A High-Performance Design of Generalized Pipeline Cellular Array

Journal:	*IEEE Computer Architecture Letters*
Manuscript ID	CAL-2020-02-0017
Manuscript Type:	Letters
Keyword:	B.2 Arithmetic and Logic Structures < B Hardware, B.2.4 High-Speed Arithmetic < B.2 Arithmetic and Logic Structures < B Hardware, B.6.1.a Cellular arrays and automata < B.6.1 Design Styles < B.6 Logic Design < B Hardware

A High-Performance Design of Generalized Pipeline Cellular Array

Zhufei Chu, *Member, IEEE,* Huiming Tian, Zeqiang Li, Yinshui Xia, and Lunyao Wang

Abstract—In this letter, we proposed a high-performance quantum-dot cellular automata (QCA) design of generalized pipeline cellular array (GPCA). The GPCA can perform all the basic arithmetic operations using only one arithmetic cell. Due to its flexibility, the high-performance GPCA design is of high interest for large-scale QCA designs. We proposed both the arithmetic unit and control unit designs of GPCA using a hybrid logic network that represented by majority-of-three (MAJ), inverter (NOT), and three-input exclusive-OR (XOR_3). The introduction of XOR_3 operation can significantly reduce the number of gates, levels, and wire-crossings, which in turn results in less area and fewer clock cycles of QCA layout. On average, the proposed design can reduce the area and latency by 39.27% and 35.89% compared with the-state-of-art, respectively.

Index Terms—majority, three-input XOR, pipeline cellular array, quantum-dot cellular automata (QCA)

1 Introduction

High-speed cellular array designs for arithmetic operations have attracted considerable interest for computer architecture. The cellular array usually consists of identical arithmetic cells or units connected in an iterative array pattern. With the transistor feature size continuing to scale down, the cellular array is becoming increasingly important. Due to their versatility and flexibility, the arrays are suitable for large-scale integration as well as pipelining and parallel processing [2], [9]. Generalized pipeline cellular array (GPCA) is one of such array [7], which can perform all the basic arithmetic operations in a pipeline manner.

Logic-gate implementation of the arithmetic cell has a significant impact on the performance of the cellular array. Since complementary metal-oxide-semiconductor (CMOS) technology is gradually reaching its physical limitation in recent years, new nanotechnology circuit (e.g., quantum-dot cellular automata (QCA) [8]) and new device (e.g., two-dimensional polarity-controllable transistors [11]) are emerging. These new computing paradigm demonstrated promising advantages of power consumption and device density. However, different from the conventional Boolean logic used in CMOS, the primitives of these technologies are based on majority-of-three (MAJ) logic.

The MAJ gate over three Boolean variables a, b, and c can be represented as $\mathbb{M}(a,b,c) = ab + ac + bc$. For the three-input XOR gate (XOR_3), it can be also represented as the MAJ forms.

$$\mathbb{X}(a,b,c) = a \oplus b \oplus c = \mathbb{M}(\bar{a}, \mathbb{M}(a,b,\bar{c}), \mathbb{M}(a,\bar{b},c)) \quad (1)$$

By setting any one of the three inputs to constant, we can obtain AND/OR/XOR operations over the remaining non-constant inputs. As an example, $ab = \mathbb{M}(a,b,0)$ and $a \oplus b = \mathbb{X}(a,b,0)$.

The Boolean expressions and computing architecture of n-bit GPCA were displayed in the 1970s [7]. In terms of new hardware implementation, a QCA design based on the MAJ logic network for GPCA is proposed [3]. Although the implementations of various arithmetic circuits in QCA have been investigated in the literature, the universal arithmetic unit, e.g., GPCA, instead of specific arithmetic circuit designs, could be beneficial in future large-scale QCA designs.

(Corresponding author: Zhufei Chu). *Z. Chu, H. Tian, Z. Li, Y. Xia, and L. Wang are with the Faculty of Electrical Engineering and Computer Science, Ningbo University, Ningbo 315211, China. (email:chuzhufei@nbu.edu.cn)*

The basic GPCA cell consists of an adder-subtractor arithmetic unit (**AU**) and a control logic unit (**CU**). Since GPCA can perform the operations of any number of bits based on these basic GPCA cell, the high-performance AU and CU designs lead to cumulative improvement for n-bit GPCA design.

Unlike the work presented in [3], which focuses on realizing the GPCA cell only by MAJ and inverter (NOT) primitives, in this letter, we redesigned the AU and CU using the combinations of MAJ, NOT, and XOR_3 operations for high-performance design. For fair comparison and validation, we also implement the proposed designs using QCA technology and simulate using QCADesigner [16]. The compact logic representations positively affect the QCA layouts, which has a reduction in area and critical path latency. The experimental results show that our design can averagely reduce the area and delay by 39.27% and 35.89%, respectively. Moreover, since the single-layer QCA design strategy is more practical for the QCA physical fabrication process, we established a single-layer QCA design of AU, thanks to our logic optimization, which results in fewer wire-crossings.

2 Related Work

The Boolean expressions of GPCA given by [7] are based on two-input XOR/AND/OR and single-input NOT operations. The digital design of GPCA using hardware design language (HDL) is studied in [14]. To exploit the QCA implementation of GPCA, the authors in [3] proposed a functional equivalent MAJ/NOT based logic network for GPCA. This is because MAJ/NOT gates are the basic building blocks of QCA. However, for the QCA XOR_3 gate, instead of straightforward realization by the MAJ forms as in (1), several independent QCA XOR_3 gates with less area and delay are proposed in the literature recently [5], [12]. Therefore, it is feasible to form a new QCA technology library that consists of MAJ, NOT, and XOR_3. The library brings new opportunities for a more efficient QCA design of GPCA.

To unleash the expressive power of the new QCA technology library, an efficient logic representation over the gate basis {MAJ, NOT, and XOR_3} is crucial for the high-performance QCA design. Recently, a graph-based logic representation XOR-majority graph (XMG) is proposed [6], which is extended from a majority-inverter graph (MIG) [4]. These advanced representations provide the expertise of MAJ based logic synthesis into the QCA designs.

3 Background

This section reviews the QCA background and the Boolean expressions of GPCA.

QCA: A QCA cell is a square structure which contains four quantum-dots at corners and two electrons. These electrons are always placed at the diagonal positions due to mutual electrostatic repulsion. Thus two distinguishable cell states exist to represent binary logic values. A number of QCA cells can be composed to build QCA logic gates and QCA wires. The QCA circuit is operated by applying a four-phases QCA clock.

Several aspects need to be considered for a high-performance QCA design. First, efficient logic representation in terms of a fewer number of logic operations and levels leads to a compact QCA layout. Second, from the perspective of QCA interconnection, wire-crossings are playing a vital role in fabrication. Wire-crossing may lead to many difficulties, including crosstalk and additional power dissipation [10]. There are two main kinds of QCA wire-crossing, as QCA can be implemented using both single-layer and multi-layer strategies. The single-layer approach deal with wire-crossing by two different quantum dot orientations, i.e., one at 45 degrees to the other. Another approach is based on clock-zone, which using phase difference to represent crossing wires [13]. In contrast, a multi-layer method is more flexible as the QCA wire can route in a three-dimensional way. However, single-layer QCA is more practical for fabrication. To show our design can be efficiently implemented using different design types, we established both single- and multi-layer AU designs in this paper.

Boolean Expressions of the GPCA: The structures of AU and CU are shown in Fig.1 (a) and (b), respectively. The AU has six inputs, i.e., A, B, C, F_i, X, and C_1, and four outputs, i.e., S, D, E and C_0. The CU has three inputs, i.e., C_0, P_i, and X, and one output F_i. The Boolean expressions of AU used in the work [7] are given as in (2).

$$\left.\begin{aligned} S &= (A \oplus (B \oplus X) \oplus C_1)F_i + A\overline{F_i} \\ C_0 &= (B \oplus X)(A + C_1) + AC_1 \\ D &= BC + CF_i = C(B + F_i) \\ E &= B + CF_i = (B + C)(B + F_i) \end{aligned}\right\} \quad (2)$$

For the CU part, the Boolean expression is defined by

$$F_i = C_0X + P_i\bar{X}. \quad (3)$$

Since QCA can natively implement MAJ and NOT operations as the fundamental logic devices, the authors in [3] proposed a functional equivalent MAJ/NOT based logic network. The Boolean expressions of AU are shown in (4), where n_0, n_1, n_2, and n_3 are internal nodes.

$$\left.\begin{aligned} S &= \mathbb{M}(\mathbb{M}(n_0, F_i, 0), \mathbb{M}(A, \bar{F}_i, 0), 1) \\ C_0 &= n_1 \\ D &= \mathbb{M}(n_3, C, 0) \\ E &= \mathbb{M}(B, C, n_3) \\ n_0 &= \mathbb{M}(\bar{n}_1, \mathbb{M}(n_2, \bar{A}, C_1), A) \\ n_1 &= \mathbb{M}(n_2, A, C_1) \\ n_2 &= \mathbb{M}(\mathbb{M}(\bar{X}, B, 0), \mathbb{M}(X, \bar{B}, 0), 1) \\ n_3 &= \mathbb{M}(F_i, B, 1) \end{aligned}\right\} \quad (4)$$

The MAJ/NOT based expression for the CU is defined by

$$F_i = \mathbb{M}(\mathbb{M}(X, C_0, 0), \mathbb{M}(\bar{X}, P_i, 0), 1). \quad (5)$$

There are twelve MAJ and five NOT gates with a logic level of eight to realize AU, while the CU needs three MAJ and one NOT gates. It can be found that only four MAJ gates are fully utilized (i.e., with no constant inputs), the remaining eleven MAJ gates are partially utilized (i.e., with one constant input). Further improvement can be achieved if we make better use of MAJ gates [1] or introducing new XOR_3 primitive.

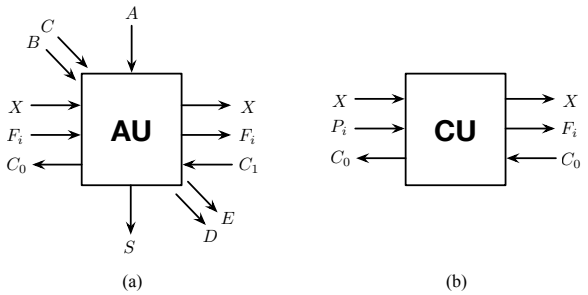

Fig. 1. Structure of the basic GPCA cells: (a) arithmetic unit, (b) control unit (adapted from the figures shown in [3], [7]).

4 Proposed Designs

We will explain the proposed Boolean expressions for all the output signals of AU first, then the design of CU will be presented.

In terms of the outputs S and C_0, from (2) we can see the subexpression $(B \oplus X)$ is appeared in both expressions. Moreover, the internal node n_2 in (4) actually preforms the same operation using MAJ/NOT primitives, i.e., $n_2 = B \oplus X$. The recent exact synthesis approach [15] indicates the two- or three-input XOR functions can be realized by at least three MAJ gates. Hence, the expression of n_2 in (4) is already optimal in size and depth. Since the XOR_3 gate was proposed in the literature, we make use of MAJ/NOT/XOR_3 based expressions for the design. For the output C_0, we replace n_2 as $\mathbb{X}(B, X, 0)$, then

$$C_0 = \mathbb{M}(\mathbb{X}(B, X, 0), A, C_1) \quad (6)$$

For the output S, since $(a \oplus b)c + a\bar{c} = a \oplus bc$ holds, we can assign $a = A$, $b = (B \oplus X) \oplus C_1$, and $c = F_i$. Then the expression of S in (2) is rewriten as

$$\begin{aligned} S &= (((B \oplus X) \oplus C_1)F_i) \oplus A \\ &= \mathbb{X}(\mathbb{M}(\mathbb{X}(\mathbb{X}(B, X, 0), C_1, 0), F_i, 0), A, 0) \quad (7a) \\ &= \mathbb{X}(\mathbb{M}(\mathbb{X}(B, X, C_1), F_i, 0), A, 0). \quad (7b) \end{aligned}$$

Equation (7b) requires only three MAJ/XOR_3 operations, in contrast, equation (7a) needs one additional XOR_3 operation but it can share the node $\mathbb{X}(B, X, 0)$ appeared in (6). We reserve both the designs for further evaluations in QCA technology.

The outputs of D and E is relatively simple. The logic network proposed in [3] is shown in Fig. 2(a), in which signals may face the wire-crossing problem. The proposed logic network for outputs D and E is shown in Fig. 2(b), in which the wire-crossing problem is solved. For the output D, the expression is

$$D = \mathbb{M}(\mathbb{M}(F_i, B, C), C, 0). \quad (8)$$

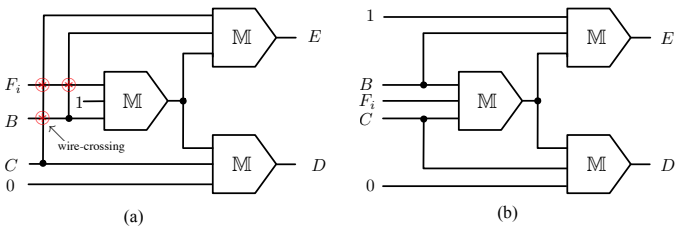

Fig. 2. Two logic networks for the outputs D and E, (a) The expressions proposed by GPCA[3] may face wire-crossing problem, (b) Our design.

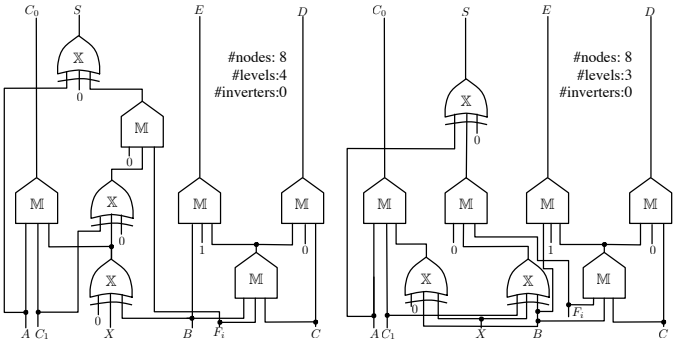

Fig. 3. The logic networks of the proposed design of AU, we make use of (a) (7a), (b) (7b) as the expression for the output S.

By expansion and simplification, one can verify the function equivalence of (8) and the D expression in (4).

$$\begin{aligned} D &= (F_iB + F_iC + BC)C = F_iBC + F_iC + BC \\ &= F_iC + BC = (F_i + B)C \\ &= \mathbb{M}(\mathbb{M}(F_i, B, 1), C, 0) \end{aligned} \tag{9}$$

Similarly, the output E is represented as

$$E = \mathbb{M}(\mathbb{M}(F_i, B, C), B, 1) \tag{10}$$

such that the subexpression $\mathbb{M}(F_i, B, C)$ is shared with (8).

The expressions of the AU are the collections of equations (6), (7), (8), and (10). Since (7) has two versions of expression, i.e., (7a) and (7b), we present the logic network of both of the two versions in Fig. 3. The GPCA cell designs adopt the AU design shown in Fig.3 (a) and (b) are named "*Design A*" and "*Design B*", respectively. Both of the AU designs require eight gates, including five MAJ gates and three XOR_3 gates. The number of logic levels of Fig. 3(b) is three while the one in Fig. 3(a) is four. Besides, no inverters are required for both of the designs. The main difference is the usage of the XOR_3 operation. Fig. 3(b) contains one fully-utilized XOR_3 node by rewriting $(B \oplus X) \oplus C_1$ into $\mathbb{X}(B, X, C_1)$. Since the node $\mathbb{X}(B, X, 0)$ has two fanouts in Fig. 3(a), we still need this node in Fig. 3(b) for functional equivalence. In terms of CU part, the design proposed in [3] needs three MAJ gates and one inverter. Due to $ab + \bar{b}c = \mathbb{M}(a, c, \overline{a \oplus b}) = \mathbb{M}(a, c, \mathbb{X}(a, b, 1))$, we can assign $a = C_0$, $b = X$, and $c = P_i$. Then the expression of F_i can be rewritten as

$$F_i = \mathbb{M}(\mathbb{X}(X, C_0, 1), P_i, C_0). \tag{11}$$

Note that a constant '1' input in XOR_3 gate leads to the inversion of the XOR operation over the other two inputs. Hence, the inverter is also eliminated thanks to the usage of XOR_3 operation.

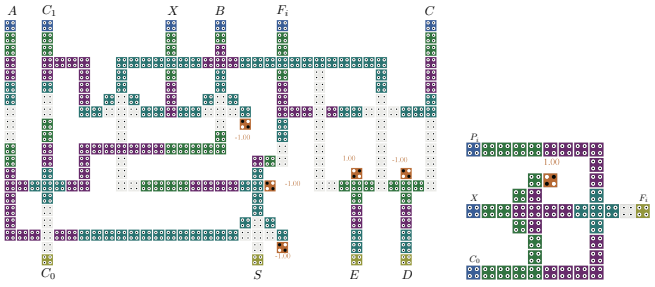

Fig. 4. The single-layer QCA layouts of the proposed (a) AU and (b) CU.

TABLE 1
Comparison Results of the Logic-gate Implementations

Cells	Designs in [3]			*Design A*			*Design B*		
	G	L	C	G	L	C	G	L	C
AU	17	8	27	8	4	5	8	3	5
CU	4	3	1	2	2	0	2	2	0
Total	21	11	28	10	6	5	10	5	5

G/L/C: the number of logic gates/levels, and wire-crossings.
Primitives used in [3] is MAJ/NOT, while MAJ/NOT/XOR_3 in the proposed design.

The comparison results of different logic-gate implementations are displayed in Table 1. By introducing the XOR_3 gate, our designs have a fewer number of logic gates/levels, and wire-crossings than previous research, which, in turn, could result in less area/delay, and design complexity of QCA layouts. *Design B* has an advantage on the logic level compared to *Design A* while the number of logic gates is the same.

5 Experimental Results and Comparison

In this section, the simulation results of our designs were presented. All the designs and simulations are implemented by QCADesigner 2.0.3. We have set the same parameters with the ones presented in [3]. The XOR_3 gate proposed in [5] is adopted in our designs. And the clock-zone based wire-crossing technique [13] is used for single-layer QCA design. The latency of the design is represented by the number of required operation clock cycles.

AU Designs: We realized the proposed AU design in both single- and multi-layer design methods. The single-layer one is more practical for QCA fabrication. But for fair comparison and evaluation with the state-of-the-art, the multi-layer approach is also established. The comparison results of AU designs are shown in Table 2, which indicate our multi-layer realization of *Design B* has a better performance in both the area and delay. Therefore, we adopt *Design B* as the building block for the subsequent n-bit GPCA designs. The single-layer QCA layouts of our proposed AU and CU are shown in Fig. 4(a) and (b), respectively, thanks to our reduced number of wire-crossings. Note that CU design has no wire-crossing, thus the layout of CU is naturally in single-layer.

Comparisons with the State-of-the-Art: The proposed AU and CU are the building blocks to construct the GPCA for performing arithmetic operations of any number of bits. Comparison results of the proposed designs with the state-of-the-art approach [3] is given in Table 3. It can be seen from the table that both the AU and CU designs achieve better performance in the area, latency, and cells count. The proposed AU design has reduced the area by 69.12% from 0.68 μm^2 to 0.21 μm^2 and the latency is reduced by 75% from 4 clock cycles to 1 clock cycle. Moreover, the cells count is reduced from 682 to 217, which is 68.18% improvement. The better performance can be also seen for the CU design. This advantage also carries over into the n-bit GPCA designs. As an example, the proposed 5-bit pipeline array design has a total number of 26,094 QCA cells and an area of 46.80 μm^2 with a latency of 56 clock cycles, while the one presented in [3] has a total number of 42,174 QCA cells and an area of 59.47 μm^2 with a latency of 71 clock cycles. On

TABLE 2
Comparison Results of the AU Designs

AU	Type	Area (μm^2)	Latency	Cells count
Design A	Multi-layer	0.25	1.5	238
Design B	Multi-layer	0.21	1	217
	Single-layer	0.31	2	245

TABLE 3
Comparison Results of the Proposed GPCA versus the Designs Presented in [3]

Designs	GPCA [3]			Ours					
	Area (μm^2)	Latency	Cells count	Area (μm^2)	%	Latency	%	Cells count	%
AU	0.68	4	682	0.21	69.12	1	75.00	217	68.18
CU	0.11	1.5	77	0.05	54.55	1	33.33	44	42.86
GPCA (n=3)	23.35	34	15098	17.06	26.94	25	26.47	10312	31.70
GPCA (n=4)	39.44	51	26698	29.80	24.44	39	23.53	17258	35.36
GPCA (n=5)	59.47	71	42174	46.80	21.30	56	21.13	26094	38.13
Average improvement					39.27%		35.89%		43.25%

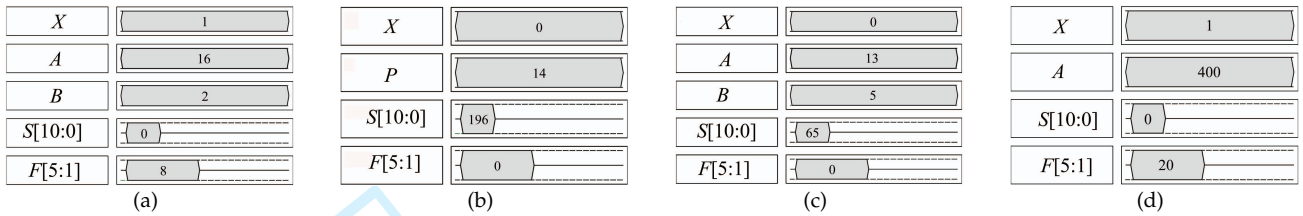

Fig. 5. Simulation results for (a) Division, (b)Squaring, (c) Multiplication, and (d)Square rooting.

average, the proposed design has reduced the area, latency, and cells count by 39.27%, 35.89%, and 43.25%, respectively.

The proposed AU and CU designs reduced the clock cycles by 3 and 0.5 compared with GPCA in [3], respectively. Thus, theoretically speaking, the delay of each level in GPCA can be reduced by 3.5 clock cycles. However, in favor of the clock design of the QCA layout, we added a 0.5 clock cycle per level in our design. As a result, the delay of the first level has 6 clock cycles and the calculation formula of the delay is as given in (12). The reader can refer to [3] for details of delay calculation.

$$Delay = 1 + \sum_{i=1}^{n}(5 + 3(i-1)) \quad (12)$$

Generally, for the n-bit GPCA design, the proposed design can reduce $3n$ clock cycles compared with the GPCA presented in [3]. For example, the latency of the 5-bit GPCA is $Delay_{n=5} = 56$, in which $3n = 3 \times 5 = 15$ clock cycles are saved.

Fig. 5 shows the simulation result of our GPCA. The input 'X' is a control line, which made logical zero or one for different arithmetic computations. For division, we set A's as $(10000)_2$ (binary number, 16 in decimal) and B's as $(10)_2$ we got the result of $(1000)_2$ in F's and the remainder of 0 in S's as show in Fig. 5(a). In Fig. 5(b), we set P's as $(1110)_2$ and we got the result of $(11000100)_2$ in S's for squaring operation. For multiplication in Fig. 5(c), we set A's $(1101)_2$ and B's as $(101)_2$ and we got the result of $(1000001)_2$ in S's. For square rooting in Fig. 5(d), we put A's as $(110010000)_2$ and we got the result of $(10100)_2$ in F's and the remainder of 0 in S's.

6 Conclusions

In this letter, we proposed a QCA design for generalized pipeline cellular array (GPCA). Unlike existing GPCA design using only the majority logic network, the proposed array is constructed by additionally introducing three-input XOR (XOR_3) gates. Thus, the GPCA is represented by a hybrid logic network by using majority, inverter, and XOR_3 primitives. The proposed arithmetic and control unit have a significant reduction of area, latency, and cells count. On average, we can achieve 39.27%, 35.89%, 43.25% reduction of area, delay and cells count compared with the state-of-the-art.

Acknowledgments

This work was supported in part by NSFC under Grant No 61871242 and in part by K.C.Wong Magna Fund in Ningbo University.

References

[1] D. Abedi and G. Jaberipur, "Decimal full adders specially designed for quantum-dot cellular automata," *IEEE Trans. Circuits Syst. II*, vol. 65, no. 1, pp. 106–110, 2018.
[2] D. P. Agrawal, "High-speed arithmetic arrays," *IEEE Trans. Comput.*, no. 3, pp. 215–224, 1979.
[3] A. F. Almatrood and H. Singh, "Design of generalized pipeline cellular array in quantum-dot cellular automata," *IEEE Comput. Archit. Lett.*, vol. 17, no. 1, pp. 29–32, 2017.
[4] L. Amarú, P.-E. Gaillardon, and G. De Micheli, "Majority-inverter graph: A novel data-structure and algorithms for efficient logic optimization," in *Design Automation Conference*, 2014, pp. 1–6.
[5] A. N. Bahar, S. Waheed, N. Hossain, and M. Asaduzzaman, "A novel 3-input XOR function implementation in quantum dot-cellular automata with energy dissipation analysis," *Alexandria Engineering Journal*, vol. 57, no. 2, pp. 729–738, 2018.
[6] W. Haaswijk, M. Soeken, L. Amarú, P.-E. Gaillardon, and G. De Micheli, "A novel basis for logic rewriting," in *Asia and South Pacific Design Automation Conference*, 2017, pp. 151–156.
[7] A. Kamal, H. Singh, and D. Agrawal, "A generalized pipeline array," *IEEE Trans. Comput.*, vol. 100, no. 5, pp. 533–536, 1974.
[8] C. S. Lent, P. D. Tougaw, W. Porod, and G. H. Bernstein, "Quantum cellular automata," *Nanotechnology*, vol. 4, no. 1, p. 49, 1993.
[9] J. C. Majithia, "Some comments concerning design of pipeline arithmetic arrays," *IEEE Trans. Comput.*, no. 11, pp. 1132–1134, 1976.
[10] R. K. Nath, B. Sen, and B. K. Sikdar, "Optimal synthesis of QCA logic circuit eliminating wire-crossings," *IET Circuits, Devices & Systems*, vol. 11, no. 3, pp. 201–208, 2017.
[11] G. V. Resta, Y. Balaji, D. Lin, I. P. Radu, F. Catthoor, P.-E. Gaillardon, and G. De Micheli, "Doping-free complementary logic gates enabled by two-dimensional polarity-controllable transistors," *ACS Nano*, vol. 12, no. 7, pp. 7039–7047, 2018.
[12] H. R. Roshany and A. Rezai, "Novel efficient circuit design for multilayer QCA RCA," *International Journal of Theoretical Physics*, vol. 58, no. 6, pp. 1745–1757, 2019.
[13] S.-H. Shin, J.-C. Jeon, and K.-Y. Yoo, "Wire-crossing technique on quantum-dot cellular automata," in *International conference on next generation computer and information technology*, 2013, pp. 52–57.
[14] H. Singh, D. P. Agrawal, S. Kamthan, and L. Alazzawi, "On simulation and design implementation of generalized pipeline cellular array," in *International Conference on Information Science, Electronics and Electrical Engineering*, vol. 3, 2014, pp. 1761–1765.
[15] M. Soeken, L. G. Amarù, P.-E. Gaillardon, and G. De Micheli, "Exact synthesis of majority-inverter graphs and its applications," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 36, no. 11, pp. 1842–1855, 2017.
[16] K. Walus, T. J. Dysart, G. A. Jullien, and R. A. Budiman, "Qcadesigner: a rapid design and simulation tool for quantum-dot cellular automata," *IEEE Trans. Nanotechnol.*, vol. 3, no. 1, pp. 26–31, 2004.