# Dynamic Supply Noise Aware Timing Analysis with JIT Machine Learning Integration

Yufei Chen* *Student Member, IEEE,* Zizheng Guo* *Student Member, IEEE,* Runsheng Wang *Member, IEEE,* Ru Huang *Fellow, IEEE,* Yibo Lin *Member, IEEE,* and Cheng Zhuo *Senior Member, IEEE*

***Abstract*—The incessant decrease in transistor size has led to reduced voltage noise margins and exacerbated power integrity challenges. This trend intensifies concerns about the efficacy of conventional static timing analysis (STA), which traditionally assumes a constant power supply level, often resulting in imprecise and overly conservative outcomes. To address this, this paper proposes a dynamic-noise-aware STA engine enhanced by just-in-time (JIT) machine learning (ML) integration. This approach employs the Weibull cumulative distribution function to accurately represent dynamic power supply noise (PSN). We perform gate-level characterization, assessing delay and transition time for each timing arc under variations in input transition time, output capacitance, and three PSN-aware parameters. The timing for each timing arc can then be predicted by a multilayer perceptron (MLP), trained with the characterization data. Finally, by incorporating JIT compilation techniques, we integrate trained MLP models into the STA engine, achieving both computational efficiency and flexibility. Experimental results show that the proposed method can accurately estimate the timing fluctuation due to dynamic PSN, with an average relative error of 4.89% for single-cell estimations and 6.27% for path delay estimations.**

## I. Introduction

With a scaling supply voltage and a continuously increasing current density, the gap between the operating voltage and the threshold voltage is narrowing. The circuit performance is then more susceptible to supply voltage fluctuation. Power supply noise (PSN) refers to the undesired fluctuations or variations in the voltage level of the power supply [1]. This type of noise can be categorized into two main types: IR drop and $Ldi/dt$ noise. IR drop refers to the low-frequency component of PSN, which results from current and resistance in the power delivery network (PDN). The scaling of interconnects leads to an increase in resistance and thus a more severe IR drop [2]. Additionally, the increased speed transients allowed by scaled transistor size result in higher $Ldi/dt$ noise due to the large parasitic inductance in the package and on-chip components [3]. These factors contribute to more complex power integrity issues that must be taken into consideration during circuit design and analysis to prevent unfavorable margins and identify optimization opportunities [4].

Static timing analysis (STA) is a technique used to validate the timing performance of a design by evaluating all possible paths for timing violations. The process involves breaking down the circuit into timing paths, modeling the delay of each timing arc, calculating the delay for each timing path, checking for timing violations, and optimizing the circuit. Each timing path in STA consists of several nets, vias, and functional models. The most basic cell of a timing path is called timing arc, which represents the timing relationship between two pins of any element. The accuracy of STA is dependent on the method used to model the delay of timing arcs. The most common approach is the look-up table (LUT) algorithm. During library characterization, several delays with different input transition times and load capacitance values are calculated by dynamic simulation. The values are then stored in a delay table. During the STA process, the STA engine uses interpolation or extrapolation of the table values to obtain the delay value for the current condition. The LUT algorithm provides an accurate and efficient way to model the delay of timing arcs in STA, and it is widely used in the industry for timing analysis. However, this algorithm assumes that the delay of timing arcs is independent of other circuit contexts, regardless of the input transition time and load capacitance value. This assumption ignores the potential impact of PSN and other noise. As the impact of noise on timing becomes increasingly apparent, the development of a model that can accurately capture the impact of noise on timing is essential.

The impact of PSN on timing has been studied extensively for many years. Conventionally, the PSN is estimated by annotating static voltage drops at each instance. Jiang et al. proposed a statistical modeling technique for PSN and integrated it with a statistical STA framework to estimate performance degradation [5]. Kim et al. proposed a vectorless approach to estimate the delay increase due to PSN by sensitizing the longest paths in the circuit [6]. Krstic et al. proposed a pattern generation technique to capture the impact of PSN on delay [7]. Hashimoto et al. replaced temporal PSN with its equivalent power/ground voltage and setup a PSN-aware STA algorithm [8]. [9] further analyzed the relationship between the equivalent voltage of PSN and timing performance, and established a more accurate model. In dynamic timing analysis (DTA), researchers have leveraged machine learning (ML) techniques to assess the impact of PSN on timing. Ye et al. introduced a support-vector-machine-based regression method to accurately predict voltage drop due to pattern-dependent

* Equal contributions.

Yufei Chen and Cheng Zhuo are with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou 310058, China (e-mail: chenyufei@zju.edu.cn; czhuo@zju.edu.cn). Corresponding Author: Cheng Zhuo.

Zizheng Guo, Runsheng Wang, Yibo Lin, and Ru Huang are with Peking University, Beijing 100871, China and Institute of Electronic Design Automation, Peking University, Wuxi, 214028, China (e-mail: gzz@pku.edu.cn; yibolin@pku.edu.cn).

IR drop based on inputs to the chip at runtime [10]. Liu et al. proposed a set of four features to effectively reduce the dimensionality of input vectors. They further compared the performance of three distinct ML models trained with these features for PSN-aware dynamic timing prediction [11]. Garyfallou et al. developed a framework based on event-driven timing simulation that identifies the underestimated dynamic timing slack and achieved significant improvements over conventional graph-based methods [12].

Static voltage drop estimations have been sufficient for earlier technology nodes where there is a sufficient amount of natural decoupling capacitance from the PDN [13]. However, their assumption of average PSN ignores the specific noise waveform shapes, which becomes a severe problem given the increasingly sharp PSN curves. PSN in advanced nodes usually comes in a burst due to the simultaneous switching activities of large amounts of instances. Due to the inability to model dynamic noise, the accuracy of the previous methods decreases as the node becomes more advanced.

To model the dynamic noise, accurate analysis of the noise waveform itself is required, which has used to be a computationally expensive task. Dynamic PSN-aware timing analysis relies heavily on the accuracy of the noise waveform analysis since the accuracy of the delay calculation directly depends on the accuracy of the noise estimation. Thanks to the recent studies on accelerating the dynamic voltage drop analysis process [14]–[18], promising improvements in both speed and accuracy have been achieved using machine learning methods from XGBoost to neural networks. These breakthroughs provide the prerequisites for the use of dynamic PSN information in STA.

Circuit timing analysis incorporating PSN is a crucial and complex undertaking. This task involves tackling a multitude of challenges that must be overcome for an accurate and reliable dynamic PSN-aware STA engine to be developed. Specifically, we identify three key challenges in the construction of such an engine. **(1) Parameterized Modeling of Noise Waveform**. A novel parameterized modeling approach is required with an appropriate trade-off between the number of parameters and the PSN information retained. **(2) Noise-Aware Timing Modeling**. A new learning-based noise-aware timing model is required to map the power noise to the timing impacts with enough accuracy. **(3) Computational Efficiency**. There can be millions of cell delays to be calculated in a single STA run, which demands a computationally efficient model to finish STA in a reasonable time budget.

In this paper, we propose a dynamic PSN-aware timing analysis method built inside an efficient STA engine. We highlight our contributions as follows:

- **Weibull Cumulative Distribution Function (CDF) based PSN waveform modeling**: We propose to model every canyon-shaped PSN curve using the CDF of Weibull distribution and reduce the modeling task to three noise shape parameters.
- **ML based accurate delay prediction**: We capture the non-linearity of delay and transition prediction tasks using data-driven ML models with outperforming accuracy.

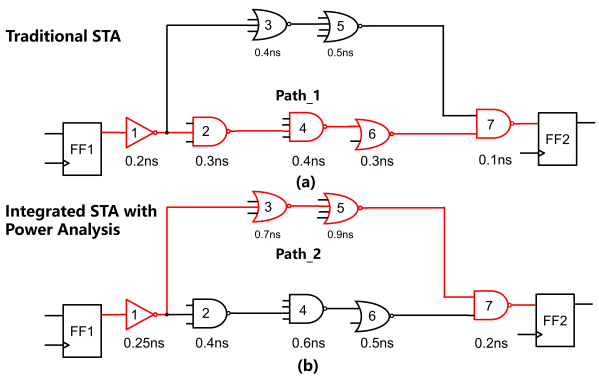



Fig. 1. Variation of circuit delay and critical path with and without power analysis integration.

- **Fast just-in-time (JIT) integration into STA engine**: We use JIT compilation techniques to embed our ML models into an STA engine, which provides both runtime efficiency and flexibility.

The rest of this paper is organized as follows. Section II demonstrates the background of STA and the motivation to integrate STA with power analysis. Section III introduces our PSN waveform modeling technique using Weibull CDF. Section IV presents our noise-aware timing model, the characterization process, and the fast JIT-based STA integration. Section V presents the experimental results. Finally, Section VI concludes the paper.

## II. Background and Motivation

### A. Limitation of Traditional Static Timing Analysis

In principle, the circuit design is broken down into timing paths during STA. The total delay of a timing path is the sum of all timing arc delays in the path. The timing arc delay is the amount of delay from the input pin to the output pin of a logic gate in a path. Typically, designers establish appropriate delay models for timing arcs during gate characterization to facilitate the calculation of timing arc delays under different scenarios. The most common delay model is the LUT algorithm. The timing arc delay is described as a function of the delay table entries, such as input transition time and output load capacitance. In the process of STA, the timing arc delay is calculated based on the current input transition time and output load capacitance through interpolation or extrapolation. After the timing path delays are determined, the timing violations will be checked based on timing constraints. For example, there are two timing paths in Fig. 1, namely Path_1 and Path_2. The total combination delay of the former path is 1.3 ns, while that of the latter path is 1.2 ns. The timing path Path_1 is critical and the minimum clock period is 1.7 ns assuming that the clock-to-q delay, the setup time, and the hold time of a flip-flop are 0.3 ns, 0.1 ns, and 0.1 ns, respectively.

In traditional STA, designers analyze the timing performance of a circuit without taking into account the impact of power distribution and power-related noise on timing. However, in modern electronic systems, power distribution and noise can significantly impact timing performance, particularly

for high-speed and low-power circuits. The impact of PSN on timing is very complex and has different effects on different devices. Neglecting PSN may result in misjudging critical paths and overlooking potential timing violations in the circuit. For example, in the circuit shown in Fig. 1, the NOR gate is more sensitive to PSN compared to other types of logic devices. As shown in Fig. 1(b), when incorporating the results of power analysis into timing analysis, the delay of each timing arc will be affected. This results in a change in the critical path of the circuit, which becomes Path_2, and the maximum combination delay increases to 2.05 ns. The motivation of this paper is to find an efficient method to integrate power supply noise analysis into STA. The key is to develop an effective method for modeling dynamic PSN and establish an efficient PSN-aware delay model.

The most comprehensive and accurate way to determine the impact of PSN in timing analysis is to perform a dynamic circuit simulation. However, the simulation of such a complicated network is infeasible, due to the long simulation time. Different approaches have been proposed to address this issue. Some works focused on the worst-case voltage drop on the supply network to ensure eliminating all potential timing violations [4], [5], [7], [19]. Some works modeled dynamic PSN with its equivalent DC and then develop a PSN-aware delay model based on this [8], [9]. However, these methods tend to amplify the impact of power supply noise to some extent, leading to overly pessimistic timing estimates.

Furthermore, to ensure the computational efficiency of the STA engine, it is necessary to utilize parameterized modeling of dynamic PSN. However, the LUT algorithm presents a significant challenge to this task. Although the LUT algorithm is capable of handling an arbitrary number of input variables, its efficiency diminishes as the number of input variables increases. The precision of the LUT is determined by its resolution, i.e., the number of data points allocated for each input variable. To accurately model highly nonlinear and complex distributions, such as the impact of PSN on timing, a considerable number of values need to be characterized for each input variable. Assuming each input variable has m tabulated values, the size of the LUT becomes $m^n$, where n represents the number of input variables. This exponential growth in LUT size with increasing input variables results in significant overhead on runtime memory and storage space. Furthermore, the time complexity of the LUT algorithm is $O(2^n)$, leading to a notable decrease in computational efficiency as the number of variables increases. To establish a PSN-aware delay model, it is necessary to introduce several variables that describe the characteristics of dynamic PSN. Considering the input transition time and output load capacitance that must be considered in STA, it is difficult to use the LUT algorithm to establish an efficient delay model. In similar problems like signal-integrity-aware error compensation [20], cell-delay modeling [21], and arrival time modeling [22], machine learning based models have been used as alternatives to LUT.

### *B. The Necessity of Modeling Dynamic Noise*

In advanced nodes, the PSN usually has a high-frequency noise component, which results in fast and large voltage fluctuations. Such a high-frequency noise cannot be effectively eliminated by decoupling capacitances as it can span as short as $1/10$ of a clock cycle. Traditional DC-based static voltage drop estimation no longer reflects the actual noise and can yield contradictory and unreasonable estimations. For example, a change in clock frequency can dramatically impact the voltage drop estimation, as shown in Fig. 2. A comparison between the static IR simulation results in Fig. 2(a) and Fig. 2(b) reveals that the static IR analysis becomes less reflective of actual circuit noise with the increasing frequency of PSN. As shown in the results in Fig. 2(a), although the noise value obtained from static IR analysis is very small, power supply noise can still have a significant impact on the devices that flip early in the clock cycle. Therefore, we need a new methodology that can consider *dynamic* PSN during circuit timing analysis.

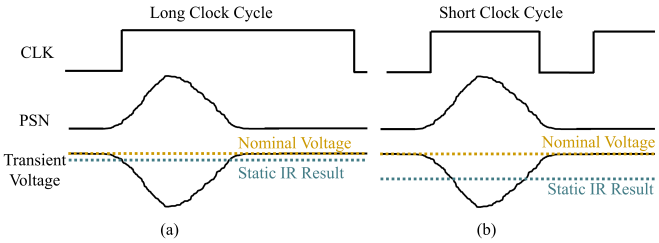



Fig. 2. Static voltage drop methods give two different voltage estimations for the same PSN curve with different clock periods.

For PSN waveform modeling, the equivalent DC method [8], [9] is a representative approach that is improved upon the static voltage drop analysis method. Instead of averaging the voltage inside the full clock cycle, they simplify the dynamic noise within a time frame around the switching event of the gate under analysis, as illustrated in Fig. 3. This shorter time frame leads to more accurate voltage estimation for every single signal transition, resulting in significant improvements in accuracy.

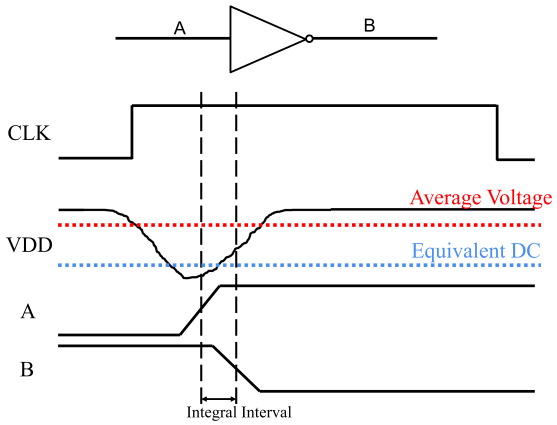



Fig. 3. The equivalent DC method focuses on the voltage within the transition period of the current gate.

However, this approach represents dynamic PSN with only integrations (i.e., equivalent DC voltage) which neglect potentially important features. As the nonlinear effects of circuit

devices increase, the errors caused by such representation become more apparent. For an And-or-invert gate in a 40 nm process, we conducted the following experiment: while ensuring that the PSN integral value during the signal flip remains constant, we varied the duration of the PSN and calculated the changes in the gate's delay and output transition time by SPICE. As shown in Fig. 4, delay and output transition time can fluctuate greatly even if the integrals are fixed. Therefore, it is essential to find an efficient method for modeling the waveform of dynamic PSN.

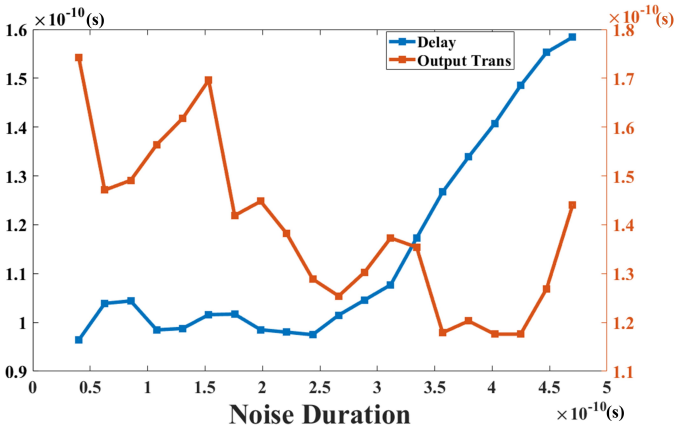



Fig. 4. Delay and output transition time can fluctuate given the same PSN integration value but different noise durations.

## III. Dynamic Noise Modeling

Caused by the toggling activities of neighboring logic gates, the power supply encounters noise in canyon-like waveforms, as depicted in Fig. 3. An accurate and concise representation of such signals is essential for building a timing model. A finite time-domain waveform like PSN cannot be effectively modeled by frequency-domain-based methods such as FFT. The reason is that these methods transform the original signal to an infinite one by convolving it with a Sinc function, and the correct shape feature of PSN in such an infinite signal has to be covered by a too wide frequency range over hundreds of GHz or even THz. Besides the difficulty to represent shapes, the noise effect on timing is also very sensitive to the *phase* of PSN signal (i.e., relative noise arrival times). All these features of PSN signals make the model setup both time-consuming and labor-intensive.

Edge-based modeling methods have been extensively studied in signal transition modeling [8], [9], [23]–[26]. However, the first-order models like equivalent DC voltage [8], [9] may become increasingly error-prone due to the complexity and non-linearity of PSN waveforms at advanced technologies. To achieve the desired accuracy, the modeling can become overly complex and high-dimensional [23]–[26]. On the other hand, unlike signal transition, PSN transition is generally much longer due to the decoupling from both decoupling capacitance and parasitics. As a result, the noise waveforms exhibit approximate monotonicity during the period of signal transition. While we can still adopt Weibull CDF like in [26] to model such monotonic edges, we also need to account for the interaction between the noise transition and its impacted signal transition.

Equation (1) gives the original CDF of the Weibull distribution. $T_0$ represents the starting point of the edge, which is the last moment when the waveform amplitude is zero. Roughly speaking, parameters $\alpha$ and $\beta$ control the shape and scale of the curve, respectively. In PSN waveform modeling, we prefer intuitive and physically meaningful parameter definitions other than $\alpha$ and $\beta$. Thus, we replace $\beta$ with the *noise transition time* $\tau$ and name $\alpha$ as *noise shape factor*. Here we define the transition time of a signal as the time gap between 10% and 90% of ideal VDD. The relation between noise transition time $\tau$ and the original $\beta$ can be derived analytically as shown in Equation (2).

$$W(t,\alpha,\beta) = 1 - e^{-((t-T_0)/\beta)^\alpha}, \qquad \forall t \geq T_0 \tag{1}$$

$$t_{10\%} = T_0 + \beta \cdot (ln\frac{10}{9})^{1/\alpha} \tag{2a}$$

$$t_{90\%} = T_0 + \beta \cdot (ln10)^{1/\alpha} \tag{2b}$$

$$\tau = t_{90\%} - t_{10\%} = \beta \cdot \left( (ln10)^{1/\alpha} - (ln\frac{10}{9})^{1/\alpha} \right) \tag{2c}$$

$$\beta = \frac{\tau}{\left( (ln10)^{1/\alpha} - (ln\frac{10}{9})^{1/\alpha} \right)} \tag{2d}$$

The problem of fitting a signal using Weibull CDF can be formulated as linear regression. In this work, we assume that the falling edge and rising edge of PSN are symmetric. In fact, due to various factors, the rise and fall edges of PSN waveforms are not perfectly symmetrical. However, in the design of digital circuits, the logic gates are implemented with dual pairs of pull-up and pull-down networks to prevent simultaneous on or off and designers strive to achieve equal delays in the pull-up and pull-down networks. Due to the design's symmetrical efforts and the effect of decoupling capacitors, the rise and fall times of the PSN rarely exhibit highly asymmetric behavior. As a result, there are many previous methods based on similar symmetrical assumptions [8], [9]. If high modeling accuracy is required, two additional parameters can be incorporated to model the rise and fall times separately to ensure higher accuracy. However, the characterization overhead will also increase, which can be partially mitigated by using adaptive sampling techniques. Therefore, we focus on the rising edge of the signal in the following discussions. First, the rising edge is normalized to the range of (0, 1). From Equation (1), the following relation can be derived.

$$\frac{ln(-ln(1-W(t-T_0)))}{\alpha} = ln(t-T_0) + ln(\tau) - ln\left( (ln10)^{1/\alpha} - (ln\tfrac{10}{9})^{1/\alpha} \right) \tag{3}$$

This can be expressed as the linear equation:

$$y = a \cdot x + b \tag{4}$$

where $y = ln(t - T_0)$, $x = ln\left(-ln\left(1 - W(t - T_0)\right)\right)$, $a = 1/\alpha$, and $b = ln(\tau) - ln\left((ln10)^{1/\alpha} - (ln\frac{10}{9})^{1/\alpha}\right)$. Given that $T_0$ is the starting point of the edge to be fitted, its value inherently depends on the characteristics of the edge. Consequently, we can adopt linear regression algorithms to

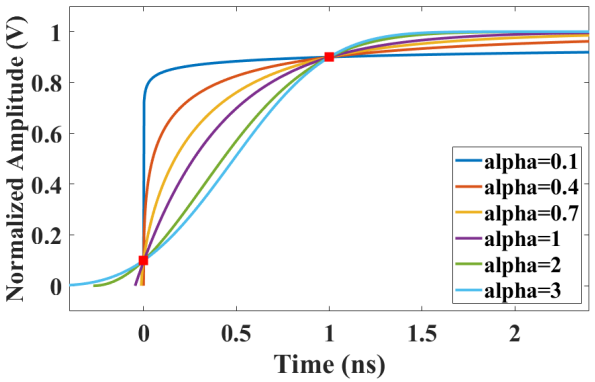


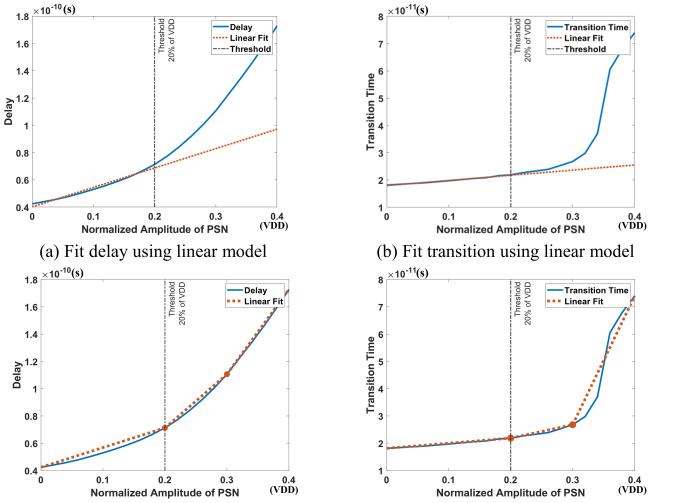

Fig. 5. The shape of Weibull CDF when the shape factor varies from 0.1 to 3.

find out $a$ and $b$. Finally, we get the noise shape factor from the reciprocal of $a$ and the noise transition time $\tau$ from $b$ by Equation (2c).

At this point, we can model all possible standard edges of dynamic PSN with noise transition time and noise shape factor. Noise transition time indicates the overall slope of the noise, while shape factors denote the convexity and concavity of the waveform. Fig. 5 depicts some curves of Weibull CDF. For clarity and convenience of description, we have aligned these curves at the 10% and 90% points by applying appropriate shifting to each curve. This alignment is employed to better elucidate the influence of the noise shape factor on the convexity and concavity of the edges. It is important to note that no alignment procedure is involved in the actual fitting process. The transition time of the curves is equal but the shape factors vary from 0.1 to 3. As the shape factor increases, the waveform transforms from a convex function into a concave function. The Weibull CDF with different shape factors may be used to represent the waveforms for different circuits. For example, when the shape factor is 0.1, it is similar to the response of a complex RC circuit; when the shape factor is 1, it is changed to the step response of a first-order RC circuit; for the shape factor of 3, the waveform is close to the standard ramp output, which simply matches the waveform of pure resistance loading.

Assuming that the noise rises and falls symmetrically, the offset between noise modeling and gate toggling time and the amplitude of the edge should be considered to model arbitrary waveform shapes. We define the time interval between the start point of modeling noise and the start point of gate toggling as time shift factor $\omega$.

According to the characteristics of the MOS transistor, when the amplitude of PSN is within a certain range the impact of noise amplitude on timing is roughly linear. As the amplitude increases, the nonlinearity between the amplitude of PSN and gate timing also increases. As shown in Fig. 6, the threshold is approximately 20% of VDD in our experimental process. When the peak value of PSN is less than 20% of VDD, the impact of PSN on gate timing is roughly linear. As shown in Fig. 6(a) and Fig. 6(b), within this range, it is sufficient to train the multi-layer perceptron (MLP) models at a normalized amplitude. The R-squared scores between the data obtained from linear interpolation and the original data are 0.976 for delay and 0.987 for transition time. The results indicate that using linear interpolation within this range introduces only small errors in gate timing calculations. However, when performing linear fitting to the entire dataset, the R-squared score for delay decreases to 0.808, and for transition time, it decreases to 0.587. This implies that using direct linear interpolation would introduce larger errors. Therefore, when the PSN peak value exceeds the threshold, it becomes necessary to train MLP models with multiple amplitudes. Interpolating between the models will significantly improve the accuracy of the prediction results. For instance, as shown in Fig. 6(c) and Fig. 6(d), the peak amplitude of PSN is 40% of VDD and three MLP models are trained at different amplitudes: 20%, 30%, and 40% of VDD. When performing interpolation between the models, it can achieve increased prediction accuracy compared to a single model. The R-squared score for delay increases from 0.808 to 0.965, and the R-squared score for transition time increases from 0.587 to 0.921.




(a) Fit delay using linear model
(b) Fit transition using linear model
(c) Fit delay using piecewise linear model
(d) Fit transition using piecewise linear model

Fig. 6. Variations of delay and transition time of a standcell with changes in noise amplitude.

Fortunately, a practical power delivery system design typically intends to limit the range of IR Drop to achieve the desired power saving and reliability [27]. Hence, training MLP models at a single amplitude is often sufficient for most practical applications.

Therefore, as depicted in Fig. 7, the dynamic PSN is finally modeled by noise transition time $\tau$, noise shape factor $\alpha$, and time shift factor $\omega$.

## IV. Library Characterization

Just knowing the logical function of a cell is not sufficient to build a functionally correct circuit. More aspects such as the transmission speed and power consumption need to be considered for functional correction. Additionally, the speed and power of a cell are influenced by many factors like operation voltages and output load. The process to collect this sort of information for each standard cell is called library

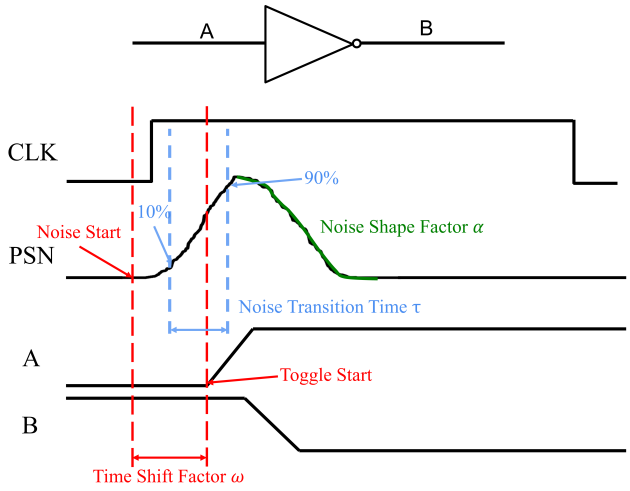



Fig. 7. The definitions of noise transition time $\tau$, noise shape factor $\alpha$, and time shift factor $\omega$, and their roles in modeling the normalized PSN.

characterization. Library characterization, therefore, plays a key role in STA by providing the basic delay and transition timing information for every timing arc.

### A. Problem Formulation

Liberty files, also known as liberty standard cell libraries or liberty timing libraries, are essential components in the electronic design automation process for integrated circuits. The results of library characterization are compiled into liberty files in the form of LUTs and these files contain detailed timing and power characteristics for each standard cell in a specific semiconductor process technology [28]. Typically, the LUT in liberty files describes the implicit mapping of input transition time $T_{trans}$ and total output load capacitance $C_{load}$ to timing information, which can be demonstrated as $T = f\left(T_{trans}, C_{load}\right)$.

The delay of the critical path determines the reported slacks of STA. The path delay can be expressed as Equation (5), where D denotes the path delay, m represents the number of timing arcs included in the path, $T_{trans,i-1}$ is the transition time of $(i-1)_{th}$ gate in the path and $C_{load,i}$ is the total output load capacitance of the $i_{th}$ gate.

$$D = \sum_{i=1}^{m} f\left(T_{trans,i-1}, C_{load,i}\right) \tag{5}$$

$$D' = D + \Delta D = D + \sum_{i=1}^{m} g(v_i) \tag{6}$$

The static IR drop based methods actually attempt to model the path delay fluctuation caused by supply voltage drop, which is expressed in Equation (6). $D'$ is the estimation of path delay considering the impact of voltage drop. $D$ is the original path delay and $g(v)$ is the function describing the impact of noise on gate performance. $v_i$ is the actual voltage of $i_{th}$ gate.

For static IR drop analysis, $v_i$ is the average voltage within a clock cycle, as shown in Equation (7). Here, $clock_{cycle} = T_i - T_{i-1}$. The error is mainly introduced by the assumption that the operating voltage of the gates is the average value.

$$v_i = \frac{1}{T_i - T_{i-1}} \int_{T_{i-1}}^{T_i} v dt \tag{7}$$

In order to solve the problem, the equivalent DC voltage based methods are devoted to finding a precise interval to calculate the accurate operating voltage of each gate. In fact, these methods aim to model Equation (8) or its variations. The dynamic PSN is represented by its integral.

$$T = f\left(T_{trans}, C_{load}, \frac{1}{T_i - T_{i-1}} \int_{T_{i-1}}^{T_i} v dt\right) \tag{8}$$

However, the transition time and time shift factor have a non-negligible impact on timing. As demonstrated in Section III, all normalized dynamic PSN can be modeled by three parameters: noise transition time $\tau$, noise shape factor $\alpha$, and time shift factor $\omega$. The problem is to find out the following function.

$$T = f\left(T_{trans}, C_{load}, \tau, \alpha, \omega\right) \tag{9}$$

### B. Multi-Layer Perceptron (MLP) Model

LUT is highly efficient for regression tasks with few input parameters. However, the space complexity of LUT is $O\left(m^n\right)$. For this task, storing the regression results will take up enormous space and make it time-consuming to compute the interpolated result.

The effectiveness of MLP in regression tasks speaks for itself. This method can extract the features automatically and eliminate to assign fitness order for each input parameter. Besides, the trained network can provide quick prediction results despite the large scale of input parameters. The train and inference of MLP can be easily deployed on GPU for parallelized acceleration.

*1) Training Data Generation:* We have parameterized the gate timing under the influence of PSN, representing it as a distribution with five variables: $T_{trans}$, $C_{load}$, $\tau$, $\alpha$, and $\omega$. By sweeping these five variables in SPICE, we can characterize the gate timing performance under the influence of PSN. The first two parameters are the inherent input of conventional STA. Their values are explictly determined in the index of LUT in liberty files. On the other hand, the ranges of noise transition time, noise shape factor, and time shift factor encompass all potential PSN occurrences on the PDN. The noise shape factor indicate the convexity and concavity of the waveform. As shown in Fig. 5, the typical shapes of the PSN can be covered with the shape factor varying in the $[0.1, 3]$. The noise transition time approximately reflects the frequency spectrum of PSN. Noise with very short transition times will be filtered out by decoupling capacitors, while noise with excessively long transition times will exhibit more static characteristics. Therefore, we need to choose a reasonable range for the noise transition time, which, in our experimental process, is set as [0.1 ns, 10 ns]. We set 10 sample points at equal intervals according to the logarithmic coordinates in the above two ranges, respectively.

The last parameter is used to eliminate the error introduced by the misalignment between the noise start time and the toggling activity. The time shift factor is defined as $\omega = T_{toggle\_start} - T_0$, where $T_0$ represents the starting point of the PSN waveforms, which is the last moment when the amplitude of PSN is zero. In the temporal dimension, the proximity of the PSN to the signal transition significantly affects gate timing. Due to the nonlinear nature of dynamic PSN, when the PSN is sufficiently close to the signal transition, the influence of time shift factor on timing also exhibits strong nonlinearity. However, as the time shift factor increases, this influence gradually diminishes. Fig. 8 illustrates the representative distribution of delay versus time shift factors in our experimental process. The transition time of the PSN is 0.6 ns and the shape factor is 1. It is observed that the impact of the time shift factor on timing is strongly non-linear within the range of -0.1 ns to 1 ns. When the time shift factor exceeds 1ns, its influence on the delay significantly diminishes. Additionally, when the time shift factor surpasses 2ns, it can be deemed to have no impact on the delay. We set 9 sample points at equal intervals from -0.1 ns to 1 ns and measure the timing when the time shift factor is 2 ns.

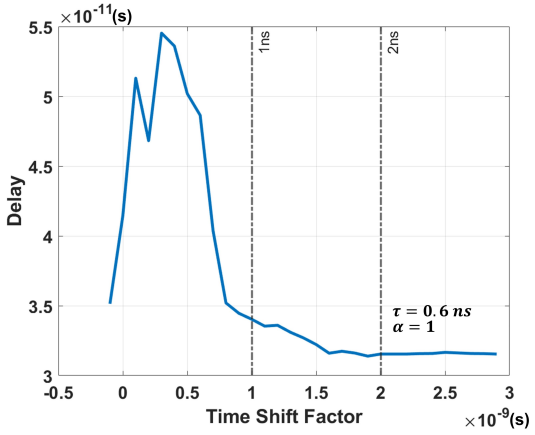



Fig. 8. Distribution of delays relative to time shift factors of a standard cell.

The timing information consists of delay and output transition time. Delay is the time interval between the 50% point of input transition to the 50% point of output transition. Output transition time is the time interval between 10% point and 90% point of output transition. The setting of characterization is demonstrated in Table I.

TABLE I
THE PARAMETER SETTINGS OF TRAINING DATA GENERATION

| | Parameter | Size | Range |
|---|---|---|---|
| **Input** | $T_{trans}$ | 10 | Liberty |
| | $C_{load}$ | 10 | Determined |
| | $\tau$ | 10 | [0.1 ns, 10 ns] |
| | $\alpha$ | 10 | $[0.1, 3]$ |
| | $\omega$ | 10 | $[-0.1ns, 2ns]$ |
| **Output** | Delay | 1 | - |
| | Output $T_{trans}$ | 1 | - |

Once the range of three parameters are determined, the Weibull-CDF-fitted PSN waveforms with different parameters

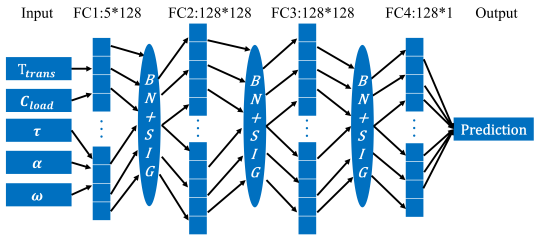



Fig. 9. The architecture of the proposed MLP model.

can be calculated and stored into piece-wise linear (PWL) files. SPICE tools can read these PWL files to obtain power supply voltage waveforms with specific noise. Subsequently, based on the input transition time and output load capacitance obtained from liberty files, the delay and output transition time for each timing arc can be measured. By measuring and recording the timing performance of all timing arcs under all possible conditions, we can obtain a dataset that will be used for training the neural network.

*2) Network Architecture:* Following a comprehensive comparison of various MLP models with different widths and depths, we have selected the architecture illustrated in Fig. 9. The detailed experimental results are present in Section V. As shown in Fig. 9, the MLP consists of an input layer, two hidden layers and a output layer. The input feature size is 5 and the output size is 1. Each hidden layer consists of 128 neurons. In order to ensure convergence, all layers in the MLP model, except for the output layer, are normalized by a 1D batch normalization layer [29]. Although Sigmoid may cause the vanishing gradient problem in deep networks, it can offer better non-linearity and normalize the output of the hidden layers. Therefore, given the shallow architecture in this work, we employ Sigmoid as the activation function to introduce non-linearity. Four different networks are required for one timing arc to cover the requirement of rise transition time, rise delay, fall transition time, and fall delay.

*3) Loss Function:* The loss function is expressed as Equation (10). Here, $L$ is the loss of a mini-batch. $B$ is the size of a mini-batch. $P$ denotes the prediction vector and $P_i$ is the $i_{th}$ value of the vector. $G_i$ is the $i_{th}$ value of ground truth. $\theta_1$ and $\theta_2$ are hyper-parameters to control the penalty items.

$$L = \sum_{i=1}^{B} \left| \frac{P_i - G_i}{G_i + eps} \right| + \theta_1 \sum_{i=1}^{B} |P_i - G_i| - \theta_2 \cdot B \cdot min\,(P) \quad (10)$$

The main goal of training is to minimize relative error. Having regard to a small deviation in the estimation of the small value will lead to a large relative error, we add the mean absolute error as a constraint. Moreover, both delay and output transition time are always positive, so the negative of minimal of each batch is used to penalize the negative prediction. In this case, the size of mini-batch is 1000. $\theta_1$ is fixed to 0.2 $ns^{-1}$ and $\theta_2$ is set to 0.001 $ns^{-1}$ initially and is decreased by the factor of 0.5 every 5 epochs. Both hyper-parameters are determined by experimental results of the MLP model trained without constraints.

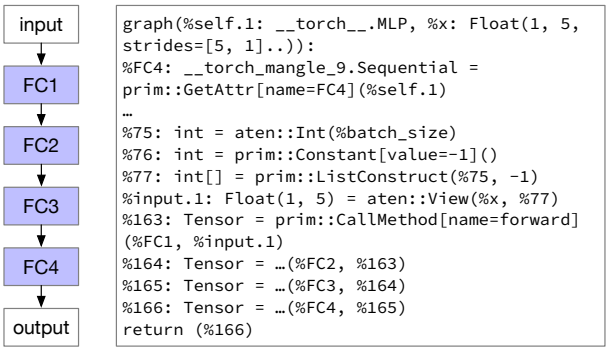



Fig. 10. Example ML model and JIT IR representation (in TorchScript).

### C. JIT Compilation of ML Models

After training the ML models, we integrate them into our STA engine. The STA engine uses our models to compute PSN-aware cell delays and perform transition propagation. This imposes two challenges:

1) The computation speed of ML models should not severely degrade the STA runtime. We need to embed an ML inference engine inside our STA engine with comparable efficiency.
2) There are different models for different types of cells, with different trained weights and even different layer types. The models and weights are part of the standard cell library and not part of the engine. The STA engine should be *model-agnostic*, in the sense that changes made to ML models only affect the input of the STA engine, not its own code.

We solve the above challenges by introducing just-in-time (JIT) compilation to our ML models. Specifically, the layers of ML models are compiled into an intermediate representation (IR) of the underlying computation graph. Fig. 10 shows an example ML model and its corresponding IR. Most state-of-the-art ML frameworks like PyTorch (TorchScript) and TensorFlow (TF-ONNX) support the creation of such an IR representation. The IR goes through its own optimization passes to maximize its runtime performance.

After optimization, this IR and all the corresponding weight parameters are *serialized* into a bit stream, which we store in the Liberty cell library file. On invocation, the STA engine *deserializes* and interprets the ML models from the input Liberty file. A lightweight ML framework runtime dedicated to IR execution is included in our STA engine, which provides optimized and vectorized implementation of ML operators. The JIT technique provides fast and flexible ML integration which is necessary for the ML application in timing analysis.

## V. Experimental Results

We implement the proposed method and make a comprehensive evaluation under an industrial 40nm technology. To meet the requirement of logical synthesis, we model several basic standard cells including inverter, NAND gate, NOR gate, and-or-invert (AOI) gate, or-and-invert (OAI) gate, and D-type flip-flop (FF). Table IV lists all characterized standard cells in detail. We generate our dataset of input parameters as proposed in Section IV-B, and then randomly partition the dataset into a training set and a test set with a ratio of 70% and 30%, respectively. The PSN amplitude during characterization is 20% of VDD. We conducted the training and evaluation of our methods on a platform equipped with an Intel i9 9980XE CPU and an NVIDIA GeForce 2080 Ti GPU with 11 GB of device memory. During training, we use PyTorch backend with Adam [30] optimization engine. After training, we JIT compile all the resulting models to TorchScript and extend the cell libraries with the bit stream of these models. We implement our dynamic PSN-aware timing analysis framework on top of the open-source STA engine OpenTimer [31].

### A. Fitness Accuracy

To demonstrate the effectiveness of the proposed Weibull CDF-based modeling method, we conducted experiments to measure the timing errors induced by the fitting. The delay and output transition time of each timing arc under the influence of four different realistic PSNs are calculated by SPICE. The waveforms of the realistic PSNs were represented in PWL format. Subsequently, we fitted the four waveforms using Weibull CDF and recalculated the delay and output transition time under the influence of fitted PSN. As shown in Table II, the experimental results indicate that our proposed modeling method can reasonably fit the PSN waveforms. The mean squared error (MSE) between fitted function and original function is below 0.05. In addition, the timing errors caused by fitting are also very small, with an absolute average relative deviation (AARD) of less than 4% for delay and less than 3% for transition time. The AARD between the prediction and ground truth can be computed using Equation (11), where k denotes the length of the testset, $P_i$ denotes the ith value of the prediction, and $G_i$ denotes the ith value of the ground truth.

$$AARD = \frac{1}{k}\sum_{i=1}^{k}\left|\frac{P_i - G_i}{G_i}\right| \tag{11}$$

TABLE II
Accuracy Results of Weibull CDF-based PSN Modeling Method

| MSE of Fitted function | AARD of Delay | AARD of Transition Time |
|---|---|---|
| 1.03E-03 | 2.28% | 1.72% |
| 4.41E-02 | 3.80% | 2.68% |
| 9.05E-03 | 3.30% | 2.52% |
| 1.82E-02 | 3.05% | 2.37% |

### B. The Architecture of MLP Models

To identify the most efficient MLP architecture, we trained multiple MLP models with varying depths and widths, and measured their accuracy and inference speed. Here, width refers to the number of neurons in each hidden layer, and depth represents the total number of layers in the model. For example, a model with a width of 128 and a depth of 5 has three hidden layers, each consisting of 128 neurons. The experimental results are shown in Table III.

The results indicate that as the width and depth increase, the number of model parameters also increases. The change in accuracy is more complex. For the models with the same depth, increasing the width will improve the model's fitting capacity, leading to a rapid increase in prediction accuracy. However, as the width continues to increase, the rate of accuracy improvement slows down until it approaches a maximum value. This is because MLP may not converge to the global optimum accurately, and increasing the width may lead to overfitting on the training set, limiting further improvement in prediction accuracy. For models with the same width, increasing the depth improves the model's ability to fit nonlinear patterns in the data. Consequently, for models with lower depths, increasing the depth results in a rapid improvement in prediction accuracy. However, as the depth becomes large, the impact of vanishing gradients becomes more prominent, making it easier for the MLP models to converge to inferior local optima, leading to a decline in prediction accuracy.

The primary consideration in selecting the MLP architecture is accuracy. To ensure high precision, we considered all structures with an AARD less than 5%, as indicated in the bolded data in Table III. When comparable accuracy is achieved, selecting a structure with fewer parameters leads to reduced storage usage. Consequently, we opted for an architecture with a depth of 4 and a width of 128 to train our MLP models. This particular configuration achieved a competitive AARD of 0.0489 with only 33921 parameters. In the characterization phase, we conducted measurements of $10^5$ delays and output transition times for each timing arc, both for rising and falling scenarios. The trained MLP models effectively captured the gate timing distribution with remarkable accuracy, utilizing only 33921 parameters. In contrast to the LUT algorithm, which is limited to linear interpolation between data points, the MLP models demonstrated the ability to learn non-linear relationships within the data. As a result, the LUT algorithm demands a substantially larger dataset and consumes significantly more storage space to achieve comparable prediction accuracy.

### C. Single-Gate Accuracy

We first demonstrate the performance of our model for predicting the delay and transition of a single gate. As shown in Fig. 11, the curve of the loss function converges fast enough to complete a single-gate training within 50 epochs.

To evaluate the accuracy of trained MLP models, we measure the AARD on the test set for each model. The average AARD among all models is 4.89%.

As shown in Fig. 12(a) and Fig. 12(a), the maximum AARD for all MLP models is less than 12%, and more than 75% of trained models have less than 6% AARD. This result demonstrates the capability of our MLP-based model to capture the complex and non-linear effects of dynamic power noise on timing.

Moreover, we examined the distribution of the predicted errors for each MLP model. A typical error distribution is illustrated in Fig. 12(c). The x-axis denotes the absolute value of prediction errors, while the y-axis represents the relative prediction errors. The distribution of predicted errors from the trained MLP models is primarily concentrated near the origin of the coordinate axis, indicating that both the relative and absolute errors are small. Furthermore, we noted that points with relatively large relative errors have very small absolute errors (less than 0.03 ns), while points with large absolute errors have very small relative errors (less than 10%). The former is because very small prediction errors can result in large relative errors for cases with low output delays, but such prediction errors have minimal impact on the final path delay error. The latter occurs with a very low frequency, and the relative errors are all less than 10%, suggesting that the model is capable of providing accurate predictions.

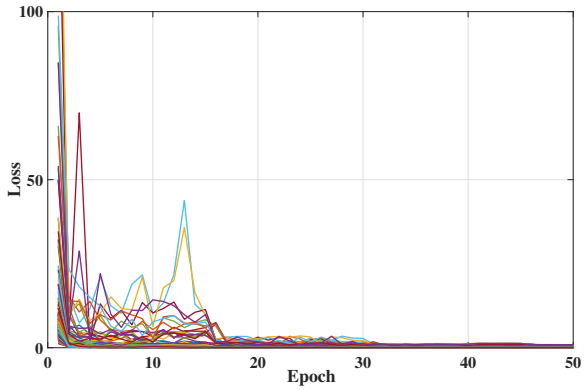
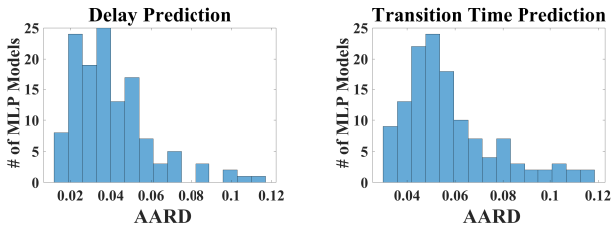
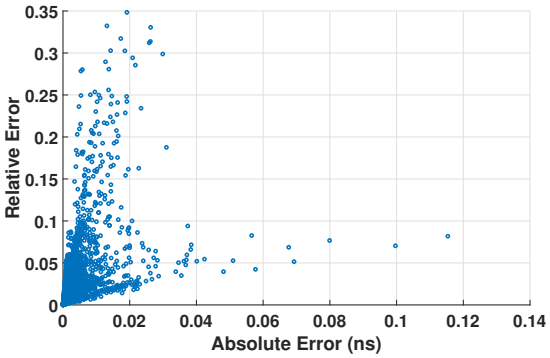



Fig. 11. Convergence maps of the loss function for all MLP models. Each curve represents the convergence of an MLP model during training.




(a) Histograms of the AARD for all delay models

(b) Histograms of the AARD for all transition time models




(c) Scatter map of the predicted error distribution from an MLP model

Fig. 12. Accuracy result of trained MLP models.

TABLE III
EXPERIMENTAL RESULTS OF THE NUMBER OF PARAMETERS AND ACCURACY FOR MLP ARCHITECTURES WITH DIFFERENT WIDTHS AND DEPTHS

| **#Layers** | **3** | | | | **4** | | | | **5** | | | | **6** | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Width** | 32 | 64 | 128 | 256 | 32 | 64 | 128 | 256 | 32 | 64 | 128 | 256 | 32 | 64 | 128 | 256 |
| **#Parameters** | 1281 | 4609 | 17409 | 67585 | 2337 | 6769 | 33921 | 133377 | 3393 | 12929 | 50433 | 199169 | 4449 | 17089 | 66945 | 264961 |
| **AARD** | 0.143 | 0.116 | 0.0971 | 0.0851 | 0.0942 | 0.0597 | **0.0489** | **0.0488** | 0.0822 | 0.0546 | **0.0494** | **0.0495** | 0.134 | 0.0952 | 0.0956 | 0.0952 |

TABLE IV
THE DETAILED INFORMATION OF CHARACTERIZED STANDARD CELLS

| **Type** | **Input** | **Scale** |
|---|---|---|
| Inverter | INV1 | $\times1, \times2, \times4, \times8, \times16$ |
| Nand | ND2,ND3,ND4 | $\times1, \times2, \times4, \times8$ |
| Nor | NR2,NR3,NR4 | $\times1, \times2, \times4, \times8$ |
| And-Or-Invert | AOI12,AOI22 | $\times1, \times2, \times4$ |
| Or-And-Invert | OAI12,OAI22 | $\times1, \times2, \times4$ |
| D-type Flip-Flop | DFF1 | $\times1, \times2$ |

### D. Paths Accuracy

Predicting the delay of a signal path with multiple gates is much more difficult, because the ML models may be influenced by the overestimation or underestimation from the previous stages. Furthermore, the discrepancy in the relative noise arrival time of different gates should be taken into consideration as well. To verify the effectiveness of our proposed method on signal paths, we compare our path delay prediction result with Synopsys HSPICE, OpenTimer [31], and the equivalent DC method [9] under 3 representative PSN waveforms listed in Table V.

We run vectorless dynamic simulation using commercial tools to obtain the dynamic supply noise, which is then parameterized using Weibull CDF modeling and fed to the proposed flow. Each waveform in Table V represents a typical scenario of vectorless dynamic voltage drop analysis. Group A represents a typical waveform generated by with lower toggling rate, exhibiting relatively small amplitude (10% of VDD) and lower frequency ($\tau$ = 5 ns). In Group A, the shape of PSN is the modeled using $\alpha = 0.3$. On the other hand, Group B represents the waveforms generated with typical toggling rate, exhibiting relatively larger amplitude (20% of VDD) and higher frequency ($\tau$ = 0.5 ns). Similarly, in Group B, the shape of PSN is the modeled using $\alpha = 0.3$. In Group C, the PSN waveform represents even higher toggling rate, exhibiting higher frequency ($\tau$ = 0.5 ns) and amplitude (20% of VDD). In group C, the shape of PSN is modelled using $\alpha = 3$. The noise waveforms start simultaneously with the first stage input of the circuit. We regard SPICE as the golden result for its accurate and rigorous waveform simulation at the cost of unacceptable runtime on large circuits. To ensure a fair comparison of results, both SPICE simulations and predictions using various models were conducted without considering interconnect delays and signal slew fluctuation caused by transmission. We invocate OpenTimer to compute the path delay under the non-linear delay models (NLDM) library of two corners, typical and worst. The typical corner ignores all power noise, and the worst corner estimates the worst-case static voltage drop.

*1) Inverter Chains:* We first conducted experiments on inverter chains of different lengths. Fig. 13 depicts the result of path delay measured by different methods. The results show that our proposed method is the closest to the actual timing performance of the inverter chain, with an average 2.3% absolute deviation between our dynamic PSN aware timer and SPICE. Due to neglecting the impact of PSN, the path delay obtained from timing analysis using the typical corner LUT is lower than the actual result. When calculating using the worst corner LUT, the circuit always operates under the worst-case noise conditions, which amplifies the effect of PSN and leads to excessively pessimistic results. Modeling PSN using the equivalent DC method can eliminate the pessimistic estimates introduced in noise modeling and make the results closer to reality. However, this method ignores other characteristics of dynamic noise, so its prediction results are still not accurate enough.

TABLE V
PSN WAVEFORM PARAMETER SETTINGS UNDER TEST

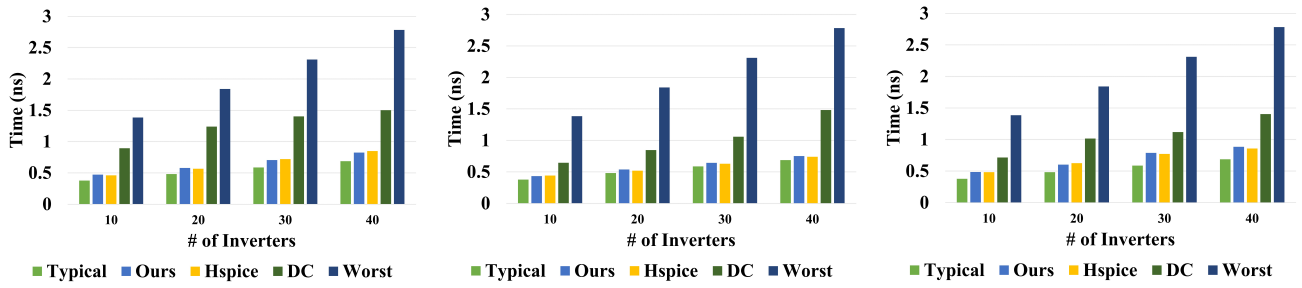
| **Group** | **A** | **B** | **C** |
|---|---|---|---|
| $\tau$ | 5 ns | 0.5 ns | 0.5 ns |
| $\alpha$ | 0.3 | 0.3 | 3 |
| **Amplitude** | 10% of VDD | 20% of VDD | 20% of VDD |

*2) Circuit Benchmarks:* To further test the performance of the proposed method in STA under real circuit scenarios, we carried out experiments on ISPD 2012 benchmarks [32]. Fig. 14 shows the comparison of results, where we remain the closest to the SPICE simulation. Compared with the static voltage drop-based methods, our proposed method successfully eliminates the substantial pessimism in PSN-induced delay.

$$AER = \frac{1}{n}\sum_{i=1}^{n}\frac{P_i - G_i}{G_i} * 100\% \tag{12}$$

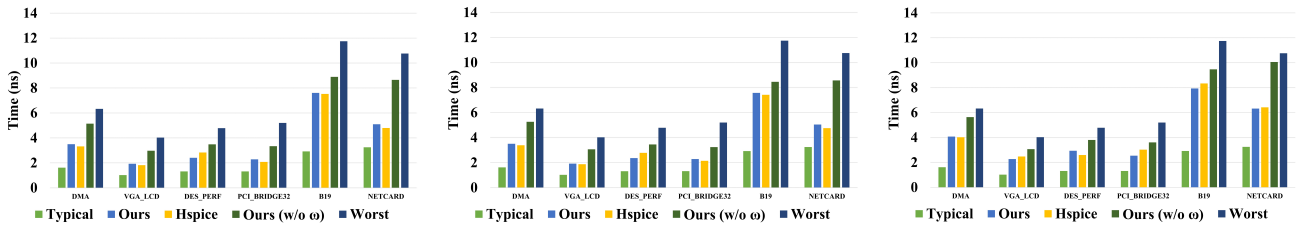
As shown in Table VI, our proposed method stands out among all approaches with an average error ratio (AER) of 6.27%. The AER can be derived from Equation (12), where n denotes the length of the testset, $P_i$ denotes the ith value of the prediction, and $G_i$ denotes the ith value of the ground truth. NLDM typical corners introduce excessive optimism with potential timing violations (negative percentages), while worst corners and ablations on the parameter $\omega$ introduce excessive pessimism (large positive percentages). These results clearly demonstrate the effectiveness of our proposed PSN-aware timing analysis method, especially our parameter-based PSN waveform modeling technique which is critical in the analysis of long paths with time-shifted waveforms.

Fig. 15 shows a stage-by-stage analysis of delay prediction for the critical path of the circuit DMA. Our model achieved




(a) Group A (b) Group B (c) Group C

Fig. 13. Path delay of inverter chains in different lengths generated by the proposed method, SPICE, equivalent DC based method, and OpenTimer.




(a) Group A (b) Group B (c) Group C

Fig. 14. Maximum path delay of different circuits in ISPD2012 generated by the proposed method, SPICE, and OpenTimer.

TABLE VI
THE AVERAGE ERROR RATIO BETWEEN OUR METHOD AND BASELINES IN PREDICTING THE MAXIMUM PATH DELAY ON ISPD 2012 BENCHMARKS.

| | AER Typical | AER Worst | AER Ours (w/o $\omega$) | AER Ours |
|---|---|---|---|---|
| DMA | -53.4% | 82.0% | 46.5% | 4.28% |
| VGA_LCD | -47.9% | 104.9% | 52.4% | 5.56% |
| DES_PERF | -51.4% | 76.7% | 30.9% | 12.6% |
| PCI_BRIDGE32 | -48.3% | 120.3% | 44.0% | 8.43% |
| B19 | -57.7% | 52.2% | 11.5% | 2.15% |
| NETCARD | -39.4% | 100.7% | 67.4% | 4.58% |
| **Average** | -49.7% | 89.5% | 42.1% | **6.27%** |

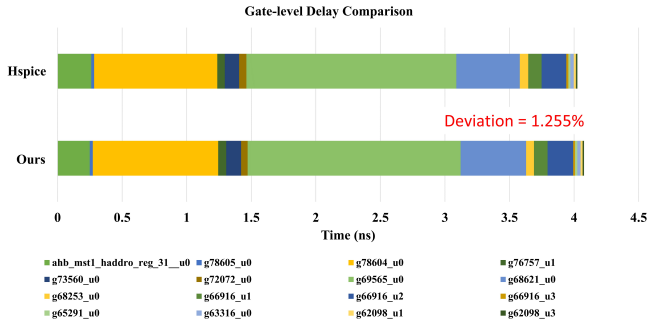



Fig. 15. Gate-level delay comparison between the proposed method and SPICE.

a low prediction error of 1.255% for the delay of this path. Our method not only gives accurate path delay estimation, but also provides fine-grained delay information down to a single gate. Besides, it can be observed that for gates with very small delays, although the relative error of the prediction may be large, this error does not have a significant impact on the final path delay. Such information can be very useful in timing-driven circuit optimization flow. When training under the same conditions, prioritizing the accuracy of high-delay scenarios can lead to more efficient training. This is the reason why we added the constraint term $\theta_1 \sum_{i=1}^{B} |P_i - G_i|$ in the loss function.

### *E. Ablation Study on Proposed Method*

In order to examine the individual contributions of each component, we conducted a series of ablation experiments to measure the impact of each component on the accuracy. We designed seven experiments, where the impact of power noise modeling methods, delay model methods, and loss function were taken into consideration. The specific settings of the experiments are illustrated in Table VI. The three types of PSN waveforms presented in Table V were also tested in the experiment and the noise started simultaneously with the signal propagation. It should be noted that for the 2-entry LUT, the entries are $T_{trans}$ and $C_{load}$; for the 3-entry LUT, the inputs are $T_{trans}$, $C_{load}$, and noise voltage; for the 4-input MLP, the inputs are $T_{trans}$, $C_{load}$, $\tau$, and $\alpha$; for the 5-input MLP, the inputs are $T_{trans}$, $C_{load}$, $\tau$, $\alpha$, and $\omega$.

In these experiments, we used different methods to calculate the critical path delay of the circuits in ISPD 2012 benchmarks. Table VI presents the AER between the predicted timing values from the seven experiments and the SPICE simulation results. Similar to the previous experiments, interconnect delays and signal slew fluctuation caused by transmission lines were not taken into account. The AER of output transition time is the average relative error between the predicted values and the ground truth measured in SPICE simulation at each stage.

From the settings in the table, it can be seen that EXP1 is equivalent to calculating path delay using typical corner LUT, while EXP2 is equivalent to calculating path delay

TABLE VII
THE SETTINGS OF ABLATION EXPERIMENTS AND CORRESPONDING RESULTS

| | EXP1 | EXP2 | EXP3 | EXP4 | EXP5 | EXP6 | EXP7 |
|---|---|---|---|---|---|---|---|
| **Noise Model** | None | Worst-case DC | Equivalent DC | Weibull-CDF | Weibull-CDF+$\omega$ | Weibull-CDF+$\omega$ | Weibull-CDF+$\omega$ |
| **Delay Model** | 2-entry LUT | 2-entry LUT | 3-entry LUT | 4-input MLP | 5-input MLP | 5-entry LUT | 5-input MLP |
| **Loss Function** | None | None | None | Proposed | Relative Error | None | Proposed |
| **AER of Path Delay** | -49.7% | 89.5% | 71.3% | 42.1% | 9.51% | 11.4% | 6.27% |
| **AER of Transition Time** | -74.5% | 63.6% | 50.8% | 39.5% | 13.7% | 16.0% | 8.67% |

using worst corner LUT. Similar to previous experimental results, EXP1 provided an overly optimistic prediction result with AER of -49.7% for path delay prediction and -74.5% AER for transition time prediction, while EXP2 provided an overly pessimistic prediction result with AER of 89.5% for path delay prediction and 63.6% for transition time prediction. Comparing the results of EXP2 and EXP3 demonstrates that modeling dynamic noise with equivalent DC can mitigate pessimistic estimates to a certain degree. The average voltage of the modeled dynamic noise in EXP3 and EXP4 is the same. EXP3 only considers the static effect of the PSN, while EXP4 fits the PSN waveform using Weibull CDF, which can more accurately reflect the actual impact of the noise. The comparison of EXP4 and EXP7 results reveals the impact of the time shift factor on pessimistic removal. In EXP4, where the time shift factor is absent, PSN is assumed to occur simultaneously with the signal propagation of each timing arc, leading to a higher level of pessimistic estimation. In contrast, in EXP7, the delay model considers the timing offset between each timing arc and PSN to adjust the impact of noise on the delay. By incorporating the time shift factor, the pessimistic estimation caused by ignoring the offset between PSN and the current timing arc is significantly reduced, especially for timing arcs that are far away from the noise source. Therefore, compared to EXP4, EXP7 reduces the AER from 42.1% to 6.39% for path delay prediction and from 39.5% to 8.67% for transition time prediction. The distinction between EXP5 and EXP7 lies in the loss function utilized for training the MLP models. With the incorporation of the two constraints proposed in this paper, the MLP models trained in EXP7 are capable of providing more precise predictions. The results of EXP6 and EXP7 show that in multi-input scenarios, using MLP instead of LUT can achieve higher modeling accuracy. The 5-entry LUT is also obtained during the characterization phase. To generate the data for training MLP models, we vary $T_{trans}$, $C_{load}$, $\tau$, $\alpha$, and $\omega$ in SPICE simulations to obtain different timing values, which are then stored in the corresponding five-dimensional matrices. When combining these matrices with the range of the five input variables, the corresponding 5-entry LUT can be generated. The LUT can only perform linear interpolation between any two data points, while the MLP learns to fit the nonlinear variations between data points through training. Considering the complex and nonlinear impact of noise on timing performance, MLP exhibits higher accuracy in modeling the timing distribution, even given the same amount of characterization data. Moreover, the 5-entry LUT contains 100,000 entries, whereas the MLP consists of 33,921 parameters. Compared to the 5-entry LUT, the MLP achieves higher space efficiency and occupies less storage space.

### F. JIT Integration Efficiency

The PSN-aware analysis should not incur too large an overhead to timing analysis runtime. By replacing the table look-up operation in NLDM models with MLP-based PSN-aware models, a larger amount of computation is needed. Fortunately, as shown in Table VIII, there is only 8%–25% runtime degradation incurred by our flow. Meanwhile, the percentage of runtime degradation is greatly *lowered* on *larger* circuits. This indicates that the model computation is not our runtime bottleneck, and the added runtime on small circuits mostly comes from one-time initialization work. The superior performance of ML models in STA flow comes from our JIT compilation technique which provides both fast and flexible ML integration.

TABLE VIII
SPEED COMPARISON BETWEEN NLDM (OPENTIMER) AND OUR PROPOSED METHOD

| | #Cells | NLDM Runtime | Ours Runtime | Runtime Degradation |
|---|---|---|---|---|
| DMA | 25301 | 14.77 s | 19.73 s | 25.1% |
| VGA_LCD | 164891 | 78.96 s | 95.72 s | 17.5% |
| DES_PERF | 111229 | 43.29 s | 50.08 s | 13.6% |
| PCI_BRIDGE32 | 33203 | 13.84 s | 18.31 s | 24.3% |
| B19 | 219268 | 152.53 s | 171.58 s | 11.1% |
| NETCARD | 958780 | 517.80 s | 562.02 s | 7.9% |

## VI. CONCLUSION

In this paper, we have presented an accurate and fast dynamic PSN modeling technique. We characterize a PSN curve using Weibull CDF functions and represent the standard cells by 2 inherent parameters and 3 PSN-aware parameters. To more accurately and efficiently model the PSN effect on timing, we adopt MLP-based delay models and integrate them into STA engine by introducing JIT compilation. Experimental results show that MLP gives accurate estimations of single-cell timing with an average relative error of 4.89%. Additional results indicate that the proposed method is still capable of more complex path delay prediction tasks. The average error between the proposed method and SPICE is as low as 2.3% on inverter chains and 6.27% on the ISPD2012 benchmarks. Our adoption of machine learning models in STA engine gives

a substantial accuracy improvement and pessimism reduction, incurring very small runtime overhead.

The proposed work demonstrates the necessity of modeling the interaction between dynamic noise and timing, which also sheds light on many future potential areas to explore. For example, we still need to further reduce the size of the MLP model and explore its application in dynamic timing analysis (DTA). Furthermore, leveraging a more precise dynamic noise-aware DTA, we can seamlessly incorporate this approach with vector-based dynamic noise analysis. This fusion has the potential to mitigate the prevailing pessimism in sign-off analysis. Finally, we would like to further investigate more complex interaction or high order effects induced by technology advancement and power supply architecture innovation, e.g., buried power rail. Thus, the proposed work will be open sourced to facilitate the community for future studies.

## ACKNOWLEDGMENTS

The authors are thankful to the editors and the anonymous reviewers for their invaluable assistance and insightful comments that significantly contributed to the improvement of this paper. The work was supported by NSFC (Grant No. 62034007, and 62141404), Major Program of Zhejiang Provincial NSF (Grant No. D24F040002) and SGC Cooperation Project (Grant No. M-0612).

## REFERENCES

[1] K. Shepard and V. Narayanan, "Noise in deep submicron digital design," in *Proceedings of International Conference on Computer Aided Design*, pp. 524–531, 1996.

[2] P. Bhattacharjee, P. Rana, and A. Majumder, "Understanding of on-chip power supply noise: Suppression methodologies and challenges," in *Recent Trends in Communication Networks*, pp. 1–10, 2019.

[3] K. Arabi, R. Saleh, and X. Meng, "Power supply noise in socs: Metrics, management, and measurement," *IEEE Design & Test of Computers*, pp. 236–244, 2007.

[4] S. Pant, D. Blaauw, V. Zolotov, S. Sundareswaran, and R. Panda, "Vectorless analysis of supply noise induced delay variation," in *ICCAD-2003. International Conference on Computer Aided Design (IEEE Cat. No.03CH37486)*, pp. 184–191, 2003.

[5] Y.-M. Jiang and K.-T. Cheng, "Analysis of performance impact caused by power supply noise in deep submicron devices," in *Proceedings 1999 Design Automation Conference (Cat. No. 99CH36361)*, pp. 760–765, 1999.

[6] H. S. Kim and D. Walker, "Statistical static timing analysis considering the impact of power supply noise in vlsi circuits," in *Seventh International Workshop on Microprocessor Test and Verification (MTV'06)*, pp. 76–82, 2006.

[7] A. Krstic, Y.-M. Jiang, and K.-T. Cheng, "Pattern generation for delay testing and dynamic timing analysis considering power-supply noise effects," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 416–425, 2001.

[8] M. Hashimoto, J. Yamaguchi, T. Sato, and H. Onodera, "Timing analysis considering temporal supply voltage fluctuation," in *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, pp. 1098–1101, 2005.

[9] T. Okumura, F. Minami, K. Shimazaki, K. Kuwada, and M. Hashimoto, "Gate delay estimation in sta under dynamic power supply noise," in *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, pp. 775–780, 2010.

[10] F. Ye, F. Firouzi, Y. Yang, K. Chakrabarty, and M. B. Tahoori, "On-chip droop-induced circuit delay prediction based on support-vector machines," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 665–678, 2016.

[11] Y.-C. Liu, C.-Y. Han, S.-Y. Lin, and J. C.-M. Li, "Psn-aware circuit test timing prediction using machine learning," *IET Computers & Digital Techniques*, pp. 60–67, 2017.

[12] D. Garyfallou, I. Tsiokanos, N. Evmorfopoulos, G. Stamoulis, and G. Karakonstantis, "Accurate estimation of dynamic timing slacks using event-driven simulation," in *2020 21st International Symposium on Quality Electronic Design (ISQED)*, pp. 225–230, 2020.

[13] C. Zhuo, K. Unda, Y. Shi, and W.-K. Shih, "From layout to system: Early stage power delivery and architecture co-exploration," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 38, no. 7, pp. 1291–1304, 2019.

[14] Z. Xie, H. Ren, B. Khailany, Y. Sheng, S. Santosh, J. Hu, and Y. Chen, "Powernet: Transferable dynamic ir drop estimation via maximum convolutional neural network," in *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, pp. 13–18, 2020.

[15] Y. Zhang, H. Ren, and B. Khailany, "Grannite: Graph neural network inference for transferable power estimation," in *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, pp. 1–6, 2020.

[16] Y.-C. Fang, H.-Y. Lin, M.-Y. Su, C.-M. Li, and E. J.-W. Fang, "Machine-learning-based dynamic ir drop prediction for eco," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 1–7, 2018.

[17] Y.-C. Lu, S. Nath, S. Pentapati, and S. K. Lim, "Eco-gnn: Signoff power prediction using graph neural networks with subgraph approximation," *ACM Transactions on Design Automation of Electronic Systems*, pp. 1–22, 2023.

[18] X. Dong, Y. Chen, J. Chen, Y. Wang, J. Li, T. Ni, Z. Shi, X. Yin, and C. Zhuo, "Worst-case power integrity prediction using convolutional neural network," *ACM Transactions on Design Automation of Electronic Systems*, pp. 1–19, 2023.

[19] J.-J. Liou, A. Krstic, Y.-M. Jiang, and K.-T. Cheng, "Modeling, testing, and analysis for delay defects and noise effects in deep submicron devices," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 756–769, 2003.

[20] A. B. Kahng, M. Luo, and S. Nath, "Si for free: machine learning of interconnect coupling delay and transition effects," in *ACM Workshop on System Level Interconnect Prediction (SLIP)*, pp. 1–8, 2015.

[21] W. Raslan and Y. Ismail, "Deep-learning cell-delay modeling for static timing analysis," *Ain Shams Engineering Journal*, p. 101828, 2022.

[22] Z. Guo, M. Liu, J. Gu, S. Zhang, D. Z. Pan, and Y. Lin, "A timing engine inspired graph neural network model for pre-routing slack prediction," in *Proceedings of the 59th ACM/IEEE Design Automation Conference*, pp. 1207—-1212, 2022.

[23] Y. Miki, M. Abe, and Y. Ogawa, "Pcheck: a delay analysis tool for high performance lsi design," in *Proceedings of the IEEE 1995 Custom Integrated Circuits Conference*, pp. 267–270, 1995.

[24] F. Dartu and L. T. Pileggi, "Modeling signal waveshapes for empirical cmos gate delay models," in *6th Intl. WorkshopPATMOS*, pp. 57–66, 1996.

[25] M. Hashimoto, Y. Yamada, and H. Onodera, "Capturing crosstalk-induced waveform for accurate static timing analysis," in *Proceedings of the 2003 international symposium on Physical design*, pp. 18–23, 2003.

[26] C. S. Amin, F. Dartu, and Y. I. Ismail, "Weibull-based analytical waveform model," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, pp. 1156–1168, 2005.

[27] C. Zhuo, S. Luo, H. Gan, J. Hu, and Z. Shi, "Noise-aware dvfs for efficient transitions on battery-powered iot devices," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 7, pp. 1498–1510, 2020.

[28] "What is library characterization." https://www.synopsys.com/glossary/what-is-library-characterization.html.

[29] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on International Conference on Machine Learning*, pp. 448–456, 2015.

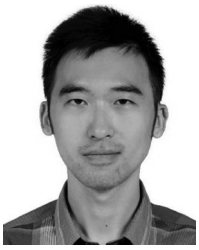
[30] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014. arxiv:1412.6980.

[31] T.-W. Huang and M. D. F. Wong, "Opentimer: A high-performance timing analysis tool," in *2015 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 895–902, 2015.

[32] M. M. Ozdal, C. Amin, A. Ayupov, S. Burns, G. Wilke, and C. Zhuo, "The ispd-2012 discrete cell sizing contest and benchmark suite," in *ACM International Symposium on Physical Design (ISPD)*, pp. 161–164, 2012.

**Yufei Chen** (Student member, IEEE) received his B.Sc degree from Zhejiang University, Hangzhou, China, in 2019, where he is currently working toward the Ph.D. degree at the College of Information Science and Electronic Engineering. His current research interests include power integrity analysis and the application of machine learning algorithms in EDA.

**Zizheng Guo** (Student member, IEEE) received his B.Sc degree from Peking University, Beijing, China, in 2022, where he is currently pursuing his Ph.D. degree at the School of Integrated Circuits. His current research interests include data structures, algorithm design and GPU acceleration for combinatorial optimization problems.



**Runsheng Wang** (Member, IEEE) (S'07–M'11) received the B.S. and Ph.D. (Hons.) degrees from Peking University, Beijing, China, in 2005 and 2010, respectively., From November 2008 to August 2009, he was a Visiting Scholar with Purdue University, West Lafayette, IN, USA. He joined Peking University in 2010 and is currently an Associate Professor with the Institute of Microelectronics. He has authored/coauthored one book, three book chapters, and over 100 scientific articles, including more than 30 articles published in the International Electron Devices Meeting (IEDM) and the Symposium on VLSI Technology (VLSIT). He has been granted 12 U.S. patents and 29 Chinese patents. His current research interests include nanoscale CMOS devices, characterization and reliability, circuit and device interaction, and new-paradigm computing.,Dr. Wang was awarded the IEEE EDS Early Career Award by the IEEE Electron Device Society (EDS), the NSFC Award for Excellent Young Scientists by the National Natural Science Foundation of China (NSFC), the Natural Science Award (First Prize) by the Ministry of Education (MOE) of China, and many other awards. He has served on the Technical Program Committee of many conferences, including IEDM and IRPS. He serves on the Editorial Board of the IEEE Transactions on Electron Devices, Scientific Reports, and Science China Information Sciences.

**Ru Huang** (Fellow, IEEE) received the B.S. (Hons.) and M.S. degrees in electronic engineering from Southeast University, Nanjing, China, in 1991 and 1994, respectively, and the Ph.D. degree in microelectronics from Peking University, Beijing, China, in 1997.,Since 1997, she has been a Faculty Member with Peking University, where she is currently a Professor with the School of Integrated Circuits. She is also the President with Southeast University. She has authored or coauthored five books, five book chapters, and more than 300 articles, including more than 80 articles in IEDM (35 IEDM articles from 2007 to 2019), VLSI Technology Symposium, IEEE EDL, and IEEE T-ED. She has delivered over 50 keynote/invited talks at conferences and seminars. She holds over 200 patents including 49 U.S. patents. Her research interests include nanoscaled CMOS devices, ultralow-power novel devices, new device for neuromorphic computing, emerging memory technology, and device variability/reliability.,Dr. Huang is also an Elected Academician of the Chinese Academy of Sciences, Beijing, and an Elected Member of TWAS Fellowship.

**Yibo Lin** (Member, IEEE) received the B.S. degree in microelectronics from Shanghai Jiaotong University, Shanghai, China, in 2013, and the Ph.D. degree in electrical and computer engineering from the University of Texas at Austin, Austin, TX, USA, in 2018 advised by Prof. David Z. Pan.,He worked as a Postdoctoral Researcher with the University of Texas at Austin from 2018 to 2019. He is currently an Assistant Professor with the School of Integrated Circuits, Peking University, Beijing, China. His research interests include physical design, machine learning applications, and heterogeneous computing in VLSI CAD.,Dr. Lin is a recipient of the Best Paper Awards at premier EDA/CAD journals/conferences, such as TCAD, DAC, DATE, and ISPD.

**Cheng Zhuo** (Senior Member, IEEE) received his B.S. and M.S. in Electronic Engineering from Zhejiang University, Hangzhou, China, and his Ph.D. in Computer Science & Engineering from the University of Michigan, Ann Arbor. He is currently a Professor at Zhejiang University and serves as the Director of the Institute of Computational Intelligence. His current research interests focus on hardware intelligence, machine learning-assisted design automation, and low power designs. He has published over 150 technical papers and received 10 Best Paper Awards and Nominations, as well as 2 international design contest awards. He is also the recipient of ACM/SIGDA Meritorious Service Award and Technical Leadership Award, JSPS Faculty Invitation Fellowship, Humboldt Research Fellowship, etc. He has served on the organization/technical program committees of many international conferences, as the area editor for Journal of CAD&CG, and as Associate Editor for IEEE TCAD, ACM TODAES, and Elsevier Integration. He is IEEE CEDA Distinguished Lecturer, a senior member of IEEE, and a Fellow of IET.