Scalable Physical Design for Silicon Dangling Bond Logic: How a 45° Turn Prevents the Reinvention of the Wheel

Simon Hofmann[1], Marcel Walter[2], and Robert Wille[3]

***Abstract*— With the ever-increasing demands of computing, post-CMOS technologies are sought after. *Field-coupled Nanocomputing* (FCN), which relies on physical field repulsion, is a class of technologies for energy-efficient computing. While the physical design for *Quantum-dot Cellular Automata* (QCA) has been researched for more than 20 years, the methodologies for its promising successor, namely *Silicon Dangling Bonds* (SiDBs), have yet to catch up. To prevent reinventing the wheel and utilizing the 20 years of development in QCA, this paper presents a methodology to create SiDB designs based on existing QCA design approaches by a 45° rotation, implemented as a remapping algorithm. The presented approach enables the direct translation of QCA layouts to SiDB ones with minimal overhead and allows to tap knowledge from decades of research.**

I. Introduction

While the need for computing power is growing rapidly, Moore's Law [1] is reaching its limits. and it is predicted that the information and telecommunications sector could reach 51% of global electricity consumption and 23% of global greenhouse gas emissions by 2030 [2]. *Field-coupled Nanocomputing* (FCN) performs computations without the flow of electric current but through the repulsion of physical fields, making it a possible candidate for the future of green computing at the nanoscale. Much of its success depends on the creation of efficient chip designs that determine the scalability and throughput of the computations that can be performed.

The most extensively researched FCN approach is *Quantum-dot Cellular Automata* (QCA) [3], which was conceptualized in 1993. Due to peculiar constraints imposed by the technology, conventional physical design methodologies are not applicable to the FCN domain. Instead, a variety of different approaches based on heuristics [4], SMT solvers [5], and hand-crafted methods [6] have been proposed by the design automation community in the last decades.

Recently, FCN gained another boost of momentum with the experimental demonstration of a working nanoscale OR gate [7] implemented using *Silicon Dangling Bonds* (SiDBs) [8] on a hydrogen-passivated silicon surface [9], [10]. These SiDBs present another magnitude of scaling improvements compared to, e. g., molecular QCA implementations [11], while offering greater flexibility in their application [7]. Along with recent fabrication accomplishments [8], [10], [12], this led to multi-million dollar investments into research enterprises such as *Quantum Silicon Inc.*

Accordingly, the question is raised again how to efficiently design SiDB layouts, as the plus-shaped QCA gates cannot be exchanged with the Y-shaped SiDB gates due to a size mismatch. A recent approach to automatically generate SiDB circuit layouts from specifications proposed hexagonal tiles for the placement of gates and presented results for small logic networks using an SMT solver [13].

Although these findings represent a significant advancement in comprehending SiDB design automation, they also prompt the inquiry of whether additional research spanning several decades may be necessary until efficient and applicable design automation solutions for a promising FCN technology are available.

In this work, we show that we can avoid this "reinvention of the wheel", by introducing a methodology that adapts findings from QCA algorithms to the SiDB domain with the simple application of a "45° turn". More precisely, to attain equivalent SiDB gate-level layouts, a transformation is employed to convert the Cartesian grids utilized as the foundation of QCA layouts into hexagonal ones as proposed by [13], thus obviating the need for multiple decades of research in the quest for a scalable SiDB physical design algorithm by directly transferring previous and future QCA findings.

Overall, the methodology proposed in this work enables the following contributions:

1) The translation of previous and future QCA layouts to SiDB ones without overhead,
2) the utilization of previously unsupported gate types in QCA physical design methods with the goal of mapping them to SiDB where they are natively supported as elementary tiles, and
3) for the first time, the realization of SiDB layouts with thousands of gates.

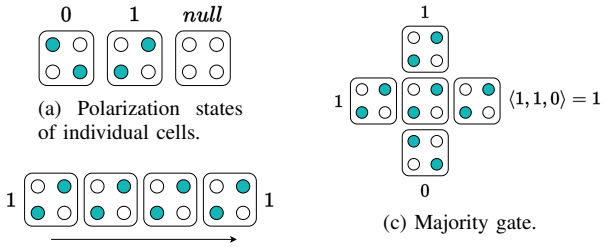
Through the avoidance of laborious efforts required to devise novel design techniques for SiDBs, researchers may concentrate their efforts on enhancing current solutions, thereby facilitating the realization of FCN as a viable post-CMOS technology.

The remainder of this paper is structured as follows: Section II reviews technical background on selected FCN technologies. Section III reviews state-of-the-art design automation methods for QCA and SiDB. To close the gap between the design of two technologies, a physical design algorithm is proposed in Section IV, which is then experimentally evaluated based on common benchmark functions in Section V. Finally, Section VI concludes the paper.

[1]Simon Hofmann is with the Chair for Design Automation, Technical University of Munich, Germany. Email: simon.t.hofmann@tum.de

[2]Marcel Walter is with the Chair for Design Automation, Technical University of Munich, Germany. Email: marcel.walter@tum.de

[3]Robert Wille is with the Chair for Design Automation, Technical University of Munich, Germany, and with the Software Competence Center Hagenberg GmbH, Austria. Email: robert.wille@tum.de

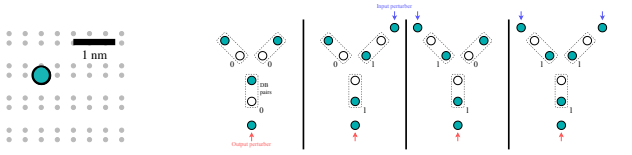


(a) Polarization states of individual cells.

(b) Wire segment.

(c) Majority gate.

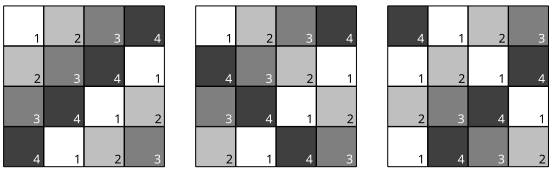
Fig. 1: Elementary QCA cells and compound structures.



(a) H-Si(100)-2×1 surface structure.

(b) Recreation of a binary-dot OR gate [7], adapted from [13].

Fig. 2: SiDBs on a H-Si(100)-2×1 lattice can implement logic gates.



(a) 2DDWave [22].

(b) USE [23].

(c) RES [24].

Fig. 3: Common clocking schemes for FCN technologies.

II. Background

FCN is a promising class of post-CMOS technologies that address the ever-increasing need for computing power and environmental concerns by providing circuits that operate at the nanoscale without the flow of electrical current [14]. This section covers their preliminaries that are required for the comprehension of the remainder of this manuscript. First, Section II-A is concerned with QCA, arguably the most intensively researched FCN technology. Afterward, Section II-B presents an overview of the more recent fabrication breakthroughs achieved with SiDBs.

A. Quantum-dot Cellular Automata

The elementary device in the QCA technology is called a *cell*. In its role, it is comparable with the transistor in conventional electronics. While individual cells can hold a single bit of information in the form of a charge state, multiple cells can be combined to form structures that compute any Boolean function. Thus, QCA (and other FCN technologies) provide logic-in-memory functionality.

A QCA cell possesses four *quantum dots* grouped together in a square frame on a substrate, as shown in Figure 1a. Polarization in the form of electron configurations can encode the binary values 0 and 1 based on the position of these charges. Electrical fields exerted by the polarization of cells influence those placed in proximity and cause their polarization to align accordingly. Thus, information can be propagated and computation conducted. In the simplest case, a line of adjacent QCA cells yields a binary wire segment as depicted in Figure 1b. By placing a cell adjacent to three input cells, the majority-of-three (MAJ3) function can be represented as is shown in Figure 1c.

Based upon these elementary building blocks, complete gate libraries have been envisioned, e. g., *QCA ONE* [15].

B. Silicon Dangling Bonds

SiDBs can be created by resorbing hydrogen atoms from a passivated silicon (H-Si(100)-2×1) surface [9] using a scanning tunneling microscope [8]. An SiDB on a H-Si(100)-2×1 surface is schematically depicted in Figure 2a. This fabrication process yields atomically-sized, chemically identical quantum dots that can be manufactured with unparalleled precision, thanks to recent breakthroughs in the domain [10], [16]–[19].

It has already been experimentally shown that SiDBs can be utilized to implement the FCN concept [7]. Instead of cells with four quantum dots, as in the QCA domain, SiDB uses only two dots in arrangements called *Binary-dot Logic* (BDL) [7]. An SiDB OR gate with a footprint of less than $30\,\mathrm{nm}^2$ was successfully demonstrated using the BDL concept [7]. A recreation of this gate structure with varying inputs is depicted in Figure 2b.

Unlike plus-shaped QCA gates, which input and output information to and from the gate via the top, left, right, and bottom corners, respectively, SiDB are Y-shaped, which results in the inputs being at the top left and right corners and the outputs at the bottom left and right corners, leading to hexagonal tiles being chosen as the optimal gate layout.

Analogously to *QCA ONE*, the *Bestagon* library [13] comprises a set of standard gates, partly designed by a reinforcement learning agent [20]. These gates can be simulated efficiently and accurately using *QuickSim* [21].

C. Technology Constraints

Several constraints imposed by the technology limit the circuit layouts that can be produced for FCN. Most FCN technologies are planar and have limited crossing capabilities, presenting a challenge for wire routing. Furthermore, the lengths of wire segments must be balanced throughout the layout to guarantee signal synchronization. A fundamental requirement for FCN circuits is that they have to be partitioned into uniform regions that are activated and deactivated periodically by external fields, to ensure signal stability and regulate the direction of information flow [25], [26].

This activation mechanism, referred to as *clocking*, is critical for all FCN implementations, as both combinational and sequential circuits must be clocked to maintain signal stability and control information flow direction. In the case of QCA, square tiles are used for clock partitioning, while hexagonal tiles are utilized in SiDB [13], [27].

The default clocking system comprises four consecutive clock signals, numbered from 1 to 4, and supports a pipeline-like flow of information, transmitting signals from tiles under

the control of clock 1 to those under clock 2, clock 3, and, finally, clock 4, before returning to clock 1 [25], [26]. However, this presents challenges for signal propagation and synchronization requiring careful management to ensure that adjacent tiles are clocked consecutively, and that wire lengths are balanced throughout the circuit to prevent delay differences and subsequent desynchronization [28].

The distribution of clock signals to each tile is a topic of significant discussion in the literature, with a general agreement that the signals can be transmitted through buried electrodes in the circuit's substrate. Numerous clocking schemes, featuring a range of regular clock zone arrangements, have been proposed [22]–[24] as illustrated in Figure 3. To support these clocking schemes, various standard gate libraries have been developed providing single-tile implementations of common logic functions and wire segments [13], [15].

III. Design Automation for Field-coupled Nanocomputing Technologies

This section discusses the design challenges and correspondingly proposed approaches in the domain of design automation methods for FCN technologies. Particularly, as before, QCA and SiDB are considered, which are the focal points of this work.

As discussed in the previous section, FCN layouts have unique characteristics that differentiate them from traditional CMOS-based computing systems. In particular, the physical design problems for FCN technologies are especially challenging due to constraints like planarity and signal balancing, as outlined in Section II-C. In compliance with expectations, placement and routing are known to be $\mathcal{NP}$-complete [29], which makes finding optimal solutions difficult even for small circuits. Furthermore, the limited tool support exacerbates the problem. Therefore, the development of efficient and effective design automation techniques for FCN is crucial, especially in the context of SiDB and QCA physical design. It is worth noting that QCA physical design has already been extensively researched for 30 years, while SiDB physical design is relatively new, with the first methods being proposed in the past few years.

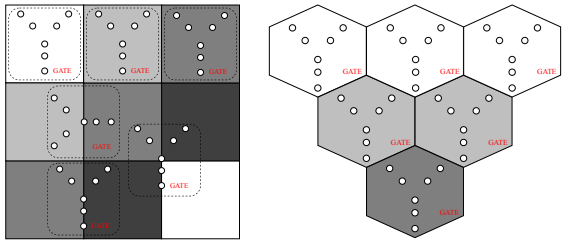
In the remainder of this section, we aim to explore the existing research works on design automation techniques for QCA and SiDB, respectively.

A. Quantum-dot Cellular Automata

Multiple automatic physical design algorithms have been proposed in the literature, which can be classified into two categories: exact and heuristic ones.

Exact physical design algorithms, e. g., [5], [30], obtained layouts from specifications that are optimal with respect to some cost metric (usually layout area). However, they suffer from performance issues due to the $\mathcal{NP}$-completeness of the task, which makes them only applicable to rather small layouts. Heuristic algorithms on the other hand, e. g., [4], [6] are more efficient by imposing restrictions onto the problem, but may not guarantee optimal solutions.

Recently, a new heuristic algorithm based on an approximation to *orthogonal graph drawing* was proposed [4], which



(a) 3 × 3 Cartesian layout. (b) 3 × 3 hexagonal layout.

Fig. 4: (a) When strictly connecting inputs to outputs, Y-shaped SiDB gates do not fit into the structure of Cartesian grids as elementary building blocks. (b) Hexagonal grids can host Y-shaped SiDB gates without modifications.

was able to automatically design layouts with hundreds of millions of tiles by making some restrictions to the search space. This algorithm, denoted as *ortho* in the remainder of this work, can handle complex QCA circuits and achieves better results compared to other heuristic algorithms. However, it is restricted to the *2DDWave* [22] clocking scheme only.

Another approach [31] uses reinforcement learning for gate placement and is able to generate solutions for functions that are unsolvable by the exact approach in a reasonable time, while using less layout area than *ortho*.

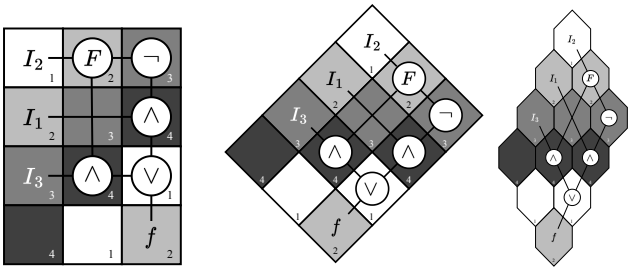
B. Silicon Dangling Bonds

Even though an assortment of automatic physical design algorithms were developed for QCA, they cannot be applied directly to the design of SiDB-based layouts. The naive substitution of QCA with SiDB gates results in a geometric discrepancy on the Cartesian grid due to the mismatch of plus-shaped QCA gates and Y-shaped SiDB ones, as depicted in Figure 4a. The Y-shaped SiDB gates receive input information from two neighboring gates located in northern direction and output it to the south. This arrangement naturally creates a unidirectional information flow from top to bottom only, as seen in Figure 4b, which is not ensured by any of the physical design algorithms discussed in Section III-A.

An initial attempt to adjust QCA-based algorithms to the physical design of SiDBs was proposed in [13], which adjusts the exact algorithm from [5] to consider the hexagonal grid geometry. However, akin to the limitations encountered in acquiring optimal QCA layouts with respect to the layout area, this approach also suffers from the exponential size of the search space and can only be employed for small layouts.

The state-of-the-art scalable physical design automation approach for QCA [4] creates layouts of any size by establishing an orthogonal drawing of the underlying logic network. As the name suggests, this approach works exclusively on Cartesian grid geometries.

To date, there is no scalable approach for SiDB design raising the question whether again decades of research is required to develop specialized design algorithms for the SiDB technology. Due to the rapid pace of development in the field, the design automation community must devise a solution to facilitate further progress.



(a) Multiplexer realized on a Cartesian layout.

(b) Layout obtained by a 45° rotation.

(c) Transformation to hexagonal tiles.

Fig. 5: Rotating a *2DDWave*-clocked layout, naturally creates a *row-wise*-clocked one.

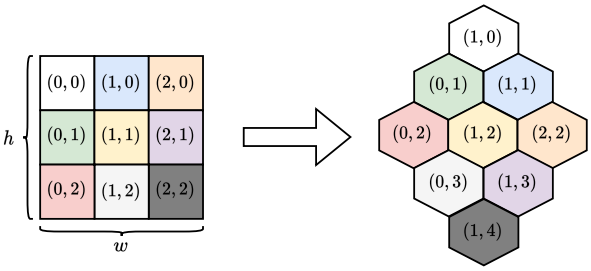


Fig. 6: Each coordinate on the Cartesian grid is transformed to a new coordinate on the hexagonal grid.

IV. Proposed Approach: The 45° Turn

This section constitutes the core contribution of the paper at hand. It outlines a proposal for the adaptation of QCA layouts to the SiDB domain.

The main idea is as follows: Rather than re-engineering physical design methodologies for SiDBs from scratch, we introduce a method that can leverage solutions attained in the realm of QCA, by rotating existing solutions. To comply with technological constraints imposed by the hexagonal grid structure, the indispensable *row-wise* clocking scheme, and the Y-shaped gates, a procedure to convert Cartesian layouts to hexagonal ones is presented, which can then accommodate SiDB-based *Bestagon* gates.

In the remainder of this section, the idea of rotating the layout is presented in detail in Section IV-A. Based on that, the implementation of the proposed method, which maps Cartesian to hexagonal layouts is presented in Section IV-B. The generated layouts can then either be realized with *Bestagon* gates or used to create larger functions, as shown in Section IV-C.

A. Layout Rotation

The main differences between Cartesian layouts and hexagonal ones suitable for SiDBs are as follows:

- Most QCA layouts rely on the *2DDWave* clocking scheme, which allows signal flow to the east and south, as indicated by the arrangement of clock phases in Figure 3a, and imposes the least amount of overhead for combinational functions [30].
- Hexagonal layouts appropriate for SiDBs must be clocked *row-wise* to enable signal propagation solely southward, as suggested by the Y-shape of the *Bestagon* gates.

Transforming the Cartesian layout by rotating it by 45° and enlarging the rectangular grid cells to hexagons overcomes these differences. Figure 5 illustrates this idea using a 3×4 Cartesian layout as an example: Figure 5a shows the realization of a 2:1 multiplexer generated using the exact approach from [5], which is suitable for QCA. In Figure 5b, the same layout is rotated to create the *row-wise* clocking scheme and ensure southern signal flow only. To obtain hexagonal grid cells, the rectangular ones in Figure 5b simply have to be stretched vertically, as demonstrated in Figure 5c.

This closes the gap between QCA and SiDB design. More precisely:

- The transformation converts the *2DDWave* clocked layout into a *row-wise*-clocked layout.
- Irrespective of the previous signal flow direction on the *2DDWave* clocked layout (east and south), the hexagonal layout conveys information southward only, as the hexagonal shapes allow signal flow to the south from their two bottom sides.

These observations lead to a remarkable connection between the physical design of QCA and SiDBs: **existing Cartesian *2DDWave*-clocked QCA layouts can be directly transformed to meet the requirements for placing SiDBs on the hexagonal *row-wise*-clocked layout.**

B. Implementation

The transformation of the Cartesian layout to the hexagonal grid is achieved through a linear mapping of each input, output, gate, and wire to the corresponding location in the hexagonal layout while preserving their connections. Tiles located on a diagonal line in the *2DDWave*-clocked layout belong to the same clock signal, which end up in the same row of the hexagonal layout after the rotation. Based on this correspondence, the proposed approach maps the Cartesian to the hexagonal coordinates.

The y coordinate of a tile on the hexagonal grid y_{hex} is determined by the respective diagonal line on the Cartesian grid, which is calculated by summing the x and y coordinates of the corresponding tile on the Cartesian layout, i. e.:

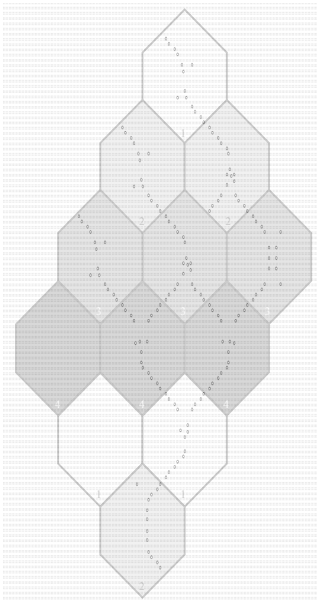
$$y_{hex} = x_{Cart} + y_{Cart} \tag{1}$$

The x coordinate of the hexagonal tile x_{hex} can be calculated using the value of y_{hex} and the height h of the Cartesian layout:

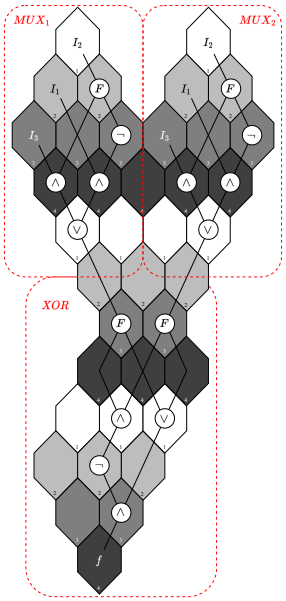
$$x_{hex} = x_{Cart} + \left\lceil \left\lfloor \frac{h}{2} \right\rfloor - \frac{y_{hex}}{2} \right\rceil \tag{2}$$

As an example, the mapping of a 3×3 Cartesian grid to a hexagonal grid is shown in Figure 6. The matching coordinate pairs (x_{Cart}, y_{Cart}) and (x_{hex}, y_{hex}) are tainted with identical colors.

The resulting layout is either visualized using the *Bestagon* libary or combined with other layouts to create composite functions, as presented in Section IV-C.



(a) Each gate is replaced by its corresponding implementation from the *Bestagon* library.



(b) Realization of the exclusive disjunction of the ouputs of two multiplexers.

Fig. 7: Generated hexagonal layouts can either be visualized using the *Bestagon* library or combined with other layouts.

C. Bestagon Gate Library & Function Combination

Once the hexagonal layout is obtained, the gates and wire segments can be replaced by the respective Y-shaped structures from the gate library. An illustration of a hexagonal layout with an SiDB lattice overlay of *Bestagon* gates is shown in Figure 7a. The layout was visualized using SiQAD [32]. Since signals always propagate from top to bottom, the Y-shaped gates can be used directly without any modification.

Alternatively, different layouts can be combined to realize composite functions, as shown in Figure 7b.

V. Experimental Evaluation

The mapping method proposed in this work has been implemented in C++17 and is included in the *fiction*[1] framework [33] as part of the *Munich Nanotech Toolkit* (MNT). Additionally, the mapping algorithm has been made available via *fiction*'s CLI as command `hex`. Having this solution established, results from *any* physical design algorithm for QCA on *2DDWave*-clocked layouts can be mapped to SiDB layouts.

To demonstrate its applicability, we utilized an *exact* approach [5] and *ortho* [4] as representatives for existing algorithms for the design of QCA layouts, mapped their generated layouts from the Cartesian to the hexagonal grid using the proposed methodology, and verified the equivalence of the obtained layouts using the formal verification technique proposed in [34]. Additionally, we compare the result quality against layouts generated by an SiDB-specific design approach available thus far (more precisely, the exact algorithm proposed in [13]). All methods have been evaluated using a broad variety of three well-established benchmark sets [6], [35], [36].

The resulting data is summarized in Table I, which lists the benchmark configurations as well as layout characteristics of the obtained solutions using both, the state-of-the-art SiDB approach and the proposed one. These results clearly confirm the point made in this work: Existing SiDB-based design automation approaches are just at the beginning. Only layouts of up to ≈ 80 tiles can be realized within $24\,\mathrm{h}$, leaving the majority of functions in Table I unsolved. Instead, using the proposed approach, decades of research culminating, e.g., in the scalable design method *ortho* can be utilized. For the first time, this allows for the realization of SiDB layouts for logic functions with thousands of gates.

Overall, this shows that, with a 45° rotation only, *all* functions in the largest benchmark sets ever considered in the FCN domain can be realized with SiDBs on hexagonal layouts, marking a milestone in the development of scalable design methods for this highly promising technology.

VI. Conclusion

Over the last three decades, researchers have developed a toolkit of algorithms for physically designing logic functions on rectangular layouts for Quantum-dot Cellular Automata. However, due to the rapid progress in SiDBs and the mismatch in size between Y-shaped SiDB gates and QCA Cartesian grids, hexagons have emerged as the preferred layout geometry of this technology. Rather than starting from scratch with new design algorithms for SiDBs, the equivalence between the rotated *2DDWave*-clocked Cartesian layout and the *row-wise*-clocked hexagonal layout enables the reuse of sophisticated design algorithms with minimal remapping overhead. The proposed mapping algorithm allows for the rapid creation of hexagonal layouts for functions with thousands of gates in just a few seconds, marking a significant advancement in the automatic design of SiDB layouts.

[1] `https://github.com/cda-tum/fiction`

References

[1] G. E. Moore, "Cramming more components onto integrated circuits," *Electronics*, vol. 38, no. 8, Apr. 1965.

[2] A. Andrae and T. Edler, "On Global Electricity Usage of Communication Technology: Trends to 2030," *Challenges*, vol. 6, pp. 117–157, Apr. 2015.

[3] C. Lent, P. Tougaw, and W. Porod, "Quantum Cellular Automata: The Physics of Computing with Arrays of Quantum Dot Molecules," in *PhysComp*, 1994, pp. 5–13.

[4] M. Walter, R. Wille, F. S. Torres, D. Große, and R. Drechsler, "Scalable Design for Field-Coupled Nanocomputing Circuits," in *ASP-DAC*, 2019, pp. 197–202.

[5] M. Walter, R. Wille, D. Große, F. S. Torres, and R. Drechsler, "An Exact Method for Design Exploration of Quantum-dot Cellular Automata," in *DATE*, 2018, pp. 503–508.

[6] G. Fontes, P. A. R. L. Silva, J. A. M. Nacif, O. P. V. Neto, and R. Ferreira, "Placement and Routing by Overlapping and Merging QCA Gates," in *ISCAS*, 2018, pp. 1–5.

[7] T. Huff *et al.*, "Binary atomic silicon logic," *Nat. Electron.*, vol. 1, no. 12, pp. 636–643, 2018.

[8] R. Achal *et al.*, "Lithography for robust and editable atomic-scale silicon devices and memories," *Nat. Commun.*, vol. 9, no. 1, 2018.

[9] M. B. Haider, J. L. Pitters, G. A. DiLabio, L. Livadaru, J. Y. Mutus, and R. A. Wolkow, "Controlled Coupling and Occupation of Silicon Atomic Quantum Dots at Room Temperature," *Phys. Rev. Lett.*, vol. 102, p. 046805, Jan. 2009.

TABLE I: Comparative experimental evaluation of the state-of-the-art exact and proposed approaches.

Benchmark Circuit					Exact Approach [13]		Proposed Approach				
Benchmark	Name	I / O	$\lvert N\rvert$	$\lvert G\rvert$	$w \times h = A$	t	$w \times h = A$	t_{exact}	t_{ortho}	t_{map}	t_{total}
Fontes18 [6]	1bitAdderMaj	3 / 1	4	8	$3 \times 7 = 21$	< 1	$4 \times 8 = 32$	< 1	—	< 1	< 1
	xor5Maj	5 / 1	4	10	$2 \times 7 = 14$	< 1	$3 \times 6 = 18$	< 1	—	< 1	< 1
	1bitAdderAOIG	3 / 2	6	11	$3 \times 9 = 27$	< 1	$5 \times 9 = 45$	< 1	—	< 1	< 1
	majority	5 / 1	8	14	$3 \times 13 = 39$	< 1	$5 \times 12 = 60$	< 1	—	< 1	< 1
	newtag	8 / 1	9	18	$3 \times 12 = 36$	< 1	$6 \times 12 = 72$	< 1	—	< 1	< 1
	cm82a	5 / 3	16	24	$4 \times 16 = 64$	615	$6 \times 14 = 84$	13	—	< 1	13
	parity	16 / 1	31	48	*timeout limit reached*		$13 \times 48 = 624$	—	< 1	< 1	< 1
ISCAS85 [35]	c432	36 / 7	173	216	*timeout limit reached*		$166 \times 435 = 72\,210$	—	< 1	< 1	< 1
	c499	41 / 32	250	323	*timeout limit reached*		$174 \times 482 = 83\,868$	—	< 1	< 1	< 1
	c880	60 / 26	286	372	*timeout limit reached*		$287 \times 679 = 194\,873$	—	< 1	< 1	< 1
	c1355	41 / 32	282	355	*timeout limit reached*		$174 \times 482 = 83\,868$	—	< 1	< 1	< 1
	c1908	233 / 375	272	880	*timeout limit reached*		$266 \times 607 = 161\,462$	—	< 1	< 1	< 1
	c2670	233 / 64	497	794	*timeout limit reached*		$576 \times 1335 = 768\,960$	—	< 1	< 1	< 1
	c3540	50 / 22	930	1002	*timeout limit reached*		$957 \times 2014 = 1\,927\,398$	—	< 1	< 1	< 1
	c5315	178 / 123	1375	1676	*timeout limit reached*		$1578 \times 3241 = 5\,114\,298$	—	< 1	1	2
	c6288	32 / 32	1424	1488	*timeout limit reached*		$1424 \times 2910 = 4\,143\,840$	—	< 1	< 1	< 1
	c7552	207 / 107	1290	1604	*timeout limit reached*		$1371 \times 3101 = 4\,251\,471$	—	< 1	1	1
EPFL [36]	ctrl	7 / 25	143	175	*timeout limit reached*		$160 \times 349 = 55\,840$	—	< 1	< 1	< 1
	int2float	11 / 7	245	263	*timeout limit reached*		$219 \times 530 = 116\,070$	—	< 1	< 1	< 1
	router	60 / 3	209	272	*timeout limit reached*		$217 \times 531 = 115\,227$	—	< 1	< 1	< 1
	dec	8 / 256	304	568	*timeout limit reached*		$548 \times 1148 = 629\,104$	—	< 1	< 1	< 1
	cavlc	10 / 11	691	712	*timeout limit reached*		$655 \times 1449 = 949\,095$	—	< 1	< 1	< 1
	priority	128 / 17	906	1051	*timeout limit reached*		$985 \times 2169 = 2\,136\,465$	—	< 1	< 1	< 1
	adder	256 / 129	766	1151	*timeout limit reached*		$828 \times 2042 = 1\,690\,776$	—	< 1	< 1	< 1
	i2c	136 / 127	1342	1605	*timeout limit reached*		$1500 \times 3165 = 4\,747\,500$	—	< 1	1	1
	bar	135 / 128	3141	3404	*timeout limit reached*		$2955 \times 6682 = 19\,745\,310$	—	< 1	4	5
	max	512 / 130	2865	3507	*timeout limit reached*		$2418 \times 6905 = 16\,696\,290$	—	< 1	7	8
	sin	24 / 25	4457	4506	*timeout limit reached*		$4323 \times 9198 = 39\,762\,954$	—	1	6	7
	voter	1001 / 1	9633	10 635	*timeout limit reached*		$5601 \times 17147 = 96\,040\,347$	—	6	28	34
	arbiter	256 / 129	11 839	12 224	*timeout limit reached*		$11513 \times 24384 = 280\,732\,992$	—	20	116	136
	square	64 / 126	15 021	15 211	*timeout limit reached*		$15243 \times 30616 = 466\,679\,688$	—	10	76	86

Runtime values are in seconds; the timeout limit is 24 h; w, h and A are the width, height, and resulting area of the layout, respectively; t_{exact}, t_{ortho} and t_{map} indicate the runtime of the exact approach, *ortho* and the proposed mapping algorithm, respectively.

[10] N. Pavliček, Z. Majzik, G. Meyer, and L. Gross, "Tip-induced passivation of dangling bonds on hydrogenated Si(100)-2×1," *APL*, vol. 111, no. 5, p. 053104, 2017.

[11] C. S. Lent, B. Isaksen, and M. Lieberman, "Molecular Quantum-Dot Cellular Automata," *JACS*, vol. 125, no. 4, pp. 1056–1063, 2003.

[12] F. Altincicek, "Atomically Defined Wires on P-Type Silicon," *Bull. Am. Phys. Soc.*, 2022.

[13] M. Walter, S. Ng, K. Walus, and R. Wille, "Hexagons Are the Bestagons: Design Automation for Silicon Dangling Bond Logic," in *DAC*, 2022, pp. 739–744.

[14] N. G. Anderson and S. Bhanja, Eds., *Field-Coupled Nanocomputing - Paradigms, Progress, and Perspectives*. Springer, 2014.

[15] D. A. Reis, C. A. T. Campos, T. R. B. S. Soares, O. P. V. Neto, and F. S. Torres, "A Methodology for Standard Cell Design for QCA," in *ISCAS*, 2016, pp. 2114–2117.

[16] T. Huff *et al.*, "Atomic White-Out: Enabling Atomic Circuitry through Mechanically Induced Bonding of Single Hydrogen Atoms to a Silicon Surface," *ACS nano*, vol. 11 9, pp. 8636–8642, 2017.

[17] J. L. Pitters, I. Dogel, and R. A. Wolkow, "Charge Control of Surface Dangling Bonds Using Nanoscale Schottky Contacts," *ACS nano*, vol. 5, pp. 1984–9, Feb. 2011.

[18] R. A. Wolkow *et al.*, "Silicon Atomic Quantum Dots Enable Beyond-CMOS Electronics," in *Field-Coupled Nanocomputing*, 2013.

[19] M. Rashidi *et al.*, "Initiating and Monitoring the Evolution of Single Electrons Within Atom-Defined Structures," *PRL*, vol. 121, p. 166801, Oct. 2018.

[20] R. Lupoiu, S. S. H. Ng, J. A. Fan, and K. J. Waluś, "Automated Atomic Silicon Quantum Dot Circuit Design via Deep Reinforcement Learning," *ArXiv*, vol. abs/2204.06288, 2022.

[21] J. Drewniok, M. Walter, S. S. H. Ng, K. Walus, and R. Wille, "*QuickSim:* Efficient *and* Accurate Physical Simulation of Silicon Dangling Bond Logic," in *IEEE-NANO*, 2023.

[22] V. Vankamamidi, M. Ottavi, and F. Lombardi, "Clocking and Cell Placement for QCA," in *IEEE-NANO*, vol. 1, 2006, pp. 343–346.

[23] C. Campos, A. Marciano, O. Vilela Neto, and F. Sill Torres, "USE: A Universal, Scalable and Efficient clocking scheme for QCA," *IEEE TCAD*, vol. 35, pp. 513–517, Feb. 2016.

[24] M. Goswami, A. Mondal, M. H. Mahalat, B. Sen, and B. K. Sikdar, "An Efficient Clocking Scheme for Quantum-dot Cellular Automata," *Int. J. Electron. Lett.*, vol. 8, no. 1, pp. 83–96, 2020.

[25] C. Lent and P. Tougaw, "A Device Architecture for Computing with Quantum Dots," *Proc. IEEE*, vol. 85, no. 4, pp. 541–557, 1997.

[26] K. Hennessy and C. S. Lent, "Clocking of Molecular Quantum-dot Cellular Automata," *J. Vac. Sci. Technol. B*, vol. 19, no. 5, pp. 1752–1755, 2001.

[27] J. Huang, M. Momenzadeh, L. Schiano, M. Ottavi, and F. Lombardi, "Tile-based QCA Design Using Majority-like Logic Primitives," *JETC*, vol. 1, no. 3, pp. 163–185, 2005.

[28] F. Sill Torres *et al.*, "Synchronization of Clocked Field-Coupled Circuits," in *IEEE-NANO*, 2018.

[29] M. Walter, R. Wille, D. Große, F. S. Torres, and R. Drechsler, "Placement and Routing for Tile-Based Field-Coupled Nanocomputing Circuits Is NP-Complete (Research Note)," *J. Emerg. Technol. Comput. Syst.*, vol. 15, no. 3, Apr. 2019.

[30] M. Walter, W. Haaswijk, R. Wille, F. S. Torres, and R. Drechsler, "One-pass Synthesis for Field-coupled Nanocomputing Technologies," in *ASP-DAC*, 2021, pp. 574–580.

[31] S. Hofmann, M. Walter, L. Servadei, and R. Wille, "Late Breaking Results From Hybrid Design Automation for Field-coupled Nanotechnologies," in *DAC*, 2023.

[32] S. S. H. Ng *et al.*, "SiQAD: A Design and Simulation Tool for Atomic Silicon Quantum Dot Circuits," *IEEE TNANO*, vol. 19, pp. 137–146, 2020.

[33] M. Walter, R. Wille, F. S. Torres, D. Große, and R. Drechsler, "fiction: An Open Source Framework for the Design of Field-coupled Nanocomputing Circuits," 2019.

[34] ——, "Verification for Field-coupled Nanocomputing Circuits," in *DAC*, 2020, pp. 1–6.

[35] F. Brglez and H. Fujiwara, "A neutral netlist of 10 combinational benchmark circuits and a targeted translator in FORTRAN," in *ISCAS*, June 1985.

[36] L. G. Amarù, P.-E. Gaillardon, and G. D. Micheli, "The EPFL Combinational Benchmark Suite," in *IWLS*, 2015.