Advanced gate-level glitch modeling using ANNs

Anastasis Vagenas[†], Dimitrios Garyfallou[†], Nestor Evmorfopoulos, and George Stamoulis
Department of Electrical and Computer Engineering, University of Thessaly, Volos, Greece
{avagenas, digaryfa, nestevmo, georges}@e-ce.uth.gr

ABSTRACT

Multiple Input Switching (MIS) effects commonly induce undesired glitch pulses at the output of CMOS gates, potentially leading to circuit malfunction and significant power consumption. Thus, accurate and efficient glitch modeling is crucial for the design of high-performance, low-power, and reliable ICs. In this work, we present a new gate-level approach for modeling glitch effects under MIS. Unlike previous studies, we leverage efficient Machine Learning (ML) techniques to accurately estimate the glitch shape characteristics, propagation delay, and power consumption. To this end, we evaluate various ML engines and explore different Artificial Neural Network (ANN) architectures. Moreover, we introduce a seamless workflow to integrate our ANNs into existing standard cell libraries, striking an optimal balance between model size and accuracy in gate-level glitch modeling. Experimental evaluation on gates implemented in 7 nm FinFET technology demonstrates that the proposed models achieve an average error of 2.19% against SPICE simulation while maintaining a minimal memory footprint.

CCS CONCEPTS

• **Hardware → Electronic design automation**; **Standard cell libraries**; • **Computing methodologies → Machine learning**.

KEYWORDS

Machine Learning, Neural Networks, Multiple Input Switching, Glitch, Timing, Power, Gate-Level Models, Standard Cell Library

ACM Reference Format:
Anastasis Vagenas, Dimitrios Garyfallou, Nestor Evmorfopoulos, and George Stamoulis. 2024. Advanced gate-level glitch modeling using ANNs . In *61st ACM/IEEE Design Automation Conference (DAC '24), June 23–27, 2024, San Francisco, CA, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3649329.3657351

1 INTRODUCTION

With continuous technology scaling, accurate and efficient glitch modeling plays an ever-increasing role in successfully designing high-performance, low-power, and reliable Integrated Circuits (ICs). During logic transitions, undesired glitch pulses are commonly generated due to Multiple Input Switching (MIS), particularly in the presence of race conditions [1–6]. Increasing operating frequencies and process-induced variations exacerbate MIS-generated glitches, especially in dense designs performing many operations, such as mobile and artificial intelligence applications [1, 2]. Once generated, these spurious transitions may propagate through the circuit, violating the gate noise margins [7] and resulting in timing and logic errors, compromising the overall system performance and reliability [8]. Moreover, glitches represent 20-30% of total power consumption [2], with contributions reaching 70% in arithmetic units [5], significantly affecting the energy efficiency of power-sensitive designs.

While SPICE simulation [9] is the most reliable method for accurate timing, power, and noise analysis considering glitches, it suffers from excessive simulation times. As a result, practical analysis is performed at the gate level using pre-characterized models [10] of standard cell libraries. However, industrial models assume Single Input Switching (SIS) events, disregarding MIS effects. To approximate glitch characteristics, commercial tools apply empirical user-defined scaling factors to the related SIS-based values [11], resulting in up to 40% glitch power estimation error compared to SPICE [5].

Over the years, numerous studies have focused on gate-level glitch modeling [2–6]. In the early study presented in [2], researchers provide fundamental guidelines for glitch power analysis but do not propose any specific model. Authors in [3] compute the individual full-swing output ramps for the switching inputs and then reschedule them to approximate glitch shape and peak voltage. However, they use additional pre-characterized parameters that neglect the impact of input signal slew, temporal distance, and output load. In [4], multiple switching inputs are collapsed into two dominant ones, and the supply current is pre-characterized for power analysis, ignoring glitch shape and positioning. Aiming for accurate timing and power modeling, authors in [5] characterize both the supply current and output voltage waveforms for each glitch-inducing MIS scenario. Multivariate polynomial regression is employed in [6], which decreases the required parameters of a glitch voltage waveform for timing analysis but compromises accuracy. To reduce characterization complexity, the above works restrict the number of switching inputs [4–6], utilize average input slews [3, 5, 6], and assume symmetrical glitches [5, 7] without precise estimation of peak voltage and time [3–7]. Furthermore, they overlook the internal power of generated glitches [3, 4, 6], which may exceed 50% of total power for large slew values [11, 12]. In a different approach, a complicated Multi-port Current Source Model (MCSM) was introduced in [13], which incorporates non-linear voltage-controlled circuit elements to capture the current and charge injected at each gate input/output pin under MIS. Nonetheless, it primarily focuses on timing and lacks evaluation for glitch generation. The common drawback of all prior works [3–6, 13] is the significant characterization effort and library size required to model a wide range of MIS scenarios, which can render them impractical for gate-level glitch modeling.

† These authors contributed equally to this work.

In this paper, we introduce a novel Machine Learning (ML) approach for gate-level glitch modeling under MIS. Contrary to waveform pre-characterization and derate-based methods [4–6, 11, 13], this is the first study that employs ML techniques to estimate glitch shape, propagation delay, and power consumption. To achieve this, we assess several ML models, including linear regression, XGBoost, and Artificial Neural Networks (ANNs), and explore different ANN architectures. Moreover, we present a framework to seamlessly integrate our ANN models into standard cell libraries, achieving an optimal balance between model size and accuracy, thus enabling advanced glitch modeling capabilities. Evaluation on combinational gates implemented in 7 nm FinFET technology reveals that our ANNs yield an average error of 2.19% over SPICE simulation while maintaining a minimal memory footprint. As a result, our efficient models can enhance the accuracy of gate-level timing, power, and reliability analysis, and effectively guide glitch reduction techniques [14].

The rest of the paper is organized as follows. In Section 2, we define the problem formulation, providing background material on MIS-induced glitch effects and the critical parameters affecting them. Section 3 presents the proposed ML approach for glitch modeling and our framework for ANN-based library generation. The overall evaluation of our models is presented in Section 4. Finally, Section 5 concludes this work.

2 BACKGROUND

CMOS logic gates are constructed using complementary PMOS-NMOS transistor networks to implement the desired logic function. The timing and power characteristics of these gates depend on several parameters, such as the transistor technology, the input signals, the gate state, and the internal/external parasitics. Although SPICE simulation [9] is considered the golden standard for accurate timing and power analysis, simulating millions of gates in modern ICs throughout the design cycle, especially during iterative optimization flows [7], is infeasible in terms of runtime and memory requirements.

To overcome this challenge, typical Electronic Design Automation (EDA) tools employ gate-level models such as the Composite Current Source (CCS) model [10] and the Non-Linear Delay/Power Model (NLDM/NLPM). These models capture the timing and power information into multi-dimensional Look-Up Tables (LUTs), pre-characterized within standard cell libraries. A crucial assumption in this approach is that only SIS events are considered, meaning that a single input switches state. SIS-based timing models capture the gate's output timing characteristics (e.g., NLDM delay/slew values or CCS output current), while SIS-based power models represent the different power components (e.g., NLPM leakage/internal power and CCS leakage/supply current). In both cases, timing and power information is modeled as a function of input slew, output load, and gate state.

In reality, however, multiple inputs may switch simultaneously, causing speed-ups, slow-downs, or even glitches [1–8, 12, 13, 15]. These MIS-induced effects impact the entire circuit's operation, introducing excessive pessimism or optimism in timing and power estimation, as well as logic errors due to glitches.

2.1 Glitch generation under MIS

A glitch is an undesired voltage swing, either complete or partial, that occurs at the output of a CMOS logic gate. Glitches can be generated due to various factors, such as MIS, crosstalk [7], and ionizing particle strikes [16]. In the case of MIS, multiple input signals switch from an initial steady state (ST_{init}) to a final steady state (ST_{fin}) that produces the same logic output, where ST_k denotes the state k of all the gate input pins. During the MIS scenario $ST_{\text{init}} \rightarrow ST_{\text{fin}}$, the individual inputs can drive the output in opposite directions, thus generating a glitch. To better illustrate this, Fig. 1 demonstrates an example of glitch generation for the $10 \rightarrow 01$ MIS scenario of a 2-input NAND2 gate, where $ST_{\text{init}} = 10$ and $ST_{\text{fin}} = 01$. As can be seen, the output initially remains at logic high for ST_{init} (phase a). Then, as the input signals transition towards ST_{fin}, the currently active transistors start to switch off in both pull-up and pull-down networks. However, different paths are formed depending on the slews and arrival times of the transitioning signals. In the provided example, the second signal arrives slightly earlier, with the gate input temporarily staying at a glitch state ST_{GL} (i.e., $ST_{\text{GL}} = 11$). This state leads to the discharge of the output load due to the open path towards the VSS rail (phase b). The duration of ST_{GL} impacts the glitch shape characteristics, such as height and width, as well as its power consumption (i.e., internal and switching power). Subsequently, as the inputs reach ST_{fin}, the pull-up network begins recharging the output load (phase c), eventually restoring the output to its logic high state (phase d).

2.2 Critical parameters for glitch characteristics

As we have seen, MIS may generate a glitch at the gate output, but not all glitches exhibit the same shape characteristics (e.g., width, height) and power consumption. A generated glitch heavily depends on the parameters of the underlying MIS scenario, and most importantly on the slew values and arrival times of the input signals, as well as the output load value. To assess the impact of each MIS variable, we sweep each variable for the MIS scenario of Fig. 1 separately while keeping the others fixed at constant values. Below, we discuss the impact of each parameter on the shape and power consumption of the generated glitch.

1) Output load: The gate output load (C_L) primarily controls the total charge required to alter the output voltage level. A smaller load necessitates less time spent in the ST_{GL} state, resulting in glitches with larger heights. This is illustrated in Fig. 2a, where for small values of C_L (0.72 fF and 2.69 fF), the glitch drops below 50% of VDD (0.35 V). However, as C_L increases (above 4.66 fF), the glitch height becomes negligible while the width and power consumption increase ever so slightly.

2) Skew: Skew (Δ_{xy}) is the temporal distance between the arrival times of two signals x and y. The value of Δ_{xy} strongly affects the time spent in the ST_{GL} state. As shown in Fig. 2b, the incremental increase of Δ_{AB} leads to an overall increase in glitch height, width, and power consumption. For large values of Δ_{AB} (above 63.75 ps), the increased duration of ST_{GL} induces significant glitches that fall below 50% of VDD.

3) Input slew: The slew value (S_x) of an input signal x also plays a crucial role in influencing the duration of the ST_{GL} state. Fig. 2c depicts this effect, where progressively smaller S_A values (below

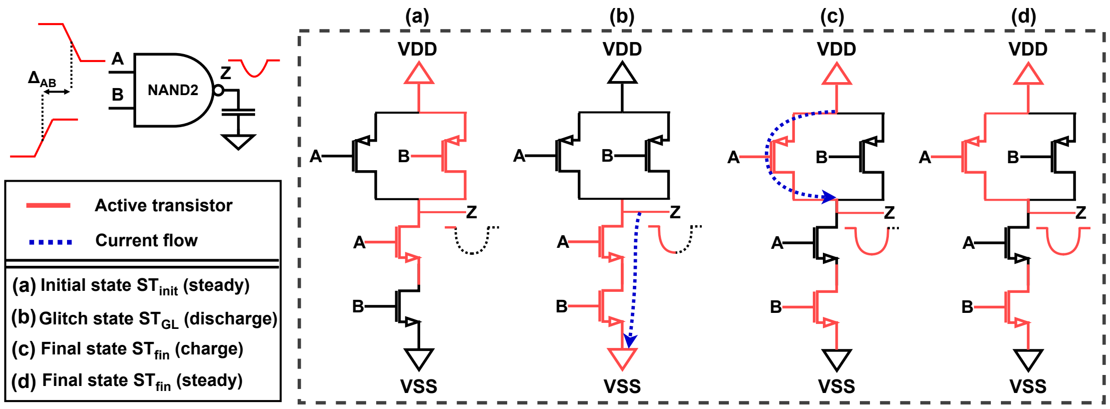

(a)
(b)
(c)
(d)
VDD
VSS
A
B
Z
NAND2
Δ_{AB}
Active transistor
Current flow
(a) Initial state ST$_{init}$ (steady)
(b) Glitch state ST$_{GL}$ (discharge)
(c) Final state ST$_{fin}$ (charge)
(d) Final state ST$_{fin}$ (steady)

Figure 1: Transistor-level demonstration of glitch generation during the 10 → 01 MIS scenario for a 2-input NAND2 gate.

42.5 ps) increase the effective duration of the ST_{GL} state, inducing significant glitches at the gate output.

Finally, the glitch characteristics and power consumption are also affected by the type of gate, with some gates exhibiting only rise or fall glitches (i.e., towards VDD or VSS rail voltage, respectively), the drive strength of the gate, the operating conditions (i.e., Process, Voltage, Temperature [PVT]), and the transistor technology.

2.3 Problem formulation

Considering the previous discussion on the parameters affecting the characteristics of an MIS-generated glitch, the problem addressed in this work is formulated as follows.

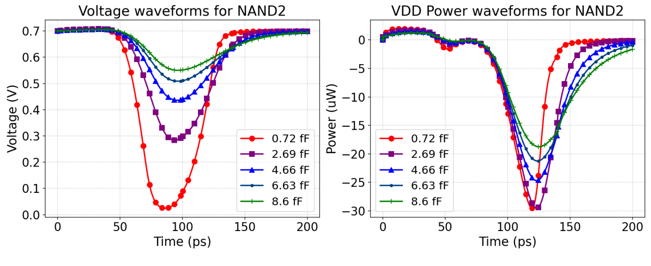
Glitch modeling problem: Given an M-input gate under an MIS scenario $ST_{\mathrm{init}} \longrightarrow ST_{\mathrm{fin}}$ of N switching input signals, the slew values $S = \{S_1, ..., S_N\}$ and arrival times $AT = \{AT_1, ..., AT_N\}$ of the input signals, and the C_L value, the objective is to effectively model the output voltage waveform shape, the propagation delay, and the power consumption of the generated glitch.

3 PROPOSED APPROACH

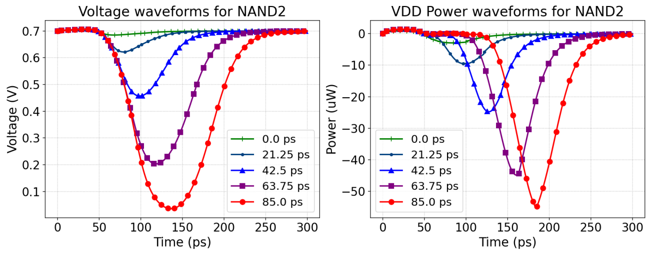
In this section, we present our approach for MIS-generated glitch modeling. First, we introduce the proposed ML models, evaluating different ML engines and ANN architectures. Then, we present our ANN-based library generation flow.

3.1 Input features

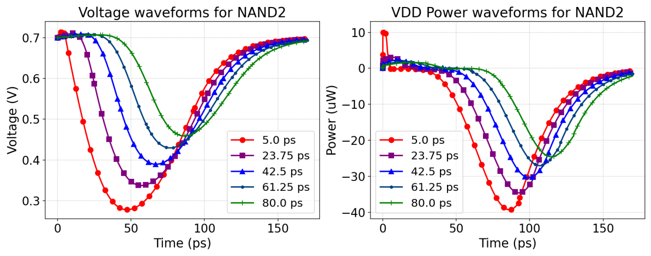
The most crucial aspect of building an ML model for glitch shape and power prediction is the selection of the input features. Based on our observations and the problem formulation presented in the previous section, we adopt the preceding MIS parameters as the input features for our ML models. Therefore, for a general M-input gate under an MIS scenario with N switching inputs, our ML models require a total of $2N$ inputs: N input slew values $S = \{S_1, ..., S_N\}$, $(N-1)$ skew values $\Delta = \{\Delta_1, ..., \Delta_{N-1}\}$, and the C_L value. Fig. 3 provides an overview of the input features for a two-input MIS scenario of a NAND2 gate. Firstly, an input signal x is characterized by three time instants, i.e., low ($T_{L(x)}$), delay ($T_{D(x)}$), and high ($T_{H(x)}$), corresponding to the points when the signal x crosses the designated library voltage thresholds (e.g., 10%, 50%, and 90% of



(a) Variable C_L value (S_A = 80 ps, S_B = 80 ps, Δ_{AB} = 40 ps).



(b) Variable Δ_{AB} value (S_A = 80 ps, S_B = 80 ps, C_L = 5.76 fF).



(c) Variable S_A value (S_B = 80 ps, Δ_{AB} = 40 ps, C_L = 5.76 fF).

Figure 2: Impact of critical parameters on glitch shape and power consumption for the 10 → 01 MIS scenario of a NAND2.

VDD, respectively). By utilizing these time instants, we can calculate the slew of a signal x, as $S_x = T_{H(x)} - T_{L(x)}$, and the skew between signals x and y, as $\Delta_{xy} = T_{D(y)} - T_{D(x)}$. For the NAND2 gate in particular, $S_A = T_{\mathrm{H(A)}} - T_{\mathrm{L(A)}}$ and $\Delta_{\mathrm{AB}} = T_{\mathrm{D(B)}} - T_{\mathrm{D(A)}}$. Finally, C_L is determined using typical effective capacitance methodologies [17].

3.2 Glitch modeling and ML model outputs

The next critical aspect is the accurate and efficient modeling of glitches regarding their shape, position, and power consumption. The glitch shape's most vital attributes are its height (H_{GL}) and width (W_{GL}), which predominantly control whether propagation to subsequent gates is possible [7]. It is crucial to note that H_{GL} also enables the calculation of glitch switching power [3], as $E_{\mathrm{SW}} = \frac{VDD \cdot H_{\mathrm{GL}} \cdot C_L}{2}$. On the other hand, most approaches and industrial models assume symmetrical glitches ignoring the time-to-peak value (TP_{GL}) [5, 10], which is essential for accurate noise and dynamic timing analysis [7, 18]. Furthermore, determining the location of a glitch relative to the switching inputs requires a delay value (D_{GL}), similar to the SIS-based NLDM delay, which corresponds to the glitch propagation delay. Lastly, the power consumption of an MIS-generated glitch can be modeled as the total energy (E_{GL}) consumed during the glitch event.

The above five parameters (H_{GL}, W_{GL}, D_{GL}, TP_{GL}, E_{GL}), shown in Fig. 3, are used as the target output of our ML models to effectively capture the glitch characteristics for an MIS scenario. Firstly, the output voltage waveform is characterized by three time instants, T_{start}, T_{end}, and T_{peak}. T_{start} and T_{end} refer to the first and last time instants when the glitch waveform crosses predetermined VDD thresholds (e.g., 5% for rise glitches and 95% for fall glitches), while T_{peak} corresponds to the time instant of the glitch peak voltage (V_{peak}). Utilizing the above, we can calculate $W_{\mathrm{GL}} = T_{\mathrm{end}} - T_{\mathrm{start}}$, $H_{\mathrm{GL}} = |V_{\mathrm{rail}} - V_{\mathrm{peak}}|$ (where V_{rail} is VDD or VSS), and $TP_{\mathrm{GL}} = T_{\mathrm{peak}} - T_{\mathrm{start}}$. In addition, we calculate $D_{\mathrm{GL}} = T_{\mathrm{peak}} - T_{D(x)}$, where x is assumed to be the latest arriving input signal. E_{GL} is obtained by integrating the entire power waveform and subsequently removing the leakage component. Note that our modeling also allows for the calculation of glitch internal power, as $E_{\mathrm{INT}} = E_{\mathrm{GL}} - E_{\mathrm{SW}}$ [3].

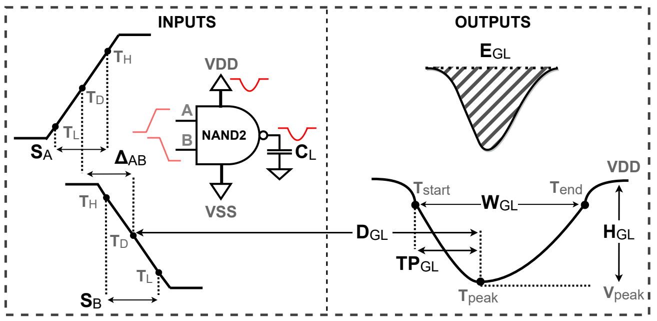


Figure 3: Features and outputs of our ML-based glitch models.

3.3 ML model selection

To assess the effectiveness of our feature selection and glitch modeling, we investigated three types of ML models, namely, linear regression (LIN), XGBoost (XGB), and ANNs. Each distinct MIS scenario of each gate is characterized by five single-output ML models, each corresponding to one of the parameters outlined in the previous section. In our experiments, multi-output models either performed poorly in accuracy or exhibited substantial size compared to multiple single-output models. The dataset for each specific MIS scenario of a gate, acquired through SPICE simulations, was split into two sets, with 80% used for training and 20% for testing. [1] Before training, we normalized all input features and scaled all output values into the appropriate scale. The normalization of the input features ensures that no input dominates over the others, while scaling the outputs enhances numerical stability. Moreover, the XGB models were trained using up to 80 trees (estimators), with a maximum depth of 6, to avoid large model sizes and overfitting. The ANNs were trained using the Adam optimizer, a maximum number of 200 epochs, a [2N, 16, 8, 1] architecture (i.e., 2N input features, two hidden layers with 16 and 8 neurons, and a single-neuron output layer), and the ReLU activation function for the hidden layers (without activation for the output). The rest of the hyper-parameters for the ANN and XGB models (e.g., batch size for ANNs and leaf samples for XGB) were selected via Bayesian search and optimization. All models utilize some form of L1/L2 regularization term to overcome overfitting, while the Mean Squared Error (MSE) loss was used during the training process. Lastly, for each gate, we report the mean R^2 score against SPICE across all ML models of all MIS scenarios as well as their combined memory footprint (when stored in binary format).

As shown in Table 1, our ANN models outperform the other models in terms of accuracy against SPICE (R^2 over 0.998) and require small amounts of memory (less than 0.06 MB) for all gates. Considering accuracy, the LIN models perform the worst while the XGB models achieve R^2 scores comparable to the ANN models in most cases (e.g., 0.996 over 0.999 for NAND2). Moreover, the ANN models, while not as memory-efficient as the LIN models, consume considerably less memory than the XGB models (with up to 350× improvement). Therefore, ANNs constitute the most appealing choice for glitch modeling, considering their superior accuracy and minimal memory footprint.

Table 1: Accuracy over SPICE (R^2 score) and total memory requirements of the examined ML models.

Gate	Accuracy (R^2)			Size (MB)		
	LIN	XGB	ANN	LIN	XGB	ANN
NAND2	0.531	0.996	0.999	0.002	2.90	0.01
NOR2	0.530	0.997	0.999	0.002	3.56	0.01
NAND3	0.445	0.991	0.999	0.01	21.51	0.06
NOR3	0.440	0.992	0.999	0.01	21.54	0.06
XOR2	0.343	0.989	0.998	0.004	7.17	0.02
XNOR2	0.347	0.989	0.998	0.004	6.61	0.02

3.4 ANN architectures

Given the highly configurable architecture of ANNs, we also explored the accuracy/size trade-off of six alternative architectures, focusing on the NAND2 and NAND3 gates. As listed in Table 2, the addition of neurons or layers to an existing ANN model leads to an improvement in accuracy for both gates. Interestingly, we observed that for the same level of accuracy, adding layers to an existing ANN architecture results in significantly (up to 2×) smaller models, compared to adding additional neurons (e.g., [2N, 16, 16, 1] vs. [2N, 8, 8, 8, 1]).

The customizable nature of ANNs can be exploited for generating glitch models to be included in a standard cell library, reducing disk

[1] You can find details about the dataset and the experimental setup in Section 4.

space requirements without compromising accuracy. It is important to highlight that a conventional LUT-based library would require five 4D or 6D LUTs (for a 2-input or 3-input gate) to capture all glitch characteristics for a specific MIS scenario. Even assuming 8 characterization points per input variable (i.e., feature), a 6D LUT would store 8^6 values. In contrast, a [6, 8, 8, 1] ANN model would be represented by 137 parameters (120 weight and 17 bias values), achieving an impressive compression ratio of 1913×.

Table 2: Accuracy over SPICE (R^2 score) and total memory requirements of the examined ANN architectures.

ANN architecture	NAND2 (N = 2)		NAND3 (N = 3)	
	Accuracy (R^2)	Size (KB)	Accuracy (R^2)	Size (KB)
[2N, 16, 1]	0.9909	3.79	0.9823	26.48
[2N, 32, 1]	0.9961	7.54	0.9911	52.73
[2N, 8, 8, 1]	0.9954	4.73	0.9944	30.23
[2N, 16, 16, 1]	0.9996	14.41	0.9995	90.23
[2N, 4, 4, 4, 1]	0.9939	2.54	0.9660	16.17
[2N, 8, 8, 8, 1]	0.9994	7.54	0.9992	47.11

3.5 Library generation flow for ANN-based glitch modeling

In this section, we provide an overview of our library generation framework for gate-level ANN-based glitch modeling. [2] As demonstrated in Fig. 4, our framework requires the SPICE models of the gates for producing the golden ML data using SPICE simulations and a configuration file that defines the available ANN architectures and model accuracy/size constraints, along with the characterization parameters (e.g., skew limits) and simulation options (e.g., timestep and solver). Optionally, a standard cell library can also be provided to extract the logic function and utilize the existing SIS slew/capacitance ranges for each gate. The core engine parses the input files and performs SPICE simulations to produce the ML training data based on the provided configuration. More specifically, the engine analyzes each gate for all glitch-inducing MIS scenarios ($ST_{\text{init}} \rightarrow ST_{\text{fin}}$), generates the netlist for each data point, and collects the golden SPICE results. Note that we initially collect 5-50k data points for each MIS scenario, depending on the number of switching inputs.

Following the dataset generation, we create the proposed ANN models. For each ANN, we select its architecture from the pool of available architectures taking into account any size limitations. The ANN is then trained using the data points of the respective MIS scenario and evaluated considering the accuracy prerequisites. In case the model does not achieve the target accuracy (i.e., a user-defined metric such as R^2 score), we opt for a more complex architecture. In the absence of other available architectures, we augment the dataset with additional data points, restarting the selection process. Once the ANN models for all gates are finalized, we store them within the standard cell library (in ASCII format) or as separate binary files. In both cases, we store the ANN architecture and parameters (weights and biases) for all MIS scenarios.

[2] Our proposed approach is available at https://github.com/anasvag575/MIS_modeling.

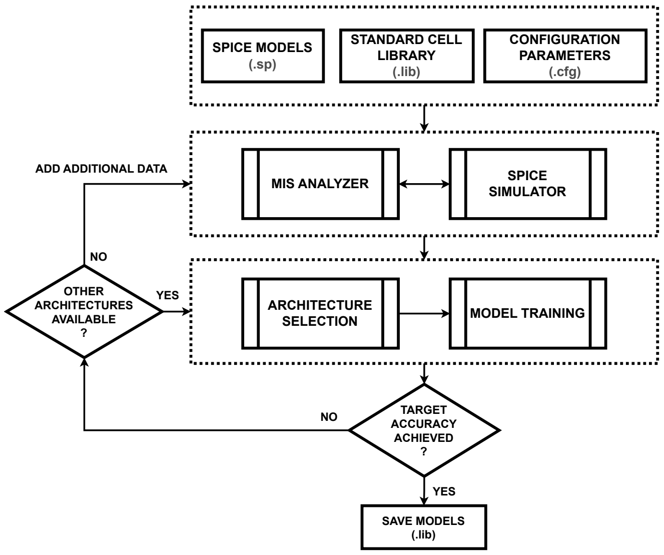


Figure 4: Proposed workflow for ANN-based glitch modeling.

4 EXPERIMENTAL EVALUATION

To evaluate our ML-based glitch modeling approach, we utilized the framework presented in Section 3.5, performing SPICE simulations of combinational gates from the ASU ASAP 7 nm FinFET PDK [19]. The selected gates include simple (e.g., NAND, OR) and complex (e.g., XNOR, AOI) gates, varying from two-input (e.g., NAND2) to five-input (e.g., NAND5) gates. During simulation, the Δ_{xy} limits are selected such that a wide range of glitch effects can be observed, while the C_L and S_x limits are extracted from the corresponding standard cell library. The golden results were obtained using the Xyce Parallel Electronic Simulator [9], and the accuracy of the models was assessed using the Normalized Root Mean Square Error (NRMSE). For the framework configuration, we used the ANN architectures of Table 2, while the target model accuracy was set to 4% NRMSE. The generation of the golden SPICE results and the selection/training procedure for all the MIS scenarios of a gate typically takes 3-50 min. and 2-20 min., respectively. Finally, we ran all the experiments on a Linux workstation with a 2.60 GHz 64-thread Intel® Xeon® CPU and 128 GB of memory.

Table 3 presents the accuracy evaluation of the proposed ANN models for the examined gates. As can be seen, our ANNs correlate extremely well with SPICE across all gates and model outputs, with a maximum NRMSE of 3.98% (for D_{GL} of AOI221), a minimum NRMSE of 0.69% (for TP_{GL} of AOI21), and 2.19% global mean NRMSE across all predictions. The most accurate predictions are achieved for E_{GL} and W_{GL}, with NRMSE ranging from 0.73% (for NAND2) to 2.51% (for AOI221) and 0.73% (for AOI21) to 3.07% (for OAI221), respectively. The worst predictions correspond to H_{GL} and D_{GL}, with NRMSE ranging from 1.04% (for NAND2) to 3.76% (for AOI221) and 1.48% (for NAND2) to 3.98% (for AOI221), respectively. Note that both H_{GL} and D_{GL} depend on the glitch peak (i.e., T_{peak}, V_{peak}), which fluctuates significantly even with small perturbations in model inputs (e.g., S_x, Δ_{xy}), resulting in lower prediction accuracy.

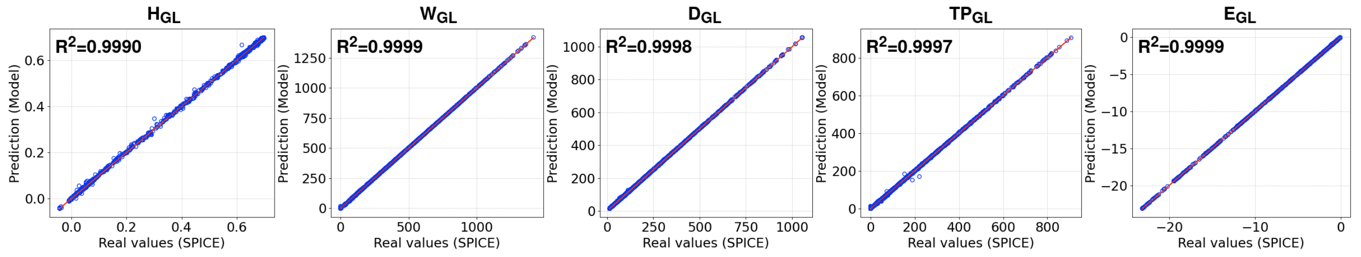


Figure 5: The predictions of the proposed ANN models for the $10 \rightarrow 01$ MIS scenario of a NAND2 gate.

Table 3: NRMSE over SPICE for the ANN models of each gate

Gate	NRMSE against SPICE					
	H_{GL}	W_{GL}	D_{GL}	TP_{GL}	E_{GL}	Mean
NAND2	1.04%	1.34%	1.48%	1.87%	0.73%	**1.29%**
NAND3	1.26%	2.11%	1.83%	2.83%	1.34%	**1.87%**
NAND4	2.12%	3.01%	2.31%	3.42%	1.94%	**2.57%**
NAND5	2.34%	3.04%	2.68%	3.85%	2.28%	**2.84%**
NOR2	2.29%	1.19%	2.12%	2.41%	1.28%	**1.86%**
NOR3	3.36%	1.84%	3.19%	3.15%	1.36%	**2.58%**
AND2	2.24%	1.37%	2.25%	1.01%	1.22%	**1.62%**
OR2	2.18%	1.79%	2.45%	1.02%	1.39%	**1.77%**
XOR2	3.68%	1.36%	3.44%	2.31%	1.55%	**2.47%**
XNOR2	2.81%	1.47%	3.72%	1.99%	1.39%	**2.28%**
AOI21	2.72%	0.73%	2.57%	0.69%	1.48%	**1.64%**
AOI211	3.58%	1.87%	3.13%	1.66%	2.01%	**2.45%**
AOI221	3.76%	2.74%	3.98%	1.92%	2.51%	**2.98%**
OAI21	1.80%	0.73%	1.96%	1.88%	0.97%	**1.47%**
OAI211	2.84%	2.12%	2.42%	3.03%	1.44%	**2.37%**
OAI221	3.23%	3.07%	3.30%	3.37%	2.07%	**3.01%**
Mean	**2.58%**	**1.86%**	**2.70%**	**2.20%**	**1.53%**	**2.19%**

Among the examined gates, the most accurate predictions are achieved for NAND2, where our ANNs exhibit 1.29% mean NRMSE considering all ML outputs. Fig. 5 illustrates the regression plots of the NAND2 models, revealing their high resistance to outliers and exceptional correlation with SPICE (R^2 over 0.999). On the other hand, the worst-performing gate is OAI221 (3.01% mean NRMSE), especially for D_{GL} and TP_{GL} predictions (3.30% and 3.37% NRMSE, respectively). Higher-input gates (NAND5 and OAI221) generally perform worse than their lower-input counterparts (NAND2 and OAI21) for all metrics, while complex gates exhibit higher mean NRMSE (2.47% for XOR2 and 2.28% for XNOR2) compared to simpler gates such as NAND2 and NOR2.

5 CONCLUSIONS

In this paper, we introduce a new ML-based approach for modeling MIS-generated glitch effects. Contrary to previous works, we employ efficient ANN models to accurately predict the glitch shape, propagation delay, and power consumption. Moreover, we present a framework that seamlessly integrates the proposed ANN models into existing libraries, exploiting the accuracy/size trade-off of different architectures, thus enabling advanced glitch modeling capabilities. Experimental results on gates implemented in 7 nm FinFET technology indicate that our models exhibit a strong correlation with SPICE, achieving 2.19% mean NRMSE across all gates, while preserving low memory requirements.

REFERENCES

[1] Synopsys - What is Glitch Power ? Accessed: Nov. 17, 2023. [Online]. Available: https://www.synopsys.com/ai/what-is-glitch-power.html
[2] M. Favalli and L. Benini, "Analysis of glitch power dissipation in CMOS ICs," in *Proc. of the International Symposium on Low Power Design (ISLPD)*, pp. 123–128, 1995.
[3] D. Rabe and W. Nebel, "New approach in gate-level glitch modelling," in *Proc. of the conference on European design automation (EURO-DAC)*, pp. 66–71, 1996.
[4] J. A. Thodiyil, "Generic gate level model for characterization of glitch power in logic cells," Nov. 25 1997, US Patent 5,691,910.
[5] M. Meixner and T. G. Noll, "Accurate estimation of CMOS power consumption considering glitches by using waveform lookup," *IEEE Trans. Circuits Syst. II Express Briefs*, vol. 64-II, no. 7, pp. 787–791, 2017.
[6] S. Abolmaali, "Efficient delay characterization method to obtain the output waveform of logic gates considering glitches," *Iranian Journal of Electrical and Electronic Engineering*, vol. 15, no. 4, p. 485, 2019.
[7] J. Bhasker and R. Chadha, *Static timing analysis for nanometer designs: A practical approach.* Springer Science & Business Media, 2009.
[8] P. Ruiz-de-Clavijo, J. Juan-Chico, M. J. Bellido, A. J. Acosta, and M. Valencia-Barrero, "HALOTIS: high accuracy logic timing simulator with inertial and degradation delay model," in *Proc. of the Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 467–471, 2001.
[9] E. Keiter *et al.*, "Xyce Parallel Electronic Simulator Users' Guide Version 7.6." Sandia National Lab. (SNL-NM), Tech. Rep., 2022.
[10] Synopsys - Composite Current Source (CCS). Accessed: Nov. 17, 2023. [Online]. Available: http://www.opensourceliberty.org/ccspaper/
[11] Synopsys - PrimePower. Accessed: Nov. 17, 2023. [Online]. Available: https://www.synopsys.com/implementation-and-signoff/signoff/primepower.html
[12] D. Rabe and W. Nebel, "Short circuit power consumption of glitches," in *Proc. of the International Symposium on Low Power Design (ISLPD)*, pp. 125–128, 1996.
[13] C. S. Amin, C. V. Kashyap, N. Menezes, K. Killpack, and E. Chiprout, "A multi-port current source model for multiple-input switching effects in CMOS library cells," in *Proc. of the 43rd Design Automation Conference (DAC)*, pp. 247–252, 2006.
[14] S. Bathla, R. M. Rao, and N. Chandrachoodan, "A Simulation-Based Metric to Guide Glitch Power Reduction in Digital Circuits," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, vol. 27, no. 2, pp. 376–386, 2019.
[15] R. Tayade, S. R. Nassif, and J. A. Abraham, "Analytical model for the impact of multiple input switching noise on timing," in *Proc. of the 13th Asia South Pacific Design Automation Conference (ASP-DAC)*, pp. 514–517, 2008.
[16] G.-I. Paliaroutis *et al.*, "Accurate soft error rate evaluation using event-driven dynamic timing analysis," in *Proc. of the International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*, pp. 1–6, 2023.
[17] D. Garyfallou *et al.*, "Gate Delay Estimation With Library Compatible Current Source Models and Effective Capacitance," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, vol. 29, no. 5, pp. 962–972, 2021.
[18] D. Garyfallou, I. Tsiokanos, N. Evmorfopoulos, G. Stamoulis, and G. Karakonstantis, "Accurate Estimation of Dynamic Timing Slacks using Event-Driven Simulation," in *Proc. of the 21st International Symposium on Quality Electronic Design (ISQED)*, pp. 225–230, 2020.
[19] ASU - ASAP7 PDK. Accessed: Nov. 17, 2023. [Online]. Available: https://github.com/The-OpenROAD-Project/asap7/