Supplementary Information

QuanDT: Quantum Digital Twin with Applications to Smart Grid

Wenxuan Ma[1,*], Mengxiang Liu[2,*], Shang Yu[3,4], Kuan-cheng Chen[2,4], Yitian Zhou[1], Rong Guo[1], Yifan Liu[1], Ruilong Deng[1,✉], Peng Cheng[1], and Jiming Chen[1]

[1]Zhejiang University, State Key Laboratory of Industrial Control Technology, College of Control Science and Engineering, Hangzhou, 310027, China

[2]Imperial College London, Department of Electrical and Electronic Engineering, London, SW7 2AZ, UK

[3]Imperial College London, Blackett Laboratory, Department of Physics, London, SW7 2AZ, UK

[4]Imperial College London, Centre for Quantum Engineering, Science and Technology, London, SW7 2AZ, UK

[*]these authors contributed equally to this work

[✉]corresponding author: dengruilong@zju.edu.cn

Contents

List of Supplementary Figures

List of Supplementary Tables

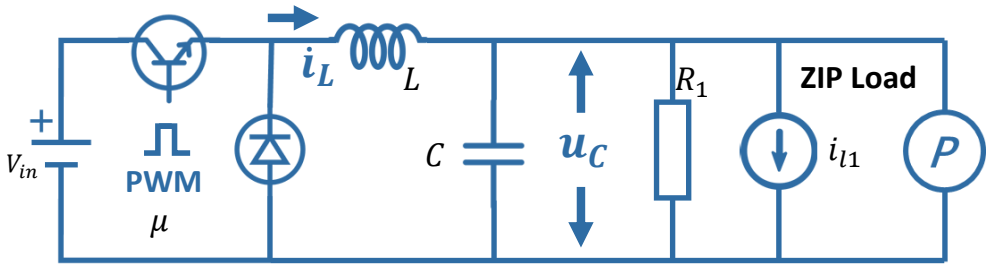
List of Supplementary Algorithms

I Details of QuanDT for DC-DC Buck Converter

A Theory Model of the System

A.1 Model of DC-DC Buck Converter with ZIP Load

A DC-DC converter is an electronic circuit that converts a direct current (DC) voltage source from one voltage level to another. With the rapid advancement of power electronics technology, DC-DC converters have become integral components in electric vehicles, DC microgrids and renewable energy generation systems, and are widely used in these domains.[1] In systems incorporating renewable energy sources such as photovoltaics and energy storage, DC-DC converters enable voltage conversion—step-up or step-down—to provide suitable voltage levels for loads with varying voltage specifications.[2] One of the most commonly used DC-DC converter topologies is the Buck converter, which is specifically designed to reduce a high DC voltage to a more manageable level.[3]



Supplementary Figure 1: A DC-DC Buck Converter with ZIP Load.

As illustrated in SFig. 1, the buck converter has a simple structure, consisting of a main circuit made up of a transistor and a diode controlled by a pulse-width modulation (PWM) signal with a duty cycle $\mu \in (0, 1)$ for switching, and passive components such as an inductor L and a capacitor C for output filtering. Together with the renewable energy source and the load, the converter forms a nonlinear second-order system.[4] Since the time scale of variations in renewable energy generation is typically much slower than the system dynamics, and considering that renewable energy sources are often coupled with energy storage systems to mitigate stochastic fluctuations, the input can be modeled as a constant DC voltage source V_{in}.[5]

Load modeling is also essential for both steady-state and dynamic analysis of power systems.[6] Modern residential, commercial, and industrial loads can often be modeled as ZIP loads, which comprise a parallel combination of resistive load R_1, current-type load i_{l1}, and power-type load P.[7] To incorporate the power-type load into the model, the current-voltage relationship $i = P/u$ around

an operating voltage V_0 can be linearized using a first-order Taylor expansion, yielding

$$i = \frac{P}{V_0} - \frac{P}{V_0^2}(u - V_0) = -\frac{P}{V_0^2}u + \frac{2P}{V_0}, \tag{1}$$

where i and u represent the current through and voltage across the power-type load, respectively. This indicates that the power-type load can be equivalently represented as a combination of a current-type load,

$$i_{l2} = \frac{2P}{V_0}, \tag{2}$$

and an impedance load,

$$R_2 = -\frac{V_0^2}{P}. \tag{3}$$

In summary, the overall ZIP load can be represented as a parallel connection of R_1 and R_2, and a total current-type load formed by i_{l1} and i_{l2}, that is

$$R = R_1 // R_2 = R_1 // \left(-\frac{V_0^2}{P}\right), \tag{4}$$

$$i_l = i_{l1} + i_{l2} = i_{l1} + \frac{2P}{V_0}. \tag{5}$$

Within one switching cycle, the system dynamics can be derived based on Kirchhoff's laws.[8] When the switch is in the ON state, the state variables, inductor current i_L and capacitor voltage u_C, satisfy

$$i_L = C\frac{du_C}{dt} + \frac{u_C}{R} + i_l, \tag{6}$$

$$u_C = V_{\text{in}} - L\frac{di_L}{dt}. \tag{7}$$

When the switch is in the OFF state, the equations become:

$$i_L = C\frac{du_C}{dt} + \frac{u_C}{R} + i_l, \tag{8}$$

$$u_C = -L\frac{di_L}{dt}. \tag{9}$$

Using state-space averaging method,[9] the unified state-space model of the DC-DC Buck converter with ZIP load over a switching period can be written as

$$\begin{bmatrix} \dot{i}_L \\ \dot{u}_C \end{bmatrix} = \begin{bmatrix} 0 & -\frac{1}{L} \\ \frac{1}{C} & -\frac{1}{RC} \end{bmatrix} \begin{bmatrix} i_L \\ u_C \end{bmatrix} + \begin{bmatrix} \frac{V_{\text{in}}}{L} & 0 \\ 0 & -\frac{1}{C} \end{bmatrix} \begin{bmatrix} \mu \\ i_l \end{bmatrix}. \tag{10}$$

A.2 Physical Controller and Fault Detector Model

Physical Controller Model: As the foundational layer for more advanced control strategies in renewable energy generation systems,[10] DC-DC converters rely on their controllers to generate PWM signals with appropriate duty cycles to regulate output voltage or current.[11] To address the demands for fast dynamic response, strong stability margins, and high robustness, a wide range of advanced control approaches—such as sliding mode control, model predictive control, and intelligent control—have been proposed for DC-DC converter applications.[12] In this paper, for simplicity and without loss of generality, we employ a basic proportional-integral (PI) control strategy to regulate the output voltage of the buck converter. The control law is defined as

$$\begin{aligned} \mu &= \mathrm{clip}\left(K_p e(t) + K_i \int_0^t e(\tau)\, d\tau;\, 0, 1\right) \\ &= \min\left(\max\left(K_p e(t) + K_i \int_0^t e(\tau)\, d\tau,\, 0\right),\, 1\right), \end{aligned} \tag{11}$$

where μ denotes the duty cycle, constrained within $(0, 1)$ by the clipping function $\mathrm{clip}(\cdot)$, which limits the controller output to a valid PWM range. The error signal $e(t)$ is given by

$$e(t) = V_{\mathrm{set}} - V_{\mathrm{meas}}, \tag{12}$$

with V_{set} representing the voltage reference and V_{meas} the actual measured output voltage.

By tuning the proportional gain K_p and integral gain K_i appropriately, this simple PI control law can achieve effective regulation of the buck converter's output voltage to meet typical performance requirements.

Fault Detector Model: With the increasing digitalization and intelligence of modern power systems, controllers play a crucial role as the core units responsible for system regulation and coordination across all levels of operation. From large-scale centralized grids to distributed energy systems, controllers are widely deployed for key tasks such as voltage regulation, frequency stabilization, power allocation, and energy dispatch. However, the complexity of controllers and their reliance on external information also make them a potential vulnerability in power system operations. A malfunction or targeted attack on a controller can lead to system performance degradation at best, or even trigger cascading failures that jeopardize the stability and safety of the entire grid.[13]

Controller failures are generally categorized into hardware, software, and communication failures. Hardware failures include processor damage, sensor malfunctions, and power interruptions, which can directly disrupt or impair control functions.[14] Software failures often stem from algorithmic flaws, data overflows, or logical errors, resulting in abnormal system responses or incorrect control actions.[15] Communication failures, such as signal delays, data loss, or synchronization

issues, can hinder information exchange between controllers, sensors, and actuators, ultimately degrading overall control performance.[16] These failures are typically unpredictable, with abrupt and cascading characteristics, posing significant challenges to secure operation of the power grid.

More concerning is the recent rise in cyber-attacks targeting controllers, characterized by increasing sophistication and diversity in attack strategies.[17] Attackers may disrupt the sensing, decision-making, and execution processes of controllers via techniques such as data injection, command tampering, or communication interference. Among them, false data injection (FDI) attacks which forge values have been shown to cause significant controller misjudgment while evading traditional anomaly detection schemes. Denial-of-service (DoS) attacks flood controllers with excessive requests, blocking communication channels and rendering control strategies ineffective. Other tactics, such as replay attacks, malicious command injection, and manipulation of control algorithms, may cause severe disruption at the control layer, which can propagate to the physical infrastructure, leading to equipment damage or even system-wide blackouts.

To address these threats, extensive research efforts have emerged from both academia and industry to enhance the security and robustness of control systems. On one hand, by incorporating detection and mitigation mechanisms, it is possible to design control strategies with attack-awareness and adaptive resilience, improving a system's survivability and responsiveness under adversarial conditions.[18] On the other hand, to defend against unforeseen or complex threats, redundancy and fault-tolerant design offer a final safeguard to ensure operational continuity and system stability.[19]

In this study, we consider scenarios in which the controller's output signal experiences a sudden drop, caused by events such as power interruptions, algorithmic output overflows, or false data injection attacks. A fault detection module is designed to simultaneously receive control commands from both the physical controller and its digital twin counterpart. Under normal operating conditions, the module forwards the control signal generated by the physical controller. However, when an abnormal condition is detected, the module switches to the control signal provided by the digital twin controller.

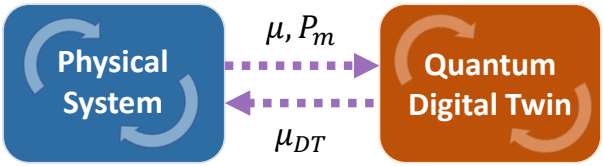
A.3 QuanDT Model

Constant power loads (CPLs) refer to a class of loads whose power consumption remains approximately constant.[20] Traditional CPLs are typically industrial automation devices operating under constant power control modes, such as variable-frequency drives, arc welding machines, computer numerical control (CNC) equipment, and laser processing systems. In addition, many types of modern power electronic devices—such as computers, communication equipment, inverter-driven motor systems, data centers, and server clusters—also behave as CPLs due to their use of tightly controlled switching power electronic converters. These loads are increasingly prevalent in modern power systems and are known to exhibit a negative impedance characteristic, potentially leading

to system instability when the CPL penetration level is high.[21] Meanwhile, with the rapid rise of large language models (LLMs), the deployment of AI infrastructure has surged, introducing a new class of high-power-consumption systems with highly volatile power profiles.[22] These systems pose significant threats to the reliability and resilience of the power grid, further underscoring the critical importance of monitoring power loads.

In the context of the Buck converter with ZIP load considered in this work, we focus on the effects of power fluctuations in CPL. As analyzed previously, a power load can be equivalently modeled as a combination of a current load i_{l2}, given by Eq. (2), and an impedance-type load R_2, given by Eq. (3). When the power varies, it can be observed from the expressions that the corresponding equivalent current and impedance components also change accordingly. Consequently, both the total equivalent impedance R, formed by the parallel connection of R_1 and R_2, and the total current input i_l, formed by the sum of i_{l1} and i_{l2}, will be affected. Ultimately, this leads to variations in both the system matrix and input of the state-space model described in Eq. (10).

DTs are expected to capture such variations and provide real-time tracking of the actual system. From the experimental studies in the main body of the manuscript, the necessity of capturing load power fluctuations is evident. For instance, in Scenario 3, the changes in power load can alter the system's dynamic characteristics by affecting its structure parameters. As a result, a machine learning model trained for one operating state cannot simply be transferred to another operating state. Instead, it requires data generated by a DT that can track the actual system in real time to build the training model.



Supplementary Figure 2: Data flow between physical system and DT.

To enable the DT to capture such variations in the system, real-time measurements of the power load P_m can be taken. Based on these measurements, the system matrix used for simulation can be updated according to Eq. (2)-(5) and Eq. (10). Additionally, as shown in the SFig.2, the QuanDT requires the real-time control signal μ from the physical controller as inputs for the simulation calculations. Meanwhile, the DT controller running simultaneously within the DT system will output the control signal μ_{DT} based on the simulation results of the QuanDT, which will serve as the backup signal.

The core of QuanDT is to solve models using quantum algorithms. The conventional method for solving Eq. (10) involves discretizing it with a certain time step. Although this method remains applicable when the system is time-varying, it may encounter stability issues when the system

stiffness is large. Another approach is to use an analytical solution form which includes matrix exponentiation and inversion as follows

$$x(t) = e^{At}x(0) + (e^{At} - I)A^{-1}Bu. \tag{13}$$

For the Buck converter with ZIP load, the system can be expressed as

$$A = \begin{bmatrix} 0 & -\frac{1}{L} \\ \frac{1}{C} & -\frac{1}{RC} \end{bmatrix}, \quad B = \begin{bmatrix} \frac{V_{\text{in}}}{L} & 0 \\ 0 & -\frac{1}{C} \end{bmatrix}, \quad x = \begin{bmatrix} i_L \\ u_C \end{bmatrix}, \quad u = \begin{bmatrix} \mu \\ i_l \end{bmatrix}. \tag{14}$$

This method, in comparison with the discretization one, offers higher accuracy and is widely used in the mainstream power system simulation tools, e.g., Simulink, Opal-RT, Typhoon HIL. However, for classical computers, the approximate calculation in Eq. (13) has at least $O(N^3)$ complexity. In DT scenario, since the system is time-varying, the system needs to be continuously solved. When the scale of matrix A becomes large, real-time computation of the system will become challenging. However, by using specific quantum circuits, efficient computation can be achieved with a logarithmic time complexity in terms of system size. Although the system case of the Buck converter is not large, it serves as a simple example to clearly illustrate the concept of QuanDT and provides a research method for QuanDT under current intermediate-scale noisy quantum (NISQ) conditions.

Specifically speaking, the solution $x(t)$ of problem (13) can be approximated by a k-th order Taylor expansion as

$$x(t) \approx \sum_{m=0}^{k} \frac{(At)^m}{m!}x(0) + \sum_{n=1}^{k} \frac{A^{(n-1)}t^n}{n!}Bu. \tag{15}$$

Let $b \triangleq Bu$, and then define

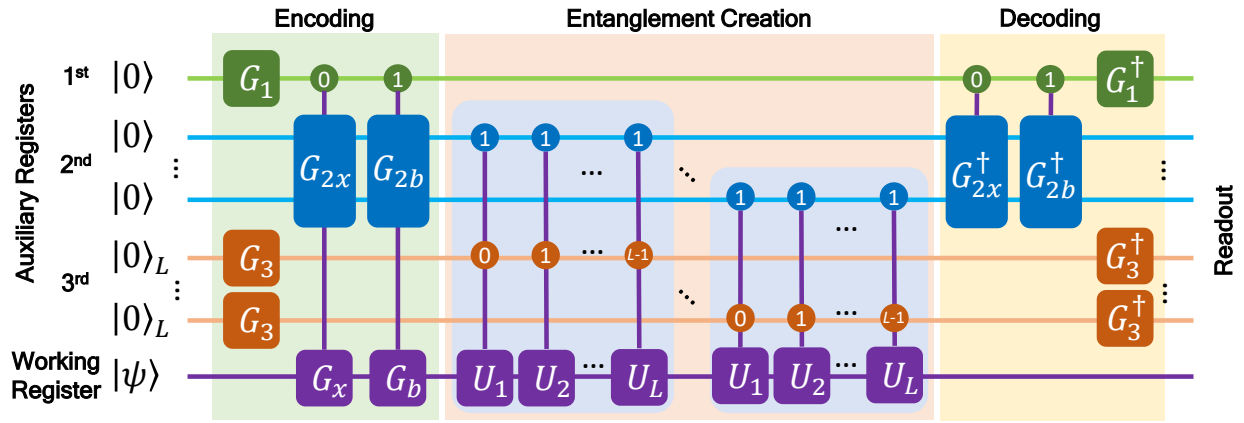
$$|b\rangle \triangleq \sum_j \frac{b_j}{\|b\|}|j\rangle, |x(0)\rangle \triangleq \sum_j \frac{x_j(0)}{\|x(0)\|}|j\rangle, \hat{A} \triangleq \sum_{i,j} \frac{A_{ij}}{\|A\|}|i\rangle\langle j| \tag{16}$$

where b_j, $x_j(0)$, and A_{ij} represent the j-th element of vectors and the element at the i-th row and j-th column of the matrix, $\|\cdot\|$ denotes the norm of vectors and the matrix. For the matrix $\hat{A}$, it can be linearly decomposed using unitary operators U_l as $\hat{A} = \sum_{l=1}^{L} \lambda_l U_l$. Then, Eq. (15) can be expressed as

$$x(t) \approx \sum_{m=0}^{k} \frac{\|x(0)\|(\|A\|t)^m(\sum_{l=1}^{L} \lambda_l U_l)^m}{m!}|x(0)\rangle + \sum_{n=1}^{k} \frac{\|b\|(\|A\|t)^{n-1}t(\sum_{l=1}^{L} \lambda_l U_l)^{n-1}}{n!}|b\rangle. \tag{17}$$

We define $|x(t)\rangle \triangleq x(t)/S$, where S is a normalization parameter, and it is the quantum state representation of the problem. Then we can get $x_j(t) = S\langle j|x(t)\rangle$. By utilizing the quantum circuit as shown in SFig. 3, an efficient computation of $|x(t)\rangle$ can be achieved.[23]

This circuit includes $\log(N)$ working registers below and three parts of auxiliary registers above, consisting of one qubit in the first part, k qubits in the second part, and k groups of qubits of L-level quantum systems. Specifically, the algorithm can be divided into three main steps, including encoding, entanglement creation and decoding.



Supplementary Figure 3: Quantum circuit to calculate $|x(t)\rangle$.

Encoding: This part aims to encode parameter information into the amplitude of the qubits and divide the calculation in two subspaces.

Firstly, we apply the operator G_1 to the first auxiliary register to encode the normalization parameter. We set

$$S_1 = \sum_{m=0}^{k} \frac{\|x(0)\|(\|A\|t)^m(\sum_{l=1}^{L}\lambda_l)^m}{m!}, \tag{18}$$

$$S_2 = \sum_{n=1}^{k} \frac{\|b\|(\|A\|t)^{n-1}t(\sum_{l=1}^{L}\lambda_l)^{n-1}}{n!}. \tag{19}$$

Then, define

$$\alpha \triangleq \frac{S_1}{\sqrt{S_1^2+S_2^2}}, \beta \triangleq \frac{S_2}{\sqrt{S_1^2+S_2^2}}, \tag{20}$$

and thus G_1 can be represented as

$$G_1 = \begin{bmatrix} \alpha & \beta \\ \beta & -\alpha \end{bmatrix}. \tag{21}$$

Afterwards, using operations G_x and G_b controlled by the first auxiliary qubit, we can encode

$|x(0)\rangle$ and $|b\rangle$ into the working qubits to form $\alpha|0\rangle|x(0)\rangle$ and $\beta|1\rangle|b\rangle$, dividing the calculation in two subspaces naturally.

Secondly, we construct the auxiliary system. Apply controlled-unitary gates G_{2x} and G_{2b} of size $2^k \times 2^k$ to the second set of auxiliary qubits. If we define

$$g_{2x}^{(m,1)} \triangleq \sqrt{\|x(0)\| \frac{(\|A\|t)^j}{j!}} \tag{22}$$

when $m = 2^k - 2^{k-j} + 1$, otherwise it is 0; and

$$g_{2b}^{(m,1)} \triangleq \sqrt{\|b\| \frac{(\|A\|t)^{j-1} t}{j!}} \tag{23}$$

when $m = 2^k - 2^{k-j} + 1$, otherwise it is 0, then the 1^{st} column's m^{th} elements of these gates are

$$G_{2x}^{(m,1)} = \frac{g_{2x}^{(m,1)}}{\|g_{2x}^{(:,1)|}\|}, \; G_{2b}^{(m,1)} = \frac{g_{2b}^{(m,1)}}{\|g_{2b}^{(:,1)|}\|}, \tag{24}$$

where $g_{2x}^{(:,1)}$ and $g_{2b}^{(m,1)}$ represent the 1^{st} column of G_{2x} and G_{2b}. Additionally, apply a unitary operator G_3 of dimension $L \times L$ to each L-level quantum system in the third set of auxiliary registers. If we define $g_3^{(:,1)}$ to be the 1^{st} column of G_3 and $g_3^{(l,1)} \triangleq \sqrt{\lambda_l}$, then $G_3^{(l,1)}$ can be expressed as

$$G_3^{(l,1)} = \frac{g_3^{(l,1)}}{\|g_3^{(:,1)}\|}. \tag{25}$$

The register state is now transformed to

$$\begin{aligned} &\alpha|0\rangle \sum_m G_{2x}^{(m+1,1)} |m\rangle \left(\sum_{l=1}^{L} G_3^{(l,1)} |l-1\rangle_L \right)^{\otimes k} |x(0)\rangle \\ &+\beta|1\rangle \sum_m G_{2b}^{(m+1,1)} |m\rangle \left(\sum_{l=1}^{L} G_3^{(l,1)} |l-1\rangle_L \right)^{\otimes k} |b\rangle. \end{aligned} \tag{26}$$

where the summation index is $m = 2^k - 2^{k-j}$. For example, when $k = 3$, the corresponding quantum state $|m\rangle$ include $|000\rangle$, $|100\rangle$, $|110\rangle$ and $|111\rangle$.

Entanglement Creation: To selectively achieve different orders of the decomposition of $\hat{A}$, a series of joint controlled-U_l gates controlled by the second and third sets of auxiliary registers, as shown in SFig. 3, will be used.

The third set of auxiliary registers is responsible for controlling the matching of the parameter

λ_l and the unitary operator U_l. Specifically, the quantum state $|l-1\rangle$ controls the function of U_l. The second set of auxiliary registers is responsible for controlling the order of operator execution and the matching of parameters. For example, when $m = 3$, the state $|110\rangle$ ensures that $\hat{A}$ is executed for two times which corresponds to j. The state of the system here will be transformed to

$$\begin{aligned}\alpha|0\rangle \sum_m G_{2x}^{(m+1,1)}|m\rangle \left(\sum_{l=1}^{L} G_3^{(l,1)}|l-1\rangle_L U_l\right)^{\otimes j} |x(0)\rangle \\ +\beta|1\rangle \sum_m G_{2b}^{(m+1,1)}|m\rangle \left(\sum_{l=1}^{L} G_3^{(l,1)}|l-1\rangle_L U_l\right)^{\otimes j} |b\rangle,\end{aligned} \tag{27}$$

where the summation index is $m = 2^k - 2^{k-j}$.

Decoding: This step will finally implement the parameters of the desired result and return the auxiliary register to the ground state. Then $|x(t)\rangle$ can be obtained in the working register when the auxiliary registers are measured in the ground state.

To decode, we only need to apply reverse operations to the auxiliary registers. That is, apply $G_3^\dagger$ to the first auxiliary register, apply controlled-unitary gates $G_{2x}^\dagger$ and $G_{2b}^\dagger$ to the second group of registers, and apply unitary gate $G_3^\dagger$ to the third group of registers. Finally, when measuring the auxiliary registers to be $|0\rangle|0\rangle^k|0\rangle_L^k$, the state of the working registers will be $|x(t)\rangle$ and $S = S_1 + S_2$.

With the aid of the robust oblivious amplitude amplification algorithm,[24] it is possible to measure the ancillary register in the ground state with near-100% probability. In such cases, the state $|x(t)\rangle$ obtained in the working register at the end of a computation cycle can serve as the initial state $|x(0)\rangle$ for the next cycle, thus eliminating the need for a full quantum state measurement and reducing cost overhead of re-encoding. To enable continuous iterative computation, it is also necessary to estimate the probability p_{sub} of the subspace where the ancillary registers are all in the ground state. This can be achieved via measurement statistics or amplitude estimation algorithms. The norm of $x(t)$ can then be calculated as

$$\|x(t)\| = S \cdot p_{\text{sub}}, \tag{28}$$

and this value will then be used in the subsequent computation cycle. Furthermore, the estimated value of p_{sub} can guide the number of amplitude amplification steps, ensuring that the desired state $|x(t)\rangle$ is obtained with high success probability efficiently.[25]

Now, we summarize the core process of our proposed QuanDT for Buck converter with ZIP load. A detailed instruction is provided in SAlg. 1.

Supplementary Algorithm 1 Core Process of QuanDT for DC-DC Buck Converter

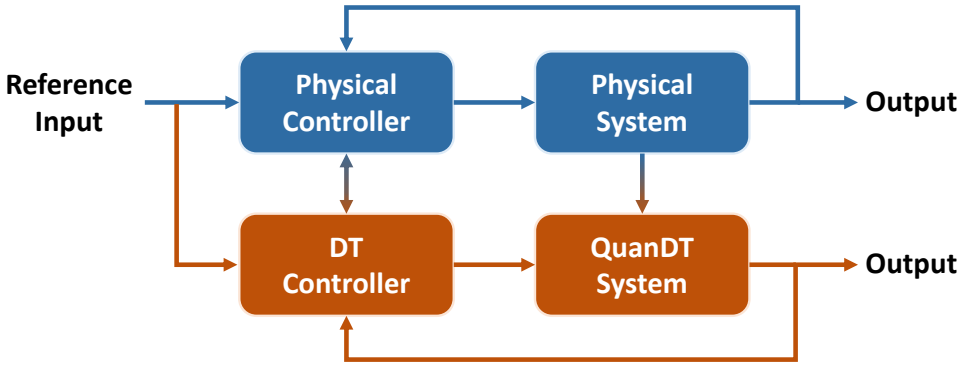
Input: the physical controller output $\mu^{(t)}$ and the measured load power $P_m^{(t)}$ at time step t
Initialize: the state vector $x(0)$, the system matrix A, and the input matrix B
repeat
    Update A and the system input u using $P_m^{(t)}$
    Compute $|x(t)\rangle$ by utilizing the quantum circuit as shown in SFig. 3 including
        Encoding parameters into the amplitude of the qubits according to Eq. (18)-(25)
        Creating entanglement to selectively achieve different orders of system matrix $\hat{A}$
        Decoding the output and get the result of $|x(t)\rangle$ and $\|x(t)\|$ according to Eq. (28)
    Set $|x(0)\rangle := |x(t)\rangle$ and $t := t + \Delta t$
until the QuanDT process is terminated

A.4 Digital Twin Controller Model

Reliability is one of the primary objectives in industrial control systems, aiming to ensure the safe and stable operation of controlled industrial processes. To address potential risks such as hardware failures, communication interruptions, or software anomalies, a common strategy for enhancing the reliability of critical functions under unpredictable conditions is the use of redundancy—that is, replicating key functionalities and distributing them across multiple components.[26] Typical redundancy architectures include hot standby, cold standby, and parallel redundancy configurations, all of which can swiftly take over control tasks when the primary system encounters failures, thereby avoiding production interruptions.[27] DT provides a convenient approach to implementing redundant control functionalities, as it serves as a virtual replica of the physical system. In the event of failures in the physical system, corresponding components in the DT can act as replacements.[19] It is important to note, however, that the effectiveness of fault-tolerant control depends heavily on the modeling accuracy and tracking capability of the DT system.[28] QuanDT, empowered by quantum computing, can facilitate more precise modeling and tracking of larger-scale systems.

The operation process of the DT controller designed in this work is illustrated in the SFig. 4. The DT system operates in parallel with the physical system. Under normal conditions, the physical controller computes control signals based on the reference input and feedback measured output, and the physical system responds accordingly under its control. Simultaneously, the DT controller receives the control signal from the physical controller and inputs it into the QuanDT system. The QuanDT system updates its internal model based on information from the physical system and computes the corresponding system response. Meanwhile, the DT controller synchronizes the

internal accumulated integral terms of the physical controller and computes the next-step control command using the reference value and the QuanDT's output. This command is forwarded to the physical controller for backup, in case of potential controller failure.



Supplementary Figure 4: Digital twin controller working data flow.

When some fault occurs, the DT controller ceases synchronization with the physical controller and takes over its role. Its control command will be sent to the physical system. Notably, during this stage, the DT controller continues to rely on the output of the QuanDT system to compute its control signals rather than the physical system. Therefore, it is crucial that the DT system maintains high-accuracy, real-time tracking of the physical system in order to ensure overall system stability.

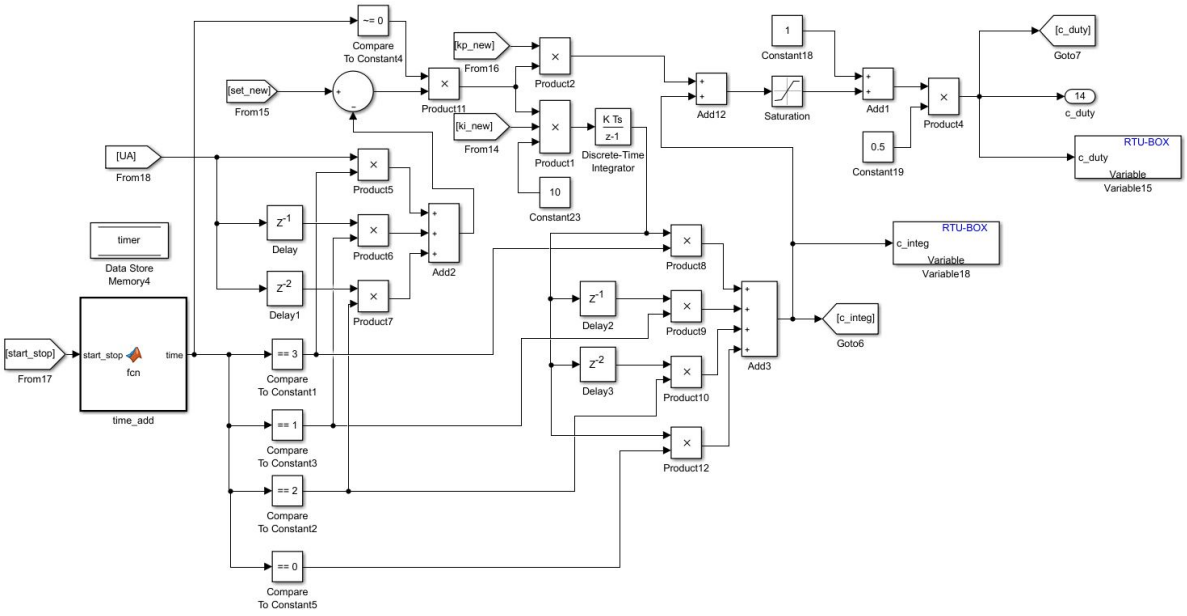
B Detailed Experiment Settings

B.1 Physical Experiment Settings

The DC voltage source employed to simulate renewable energy is a high-power, high-performance programmable power supply, model DCPS2420, manufactured by LANYI. The Buck converter utilizes an IGBT-based half-bridge power electronics module, model RTM-PEH8025IF from Rtunit, which incorporates an integrated current measurement module for monitoring the inductor current i_L. The LC filter module is also supplied by Rtunit. The ZIP load is simulated using three high-speed, high-precision programmable DC electronic loads, model IT8813B, provided by ITECH. Among these, the power load is connected to the host computer via USB, enabling power fluctuation control through ITECH's IT9000 software. These electrical devices are interconnected as illustrated in SFig. 1. During the experiments, the power measurement P_m of the power load is obtained by multiplying the capacitor voltage u_C, measured by a voltmeter (model RTU-REM-VS8003, Rtunit), and the current flowing through the power load, measured by a ammeter (model RTU-PEM-CS503, Rtunit). The actual parameters of the electrical circuits used in experiments are listed in STab. 1.

Parameter	Value	Unit
V_{in}	68	V
L	0.0034	H
C	0.00113	F
R_1	20	Ω
I_{l1}	2	A
P	$25 + 10\sin(\pi t)$ *	W

Supplementary Table 1: Parameters of the electrical circuits in physical experiments. * Due to the performance limitation of the electronic load, the power load cannot change its power continuously; instead, it adjusts the power in discrete steps every 0.1 second according to the Sine function value.



Supplementary Figure 5: Implementation of physical controller in Rtunit Studio.

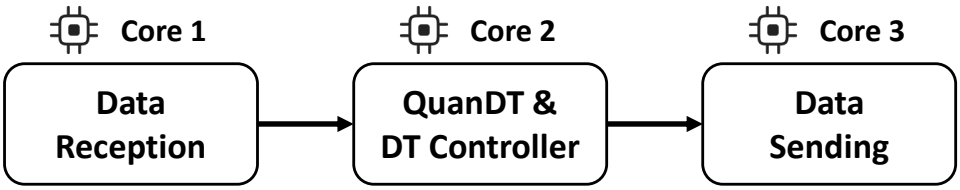
The physical controller employed is the RTU-BOX204 real-time digital controller, also from Rtunit. The RTU-BOX204 is equipped with DAC and ADC modules for receiving sensor data and outputting PWM signals. Control programs can be designed as shown in SFig. 5 using the accompanying host software Rtunit Studio and the Simulink model library Rtunit Toolbox. The controller code is then generated using RTU-Coder, which can be compiled and downloaded to the controller. In the experiment, the controller described in Section A.2 was implemented with an operating cycle set to 3 ms. The parameters of the physical controller are presented in STab 2.

Parameter	K_p	K_i
Value	0.001	10

Supplementary Table 2: Parameters of the physical controller in physical experiments.

In the physical experiment, both the QuanDT and the DT controller are implemented on a Raspberry Pi 5 equipped with 8 GB of memory. This Raspberry Pi was configured with a 512 GB KINGHICO SSD via a Waveshare PCIe to M.2 adapter board and runs the 64-bit Raspberry Pi OS Lite operating system. Additionally, the Raspberry Pi is equipped with a Waveshare dual-channel isolated CAN bus expansion board to facilitate CAN bus communication with the physical system.

To ensure the real-time performance of the QuanDT, the functionalities on the Raspberry Pi are implemented using a Julia-based program. Julia is a high-level dynamic programming language well-suited for high-performance scientific computing. In the implementation, the `Distributed` library is used to divide the program into three serial subprocesses according to the functional blocks shown in the SFig. 6, with each subprocess running on a separate core of the Raspberry Pi. The three subprocesses communicate with each other through channel-based inter-process communication, enabling efficient data exchange during runtime.



Supplementary Figure 6: The program flow of the code implemented using Julia.

The *Data Reception* subprocess is responsible for receiving CAN protocol data from the RTU-BOX204 controller. This includes the actual control outputs and power measurements of the load required for the self-updating and evolution of the QuanDT, as well as the accumulated integral value within the physical controller, which is used to synchronize the DT controller. In addition, some status indicators are received to adjust the operation of the DT system. The *Data Sending* subprocess is primarily responsible for transmitting control information generated by the DT controller to the RTU-BOX204 in the form of CAN protocol data, guiding the operation of the physical system in the event of abnormal conditions. The *QuanDT and the DT controller* operate collaboratively, and the detailed algorithm in use is provided in SAlg. 2.

In the real-time tracking experiments of Scenario 1, when voltage fluctuations are relatively small, V_0 can be set to the reference value. In cases where the voltage fluctuates significantly—for

example, when the reference voltage changes abruptly—V_0 can be set to the value computed by the DT at the previous time step. That is

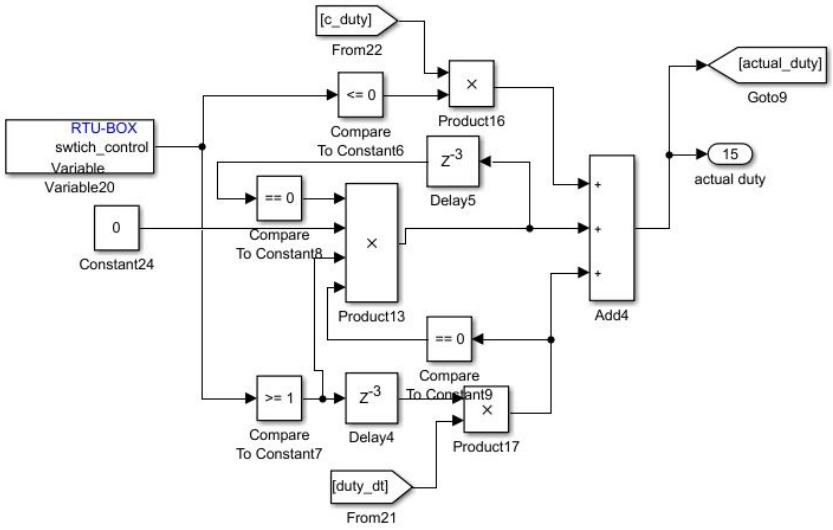
$$V_0 = u_C(t). \tag{29}$$

It is worth noting that, from a practical perspective, this requires additional measurement steps to extract information stored in quantum states. Since the measurement of some single state is acceptable for large-scale systems, and given that many other methods exist for model updates, this method variant is implemented for the purpose of methodological continuity.

Supplementary Algorithm 2 QuanDT & DT Controller in Use

Input: the output $\mu^{(t)}$ and the integral value $Integ^{(t)}$ of the physical controller, and the measured load power $P_m^{(t)}$ at time step t
Ouput: the DT controller output $\mu_{DT}^{(t+\Delta t)}$
Initialize: the state vector $x(0) = [i_L(0), u_C(0)]$, the system matrix A, the input matrix B, the output μ_{DT} and the integral value $Integ^{(t)}$ of the DT controller
repeat
  if under normal conditions **then**
    Align the DT controller with the physical controller using $\mu^{(t)}$ and $Integ^{(t)}$
  else if under abnormal conditions **then**
    Set $\mu^{(t)} := \mu_{DT}^{(t)}$
  end if
  Update A and the system input u using $P_m^{(t)}$ according to Eq. (2)-(5) and Eq. (10)
  Compute $|x(t)\rangle$ as shown in SFig. 3 based on `Yao` and `QuDiffEq` library with $k = 3$
  Calculate the control output $\mu_{DT}^{(t+\Delta t)}$ of the DT controller
  Set $|x(0)\rangle := |x(t)\rangle$ and $t := t + \Delta t$
until the QuanDT process is terminated

In the redundant control experiment under Scenario 2, a fault is introduced by abruptly reducing the control signal output from the physical controller as shown in SFig. 7. Simultaneously, the fault detector identifies the anomaly and switches the controller signal. Then, the physical system sends status information to the DT system. As shown in SAlg. 2, when an abnormal condition is detected, the DT controller will be activated. Supported by the QuanDT, the DT controller will take over and maintain stable system operation in place of the physical controller.



Supplementary Figure 7: Implementation of fault generation and control switching in Rtunit Studio.

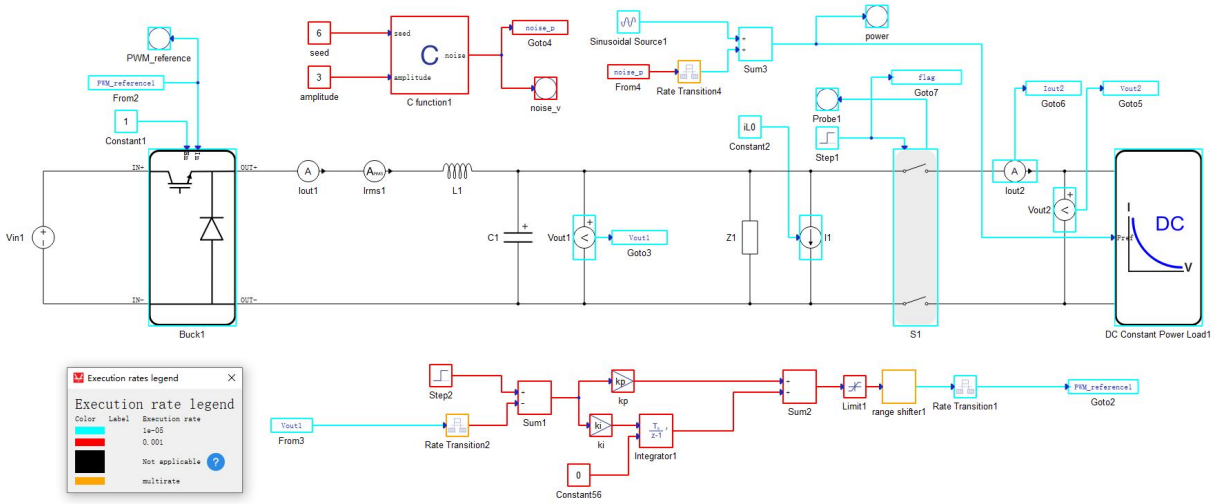
B.2 HIL Experiment Settings

The physical electrical devices and the controller are implemented using the Typhoon hardware-in-the-loop (HIL) 602+ simulator. It enables high-fidelity simulation of power electronic systems with ultra-high temporal resolution. As illustrated in the SFig. 8, both electrical and control scenarios are designed using the Schematic Editor provided in the Typhoon HIL Control Center.

The electrical devices are connected following the same configuration as shown in SFig. 1. In the simulation, the constant power load is modeled as a controlled current source with an added snubber circuit. To emulate realistic disturbances, the power reference of the constant power load is artificially perturbed using Gaussian-distributed random noise generated via the Box–Muller transform applied to a uniform distribution.

The physical controller follows the same implementation as described in Section A.2, with a control cycle of 1 ms. To ensure synchronization between the measurement device and the controller, the reference power of the constant power load is also updated at a 1 ms interval. The electrical devices, on the other hand, are simulated with the highest available resolution of 0.01 ms. The actual parameters of the electrical circuits and controllers used in the experiment are listed in STab. 3 and 4.

In the HIL experiments, both the QuanDT and the DT controller are implemented on a workstation equipped with 32 GB of RAM. The workstation features an Intel® Core™ i9-10850K CPU @ 3.60 GHz and runs Windows 10 22H2 Professional. The implementation of the QuanDT and the DT controller follows the same approach as previously described. The main difference is that communication between the physical system and the DT system is carried out via UDP.



Supplementary Figure 8: Implementation of the electrical devices and the controller in the Schematic Editor of Typhoon HIL Control Center.

Parameter	**Value**	**Unit**
V_{in}	68	V
L	0.003	H
C	0.001	F
R_1	20	Ω
I_{l1}	2	A
P	$25 + 10\sin(\pi t)$ *	W
Power Noise	$\mathcal{N}(0,\ 3^2)$ *	W
Snubber Resistance	10^5	Ω

Supplementary Table 3: Parameters of the electrical circuits in HIL experiments. * The power load adjusts the power in discrete steps every 1 ms.

Parameter	K_p	K_i
Value	0.001	0.3

Supplementary Table 4: Parameters of the physical controller in HIL experiments.

C QML Application in Detail

C.1 Dataset construction and processing

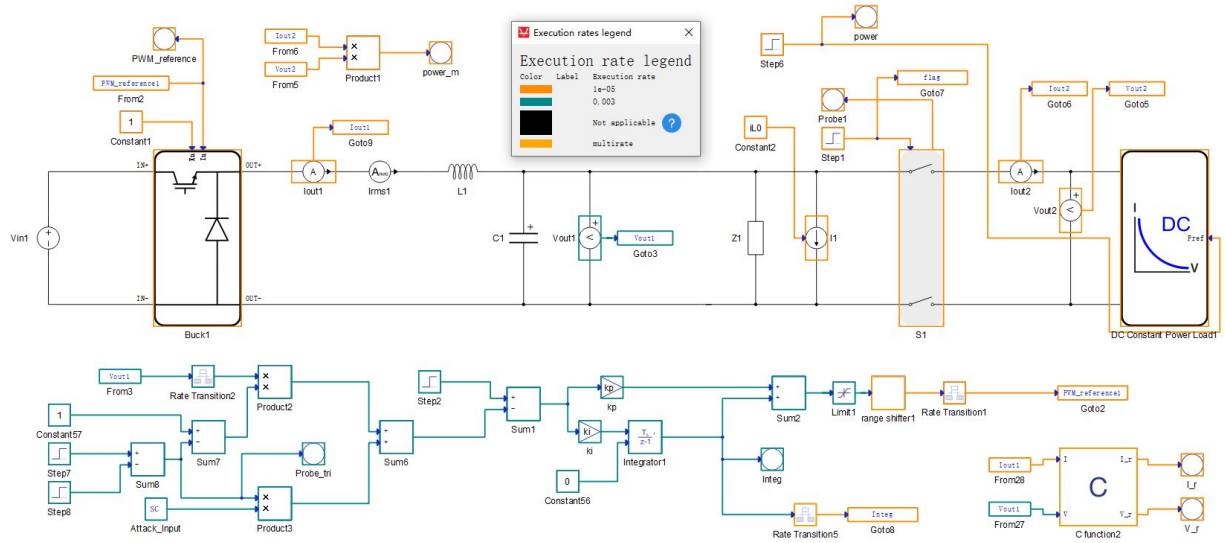
In modern power systems, intelligent sensors are widely deployed across all stages of generation, transmission, distribution, and consumption to enable real-time acquisition of critical state variables, thereby supporting system monitoring, control, and protection. However, sensors themselves are vulnerable to physical damage, aging, and communication interference, which can lead to faults or abnormal data.[29] Furthermore, as power systems become increasingly integrated with information and communication technologies, cyberattacks targeting sensor data are becoming more prevalent. These attacks may mislead control systems into making erroneous decisions, thereby jeopardizing the safety and stability of the power grid.[30] Therefore, timely detection of anomalies caused by sensor faults and malicious attacks is essential for ensuring the reliable operation.

At present, researchers have proposed various methods for detecting sensor faults and identifying potential malicious attacks. These approaches can be broadly categorized into model-based methods and data-driven methods. Model-based methods rely on the physical models of power systems, with commonly used techniques including state estimation and Kalman filtering.[31] In contrast, data-driven methods leverage machine learning or deep learning models trained on historical data to capture the statistical patterns of normal system operations and identify anomalies that deviate from these patterns.[32] In data-driven approaches, high-quality data form the foundation for effective model training and performance improvement.[33] Sufficient, diverse, and representative data help capture both normal and abnormal system behaviors, thereby enhancing the sensitivity and robustness of anomaly detection algorithms to sensor faults and cyberattacks.

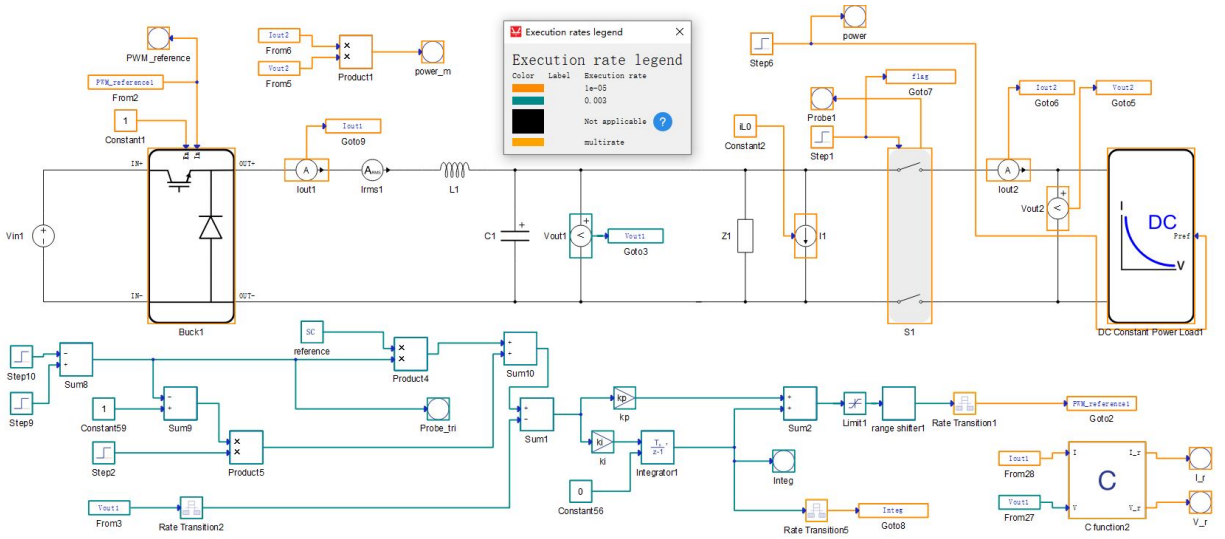
DTs, by constructing virtual models that closely mirror physical systems, can generate rich and accurate datasets under various operating conditions. This capability effectively alleviates challenges such as imbalanced data distributions and the scarcity of anomalous samples in real-world scenarios, thereby supporting the development of balanced, high-quality datasets.[34] Moreover, modern power systems often exhibit significant nonlinearity and time-varying characteristics—such as those found in ZIP loads—leading to diverse dynamic responses under different operating conditions.[35] DTs can flexibly adjust model parameters in accordance with specific scenarios to generate representative dynamic data, which enhances the ability of data-driven methods to identify and adapt to abnormal behaviors.[36]

In the dataset construction process, the Typhoon HIL 602+ simulator is employed to simulate data generated by the QuanDT. Different power load setpoints are configured to emulate the QuanDT under various operating conditions. Two scenarios are established to collect both anomalous and normal data: one involving sensor attacks, as illustrated in SFig. 9, and the other involving normal reference value changes, as shown in SFig. 10. The electrical devices are simulated with a cycle of

0.01 ms and the controller is with a control cycle of 3 ms.



Supplementary Figure 9: Implementation for sensor attack scenario dataset.



Supplementary Figure 10: Implementation for reference change scenario dataset.

In combination with the TyphoonTest IDE software, attack values and reference values are automatically and randomly configured, and the corresponding data are collected accordingly. The final dataset consists of 2,000 samples for each of the following scenarios: abnormal conditions under low power load, normal conditions under low power load, abnormal conditions under high power load, and normal conditions under high power load. Each sample contains five time steps of system current and voltage measurements, recorded at 9 ms intervals. To simulate the output format of the QuanDT, all current and voltage values are normalized such that the system state at

each time step can be encoded into a single qubit. The relevant parameters involved in the data collection process are summarized in STab. 5 and 6.

Parameter	**Value**	**Unit**
V_{in}	68	V
L	0.0005	H
C	0.001	F
R_1	5	Ω
I_{l1}	30	A
P in High Power State	250	W
P in Low Power State	10	W
Snubber Resistance	10^5	Ω
Attack	$\mathcal{U}(45, 47) \cup (49, 51)$	V
Reference Value	$\mathcal{U}(47.5, 48.5)$	V

Supplementary Table 5: Parameters of the electrical circuits for data collection.

Parameter	K_p	K_i
Value	0.001	1

Supplementary Table 6: Parameters of the controller in SFig. 9 and 10.

C.2 Quantum and Classical Model

The quantum and classical models differ in both their inputs and network architectures. For the considered QuanDT for DC-DC Buck converter, the system output at time step k is a quantum state in a single qubit, that is

$$\left|x_q^{(k)}\right\rangle = \frac{x^{(k)}}{\|x^{(k)}\|} = \frac{1}{\sqrt{i_L^2 + u_C^2}} \begin{bmatrix} i_L \\ u_C \end{bmatrix}, \tag{30}$$

where i_L and u_C denote the inductor current and capacitor voltage, respectively. The corresponding classical representation can be obtained by measurement and is expressed as

$$x_c^{(k)} = \frac{1}{\sqrt{i_L^2 + u_C^2}} \begin{bmatrix} i_L \\ u_C \end{bmatrix}. \tag{31}$$

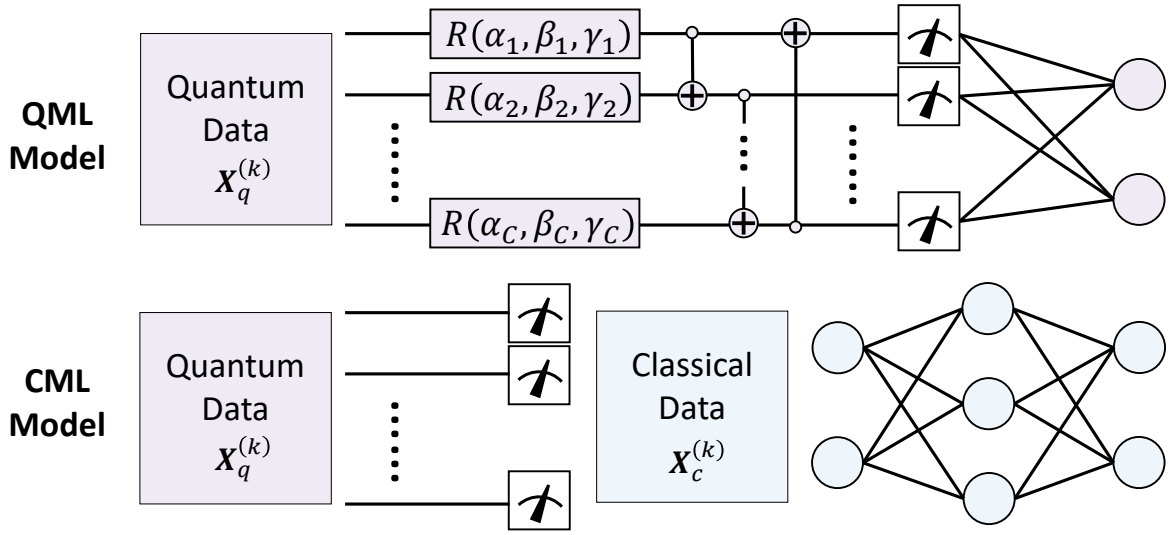
A sequence of C past data points from time step k forms

$$X_q^{(k)} = \left[\left| x_q^{(k)} \right\rangle, \left| x_q^{(k-1)} \right\rangle, \ldots, \left| x_q^{(k-C+1)} \right\rangle \right], \tag{32}$$

and its classical counterpart is

$$X_c^{(k)} = \left[x_c^{(k)}, x_c^{(k-1)}, \ldots, x_c^{(k-C+1)} \right]. \tag{33}$$

The input to the QML model is $X_q^{(k)}$, while $X_c^{(k)}$ serves as the input to the classical machine learning model. Notably, the quantum representation leads to exponential savings in the required number of bits compared to classical representations. Moreover, directly processing quantum data avoids the overhead associated with measurement and classical conversion.



Supplementary Figure 11: A model architecture diagram of quantum machine learning (QML) and classical machine learning (CML).

The QML model adopts a classical-quantum hybrid architecture as illustrated in the SFig. 11. The input $X_q^{(k)}$ contains data from C time steps, each corresponding to a single qubit. The input is first processed by a strongly entangling layer U_{strong}, which consists of a parameterized arbitrary

rotation layer,

$$U_{\text{rot}}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) = \bigotimes_{j=1}^{C} R_Z(\alpha_j) R_Y(\beta_j) R_Z(\gamma_j), \tag{34}$$

and a ring entangling layer defined as

$$U_{\text{ring}} = \text{CNOT}_{C-1,0} \, \text{CNOT}_{C-2,C-1} \cdots \text{CNOT}_{0,1}. \tag{35}$$

The overall strongly entangling layer is then given by

$$U_{\text{strong}}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) = U_{\text{ring}} \cdot U_{\text{rot}}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}), \tag{36}$$

and the resulting quantum state is

$$|\phi\rangle = U_{\text{strong}}(\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) \left| X_q^{(k)} \right\rangle. \tag{37}$$

Let the expectation value of the local Pauli-Z operator on the j-th qubit be

$$\langle Z_j \rangle = \left\langle \phi \middle| Z_j \middle| \phi \right\rangle, \tag{38}$$

where Z_j denotes the Pauli-Z operator acting on the j-th qubit. By measuring the local Pauli-Z expectation value on each qubit, we obtain the vector

$$\boldsymbol{z} = [\langle Z_0 \rangle, \langle Z_1 \rangle, \ldots, \langle Z_{C-1} \rangle]. \tag{39}$$

It is worth noting that this measurement is efficient since only local observables are involved.

Finally, the resulting vector $\boldsymbol{z}$ is passed through a linear layer,

$$L(\boldsymbol{z}) = W\boldsymbol{z} + \boldsymbol{b}, \tag{40}$$

yielding the logits for the current situation, where the index of the maximum value is taken as the model prediction.

For the classical machine learning model, a multilayer perceptron (MLP) architecture is used as illustrated in the SFig. 11. The input $X_c^{(k)}$, containing data from C time steps, corresponds to an input dimension of $2C$. The input is first processed by a hidden layer,

$$H\left(X_c^{(k)}\right) = \sigma\left(L\left(X_c^{(k)}\right)\right), \tag{41}$$

where σ is the activation function, taken to be ReLU in our experiments. The output is then passed

through another linear layer,

$$L\left(H\left(X_c^{(k)}\right)\right), \tag{42}$$

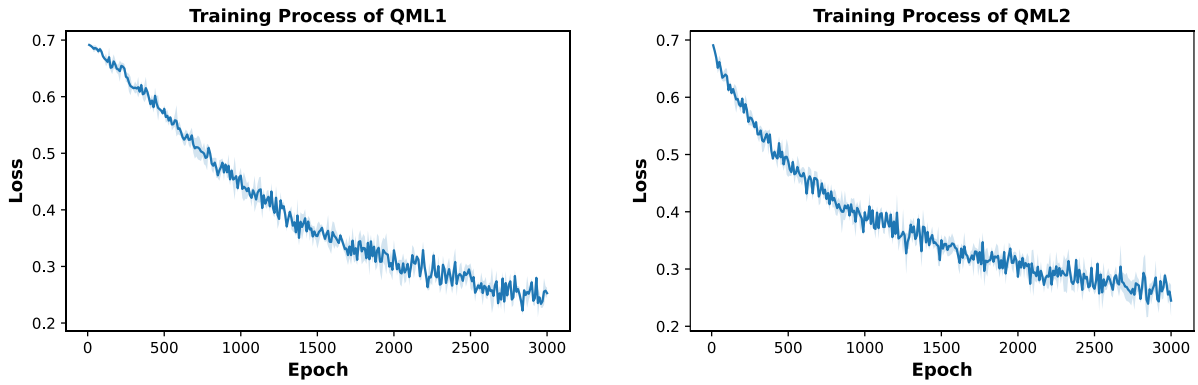
producing the logits for the current situation, with the predicted class corresponding to the index of the maximum value. To ensure fair comparison, we adjust the dimension of the hidden layer such that the classical and quantum models have comparable parameter space dimensionality.

Notably, in the general case, if the QuanDT produces p qubits as output, then the input dimension of the quantum model is $C \cdot p$, while that of the classical model is $C \cdot 2^p$. Thus, the use of QML in conjunction with a QuanDT can significantly reduce the model input dimensionality.

C.3 Experiment Settings

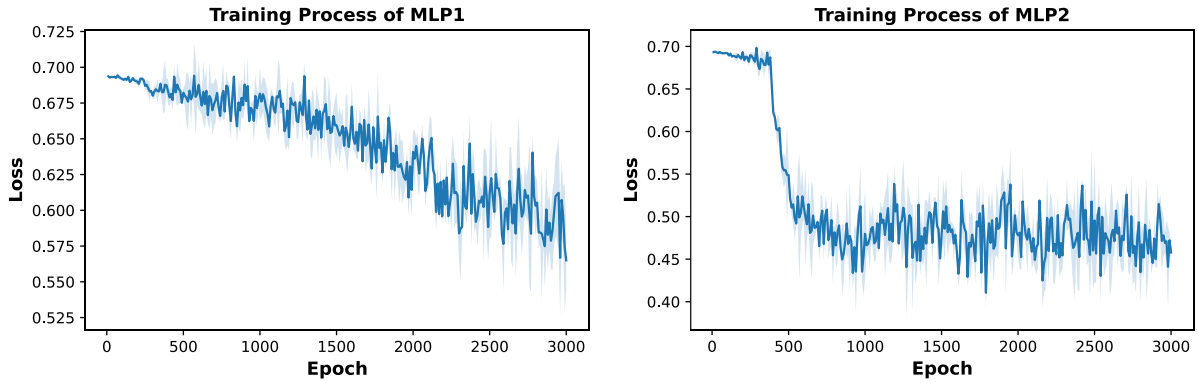
In the experiments, both QML and MLP models are trained separately on the low-power-load and high-power-load datasets. Each dataset is split into a training set and a test set with an 4:1 ratio for model training and evaluation. The models are implemented using the Pennylane and PyTorch frameworks.

For the QML model, the input are 5 qubits. A strongly entangling layer, involving $5 \times 3 = 15$ trainable parameters, is applied. This is followed by a linear layer with input dimension 5 and output dimension 2, consisting of $5 \times 2 + 2 = 12$ parameters. In total, the QML model has 27 trainable parameters. The MLP model take as input a 10-dimensional vector, which passes through a hidden layer of dimension 2, involving $10 \times 2 + 2 = 22$ parameters. The output linear layer has 2 outputs, resulting in $2 \times 2 + 2 = 6$ additional parameters. Thus, the total number of trainable parameters in the MLP model is 28.



Supplementary Figure 12: QML training loss as a function of training epochs.

The training is based on Pennylane's `default.qubit` backend to simulate quantum computing. Cross-entropy loss is used as the objective function, and parameter updates are performed via backpropagation using the Adam optimizer with an initial learning rate of 0.001 for 3000 epochs.



Supplementary Figure 13: MLP training loss as a function of training epochs.

During each epoch, the dataset is reshuffled and divided into batches of size 100 for training. The same randomly initialized model parameters are used across different training runs, while distinct random seeds are used to shuffle the datasets, resulting in different training trajectories. The training curves of both QML and MLP models under the two load conditions are shown in SFig. 12 and SFig. 13, respectively. For the figures, the dark blue line denotes the mean training loss as a function of the number of training epochs, and the light blue shaded region indicates one standard deviation over multiple runs with different random seeds.

For evaluation, the performance of the QML and MLP models is compared horizontally under both operating conditions. Furthermore, cross-scenario evaluations are conducted: the model trained on the high-power-load dataset is tested on the low-power-load dataset, and the model trained on the low-power-load dataset is tested on the high-power-load dataset. These evaluations aim to demonstrate the necessity of real-time DT tracking with the physical system for reliable machine learning deployment. In addition, the results highlight the potential of QuanDT in enhancing model adaptability and robustness for industrial applications.

Supplementary References

[1] Dai, X., Liu, G.-P., Deng, Q. & Zeng, W. Global optimal cooperative control of multiple dc–dc converter systems for dynamic consensus. *IEEE Transactions on Power Electronics* **36**, 14340–14352 (2021).

[2] Lei, Z., Zhou, H., Dai, X., Hu, W. & Liu, G.-P. Digital twin based monitoring and control for dc-dc converters. *Nature Communications* **14**, 5604 (2023).

[3] Gheisarnejad, M., Farsizadeh, H. & Khooban, M. H. A novel nonlinear deep reinforcement learning controller for dc–dc power buck converters. *IEEE Transactions on Industrial Electronics* **68**, 6849–6858 (2020).

[4] Solsona, J. A., Jorge, S. G. & Busada, C. A. Nonlinear control of a buck converter which feeds a constant power load. *IEEE Transactions on Power Electronics* **30**, 7193–7201 (2015).

[5] Tucci, M., Riverso, S., Vasquez, J. C., Guerrero, J. M. & Ferrari-Trecate, G. A decentralized scalable approach to voltage control of dc islanded microgrids. *IEEE Transactions on Control Systems Technology* **24**, 1965–1979 (2016).

[6] Arif, A. *et al.* Load modeling—a review. *IEEE Transactions on Smart Grid* **9**, 5986–5999 (2017).

[7] Bokhari, A. *et al.* Experimental determination of the zip coefficients for modern residential, commercial, and industrial loads. *IEEE Transactions on Power Delivery* **29**, 1372–1381 (2013).

[8] Tan, R. H. & Hoo, L. Y. Dc-dc converter modeling and simulation using state space approach. In *2015 IEEE Conference on Energy Conversion (CENCON)*, 42–47 (IEEE, 2015).

[9] Martinez-Salamero, L., Cid-Pastor, A., Giral, R., Calvente, J. & Utkin, V. Why is sliding mode control methodology needed for power converters? In *Proceedings of 14th International Power Electronics and Motion Control Conference EPE-PEMC 2010*, S9–25 (IEEE, 2010).

[10] Han, R., Tucci, M., Martinelli, A., Guerrero, J. M. & Ferrari-Trecate, G. Stability analysis of primary plug-and-play and secondary leader-based controllers for dc microgrid clusters. *IEEE Transactions on Power Systems* **34**, 1780–1800 (2018).

[11] Alsmadi, Y. M., Utkin, V., Haj-ahmed, M. A. & Xu, L. Sliding mode control of power converters: Dc/dc converters. *International Journal of Control* **91**, 2472–2493 (2018).

[12] Xu, Q. *et al.* Review on advanced control technologies for bidirectional dc/dc converters in dc microgrids. *IEEE Journal of Emerging and Selected Topics in Power Electronics* **9**, 1205–1221 (2020).

[13] Jadidi, S., Badihi, H. & Zhang, Y. Active fault-tolerant and attack-resilient control for a renewable microgrid against power-loss faults and data integrity attacks. *IEEE Transactions on Cybernetics* **54**, 2113–2128 (2023).

[14] Shu, X., Guo, Y., Yang, H., Wei, K. *et al.* Reliability study of motor controller in electric vehicle by the approach of fault tree analysis. *Engineering failure analysis* **121**, 105165 (2021).

[15] Liu, P. *et al.* Fuzzing proprietary protocols of programmable controllers to find vulnerabilities that affect physical control. *Journal of Systems Architecture* **127**, 102483 (2022).

[16] Vitturi, S., Zunino, C. & Sauter, T. Industrial communication systems and their future challenges: Next-generation ethernet, iiot, and 5g. *Proceedings of the IEEE* **107**, 944–961 (2019).

[17] Liu, M. *et al.* Enhancing cyber-resiliency of der-based smart grid: A survey. *IEEE Transactions on Smart Grid* **15**, 4998–5030 (2024).

[18] Liu, M. *et al.* Pddl: Proactive distributed detection and localization against stealthy deception attacks in dc microgrids. *IEEE Transactions on Smart Grid* **14**, 714–731 (2022).

[19] Lei, Z., Zhou, H., Dai, X., Hu, W. & Liu, G.-P. Digital twin based monitoring and control for dc-dc converters. *Nature Communications* **14**, 5604 (2023).

[20] AL-Nussairi, M. K., Bayindir, R., Padmanaban, S., Mihet-Popa, L. & Siano, P. Constant power loads (cpl) with microgrids: Problem definition, stability analysis and compensation techniques. *Energies* **10**, 1656 (2017).

[21] Singh, S., Gautam, A. R. & Fulwani, D. Constant power loads and their effects in dc distributed power systems: A review. *Renewable and Sustainable Energy Reviews* **72**, 407–421 (2017).

[22] Li, Y., Mughees, M., Chen, Y. & Li, Y. R. The unseen ai disruptions for power grids: Llm-induced transients. *arXiv preprint arXiv:2409.11416* (2024).

[23] Xin, T. *et al.* Quantum algorithm for solving linear differential equations: Theory and experiment. *Physical Review A* **101**, 032307 (2020).

[24] Berry, D. W., Childs, A. M., Cleve, R., Kothari, R. & Somma, R. D. Simulating hamiltonian dynamics with a truncated taylor series. *Physical review letters* **114**, 090502 (2015).

[25] Brassard, G., Hoyer, P., Mosca, M. & Tapp, A. Quantum amplitude amplification and estimation. *arXiv preprint quant-ph/0005055* (2000).

[26] Ma, R. *et al.* Stealthy attack against redundant controller architecture of industrial cyber-physical system. *IEEE Internet of Things Journal* **6**, 9783–9793 (2019).

[27] Leander, B., Johansson, B., Mubeen, S., Ashjaei, M. & Lindström, T. Redundancy link security analysis: An automation industry perspective. In *2024 IEEE 29th International Conference on Emerging Technologies and Factory Automation (ETFA)*, 1–8 (IEEE, 2024).

[28] Liu, G.-P. Control strategies for digital twin systems. *IEEE/CAA Journal of Automatica Sinica* **11**, 170–180 (2024).

[29] Chandra, M. S. S. & Mohapatro, S. Sensor fault tolerant control in modern and green power applications: A review. *IEEE Transactions on Instrumentation and Measurement* (2024).

[30] Duo, W., Zhou, M. & Abusorrah, A. A survey of cyber attacks on cyber physical systems: Recent advances and challenges. *IEEE/CAA Journal of Automatica Sinica* **9**, 784–800 (2022).

[31] Liu, M. *et al.* Converter-based moving target defense against deception attacks in dc microgrids. *IEEE Transactions on Smart Grid* **13**, 3984–3996 (2021).

[32] Peng, S., Liu, M., Chai, L. & Deng, R. Dst-gnn: A dynamic spatiotemporal graph neural network for cyberattack detection in grid-tied photovoltaic systems. *IEEE Transactions on Smart Grid* (2024).

[33] Goodfellow, I., Bengio, Y. & Courville, A. *Deep Learning* (MIT Press, 2016).

[34] Jain, P. *et al.* A digital twin approach for fault diagnosis in distributed photovoltaic systems. *IEEE Transactions on Power Electronics* **35**, 940–956 (2019).

[35] Henriquez-Auba, R., Lara, J. D. & Callaway, D. S. Small-signal stability impacts of load and network dynamics on grid-forming inverters. In *2024 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)*, 1–5 (IEEE, 2024).

[36] Chen, C., Fu, H., Zheng, Y., Tao, F. & Liu, Y. The advance of digital twin for predictive maintenance: The role and function of machine learning. *Journal of Manufacturing Systems* **71**, 581–594 (2023).