BitPlanarNet: Learning-to-Layout Planar Gate Networks for Emerging Devices

Samuel S. H. Ng, Marcel Walter, Max Yuan, Jan Drewniok, Ryan H. Foote, Robert Lupoiu, Jonathan A. Fan, Robert Wolkow, Robert Wille, and Konrad Walus

***Abstract*— Emerging planar computing platforms such as atomic-scale computing and silicon photonics combine gates and interconnects in a single 2D plane, making wiring and crossings dominant bottlenecks. This work reframes differentiable logic network training as one-pass synthesis: BitPlanarNet trains a strictly planar gate network that maps one-to-one to device primitives. Building on differentiable logic gate networks (DLGNs), each neuron selects a 2-input/2-output primitive and connects only to nearest neighbors, yielding a gate-level layout at discretization. It is demonstrated that this concept is physically viable by successfully fabricating a $33 \times 25\,\mathrm{nm}^2$ representative learned dangling-bond logic layout on a hydrogen-passivated silicon surface using a scanning tunneling microscope. On image-classification tasks, BitPlanarNet achieves accuracy comparable to unconstrained DLGNs—remaining within approximately 1 percentage point on MNIST and 5 percentage points on CIFAR-10—while guaranteeing planar connectivity, providing a versatile methodology for translating learned tasks directly into realizable gate layouts on emerging planar technologies.**

I. Introduction

The rapid expansion of AI workloads across sectors continues to strain conventional CMOS technology, where scaling limits in density and energy efficiency restrict further improvements in compute capability. These pressures have motivated the exploration of emerging computing platforms such as atomic-scale computing [1], silicon photonics [2], and microfluidic logic [3]. These platforms promise a combination of advantages such as ultra-low energy switching, extreme compactness, high-frequency operation, or the ability to operate in environment-constrained settings.

A defining characteristic of the platforms considered in this work is strict planarity: gates and interconnects must coexist on a single 2D surface with limitations on fan-out and crossings. This constraint makes placement and routing a dominant cost, and severely limits the scalability of conventional ML accelerators on planar devices. Past attempts at implementing arithmetic blocks on field-coupled nanocomputing (FCN) technologies such as quantum-dot cellular automata (QCA) [4], [5] and atomic-scale computing [6] indicate that interconnects quickly dominate layout cost, underscoring core challenges posed by planarity. Similar interconnect-related scaling limits are recognized in silicon-photonic Mach-Zehnder interferometer (MZI) meshes, where accumulated optical loss, crosstalk, and control overhead constrain mesh depth and effective fan-out [7]. What is missing is a learning framework that respects planar wiring locality and produces a network whose topology is directly realizable as a planar layout. This work treats training as one-pass synthesis, enabling hardware to learn arbitrary operations—circuits or tasks—directly as planar gate graphs.

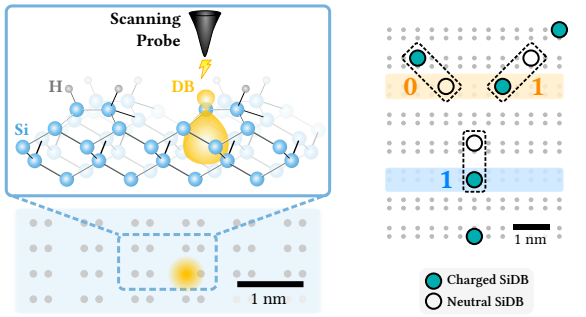
Differentiable logic gate networks (DLGNs) [8] provide a framework for training Boolean gate networks on ML tasks, yet existing formulations permit unrestricted inter-layer wiring with crossings. To address this gap, this work introduces BitPlanarNet, a DLGN-based architecture co-designed with the physical rules of planar devices. BitPlanarNet enforces planarity by restricting connectivity to nearest neighbors, constraining successive layer widths to differ by exactly one, and supporting 2-input/2-output ($2\to2$) primitive libraries aligned with platform-native logic components. Training yields a planar, combinational graph that maps directly to physical layout with minimal synthesis or placement-and-routing overhead.

To validate this co-design principle, BitPlanarNet is implemented and fabricated using silicon dangling-bonds (DBs)—the building blocks for atomic-scale computing—with a custom-designed gate library. A one-pass synthesis workflow is demonstrated: from training, to a discretized logic graph, to a dot-accurate DB layout. Furthermore, manufacturing feasibility is proven through fabricating an atomically-precise $33 \times 25\,\mathrm{nm}^2$ sub-layout of the trained network using a scanning tunneling microscope (STM), highlighting the practicality of the proposed train-to-layout pipeline. On image-classification tasks chosen to match those used in the original DLGN evaluation [8], the approach attains comparable accuracies, reaching deltas of ≈ 1 percentage point (p.p.) on MNIST and 5 p.p. on CIFAR-10, while adding direct mappability to planar hardware. These results position BitPlanarNet as a learning-to-layout framework for planar devices, where training yields device-matched gate networks that translate directly to manufacturable layouts.

II. Background

Planar computing platforms exhibit common constraints: logic gates and interconnect share a 2D surface, practical

S. Ng and K. Walus are with the Department of Electrical and Computer Engineering, University of British Columbia, Vancouver, BC, Canada ({samueln, konradw}@ece.ubc.ca); M. Walter, J. Drewniok, and R. Wille are with the Chair for Design Automation, Technical University of Munich, BY, Germany ({marcel.walter, jan.drewniok, robert.wille}@tum.de); M. Walter and R. Wille are also with the Munich Quantum Software Company GmbH, Garching near Munich, BY, Germany. R. Wille is also with the Software Competence Center Hagenberg GmbH, Hagenberg, OÖ, Austria; M. Yuan, R. H. Foote, and R. Wolkow are with the Department of Physics, University of Alberta, Edmonton, AB, Canada ({myuan1, rhfoote, rwolkow}@ualberta.ca); M. Yuan, R. H. Foote and R. Wolkow are also with Quantum Silicon Inc., Edmonton, Alberta, Canada. R. Lupoiu and J. A. Fan are with the Department of Electrical Engineering, Stanford University, Stanford, CA, United States of America ({rclupoiu, jonfan}@stanford.edu).



(a) (b)

Fig. 1. Atomic-scale computing with DBs. (a) H-Si(100)–2×1 surface with an STM probe which created a DB by applying voltage pulses. (b) *SiQAD* [9] reproduction of an experimentally demonstrated OR gate [1].

fan-out is limited, and crossings are costly. To ground these constraints, atomic-scale computing is reviewed as a representative planar platform. Atomic-scale computing is a charge-based computing platform in the FCN family built with DBs; it distinguishes itself by enabling atomically precise fabrication and a rich library of compact logic primitives. DBs can be patterned atom-by-atom on hydrogen-passivated Si(100)–2×1 surfaces using an STM probe as depicted in Fig. 1a. Combined with their ability to hold discrete charge states [10], DBs form the basis for logic primitives that encode bit values positionally via a charge shared between two DBs in proximity [1]. Careful arrangements of these primitives yield logic gates, including an experimentally demonstrated OR gate occupying only $\approx 5 \times 6\,\mathrm{nm}^2$ as depicted in Fig. 1b, thereby highlighting the extreme density potential of charge-based logic on silicon surfaces [1].

The emergence of *SiQAD*, a CAD tool for DB layouts, has spurred simulation-informed component studies [11], [12] and robustness-driven figures of merit (FoMs) [13], [14]. Automated gate-design tools [15] have produced gate libraries covering 2→1 and 2→2 functions [16], culminating in the *Bestagon* gate set [16] which is designed for DB synthesis flows. This enabled *fiction* [17], an EDA framework specialized in FCN, to technology map gate-level layouts to *Bestagon* and perform placement and routing for DB logic [17]. Together, these tools provide an end-to-end flow from register-transfer level (RTL) Verilog to manufacturable DB layouts that have enabled application-scale studies such as a matrix-vector multiplier for ML acceleration [6].

Directionality and power delivery in DB logic come from clocked domains that partition the surface and steer data flow [9], [18]. Hanging or buried electrodes apply sinusoidal biases that modulate band bending, creating charge-rich "active" domains separated by low-potential, charge-depleted buffers. As phases advance, the active front shifts to propagate information [9], [18]. Row-wise clocking provides the most area-efficient clocking and favors unidirectional, purely combinational layouts [18]. The interface to CMOS is via electrostatically biased input electrodes [18] and charge-sensitive output sensors such as single-electron transistors [19].

III. Related Neural Architectures

AI acceleration is central to modern computing. This section briefly positions BitPlanarNet against prior planar AI accelerators and the DLGN family on which it builds. Starting with the former, modern neural workloads are typically organized around linear algebra primitives, with matrix-vector products executed on dedicated matrix multiply units (MXUs) [20]. For atomic-scale computing, prior work has explored MXU-style computation on DBs [6], but the resulting layouts were dominated by interconnect and crossing structures rather than logic gates. Similar routing pressure has also been reported in other FCN technologies [4], [5]. These observations motivate learning architectures whose topology already respects planar locality without subsequent reliance on conventional synthesis and routing.

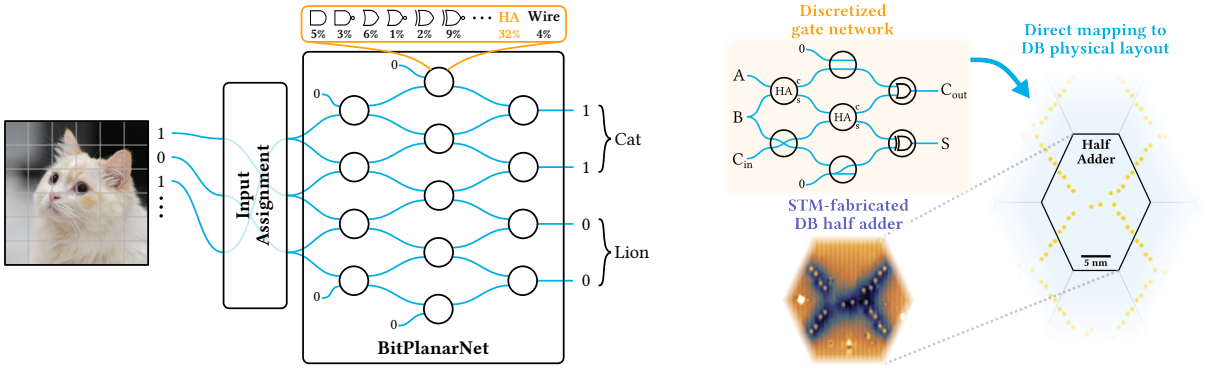
Petersen *et al.* introduced DLGNs [8], in which each neuron learns a soft mixture over Boolean gates and is discretized after training to yield a logic network. This makes DLGNs a natural starting point for trainable logic on emerging hardware. However, prior DLGNs do not enforce planar nearest-neighbor connectivity, do not target device-specific 2→2 primitive libraries, and do not directly yield layout-ready planar graphs. Subsequent work has improved DLGNs through convolutional structure and residual initialization [21], explored learnable interlayer connectivity [22], and further studied recurrent and more scalable variants [23]–[25]. In parallel, differentiable lookup table (LUT) networks have also been proposed [26], but they do not restrict learned functions to compact technology-matched primitive libraries. BitPlanarNet addresses these gaps by co-designing the trainable logic network with planar device constraints and fabrication-oriented primitive sets.

IV. BitPlanarNet: Proposed Gate Network for Planar Devices

This section presents the manuscript's primary contribution, adapting DLGNs for strictly planar fabrics through a network formulation employing 2→2 gates, supporting regularizers, and input assignment strategies.

A. Network Formulation

BitPlanarNet adapts DLGNs to planar devices by imposing connectivity constraints and expanding the gate vocabulary to match device-level primitives. As in prior DLGNs, binary logic is relaxed during training: inputs and intermediate activations lie in $[0, 1]$, each discrete primitive is replaced by a smooth surrogate, and training operates on differentiable mixtures that are discretized after optimization. The objective is to train networks that translate to manufacturable, strictly planar layouts with minimal synthesis overhead, while maintaining end-to-end trainability and combinational signal flow. The visual summary in Fig. 2a depicts a nearest-neighbor 2→2 network with a software-side input assignment to optimize feature ordering (Section IV-C), mixture-driven primitive selection, and multi-bit class aggregation. Fig. 2b showcases a discretized gate network directly mapped to a



(a) BitPlanarNet architecture.

(b) Learning-to-layout: trained network fabricated on DBs.

Fig. 2. Overview of the BitPlanarNet architecture, from learned planar gates to silicon DBs.

DB layout; the core half-adder tile is then fabricated with an STM with atomic precision (Section V).

Unlike the DLGNs of [8], which permit randomized interlayer connectivity without constraints on crossings or fan-out, this work enforces strictly planar connectivity: each neuron draws inputs from its two nearest predecessors in the preceding layer and delivers its outputs to the nearest successors as shown in Fig. 2. Layer sizes change by at most one neuron between successive layers to preserve planarity and constant local fan-in/fan-out, with edge tiles in expanding layers taking 0 on their outer input. This topology directly enforces local wiring and eliminates crossings.

Beyond connectivity, this work also extends the primitive gate model from Petersen *et al.*'s 2→1 Boolean functions to support multi-output (2→2) primitives. The broadened gate vocabulary increases expressiveness per neuron by collapsing common two-pin patterns (e.g., half-adder) into single primitives without increasing area. The extension is realized by evaluating, for each output pin $s \in \{1,2\}$, a shared mixture over a configurable primitive library $\mathcal{G}$:

$$a_{\mathrm{o},s} = \Big[\sum_{g\in\mathcal{G}} \pi(g)\,\widetilde{g}(a_{\mathrm{i},1}, a_{\mathrm{i},2})\Big]_s, \pi(g) = \frac{e^{w(g)/\sigma}}{\sum_{h\in\mathcal{G}} e^{w(h)/\sigma}}. \quad (1)$$

Here, $[\cdot]_s$ selects the output pin, each $\widetilde{g}$ is a smooth, component-wise extension of a 2→2 primitive, and π is a temperature-controlled softmax over learned per-primitive logits w (larger σ yields a more diffuse mixture; smaller σ sharpens selection). At discretization, a single primitive is selected per neuron and applied to both pins, $g_{\mathrm{sel}} = \mathrm{argmax}_{g\in\mathcal{G}}\,\pi(g)$, yielding a combinational planar gate graph consisting strictly of permitted primitives.

For classification tasks, the final-layer outputs are grouped into class-specific vote blocks, following prior DLGNs but now operating on the planar BitPlanarNet outputs. Let n denote the number of output bits, K the number of classes, and $v{=}n/K$ the votes assigned to each class. The aggregated logit for class $k \in \{1,\dots,K\}$ is defined as

$$\hat{y}_k = \frac{1}{\tau}\sum_{j=(k-1)v+1}^{kv} a_j + \beta\,, \quad (2)$$

where $\tau{>}0$ is a normalization temperature and β is an offset.

B. Loss

Training is governed by a classification objective with an additional entropy regularizer. For classification tasks, the class logits are supplied by the vote aggregation in Eq. (2) and cross-entropy is applied, denoted here as $\mathcal{L}_{\mathrm{task}}$. The entropy regularizer closes the discretization gap between the train-time gate mixture and inference-time discrete gate selection by introducing an entropy loss $\mathcal{L}_{\mathrm{ent}}$ that penalizes indecisive mixtures and pushes the network towards a clear primitive choice per site. Under residual initialization [21], each gate's primitive mixture is initialized with high probability on wire-like primitives; without masking, the entropy term would penalize the temporary increase in entropy needed to move toward task-specific gates. Accordingly, a binary mask $m_g \in \{0,1\}$ is introduced to omit those primitives from the entropy penalty. At each site j, letting $\varepsilon_\bullet$ denote small positive constants for stability, the renormalized gate mixture $r_j(g)$ is obtained by normalizing $\pi_j(g)$ to the probability mass φ_j of the masked set:

$$\varphi_j = \sum\nolimits_{g\in\mathcal{G}} m_g\,\pi_j(g), \quad r_j(g) = m_g\,\frac{\pi_j(g)}{\varphi_j + \varepsilon_\pi}, \quad (3)$$

The normalized site entropy is 0 when $\sum_g m_g \le 1$; otherwise it is defined as

$$H_j = -\frac{1}{\log\left(\sum_g m_g\right)} \sum_{\substack{g\in\mathcal{G}:\\ m_g=1}} r_j(g)\,\log\left(r_j(g)+\varepsilon_H\right), \quad (4)$$

where $H_j \in [0,1]$ reduces with sharp primitive selections. The entropy regularizer averages entropies across J active sites scaled to a configurable entropy weight λ_S:

$$\mathcal{L}_{\mathrm{ent}} = \lambda_S\,\frac{1}{J}\sum\nolimits_{j=1}^{J} \varphi_j\,H_j\,. \quad (5)$$

Here, H_j is scaled by φ_j to avoid penalizing neurons for choosing masked primitives. The overall training objective is thus $\mathcal{L} = \mathcal{L}_{\mathrm{task}} + \mathcal{L}_{\mathrm{ent}}$, allowing the optimizer to promote decisive gate selections during task optimization.

C. Input Assignment

In contrast to prior DLGNs, which permit unconstrained interlayer wiring with crossings [8], BitPlanarNet maintains

strictly planar, nearest-neighbor connectivity, which introduces two constraints. First, because arbitrary fan-out is unavailable, wider hidden layers are obtained by duplicating primary inputs across multiple input neurons. Second, limited hop distance makes input ordering consequential, since it determines which features can interact locally.

Accordingly, this work considers both fixed and learned input assignments. A simple baseline is to duplicate inputs and randomize their order, but random shuffles do not preserve spatial locality. Fixed assignments therefore also include locality-preserving image traversals such as row-major, column-major, serpentine, spiral, Hilbert [27], and Morton [28]. With duplication, copies may follow identical or different traversals to diversify local contexts while keeping nearby features close along the input row.

When fixed geometry priors are insufficient or absent, a duplication-aware Sinkhorn layer [29] instead learns an assignment matrix $A \in \mathbb{R}^{T\times S}$ from the original S features to the duplicated input width $T = S\times D$. This is related in spirit to learnable-connectivity DLGNs [22], but is used here only to reorder primary inputs under fixed planar nearest-neighbor wiring. Each row of A is annealed toward a near one-hot choice before final argmax discretization. The effects of these strategies are explored in Section VI.

V. Case Study of BitPlanarNet on Silicon DBs

Using the BitPlanarNet formulation, this section applies it to atomic-scale computing with DBs as a case study, demonstrating the full learning-to-layout premise: a trained BitPlanarNet was discretized into a gate network that was mapped directly to a DB layout, and a sub-layout was fabricated via STM. There already exist multiple gate libraries proposed for DB logic and verified in CAD [9], [11], [16]. Among them, the *Bestagon* library [16] is particularly relevant because it standardizes input/output (I/O) pin locations on hexagonal tiles and interoperates with the *fiction* EDA flow [17], [30]. However, *Bestagon* is primarily optimized around $2\to1$ logic and uses most $2\to2$ tiles for interconnect, thus not taking full advantage of BitPlanarNet's $2\to2$ neuron model.

To match BitPlanarNet more directly, this work introduces a BitPlanarNet-specific DB gate library that preserves *Bestagon*'s I/O pin positions while expanding support for $2\to2$ logic primitives. The tiles were designed with established DB simulation and robustness-evaluation tools [9], [13], [15], [16]. For comparative studies, the resulting $2\to2$ tiles are grouped into three primitive sets:

- $\mathcal{G}_{16}$ implements the 16 Boolean functions mirrored on both outputs, providing a baseline against Petersen *et al.* [8];
- $\mathcal{G}_{20}$ adds interconnects (e.g., crossings) and a half-adder tile;
- $\mathcal{G}_{34}$ curates 7 more expressive $2\to2$ components, e.g., pass-through+XOR (CNOT-like) and AND/OR (min/max).

BitPlanarNet can use any of these sets as the primitive library. An additional set of 16 $2\to1$ output tiles, matching the truth tables of $\mathcal{G}_{16}$, completes the 50-tile library. Because all tiles share a tileable geometry, a discretized BitPlanarNet can be mapped directly into a dot-accurate physical layout without an additional placement-and-routing stage.

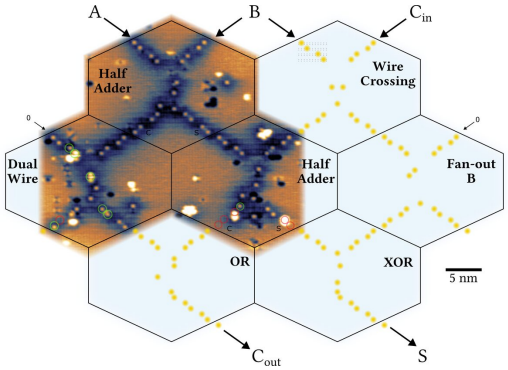


Fig. 3. A DB layout created from a BitPlanarNet trained on the full-adder truth table using $\mathcal{G}_{34}$ primitives, discretized into a planar gate network, and mapped to DBs. A $33 \times 25\,\mathrm{nm}^2$ sub-layout is fabricated using an STM, composited on the top left alongside the full trained version, demonstrating a one-pass end-to-end flow from network training to physical patterning. Green circles are added at DBs that are hard to see due to imaging artifacts; red circles indicate missing DBs or those very near charged defects.

As a proof-of-concept on DBs, a compact BitPlanarNet was trained on the full-adder truth table using $\mathcal{G}_{34}$. To satisfy the architectural constraint that adjacent layer widths differ by exactly one, inputs A, B, and C_{IN} were used to drive a three-layer network of widths $(2, 3, 2)$, producing S and C_{OUT}. The discretized gate network (Fig. 2b) was translated to Verilog and validated with testbenches, confirming full-adder behavior; it was then mapped directly to a DB layout using the proposed tile library.

To validate manufacturability of the resulting DB layout, three logic gates were fabricated with atomic precision using low temperature ($4.5\,\mathrm{K}$) STM on degenerately arsenic-doped, hydrogen-passivated Si(100) with a 2×1 surface reconstruction. The fabricated layout is shown in Fig. 3 composited onto the trained full-adder. DBs were patterned by applying voltage pulses via an ultra-sharp STM tip to break hydrogen-silicon bonds [31], with correction capabilities via hydrogen-functionalized tips [32]. In Fig. 3, DBs appear as circular features; dark pits are neutral missing dimers which do not interact with DBs electrostatically, while bright features are stray DB defects which can be erased when manufacturing production devices or mitigated using defect-aware gate redesign [33]. Together, these results substantiate the co-design premise and trace an end-to-end path from learned planar gate networks to physical fabrication.

VI. Experiments

With fabrication-validated DB primitives confirming the feasibility of the planar gate library, an experimental evaluation was conducted to benchmark BitPlanarNet on representative ML tasks. As image-classification tasks, MNIST [34] and CIFAR-10 [35] were chosen to establish comparability with Petersen *et al.*'s evaluation of DLGNs [8], with no image augmentation applied in either case. These datasets are modest by current vision-benchmark standards, but they remain the canonical image tasks used in the original DLGN work and therefore provide the clearest baseline for measuring the accuracy cost of strict planarity. Since BitPlanarNet

TABLE I

BITPLANARNET IMAGE CLASSIFICATION ACCURACY.

DATASET	PARAMETERS	l	D	r	ACCURACY
MNIST	44 880	6	10	1	96.75 %
	376 320	6	80	1	97.28 %
CIFAR-10	491 520	4	10	4	52.88 %
	1 228 800	4	20	5	55.11 %
	1 966 080	4	20	8	56.23 %
	3 932 160	4	20	16	57.51 %

l = layer count, D = input duplication, r = bits-per-channel.

operates on binary inputs, the floating-point image data had to be mapped to binary values; this work adopts the binarization strategies from [8]: for MNIST, any grayscale value >0 was mapped to logic 1; for CIFAR-10, unary encoding was used, which compared each color channel against r fixed thresholds to yield r bits per channel (e.g., if $r = 3$, each color channel is thresholded at $\{0.25, 0.5, 0.75\}$). This binarization further distinguishes the study from conventional floating-point CNN pipelines, so the central experimental question is not absolute state-of-the-art accuracy, but whether one-pass planar synthesis can be achieved without a disproportionate loss relative to unconstrained DLGNs.

Hyperparameters were selected via human-guided randomized sweeps facilitated by Ray Tune [36] and scheduled with the asynchronous successive halving algorithm (ASHA) [37]. The input duplication factor D (Section IV-C) was set as multiples of 10 so that each class-specific output block can observe all inputs at least once. Because adjacent layers must differ by exactly one neuron, widths cannot match those in [8] exactly; l and r were adjusted as needed to obtain comparable overall sizes. This requirement also entails that the actual layer dimension differs per layer, but those variations are insignificant compared to the total parameter count. Because learned input assignment was memory intensive, it was only enabled for MNIST, while randomized assignments were used for CIFAR-10. Training used a 90 %/10 % train/validation split, ran for up to 100 epochs (with early stopping under ASHA), validated in argmax mode every 10 epochs, and reported test accuracy from the best validation checkpoint. AdamW [38] was used with decoupled weight decay to stabilize gate selection under the entropy penalty. All three DB primitive libraries were included in the hyperparameter search.

Results in Table I summarize the best test accuracies achieved for each dataset and model size tested for BitPlanarNet. On MNIST, BitPlanarNet remains close to Petersen *et al.*'s DLGN results [8] at similar parameter counts at ≈ 1 p.p. delta, suggesting that strict planarity does not preclude competitive accuracy on simpler datasets. On CIFAR-10, whose larger $32\times32\times3$ input volume stresses long-range feature mixing, BitPlanarNet remains within ≈ 5 p.p. of unconstrained DLGNs while guaranteeing planar connectivity. In future work, enabling learned input assignment to place correlated features in proximity, together with increased network depth to enlarge the receptive field, is expected to narrow the remaining delta.

TABLE II

ABLATION STUDY OF BITPLANARNET ON MNIST 45K.

INPUT ASSIGNMENT	$\mathcal{G}_{16}$	$\mathcal{G}_{20}$	$\mathcal{G}_{34}$
Learned (baseline)	95.25 %	95.87 %	**96.15 %**
Deterministic input assignments			
Spiral	94.54 %	94.92 %	**95.33 %**
Hilbert	93.96 %	**94.99 %**	94.80 %
Column-major	94.38 %	**94.97 %**	94.76 %
Morton	94.08 %	94.40 %	**94.75 %**
Serpentine	93.82 %	94.21 %	**94.36 %**
Row-major	93.68 %	94.25 %	**94.32 %**
Randomized input assignments (10 runs)			
Random (avg.)	94.34 %	94.74 %	**95.02 %**
Random (min.)	94.03 %	94.18 %	**94.77 %**
Random (max.)	94.53 %	95.02 %	**95.38 %**
Ablate $\mathcal{L}_{\mathrm{ent}}$ $(\lambda_S = 0)$			
Learned	51.21 %	91.01 %	**92.60 %**

Learning rate $= 0.3$, $\tau = 5$ (Eq. (2)), $\sigma = 1.6$ (Eq. (1)), $\lambda_S = 10^{-2}$ (Eq. (5)), and weight decay $= 10^{-4}$.

A targeted ablation on the 44 880-parameter MNIST model probes input assignment, entropy regularization, and primitive-library expressiveness. Table II shows that learned assignment performs best, while deterministic traversals and random assignments remain within roughly 1–2 p.p., indicating a modest but consistent benefit from learning the input order. Removing $\mathcal{L}_{\mathrm{ent}}$ causes the largest degradation, while $\mathcal{G}_{34}$ consistently outperforms $\mathcal{G}_{16}$ across settings, underscoring the value of entropy regularization and expressive device-matched primitives.

Overall, the findings suggest that BitPlanarNet preserves much of the accuracy of unconstrained DLGN on MNIST while ensuring planar connectivity and direct mappability to a single device plane. On CIFAR-10, accuracy remains competitive given stricter wiring locality, with identified opportunities for improvement in future work. As the objective is hardware-realizable layouts, this work prioritizes accuracy and planarity over wall-clock training time. The source code for BitPlanarNet is available in the public repository [39].

VII. CONCLUSION

This work introduces BitPlanarNet, a learning-to-layout framework that reframes training as one-pass synthesis and provides a practical bridge from high-level task objectives to atomically precise planar circuitry. By adapting differentiable logic gate networks to planar constraints, BitPlanarNet combines native support for $2\to2$ logic primitives with entropy-regularized primitive selection and input assignment strategies to enable ML-based one-pass synthesis that produces manufacturable planar networks from task objectives. This premise was validated on atomic-scale computing through the design of a BitPlanarNet-specific primitive library, the one-to-one mapping of a trained full adder to a dot-accurate silicon DB layout, and the fabrication of a representative sub-layout via STM. The ability to manufacture circuits with atomic precision is a longstanding challenge; the DB structures realized in this work demonstrate state-of-the-art

lithography and an end-to-end path from a trained network to atomically precise circuitry. As patterning and verification tools advance, BitPlanarNet's configurable dimensions allow layouts to scale with fabrication capabilities toward larger, defect-aware structures. On MNIST and CIFAR-10, BitPlanarNet remains within ≈ 1 p.p. and 5 p.p. of DLGNs at comparable model sizes, showing that the proposed planar constraints preserve competitive accuracy on the benchmark scales established for this class of logic networks while enabling direct layout realizability for planar hardware.

Future work includes incorporating recent DLGN training advances [24], [25], scaling learned input assignment and deeper planar stacks, pruning post-training layouts to avoid fabricating redundant branches, and targeting additional planar platforms such as silicon photonics [2], photonic crystals, and microfluidic logic [3], as well as sensing and edge-classification workloads. By reframing training as one-pass synthesis, BitPlanarNet provides a direct route from task objectives to manufacturable, technology-matched, and scalable planar gate layouts on emerging planar technologies.

Acknowledgment

This work used Anthropic and OpenAI LLMs for coding and limited writing assistance. All outputs were verified by the authors, who assume full responsibility for the content.

References

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

[2] F. P. Sunny et al., "A Survey on Silicon Photonics for Deep Learning," *J. Emerg. Technol. Comput. Syst.*, vol. 17, no. 4, 61:1–61:57, 2021.

[3] M. Prakash and N. Gershenfeld, "Microfluidic bubble logic," *Science (New York, N.Y.)*, vol. 315, no. 5813, pp. 832–835, 2007.

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

[5] F. Sill Torres et al., "On the impact of the synchronization constraint and interconnections in quantum-dot cellular automata," *Microprocessors and Microsystems*, vol. 76, p. 103 109, 2020.

[6] 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.

[7] M. A. Al-Qadasi et al., "Scaling up silicon photonic-based accelerators: Challenges and opportunities," *APL Photonics*, vol. 7, no. 2, p. 020 902, 2022.

[8] F. Petersen et al., "Deep Differentiable Logic Gate Networks," in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., vol. 35, Curran Associates, Inc., 2022, pp. 2006–2018.

[9] 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.

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

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

[12] S.-S. Ahmadpour et al., "An Energy-Aware Nano-Scale Design of Reversible Atomic Silicon based on Miller Algorithm," *IEEE Design & Test*, pp. 1–1, 2023.

[13] 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, 2024, pp. 91–96.

[14] 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.

[15] J. Drewniok et al., "QuickCell: Fast Automatic Design of Standard Cells for Silicon Dangling Bond Logic," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1–1, 2025.

[16] 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.

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

[18] 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, 2020, pp. 134–139.

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

[20] N. P. Jouppi et al., *TPU v4: An Optically Reconfigurable Supercomputer for Machine Learning with Hardware Support for Embeddings*, arXiv: 2304.01433 [cs.AR], 2023.

[21] F. Petersen et al., "Convolutional Differentiable Logic Gate Networks," in *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[22] K. Fojcik, R. Zioma, and J. Armaitis, *LILogic Net: Compact Logic Gate Networks with Learnable Connectivity for Efficient Hardware Deployment*, arXiv: 2511.12340 [cs.LG], 2025.

[23] S. Bührer et al., "Recurrent Deep Differentiable Logic Gate Networks," in *Proceedings of the 2nd International Workshop on Edge and Mobile Foundation Models*, Hong Kong China: ACM, 2025, pp. 31–36.

[24] S. Yousefi et al., *Mind the Gap: Removing the Discretization Gap in Differentiable Logic Gate Networks*, arXiv: 2506.07500 [cs], 2025.

[25] L. Rüttgers et al., *Light Differentiable Logic Gate Networks*, arXiv: 2510.03250 [cs], 2025.

[26] A. T. L. Bacellar et al., "Differentiable weightless neural networks," in *Proceedings of the 41st International Conference on Machine Learning*, ser. ICML'24, Vienna, Austria: JMLR.org, 2024.

[27] H. Sagan, *Space-Filling Curves* (Universitext). New York, NY: Springer New York, 1994.

[28] G. M. Morton, "A computer oriented geodetic data base; and a new technique in file sequencing," International Business Machines Co. Ltd., Ottawa, Ontario, Canada, Research Report, 1966.

[29] G. Mena et al., *Learning Latent Permutations with Gumbel-Sinkhorn Networks*, arXiv: 1802.08665 [stat.ML], 2018.

[30] 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, 2023, pp. 872–877.

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

[32] 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, 2017.

[33] 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, 2024, pp. 30–35.

[34] L. Deng, "The MNIST Database of Handwritten Digit Images for Machine Learning Research [Best of the Web]," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[35] A. Krizhevsky, "Learning Multiple Layers of Features from Tiny Images," University of Toronto, Tech. Rep., 2009.

[36] R. Liaw et al., *Tune: A Research Platform for Distributed Model Selection and Training*, arXiv: 1807.05118 [cs.LG], 2018.

[37] L. Li et al., "A System for Massively Parallel Hyperparameter Tuning," in *Proceedings of Machine Learning and Systems*, I. Dhillon, D. Papailiopoulos, and V. Sze, Eds., vol. 2, 2020, pp. 230–246.

[38] I. Loshchilov and F. Hutter, *Decoupled Weight Decay Regularization*, arXiv: 1711.05101 [cs.LG], 2019.

[39] S. S. H. Ng, *BitPlanarNet source code*, https://github.com/samuelshng/bitplanarnet, 2026.