RTL-to-Atoms Synthesis of a Machine Learning Accelerator on Atomic-Scale Computers

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

***Abstract*—As CMOS scaling slows and AI demand soars, atomic-scale platforms such as quantum-dot logic based on silicon dangling bonds (SiDBs) offer a promising path toward energy-efficient computation, yet practical design flows from register-transfer level (RTL) specifications to manufacturable layouts remain limited. This work presents an RTL-to-atoms synthesis framework for a quantized matrix multiply unit targeting SiDB-based field-coupled nanocomputing (FCN). Building on recent advances in SiDB-aware electronic design automation (EDA), the framework combines a hierarchical, parameterized RTL architecture with platform-optimized arithmetic logic units, reducing the synthesized logic core of processing elements by about 15 % compared to prior flows. Key improvements were also made to the synthesis workflow to better optimize for SiDB logic and incorporate figure-of-merit awareness, which together ensure that the synthesized layouts achieve favorable area scaling and throughput while balancing operational robustness. Evaluations across multiple bit-widths show substantial footprint reductions for configurations within the tractable range of the latest placement-and-routing algorithms while preserving testbench-validated correctness from RTL to dot-accurate SiDB layouts, thereby establishing a reproducible benchmark for EDA on atomic-scale computing. This represents a significant milestone, bridging manually intensive workflows with scalable, automated methodologies, providing a valuable foundation for future design efforts for SiDB-based accelerators.**

***Index Terms*—Electronic design automation, computer-aided design, machine learning, quantum dots, silicon dangling bonds**

I. Introduction

GLOBAL computing infrastructure increasingly devotes its resources to AI inference and training, making the associated energy demand a central constraint and spurring efforts to develop hardware platforms that operate far more efficiently than traditional CMOS processors. Among these, field-coupled nanocomputing (FCN) has emerged as an appealing post-CMOS computing paradigm, in which local field interactions encode logic states in the position of charges [1] or in the magnetic polarity of nanomagnets [2], [3], enabling both computation and signal transmission.

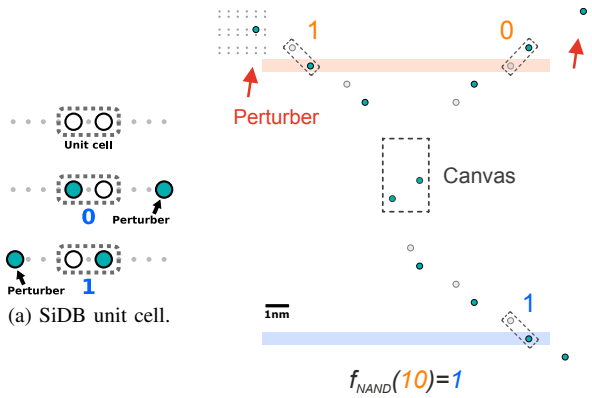
This work was supported by the Natural Sciences and Engineering Research Council of Canada under Grant RGPIN-2022-04830.

S. Ng and K. Walus were with the Department of Electrical and Computer Engineering, University of British Columbia, Canada ({samueln, konradw}@ece.ubc.ca); M. Walter, S. Hofmann, J. Drewniok, and R. Wille were with the Chair for Design Automation, Technical University of Munich, Germany ({marcel.walter, simon.t.hofmann, jan.drewniok, robert.wille}@tum.de). M. Walter, S. Hofmann, and R. Wille were also with the Munich Quantum Software Company GmbH, Germany. R. Wille was also with the Software Competence Center Hagenberg (SCCH) GmbH, Austria.
Copyright (c) 2026 IEEE. Personal use of this material is permitted. However, permission to use this material for any other purposes must be obtained from the IEEE by sending a request to pubs-permissions@ieee.org.

Among the various physical implementations of FCN, quantum dots made of silicon dangling bonds (SiDBs) stand out as a promising platform due to their atomically precise fabrication [4], [5] and discretely controllable charge states [6]. Following the experimental demonstration of compact SiDB logic gates [7], a supporting ecosystem of CAD [8], simulation [9]–[11], and electronic design automation (EDA) [12] tools emerged. These advances opened a path from gate-level logic to manufacturable SiDB layouts and made automated application-scale exploration increasingly feasible.

Prior to the maturity of SiDB EDA tools, application-scale research targeting SiDBs was scarce and relied on manually verified SiDB building blocks, which were then extrapolated into system-level designs to estimate their behavior and implementation costs [9], [13]. Among them was a quantized matrix multiply unit (MXU) inspired by Google's tensor processing unit (TPU) [14] and optimized for SiDB's architectural constraints [13], where simulation-proven SiDB logic components were used to approximate the area cost and performance figures. To facilitate practical exploration, an earlier conference version of this work [15] implemented the MXU as a register-transfer level (RTL) Verilog design, employing clocked registers as an abstract proxy for the underlying FCN clocking behavior. With a workflow spanning multiple EDA frameworks, the study synthesized the RTL implementation into dot-accurate SiDB layouts, realizing an automated, scalable, and verifiable end-to-end SiDB design flow using state-of-the-art EDA tools.

Despite this progress, the conference version of the RTL-to-atoms flow in [15] left several aspects only coarsely explored: the choice of arithmetic logic units (ALUs) was left to logic synthesis frameworks, which do not take FCN's unique architectural preferences into account. Weights and activations were fixed to a single 8-bit configuration, preventing systematic study of how the synthesized layouts scale with precision. This work therefore provides a substantial extension upon the original methodology at multiple layers: at the RTL design layer, parameterized quantization enabled a systematic study of downstream trade-offs in the synthesis pipeline; for RTL-to-netlist synthesis, the introduction of technology-appropriate ALUs reduced the resulting area; and for netlist-to-atoms synthesis, multiple SiDB-focused refinements significantly improved the quality of the results. These contributions transform a one-off demonstration into a concrete framework optimized for SiDB logic synthesis, revealing how architectural and layout decisions shape the strengths and limitations of current tools.



(a) SiDB unit cell.

(b) *Bestagon* NAND gate.

Fig. 1. (**a**) SiDB logic unit cell: a pair of closely-spaced SiDBs share a charge whose position encodes logic `0` or `1`, controlled by the location of a nearby "perturber" SiDB. (**b**) Example NAND gate from the *Bestagon* library, with inputs applied at the upper pins via perturbers, and the output sensed at the lower pin; the central region hosts SiDBs that implement the gate function. With permission, (a) is reproduced from [13], (b) is adapted from [16].

II. Background

SiDBs can be fabricated on hydrogen-passivated Si(100)-2×1 using the tip of a scanning-tunneling microscope [4], [5]. Each SiDB can hold discrete negative, neutral, or positive charge states. In dense ensembles, the charge state of each SiDB is determined by their collective interaction, bulk doping levels, and external electrostatic influences [5]. Binary logic states can be encoded by the position of a charge shared among neighboring SiDBs (Fig. 1a), enabling the experimental demonstration of an OR gate with a footprint of only $5 \times 6\,\mathrm{nm}^2$ [7]. Further exploration was enabled by *SiQAD* [8], a CAD tool specialized for SiDB logic, enabling the introduction of the *Bestagon* standard-tile library [17], featuring standardized input/output (I/O) pin locations and a central canvas on which SiDBs are placed to implement logic (Fig. 1b). The *Bestagon* library has allowed *fiction* [12], an EDA framework specialized for FCN, to take gate-level netlists as input and synthesize dot-accurate, fabricable SiDB layouts [17].

SiDB logic, like other FCN families, uses spatially partitioned clock zones to enforce directed data flow [8], [9]. Hanging or buried electrodes apply phase-shifted sinusoidal potentials that modulate the surface band bending and thereby the local charge density [9]. Adjacent electrodes are offset by 90° in phase, producing "active" regions where charges are present, and computation occurs, separated by "inactive" regions that are effectively charge-free buffers. An area-efficient arrangement places these clock zones in rows to support unidirectional, purely combinational logic [9], [18]. Interfacing with CMOS can be achieved by biasing inputs electrostatically via electrodes [9], while outputs can be sensed using charge-sensitive devices such as single-electron transistors [19]–[21].

III. Related Work

A. Electronic Design Automation for FCN

Beyond *fiction*, other FCN EDA frameworks have focused on nanomagnetic logic (NML) which stores and propagates information via the magnetization of interacting nanomagnets rather than mobile charges. Among them, *ToPoliNano* [22] provides a top-down flow from structural VHDL to clocking-aware in-plane NML layouts, and *FUNCODE* [23] infers VHDL netlists from custom NML layouts for HDL-based analysis. These works demonstrate broader FCN CAD capabilities, whereas the present work targets the emerging atomic-scale SiDB logic platform, for which RTL-to-atoms design flows are maturing.

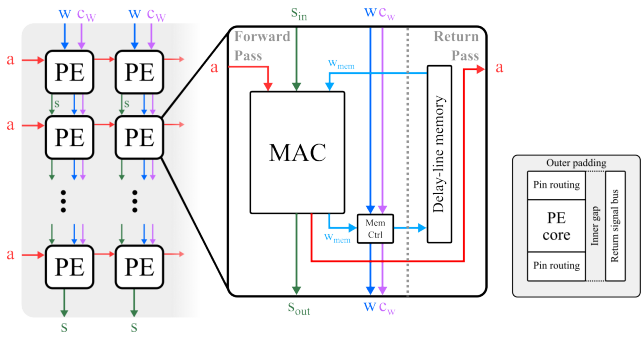


Fig. 2. A 2D systolic array MXU receiving quantized activations a, weights w, control signals c_w, and partial sums s. Each PE consists of a MAC unit, memory controller, and memory for weight storage. Signals propagate downward on the left (forward pass) and upwards on the right (return pass). A breakdown of components for dimension estimations is shown on the right (see Section V-C).

B. SiDB ML Accelerators

Before dedicated SiDB-aware EDA tooling became available, studies of SiDB applications were limited and largely relied on manual CAD prototyping with extrapolated area and latency estimates. One such example was an SiDB MXU for machine learning (ML) acceleration [13], inspired by Google's TPUv1 [14] and organized as a systolic array of processing elements (PEs) (Fig. 2). At the PE level, the core operation is a multiply–accumulate (MAC), $s_{\text{out}} = s_{\text{in}} + (w \cdot a)$. The design operates in preload and compute phases and comprises a *forward pass*, which contains the MAC and memory controller, and a *return pass*, which closes the delay-line memory loop and aligns activations between neighboring PEs. Both paths are purely combinational, enabling row-wise clocking and throughput-oriented pipelining. While that study reported promising extrapolated area and power estimates against the TPUv1 [13], [14], it did not produce a manufacturable layout or formal verification. The preceding conference version of this work [15] bridged this gap by demonstrating an RTL-to-atoms flow for a quantized SiDB MXU, which the present manuscript substantially refines through technology-specific optimizations and multi-bit-width scaling analysis.

IV. Proposed Methodology

This section presents the proposed end-to-end RTL-to-atoms design and synthesis flow, broken down to the hierarchical RTL design, RTL-to-netlist conversion, and netlist-to-atoms physical design with optimizations for SiDBs.

A. Matrix Multiply Unit in Hierarchical RTL

To achieve this work's objectives, the RTL implementation must satisfy the following requirements: 1) the RTL must be

Algorithm 1 Forward Pass Logic in Processing Element

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_{load}
$s_{\text{in}} \leftarrow$ input partial sum
$a \leftarrow$ input activation
$w_{\text{mem}} \leftarrow$ weight loaded from delay-line memory
Outputs:
s_{out}: new partial sum
w_{memOut}: 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
  $s_{\text{out}} \leftarrow s_{\text{in}} + \text{signExtend}(p, 24)$ ▷ Sum with partial sum
  if $m =$ PRELOAD **and** $PE_{\text{target-}y} = PE_y$ **then**
    $w_{\text{memOut}} \leftarrow w_{\text{load}}$ ▷ Update stored weight
  end if
  return s_{out}, w_{memOut}, and other pass-through wires
end procedure

purely combinational for *fiction* compatibility; and 2) operation of the pipelined PE must be verifiable so that test benches can establish operational correctness, addressing the lack of formal verification in prior work [13]. Because the PE includes forward and return paths that form an internal feedback loop, a direct RTL implementation of the full PE would be incompatible with *fiction*. To reconcile these constraints, this work compartmentalizes the hierarchical RTL design:

Combinational core: The PE's forward-pass logic is implemented in strictly combinational RTL, as shown in Alg. 1, ensuring compatibility with subsequent synthesis steps.

Clocked shell: A higher-level RTL module captures the clocked operation of the full PE, defining the component's input and output signals, pipelined internal signal steering, and synchronized signal timing for the return pass.

Both are parameterized by the weight and activation bit-widths, allowing the study of their scaling behavior in Section V.

Correctness was validated with test benches across both layers. These tests confirmed correct weight storage in preload mode and accurate MAC results in compute mode for representative inputs. In particular, interleaved inputs for concurrent MAC operation across all pipeline stages were also verified. All simulations were performed using Icarus Verilog [24], establishing logical correctness of the PE prior to synthesis and physical mapping. The codebase is fully open source.[1]

B. RTL-to-Netlist

With the RTL design verified, the first stage of the synthesis flow is to generate a gate-level netlist. To accomplish this, this work uses *Yosys* [25] to map arithmetic operators such as accumulation ($+$) and multiplication ($\star$) into gate-level ALUs. Absent explicit directives, *Yosys* selects CMOS-optimized ALUs, which underperform in FCN due to differences in area and latency trade-offs under the strictly planar fabric [26]. To ensure technology-appropriate choices, this work supplies explicit gate-level descriptions of a ripple-carry adder and an array multiplier to *Yosys*, both of which have been shown to be suitable for FCN thanks to their regular structure, which eases wiring congestion [26]. The mapped netlist is then rewritten as an And-Inverter Graph (AIG) and optimized with ABC's *&deepsyn* [27] strategy to reduce its node count, and re-validated against test benches to confirm functional correctness.

[1]GitHub repository: https://github.com/samuelshng/sidb-mxu-verilog

C. Netlist-to-Atoms

The optimized AIG is then passed to *fiction* for the second stage of the synthesis flow: physical design. This stage transforms the netlist into a dot-accurate SiDB layout through a multi-step process, which this work enhances with several optimizations for the SiDB platform.

1) Synthesis Flow: This work starts with the SiDB-specific flow in *fiction* proposed by [17]:

Technology mapping maps the AIG onto the *Bestagon* gate library [17], exploiting the library's richer primitives to shrink the logic compared with an AND/INV-only basis.

Placement and routing with the *ortho* [28] or graph-oriented layout design (*gold*) [29] algorithm produces a layout on a Cartesian grid.

Post-layout optimization (*PLO*) [30] reduces the footprint of the placed and routed design in a post-processing step.

Hexagonalization projects the Cartesian layout onto a hexagonal grid [31] to align with *Bestagon* gates.

Equivalence checking [32] using Boolean satisfiability confirms that the hexagonalized layout preserves the gate-level behavior.

Atom mapping maps each tile in the verified layout to the corresponding *Bestagon* gate's SiDB arrangement, yielding a dot-accurate, manufacturable SiDB layout.

While this established workflow provides a robust foundation, its application to the SiDB platform revealed opportunities for technology-specific enhancements as detailed below.

2) FoM-aware Technology Mapping: Previous SiDB synthesis flows within *fiction* treated all gates in the technology library as equivalent, without accounting for physical robustness. Recent work unified major SiDB robustness factors, including operational domain size, thermal stability, band-bending effects, and defect sensitivity, into a single figure of merit (FoM) cost function [33], which this work adopts during technology mapping to prioritize more robust gates despite possible trade-offs in area or dot count.

3) Placement and Routing Optimizations: The placement-and-routing framework within *fiction* was first established for quantum-dot cellular automata circuits on a Cartesian grid [12]. Support for SiDB logic was later enabled through a coordinate mapping that rotates the Cartesian layout by $45°$ and projects it onto the hexagonal SiDB lattice [31]. This transformation can distort aspect ratio: layouts that are compact on the Cartesian grid may become tall after hexagonalization when diagonal signal paths map to long vertical paths. While the older *ortho* algorithm offers no direct compensation, the more recent *gold* algorithm [29] exposes a customizable placement-and-routing cost objective. This work therefore introduces new *gold* cost objectives that favor more balanced footprints and mitigate the distortion induced by the Cartesian-to-hexagonal mapping, as evaluated in Section V-B.

4) Hexagonalization for SiDB Logic: The hexagonalization algorithm was also revised in this work to align generated layouts with SiDB clocking expectations. Under the Cartesian layout's 2DDWave clocking scheme [34], primary inputs are placed along the north and west borders, and data advances diagonally toward the southeast. The $45°$ rotation left many I/O pins recessed from the north and south edges, which reduces the attainable throughput because some signals must be held across multiple cycles for synchronization. The updated flow stretches all I/O pins to the north and south boundaries after projection, producing layouts whose ports are time-aligned as they enter and exit the hexagonal fabric. An optional constraint can be toggled to perform the pin extensions without wire crossings as they are typically less robust against environmental defects.

Although the present flow is instantiated for the H-Si(100)-2×1 surface, the underlying tile-based methodology is not inherently limited to that orientation. Recent work in [35] extended SiDB design support to H-Si(111)-1×1 and demonstrated, in simulation, a hexagonal standard-tile library. This indicates that the proposed RTL-to-atoms methodology can in principle be transferred to other surface orientations if future experimental validation makes this a favorable approach.

V. Experiments

This section evaluates the RTL-to-atoms flow using the synthesized PE-core layouts, starting with the experimental protocol and configurations, then the resulting synthesis metrics across the evaluated configurations, and lastly a discussion on the findings and comparisons with prior manual estimations.

A. Experimental Protocol

This comparative study characterizes how activation and weight bit-widths scale and reports synthesis results across various synthesis configurations. To this end, the combinational PE-core (Section IV-A) is synthesized for 2-, 4-, and 8-bit weights and activations, henceforth denoted W2A2, W4A4, and W8A8, respectively. These selections correspond to widely studied quantized regimes for ML workloads where the dominant operations reduce to dense matrix multiplications [14], [36], [37]. The RTL-to-netlist flow from this work instructs *Yosys* to use technology-appropriate ALUs to produce an AIG (Section IV-B), which is then optimized using ABC's *&deepsyn* strategy [27]. In the netlist-to-atoms flow, two *Bestagon* library [17] variants are compared during technology mapping: a uniform cost baseline, and an FoM-aware counterpart. *ortho* and *gold* are used for placement and routing.

B. Results

The best synthesis results obtained for the PE-core under the tested configurations are presented in TABLE I. Compared to prior results from [15], which left ALU selection to *Yosys*'s CMOS-aligned defaults, the explicit selection of SiDB-aligned ALUs yields a $15\,\%$ area reduction, underscoring the importance of steering EDA tools toward technology-specific component choices. Turning to the FoM-informed technology mapping results, the corresponding layouts are consistently larger than those obtained by treating all gates with equal preference, indicating that the technology mapper has 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 SiDB logic implementation.

TABLE I
Synthesis comparison across MXU and EDA configurations

ALUs	PE	Alg	Uniform TM $w_{\text{core}} \times h_{\text{core}} = A_{\text{uni}}$	A_{FoM}
Default [15]	W8A8	*ortho+PLO*	$515 \times 1043 = 537\,145$	541 284
Optimized (This work)	W8A8	*ortho+PLO*	$474 \times 967 = 458\,358$	470 830
	W4A4	*ortho+PLO*	$143 \times 333 = 47\,619$	56 744
		gold+PLO	$116 \times 241 = 27\,956$	70 070
	W2A2	*ortho+PLO*	$67 \times 151 = 10\,117$	10 833
		gold+PLO	$48 \times 98 = 4704$	5457

"Default" and "optimized" ALUs indicate *Yosys*'s ALU selection mode; w and h denote the tile counts along the width and height; $A = w \times h$; A_{uni} and A_{FoM} denote the *Bestagon* variant used for technology mapping.

TABLE II
Cost objective comparison for *gold* algorithm

PE	Cost obj	$A_{\min}$	$\bar{A}$
W4A4	Area (baseline)	27 956	37 483.61
	$x + y$	27 956 (±0.00 %)	37 303.28 (−0.48 %)
	$x^2 + y^2$	28 175 (+0.78 %)	36 716.71 (−2.05 %)
W2A2	Area (baseline)	5145	10 156.22
	$x + y$	5145 (±0.00 %)	8107.02 (−20.18 %)
	$x^2 + y^2$	4704 (−8.57 %)	8121.47 (−20.03 %)

$A_{\min}$ and $\bar{A}$ represent the minimum and mean areas under uniform technology mapping. All trials ran *gold* in single-core mode (400 s timeout on AMD Ryzen 9 9950X3D, maximum effort mode, input pin tile skipping swept 1–6 with randomization). 630 trials were performed per configuration with unsuccessful results discarded. *PLO* had maximum gate relocations set to 1.

Comparing placement and routing algorithms, while the W8A8 netlist exceeds *gold*'s tractable problem size, W4A4 and W2A2 remain sufficiently small for *gold* to successfully place and route, achieving a $\approx 40\,\%$ to $50\,\%$ area reduction relative to *ortho*-based counterparts. However, the FoM-aware synthesis of W4A4 using *gold* exhibits a pronounced area increase. This work interprets the outlier as evidence that, under FoM-aware technology mapping, the AIG is already near the upper bounds of *gold*'s ability to provide consistent solutions: only 1 trial succeeded among 630 trials for this configuration. Nevertheless, configurations that remain within *gold*'s effective scaling regime yield area reductions that directly benefit manufacturability and reduce power consumption in practice.

Part of the improvement achieved with *gold* arises from refining the cost objectives in the SiDB-specific synthesis workflow (Section IV-C3). Two additional *gold* cost objectives were implemented—Manhattan distance $(x + y)$ and squared Euclidean distance $(x^2 + y^2)$—and compared against the default area-based objective in TABLE II. For the W2A2 configuration, both the best and average layout areas improve, with the gains being more pronounced in the average case and the squared Euclidean objective yielding the smallest minimum area. For

W4A4, modest improvements persist in the average case, but no benefit is observed in the minimum area. This behavior is consistent with W4A4 already operating near the upper end of *gold*'s practical scaling range, where a reduced number of successful runs can skew the distribution of best-case results.

C. Discussion

The presented synthesis results cover the PE-core, i.e., the logic-bearing portion of the PE; the omitted structures are used for signal propagation (see Section IV-A). As shown in Fig. 2, a full PE-level estimation must additionally account for *pin routing* from the PE-core to the outer I/O of the PE, the *return signal bus* carrying the activation and stored weight, and clocking-related spacing between neighboring regions. Their widths and heights are denoted by $w_{\text{core}}, h_{\text{core}}$ for the PE-core, $w_{\text{routing}}, h_{\text{routing}}$ for each pin-routing block, and $w_{\text{bus}}, h_{\text{bus}}$ for the return signal bus. Automating these steps calls for further streamlining of SiDB EDA tools and is left as future engineering work. However, a first-order pessimistic estimate can still be derived for comparison with the manual W8A8 SiDB MXU estimates in [13] under the following assumptions:

1) Since the RTL-to-atoms flow in *fiction* does not enforce a fixed physical ordering of I/O pins, the worst case is assumed in which an outer-edge pin lies opposite its location on the PE-core, giving $h_{\text{routing}} = w_{\text{core}}$ for each pin-routing block.
2) The return signal bus carries the 8-bit weight and activation, giving $w_{\text{bus}} = 16$ and a height equal to that of the forward pass.
3) The clocking interfaces at the edge of PEs and between the forward and return passes are conservatively assumed to require one full logical pipeline stage of separation. Under 4-phase clocking (see Section II), this yields an inner gap of $200\,\text{nm}$ and an outer padding of $100\,\text{nm}$ per side for an inter-electrode spacing of $\approx 50\,\text{nm}$ [13].

Using the tile dimensions in TABLE I, the physical dimensions of the PE-core can be computed by $w \times 23.06\,\text{nm}$ and $h \times 13.06\,\text{nm}$.[2] When compared against the manual SiDB MXU estimates from [13] ($5000\,\text{nm} \times 8150\,\text{nm}$ per PE), the synthesized PE-core represents a $3.4\times$ area increase, and the pessimistic full-PE estimation represents an increase of $7.2\times$.

This discrepancy arises from three main factors. First, [13] was intentionally framed as a blueprint and therefore relied on manual architectural insight and component-level extrapolation rather than a fully placed-and-routed implementation, whereas the present work derives the PE-core through an automated and verifiable synthesis pipeline that produces dot-accurate layouts. This automation, however, offers only indirect control over ALU structures through *Yosys*, causing arithmetic boundaries to blur and limiting the extent to which that structure can be exploited later; the lack of pin-order enforcement in *fiction* further increases routing cost. Second, the *Bestagon* gate library [17] deliberately uses a larger footprint to maintain sufficient separation between neighboring logic canvases and thereby reduce inter-gate interference, making it less dense than the hand-estimated components in [13]. For perspective, [13] assumed wire spacing of $11\,\text{nm}$ versus *Bestagon*'s $23\,\text{nm}$, and $60\,\text{nm} \times 62\,\text{nm}$ full adders versus a smallest possible *Bestagon* counterpart of $250\,\text{nm} \times 65\,\text{nm}$,[3] with no guarantee that the synthesized layout employs this smallest arrangement. Third, the W8A8 netlist timed out in *gold*, consistent with the exponential growth in partial placements reported in [29], and thus lies beyond the algorithm's current practical scaling range, preventing the $\approx 40\,\%$ to $50\,\%$ PE-core area reductions observed for smaller configurations.

Nevertheless, the successful end-to-end design flow achieved in this work demonstrates what state-of-the-art FCN EDA tools can realize on the SiDB platform, combining manufacturable layouts with full testbench-based verification and highlighting opportunities for further optimization.

VI. Conclusion and Future Work

This work has demonstrated an end-to-end RTL-to-atoms synthesis flow for a quantized SiDB-based MXU PE with fully verifiable behavior across abstraction levels, enabled by a hierarchical RTL organization that separates a combinational PE-core from its clocked shell. By explicitly selecting SiDB-appropriate ALUs in *Yosys* and tailoring technology mapping, placement-and-routing, and hexagonalization to SiDB logic, the presented framework achieves a $\approx 15\,\%$ reduction in synthesized PE-core area over prior flows and realizes additional footprint savings of $\approx 40\,\%$ to $50\,\%$ for configurations within the scaling range of newer placement-and-routing algorithms, while making explicit the trade-off between footprint and device-level robustness under FoM-aware mapping. Together, these components establish a reproducible benchmark for atomic-scale EDA that links RTL design decisions, EDA tooling, and manufacturable SiDB layouts.

While logical correctness in this work focuses on RTL verification and layout synthesis, the realized behavior of the MXU can still be perturbed by physical non-idealities such as charged surface defects, parameter variations, thermal effects, and band-bending perturbations [33], [38]. Addressing these effects is a natural next step through defect-aware and robustness-aware CAD/EDA methods [39], while dynamic noise under field-modulated operation will likely require more accurate non-equilibrium simulation; the hierarchical RTL provides a suitable starting point for such future evaluation.

Beyond these reliability considerations, the findings also reveal several avenues for future work to bridge remaining gaps with manual expert designs. Preservation of frequently used arithmetic structures across the synthesis stages (i.e., not decomposing them into elementary gates but keeping them as larger standard cells or macros) will allow optimized ALU layouts to be directly applied in the physical layout stage. For physical design, the scalability and robustness of *gold* can be improved by implementing native support for the hexagonal grid as it improves space utilization by removing the Cartesian-to-hexagonal grid projection; it also enables the use of more compact gates such as the half adder tile which

[2] These scaling factors differ from [15] due to corrected calculation factors.

[3] Estimated using *fiction*-supported tiles, which excludes the single-tile half adder [17].

is available in the *Bestagon* gate library [17] but currently not supported by *fiction*. Finally, automating full PE- and MXU-level synthesis—including routing of the return path and of the PE-core's pins to the outer PE module—will enable application-scale studies that jointly consider device reliability, clocking constraints, and workload characteristics. Taken together, the successful demonstration of an end-to-end RTL-to-atoms flow for a quantized SiDB-based MXU PE provides a concrete methodological foundation for these avenues and serves as the blueprint for future scalable accelerator studies on SiDBs.

ACKNOWLEDGEMENT

OpenAI LLMs were used for limited writing aid for wording and clarity; they were also used to assist with coding and debugging for Verilog, *Yosys* scripts, and *gold* improvements. All outputs were reviewed and verified by the authors, who assume full responsibility for the content.

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

[3] D. Giri *et al.*, "Modeling, Design, and Analysis of MagnetoElastic NML Circuits," *IEEE Transactions on Nanotechnology*, vol. 15, no. 6, pp. 977–985, Nov. 2016.

[4] 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.

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

[6] 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.

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

[8] 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.

[9] H. N. Chiu *et al.*, "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.

[10] 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.

[11] W. Lambooy *et al.*, "Mastering the Exponential Complexity of Exact Physical Simulation of Silicon Dangling Bonds," in *Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2026.

[12] M. Walter *et al.*, "Fiction: An open source framework for the design of field-coupled nanocomputing circuits," *ArXiv*, vol. abs/1905.02477, 2019.

[13] S. S. H. Ng *et al.*, "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.

[14] N. P. Jouppi *et al.*, "In-datacenter performance analysis of a tensor processing unit," 2017.

[15] S. S. H. Ng *et al.*, "Building a Machine Learning Accelerator with Silicon Dangling Bonds: From Verilog to Quantum Dot Layout," in *2025 IEEE 25th International Conference on Nanotechnology (NANO)*, 2025, pp. 483–488.

[16] 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.

[17] M. Walter *et al.*, "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.

[18] C. S. Lent and P. D. Tougaw, "A device architecture for computing with quantum dots," *Proceedings of the IEEE*, vol. 85, no. 4, pp. 541–557, Apr. 1997.

[19] M. Fuechsle *et al.*, "A single-atom transistor," *Nature Nanotechnology*, vol. 7, no. 4, pp. 242–246, Apr. 2012.

[20] A. A. Prager, A. O. Orlov, and G. L. Snider, "Integration of CMOS, single electron transistors, and quantumdot cellular automata," in *2009 IEEE Nanotechnology Materials and Devices Conference, NMDC 2009*, 2009.

[21] S. Bohloul *et al.*, "Quantum transport in gated dangling-bond atomic wires," *Nano Letters*, vol. 17, no. 1, pp. 322–327, Jan. 2017.

[22] F. Riente *et al.*, "ToPoliNano: A CAD Tool for Nano Magnetic Logic," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 36, no. 7, pp. 1061–1074, 2017.

[23] U. Garlando, F. Riente, and M. Graziano, "FUNCODE: Effective Device-to-System Analysis of Field-Coupled Nanocomputing Circuit Designs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 40, no. 3, pp. 467–478, 2021.

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

[25] C. Wolf, J. Glaser, and J. Kepler, "Yosys-a free verilog synthesis suite," 2013.

[26] S. W. Kim and E. E. Swartzlander, "Multipliers with coplanar crossings for quantum-dot cellular automata," in *10th IEEE International Conference on Nanotechnology*, Aug. 2010, pp. 953–957.

[27] 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.

[28] M. Walter *et al.*, "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.

[29] S. Hofmann, M. Walter, and R. Wille, "Graph-Oriented Layout Design for Field-Coupled Nanocomputing via Parallel Multi-Objective Search Space Exploration," *IEEE Transactions on Circuits and Systems I: Regular Papers*, pp. 1–14, 2025.

[30] ——, "Efficient and Scalable Post-Layout Optimization for Field-Coupled Nanotechnologies," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 44, no. 10, pp. 3790–3803, 2025.

[31] ——, "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.

[32] M. Walter *et al.*, "Verification for field-coupled nanocomputing circuits," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*, 2020, pp. 1–6.

[33] J. Drewniok *et al.*, "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.

[34] 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.

[35] S. S. H. Ng *et al.*, "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.

[36] S. K. Esser *et al.*, "Learned Step Size Quantization," 2020.

[37] R. Banner, Y. Nahshan, and D. Soudry, "Post training 4-bit quantization of convolutional networks for rapid-deployment," in *Advances in Neural Information Processing Systems*, vol. 32. Curran Associates, Inc., 2019.

[38] S. S. H. Ng *et al.*, "Simulating Charged Defects in Silicon Dangling Bond Logic Systems to Evaluate Logic Robustness," *IEEE Transactions on Nanotechnology*, vol. 23, pp. 231–237, 2024.

[39] J. Drewniok *et al.*, "On-the-fly Defect-Aware Design of Circuits based on Silicon Dangling Bond Logic," in *2024 IEEE 24th International Conference on Nanotechnology (NANO)*. Gijon, Spain: IEEE, Jul. 2024, pp. 30–35.