International Journal of Innovative Research and Scientific Studies, 9(6) 2026, pages: 51-68

IJIRSS
International Journal of Innovative Research & Scientific Studies

ISSN: 2617-6548

URL: www.ijirss.com

Hardware-in-the-loop-based SVPWM control of a PMSG-based wind energy conversion system

Omar Stihi[1*], Oumar Kone[2], Mehdi Fazilat[2], Fares Boudjema[1], Mohamed Tadjine[3], Nadjet Zioui[2]

[1]*École Nationale Polytechnique, 10 Rue des Frères OUDEK, El Harrach 16200, Algiers, Algeria.*
[2]*Université du Québec à Trois-Rivières, 3351 Bd des Forges, Trois-Rivières, QC G8Z 4M3, Canada.*
[3]*École Nationale supérieure des Technologies Avancées, Cité Diplomatique Ex Centre Biomédical Dergana-Bordj El Kiffan, 16087, Algiers, Algeria.*

Corresponding author: Omar Stihi (*Email: omar.stihi@g.enp.edu.dz*)

Abstract

Simulation-based validation of Space Vector Pulse Width Modulation (SVPWM) algorithms often fails to capture numerical, timing, and communication limitations encountered during deployment on embedded hardware. This study proposes a low-cost serial-communication-coupled embedded validation framework to assess the SVPWM switching-state computation of the machine-side converter in a Permanent Magnet Synchronous Generator (PMSG)-based horizontal-axis wind turbine energy conversion system. The aerodynamic model of the wind turbine, the electrical dynamics of the PMSG, the bidirectional converter, the speed control loop, and the d-q current control loops were developed using MATLAB/Simulink. In contrast, the computationally critical SVPWM functions, including sector identification, hold time calculation, and switching state generation, were implemented on an Arduino UNO microcontroller. The framework employs serial communication between the simulation environment and embedded controller, providing a practical intermediate validation stage. It is intended as a preliminary embedded validation platform rather than a cycle-by-cycle PWM-synchronous real-time hardware-in-the-loop (HIL) system. Comparative analysis demonstrates that the Arduino-executed SVPWM algorithm successfully reproduces the dominant transient and steady-state characteristics observed in the ideal Simulink implementation. Generator-speed, d-q current, and d-q voltage responses exhibit close agreement, while the remaining discrepancies are minor and attributable to finite-precision arithmetic, serial data exchange, embedded execution delays, and communication latency. The proposed framework effectively bridges the gap between simulation-only verification and advanced real-time implementation by enabling practical assessment of embedded SVPWM computation under realistic operating constraints. The framework offers an accessible and cost-effective approach for preliminary validation of embedded SVPWM implementations in PMSG-based wind energy systems, reducing development risk before transitioning to high-fidelity real-time HIL platforms or full-scale power-hardware experiments.

Keywords: Embedded SVPWM implementation, Hardware-coupled validation, Machine-side converter, Permanent magnet synchronous generator, Space vector pulse width modulation, Wind energy conversion system.

DOI: 10.53894/ijirss.v9i6.11718
Funding: This study received no specific financial support.
History: Received: 18 March 2026 / **Revised:** 22 May 2026 / **Accepted:** 25 May 2026 / **Published:** 2 June 2026

Competing Interests: The authors declare that they have no competing interests.
Authors' Contributions: All authors contributed equally to the conception and design of the study. All authors have read and agreed to the published version of the manuscript.
Transparency: The authors confirm that the manuscript is an honest, accurate, and transparent account of the study; that no vital features of the study have been omitted; and that any discrepancies from the study as planned have been explained. This study followed all ethical practices during writing.
Publisher: Innovative Research Publishing

1. Introduction

Wind Energy Conversion Systems have become a central element in the continuing shift toward renewable electricity generation, since they allow electrical energy to be extracted at large scale from a resource that is renewable and broadly available [1-3]. Within variable-speed wind turbines, Permanent Magnet Synchronous Generators (PMSGs) have attracted sustained attention because they combine high efficiency, high power density, and compact construction with practical compatibility for direct-drive and reduced-gearbox arrangements [4-6]. These features are especially relevant for Horizontal-Axis Wind Turbines (HAWTs), where the generator must support efficient electromechanical conversion while limiting mechanical complexity and maintaining acceptable operation under changing wind conditions. For this reason, PMSG-based Wind Energy Conversion System (WECSs) remain a significant focus in wind energy research, particularly in studies concerned with dynamic modeling, converter-level control, real-time implementation, and validation of embedded control strategies.

The broader adoption of PMSGs in variable-speed wind turbines is also tied to the use of full-scale power electronic conversion stages. In a common PMSG-based WECS configuration, the generator is interfaced with the electrical network through a back-to-back converter, which separates the turbine mechanical speed from the grid frequency [7-9]. This arrangement typically includes a Machine-Side Converter (MSC), a Direct Current (DC)-link capacitor, and a Grid-Side Converter (GSC). The MSC is responsible for regulating the electromechanical conversion process, commonly through generator torque, speed, or stator-current control, whereas the GSC regulates the DC-link voltage and governs controlled power injection into the grid. This division of functions permits variable-speed turbine operation while preserving compatibility with grid-side electrical requirements. Consequently, the converter structure and its switching strategy are not only implementation details, but also key factors affecting current quality, dynamic behavior, and the reliability of real-time control deployment [10, 11]. Control of PMSG-based Wind Energy Conversion Systems is usually developed in the synchronous rotating d-q reference frame, where stator quantities are separated into current components that can be regulated with limited cross-coupling [4, 12]. For surface-mounted PMSGs, the direct-axis current is commonly assigned a zero reference, while the quadrature axis current becomes the main variable for controlling electromagnetic torque and, through it, generator speed. This choice leads naturally to a cascaded structure, in which the outer speed controller generates the current reference and the inner current controllers compute the voltage commands supplied to the machine-side converter. Proportional-integral (PI) and proportional-integral-derivative (PID) controllers are still used extensively in this setting, mainly because they are transparent to tune, computationally light, and compatible with real-time digital platforms. When these controllers are embedded in a vector-control scheme and paired with a suitable modulation method, the resulting system can provide acceptable current regulation and speed tracking behavior over the operating range considered for variable wind conditions [13-15].

The modulation stage then determines how the converter realizes the voltage commands produced by the control loops. In voltage-source converters, Space Vector Pulse Width Modulation (SVPWM) is widely used because it maps the inverter switching states into voltage vectors in the stationary $\alpha - \beta$ plane and reconstructs the reference voltage by selecting appropriate active and zero vectors [13, 16]. Relative to conventional Sinusoidal Pulse Width Modulation (SPWM), SVPWM generally makes better use of the direct-current (DC)-link voltage and can improve harmonic performance, which is relevant for maintaining converter current quality in wind energy systems. Recent PMSM-drive work has also used classical SVPWM as the benchmark modulation method for speed-control evaluation against quantum-inspired SVPWM, reporting improved RMS speed-error, d-axis current, q-axis current, and harmonic-performance indicators for the quantum variant [17]. Its practical value is also linked to its algorithmic form, since the procedure can be implemented through reference-vector calculation, sector identification, dwell-time calculation, and switching state generation [17, 18]. For this reason, SVPWM remains a suitable modulation option for embedded converter control in PMSG-based WECS applications. Although SVPWM can perform well in a simulated control system, this result does not by itself confirm that the same behavior will be preserved after deployment on physical digital hardware. In offline simulation, the modulation routine is usually evaluated without the numerical, timing, and communication limits imposed by an embedded implementation. A real controller, however, must operate under finite numerical precision, fixed sampling periods, computational latency, timer quantization, interrupt scheduling, and implementation-specific rounding effects [19, 20]. These factors may alter the final switching commands delivered to the converter rather than merely affecting the internal execution of the algorithm. For a PMSG-based WECS, such deviations are not necessarily negligible, since small errors in switching-state generation can propagate into current regulation, electromagnetic torque response, and generator speed

dynamics. For this reason, simulation-only validation remains incomplete when the purpose is to determine whether an SVPWM controller retains its expected dynamic behavior after real-time digital implementation.

This implementation gap has encouraged the use of Hardware-in-the-Loop (HIL) and Processor-in-the-Loop (PIL) methodologies for validating power-electronic converter controllers before full HIL-based embedded validation deployment. In these validation schemes, the plant model, for example a PMSG-based WECS, is executed in a controlled real-time or simulation environment, while the controller logic, modulation routine, or embedded code is run on actual digital hardware [19, 21]. HIL and PIL therefore provide an intermediate stage between purely numerical simulation and complete laboratory prototyping. Their main value is not to replace experimental testing, but to expose timing errors, processor-induced delays, numerical effects, and communication constraints at a stage where the controller can still be modified with relatively low cost and risk. Prior work has shown that real-time validation platforms, including dSPACE and OPAL-RT, are effective for testing wind turbine controllers, converter modulation strategies, and fault-tolerant control schemes under operating conditions that are closer to physical implementation [22, 23]. These platforms offer considerable flexibility and computational capability, but their cost and infrastructure requirements may limit their practicality for early stage embedded-control validation. In parallel, research on PMSG-based wind energy conversion systems has increasingly moved toward advanced control design, grid-support functions, and real-time digital implementation. Model predictive control, adaptive backstepping, sliding mode control, direct power control, and grid-forming control have been examined for improving power quality, transient response, low voltage ride-through capability, and frequency-support behavior [24, 25]. In the closely related permanent-magnet synchronous-machine drive literature, advanced direct-torque-control variants have also been investigated to improve torque ripple, commutation behavior, and harmonic performance [26]. A related body of implementation-oriented work commonly employs digital signal processors (DSPs), field-programmable gate arrays (FPGAs), or professional real-time simulation platforms, mainly because converter control and modulation algorithms impose strict timing and computational requirements [27, 28]. More broadly, recent motor-drive research has also explored implementable variants of PWM algorithms, including quantum-inspired PWM formulations for DC motor control, confirming the continuing interest in modulation-layer implementation for electrical drives [29]. These studies are valuable, yet their emphasis is generally placed on sophisticated control laws, converter-level complexity, or high-performance hardware. By contrast, the direct execution of the SVPWM switching-state computation on Arduino-class microcontroller hardware, within a Simulink coupled PMSG wind turbine HIL framework, remains less explicitly addressed.

This limitation matters because a low-cost embedded HIL arrangement can offer an early validation path for digital modulation algorithms before a complete wind turbine, generator, and power converter test bench is available. The purpose of such a setup is not to substitute professional real-time simulators or full HIL-based embedded validation platforms. Rather, it serves as an intermediate environment for checking whether the embedded realization of the modulation layer remains consistent with the corresponding simulation model. For SVPWM, this point is especially relevant, since the algorithm contains time-sensitive computations that ultimately produce discrete switching states. Implementing these computations on an Arduino-class microcontroller makes it possible to observe how restricted computational capacity, serial communication, numerical representation, and execution timing affect the closed-loop response of the simulated PMSG-based WECS. A comparison between simulation only and low-cost validation responses can therefore indicate whether the embedded implementation preserves the expected generator-speed, current-control, and voltage-control behavior.

To address this implementation gap, this paper develops and validates a hardware-coupled implementation (Arduino-coupled embedded SVPWM validation framework) of SVPWM control for a PMSG-based HAWT energy conversion system. The proposed framework keeps the wind turbine, PMSG, back-to-back converter, and control-reference generation models in MATLAB/Simulink, while the SVPWM switching-state computation is executed on an Arduino UNO microcontroller. The microcontroller receives the voltage-reference components in the $\alpha - \beta$ plane, computes the SVPWM timing and switching states, and returns the resulting switching information to the Simulink model through serial communication. The purpose of the study is deliberately limited. It does not introduce a new SVPWM algorithm or a new speed-control law, but instead examines whether a low-cost embedded implementation of the modulation stage can reproduce the expected behavior of the ideal simulation-based controller when both are connected to the same PMSG wind turbine model.

Accordingly, the contribution of the work is therefore practical and validation-oriented rather than algorithmic. First, a MATLAB/Simulink model of a HAWT-PMSG energy conversion system is developed, including the turbine aerodynamic model, PMSG electrical dynamics, back-to-back converter structure, and *d*-*q*-axis current control strategy. Second, the SVPWM timing and switching-state generation process is implemented on an Arduino UNO microcontroller, so that the modulation task is carried out on physical embedded hardware rather than being retained entirely inside the simulation environment. Third, a serial-communication-based interface is established between Simulink and the microcontroller, enabling hardware-coupled exchange of reference voltage components and switching commands at the communication update rate of the test setup. This exchange rate is lower than the PWM carrier frequency and is therefore used to evaluate the embedded SVPWM logic under communication and execution constraints, rather than to demonstrate cycle-by-cycle real-time PWM synchronization. Fourth, the simulation-only and Arduino-based embedded SVPWM switching logic implementations are comparatively evaluated in terms of generator speed, *d*-*q* current components, and *d*-*q voltage* control signals. This comparison is used to assess the practical feasibility, within the considered Arduino-based embedded SVPWM switching logic setting, of using low-cost microcontroller hardware for embedded SVPWM validation in PMSG-based wind energy applications.

The remainder of this paper is organized as follows. Section 2 presents the modeling and methodological framework, including the HAWT aerodynamic model, the PMSG electrical model, the back-to-back converter structure, the SVPWM formulation, the speed-control strategy, and the proposed Arduino-in-the-loop validation communication architecture. Section 3 presents and discusses the simulation and Arduino-based embedded SVPWM switching logic results, with emphasis on the comparison of generator speed, *d-q* current components, and *d-q* voltage control signals. Finally, Section 4 summarizes the main findings and concludes the paper.

2. Materials and Methods

2.1. Modeling of the PMSG-based HAWT System

In the HAWT-based WECS considered in this work, the aerodynamic subsystem represents the first stage of energy conversion, where the kinetic energy available in the incoming wind is converted into mechanical power at the turbine rotor. For the present control-oriented study, the aerodynamic subsystem is represented using the conventional power-coefficient formulation. This reduced-order model describes the captured turbine power as a function of wind speed, rotor swept area, air density, and the aerodynamic power coefficient $C_p(\lambda,\beta)$. The C_p-based formulation is widely used in wind turbine control studies because it preserves the dominant relationship between rotor speed, wind speed, blade pitch angle, and extracted mechanical power, while avoiding the complexity of detailed blade element calculations [30-32]. Under a uniform incoming wind speed, the mechanical power captured by the WT rotor is expressed as:

$$P_w = \frac{1}{2}\rho A {v_w}^3 C_p(\lambda,\beta) \tag{1}$$

In this formulation, ρ denotes the air density, which directly affects the amount of kinetic energy available in the wind flow. The parameter A represents the area swept by the turbine rotor and is defined as $A = \pi R^2$, where R is the rotor radius. Therefore, a larger rotor radius increases the swept area and enables the turbine to capture more wind energy. The variable v_w indicates the wind speed, which has a strong influence on the extracted aerodynamic power. The coefficient $C_p(\lambda,\beta)$ is the power coefficient of the wind turbine and expresses the efficiency with which the turbine converts the available wind power into mechanical power. This coefficient depends on the tip-speed ratio λ and the blade pitch angle β. The tip-speed ratio is defined as $\lambda = (\omega_{rt}R)/v_w$, where ω_{rt} is the angular speed of the turbine rotor. Thus, λ relates the tangential speed at the blade tip to the incoming wind speed and plays a central role in determining the aerodynamic operating condition of the turbine. The expression of the power coefficient $C_p(\lambda,\beta)$ is as in Equation 2 that has been determined empirically:

$$\begin{cases} C_p(\lambda,\beta) = c_1\left(\frac{c_2}{\lambda_i} - c_3\beta - c_4\right) exp\left(-\frac{c_5}{\lambda_i}\right) + c_6\lambda \\ \lambda_i = \left(\frac{1}{\lambda + 0.08\beta} - \frac{0.035}{\beta^3 + 1}\right)^{-1} \end{cases} \tag{2}$$

c_1 to c_6 are constant parameters of the model. The power extracted by the WT is constrained by the Betz limit to around 59.26% of the kinetic energy of the wind. The mechanical torque T_m on the rotor shaft of the turbine is determined using the following expression:

$$T_m = \frac{P_w}{\omega_{rt}} \tag{3}$$

In many systems, there is a mechanical transmission, between the turbine rotor and the generator, including a potential gearbox, that adapts the low-speed, high-torque output of the turbine rotor to the high speed required by the generator. This coupling is modeled so that the generator angular speed ω_{gen} is related to the turbine speed ω_{rt} by the gear ratio k_{gb} such as $k_{gb} = \frac{\omega_{gen}}{\omega_{rt}}$. In this work, the drivetrain is represented by a simplified single-mass mechanical model with equivalent inertia and viscous friction. Therefore, the rotor-generator dynamics are described by the following fundamental motion equation [33]:

$$J\frac{d\omega_{gen}}{dt} + f\omega_{gen} = \frac{T_m}{k_{gb}} - T_{el} \tag{4}$$

In the rotor dynamic model, T_{el} denotes the electromagnetic torque generated by the electrical machine. The parameter f represents the equivalent viscous friction coefficient, accounting for speed-dependent mechanical losses. The term J is the equivalent total inertia of the drivetrain, combining the wind turbine and hub inertia (J_{wt}) with the generator rotor and shaft inertia (J_{gen}). Considering the gearbox ratio k_{gb}, it is expressed as $J = J_{wt}/k_{gb}^2 + J_{gen}$. If lumped parameters are used directly in the simulation model, J and f denote the corresponding generator-side equivalent inertia and viscous friction coefficient.

In the mechanical drivetrain equation, ω_{gen} represents the mechanical angular speed of the generator shaft. However, the voltage equations of the PMSG are formulated in the synchronous rotating $d-q$ reference frame, where the speed-dependent coupling terms must be expressed using the electrical angular speed. Therefore, the electrical angular speed is defined as $\omega_e = p\omega_{gen}$, where p denotes the number of pole pairs. This definition is used in the following stator voltage equations to distinguish the mechanical generator speed from the electrical speed associated with the rotating magnetic field.

In the electrical part of the wind energy conversion system, the captured mechanical power is converted into electric power by a Permanent Magnet Synchronous Generator (PMSG). For modeling the PMSG, the standard approach is to use the synchronous-rotating reference $d-q$ frame to express the stator voltage and flux equations derived from the electrical scheme of the PMSG depicted in Figure 1. The stator voltage equations can be written as follows [33-35]. According to the equivalent electrical scheme of the PMSG shown in Figure 1, and using the generator sign convention, the stator voltage equations in the synchronous rotating *d-q* frame are expressed as follows:

$$\begin{aligned} v_d &= -R_s i_d - L_{sd}\frac{di_d}{dt} + L_{sq} i_q \omega_e \\ v_q &= -R_s i_q - L_{sq}\frac{di_q}{dt} + L_{sd} i_d \omega_e + \omega_e \varphi_m \end{aligned} \tag{5}$$

where $\omega_e = p\omega_{\text{gen}}$ is the electrical angular speed, ω_{gen} is the mechanical angular speed of the generator, and p is the number of pole pairs.

In the PMSG electrical model, v_d and v_q denote the stator voltage components along the direct and quadrature axes, respectively, while i_d and i_q represent the corresponding stator current components in the synchronous rotating $d-q$ reference frame. The parameter R_s is the stator resistance, and φ_m denotes the permanent-magnet flux linkage. The terms L_{sd} and L_{sq} represent the stator inductances along the direct and quadrature axes, respectively. In these voltage equations, the speed-dependent coupling and back-electromotive-force terms are expressed using the electrical angular speed ω_e, which is related to the mechanical generator speed by $\omega_e = p\omega_{gen}$, where p is the number of pole pairs.

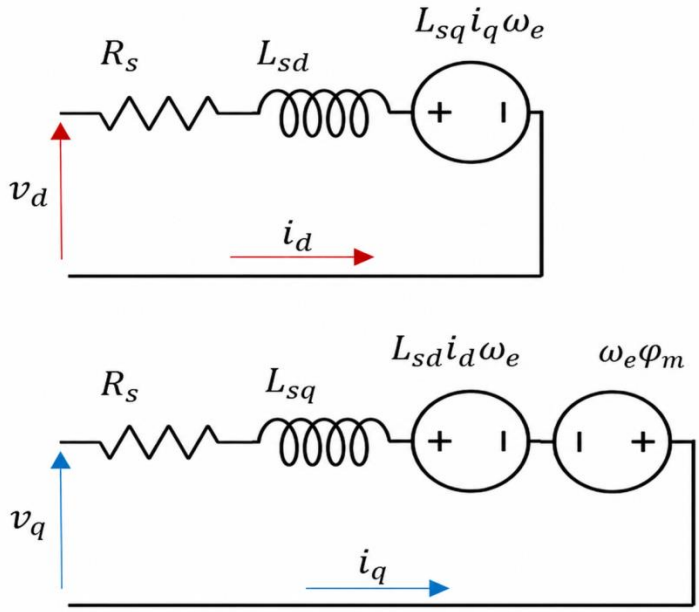

Figure 1.
The PMSG equivalent electrical scheme.

From the stator voltage equations in the synchronous rotating *d-q* reference frame, the electromagnetic torque of the PMSG can be expressed as [36, 37]:

$$T_{el} = \frac{3}{2} p\left[\varphi_m i_q + \left(L_{sd} - L_{sq}\right) i_d i_q\right] \tag{6}$$

With the current direction adopted in this paper, T_{el} is interpreted as the generator electromagnetic braking torque. Therefore, a positive i_q corresponds to positive electromagnetic torque opposing the mechanical input torque in (6). The second term, $\left(L_{sd} - L_{sq}\right) i_d i_q$, represents the reluctance torque component associated with the saliency of the machine. This formulation is consistent with the d - q-axis PMSG model, where the electromagnetic torque depends on the permanent-magnet flux linkage, the stator current components, and the difference between the direct- and quadrature-axis inductances.

For a surface-mounted PMSG, the rotor magnetic structure is commonly considered non-salient, and the direct- and quadrature-axis inductances are approximately equal, i.e., $L_{sd} \approx L_{sq}$. Therefore, the reluctance torque component becomes negligible, and the electromagnetic torque is simplified as:

$$T_{el} = \frac{3}{2} p \varphi_m i_q \tag{7}$$

To interface the PMSG with the grid, a full-scale back-to-back converter is used, as shown in Figure 2. It consists of two three-phase voltage-source converters sharing a common DC-link capacitor: the machine-side converter on the

generator side and the grid-side converter [38, 39]. The MSC regulates the generator-side variables and applies the voltage references generated by the d - q current controllers through the SVPWM algorithm. The GSC regulates the DC-link voltage and controls power injection into the grid. This structure enables variable-speed PMSG operation while maintaining grid compatible output conditions.

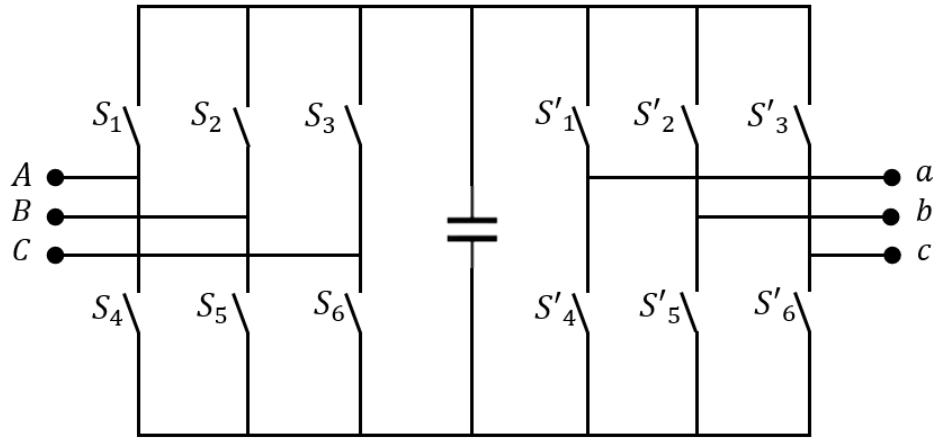

Figure 2.
Diagram of the B2B converter used in the PMSG based WT.

In the considered PMSG-based wind energy conversion system, the generator is interfaced with the grid through a full-scale back-to-back voltage-source converter composed of a machine-side converter (MSC), a DC-link capacitor, and a grid-side converter (GSC). Under generator-side power flow, the MSC operates as an active controlled rectifier, converting the variable-frequency three-phase stator voltages of the PMSG into DC power and transferring this power to the common DC-link. The GSC then converts the DC-link power into grid-synchronized three-phase AC power. This intermediate DC stage decouples the electromechanical operating point of the wind turbine-generator set from the grid frequency, thereby enabling variable-speed operation of the PMSG while maintaining fixed-frequency power injection at the grid terminals [40]. This functional separation is a standard feature of full-scale converter-based Type- 4 wind turbine systems, where the converter interface electrically separates the turbine rotor dynamics from the grid-side electrical variables [41]. In this work, the grid-side converter and grid interface are included only as part of the simulated WECS environment required to represent the back-to-back converter structure and DC-link interaction. The embedded validation focus is limited to the SVPWM switching-state computation of the machine-side converter.

The MSC is responsible for regulating the generator-side electromechanical conversion process. In vector controlled PMSG drives, this task is commonly achieved by controlling the stator current components in the synchronous d-q reference frame. The d-axis current is generally set to zero for a surface-mounted PMSG, while the q-axis current is used to regulate the electromagnetic torque and, consequently, the generator speed. The outputs of the current controllers are the voltage references v_d^* and v_q^*, which are transformed into the stationary $\alpha\beta$ frame and supplied to the modulation stage. In the present work, these voltage references are applied to the MSC through the SVPWM algorithm, so that the requested converter voltage vector is synthesized by appropriate switching of the converter legs [42].

The GSC controls the grid-side power exchange and stabilizes the DC-link voltage by balancing the power received from the MSC with the power delivered to the grid. In a conventional grid-connected configuration, the GSC regulates the DC-link voltage through the active-current component, while the reactive-current component is used to impose the desired reactive power or power-factor condition at the point of common coupling. In a conventional grid-connected wind turbine converter structure, the generator-side converter primarily governs the electromechanical power extracted from the PMSG, whereas the grid-side converter regulates the DC-link voltage and controls the reactive-power exchange with the grid [25, 39]. This separation allows the active-power flow from the generator and the reactive-power requirement at the grid interface to be controlled independently. It is particularly important during grid-side disturbances, where variations in grid voltage or injected power can create an instantaneous mismatch between the power delivered by the MSC and the power transferred by the GSC. If this imbalance is not properly regulated, the excess or deficient energy is absorbed by the DC-link capacitor, resulting in DC-link voltage oscillations, degraded power quality, and increased harmonic stress on the converter and connected electrical components.

Let $S_1 - S_6$ denote the switching functions of the MSC and $S'_1 - S'_6$ those of the GSC. For each converter leg, a switching function equal to unity indicates that the corresponding upper switch is conducting, whereas a value of zero indicates that the complementary lower switch is conducting, assuming ideal complementary gating and neglecting dead-time effects. Based on this switching representation, the phase-to-neutral voltages of the MSC, v_{AN}, v_{BN}, and v_{CN}, and those of the GSC, v'_{AN}, v'_{BN}, and v'_{CN}, can be expressed as functions of the switching states and the DC-link voltage as follows:

$$\begin{pmatrix} v_{AN} \\ v_{BN} \\ v_{CN} \end{pmatrix} = \frac{V_{DC}}{3} \begin{pmatrix} 2S_1 - S_3 - S_5 \\ 2S_3 - S_1 - S_5 \\ 2S_5 - S_3 - S_1 \end{pmatrix} \quad (8)$$

$$\begin{pmatrix} v'_{AN} \\ v'_{BN} \\ v'_{CN} \end{pmatrix} = \frac{V_{DC}}{3} \begin{pmatrix} 2S'_1 - S'_3 - S'_5 \\ 2S'_3 - S'_1 - S'_5 \\ 2S'_5 - S'_3 - S'_1 \end{pmatrix} \quad (9)$$

The MSC and GSC are coupled through a common DC-link capacitor, which acts as the intermediate energy storage element between the generator-side and grid-side power conversion stages. In this configuration, the DC-link voltage is governed by the instantaneous current balance at the capacitor terminals. Therefore, assuming ideal converter switches and neglecting converter losses, the DC-link voltage dynamics can be expressed as:

$$\frac{dv_{DC}}{dt} = \frac{1}{C_{DC}} \left(i_{DC,\,\text{gen}} - i_{DC,\,\text{grid}} \right) \quad (10)$$

where C_{DC} is the DC-link capacitance, $i_{DC,gen}$ is the DC current supplied by the machine-side converter to the DC-link, and $i_{DC,\,\text{grid}}$ is the DC current drawn by the grid-side converter from the DC-link. With this sign convention, $i_{DC,\,\text{gen}} > i_{DC,\,\text{grid}}$ charges the capacitor and increases v_{DC}, whereas $i_{DC,\,\text{gen}} < i_{DC,\,\text{grid}}$ discharges the capacitor and decreases v_{DC}. This current-balance representation is appropriate for PMSG-based wind energy systems because the DC-link capacitor voltage directly reflects the power mismatch between the generator-side and grid-side converters, and its stabilization is a central requirement in full-scale converter-interfaced PMSG wind turbines [38].

The DC currents associated with each converter can be obtained from the corresponding phase currents and upper-switching functions. Let i_A, i_B, and i_C denote the MSC phase currents, and let i'_A, i'_B, and i'_C denote the GSC phase currents. Using the switching functions defined previously, the instantaneous DC-side currents are given by:

$$\begin{cases} i_{DC,gen} = S_1 i_A + S_3 i_B + S_5 i_C \\ i_{DC,grid} = S'_1 i'_A + S'_3 i'_B + S'_5 i'_C \end{cases} \quad (11)$$

where S_1, S_3, and S_5 correspond to the upper switches of the three MSC legs, while S'_1, S'_3, and S'_5 correspond to the upper switches of the three GSC legs. These expressions show that the DC-link current contribution of each converter is determined by the phase current connected to the positive DC rail at each switching instant. Consequently, the DC-link voltage is not an independent state but a dynamic variable determined by the instantaneous switching states and the power exchanged between the PMSG, the converter system, and the grid. This modeling approach is consistent with recent studies on PMSG wind turbine generators, where oscillating or unbalanced active power at the converter terminals is shown to propagate through the DC-link and produce voltage oscillations if not properly regulated.

2.2. Space Vector Pulse Width Modulation (SVPWM)

The SVPWM is an advanced technique used to control the output voltage of a three-phase inverter where the goal is to create a three-phase AC voltage with a specific amplitude and frequency. The technique provides better voltage use and less harmonic distortion compared to traditional modulation techniques. It controls the switching of the inverter's switches in such a way that it generates the desired output voltage waveform while minimizing the harmonic content.

The technique considers the space vector diagram of the inverter's output voltages. The space vector is a representation of the output voltage in a two-dimensional complex plane, where the voltage vectors are spaced 60° apart. The technique involves switching between different voltage vectors to generate the desired reference voltage.

The output of the three-phase inverter can be represented as a combination of 8 possible voltage vectors in the plane, with each vector corresponding to one of the 8 switching states of the inverter. These 8 voltage vectors are 6 active vectors (V1 to V6) that represent the voltages when two switches are ON, and one is OFF and vice versa, and 2 zero vectors (V0 and V7) that represent situations where all the upper or lower switches are simultaneously OFF or ON.

The space vector diagram results in a hexagon where the six active voltage vectors are positioned around the perimeter. The zero vectors are positioned at the center of the hexagon as depicted in Figure 3.

The three-phase voltages can be expressed in the $\alpha - \beta$ frame through the Clarke transform using the following relationship [43]:

$$\begin{pmatrix} v_\alpha \\ v_\beta \end{pmatrix} = \frac{2}{3} \begin{pmatrix} 1 & -1/2 & -1/2 \\ 0 & \sqrt{3}/2 & -\sqrt{3}/2 \end{pmatrix} \begin{pmatrix} v_{AN} \\ v_{BN} \\ v_{CN} \end{pmatrix} \quad (12)$$

The desired reference vector can be expressed as the following:

$$v_{ref} = \sqrt{{v_\alpha}^2 + {v_\beta}^2} \quad (13)$$

The module of the reference vector v_{ref} specifies the control of the output voltage waveform. With the value of the DC-bus voltage v_{DC}, it is also used to determine the modulation index m as in Equation (14).

$$m = \frac{3}{2}\frac{v_{ref}}{v_{DC}} \tag{14}$$

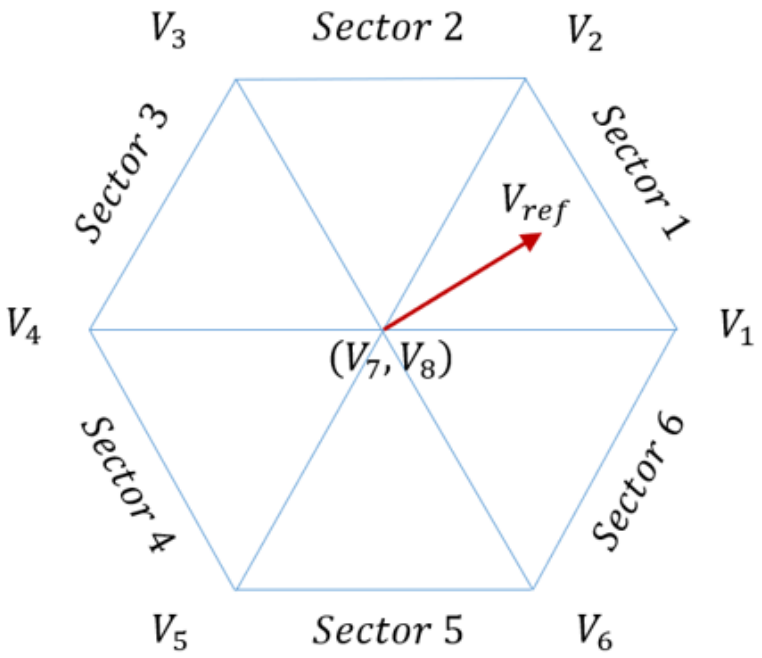

Figure 3.
Location of the reference in the space vector hexagon.

The reference voltage vector is decomposed into a combination of two adjacent active vectors and the duration of time each active vector needs to be applied. The reference vector lies between two of the active vectors in the space vector diagram, or sector. By calculating the time duration for each active vector and zero vector, the inverter switches in such a way that the resulting voltage approximates the reference.

The modulation strategy involves calculating how long each of the active and zero vectors should be applied. The two active vectors contribute to most of the reference voltage, and the zero vectors are used to balance the power and reduce harmonic distortion.

The switching times can be calculated geometrically by finding the time duration for each of the active vectors using the projection of the reference vector onto the adjacent active vectors in the space vector diagram. This will result in the following switching time formula:

$$\begin{aligned} T_1 &= \sqrt{3}T_s\frac{V_{\text{ref}}}{V_{DC}}\sin\left(\frac{n\pi}{3} - \theta\right) \\ T_2 &= \sqrt{3}T_s\frac{V_{\text{ref}}}{V_{DC}}\sin\left(\theta - \frac{(n-1)\pi}{3}\right) \\ T_0 &= T_s - (T_1 + T_2) \end{aligned} \tag{15}$$

T_1 and T_2 represent the dwell times assigned to the two adjacent active voltage vectors used to synthesize the reference voltage vector, whereas T_0 represents the dwell time assigned to the zero-voltage vector. The switching period is denoted by T_s. The sector number is denoted by $n = 1, \ldots, 6$, and θ represents the absolute angular position of the reference voltage vector in the stationary $\alpha - \beta$ reference frame, normalized within $[0, 2\pi]$.

2.3. Control Strategy

The adopted control strategy follows a simplified vector-control structure for the machine-side converter. The outer speed-control loop generates the quadrature-axis current reference $\boldsymbol{i_q^*}$, while the direct-axis current reference is set to zero, $\boldsymbol{i_d^* = 0}$, as commonly adopted for surface-mounted PMSG operation. As illustrated in Figure 4, the generated reference quantities are transformed into stationary-frame components before being processed by the SVPWM stage. In the implemented model, the SVPWM block uses these stationary-frame reference components to determine the converter switching states $\boldsymbol{S_1}$ to $\boldsymbol{S_6}$, which are then applied to the machine-side converter. It should be noted that Figure 4 provides a simplified signal-flow representation of the control and modulation process; the detailed current regulation and voltage-reference generation stages are represented in lumped form. The speed controller is implemented using a PID block with the derivative gain set to zero; therefore, its effective behavior is proportional-integral. The controller gains are $\boldsymbol{K_p = 10}$, $\boldsymbol{K_i = 40}$, and $\boldsymbol{K_d = 0}$.

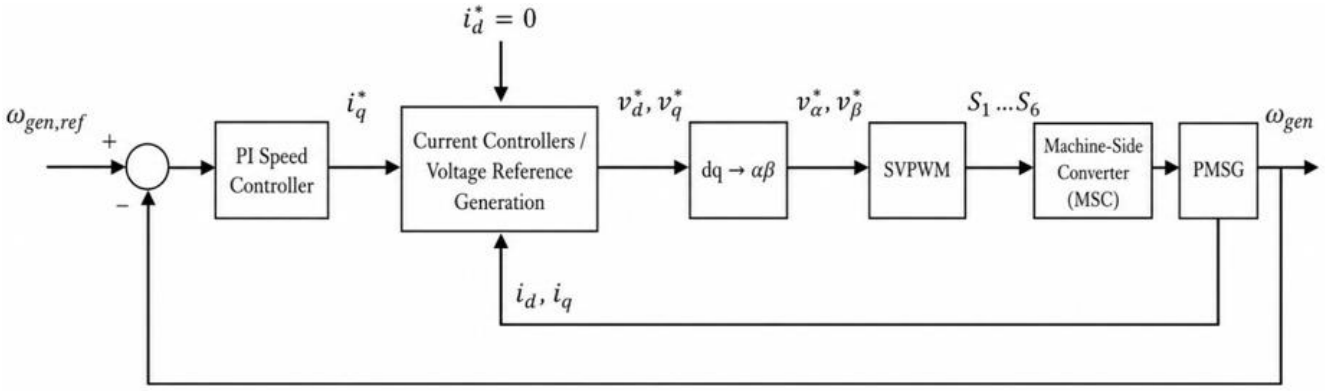

Figure 4.
The block diagram of the control strategy.

2.4. Structure of the System

The following describes the communication method used in an Arduino-based SVPWM switching logic system for testing an SVPWM control algorithm. The objective is to run the SVPWM calculation on a real microcontroller (Arduino UNO) while leaving the PMSG simulation and the generation of reference signals to Simulink simulation. In the proposed structure, the plant model remains in simulation, whereas the converter switching logic is executed on embedded hardware. The data flow is organized as follows:

- Simulink simulates the PMSG model and generates the voltage-reference components.
- The Arduino UNO receives the stationary-frame voltage references (v_α, v_β), calculates the SVPWM timing variables and the corresponding switching states S_1 to S_6, and sends the encoded switching information back to Simulink in one byte.
- Simulink decodes this byte and applies the corresponding switching commands to the simulated machine-side converter.

In the proposed implementation, the Arduino routine computes the SVPWM sector, dwell-time variables, and corresponding switching-state information. However, because the Universal Asynchronous Receiver/Transmitter (UART) exchange time exceeds the PWM carrier period, the Simulink converter does not receive a carrier-synchronous sequence of intra-period switching transitions. Instead, the returned switching byte is decoded and held until the next serial update. Therefore, the proposed setup validates the embedded SVPWM switching-state computation and the serial hardware-coupled data path, rather than full cycle-by-cycle SVPWM gate-pulse generation.

It should be noted that the serial communication loop and the PWM carrier period are not identical time scales in the proposed setup. The PWM period used in the SVPWM timing calculation is $T_s = 50.5\,\mu s$, corresponding to 19.8 kHz. However, the serial communication link operates at $500{,}000\ \text{bit/s}$. With a standard UART as serial communication link frame of approximately 10 bits per byte, one byte requires about 20μ s. Therefore, transmitting two single-precision $\alpha - \beta$ values require at least 8 bytes, or approximately 160μ s, before considering the returned switching byte, Simulink overhead, buffering, and microcontroller execution time. Consequently, the serial exchange cannot be interpreted as a complete per-PWM-period communication cycle. In this work, the serial link is used as a hardware-coupled validation interface, while the exchanged switching information is applied at the communication update rate.

The main advantage of this structure is that it enables the embedded SVPWM code to be evaluated under practical numerical, execution, formatting, and serial communication constraints without requiring a physical generator, converter, or motor test bench. It also allows deviations between the ideal Simulink model and the Arduino-based implementation to be observed, including the effects of finite precision, rounding, latency, and data formatting. The serial communication and SVPWM parameters are summarized in Tables 1 and 2, respectively.

Table 1.
Serial block parameters.

Parameter	Value	Description
Port	COMXX	Serial port of Arduino UNO, determined from the device manager
Baud Rate	500,000	Serial communication speed
Data Bits	8	Word data size
Parity	None	No parity bit
Stop Bits	1	One stop bit
Byte Order	Little-endian	Byte order used for multi-byte data transmission

Table 2.
SVPWM parameters.

Parameter	Value	Description
Tz	1.0 / 19800.0 ≈ 50.5 μs	PWM period equal to 1/ (19.8 kHz)
VDC	655 V	Voltage of DC-bus
PWM_TOP	404	Maximum value of the timer counter (Center-aligned)

From Tables 1 and 2, the UART communication time exceeds one PWM carrier period when the $\alpha - \beta$ references are exchanged as single-precision values. At $500{,}000$ bit/s, transmitting two single-precision reference values and receiving one encoded switching byte requires at least 180μ s, excluding software and execution overhead. This is longer than the 50.5 μs PWM period. Therefore, the proposed setup should not be interpreted as a cycle-by-cycle PWM-synchronous HIL implementation; the latest received switching information is held in the Simulink converter model until the next serial update.

The block diagram of the solution is depicted in Figure 5 with the data flow between the simulation and hardware blocks.

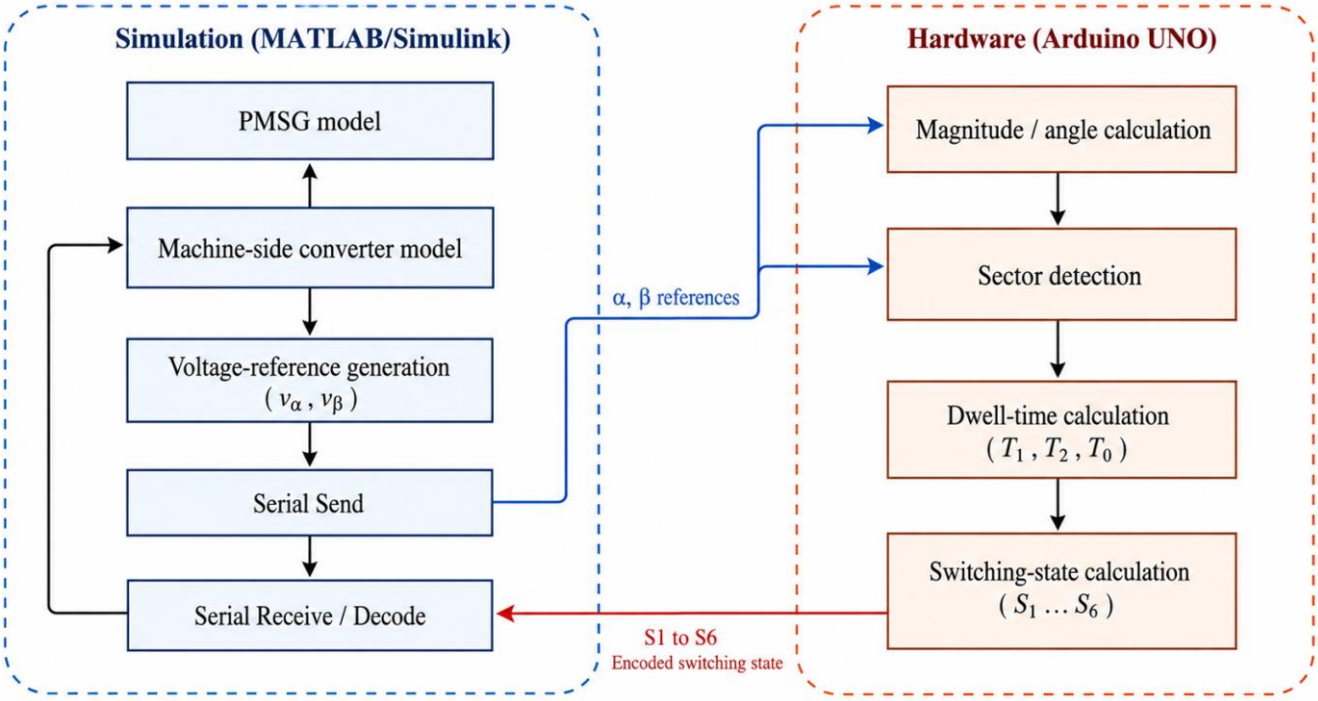

Figure 5.
The block diagram of the proposed control implementation.

3. Results and Discussion

The proposed simulation and hardware-coupled validation framework was implemented in MATLAB/Simulink R2023b. The PMSG-based horizontal-axis wind turbine energy conversion system was kept in Simulink, including the aerodynamic model, generator dynamics, back-to-back converter, *d-q* current-control loops, and speed-control loop. In contrast, the space-vector pulse-width modulation switching-state computation was assigned to an Arduino UNO microcontroller, so that the modulation stage could be assessed under the execution and communication constraints of low-cost embedded hardware. This separation was used to compare the simulation-only implementation and the embedded SVPWM switching logic implementation under the same model structure and operating conditions. The results in this section therefore examine the effect of replacing the ideal Simulink SVPWM block with the physical microcontroller-based SVPWM routine. The comparison is based on the generator speed response, the d- and q-axis current components, and the d- and q-axis voltage control signals, as reported in Figures 7–9. These variables were selected because they capture the main mechanical, electrical, and control-level responses of the closed-loop PMSG system.

Figure 6 illustrates the serial data-exchange architecture used to couple the MATLAB/Simulink environment with the Arduino UNO embedded hardware. In Simulink, the voltage-reference generation block produces the stationary-frame components v_α and v_β from the $\mathrm{d} - \mathrm{q}$ current controllers and inverse transformation; these signals are multiplexed, converted to single precision, and transmitted through the Serial Send block on COM3. The UART serial link is configured at 500,000 bit/s with 8 data bits, no parity, one stop bit, and little-endian byte ordering; therefore, the transmitted reference packet contains two single-precision values, corresponding to at least 8 bytes. On the Arduino side, the serial receives buffer and unpacking stage recover the floating-point references and feed the embedded SVPWM routine. This routine calculates $|V_{\mathrm{ref}}|$ and θ, identifies the sector $n = 1 - 6$, computes the dwell times T_1, T_2, and T_0, selects the corresponding

switching state, and packs $S_1 - S_6$ into one encoded byte. The encoded byte is returned through the serial transmit buffer, decoded in Simulink by the integer-to-bit converter, held by the zero-order hold between serial updates, and applied as the gate vector to the simulated machine-side converter. Because the UART exchange time exceeds the 50.5μ s PWM carrier period, the interface is used for serial hardware-coupled validation rather than cycle-by-cycle PWM-synchronous HIL operation.

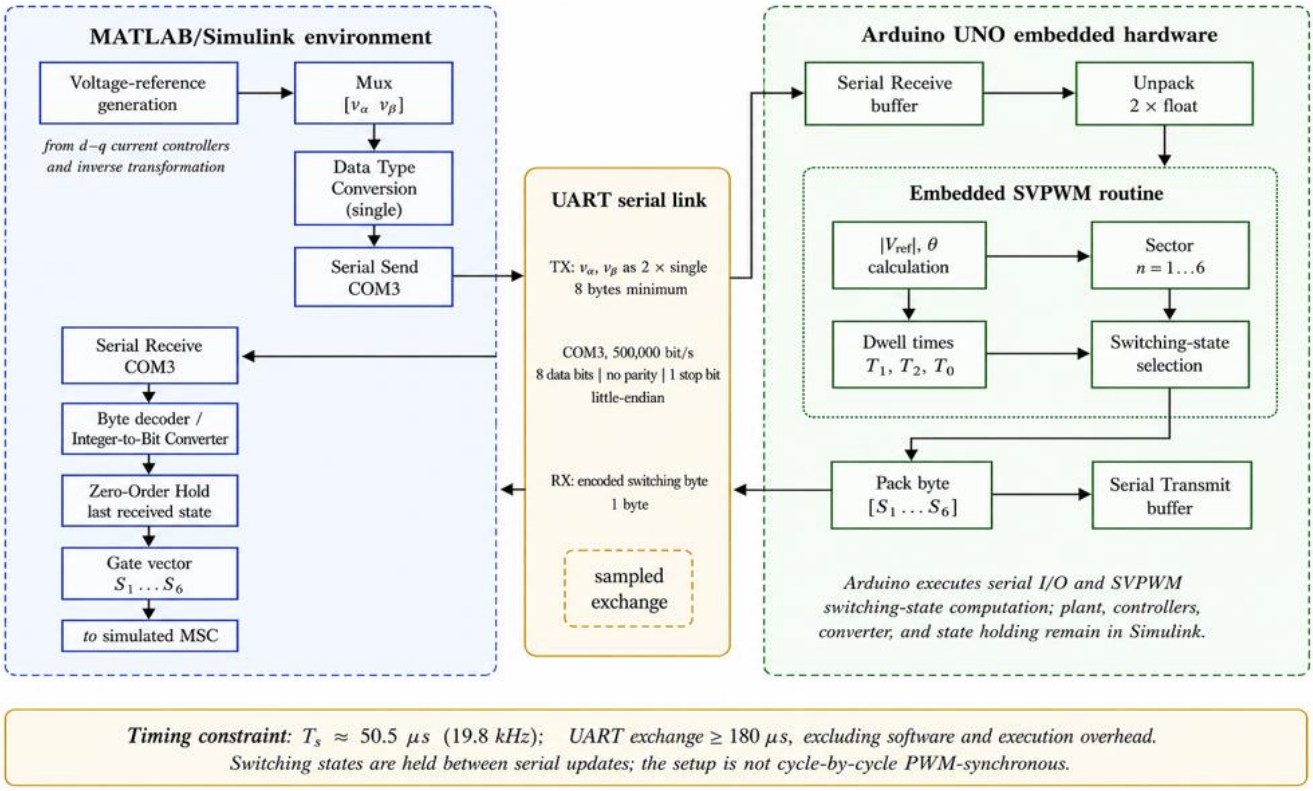

Figure 6.
Serial data-exchange architecture between the MATLAB/Simulink model and the Arduino UNO board.

3.1. Results Presentation

Because the proposed Arduino-coupled SVPWM implementation is intended to determine whether the embedded SVPWM routine can preserve the closed-loop dynamic behavior of the PMSG system, the generator speed response is used first as a system-level performance indicator. This variable is not only a mechanical output, since it also reflects the combined action of the outer speed controller, the inner current-control loops, the machine-side converter, and the SVPWM switching-state generation. For this reason, an implementation error introduced by the embedded modulation stage would be expected to appear in the speed response as a tracking deviation, additional delay, overshoot, or sustained oscillation. To isolate the effect of the embedded modulation stage, the plant model, controller parameters, DC-link voltage, speed-reference profile, and simulation time window were kept identical in both implementations. The only deliberate change was the transfer of the SVPWM switching-state computation from the Simulink environment to the Arduino UNO microcontroller.

Under this evaluation criterion, the response obtained with the conventional Simulink-based SVPWM block is compared with the response obtained when the SVPWM computation is executed on the Arduino UNO through the Arduino-based SVPWM switching logic interface. The comparison, reported in Figure 7, uses an initial reference speed of 100 rad/s, followed by an increase to 150 rad/s at 0.1 s, so that both the initial acceleration and the reference-step transition can be examined. In both cases, the generator speed follows the imposed reference with closely similar dynamic behavior. The embedded SVPWM switching logic response remains close to the simulation-only response during the start-up interval, after the speed change, and in the steady-state region. The enlarged views also show that the remaining deviations are small, although they are not completely absent, which is expected because the Arduino-based SVPWM switching logic path includes microcontroller execution time, finite numerical precision, data formatting, and serial communication latency. Within the tested operating condition, this agreement indicates that the transfer of the SVPWM switching-state computation from Simulink to the Arduino UNO does not materially alter the dominant speed-control dynamics of the PMSG system.

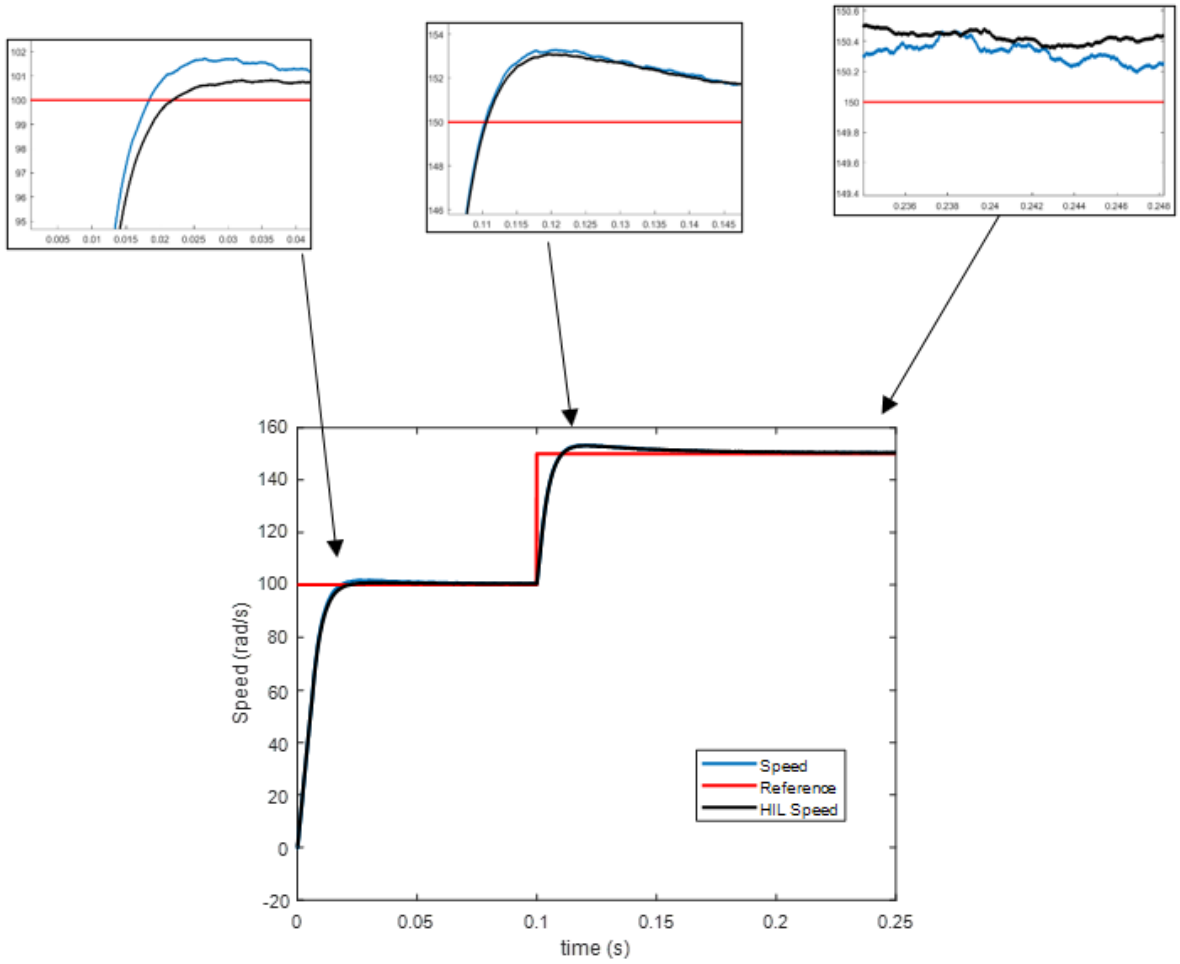

Figure 7.
Generator-speed response using Simulink SVPWM and Arduino-coupled SVPWM switching-state computation.

After the system-level speed-tracking response has been verified, the validation must also examine the electrical behavior of the Permanent Magnet Synchronous Generator, because errors in the embedded SVPWM routine would be expected to appear first in the current-control dynamics. In the adopted *d-q* vector-control scheme, the direct-axis current is associated with the flux-related channel and is normally regulated around its reference value, whereas the quadrature-axis current provides the main torque-producing contribution requested by the outer speed-control loop. A credible SVPWM implementation should therefore reproduce not only the mechanical speed response, but also the current trajectories enforced by the inner controllers. Figure 8 compares the d- and q-axis current components obtained with the simulation-only SVPWM block and with the Arduino-based implementation. The transient observed at 0.1 s is caused by the imposed increase in the speed reference, which changes the torque demand and is consequently reflected mainly in the q-axis current. For both current components, the proposed SVPWM waveforms remain close to the corresponding simulation-only results, suggesting that the embedded SVPWM switching-state computation does not introduce a noticeable distortion in the current-control behavior under the tested condition. The enlarged views are useful because they expose local deviations between the two implementations over short time intervals. These deviations are consistent with embedded computation, finite-precision arithmetic, byte-level data exchange, serial communication delay, and sample-hold behavior between communication updates. Despite these implementation-level effects, the d- and q-axis currents remain bounded and dynamically consistent with the simulation-only case, which supports the conclusion that the proposed arrangement preserves the essential electrical response of the controlled PMSG system for the considered operating scenario.

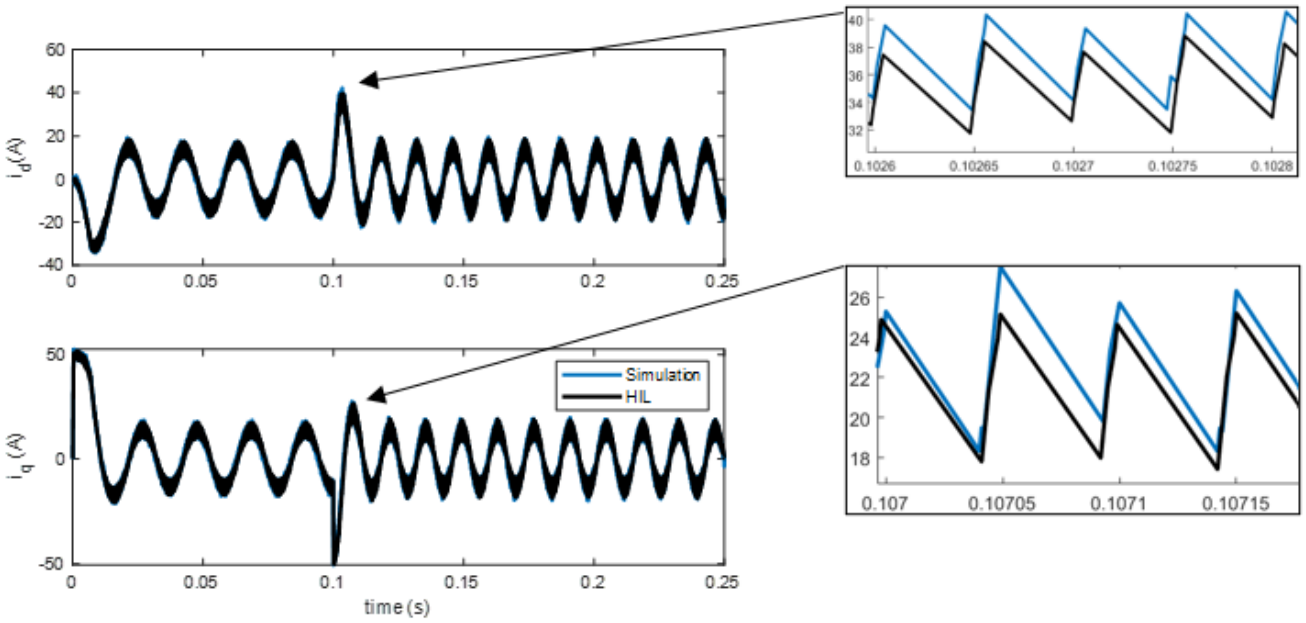

Figure 8.
d–q current components under Simulink and Arduino-coupled SVPWM.

A final comparison is needed on the voltage-command side, since the *d-q* axis voltage components are the immediate outputs of the inner current controllers and form the reference signals that the SVPWM stage must synthesize. The speed response gives a system-level indication of mechanical tracking, and the current waveforms describe the electrical control behavior, but the voltage components are closer to the modulation process itself. For this reason, Figure 9 is used to examine whether transferring the SVPWM switching-state computation from the ideal MATLAB/Simulink environment to the proposed interface changes the voltage behavior imposed on the machine-side converter. The obtained v_d and v_q waveforms reflect the switching behavior imposed by the simulated converter model using the switching information received from the Arduino-based SVPWM routine. Because the serial update rate is lower than the PWM carrier frequency, these voltage responses should be interpreted as evidence of closed-loop consistency under the adopted serial communication-coupled embedded SVPWM setup, not as proof of cycle-by-cycle PWM timing equivalence. Across the full simulation interval, including the transient around the speed-reference change at $0.1s$, the simulation-only and proposed SVPWM implementation responses remain closely aligned. This agreement suggests that the embedded SVPWM routine reproduces the voltage-synthesis behavior required by the current controllers, even with the finite-precision computation and serial communication constraints introduced by the microcontroller. Taken together with the speed and current results, the voltage comparison indicates that the proposed Arduino-coupled SVPWM arrangement preserves the main control and modulation characteristics of the simulated PMSG system under the considered operating condition.

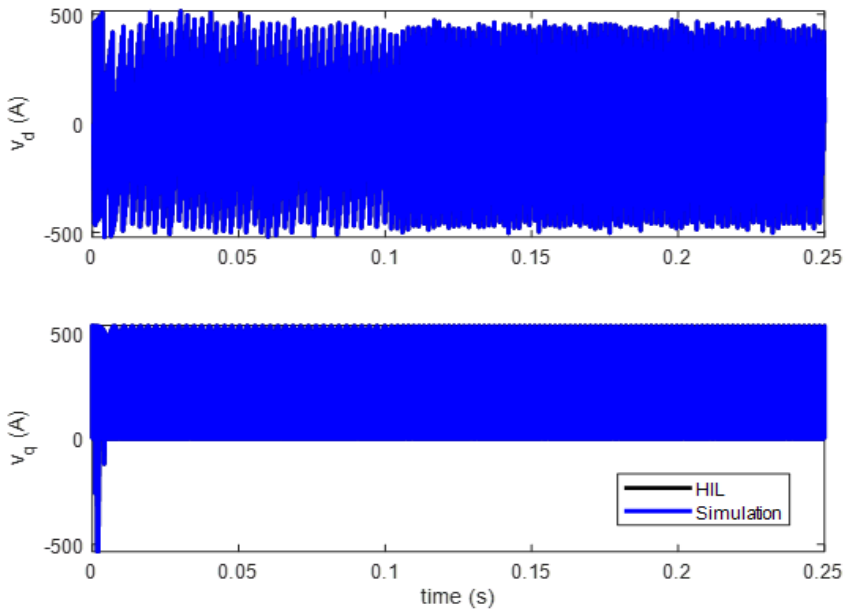

Figure 9.
d–q voltage control signals for Simulink and Arduino-coupled SVPWM.

Taken together, these three result groups provide a layered assessment of the proposed implementation of SVPWM. The speed response verifies that the system-level mechanical dynamics are largely preserved, the d- and q-axis current components confirm the consistency of the inner current-control behavior, and the voltage-control signals provide a closer indication of the modulation action applied to the machine-side converter. The close agreement between the simulation-only and proposed Arduino-coupled SVPWM results across these levels indicates that the Arduino-based SVPWM switching-state computation can reproduce the dominant behavior of the ideal Simulink implementation for the tested operating condition. At the same time, the small deviations visible in the enlarged views and high-frequency electrical waveforms reflect implementation-dependent effects, including embedded execution, finite numerical precision, byte-level data exchange, and serial communication latency. These observations provide the basis for the following discussion, where the effectiveness, interpretation, and limitations of the proposed Arduino-in-the-loop validation are examined in more detail.

3.2. Discussion

The results indicate that the proposed serial communication-coupled embedded SVPWM framework implementation preserves the main closed-loop behavior of the simulation-only SVPWM-controlled PMSG system under the tested operating condition. This conclusion refers to the dominant speed, current, and voltage trends observed at the simulation level and should not be interpreted as verification of per-PWM-cycle real-time synchronization. The SVPWM algorithm was not reformulated theoretically. Instead, the switching-state computation was transferred to an Arduino UNO microcontroller and then coupled with the closed-loop Simulink model of the PMSG wind turbine system. This transfer is not a trivial coding substitution, since the modulation stage is no longer evaluated only under ideal numerical and timing assumptions. It is exposed to embedded-implementation constraints, including finite numerical precision, limited processing capacity, byte-level data exchange, serial communication latency, and rounding effects introduced by the hardware path.

This distinction matters because the modulation block is the practical interface between the control algorithm and the converter switching action. In a PMSG-based wind energy conversion system, errors at this interface can affect the voltage applied to the generator, the current-control response, the electromagnetic torque, and, through these variables, the mechanical speed dynamics. For that reason, the comparison performed here should not be interpreted only as a confirmation that the Arduino code executes. The more relevant question is whether switching states generated by the physical microcontroller can be inserted into the closed-loop PMSG model without disturbing the expected behavior of the controlled system.

The rotor-speed response gives the first system-level indication of this consistency. Rotor speed reflects the combined effect of aerodynamic input, electromagnetic torque, the outer speed controller, the inner current controllers, the converter switching states, and the SVPWM process. It is therefore an integrated, relatively low-bandwidth indicator of closed-loop behavior. A sufficiently large timing error or communication delay may appear as degradation in the speed response, for example increased tracking error, delayed transient behavior, overshoot, oscillation, or steady-state offset. However, because rotor-speed dynamics are relatively slow, speed response alone cannot verify cycle-level PWM timing accuracy. The comparison in Figure 7 does not show such degradation. Both the simulation-only and proposed Arduino-coupled SVPWM implementations track the reference profile, including the transition from 100 rad/s to 150 rad/s at 0.1 s, with closely comparable transient and steady-state behavior. Within this tested speed-reference variation, the Arduino-based modulation stage therefore does not materially change the dominant mechanical response of the controlled PMSG system.

Speed response alone, however, cannot validate the embedded modulation process. Mechanical dynamics are slower than electrical dynamics, and they may conceal some modulation-level deviations. The *d-q* current responses therefore provide a more sensitive view of how the embedded SVPWM implementation affects the inner electrical control loops. In the adopted vector-control strategy, the direct-axis current is regulated around its reference, whereas the quadrature-axis current supplies the torque-producing component demanded by the speed controller. The current trajectories are consequently linked both to current-controller performance and to the voltage synthesis produced by the modulation stage. As shown in Figure 8, the proposed Arduino-coupled SVPWM implementation d- and q-axis currents remain dynamically consistent with the simulation-only currents. The transient near 0.1 s is physically expected, because the increase in speed reference requires a corresponding change in torque-producing current. Exact waveform identity is not expected once the hardware path is introduced. The relevant result is that boundedness, dynamic evolution, and control action are preserved. The small differences visible in the enlarged views are consistent with embedded computation time, finite-precision arithmetic, serial transmission, byte formatting, and decoding before the switching commands are applied to the simulated converter.

The voltage-control signals provide a closer view of the modulation-related behavior. The voltage components are the immediate outputs of the current controllers and define the reference voltage vector that the SVPWM routine must synthesize. Their comparison is therefore directly relevant to whether the embedded implementation preserves the voltage-generation behavior required by the control system. As shown in Figure 9, the proposed Arduino-coupled SVPWM implementation voltage responses remain closely aligned with the simulation-only responses, including during the transient caused by the speed-reference step. The dense high-frequency appearance of these waveforms is consistent with voltage-source converter switching and, by itself, should not be read as evidence of instability. The agreement between the two implementations instead suggests that the Arduino-based SVPWM routine reproduces the required modulation behavior with sufficient consistency for the considered operating condition.

Viewed together, the speed, current, and voltage comparisons provide a layered validation of the implementation. The speed response supports preservation of the global electromechanical behavior. The *d-q* currents indicate that the inner electrical control loops remain consistent. The *d-q* voltage signals further show that the modulation-related behavior remains compatible with the voltage demands imposed by the current controllers. Agreement across these levels supports the conclusion that the embedded SVPWM routine has been integrated correctly into the closed-loop PMSG wind turbine model for the considered test case.

The practical value of the proposed framework lies in its position between offline simulation and complete experimental testing. In a conventional simulation, the SVPWM block operates under ideal numerical and timing conditions. Such a setting is useful for control design, but it cannot show whether the same algorithm remains acceptable when executed on physical digital hardware. A complete experimental platform, by contrast, would require a wind turbine emulator, PMSG, power converter, sensors, gate drivers, protection circuitry, and grid-interface equipment, which increases cost, instrumentation effort, and safety requirements. proposed Arduino-coupled SVPWM implementation offers an intermediate validation stage. The modulation code runs on a real low-cost microcontroller, while the PMSG wind energy conversion system remains in a flexible and safe simulation environment. This makes it possible to detect implementation-related problems before moving to more demanding real-time platforms or power-hardware experiments.

This point is especially relevant for SVPWM, because its practical behavior depends not only on the modulation equations but also on their digital realization. Sector identification, dwell-time calculation, switching-state selection, scaling, rounding, and timing synchronization must all be implemented consistently. An error in any of these steps can generate incorrect converter switching commands even when the control law itself is properly designed. The present results show that an Arduino-class microcontroller can reproduce the expected switching-state behavior with sufficient consistency for embedded SVPWM validation in the considered PMSG-based horizontal-axis wind turbine application.

The scope of this validation should nevertheless remain bounded. The proposed Arduino-coupled SVPWM implementation setup is not a full experimental validation of the complete wind energy conversion system. The wind turbine, PMSG, back-to-back converter, DC-link, and grid-side elements are still represented by simulation models. Several physical non-idealities are therefore not fully captured, including semiconductor switching delays, dead-time effects, gate-driver limitations, sensor noise, Analog-to-Digital Converter quantization, thermal behavior, electromagnetic interference, parasitic elements, protection circuits, and real grid disturbances. These effects may become important in a laboratory prototype or field implementation. The present results demonstrate the consistency of embedded SVPWM switching-state computation within the tested proposed SVPWM implementation configuration, but they should not be interpreted as evidence of complete converter-hardware readiness or grid-connected system readiness.

Another limitation is that the comparison remains mainly waveform-based. Although the graphical agreement between the simulation-only and the proposed serial hardware-coupled SVPWM validation framework supports the feasibility of the proposed approach, a stronger validation would require quantitative performance indicators. Future work should include metrics such as maximum speed-tracking error, steady-state speed error, settling time, Root Mean Square current deviation, current-ripple amplitude, voltage deviation, and harmonic distortion indices. Additional operating scenarios should also be examined, including variable wind-speed profiles, multiple speed-reference changes, DC-link voltage disturbances, grid-side perturbations, load variations, PMSG parameter mismatch, and operation near converter voltage limits.

Overall, the obtained results support the main objective of the study. The Arduino-based SVPWM implementation reproduces the essential behavior of the ideal Simulink implementation for the considered operating condition, while introducing physical microcontroller execution into the modulation path. The proposed framework can therefore serve as a practical validation tool for embedded SVPWM control in PMSG-based horizontal-axis wind turbine energy conversion systems. Its role is not to replace high-fidelity real-time simulators or full experimental test benches. Its value is to provide an accessible step for checking the digital implementation of the modulation layer before more advanced hardware validation is attempted.

4. Conclusion

This paper presented a serial co-simulation-based framework for the embedded implementation of Space Vector Pulse Width Modulation in a Permanent Magnet Synchronous Generator based horizontal-axis wind turbine energy conversion system. The aerodynamic turbine model, PMSG, back-to-back converter, speed-control loop, and current-control loops were kept in MATLAB/Simulink, whereas the SVPWM switching-state computation was transferred to an Arduino UNO microcontroller and interfaced with the simulated converter through serial communication. The purpose was deliberately limited. The study did not seek to propose a new modulation method or a new speed-control strategy, but rather to verify whether moving the SVPWM computation from an ideal simulation block to physical embedded hardware would preserve the expected closed-loop behavior of the controlled PMSG system.

The comparison between the simulation-only and Arduino-in-the-loop validation cases showed close agreement under the tested operating condition. The generator speed response followed the same general behavior during initial acceleration, during the reference change from 100 rad/s to 150 rad/s, and during the subsequent steady-state intervals. This suggests that the Arduino-based SVPWM routine did not materially disturb the dominant electromechanical dynamics of the closed-loop system in this case. The d-axis and q-axis current responses also remained dynamically consistent with the simulation-only implementation, despite the added effects of finite-precision computation, byte-level data exchange, embedded execution time, and serial communication latency. The *d-q* voltage-control signals showed comparable behavior in both implementations, indicating that the embedded SVPWM routine preserved the main voltage-control response required by the current controllers under the adopted serial communication-coupled embedded SVPWM configuration. This result

should not be interpreted as evidence of cycle-by-cycle PWM timing equivalence, because the serial communication update time is longer than the 50.5 μs PWM carrier period.

The contribution of the work is therefore practical rather than algorithmic. The proposed framework offers an intermediate validation stage for the embedded SVPWM switching-state computation path between offline simulation and full experimental testing of a PMSG-based wind energy conversion system. By assigning the SVPWM switching-state computation to a real low-cost microcontroller, the setup makes several implementation-level effects visible, including sector identification, dwell-time calculation, switching-state generation, scaling, rounding, data formatting, and communication delay. Within the considered operating condition, the results indicate that an Arduino-class microcontroller can serve as an embedded validation platform for SVPWM implementation in a simulated PMSG-based HAWT system.

The validation scope remains intentionally bounded. The turbine, PMSG, power converters, DC-link, and grid-side elements were still represented by simulation models, so the setup does not fully account for semiconductor switching delays, dead-time effects, gate-driver constraints, sensor noise, analog-to-digital converter quantization, thermal behavior, electromagnetic interference, parasitic elements, protection circuits, or real grid disturbances. Future work should extend the assessment to variable wind-speed profiles, multiple speed-reference changes, DC-link voltage disturbances, grid-side perturbations, load variations, machine-parameter mismatch, and operation near converter voltage limits. Quantitative indicators, including maximum speed-tracking error, steady-state error, settling time, root mean square current deviation, current ripple, voltage deviation, and harmonic distortion indices, should also be reported. Overall, the Arduino-based framework provides an accessible preliminary validation tool for embedded SVPWM logic in PMSG-based HAWT energy conversion systems. Its validity is limited to preliminary serial communication-coupled embedded SVPWM validation assessment and should be extended with measured latency analysis, faster communication hardware, or real-time HIL platforms before claiming carrier-synchronous or cycle-by-cycle PWM validation.

References

[1] F. Merahi, H. Mernache, D. Aouzelag, A. E. Badoud, M. Bajaj, and I. Zaitsev, "Power control of an autonomous wind energy conversion system based on a permanent magnet synchronous generator with integrated pumping storage," *Scientific Reports,* vol. 14, no. 1, p. 29776, 2024. https://doi.org/10.1038/s41598-024-81522-8

[2] A. Bouchakour *et al.*, "MPPT algorithm based on metaheuristic techniques (PSO & GA) dedicated to improve wind energy water pumping system performance," *Scientific Reports,* vol. 14, no. 1, p. 17891, 2024. https://doi.org/10.1038/s41598-024-68584-4

[3] G. E. Barter, L. Sethuraman, P. Bortolotti, J. Keller, and D. A. Torrey, "Beyond 15 MW: A cost of energy perspective on the next generation of drivetrain technologies for offshore wind turbines," *Applied Energy,* vol. 344, p. 121272, 2023. https://doi.org/10.1016/j.apenergy.2023.121272

[4] B. Majout *et al.*, "A review on popular control applications in wind energy conversion system based on permanent magnet generator PMSG," *Energies,* vol. 15, no. 17, p. 6238, 2022. https://doi.org/10.3390/en15176238

[5] H. Liu, Y. Ren, Q. Meng, B. He, R. Xu, and C. Fang, "Frequency regulation capability improvement control strategy for grid-forming direct-drive wind turbines," *International Journal of Energy Research,* vol. 2026, no. 1, p. 2248156, 2026. https://doi.org/10.1155/er/2248156

[6] D. Abdeljalil, M. Krichen, N. Benhalima, N. Benhadj, and R. Neji, "Performance driven multi objective optimization of 2 MW integrated Pseudo Direct Drive permanent magnet synchronous wind generator," *Scientific Reports,* vol. 16, p. 10130, 2026. https://doi.org/10.1038/s41598-026-40096-3

[7] J. Zhang and J. Li, "Hybrid deloading control strategy in mmc-based wind energy conversion systems for enhanced frequency regulation," *Energies,* vol. 17, no. 5, p. 1253, 2024. https://doi.org/10.3390/en17051253

[8] A. A. A. Radwan, M. A. Elshenawy, Y. A.-R. I. Mohamed, and E. F. El-Saadany, "Grid-forming current-source converter for a full-scale wind energy conversion system," *IEEE Open Journal of Power Electronics,* vol. 6, pp. 327-343, 2025. https://doi.org/10.1109/OJPEL.2025.3537939

[9] S. Liu, R. G. Cirstea, H. Wu, T. Bosma, and X. Wang, "Comparative evaluation of converter control impact on torsional dynamics of type-IV grid-forming wind turbines," *IEEE Transactions on Sustainable Energy,* vol. 15, no. 4, pp. 2803-2814, 2024. https://doi.org/10.1109/TSTE.2024.3444474

[10] M. Ishaque, J. A. Laghari, M. A. Bhayo, S. Chandio, and I. Mahariq, "Hybrid ANFIS-PI-based robust control of wind turbine power generation system," *International Transactions on Electrical Energy Systems,* vol. 2024, no. 1, p. 2389751, 2024. https://doi.org/10.1155/2024/2389751

[11] H. Benbouhenni *et al.*, "Enhancing the power quality of dual rotor wind turbines using improved fuzzy space vector modulation and super twisting sliding techniques," *Scientific Reports,* vol. 15, no. 1, p. 7290, 2025. https://doi.org/10.1038/s41598-025-90914-3

[12] N. Kumar and B. Sridhar, "Fault ride-through capability enhancement of grid-integrated PMSG wind energy conversion systems," *International Journal for Multidisciplinary Research,* vol. 7, no. 5, 2025. https://doi.org/10.36948/ijfmr.2025.v07i05.55132

[13] S. Shuvo, E. Hossain, and Z. R. Khan, "Fixed point implementation of grid tied inverter in digital signal processing controller," *IEEE Access,* vol. 8, pp. 89215-89227, 2020. https://doi.org/10.1109/ACCESS.2020.2993985

[14] A. G. Abo-Khalil and M. Alobaid, "Optimized control for PMSG wind turbine systems under unbalanced and distorted grid voltage scenarios," *Sustainability,* vol. 15, no. 12, p. 9552, 2023. https://doi.org/10.3390/su15129552

[15] A. Talaoubrid, Y. Ait Gougam, R. Dermouche, and N. Zioui, "Experimental comparison of the performance of PI and IP controllers for a field-oriented controlled permanent magnet synchronous motor drive," *International Journal of Dynamics and Control,* vol. 12, no. 8, pp. 2918-2928, 2024. https://doi.org/10.1007/s40435-024-01395-7

[16] H. Ren, C. Gan, K. Ni, H. Shi, C. Zhang, and R. Qu, "Flexible active vector distribution strategy for current ripple and switching loss reduction in three-phase voltage source inverter," *IEEE Transactions on Industry Applications,* vol. 61, no. 2, pp. 3234-3245, 2024. https://doi.org/10.1109/TIA.2024.3523462

[17] N. Zioui, A. Mahmoudi, M. Fazilat, O. Kone, D. Reda, and M. Tadjine, "Quantum space vector pulse width modulation for speed control of permanent magnet synchronous machines," *e-Prime-Advances in Electrical Engineering, Electronics and Energy,* vol. 13, p. 101074, 2025. https://doi.org/10.1016/j.prime.2025.101074

[18] A. Ammar, H. Y. Kanaan, N. Moubayed, M. Hamouda, and K. Al-Haddad, "Original approach toward three-phase indirect matrix converters through hybrid PWM modulation and DSP implementation," *IEEE Access,* vol. 8, pp. 45837-45852, 2020. https://doi.org/10.1109/ACCESS.2020.2977502

[19] H. Abouobaida *et al.*, "A three-level inverter-based model predictive control design for optimal wind energy systems," *IEEE Access,* vol. 13, pp. 42414-42427, 2025. https://doi.org/10.1109/ACCESS.2025.3547996

[20] A. Kumar, S. Prakash, S. Ghosh, N. S. Naidu, and P. Dworak, "First-order dynamic controller design for grid-tied permanent magnet synchronous generator," *IEEE Journal of Emerging and Selected Topics in Power Electronics,* vol. 14, no. 2, pp. 1600 - 1614, 2026. https://doi.org/10.1109/JESTPE.2026.3656931

[21] Y. E. Mourabit *et al.*, "Enhanced performance in PMSG-based wind turbine systems: Experimental validation of adaptive backstepping control design," *Energies,* vol. 16, no. 22, p. 7481, 2023. https://doi.org/10.3390/en16227481

[22] H. Benbouhenni, M. Yessef, N. Bizon, B. Bossoufi, and T. A. Alghamdi, "Experimental assessment of a dual super-twisting control technique of variable-speed multi-rotor wind turbine systems," *IEEE Access,* vol. 12, pp. 103744-103763, 2024. https://doi.org/10.1109/ACCESS.2024.3434534

[23] M. Shayeghan, M. Di Benedetto, A. Lidozzi, and L. Solero, "HIL-based fault-tolerant vector space decomposition control for a six-phase PMSM fed by a five-level CHB converter," *Energies,* vol. 18, no. 3, p. 507, 2025. https://doi.org/10.3390/en18030507

[24] Y. O. Cota and A. J. Sguarezi Filho, "Finite control set MPC for DFIG direct power control in distorted grids," *IEEE Access,* vol. 14, pp. 41753 - 41763, 2026. https://doi.org/10.1109/ACCESS.2026.3671001

[25] D. Wu *et al.*, "Grid integration of offshore wind power: Standards, control, power quality and transmission," *IEEE Open Journal of Power Electronics,* vol. 5, pp. 583-604, 2024. https://doi.org/10.1109/OJPEL.2024.3390417

[26] D. Reda, A. Talaoubrid, M. Fazilat, N. Zioui, and M. Tadjine, "A quantum direct torque control method for permanent magnet synchronous machines," *Computers and Electrical Engineering,* vol. 122, p. 109994, 2025. https://doi.org/10.1016/j.compeleceng.2024.109994

[27] A. A. M. Al-Dwa *et al.*, "Real-time implementation of a new modified 3DSVPWM control method for eliminating zero-sequence circulating current in parallel three-phase four-leg source voltage inverters," *IEEE Access,* vol. 12, pp. 101121-101138, 2024. https://doi.org/10.1109/ACCESS.2024.3430979

[28] S. Divya and D. K. Kumar, "Design and implementation of direct torque control of IPMSM drive using three-level inverter and FPGA for electric vehicle," *International Journal of Electrical and Electronics Research,* vol. 13, no. 1, pp. 10-16, 2025. https://doi.org/10.37391/IJEER.130102

[29] S. Saidat, R. Boumekhita, M. Tadjine, and N. Zioui, "Quantum pulse-width modulation design and implementation for a DC motor drive," *Quantum Information Processing,* vol. 23, no. 3, p. 88, 2024. https://doi.org/10.1007/s11128-024-04284-2

[30] H. A. Madsen, T. J. Larsen, G. R. Pirrung, A. Li, and F. Zahle, "Implementation of the blade element momentum model on a polar grid and its aeroelastic load impact," *Wind Energy Science,* vol. 5, no. 1, pp. 1-27, 2020. https://doi.org/10.5194/wes-5-1-2020

[31] M. A. Bhayo *et al.*, "An experimental hybrid control approach for wind turbine emulator," *IEEE Access,* vol. 11, pp. 58064-58077, 2023. https://doi.org/10.1109/ACCESS.2023.3283420

[32] Y. Kim, H. A. Madsen, M. Aparicio-Sanchez, G. Pirrung, and T. Lutz, "Assessment of blade element momentum codes under varying turbulence levels by comparing with blade resolved computational fluid dynamics," *Renewable Energy,* vol. 160, pp. 788-802, 2020. https://doi.org/10.1016/j.renene.2020.06.006

[33] A. Safaeinejad, M. Rahimi, D. Zhou, and F. Blaabjerg, "Pitch control scheme considering entire dynamics and full-load region in PMSG-based wind turbines," *IEEE Transactions on Sustainable Energy,* vol. 16, no. 2, pp. 955-969, 2024. https://doi.org/10.1109/TSTE.2024.3493961

[34] J. Liu, C. Zhao, and Z. Xie, "Power and current limiting control of wind turbines based on PMSG under unbalanced grid voltage," *IEEE Access,* vol. 9, pp. 9873-9883, 2021. https://doi.org/10.1109/ACCESS.2021.3049839

[35] A. Ghafoor, J. Apsley, D. Al Kez, and S. Djurović, "Operation of a wind turbine permanent magnet synchronous generator (PMSG) for ancillary frequency support services," *International Journal of Electrical Power & Energy Systems,* vol. 167, p. 110623, 2025. https://doi.org/10.1016/j.ijepes.2025.110623

[36] A. Bakeer, S. Hussain, A. Chub, H. S. Salama, and G. Magdy, "Energy storage-enabled fractional-order virtual synchronous generator for DC-link voltage regulation in DC microgrid under load and renewable disturbances," *Scientific Reports,* vol. 16, p. 12355, 2026. https://doi.org/10.1038/s41598-026-45850-1

[37] J. Baran and A. Jąderko, "An MPPT control of a PMSG-based WECS with disturbance compensation and wind speed estimation," *Energies,* vol. 13, no. 23, p. 6344, 2020. https://doi.org/10.3390/en13236344

[38] J. Kim, S. H. Lee, and J.-W. Park, "Inertia-free stand-alone microgrid—Part II: Inertia control for stabilizing DC-link capacitor voltage of PMSG wind turbine system," *IEEE Transactions on Industry Applications,* vol. 54, no. 5, pp. 4060-4068, 2018. https://doi.org/10.1109/TIA.2018.2840083

[39] Z. Wang *et al.*, "Enhanced DC-link voltage control of PMSG-based wind turbine generators by machine-side converter during asymmetrical grid faults," *International Journal of Electrical Power & Energy Systems,* vol. 155, p. 109638, 2024. https://doi.org/10.1016/j.ijepes.2023.109638

[40] M. Rahimi, "Modeling, control and stability analysis of grid connected PMSG based wind turbine assisted with diode rectifier and boost converter," *International Journal of Electrical Power & Energy Systems,* vol. 93, pp. 84-96, 2017. https://doi.org/10.1016/j.ijepes.2017.05.019

[41] A. Khan, D. Aragon, M. Seyedmahmoudian, S. Mekhilef, and A. Stojcevski, "Inertia emulation control of PMSG-based wind turbines for enhanced grid stability in low inertia power systems," *International Journal of Electrical Power & Energy Systems,* vol. 156, p. 109740, 2024. https://doi.org/10.1016/j.ijepes.2023.109740

[42] M. Jahanpour-Dehkordi, S. Vaez-Zadeh, and J. Mohammadi, "Development of a combined control system to improve the performance of a PMSG-based wind energy conversion system under normal and grid fault conditions," *IEEE Transactions on Energy Conversion,* vol. 34, no. 3, pp. 1287-1295, 2019. https://doi.org/10.1109/TEC.2019.2912080

[43] K. B. Tawfiq, P. Sergeant, and A. S. Mansour, "Comparative analysis of space vector pulse-width modulation techniques of three-phase inverter to minimize common mode voltage and/or switching losses," *Mathematics,* vol. 12, no. 18, p. 2832, 2024. https://doi.org/10.3390/math12182832