Building a Machine Learning Accelerator with Silicon Dangling Bonds: From Verilog to Quantum Dot Layout

Samuel S. H. Ng, Marcel Walter, Jan Drewniok, Simon Hofmann, Robert Wille, and Konrad Walus

Abstract— **At a time when traditional CMOS technologies approach their fundamental scaling limits and artificial intelligence continues to escalate global computational demands, emerging post-CMOS technologies like Silicon Dangling Bonds (SiDBs) provide promising pathways towards energy-efficient computation. SiDBs offer atomic-scale precision and discrete charge control, enabling the realization of ultra-dense computational logic. However, manual layout design and verification have historically restricted the exploration and scalability of SiDB-based logic systems. To this end, this work demonstrates an automated, end-to-end Electronic Design Automation (EDA) flow for designing and synthesizing a core component of a Matrix Multiply Unit (MXU) from high-level Register-transfer Level (RTL) Verilog descriptions down to dot-accurate SiDB layouts. Leveraging recent advances in SiDB-focused EDA tooling, we demonstrate the first fully automated design flow capable of translating RTL descriptions into manufacturable quantum-dot layouts. The proposed hierarchical Verilog approach addresses existing EDA constraints while facilitating comprehensive operational verification via test benches. Additionally, our design process incorporates reliability-focused Figures Of Merit (FoMs), ensuring the selection of robust logic gates throughout synthesis. Our synthesized MXU Processing Element (PE) layout represents a significant milestone in SiDB logic design, bridging previously manually-intensive workflows with scalable, automated methodologies. Despite achieving larger footprints than hand-crafted designs, the presented approach provides a valuable foundation for future optimization and widespread adoption of SiDB-based computing architectures.**

I. Introduction

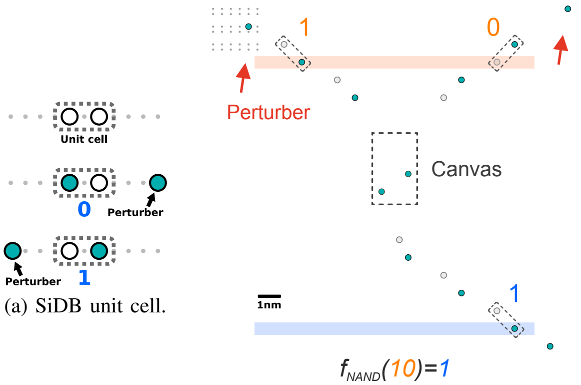
As CMOS scaling faces increasing barriers and challenges, and with global energy demands rapidly escalating due to the widespread adoption of artificial intelligence across diverse application domains, there is renewed urgency to develop highly energy-efficient computing hardware. Field-coupled Nanocomputing (FCN) has emerged as a compelling alternative for post-CMOS logic systems, where logic states encoded in the position of charges in quantum dots [1], [2] or the magnetic polarity of nano-magnets [3], [4] enable computation and signal transmission through local field interactions. Among the various FCN implementation options, Silicon Dangling Bonds (SiDBs) stand out as a promising candidate due to their ability to be fabricated at atomically precise locations [5]–[7] and their discretely controllable charge states [8], [9]. Following a successful experimental demonstration of an SiDB OR gate measuring $5 \times 6\,\mathrm{nm}^2$ [10], Computer-aided Design (CAD) tools supporting different levels of SiDB logic exploration have emerged. On the physical design level, the introduction of *SiQAD* [11] and an ecosystem of specialized simulators [12]–[14] allow users to create and simulate SiDB systems.

The advent of SiDB CAD capabilities has spurred research interest into designing SiDB logic gates and circuits [15]–[17], where designers carefully designed quantum-dot layouts in SiQAD and verify their behavior with SiQAD's physical simulation capabilities [11]–[14]. However, the fully manual design flow of these layouts caused them to be time-consuming ordeals. Fortunately, innovations that introduce automation to multiple levels of the SiDB design flow opened new pathways for large-scale SiDB logic implementation and studies. At the quantum-dot level, automated SiDB gate designers [18], [19] have enabled the creation of standard-tile libraries [20], [21]. The implementation of SiDB support to *fiction* [22], a state-of-the-art Electronic Design Automation (EDA) framework specialized for FCN systems, has enabled higher-level exploration of SiDB applications by automating the synthesis from gate-level netlists—structured descriptions of digital circuits composed of interconnected logic gates—down to dot-accurate SiDB layouts at scales impractical for manual design.

Prior to the availability of SiDB EDA tools, the time-consuming nature and difficulty of scaling up SiDB designs had also limited existing research into large-scale SiDB applications. Among published SiDB applications is a Matrix Multiply Unit (MXU) which is inspired by Google's Tensor Processing Unit (TPU) [23], [24], where the designer used simulation-proven SiDB logic components (up to a half-adder) to extrapolate and approximate the area cost and performance figures of a full SiDB MXU. To move beyond approximations and toward practical implementation, we realize the MXU in Register-transfer Level (RTL) Verilog—a high-level hardware description language commonly used to define digital circuits through clear, structured specifications of data flow and control logic between hardware registers. Leveraging state-of-the-art EDA tools, our RTL implementation can be automatically synthesized into dot-accurate SiDB layouts, enabling a fully automated, scalable, and verifiable end-to-end SiDB design flow—from RTL descriptions, through logic synthesis and optimization, down

This work was supported by the Natural Sciences and Engineering Research Council of Canada under Grant RGPIN-2022-04830.

S. S. H. Ng and K. Walus were with the Department of Electrical and Computer Engineering, University of British Columbia, Vancouver, BC, Canada ({samueln, konradw}@ece.ubc.ca); M. Walter, J. Drewniok, S. Hofmann, and R. Wille were with the Chair for Design Automation, Technical University of Munich, BY, Germany ({marcel.walter, jan.drewniok, simon.t.hofmann, robert.wille}@tum.de).
M. Walter, S. Hofmann, and R. Wille were also with the Munich Quantum Software Company GmbH, Garching near Munich, BY, Germany. R. Wille was also with the Software Competence Center Hagenberg (SCCH) GmbH, Hagenberg, OÖ, Austria.



(a) SiDB unit cell.

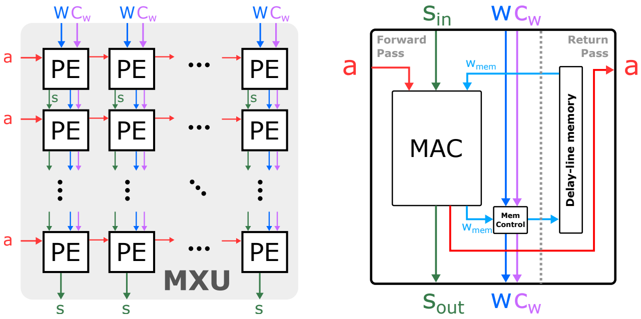
(b) *Bestagon* NAND gate.

Fig. 1. (**a**) A logic unit cell made of a pair of SiDBs sharing a single negative charge [10] simulated in *SiQAD* [11]. **Top:** the unit cell is illustrated without charges for readability; **middle:** unit cell simulated with charge, alongside an additional SiDB placed to the right (dubbed a *perturber*), biasing the unit cell to take on logic state 0; **bottom:** like above but with the perturber moved to the left, biasing the unit cell to take on logic 1. Reprinted from [23] with permission. (**b**) A NAND gate from the *Bestagon* library [20] which defines standard locations for I/O pins and a canvas at the center which allows flexible placement of SiDBs to implement logic gates. Input pins are located at the top with input logic states set by input perturbers—a closer perturber pushes the input wire to the logic 1 position, while a further one allows charges to take on logic 0 state. The output is read out at the output pin located at the bottom. Adapted from [25] with permission.

to quantum-dot layouts. To our knowledge, this represents the first realization of a practically relevant SiDB application design flow validated at every step from RTL description to manufacturable layout, thus setting a crucial benchmark for future SiDB application development and validation. The rest of the manuscript is structured as follows: Section II covers background information on SiDB logic operation and design automation frameworks; Section III presents prior work on the SiDB MXU; Section IV describes our proposed methodology for designing the High-level Synthesis (HLS) Verilog, test bench coverage for operational validation, and workflow for mapping the Verilog description down to dot-accurate layouts; Section V presents the synthesis results, comparisons between different settings and against past work, and discussions about their significance; Section VI concludes the manuscript and discusses potential future work.

II. Background

SiDBs can be manufactured on the surface of hydrogen-passivated Silicon(100)-2×1 by removing each hydrogen atom with the tip of a scanning-tunneling microscope, whereby the application of a current near a hydrogen atom desorbs it from the surface, leaving behind a vacancy [6], [7]; SiDBs can also be erased by reintroducing a hydrogen atom to the vacancy with a functionalized tip [5]. Each SiDB can hold discrete charge states—negative, neutral, or positive—that is determined by bulk doping level and external electrostatic influences [7]. Charges can be shared among SiDBs in close proximity, thus allowing closely-spaced SiDB-pairs to represent binary logic states based on the location of the charge in the SiDB-pair, as illustrated in Fig. 1a. Ensembles of these SiDB-pairs with careful placement have been experimentally proven to implement logic gates, such as the $5 \times 6\,\mathrm{nm}^2$ OR gate which employs two SiDB pairs as inputs and one SiDB pair as output [10]. CAD-assisted explorations have continued to push the exploration of SiDB gate and circuit implementations [11], [15], [17], [26], culminating in the proposal of standard-tile libraries that implement a collection of foundational SiDB gates with standardized input and output pin locations [20]. One such gate from the *Bestagon* gate library is illustrated in Fig. 1b [20]. The advent of these libraries has enabled the incorporation of SiDB support in *fiction* [20], [22], which provides a suite of tools that enable EDA for FCN technologies including technology mapping, placement and routing, exporting technology-specific designs via gate libraries, and more. *fiction*'s capabilities allow this work to synthesize gate-level Verilog descriptions down to quantum-dot level layout descriptions that are supported by *SiQAD* [11], a specialized CAD tool for SiDB logic design and simulation.



(a) MXU systolic array.

(b) PE internal layout.

Fig. 2. (**a**) A systolic array MXU taking in quantized activations (a), weights (w), weight control signals (C_w), and partial sums (s). Adapted from [23] with permission. (**b**) Layout within a PE showing key components: MAC arithmetic unit, memory controller, and delay-line memory for weight storage. The signal paths are also illustrated, showing that signals propagate downwards on the left side of the PE and upwards on the right side, the two sides are thus dubbed the forward- and return-passes.

III. Related Work

Out of the scarce selection of application-scale SiDB implementations, the proposed blueprint of an SiDB MXU [23] intended to accelerate matrix-multiply operations in machine learning applications forms the basis for this work. The implementation took inspiration from key design choices from Google's TPUv1 [24], including the use of 8-bit quantized integer arithmetic in lieu of floating point operations to reduce compute costs, and a systolic array architecture for efficient data flow which works well with matrix multiplication. Systolic arrays consist of regular arrangements of PEs—compute modules that share the same design, arranged in a way to operate on its inputs and pass the results onto the next PE. In the blueprint, each PE performs the MAC operation which can be written as [23]:

$$s_{\text{out}} = s_{\text{in}} + (a \times w), \tag{1}$$

where s_{in} is the input partial sum computed by a prior PE, a is the activation, w is the weight, and s_{out} is the output partial sum. Fig. 2a illustrates the systolic array layout of the SiDB MXU and associated high-level data flow.

The MXU has two modes of operation: 1) preloading, where weights (w) are loaded into the systolic array from the top until they reach their destined PEs, upon which they're stored in the corresponding PE's memory; and 2) computing, where activations (a) are loaded into the systolic array from the side for the MAC operation to be carried out in the PEs. Activations continue traversing rightward, multiplying with subsequent weights, while partial sums (s) traverse downward to be summed with the next $a \times w$ product.

At a more fine-grained level, each PE must be able to determine the operation mode based on input flags, and perform MAC computation when operating in the computing mode. Key components in each PE, including the MAC unit and the memory controller, can be designed and arranged such that most operations are performed combinationally with unidirectional signal flow; some signals must subsequently flow in the reverse direction to complete the PE. Fig. 2b illustrates this arrangement of components in the PE, with all of the logical operations performed on the left side of the PE when signals flow from top to bottom, while reverse signals on the right side allow activations to be passed to the neighboring tile and the stored weight in the delay-line memory to be passed back to the MAC's input. We dub the left side the *forward-pass* and the right side the *return-pass* in reference to the direction of signal propagation. The implementation is deeply pipelined, a trait that is common among FCN technologies in consideration of operational reliability and in order to facilitate signal propagation. This means that, at each clock cycle, different inputs can be interleaved to increase the computational throughput of the MXU. The blueprint used logic components which were proven in simulation as the basis to approximate the dimensions of larger components [23], which includes various foundational two-input, one-output logic gates and a half-adder introduced in [11] without the use of standard-tile libraries.

The prior work reported up to 10× improvement in area efficiency and up to $10^8\times$ in power efficiency when compared against Google's TPUv1 [23], [24]. However, the approximated nature of the proposal means that it offers neither an exact SiDB layout that can be manufactured and tested, nor any verification in operational correctness.

To this end, our work harvests latest FCN EDA techniques to bridge the gap in the verification of logic designs and the lack of accurate physical layouts in SiDB applications by implementing the PE in Verilog, verifying fully pipelined interleaved operations with test benches, and demonstrating a design flow that synthesizes the description to dot-accurate SiDB layouts, as detailed in the following section.

IV. Methodology

In this section, we describe the full methodology at the core of this work. Starting with Section IV-A, we discuss our hierarchical Verilog development approach which satisfies 1) the restrictions on input netlists imposed by FCN EDA tools and 2) our desire to verify the operational correctness of the SiDB MXU; then in Section IV-B, we present the workflow employed by this work to map the RTL Verilog all the way down to synthesized quantum-dot layouts.

Algorithm 1 Processing Element's Forward Pass Logic

```
Inputs:
    $PE_y \leftarrow$ y-index assigned to each PE (constant per PE)
    $m \leftarrow$ PRELOAD (0) or COMPUTE (1) mode
    $w_{\text{load}} \leftarrow$ weight to be loaded into memory
    $PE_{\text{target-}y} \leftarrow$ target Y-index of $w_{\text{load}}$
    $s_{\text{in}} \leftarrow$ input partial product
    $a \leftarrow$ input activation
    $w_{\text{mem}} \leftarrow$ weight loaded from delay-line memory
Outputs:
    $s_{\text{out}}$: new partial sum
    $w_{\text{mem-out}}$: weight to write to delay-line memory
procedure ProcessingElementLogic
    $p \leftarrow \text{signed}(w_{\text{mem}}) \times \text{signed}(a)$        ▷ Multiply stored $w$ with $a$
    $p_{\text{signExtended}} \leftarrow \text{signExtend}(p, 24)$
    $s_{\text{out}} \leftarrow s_{\text{in}} + p_{\text{signExtended}}$        ▷ Sum with partial sum
    if $m =$ PRELOAD and $PE_{\text{target-}y} = PE_y$ then
        $w_{\text{mem-out}} \leftarrow w_{\text{load}}$        ▷ Update stored weight
    end if
    return $s_{\text{out}}$, $w_{\text{mem-out}}$, and other pass-through wires
end procedure
```

A. Matrix Multiply Unit In Hierarchical Verilog

We approach the Verilog implementation of the MXU with the following requirements in mind:

1) The Verilog implementation must obey the constraints of *fiction*, which expects combinational gate-level netlists as inputs, with no feedback loops or registers. Signal flow in the placed and routed SiDB layouts would also be unidirectional.
2) Operation of the pipelined processing element must be verifiable—whereas prior work on the SiDB MXU [23] did not include formal verifications of the MXU's operation, this work has the opportunity to verify our Verilog implementation with test bench coverage.

As described in Section III, each PE in the SiDB MXU includes a forward pass where most of the core logic takes place, and a return pass which is required for signal synchronization and the operation of the delay-line memory. However, this creates a feedback loop which violates *fiction*'s input expectations. To work around this, our RTL Verilog implementation is split hierarchically. The core logic in the forward pass is implemented as a combinational Verilog description and shown in Alg. 1. Operation of the full PE is then captured by a higher level RTL Verilog description that defines the component's input and output signals, pipelined internal signal steering, and synchronized signal timing for the return pass.

Test benches can thus be written for both levels of Verilog descriptions. At the lower level, test benches have been implemented to verify correct MAC operation with signed numbers, including edge cases like inputting large signed integers that verify proper operation at the limit of the 8-bit quantized implementation. At the full PE level, test benches have been written to verify operation in preloading and compute modes. Since it is expected that the final SiDB implementation would be deeply pipelined, the test benches also verify pipelined operation by presenting interleaved inputs across multiple clock cycles to verify the PE's ability to handle multiple sets of weights and activations simultaneously going through different pipeline stages. Simulations are performed using Icarus Verilog [27]. These test benches ensure that the PE design is logically correct before proceeding to synthesis and physical mapping. You can find the full open-source Verilog implementation on GitHub [28].

B. Design Automation Workflow

After verifying the Verilog description, the RTL design must be synthesized into a gate-level netlist in order for *fiction* to process the design. We use Yosys [29] to synthesize the RTL Verilog description into an And-Inverter Graph (AIG) netlist and ABC's *&deepsyn* [30] optimization strategy to reduce the network's node count. The resulting gate-level Verilog is then provided to *fiction* [22] for synthesis to an SiDB layout using its FCN-specific EDA tooling in the following multi-step process:

1) Technology mapping is conducted to map the AIG into gates that are available in the *Bestagon* gate library [20], which further reduces the node count via the expressive power of a full gate library over AND/INV primitives.
2) Placement and routing with the *ortho* algorithm [31] produces a layout on a Cartesian grid representing the full logic implementation with valid gate placements that are connected by wires.
3) Post-layout optimizations are run to reduce the footprint of the layout [32], [33].
4) The Cartesian layout is then remapped onto a hexagonal grid [34] in preparation for applying the *Bestagon* library.
5) SAT-based equivalence checking [35] is performed to verify that the mapped, optimized, and hexagonalized layout faithfully implements the gate-level Verilog specification.
6) The *Bestagon* library [20] is applied to the hexagonalized layout, yielding a dot-accurate SiDB implementation of the design.

As part of this investigation, we have also contributed various improvements to *fiction* to further optimize the EDA tools for SiDB workflows. Recent works on SiDB logic robustness have culminated into the proposal of unified Figures Of Merit (FoMs) for SiDB gates [36], which comprehensively capture currently known cost functions that can impact the robustness of SiDB gates: operational domain size [11], [37], thermal resilience [38], band bending [9], defect [39]–[41], and more. The scores achieved by each metric are weighted and averaged to obtain χ, a FoM assigned to the gate. Our gate library uses χ as the cost function for the technology mapper that will thus favor the most stable gates even if a certain logic could have been implemented with fewer, albeit less stable, selection of gates. The impact of this in the final layout will be compared in Section V.

We have also improved the hexagonalization step to better align with SiDB layout expectations. The Cartesian layout prior to hexagonalization employs the 2DDWave clocking scheme [42] where all input pins are placed along the top and left borders, and signals subsequently flow diagonally from top-left to bottom-right. In prior implementations of hexagonalization, the Cartesian layout is then rotated by 45° and projected onto a hexagon grid [34] such that signals flow top-down. As a result, input pins on the hexagonal layout tend to have a y offset from the top. This can negatively impact the theoretical throughput of the layout as some input pins might have to be held at the same value over multiple clock cycles in order to synchronize with other inputs. These constraints can be alleviated by extending all input pins to the top edge of the layout; we have thus updated the hexagonalization algorithm to perform this extra step which yields fully synchronized circuit layouts.

V. Results and Discussion

Following the EDA workflow laid out in Section IV-B, we have successfully synthesized the RTL Verilog description of the forward-pass PE all the way down to a dot-accurate SiDB layout. We present the results in Table I, which includes: the network dimension in terms of tile count, the physical dimension in nm^2, as well as the counts of gates, wires, crossings, and SiDBs used in the layout. Note that in this table, every tile that is mapped to a wire in the final layout is counted as 1 wire, thus a single wire signal that spans across 10 tiles would increment the value by 10. We can see that the layout resulting from FoM-informed technology mapping is slightly larger than the simpler alternative of treating all gates with equal preference, indicating that the technology mapper has indeed chosen costlier solutions that make use of more robust gates. This area trade-off comes with the benefit that SiDB gates with higher reliability metrics are more often employed in the layout, which can ultimately yield a more robust device. It is also important to note that the FoM values from [36] are derived from a specific variant of the *Bestagon* library optimized under a particular set of physical conditions. Choosing different libraries or physical conditions may yield different trade-offs. Also notable is that the original hexagonalization algorithm would have imposed a $\frac{1}{12}\times$ throughput limit compared to full throughput, a limitation that is now alleviated by extending the input pin wires (as discussed in Section IV-B) at no increase in area cost.

Although the synthesized SiDB layout only covers the forward pass of the PE's operation, it already represents all of the logic operations; the parts that are not included in the synthesized layout are purely for signal propagation, as described in Section IV-A. We thus believe that there is value in comparing our results with the previously proposed blueprint [23]. The blueprint reported an area footprint of $5000 \times 8150\,\mathrm{nm}^2$, which means that the area we've achieved is $\sim 30\times$ higher than those estimated by the blueprint, and

TABLE I

PLACED AND ROUTED MXU PROCESSING ELEMENT (FORWARD PASS)

Experiment	Tiles ($w \times h$)	Physical dimensions (nm^2)	Gates	Wires	Crossings	SiDBs
All gates equally preferred	515×1043	30,930 × 35,474	613	120,945	11,880	1,645,777
FoM-informed	516×1049	30,990 × 35,678	607	124,048	11,057	1,685,608

the gap would further widen if we were to synthesize the entire PE. Why is there such a substantial increase in area compared to previous estimates?

We believe there are multiple contributing factors. Whereas the previously proposed blueprint used hand-designed components with fairly high logic gate density [11], [23], this work uses the *Bestagon* gate library which deliberately chose a large tile template to ensure sufficient separation between the logic canvas of neighboring logic tiles in order to minimize inter-gate interference [20]. Furthermore, the blueprint study benefited from deliberately chosen multiplier and adder implementations that were, themselves, systolic array designs optimized for FCN implementation. In this study, the choice of arithmetic unit implementation is completely left to Yosys and ABC's synthesis tools, which are optimized for traditional CMOS devices with very different architectural trade-offs. Furthermore, once we perform logic optimization with ABC, the boundaries of arithmetic elements get completely blurred, hindering the possibility of potential optimization algorithms that could benefit from such arithmetic knowledge. Lastly, due to the large size of the netlist required by this application (in relation to other FCN applications), it is computationally intractable to use the optimal placement and routing algorithms offered by *fiction* such as the *exact* [43] and *gold* [44] solvers. Instead, we had to rely on algorithms which prioritized runtime at the sacrifice of final area cost. Nevertheless, the successful end-to-end design flow achieved by this study presents an encouraging demonstration of what state-of-the-art FCN EDA tools can achieve on the SiDB platform, highlighting opportunities for further optimization.

VI. CONCLUSION AND FUTURE WORK

In past studies, the design and verification of large-scale SiDB applications remained disconnected from their physical implementations, constrained by immature EDA tooling and manually-intensive workflows at the time [23], [45]. Enabled by recent advances in SiDB-focused EDA frameworks, this work presents the first automated, end-to-end design flow for implementing a processing element (PE) of a SiDB-based matrix multiply unit designed specifically for machine learning acceleration. Our hierarchical Verilog structure facilitates the synthesis of combinational components down to the SiDB level while enabling validation of the complete operational functionality through higher-level RTL modules. By strategically separating core combinational logic from higher-level pipeline control, we successfully synthesized a majority of the PE's functionality into a dot-accurate SiDB layout, selectively omitting specific signal components from synthesis to accommodate EDA tooling restrictions. Although the synthesized layout exhibits a larger area compared to previous hand-crafted designs [23], our effort provides a concrete demonstration of automated design methodologies applied to SiDB-based circuits, and underscores opportunities for further optimizations in SiDB-specific EDA flows. Additionally, this study is the first to incorporate FoM considerations into *fiction*'s technology mapping process, prioritizing gate reliability within synthesized layouts.

One limitation faced by this study is the lack of control over the implementation of fundamental arithmetic units such as adders and multipliers. Since the design trade-offs of SiDB logic differ significantly from CMOS designs, it is possible that our workflow for synthesizing RTL Verilog to gate-level Verilog using established tools like Yosys and ABC is introducing unnecessary overhead due suboptimal implementation choices. Further improvements to this synthesis step, as well as further developments to *fiction*'s placement and routing algorithms, can bridge the gap between synthesized layouts and expert-designs and incentivize further development of novel applications optimized for SiDBs.

REFERENCES

[1] C. S. Lent, B. Isaksen, and M. Lieberman, "Molecular quantum-dot cellular automata," *Journal of the American Chemical Society*, vol. 125, no. 4, pp. 1056–1063, 2003.

[2] Y. Ardesi, F. Ravera, G. Piccinini, and M. Graziano, "Modeling Molecules for Field-Coupled Nanocomputing Circuit Design," in *2024 IEEE 24th International Conference on Nanotechnology (NANO)*. Gijon, Spain: IEEE, Jul. 2024, pp. 430–435.

[3] G. Bernstein *et al.*, "Magnetic QCA systems," *Microelectronics Journal*, vol. 36, no. 7, pp. 619–624, 2005.

[4] D. Giri, M. Vacca, G. Causapruno, M. Zamboni, and M. Graziano, "Modeling, Design, and Analysis of MagnetoElastic NML Circuits," *IEEE Transactions on Nanotechnology*, vol. 15, no. 6, pp. 977–985, Nov. 2016.

[5] T. R. 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, no. 9, pp. 8636–8642, Sep. 2017.

[6] R. Achal *et al.*, "Lithography for robust and editable atomic-scale silicon devices and memories," *Nature Communications*, vol. 9, no. 1, p. 2778, Jul. 2018.

[7] J. Pitters *et al.*, "Atomically Precise Manufacturing of Silicon Electronics," *ACS Nano*, p. acsnano.3c10412, Feb. 2024.

[8] 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," *Physical Review Letters*, vol. 102, no. 4, p. 046805, Jan. 2009.

[9] J. L. Pitters, I. A. Dogel, and R. A. Wolkow, "Charge control of surface dangling bonds using nanoscale Schottky contacts," *ACS Nano*, vol. 5, no. 3, pp. 1984–1989, Mar. 2011.

[10] T. Huff *et al.*, "Binary atomic silicon logic," *Nature Electronics*, vol. 1, no. 12, pp. 636–643, Dec. 2018.

[11] S. S. H. Ng *et al.*, "SiQAD: A design and simulation tool for atomic silicon quantum dot circuits," *IEEE Transactions on Nanotechnology*, vol. 19, pp. 137–146, 2020.

[12] H. N. Chiu, S. S. H. Ng, J. Retallick, and K. Walus, "PoisSolver: A tool for modelling silicon dangling bond clocking networks," in *2020 IEEE 20th International Conference on Nanotechnology (IEEE-NANO)*. Montreal, QC, Canada: IEEE, Jul. 2020, pp. 134–139.

[13] 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 International Conference on Nanotechnology (IEEE NANO)*, 2023, pp. 817–822.

[14] J. Drewniok, M. Walter, and R. Wille, "The Need for Speed: Efficient Exact Simulation of Silicon Dangling Bond Logic," in *Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2024.

[15] M. D. Vieira *et al.*, "Three-Input NPN Class Gate Library for Atomic Silicon Quantum Dots," *IEEE Design & Test*, pp. 1–1, 2022.

[16] A. N. Bahar and K. A. Wahid, "Design of QCA-Serial parallel multiplier (QSPM) with energy dissipation analysis," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 67, no. 10, pp. 1939–1943, 2020.

[17] S.-S. Ahmadpour, N. J. Navimipour, A. N. Bahar, M. Mosleh, and S. Yalcin, "An Energy-Aware Nano-Scale Design of Reversible Atomic Silicon based on Miller Algorithm," *IEEE Design & Test*, pp. 1–1, 2023.

[18] R. Lupoiu, S. S. H. Ng, J. A. Fan, and K. Walus, "Automated Atomic Silicon Quantum Dot Circuit Design via Deep Reinforcement Learning," Apr. 2022.

[19] J. Drewniok, M. Walter, and R. Wille, "Minimal Design of SiDB Gates: An Optimal Basis for Circuits Based on Silicon Dangling Bonds," in *Proceedings of the 18th ACM International Symposium on Nanoscale Architectures*. Dresden Germany: ACM, Dec. 2023, pp. 1–6.

[20] M. Walter, S. S. H. Ng, K. Walus, and R. Wille, "Hexagons are the Bestagons: Design automation for silicon dangling bond logic," in *Proceedings of the 59th ACM/IEEE Design Automation Conference*, ser. DAC '22. San Francisco, CA: Association for Computing Machinery, 2022, p. 6.

[21] S. S. H. Ng, J. Drewniok, M. Walter, J. Retallick, R. Wille, and K. Walus, "Unlocking Flexible Silicon Dangling Bond Logic Designs on Alternative Silicon Orientations," in *2024 IEEE 24th International Conference on Nanotechnology (NANO)*. Gijon, Spain: IEEE, Jul. 2024, pp. 57–62.

[22] M. Walter, R. Wille, F. Sill, D. Große, and R. Drechsler, "Fiction: An open source framework for the design of field-coupled nanocomputing circuits," *ArXiv*, vol. abs/1905.02477, 2019.

[23] S. S. H. Ng, H. N. Chiu, J. Retallick, and K. Walus, "A Blueprint for Machine Learning Accelerators Using Silicon Dangling Bonds," in *2023 IEEE 23rd International Conference on Nanotechnology (NANO)*. Jeju City, Korea, Republic of: IEEE, Jul. 2023, pp. 1–6.

[24] N. P. Jouppi *et al.*, "In-datacenter performance analysis of a tensor processing unit," 2017. [Online]. Available: https://arxiv.org/abs/1704.04760

[25] J. Drewniok, M. Walter, and R. Wille, "QuickTrace: An efficient contour tracing algorithm for defect robustness simulation of silicon dangling bond logic," in *Proceedings of the 2025 IEEE International Symposium on Circuits and Systems (ISCAS)*. London, United Kingdom: IEEE, May 2025.

[26] A. N. Bahar, K. A. Wahid, S. S. Ahmadpour, and M. Mosleh, "Atomic silicon quantum dot: A new designing paradigm of an atomic logic circuit," *IEEE Transactions on Nanotechnology*, vol. 19, pp. 807–810, 2020.

[27] Icarus Verilog GitHub repository. [Online]. Available: https://github.com/steveicarus/iverilog

[28] S. S. H. Ng, M. Walter, J. Drewniok, S. Hofmann, R. Wille, and K. Walus. Verilog implementation of a matrix-multiply unit for silicon dangling bonds. [Online]. Available: https://github.com/samuelshng/sidb-mxu-verilog

[29] C. Wolf, J. Glaser, and J. Kepler, "Yosys-a free verilog synthesis suite," 2013. [Online]. Available: https://api.semanticscholar.org/CorpusID:202611483

[30] R. Brayton and A. Mishchenko, "ABC: an academic industrial-strength verification tool," in *Proceedings of the 22nd International Conference on Computer Aided Verification*, ser. CAV'10. Berlin, Heidelberg: Springer-Verlag, 2010, p. 24–40. [Online]. Available: https://doi.org/10.1007/978-3-642-14295-6_5

[31] M. Walter, R. Wille, F. S. Torres, D. Große, and R. Drechsler, "Scalable design for field-coupled nanocomputing circuits," in *Proceedings of the 24th Asia and South Pacific Design Automation Conference*, ser. ASPDAC '19. New York, NY, USA: Association for Computing Machinery, 2019, pp. 197–202.

[32] S. T. Hofmann, M. Walter, and R. Wille, "Post-Layout Optimization for Field-coupled Nanotechnologies," in *Proceedings of the 18th ACM International Symposium on Nanoscale Architectures*. Dresden Germany: ACM, Dec. 2023, pp. 1–6.

[33] ——, "Late Breaking Results: Wiring Reduction for Field-coupled Nanotechnologies," in *Proceedings of the 61st ACM/IEEE Design Automation Conference*. San Francisco CA USA: ACM, Jun. 2024, pp. 1–2.

[34] S. Hofmann, M. Walter, and R. Wille, "Scalable Physical Design for Silicon Dangling Bond Logic: How a 45° Turn Prevents the Reinvention of the Wheel," in *2023 IEEE 23rd International Conference on Nanotechnology (NANO)*. Jeju City, Korea, Republic of: IEEE, Jul. 2023, pp. 872–877.

[35] M. Walter, R. Wille, F. S. Torres, D. Große, and R. Drechsler, "Verification for field-coupled nanocomputing circuits," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*, 2020, pp. 1–6.

[36] J. Drewniok, M. Walter, S. S. H. Ng, K. Walus, and R. Wille, "Unifying Figures of Merit: A Versatile Cost Function for Silicon Dangling Bond Logic," in *2024 IEEE 24th International Conference on Nanotechnology (NANO)*. Gijon, Spain: IEEE, Jul. 2024, pp. 91–96.

[37] M. Walter, J. Drewniok, S. S. H. Ng, K. Walus, and R. Wille, "Reducing the Complexity of Operational Domain Computation in Silicon Dangling Bond Logic," in *International Symposium on Nanoscale Architectures (NANOARCH)*, 2023.

[38] J. Drewniok, M. Walter, and R. Wille, "Temperature Behavior of Silicon Dangling Bond Logic," in *IEEE International Conference on Nanotechnology (IEEE NANO)*, 2023, pp. 925–930.

[39] S. S. H. Ng, J. Croshaw, M. Walter, R. Wille, R. Wolkow, and K. Walus, "Simulating Charged Defects in Silicon Dangling Bond Logic Systems to Evaluate Logic Robustness," *IEEE Transactions on Nanotechnology*, vol. 23, pp. 231–237, 2024.

[40] T. R. Huff *et al.*, "Electrostatic landscape of a Hydrogen-terminated silicon surface probed by a moveable quantum dot," *ACS Nano*, vol. 13, no. 9, pp. 10 566–10 575, Sep. 2019.

[41] J. Croshaw, T. Dienel, T. Huff, and R. Wolkow, "Atomic defect classification of the H–Si(100) surface through multi-mode scanning probe microscopy," *Beilstein Journal of Nanotechnology*, vol. 11, pp. 1346–1360, Sep. 2020.

[42] V. Vankamamidi, M. Ottavi, and F. Lombardi, "Two-dimensional schemes for Clocking/Timing of QCA circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 1, pp. 34–44, 2008.

[43] M. Walter, R. Wille, D. Große, F. S. Torres, and R. Drechsler, "An exact method for design exploration of quantum-dot cellular automata," in *2018 Design, Automation Test in Europe Conference Exhibition (DATE)*, 2018, pp. 503–508.

[44] S. Hofmann, M. Walter, and R. Wille, "A* is Born: Efficient and Scalable Physical Design for Field-coupled Nanocomputing," in *2024 IEEE 24th International Conference on Nanotechnology (NANO)*. Gijon, Spain: IEEE, Jul. 2024, pp. 80–85.

[45] H. N. Chiu, "Simulation and analysis of clocking and control for field-coupled quantum-dot nanostructures," Master's thesis, University of British Columbia, 2020.