IJECBE (2026), **4**, **1**, 1–16
Received (22 May 2026) / Revised (15 June 2026)
Accepted (18 June 2026) / Published (30 March 2026)
https://doi.org/10.62146/ijecbe.v4i1.244
https://ijecbe.ui.ac.id
ISSN 3026-5258

International Journal of Electrical, Computer and Biomedical Engineering

RESEARCH ARTICLE

Implementation and Hardware-Level Validation of a 36 V BLDC Motor Drive Using Artix-7 FPGA

Rafi Kamil Arief[†], Fajar Rahino Triputra[‡], and Arief Udhiarto[*†]

†Department of Electrical Engineering, Faculty of Engineering, Universitas Indonesia, Depok, Indonesia
‡Research Center for Aeronautics Technology, National Research and Innovation Agency (BRIN), Indonesia
*Corresponding author. Email: arief.udhiarto@ui.ac.id

Abstract

This paper presents an Artix-7 FPGA-based implementation and validation of a 36 V BLDC motor drive for open-loop no-load operation. The proposed controller combines Hall-sensor sixstep commutation, counter-based PWM generation, bidirectional phase sequencing, and deadtime-protected gate output logic in programmable hardware. The study emphasizes hardwarelevel validation, including the consistency of Hall-sector decoding, PWM scaling, and inverter output behavior, rather than closed-loop speed regulation. The design was verified using Verilog simulation and experimentally tested with a three-phase inverter, a 20 kHz PWM carrier, and a 10 - bit duty cycle. The measurements confirmed the expected PWM-driven phase, low-side return path, and floating phase behavior for adjacent Hall states. No-load characterization was then performed in both clockwise and counterclockwise directions. The measured speed increased almost linearly with duty cycle and reached approximately 2425 r/min at full duty cycle, while the supply current increased to about 0.49 A. The minimum start duty was 2% in both directions, and the minimum sustain duty was 1.8% and 1.7% for clockwise and counterclockwise rotation, respectively

Keywords: BLDC motor drive, FPGA, Hall sensor, PWM control, six-step commutation, dead-time insertion

1. Introduction

Brushless direct current (BLDC) motors have become integral components in modern electric drives, aerospace actuators, and high-performance embedded control systems

due to their high torque-to-weight ratio, high efficiency, compact size, and high reliability compared to traditional brushed motors [1], [2], [3], [4]. Their compact design and highpower density make them particularly suitable for demanding applications, such as unmanned aerial vehicles (UAVs) [5]. Unlike conventional DC motors that use mechanical brushes and commutators, BLDC motors require electronic commutation to manage electrical input to generate a rotating magnetic field [6]. Hence, highly accurate synchronization between the rotor position and inverter switching becomes essential for stable operation.

The electrical behavior of BLDC motors has been extensively modeled in the literature [1], [7], [8]. Classical BLDC drive models describe the relationship among phase voltage, phase current, trapezoidal back electromotive force (back-EMF), electromagnetic torque, and rotor speed [1]. In sensored BLDC drives, the rotor position is commonly obtained from three Hall sensors placed 120 electrical degrees apart [6], [9]. These sensors detect the rotor's magnetic field and generate digital signals that divide a full electrical rotation into six distinct sectors [4], [10]. This feedback then allows the controller to implement six-step (trapezoidal) commutation by selecting active high-side and low-side switches so that the necessary electromagnetic torque can be produced for motor rotation [9].

Traditionally, BLDC motor control has been implemented using microcontrollers or digital signal processors. However, software-based controllers face significant limitations in high-precision and real-time tasks due to their sequential execution [1]. Furthermore, software-induced pulse width modulation (PWM) jitters in microcontroller unit (MCU)- based BLDCs increase motor-current fluctuations, ripple-induced losses, and thermal accumulation [5]. As a solution, field-programmable gate array (FPGA) based controllers offer a robust alternative by providing a fully user-defined parallel hardware architecture [11], [12]. Unlike the sequential processing of software-based microcontrollers, FPGAs allow for deterministic timing and concurrent execution of multiple control tasks with nanosecond precision, such as Hall decoding, PWM generation, and enhanced protection [13], [14], [15], [16], [17].

Earlier works have demonstrated FPGA-based digital PWM control, stability analysis, speed-control experiments, and very high-speed integrated circuit hardware description language (VHDL) or Verilog implementation for BLDC drives. Later studies extended this direction toward reduced-complexity application-specific integrated circuit (ASIC) oriented hardware, rapid system on chip (SoC) deployment, hybrid model-predictive commutation, real-time simulation, reduced instruction set computer five (RISC-V) traction control, and comparative proportional integral derivative (PID) and dead time compensation (DTC) implementation [3]. However, many studies emphasize advanced control algorithms, PWM techniques, or performance improvements, while the underlying hardware validation path is often treated as an assumed prerequisite. In practice, a BLDC motor may still rotate even when Hall-sector mapping, phase assignment, PWM scaling, or bidirectional commutation is not fully verified, which makes it difficult to separate controller algorithm performance from basic hardware implementation errors.

Therefore, the development direction of FPGA-based BLDC should not only focus on increasingly complex control algorithms but also on establishing a trust-

worthy hardware baseline. FPGA and complex programmable logic device (CPLD) devices can generate precise high-frequency PWM signals when the clock, counter period, and duty-cycle resolution are selected consistently [18]. At the same time, dead time is not a secondary implementation detail. It is required to prevent simultaneous conduction in inverter legs, yet it can introduce output-voltage error, current distortion, and torque pulsation [19], [20]. Thus, experimental validation should not only observe that the motor rotates but also show that Hall-state mapping, PWM frequency, duty-cycle scaling, and dead-time-protected gate outputs are mutually consistent.

Accordingly, this work presents an Artix-7 implementation of a Hall-sensor six-step BLDC controller and links the complete control path from Hall inputs, PWM command, and direction selection to the six inverter gate outputs. The analysis relates the implemented timing parameters, including the 20 kHz PWM carrier, 10 - bit duty command, Hall-sector transition rate, and 200 ns dead-time interval, to the measured no-load behavior. The experimental contribution is a repeatable no-load characterization over 13 duty-cycle setpoints in both clockwise (CW) and counterclockwise (CCW) directions, with five speed readings at each point. Additional identification of the minimum start and sustain duty near the low-duty operating boundary is also provided. The scope is intentionally limited to noload hardware validation, while loaded torque and closed-loop speed regulation are reserved for subsequent work.

2. System Design

2.1 General Concept

The proposed system is designed as an FPGA-based Hall-sensor BLDC motor controller for open-loop no-load hardware validation. The controller is implemented on a Digilent CMOD A7 board containing a Xilinx Artix-7 XC7A35T FPGA. A custom BLDC control IP core was developed in Verilog and connected to a MicroBlaze processor through an AXILite interface. The MicroBlaze firmware provides a universal asynchronous receivertransmitter (UART)-based supervisory interface to configure the BLDC setup, while FPGA logic performs real-time motor control functions, including six-step commutation, PWM generation, gate-signal composition, and dead-time insertion. This architecture separates command handling from deterministic gate-signal generation, so that it is only responsible for updating registers, while the FPGA fabric directly produces the six inverter gate commands according to the selected Hall sector, direction, and duty cycle.

2.2 System Architecture

Fig. 1 shows the overall system architecture with the CMOD A7-35T board accommodating two functional domains, namely, a Microblaze soft-core processor and programmable logic (PL). On the control path, the programmable logic receives three Hall sensor signals from the motor and then produces six PWM gate outputs to an external threephase driver and inverter powered by a 36 V power supply. After that, the three channels of a Rigol DHO924S oscilloscope are connected directly to the U, V, and W phase terminals of the BLDC motor to capture the applied phase waveforms,

and the recorded data are transferred to a PC via USB. A USB–UART link between the PC and MicroBlaze is used to issue configuration commands during experiments.

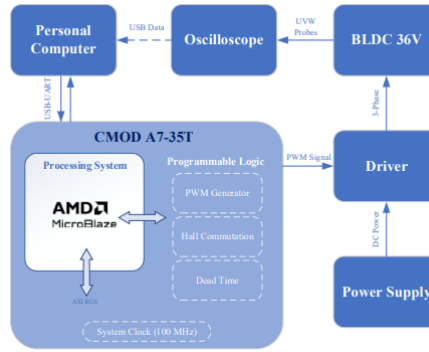

Figure 1. System architecture of the Artix-7 FPGA-based BLDC motor drive and measurement setup

At the PL layer, real-time control is implemented as a custom IP core comprising three hardware submodules: a counter-based PWM generator, a combinational Hall commutation decoder, and dead-time logic for the inverter gate signals, as illustrated in Fig. 2.

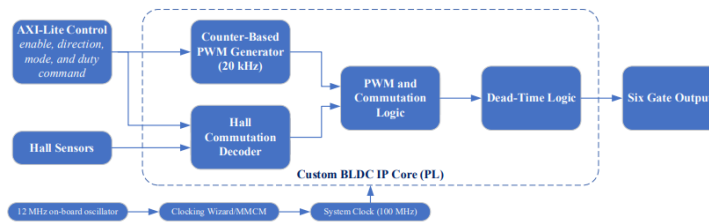

Figure 2. Internal control and signal flow of the custom BLDC IP core.

The timing basis of the programmable logic is derived from the 12 MHz onboard crystal oscillator of the CMOD A7-35T, which is converted to a 100 MHz system clock using the mixed-mode clock manager (MMCM) configured through the Xilinx Clocking Wizard IP. The resulting clock is distributed to the AXI-Lite interface, PWM generator, and dead-time logic, while the Hall commutation decoder operates combinationally from the Hall-sensor and direction inputs. The PWM generator uses this clock to produce the fixed 20 kHz carrier, as detailed in the PWM generation subsection.

The MicroBlaze processor updates the movement and holding duty-cycle commands through separate AXI-Lite registers. Depending on the operating mode, the corresponding command is selected and converted into the comparison threshold used by the PWM generator. In parallel, the Hall-sensor state and direction command are processed by the commutation decoder to select the active high-side and low-side phases. The PWM and phase-selection signals are subsequently combined and

passed through the dead-time and interlock logic before reaching the six inverter gate outputs. An inactive enable signal or an invalid Hall state forces all gate outputs low.

The BLDC IP is connected to the MicroBlaze through an AXI-Lite memory-mapped register interface comprising four 32 - bit registers. Register 0, located at offset 0x00, serves as the control register, where bit 0 corresponds to the enable signal, bit 1 determines the rotation direction, bit 2 enables continuous rotation operation, and bit 3 selects the manual-sector control mode. Register 1, at offset 0x04, stores the 10 - bit duty-cycle command used during rotational operation, whereas Register 2, at offset 0x08, stores the duty-cycle command used to maintain the rotor at a fixed sector. Register 3, located at offset 0x0C, contains the 3 - bit Hall sector index applied when the controller operates in manual mode. After these register values are written by the firmware, the generation of gate-drive signals is performed autonomously within the FPGA fabric, thereby eliminating the need for further processor intervention during motor actuation.

The control algorithm is executed in four stages. First, the MicroBlaze writes the enable status, rotation direction, operating mode, and duty-cycle command to the AXI-Lite registers. Second, the BLDC IP selects the appropriate duty-cycle command and commutation source: live Hall-sensor inputs for continuous rotation, the captured Hall state for holding operation, or the register-defined sector for manual operation. Third, the selected Hall state and direction are decoded to determine the active high-side, low-side, and floating phases, while an invalid Hall state disables all gate outputs. Finally, the PWM signal is applied to the selected high-side phase, combined with the low-side command, and passed through the dead-time and interlock logic before the six gate signals are transferred to the inverter driver. These operations are continuously executed in the FPGA fabric after the control registers have been updated.

2.3 Hall-Sensor-Based Six-Step Commutation

The BLDC controller uses three-bit Hall-sensor signals to determine the rotor sector for six-step commutation. The Hall vector is defined as (1) [21].

$$H = \{H_U, H_V, H_W\} \tag{1}$$

where HU, HV, and HW are the digital Hall-sensor inputs associated with the motor phases. In normal operation, only six Hall states are valid operations, which are 100, 101, 011, 010, and 110. The remaining states, 000 and 111, are treated as invalid conditions and the commutation outputs are disabled.

For each valid Hall state, the controller generates two phases-selection vectors, denoted as *hi*[2: 0] and *lo*[2: 0], where the bit order corresponds to phase U, V, and W. The *hi* vector selects the phase connected to the high-side PWM, while the *lo* vector selects the phase connected to the low-side return path. The remaining phase is left floating, so that each commutation sector energizes only two motor phases to create a trapezoidal six-step operation. In clockwise operation, the implemented Hall to phase mapping is summarized in Table 1.

Table 1. Hall-to-phase commutation sequence for clockwise rotation

Hall Signal	High-side PWM	Low-side ON	Floating Phase	Current Flow
100	U	W	V	U → W
101	U	V	W	U → V
001	W	V	U	W → V
011	W	U	V	W → U
010	V	U	W	V → U
110	V	W	U	V → W

2.4 PWM and Counter Generation

The PWM generator is implemented as a counter-based digital PWM block driven by the 100 MHz FPGA system clock operating at nanosecond resolution. The PWM switching frequency was fixed at 20 kHz as a practical design compromise rather than as an experimentally optimized value. Frequencies within the 10-20 kHz range have been employed in previous FPGA-based BLDC drives, including a 20 kHz implementation reported by Wang et al. [4] and a 10-20 kHz range reported by Pindoriya et al. [22]. The reason behind the selection of 20 kHz PWM frequency is as a compromise between switching losses and motor current quality. Increasing the PWM frequency above 20 kHz can reduce current ripple and move switching-related acoustic components beyond the audible range [23], [24]. However, it also increases the number of switching transitions per second, resulting in higher MOSFET switching losses, gate-driver losses, and inverter heating [25]. Conversely, frequencies below approximately 10 kHz reduce switching losses but generally produce larger phase-current and torque ripple and may generate audible motor noise [26]. Therefore, 20 kHz provides a practical balance between switching efficiency, current ripple, acoustic behavior, and the timing resolution. The PWM period is determined by the selected carrier frequency, as expressed as (2).

$$N_{PWM} = \frac{f_{clk}}{f_{PWM}} = \frac{100MHz}{20kHz} = 5000\ ticks \tag{2}$$

where N_{PWM}, f_{clk}, and f_{PWM} denote the number of clock ticks per PWM period, the FPGA clock frequency and the PWM carrier frequency, respectively. Since each clock tick corresponds to 10 ns, the PWM generator can update the output state with a native timing granularity of 10 ns over a 50 us switching period. This counter-based approach follows the common FPGA implementation of digital PWM, where a free-running counter is compared with a duty-cycle threshold to determine the active interval of each PWM cycle. The duty-cycle command only determines the fraction of the PWM period during which the gate signal remains active. Therefore, increasing the number of available PWM timing levels improves the control resolution, but does not reduce the maximum attainable motor speed. In an ideal inverter stage, PWM does not change the DC-link voltage itself. Instead, it changes the effective voltage applied to the motor winding over one switching period, which can be expressed as (3) [27].

$$V_{eff} \approx d \times V_{dc} \tag{3}$$

where V_{eff} is the effective applied winding voltage, Vdc is the DC-link voltage from external supply, and d is the PWM duty ratio.

In the implemented RTL, the duty-cycle value is not used directly as the PWM output. Instead, it is first converted into an on-time threshold expressed in clock ticks. The duty command is supplied through the AXI-Lite register interface. Inside the PWM module, the active command is selected according to the operating mode (static or rotating operation). The selected command is then scaled to the PWM period counter, which can be expressed as (4).

$$N_{on} = \frac{D_{cmd} \times N_{PWM}}{1024} \tag{4}$$

where N_{on} is the on-time threshold in clock ticks, D_{cmd} is selected active command, and N_{PWM} is the number of clock ticks in one PWM period. Since the PWM output remains active for N_{on} ticks within a total period of N_{PWM} ticks, the resulting duty ratio is obtained as (5).

$$d = \frac{N_{on}}{N_{PWM}} \tag{5}$$

By substituting the on-time threshold equation, the duty ratio can be expressed as (6) and (7).

$$d = \frac{\left(\frac{D_{cmd} \times N_{PWM}}{1024}\right)}{N_{PWM}} \tag{6}$$

$$d = \frac{D_{cmd}}{1024} \tag{7}$$

The PWM waveform is generated synchronously with the FPGA system clock. A period counter advances once per clock cycle and restarts after completing one PWM period. At each clock tick, the counter value is compared with the on-time threshold. The PWM signal is active while the counter value is inside the active interval and becomes inactive for the remainder of the period. Therefore, the duty-cycle command linearly controls the active fraction of each PWM period, while the 100 MHz clock determines the timing granularity of the generated PWM signal.

2.5 Dead-Time Insertion

Dead-time logic is required to ensure that complementary switches are not turned on simultaneously during switching transitions. In an inverter leg, shoot-through occurs when the high-side and low-side devices in the same phase conduct at the same time, creating a direct current path from the positive DC bus to ground through both switches. For this reason, the generated PWM and commutation signals are passed through dead-time logic before being delivered to the gate drivers. The dead-time interval can be expressed as (8).

$$DT_{ticks} = \frac{t_{dead}}{T_{clk}} \tag{8}$$

where DTticks is the number of FPGA clock cycles for dead-time insertion, tdead is the required dead-time interval, and Tclk is the FPGA clock period. As a result, the gate output is filtered by dead-time protection.

2.6 Experimental Setup

The experimental setup was arranged to validate the FPGA-based BLDC motor drive under static and rotating operating conditions. The implemented design was deployed on the Artix-7 FPGA platform and connected to a three-phase inverter gate-driver stage, an external DC power supply, and a BLDC motor. The DC power supply provided the 36 V input to the inverter stage, while the motor-driver assembly contained the CMOD A7-35T controller, gate-driver circuit, and three-phase inverter used to energize the motor phases. The PC controller was used to program the FPGA, transmit operating commands, and configure the duty-cycle and operating mode through the MicroBlaze interface. The digital oscilloscope measured the phase-terminal and gate-related switching waveforms, whereas the tachometer provided an independent measurement of the motor rotational speed. The validation was performed first in static or manual-sector mode to verify that each Hallsector command produced the corresponding gate-output pattern, and then in rotating mode to observe the motor response under PWM duty control. The overall laboratory setup used for the validation is shown in Fig. 3, where the main specifications of the BLDC motor used are summarized in Table 2

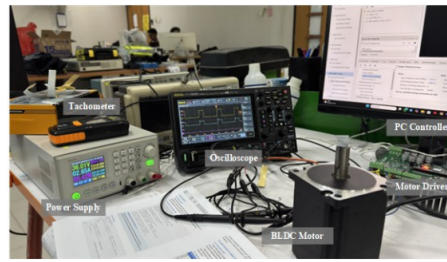

Figure 3. Experimental setup of the FPGA-based BLDC motor drive system

3. Results

3.1 Testbench Simulation of BLDC Six-Step Commutation

The first validation stage was carried out through Verilog testbench simulation to verify the basic Hall-sector decoding, PWM generation, and gate-output relation of the proposed system. The simulation was configured for clockwise rotation with a PWM value of 51 counts, corresponding to approximately 5% duty cycle. Since the motor has four pole pairs, one mechanical revolution is represented by 24 Hall-sector transitions.

Table 2. Specifications of the BLDC motor used in the experiments

Parameter	Specification
Motor model	86BLF105
Motor type	Three-phase BLDC motor with Hall sensors
Number of poles	8 (4 pole pairs)
Number of phases	3
DC-bus voltage	36 V
Rated speed	3000 rpm $\pm$ 10%
No-load current	0.8 A
Motor weight	2.5 kg

Fig. 3 shows the simulated six-step commutation waveform generated by the BLDC controller. The Hall sensor inputs are represented by sens_hu, sens_hv, and sens_hw, while the corresponding inverter gate commands are represented by gate_hu, gate_hv, gate_hw, gate_lu, gate_lv, and gate_lw. The Hall sequence produces periodic commutation-sector transitions, and the FPGA logic responds by activating the appropriate high-side and low-side gate pair according to the implemented commutation table. In each sector, the selected high-side switch is modulated by the PWM signal, while the corresponding low-side switch provides the return current path. The simulation also includes shoot-through indicator signals (short_u, short_v, and short_w) where all shootthrough indicators remain low during the entire simulated interval. This confirms that the generated gate pattern does not command both switches of the same phase leg to conduct at the same time. Therefore, the simulation verifies that the commutation logic and deadtime insertion on the gate operate consistently before hardware testing is performed

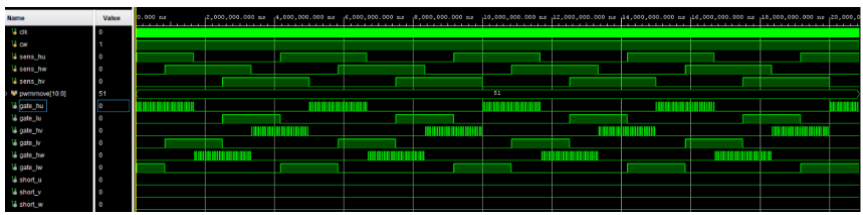

Figure 4. Simulated six-step commutation waveform of the FPGA-based BLDC controller at 5% PWM duty cycle

3.2 FPGA resource-utilization comparison

Following functional verification, the post-synthesis resource utilization of the proposed BLDC control IP was evaluated and compared with the FPGA-based three-loop BLDC servo controller reported by Wang et al. [4], as summarized in Table 3.

The proposed BLDC controller requires 99 LUTs, 157 flip-flops, no BRAM, and one DSP block, whereas the three-loop BLDC servo controller reported by Wang et al. [4] uses 8,490 LUTs, 1,245 flip-flops, and 177 DSP blocks, with BRAM utilization

Table 3. FPGA resource-utilization comparison

Resource	Proposed BLDC controller	Wang et al. (2021)
Look-up tables (LUTs)	99	8,490
Flip-flops (FFs)	157	1,245
Block RAM (BRAM)	0	Not reported
Digital signal processing (DSP blocks)	1	177

not reported. The relatively low resource utilization of the proposed architecture results from its streamlined hardware functions, comprising Hall-sensor six-step commutation, counter-based PWM generation, bidirectional phase sequencing, dead-time insertion, and an AXI control interface. No BRAM is required because the controller does not employ lookup tables, data buffering, or stored motor models, while the single DSP block is inferred for the multiplication used to convert the commanded duty value into PWM counter ticks. This compact implementation leaves substantial FPGA resources available for future control or monitoring functions and enables deployment on the smaller XC7A35T device. Nevertheless, the lower resource count should not be interpreted as an absolute efficiency advantage because Wang et al. implemented a more computationally intensive three-loop servo system incorporating current, speed, and position control. Therefore, the comparison primarily demonstrates the resource-functionality trade-off between the two architectures.

3.3 Static PWM and Commutation Verification

Figure 5 presents the measured phase voltage waveforms for two adjacent Hall states, where Channel 1, Channel 2, and Channel 3 represent phase U, phase V, and phase W, respectively. Fig. 5(a) shows the waveform for Hall state 100. In this commutation state, phase U is selected as the PWM-driven high-side phase, phase W is connected to the lowside return path, and phase V remains floating. The measured PWM frequency is 20 kHz with a duty cycle of approximately 5.8%. Phase U exhibits a periodic PWM waveform with a peak-to-peak voltage of approximately 40.6 V, slightly higher than the 36 V DC input due to switching transients and voltage ringing caused by parasitic elements and fast inverter switching. Although phase V is not directly driven in this sector, it shows a synchronized transient response of approximately 25.5 Vpp at its terminal. This observation is consistent with previous BLDC studies showing that the unexcited phase retains a measurable terminal voltage and that phase-terminal voltage is affected by the motor neutral-point potential and inverter freewheeling state [28], [29]. Nevertheless, the separate contribution of each mechanism to the measured 25.5 Vpp response was not determined in this experiment. In contrast, phase W remains close to the reference level because it provides the low-side conduction path, with only small residual voltage caused by conduction loss, switching noise, or measurement parasitics.

Fig. 5(b) shows the waveform for Hall state 101. In this sector, phase U remains the PWM-driven high-side phase, while the low-side return path shifts from phase W to phase V, leaving phase W as the floating phase. The result shows that phase U still

produces a periodic 20 kHz PWM pulse with a peak-to-peak voltage of approximately 40.6 V, phase V stays near the reference level as the low-side conduction path, and phase W exhibits a synchronized transient response of approximately 24.7 Vpp as the floating phase. This result is also consistent with the implemented six-step commutation table, where two adjacent Hall sectors can share the same high-side phase while the low-side phase changes to advance the stator excitation vector.

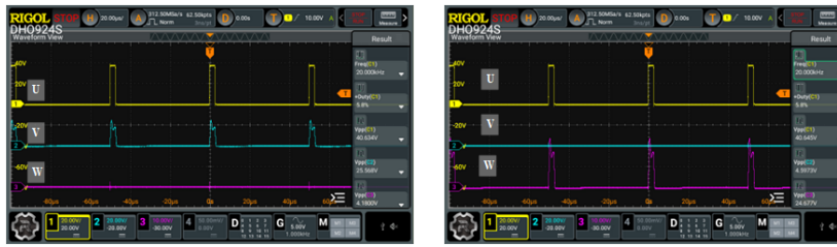

Figure 5. Measured phase voltage waveforms for static Hall states: (a) Hall 100 and (b) Hall 101.

3.4 No-Load Speed Response

Fig. 6(a) shows the no-load speed and power supply current characteristics of the motor under CW and CCW rotation. The motor speed increased almost linearly with the PWM duty cycle and reached approximately 2400 r/min at 100% duty cycle. The CW and CCW speed responses were nearly identical, indicating symmetrical no-load operation in both rotational directions. The linear regression results show that the CW speed response can be expressed as (9), while the CCW response is given by (10), where S denotes the motor speed in r/min and d denotes the PWM duty cycle as a percentage. The similar slope values indicate comparable speed gains for both directions.

$$S_{CW} = 24.7d - 30.83 \; R^2 = 0.99999 \tag{9}$$

$$S_{CCW} = 24.66d - 30.68 \; R^2 = 0.99998 \tag{10}$$

A similar trend is observed in the current flow. The current increases gradually with increasing duty cycle increases and reaches approximately 0.49 A at full duty cycle. The fitted current equations are shown in (11) and (12) for CW and CCW, respectively. These results indicate that the no-load current demand is almost identical for both directions, suggesting comparable electrical and mechanical losses during CW and CCW operations.

$$I_{CW} = 0.00496d - 0.00741 \; R^2 = 0.99755 \tag{11}$$

$$I_{CCW} = 0.00492d - 0.00441 \; R^2 = 0.99915 \tag{12}$$

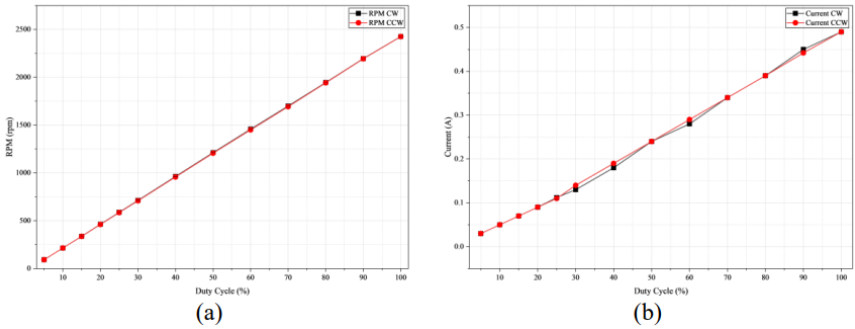

Figure 6. No-load characteristics of the BLDC motor under CW and CCW rotation: (a) speed response versus PWM duty cycle and (b) supply current versus PWM duty cycle.

Fig. 7 shows the measured PWM output waveform at approximately 20 kHz switching frequency and 50% duty cycle. The waveform confirms that the FPGA-generated PWM signal was properly translated to the inverter output. The high and low intervals of phase U correspond to the PWM-on and PWM-off intervals, respectively, while the narrow peak at the switching edge represents a short-duration switching-related voltage component rather than the steady-state phase-voltage level. High-frequency components in measured motorterminal voltages have also been reported in PWM-based three-phase BLDC inverter operation [28]. However, because the gate-node voltages and phase currents were not measured simultaneously, the specific physical origin of the transient observed in Fig. 7 was not separately identified in this experiment.

Figure 7. Measured PWM output waveform at 20 kHz switching frequency and 50% duty cycle.

3.5 Minimum Start and Sustain Duty

The minimum start and sustain duty results are summarized in Table 3.2. The motor started from rest at an active PWM command of 20 counts, corresponding to 2% duty cycle, in both CW and CCW directions. This result indicates that the starting requirement under no-load condition was nearly identical for both rotational directions. After the motor reached rotation, the duty command was gradually reduced to determine the sustain boundary. The minimum sustain duty was 1.8%

for CW and 1.7% for CCW, where the motor operated near the stall boundary. The start-sustain hysteresis was small, equal to 1 count for CW and 2 counts for CCW, indicating that only a slightly lower command was required to maintain rotation than to initiate it.

Table 4. Minimum start and sustain duty

Parameter	CW	CCW
Minimum start duty	2%	2%
Minimum sustain duty	1.8%	1.7%
RPM at minimum sustain	10 RPM	9 RPM

Fig. 7 shows the output waveform at the minimum CW start condition. At 2% duty cycle, equivalent to an active PWM command of 20 counts, the motor successfully started from rest. The waveform exhibits a fast-switching transition with a rise time of approximately 30 ns and a small overshoot of about 4.9%, followed by a short settling interval. This result supports the selection of 2% duty cycle as the minimum CW start duty cycle.

Figure 8. Output waveform at the minimum CW start with 2% duty cycle

4. Discussion

The experimental results show that the implemented FPGA controller provides a consistent control path from Hall-sensor inputs to inverter gate outputs. The simulation confirms the intended Hall-sector decoding, PWM modulation, low-side return selection,and dead-time application before hardware operation. This is important because BLDC rotation alone is not sufficient to verify the controller implementation, where incorrect phase mapping or timing may still produce motion even though it may mask hidden commutation errors. The agreement between simulation and measured static waveforms indicates that the FPGA logic, AXI-based duty command, and inverter output stage operate coherently as a complete drive chain.

The static commutation measurements further confirm the expected six-step behavior. In two adjacent Hall states, the same high-side phase remains PWM-driven while the low-side return phase changes, causing the floating phase to shift accordingly. The voltage observed on the floating phase should therefore not be interpreted as active phase excitation, but as a terminal-voltage response consistent

with winding coupling, neutral-point movement, and inverter freewheeling states during PWM switching. The individual contribution of these mechanisms was not separately measured.

The no-load speed response supports the correctness of the PWM scaling used in the FPGA implementation. Over the tested range, the motor speed increases almost linearly with duty cycle. This indicates that the 10 - bit active PWM command is translated into a proportional effective voltage at the motor terminals. The CW and CCW responses are also nearly identical, with only small differences in speed and power supply current. This result suggests that the bidirectional commutation table and phase assignment do not introduce a significant directional bias under no-load operation.

The power supply current trend is consistent with the no-load operating condition. Since no external mechanical load is applied, the measured current mainly represents the current required to overcome mechanical friction, windage, inverter conduction loss, switching loss, and the electrical requirement for maintaining rotation. The gradual increase in current with duty cycle indicates that the drive remains stable across the tested operating range,while the close CW and CCW current profiles imply comparable loss characteristics in both directions.

The minimum start and sustain duty results define the low-speed operating boundary of the open-loop drive. The motor starts from rest at 2% duty cycle in both directions, while it can sustain rotation at slightly lower duty levels after motion has been established. This small start-sustain hysteresis is expected because the torque required to overcome static friction is higher than the torque required to maintain rotation near the stall boundary.

5. Conclusion

This work presented the design and experimental validation of an Artix-7 FPGA-based BLDC motor drive using Hall-sensor six-step commutation. The implemented controller integrates Hall-sector decoding, counter-based PWM generation, direction control, and dead-time insertion on inverter gate outputs. Simulation and static waveform measurements confirmed that the FPGA logic generated the expected commutation sequence and PWM output pattern before rotating operation was evaluated.

The no-load experimental results show that the motor speed increased almost linearly with PWM duty cycle and reached approximately 2425 r/min at full duty cycle, with nearly identical CW and CCW responses. The power supply current increased gradually up to about 0.49 A, indicating stable no-load operation over the tested duty range. The motor started at 2% duty cycle in both directions and sustained rotation down to 1.8% for CW and 1.7% for CCW. These results demonstrate that the proposed FPGA-based drive provides a consistent hardware baseline for bidirectional BLDC operation. Future work will extend the validation to loaded operation, current ripple analysis, thermal behavior, and closed-loop speed control.

References

[1] C. L. Xia. *Permanent Magnet Brushless DC Motor Drives and Controls*. John Wiley & Sons, 2012. DOI: 10.1002/9781118188347.

[2] N. Milivojevic et al. "Stability Analysis of FPGA-Based Control of Brushless DC Motors and Generators Using Digital PWM Technique". In: *IEEE Transactions on Industrial Electronics* 59.1 (2012), pp. 343–351. DOI: 10.1109/TIE.2011.2146220.

[3] P. Mishra et al. "Implementation and Validation of Quadral-Duty Digital PWM to Develop a Cost-Optimized ASIC for BLDC Motor Drive". In: *Control Engineering Practice* 109 (2021), p. 104752. DOI: 10.1016/J.CONENGPRAC.2021.104752.

[4] W. Bolun et al. "A Method for Rapid Deployment of Brushless DC Motor Servo System Model Based on FPGA". In: *2021 6th International Conference on Automation, Control and Robotics Engineering (CACRE)*. 2021, pp. 379–384. DOI: 10.1109/CACRE52464.2021.9501387.

[5] H. A. Hergul et al. "FPGA-Based Performance Enhancement of BLDC Motor Drives for UAV Applications". In: *Proceedings of the 5th International Conference on Informatics and Software Engineering (IISEC)*. 2026, pp. 595–598. DOI: 10.1109/IISEC69317.2026.11418448.

[6] D. Mohanraj et al. "A Review of BLDC Motor: State of Art, Advanced Control Techniques, and Applications". In: *IEEE Access* 10 (2022), pp. 54833–54869. DOI: 10.1109/ACCESS.2022.3175011.

[7] Ali Emadi. *Handbook of Automotive Power Electronics and Motor Drives*. Taylor & Francis, 2005.

[8] P. Pillay and R. Krishnan. "Modeling, Simulation, and Analysis of Permanent-Magnet Motor Drives, Part II: The Brushless DC Motor Drive". In: *IEEE Transactions on Industry Applications* 25.2 (1989), pp. 274–279. DOI: 10.1109/28.25542.

[9] J. Kuruvilla et al. "Speed Control of BLDC Motor Using FPGA". In: *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering* 4.5 (2015). URL: https://www.ijareeie.com/upload/2015/april/72_31_SPEED.pdf.

[10] H. P. Putra, H. Suryoatmojo, and S. Anam. "Perbaikan Faktor Daya Menggunakan Cuk Converter pada Pengaturan Kecepatan Motor Brushless DC". In: *Jurnal Teknik ITS* 5.2 (2016), pp. 156–163. URL: https://media.neliti.com/media/publications/193669-ID-perbaikan-faktor-dayamenggunakan-cuk-co.pdf.

[11] H. M. A. Abbas, R. F. Chisab, and M. J. Mnati. "Monitoring and Controlling the Speed and Direction of a DC Motor Through FPGA and Comparison of FPGA for Speed and Performance Optimization". In: *International Journal of Electrical and Computer Engineering* 11.5 (2021), pp. 3903–3912. DOI: 10.11591/ijece.v11i5.pp3903-3912.

[12] X. Lin-Shi et al. "Implementation of Hybrid Control for Motor Drives". In: *IEEE Transactions on Industrial Electronics* 54.4 (2007), pp. 1946–1952. DOI: 10.1109/TIE.2007.898303.

[13] P. Mishra, A. Banerjee, and M. Ghosh. "FPGA-Based Real-Time Implementation of Quadral-Duty Digital-PWM-Controlled Permanent Magnet BLDC Drive". In: *IEEE/ASME Transactions on Mechatronics* 25.3 (2020), pp. 1456–1467. DOI: 10.1109/TMECH.2020.2977859.

[14] K. Rauma et al. "FPGA Based Dead-Time Compensation for PWM Inverters". In: *European Conference on Power Electronics and Applications*. 2005. DOI: 10.1109/EPE.2005.219394.

[15] S. C. Huerta et al. "FPGA-Based Digital Pulsewidth Modulator with Time Resolution Under 2 ns". In: *IEEE Transactions on Power Electronics* 23.6 (2008), pp. 3135–3141. DOI: 10.1109/TPEL.2008.2005370.

[16] H. Chen, S. Liu, and G. Lai. "High-Precision Dead-Time Intellectual Property Core and Its Compensation for Inverters". In: *Wuhan University Journal of Natural Sciences* 28.3 (2023), pp. 271–276. DOI: 10.1051/WUJNS/2023283271.

[17] U. Ayaz, S. Tugan, and C. Gurbuz. "Dead Time Control with FPGA for Electric Vehicle 3 Phases Inverter Application". In: *14th International Conference on Electrical and Electronics Engineering (ELECO)*. 2023. DOI: 10.1109/ELECO60389.2023.10416062.

[18] E. Koutroulis, A. Dollas, and K. Kalaitzakis. "High-Frequency Pulse Width Modulation Implementation Using FPGA and CPLD ICs". In: *Journal of Systems Architecture* 52.6 (2006), pp. 332–344. DOI: 10.1016/J.SYSARC.2005.09.001.

[19] D. Leggate. "Pulse-Based Dead-Time Compensator for PWM Voltage Inverters". In: *IEEE Transactions on Industrial Electronics* 44.2 (1997), pp. 191–197. DOI: 10.1109/41.564157.

[20] A. R. Munoz and T. A. Lipo. "On-Line Dead-Time Compensation Technique for Open-Loop PWM-VSI Drives". In: *IEEE Transactions on Power Electronics* 14.4 (1999), pp. 683–689. DOI: 10.1109/63.774205.

[21] S. C. Huerta et al. "FPGA-Based Digital Pulsewidth Modulator with Time Resolution Under 2 ns". In: *IEEE Transactions on Power Electronics* 23.6 (Nov. 2008), pp. 3135–3141. DOI: 10.1109/TPEL.2008.2005370.

[22] R. M. Pindoriya et al. "FPGA Based Digital Control Technique for BLDC Motor Drive". In: *IEEE Power and Energy Society General Meeting*. 2018. DOI: 10.1109/PESGM.2018.8586472.

[23] P. Ivo and P. Petr. "Integrated Control and Power Electronic for BLDC Motor". In: *DAAAM International Symposium*. Vol. 22. 1. 2011. URL: https://www.daaam.info/Downloads/Pdfs/proceedings/proceedings_2011/733.pdf.

[24] P. C. Palavicino et al. *Electrification of Hydraulic Systems Using High-Efficiency Permanent Magnet Motors*. Tech. rep. 2020. DOI: 10.25368/2020.53.

[25] A. Andersson, D. Lennstrom, and A. Nykanen. "Influence of Inverter Modulation Strategy on Electric Drive Efficiency and Perceived Sound Quality". In: *IEEE Transactions on Transportation Electrification* 2.1 (2016), pp. 24–35. DOI: 10.1109/TTE.2015.2514162.

[26] W. C. Lo et al. "Acoustic Noise Radiated by PWM-Controlled Induction Machine Drives". In: *IEEE Transactions on Industrial Electronics* 47.4 (2000), pp. 880–889. DOI: 10.1109/41.857968.

[27] D. J. Gauthier et al. "Digital Twin of a DC Brushless Electric Motor-Propeller System with Application to Drone Dynamics". In: *AIAA SciTech Forum and Exposition*. 2024. DOI: 10.2514/6.2024-2359.

[28] J. C. Gamazo-Real, V. Martinez-Martinez, and J. Gomez-Gil. "ANN-Based Position and Speed Sensorless Estimation for BLDC Motors". In: *Measurement* 188 (2022), p. 110602. DOI: 10.1016/j.measurement.2021.110602.

[29] Y. Li et al. "A Sensorless Commutation Error Correction Method for High-Speed BLDC Motors Based on Phase Current Integration". In: *IEEE Transactions on Industrial Informatics* 16.1 (Jan. 2020), pp. 328–338. DOI: 10.1109/TII.2019.2917608.