Design of Efficient BCD Adders in Quantum-Dot Cellular Automata

G. Cocorullo, P. Corsonello, F. Frustaci, and S. Perri

***Abstract*—Among the emerging technologies recently proposed as alternatives to the classic CMOS, quantum-dot cellular automata (QCA) is one of the most promising solutions to design ultralow-power and very high speed digital circuits. Efficient QCA-based implementations have been demonstrated for several binary and decimal arithmetic circuits, but significant improvements are still possible if the logic gates inherently available within the QCA technology are smartly exploited. This brief proposes a new approach to design QCA-based BCD adders. Exploiting innovative logic formulations and purpose-designed QCA modules, computational speed significantly higher than existing counterparts is achieved without sacrificing either the occupied area or the cell count.**

***Index Terms*—BCD adders, decimal arithmetic, quantum-dot cellular automata (QCA).**

I. Introduction

QUANTUM-DOT cellular automata (QCA) technology [1] has become one of the most attractive approaches for the development of next-generation ultradense low-power high-performance digital circuits. Among the logic circuits recently proposed, arithmetic submodules represent examples of the most investigated structures [2]–[9]. Independently of the performed logic function, nonelementary digital modules are designed smartly combining inverters and majority gates (MGs), which are the only fundamental logic gates inherently available within the QCA technology.

Decimal arithmetic has recently received a great deal of attention since several financial, commercial, and Internet-based applications increasingly require higher precision. In these contexts, the errors coming from the conversion between decimal and binary number representations could not be tolerated, and several recent microprocessors [10]–[14] include hardware decimal arithmetic units in their core based on the IEEE 754–2008 standard [15]. The design of such digital circuits requires proper strategies at both logic and layout levels to improve performance and area behaviors [16]–[25].

This brief proposes a novel approach to design QCA-based n-digit (with $n \geq 1$) BCD adders able to achieve computational speed higher than existing counterparts without sacrificing either the occupied area or the cell count. Such advantages are obtained by exploiting an innovative logic strategy together with purpose-designed QCA modules. The latter have been optimized by taking into account that the most time critical decimal addition between two n-digit numbers occurs when a carry is generated by adding the least significant digits of the operands, it is then propagated through the subsequent $n-2$ digit positions, and finally, it is absorbed at the last digit position where the most significant sum digit is computed. The novel 1-digit decimal adder generates, propagates, and absorbs a carry signal with delay times up to 27%, 45%, and 44% lower than the faster existing counterparts that are those described in [23]. Differently from all previous works, we extended our work to the design of a 2-digit QCA-based BCD adder. Its characterization demonstrates that the 2-digit sum computation is performed within only 18 clock phases, and the circuit spans over an area of only 2.74 μm^2.

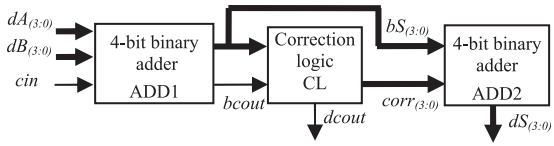

Fig. 1. Structure of the BCD adders presented in [21]–[23].

II. Background and Related Works

The semiconductor QCA nanostructure that we refer in the following uses a square cell with four quantum dots and two free electrons as the basic element to realize both logic structures and interconnections [1]. Owing to coulombic repulsion, the generic cell can assume only two possible stable states, also named polarizations, which are associated with the binary states 1 and 0. Even though adjacent cells interact through electrostatic forces and tend to align their polarizations, QCA cells do not have intrinsic data flow directionality. The latter is provided by means of four clock signals clk_x (with x ranging from 0 to 3), shifted by 90° from each other [1]. The fundamental logic gates available within the QCA technology are the inverter and the MG. Given three inputs a, b, and c, the MG performs the logic function reported in (1) provided that all input cells are associated with the same clock signal clk_x, whereas the remaining cells of the MG are associated with the clock signal clk_{x+1}

$$M(a,b,c) = a \cdot b + a \cdot c + b \cdot c. \tag{1}$$

The 1-digit BCD adders previously proposed in [23]–[25] use the top-level architecture in Fig. 1: They first add the input digits in binary format and then perform the conversion to BCD by adding 6 if the obtained binary result is greater than 9 and 0 otherwise. From the logic point of view, the aforementioned designs mainly differ from each other in the adder structure

Manuscript received January 14, 2016; revised April 7, 2016; accepted June 9, 2016. Date of publication June 14, 2016; date of current version April 28, 2017. This brief was recommended by Associate Editor M. Alioto.

The authors are with the Department of Informatics, Modeling, Electronics and System Engineering, University of Calabria, 87036 Rende, Italy (e-mail: g.cocorullo@unical.it; p.corsonello@unical.it; ffrustaci@deis.unical.it; perri@deis.unical.it).

Color versions of one or more of the figures in this brief are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSII.2016.2580901

used. In fact, ripple-carry adders are utilized in [24] and [25], whereas the two implementations presented in [23] exploit the carry-flow [4] and the carry-look-ahead [3] architectures.

Conversely, the decimal adder demonstrated in [16] serially processes digits encoded by the Johnson–Mobius code.

III. Novel QCA Adder

The BCD adder here presented follows the traditional top-level structure illustrated in Fig. 1, but it exploits the novel logic expressions demonstrated in the following by *Theorems* 1 and 2. As the main result, the proposed approach leads to the best tradeoff between the overall occupied area and the speed performances.

To understand the new design strategy, let us examine first the 4-b binary adder ADD1. It receives the digits $dA_{(3:0)}$ and $dB_{(3:0)}$ and the carry cin as inputs and computes the binary results bcout and $bS_{(3:0)}$. [26, Lemma 4] demonstrates that, for QCA-based rippling adders, the optimal logic structure for propagating a carry C_i through a single bit position is represented by (2a) that introduces only one MG between C_i and C_{i+1}

$$C_{i+1} = M(dA_i, dB_i, C_i) \tag{2a}$$

$$C_{i+2} = M(C_i, M(dA_{i+1}, dB_{i+1}, g_i), M(dA_{i+1}, dB_{i+1}, p_i)) \tag{2b}$$

Consequently, the propagation of C_i through two bit positions would require two cascaded MGs to obtain the carry C_{i+2}. Conversely, as discussed in [6] and given in (2b), by exploiting the auxiliary generate and propagate signals $g_i = dA_i \cdot dB_i$ and $p_i = dA_i \cdot dB_i$, the carry C_{i+2} can be computed by propagating C_i through just one MG. *Theorem 1* demonstrates a novel way to propagate C_i through two consecutive bit positions that also introduces just one MG between C_i and C_{i+2} but avoiding the computation of g_i and p_i.

Theorem 1: Let us consider the consecutive bits dA_{i+1}, dA_i dB_{i+1}, and dB_i of the addends $dA_{(3:0)}$ and $dB_{(3:0)}$. If Ci is the carry signal inputted at the ith bit position, then the carry C_{i+2} produced at the $(i+1)$th bit position is given by

$$C_{i+2} = M(C_i, M(dA_{i+1}, dB_{i+1}, dA_i), M(dA_{i+1}, dB_{i+1}, dB_i)). \tag{3}$$

Proof: By applying the conventional Carry-Look-Ahead (CLA) logic and considering that the propagate and generate signals P_{i+1}, P_i, G_{i+1}, and G_i are defined as $P_{i+1} = dA_i + dB_{i+1}$, $P_i = dA_i + dB_i$, $G_{i+1} = dA_{i+1} \cdot dB_i + 1$, and $G_i = dA_i \cdot dB_i$, we have

$$\begin{aligned}
C_{i+2} &= G_{i+1} + P_{i+1} \cdot G_i + P_{i+1} \cdot P_i \cdot C_i \\
&= G_{i+1} + P_{i+1} \cdot G_i + P_{i+1} \cdot P_i \cdot C_i + G_{i+1} \cdot (C_i + P_{i+1} \cdot P_i) \\
&= G_{i+1} + P_{i+1} \cdot dA_i \cdot dB_i + P_{i+1} \cdot (dA_i + dB_i) \\
&\quad \cdot C_i + C_i \cdot G_{i+1} + G_{i+1} \cdot P_{i+1} \cdot (dA_i + dB_i) \\
&= C_i \cdot (G_{i+1} + P_{i+1} \cdot dA_i) + C_i \cdot (G_{i+1} + P_{i+1} \cdot dB_i) \\
&\quad + (G_{i+1} + P_{i+1} \cdot dA_i) \cdot (G_{i+1} + P_{i+1} \cdot dB_i) \\
&= M(C_i, G_{i+1} + P_{i+1} \cdot dA_i, G_{i+1} + P_{i+1} \cdot dB_i) \\
&= M(C_i, dA_{i+1} \cdot dB_{i+1} + dA_{i+1} \cdot dA_i + dB_{i+1} \cdot dA_i, dA_{i+1} \\
&\qquad \cdot dB_{i+1} + dA_{i+1} \cdot dB_i + dB_{i+1} \cdot dB_i) \\
&= M(C_i, M(dA_{i+1}, dB_{i+1}, dA_i), M(dA_{i+1}, dB_{i+1}, dA_i)).
\end{aligned}$$

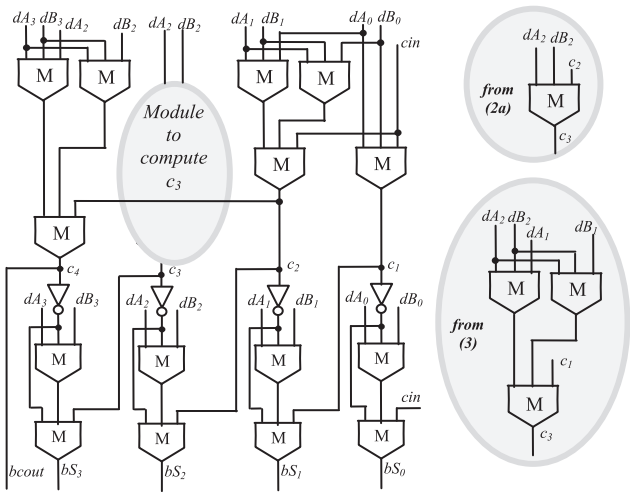

Fig. 2. New ADD1 module.

As a consequence of *Theorem 1*, the 4-b adder ADD1 can be realized by computing the carries as reported in

$$\begin{aligned}
C_1 &= M(dA_0, dB_0, cin) \\
C_2 &= G_1 + P_1 \cdot G_0 + P_1 \cdot P_0 \cdot C_{\text{in}} \\
&= M\left(C_{\text{in}}, M(dA_1, dB_1, dA_0), M(dA_1, dB_1, dB_0)\right) \\
C_3 &= M(dA_2, dB_2, C_2) \\
C_4 &= bcout = G_3 + P_3 \cdot G_2 + P_3 \cdot P_2 \cdot C_2 \\
&= M\left(C_2, M(dA_3, dB_3, dA_2), M(dA_3, dB_3, dB_2)\right).
\end{aligned} \tag{4}$$

Finally, the ith bit of the binary sum $bS_{(3:0)}$ is computed by applying the bit sum representation demonstrated in [5] and here given in

$$bS_i = M\left(C_i, \overline{C_{i+1}}, M(dA_i, dB_i, \overline{C_{i+1}})\right). \tag{5}$$

Fig. 2 depicts the QCA circuit purpose-designed for the ADD1 module exploiting the aforementioned described logic. It is worth noting that, to compute C_3, both (2a) and (3) could be used. The resulting circuits are depicted in the insets of Fig. 2. The former is actually exploited since propagating C_1 instead of C_2 would not reduce the overall delay of ADD1. Indeed, using (3) would lead to unnecessary additional MGs.

Due to *Theorem 1*, the novel circuit exhibits a critical computational path of five MGs and one inverter, which is one MG shorter than the conventional Ripple-Carry Adder (RCA) [26]. Moreover, the novel ADD1 module uses only 16 MGs and 4 inverters, thus overcoming also the conventional 4-b CLA [3], which is implemented using 43 MGs and 4 inverters.

The decimal carry-out $dcout$ and the decimal sum $dS_{(3:0)}$ are then calculated following the novel approach demonstrated in *Theorem 2*.

Theorem 2: If bcout and $bS_{(3:0)}$ are the binary carry-out and the binary sum computed by adding the BCD digits $dA_{(3:0)}$ and

$dB_{(3:0)}$, then the BCD carry-out $dcout$ is given by (6) and the decimal sum $dS_{(3:0)}$ is obtained by (7)

$$dcout = M\left(bcout, M(1, bcout, bS_3), M(bS_3, bS_2, bS_1)\right) \quad (6)$$

$$dS_0 = bS_0 \quad (7a)$$

$$dS_1 = M\left(\overline{M(0, bS_1, dcout)}, M(1, bS_1, dcout), 0\right) \quad (7b)$$

$$dS_2 = M\left(\overline{M(bS_2, dcout, M(0, bS_1, dcout)}\right), M\left(\overline{M\left(bS_2, dcout, M(0, bS_1, dcout)\right)}, bS_2, dcout\right), M(0, bS_1, dcout)) \quad (7c)$$

$$dS_3 = M\big(1, M(\overline{dcout}, bS_3, 0), M\big(M(dcout, \overline{bS_3}, 0), bS_1, 0\big)\big). \quad (7d)$$

Proof: As demonstrated in [21]–[23]

$$dcout = bcout + bS_3 \cdot bS_2 + bS_3 \cdot bS_1$$

that can be rewritten as

$$\begin{aligned} dcout &= bcout \cdot (\text{bcout} + bS_3) + bcout \\ &\quad \cdot (bS_3 \cdot bS_2 + bS_3 \cdot bS_1 + bS_2 \cdot bS_1) + (bcout + bS_3) \\ &\quad \cdot (bS_3 \cdot bS_2 + bS_3 \cdot bS_1 + bS_2 \cdot bS_1) \\ &= M(bcout, bcout + bS_3, bS_3 \cdot bS_2 + bS_3 \cdot bS_1 + bS_2 \cdot bS_1) \\ &= M\left(bcout, M(1, bcout + bS_3), bS_3 \cdot bS_2 + bS_3 \right. \\ &\quad \left. \cdot bS_1 + bS_2 \cdot bS_1\right). \end{aligned}$$

The decimal sum $dS_{(3:0)}$ is computed by adding $bS_{(3:0)}$ to the correction operand $corr_{(3:0)}$ that is equal to 6 when $bS_{(3:0)} > 9$; otherwise, it is set to 0. Due to this, $dS_0 = bS_0 + corr_0 = bS_0$, which trivially proves the correctness of (7a).

To demonstrate the remaining equations, let us consider the binary outputs of ADD1 and their corresponding BCD representation. When $bS_{(3:0)} > 9$, $dcout = 1$ and the operand $corr_{(3:0)}$ can be trivially obtained by setting $corr_3 = corr_0 = 0$ and $corr_2 = corr_1 = dcout$. Considering that $corr_0$ is always equal to 0, it is easy to understand that adding bS_0 and $corr_0$ will always produce a zero carry-out. Therefore, dS_1 can be defined as the XOR between bS_1 and $dcout$

$$dS_1 = \overline{bS_1} \cdot dcout + bS_1 \cdot \overline{dcout}$$

which can be rewritten as

$$\begin{aligned} dS_1 &= (\overline{bS_1} + \overline{dcout}) \cdot (bS_1 + dcout) \\ &= (\overline{bS_1} \cdot \overline{dcout}) \cdot (bS_1 + dcout) \\ &= M\left(\overline{M(0, bS_1, dcout)}, M(1, bS_1, dcout), 0\right) \end{aligned}$$

thus proving (7b).

It can be easily verified also that dS_2 is given by

$$dS_2 = (bS_2 + dcout) \cdot \overline{bS_2} \cdot \overline{bS_1} + dcout \cdot bS_2 \cdot bS_1$$

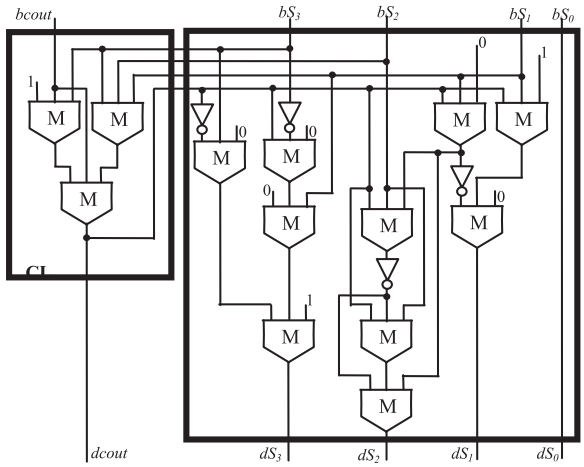

Fig. 3. Module CL and ADD2 of the new BCD adder.

that can be rewritten as follows:

$$\begin{aligned} dS_2 &= (bS_2 + dcout) \cdot \overline{dcout \cdot bS_2} \cdot \overline{dcout \cdot bS_1} + dcout \cdot bS_2 \cdot bS_1 \\ &= \big((bS_2 + dcout) \cdot \overline{dcout \cdot bS_2} \cdot \overline{dcout \cdot bS_1} + dcout \cdot bS_2\big) \\ &\quad \cdot dcout \cdot dS_1 + \overline{dcout \cdot bS_2} \cdot \overline{dcout \cdot bS_1} + dcout \cdot bS_1 \\ &\quad + \overline{dcout \cdot bS_2} \cdot \overline{dcout \cdot bS_1} \cdot dcout \cdot bS_2 + (bS_2 + dcout) \\ &\quad \cdot \overline{dcout \cdot bS_2} \cdot \overline{dcout \cdot bS_1} \\ &= M(\overline{dcout \cdot bS_2} \cdot \overline{dcout \cdot bS_1}, dcout \cdot bS_2 + (bS_2 + dcout) \\ &\quad \cdot \overline{dcout \cdot bS_2} \cdot \overline{dcout \cdot bS_1}, bS_1 \cdot dcout) \\ &= M(\overline{dcout \cdot bS_2 + dcout \cdot bS_1}, M(dcout, bS_2, \overline{dcout \cdot bS_2} \\ &\quad \cdot \overline{dcout \cdot bS_1}, bS_1 \cdot dcout) \\ &= M(\overline{dcout \cdot bS_2 + dcout \cdot bS_1 + dcout \cdot bS_2 \cdot bS_1}, \\ &\quad M(dcout, bS_2, \overline{dcout \cdot bS_2 + dcout \cdot bS_1}, bS_1 \cdot dcout) \\ &= M(\overline{M(dcout, bS_2, dcout \cdot bS_1)}, \\ &\quad M\Big(dcout, bS_2, \overline{M(dcout, bS_2, dcout \cdot bS_1)}, bS_1 \cdot dcout\Big) \\ &= M\left(\overline{M(dcout, bS_2, M(0, bS_1 \cdot dcout)}\right), \\ &\quad M\left(dcout, bS_2, \overline{M\left(dcout, bS_2, M(0, bS_1 \cdot dcout)\right)}, \right. \\ &\quad \left. M(0, bS_1 \cdot dcout)\right). \end{aligned}$$

Finally, dS_3 can be obtained as

$$dS_3 = \overline{dcout} \cdot bS_3 + dcout \cdot \overline{bS_3} \cdot bS_1$$

that can be rewritten as

$$\begin{aligned} dS_3 &= M(\overline{dcout}, bS_3, 0) + M(dcout, \overline{bS_3}, 0) \cdot bS_1 \\ &= M\left(1, M(\overline{dcout}, bS_3, 0), M(dcout, \overline{bS_3}, 0) \cdot bS_1\right) \\ &= M\big(1, M(\overline{dcout}, bS_3, 0), M\big(M(dcout, \overline{bS_3}, 0) \cdot bS_1, 0\big)\big) \end{aligned}$$

thus demonstrating also (7d).

The equations demonstrated earlier have been exploited to design the correction module CL and the 4-b adder ADD2, as illustrated in Fig. 3. It is worth noting that the proposed circuit allows transforming the binary results coming from ADD1 to the BCD format through only four cascaded MGs, which are well below the eight and seven required by the ADD2 adders used with the same objective in [23]–[25] respectively.

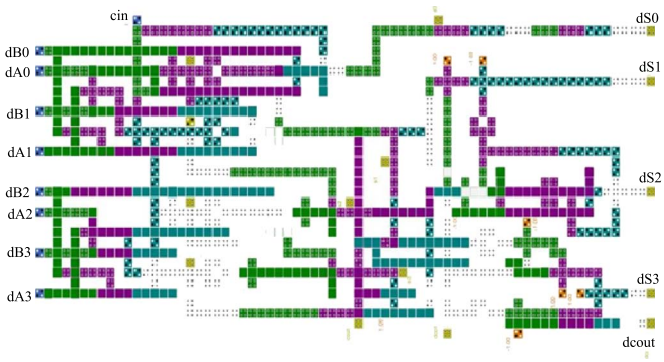

Fig. 4. Layout of the new 1-digit BCD adder.

TABLE I
COMPARISON RESULTS

Adder	# Cells	Area [μm^2]	# Clock Cycles
[24]	-	5.8	-
[25]	1348	2.28	8
[16]	1130	1.77	10
[23] CFA-based	932	1.36	4.75
[23] CLA-based	1838	1.86	2.5
New	1065	0.89	3
New 2DDWAVE	1196	1.36	3.5

IV. RESULTS

In this section, two different implementations of the novel 1-digit BCD adder are characterized. The first one adopts the clock scheme used in [16] and [23]–[25], whereas in the second implementation, the 2-D wave clocking (2DDWAVE) mechanism proposed in [28] is applied. Finally, a 2-digit BCD adder designed as described here is presented. All the proposed circuits were implemented using the QCADesigner tool [27] adopting the simulation setup as in [6]. The first implemented layout is reported in Fig. 4. The generic operation is performed within 12 clock phases (i.e., three clock cycles). Moreover, the adder uses only 1065 cells within an overall area of just 0.89 μm^2. Data summarized in Table I allow a fair comparison with existing competitor circuits to be performed. When compared to the CFA-based circuit presented in [23], the 1-digit BCD adder proposed here occupies $\sim$35% less area and exhibits a computational time $\sim$36% lower. With respect to the CLA-based adder, the area reduction achieved with the novel circuit is $\sim$52%, even though its computational time is 20% higher. It is worth underlining that this time overhead is due to the limited wire lengths adopted within the novel adder, in which, in contrast to [23], wires longer than 16 cascaded cells are avoided.

Nevertheless, the advantages offered by the novel BCD adder can be better appreciated considering that the high-precision computations, in which decimal arithmetic is required, usually process n-digit operands, with $n > 1$. An n-digit decimal adder can be structured as illustrated in Fig. 5, in which n 1-digit adders (DAs) are cascaded in a ripple-carry fashion. The ith DA, with $i = 0, \ldots, n-1$, receives as inputs the digits $dA^i_{(3:0)}$ and $dB^i_{(3:0)}$ of the operands together with the carry signal $dcout^i$ produced by the previous DA and furnishes the digit $dS^i_{(3:0)}$ of the decimal sum and the carry $dcout^{i+1}$.

As it is well known, adder architectures based on the ripple-carry logic perform the most delay critical addition when the carry signal $dcout_1$, generated by the least significant DA, is

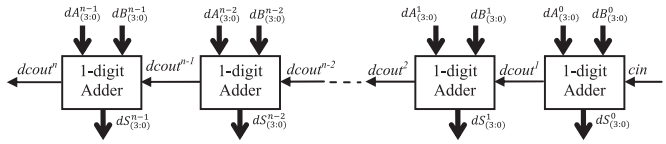

Fig. 5. n-digit BCD adder.

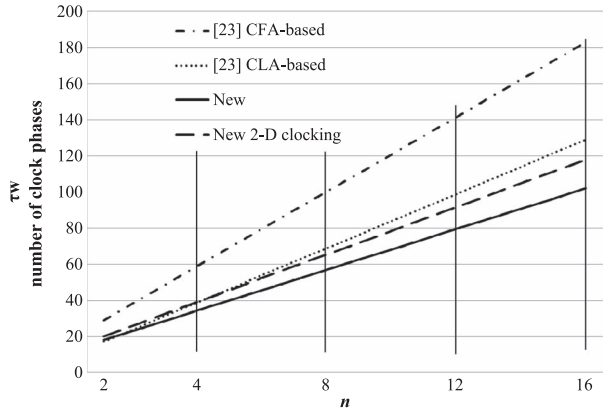

Fig. 6. Worst case delay versus n.

TABLE II
COMPARISON RESULTS

	# clock phases		
	τ_g	τ_p	τ_a
[23] CFA-based	11	11	18
[23] CLA-based	8	8	9
New	8	6	10
New 2-D clocking	9	7	11

propagated through the $n-2$ subsequent DAs to be finally absorbed by the most significant DA that furnishes the digit $dS^{n-1}_{(3:0)}$ and the carry-out $dcout^n$. Therefore, the worst case delay τ_w of the n-digit decimal adder depicted in Fig. 6 is given by (8), where τ_g and τ_p are the delays required to generate $dcout^1$ and to propagate it to the next DA. The most significant DA in the chain will absorb the received carry signal with the delay τ_a

$$\tau_w = \tau_g + (n-2) \cdot \tau_p + \tau_a. \tag{8}$$

Table II reports the generation, propagation, and absorption delays, expressed in terms of the number of clock phases, of the novel 1-digit BCD adder here proposed and those achieved by the CFA- and the CLA-based adders described in [23]. The aforementioned given data were then exploited to estimate the overall computational delay τ_w of the compared architectures versus n. Results reported in Fig. 6 clearly show that the proposed QCA-based BCD adder architecture can reach better speed performances than its competitors.

Two DAs designed as proposed here have been cascaded to implement the 2-digit decimal adder. The produced layout uses 2574 cells, occupies an overall area of 2.74 μm^2, and operates within 18 clock phases. It is worth noting that it is more than twice as large as the 1-digit adder because additional cells are needed to properly synchronize input and output signals, maintaining the geometric regularity of the layout.

As deeply explained in [28], the zone clocking scheme adopted within QCA-based designs can significantly affect the topology of the underlying clock distribution network, thus leading to more or less feasible designs. In particular, some clocking mechanisms allow the clocking circuitry required

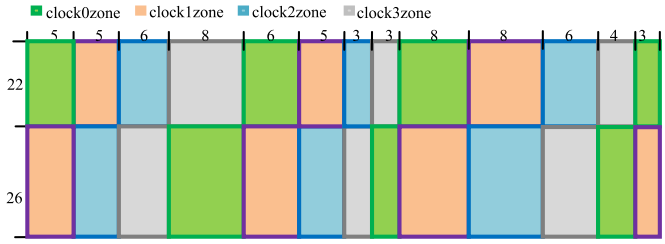

Fig. 7. Adopted clocking scheme.

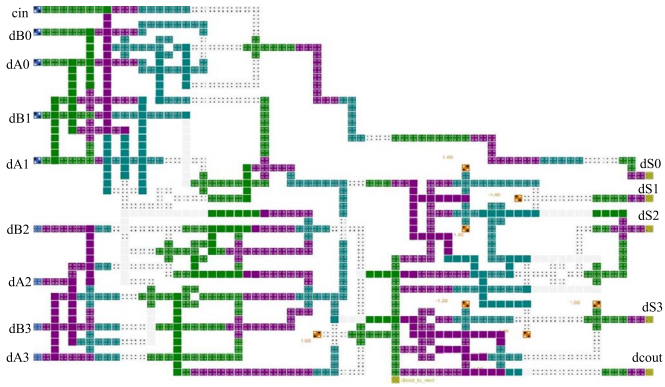

Fig. 8. New BCD adder based on the 2DDWAVE clocking scheme.

to polarize the QCA cells to be significantly simplified, and consequently, they lead to more feasible designs.

In order to demonstrate that the BCD adder here proposed can be efficiently implemented with different clocking schemes, it was designed by also applying the 2DDWAVE mechanism proposed in [28] and partitioning the cells as schematized in Fig. 7, where the size of each clock zone is also shown in terms of the number of cells.

The BCD adder implemented in this way is illustrated in Fig. 8. As summarized in Table I, the adder performs its generic operation within 14 clock phases (i.e., 3 1/2 clock cycles) and uses 1196 cells within an overall area of only 1.36 μm^2. Delay and area overheads due to the different partitioning approach used with the 2-D clocking scheme are the reasonable price to pay to simplify the clock circuitry and to achieve higher feasibility. Nevertheless, if the CFA- and the CLA-based counterparts will be realized using the 2-D clocking scheme, then it would be reasonable to expect similar impacts on performances and area. Thus, the proposed circuit would maintain the advantages as discussed earlier.

V. Conclusion

A new design approach has been presented and demonstrated to achieve efficient QCA-based implementations of decimal adders. Unconventional logic formulations and purpose-designed logic modules here proposed allow outperforming decimal adders known in the literature. In fact, the new 1-digit BCD adder exhibits computational delay and area occupancy up to 36% and 52% lower than existing competitors. These advantages become even more evident when two n-digit decimal numbers must be summed. The 2-digit adder designed as proposed here operates within only 18 clock phases and occupies an area of just 2.74 μm^2. Finally, by exploiting the 2-D wave clocking scheme, a more feasible implementation of the new adder has been obtained without compromising the advantages achieved over its direct competitors.

References

[1] C. S. Lent, P. D. Tougaw, W. Porod, and G. H. Bernestein, "Quantum cellular automata," *Nanotechnology*, vol. 4, no. 1, pp. 49–57, 1993.
[2] V. Pudi and K. Sridharan, "New decomposition theorems on majority logic for low-delay adder design in quantum dot cellular automata," *IEEE Trans. Circuits Syst. II, Exp. Brief*, vol. 59, no. 10, pp. 678–682, Oct. 2012.
[3] H. Cho and E. E. Swartzlander, Jr., "Adder design and analyses for quantum-dot cellular automata," *IEEE Trans. Nanotechnol.*, vol. 6, no. 3, pp. 374–383, May 2007.
[4] H. Cho and E. E. Swartzlander, Jr., "Adder and multiplier design in quantum-dot cellular automata," *IEEE Trans. Comput.*, vol. 58, no. 6, pp. 721–727, Jun. 2009.
[5] V. Pudi and K. Sridharan, "Efficient design of a hybrid adder in quantum-dot cellular automata," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 19, no. 9, pp. 1535–1548, Sep. 2011.
[6] S. Perri, P. Corsonello, and G. Cocorullo, "Area-delay efficient binary adders in QCA," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 22, no. 5, pp. 1174–1179, May 2014.
[7] L. Lu, W. Liu, M. O'Neill, and E. E. Swartzlander, Jr., "QCA systolic array design," *IEEE Trans. Comput.*, vol. 62, no. 3, pp. 548–560, Mar. 2013.
[8] S. Perri, P. Corsonello, and G. Cocorullo, "Design of efficient binary comparators in quantum-dot cellular automata," *IEEE Trans. Nanotechnol.*, vol. 13, no. 2, pp. 192–202, Feb. 2014.
[9] S. Perri and P. Corsonello, "New methodology for the design of effic ient binary addition in QCA," *IEEE Trans. Nanotechnol.*, vol. 11, no. 6, pp. 1192–1200, Jun. 2012.
[10] C. A. Krygowski *et al.*, "Functional verification of the IBM system z10 processor chipset," *IBM J. Res. Dev.*, vol. 53, no. 1, pp. 1–11, Jan. 2009.
[11] B. Sinharoy *et al.*, "IBM POWER8 processor core microarchitecture," *IBM J. Res. Dev.*, vol. 59, no. 1, pp. 1–21, Jan./Feb. 2015.
[12] L. Eisen *et al.*, "IBM POWER6 accelerators: VMX and DFU," *IBM J. Res. Dev.*, vol. 51, no. 6, pp. 1–21, Nov. 2007.
[13] Sparc64^{TM} X/X+ Specification Ver. 29.0 Fujitsu, Kawasaki, Japan, Jan. 2015. [Online]. Available: http://www.fujitsu.com
[14] "SilMinds DFP adder units," SilMinds, Cairo, Egypt. [Online]. Available: http://www.silminds.com/ip-products/dfp-units/56
[15] *IEEE Standard for Floating-Point Arithmetic, IEEE Working Group of the Microprocesser Standards Subcommittee*, IEEE Std. 754-1985, 2008.
[16] M. Gladshtein, "Quantum-dot cellular automata serial decimal adder," *IEEE Trans. Nanotechnol.*, vol. 10, no. 6, pp. 1377–1382, Jun. 2011.
[17] E. M. Schwarz, J. S. Kapernick, and M. FCowlishaw, "Decimal floating-point support on the IBM system z10 processor," *IBM J. Res. Dev.*, vol. 53, no. 1, pp. 1–10, Jan. 2010.
[18] R. D. Kenney and M. J. Schulte, "High-speed multioperand decimal adders," *IEEE Trans. Comput.*, vol. 54, no. 8, pp. 953–963, Aug. 2005.
[19] C. Tsen, S. Gonzalez-Navarro, M. J. Schulte, and K. Compton, "Hardware designs for binary integer decimal-based rounding," *IEEE Trans. Comput.*, vol. 60, no. 5, pp. 614–627, May 2011.
[20] L. Han and S. B. Ko, "High-speed parallel decimal multiplication with redundant internal encodings," *IEEE Trans. Comput.*, vol. 62, no. 5, pp. 956–968, May 2013.
[21] A. A. Bayrakci and A. Akkas, "Reduced delay BCD adder," in *Proc. IEEE Int. Conf. Appl.-Specific, Syst., Archit. Process.*, Montreal, QC, Canada, Jul. 2007, pp. 266–271.
[22] C. Sundaresan, C. V. S. Chaitanya, P. R. Venkateswaran, S. Bhat, and J. M. Kumar, "Modified reduced delay BCD adder," in *Proc. IEEE Int. Conf. Biomed. Eng. Informat.*, Shanghai, China, Oct. 2011, pp. 2148–2151.
[23] W. Liu, L. Lu, M. O'Neill, and E. E. Swartzlander, "Cost-efficient decimal adder design in quantum-dot cellular automata," in *Proc. IEEE Int. Symp. Circuits Syst.*, Seoul, South Korea, May 2012, pp. 1347–1350.
[24] M. Taghizadeh, M. Askari, and K. Fardad, "BCD computing structures in quantum-dot cellular automata," in *Proc. IEEE Int. Conf. Comput. Commun. Eng.*, Kuala Lampur, Malaysia, May 2008, pp. 1042–1045.
[25] F. Kharbash and G. M. Chaudhry, "The design of quantum-dot cellular automata decimal adder," in *Proc. IEEE Int. Multitopic Conf.*, Karachi, Pakistan, Dec. 2008, pp. 71–75.
[26] V. Pudi and K. Sridharan, "Low complexity design of ripple carry and Brent-Kung adders in QCA," *IEEE Trans. Nanotechnol.*, vol. 11, no. 1, pp. 105–119, Jan. 2012.
[27] K. Walus and G. A. Jullien, "Design tools for an emerging SoC technology: Quantum-dot cellular automata," *Proc. IEEE*, vol. 94, no. 6, pp. 1225–1244, Jun. 2006.
[28] V. Vankamamidi, M. Ottavi, and F. Lombardi, "Two-dimensional schemes for clocking/timing of QCA circuits," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 27, no. 1, pp. 34–44, Jan. 2008.