation.
In Figure 4-30, show the simulation result when Set and Reset inputs are SR = ‘00’ or ‘11’. Hence ‘O_1’ and ‘O_2’ have the respective outputs after twenty one clock cycles indicated in the diagram.
4.5 Design Parameters and Analysis of Simulation Results

The first ever configurable logic block with 5-input LUT was successfully designed and simulated at the quantum level using QCA technology. The slice of a CLB has been successfully modeled, implemented, and simulated using the QCA Designer tool. The simulated results of a single slice of a CLB were verified with its expected outputs to confirm the proper functioning of the CLB slice. The proposed architecture has also been optimized with respect to various design parameters like the number of QCA cells used to implement the logic circuit, the area occupied by the logic circuit and the overall delay of the CLB slice.

The design parameters of various sub-components of the CLB have been tabulated in Table 4-4.

<table>
<thead>
<tr>
<th>Module</th>
<th>Number of QCA cells</th>
<th>Area in nm²</th>
<th>Latency</th>
<th>Contribution to the overall latency (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decoder</td>
<td>4508</td>
<td>11604193.51</td>
<td>5</td>
<td>23.81</td>
</tr>
<tr>
<td>Memory Array</td>
<td>1800</td>
<td>2937211.20</td>
<td>1</td>
<td>4.76</td>
</tr>
<tr>
<td>Multiplexer</td>
<td>5541</td>
<td>19585857.92</td>
<td>10</td>
<td>47.61</td>
</tr>
<tr>
<td>Flip-Flops</td>
<td>113</td>
<td>289367.45</td>
<td>2</td>
<td>9.52</td>
</tr>
<tr>
<td>LUT</td>
<td>11689</td>
<td>31203260.39</td>
<td>18</td>
<td>-</td>
</tr>
<tr>
<td>CLB Slice</td>
<td>24,989</td>
<td>90228553.25</td>
<td>21</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 4-4 Metrics of Various Components of the CLB.
The proposed architecture is compared with the existing architectures with respect to read latency, write latency and area. The existing CLB architectures are all based on design of a 4-input LUT. Therefore, to have a fair comparison with these architectures, a 4-input LUT was modeled and implemented using the same technique. Table 4-5 shows the comparison of latency of various architectures with the proposed architecture.

Table 4-5 Latency Comparison.

<table>
<thead>
<tr>
<th>Latency in Clock Cycles</th>
<th>Decoder</th>
<th>LUT</th>
<th>Routing</th>
<th>CLB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read latency [40]</td>
<td>NA</td>
<td>11</td>
<td>19</td>
<td>30</td>
</tr>
<tr>
<td>Read latency [45]</td>
<td>NA</td>
<td>9</td>
<td>7</td>
<td>16</td>
</tr>
<tr>
<td>Read latency of the proposed CLB</td>
<td>NA</td>
<td>10</td>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>Write latency [40]</td>
<td>5</td>
<td>30</td>
<td>NA</td>
<td>134</td>
</tr>
<tr>
<td>Write latency [45]</td>
<td>Data Not Available</td>
<td>9</td>
<td>NA</td>
<td>19</td>
</tr>
<tr>
<td>Write latency of the proposed CLB</td>
<td>2</td>
<td>10</td>
<td>NA</td>
<td>12</td>
</tr>
</tbody>
</table>

Table 4-6 shows the metrics of the area occupied by various architectures along the proposed architecture. In Figure 4-31, the graphical representation of the latency contribution of each CLB component of a 4-input LUT is shown. The area occupied by various CLB components is shown in Figure 4-32.
Table 4-6 Area Comparison.

<table>
<thead>
<tr>
<th></th>
<th>QCA Cells in CLB</th>
<th>Area of the CLB in µm²</th>
<th>QCA Cells in CLB</th>
<th>Area of the CLB in µm²</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLB of [40]</td>
<td>27,265</td>
<td>94.0</td>
<td>27,265</td>
<td>94.0</td>
</tr>
<tr>
<td>CLB of [45]</td>
<td>22,558</td>
<td>48.0</td>
<td>22,558</td>
<td>48.0</td>
</tr>
<tr>
<td>Proposed CLB</td>
<td>11,372</td>
<td>36.0</td>
<td>11,372</td>
<td>36.0</td>
</tr>
</tbody>
</table>

Figure 4-31 Latency Contribution of Each Component to the Overall Latency of the CLB.

Figure 4-32 Area Occupied by Various Components of the CLB.
From the results tabulated above, it can be inferred that the architecture of the proposed CLB with a 4-input LUT is better than the rest of the existing architectures with respect to all design parameters. But area cannot be used as a parameter for a legitimate comparison as the cell separation distance is different for different architectures. The distance between the two QCA cells in the proposed architecture and [40] is 2nm whereas the distance between the two QCA cells in [45] is 20nm which is ten times more when compared with the other two architectures. Therefore, the number of QCA cells used for implementation is used as the design parameter for comparison. As observed from Table 4-6, the number of QCA cells used for implementing the proposed architecture in QCA is half of the number of QCA cells used to implement the same logic in [40] and almost half of the number of QCA cells used in [45].
Chapter 5

Design of Built-in Self Test for Quantum-Dot Cellular Array Nano FPGAs

5.1 Introduction

In the past, BIST models have been developed for FPGAs. The basic idea of BIST is to make the FPGA test itself by allowing the FPGA to perform a self test without the need for external test equipment or a dedicated circuitry on chip to perform the test. On-board, it gives an FPGA the power to execute the self test algorithm, analyze the test result and perform the diagnosis test on the BIST results to identify the faults in the FPGA core, programming logic and the interconnects. Section 5.2 gives a brief overview of the fundamentals of BIST, merits and demerits and classification of the BIST techniques. Section 5.3 explains the various classification of BIST and also explains the major components in the BIST architecture. Section 5.4 illustrates the implementation of the BIST architecture in QCA. Section 5.5 describes fault modeling in QCA followed by the section which describes in detail the implementation and analysis of faulty circuits. The design parameters are discussed in the final section.

5.2 Built-In Self Test

BIST is a technique used to develop features into ICs to perform self testing. BIST allows the IC to perform a self check by using circuits on board, thereby reducing
the dependence on external automated test equipment (ATE). The self testing technique of BIST offers more efficient, less expensive and a faster means of testing an IC. Therefore, BIST can be classified as a Design-for-Testability (DFT) technique [41]. It offers a hierarchical approach to test highly complex digital systems, and its DFT feature eliminates the need for external hardware for conducting the test [42] [46].

BIST is used for testing Complex Programmable Logic Devices (CPLDs), particularly FPGAs. The re-programmability of these devices makes them complex test structures. However, the same re-programmability feature can be used to incorporate BIST in the system function which is programmed in the device during system operation. Once testing is complete, the device can be reprogrammed to reinsert the desired system function. Test configurations can be stored in the system memory and are used for parallel testing of devices. When the locations of faults are also detected by BIST, the system can be reconfigured to bypass the faults to provide fault tolerant operation [47].

Advantages of BIST:

1. It eliminates the need for external testing using ATE which is very costly.

2. Complex test structures can be implemented onto the chip which results in better fault coverage.

3. The test time can be drastically reduced if the BIST is implemented to run in parallel.

4. It provides better serviceability.

Disadvantages of BIST:
1. It presents an additional overhead during the fabrication process to incorporate the BIST circuits on board. However, no additional circuitry is needed for BIST for testing FPGAs.

2. It introduces increased access time and timing problems which lead to a degradation in the performance.

3. It increases the package size of the device to incorporate interface for BIST. Again, this does not apply for implementing BIST on FPGAs.

There are many kinds of BIST techniques including the following:

1. Logic BIST (LBIST)

2. Memory BIST (MBIST).

5.2.1 Logic BIST (LBIST)

LBIST is designed for testing random logic. It generally uses a pseudo-random pattern generator to generate input patterns that are given to the device's internal scan chain and a multiple input signature register as inputs. The response obtained from it is used to test the input patterns. An error in multiple input signature register output indicates a defect in the device.

5.2.2 Memory BIST (MBIST)

MBIST is specifically used for testing memories. It typically consists of test circuits that apply, read, and compare test patterns designed to expose defects in the memory device.
5.3 Classification of BIST

The BIST is broadly classified into two main categories.

1. On-Line BIST

2. Off-Line BIST

In On-Line BIST, testing occurs during normal operation conditions. The circuit contains specialized coding scheme which are incorporated in the circuit during the design phase.

Off-line BIST deals with testing a system when it is not carrying out its normal operation. The circuit enters into a ‘Test mode’ and off-line testing techniques are applied. Off-line testing is carried out using on-chip Test Pattern Generators and Output Response Analyzers.

The major components of a BIST model are shown in Figure 5-1 that comprises of the following: Test Pattern Generator (TPG), an Output Response Analyzer (ORA) and a Circuit under Test (CUT).

![Figure 5-1 Block Diagram of Various Components of BIST Model.](image)

5.3.1 Test Patter Generator

It generates a set of input vectors which are applied to the inputs of the circuit under test. Different types of TPGs are explained below:

- **Exhaustive testing**: It assumes that the CUT is an n-input, m-output combination. It generates all the possible combinations of the inputs \(2^n\). This kind of test
pattern is used for complete testing of all static faults. It detects all possible faults. Usually binary counters are used to generate the input pattern. The major drawback of this scheme is that it cannot be applied for sequential circuits as it becomes unfeasible if ‘n’ is greater than 22.

- **Pseudorandom Testing:** The test pattern generated is less than \(2^n\). These patterns are random, but deterministic and repeatable. An acceptable level of faults is detected by selecting a suitable Test Length.

- **Pseudo-exhaustive Testing:** The test pattern is a combination of both exhaustive and pseudorandom testing. It has all the benefits of exhaustive testing but only has a fewer test patterns. It segments the circuits and exhaustively tests each segment.

5.3.2 Circuit Under Test

It comprises of the entire circuit or a portion of it which has to be tested. The input test vectors are applied to the CUT. These test vectors traverse through the CUT and are fed into the ORA.

5.3.3 Output Response Analyzer

In the ORA, the outputs from the CUT are compared with known good responses. Output Response Analyzers validate the responses from the CUT by comparing the CUT responses to the responses of a known fault free circuit.

5.4 QCA Implementation of BIST

This section explains the design, modeling, and implementation of the BIST model in QCA for testing the CLB slice of the nano FPGA. As discussed earlier, the BIST model has three major functional blocks:
• Test Pattern Generator (TPG)

• Circuit under Test (CUT)

• Output Response Analyzer (ORA)

Test Pattern Generator (TPG): The Test Pattern Generator is used to generate a set of test vectors that are applied to the Circuit under Test. In the proposed BIST architecture, pseudo-exhaustive testing is used. The CLB slice is pseudo-exhaustively tested for all possible combinations of inputs. Thus a 5 bit-binary counter is required as TPG because the CLB houses two 5-input LUTs. The counter would repeatedly provide all the thirty two combinations of the inputs to the two LUTs in the CLB slice.

To implement a binary counter in QCA, the ‘D’ latch is used. The latch is used to hold the output till the next clock cycle. The block diagram for the TPG is shown in Figure 5-2. The QCA layout of the cascaded five D-Latches shown in Figure 5-3 is used to generate the inputs for the LUT. The enable input of the first latch is always kept high, and a series of 0’s and 1’s are given as inputs to $D_0$. The outputs of each latch $Q_0$, $Q_1$, $Q_2$, $Q_3$ and $Q_4$ are the five outputs, with $Q_0$ as the least significant bit and $Q_4$ as the most significant bit.

![Block Diagram Representation of the Test Pattern Generator.](image-url)
Figure 5-3 QCA Layout of the Test Pattern Generator.

The outputs from the TPG are given as inputs to the decoder and the multiplexer.

The simulation results shown in Figure 5-4 verify the successful operation of the counter.

The responses of the TPG are enclosed in the red rectangle in Figure 5-4.

Figure 5-4 QCA Simulation of the Test Pattern Generator.

*Output Response Analyzer:* The output response analyzer analyzes, compares and validates the CUT. The circuit is deemed fault-free if it outputs logic ‘0’ and faulty if it outputs logic ‘1’. To declare a component a pass or a fail, it’s sufficient to compare the output of the circuit under test and response from a fault free circuit. Comparator based ORAs are used for the analysis. Therefore, a simple *XOR* gate is used to function as a comparator. The QCA implementation of XOR gate is shown in Figure 5-5. The QCA
simulation is shown in Figure 5-6. There is a latency of one clock cycle as indicated by the yellow circle in the figure.

![QCA Layout of the Output Response Analyzer.](image1)

**Figure 5-5** QCA Layout of the Output Response Analyzer.

![QCA Simulation of the Output Response Analyzer.](image2)

**Figure 5-6** QCA Simulation of the Output Response Analyzer.

5.5 Modeling Faults in QCA

This part of the research focuses mainly on the manufacturing defects that arise during fabrication. As the fabrication of the QCA devices is still in its nascent stage, it leads to a wide range of fabrication errors. These errors or defects occur mainly due to the manufacturing process used to develop these devices. Molecular self-assembly
process is currently being employed to fabricate these devices which make them more susceptible to defects. The fabrication of QCA devices mainly consists of the synthesis of the QCA cell and their deposition. Errors includes a cell being misaligned with its adjacent cell, change in orientation of the cells, i.e., presence of rotated cells where it is not intended. These errors mainly occur during the deposition phase rather than in the synthesis phase.

The manufacturing defects which result in a QCA cell having missing or extra electrons and/or dots is unlikely to happen due to the advancement in the process involved for purifying these organic molecules. The other defects that arise from the compounds used during the lithographic process are the presence of a charged particle which influences other cells in the circuit. The charged particle present can either be lightly charged or heavily charged. This heavily charged particle can cause a QCA cell to be fixed in a definite orientation which fails to polarize in accordance with the driver cell. We call these faults as ‘Stuck at Polarization’ faults and denote them as ‘s@p’. These faults are tested and analyzed in QCA circuits in the following subsections.

5.5.1 Cell Displacement

Cell displacement is a defect in which the QCA cell is displaced from its original position. The cell displacement can be modeled by slightly moving the QCA cell in the QCA Designer. Figure 5-7 (a) shows a fault free majority voter in which the distance between two adjacent cells is 2nm and Figure 5-7 (b) shows its simulation with a delay of one clock cycle.
Figure 5-7 (a) QCA Layout of a Fault-Free Majority Voter. (b) QCA Simulation of the Majority Voter.

Figure 5-8 (a) illustrates one of the input QCA cells being displaced from its original position by 42 nm. From its simulation it can be observed that the output of this faulty gate is exactly the same as the normal fault free gate. This is because of the parameter called ‘Radius of Effect’ which was set to 65nm in the bistable options before the gate was simulated which means that the cell will have an effect on any other QCA cell placed within its radius of 65nm. This can be verified from the simulation shown in Figure 5-8 (b) in which there was a delay of one clock cycle.

When the distance of displacement of the QCA cell was increased to 62nm from its original position which is very close to the set value of 65nm, the circuit was no longer functioning as a Majority Voter. The QCA layout of this faulty gate is shown in Figure 5-9 (a) where the distance of displacement of the QCA cell is indicated and its simulation with one clock cycle delay is shown in Figure 5-9 (b).
Figure 5-8 (a) QCA Layout of a Majority Voter with a Cell Displaced by 42nm. (b) QCA Simulation of the Faulty Circuit.

Figure 5-9 (a) QCA Layout of a Majority Voter with a Cell Displaced by 62nm. (b) QCA Simulation of the Faulty Circuit.

The circuit functioned in the same way when the distance of misplaced cell from its original position was increased beyond 65 nm. This is illustrated in Figure 5-10 (a) where the cell is displaced by 82 nm from its original position. Its simulation with a delay of one clock cycle is shown in Figure 5-10 (b).
5.5.2 Cell Omission

Cell omission defect is a type of defect which has missing QCA cells in the circuit compared to the fault-free circuit. This can be modeled in the QCA Designer by purposefully omitting QCA cells and then observing its effect.

In Figure 5-11 (a), the decision making QCA cell in the majority voter gate was omitted. A normal majority voter gate implements the logic function $Z(A, B, C) = A.B + B.C + C.A$. But due to the omission of the decision cell, the logic function implemented changes to $Z(A, B, C) = A.C' + A.B'$ considering input A has a stronger polarization compared to the other two inputs. This can be verified from the simulation shown in Figure 5-11 (b) with a delay of one clock cycle.

During the fabrication process of a QCA circuit there are chances that circuit fabricated may end up having more than one cell omitted. This kind of circuit is also implemented in the QCA Designer. A majority gate having multiple cells omitted is shown in Figure 5-12 (a). The gate has three QCA cells omitted at the output line and as the line is left floating with no drivers present in the radius of 65 nm, the output obtained indicates a zero polarization. The simulation result is shown in Figure 5-12 (b).
Figure 5-11 (a) QCA Layout of Majority Voter with a Cell Omitted. (b) QCA Simulation of the Faulty Circuit.

Figure 5-12 (a) QCA Layout of the Majority Voter with Multiple Cells Omitted. (b) QCA Simulation of the Faulty Circuit.

If a QCA cell is omitted in a binary wire in the configuration shown in Figure 5-13 (a), the signal gets inverted at the output which is shown in Figure 5-13 (b).

Figure 5-13 (a) QCA Layout of a Binary Wire with Missing Cell. (b) QCA Simulation of the Faulty Wire.
5.5.3 Stuck at Polarization

These kinds of faults arise due to the presence of a heavily charged particle which causes a QCA cell to be fixed in a definite orientation which fails to polarize in accordance with the driver cell. These faults are called as ‘Stuck at Polarization’ faults and are denoted by ‘s@p’. Such cells fail to polarize according to the driver and stay at a constant polarization of -1 or +1. These faulty cells polarize the QCA cells following them leading to an erroneous value at the output. The stuck at polarized faults are represented as ‘s@p-1’ and ‘s@p1’ for stuck at polarization -1 and stuck at polarization +1, respectively. These defects could be modeled in the QCA Designer by fixing the values of particular QCA cell in a circuit to logic high or logic low permanently.

Figure 5-14 shows the MV logic diagram of a 2-to-4 decoder. As indicated in the figure, an s@1 is induced at one of the input lines, i.e., Line B. The QCA layout is shown in Figure 5-15. Figure 5-16 (a) and (b) shows the expected output waveform and the actual output waveform respectively.

Figure 5-14 MV Logic of a 2-to-4 Decoder with an s@1 Fault in the Input Line B.
Figure 5-15 QCA Layout of a 2-to-4 Decoder with Induced s@1 Fault at the Input Line B.

Figure 5-16 (a) QCA Simulation of an Ideal 2-to-4 Decoder.
The red circles in Figure 5-16 (b) indicate the errors due to stuck at polarization fault, s@1. Due to this defect the truth table of the faulty circuit is reduced as shown in Table 5-1.

### Table 5-1 Truth Table of the Faulty Decoder.

<table>
<thead>
<tr>
<th>B</th>
<th>A</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>Y2</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Y3</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Y2</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Y3</td>
</tr>
</tbody>
</table>

As seen from the above table, because one of the input lines i.e. ‘Line B’ is induced with an s@1 fault the output lines Y₀ and Y₁ are always low as observed from Truth Table 5-1. This was verified from the simulation.

Another example for stuck at polarization fault, s@-1 fault, is shown in Figure 5-17. From the figure, it can be seen that one of the output lines (Y₀) is induced with as@-1 fault. This faulty output line is always low, i.e., ‘0’, irrespective of the inputs.
The QCA layout is shown in Figure 5-18 (a) and the simulation is shown in Figure 5-18 (b).

Figure 5-17 MV Logic of a 2-to-4 Decoder with an $s@-1$ Fault in the Output Line.

Figure 5-18 (a) QCA Layout of a 2-to-4 Decoder with Induced $s@-1$ Fault at the Output Line.
5.6 Implementation of the Proposed BIST Architecture in QCA

Before implementing the BIST model in the proposed CLB design, a slice of CLB was pseudo-exhaustively tested and its output was verified using the truth table. This tested CLB is used as an ideal CLB to test other slices of CLB. The outputs from the TPG are given as the inputs to the decoder and the multiplexer for both the ideal and the test CLB. The ORA compares the outputs from each of the CLB slices and declares the circuit pass/fail. Figure 5-19 shows the block diagram of the BIST model designed and implemented in QCA.
A fault-free CLB was tested using the proposed BIST model. Figure 5-20 shows the QCA layout of this fault-free model and its simulation is shown in Figure 5-21. The components of the BIST model is also indicated in the figure. There is a delay of twenty seven clock cycles at the output. As the above model is a fault-free model, the ORA outputs a logic ‘0’ indicating no faults in the circuit under test.

Figure 5-20 Fault-Free Model of BIST.
Figure 5-21 QCA Simulation of a Fault-Free Model in BIST.
To validate the BIST model, three different test cases have been developed, implemented and analyzed. Specifically, the following fault models and test cases are studied:

Faulty Model 1: Stuck at Polarization fault (s@1) was created at the input of the flip flop in the test CLB.

Faulty Model 2: Stuck at Polarization fault (s@-1) was created at the output lines of the decoder in the test CLB.

Faulty Model 3: Faults caused by the omission of QCA cells was modeled by omitting multiple QCA cells at the output of the multiplexer circuit in the test CLB.

These above test cases are indicated in the block diagram shown in Figure 5-22.

![Figure 5-22 Fault Modeling for BIST Architecture.](image)
Faulty model 1 has been modeled in the QCA Designer. The stuck at polarization fault, ‘s@1’, was created at the input of the flip flop of the test CLB. Figure 5-23 illustrates the fault created in the QCA layout of the test CLB. Stuck at polarization fault i.e. s@1 was created at the input of the flip flop. In the inset, fault modeled in MV logic is also shown.

Figure 5-23 QCA Layout of Faulty Model 1 with BIST.
As the input line of the Flip Flop in the test CLB is stuck at polarization 1, i.e. \(s@1\), the output of the flip flop is always ‘1’ irrespective of the value on the data line. This is true when \(RS = ‘11’\). But for reset and set conditions, this fault would not have any influence and the output value is ‘0’ or ‘1’ based on the set and reset condition. Figure 5-24 shows the simulation of this fault. The faults are indicated by the red rectangular box.
Faulty model 2 was designed and implemented with the stuck at polarization fault (s@-1), introduced at one of the output lines of the decoder. The fault was created in the output line, Y₀. The input to the memory array, m₀, is always a ‘1’ due to this error. This model is shown in Figure 5-25. In the inset, fault is shown in MV logic. The simulation is shown in Figure 5-26.
Figure 5-26 QCA Simulation of Faulty Model 2 with BIST.
The s@1 error induced a faulty output at the output line, Y₀, of the decoder, this propagates through the circuit and this is shown in the red circle in the simulation where the output of the faulty CLB differs from output of the ideal CLB.

Faulty model 3 was designed and implemented in QCA by deleting multiple QCA cells in the multiplexer. The QCA cells were deleted in the last stage of the multiplexer i.e., at the input of 2 x 1 Mux. This model is shown in Figure 5-27. In the inset, fault is shown in MV logic. The simulation is shown in Figure 5-28. It can be observed that the output of the test CLB is unpredictable which is indicated in the figure.
Figure 5-28 QCA Simulation of Faulty Model 3 with BIST.
5.7 Design Parameters and Analysis of Simulation Results

The nano CLB was successfully integrated and simulated with the BIST model. The simulation results were generated using the ‘QCA Designer’ software (version 2.0.3). A bistable approximation was used to simulate the results presented in this research. The radius of effect is 65nm. The total sample ranges are from 12,800 to 64,000 depending on the component of the circuit tested. Distance between two corresponding QCA cells is 2nm as indicated in the QCA Designer. Default values provided by the QCA Designer were used to define other parameters like convergence tolerance, clock amplitude factor, layer separation and maximum iterations per sample.

From the simulation results, it was observed that there was a latency of eight clock cycles from the input of the BIST model to its output. The total latency of the entire BIST model i.e., from the input of the CLB to the output of the BIST model was calculated to be twenty seven clock cycles. Various other design parameters like area, number of QCA cells used and latency are tabulated below in Table 5-2.

Table 5-2 Design Parameters.

<table>
<thead>
<tr>
<th>Module</th>
<th>Area Occupied in μm²</th>
<th>Number of QCA Cells</th>
<th>Latency in Clock Cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Look-Up Table</td>
<td>42.67</td>
<td>11,696</td>
<td>16</td>
</tr>
<tr>
<td>Configurable Logic Block</td>
<td>89.27</td>
<td>24,989</td>
<td>19</td>
</tr>
<tr>
<td>BIST Model</td>
<td>102.8</td>
<td>27,195</td>
<td>27</td>
</tr>
</tbody>
</table>

Table 5-3 gives the statistical data for the various design parameters of the BIST model.
Table 5-3 Design Parameters of the Components in BIST.

<table>
<thead>
<tr>
<th>Module</th>
<th>Area Occupied in μm²</th>
<th>Number of QCA Cells</th>
<th>Latency in Clock Cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td>TPG</td>
<td>1.41</td>
<td>498</td>
<td>2</td>
</tr>
<tr>
<td>ORA</td>
<td>0.13</td>
<td>79</td>
<td>1</td>
</tr>
</tbody>
</table>

The first successful implementation of CLB slice with 5-input LUT along with the implementation of the BIST model at the quantum level using the Quantum-Dot Cellular Automata technology has been presented in this work. The proposed BIST architecture was verified using three faulty models designed and implemented in QCA. The BIST model can successfully detect the stuck at polarization faults and faults caused by omitting multiple QCA cells.
Chapter 6

Summary and Conclusions

6.1 Summary and Conclusions

This chapter is dedicated to the discussion of the results obtained from the work presented. The goal of this work was to model, design, implement and simulate the first novel QCA architecture of a nano configurable logic block with 5-input look-up table which emulates a Xilinx Virtex 5 FPGA architecture. During the design phase, the translation of the gate level to the quantum level logic was done on a one-to-one mapping. In the implementation phase of the research, multi-layer crossover was used to completely eliminate the need for rotated QCA cells. Using this technique enabled us to optimize the digital circuits with respect to both the area occupied and latency. It also eliminates the complexity of fabricating the proposed design in QCA technology by reducing sensitive fabrication errors.

The results obtained from the research were compared to various existing architectures [45] [40]. From the simulation of the CLB with 4-input LUT, it was observed that the write cycle had a latency of sixteen clock cycles as compared to 19 clock cycles in [45] and 134 clock cycles in [40]. The proposed CLB with 4-input LUT uses a total of 11,372 QCA cells covering an area of 3684528 nm². The gap between the two corresponding cells is 2 nm as indicated in the QCA Designer. The results obtained
from the comparison indicate that the proposed design is better than the existing architectures both with respect to area and latency.

In the final phase of the research, the proposed CLB was tested pseudo-exhaustively and a novel Built in Self Test (BIST) model was implemented in QCA in the proposed CLB slice for the first time. The functional blocks of the BIST were implemented in QCA technology and the design was verified and simulated using the QCA Designer software. The proposed BIST model was used to verify the CLB with stuck at polarization faults and faults due to omitted cells successfully.

The simulation results were generated using the ‘QCA Designer’ software (version 2.0.3). Bistable approximation was used to simulate the results presented in this research. The radius of effect is 65nm. The total sample ranges are from 12,800 to 64,000 depending on the component of the circuit tested. Default values provided by the QCA Designer were used to define other parameters like convergence tolerance, clock amplitude factor, layer separation and maximum iterations per sample.

6.2 Contributions

The intellectual contributions made in the research are listed below:

- Successfully designed, modeled, implemented and simulated the first 5-input LUT. A slice of a nano CLB whose architecture closely resembles the CLB architecture of Xilinx Virtex 5 FPGA was successfully implemented and simulated in QCA using the QCA Designer CAD tool. Standard 4-dot QCA cells were used for implementing the physical layouts of all the components.
• Various fabrication faults were modeled and analyzed using the CAD tool.
• Test models injected with stuck at polarization faults in various components of the CLB were developed.
• The proposed architecture of the nano CLB was compared with the existing architectures and found to achieve better design optimization with respect to area occupied, number of QCA cells used, and latency.
• Successfully designed, modeled, implemented and simulated a BIST model for testing the CLB for stuck at polarization faults and faults caused due to cells omitted. The results were later analyzed and confirmed that the BIST model designed achieved 100% fault coverage.

6.3 Future Work

The proposed CLB was translated from the gate level to the QCA logic using a one-to-one mapping which often does not exploit the advantages of the QCA technology. Therefore, optimization during the translation of the gate level to the MV logic by eliminating redundant logic should be investigated.

The QCA Designer tool has many limitations. Research is needed to develop other tools which will give the user better control in the design, layout, and simulation of QCA circuits.

Research is also needed for developing new fault models for testing other defects in QCA circuits which have not been covered in this work. For example, the proposed BIST model can be modified to test transient faults during normal operation apart from permanent faults caused during the fabrication process.
References


[35] Data flow in molecular QCA: Logic can “sprint,” but the memory wall can still be a “hurdle” (2005).


[38] Semiconductor Industry Association International Technology Roadmap for Semiconductors 1999 (Sematech, Austin, TX, 1999)


Appendix I

QCA Designer Manual

Source: http://www.mina.ubc.ca/qcadesigner_manual

1.0 Adding Objects

1.1 Creating Cells

QCA Designer provides several methods for adding cells to your design. When cells are created they are placed into the currently selected cell layer. Their properties are set when a new layer is created. You can also edit cell properties for a particular layer by bringing up the layer properties dialog from the layers toolbar.

To add an individual cell to your design, click the Cell button, then click in the design area and a cell will be created at the click point.

To add an entire QCA wire at a time, you may use the array tool. Click the Array button then click and drag in the design area. When you release the left mouse button, the array of cells will be created.

Note: QCA Designer does not allow cells to overlap on the same layer (they can overlap on different layers). If you create an array that overlaps in some points, only those cells that do not overlap will be created.
1.2 Adding Labels

To add a label to a drawing objects layer, click and click the place on the design where you would like it to appear. You can double-click on the label to change its text.

1.3 Adding Substrates

Though it's very rarely useful, you can add substrates to designs and remove them from designs. To add a substrate to a design, select an active substrate layer and click . Then, click the design at the point where you would like the top left corner of the substrate to appear and drag the mouse to stretch the new substrate to its desired width and height.

2.0 Bus Layout

As layouts become more complex, you may wish to group inputs and outputs logically into buses. A bus is simply a named collection of inputs or of outputs. Once your design contains inputs and/or outputs, you can group them into buses using the bus layout dialog. You can invoke this dialog by choosing "Main Menu: Tools→Bus Layout...", or by clicking on the layers toolbar. The dialog box is shown in Figure A-1
Figure A-1 Dialog Box for Managing Bus Layers.

For example, to create an input bus, select several inputs listed on the left, and click "Create Bus". You can hold down Ctrl or Shift to select multiple inputs.

Using the buttons on the right, you can raise the significance of cells within a bus, and you can modify the order the buses are stored in. This is the order the simulation results will be displayed in. Inputs and outputs not grouped into buses remain at the end of the list.

3.0 Clocking

QCA Designer provides several means by which you set the clocking for your designs. However, before you can change the clocking, you must first make a selection. Once you have a selection you can do any of the followings to change the clocking on your selection:

To increment (modulo 4) the clock of every cell by one, choose "Tools→Increment Cell Clocks" from the menu, or press Ctrl+I.

Menu: Tools → Increment Cell Clocks
To assign a selection of cells to a single clock, use the clock select pull down in the layers toolbar.

Step 1: Make a selection as shown in Figure A-2.

![Figure A-2 Cell Selection for Clock Assignment.](image)

Step 2: Choose the clock from the clock select pull down as shown in Figure A-3.

![Figure A-3 Clock Assignment for Selected Cells.](image)

Step 3: Release the selection as shown in Figure A-4.

![Figure A-4 Cells with Clock Assigned.](image)
You can set default clocking zone to something other than 0 in the layer properties dialog.

4.0 Creating and Importing Blocks

The ability to manipulate blocks of cells allows you to import other designs into the current one, as well as create blocks from components of the current design.

4.1 Create Block

To create a block, select all the objects in the design you wish to use in the block. Once you have selected all the objects, click Create Block from the Tools menu as shown in Figure A-5. This will allow you to choose a filename for the block consisting of all your selected objects.

TIP: You can use SHIFT to add to the selection, as well as CTRL to remove from the selection.

![Create Block on Toolbar](image)

Figure A-5 Toolbar Selection to Create a Block.

4.2 Import Block

You can also incorporate any **QCA Designer** design file into the current design. It will become a new selection you can drag to a place of your choice.
Before an imported block can become a new selection in your design, you must specify, for each layer in the block, which design layer it is to be merged with. The layer mapping dialog pops up after you have chosen the design file, as shown in Figure A-6, to merge with your current design.

![Map Layers dialog](image)

Figure A-6 Managing Layers in Multi-Layer Crossover.

Therein, for each layer present in the imported block, you can either choose a design layer to merge it with, or you can choose to place the block layer into a newly created design layer. To choose a destination layer wherein to merge a given block layer, click on the layer in the "Destination Layer" column. You can now choose from a list of available destination layers.

5.0 Managing Layers

Layers allow for a separation of objects into different groups which can be manipulated together. They also separate different objects by virtue of their type, i.e. cells go into cell layers. Layers exist for cells (Cell Layer Icon), drawing objects (Drawing Layer Icon), substrates (Substrate Icon), etc. Each layer maintains a set of template objects for each object type it may contain. When you create new objects, they will have the same properties as these templates. Set the
properties for the template objects by clicking on the Layer Properties button. The present listing of layers can be viewed by pulling down the layers pull down as shown in Figure A-7.

As you can see, the present design consists of seven layers. Notice that the last four layers are cell layers and are displayed with the icon. When a layer is set visible, the objects in that layer are displayed in the design area. When a layer is set active, the user can manipulate the objects in that layer. Of course when a layer is not visible it is also not active.

5.1 Layer Ordering

The order in which layers appear in the layers list is important for cell layers because it determines their physical relation to one another, i.e. one layer of cells is directly above another but below yet another. To reorder the list of layers click the reorder button in the layers toolbar. Once the layers list pops up you can drag and drop layers to achieve the correct order as shown in Figure A-8.
The cell layer which appears at the top most position in the list is considered to be at the lowest physical location. Layers appearing below this layer are considered to be physically stacked on top of this layer.

5.2 Creating a New Layer

To create new layers click the button in the toolbar. The layer properties dialog will pop up as shown in Figure A-9.
In this dialog you can set the layer status, the layer description (which is the layer name), as well as the default properties for objects in that particular layer. When satisfied with the new layer, click OK.

5.3 Deleting a Layer

To delete the presently selected layer click the button.

5.4 Concentrating On a Layer

When working for a protracted period of time with a certain layer, you may want to make sure that no other active layer interferes with your work. To this end, you can quickly hide all layers except the selected one, by pressing Ctrl+H, or by choosing "Main Menu: Tools→Hide All But The Selected Layer".

6.0 Manipulating Objects

You can select objects on any active layer.

You can select a single object by left-clicking on it.

You can select any group of objects by left-clicking on the workspace and dragging the mouse to create a selection rectangle. The objects contained in that selection rectangle will be highlighted in red. There are many things you can do with a selection.

You can add objects to your selection by holding down Shift and creating a selection rectangle, as above. Deselected objects in the rectangle will be selected.

You can remove objects from your selection by holding down Ctrl and creating a selection rectangle. Selected objects in the rectangle will be deselected.
You can create a copy of your selection by clicking "Copy". A copy of the selected objects will be created, unless your current selection cannot be dropped. If the location where you want to place the copy is off-screen, you can use the mouse wheel to scroll there. Alone, the mouse wheel scrolls the workspace up→down. To scroll left→right, hold down Ctrl while scrolling.

To mirror a selection of objects, first select the objects you would like to mirror. Then, click "Mirror". From the menu that pops up, choose either "Horizontally", if you want to mirror across the x-axis, or "Vertically", if you want to mirror across the y-axis.

To rotate a block of objects by 90°, first make a selection, and then click "Tools→Rotate Selection" as shown in Figure A-10.

![Figure A-10 Toolbar Selection for Rotating Cells.](image)

Your selection will be rotated 90°. Click the menu item again and again to rotate by 180° and 270°, respectively.
TIP: To rotate a copy of your selection, first make a selection, click "Copy", and then click "Tools→Rotate Selection".

To delete a selection of objects, first make such a selection. Then press the "Delete" key to delete your selection.

When you drag a selection and your objects are snapped to the grid, you cannot move those objects around arbitrarily. However, by choosing Translate, you can move a selection by an arbitrary number of nanometers.

To accommodate multilayer circuits, it is now possible to change the appearance of cells. This does not affect their simulated behavior. It is merely a means of visually identifying cells overlapping on different layers. To modify the appearance of selected cells, click Alt Style and choose from among 3 appearances from the popup menu.

TIP: In the cell libraries, we observe the following convention:

Cells on the main layer are left unaltered, except for those underlying "via" cells.

Cells on "via" layers and those accessing "via" layers are drawn using the circular appearance.

Cells on "crossover" layers are drawn using the "X" appearance, except those accessing "via" layers.

This creates crossovers according to the convention shown in Figure A-11:
7.0 Moving Around

You can use the mouse to move around QCA Designer. To scroll your design vertically, simply turn the mouse wheel. To scroll your design horizontally, hold down Ctrl while turning the mouse wheel.

A "Pan" tool is also available. To use it, click Pan. The shape of the mouse cursor will become Pan. Click and drag a point on the design to the location you want to have it appear at.

TIP: To scroll your design such that a specific QCA cells appears in a specific location relative to the rest of the QCA Designer window, click "Pan" and simply "grab" the design by that cell.

TIP: To pan large distances quickly, say, to the right, pan from near the right edge of the workspace to near the left edge. This is often faster than using the mouse wheel and is the only option if the mouse doesn't have a wheel.

8.0 Object Properties
Most QCA Designer objects have associated properties. To modify them, double-click on the object.

8.1 Cell Functions

The function of a cell can be changed by double-clicking on the cell. This will bring up the cell function dialog as shown in Figure A-12:

![Cell Function Dialog Box](image)

Figure A-12 Dialog Box for Manipulating Cell Function.

8.1.1 Normal Cell

A normal cell has no special function other than to exist and switch according to the influence of neighboring cells.

8.1.2 Fixed Polarization

A fixed polarization cell will remain at the chosen polarization during the simulation regardless of the effect of neighboring cells or the clock.
8.1.3 Input Cell

Input cells will have a polarization that is determined by the simulation engine. The input values will be set according to the Simulation Type. You can simulate all possible input combinations by choosing the Exhaustive Verification or select input vectors manually using a Vector Table.

8.1.4 Output Cell

Output cells act just like normal cells in that they are directly affected by their neighbors. The polarization of the output cells is recorded throughout the simulation and plotted in the graph dialog when the simulation is complete.

8.1.5 Clock

This allows you to change the clock for the selected cell.

8.2 Substrate properties

The grid spacing a substrate maintains can be changed by double-clicking on the substrate and specifying the desired grid spacing in its property dialog as shown in Figure A-13.

![Grid Spacing Dialog Box](image)

Figure A-13 Dialog Box to Change Substrate Properties.
8.3 Label properties

The text of a label can be modified by double-clicking on it and typing it into its property dialog as shown in Figure A-14.

Figure A-14 Dialog Box to Change Label Properties.

9.0 Exhaustive Verification

The exhaustive verification simulation type will simulate all possible combinations of the input vectors. If you have \( n \) inputs, then this will create and simulate all \( 2^n \) possible vectors in increasing order. To select the exhaustive simulation type click "Simulation→Simulation Type Setup", and make sure that Exhaustive Verification is selected as shown in Figure A-15, and click OK.

Figure A-15 Exhaustive Verification Selection.

10.0 Vector Table

The vector table simulation type allows you to specify the input vectors, as well as the order in which they are applied to the inputs of your circuit. The vector table simulation is only available if you already have cells designated as inputs in your circuit. For each available input you can specify a binary value in each vector. To select the
Vector Table simulation type clicks "Simulation→Simulation Type Setup", and make sure that Vector Table is selected as shown Figure A-16.

You will notice that all the inputs in your design are indicated with their associated names, and they are grouped into the buses you created for them. Each of the inputs is Active by default. You can make any input Inactive by toggling the Active checkbox as shown in Figure A-17. When an input is Inactive it is simulated as if it were just a regular cell in the design; i.e. it is allowed to switch with the influence of its neighbors. To append vectors to the end, click the button. You can also insert a new vector in front of an existing one by clicking on the column header of the existing one and clicking on the button:

![Figure A-16 Vector Table](image)

Figure A-16 Vector Table
Now, to specify the value of a vector, you can either specify the value of each bus, or you can check the box next to each input to set individual bits high or low as shown in Figure A-18.
To delete a vector, click on its column header and click the button.

When you have completed your vector table you may save it using the button. You can also open and use a previously saved vector table. If the other vector table contains fewer inputs than your design then each extra input not loaded from the vector table will be padded with 0s. On the other hand, if the vector table contains more inputs, it will be truncated. Click the button to use the vector table. The next simulation you perform will be based on the chosen vectors. Note: if you add or remove inputs to your design, make sure to go back and edit your vector table.

11.0 Engine Options

The bistable engine options are shown in Figure A-19.

![Bistable Engine Options](image)

Figure A-19 Bistable Engine Options.
11.1 Number of Samples

The total simulation is divided by the number of samples. For each sample, the simulation engine looks at each cell and calculates its polarization based on the polarization of its effective neighbors (determined by the radius of effect). The larger this number the longer the simulation will take to complete. However, if you choose too small a number you may not get the expected results, because there will not be sufficient samples during input transitions. If you suspect that your results should be something other than what they are, it is recommend that you try to increase this number.

TIP: If other parameters are default, try setting the number of samples to 1000 times the number of vectors you are simulating. For exhaustive simulation set to 2000 times $2^n$, where $n$ is the total number of inputs in your design.

11.2 Convergence Tolerance

During each sample, each cell is converged by the simulation engine. The sample will complete when the polarization of each cell has changed by less than this number; i.e. loop while any design cell has $(\text{old}\_\text{polarization} - \text{new}\_\text{polarization}) > \text{convergence}\_\text{tolerance}$.

11.3 Radius of Effect

Because the interaction effect of one cell onto another decays inversely with the fifth power of the distance between cells, we do not need to consider each cell as affecting every other cell. This number determines how far each cell will look to find its neighbors. Make sure that at least the next-to-nearest neighbors are included in this
radius. If you only allow for the nearest neighbor then do not expect designs with coplanar crossovers to work. Figure A-20 should help clear this up.

![Figure A-20 Radius of Effect.](image)

Note that with multilayer capability the radius of effect is extended into the third dimension. Therefore in order to include cells in adjacent layers, make sure that the layer separation is less than the radius of effect.

11.4 Relative Permittivity

The relative permittivity of the material system you want to simulate. For GaAs/AlGaAs it is roughly 12.9 which is the default value. This is only used in calculating the kink energy.