Quantum-Dot Cellular Automata Based On Trainable Associative Memory Neural Network for Implementing Reconfigurable Logic Gates

M Saravanan
Department of Electronics and Communication Engineering, Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology, Chennai, Tamilnadu, India
Email: 1msarawins@ieee.org

Abstract. CMOS technology has reached its physical scaling limits, power consumption limits, and hence CMOS technology has to be replaced by other emerging technologies. Quantum-Dot Cellular Automata (QCA) has proven to be a better solution since it offers better scaling and low power consumption for processing digital signals. However, due to a lack of optimization support improvements, dynamical QCA paradigms require multi-layer designs and increased computations. Hence in this paper, a reconfigurable logic gate based on QCA paradigms is proposed. The design utilizes training of memory cells associated with QCA paradigms and hence can be switched between AND, OR, and MUX logic circuit designs dynamically. Compared to traditional dynamical QCA, the model is designed on a single-layer substrate with a minimum number of cells and size.

Keywords: Quantum-dot Cellular Automata; reconfigurable logic gates; artificial neural networks; CNN optimization; CMOS Technology.

1. Introduction
In the last three decades, silicon technology has attained tremendous growth in all electronic applications, from small-scale integration to Ultra large-scale integration. It is recently slowly reaching its limits on physical scaling limits and device density, and hence there is a need for alternate device technologies. Many reaches are carried on QCA, single-electron transistor, and carbon nanotube (CNT) to replace classical CMOS technology. The QCA model was initially proposed by Craig Lent [1], and logical devices are realized based on QCA paradigms [2] due to its better performances in terms of device scaling, clock frequency, power handling [3]. To date, several models achieving different performances in the logical designing based on QCA for ALU, divider, decoder, and memory circuits are implemented [4]. Despite lots of advantages on QCA models compared to classical CMOS technologies, QCA models are less supportive of the optimization implementation of Boolean logic functions [5]. This puts the need for reconfigurable QCA models to replace multiple QCA models with a single QCA model in which operation states are controlled dynamically [6]. The control signals are determined and trained based on a correlation matrix memory (CMM) neural network. QCA based neural network for the training memory cell of a QCA model is presented [7].

In this paper, dynamic switching of logic functions AND gate, OR gate, and Multiplexer is implemented based on QCA paradigms. The switching involves the training of memory cells associated with QCA paradigms in the first few clock cycles and operates at desired logic functions. The proposed
model operates at a single clock cycle and is designed on the single-layer substrate compared to other conventional programmable QCA models, which reduces the system latency and serves for better integration with other Nano-electronics.

2. Preliminaries

2.1. Quantum-Dot Cellular Automata (QCA)

QCA is a transistor-free technology in which the position of electron pairs in the cell determines the cell's polarization states. In general, the electron pair is positioned away from each other due to mutual Coulombic repulsion [8]. Upon proper excitation, the electron pair jumps into its dots as a process of tunneling effect. The polarization state of the QCA cell is depicted in Figure 1, and its corresponding polarization state is derived from equation (1), where \( p_i \) is the charge density in quantum dots.

\[
P = \frac{(p_1 + p_3) - (p_2 + p_4)}{p_1 + p_2 + p_3 + p_4}
\]

Figure 1: The polarization state of QCA Cell

Figure 2 depicts QCA implementation of basic logic structures. Figure 2(a) shows the wire's realization in which the propagation of information takes place. Note that there is no electrical power flow through the line, resulting in low power consumption in QCA designs [9]. Figure 2 (b) shows inverter logic while Figure 2(c) and 2(d) shows AND and OR logic functions, respectively. Figure 2(e) depicts memory cell implementation using the QCA cell array. When the cell is biased with logic 1 (logic 0), the corresponding bit logic 1 (logic 0) will get propagated inside the QCA array loop, thereby storing the one-bit information.

2.2. Correlation matrix memory (CMM) neural network

In the associated memory cell, the response is obtained by proper excitation of stimulus signal. In a biological system, this response is obtained due to stimuli generated from two sets of the neuron [11]. The first set of a neuron's response is given as stimuli for the second set of the neuron, and
thus it propagates. This can be modeled through correlation matrix memory (CMM) shown in Figure 3. It involves two layers of a neural network to represent two sets of biological neurons. The binary vectors are given as stimuli to the neural network, and the two layers are used for training the connection between the layers.

\[
\begin{align*}
(X)(M)(O) \\
(ab)(acadbcd)(cd)
\end{align*}
\]

![Figure 3: CMM Architecture (b) CMM associated neural network](image)

The matrix M is filled with zeroes initially (before training of neural network). The network has K sets of input-output pairs, and hence after training, the matrix M is updated with corresponding association sets. Thus, the output of the CMM neural network is given as

\[
O = MX
\]

Were
\[X\] is the input pattern,
\[O\] is the output pattern obtained as a result of training associated matrix [\(M\)].

3. Implementation
CMM network involves capturing binary input data and training the data based on an associated matrix. In the QCA model, the capturing of binary input can be realized by the mean of the logic gate [12], either AND or OR gate, as shown in Figure 1.

![Figure 4: Logical diagram for CMM architecture.](image)

The training of data is realized using the memory cell, as shown in Figure 1(b). The standard QCA cell design parameters are given in Table 1, shown below.
Table 1: QCAD Parameters.

| Parameter            | Description                        | Standard Value |
|----------------------|------------------------------------|----------------|
| Quantum Dot size     | Size of a quantum dot              | 5 nm           |
| Quantum Cell area    | Dimensions of each cell            | 18 nm x 18 nm  |
| Quantum Cell distance| Distance between two cells         | 20 nm          |
| Layer distance       | Distance between QCA layers in case of multi-layer crossing | 11.5 nm        |
| εr                   | Relative permittivity of material for QCA system | 12.9           |
| Temp                 | Operating temperature              | 1K             |
|                      | Maximum distance                  |                |

The 2-bit CMM neural network's logical design is shown in Figure 4, and its corresponding QCA paradigm is shown in Figure 5 given below.

![Figure 5: QCA Implementation of 2-Bit CMM Neural network.](image)

Table 2: Weights assigned for different operation functions

| S0 | S1 | C   |
|----|----|-----|
| -1 | 0  | AND Gate (A&B) |
| 1  | 0  | OR Gate (A|B) |
| 1  | -1 | A    |
| -1 | 1  | B    | MUX |

After setting the memory cell M based on the weights (S0 and S1), the associated value is trained and is recalled at output C.

4. Results and Discussions

The different training state corresponding to different weights is discussed below. Figure 6 shows the QCA implementation of state one, and its corresponding waveform output generated at the output layer C.
A and B are the inputs given to the QCA cell shown in Figure 6. S0 and S1 are the weights assigned to the memory cell associated with inputs. Upon no biasing (S0 and S1 are set to zero) in the weights cell, the corresponding output cell C gives a random value buffered from A or B inputs. The corresponding function is binary value trained inside the memory cell [13]. When the weights are tuned to S0=-1 and S1=0, the memory cell is updated, and trains for logical AND operation on inputs A and B, and the resultant value is given at output cell C as shown in Figure 6. The training of the memory cell is updated each time during the negative edge of the clock time. Figure 7 shows the QCA implementation of state two and its corresponding waveform output generated at the output layer C. A and B are the inputs given to the QCA cell shown in Figure 7, and S0 and S1 are the weights assigned to the memory cell associated with inputs. Similar to Figure 6, upon no biasing (S0 and S1 are set to zero) in the weights cell, the corresponding output cell C gives a random value buffered from inputs either A or B[14]. The corresponding function is binary value trained inside the memory cell. When the weights are tuned to S0=1 and S1=0, the memory cell is updated and trains for logical OR operation on inputs A and B, and the resultant value is given at output cell C, as shown in Figure 7. The training of the memory cell is updated each time during the negative edge of the clock time.
Figure 7: State 2 (a) QCA implementation (b) Output waveform.

Figure 8 shows the QCA implementation of state three and its corresponding waveform output generated at the output layer C. Similar to Figure 7, upon no biasing (S0 and S1 are set to zero) in the weights cell, the corresponding output cell C gives a random value buffered from inputs either A or B and the corresponding function is binary value is trained inside the memory cell.
The memory cell trains for OR operation corresponding to previous iterations and is shown in Figure 8 at initial clock cycles for four cycles. When the weights are tuned to $S_0=1$ and $S_1=-1$, the memory cell is updated and trains for logical MUX operation on inputs $A$ and $B$, selecting second input $B$ and resultant value reflected after fourth clock cycle at output cell $C$ as shown in Figure 8. The training of the memory cell is updated each time during the negative edge of the clock time.

**Figure 8:** State 3 (a) QCA implementation (b) Output waveform.

**Figure 9:** State 4 (a) QCA implementation (b) Output waveform.
Figure 9 shows the QCA implementation of state four and its corresponding waveform output generated at the output layer C. The memory cell trains for, AND operation corresponds to previous iterations and is shown in Figure 9 at initial clock cycles for four cycles. When the weights are tuned to $S0=1$ and $S1=1$, the memory cell is updated and trains for logical MUX operation on inputs A and B, selecting second input B and resultant value reflected after fourth clock cycle at output cell C as shown in Figure 9. The training of the memory cell is updated each time during the negative edge of the clock time. The above figures discuss the implementation of QCA memory cells with a CMM neural network [15]. This design could serve as an efficient platform for implementing pattern matching abilities using CMM neural network, and also, it is more efficient compared to conventional CMOS technology as it has the potential to store $415\text{nm}^2$ per bit. The model comprises 35 QCA cells each of $18\times18\text{nm}^2$ area space, and thereby the total QCA architecture occupies a total area of $29584\text{nm}^2$. The area utilization factor (AUF) is given by

$$AUF = \frac{\text{Area required for the total number of cells}}{\text{Actual area occupied in QCA implementation}}$$

One of the proposed model’s principal advantages is that the proposed work is designed using a single clock frequency that avoids clock delay. This increases the speed of training of CMM architecture. Table 3 displays performance parameters of Proposed QCA model. A Hartree-Fock approximation is used to calculate the power dissipation by a QCA cell, as given below.

$$H = \left[-\frac{E_k}{2} \sum_{i,j} C_{if},j - \gamma - \frac{E_k}{2} \sum_{i} C_{if},j\right]$$

$$= \left[-\frac{E_k}{2} (C_{j-1} + C_{j+1}) - \gamma - \frac{E_k}{2} (C_{j-1} + C_{j+1})\right]$$

The kink energy between two neighboring cells i and j is given by

$$E_{i,j} = \frac{1}{4\pi\varepsilon_0\varepsilon_r} \sum_{n=1}^{4} \frac{q_{n,i} q_{n,j}}{|r_{n,i} - r_{n,j}|}$$

Power dissipation for the QCA cell can be derived from

$$P_{\text{Total}} = \frac{d}{dt} E = \frac{\hbar}{2} \left(\frac{d}{dt} \lambda\right) \Gamma + \frac{\hbar}{2} \left(\frac{d}{dt} \lambda\right) \Gamma$$

Table 3: Performance parameters of Proposed QCA model

| Parameter                  | Specification |
|---------------------------|---------------|
| No. of Cells required     | 35            |
| AUF                       | 0.9218        |
| Dissipation of power at 1K| $\gamma = 0.5E_k$ | 6.2 |
|                           | $\gamma = 1.0E_k$ | 10.56 |
|                           | $\gamma = 1.5E_k$ | 14.82 |

5. Conclusion
The paper discusses the implementation of the Correlation matrix memory (CMM) neural network using QCA paradigms. A 2-bit binary data is taken as stimuli for the CMM network, and the memory cell acts as an intermediate layer and is associated with the input layer. It is used for training based on the input layer. The output layer selects desired logic functions based on weights associated with memory cells. The proposed model has higher memory density and consumes less power when compared with traditional CMOS technology.

References
[1] Lent S, Tougaw D, Porod W, Bernstein H (1993) Quantum cellular automata. Nanotechnology 4(1): 49-57.
[2] Tougaw P, Lent S (1994) Logical devices implemented using quantum cellular automata. Journal of Applied Physics 75: 1818-1825.
[3] Snider GL, Orlov AO, Amlani I, Zuo X, Bernstein GH, Lent CS, Porod W (1999) Quantum-dot cellular automata: Review and recent experiments. Journal of Applied Physics 85(8): 4283–4285.

[4] Mo Liu, Lent CS (2003) High-speed metallic quantum-dot cellular automata. Third IEEE Conference on Nanotechnology (IEEE-NANO 2003) 465-468

[5] Kim SW, Swartzlander EE (2010) Multipliers with coplanar crossings for Quantum-Dot Cellular Automata. 10th IEEE International Conference on Nanotechnology 953–957.

[6] Balali M, Rezai A, Balali H, Rabiei F, Emadi S (2017) Towards coplanar quantum-dot cellular automata adders based on efficient three-input XOR gate. Results in Physics 7: 1389–1395.

[7] Sasamal TN, Singh AK, Ghanekar U (2016) Design of non-restoring binary array divider in majority logic-based QCA. Electronics Letters 52(24): 2001–2003.

[8] Chougule PP, Sen B, Dongale TD (2017) Realization of processing In-memory computing architecture using Quantum Dot Cellular Automata. Microprocessors and Microsystems 52: 49–58.

[9] Kalogeiton VS, Papadopoulos DP, Liolis O, Mardiris VA, Sirakoulis GC, Karafyllidis IG (2017) Programmable Crossbar Quantum-Dot Cellular Automata Circuits. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 36(8): 1367–1380.

[10] Kohonen T (1972) Correlation Matrix Memories. IEEE Transactions on Computers, C-21(4): 353–359. doi:10.1109/etc.1972.5008975

[11] James Stovold (2019) Trainable Associative Memory Neural Networks in Quantum-Dot Cellular Automata. International Conference on Unconventional Computation and Natural Computation (UCNC 2019).

[12] Lange W (1975) Cell number and cell density in man and some other mammals in the cerebellar cortex. Cell and Tissue Research 157(1): 115–124. doi:10.1007/bf00223234

[13] Cunha, F., Racicot, K., Nahirney, J., Heuston, C., Wylie, D.R., and Iwaniuk, A.N., 2020. Allometric Scaling Rules of the Cerebellum in Galliform Birds. Brain, Behavior and Evolution, 95(2), pp.78-92.

[14] Attaai, A.H., Noreldin, A.E., Abdel-Maksoud, F.M. and Hussein, M.T., 2020. An updated investigation on the dromedary camel cerebellum (Camelus dromedarius) with special insight into the distribution of calcium-binding proteins. Scientific reports, 10(1), pp.1-17.

[15] Zimatkin, S.M., Yemelyanchik, S.V. and Karnyushko, O.A., 2020. Calbindin Immunoreactivity in Rat Cerebral Cortex and Cerebellum Neurons. Neuroscience and Behavioral Physiology, pp.1-4.