PowerCube: A Formula-in-the-Loop Framework for Cross-Stage Power Map Prediction

Yuxiang Zhao[1], Jing Mai[2], Zishu Li[1], Haoyi Zhang[1],
Jincheng Lou[1], Qing He[3], Runsheng Wang[1, 4, 5], Yibo Lin[1, 4, 5 *]

[1]*School of Integrated Circuits, Peking University*, Beijing, China
[2]*School of Computer Science, Peking University*, Beijing, China
[3]*College of Electronic and Information Engineering, Tongji University*, Shanghai, China
[4]*Beijing Advanced Innovation Center for Integrated Circuits*, Beijing, China
[5]*Institute of Electronic Design Automation, Peking University*, Wuxi, China

Email: {yuxiangzhao, hy.zhang, jinchenglou}@stu.pku.edu.cn, {jingmai, r.wang, yibolin}@pku.edu.cn,
cangxige112@gmail.com, qhe@tongji.edu.cn

[*]Co-corresponding authors

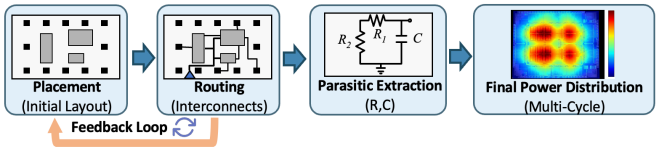
***Abstract*—Accurate early-stage power estimation is critical in modern IC design, yet existing cross-stage predictors yield only a single total power value, lacking granular 2D maps for localized optimization. Thus, we propose PowerCube, the first cross-stage prediction framework dedicated to fine-grained 2D power maps. To capture complex post-routing variations while preserving physical laws, our novel "formula-in-the-loop" architecture decouples transient physical parameter inference from deterministic power computation. First, a Spatial-Signal Dual-Branch Network calibrates early layout features into precise load and slew anchors. These anchors drive a differentiable calculation engine (Diff-LUT) to evaluate internal power, combined with exact analytical equations to compute switching and leakage components. Subsequently, a Synergistic Module bridges these localized metrics with layout-scale contexts to synthesize high-fidelity maps. Evaluated on ASAP7 7nm across 8 designs, PowerCube eliminates Out-of-Memory bottlenecks on massive layouts (e.g., 1.5-million-cell *nvdla-large*) and achieves an average 414× runtime speedup over commercial flows. Furthermore, it outperforms baselines by improving average R^2 by 26.44% and reducing NMAE by 28.72%, delivering a fast, highly scalable, and sign-off-accurate solution.**

***Index Terms*—Power prediction, Deep learning, Differentiable modeling**

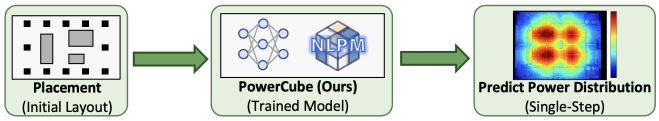
I. Introduction

Power has emerged as a primary bottleneck in modern IC design. The "Power Wall" inherently restricts the scalability of AI architectures, while stringent energy constraints for edge-deployed Large Language Models (LLMs) severely limit their deployment scenarios. Beyond performance, uneven power distribution poses severe reliability risks, such as transient IR-drop and thermal hotspots, which can significantly shorten a chip's operational life.

Consequently, early-stage power prediction is vital to enable "Shift-Left" design strategies [1]. However, it faces a fundamental paradox: initial estimations lack post-routing parasitics and Clock Tree Synthesis (CTS) data, whereas traditional post-layout simulations demand computationally prohibitive runtimes (often spanning tens of hours) that hinder iterative design space exploration. While recent Machine Learning (ML) frameworks, such as PowPrediCT [2] and ATLAS [3], have advanced cross-stage power modeling, they suffer from two critical limitations. As summarized in Table I, prior works predominantly predict a single total power value and operate as end-to-end black boxes devoid of physical laws. These coarse assessments completely mask localized spatial heterogeneity, failing to support granular hotspot mitigation.



(a) Classic power simulation flow.



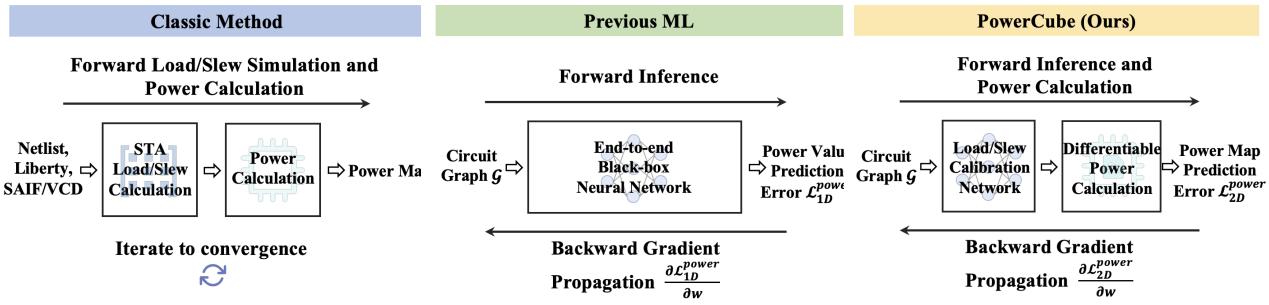
(b) Our PowerCube framework.

Fig. 1. **Comparison of power evaluation flows.** (a) Conventional simulation relies on time-consuming routing and extraction, causing long feedback loops. (b) PowerCube enables single-step power map prediction directly from initial placement.

To address these critical challenges, we propose **PowerCube**, the first cross-stage power map prediction framework driven by a novel physically-informed methodology. Figure 1 illustrates

TABLE I
Comparison of representative ML-based power models.

Power Models	Applied Stage	Formula-in-the-Loop	Target Fine-Grained 2D Map
PRIMAL [4]	RTL	No	No
APOLLO [5]			
SNS [6]			
MasterRTL [7]			
ATLAS [3]	Netlist		
PowPrediCT [2]	Layout		
PowerCube (Ours)	**Layout**	**Yes**	**Yes**



(a) Classic Simulation Method (b) Previous end-to-end black-box Method (c) Our PowerCube Framework

Fig. 2. **Method Comparison**. (a) Classic iterative Power simulation. (b) Previous end-to-end black-box ML approaches predicting 1D power values. (c) Our PowerCube framework featuring a differentiable power calculation module for 2D power map prediction.

the classic iterative power simulation flow alongside our PowerCube architecture. As illustrated in Figure 2, our framework introduces a differentiable power calculation module to replace the pure neural network. In contrast to prevailing black-box approaches, PowerCube fundamentally decouples the inference of pin-level electrical parameters from layout-scale post-routing effects, resolving both memory and runtime bottlenecks. Its superiority manifests in three core aspects:

Firstly, a "Formula-in-the-Loop" Paradigm. Traditional analysis is bottlenecked by the exhaustive physical design flow required to extract exact load and slew. PowerCube abandons end-to-end black-box regression. Instead, it employs neural networks exclusively to cross-stage *calibrate* these early-stage physical parameters, while retaining deterministic industrial analytical equations for the final power derivation. This establishes a highly interpretable, physics-compliant estimation paradigm.

Secondly, Physical Calibration via SSDN. To mirror authentic physical signal propagation, we propose the Spatial-Signal Dual-Branch Network (SSDN). To handle large-scale layouts efficiently, a lightweight GCN first calibrates the capacitive load using early layout priors. Subsequently, a topology-aware GNN computes signal slew via layer-wise DAG aggregation. Crucially, by explicitly injecting the calibrated load into the slew branch, SSDN strictly honors their underlying physical co-determination.

Thirdly, Layout-Scale Synergy and Diff-LUT. We introduce a fully differentiable calculation engine coupled with a Spatial-Routing Synergistic Module. Unlike prior works that suffer from massive memory overheads via global attention, our **Diff-LUT** performs $\mathcal{O}(1)$ exact differentiable interpolation exclusively on unique cell LUTs, enabling lightning-fast inference. Concurrently, analytical models explicitly derive switching and leakage power. Most importantly, to bridge the inherent gap between isolated pin-level computations and layout-scale post-routing variations (e.g., routing congestion and CTS buffer clustering), these localized metrics are seamlessly fed into our synergistic U-Net module to synthesize high-fidelity 2D power maps.

Our main contributions are summarized as follows:

- We propose **PowerCube**, the first cross-stage prediction framework dedicated to fine-grained 2D power maps. It pioneers a novel "formula-in-the-loop" architecture that fundamentally decouples the iterative inference of transient physical parameters from deterministic power computations, thereby strictly preserving exact physical laws.
- We design a highly efficient Differentiable Power Calculation Engine. Driven by the cross-stage physical calibration of SSDN, our $\mathcal{O}(1)$ Diff-LUT seamlessly embeds discrete standard-cell Non-Linear Power Models (NLPM) into the gradient flow, eliminating the severe memory bottlenecks of prior attention-based methods.
- We introduce a Spatial-Routing Synergistic Module that implicitly captures post-routing parasitics and CTS variations, successfully mapping localized physical anchors to global layout topologies.
- Evaluated on the ASAP7 7nm dataset across 8 industrial designs, PowerCube achieves a paradigm shift in both speed and spatial accuracy. It inherently resolves the Out-of-Memory (OOM) failures on massive designs (e.g., 1.5-million-cell *nvdla-large*). By bypassing the complete flow, it delivers an average runtime speedup of 414× (up to 639× peak). Furthermore, it improves the average R^2 metric by 26.44% and significantly reduces the NMAE by 28.72% over state-of-the-art baselines.

II. Methodology

The overall pipeline of PowerCube is illustrated in Figure 3. We first construct a directed circuit graph representing early-stage layouts (Sec II-A). The Spatial-Signal Dual-Branch Network (SSDN, Sec II-B) then calibrates cross-stage pin-level load and slew. These physical anchors then drive a fully differentiable calculation engine (Sec II-C), which combines Diff-LUT and analytical equations to execute deterministic power computations. Finally, to capture complex post-routing variations, a Spatial-Routing Synergistic Module (Sec II-D) bridges these localized metrics with spatial layout contexts, synthesizing the ultimate high-fidelity 2D power maps.

A. Circuit Graph Construction

To faithfully capture the exact physical mechanisms of circuit signal propagation, we formulate the circuit as a fine-grained, pin-centric directed graph, inherently mirroring the structural paradigm of Static Timing Analysis (STA). Nodes $\mathcal{V}$ represent pins, each initialized with a multi-dimensional feature vector $\mathbf{h}_v \in \mathbb{R}^d$. Edges $\mathcal{E} = \mathcal{E}_{net} \cup \mathcal{E}_{cell}$ denote net connections and internal cell timing arcs, enabling the network to directly trace

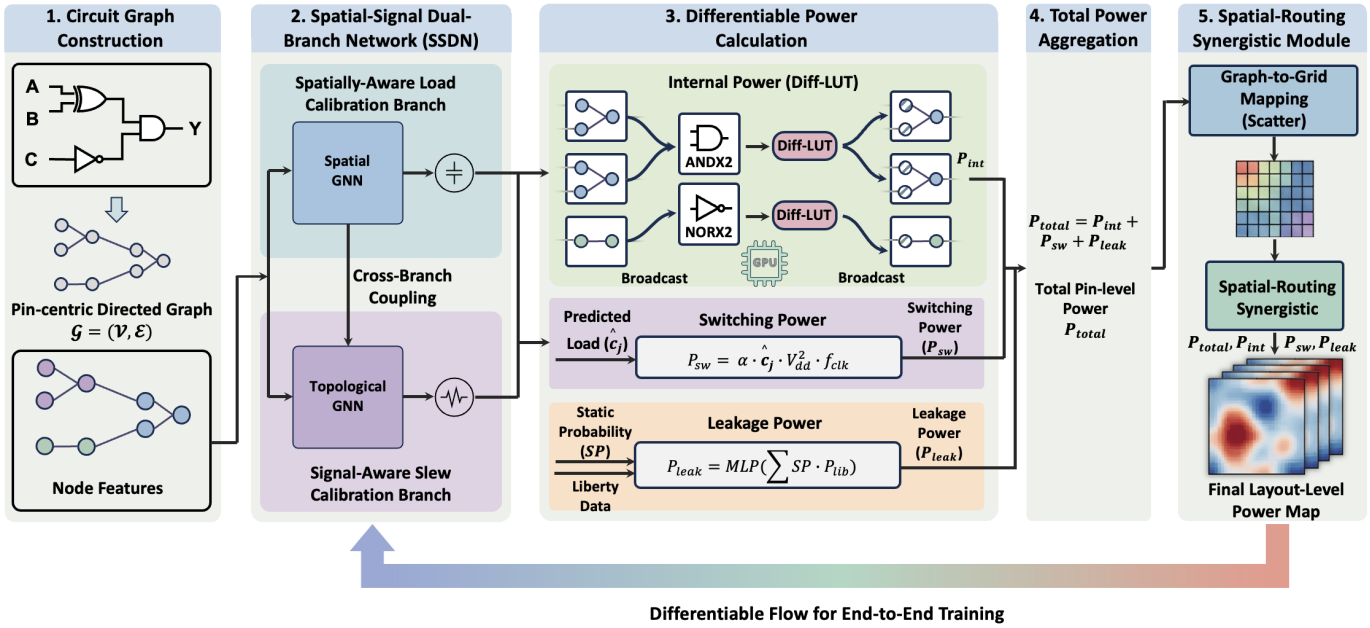


Fig. 3. **Overall architecture of the proposed PowerCube framework. The pipeline consists of five key stages:** (1) Circuit Graph Construction; (2) Spatial-Signal Dual-Branch Network (SSDN);(3) Differentiable Power Calculation; (4) Total Power Aggregation; and (5) Power Map Refinement Module.

TABLE II
DETAILED DEFINITIONS OF PIN-LEVEL INITIAL NODE FEATURES ($\mathbf{h}_v$).

Category	Notation	Description
Spatial	$\mathbf{p}_v = (x, y)$	Normalized 2D layout coordinates on the floorplan.
Activity	$\mathbf{s}_v = (\alpha, SP)$	Toggle rate (α) & static probability (SP) from zero-delay simulation.
Electrical	$\mathbf{e}_v = (S_{init}, C_{init})$	Early estimates of input slew and capacitive load via parasitics.
Topological	N_{fo}	Fan-out count, implicitly guiding routing congestion prediction.

signal evolution and spatial topological dependencies (Figure 3 (1)).

As detailed in Table II, the initial node attribute $\mathbf{h}_v$ systematically fuses spatial, layout-independent functional activities, and coarse early-stage electrical priors. Specifically, these heterogeneous features are channel-wise concatenated to form the explicit node representation (i.e., $\mathbf{h}_v = \mathbf{p}_v \parallel \mathbf{s}_v \parallel \mathbf{e}_v \parallel N_{fo}$), providing the necessary physical context for the downstream cross-stage calibration.

B. Spatial-Signal Dual-Branch Network (SSDN)

Internal and switching power are inherently contingent upon capacitive load and signal transition time (slew). However, since early layout parasitics completely lack actual routing parasitics and CTS, early-stage proxy metrics are notoriously inaccurate. Therefore, SSDN fundamentally functions as a highly non-linear *physical feature calibrator* (Figure 3 (2)), mapping coarse placement-stage estimates to precise sign-off pin metrics.

1) Spatially-Aware Load Calibration: A Graph Convolutional Network (GCN) processes the layout graph $\mathcal{G}$ to capture spatial pin localities and implicitly compensate for missing realistic routing parasitic capacitance. The node attributes, initialized as $\mathbf{h}_i^{(0)} = \mathbf{h}_i$, are updated iteratively through the message passing mechanism:

$$\mathbf{h}_i^{(l+1)} = \sigma\left(\sum_{j \in \mathcal{N}(i) \cup \{i\}} \frac{1}{\sqrt{\tilde{d}_i \tilde{d}_j}} \mathbf{W}^{(l)} \mathbf{h}_j^{(l)}\right), \quad (1)$$

where $\mathcal{N}(i)$ denotes the set of immediate neighbor nodes of node i, and $\tilde{d}_i = |\mathcal{N}(i)| + 1$ represents the degree of node i with an added self-loop. $\mathbf{W}^{(l)}$ is the trainable weight matrix at the l-th layer ($l \in \{0, 1, \ldots, L-1\}$), and $\sigma(\cdot)$ denotes a non-linear activation function (e.g., ReLU). After stacking L layers, the final node embeddings $\mathbf{f}_i^{load} = \mathbf{h}_i^{(L)}$ are fed into a multilayer perceptron (MLP) regression head to predict the calibrated scalar load $\hat{c}_i = \text{MLP}_{load}(\mathbf{f}_i^{load})$. Simultaneously, $\mathbf{f}_i^{load}$ is explicitly forwarded to the slew branch to serve as essential physical constraints.

2) Signal-Aware Slew Calibration: To emulate authentic signal propagation and enforce STA causality, nodes are evaluated via a topological sort π on the DAG representation of $\mathcal{G}$. A node's latent slew $\mathbf{s}_i$ fuses its upstream context, intrinsic attributes, and the calibrated load embeddings:

$$\mathbf{s}_i = \text{MLP}_{slew}\left(\text{AGG}\left(\{\mathbf{s}_j \mid j \in \mathcal{N}_{in}(i)\}\right) \parallel \mathbf{h}_i \parallel \mathbf{f}_i^{load}\right) \quad (2)$$

A linear projection then maps this latent representation to the continuous scalar slew $\hat{s}_i = \mathbf{W}_{reg}\mathbf{s}_i$.

Crucially, the calibrated $\hat{c}_i$ and $\hat{s}_i$ now serve as explicit physical variables driving the downstream differentiable calculation engine to generate physically compliant, localized power anchors.

C. Differentiable Power Calculation

We formulate a decoupled, fully differentiable engine (Figure 3 (3)) that inherently mirrors industrial sign-off logic, guaranteeing end-to-end physical interpretability instead of relying on black-box regression.

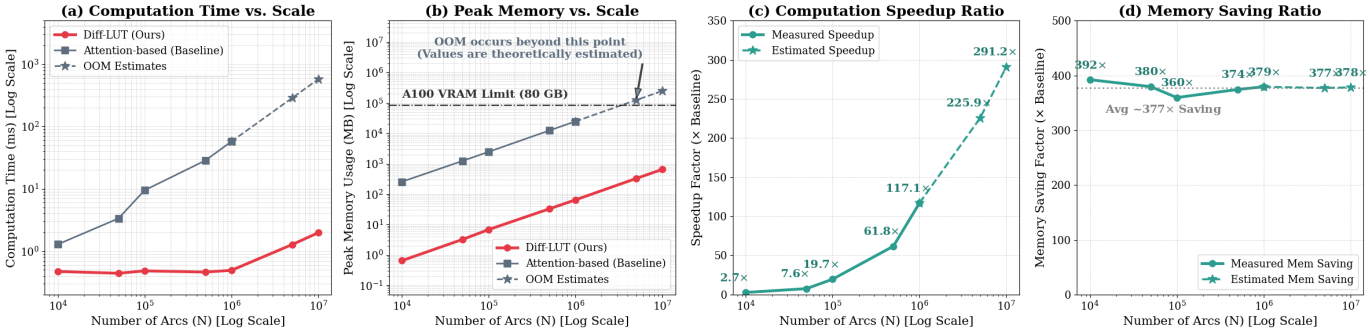
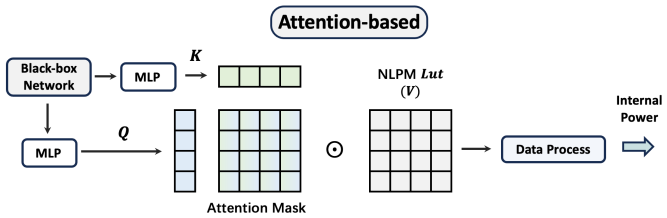
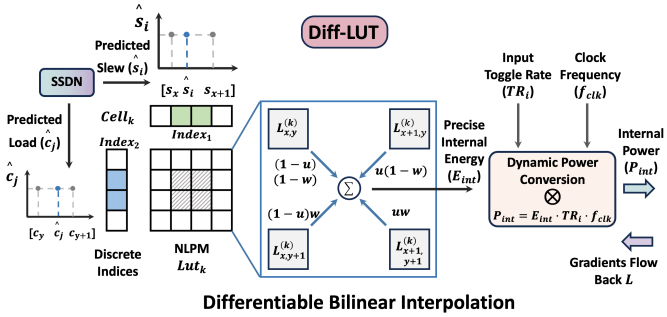


Fig. 4. **Scalability of Diff-LUT vs. attention baseline.** (a) Computation time and (b) peak memory, showing the baseline encountering OOM on an 80GB GPU. Diff-LUT efficiently scales, achieving (c) up to **291**× speedup and (d) ∼**377**× average memory savings.



(a) Previous attention-based method.



(b) Our Diff-LUT method.

Fig. 5. **Comparison of internal power calculation.** (a) Attention-based approximation. (b) Proposed Diff-LUT with exact, differentiable physical modeling.

1) ***Internal Power via Diff-LUT****:* Internal power depends non-linearly on slew and load, strictly defined via discrete 2D NLPM $\mathbf{L}^{(k)} \in \mathbb{R}^{m\times n}$ in standard cell libraries . To enable gradient-based training without breaking physical laws, our *Diff-LUT* performs differentiable exact interpolation. For an arc with calibrated input slew $\hat{s}_i$ and output load $\hat{c}_j$, Diff-LUT first clamps predictions within library bounds, locates intervals $[s_x, s_{x+1}]$ and $[c_y, c_{y+1}]$, and computes local coordinates:

$$u = \frac{\hat{s}_i - s_x}{s_{x+1} - s_x}, \quad w = \frac{\hat{c}_j - c_y}{c_{y+1} - c_y} \tag{3}$$

The precise internal energy E_{int} is derived via bilinear interpolation:

$$\begin{aligned} E_{int} = (1-u)(1-w)\mathbf{L}^{(k)}_{x,y} + u(1-w)\mathbf{L}^{(k)}_{x+1,y} \\ + (1-u)w\mathbf{L}^{(k)}_{x,y+1} + uw\mathbf{L}^{(k)}_{x+1,y+1} \end{aligned} \tag{4}$$

Dynamic internal power is then $P_{int} = E_{int} \cdot \alpha_i \cdot f_{clk}$, where α_i is the input toggle rate.

Gradient Flow Analysis. While discrete search operations for bounding indices (x and y) are inherently non-differentiable (exhibiting zero gradients almost everywhere), the Diff-LUT engine remains *sub-differentiable* overall. Specifically, these indices merely act as selectors for constant empirical values from $\mathbf{L}^{(k)}$. During backpropagation, gradients flow to upstream SSDN predictions ($\hat{s}_i$ and $\hat{c}_j$) exclusively through the continuous fractional coordinates u and w of the bilinear kernel. Analogous to Spatial Transformer Networks [8] and differentiable EDA frameworks like DREAMPlace [9], treating discrete LUT entries as constants enables end-to-end training via the gradients of interpolation weights. This ensures a smooth gradient flow, bridging discrete physical characterization and continuous neural optimization.

TABLE III
COMPARISON OF EFFICIENCY, MEMORY FOOTPRINT, AND INTERPRETABILITY.

Metric	Previous Methods	Diff-LUT (Ours)
Efficiency	$\mathcal{O}(m \times n)$ per pin	$\mathcal{O}(1)$ **per pin**
Memory	$\mathcal{O}(N_{\text{arc}})$	$\mathcal{O}(N_{\text{cell\_type}})$
Interpretability	Heuristic approx.	**Exact physical logic**

Comparison with Previous Methods. As depicted in Figure 5, recent ML-based models [2] employ heuristic attention mechanisms for LUT approximation, computing global weighted sums over grids for every cell arc. In contrast, our GPU-accelerated Diff-LUT mathematically replicates exact sign-off interpolation, accessing only four adjacent points. This drops inference complexity to absolute $\mathcal{O}(1)$ per pin, entirely eliminating the massive $\mathcal{O}(N_{arc})$ memory bottleneck caused by prior attention models (validated in Table III and Figure 4).

2) ***Switching and Leakage Power****:* Unlike internal power, switching power P_{sw} exhibits an explicit closed-form expression. We integrate the layout-dependent $\hat{c}_j$ from SSDN with the layout-independent α_j:

$$P_{sw,j} = \frac{1}{2}\alpha_j \cdot \underbrace{\hat{c}_j}_{\textit{calibrated}} \cdot V_{dd}^2 \cdot f_{clk} \tag{5}$$

It is paramount to note that α_j is rapidly extracted via early-stage gate-level logic simulation, whereas the layout-dependent parasitic $\hat{c}_j$ is exactly what SSDN bridges across stages. This strictly linear formulation backpropagates a physically-grounded constant gradient ($\frac{\partial P_{sw,j}}{\partial \hat{c}_j}$) to provide direct supervision for the SSDN load branch.

Leakage power P_{leak} is a static metric driven by the static probability (SP) of input states (also from early logic

TABLE IV
PERFORMANCE COMPARISON OF POWERCUBE VERSUS POWPREDICT ACROSS EIGHT HARDWARE DESIGNS.

Architecture	Cells	Metric	Internal Power			Switching Power			Leakage Power			Total Power			
			ATLAS§	PowPrediCT§	Ours	ATLAS§	PowPrediCT§	Ours	ATLAS§	PowPrediCT§	Ours	ATLAS§	PowPrediCT§	Ours	*Improv.*(%)
zero-riscy	35,969	Pearson's r	0.82	0.59	**0.97**	0.85	0.58	**0.94**	0.84	0.60	**0.98**	0.80	0.56	**0.95**	**18.75%**
		R^2	0.49	0.21	**0.94**	0.11	0.17	**0.89**	−0.32	0.29	**0.96**	0.37	0.12	**0.90**	**143.24%**
		SSIM	0.28	0.51	**0.95**	0.29	0.53	**0.91**	−0.06	0.53	**0.96**	−0.01	0.57	**0.94**	**64.91%**
		NMAE	0.86	0.76	**0.18**	1.28	0.80	**0.24**	1.61	0.70	**0.16**	0.97	0.83	**0.23**	**72.29%**
RISCY	46,184	Pearson's r	0.78	0.76	**0.97**	0.84	0.75	**0.95**	0.83	0.77	**0.98**	0.79	0.71	**0.95**	**20.25%**
		R^2	−0.05	0.51	**0.94**	−0.02	0.48	**0.90**	−0.06	0.58	**0.95**	−0.04	0.41	**0.88**	**114.63%**
		SSIM	0.05	0.51	**0.93**	0.09	0.56	**0.90**	0.05	0.44	**0.94**	0.21	0.60	**0.92**	**53.33%**
		NMAE	1.18	0.55	**0.18**	1.20	0.58	**0.22**	1.15	0.52	**0.16**	1.22	0.63	**0.24**	**61.90%**
RISCY-FPU	65,464	Pearson's r	0.66	0.68	**0.96**	0.73	0.66	**0.94**	0.72	0.71	**0.97**	0.50	0.62	**0.94**	**51.61%**
		R^2	0.06	0.11	**0.92**	0.21	−0.06	**0.88**	0.03	0.47	**0.94**	0.04	−0.47	**0.88**	**2100.00%**
		SSIM	0.13	0.45	**0.91**	0.21	0.47	**0.87**	0.15	0.47	**0.92**	0.35	0.53	**0.92**	**73.58%**
		NMAE	0.91	0.71	**0.20**	0.86	0.81	**0.24**	0.90	0.52	**0.16**	0.91	0.96	**0.24**	**73.63%**
Vortex-small	113,961	Pearson's r	0.31	0.69	**0.94**	0.41	0.68	**0.93**	0.38	0.70	**0.94**	0.29	0.65	**0.90**	**38.46%**
		R^2	−2.98	0.43	**0.87**	−2.13	0.41	**0.83**	−5.04	0.46	**0.89**	−3.37	0.31	**0.74**	**138.71%**
		SSIM	−0.12	0.48	**0.85**	0.31	0.50	**0.83**	0.33	0.52	**0.88**	0.29	0.50	**0.81**	**62.00%**
		NMAE	0.91	0.61	**0.27**	1.01	0.63	**0.31**	0.93	0.58	**0.25**	1.08	0.69	**0.39**	**43.48%**
nvdla-small	270,072	Pearson's r	0.46	0.69	**0.89**	0.52	0.68	**0.89**	0.48	0.71	**0.90**	0.42	0.63	**0.85**	**34.92%**
		R^2	0.16	0.36	**0.79**	−0.92	0.27	**0.79**	0.14	0.47	**0.81**	0.01	0.17	**0.71**	**317.65%**
		SSIM	0.18	0.45	**0.82**	−0.03	0.46	**0.83**	0.30	0.47	**0.84**	0.33	0.51	**0.83**	**62.75%**
		NMAE	1.06	0.70	**0.36**	1.86	0.77	**0.36**	0.89	0.63	**0.33**	1.14	0.82	**0.42**	**48.78%**
openc910-1	754,981	Pearson's r	0.60	0.57	**0.86**	0.66	0.49	**0.83**	0.61	0.57	**0.89**	0.61	0.43	**0.83**	**36.07%**
		R^2	−0.43	−0.68	**0.73**	−2.23	−0.56	**0.63**	−3.12	0.20	**0.79**	−0.74	−2.45	**0.68**	**191.89%**
		SSIM	−0.06	0.17	**0.65**	0.27	0.21	**0.65**	−0.01	0.19	**0.68**	0.05	0.28	**0.80**	**185.71%**
		NMAE	0.81	0.83	**0.29**	1.22	0.80	**0.36**	1.56	0.52	**0.25**	0.80	1.25	**0.33**	**58.75%**
Vortex-large	1,018,221	Pearson's r	0.62	−0.37	**0.87**	0.65	−0.38	**0.85**	0.65	−0.38	**0.89**	0.56	−0.33	**0.82**	**46.43%**
		R^2	0.20	−937.31	**0.75**	−0.43	−1285.67	**0.72**	−0.01	−812.78	**0.79**	0.16	−673.42	**0.64**	**300.00%**
		SSIM	0.32	0.01	**0.78**	0.28	0.01	**0.76**	0.37	0.01	**0.81**	0.42	0.01	**0.75**	**78.57%**
		NMAE	0.84	32.75	**0.37**	1.41	38.96	**0.40**	0.85	28.66	**0.33**	0.74	26.80	**0.47**	**36.49%**
nvdla-large	1,478,865	Pearson's r	OOM*		**0.87**	OOM*		**0.88**	OOM*		**0.88**	OOM*		**0.83**	N/A†
		R^2			**0.76**			**0.78**			**0.78**			**0.68**	
		SSIM			**0.75**			**0.76**			**0.76**			**0.77**	
		NMAE			**0.35**			**0.34**			**0.34**			**0.42**	

§ **We made minor modifications to the original ATLAS and PowPrediCT baselines to adapt them for power map prediction tasks.**
* **OOM**: **Out-of-memory error occurred during the baseline simulation due to circuit complexity.**
† **N/A**: **Improvement is not applicable because the baselines failed to produce valid references.**
‡ **The shading column (Ours) highlights our proposed PowerCube method.**
¶ **All percentage improvements in blue are evaluated relative to the highest-performing baseline (ATLAS or PowPrediCT) for each metric.**

simulation). It is modeled as the mathematical expectation across all valid states $\mathcal{S}$:

$$P_{leak} = \sum_{s \in \mathcal{S}} SP_s \cdot P_{lib,s} \tag{6}$$

Being independent of layout parasitics, it requires no gradient backpropagation but provides a crucial, physics-compliant static baseline for spatial maps.

D. Spatial-Routing Synergistic Module

Although the preceding SSDN and Diff-LUT engine execute deterministic computations to guarantee pin-level electrical accuracy, they evaluate individual instances purely based on graph connectivity, remaining void of actual 2D placement contexts. Consequently, these isolated metrics are agnostic to layout-scale post-routing variations, such as global routing congestion and CTS buffer clustering.

*1) **Physics-to-Spatial Projection**:* To explicitly construct the 4-channel physical baseline, we first define the comprehensive power feature vector for each cell v as $\mathbf{P}_v = [P_{int,v}, P_{sw,v}, P_{leak,v}, P_{total,v}]^\top$, where the total node power is derived as $P_{total,v} = P_{int,v} + P_{sw,v} + P_{leak,v}$. We then scatter these localized physical anchors onto a 2D layout grid. The initial multi-channel pre-map at the (x, y)-th GCell, denoted as the vector $\mathbf{M}_{init}(x, y) \in \mathbb{R}^4$, is computed by aggregating the power vectors of all cells falling within this spatial bin:

$$\mathbf{M}_{init}(x, y) = \sum_{v \in G_{x,y}} \mathbf{P}_v, \tag{7}$$

yielding the complete 4-channel physical baseline tensor $\mathbf{M}_{init} \in \mathbb{R}^{4 \times h \times w}$.

*2) **Spatial Synergy and Supervision**:* Backend optimizations inevitably perturb the initial spatial distribution (e.g., CTS heavily concentrates buffers around sequential elements). To implicitly model these cross-stage spatial dynamics, we concatenate M_{init} with auxiliary layout features F_{layout} (e.g., cell density, pin density, macro map). A synergistic U-Net deduces post-routing coupling from this fused representation:

$$M_{final} = \text{U-Net}(M_{init} \oplus F_{layout}; \theta) \tag{8}$$

The framework is supervised end-to-end using a pixel-wise L_1 loss against sign-off ground truth M^c_{GT}:

$$\mathcal{L} = \sum_{c \in \{int, sw, leak, total\}} \|M^c_{final} - M^c_{GT}\|_1 \tag{9}$$

This joint supervision ensures robustness against extreme power hotspots and implicitly regularizes the network to respect underlying physical additivity ($P_{total} = P_{int} + P_{sw} + P_{leak}$) during backpropagation.

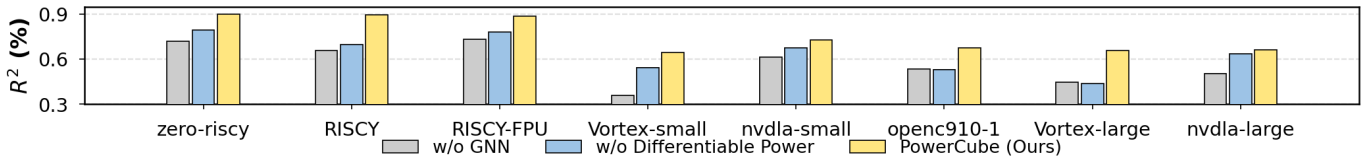


Fig. 6. **Ablation study.** The complete PowerCube consistently achieves the highest R^2 accuracy across all designs compared to its ablated variants.

TABLE V
RUNTIME COMPARISON BETWEEN INNOVUS AND POWERCUBE (OURS) IN SECONDS

Design	Innovus Full Flow				PowerCube (Ours)		
	CTS	Routing	PA*	Total	Pre.†	Infer.‡	Total
zero-riscy	32	466	115	613	10.45	0.35	10.80
RISCY	33	592	178	803	13.82	0.24	14.06
RISCY-FPU	35	787	274	1096	16.91	0.37	17.28
Vortex-small	37	1327	765	2129	31.24	0.63	31.87
nvdla-small	88	4182	3173	7443	92.15	0.94	93.09
openc910-1	130	7289	22511	29930	108.43	0.78	109.21
Vortex-large	165	8299	84693	93157	145.76	0.94	146.70
nvdla-large	211	12481	102276	114968	178.50	1.33	179.83
Average	**91**	**4428**	**26748**	**31267**	**74.66**	**0.70**	**75.36**

* **PA**: Power Analysis †**Pre.**: Data Preprocessing ‡**Infer.**: Inference

III. EXPERIMENT

A. Experimental Setup

We evaluate PowerCube on an NVIDIA A100 GPU using 8 diverse CircuitNet 2.0 [10] RTL designs. We generate 120 post-routing layouts (15 per design) via Cadence Innovus [11] on the ASAP7 7nm PDK [12], comprehensively sweeping five clock frequencies (100 to 1000 MHz) and three target utilizations. Ground truth maps (M_{GT}) are extracted via Voltus. All models are trained for 100 epochs (AdamW, LR=0.001) using strict leave-one-out cross-validation across the 8 designs to rigorously assess zero-shot generalization.

B. Evaluation Metrics

We assess the fidelity of 2D power maps using Pearson Correlation (r), Coefficient of Determination (R^2), Normalized Mean Absolute Error (NMAE), and Structural Similarity Index (SSIM). These metrics provide a quantitative basis for ensuring compliance with thermal limits and preventing IR-drop violations.

C. Main Results and Physical Insights

Table IV shows PowerCube universally outperforms PowPrediCT [2] and ATLAS [3]. Crucially, PowerCube scales effortlessly to the 1.5-million-cell *nvdla-large* design, whereas both baselines trigger OOM failures, proving our $\mathcal{O}(1)$ Diff-LUT successfully shatters standard GNN memory bottlenecks. On mid-sized designs, PowerCube consistently eclipses baselines; on *RISCY*, it achieves a 61.90% NMAE reduction and a 114.63% R^2 boost. Furthermore, on complex layouts (*openc910-1*, *Vortex-large*), baselines suffer severe structural collapse (negative R^2). Conversely, our "formula-in-the-loop" architecture robustly maintains SSIM ≥ 0.75 and $r \geq 0.82$ by anchoring predictions with exact physical laws.

D. Ablation Study

We systematically remove core components (Figure 6). **1) *w/o GNN (SSDN):*** Bypassing graph-based calibration degrades R^2 (e.g., an 0.24 absolute drop on *RISCY*), proving STA causality modeling is essential for dynamic load/slew dependencies. **2) *w/o Diff-Power (Diff-LUT):*** Replacing our differentiable engine with black-box MLP regression causes a 33% R^2 plunge on *Vortex-large*, confirming standard deep learning profoundly struggles to extrapolate non-linear power logic without explicit physical inductive biases.

E. Runtime Analysis and Scalability

Table V compares end-to-end execution times against the Innovus flow. Baseline sign-off Power Analysis explodes non-linearly on *nvdla-large* to over 28 hours (102,276 seconds), severely bottlenecking iteration cycles. In profound contrast, PowerCube's neural inference executes in merely 1.33 seconds. Overall, PowerCube delivers an average 414x (up to 639x peak) speedup across 8 designs.

Summary: By shattering memory and runtime bottlenecks while maintaining exact physical alignment, PowerCube proves that synergizing pin-level calibration, exact analytical formulas, and layout-scale contexts is non-negotiable for ultra-fast, sign-off-grade prediction.

IV. CONCLUSION

In this paper, we presents PowerCube, the first cross-stage framework dedicated to fine-grained 2D power map prediction. By introducing a "formula-in-the-loop" architecture, we fundamentally decouple neural parameter calibration from deterministic power computation. Our $\mathcal{O}(1)$ Diff-LUT engine embeds exact standard-cell physical constraints directly into the learning process, effectively overcoming the limitations of purely data-driven black-box models. Extensive evaluations on 8 industrial designs at the ASAP7 7nm node demonstrate that PowerCube resolves the OOM bottlenecks of prior baselines, seamlessly scaling to multimillion-gate layouts (e.g., *nvdla-large*). Furthermore, PowerCube establishes a new state-of-the-art by improving average R^2 by 26.44% and reducing NMAE by 28.72%, while delivering a 414× average runtime speedup over commercial sign-off tools. By bridging deep learning efficiency with physical rigor, PowerCube provides a highly scalable and robust solution for next-generation "Shift-Left" IC design methodologies.

ACKNOWLEDGMENTS

This work was supported in part by the National Key Research and Development Program of China under Grant 2021ZD0114702; in part by the Natural Science Foundation of Beijing, China, under Grant Z230002.

REFERENCES

[1] C. Z. e. a. Yun LIANG, "The shift-left design paradigm of eda: progress and challenges," *SCIENTIA SINICA Informationis*, no. 1, 2024.

[2] Y. Du, Z. Guo, X. Jiang, Z. Chai, Y. Zhao, Y. Lin, R. Wang, and R. Huang, "Powpredict: Cross-stage power prediction with circuit-transformation-aware learning," in *Proceedings of the 61st ACM/IEEE Design Automation Conference*, ser. DAC '24, 2024.

[3] W. Li, Y. Lu, W. Fang, J. Wang, Q. Zhang, and Z. Xie, "Atlas: A self-supervised and cross-stage netlist power model for fine-grained time-based layout power analysis," *arXiv preprint arXiv: 2508.12433*, 2025.

[4] Y. Zhou, H. Ren, Y. Zhang, B. Keller, B. Khailany, and Z. Zhang, "Primal: Power inference using machine learning," in *DAC*, 2019.

[5] Z. Xie *et al.*, "Apollo: An automated power modeling framework for runtime power introspection in high-volume commercial microprocessors," in *MICRO*, 2021.

[6] C. Xu, C. Kjellqvist, and L. W. Wills, "Sns's not a synthesizer: a deep-learning-based synthesis predictor," in *ISCA*, 2022.

[7] W. Fang, Y. Lu, S. Liu, Q. Zhang, C. Xu, L. W. Wills, H. Zhang, and Z. Xie, "Masterrtl: A pre-synthesis ppa estimation framework for any rtl design," in *ICCAD*, 2023.

[8] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, "Spatial transformer networks," in *Advances in neural information processing systems (NeurIPS)*, vol. 28, 2015.

[9] Y. Lin, S. Dhar, W. Li, H. Ren, B. Khailany, and D. Z. Pan, "DREAMPlace: Deep learning toolkit-enabled GPU acceleration for modern VLSI placement," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, vol. 40, no. 4, pp. 748–761, 2020.

[10] X. Jiang, Z. Chai, Y. Zhao, Y. Lin, R. Wang, R. Huang *et al.*, "Circuitnet 2.0: An advanced dataset for promoting machine learning innovations in realistic chip design environment," in *The Twelfth International Conference on Learning Representations*, 2024.

[11] "Cadence Innovus Implementation System," http://www.cadence.com.

[12] L. T. Clark, V. Vashishtha, L. Shifren, A. Gujja, S. Sinha, B. Cline, C. Ramamurthy, and G. Yeric, "Asap7: A 7-nm finfet predictive process design kit," *Microelectronics Journal*, 2016.