Exploiting the Third Dimension: Stackable Quantum-dot Cellular Automata

Willem Lambooy
Radboud University
Institute for Computing and Information Sciences
Nijmegen, Gelderland, Netherlands
willem.lambooij@ru.nl

Marcel Walter
Technical University of Munich
Chair for Design Automation
Munich, Bavaria, Germany
marcel.walter@tum.de

Robert Wille*
Technical University of Munich
Chair for Design Automation
Munich, Bavaria, Germany
robert.wille@tum.de

ABSTRACT

The exponential growth of transistor density in integrated circuits is doomed to fail at the limits of physics in the foreseeable future. *Quantum-dot Cellular Automata* (QCA) is a post-CMOS contestant from the emerging *Field-coupled Nanocomputing* (FCN) paradigm which offers computations with tremendously low power dissipation. Recent physical accomplishments in this area also motivated the developments of corresponding design automation methods. However, although the higher integration density of QCA makes this technology a promising candidate for stacked, i. e. cuboid-like, chip architectures, all design automation solutions proposed thus far are limited to 2-dimensional architectures only. This work showcases the potential when the third dimension is additionally utilized. To this end, we must overcome certain obstacles for which corresponding solutions are proposed. Case studies on important regular structures such as bitwise AND/OR, binary adders, or multiplexers—for which we provide automatic generation scripts—confirm that exploiting the third dimension in this fashion yields a prodigious reduction in area occupation and cell count, differing by several orders of magnitude compared to the state of the art.

CCS CONCEPTS

• **Hardware → Quantum dots and cellular automata**; *Wire routing*; **Physical synthesis**; *Placement*; *Clock-network synthesis*; Software tools for EDA.

1 INTRODUCTION & MOTIVATION

The contemporary semiconductor chip technology reduces transistor sizes at a rate that is asymptotic; a direct consequence of the increasing effect of the uncertainty principle that plays at the quantum scale. Exactly this effect enables the formation of *Quantum-dot Cellular Automata* (QCA): a technology concept from the emerging *Field-coupled Nanocomputing* (FCN) paradigm that operates on the Coulombic repulsion between individual charges [1].

Correspondingly, QCA performs computation with extremely low energy dissipation, which allows it to be integrated much denser than current CMOS technology nodes allow for.

The task to generate a QCA circuit layout from a specification, which is usually given by the means of a logic network, is called *physical design*. It involves considerations of gate placement, wire routing, clock zone assignment, and timing awareness. The problem is known to be computationally intractable, even under various relaxations [26].

Nevertheless, a handful of automatic approaches that tackle the physical design of QCA systems exist; e. g., [3, 23–25, 27]. However, these algorithms suffer from greedy layout consumption by long wire segments for routing or do not scale on large input networks. Main reasons for these shortcomings are stricter technology-specific constraints that render the application of established VLSI approaches unsuitable.

Overcoming this major roadblock in the process of developing a complete design automation flow for QCA systems is the goal of current scientific endeavors in the community. Currently, all composed circuits have to be obtained either manually or via the aforementioned placement and routing techniques.

Additionally, to the best of the authors' knowledge, all existing automatic methods target 2-dimensional QCA technologies—with only short wire segment crossovers—even though QCA's intrinsic property of operating with radial fields as well as its higher possible integration density make it a prime candidate for 3-dimensional, i. e., stacked architectures. These architectures have in fact been introduced in the literature with the potential physical implementation of 2D electron gases [4], and their benefits are known from similar technologies like *perpendicular nanomagnetic logic* (pNML) [17].

However, certain obstacles need to be overcome when exploring the third dimension for QCA layout generation. Unwanted signal inversions must be handled, and an inconsistent cell spacing needs to be adopted to facilitate equal strength signal transmission in each dimension.

This work investigates these physical issues and proposes solutions that enable the exploitation of the third dimension in QCA layout generation. Furthermore, it introduces novel QCA circuit classes, namely *linearly* and *arboreally stackable* designs, that can be stacked in the third dimension to increase their bitwidth without costly placement and routing. Additionally, since these circuits expand in the third dimension, they only require modest chip area in the plane; in fact, an improvement of several orders of magnitude in comparison to the state of the art is achieved. Thereby, they can be used as extensible 3D building blocks for *cuboid* chip architectures. Alongside the initial definition of these circuit classes, this work also proposes a programmatic solution in Haskell to the automatic obtainment of n-bit QCA circuitry formed from said classes. The code and all obtained design files are made publicly available at https://www.github.com/wlambooy/QCA-STACK.

The remainder of the paper is structured as follows: Section 2 discusses select background on QCA. Section 3 introduces the notion of stackable QCA layouts, discusses their potential as well as physical issues that come with the approach, and proposes corresponding solutions. Section 4 introduces two methods for creating seamlessly stackable QCA layouts from base designs with the help of running examples. Section 5 showcases results obtained from the proposed methods and compares them against the state of the art. Finally, Section 6 concludes the paper.

2 PRELIMINARIES

There exist a variety of physical implementations that build upon the QCA concept of which 2D electron gases and pNML are just two particularly well suited ones for 3D architectures [4, 17]. This work shall, however, not favor any particular QCA technology.

At the core of a quantum-dot cellular automaton is the QCA cell, a 2-dimensional plane with sides of equal length. Inside are four quantum-dots that are both evenly spaced apart from each other and from the center and the nearest corner of the cell [13]. Each quantum-dot is a nanoscopic (molecular) structure that functions as a site that can be occupied by at most one electron [10]. When exactly two electrons are put into a single cell, the repelling *Coulomb forces* between them will always push them into opposite corners. From this, two possible stable states emerge that are denoted as $P = -1$ and $P = +1$, and can be read as logical 0 and logical 1 respectively [12, 13].

Example 2.1. Figure 1 depicts eight QCA cells aligned adjacently, where each square represents the outline of a cell, each circle represents a quantum dot, and each bullet represents a charge located

*Also with Software Competence Center Hagenberg (SCCH).

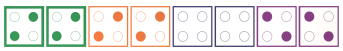


Figure 1: A clocked QCA cell wire

at such a dot. The green and orange cells are polarized to logical 1, while the purple cells are polarized to logical 0. The blue cells are unpolarized and represent a *null* state without retrievable information. This makes QCA a binary, but not a ternary technology.

While it is possible for electrons to tunnel to nearby unoccupied quantum dots in the same cell, they cannot tunnel to quantum dots in adjacent cells [13]. Nevertheless, their charge influences the behavior of neighboring cells so that they align their polarizations accordingly. A line of QCA cells thus forms a wire through which information is transferred by Coulomb interaction alone—without the flow of electric current [9]. The cells in Figure 1 are aligned as such a binary wire segment. It can be seen that the first four cells aligned their polarizations. However, a cell's influence rapidly decays over distance. As the number of cells and the temperature of a QCA system increase, cells occasionally assume incorrect polarizations with an increasing probability. The energy difference between the ground state (the state of lowest energy) and the first excited state (a state with incorrect polarization) is referred to as *kink energy* [9, 13]. This notion can be used to calculate cell polarizations given external parameters.

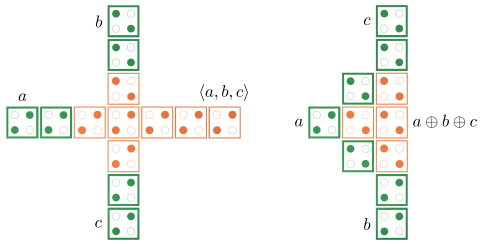
To address this problem, a larger QCA system is usually divided into smaller segments and provided with an electric field generator that controls the tunneling characteristics of the cells; a concept known as *clocking* [5, 11]. Intuitively, a clock field applied in this way ensures that the affected cells assume a fixed polarization and cannot change their state until the field is turned down again. If four repeating field signals are distributed across a QCA system (each of them shifted clockwise by 90°), a pipeline-like information flow of affected cells is achieved.

Example 2.2. Consider again Figure 1. Here, the color code represents the different clocks fields employed to the cell. Green cells are in a fixed state of stable information (*hold*), and are therefore depicted with a slightly thicker cell wall. Conversely, the orange cells are currently switching to this state (*switch*). At the same time, the blue cells are depolarized (*relax*) and the purple cells are reverting from their previous polarization (*release*).

Clocking is one of the most fundamental concepts of information propagation in QCA. The de-facto state-of-the-art QCA design rules utilize a *tile-based* scheme that tiles the circuit into uniformly sized clock zones, e. g., 5×5 cells, and assigns clock numbers to these blocks [6]. For the ease of depiction, the QCA systems investigated in this work use clock zone sizes that are smaller than commonly used ones. However, they can easily be translated into feasible layouts by duplicating their cells to obtain respectively sized blocks. Using those clock zones and additionally arranging QCA cells in specific topological structures, their Coulombic influence can yield computational properties, i. e., can be utilized to implement combinational logic [9, 22].

Example 2.3. Figure 2a depicts a MAJ gate that implements $\langle a, b, c\rangle := ab + ac + bc$ [22]. Figure 2b showcases an XOR gate [18]. In both gates, the Coulombic pressure exerted by the inputs on the center cell causes it to assume a polarization in accordance to the strongest forces acting upon it—eventually yielding the output value that realizes the respective function.

The composition of cells into larger computational structures is called *physical design* and is usually performed on an abstraction level that is independent of concrete cell positions [3, 23, 25, 27]. Instead, gates or components are utilized whose internal cell structures are known beforehand [16]. A sufficiently large composition of cells is called a *(circuit) layout*. In the past, QCA circuit layouts presented in the literature were almost exclusively *semi-planar*, i. e., all their logic



(a) Majority [22] **(b) XOR [18]**

Figure 2: Planar 3-input QCA gates

elements were placed in the plane with only short wire-crossings in a second layer [3, 23, 25, 27].[1]

The automatic obtainment of layouts from specifications is one of the most crucial tasks in QCA design automation and is known to be $\mathcal{NP}$-complete [26]. Due to its peculiar constraints, conventional algorithms for transistor logic cannot be applied to the QCA domain.

3 INTRODUCING STACKABLE QCA LAYOUTS

A major problem in today's physical design of QCA layouts—even for regular structures such as n-bit logical operations, n-bit binary adders, or multiplexers—is that the correspondingly needed routing requires a substantial overhead in terms of area. In this work, we propose to overcome this shortcoming by exploiting the third dimension of QCA, and introducing what we coin *stackable QCA layouts*. These denote designs that can be seamlessly extended in the third dimension to increase their bitwidth. This section provides a corresponding motivation and also discusses the obstacles that come with 3D QCA design. Based on that, the remainder of this work then provides solutions that exploit this potential and concurrently overcome the obstacles.

3.1 Potential of 3-Dimensional QCA

Using the basic concepts reviewed in the previous section allows to realize arbitrary functionality in QCA by composing corresponding gates together as discussed in Example 2.3 and illustrated in Figure 2. Similarly, strictly regular circuit components such as n-bit AND/OR operations, n-bit binary adders, or multiplexers can be realized by composing the single building blocks (i. e., the AND/OR gates, full adders, or MUX gates) together. However, while the underlying concepts are rather simple, realizing those layouts in QCA requires a substantial area overhead due to its 2-dimensional architecture. This makes routing, i. e., connecting the single building blocks, rather tedious as illustrated by the following example.

Example 3.1. Figure 3 shows the QCA layout of a 4-bit *Ripple Carry Adder* (RCA) as it was proposed in [16]. While the layout is defined by a diagonal composition of four *full adders* in the plane, the figure also clearly shows that connecting wires consume a substantial amount of the layout area.

The main reason for this circumstance is that only one layer is available for wire routing, which causes a need for long detours. Another reason is the compensation of delay differences. Due to QCA's pipeline-like clocking behavior, circuits propagate information in discrete time steps, and meticulous care must be taken to ensure that all signal transmission times in the layout are identical. This is often solved by long and convoluted wires, as depicted in the figure. Consequently, a lot of chip area is wasted.

This obstacle is known by the term *routing congestion*. Such problems can be found all over the literature, e. g., [14, 20]. In all these cases, designers have to handle this severe drawback that comes with the 2-dimensionality of QCA layouts.

[1] Furthermore, *coplanar* wire-crossings allowed the construction of crossover segments in the plane by rotating the crossing cells by 45° [22]. However, this technique requires additional area for converting a non-rotated signal to a rotated one and vice versa.

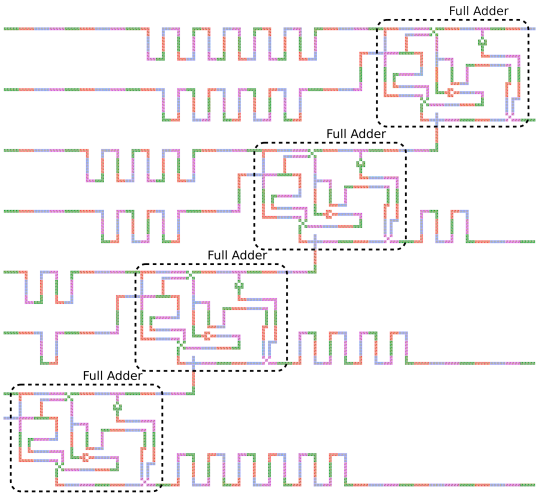


Figure 3: Recreation of the planar 4-bit QCA RCA proposed in [16]. Long wire detours have been created for routing and delay compensation.

At the same time, however, the underlying Coulomb interaction that provides the basis for QCA is by no means a 2-dimensional phenomena. Its force does not only act between cells in the plane but naturally extends in all directions—including the third dimension. Physically, this idea has already been considered in [4] where a 3D QCA architecture was proposed, which acts as justification for this work to explore it for layout generation.

Conceptually, using interactions of QCA cells not exclusively in 2-dimensional directions, but additionally exploiting the third dimension, suddenly offers plenty of further opportunities to connect building blocks and, hence, avoids the routing congestion problem illustrated above. At the same time, simply using the Coulomb interaction in the third dimension is not straightforward, as it imposes some obstacles to overcome. Those are discussed next.

3.2 Resulting Obstacles

In order to avoid the routing congestion problem illustrated above, we propose to exploit the third dimension, i. e., to *stack* QCA cells on top of each other—leading to the stackable QCA layouts proposed in this work. However, to really utilize this concept, two major obstacles have to be considered:

(1) In contrast to adjacent cells in 2-dimensional layouts, stacking QCA cells causes unwanted signal inversions that need to be given particular attention.
(2) The physical spacing of stacked cells has to be adjusted so that the Coulombic interaction acts on adjacent cells with the same force regardless if they are in the 2-dimensional plane or in a stack.

More precisely, the unwanted signal inversions are illustrated in the following example.

Example 3.2. Figure 4a depicts three stacked QCA cells. Assuming the bottom cell to have a fixed polarization of $P = -1$, i. e., logical 0, its signal is propagated upwards via Coulombic forces as discussed in Section 2. The tendency of charges to occupy quantum-dots that are farthest from each other yields a logical inversion per cell in the third dimension. Thus, the middle cell assumes polarization $P = +1$ (logical 1), and the top one again resolves to $P = -1$ (logical 0). As shown by this, an odd number of cells is required in each stack to obtain unaltered signals at the target level.

Without loss of generality, we propose to address this obstacle by stipulating that the addition of another QCA layer in the third dimension always requires exactly one additional *via layer*, i. e., a layer that merely contains propagating cells but no logic. To this end, for propagating information up or downwards, QCA cell stacks of height 3 are used, but can easily be extended to 5, 7, etc. if needed for, e. g., technology constraint reasons. This restriction is used in the remainder to construct multi-layered QCA, functionally consisting of layer pairs connected by one conducting via layer.

The obstacle concerning the dedicated spacing in between cells is rooted in the fact that, naturally, the Coulomb interactions between cells should be the same in all dimensions (e. g., an interaction in the second dimension must not be stronger or weaker than an interaction in the third dimension). Since these forces are stronger the closer the cells are placed together, this needs to be properly adjusted. For the second dimension, established standard values for the spacing of cells already exist (and are, e. g., also preset in physical simulators such as *QCADesigner* [29]). For the third dimension, those values still have to be determined. The following example illustrates how this can be accomplished by utilizing the kink energy as touched upon in Section 2.[2]

Example 3.3. When horizontal and vertical spacing are set to be equivalent, the absolute kink energy $|E^k|$ between some adjacent cells c_0 and c_1 in the plane evaluates as follows: $|E^k_{c_0,c_1}| \approx 0.82$. This factor is largely decided by the shortest distances between the quantum-dots in c_0 and the ones in c_1. A cell c_2 that is adjacent to c_1 in the third dimension, i. e., one that is directly above or below it, does not share the shortest distances of the previously considered coplanar adjacency. Hence, their mutual absolute kink energy differs: $|E^k_{c_0,c_2}| \approx 0.11$.

Based on that, the required layer spacing can be determined by setting the kink energy between a cell and its directly adjacent coplanar neighbor equivalent to the negated kink energy between said cell and its vertical (non-coplanar) neighbor. This *pragmatical layer spacing* was calculated to be approximately $0.581215 \cdot H$, where H denotes the horizontal distance between the centers of two directly adjacent coplanar cells. Evaluating the absolute kink energy equations from Example 3.3 again with said layer spacing set, the following results: $E^k_{c_0,c_1} \approx -E^k_{c_0,c_2}$. Hence, intuitively speaking, the cell spacing in the third dimension must be set to $\approx 58\,\%$ of the cell spacing in the plane to facilitate that signal transmission in each spatial dimension is of equal strength.[3] Thus, applying this method allows for the proposed stackable QCA designs.

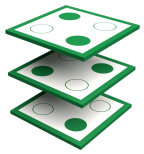
4 METHODOLOGIES FOR STACKING QCA

Having the concept of stackable QCAs and solutions for the correspondingly resulting obstacles, this section now proposes corresponding design methodologies that exploit the additional potential of the third dimension. This potential substantially depends on the respectively given layout to be realized, and on the needed routing of the corresponding building blocks. If employed in a smart fashion, stackable QCAs may not require any or only small overhead and, hence, may overcome the routing congestion problem illustrated in Example 3.1 and Figure 3. In this section, we are proposing two corresponding methodologies that accomplish that: one leading to linearly stackable layouts and another leading to arboreally stackable layouts. Both of them are described by means of representative examples, while further cases are considered and summarized later in Section 5.

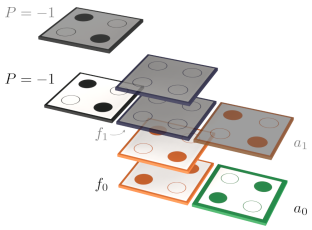
4.1 Linearly Stackable Layouts

Input sizes for most logical and arithmetical functions grow uniformly linearly with an increase in dimensionality. Unmissable functions that fall under this characterization are the bitwise n-bit AND/OR as well as the n-bit binary adder. This section demonstrates a method of designing QCA layouts for elementary building blocks of such linear

[2] Kink energy factors were obtained using the adapted bistable approximation method found in [8].
[3] It is to be noted that the precise values might change with the establishment of novel QCA implementations. However, the general concept preserves validity.



(a) Three QCA cells stacked on top of each other. The signal from the bottom cell is retained in the top cell due to the double signal inversion.



(b) The proposed stackable bitwise AND/OR gate showing both the base design (non-gray cells) and the once-stacked layout. The fixed polarization cell toggles the gate function ($P = +1$ for OR and $P = -1$ for AND).

Figure 4: Basic stacked QCA

functions, in a manner that allows for *linear* stacking to achieve an increase in bitwidth. In the following, this is described and illustrated by using a bitwise n-bit AND/OR as a simple and intuitive example to get familiar with the concept, and a n-bit RCA as a more sophisticated representative.

4.1.1 Bitwise n-bit AND/OR. A main basic logic gate in QCA is the 3-input MAJ voter as reviewed in Section 2. From that, 2-input AND/OR gates are created by assigning a fixed polarization of logical 0/1 to one of the inputs. Consequently, n-bit AND/OR gates are then derived from the 2-input primitives, e. g., via costly placement and routing.

Instead of having all non-constant AND/OR inputs in the same plane, the linearly stackable QCA scheme utilizes one layer per input and propagates the resulting signal upwards. By this, a 2^n-bit design can be obtained by stacking the 2^{n-1}-bit design on top of itself. Figure 4b shows the resulting (once-stacked) bitwise AND/OR gate (it is to be noted that the only difference between the AND and OR functionality is the polarization of the fixed cells). Here, the grayed out cells represent the new stack, which introduces a new input cell such that the once-stacked design outputs $a_0 \odot a_1$, with $\odot \in \{\cdot, +\}$. Each added 1-bit stack moves the output cell up by two layers (one via layer and one regular layer).

By polarizing the input cell a_0 to logical 1, the output cell f_0 is polarized to logical 1 as well. When stacked once with input cell a_1 set to logical 1 as shown, the new output cell f_1 is also assuming a logical 1 state. Thereby, the output signal is eventually continuously propagated upwards through the individual AND/OR-gates, and is ultimately leading to $f_{n-1} \odot a_n$ at the nth AND/OR-gate, where f_n is the value of the output signal after the nth AND/OR-gate.[4]

Note that, at a first glance, it may seem impractical that the output is located at height n (not counting via layers)—in particular, if the output of an n-bit 3D design is to be used as input to subsequent circuitry. However, the subsequent circuitry can simply be turned upside down to have its first input at the top instead. Hence, the stackable solution certainly offers a substantial area improvement compared to the approach by [16] depicted in Figure 3.

4.1.2 n-bit RCA. As a more complex representative of the proposed linear stacking method, a stackable RCA is presented whose base design is inspired by [18].

The concept of an RCA was briefly touched upon in Example 3.1. As already mentioned, it is usually implemented as a chain of full adder circuits that each perform the following two three-input functions of

[4]There are methods to compute the bitwise AND/OR in $\log(n)$ steps, hence, this linear method of scaling does not produce an optimal realization but is supposed to serve as a simple representative of a linearly stackable design. Given the right scaling algorithm, the design—be it with minor tweaks—can be scaled arboreally to produce a more efficient circuit. This can be derived from the fact that each two branches, B_1 and B_2, can be merged to one branch R with the same operation: $R = B_1 \odot B_2$, for example. A realization of this concept is explored later in this paper in Table 1.

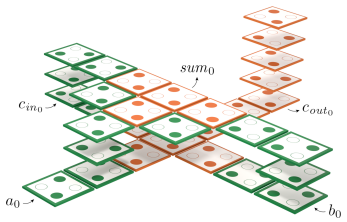


Figure 5: The proposed full adder. The bottom layer implements $c_{out_0} = \langle a_0, b_0, c_{in_0} \rangle$, which is propagated upwards on the right. The upper layer implements $sum_0 = a_0 \oplus b_0 \oplus c_{in_0}$ with its output in the center.

the shared variables a, b, c_{in}:

$$sum := a \oplus b \oplus c_{in},$$
$$c_{out} := \langle a, b, c_{in} \rangle,$$

i. e., the respective XOR and MAJ of the three input variables.

For the proposed stackable design, the MAJ and XOR gate from Figure 2 are located above one another (with the MAJ gate in the bottom layer), so that the input cells in the lower layer propagate upwards to serve as inputs to upper plane. This constitutes a single full adder, which is depicted in Figure 5. While the c_{out_0} output propagates upwards (as it must feed into the next stage), sum_0 can be passed out of the stack as primary output. There has been left sufficient space to do so by inserting extra wire cells.

A full adder can be considered a 1-bit RCA. We propose to utilize a 2-bit RCA as the base design for the stackable n-bit RCA. Said 2-bit RCA is obtained by duplicating the 1-bit design, rotating it by $-90°$, mirroring it, and then stacking it on top of itself; thereby feeding c_{out_0} as input to c_{in_1} and exhibiting two new inputs: a_1 and b_1. This 2-bit RCA is depicted in Figure 6a. Having this building block, no more rotation and mirroring is necessary to increase its bitwidth arbitrarily further. Instead, it can be seamlessly stacked, doubling its input width with every step. Figure 6b showcases the 4-bit RCA that results from stacking the base design once.

Most arithmetical and logical circuits can be realized in this way. However, there are also special cases which do not allow linear stacking despite a certain regularity. These are circuits whose inputs do not all grow linearly with increasing bitwidth such as the multiplexer, whose *select* input grows only logarithmically in size. A stacking solution for such circuits is presented in the following section.

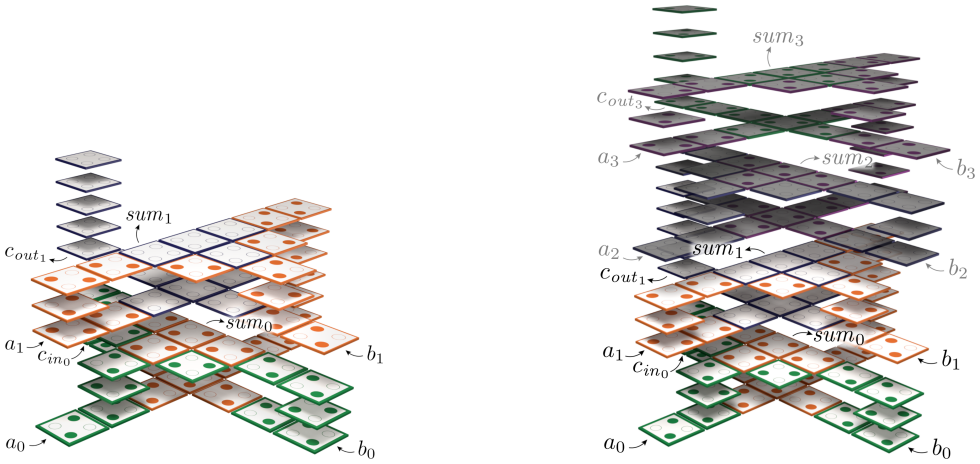
4.2 Arboreally Stackable Layouts

Having introduced the linearly stackable circuits and, thus, the general idea of stackable QCA, this section is dedicated to stacking designs arboreally. As mentioned in the previous section, certain function classes possess inputs that do not grow linearly but logarithmically with a linear increase in overall bitwidth. In this section, the n-to-1 multiplexer (n:1-MUX) is used as a representative of said circuit class.

The n:1-MUX is categorized by n *data* inputs $d_0, \ldots, d_{n-1}$ plus $\lceil \log(n) \rceil$ *select* inputs $s_0, \ldots, s_{\lceil \log(n) \rceil - 1}$, as well as 1 primary output $MUX := d_i$, where i is the decimal representation of the binary string $[s_{\lceil \log(n) \rceil - 1} \ldots s_0]_2$. Intuitively, the s inputs select a d input to pass through by encoding its index in unsigned binary notation.

The logarithmic growth of the n:1-MUX imposes a challenge for the proposed stacking method since a linear stacking would introduce a linear amount of both new *data* and *select* inputs. It can, however, be observed that the n:1-MUX grows similarly to a binary tree where the number of leaves is equivalent to the number of *data* inputs and the number of levels is equivalent to the number of *select* inputs.

It is therefore proposed to create a base design that is arboreally stackable, i. e., that can be used as nodes of a binary tree. This means that two base designs are always combined by one in the next higher layer. This requires an additional routing layer (plus two via layers)



(a) 2-bit RCA (base design) **(b) 4-bit RCA (stacked once)**

Figure 6: The 2-bit RCA base design and the respectively obtained once-stacked 4-bit RCA.

on which the corresponding inputs and outputs are connected. The routing differs slightly depending on the base design's topology, but is generally straightforward and can be determined by a linear algorithm.

Similar to the previous section, where a 2-bit RCA was constructed as the base design, a 4:2-MUX is used here. The layout consists of two 2:1-MUX that were inspired by [19] and fused together in one layer where the *select* inputs were merged into one common input. For the root node of the resulting binary tree, a third 2:1-MUX is stacked on top to bundle the 2-bit output signal into a single 1-bit signal. The resulting 4:1-MUX root design is visualized in Figure 7a, and the once arboreally stacked 8:1-MUX in Figure 7b. The emerging binary tree structure is provided in the latter.

When stacking, the inputs a_0, a_1, b_0 and b_1 are fed in from the bottom, while the output feeds its signal to the top. Level n of the resulting binary tree (assuming the root to be in level 0) then consists of $n + 1$ 4:2-MUXs placed in the same plane with regular spacing which allows for seamless stacking. Additionally, each level must be connected by a supplementary routing layer (which yields a total of three added layers because of the obligatory extra two via layers).[5]

Table 1: Proposed linearly vs. arboreally stackable AND/OR design characteristics

Bits		x	y	z	V	Cells	nm^2	D
2-bit	lin.	3	1	5	15	8	1044	2
	arb.	3	2	3	18	8	2204	1
4-bit	lin.	3	1	9	27	16	1044	4
	arb.	7	2	7	98	28	5244	3
8-bit	lin.	3	1	17	51	32	1044	8
	arb.	15	4	11	660	88	23244	5
16-bit	lin.	3	1	33	99	64	1044	16
	arb.	31	8	14	3472	280	97644	9

x, y, z Volumetric aspect ratio V Layout volume in cells
Cells Number of QCA cells nm^2 Layout area in nm^2
D Delay in clock phases

5 EXPERIMENTAL EVALUATION

The proposed procedures for the obtainment of arbitrary linearly and arboreally stackable QCA circuits from base designs were implemented in Haskell. They receive a base design as input and stack it an arbitrary amount of times by applying their respective strategy. These algorithms were used to generate 3D QCA n-bit layouts of the proposed AND/OR, RCA, and MUX. While the n-bit RCA can only be obtained from linear stacking and the n:1-MUX only from arboreal stacking, the AND/OR gate operates with both algorithms, resulting in either a linear implementation or a logarithmic one.

Table 1 provides a quick comparison of the linearly and the arboreally stackable AND/OR gate for increasing bitwidths. The column V lists the resulting layout volume, i. e., $x \cdot y \cdot z$. Column *Cells* denotes the total number of QCA cells present in the designs, and column nm^2 shows the resulting layout footprint in nm^2, using default values for cell sizes established by QCADesigner: 18 nm × 18 nm with a spacing of 2 nm in both x and y direction. The column *Delay* denotes the signal delay from primary inputs to primary outputs in clock phases (fourths of full clock cycles). Signal delay and cell count are proxy criteria for technology-dependent speed and power consumption respectively.

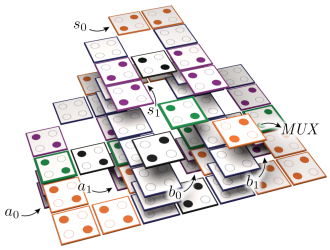
In addition, Table 2 summarizes some layout characteristics for the RCA and MUX and compares them to various designs found in the literature where applicable. The columns x, y, and z list the layout dimensions in cells in width, depth and height respectively. All other columns are analogous to Table 1. As can be seen in both tables, the footprint of linearly stackable designs does not increase with an increase in bitwidth; the layouts are merely growing in the third dimension, thus avoiding large layout consumption in the plane.

This benefit becomes especially apparent when comparing the 3D layouts of the RCA and the MUX against the state of the art. While the footprints of a 16-bit RCA already approaches 30 billion nm^2, our proposed design stays at constant 19 044 nm^2. A similar trend can be observed for the MUX, where a 8:1-MUX from the literature consumed 580 000 nm^2, while our proposed one consumes merely 41 124 nm^2.

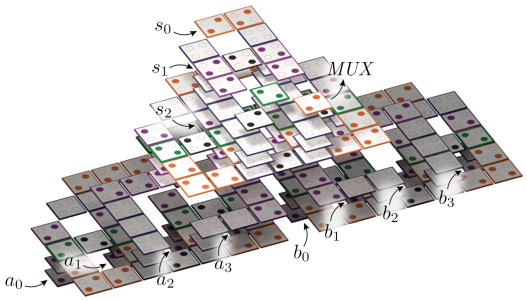
Furthermore, the regularity and straightforward stackability allows generating designs in bitwidths that could not be found in the literature, e. g., a 1024-bit RCA and a 1024:1-MUX. This clearly showcases the benefits of our proposed methods.

Although the proposed proof-of-concept designs utilize single-cell clocking which has been shown to be unrealistic in [2, 5], the designs can be made realistic through tiling using the technology-independent framework *fiction* [28]. This calls for the development of 3D clocking

[5] It can be observed that the information flow in the arboreally stackable n:1-MUX follows the general form of an upwards traveling pulse wave. Presumably, when physically implementing this structure, it is clockable via a 3D adaptation of the *Columnar* clocking scheme [11], where the respective non-via layers alternate between the two clocking directions, and additional clock generators are added for the via layers to enable upward signal propagation.



(a) 4:1-MUX (root design) consisting of two fused 2:1-MUX in the bottom layer and one 2:1-MUX stacked on top.



(b) Resulting 8:1-MUX consisting of two 4:2-MUX base designs in the bottom layer and one 2:1-MUX root design in the top layer.

Figure 7: The 4:1-MUX root design and the respectively once-arboreally stacked 8:1-MUX.

Table 2: Proposed stackable RCA and MUX design characteristics and comparison against state of the art

		Proposed Stackable Layouts						State of the art		
Name	Bits	x	y	z	V	Cells	nm^2	Ref.	Cells	nm^2
RCA	2	7	7	9	441	61	19044	[30]	85	70924
	4	7	7	17	833	121	19044	[16]	2616	$4.31 \cdot 10^5$
	8	7	7	33	1617	241	19044	[16]	≈ 144000	$4.22 \cdot 10^7$
	16	7	7	65	3185	481	19044	[7]	17278	$2.91 \cdot 10^7$
	⋮									
	1024	7	7	4097	200753	30721	19044	–	–	–
MUX	4:1	7	6	7	294	61	16284	[21]	73	66564
	8:1	15	7	11	1155	165	41124	[15]	293	$5.8 \cdot 10^5$
	16:1	31	11	15	5115	449	134724	–	–	–
	32:1	63	20	19	23940	1359	500684	–	–	–
	⋮									
	1024:1	2047	521	39	$4.16 \cdot 10^7$	896627	$4.26 \cdot 10^8$	–	–	–

x, y, z Volumetric aspect ratio
Cells Number of QCA cells
V Layout volume in cells
nm^2 Layout area in nm^2

schemes, which currently do not exist.[6] Analyses on the absolute kink energy can be performed to assess local stability.

Obtained layouts were simulated and found to behave as expected. The Haskell code as well as the generated design files and example output simulations are made publicly available at https://www.github.com/wlambooy/QCA-STACK.

6 CONCLUSIONS

Established physical design algorithms for QCA circuit layouts almost exclusively consider a 2-dimensional architecture. However, this results in large area requirements due to routing congestion and delay compensation. In this work, two methods were presented that allow realizing regular QCA circuit structures by seamless stacking in the third dimension. For this purpose, corresponding base designs for AND/OR, RCA, and MUX were presented. These do not only implement relevant building blocks for ALUs, but, in the case of MUX, also look-up tables, which implement arbitrary Boolean functions. In fact, the methods and base designs proposed here represent the first steps towards cuboid QCA CPUs and FPGAs.

Furthermore, in comparison to state-of-the-art designs, the proposed approaches consume several orders of magnitude less layout area due to the exploitation of the third dimension. Additionally, circuit bitwidths could be achieved that were not found in the literature before. Thereby, the presented theoretical consideration might inspire future advancement of technological implementations of the QCA concept.

[6] By extending via layers, space is created for vertical clocking electrodes used for propagating information up or down, besides the conventional horizontal ones that drive planar computation.

REFERENCES

[1] N. G. Anderson et al. 2014. *Field-coupled Nanocomputing: Paradigms, Progress, and Perspectives.* Springer, New York.
[2] H. N. Chiu, S. S. H. Ng, J. Retallick, and K. Walus. 2020. PoisSolver: a tool for modelling silicon dangling bond clocking networks. In *IEEE-NANO.* 134–139.
[3] G. Fontes et al. 2018. Placement and Routing by Overlapping and Merging QCA Gates. In *ISCAS.* IEEE.
[4] A. Gin et al. 1999. An alternative geometry for quantum-dot cellular automata. *Journal of Applied Physics* 85, 12 (1999), 8281–8286.
[5] K. Hennessy et al. 2001. Clocking of Molecular Quantum-dot Cellular Automata. *Journal of Vacuum Science & Technology B* 19, 5 (2001), 1752–1755.
[6] J. Huang et al. 2005. Tile-based QCA Design Using Majority-like Logic Primitives. *JETC* 1, 3 (2005), 163–185.
[7] P. Kumar et al. 2019. Optimization of the area efficiency and robustness of a QCA-based reversible full adder. *J. Comput. Electron.* 18, 4 (2019), 1478–1489.
[8] W. Lambooy, C. Kop, and S. Smetsers. 2021. Linearly and Arboreally Stackable Quantum-dot Cellular Automata and Their Discrete Simulation.
[9] C. S. Lent et al. 1993. Lines of Interacting Quantum-dot Cells: A Binary Wire. *Journal of Applied Physics* 74, 10 (1993), 6227–6233.
[10] C. S. Lent et al. 2003. Molecular Quantum-Dot Cellular Automata. *J. Am. Chem. Soc.* 125, 4 (2003), 1056–1063.
[11] C. S. Lent and P. D. Tougaw. 1997. A Device Architecture for Computing with Quantum Dots. *Proc. IEEE* 85, 4 (1997), 541–557.
[12] C. S. Lent, P. D. Tougaw, and W. Porod. 1994. Quantum Cellular Automata: The Physics of Computing with Arrays of Quantum Dot Molecules. In *Workshop on Physics and Computation.* IEEE, 5–13.
[13] C. S. Lent, P. D. Tougaw, W. Porod, and G. H. Bernstein. 1993. Quantum Cellular Automata. *Nanotechnology* 4, 1 (1993), 49.
[14] V. Pudi et al. 2011. Low complexity design of ripple carry and Brent–Kung adders in QCA. *TNANO* 11, 1 (2011), 105–119.
[15] H. Rashidi et al. 2016. High-performance multiplexer architecture for quantum-dot cellular automata. *J. Comput. Electron.* 15, 3 (2016), 968–981.
[16] D. A. Reis et al. 2016. A Methodology for Standard Cell Design for QCA. In *ISCAS.* 2114–2117.
[17] F. Riente et al. 2017. MagCAD: A Tool for the Design of 3D Magnetic Circuits. *JXCDC* 3 (2017), 65–73.
[18] H. R. Roshany et al. 2019. Novel efficient circuit design for multilayer QCA RCA. *Int. J. Theor. Phys.* 58, 6 (2019), 1745–1757.
[19] S. S. Roy. 2017. A Novel Optimized Multiplexer Design in Quantum-Dot Cellular Automata. *Int. J. Res. Appl. Sci. Eng. Technol.* 5, VIII (2017).
[20] A. Safavi et al. 2013. An overview of full adders in QCA technology. *International Journal of Computer Science & Network Solutions* 1, 4 (2013), 12–35.
[21] R. Tiwari et al. 2018. Design and Implementation of 4:1 Multiplexer for Reversible ALU using QCA. In *ICMETE.* 191–196.
[22] P. D. Tougaw et al. 1994. Logical devices implemented using Quantum Cellular Automata. *Journal of Applied Physics* 75, 3 (1994), 1818–1825.
[23] A. Trindade et al. 2016. A Placement and Routing Algorithm for Quantum-dot Cellular Automata. In *SBCCI.*
[24] V. Vankamamidi et al. 2006. Clocking and Cell Placement for QCA. In *IEEE-NANO,* Vol. 1. IEEE, 343–346.
[25] M. Walter et al. 2018. An Exact Method for Design Exploration of Quantum-dot Cellular Automata. In *DATE.* 503–508.
[26] M. Walter et al. 2019. Placement & Routing for Tile-based Field-coupled Nanocomputing Circuits is $\mathcal{NP}$-complete. *JETC* 15, 3 (2019), 10 pages.
[27] M. Walter et al. 2019. Scalable Design for Field-coupled Nanocomputing Circuits. In *ASP-DAC.* 197–202.
[28] M. Walter, R. Wille, F. Sill Torres, D. Große, and R. Drechsler. 2019. fiction: An Open Source Framework for the Design of Field-coupled Nanocomputing Circuits. arXiv:1905.02477
[29] K. Walus et al. 2004. QCADesigner: A Rapid Design and Simulation Tool for Quantum-dot Cellular Automata. *TNANO* 3, 1 (2004), 26–31.
[30] M. Zahmatkesh et al. 2019. Robust coplanar full adder based on novel inverter in quantum cellular automata. *Int. J. Theor. Phys.* 58, 2 (2019), 639–655.