Energy 338 (2025) 138753

Contents lists available at ScienceDirect

Energy

journal homepage: www.elsevier.com/locate/energy

AI-accelerated physics-informed transient real-time digital-twin of SMR-based multi-domain submarine power distribution

Songyang Zhang *, Weiran Chen, Yuzhong Zhang, Venkata Dinavahi

Department of Electrical and Computer Engineering, University of Alberta, Edmonton, T6G 2V4, Alberta, Canada

ARTICLE INFO

Keywords:
Artificial intelligence
Digital-twin
Field-programmable gate arrays
Machine learning
Multi-domain system
Physics-informed neural networks
Real-time systems
Small modular reactors

ABSTRACT

Small Modular Reactors (SMRs) have emerged as promising solutions for next-generation marine propulsion systems due to their enhanced efficiency, reduced maintenance requirements, and extended operational capabilities. However, traditional transient modeling methods for these systems often rely on conventional numerical integration techniques, which encounter significant challenges when dealing with nonlinear system dynamics, leading to considerable computational latency and extensive parameter tuning efforts. To address these limitations, this paper introduces an artificial intelligence (AI)-accelerated physics-informed real-time digital-twin (RTDT) for an SMR-based multi-domain submarine power distribution system. The proposed approach integrates physics-informed machine learning (PIML) methodologies, combining neural network models with explicit physical constraints. Leveraging the parallel computing capabilities of the Xilinx® UltraScale+ FPGA hardware platform, the proposed framework significantly reduces computational latency. The emulation results validate the effectiveness and efficiency of the proposed PIML-based RTDT, achieving mean percentage absolute errors (MPAEs) consistently below 1%, thus demonstrating superior performance compared to classical numerical methods.

1. Introduction

Small Modular Reactors (SMRs) have garnered increasing attention for their potential to revolutionize next-generation power systems by offering enhanced efficiency, modular construction, and improved safety characteristics. SMRs operate at lower capacities – typically ranging from 20 to 300 MWe – making them suitable for flexible deployment in remote or distributed energy systems [1–3]. According to the International Atomic Energy Agency (IAEA), SMRs can run continuously for up to 30 years without refueling, thereby significantly reducing operational costs, maintenance downtime, and refueling logistics [4,5]. These characteristics, combined with simplified passive safety features and reduced construction times, position SMRs as viable alternatives to large-scale nuclear reactors [6–8]. Recent research has explored the feasibility of SMRs in diverse energy domains. For instance, integration with cogeneration systems [9], ammonia-fueled subsystems [10], and multi-unit renewable-friendly architectures [11] illustrates the adaptability of SMRs. Novel control strategies have been proposed to handle uncertainties in sensor feedback and actuator reliability [12], while fault diagnosis under varying operating conditions has been explored using transfer learning and deep learning methods [13]. These efforts underscore the growing complexity of SMR systems and the need for intelligent modeling and control solutions.

SMR-based submarines offer substantial advantages over conventional diesel-electric platforms [14,15], enabling prolonged underwater endurance and reduced refueling frequency—capabilities that have driven nations such as the United States to invest over $69 billion beyond initial projections in next-generation strategic submarine technologies [16]. For such high-stakes and non-conventional applications, comprehensive pre-deployment simulation and analysis of the overall system and its subsystems are essential. In particular, real-time digital-twin (RTDT) emulation facilitates rapid and seamless integration testing with existing controllers, algorithms, and systems [17]. Traditional numerical modeling approaches are limited in this context. First, offline computer-based simulations lack the ability to interface with real-world controllers and systems in real time. Second, they rely heavily on accurate parameters and become computationally intensive when dealing with highly nonlinear subsystems. As such, RTDTs have emerged as a powerful solution [18–20]. RTDTs, when deployed on FPGA platforms, offer parallelized, low-latency emulation capabilities that support hardware-in-the-loop integration with real systems [17]. Although previous studies have explored the use of field-programmable gate arrays (FPGAs) for real-time emulation of SMR-based submarines or nuclear-powered systems [16], most still rely on conventional numerical analysis methods. While such approaches are well-established

* Corresponding author.
E-mail address: zhang.songyang@ualberta.ca (S. Zhang).

https://doi.org/10.1016/j.energy.2025.138753
Received 6 April 2025; Received in revised form 27 September 2025; Accepted 30 September 2025
Available online 4 October 2025

and validated, they require extensive model tuning and significant computational resources when dealing with nonlinear system behaviors. In contrast, recent advancements in neural network-based modeling offer unique advantages, especially in handling nonlinear dynamics due to their inherent compatibility with parallel hardware acceleration [21, 22].

Earlier studies on neural network applications in nonlinear power systems modeling were predominantly data-driven [21–23]. More recently, hybrid approaches have emerged that integrate domain knowledge with machine learning models, combining the strengths of data-driven and physics-based methods. The rise of physics-informed neural networks [24–26] has introduced a novel paradigm, where models not only learn from empirical nonlinear datasets but also incorporate physical constraints to enhance interpretability. The success of PINN-based modeling has been demonstrated across a wide range of systems, including state-of-health estimation for lithium-ion batteries [27–29], modeling of borehole thermal energy storage [30], and wind forecasting [31]. These studies have shown that PINNs can significantly reduce data requirements while preserving physical realism.

Despite these advances, most existing applications of PINNs focus on isolated subsystems or simplified models [32,33]. Comprehensive real-time modeling for multi-domain systems – particularly those involving nuclear, mechanical, and electrical couplings – remains largely unexplored. For instance, although promising results have been reported by combining PINNs with other neural architectures, such as temporal convolutional networks for wind speed and power forecasting [34], long short-term memories (LSTM) for photovoltaic temperature calibration and power prediction [35], and multiple multilayer perceptions for lithium-ion battery evaluation [36], these studies did not account for inference hardware deployment, model complexity, or computation latency. Additionally, most energy system PINN applications are performed offline without real-time integration into hardware platforms. To address these gaps, this paper proposes an AI-accelerated, physics-informed machine learning (PIML) framework for RTDT modeling of an SMR-based multi-domain submarine power distribution system. The proposed method is deployed on a Xilinx® UltraScale+ FPGA platform to enable ultra-fast, high-fidelity emulation. This work demonstrates the potential of combining RTDT and PIML to meet the stringent performance and safety demands of modern naval nuclear platforms. The key contributions of this work are as follows:

(1) A unified PIML-based modeling framework is proposed for SMR-based multi-domain submarine power systems. By embedding physical constraints into neural network structures (PIFNN and PIRNN), the framework achieves interpretable, high-fidelity nonlinear modeling suitable for FPGA-based RTDT emulation.

(2) A comprehensive implementation of neural network parallelization strategies is performed on FPGA hardware. Fixed-point models with loop unrolling demonstrate optimal trade-offs between resource usage and performance, achieving up to 60% latency reduction compared to conventional serial forward Euler (FE) and Trapezoidal rule (TZR) methods.

(3) Quantitative emulation results across thermal, pressure, turbine, and electrical domains confirm that the proposed PIML models achieve high numerical accuracy, with mean percentage absolute error (MPAE) consistently below 1% when benchmarked against reference EMT simulations.

This paper presents an AI-accelerated, physics-informed real-time transient digital-twin of an SMR-based multi-domain submarine power distribution system, implemented on a Xilinx® UltraScale+ FPGA hardware platform. The remainder of this paper is organized as follows: Section 2 reviews conventional modeling methodologies. Section 3 introduces machine learning (ML)-based modeling approaches. Section 4 details the modeling framework for multi-domain systems. Section 5 describes the implementation of the proposed PIML-based models and the corresponding hardware platform. Section 6 presents the real-time digital-twin results for the SMR-based multi-domain submarine system. Finally, Section 7 concludes the paper.

2. Conventional numerical modeling methods

Accurate numerical modeling methods are essential to discretize the differential governing equations for the dynamic analysis of power systems, particularly in circuit simulation, control design, and stability assessment. Various well-established numerical techniques have been developed, including explicit and implicit integration schemes, iterative solvers, and multi-stage numerical approaches. This section provides an overview of commonly used modeling methods along with their mathematical formulations.

2.1. Forward Euler method

The FE method is a first-order explicit numerical integration approach that approximates the evolution of a differential equation using a first-order Taylor series expansion. Given a differential equation of the form:

$$\frac{dx}{dt} = f(x,t), \tag{1}$$

the numerical update at the next time-step is computed as:

$$x_{k+1} = x_k + hf(x_k,t_k), \tag{2}$$

where h is the time-step size. Despite its computational simplicity, the FE method suffers from stability issues, particularly when applied to stiff systems.

2.2. Fourth-order Runge–Kutta (RK4) method

The fourth-order Runge–Kutta (RK4) method is a widely adopted explicit numerical integration scheme that improves both accuracy and stability compared to the FE method. It involves four intermediate stages, defined as:

$$k_1 = hf(x_k,t_k), \tag{3}$$

$$k_2 = hf\left(x_k + \frac{k_1}{2}, t_k + \frac{h}{2}\right), \tag{4}$$

$$k_3 = hf\left(x_k + \frac{k_2}{2}, t_k + \frac{h}{2}\right), \tag{5}$$

$$k_4 = hf(x_k + k_3, t_k + h). \tag{6}$$

The final state update is then computed as:

$$x_{k+1} = x_k + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4). \tag{7}$$

The RK4 method offers a favorable trade-off between computational efficiency and numerical accuracy, making it well-suited for a wide range of power electronic simulations.

2.3. Implicit Runge–Kutta (IRK) method

The implicit Runge–Kutta (IRK) method is a class of implicit numerical integration techniques particularly well-suited for solving stiff differential equations commonly encountered in power electronic circuits. The general IRK formulation is expressed as:

$$X_i = x_k + h\sum_{j=1}^{s} a_{ij} f(X_j, t_k + c_i h), \tag{8}$$

where a_{ij} and c_i are coefficients defined by the Butcher tableau. The final update follows:

$$x_{k+1} = x_k + h\sum_{i=1}^{s} b_i f(X_i, t_k + c_i h). \tag{9}$$

IRK methods, such as the Gauss–Legendre scheme, exhibit superior numerical accuracy and unconditional stability. However, they require

solving nonlinear systems of equations at each time-step, increasing computational complexity.

In numerical simulations, the TZR is one of the most commonly used methods due to its balance between accuracy, computational efficiency, and stability. The TZR method can be regarded as a second-order IRK method, given by:

$$x_{k+1} = x_k + \frac{h}{2}\left[f(x_k, t_k) + f(x_{k+1}, t_{k+1})\right]. \tag{10}$$

As an implicit integration method, the TZR ensures A-stability and is particularly effective for stiff system simulations.

Conversely, implicit solution processes are inherently more complex, and for nonlinear problems, the Newton–Raphson method is often the only viable approach for achieving numerical convergence.

3. Machine learning methods

Conventional time-stepping methods, such as FE method, RK4, and other numerical discretization techniques – commonly used in electromagnetic transients (EMT) simulations – employ discrete point-to-point calculations, iteratively solving for system states at each time-step. These EMT methods are widely adopted due to their accuracy and efficiency in dynamic system analysis. In contrast to traditional physics-based computational methods that rely on analytical formulations, an alternative approach involves data-driven solutions leveraging neural networks for nonlinear problem-solving. This methodology, primarily composed of matrix operations and activation functions, offers significant advantages in terms of parallel acceleration on hardware platforms compared to conventional computational techniques. However, conventional data-driven ML models often result in "black-box" solutions with limited interpretability. To address this, physics-informed feedforward neural network (PIFNN) integrate fundamental physical principles with ML techniques, combining the advantages of traditional physics-based modeling and data-driven learning for enhanced accuracy and reliability.

3.1. Physics-informed neural network

A traditional PINN approximates the solution function $u(x,t)$ directly over a continuous domain. In contrast, the PIFNN and PIRNN models are designed to approximate the instantaneous rate of change (the right-hand side of the differential equations), i.e., $\frac{du}{dt}$. This architectural choice allows the models to integrate seamlessly with a time-stepping approach, making them highly suitable for dynamic, real-time emulation. In particular, the recurrent structure of the PIRNN draws a direct parallel with established numerical integration methods like the TZR. Both these methods and traditional PINNs integrate physical constraints into the training loss function. However, this approach goes a step further by tailoring the network architecture itself to the nature of the dynamic problem. The PIRNN's design, which inherently processes information sequentially, is more effective at capturing the time-dependent constraints of dynamic systems compared to a non-recurrent PINN. This integration of a specialized architecture with physics-informed learning is a key innovation that enables the model to achieve high accuracy and low latency in a RTDT framework.

3.1.1. Proposed PIFNN-based vs. Traditional time-stepping approaches

Fig. 1(a) illustrates a comparison between traditional EMT-based time-stepping numerical integration and PIFNN-based time-stepping methods. The blue dashed line represents the traditional time-stepping approach, where system states are computed sequentially at each discrete time-step Δt. In contrast, the green dashed line represents the PIFNN-based approach, which leverages a trained NN to predict future system states in a data-driven manner. The PIFNN method learns from previous states to approximate future states, reducing reliance on explicit numerical solvers while maintaining accuracy. The PIFNN-based approach, which functions as a time-stepping method, is particularly suitable for simulations involving multivariable dynamic variations.

3.1.2. Continuous-time PIFNN approximation

The PIFNN framework can be also structured to achieve functionalities beyond conventional time-stepping methods. Fig. 1(b) introduces an alternative approach, the pre-solving continuous-time PIFNN (PSCT-PIFNN), which approximates an analytical function $y_t = G(t)$ that closely fits the field-measured reference curve. In this framework, the PSCT-PIFNN effectively incorporates the solution process during the NN training phase. Once trained, the PSCT-PIFNN can directly compute the corresponding physical quantities by simply inputting the time variable, eliminating the need for iterative numerical integration. Unlike time-stepping approaches, PSCT-PIFNN leverages both initial and boundary conditions, along with sampled internal points, to construct a continuous-time representation of the system dynamics. This method removes the dependency on discrete iterations, offering a smooth, physics-informed approximation that efficiently captures system behavior. Compared to point-to-point time-stepping methods, which are more suited for dynamic systems with external input variations, the PSCT-PIFNN approach is particularly advantageous for systems governed solely by initial conditions with no external excitations after initialization.

3.2. Physics-informed recurrent neural network

The physics-informed recurrent neural network (PIRNN) is a data-driven approach designed to integrate domain knowledge with deep learning techniques. As illustrated in Fig. 2, PIRNN exhibits a computational structure that closely resembles traditional numerical methods, particularly the EMT methods.

3.2.1. Two-sequence-step PIRNN and trapezoidal rule-based processes

- **Traditional TZR-Based Computation:** The EMT function is applied at both the current state $X(t)$ and the next-step estimated state $X(t+\Delta t)$. The two estimated increments ΔX_t and $\Delta X_{t+\Delta t}$ are averaged to obtain the final state update:

$$X(t+\Delta t) = X(t) + \frac{\Delta X_t + \Delta X_{t+\Delta t}}{2}. \tag{11}$$

  This iterative approach ensures numerical stability while approximating the system dynamics.
- **PIRNN computational process:** The RNN cells serve as function approximators analogous to the EMT function in the TZR. The state update follows a similar averaging process, where the next-step prediction is computed as:

$$X(t+\Delta t) = X(t) + \frac{\Delta X_t^{NN} + \Delta X_{t+\Delta t}^{NN}}{2}. \tag{12}$$

  The physics-informed process ensures that the NN maintains consistency with the underlying physical laws, similar to how the TZR enforces numerical stability in traditional models.

3.2.2. Pirnn's equivalence to traditional methods

- **Structural parity:** PIRNN's sequential architecture and recursive updates mirror the iterative approach used in traditional EMT solvers. By using RNN cells in place of conventional EMT functions, PIRNN effectively reconstructs the time-stepping mechanism in a data-driven yet physics-consistent manner.
- **Physics-informed intermediate data:** Just as the TZR incorporates numerical corrections from physics-based calculations, PIRNN embeds physics-informed constraints into its update equations, ensuring that the network's predictions remain physically meaningful.
- **Flexible consistent:** Unlike purely data-driven RNNs, PIRNN maintains the core numerical principles of traditional methods, allowing it to generalize across different system conditions while still producing results consistent with numerical solvers.

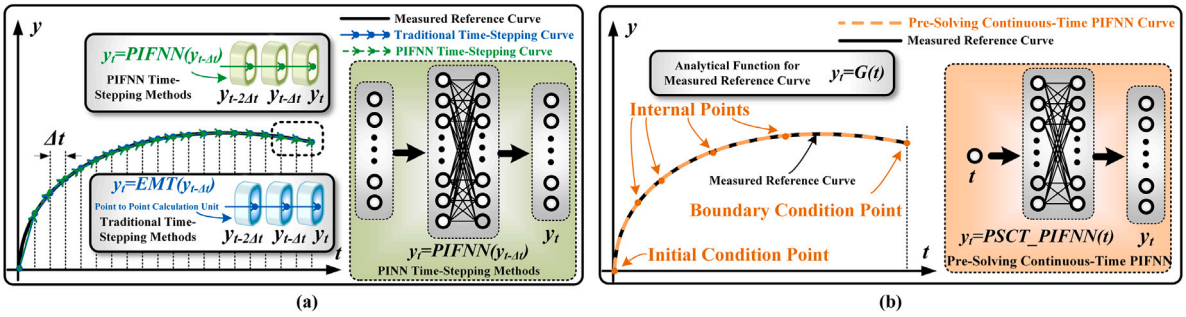

Fig. 1. Comparison of traditional and PIFNN-based methods: (a) PIFNN-based time-stepping vs. traditional time-stepping approaches; (b) pre-Solving continuous-time PIFNN approximation.

Table 1
Comparison of modeling methods.

Feature	FE	RK4	IRK	FNN in [21]	RNN in [21]	PFNN in [22]	PIFNN	PIRNN
Complexity	+	++	++++	+	+++	++	++	+++
Execution Time	++	+++	++++	++	+++	++	++	+++
Resource Consumption	+	++	+++	++	+++	++	++	+++
Accuracy	+	++	+++	++	++	+++	+++	+++
Generality	+++	+++	+++	++	++	++	+++	+++
Stability	+	++	+++	+	+	++	+++	+++
Scalability	NO	NO	NO	Yes	Yes	Yes	Yes	Yes
Hardware Acceleration Feasibility	+	+	+	+++++	++++	+++++	+++++	++++
Data Dependency	No	No	No	++++	++++	++	+	+
Training Efficiency	–	–	–	++	++	+++	++++	++++
Constraints of Physical Laws	Yes	Yes	Yes	No	No	No	Yes	Yes

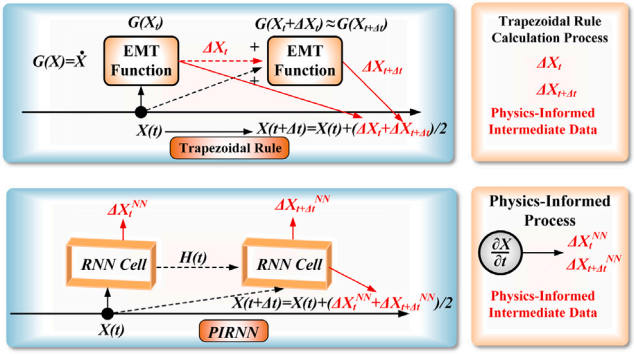

Fig. 2. Trapezoidal rule method vs. PIRNN.

3.3. Comparison of modeling methods

The utilized models in this work, PIFNN and PIRNN, provide significant advantages over numerical methods (FE, RK4, IRK) and existing neural approaches (FNN, RNN, physics-feature neural network (PFNN)), as shown in Table 1. Specifically, compared to IRK (most complex), both proposed methods feature lower complexity, leading to reduced execution time and less resource consumption. This enables more efficient hardware implementation compared to traditional numerical approaches (FE, RK4, IRK). In terms of accuracy, PIFNN and PIRNN achieve superior performance, clearly outperforming simpler numerical methods like FE and RK4, and matching the high precision of IRK and PFNN. They also offer enhanced generality, superior to pure neural methods such as FNN and RNN. Stability of the proposed methods matches the best available (IRK) and exceeds traditional numerical integration methods (FE, RK4). Unlike numerical methods (FE, RK4, IRK), both PIFNN and PIRNN possess scalability. Additionally, the proposed ML models effectively incorporate constraints of physical laws, an important advantage missing in standard neural approaches (FNN, RNN, PFNN). Finally, the lower data dependency and improved training efficiency of PIFNN and PIRNN distinguish them significantly from data-intensive methods like FNN. To clarify the context for the evaluation, the results in Table 1 are based on models at the subsystem and equipment levels, including DC load networks, converters, and motors. These models were conducted with a time-step of 100 ns to 1 ms, which are typical scales for RTDT applications. This ensures the comparison of methods is both meaningful and relevant.

Constraint Handling: Both these methods and traditional PINNs integrate physical constraints into the training loss function. However, this approach goes a step further by tailoring the network architecture itself to the nature of the dynamic problem. The PIRNN's design, which inherently processes information sequentially, is more effective at capturing the time-dependent constraints of dynamic systems compared to a non-recurrent PINN. This integration of a specialized architecture with physics-informed learning is a key innovation that enables the model to achieve high accuracy and low latency in a RTDT framework.

4. Multi-domain submarine power system modeling

This section presents a comprehensive modeling framework for multi-domain systems. In this study, an SMR-based multi-domain submarine power distribution system is selected as the case study to validate the proposed modeling approach. The overall structure of the SMR-based multi-domain submarine system is depicted in Fig. 3, while its abstract conceptual topology is illustrated in Fig. 4. The nuclear generation section [37–39] integrates neutron kinetics, thermal-hydraulic dynamics, and turbine-generator control dynamics to form a holistic representation of the nuclear propulsion system. The multi-domain modeling approach decomposes the system into multiple interacting subsystems, each governed by a set of mathematical equations capturing transient behaviors. For the electrical power conversion and distribution section, the permanent magnet synchronous motor (PMSM) Similarly in is chosen as the focal case study for multi-domain analysis. PMSM-based propulsion offers high efficiency and dynamic performance, making it well-suited for submarine applications.

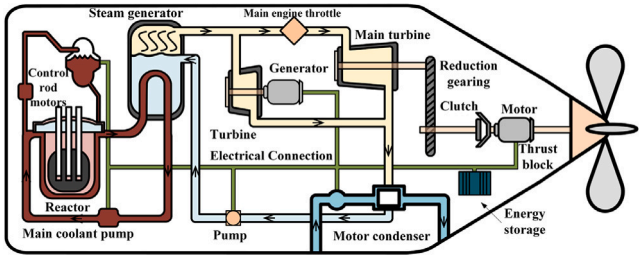

Fig. 3. Overview of SMR-based multi-domain submarine system.

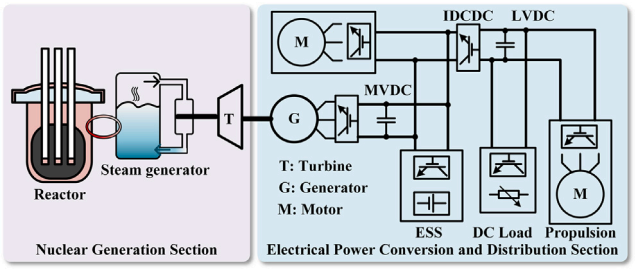

Fig. 4. Abstract conceptual nuclear submarine power distribution.

4.1. Neutron kinetics

The neutron kinetics model captures the time-dependent behavior of neutron flux and delayed neutron precursor concentrations within the reactor core, providing insights into power regulation and reactivity feedback.

The point kinetics equations governing neutron population dynamics, including neutron flux deviation and delayed neutron precursor concentrations, are given by:

$$\frac{d\Delta n(t)}{dt} = \frac{\Delta\rho(t)}{l} - \frac{\beta}{l}\Delta n(t) + \sum_{i=1}^{6}\lambda_i \Delta C_i(t), \tag{13}$$

$$\frac{d\Delta C_i(t)}{dt} = \frac{\beta_i}{l}\Delta n(t) - \lambda_i \Delta C_i(t), \tag{14}$$

where $\Delta n(t)$ is the neutron density deviation; $\Delta C_i(t)$ represents the concentration of delayed neutron precursors for the ith group; $\Delta\rho(t)$ is the reactivity deviation, influenced by external and feedback effects; β is the total delayed neutron fraction, representing the proportion of neutrons emitted as delayed neutrons; β_i is the fraction of delayed neutrons in the ith precursor group. λ_i is the decay constant of the delayed neutron precursor for the ith group; l is the neutron generation time, defining the characteristic time scale for prompt neutron population changes.

The reactivity deviation $\Delta\rho(t)$, incorporating control action and temperature feedback effects, is formulated as:

$$\Delta\rho(t) = \rho_{\text{ext}} + \alpha_F \Delta T_f + \alpha_C \left(\frac{\Delta T_{C1} + \Delta T_{C2}}{2}\right), \tag{15}$$

where ρ_{ext} is the externally induced reactivity change, e.g., from control rod motion; α_F and α_C are the reactivity feedback coefficients for fuel and coolant temperature changes, respectively; ΔT_f represents the deviation in fuel temperature; ΔT_{C1} and ΔT_{C2} denote the temperature deviations at different coolant nodes. These functions approximate the integral effects of reactivity on neutron flux evolution and help in the efficient numerical computation of neutron kinetics.

4.2. Coolant and thermal hydraulics

This subsection describes the thermal-hydraulic interactions between fuel and coolant, as well as coolant circulation.

4.2.1. Fuel and coolant temperature dynamics

This subsystem describes the thermal energy exchange between fuel and coolant, affecting reactivity and reactor stability. The temperature dynamics are governed by:

$$\frac{dT_f}{dt} = \frac{f_F P_{TH} n}{m_F C_p^F} - \frac{Ah(T_f - T_{C1})}{m_F C_p^F}, \tag{16}$$

$$\frac{dT_{C1}}{dt} = \frac{(1-f_F)P_{TH} n}{m_C C_p^C} + \frac{Ah(T_f - T_{C1})}{m_C C_p^C} - \frac{2W_c(T_{C1} - T_{cin})}{m_C}, \tag{17}$$

$$\frac{dT_{C2}}{dt} = \frac{(1-f_F)P_{TH} n}{m_C C_p^C} + \frac{Ah(T_f - T_{C1})}{m_C C_p^C} - \frac{2W_c(T_{C2} - T_{C1})}{m_C}. \tag{18}$$

where T_{C1} and T_{C2} are coolant temperatures at two different nodes; P_{TH} is the thermal power, which is proportional to the neutron flux n; f_F is the fraction of power transferred to the fuel; m_F and m_C are the masses of fuel and coolant, respectively; C_p^F and C_p^C are the specific heat capacities of fuel and coolant; A is the heat transfer area; h is the heat transfer coefficient; W_c is the coolant flow rate; T_{cin} is the inlet coolant temperature.

4.2.2. Hot leg and inlet temperature evolution

The model captures coolant circulation between the reactor core and steam generator, ensuring efficient heat transfer and maintaining operational stability. The governing equations for temperature changes in the hot leg and inlet coolant are:

$$\frac{dT_{HL}}{dt} = \frac{T_{C2} - T_{HL}}{t_H}, \tag{19}$$

$$\frac{dT_{cin}}{dt} = \frac{2T_p - T_{cin} - T_{HL}}{t_C}. \tag{20}$$

where T_{HL} is the hot leg temperature, representing the temperature of the coolant leaving the reactor core; t_H is the characteristic time constant for hot leg temperature evolution; T_p is the pressurizer temperature, which helps regulate system pressure and affects coolant inlet conditions; t_C is the characteristic time constant governing the inlet temperature changes.

4.3. Pressure and temperature dynamics

This subsystem captures pressure and thermal feedback regulation. The governing equations for pressure and temperature variations are:

$$\frac{dT_p}{dt} = \frac{k_{pm}T_m + k_{pc}T_{HL} - T_p}{t_p}, \tag{21}$$

$$\frac{dT_m}{dt} = \frac{k_{mp}T_p + k_{ms}P_s - T_m}{t_m}, \tag{22}$$

$$\frac{dP_s}{dt} = \frac{k_{psm}T_m + k_{psy}(1-y) - P_s}{t_{ps}}. \tag{23}$$

where T_m is the coolant mixing temperature, representing an intermediate state in the thermal regulation process; P_s is the system pressure, influencing boiling point regulation and reactor safety margins; k_{pm}, k_{pc}, k_{mp}, and k_{ms} are proportionality constants governing heat and pressure feedback interactions; k_{psm} and k_{psy} regulate pressure dynamics based on coolant temperature and control actions; t_p, t_m, and t_{ps} are characteristic time constants associated with pressurizer, coolant mixing, and pressure response dynamics.

4.4. Turbine dynamics

The turbine model describes the dynamic conversion of steam thermal energy into mechanical power for electricity generation. The model comprises high-pressure and intermediate-low pressure turbine stages, regulated by a steam valve and governor control system to maintain

stable power output under varying operational conditions. The general mathematical formulation of the turbine dynamics is represented by the following equations:

$$\frac{dP_{HP}}{dt} = \frac{F_{HP}\mu(P_{s,ref} - P_s) - P_{HP}}{T_{CH}}, \tag{24}$$

$$\frac{dP_{LP}}{dt} = P_{RH}, \tag{25}$$

$$\frac{dP_{RH}}{dt} = \frac{F_{LP}\mu(P_{s,ref} - P_s) - P_{LP} - (T_{CH} + T_{RH})P_{RH}}{T_{CH}T_{RH}}, \tag{26}$$

where: P_{HP} is the high-pressure turbine state variable representing the integral power output; P_{LP} is the intermediate-low pressure turbine state variable representing the integral power output; P_{RH} is an auxiliary state variable representing reheater dynamics; μ is the valve opening fraction controlling steam admission; P_s is the actual steam pressure affecting the turbine efficiency; $P_{s,ref}$ is the reference steam pressure for optimal turbine operation; F_{HP} and F_{LP} are power distribution coefficients for turbine stages; T_{CH} and T_{RH} are time constants defining dynamic responses of the turbine stages. These equations characterize the relationship between valve positioning, pressure deviations, and power regulation, capturing transient behaviors in the steam flow and mechanical output of the turbine system.

4.5. Control of the SMR

This subsystem regulates power generation and turbine operations, ensuring stable reactor performance under varying load conditions. The control system dynamically adjusts turbine operation based on deviations from the reference power output. The governing equations representing the control dynamics are expressed as follows:

$$\frac{dg}{dt} = K_I\left(P_{ref} - (P_{HP} + P_{LP})\right), \tag{27}$$

$$\mu = \mathrm{sat}(y_g), \tag{28}$$

$$\frac{dy_g}{dt} = G_2\left(G_1\left(g + K_P\left(P_{ref} - (P_{HP} + P_{LP})\right) - \mu\right) - y_g\right), \tag{29}$$

where: P_{ref} is the reference power set-point for turbine control; K_I is the integral gain parameter for correcting long-term deviations in turbine power output; K_P is the proportional gain parameter for immediate response based on instantaneous power deviations; G_1 and G_2 are feedback loop gain parameters, regulating the governor response; y_g is the governor intermediate control variable; μ is the valve opening, constrained by the saturation function (sat) to limit valve positioning within operational bounds.

This control scheme effectively adjusts turbine outputs to match operational demands, maintaining reactor stability and safety under varying load conditions.

4.6. PMSM driving subsystem

For the electrical domain, the PMSM driving subsystem is selected as a representative study case, as illustrated in Fig. 4. Given its role in electromechanical energy conversion, an accurate electromagnetic representation is essential to capture its dynamic behavior precisely. The model employed in this study is formulated based on detailed electromagnetic modeling implemented in PSCAD/EMTDC®, incorporating short-circuited windings to account for internal leakage reactance, thereby enhancing the accuracy of dynamic simulations. The governing equations describing the electrical dynamics of the PMSM are given as:

$$\begin{cases} v_d = Ri_d - \omega_r\lambda_q + \dfrac{d\lambda_d}{dt}, & \lambda_d = L_d i_d + L_{md} i'_{kd} + \lambda_m, \\ v_q = Ri_q + \omega_r\lambda_d + \dfrac{d\lambda_q}{dt}, & \lambda_q = L_q i_q + L_{mq} i'_{kq}, \\ 0 = r'_k i'_{kd} + \dfrac{d\lambda'_{kd}}{dt}, & \lambda'_{kd} = L_{md} i_d + L'_{kd} i'_{kd} + \lambda_m, \\ 0 = r'_k i'_{kq} + \dfrac{d\lambda'_{kq}}{dt}, & \lambda'_{kq} = L_{mq} i_q + L'_{kq} i'_{kq} \end{cases} \tag{30}$$

$$T_e = \frac{3p}{4}\left[\lambda_{pm} i_q + \left(L_d - L_q\right) i_d i_q\right], \tag{31}$$

$$J\frac{d\omega_r}{dt} = T_e - T_{load} - B\omega_r, \tag{32}$$

$$\frac{d\theta_m}{dt} = \omega_r. \tag{33}$$

where: v_d, v_q are the d-axis and q-axis voltage components; i_d, i_q are the d-axis and q-axis current components; ω_r is the rotor angular velocity; λ_d, λ_q are the d-axis and q-axis flux linkages; R is the stator winding resistance; r'_k represents the equivalent resistance for the short-circuit condition; i'_{kd}, i'_{kq} are the short-circuit current components in the d and q axes; λ'_{kd}, λ'_{kq} are the short-circuit flux linkages; L_d, L_q are the stator inductances along the d and q axes; L_{md}, L_{mq} are the mutual inductances affecting the d and q axes; L'_{kd}, L'_{kq} are the leakage inductances associated with the short-circuited winding; λ_m is the permanent magnet flux linkage; T_e is the electromagnetic torque; T_{load} is the mechanical load torque applied to the rotor; ω_r is the rotor angular velocity; J is the rotor moment of inertia; B is the damping coefficient; θ_m is the rotor mechanical angle.

5. FPGA-based hardware real-time digital-twin implementation

This section presents the hardware implementation of RTDT for the SMR-based submarine power system, with a focus on comparing computational algorithms for SMR modeling.

5.1. Algorithm implementation

Fig. 5 illustrates the implementation of conventional mathematical modeling algorithms, comprising two primary components: linear state-space computations and nonlinear function evaluations, as shown in Fig. 5(a). The elements of the state-space matrix A, calculated based on the physical models described in Section 4, are presented in Table 2. The integration of these computational blocks within the TZR numerical integration framework is demonstrated in Fig. 5(b).

Fig. 6 depicts the implementation of the SMR physical process using PIFNN and PIRNN. Based on the physical processes described in Section 4, each subsystem is modeled independently using either a PIFNN or a PIRNN. This modular design reduces the computational matrix size of each neural network and enables efficient parallel deployment on FPGA hardware for real-time acceleration. Both network types are well-suited for parallel computation, offering significant speed-up when mapped onto reconfigurable logic. The choice between PIFNN and PIRNN architectures for different subsystems was a deliberate design decision based on their respective modeling requirements. The first three processes – fuel coolant temperatures, hot leg and inlet temperatures, and pressure and temperature dynamics – exhibit dynamics that can be effectively modeled at each time step without strong sequential dependencies. For these, the PIFNN architecture was selected as it provides an excellent balance between accuracy and computational efficiency, thereby helping to conserve hardware resources and minimize latency. In contrast, the turbine dynamics model is a highly nonlinear system with inherent time-dependent dynamics. The recurrent nature of PIRNN is much better suited to capture this complex, sequential behavior, ensuring higher modeling fidelity for this critical component. This targeted approach allows us to optimize each model for its specific

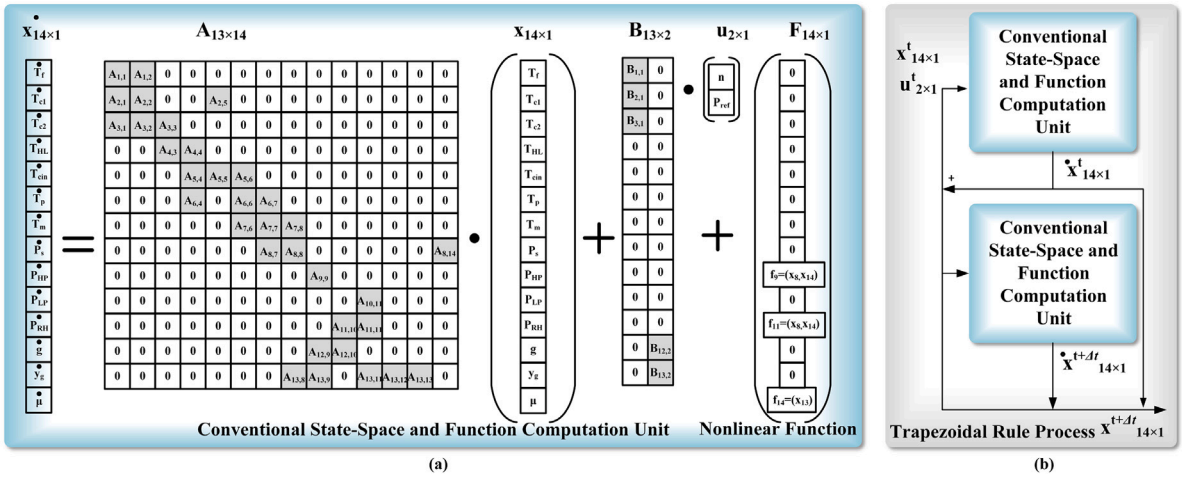

Fig. 5. Traditional mathematical model algorithm implementation: (a) the state-space and function computation; (b) the Trapezoidal rule process.

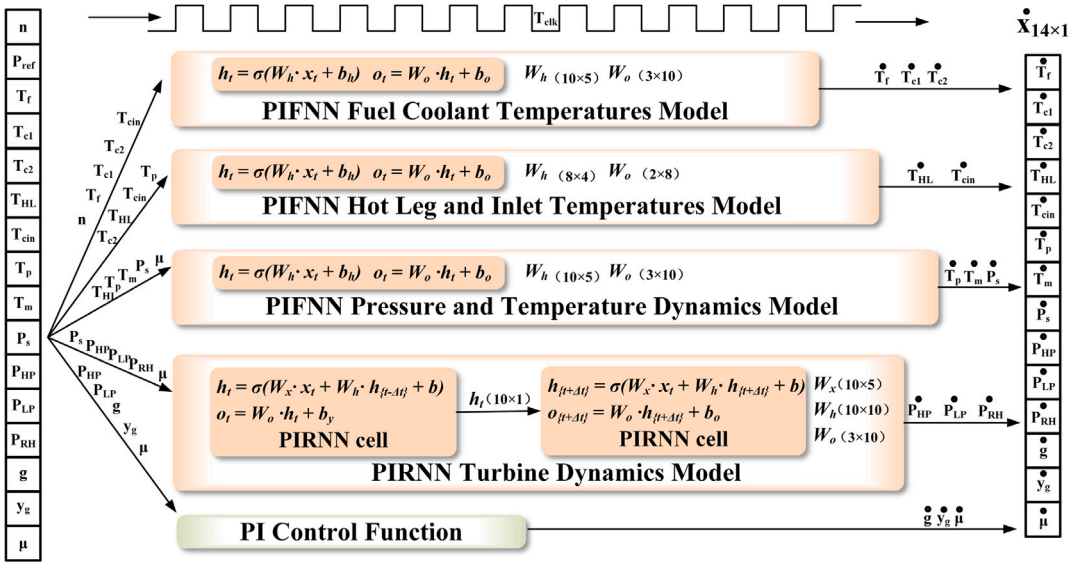

Fig. 6. Physics-informed algorithm implementation.

Table 2
Element-wise description of matrix **A**.

Element	Description	Element	Description
$A_{1,1}$	$-\frac{Ah}{m_F C_{pf}}$	$A_{1,2}$	$\frac{Ah}{m_F C_{pf}}$
$A_{2,1}$	$\frac{Ah}{m_C C_{pc}}$	$A_{2,2}$	$-\left(\frac{Ah}{m_C C_{pc}}+\frac{2W_{c,N}}{m_C}\right)$
$A_{3,1}$	$\frac{Ah}{m_C C_{pc}}$	$A_{3,2}$	$-\frac{Ah}{m_C C_{pc}}+\frac{2W_{c,N}}{m_C}$
$A_{3,3}$	$-\frac{2W_{c,N}}{m_C}$	$A_{4,4}$	$\frac{1}{\tau_H}$
$A_{4,5}$	$-\frac{1}{\tau_H}$	$A_{5,4}$	$-\frac{1}{\tau_C}$
$A_{5,5}$	$-\frac{1}{\tau_C}$	$A_{5,6}$	$\frac{2}{\tau_C}$
$A_{6,4}$	$\frac{k_{pc}}{\tau_p}$	$A_{6,6}$	$-\frac{1}{\tau_p}$
$A_{6,7}$	$\frac{k_{pm}}{\tau_p}$	$A_{7,6}$	$\frac{k_{mp}}{\tau_m}$
$A_{7,7}$	$-\frac{1}{\tau_m}$	$A_{7,8}$	$\frac{k_{ms}}{\tau_m}$
$A_{8,7}$	$\frac{k_{psm}}{\tau_{ps}}$	$A_{8,8}$	$-\frac{1}{\tau_{ps}}$
$A_{9,9}$	$-\frac{1}{T_{CH}}$	$A_{10,10}$	1
$A_{11,9}$	$-\frac{1}{T_{CH}T_{RH}}$	$A_{11,10}$	$-\frac{T_{CH}+T_{RH}}{T_{CH}T_{RH}}$
$A_{12,12}$	$-K_I$	$A_{13,12}$	$G_2 \cdot G_1$

task while maintaining the overall real-time performance of the digital twin.

Table 3 summarizes the computational complexity of commonly used neural network operations, which forms the theoretical foundation for parallel algorithm acceleration. For example, in the case of element-wise addition between two vectors, the serial computational complexity is $\mathcal{O}(n)$, requiring n clock cycles. However, with n parallel processing units, the operation can be completed in a single cycle. Similarly, although both vector addition and vector multiplication exhibit $\mathcal{O}(n)$ complexity under serial execution, their complexity can be reduced to $\mathcal{O}(\log_2 n)$ through an optimized tree-structured parallel architecture. This parallelism characteristic is a key factor in matrix-level acceleration. It is worth noting that the analysis above represents the ideal case, excluding the overhead caused by memory access, register usage, and data transfer. In practice, there is a trade-off between resource utilization and computational speed. Nevertheless, neural networks offer significant structural advantages for parallel implementation compared to traditional state-space or nonlinear function evaluations. The regular, repeating structure of neural network layers makes them easier to optimize and deploy efficiently on hardware platforms. Moreover, once optimized, a neural network can be flexibly reused across different modeling targets. For instance, the same trained network structure can

Table 3
Performance metrics comparison for vector/matrix operations.

Metric	Operation Type				
	Load	Vector Mult. or Add.	Elements Add.	Activation Func.	Matrix Mult.
Operation	Mem. read	$\mathbf{x}^T\mathbf{y}$ or $\mathbf{x}+\mathbf{y}$	$\sum_{i=1}^{n} x_i$	*Relu*	$\mathbf{A}_{i\times n}\mathbf{B}_{n\times h}$
Variable Dimensions	n	n	n	n	$i\times n\times h$
Time Complexity	CPU: $O(n)$	CPU: $O(n)$	CPU: $O(n)$	CPU: $O(n)$	CPU: $O(inh)$
Serial Clock Cycle	n	n	n	n	$ih(2n-1)$
Parallel Operation	FPGA: n	FPGA: n	FPGA: $\lfloor \frac{n}{2} \rfloor$	FPGA: n	FPGA: p and $\lfloor \frac{n}{2} \rfloor$
Parallel Complexity	FPGA: $O(1)$	FPGA: $O(1)$	FPGA: $O(\log n)$	FPGA: $O(1)$	FPGA: $O(inh/p)$
Parallel Clock Cycle	1	1	$\lceil \log_2 n \rceil$	1	$inh/p+\lceil \log_2 n \rceil$

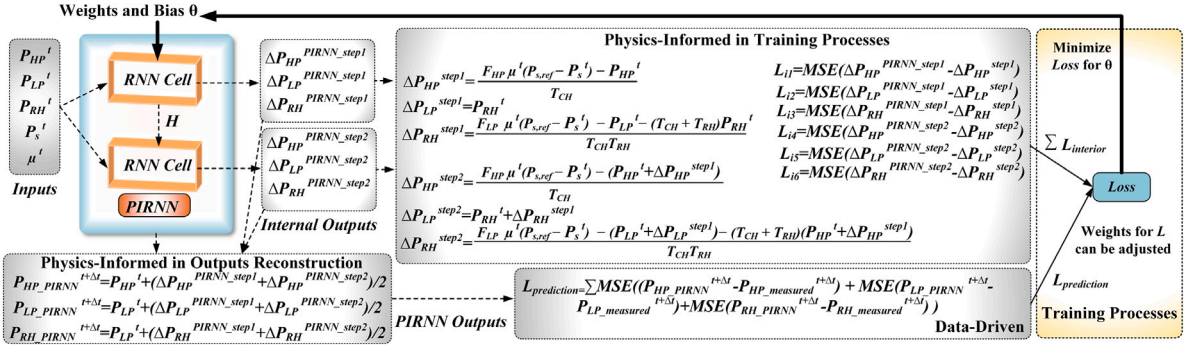

Fig. 7. PIRNN training process for turbine dynamics.

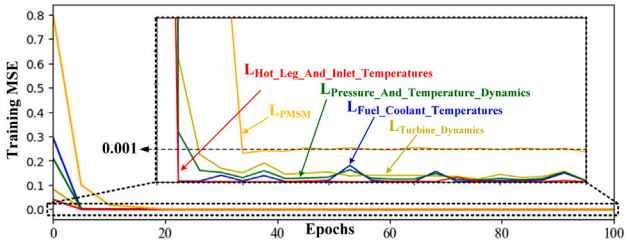

Fig. 8. MSE of models during training processes.

be adapted to calculate fuel coolant temperatures, estimate hot leg and inlet temperatures, or model pressure and temperature dynamics. This reusability enhances both the generalization and scalability of the modeling framework, particularly in multi-domain system simulations.

5.2. Training process and parameter design

To embed physical constraints into the PIFNN and PIRNN models, a hybrid strategy of soft and hard constraints is adopted in this work.

Soft constraints: Physical laws are incorporated into the training loss function as penalty terms. As illustrated in Fig. 7, intermediate variables from the recurrent cells are employed to compute the residuals of the turbine dynamics equations. The mean squared error (MSE) of these residuals defines the physics-informed loss term (L_{interior}), which is minimized together with the conventional data-driven loss ($L_{\text{prediction}}$). This mechanism guides the network to capture physically consistent representations without causing strict conflicts.

Hard constraints: To guarantee physical consistency in the final predictions, a deterministic reconstruction step is introduced. In this stage, the intermediate network outputs are combined with the predicted differentials through explicit physical formulations, thereby ensuring that the reconstructed variables (e.g., P_{HP}, P_{LP}, P_{RH}) strictly adhere to the governing equations. Since this reconstruction is a direct calculation rather than an optimization procedure, potential conflicts among physical constraints are effectively avoided. This two-level strategy enables the PIRNN to integrate physical laws both during training and at the output reconstruction stage, providing robust and interpretable predictions.

The detailed configurations of the PIFNN and PIRNN are as follows. For the SMR PIFNN, hidden layer depths ranging from 1–3 layers and 5–20 neurons were tested, following previous ML modeling practices [20–22]. The final architecture employs a single hidden layer. Specifically, the PIFNN fuel coolant temperatures model and the PIFNN pressure and temperature dynamics model use 10 hidden neurons, while the PIFNN hot leg and inlet temperatures model uses 8 hidden neurons (approximately twice the input dimension). For PIRNN, motivated by the TRZ method with two iterative cycles, the recurrent sequence length is set to 2, with 10 hidden neurons (also twice the input dimension). Detailed matrix operations and dimensional information are provided in Fig. 6. The activation function is chosen as ReLU due to its hardware efficiency: it requires only two cycles to implement with simple if–else logic, in contrast to Tanh and Sigmoid, which typically require LUT-based approximations occupying BRAM resources.

The training dataset consists of two parts: (i) randomly generated samples within the input domain, and (ii) measurement data (represented by simulation data in this study). For the random dataset, 20,000–50,000 uniformly generated input samples are employed, with the network outputs trained to match the corresponding physical model responses. For the measurement/simulation data, the raw dataset ranges from 0.5–1.0 million samples. After applying a 20:1 downsampling, 25,000–50,000 samples remain, of which 70% are used for training and 30% for testing and evaluation. Data preprocessing includes Z-score normalization, min–max normalization (where appropriate), and the removal of a small number of extreme outliers. To improve robustness, small Gaussian perturbations are injected into the training targets of the randomly generated dataset. This enables the network to learn to suppress such disturbances, thereby improving generalization against measurement noise.

The networks are trained using the Adam optimizer [40] with an initial learning rate of 0.001. The resulting training MSE reaches approximately 10^{-4}, while the test MPAE remains below 0.1%, taking PIRNN as an example (Fig. 8). In the training process of PINNs, loss functions are employed to quantify prediction accuracy, and those

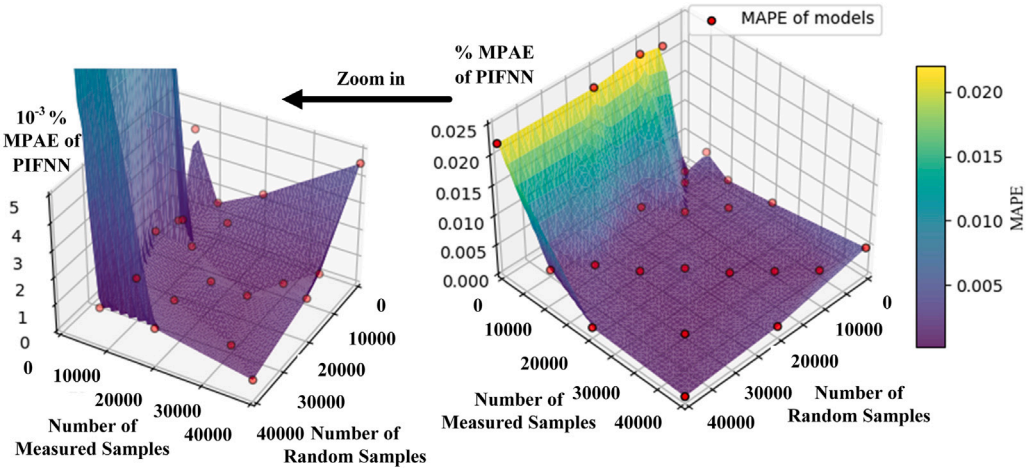

Fig. 9. MPAE of PIFNN models with different numbers of training samples.

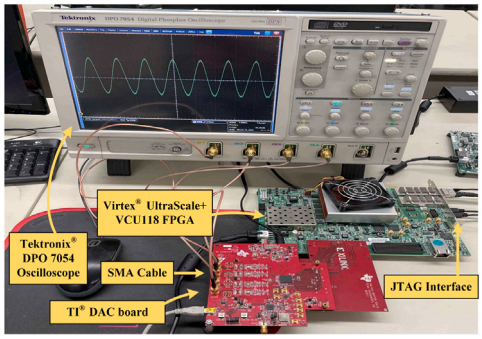

Fig. 10. Hardware setup of the RTDT for SMR-based submarine power distribution system.

adopted in this work are:

$$\mathrm{MPAE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100, \tag{34}$$

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2, \tag{35}$$

$$\mathrm{PAE}_i = \left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100. \tag{36}$$

Fig. 9 illustrates the modeling accuracy achieved with varying numbers of training samples, highlighting two key observations. Firstly, including measured data in the training set significantly improves model performance, resulting in a substantially lower MPAE. Secondly, increasing the number of training samples can slightly enhance the prediction accuracy, indicating diminishing returns with larger datasets. Notably, the PIML framework benefits from the inclusion of measured datasets, which inherently increases the effective number of training samples. This approach mitigates issues associated with limited measurement data, thereby improving modeling robustness and generalization capabilities in scenarios with constrained data acquisition.

5.3. Hardware emulation

Fig. 10 depicts the hardware setup utilized for implementing the SMR-based submarine system on the Xilinx VCU118 FPGA development board. Optimized neural network models derived from the training stage are deployed onto the FPGA platform for real-time emulation. Specifically, the VCU118 board employs the XCVU9P FPGA device, operating at a clock frequency of 100 MHz, corresponding to a computational cycle duration of 10 ns per clock cycle. The primary hardware resources on this FPGA include 4,320 K block random-access memory units (BRAMs), 6,840 digital signal processors (DSPs), 2,364,480 flip-flops (FFs), and 1,182,240 lookup tables (LUTs).

To improve emulation efficiency on FPGA platforms, several optimized configurations were implemented, including PIFNN-Opt (double), PIFNN-Opt (float), PIFNN-Opt (fp), PIRNN-Opt (double), PIRNN-Opt (float), and PIRNN-Opt (fp). The PIFNN-Opt and PIRNN-Opt models under double- and single-precision floating-point arithmetic employ pipelining to achieve a balanced trade-off between resource utilization and latency. In contrast, PIRNN-Opt (fp) adopts loop unrolling with an unroll factor of 10, thereby exploring the minimum achievable latency for recurrent architectures. It is worth noting that BRAM utilization is negligible across all designs, as no on-chip memory blocks are allocated for storing large runtime parameters or temporary data. This reflects an engineering trade-off that prioritizes DSP, LUT, and FF resources, while minimizing reliance on BRAM.

The influence of numerical precision on both model performance and hardware resource consumption was further investigated. Table 5 compares implementations using double precision, single precision (float), and fixed-point arithmetic (32-bit word length with 10 integer bits). Results indicate that while double-precision arithmetic achieves the highest numerical accuracy, it demands significantly more DSP and LUT resources. Single precision reduces resource utilization and latency by over 40% relative to double precision, with only a negligible reduction in accuracy (relative error $< 0.05\%$). Fixed-point implementations provide the most favorable trade-off. With 32-bit fixed-point numbers (10 integer bits and 22 fractional bits), the minimum representable resolution is 2^{-22}, and the interval between adjacent numbers is correspondingly fine. This design yields relative errors within 0.1% compared to floating-point results, while simultaneously reducing latency to as few as 24 cycles for PIFNN and 22 cycles for PIRNN. Resource consumption is also further reduced, confirming the practicality of fixed-point quantization in real-time SMR simulations.

These results demonstrate that an appropriate quantization scheme enables a balanced compromise between accuracy and efficiency. Although double-precision arithmetic guarantees maximum numerical fidelity, its hardware cost is prohibitive for real-time deployment. Single-precision offers a resource-efficient alternative with minimal loss in accuracy, whereas the fixed-point design achieves the lowest latency and resource footprint without compromising practical performance. Taken together, the fixed-point configuration provides a robust and efficient solution for FPGA-based SMR applications.

Table 4
SMR TZR models hardware resource consumption on Xilinx® UltraScale+™ XCVU9P FPGA.

Model	DSP	FF	LUT	Latency (cycles)
Neutron kinetics	136	20,553	14,418	122
Fuel/Coolant Temp.	13	1,107	987	17
Hot/Inlet Temp.	7	592	622	14
Press./Temp.	18	1,316	1,264	16
Turbine Dyn.	28	2,095	2,035	26
Control	5	467	923	22
Conventional SMR	207	26,130	20,249	312

Table 5
SMR PINN models hardware resource consumption on Xilinx® UltraScale+™ XCVU9P FPGA.

Model	DSP	FF	LUT	Latency (cycles)
PIFNN-NoOpt (float)	50	6,062	4,469	96
PIFNN-Opt (double)	70	6,086	5,555	52
PIFNN-Opt (float)	25	3,085	2,244	52
PIFNN-Opt (fp)	30	1,217	1,818	24
PIRNN-NoOpt (float)	77	8,859	7,266	296
PIRNN-Opt (double)	115	9,870	13,800	141
PIRNN-Opt (float)	42	4,847	6,762	112
PIRNN-Opt (fp)	164	6,450	13,124	22
PIML SMR	258	35,122	28,835	122

Table 6
Parameters of initial states and triggers.

Feature	$n(t_0)$	$T_f(t_0)$	$T_{c1}(t_0)$	$T_{c2}(t_0)$	$T_{HL}(t_0)$
State1	2.500e8	8.265e2	2.968e2	3.117e2	3.117e2
State2	2.470e8	8.213e2	2.980e2	3.127e2	3.127e2
Feature	$T_{cin}(t_0)$	$T_p(t_0)$	$T_m(t_0)$	$P_s(t_0)$	$P_{HP}(t_0)$
State1	2.819e2	2.968e2	2.929e2	9.203e−1	3.000e−1
State2	2.833e2	2.980e2	2.941e2	9.242e−1	2.100e−1
Feature	$P_{LP}(t_0)$	$P_{RH}(t_0)$	$g(t_0)$	$y_g(t_0)$	$\mu(t_0)$
State1	7.000e−1	−2.154e−8	1	0	1
State2	4.900e−1	−5.575e−8	7.028e−1	−4.517e−6	7.028e−1
Time S1	0.5 s	1.51 s	2.52 s	3.53 s	4.54 s
P_{ref}	0.9 p.u.	0.7 p.u.	0.8 p.u.	0.9 p.u.	1 p.u.
Time S2	0.5 s	2 s	3.5 s		
P_{ref}	0.1 p.u.	1.9 p.u.	0.7 p.u.		

6. Hardware results and discussion

To assess the generalization and robustness of the proposed framework, comparative simulations were performed under two distinct scenarios, denoted as State 1 and State 2, corresponding to initial condition 1 (IC1) and initial condition 2 (IC2), respectively. State 1 represents the baseline operating point, whereas State 2 is defined by a different set of initial parameters that were not included in the training process. In both scenarios, the system was subjected to transient power demand variations, including abrupt load additions and shedding. The results indicate that the framework consistently captures the dynamic response and maintains stability across different initial conditions.

6.1. Flexibility of hardware resource utilization

The FPGA resource utilization for each subsystem of the SMR model is summarized in Table Table 4 and Table Table 5. The conventional modeling approach, based on serial computations using state-space equations and explicit function evaluations, requires relatively fewer hardware resources but exhibits significantly higher latency. In contrast, the proposed PIML-based SMR modeling framework introduces neural network models for certain subsystems, which slightly increases hardware resource consumption but significantly reduces overall computational latency.

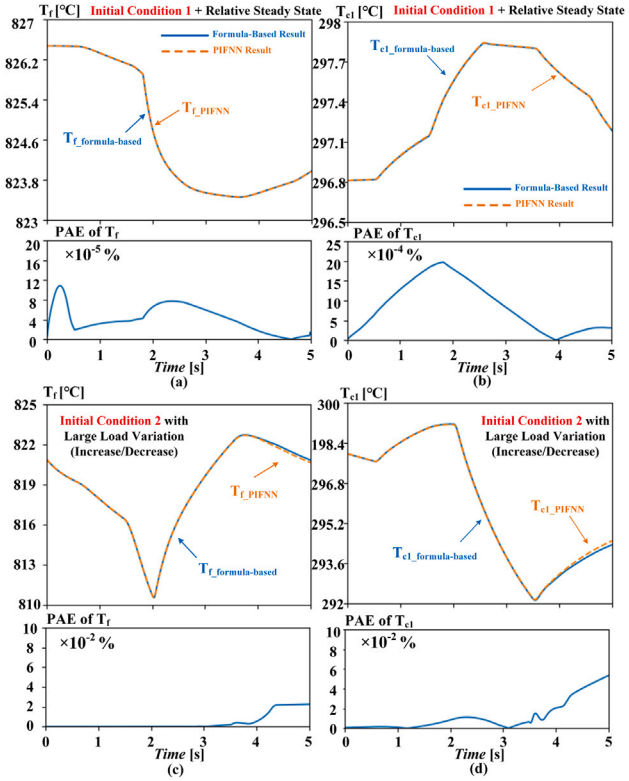

Fig. 11. PIFNN results for the fuel and coolant temperature dynamics: (a) IC1 fuel temperature; (b) IC1 coolant temperatures at the first node; (c) IC2 fuel temperature; and (d) IC2 coolant temperatures at the first node.

As shown in Table Table 5, the PIML method provides flexibility in the model design space by supporting different levels of neural network optimization. For floating-point implementations, pipeline optimization is adopted as a balanced solution between resource consumption and latency. For fixed-point implementations, loop unrolling is applied to minimize execution latency. Despite increased DSP utilization in some configurations (e.g., PIRNN-Opt(fp)), the resource growth remains within acceptable bounds, while latency is reduced dramatically.

This flexibility is a key benefit of the PIML modeling paradigm: it allows designers to tune the trade-offs between hardware consumption, performance latency, and model complexity based on system-level constraints. Moreover, the consistent structural patterns of neural networks facilitate unified hardware optimization across multiple subsystems, enabling efficient reuse of accelerator designs.

Finally, it is worth noting that the resource figures presented do not yet exploit the full optimization capabilities of low-level hardware synthesis tools. For instance, Xilinx Vivado HLS currently uses default serial accumulation for vector addition and matrix multiplication, whereas adopting tree-structured parallel computations may further improve resource efficiency and reduce latency.

6.2. Efficiency in computational latency

As summarized in Table Table 4 and Table Table 5, the conventional SMR exhibits significantly higher latency, consuming up to 312 computational cycles. In contrast, the proposed the PIML-based SMR

modeling approach leads to a modest increase in hardware resource usage, it effectively reduces the computational latency to approximately 40% of the conventional method. A key advantage of the proposed PIML framework lies in its capacity to meet stringent real-time emulation constraints, particularly in large-scale multi-domain systems such as submarine power distribution. For instance, conventional simulation platforms such as MATLAB are unable to achieve real-time performance when modeling the full SMR power distribution system. MATLAB-based simulations typically compute the entire system sequentially using state-space formulations, and despite high central processing unit (CPU) clock speeds, the computational burden results in a poor simulation ratio. For a system using a time-step of 100 μs, the computation-to-simulation-time ratio can reach 10:1–requiring 10 seconds of computational time to simulate just 1 second of system behavior.

By contrast, the proposed FPGA-based approach using conventional numerical methods, with subsystem-level parallelization, can complete SMR computation within 312 clock cycles–equivalent to 3.12 μs at a 100 MHz clock frequency. This leads to a computation-to-simulation-time ratio of approximately 1:32, enabling faster-than-real-time emulation. For the electrical subsystem, which typically requires a finer resolution of 2 μs, real-time execution is also achievable due to parallel hardware scheduling.

Furthermore, the PIML-accelerated FPGA implementation offers even greater computational efficiency. The PIML-based SMR achieves full step computation within just 112 clock cycles (1.12 μs), yielding a computation-to-simulation-time ratio ratio of 1:90–an approximately 60% reduction in latency compared to the conventional FPGA-based numerical method. For the power system with a 2 μs emulation time-step, real-time execution is similarly sustained. These results underscore the superior efficiency and scalability of the proposed method in handling complex multi-physics systems with demanding timing constraints.

6.3. Accuracy preservation in PIML models

Table 6 presents the key parameters of the two initial states, designated as State 1 and State 2, corresponding to IC1 and IC2, respectively. For each state, the initial values of neutron density, fuel and coolant temperatures, hot-leg and inlet temperatures, pressures, control variables, and other relevant system variables are provided. The table also specifies the timing of transient events (S1 and S2) and the associated reference power setpoints that trigger system responses. This detailed specification enables clear analysis of the system's dynamic behavior under varying ICs.

Fig. 11 illustrates PIFNN results for fuel temperature T_f and coolant temperature at the first node T_{c1}. A notable transient occurs around 1.5 s, where the fuel temperature decreases sharply from 826.2 °C to approximately 823.8 °C in Fig. 11(a). In Fig. 11(b), the coolant temperature at the first node shows a corresponding transient increase from 296.8 °C to approximately 297.7 °C between 1 s and 2 s, closely matching the formula-based reference. For the second IC, which involves a large load variation, as shown in Figs. 11(c) and (d), the system exhibits a more pronounced transient. The fuel temperature T_f decreases significantly from approximately 825 ° C to a minimum of about 811 ° C around 3 s, followed by a gradual recovery. Simultaneously, the coolant temperature T_{c1} initially peaks near 300 ° C before sharply dropping to approximately 292 ° C. Despite this highly dynamic scenario, the PIFNN results remain in strong agreement with the formula-based reference, with the percentage absolute error (PAE) on the order of 10^{-2}%, confirming the accuracy under large-scale transients.

In Fig. 12, the hot-leg temperature T_{HL} and inlet coolant temperature T_{cin} demonstrate similar transient dynamics. In the baseline state, T_{HL} rises from 311.7 °C at 1.5 s to about 312.8 °C around 3 s, closely

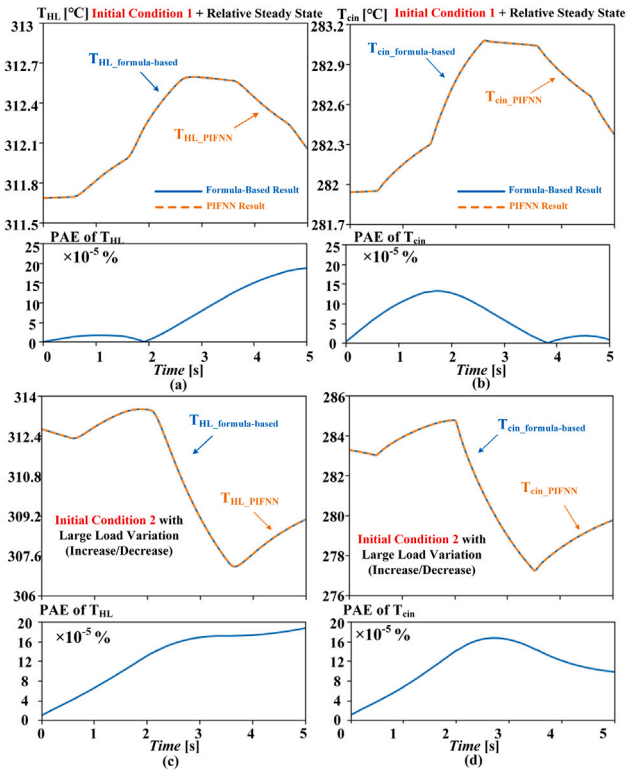

Fig. 12. PIFNN results for the hot leg and inlet temperature dynamics: (a) IC1 hot leg temperature; (b) IC1 inlet coolant temperature; (c) IC2 hot leg temperature; and (d) IC2 inlet coolant temperature.

followed by a subsequent decline. Meanwhile, the inlet coolant temperature shows a concurrent increase, peaking near 282.9 °C at about 2.5 s. Under the large load variation condition shown in Figs. 12(c) and (d), both temperatures exhibit a more substantial transient, characterized by a significant drop. The hot-leg temperature T_{HL} decreases from approximately 313.5 ° C to a minimum of about 308.5 ° C near 3.5 s. Correspondingly, the inlet coolant temperature T_{cin} undergoes a sharp decline from around 283 ° C to a low of approximately 278 ° C. In this highly dynamic scenario, the PIFNN predictions remain in excellent agreement with the formula-based results, maintaining a PAE on the order of 10^{-5}%, demonstrating the robustness and precision of the model.

In the baseline state, Figs. 13(a) and (b), presents the coolant mixing temperature T_m and system pressure P_s. T_m increases from 292.8 °C at 0.5 s, reaching a maximum of approximately 293.9 °C at around 2.5 s before decreasing. Simultaneously, system pressure P_s rises notably from 18.4 MW to approximately 18.48 MW within the same period. In the case of a large load variation, illustrated in Figs. 13(c) and (d), the system dynamics are significantly more pronounced. The coolant mixing temperature T_m initially rises to a peak of approximately 295.8 ° C, then drops sharply to a minimum of about 288.5 ° C around 3.5 s. The system pressure P_s exhibits more complex behavior; it initially peaks, then decreases substantially to below 17.9 MW, followed by gradual recovery. In this scenario, the PIFNN model successfully captures the overall dynamic trends. Although a slightly larger deviation from the formula-based results is observed for P_s during the rapid transient, the PAE for both variables remains low (on the order of 10^{-1}%).

Fig. 14 depicts turbine dynamics using PIRNN, showing the high-pressure turbine power output P_{HP} and intermediate-low pressure turbine power output P_{LP}. A prominent transient in P_{HP} occurs between 1 s and 3 s, dropping sharply from about 5.9 MW to 4.4 MW, then recovering. Similarly, P_{LP} exhibits a steep decline from 13.9 MW

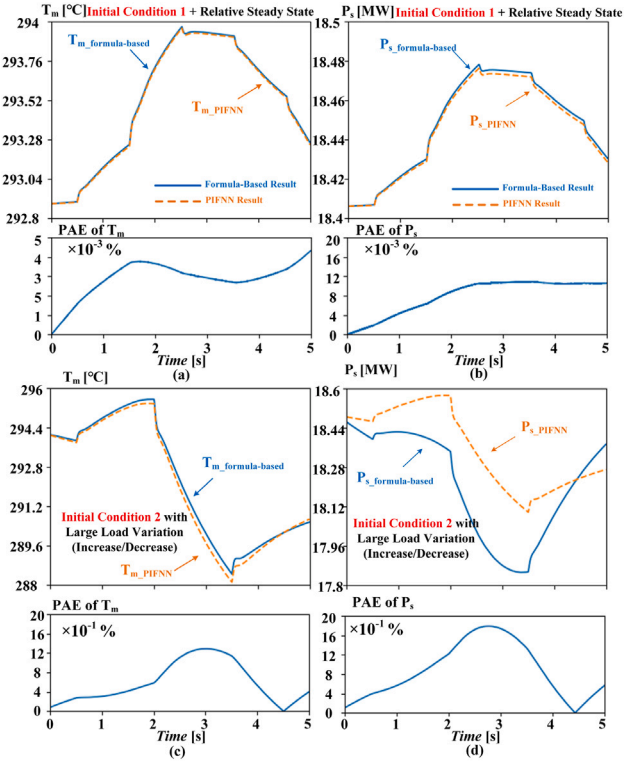

Fig. 13. PIFNN results for the pressure and temperature dynamics: (a) IC1 coolant mixing temperature; (b) IC1 system pressure; (c) IC2 coolant mixing temperature; and (d) IC2 system pressure.

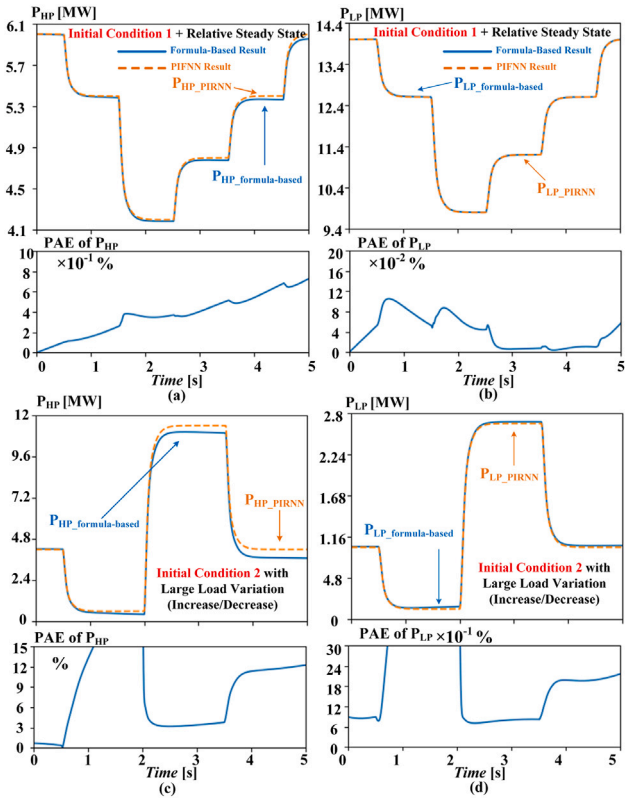

Fig. 14. IRNN results for turbine dynamics: (a) IC1 high-pressure turbine power output; (b) IC1 intermediate-low pressure turbine power output; (c) IC2 high-pressure turbine power output; and (d) IC2 intermediate-low pressure turbine power output.

at 1 s to around 10.2 MW at 2.5 s, followed by a rapid recovery. Under the second initial condition involving a large load variation, depicted in Figs. 14(c) and (d), the system undergoes a significant step-increase in power. At approximately 2.5 s, the high-pressure turbine power P_{HP} surges abruptly from 2.4 MW to 10.8 MW, while the intermediate-low pressure power P_{LP} jumps from near zero to 2.6 MW. The PIRNN model accurately captures both the timing and magnitude of these sharp, step-like changes. Although a momentary spike in the PAE occurs at the rapid transition, the model output quickly converges with the reference, demonstrating its robustness under severe transient conditions.

Fig. 15 compares PMSM dynamics modeled by traditional EMT and PIFNN methods. The PIFNN accurately reproduces three-phase currents, torque transients, and speed variations, with peak PAE under 5%. Significant torque fluctuations at around 1s and speed step changes near 2.5s are closely matched, demonstrating PIFNN effectiveness for PMSM real-time emulation. Overall, the exceptional alignment of neural network predictions with formula-based references, evidenced by low MPAE (less than 1% in general), underscores the efficacy of PIFNN and PIRNN approaches for accurate, real-time FPGA-accelerated emulation in complex multi-domain modeling scenarios.

7. Conclusion

In this paper, an AI-accelerated, physics-informed transient RTDT for an SMR-based submarine multi-domain power distribution system was developed. Key conclusions are:

(1) The proposed PIML-based modeling framework successfully integrates PIFNN and PIRNN architectures for subsystem-level nonlinear modeling. When deployed on FPGA hardware, it reduces computational latency by approximately 60% compared to traditional serial finite-element (FE) and trapezoidal rule (TZR) methods (as shown in Table 4), enabling faster-than-real-time execution for SMR subsystems with a latency of only 1.12 μs per 100 μs time-step.

(2) Comprehensive FPGA implementation analysis demonstrated that fixed-point models with loop unrolling achieved the lowest latency (22–24 cycles), while incurring only moderate increases in DSP and logic usage compared to floating-point pipelined implementations. This validates their efficiency in balancing execution speed with hardware resource constraints.

(3) Extensive RTDT emulation results across thermal, pressure, and electrical subsystems show strong agreement between PIML predictions and reference EMT simulations. The models consistently achieved MPAE below 1%, and successfully captured dynamic transients in all domains, confirming their suitability for RTDT applications in multi-domain submarine power systems.

Taken together, these results validate the proposed PIML framework as a powerful methodology for developing high-performance digital twins that successfully bridge the gap between computational speed and physical accuracy. It is important to acknowledge the study's limitations: the current model is tailored to the specific SMR architecture presented, and its generalizability to different reactor types warrants further investigation. Additionally, since validation was performed against reference simulations, testing with physical hardware controllers remains a crucial step to fully confirm its real-world applicability. Future work will focus on further optimizing neural network architectures and exploring additional system domains to enhance the scalability and generalizability of the proposed methodology. The developed RTDT framework can also be extended and applied to other advanced transportation systems, such as marine vessels, high-speed rail, and aerospace applications.

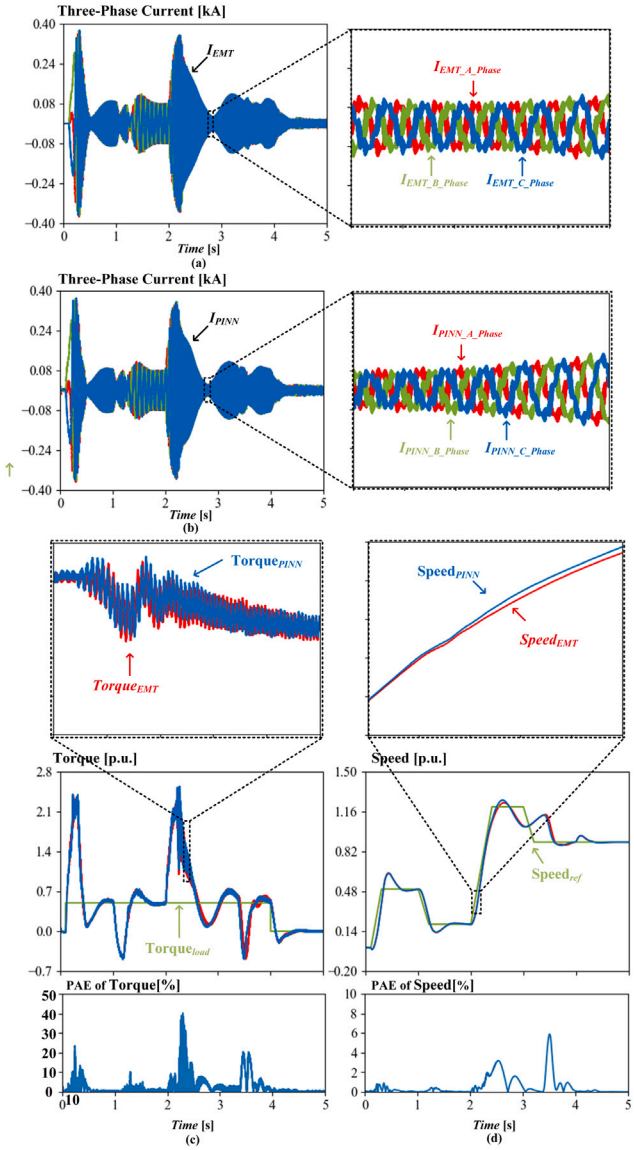

Fig. 15. PIFNN results for PMSM: (a) three-phase currents of FE model; (b) three-phase currents of PIFNN model; (c) torque; (d) speed.

Abbreviations

AI	Artificial Intelligence
BRAM	Block Random-Access Memory
CPU	Central Processing Unit
DSP	Digital Signal Processor
EMT	Electromagnetic Transient
FE	Forward Euler
FF	Flip-Flop
FPGA	Field-Programmable Gate Array
FNN	Fully Connected Neural Network
HLS	High-Level Synthesis
IRK	Implicit Runge-Kutta
IC	Initial Condition
LUT	Look-Up Table
MPAE	Mean Percentage Absolute Error
MSE	Mean Squared Error
PAE	Percentage Absolute Error
PFNN	Physics-Feature Neural Network
PIFNN	Physics-Informed Feedforward Neural Network
PIRNN	Physics-Informed Recurrent Neural Network
PIML	Physics-Informed Machine Learning
PINN	Physics-Informed Neural Network
PMSM	Permanent Magnet Synchronous Motor
PSCT-PINN	Pre-Solving Continuous-Time PINN
RK4	Fourth-Order Runge-Kutta
RNN	Recurrent Neural Network
RTDT	Real-Time Digital-Twin
SMR	Small Modular Reactor
TZR	Trapezoidal Rule

CRediT authorship contribution statement

Songyang Zhang: Writing – review & editing, Writing – original draft, Visualization, Software, Methodology, Formal analysis, Data curation. **Weiran Chen:** Writing – review & editing, Software, Data curation. **Yuzhong Zhang:** Writing – review & editing, Data curation. **Venkata Dinavahi:** Writing – review & editing, Supervision, Resources, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this manuscript.

Acknowledgments

Songyang Zhang acknowledges the financial support from Mitacs. Venkata Dinavahi acknowledges the financial support from the Natural Sciences and Engineering Research Council (NSERC) of Canada.

Data availability

Data will be made available on request.

References

[1] Skjong E, Volden R, Rødskar E, Molinas M, Johansen TA, Cunningham J, present Past. And future challenges of the marine vessels electrical power system. IEEE Trans Transp Electrific. 2016;2(4):522–37.
[2] Koziol M. The case for nuclear cargo ships. IEEE Spectr 2025. Online: https://spectrum.ieee.org/nuclear-powered-cargo-ship. [Accessed 30 March 2025].
[3] European Commission. Small modular reactors explained. European commission – energy. 2025, Online: https://energy.ec.europa.eu/topics/nuclear-energy/small-modular-reactors/small-modular-reactors-explained_en. [Accessed 30 March 2025].
[4] International atomic energy agency, what are small modular reactors (SMRs)?. 2025, Online: https://www.iaea.org/newscenter/news/what-are-small-modular-reactors-smrs. [Accessed 30 March 2025].
[5] Senemmar S, et al. Navigating the future: Exploring small modular reactors in the maritime sector. IEEE Electrification Mag 2024;12(4):30–4210, 1109/MELE.2024.
[6] Vujić J, Bergmann RM, Škoda M. Small modular reactors: Simpler, safer, cheaper? Energy 2012;45(1):288–95.
[7] Sainati T, Locatelli G, Brookes N. Small modular reactors: Licensing constraints and the way forward. Energy 2015;82:1092–5.
[8] Locatelli G, Boarin S, Pellegrino F, Ricotti ME. Load following with small modular reactors (SMR): A real options analysis. Energy 2015;80:41–54.
[9] Kang SW, Yim MS. Coupled system model analysis for a small modular reactor cogeneration (combined heat and power) application. Energy 2023;262:125481.
[10] Akgun I, Dincer I. Development of a smart powering system with ammonia fuel cells and internal combustion engine for submarines. Energy 2024;294:130747.
[11] Byun HH, Yim MS. Genetic algorithm-based non-synchronous unit-specific optimal load-following control of multi-unit small modular reactor. Energy 2025;314:134091.
[12] Liu H, Song Y, Xiao Q, Xu Q. Neural networks and adaptive finite-time state observer-based preassigned-time fault-tolerant control of load following for a PWR-smr under CRDM faults and sensor noises. Energy 2025;323:135689.

[13] Luo R, Li Y, Guo H, Wang Q, Wang X. Cross-operating-condition fault diagnosis of a small module reactor based on CNN-LSTM transfer learning with limited data. Energy 2024;313:133901.
[14] Zafar S, Khan A. Integrated hydrogen fuel cell power system as an alternative to diesel-electric power system for conventional submarines. Int J Hydrog Energy 2024;51:1560–72.
[15] V. Başhan. Comparative evaluation and selection of submarines with air-independent propulsion system. Int J Hydrog Energy 2022;47(86):36659–71.
[16] Chen W, Liang T, Dinavahi V. Comprehensive real-time hardware-in-the-loop transient emulation of MVDC power distribution system on nuclear submarine. IEEE Open J Ind Electron Soc 2020;1:326–39.
[17] Dinavahi V, Lin N. Real-time electromagnetic transient simulation of AC-DC networks. New Jersey: Wiley-IEEE Press; 2021.
[18] Li W, Li Y, Garg A, Gao L. Enhancing real-time degradation prediction of lithium-ion battery: A digital twin framework with CNN-LSTM-attention model. Energy 2024;286:129681.
[19] Barone G, Buonomano A, Papa GDel, Giuzio GF, Palombo A, Russo G. Towards sustainable ships: Advancing energy efficiency of HVAC systems onboard through digital twin. Energy 2025;317:134435.
[20] Zhang S, Dinavahi V, Liang T. Towards hydrogen-powered electric aircraft: Physics-informed machine learning based multi-domain modeling and real-time digital twin emulation on FPGA. Energy 2025;322:135451.
[21] Zhang S, Liang T, Dinavahi V. Hybrid ML-EMT-based digital twin for device-level HIL real-time emulation of ship-board microgrid on FPGA. IEEE J Emerg Sel Top Ind Electron 2023;4(4):1265–77.
[22] Zhang S, Liang T, Dinavahi V. Real-time HIL emulation of DRM with machine learning accelerated WBG device models. IEEE Open J Power Electron 2023;4:567–78.
[23] Zhang Y, Zhang S, Dinavahi V. A survey of machine learning applications in advanced transportation systems: Trends. Tech Futur Dir ETransportation 2025;24:100417.
[24] Rudy SH, Brunton SL, Proctor JL, Kutz JN. Data-driven discovery of partial differential equations. Sci Adv 2017;3(4):e1602614.
[25] Han J, Jentzen A, W E. Solving high-dimensional partial differential equations using deep learning. Proc Natl Acad Sci USA 2018;115(34):8505–10.
[26] Raissi M, Perdikaris P, Karniadakis GE. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J Comput Phys 2019;378:686–707.
[27] Wang Q, Ye M, Li B, Lian G, Li Y. Co-estimation of state of charge and capacity for battery packs in real electric vehicles with few representative cells and physics-informed machine learning. Energy 2024;306:132520.
[28] Ye J, Xie Q, Lin M, Wu J. A method for estimating the state of health of lithium-ion batteries based on physics-informed neural network. Energy 2024;294:130828.
[29] Lin C, Tuo X, Wu L, Zhang G, Lyu Z, Zeng X. Physics-informed machine learning for accurate SOH estimation of lithium-ion batteries considering various temperatures and operating conditions. Energy 2025;318:134937.
[30] Li P, Guo F, Li Y, Yang X, Yang X. Physics-informed neural network for real-time thermal modeling of large-scale borehole thermal energy storage systems. Energy 2025;315:134344.
[31] Lagomarsino-Oneto D, Meanti G, Pagliana N, Verri A, Mazzino A, Rosasco L, Seminara A. Physics informed machine learning for wind speed prediction. Energy 2023;268:126628.
[32] Li X, et al. Temporal modeling for power converters with physics-in-architecture recurrent neural network. IEEE Trans Ind Electron 2024;71(11):14111–23.
[33] Fassi Y, Heiries V, Boutet J, Boisseau S. Toward physics-informed machine-learning-based predictive maintenance for power converters—A review. IEEE Trans Power Electron 2024;39(2):2692–720.
[34] Mi L, Han Y, Long L, Chen H, Cai CS. A physics-informed temporal convolutional network-temporal fusion transformer hybrid model for probabilistic wind speed predictions with quantile regression. Energy 2025;326:136302.
[35] Wang K, Wang L, Meng Q, Yang C, Lin Y, Zhu J, Zhao Z, Zhou C, Zheng C, Gao X. Accurate photovoltaic power prediction via temperature correction with physics-informed neural networks. Energy 2025;328:136546.
[36] Tian A, He L, Ding T, Dong K, Wang Y, Jiang J. A generic physics-informed neural network framework for lithium-ion batteries state of health estimation. Energy 2025;332:137215.
[37] Kerlin TW, Katz EM. Pressurized-water-reactor modeling for long-term power-system-dynamics simulations. In: Final rep. no. EPRI-EL–3087-vOL. 2. Tennessee Univ.; 1983.
[38] Ichikawa T, Inoue T. Light water reactor plant modeling for power system dynamics simulation. IEEE Trans Power Syst 1988;3(2):463–71.
[39] Di Lascio MS, Moret R, Poloujadoff M. Reduction of program size for long-term power system simulation with pressurized water reactor. IEEE Power Eng Rev PER 1983;3(3). 43–43.
[40] Kingma DP, Ba J. Adam: A method for stochastic optimization. In: Proc. 3rd int. conf. learn. represent.. San Diego, CA, USA; 2015, p. 1–15.