FPGA-Based Digital Twin Implementation for Power Converter System Monitoring

Justus Nwoke, Marco Milanesi, Jairo Viola and YangQuan Chen
Mechatronics, Embedded Systems and Automation (MESA) Lab
University of California, Merced
5200 North Lake Road, Merced, CA 94343, USA
{jnwoke, mmilanesi2, jviola, ychen53}@ucmerced.edu

***Abstract*—Implementing closed-loop control requires ensuring a robust system response against undesired disturbances and random fault events. To overcome these challenges, the controllers must be able to detect and adapt the process behavior against undesired events by adjusting its parameters accordingly. Developers have utilized the concept of a digital twin—a real-time representation of the physical process—to design and update the physical controller effectively. However, traditional digital twin implementations often involve significant data exchange between the digital system and the real asset through cloud platforms, leading to data latency and privacy issues. To mitigate these concerns, we propose an FPGA-based digital twin implementation where the information is directly sourced into the Digital Twin from the physical asset, which runs in parallel with the real system. This setup eliminates the need for big data transfers and cloud uploads, ensuring enhanced data privacy and facilitating a faster and more efficient digital twin implementation and update process. To demonstrate the capabilities of embedded Digital Twin, we present a case study involving monitoring a power converter system during a sensor fault scenario.**

***Index Terms*—Digital Twin, Flyback Converter, Smart Control Engineering, FPGA, Embedded Digital Twin, Industry 4.0.**

I. Introduction

The concept of real-time controller improvement has gained significant importance in achieving effective process in the loop control despite unpredictable disturbances. Until recently, classical control methods relied on approaches like model predictive control (MPC) [1], adaptive control methods [2], and offline optimization methods [3]. However, these methods were often either too mathematically complex or impractical for implementation on microcontrollers.

To address this challenge, developers devised a solution: creating a replica of the real system that operates in parallel with it, using the replica's performance to update the controller of the actual system. This idea, known as Digital Twin (DT), has been explored and implemented for various systems [4], [5]. While the digital twin concept has proven effective, it requires the exchange of large datasets between the replica and the physical assets, often requiring cloud storage and transmission, known as cloud-based implementation. Numerous cloud-based DT applications have been explored across industries, including industrial robots [6]–[8], mobile robots [9], line-following robots [10], machine integration for smart manufacturing [11], smart factory [12], and smart farming [13].

However, cloud-based implementations present challenges such as transmission protocol delays from the server, privacy issues, and potential data loss during transmission. To mitigate these issues, edge computing-based or cloud-edge-based DT implementations [14]–[16] have been explored. In standalone edge implementations, the digital twin directly receives data from physical assets for real-time updates, bypassing the intermediate cloud interface. This approach improves real-time digital twin operations and ensures data privacy, which is especially crucial in highly competitive industry sectors. Cloud-edge implementation combines the benefits of both edge and cloud implementations, including increased data storage memory. Moreover, direct implementations, like standalone edge implementation on devices such as Arduino, have been extended to include lower-level hardware implementation, such as Field Programmable Gate Array (FPGA) embedded implementation [17], [18]. The embedded-based implementation offers the advantage of high-speed execution as embedded systems like FPGA boards support concurrent execution of algorithms, unlike edge devices.

Therefore, this paper proposes an embedded digital twin implementation on an FPGA for monitoring a Power Converter system. The Digital Twin of the physical system is built using MATLAB/Simulink following the development framework [19]. The resulting DT model is translated into HDL code using the Matlab HDL coder toolbox for efficient voltage regulation of a Flyback Converter system. This digital twin implementation acts as a reference for the performance assessment of the physical asset, enabling event awareness capabilities. The main contribution of this paper lies in the hardware-level implementation of the digital twin to monitor the status of a Flyback Converter using an FPGA, with real-time data exchanged directly from physical assets, eliminating the need for an intermediary cloud exchange that introduces latency and data privacy issues. This work advances the framework of smart control engineering, equipping controllers with information for smarter decision-making. The paper is organized as follows: Section II introduces the digital twin

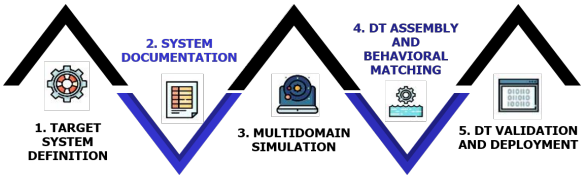

Figure 1. Digital Twin Five-step development framework [19]

framework. Section III presents the case study. Section IV describes the DT implementation and its integration into the FPGA. Section V presents the DT results and discussion. Finally, the conclusion and future works are provided in Section VI.

II. Digital Twin Framework

This paper utilizes the development framework proposed by [19] to construct the Digital Twin (DT) case study. This framework comprises five distinct steps: target system definition, system documentation, multidomain simulation, assembly and behavioral matching, and validation and deployment, as depicted in Fig. 1. In the initial step, the current status of the physical system to be replicated via the Digital Twin is determined, with two possible scenarios. The first scenario, known as Conceptual design, involves employing the DT for the initial designing task when a physical prototype is not available. The second scenario involves an operating physical system, where the DT serves as a supporting tool to enhance system operation.

In the second step, all available information about the system is collected to create the most accurate representation, including details about the control algorithms employed, data sheets of sensors and actuators, troubleshooting and problem records, cumulative experience of system engineers and operators, and the system's data streams. The third step involves employing a set of simulation models to represent the behavior of the real system. The simulation domains are defined based on the system's physical and constitutive laws, and appropriate computational tools are selected for multiphysics simulation.

Once the simulation models are completed, the fourth step, known as behavioral matching, is performed. This step involves determining the unknown parameters of the system using real data collected from the system at different operating points. Optimization fitting techniques, such as nonlinear least squares, are used to make the Digital Twin's simulation behavior as close as possible to that of the real asset.Finally, after performing the behavioral matching, the Digital Twin is ready for the last step of real-life validation and deployment. It is deployed as either a software service or hardware description language module that runs in parallel with the real system and receives live data streams to perform further analysis, such as prognosis or fault detection.

Table I
Flyback Converter Documentation

Component	Features
High Speed Optocouplers (H11N1)	High Data Rate, 5 MHz Operating Voltage: $4 \div 15\,V$ Rise Time: $7.5\,ns$
Optocoupler Pull up Resistance Base Resistor Collector Resistor	$R_1 = 390\,\Omega$ $R_2 = 10\,k\Omega$ $R_3 = 10\,k\Omega$
NPN 2N3904 BJT Transistor (Q_1)	Switch. Freq.: $> 250\,MHz$
Input Snurber Filter	$D_1 =$ 1N4148 Diode $R_4 = 6.8\,k\Omega$ $C_1 = 100\,pF$
IRFZ44 Mosfet Transistor	Max. Drain-Source Volt.: $55\,V$ Maximum Drain Current: $49\,A$
750315038 DC/DC Transformer ($T1$)	Operating Frequency: $250 \div 600\,kHz$ Inductance: $36.5\,\mu H$
Output Snurber Filter	$D_2 =$ 1N4148 Diode $R_5 = 22\,\Omega$ $C_2 = 1\,nF$
Resistive Load	$R_L = 120k\Omega$

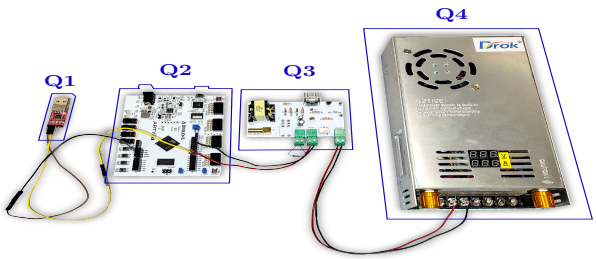

Figure 2. Flyback Converter System Based on FPGA

III. Case Study: Flyback Converter

In this paper, the voltage regulation control system shown in Fig. 2 based on a Flyback converter is used as case study for Digital Twin assembly and deployment using the development framework described in Section II.

A. DT Target System Definition and Documentation

The application is composed of a Flyback converter PCB board (Q3) and a power supply (Q4). The voltage regulation of (Q3) is controlled using a proportional-integral (PI) controller implemented in Verilog employing a Xilinx Arty S7-50 FPGA (Q2) [20]. The voltage value, control action, and error data from the physical asset are logged via serial port communication, using a USB-UART Converter (Q1). In the second step, for a detailed information on each component, please check [20].

B. Multidomain Simulation

In the third step, multidomain simulation, the system can be represented using two simulation domains: Electrical and Digital. The first domain is composed of the power supply and the flyback converter board while the second one, which

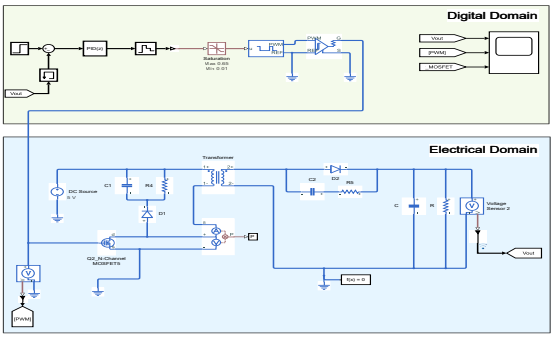

Figure 3. Assembled DT multidomain simulation

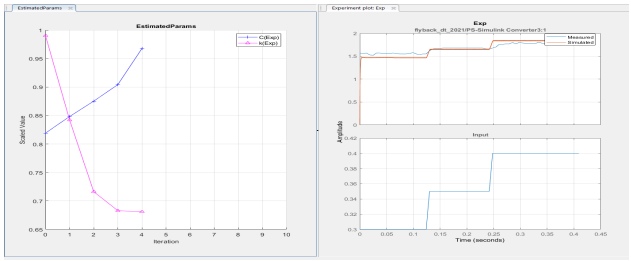

Figure 4. Behavioral matching using SLDO optimization

corresponds to the digital domain, is composed of the PID control algorithm and the analogue-to-digital interfaces to communicate the control side with the flyback Converter. Figure 3 shows the multidomain simulation of the system using Matlab/Simulink/Simscape.

C. Behavioral Matching

In the fourth step, behavioral matching, the multidomain simulation response is adjusted to replicate the system response using input/output data from the physical asset by optimizing the model's electrical domain parameters to determine the asset's current values. The flyback converter datasheet defines these parameters' initial values, shown in Table II. The Simulink Design Optimization (SLDO) Toolbox is used to determine the real parameters of the multiphysics model.

Figure 4 shows the behavioral matching results from the velocity open-loop experiment performed to update the multiphysic model with a new set of parameters shown in the last column of Table II.

D. DT Model Reduction for Hardware Implementation

Once the behavioural matching is completed, the DT for the power converter system is consistent and could be considered for deployment. However, it is crucial to notice that the current multidomain simulation model on Fig. 3 is computationally expensive for real-time execution on the FPGA due to its multiphysics nature. Therefore, a data-driven representation of

Table II
FLYBACK CONVERTER PARAMETERS BEFORE/AFTER BEHAVIORAL MATCHING

Parameter	Manufacturer values	Value after SLDO
Mutual inductance M [H]	2.3673e-05	1.5735e-5
Coupling coefficient k	0.99	0.6580
Capacitor C[F]	1e-4	1.1943e-4

the DT is required, which can be synthesized into Hardware Description Language (HDL) to ensure its real-time execution on the FPGA hardware level. Thus, a discrete transfer function of the flyback converter system is identified based on the Digital Twin response given by (1) with sampling time $t_s = 1e-5\,s$, where $y(z)$ is the voltage value, and $u(z)$ the flyback converter duty cycle.

$$P(z) = \frac{y(z)}{u(z)} = \frac{0.2781z^2 + 0.5561z + 0.2781}{z^2 + 0.6723z + 0.9396} \tag{1}$$

The equation in (1) is derived from the modelling of a flyback converter using the state space averaging technique described in Raj et al. [21]. The parameters $R_{sw} = 0.01$, $n = 1/2.33$, $Rc = 0$, $R = 120e3$, $V_g = 5$, and $V_d = 0.7$ are used in this derivation. The small signal model solution, with $X = [0, 0]$ as the initial condition, is obtained for the converter operating around a duty cycle of 30%. The Tustin discretization method is applied with a sampling time of $t_s = 1e-5s$ resulting as (1).

Likewise, the PI controller used for voltage regulation is represented by (2) where k_p is the proportional gain, t_i is the integral time, and $e(z)$ is the system error.

$$PI(z) = \frac{u(z)}{e(z)} = k_p + k_i\left(\frac{t_s}{z-1}\right) \tag{2}$$

The data-driven model in Simulink representing the Digital Twin of the power converter is shown in Fig. 6. The DT model is optimized to operate with fixed point arithmetic and incorporates the complete closed-loop architecture of the power converter system. The inputs are the current system setpoint and the proportional and integral gains for the PI controller. The model outputs include the DT error, proportional and integral actions, the manipulated variable (MV) corresponding to the MOSFET pulse width, and the process variable (PV), the voltage.

IV. DIGITAL TWIN HDL GENERATION AND INTEGRATION

The workflow that represents the whole synthesis process from the HDL Code generation to the DT implementation on the FPGA board is illustrated in Figure 5. It can be divided into four sequential steps: Matlab Test sequences, Test Bench, Synthesis and Implementation.

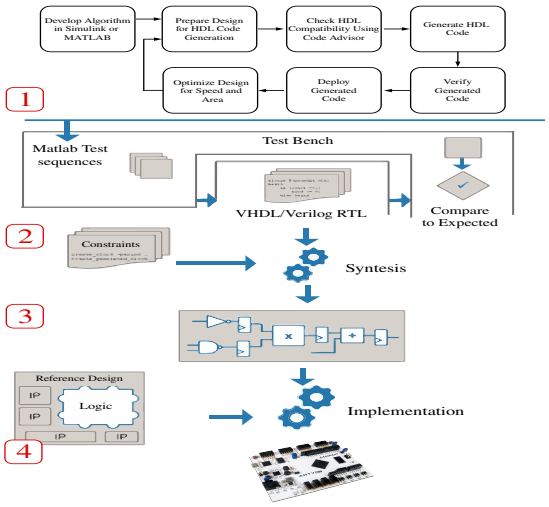

Figure 5. HDL workflow scheme for Digital Twin Implementation. Adapted from [22].

A. Matlab Test Sequences

Before implementing the module, a set of tests is conducted to ensure the model's consistency. The first test involves using the Model Checker, which verifies the model's configuration settings, ports and subsystem settings, block settings, support for native floating point, and compliance with industry-standard rules. Afterwards, HDL/Verilog code is generated along with a bit-true and cycle-accurate model to the generated HDL code. This model demonstrates the impact of block implementations, as well as speed and area optimizations that were specified. Lastly, the HDL coder creates a validation model that compares the original model to the generated model because the generated model is often substantially different from the original model. The validation model inserts delays at the outputs of the original model to compensate for latency differences and compares the outputs of the two models. Once it has been confirmed that there are no errors in the outputs, a testbench can be generated to evaluate the module's response before it is instantiated in the top module. This step ensures the design works as intended before FPGA programming occurs.

B. Test Bench

The Test Bench is a Xilinx Vivado project produced by Simulink to verify the Design Under Test (DUT). HDL Verifier automates this co-simulation process by creating expected outputs and comparing them within the testbench code. A message is generated indicating whether the test passed or failed and any potential identified mismatching. Furthermore, the results of the Simulink DT model can be compared with the behavioural simulation within the Vivado Test Bench simulation, indicating a correct execution of the system dynamics on the Register Transfer Level (RTL) level of FPGA.

C. Synthesis

Now that the module has been confirmed to perform as intended, it can be instantiated in the top module. The setpoint and PI controller parameters must be provided to ensure that it behaves like the control module. Since tasks can be performed in parallel on FPGAs, both the control module and the Digital Twin module can be run simultaneously. During synthesis, digital logic gates are generated from the Register Transfer Level (RTL) code while attempting to achieve the desired register-to-register clock frequency goals and minimizing the use of FPGA resources.

D. Implementation

This process involves determining the physical resources on the FPGA with the corresponding logic and how to connect (route) them. This process creates a bitstream to be loaded onto the device for FPGA programming. Likewise, the monitoring of the Digital Twin and the physical asset is performed using the Labview-based interface shown in Fig. 8.

E. Digital Twin Integration

Using the proposed framework, the flyback converter Digital Twin (DT) shown in Fig. 6 is synthesized using the Matlab HDL Code Generation toolbox, which enables HDL code generation in Verilog and VHDL from MATLAB and Simulink models. In this paper, the DT model is synthesized in Verilog for execution in a Xilinx FPGA, where it runs the closed-loop control of the system. The output of the MATLAB HDL synthesis is a module that can be integrated into the top module of the FPGA hardware design.

Figure 7a illustrates the schematic of the embedded DT architecture. In the virtual domain, the DT functions as one submodule of the HDL code, receiving the current process setpoint to perform voltage regulation. Simultaneously, in the physical domain, the case study hardware connects to the FPGA, containing the PI controller responsible for voltage regulation, which can interact with the DT in real time.

Similarly, Fig. 7b displays the block diagram of the HDL code running within the FPGA, encompassing voltage regulation, digital twin, and various required hardware components for closed-loop control, such as setpoint selection, ADC reading, PI control, PWM generation, and serial communication. The power converter system's digital twin executes in parallel to the voltage regulation closed-loop control. Additionally, a DT-Asset error module calculates the real-time error between the asset and the digital twin, serving as an awareness signal for event detection over the physical system. The HDL code containing the DT and the closed-loop control can be downloaded from [23], and the final result video can be found from [24].

V. Results and Discussion

After implementing the Digital Twin of the Flyback Converter on the FPGA, its response is compared against the

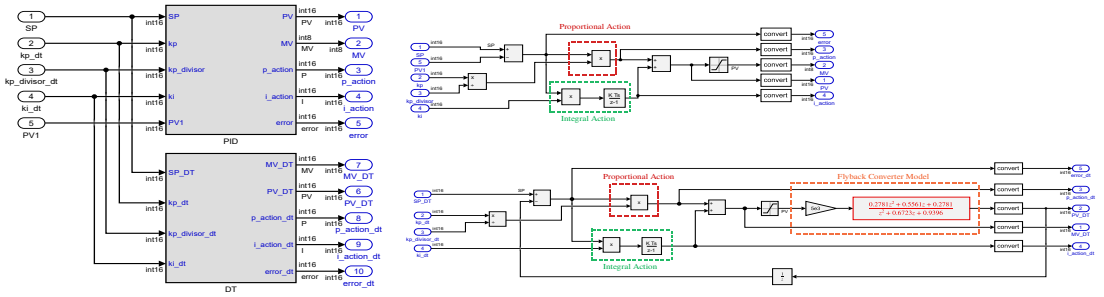

Figure 6. DT of the Flyback Converter on Matlab/Simulink with Data-Driven Model Dynamics

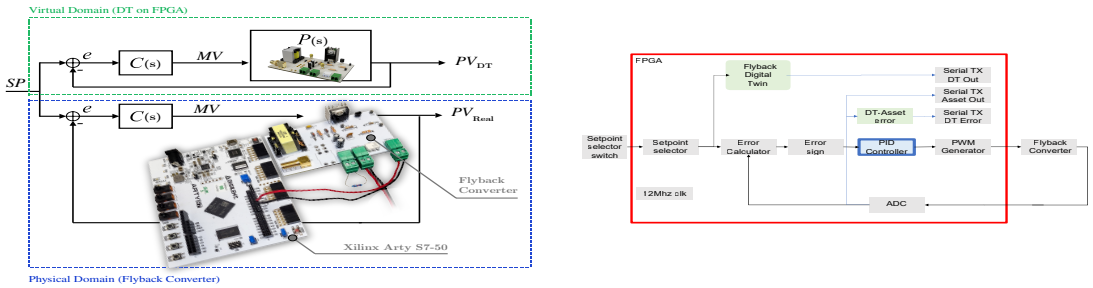

Figure 7. Digital Twin Parallel Execution a) Architecture and b) HDL Design Block Diagram.

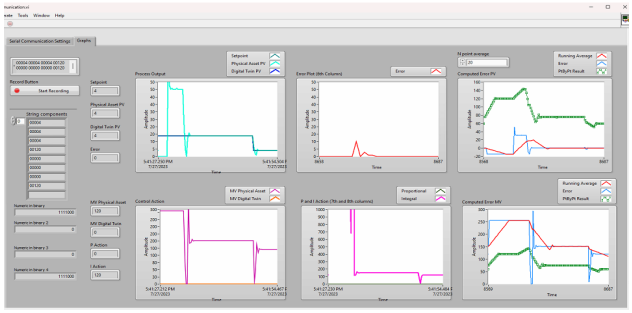

Figure 8. Labview GUI for Flyback testing and implementation

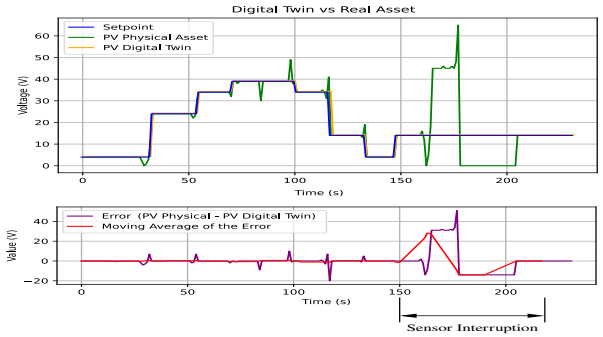

Figure 9. Comparison between the output voltage of the Digital Twin and Physical Asset and 15 Samples Moving Average Error (DT minus Asset) for ADC communication interruption detection (purple line).

output of the physical asset, as depicted in Fig 9. The response of the digital twin closely follows the physical asset output with minimal deviation. In Fig 9, the bottom plot illustrates the minimum error between the digital twin and the physical asset. Although the error should ideally be zero, some regions with minor spikes are observed. These spikes are a result of random noise affecting the ADC signal of the physical system.

However, during steady-state execution for multiple set-points, both the DT and the physical asset exhibit consistent behavior, as demonstrated by the results. Furthermore, the behavioral matching process effectively minimized the error between the physical asset and the digital twin. To achieve even better accuracy, we introduce more complexity into the digital twin model, such as accounting for noisy sensor behavior or nonlinear effects on the Flyback converter.

A. DT enabling capability: Events detection

The Digital Twin response provides valuable error signals that enable event detection over the physical asset. Using a moving average of the error signal, we can monitor an undesirable effect—the loss of connection of the ADC—while the closed-loop control is active. Figure 9 illustrates this with an example of the error signal's moving average for detecting ADC communication interruptions during the asset's steady-state operation. As observed, the undesired event is induced by physically disconnecting the ADC signal from the FPGA.

By implementing a statistical warning policy based on the moving average standard deviation, following industrial standards like six-sigma [25], we can detect and anticipate such events. This integration of capabilities into local supervisory algorithms makes the control systems smarter by continuously monitoring the current health status of the system. Additionally, enabling the Digital Twin's parallel execution directly on the embedded system unlocks novel capabilities, providing an additional information source for local supervisory algorithms, and enhancing the control system's intelligence in monitoring and maintaining the system's health.

VI. Conclusions and future works

An FPGA-based digital twin implementation is employed to optimize a power converter system controller in real-time, showcasing the benefits of a digital framework that circumvents the cloud-data exchange intermediary protocol. This approach offers edge digital twin operation ensuring data privacy, which is crucial for safe process operation.

The case study focused on voltage regulation for a Flyback Converter, where the Digital Twin of the system was synthesized using the HDL coder of the Flyback Converter to execute it on an FPGA board, enabling monitoring of the physical asset's desired operating condition. The obtained results demonstrate precise output with reduced error between the physical and digital assets. Moreover, real-time error computation between the Digital Twin and physical asset enables novel analytics for event detection within the system, such as detecting feedback sensor communication interruptions through moving averages error. Consequently, this work effectively showcases the implementation of digital twin technology on a low-level hardware-embedded controller, introducing intelligent capabilities to the embedded domain.

Future endeavors will involve utilizing the insights gained from observations for predictive maintenance, fault diagnostics, and smart control engineering applications.

References

[1] S. Vazquez, J. Rodriguez, M. Rivera, L. G. Franquelo, and M. Norambuena, "Model predictive control for power converters and drives: Advances and trends," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 2, pp. 935–947, 2017.

[2] Y. Yin, J. Liu, W. Luo, L. Wu, S. Vazquez, J. I. Leon, and L. G. Franquelo, "Adaptive control for three-phase power converters with disturbance rejection performance," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 2, pp. 674–685, 2021.

[3] S. E. De León-Aldaco, H. Calleja, and J. Aguayo Alquicira, "Metaheuristic optimization methods applied to power converters: A review," *IEEE Transactions on Power Electronics*, vol. 30, no. 12, pp. 6791–6803, 2015.

[4] A. J. Wileman, S. Aslam, and S. Perinpanayagam, "A component level digital twin model for power converter health monitoring," *IEEE Access*, vol. 11, pp. 54143–54164, 2023.

[5] G. D. Nezio, M. d. Benedetto, A. Lidozzi, and L. Solero, "Dc-dc boost converters parameters estimation based on digital twin," *IEEE Transactions on Industry Applications*, pp. 1–9, 2023.

[6] P. Aivaliotis, K. Georgoulias, Z. Arkouli, and S. Makris, "Methodology for enabling digital twin using advanced physics-based modelling in predictive maintenance," *Procedia Cirp*, vol. 81, pp. 417–422, 2019.

[7] D. Shangguan, L. Chen, and J. Ding, "A hierarchical digital twin model framework for dynamic cyber-physical system design," in *Proceedings of the 5th international conference on mechatronics and robotics engineering*, pp. 123–129, 2019.

[8] K. Yan, W. Xu, B. Yao, Z. Zhou, and D. T. Pham, "Digital twin-based energy modeling of industrial robots," in *Methods and Applications for Modeling and Simulation of Complex Systems: 18th Asia Simulation Conference, AsiaSim 2018, Kyoto, Japan, October 27–29, 2018, Proceedings 18*, pp. 333–348, Springer, 2018.

[9] J. A. Erkoyuncu, P. Butala, R. Roy, *et al.*, "Digital twins: Understanding the added value of integrated models for through-life engineering services," *Procedia Manufacturing*, vol. 16, pp. 139–146, 2018.

[10] J. Fitzgerald, P. G. Larsen, and K. Pierce, "Multi-modelling and co-simulation in the engineering of cyber-physical systems: towards the digital twin," *From Software Engineering to Formal Methods and Tools, and Back: Essays Dedicated to Stefania Gnesi on the Occasion of Her 65th Birthday*, pp. 40–55, 2019.

[11] Y. Lu, C. Liu, I. Kevin, K. Wang, H. Huang, and X. Xu, "Digital twin-driven smart manufacturing: Connotation, reference model, applications and research issues," *Robotics and Computer-Integrated Manufacturing*, vol. 61, p. 101837, 2020.

[12] M. A. Hailan, B. M. Albaker, and M. S. Alwan, "Transformation to a smart factory using nodemcu with blynk platform," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 30, no. 1, pp. 237–245, 2023.

[13] R. G. Alves, R. F. Maia, and F. Lima, "Development of a digital twin for smart farming: Irrigation management system for water saving," *Journal of Cleaner Production*, p. 135920, 2023.

[14] L. Girletti, M. Groshev, C. Guimarães, C. J. Bernardos, and A. de la Oliva, "An intelligent edge-based digital twin for robotics," in *2020 IEEE Globecom Workshops (GC Wkshps*, pp. 1–6, 2020.

[15] H. Huang, L. Yang, Y. Wang, X. Xu, and Y. Lu, "Digital twin-driven online anomaly detection for an automation system based on edge intelligence," *Journal of Manufacturing Systems*, vol. 59, pp. 138–150, 2021.

[16] R. Martinez-Velazquez, R. Gamez, and A. El Saddik, "Cardio twin: A digital twin of the human heart running on the edge," in *2019 IEEE International Symposium on Medical Measurements and Applications (MeMeA)*, pp. 1–6, IEEE, 2019.

[17] M. R. Sarac and O. Aydogmus, "Design and implementation of an fpga-based digital twin for an electric motor," in *Proceedings of Fourth International Conference on Communication, Computing and Electronics Systems: ICCCES 2022*, pp. 613–623, Springer, 2023.

[18] S. Memis and R. Yeniceri, "Towards fpga based digital twin of uav swarms: An area efficient hardware accelerator of transformation matrix of 6-dof block," in *AIAA SCITECH 2023 Forum*, p. 2130, 2023.

[19] J. Viola and Y. Chen, "Digital twin enabled smart control engineering as an industrial ai: A new framework and case study," in *2020 2nd International Conference on Industrial Artificial Intelligence (IAI)*, pp. 1–6, 2020.

[20] Xilinx Inc, "Arty s7-50: Spartan-7 fpga for makers and hobbyists." Available at: https://www.xilinx.com/products/boards-and-kits/1-pnziih.html, 2023.

[21] A. S. Raj, A. M. Siddeshwar, K. Guruswamy, C. Maheshan, *et al.*, "Modelling of flyback converter using state space averaging technique," in *2015 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)*, pp. 1–5, IEEE, 2015.

[22] Mathworks, "Fpga programming." Available at: mathworks.com/discovery/fpga-programming, 2023.

[23] Repository, "Github." Available at: github.com/marco-milanesi/FlybackConverter-FPGA-based-Digital-Twin, 2023.

[24] MESALAB, "final result video," 2023.

[25] S. Tissir, A. Cherrafi, A. Chiarini, S. Elfezazi, and S. Bag, "Lean six sigma and industry 4.0 combination: scoping review and perspectives," *Total Quality Management & Business Excellence*, vol. 34, no. 3-4, pp. 261–290, 2023.