Defects and Faults in Quantum Cellular Automata at Nano Scale

Mehdi Baradaran Tahoori, Mariam Momenzadeh, Jing Huang, Fabrizio Lombardi
Department of Electrical and Computer Engineering,
Northeastern University, Boston, MA, 02115
{mtahoori, mmomenza, hjing, lombardi@ece.neu.edu}

Abstract

There has been considerable research on quantum dot cellular automata (QCA) as a new computing scheme in the nano-scale regimes. The basic logic element of this technology is majority voter. In this paper, a detailed simulation-based characterization of QCA defects and study of their effects at logic-level are presented. Testing of these devices is investigated and compared with conventional CMOS-based designs. Unique testing features of designs based on this technology are presented and interesting properties have been identified.

1. Introduction

As the CMOS technology approach it's fundamental physical limits, there has been extensive research in recent years in nanotechnology for future generation IC. It is anticipated that these technologies can achieve a density of 10^{12} devices/cm^2 and operate at THZ frequencies. Among these new devices, *quantum dot cellular automata* (QCA) not only gives a solution at nano scale, but also it offers a new method of computation and information transformation [15]. In terms of feature size, it is projected that a QCA cell of few nanometer size can be fabricated through molecular implementation by a self-assembly process. One of the fundamental issues in the testing community is the radical shift in computation and fabrication technology and its effect on the test flow. Since the manufacturing process for nano devices is ill-defined, it is extremely difficult to address manufacturing testing problems. However, it would be inappropriate to ignore testing of these device till manufacturing state.This paper tries to address this issue for one of the proposed trends in nanometer era.

For QCA, the cells must be aligned precisely at nano scales to provide correct functionality, so proper testing of these devices for manufacturing defects and misalignment plays a major role for quality of QCA based circuits. The basic logic element in this technology is the majority voter. Since the basic logic elements of QCA-based designs are different from conventional CMOS designs, they need different testing schemes.

In this paper, the defect characterization of these devices has been extensively studied; effects of defects are investigated at logic-level . Also, testing of QCA is compared with testing of conventional CMOS implementations. Defect injection is exploited to study the behavior of QCA-based circuits in the presence of defects and to measure the effectiveness of different test sets in detecting them. The approach proposed in this work is based on simulating different manufacturing misalignments, investigating their effects at logic level and identifying the test vectors for detection of all faults. Different fabrication schemes of the majority voter at cell level are studied; these different implementations are compared in terms of defect tolerance and testability. Although in the current CMOS process only a small portion of the actual defects behaves like stuck-at faults, the stuck-at fault model is still widely used as the test sets generated based on this model are quite acceptable. So it is possible to investigate effectiveness of stuck-at test sets for QCA defects even though QCA defect mechanisms cannot be modeled by the stuck-at model. This is addressed in this paper.

The rest of this paper is organized as follows. In Sec. 2, a review of QCA is presented. In Sec. 3, testing of QCA-based design at logic level is discussed. In Sec. 4, the defect characterization of QCA is presented. In Sec. 5, test set, defect and fault coverage are discussed. Finally, Sec. 6 concludes the paper.

2. Review

QCA is a novel nano device that stores logic states not as voltage levels but rather based on the position of individual electrons. A quantum cell can be viewed as a set of four charge containers or dots, positioned at the corners of a square. The cell contains two extra mobile electrons which can quantum mechanically tunnel between dots, but not cells. The electrons are forced to the corner positions by Coulomb repulsion. The two possible polarization states represent logic 0 and logic 1, as shown in Fig 1. Unlike conventional logic in which information is transferred from one device to another by electrical current, QCA does so by Coulomb interaction which connects the state of one cell to the state of its neighbors. This results in a technology in which information transfer (interconnection) is the same as information transformation (logic manipulation). Figure 1 illustrates the cell-cell response function, where the polarization P1 is induced in cell 1 by the fixed polarization of its neighbor P2 [18]. P$= +1$ and P$= -1$ states indicate logic values "1" and "0" respectively. Power dissipation in QCA circuits is ultra low compared with conventional CMOS circuits [15][18][19].

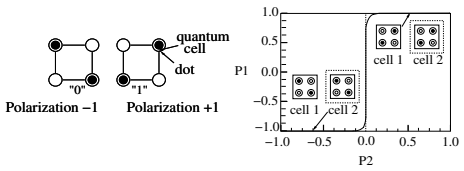

Fig. 1. QCA Cell and Cell-cell response [18]

The basic logic gate in QCA is the *majority voter* (MV). The

majority voter with logic function MV(A, B, C) = AB + AC + BC, can be realized by only 5 QCA cells (compared to a CMOS implementation which requires 16 transistors), as shown in Fig. 2(b). Logic AND and logic OR functions can be implemented from a majority voter by setting one input permanently to 0 and 1, respectively. The QCA Inverter is shown in Fig. 2(a). Unlike conventional CMOS in which it is the simplest block, it consumes considerable area in QCA. The binary wire (as interconnect) and the inverter chain are shown in Fig. 2(c)(d).

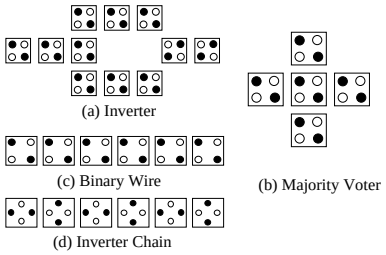

Fig. 2. QCA devices

The concept of clocking for QCAs has been introduced in [5]. Some designs based on QCAs including microprocessors, FPGA and memories have been proposed [13][14][20][22][7]. A study of the fault tolerant properties of the majority voter under some manufacturing misalignments [3][4][6] show that MV is more vulnerable to misalignment in the vertical direction than in the horizontal direction. A misalignment (at least equal to half a cell width in the vertical direction) causes the MV to malfunction. Based on this simulation-based study, a fault tolerant MV block has been proposed.

Currently, micro-sized QCA devices have been fabricated with metal cells which operates at 50mK [15][1]. In [15], an experimental demonstration of a basic QCA cell composed of four metal dots, connected with tunnel junctions and capacitors is presented. In [2], building of basic logic elements with these cells is demonstrated. It is anticipated that molecular scale (∼2nm) yields operation of QCA at room-temperature. In [11], some possible molecular realizations of QCA have been proposed. It describes the progress toward making QCA molecules and advances for surface attachment chemistry compatible with QCA.

3. Logic-Level Testing

The overall structure of the QCA implementation of (combinational) logic designs is shown in Fig. 3. The block consists of an interconnection of majority voters and inverters. There are two system-level control lines, U_0 and U_1, which are connected to majority voters. U_0 is connected to logic "0" and sets some majority voters as the AND function, whereas U_1 is connected to logic "1" and sets other majority voters as the OR function. A simple example is shown in Fig. 4. These control lines can provide more controllability since these lines can be seen as extra input lines during test time. This unique feature of QCA can be exploited to achieve better testability.

Since logic designs are implemented as a network of majority voters and inverters (as the universal logic set) in QCA technology, it is important to investigate the properties of these network, especially for test execution. As shown through the following statements, these networks have unique and interesting

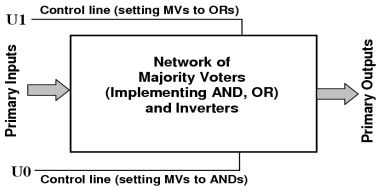

Fig. 3. The QCA implementation of logic networks using majority voters (implementing AND and OR) and Inverters

testing features which cannot be achieved in conventional CMOS implementations.

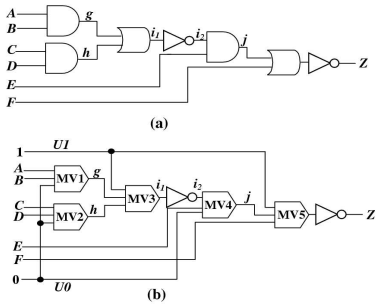

Fig. 4. (a) a simple AND-OR logic (b) MV-based implementation

Consider a majority voter with input lines A, B, and C, and the output line Z ($Z = AB + AC + BC$).

Property 1. *Consider a majority voter with input values a, b, and c, (for lines A, B, and C, respectively) and output z. If the all inputs are flipped, $abc \rightarrow a'b'c'$, the output will be also flipped, $z \rightarrow z'$.*

Note that this is not the case for other logic functions such as AND, NOR, etc. For example, consider a three input AND gate with inputs 100 and output 0. If the inputs are flipped to 011, the output will remain 0.

Property 2. *If there is inversion at any input and/or the output of the majority voter, property 1 still holds.*

Property 3. *Consider a majority voter with input pattern abc (for lines A, B, and C, respectively). The stuck-at-v fault on any input or output line of the voter is detectable (fault effect appears at output line) by abc if and only if the stuck-at-v' fault on that line is detectable by $a'b'c'$.*

Proof. Consider l stuck-at-v fault. If l is an input line, consider the l is A, without loss of generality. The fault is detected if and only if the value of a is v' and the other inputs, b and c, have opposite values. As a result, a' is v and b' and c', have opposite values. Hence, $a'b'c'$ detects the l stuck-at-v'.

Again, this property does not hold for other logic functions. As an example, consider an AND gate with test vector 11 which detects stuck-at-0 at the top input (and the bottom input too). The complement of this vector, 00, does not detect any single stuck-at-1 on the inputs.

Property 4. *If there are some inversions at any inputs and/or the output of the majority voter, property 3 still holds.*

The interesting property of majority voters is that the above properties hold for any arbitrary network of majority voters (including inverters).

Property 5. *Consider an arbitrary network of majority voters (and inverters) with primary input vector V. If all bits of V are flipped, $V \rightarrow V'$, all nodes in the network will be flipped.*

Proof. The proof is based on induction on the level (distance) of each majority voter in the network from the primary inputs, by forming a topological order of the majority voters in the network. The step of induction is property 2.

Property 6. *Consider an arbitrary network of majority voters (and inverters) with primary input vector V. For any node n in the network, n stuck-at-u is detected by V, if and only if n stuck-at-u' is detected by V'.*

Proof. The proof is similar to the proof of property 5. The step of induction is property 4.

Property 5 and 6 are very interesting and proved unique features of a network of MVs (and inverters). Based on property 5, the test vector pair (V, V'), where V is any arbitrary vector, causes a transition on all nodes of the network. Also, the three vectors (V, V', V) cause both fall and rise transitions on all nodes in the network. Hence, a 100% toggle fault coverage test set is applicable.

Based on property 6, the fault list for any network of majority voters (and inverters) can be divided into two parts: just one fault per each node, because if a vector V detects one stuck-at fault on that node, V' will detect the other stuck-at fault on that node. As a corollary, this feature can be exploited to reduce the size of the fault list, and hence ATPG execution, for the control inputs (to be generated by ATPG) into half.

To generate tests for stuck-at faults in a network of MVs and inverters, conventional (combinational) ATPG tools can be exploited. The network of MVs and inverters is first transformed into a hierarchical gate-level netlist. Each MV is replaced by a hierarchical cell implementing the majority function. We only consider pin faults on the inputs of these hierarchical cells which correspond to the inputs of MVs. As explained above, only half of the pin faults must be considered for the test generation.

4. Defect Characterization

In this section, the robustness of the QCA majority voters and binary wires, as well as some QCA circuits is investigated. The basic functionality of a QCA device is based on the Coulombic interaction among neighboring QCA cells (depending on the accuracy and geometry of its implementation). Various configurations of QCA devices have been studied using the QCADesigner[1] v1.20 simulation tool. For accuracy, the bistable model is employed. This is a quantum mechanical engine using the Jacobi algorithm to calculate the eigenvalues/vectors of the Hamiltonian matrix.

4.1. Defect and Failure Modes

To perform a defect characterization of QCA devices and circuits and study their effects at logic-level, appropriate defect mechanisms and models must be considered which 1) can be simulated using the available simulation tool and 2) be realistic for manufacturing and fabrication defects. According to [10], in the present stage of QCA manufacturing, defects are possible in both synthesis phase and deposition phase. Manufacturing defects may cause a cell to have missing or extra dots or/and electrons. This will be fatal to the correct operation of the cell. However, defects are much more likely to occur in the deposition part than in the synthesis part which will result in cell misplacement. A missing dot (or additional dot) is very unlikely due to the ease of purification of small inorganic molecules [10]. For example, Nuclear Magnetic Resonance (NMR) has an estimate minimum purity of 99% for model compounds such as the Creutz-Taube Ion (a 2-dot cell model). Moreover, electrochemical measurements for the CT Ion show that fewer than one molecule in 10^5 are in the incorrect charge state. Yet placing the individual cells in specific locations during the deposition part is difficult, various types of cell misplacement faults may occur such as cell misalignment, rotated cells, etc. Therefore in this paper we assume that all the cells are perfectly manufactured and operate correctly and study the effects of following types of cell misplacement faults:

- A *cell displacement* is a defect in which the defective cell is misplaced from its original direction. Several cell displacement defects are shown in Fig. 5.
- In a *cell misalignment* defect, the direction of the defective cell is misplaced. Some examples of cell misalignments are shown in Fig.6.
- In a *cell omission* defect, a particular cell is missing as compared to the original (defect-free) arrangement. The electron missing defect where the defective cell have no electrons can be modeled by this type of defects.

In this work, the following defects are considered and simulated for QCA devices: all possible combinations of displacement of cells with respect to the central cell under different distances, misalignment of cells in different directions. For QCA circuits, cell omission defects are also simulated.

4.2. Majority Voter Defect Analysis

A defect free majority voter which has a $5nm$ dot size, a $20nm \times 20nm$ cell size , with a $5nm$ cell distance is considered, as shown in Fig. 5(a). Different defects in the majority voter, including cell displacement and misalignment have been considered and simulated. The results for cell displacement and misalignment are shown in Table 1 and Table 2, respectively. Only faulty entries are shown in the tables, in the form of (fault-free/faulty) values.

The data shows that in most cases the horizontal input cell (i.e. cell B) is dominant; this cell seems to have a bigger impact on the center cell than A and C. For misalignment, any single cell misalignment greater or equal to half a cell causes malfunction (fault at logic-level). In some cases the fault margin is smaller. MM:{please check the following:} A comparison between misalignment and displacement defects illustrates that the misalignment defects have more catastrophic effects on the functionality of a majority voter , with the same defective-distance as the displacement defect.

[1]QCADesigner is the product of an ongoing collaboration between the University of Calgary ATIPS Laboratory and the University of Notre Dame.

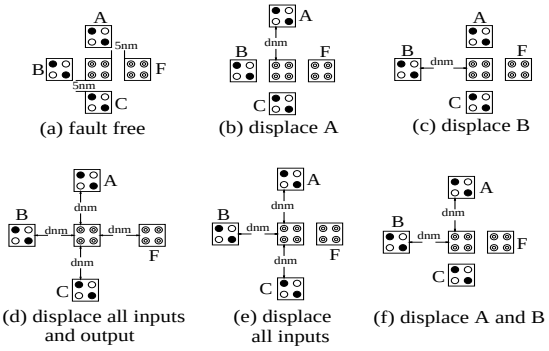

Fig. 5. Displacement in Majority Voter

Table 1
Results for Displacement in Majority Voter

displace cell A: fig 5(2)	
$d \leq 15nm$ Normal Operation	$d \geq 20nm$, F=B

displace cell B: fig 5(3)	
$d \leq 40nm$ Normal Operation	$d \geq 45nm$
	A B C | F
	001 | Z (no polarization)
	011 | Z(no polarization)
	100 | Z (no polarization)
	110 | Z (no polarization)

displace all input/output cells: fig 5(4)	
$d \leq 10$ or $30 \leq d \leq 40nm$ Normal Operation	$15 \leq d \leq 25nm$
	A B C | F
$d \geq 45nm$ F=Z (no polarization)	010 | 0/1
	101 | 1/0

displace all input cells: fig 5(5)	
$d \leq 15$ or $d = 40nm$ Normal Operation	$d \geq 45nm$ F=Z (no polarization)
$20 \leq d \leq 25$ or $d = 35nm$	$d = 30nm$
A B C | F	A B C | F
010 | 0/1	000 | 0/1
101 | 1/0	010 | 0/1
	101 | 1/0
	111 | 1/0

displace cells A and B: fig 5(6)	
$d \leq 5nm$ Normal Operation	$d \geq 10nm$, F=C

Table 2
Results for Misalignment in Majority Voter

move A toward west: fig 6(1)	
$d \leq 5nm$ Normal Operation	$d \geq 10nm$, F=B

move A toward east: fig 6(2)	
$5 \leq d \leq 15nm$	$d = 20$ or $d = 30nm$
A B C | F	Normal Operation
001 | 0/1	
010 | 0/1	$d = 25nm$
101 | 1/0	F=A
110 | 1/0	

move C toward west: fig 6(3)	
$d \leq 5nm$ Normal Operation	$d \geq 10nm$ F=B

move C toward east: fig 6(4)	
$5 \leq d \leq 15nm$	$d = 20$ or $d = 30nm$
A B C | F	Normal Operation
010 | 0/1	
011 | 1/0	$d = 25nm$
100 | 0/1	F=C
101 | 1/0	

move A,C toward west: fig 6(5)	
$d \geq 5nm$ F=B	

move A,C toward east: fig 6(6)	
$d = 5, 20, d \geq 30nm$ F=B	$10nm \leq d \leq 15nm$
	A B C | F
$d = 25nm$ Normal Operation	000 | 0/1
	010 | 0/1
	101 | 1/0
	111 | 1/0

move B toward south/north: fig 6(7)	
$d \leq 5nm$ Normal Operation	$d \geq 45nm$
	A B C | F
	001 | 0/1
	011 | 1/0
	100 | 0/1
	110 | 1/0

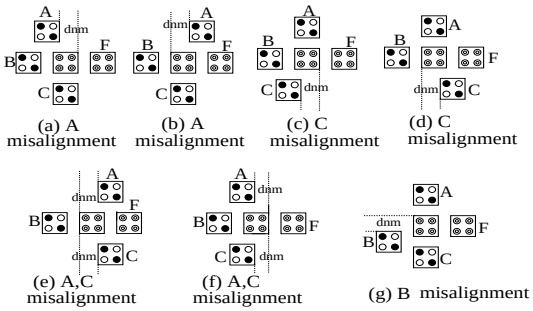

Fig. 6. Misalignment in Majority Voter

4.3. Binary Wires and Inverter Chains

The effect of cell displacement defects on two parallel binary wires as well as two parallel inverter chains have been investigated.

4.3.1. Double Binary Wires: Two defect-free binary wires are shown in Figure 7(a); these wires are denoted as the upper wire ($i1$ to $o1$) and the lower wire ($i2$ to $o2$). The cells used in this simulation have a size of $20nm \times 20nm$, and the dot diameter is $5nm$. In the defect-free case, the cells in the same wire are separated by $15nm$. The distance between the wires is $60nm$.

The displacement defects are simulated by moving one or two cells in the lower wire toward the upper wire (by a displacement d) as shown in Figure 7(b). The simulation results are shown in Table 3. These results show that in most cases the lower wire is dominated by the upper wire. $o1$ and $o2$ are either equal to $i1$ or $i1'$, depending on which cell(s) are displaced and the value of the displacement, d. In most cases, the upper wire functions normally, i.e. $i1 = o1$. However, it can be observed that in some cases the upper wire behaves as an inverter. Clearly, unlike CMOS designs, the coupling defects at QCA device-level do not behave as the *wired bridging fault* model. However, these defects manifest themselves as a dominant model (at logic level) in which the output of a wire is determined by the value of the coupled wire.

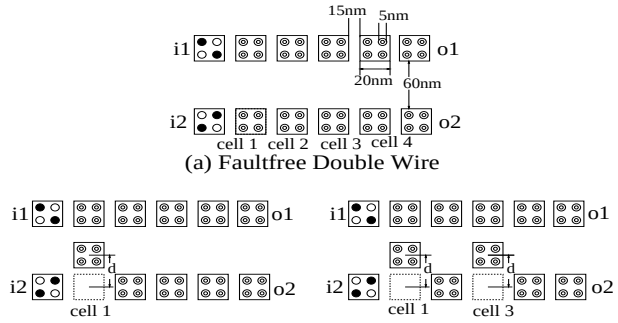

Fig. 7. Displacment in Binary Double Wire

4.3.2. Double Inverter Chains: The double inverter chain is shown in Figure 8(a). The simulation results for moving one of the cells in the bottom wire toward the upper wire, with displacement d, (as shown in Figure 8(b)) are presented in Table 4. The displacement defects behave as according to

Table 3
Results for Double Binary Wires

move cell1 OR cell2		
$d \leq 40nm$ Normal	$d = 45 - 50nm$ $o1 = i1, o2 = i1$	$d \geq 55nm$ $o1 = i1, o2 = Z$
move cell3 OR cell4		
$d \leq 35nm$ Normal	$d = 40 - 50nm$ $o1 = i1, o2 = i1'$	$d \geq 55nm$ $o1 = i1, o2 = Z$
move cell1 AND cell2		
$d \leq 35nm$ Normal	$d = 40 - 50nm$ $o1 = i1, o2 = i1$	$d \geq 55nm$ $o1 = i1, o2 = Z$
move cell1 AND cell4; OR move cell 2 AND cell 3; OR move cell3 AND cell4		
$d \leq 35nm$ Normal	$d = 40 - 50nm$ $o1 = i1, o2 = i1'$	$d \geq 55nm$ $o1 = i1, o2 = Z$

move cell1 AND cell3			
$d \leq 35nm$ Normal	$d = 40 - 50nm$ $o1 = i1, o2 = i1$	$d = 45nm$ $o1 = i1, o2 = i1'$	$d \geq 55nm$ $o1 = i1, o2 = Z$

move cell2 AND cell4				
$d \leq 15nm$ Normal	d=20-25nm d=40-45nm $o1 = i1$	d=30-35nm $o1 = i1$ $o2 = i1$	$d = 50nm$ $o1 = i1'$ $o2 = i1$	$d \geq 55nm$ $o1 = i1$ $o2 = Z$

Table 4
Results for Double Inverter Chains

Fault Free: $o1 = i1'; o2 = i2'$		
move cell1 OR cell2 OR cell3		
$d \leq 35nm$ Normal	$d = 40nm - 50nm$ $o1 = i1', o2 = i1'$	$d \geq 55nm$ $o1 = i1', o2 = Z$
move cell4		
$d \leq 30nm$ Normal	$d = 35nm - 50nm$ $o1 = i1', o2 = i1'$	$d \geq 55nm$ $o1 = i1', o2 = Z$

the *dominating bridging fault* model at logic level. Moreover, a comparison with the binary wires shows that the binary wires are more defect tolerant than inverter chains in the case of displacement coupling defects.

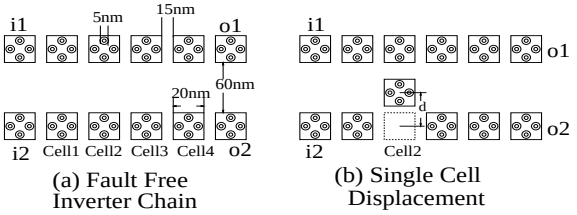

Fig. 8. Displacment in Double Inverter Chains

4.4. Defects and Faults in a Full-Adder

A QCA implementation of a full adder using three majority voters and two inverters is shown in Fig.9. The corresponding QCA layout is shown in Fig.10 which contains 145 cells. The cells are 18nm × 18nm with dot size of 5nm. 40 different single cell omission defects have been simulated in this circuit.

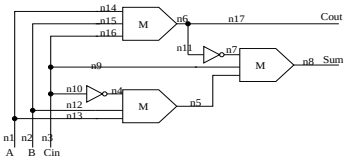

Fig. 9. One-bit QCA full adder

4.4.1. Defects in Wires and Inverter Chain: Removing a single cell from a binary wire does not affect its functionality at logic-level although it may result in some delay faults. However, a single cell omission in a wire implemented as an inverter chain results in an unwanted complementation at the output of the chain. Those binary wires which change direction in the layout (e.g. L shape) are very sensitive to the defects on the corner cells. Cell omission defect at the corner cell is equivalent to unwanted complementation fault at logic-level.

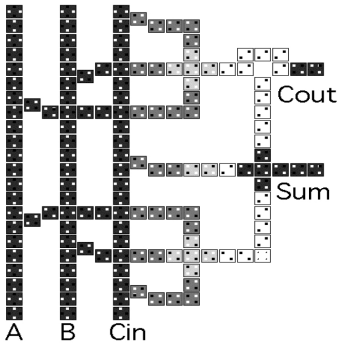

Fig. 10. One-bit QCA full adder layout

4.4.2. Defects in Wire Crossing: In QCA implementation, two different wires (horizontal and vertical) can cross each other in the same layer (*co-planar wire crossing*). In this case, one of them is implemented as a binary wire, while the other one is implemented as an inverter chain (i.e. the cells in the other wire are rotated). In the fault-free case, the wires are unaffected by each other and can carry different signal values.

However, this structure is very vulnerable to cell omission defects at or near to the crossing point. The cell omission defect at the cross point results in an unwanted complementation on the inverter chain and the binary wire is *dominated* by the faulty value of the inverter chain (dominating bridging fault). Cell omission defects for the cells adjacent to the crossing point have similar effects, i.e. the value of the binary wire is dominated by the faulty value of the inverter chain.

4.4.3. Defects in the Majority Voter: The results of defects in a majority voter of the full-adder is consistent with the defect characterization results for a single majority voter: the horizontal input has more impact on the output than the vertical inputs. Cell omission defect on the horizontal input cell does not affect the functionality. However, a cell omission defect on any of vertical inputs causes the output to be dominated only by the horizontal input, i.e. the output is shorted to the horizontal input.

The cell omission defect on the center cell of a majority voter with vertical input values a and b, and horizontal input c changes the function to be the majority of a', b', and c. This can be interpreted as unwanted complementation faults on both vertical inputs.

5. Test Sets Coverage and Fault Model

Despite the fact that stuck-at fault does not accurately model a large portion of the defects found in the modern CMOS process, it seems to be the most effective fault model in terms of the detection of the defective parts [12]. Therefore although from our simulation results we see that QCA defects do not behave like stuck-at faults, it is still interesting to evaluate the effectiveness of different stuck-at test sets for the simulated defects on a single majority voter. The main results of this evaluation are as follow:

- In all simulations, *super exhaustive* input patterns (i.e. all possible input transitions) are used. The results show that

there is no sequence dependent behavior at logic level; i.e. none of the manufacturing misalignments introduces a state dependency at logic level.

- Except for a single case (i.e. the displacement of all inputs and output cells) faults are detected using a subset of some 100% stuck-at fault test sets. Note that not all of these 100% stuck-at test sets are equal.
- A particular 100% 2-detect stuck-at test set (each fault is detected by two vectors) can detect all manufacturing defects, except for one case, i.e. the simultaneous displacement of the top and left inputs.
- Moreover, a particular 100% single stuck-at test set (001,010,011,101) can detect all simulated defects.

The results for the full-adder circuit shows that none of the defects behave as stuck-at faults at logic-level. However, cell omission defects in wires implemented as inverter chains mainly result in *unwanted complementation* faults in which at extra inverter is present in the faulty wire. Cell omission defects at corner cells in the binary wires also behave this way.

We also considered stuck-at test sets for the full-adder (Fig.9) and computed the corresponding defect coverage with respect to cell omission defects. Note that for a full-adder, any two vectors $\{(a,b,c),(a',b',c')\}$ will result in 100% stuck-at coverage for pin faults, e.g. $\{(010),(101)\}$. However, this test set can detect only 17 out of 28 cell omission defects (Note that 12 of 40 simulated cell omission defects do not affect its functionality). By considering all internal nodes (n1 to n17 in Fig.9), $\{000, 001, 011, 100, 101\}$ is a 100% single stuck-at test set. This test set can detect all 28 detectable defects. This shows that the specific QCA implementation must be considered for test generation to achieve a high defect coverage.

6. Conclusion

Quantum cellular automata (QCA) are novel devices which are promising in the era of nano scale computing. In this paper, testing of QCA based designs has been investigated. A detailed defect characterization for QCA logic devices and some representative circuits has been presented. As shown in this paper, the coupling mechanisms and behavior of defects at logic-level (i.e. the faults) are not similar to those in a conventional CMOS. For example, an *unwanted complementation* fault at logic-level has been observed for a considerable number of cases of *cell omission* defects. Hence, appropriate fault models for QCA must be developed and used for test generation.

Some interesting and unique properties of QCA implementation of logic networks have been investigated. As shown in this paper, a network of majority voters (and inverters) has unique testing properties: Any (V, V', V) test set achieves 100% toggle fault coverage. And V detects n stuck-at-u if and only if V' detects n stuck-at-u'. The effectiveness of different stuck-at test sets in detecting QCA defects has been studied. Our results show that to achieve high defect coverage, the specific QCA implementations of each function must be considered for test generation.

References

[1]I. Amlani, A.O. Orlov, G.L. Snider, and C.S. Lent, "Demonstration of a Six-dot Quantum Cellular Automata System", *Applied Physics Letters*, vol 72, pp. 2179-2181, 1998.

[2]I. Amlani, A.O. Orlov, G. Toth, C.S. Lent, G.H. Bernstein, and G.L. Snider ,"Digital Logic Gate Using Quantum-Dot Cellular Automata", *Science* 284, pp. 289-291, 1999.

[3]C.D.Armstrong, W.M.Humphreys, A.Fijany, "The Design of Fault Tolerant Quantum Dot Cellular Automata Based Logic ", *11th NASA Symposium on VLSI Design*, 2003

[4]C.D.Armstrong, W.M.Humphreys, "The Development of Design Tools for Fault Tolerant Quantum Dot Cellular Automata Based Logic", *2nd International Workshop on Quantum Dots for Quantum Computing and Classical Size Effect Circuits*, 2003.

[5]R. Compano, L. Molenkamp, D.J. Paul, "Technology Roadmap for Nanoelectroincs", *European Commission IST programme, Future and Emerging Technologies*.

[6]A. Fijany and B. N. Toomarian, "New design for quantum dots cellular automata to obtain fault tolerant logic gates", *Journal of Nanoparticle Research*, vol. 3, pp. 27-37, 2001.

[7]S.E. Frost, A.F. Rodrigues, A.W. Janiszewski,R.T. Rausch, P.M. Kogge, "Memory in Motion: A Study of Storage Structures in QCA", *1st Workshop on Non-Silicon Computation*, 2002.

[8]M. Governale, M. Macucci, G. Iannaccone, C. Ungarelli, "Modeling and Manufacturability Assessment of Bistable Quantum-Dot Cells", *Journal of Applied Physics* vol 85, pp. 2962, 1999.

[9]K. Hennessy, C.S. Lent, "Clocking of Molecular Quantum-Dot Cellular Automata", *Journal of Vaccum Science and Technology*, vol 19(5), pp. 1752-1755, 2001.

[10]Personal communication with Professor Marya Lieberman, Department of Chemistry and Biochemistry, University of Notre Dame, IN, USA.

[11]M. Lieberman, S. Chellamma, B. Varughese, Y. Wang, C.S. Lent, G.H. Bernstein, G. Snider, F. Peiris, "Quantum-Dot Cellular Automata at a Molecular Scale", *Annals of the New York Academy of Sciences*, vol 960, pp. 225-239, 2002.

[12]E.J. McClusky and C.W. Tseng, "Stuck-Fault Tests vs. Actual Defects", *Proceedings of International Test Conference*, pp. 336-343, 2000.

[13]M.T. Niemier, A.F. Rodrigues, P.M. Kogge, "A Potentially Implementable FPGA for Quantum Dot Cellular Automata", *1st Workshop on Non-Silicon Computation*, 2002.

[14]M.T. Niemier, P.M. Kogge, "Logic-in-Wire: Using Quantum Dots to Implement a Microprocessor", *International Conference on Electronics, Circuits, and Systems (ICECS '99)*,1999.

[15]A.O. Orlov, I. Amlani, G.H. Bernstein, C.S. Lent, G.L. Snider, "Realization of a Functional Cell for Quantum-Dot Cellular Automata", *Science*, vol 277, pp 928-930, 1997.

[16]QCADesigner Home Page: www.atips.ca/projects/qcadesigner/

[17]C.G. Smith, "Computation Without Current", *Science*, vol 284, pp. 274, 1999.

[18]P.D. Tougaw and C.S. Lent, "Logical Devices Implemented Using Quantum Cellular Automata", *Journal of Applied Physics*, vol 75(3), pp. 1818-1825, 1994.

[19]P.D. Tougaw and C.S. Lent, "Dynamic Behavior of Quantum Cellular Automata", *Journal of Applied Physics*, vol 80(8), pp. 4722-4736, 1996.

[20]V.S. Dimitrov, G.A. Jullien, K. Walus, "Quantum-Dot Cellular Automata Carry-Look-Ahead Adder and Barrel Shifter", *IEEE Emerging Telecommunications Technologies Conference*, 2002.

[21]K. Walus, R.A. Budiman, and G.A. Jullien, "Effects of morphological variations of self-assembled nanostructures on quantum-dot cellular automata (QCA) circuits", *Frontiers of Integration, An International Workshop on Integrating Nanotechnologies*, 2002.

[22]K. Walus, A. Vetteth, G.A. Jullien, V.S. Dimitrov, "RAM Design Using Quantum-Dot Cellular Automata", *NanoTechnology Conference*,vol 2, pp. 160-163, 2003.