High-Fidelity Real-Time Simulation of Power Electronics Converters via FPGA-Accelerated Dynamic Connectionist Neural Network

Haowen Weng, Zixiang Liao, Yinbin Chen, and Can Wang, *Member, IEEE*

***Abstract*—The precise modeling of power electronic switches is essential for accurate real-time simulation of power converter systems, particularly under high-frequency and dynamic operating conditions. Existing simulation techniques often rely on ideal switch models or simplified behavioral models, which cannot accurately capture transient behaviors like voltage and current spikes, electromagnetic interference, and switching losses. To address these limitations, this article proposes a novel connectionist neural network-based transient switch modeling approach that integrates time-node coupling mechanisms for enhanced dynamic behavior prediction. A high-quality dataset is generated through extensive offline simulations using a verified physics-based insulated gate bipolar transistor (IGBT) model. The connectionist neural network model leverages feedback connections across time steps, significantly improving its ability to describe transient waveforms under various operating conditions. Additionally, a dynamic neuron adjustment strategy is introduced to reduce hardware resource usage by dynamically allocating neurons based on the complexity of different time nodes. This ensures high modeling accuracy while alleviating computational burden on field programmable gate array platforms.**

***Index Terms*—Electromagnetic transient real-time simulation, field programmable gate array (FPGA), neural network, physics-based switch modeling, real-time simulation architecture.**

I. Introduction

THE rapid development of renewable energy, smart grids, and electric transportation has significantly increased the demand for efficiency and reliability in power electronic systems, thereby driving advancements in real-time simulation technologies [1]. Traditional design methods relying on experience are no longer sufficient to meet the modeling and verification requirements of high-frequency switching devices, while digital design methods have been widely adopted due to their efficiency and automation advantages [2]. In real-time simulation, there is always a tradeoff between the accuracy of nonlinear device modeling and computational efficiency [3]. Current commercial simulators predominantly employ idealized system-level models, such as binary resistance model [4] and discrete circuit models [5], but these models still suffer from insufficient accuracy in capturing switching transient characteristics, thereby impacting system control optimization and reliability assessment. In fact, accurately capturing key transient phenomena such as voltage and current spikes, electromagnetic interference (EMI) noise, and switching losses is crucial for protection, thermal analysis, and control strategy validation [6]. Therefore, integrating device-level modeling into real-time simulation platforms has become an important direction for enhancing modeling fidelity and practical value.

Device-level modeling methods can generally be classified into analytical and behavioral models. The former, such as the Hefner and Diebolt [7], Kraus and Mattausch [8], and Sheng et al. models [9], are based on semiconductor physics and provide high-fidelity offline simulations. However, due to their reliance on complex nonlinear equations and process-dependent parameters, these models are computationally intensive and unsuitable for real-time applications [10]. In contrast, behavioral models aim to improve efficiency by simplifying internal device dynamics. Representative approaches include nonlinear behavioral models, piecewise linear transient models, and curve-fitting models [11]. Nonlinear behavioral models emulate switching transients through equivalent circuits, and are often combined with methods, such as admittance matrix partitioning [12], dynamic time-stepping [13], and event-driven state machines [14]. While these techniques improve performance, they remain limited in temporal resolution and require significant hardware resources. Curve-fitting methods [15], including lookup tables [16], polynomial fitting [17], and Norton equivalent-based prediction [18], reduce computational overhead but exhibit limited adaptability to varying operating conditions and are difficult to generalize across complex dynamic scenarios. Recent work [19] presented an FPGA-based modeling method for SiC MOSFETs, achieving a 10 ns simulation time step and keeping RMS error within 5% on an LLC topology. However, the method requires high hardware resources, with 132 DSPs per half-bridge module, and its step size is limited by synchronization with system-level models in complex circuits. These constraints reflect the need for more efficient and decoupled device-level modeling approaches.

Received 17 April 2025; revised 30 June 2025; accepted 29 July 2025. Date of publication 4 August 2025; date of current version 22 October 2025. This work was supported in part by the GuangDong Basic and Applied Basic Research Foundation under Grant 2022A1515240078 and in part by the Shenzhen Science and Technology Plan Project under Grant JCYJ20220818102416036 and KCXST20221021111403009. Recommended for publication by Associate Editor T. Roinila. *(Corresponding author: Can Wang.)*

The authors are with the School of Robotics and Advanced Manufacturing, Harbin Institute of Technology, Shenzhen 518055, China (e-mail: 22s153261@stu.hit.edu.cn; 23s053047@stu.hit.edu.cn; 24s153180@stu.hit.edu.cn; can.wang@hit.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TPEL.2025.3595452.

Digital Object Identifier 10.1109/TPEL.2025.3595452

In the face of challenges related to high precision and computational efficiency, there has been growing interest in data-driven modeling methods, which bypass complex physical modeling and directly learn system dynamics from input-output data. Common methods include time series models, support vector machines, and various neural networks, such as recurrent neural networks (RNNs) and long short-term memory networks [20]. However, existing technologies still face issues such as deployment complexity, poor generalization capabilities, and excessive resource demands on FPGA hardware platforms.

For example, Krishnamoorthy and Narayanan Aayer [21] relie on traditional machine learning to establish accurate models, but requires frequent retraining for different devices or topological structures. Liang et al. [22] propose a hybrid network combining k-NN and RNN, which improves modeling efficiency, but its resolution is limited due to the feedback dependency of its recurrent structure. Hari et al. [23] employ a deep feedforward network for GaN device modeling, achieving high accuracy but resulting in significant resource consumption when deployed on an FPGA. Shang et al. [24] propose an electrothermal coupling modeling method supporting a 100 ns time step, but it cannot adequately respond to faster transient processes. Zhang et al. [25] employ feature extraction and dynamic step size strategies to reduce computational complexity, but suffers from poor generalization and significant resource consumption. To address the above challenges, researchers in [26] proposed dividing the time series into discrete nodes and modeling each node separately using a feedforward neural network (FNN), thereby achieving switch modeling tasks with a time step of 5 ns and verifying the application potential of this type of structure in high-speed real-time simulation. However, the output predictions of each time node in this model are independent of one another, failing to fully leverage the inherent correlations within the sequence data, which limits the model's expressive capability and fitting accuracy. Additionally, this method still requires up to 128 DSP48s resources to model a single switch on an FPGA, imposing constraints on deployment scale. Given the real-time simulation requirements for high precision, high resolution, and low resource consumption, general-purpose neural networks struggle to meet these demands, necessitating the design of specialized models with more suitable structures and higher modeling efficiency.

To overcome these limitations, this article proposes a novel connectionist neural network architecture specifically designed for switch transient modeling. This architecture introduces explicit coupling between time nodes, enhancing model accuracy while supporting parallel pipeline deployment on FPGAs. Additionally, to reduce resource consumption, this article proposes a dynamic neuron allocation strategy that adjusts the number of hidden layer neurons based on the complexity of each time node, thereby reducing the total number of neurons in the neural network while maintaining accuracy. Experiments show that this model only requires a small amount of 47 DSP48s resources to complete the modeling task of power electronic switching devices or half-bridge arm models, while maintaining a time resolution of 5 ns and high modeling accuracy.

The rest of this article is organized as follows. Section II introduces system-level modeling and simulation algorithms.

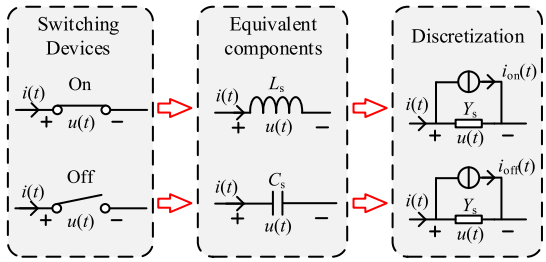

Fig. 1. Equivalent model of ADC modeling method.

Section III presents the proposed switching transient model based on a connectionist neural network. Section IV introduces the dynamic neuron allocation strategy. Section V demonstrates the model's effectiveness through case studies. Finally, Section VI concludes the article.

II. System-Level Modeling and Simulation Algorithms

The modeling method proposed in this article consists of two indispensable parts: system-level modeling and transient modeling. The former provides the necessary input conditions for the latter. Therefore, this section will discuss system-level switch modeling techniques and electromagnetic transient simulation algorithms based on node analysis.

A. System-Level Switch Modeling Methodology

The switch modeling method serves as a core component of real-time simulation in power electronic systems, directly determining the simulation's accuracy and efficiency. To address the inefficiency caused by variations in the nodal admittance matrix in ideal switch models and binary resistance models during simulation, while also meeting the stringent real-time requirements, mainstream real-time simulators primarily adopt the associated discrete circuit (ADC) approach. In the ADC modeling method, the switch's turn-ON and turn-OFF states are equivalently represented by a small inductance L_s and a small capacitance C_s, respectively, as illustrated in Fig. 1.

The historical current source expression of the ADC model has been extended in [5], establishing its relationship with the switch voltage and current from the previous time step, as expressed by

$$\begin{cases} i_{\text{on}}(t) = \alpha_{\text{on}} Y_{\text{s}} u(t-\Delta t) + \beta_{\text{on}} i(t-\Delta t) \\ i_{\text{off}}(t) = \alpha_{\text{off}} Y_{\text{s}} u(t-\Delta t) + \beta_{\text{off}} i(t-\Delta t) \end{cases} \tag{1}$$

where α_{on} and α_{off} are the voltage coefficients during turn-ON and turn-OFF, respectively, while β_{on} and β_{off} denote the corresponding current coefficients. The parameter Δt represents the real-time simulation time step.

These voltage and current coefficients are optimized through steady-state and transient response matching to achieve optimal parameter settings. Steady-state response matching ensures that the model behaves identically to an ideal switch in steady-state conditions, meaning the switch voltage is zero when conducting and the switch current is zero when turning OFF. Transient response matching optimizes damping parameters and adjusts the pole positions of the equivalent circuit to bring system poles as close as possible to the origin, thereby achieving rapid transient

response convergence and effectively reducing virtual power losses. The method for setting damping parameters references the parameter optimization strategy outlined in [5].

To further eliminate the issue of virtual power loss in the ADC switch modeling method, an initial error correction algorithm is employed, as described in [27]. This approach directly substitutes the steady-state value of the current or voltage from the previous step during switch transitions, thereby mitigating the initial error and transient oscillations caused by switching events. Consequently, the problem of virtual power loss in constant admittance switch models is effectively resolved.

B. Electromagnetic Transient Simulation Algorithm

Traditional electromagnetic transient programs (EMTPs), based on nodal analysis, involve extensive serial computation processes. These serial calculations significantly increase the computation time within a single time step, exacerbate the propagation of truncation errors, and elevate the FPGA hardware resource usage, making them unsuitable for real-time simulation. To address these challenges, this article reconfigures the node analysis method with the aim of optimizing the computational efficiency of electromagnetic transient simulation while improving parallelism. This restructuring method achieves independent parallel calculation of node voltage and branch current by merging and optimizing intermediate calculation steps. The algorithm can be expressed as follows:

$$\begin{cases} \boldsymbol{i}_{\text{temp}}(t) = \boldsymbol{j}_a(t) + \boldsymbol{j}_s(t) \\ \boldsymbol{v}_b(t) = \boldsymbol{B}\boldsymbol{i}_{\text{temp}}(t) \\ \boldsymbol{i}_b(t) = \boldsymbol{C}\boldsymbol{i}_{\text{temp}}(t) \\ \boldsymbol{j}_a(t+\Delta t) = (\boldsymbol{\alpha}+\boldsymbol{\beta})\boldsymbol{i}_b(t) - \boldsymbol{\alpha}\boldsymbol{i}_{\text{temp}}(t) \end{cases} \tag{2}$$

$$\begin{cases} \boldsymbol{B} = -\boldsymbol{A}^T\boldsymbol{Y}_n^{-1}\boldsymbol{A} \\ \mathbf{C} = -\boldsymbol{Y}_b\boldsymbol{A}^T\boldsymbol{Y}_n^{-1}\mathbf{A} + \mathbf{E} \end{cases} \tag{3}$$

where $\boldsymbol{i}_{\text{temp}}$ represents the temporary current, which is used to store the sum of all current sources in all branches. $\boldsymbol{j}_a$ denotes the current source derived from the discretization of branch elements, while $\boldsymbol{j}_s$ represents the current source. $\boldsymbol{v}_b$ and $\boldsymbol{i}_b$ correspond to the branch voltage and branch current, respectively. α and β are diagonal coefficient matrices used in switch modeling. Furthermore, $\boldsymbol{Y}_{\mathbf{n}}$ and $\boldsymbol{Y}_{\mathbf{b}}$ refer to the nodal admittance matrix and the branch admittance matrix. $\boldsymbol{A}$ is the node-to-branch incidence matrix, and $\boldsymbol{E}$ denotes the identity matrix.

By reconfiguring the EMTP algorithm, the process is reduced to two sequential steps with the number of pre-stored matrices minimized to $\boldsymbol{B}$ and $\boldsymbol{C}$ requirements, resulting in improved computational efficiency and reduced resource utilization.

III. Switching Transient Model Based on Connectionist Neural Network

Existing methods for modeling the transient behavior of power electronic switches have reduced the computational complexity to a certain extent, but they still fall short of the high-resolution requirements for real-time simulation. This is due to the extremely short time scale of the switching process, which requires a very small time step. For this reason, this article proposes a data-driven modeling method based on connectionist neural networks to achieve accurate prediction of switching transients while maintaining high resolution.

A. Data Acquisition for Neural Networks Dataset

The accuracy of neural network models largely hinges on the diversity and quality of the training data, making the acquisition of high-quality datasets a critical prerequisite. Data can be obtained either through physical testing platforms or offline simulations. However, constructing physical test platforms is both time-consuming and labor-intensive, contradicting the primary intent of real-time simulation. Consequently, current research predominantly relies on offline simulation software for data generation, which not only significantly reduces time and costs but also offers greater flexibility in controlling operating conditions, ensuring comprehensive and representative training datasets.

In the context of real-time simulation, in order to improve computational efficiency and reduce complexity, this article uses a model controlled by the steady-state results to approximate complex switching devices and their peripheral components (e.g., input/output inductors and capacitors). Specifically, the model uses the steady-state voltage V_{ce}, steady-state current I_c, and junction temperature T of the IGBT as inputs, and outputs the transient voltage v_{ce} and transient current i_c during the switching process. While gate resistance R_g and gate voltage V_g can influence the transient behavior, these parameters are typically fixed and optimized for specific applications during circuit design. Therefore, they are not treated as dynamic variables in the simulation. This article focuses on the influence of V_{ce}, I_c, and T on transient characteristics, ignoring the dynamic variation of R_g and V_g to further simplify the model and improve simulation efficiency.

To ensure the accuracy of the switching device model in LTspice, an IGBT with an antiparallel diode (FGY160T-65SPD-F085) is chosen as an example to construct a neural network-based switching model in this article. The IGBT simulation model used is provided by Onsemi, which is based on a physical mechanism and calibrated by the manufacturer to have high accuracy and reliability [28]. In order to obtain training data covering various working conditions, a Python-LTspice automation program was developed to simulate offline simulation under different switch input parameters. Its data acquisition flow and circuit are shown in Fig. 2. LTspice is used to build the test circuit, and the working state of device S_1 is set by adjusting the voltage and current sources, while the Python script completes the parameter configuration, simulation execution and data extraction. By collecting 750 ns turn-ON data from the turn-ON signal and 2500 ns turn-OFF data from the turn-OFF signal to fully reflect the process of the device from transient to steady state. Considering the FPGA clock frequency is 200 MHz, the data time step is set to 5 ns. In order to make the dataset convenient for neural network training, the data structure was transposed. In the processed dataset, each array corresponds to a specific time node, and the elements within the array represent the response of the IGBT under different working conditions at that time. The

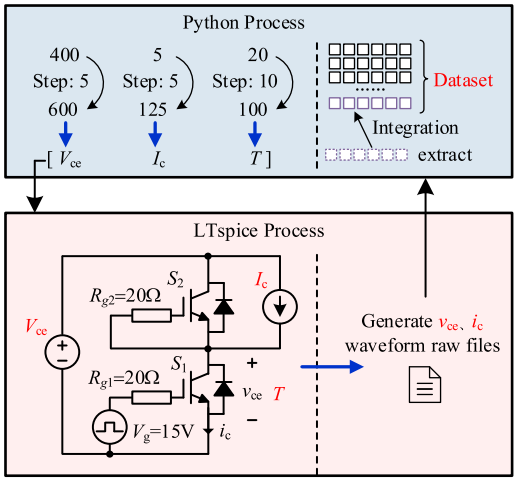

Fig. 2. Diagram of data acquisition from Python to LTspice.

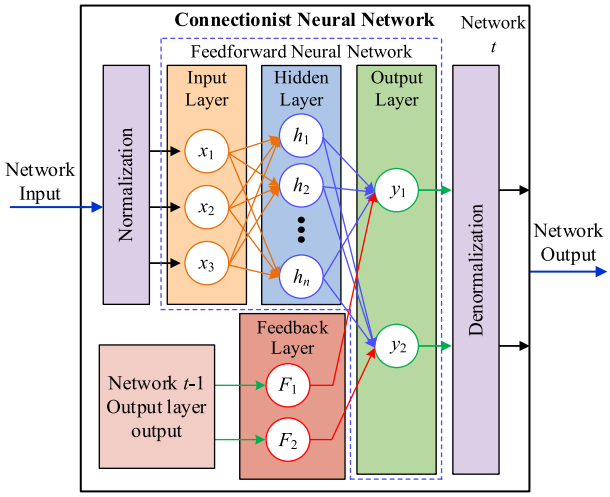

Fig. 3. Connectionist neural network architecture.

dataset is represented as follows:

$$\begin{aligned} \boldsymbol{X} &= \quad [\boldsymbol{X}_1 \quad \cdots \quad \boldsymbol{X}_i \quad \cdots \quad \boldsymbol{X}_{OP}]^{\mathrm{T}} \\ \boldsymbol{Y}_1 &= \quad [\boldsymbol{Y}_{11} \quad \cdots \quad \boldsymbol{Y}_{1i} \quad \cdots \quad \boldsymbol{Y}_{1OP}]^{\mathrm{T}} \\ &\vdots \\ \boldsymbol{Y}_j &= \quad [\boldsymbol{Y}_{j1} \quad \cdots \quad \boldsymbol{Y}_{ji} \quad \cdots \quad \boldsymbol{Y}_{jOP}]^{\mathrm{T}} \\ &\vdots \\ \boldsymbol{Y}_{TN} &= [\boldsymbol{Y}_{TN1} \quad \cdots \quad \boldsymbol{Y}_{TNi} \quad \cdots \quad \boldsymbol{Y}_{TNOP}]^{\mathrm{T}} \end{aligned} \tag{4}$$

where $\boldsymbol{X}$ represents the operating condition matrix, where $\boldsymbol{X}_i$ denotes the matrix corresponding to the ith operating condition, composed of the steady-state voltage V_{ce}, steady-state current I_c, and junction temperature T of the IGBT. In this expression, OP denotes the total number of operating conditions in the dataset, which is 9225 in this article. $\boldsymbol{Y}_j$ represents the transient output matrix at the jth time node for all operating conditions, and $\boldsymbol{Y}_{ji}$ denotes the output voltage v_{ce} and output current i_c of the IGBT at the jth time node under the ith operating condition. Finally, TN refers to the total number of time nodes.

B. Connectionist Neural Network Architecture

Although several neural network architectures have been developed, complex models are difficult to be deployed efficiently due to the requirement of real-time simulation on FPGA hardware resources. For this reason, this article proposes a lightweight connectionist neural network architecture to enhance the modelling capability of dynamic characteristics of power electronic switches. The network takes advantage of the continuity and time-dependent nature of the transient process of power electronic switches, and by introducing the output of the previous moment as part of the current network input, it enables the model to learn the temporal characteristics, thus improving the prediction accuracy. The structure of the network is shown in Fig. 3, which adds a feedback layer to the traditional FNN, and its output results in the following computational expression:

$$y(t) = Y(t) + W_F(t)y(t-1) \tag{5}$$

where $y(t)$ and $W_F(t)$ denote the output of the neural network output layer and the feedback layer weights at time node t, respectively, $Y(t)$ denotes the original output layer result of the neural network at time node t, i.e., the hidden layer output weighted and biased result.

As indicated by (5), the computation of $y(t)$ depends on the result from the preceding time step, $y(t\text{-}1)$. This sequential dependence requires serial processing, which hinders parallel execution and independent staging of the computation, leading to a stagnant pipeline. To solve this, a computational step is designed specifically for connectionist neural networks. The first step is to expand the $y(t\text{-}1)$ expression as

$$y(t) = Y(t) + W_F(t)\left[Y(t-1) + W_F(t-1)y(t-2)\right]. \tag{6}$$

Subsequently, a further expansion of $y(t\text{-}2)$ is given as

$$\begin{aligned} y(t) &= Y(t) + W_F(t)W_F(t-1)\left[\frac{Y(t-1)}{W_F(t-1)}\right. \\ &\quad \left. + Y(t-2) + W_F(t-2)y(t-3)\right]. \end{aligned} \tag{7}$$

By transforming the original dependency on $y(t\text{-}1)$ into a dependency on earlier time nodes, such as $y(t\text{-}3)$, sufficient time is created for performing the required weighting working conditions without pipeline stalls. Finally, the computation of $y(t)$ can be executed using the high-throughput pipeline architecture illustrated in Fig. 4. The entire process is partitioned into four consecutive stages, each operating concurrently on the FPGA, delivering one valid output every 5 ns at a 200 MHz clock rate. The data expressions annotated in gray depict the temporal alignment of signals and intermediate results across pipeline stages to ensure correct synchronization.

Fig. 5 compares the mean square error (MSE) of the connectionist neural network model (the proposed model) with that of the FNN (the traditional model) at each time node. To ensure a fair and meaningful comparison, the baseline FNN model is configured with the same input layer, hidden layer, and output layer architecture as the proposed connectionist neural network model, excluding the feedback connections unique to the latter.

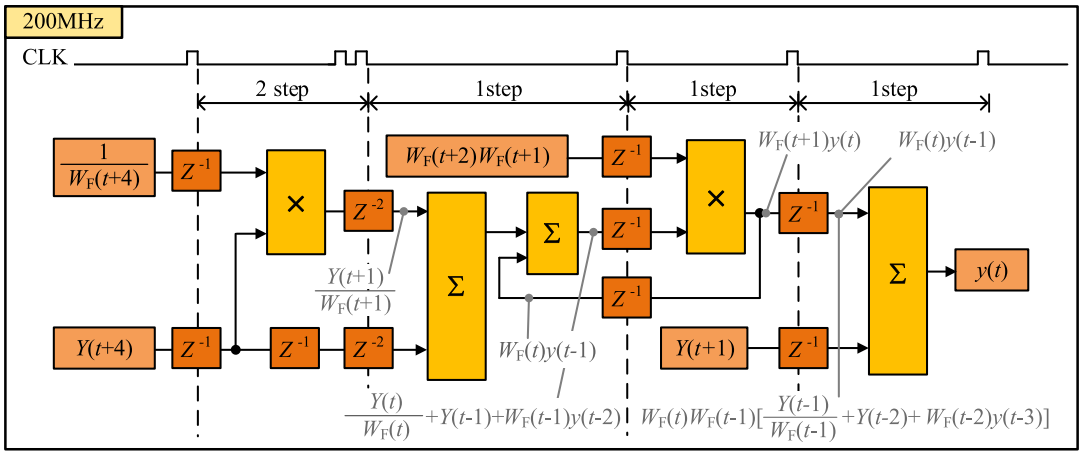

Fig. 4. Schematic diagram of high-throughput pipeline FPGA implementation of connectionist neural network.

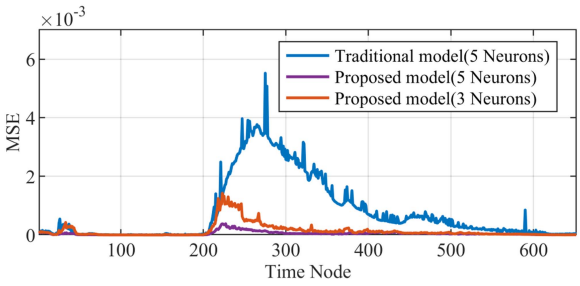

Fig. 5. Comparison of MSE for the neural network-based switch model at different time nodes.

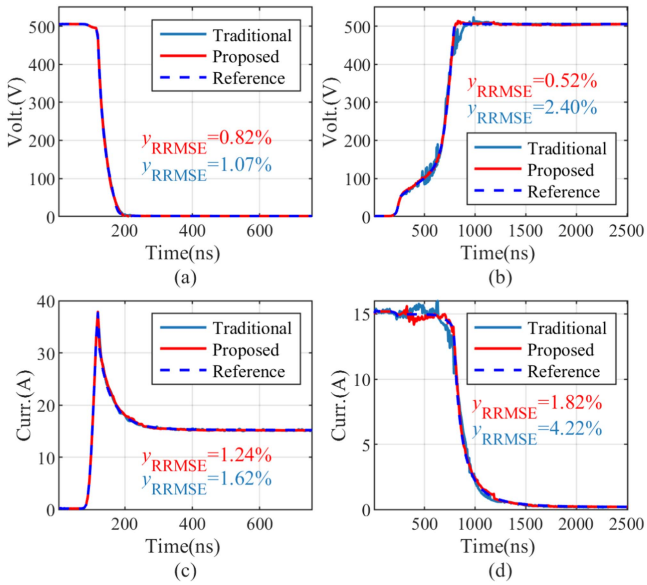

Fig. 6. Comparison of output responses among the proposed model, traditional model, and reference model. (a) v_{ce} of turn-ON transient. (b) v_{ce} of turn-OFF transient. (c) i_c of turn-ON transient. (d) i_c of turn-ON transient.

The number of hidden neurons differs between the two models to explore performance under varying model complexities. Moreover, both models are trained using an identical dataset, training algorithm, and number of epochs, thereby ensuring consistency across experimental conditions and isolating the structural differences as the sole variable affecting performance. The comparison results show that the accuracy of the proposed model is significantly higher than that of the traditional model with the same number of hidden layer neurons. Even when the number of hidden layer neurons is reduced to three, the proposed model still maintains excellent accuracy. Fig. 6 illustrates the voltage and current waveforms of the switching device under operating conditions characterized by a steady-state voltage V_{ce} of 505 V, steady-state current I_c of 15 A, and a device temperature T of 50°C. The simulation results of the traditional model with five hidden layer neurons, the proposed model with three hidden layer neurons, and the LTspice model (reference model) are compared. It is evident that the proposed model achieves superior alignment with the reference curve, even with a reduced number of hidden layer neurons. The relative root mean square error y_{RRMSE} is employed as the error metric, which is defined as

$$y_{\mathrm{RRMSE}} = \sqrt{\frac{\sum_{j=1}^{N}\left(y_{\mathrm{pro},j}-y_{\mathrm{ref},j}\right)^2}{\sum_{j=1}^{N}\left(y_{\mathrm{ref},j}\right)^2}} \times 100\% \qquad (8)$$

where $y_{\mathrm{ref},j}$ and $y_{\mathrm{pro},j}$ denote the output values of the reference model and the proposed model at the jth time step, and N denotes the total time nodes of the transient process.

IV. Optimization of Neural Network Computational Resources for Real-Time Simulation

A. Complexity of Temporal Nodes in Switching Transients

In connectionist neural network-based switch transient modeling, achieving high-fidelity real-time simulation requires an accurate approximation of the transient behavior of power electronic switches. However, the complexity of the dataset varies greatly in different time nodes, resulting in the traditional fixed neuron allocation strategy failing to balance the model accuracy and hardware resource utilization. Specifically, the time nodes with high data complexity are limited by the insufficient number

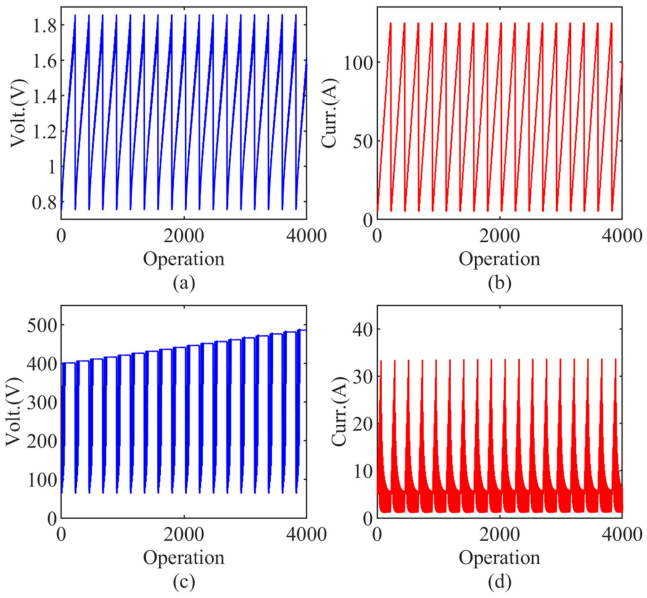

Fig. 7. Time-node dataset visualization. (a) Voltage at time node 170. (b) Current at time node 170. (c) Voltage at time node 267. (d) Current at time node 267.

of hidden layer neurons which will result in lower fitting accuracy, while the redundant hidden layer neurons in the time nodes with low data complexity will result in the inefficient utilization of computational resources, thus restricting the performance of the model on the FPGA platform.

As shown in Fig. 7, only the first 4000 working conditions are selected for visualization in order to show the distribution of the dataset under different time nodes more clearly. It is easy to find that at relatively simple time nodes, e.g., the 170th time node in Fig. 7(a) and (b), the voltage and current datasets exhibit obvious linear characteristics, which enables the neural network to achieve accurate fitting with fewer hidden layer neurons. In contrast, at more complex time nodes, such as time node 267 in Fig. 7(c) and (d), the dataset exhibits stronger nonlinear characteristics. This is mainly due to the fact that this time node is at a critical stage of the switching transient process, and the transient processes under different working conditions are not synchronized. For example, some conditions are still in the initial stage of transient at this time node, while others are close to steady state. This phase difference leads to a more complex distribution of voltage and current data sets. Therefore, more neurons need to be allocated to this time node compared to the simple time node to enhance the expressive power of the network.

B. Dynamic Neuron Allocation Strategy

In order to improve the modeling accuracy and take into account the hardware resource utilization, a dynamic neuron allocation strategy (DNAS) is introduced in this article. Unlike the fixed neuron allocation method that cannot satisfy the computational demands of different time nodes, the proposed strategy reallocates the neurons in different time steps to ensure

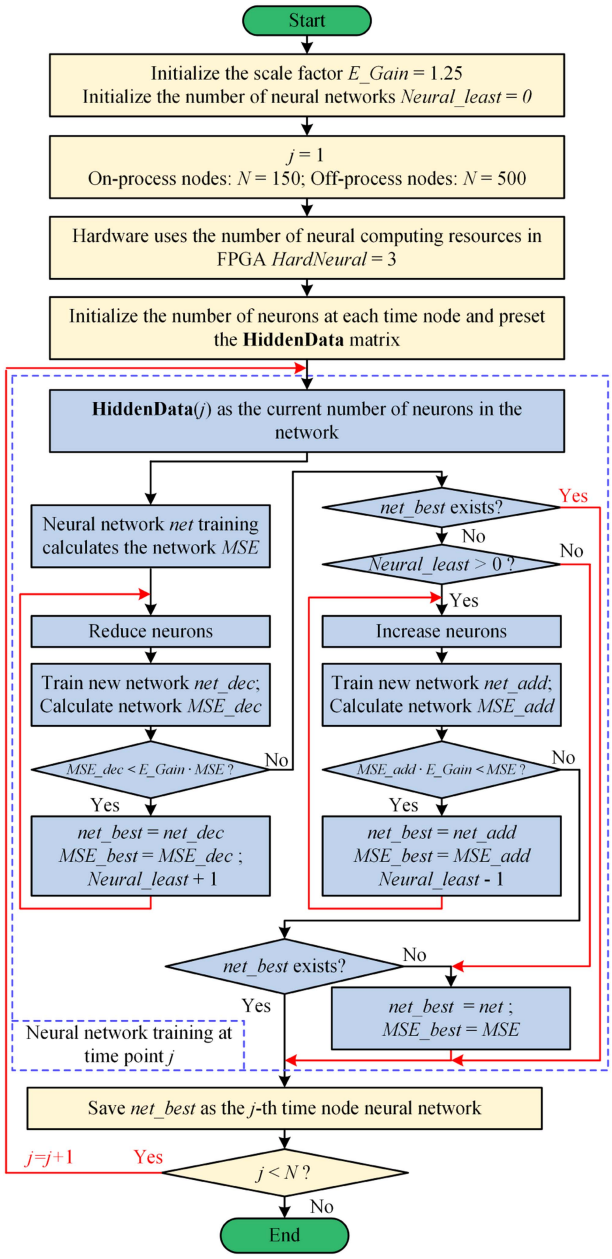

Fig. 8. Neural network training flow chart of neuron dynamic adjustment strategy.

the optimal resource utilization in FPGA-based real-time simulation. The basic principle of DNAS is to determine the optimal allocation of neurons using the model's error distribution in all time nodes. The basic principle of DNAS is to determine the optimal allocation of neurons using the model's error distribution across all time nodes. For time nodes with minimal voltage and current variations, the number of neurons is reduced to free up computational resources. On the contrary, at time nodes with a large number of nonlinear features, the previously released neurons are allocated to improve the fitting accuracy.

As shown in Fig. 8, the proposed strategy uses a structured training process. Prior to training, several parameters are initialized, including the error scaling factor E_Gain, which is set to

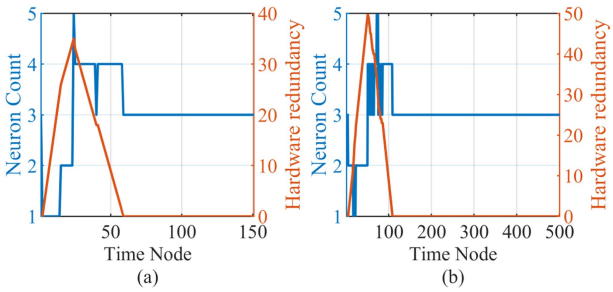

Fig. 9. Distribution of hidden layer neurons and hardware neuron redundancy. (a) Distribution of turn-ON. (b) Distribution of turn-OFF.

1.25 in this article. This parameter controls the neuron tuning decision, ensuring that neurons are reduced only when the resulting MSE is lower than a predefined threshold in multiples of *E*_Gain. In addition, a neuron redundancy buffer (Neural_least) was introduced for recording the number of redundant neurons throughout the training process. If the current time node Neural_least is zero, the neuron increase is limited. A predefined distribution matrix (**HiddenData**) was also created using an empirical method to initialize the distribution of the number of neurons at each time node. The training process iteratively evaluates each time step, first determining if the neurons can be reduced. If not, the system identifies high complexity nodes and attempts to increase the neuron allocation. However, all increases in neurons are predicated on the presence of redundant neurons at that time node.

The analysis in Fig. 9 shows that in the initial phase of the switching process, mainly low-complexity time nodes are involved, so the number of neurons can be reduced. Whereas in the intermediate phases, the number of neurons needs to be increased due to the high complexity time nodes involved. It is worth noting that the neural network architecture is implemented on FPGAs using a pipelined structure, so that some neurons are in a taskless state in time nodes where the number of required neurons is lower than the number of available neurons in the hardware. At the same time, the neural network model generates transient waveforms using steady-state inputs, which remain constant for a short period of time. This feature allows the taskless neurons to act as redundant neurons, precomputing tasks for subsequent time nodes and storing the results of the computations to be released when needed. The whole process is able to improve the overall accuracy of the model without adding additional hardware neuron computation units, while not affecting the original pipeline computation structure.

Fig. 10 illustrates the module structure and task scheduling flow of the dynamic neuron allocation strategy on FPGA. At the time node t_m, due to lower modeling complexity, only the first (*n-j*) neurons are activated for forward computation, leaving *j* neurons unused at this moment. These idle neurons are reassigned by the neuron distribution engine to perform pre-computation tasks for subsequent complex time nodes (e.g., t_n). The precomputed results, along with their associated target time node indices, are stored into BRAMs labeled as storage BRAMs in the figure. When the simulation reaches t_n, the distribution engine retrieves the corresponding pre-computed outputs

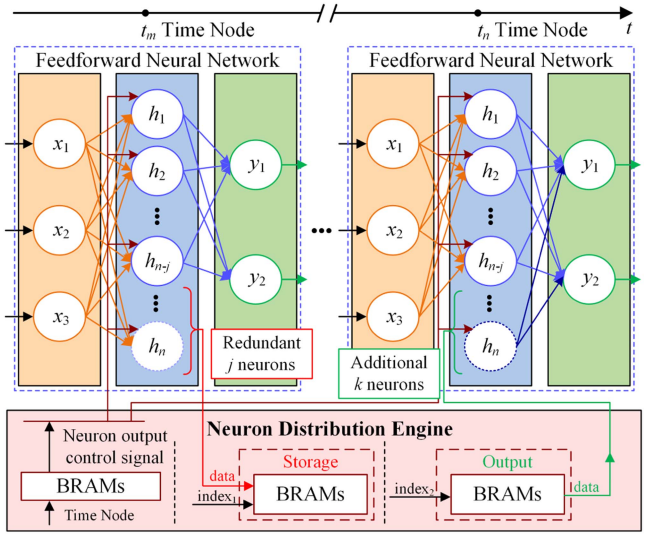

Fig. 10. Diagram of the proposed neuron dynamic adjustment strategy in FPGA.

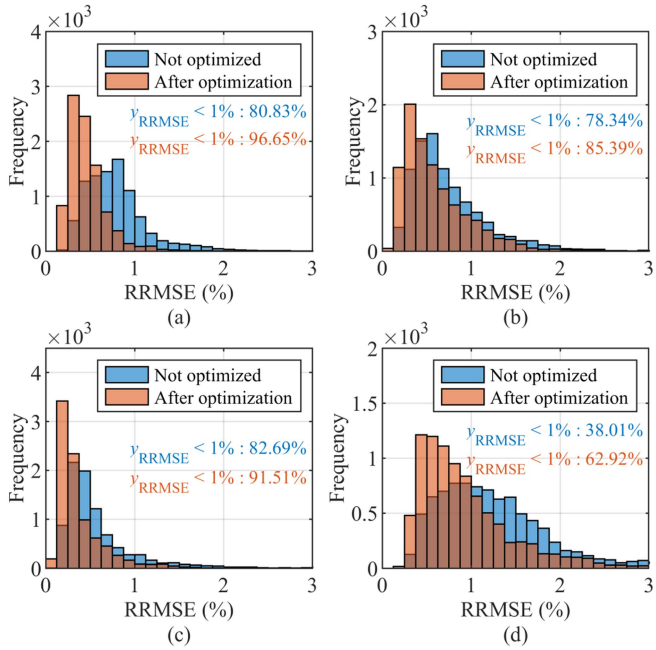

Fig. 11. Histogram of RRMSE statistics for the switch model under multiple operating conditions. (a) v_{ce} of turn-ON transient. (b) v_{ce} of turn-OFF transient. (c) i_c of turn-ON transient. (d) i_c of turn-ON transient.

from output BRAMs using index_2, and feeds the data into the neural network module as supplementary outputs, thereby avoiding real-time computation delay under complex transient conditions. Simultaneously, the distribution engine reads the neuron activation index (index_1) at each time node from BRAM to control which neurons are engaged in the forward pass. The entire mechanism is driven by preloaded index tables stored in BRAM, enabling dynamic computational resource scheduling on top of a static hardware structure, effectively balancing FPGA resource utilization and neural network modeling accuracy.

As shown in Fig. 11, the RRMSE distributions of the proposed 3-neuron connectionist neural network model before and after

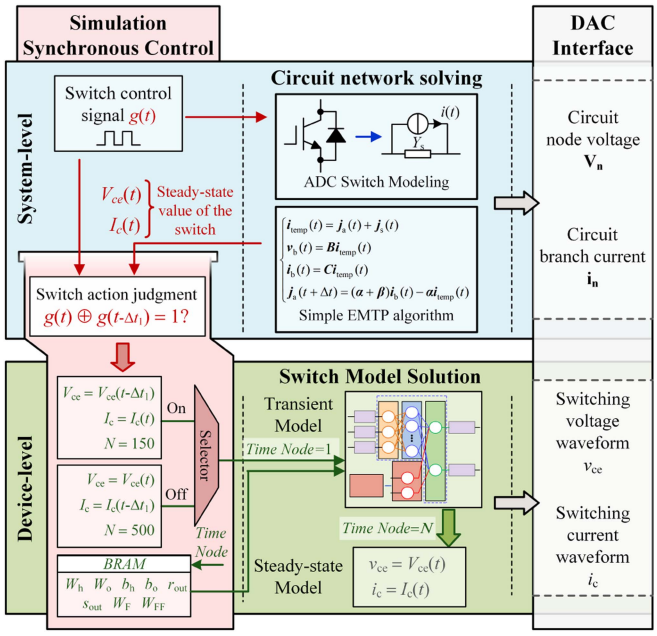

Fig. 12. Real-time simulation platform architecture.

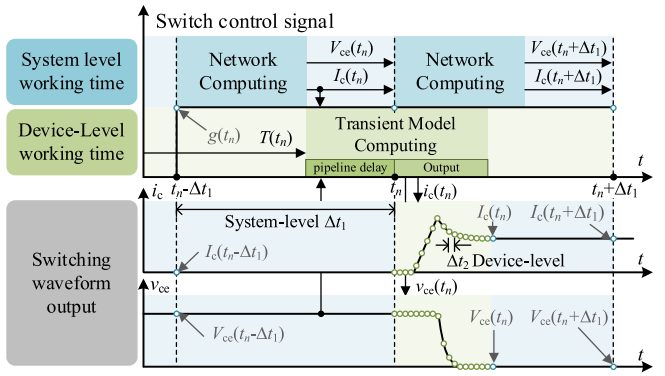

Fig. 13. Time-domain synchronization diagram between system-level and device-level simulations.

optimization are evaluated under 9225 operating conditions in this article. The most significant improvements can be seen in Fig. 11(a) and (d), where the percentage of samples with errors less than 1% significantly increases from 80.83% to 96.65% and from 38.01% to 62.92%, respectively, indicating a significant reduction in the incidence of high errors. Meanwhile, Fig. 11(b) and (c) shows slight improvements, indicating that although the optimization is still effective, its impact is less pronounced in these cases. These findings confirm that the proposed strategy effectively improves the model accuracy.

V. Integrated Framework and Applications of Real-Time Simulation

A. Real-Time Simulation Architecture

The real-time simulation architecture in this article is divided into two parallel parts: device-level simulation with nanosecond granularity and system-level simulation with microsecond time steps [29]. As shown in Fig. 12, the real-time simulation architecture consists of system-level simulation, device-level simulation, simulation synchronization control, and DAC interface, all of which run concurrently in the FPGA environment.

At the system level, the simulation utilizes large time steps in the microsecond range to update the circuit network matrix for solving based on the switching control signals. In contrast, device-level simulation utilizes steady-state data provided by system-level execution to generate transient waveforms with nanosecond resolution. When the time node escalates to N (the maximum time node for the current switching state), the device-level simulation completes its transient waveform output and transitions to providing the steady-state value as the current output.

The solution of the transient model is implemented by a neural network-based switching model consisting of 650 interconnected neural network units and a neuron allocation engine implemented in an FPGA using variable coefficients connectionist neural networks and memory modules with data selection. In addition to this, it is necessary to set the appropriate cycle delays for the data according to the pipeline position in which they are located to ensure that the signals are synchronized with the coefficient updates. To minimize the resource consumption and delay incurred by the FPGA to compute the nonlinear activation function. Given that computing a nonlinear activation function (e.g., tan$h$) in an FPGA consumes significant resources and incurs delays, the function is implemented using a lookup table approach [30]. It is important to note that in order to mitigate any inaccuracies introduced by the lookup table, the tan$h$ function is replaced with its lookup table during the neural network training phase.

The simulation synchronization control section determines the switching state by comparing the switching control signals $g(t)$ and $g(t\text{-}\Delta t_1)$. When a switching action is detected, the time node register is initialized and incremented according to the simulation time. The synchronization control module is also responsible for transferring the steady-state voltages and currents at the current simulation instant to the device-level simulation section based on the detected switching status. Since the switching model is implemented using a variable-coefficient connectionist neural network, its network weights are not fixed throughout the transient simulation process but are dynamically updated according to the time node. Consequently, during real-time simulation, the synchronization control module must retrieve the corresponding weight parameters from the FPGA's block RAM using the current time-node index. This ensures that the neural network utilizes the appropriate model coefficients at each step, thereby accurately generating the transient response of the switching device. To more clearly illustrate the timing coordination between system-level and device-level simulations, Fig. 13 presents a timing diagram outlining the simulation workflow. The system-level simulation proceeds with a time step of Δt_1. Once a switching event is detected and the system-level circuit solution is completed, the steady-state parameters are provided as input to activate the device-level

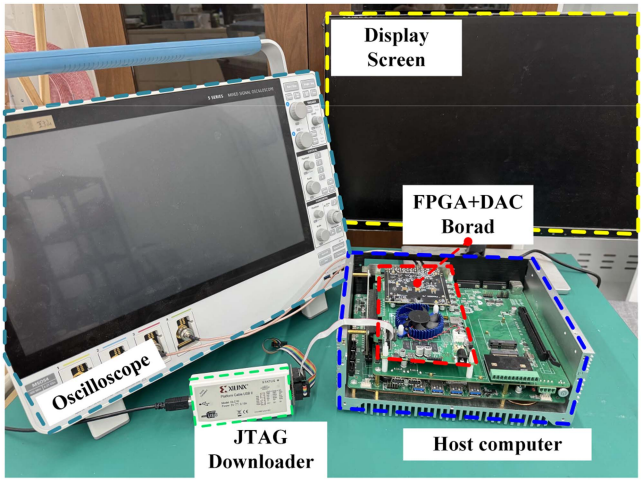

Fig. 14. Real-time simulation hardware platform.

model. The device-level simulation then generates high resolution transient waveforms with a time step of Δt_2. It should be noted that, due to the pipelined architecture of the neural network implemented on the FPGA, there is an inherent startup latency of 32 FPGA clock cycles before the first valid output is produced. This delay originates from the pipeline fill stage required to achieve high-throughput parallel computation. Finally, once the device-level model completes its output, the system switches back to using the results from the system-level simulation for subsequent processing.

As shown in Fig. 14, the real-time simulation hardware architecture consists of several key components, including a JTAG downloader for communication with the host and bitstream download, an FPGA board for real-time simulation model computation, a DAC board for converting the simulation results to output, an oscilloscope, and a host monitor. The main FPGA chip used is the Xilinx Kintex-7 XC7K325T, which includes 407 600 registers, 203 800 lookup tables, 840 DSP48s and 445 RAM blocks. The simulation model is designed and developed in the LabVIEW FPGA environment using the graphical programming language provided by NI LabVIEW. The main chip on the DAC board is the DAC3174 high-performance 14-bit dual-channel digital-to-analog converter (DAC) from Texas Instruments, which supports a conversion rate of up to 500 MSPS.

B. Bidirectional Buck-Boost Converter Real-Time Simulation

This article presents a real-time simulation example using a bidirectional buck-boost converter, which is widely used in energy storage systems and electric vehicles, and is an ideal case study for verifying the performance of the switching transient modeling approach. The topology of the bidirectional buck-boost converter is shown in Fig. 15. The main parameters are detailed in Table I. In this case, the system-level simulation was executed with a time step of 400 ns, while the device-level transient model operated with a fine resolution of 5 ns to accurately capture the switching dynamics.

In the real-time simulation of the bidirectional buck-boost converter, the oscilloscope captures the output waveform of

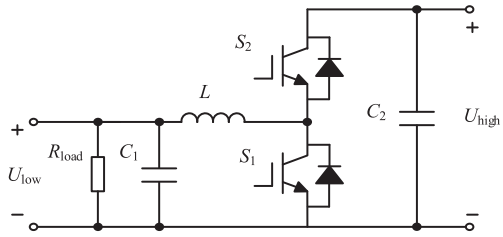

Fig. 15. Topology of bidirectional Buck-Boost converter.

TABLE I
MAIN PARAMETERS OF CONVERTER

Parameter	Value
Low voltage side capacitor C_1	220 nF
High voltage side capacitor C_2	100 μF
Inductance L	9.1 mH
High voltage side voltage U_{high}	600 V
Low voltage side load resistance R_{load}	2.64 Ω
Switching frequency f_{sw}	20 kHz
Switch S_2 control signal duty cycle D	44 %

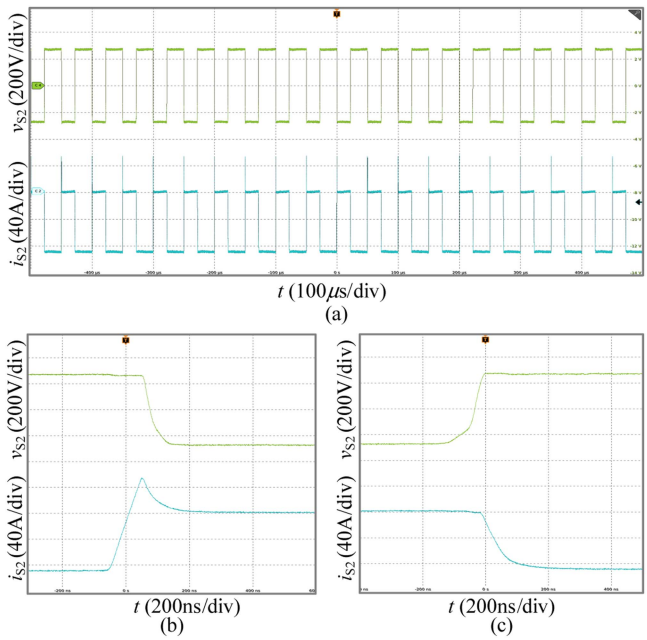

Fig. 16. Oscilloscope waveform of switch S_2 in real-time simulation. (a) Voltage and current waveforms of switch S_2. (b) Zoomed-in view of waveforms during the turn-ON of S_2. (c) Zoomed-in view of waveforms during the turn-OFF of S_2.

switch $S2$ as shown in Fig. 16. It is pertinent to note that the displayed waveform is the result of the DAC scaling the simulation outputs to a range of -5 to 5 V. To assess the performance of the modeling method in real-time simulation and further validate its effectiveness, this article compares the results of the real-time simulation of the FPGA with the results of the offline simulation of LTspice, and the results are shown in Fig. 17. The comparison results show that the simulation results of the proposed model

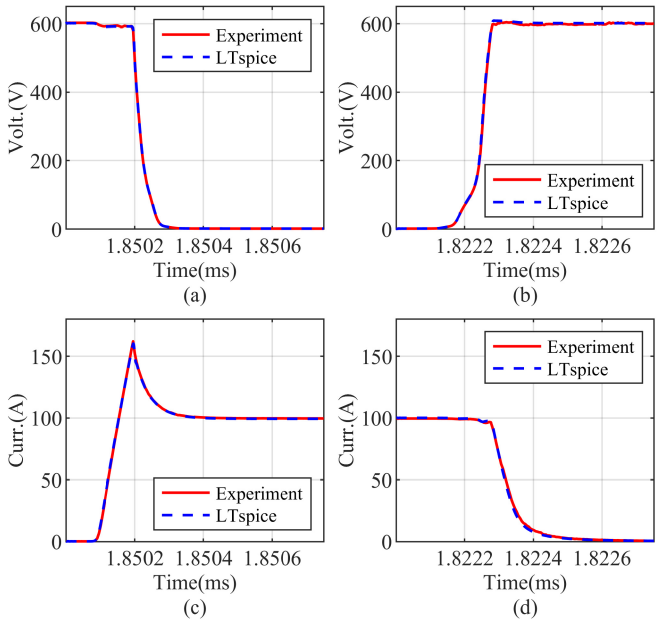

Fig. 17. Comparison of real-time simulation results. (a) v_{ce} of turn-ON transient. (b) v_{ce} of turn-OFF transient. (c) i_c of turn-ON transient. (d) i_c of turn-ON transient.

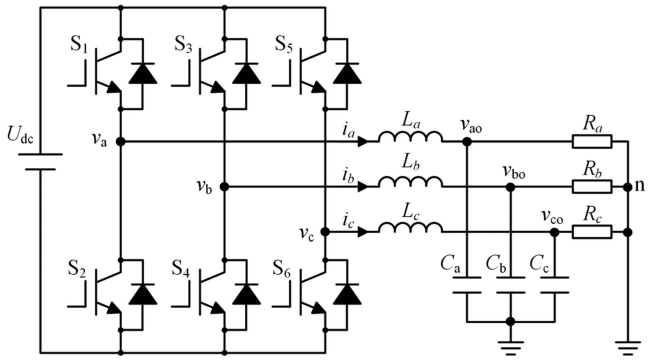

Fig. 18. Topology of three-phase inverter.

are in good agreement with those of the LTspice model, thus confirming the high fidelity of the switching transient model constructed by the connectionist neural network architecture.

C. Three-Phase Inverter Real-Time Simulation

Three-phase inverters are key components in power electronic systems and are widely used in photovoltaic power generation, motor drives and energy storage systems. Its main function is to convert dc power into three-phase ac power for supplying to loads or grid connection. Fig. 18 illustrates the topology of a three-phase inverter, which usually consists of multiple switching devices, inductors and capacitors. The main principle of operation is to adjust the relationship between the dc voltage and the output AC voltage by controlling the state of the switching devices. The high voltage on the dc side requires the use of high power electronic devices for switching. For this modeling example, an IGBT with a withstand voltage of 1200 V and an antiparallel diode (model FGY100T120RWD) is selected to meet the requirements of high-voltage working conditions, and controlled using sinusoidal pulsewidth modulation (SPWM). The main component parameters used in the three-phase inverter model are given in Table II.

TABLE II
MAIN PARAMETERS OF THREE-PHASE INVERTER

Parameter	Value
DC voltage U_{dc}	700 V
AC load inductance L_a, L_b, L_c	1.5 mH
AC load capacitance C_a, C_b, C_c	60 μF
AC load resistance R_a, R_b, R_c	7.26 Ω
Switching frequency f_{sw}	10 kHz

The three-phase inverter topology modeling uses FPGA as the core platform for system-level and device-level modeling of the circuit network. The host computer transmits the SPWM signals to the FPGA through the PCIe interface to realize the real-time control of the inverter. After receiving the signals, the FPGA updates the system-level switching model, calculates the steady-state simulation results of each node by using the electromagnetic transient simulation algorithm, and further generates the transient simulation results of the switching action through the switching transient model. The three-phase inverter CPU serves as a real-time simulation example of the circuit controller executing the SPWM waveform generation algorithm. The modulation frequency of the sine wave is 50 Hz, the frequency of the triangle wave is 10 kHz, the modulation ratio is 0.889, and the dead time is 800 ns. For the three-phase inverter case, a system-level time step of 1 μs was adopted and the device-level model was simulated at 5 ns intervals for transient accuracy.

The real-time simulation results are compared with the offline simulation results as shown in Fig. 19. Fig. 19(a) shows the waveforms of the output voltage (v_a) of the a-phase bridge arm, the output voltage (v_{ao}) of the load side, and the output current (i_a) obtained from the LTspice offline simulation. Fig. 19(b) shows the corresponding waveforms captured by the oscilloscope of the real-time simulation platform. The comparison results show that the real-time simulation results are very consistent with the offline simulation results, verifying the accuracy and effectiveness of the real-time simulation method.

Fig. 19(c) shows the output voltage and current waveforms of the a-phase upper bridge arm IGBT S_1 obtained by the real-time simulation platform. Due to the transient characteristics and detailed complexity of the IGBT process, direct comparison with an oscilloscope may not be accurate enough. Therefore, it is necessary to further transmit the transient data of the IGBT to the host computer for analysis through the PCIe interface. The comparison results of the turn-ON and turn-OFF processes of the switching process are given in Figs. 20 and 21 for real-time simulation and offline simulation, respectively. Fig. 20 shows that the transient voltage and current waveforms during the turn-ON process are highly consistent. On this basis, Fig. 21 provides a more detailed zoomed-in comparison of the switching

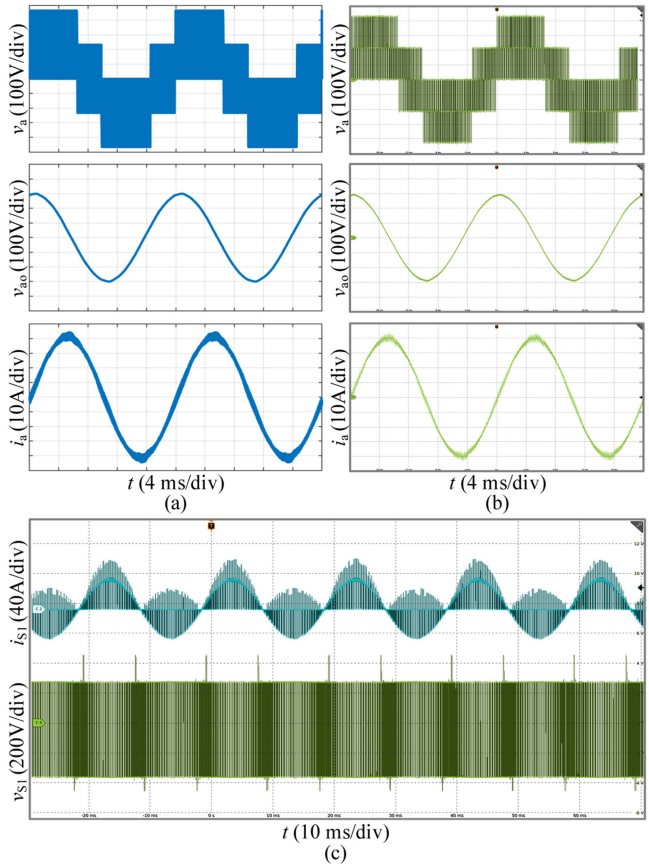

Fig. 19. Comparison of real-time simulation results of three-phase inverter. (a) System-level LTspice simulation results. (b) System-level real-time simulation oscilloscope capture results. (c) Phase upper bridge arm switch oscilloscope waveform.

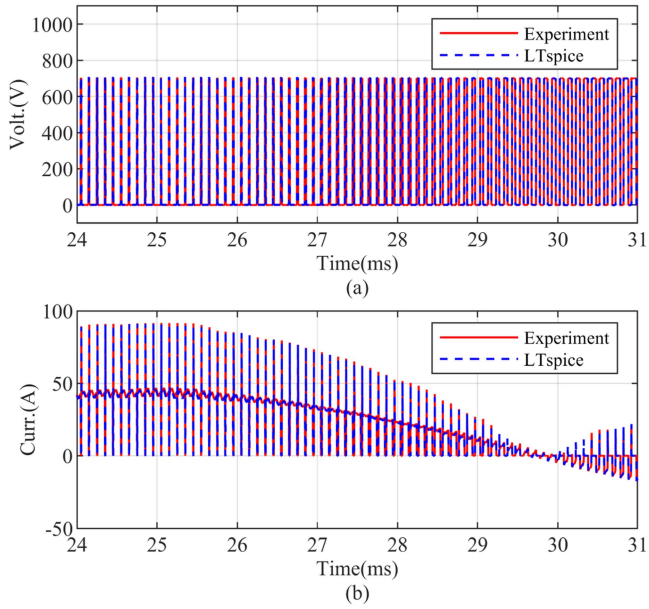

Fig. 20. Comparison of output voltage and current waveforms of S_1. (a) Output voltage v_{ce} waveform. (b) Output current i_c waveform.

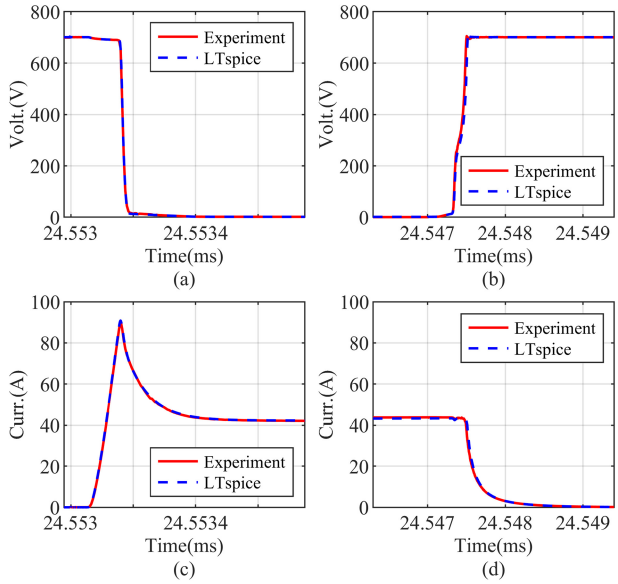

Fig. 21. Zoomed-in comparison of output voltage and current waveforms of S_1. (a) v_{ce} of turn-ON transient. (b) v_{ce} of turn-OFF transient. (c) i_c of turn-ON transient. (d) i_c of turn-ON transient.

TABLE III
DEVICE UTILIZATION ON XC7K325T FPGA

Device	Total	Used		Percentage of Total Resources	
		Case1	Case2	Case1	Case2
Slice Registers	407 600	25 419	46 173	6.2%	11.3%
Slice LUTs	203 800	26 989	50 787	13.2%	24.9%
Block RAMs	445	96	171	21.6%	38.4%
DSP48s	840	91	291	10.8%	34.6%

transient characteristics, showing that the voltage and current waveforms are almost identical during the turn-ON and turn-OFF phases. These detailed comparisons further confirm the accuracy advantage of the proposed model in transient simulation, and also demonstrate the versatility of the simulation example by incorporating another switching device, thus demonstrating the versatility of the modeling methodology proposed in this article.

D. FPGA Resource Utilization and Comparative Analysis

Table III gives the overall FPGA resource utilization for the two real-time simulation case studies, where case 1 corresponds to the Buck-Boost converter and case 2 to the three-phase inverter. The reported values include not only the implementation of the neural network-based switching models but also the supporting modules of the real-time simulation platform, such as the PCIe communication interface, DAC conversion logic, and waveform output control.

To further demonstrate the advantages of the proposed method, Table IV gives a comprehensive comparison between our approach and representative existing device-level modeling

TABLE IV
COMPARISON WITH EXISTING REAL-TIME DEVICE-LEVEL MODELING METHODS

	Method type	Time-step	DSP48s	RMS error
Wang et al. [19]	Equivalent Circuit	10 ns	132	<5%
Li et al.[26]	Data-driven	5 ns	128	<5%
Proposed Model	Data-driven	5 ns	47	<2%

techniques in terms of FPGA resource utilization and modeling accuracy. Compared with [19] and [26], the connectionist neural network proposed in this article achieves comparable or even higher accuracy under switching transients. Specifically, the proposed method maintains high fidelity with an RMS error below 2% across most operating conditions, which indicates excellent alignment with physics-based reference models. At the same time, due to the network's suitability for switching transient modeling and the use of a dynamic neuron allocation strategy, it significantly reduces DSP48s consumption.

In addition to resource and accuracy considerations, the applicability of the proposed model to various topologies and system simulation frameworks is also emphasized. Unlike the method in [19], which tightly couples device-level and system-level solvers the modeling frameworks in [26] and in this article adopt a decoupled architecture. This design effectively removes the constraint of system-level simulation step size on the transient model, thereby improving compatibility with diverse converter topologies and system-level solvers. As a result, the proposed model exhibits enhanced generality and scalability in practical real-time simulation scenarios, making it more suitable for large-scale and heterogeneous system applications.

VI. CONCLUSION

In this article, a connectionist neural network-based switching transient modeling method and a dynamic neuron allocation strategy are proposed for real-time simulation of power electronic converters, which significantly improves the simulation accuracy and real-time performance. Compared with the existing methods, the proposed method not only achieves higher simulation accuracy but also significantly reduces the demand for FPGA hardware resources. It is particularly suitable for accurate modeling of switching devices that do not exhibit complex parasitic oscillations during transient processes. By accurately capturing and restoring the voltage and current transient waveforms during the switching action, the proposed model is able to effectively evaluate the key performance indicators such as switching loss, thermal characteristics, and EMI caused by high-speed switching in a real-time simulation environment. Future research will explore the integration of IGBT thermal modeling and the EMI noise caused by high switching speed, aiming to further improve the applicability of real-time simulation.

REFERENCES

[1] X. Liang, "Emerging power quality challenges due to integration of renewable energy sources," *IEEE Trans. Ind. Appl.*, vol. 53, no. 2, pp. 855–866, Mar./Apr. 2017.

[2] A. Benigni, T. Strasser, G. De Carne, M. Liserre, M. Cupelli, and A. Monti, "Real-time simulation-based testing of modern energy systems: A review and discussion," *IEEE Ind. Electron. Mag.*, vol. 14, no. 2, pp. 28–39, Jun. 2020.

[3] H. Bai, C. Liu, S. Zhuo, R. Ma, D. Paire, and F. Gao, "FPGA-based device-level electro-thermal modeling of floating interleaved boost converter for fuel cell hardware-in-the-loop applications," *IEEE Trans. Ind. Appl.*, vol. 55, no. 5, pp. 5300–5310, Sep./Oct. 2019.

[4] M. S. Vekić, S. U. Grabić, D. P. Majstorović, I. L. Čelanović, N. L. Čelanović, and V. A. Katić, "Ultralow latency HIL platform for rapid development of complex power electronics systems," *IEEE Trans. Power Electron.*, vol. 27, no. 11, pp. 4436–4444, Nov. 2012.

[5] K. Wang, J. Xu, G. Li, N. Tai, A. Tong, and J. Hou, "A generalized associated discrete circuit model of power converters in real-time simulation," *IEEE Trans. Power Electron.*, vol. 34, no. 3, pp. 2220–2233, Mar. 2019.

[6] J. Zhang, T. Lu, W. Zhang, X. Bian, and X. Cui, "Characteristics and influence factors of radiated disturbance induced by IGBT switching," *IEEE Trans. Power Electron.*, vol. 34, no. 12, pp. 11833–11842, Dec. 2019.

[7] A. R. Hefner and D. M. Diebolt, "An experimentally verified IGBT model implemented in the Saber circuit simulator," *IEEE Trans. Power Electron.*, vol. 9, no. 5, pp. 532–542, Sep. 1994.

[8] R. Kraus and H. J. Mattausch, "Status and trends of power semiconductor device models for circuit simulation," *IEEE Trans. Power Electron.*, vol. 13, no. 3, pp. 452–465, May 1998.

[9] K. Sheng, S. J. Finney, and B. W. Williams, "A new analytical IGBT model with improved electrical characteristics," *IEEE Trans. Power Electron.*, vol. 14, no. 1, pp. 98–107, Jan. 1999.

[10] B. Shi, Z. Zhao, and Y. Zhu, "Piecewise analytical transient model for power switching device commutation unit," *IEEE Trans. Power Electron.*, vol. 34, no. 6, pp. 5720–5736, Jun. 2019.

[11] H. Bai, C. Liu, E. Breaz, K. Al-Haddad, and F. Gao, "A review on the device-level real-time simulation of power electronic converters: Motivations for improving performance," *IEEE Ind. Electron. Mag.*, vol. 15, no. 1, pp. 12–27, Mar. 2021.

[12] N. Lin and V. Dinavahi, "Detailed device-level electrothermal modeling of the Proactive hybrid HVDC breaker for real-time hardware-in-the-loop simulation of DC grids," *IEEE Trans. Power Electron.*, vol. 33, no. 2, pp. 1118–1134, Feb. 2018.

[13] Z. Shen and V. Dinavahi, "Dynamic variable time-stepping schemes for real-time FPGA-based nonlinear electromagnetic transient emulation," *IEEE Trans. Ind. Electron.*, vol. 64, no. 5, pp. 4006–4016, May 2017.

[14] H. Bai, H. Luo, C. Liu, D. Paire, and F. Gao, "A device-level transient modeling approach for the FPGA-based real-time simulation of power converters," *IEEE Trans. Power Electron.*, vol. 35, no. 2, pp. 1282–1292, Feb. 2020.

[15] L. Idkhajine and E. Monmasson, "Embedded fully FPGA-based real-time simulators for static power converters with power switch characteristics approximated by identification," *IEEE Trans. Ind. Electron.*, vol. 69, no. 9, pp. 9624–9633, Sep. 2022.

[16] N. Lin and V. Dinavahi, "Behavioral device-level modeling of modular multilevel converters in real time for variable-speed drive applications," *IEEE J. Emerg. Sel. Topics Power Electron.*, vol. 5, no. 3, pp. 1177–1191, Sep. 2017.

[17] N. Lin and V. Dinavahi, "Dynamic Electro-magnetic-Thermal modeling of MMC-based DC–DC converter for real-time simulation of MTDC grid," *IEEE Trans. Power Del.*, vol. 33, no. 3, pp. 1337–1347, Jun. 2018.

[18] Z. Huang and V. Dinavahi, "A fast and stable method for modeling generalized nonlinearities in power electronic circuit simulation and its real-time implementation," *IEEE Trans. Power Electron.*, vol. 34, no. 4, pp. 3124–3138, Apr. 2019.

[19] S. Wang, X. Guo, K. Li, Y. Yin, Z. Sun, and X. You, "An equivalent switching model for FPGA-based real-time simulation of SiC MOSFET transient behaviors in power electronic converters," *IEEE Trans. Power Electron.*, vol. 40, no. 9, pp. 13063–13074, Sep. 2025.

[20] S. Subedi, Y. Gui, and Y. Xue, "Applications of data-driven dynamic modeling of power converters in power systems: An overview," *IEEE Trans. Ind. Appl.*, vol. 61, no. 2, pp. 2434–2456, Mar./Apr. 2025.

[21] H. S. Krishnamoorthy and T. Narayanan Aayer, "Machine learning based modeling of power electronic converters," in *Proc. 2019 IEEE Energy Convers. Congr. Expo.*, 2019, pp. 666–672.

[22] T. Liang, Z. Huang, and V. Dinavahi, "Adaptive real-time hybrid neural network-based device-level modeling for DC traction HIL application," *IEEE Access*, vol. 8, pp. 69543–69556, 2020.

[23] N. Hari, S. Chatterjee, and A. Iyer, "Gallium nitride power device modeling using deep feed forward neural networks," in *Proc. 1st Workshop Wide Bandgap Power Devices Appl. Asia*, 2018, pp. 164–168.

[24] B. Shang, T. Cheng, T. Liang, N. Lin, and V. Dinavahi, "Real-time nonlinear behavioral electrothermal device-level emulation of IGBT on heterogeneous adaptive compute acceleration platform," *IEEE Open J. Ind. Electron. Soc.*, vol. 3, pp. 663–673, 2022.

[25] S. Zhang, T. Liang, and V. Dinavahi, "Real-time HIL emulation of DRM with machine learning accelerated WBG device models," *IEEE Open J. Power Electron.*, vol. 4, pp. 567–578, 2023.

[26] Q. Li, H. Bai, E. Breaz, R. Roche, and F. Gao, "ANN-aided data-driven IGBT switching transient modeling approach for FPGA-based real-time simulation of power converters," *IEEE Trans. Transp. Elect.*, vol. 9, no. 1, pp. 1166–1177, Mar. 2023.

[27] C. Wang, Q. Wang, H. Weng, and X. Pan, "A modified algorithm for the L/C-based switch model of power converters in real-time simulation based on FPGA," *IEEE Trans. Ind. Appl.*, vol. 60, no. 5, pp. 7030–7037, Sep./Oct. 2024.

[28] Technical Documentation. Physically Based, Scalable SPICE Modeling Methodolo-gies for Modern Power Electronic Devices. Accessed: Aug. 2021. [Online]. Available: https://www.onsemi.com/video/physically-based-scalablespice-modeling-methodologies-for-modern-power-electronic-devices

[29] H. Bai, C. Liu, A. K. Rathore, D. Paire, and F. Gao, "An FPGA-based IGBT behavioral model with high transient resolution for real-time simulation of power electronic circuits," *IEEE Trans. Ind. Electron.*, vol. 66, no. 8, pp. 6581–6591, Aug. 2019.

[30] B. Zamanlooy and M. Mirhassani, "Efficient VLSI implementation of neural networks with hyperbolic tangent activation function," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 22, no. 1, pp. 39–48, Jan. 2014.