Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Design and Implementation of a Semi-Autonomous Battery-Powered Electric Vehicle Controller Using an Arduino Microcontroller

Fabrice Nyagushimwa

fabricenyagushimwa@gmail.com

University of Rwanda

Theogene Nsanganiyimana

University of Rwanda

Alex Safari

University of Rwanda

Article

Keywords: Electric vehicle, Arduino Uno, H-Bridge, MOSFET, PWM, battery management, DC motor control, state of charge, Proteus simulation

Posted Date: September 11th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10739756/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations:
No competing interests reported.

Figures 1 is not available with this version

Design and Implementation of a Semi-Autonomous Battery-Powered Electric Vehicle Controller Using an Arduino Microcontroller

Fabrice Nyagushimwa[1,*]
Dept. of EEE, University of Rwanda
Kigali, Rwanda
nyagushimwafabrice@yahoo.com
*Corresponding author

Theogene Nsanganiyimana[1]
Dept. of EEE, University of Rwanda
Kigali, Rwanda
nsanganiye78@gmail.com

Alex Safari[2]
Dept. of EEE, University of Rwanda
Kigali, Rwanda
safarialex@gmail.com

Abstract—The rapid proliferation of internal combustion engine (ICE) vehicles has led to mounting concerns over greenhouse gas emissions, fossil fuel depletion, and urban air quality degradation. Electric vehicles (EVs) present a viable alternative, offering zero tailpipe emissions and superior energy efficiency. This paper presents the design and simulation of a semi-autonomous, battery-powered electric vehicle controller using an Arduino Uno microcontroller. The system integrates a MOSFET-based H-Bridge motor driver for bidirectional 600 W DC motor control, a pulse-width modulation (PWM) speed regulation scheme, a 24 V lead-acid battery power supply, and a 20×4 LCD user interface for real-time monitoring of driving mode, motor speed, and battery state of charge (SOC). Simulation in Proteus 8.1 verified correct forward, reverse, and regenerative braking operation. Results confirm that PWM-based control provides efficient, low-loss speed regulation and that SOC monitoring successfully protects the battery from deep discharge. The architecture is low-cost and employs components commercially available in developing markets.

Keywords—Electric vehicle; Arduino Uno; H-Bridge; MOSFET; PWM; battery management; DC motor control; state of charge; Proteus simulation.

I. Introduction

The global transportation sector is a major contributor to anthropogenic CO_2 emissions. Road transport generates a disproportionate share of nitrogen oxides (NO_x), unburned hydrocarbons (HC), and particulate matter (PM) all posing serious public health and environmental risks [1]. Many governments have therefore adopted zero-emission vehicle (ZEV) mandates to accelerate the transition from ICE-powered vehicles to battery electric vehicles (BEVs) [2].

Electric vehicles eliminate direct tailpipe emissions by replacing the combustion engine with a traction motor, battery pack, and electronic controller. Even accounting for upstream power-plant emissions, the carbon footprint of a BEV is substantially lower than an equivalent ICE vehicle, especially when charged from renewable sources [3].

In developing economies such as Rwanda, the high upfront cost of commercial EVs remains a barrier to mass adoption. Open-source microcontrollers such as the Arduino enable low-cost prototype development that supports localized EV research, skills development, and indigenous electric mobility solutions.

The principal contributions of this work are: (i) an MOSFET H-Bridge bidirectional drive circuit rated for 24 V / 600 W; (ii) PW based speed control on an Arduino Uno; (iii) real-time SOC monitoring with LCD driver feedback; and (iv) full Proteus 8.1 simulation verification.

II. Literature Review

Electric vehicles were prevalent in the early twentieth century before the mass production of gasoline vehicles and the associated petroleum infrastructure made ICE cars substantially cheaper and longer-range [4]. The energy crises of the 1970s and 1980s rekindled interest in EVs, and the global economic recession of the late 2000s further accelerated the shift toward hybrid and fully electric platforms [5].

A. DC Motor Principles

The DC motor converts electrical energy into mechanical energy through electromagnetic interaction between its armature winding and the stator field. Fleming's left-hand rule governs the direction of the resulting force, whose magnitude is given by F = B·I·L (N), where B is flux density (T), I is armature current (A), and L is conductor length (m). DC motors deliver maximum torque from standstill, making them well suited to EV traction [7].

The separately-excited DC motor used in this design is modeled by the coupled electrical and mechanical equations. The armature voltage equation is given by (1),

$$V(t) = I_a(t)R_a + L_a\frac{dI}{dt} + E_b(1)$$

where R_a and L_a are the armature resistance and inductance and E_b = K_eω is the back-EMF, proportional to the motor's angular speed ω through the back-EMF constant K_e (V·s/rad). The developed electromagnetic torque is given by (2),

$$T_m = K_t I_a \qquad (2)$$

where K_t is the torque constant (N·m/A); for an SI-consistent motor model K_t = K_e. The mechanical dynamics are governed by (3),

$$J\frac{d\omega}{dt} = T_m - T_L - B\omega \qquad (3)$$

with J the combined rotor/load inertia (kg·m²), T_L the load torque, and B the viscous friction coefficient (N·m·s/rad).

Combining the electrical and mechanical equations and taking the Laplace transform (zero initial conditions) gives the open-loop speed transfer function in (4).

$$\frac{\omega(s)}{V(s)} = \frac{K_t}{(L_a s + R_a)(Js + B) + K_t K_e} \qquad (4)$$

Under quasi-steady-state operation (dI_a/dt ≈ 0, dω/dt ≈ 0 at constant load) this reduces to the algebraic relation in (5),

$$\omega_{ss} = \frac{V - I_a R_a}{K_e} \qquad (5)$$

which is used in Section III to relate PWM duty cycle to commanded shaft speed. Because L_a is small for brushed DC traction motors of this class, the electrical time constant τ_e = L_a/R_a is typically one to two orders of magnitude smaller than the mechanical time constant τ_m = JR_a/(K_eK_t + BR_a); the PWM switching frequency (Section III-C) therefore only needs to exceed 1/τ_e, while the controller's effective bandwidth is set by τ_m.

B. H-Bridge Topology

Bidirectional motor control requires reversing current polarity across the motor terminals. The H-Bridge achieves this with four switching elements in an H-shaped configuration [9]. Activating diagonal switch pairs reverses the applied motor voltage and rotation direction. Power MOSFETs are preferred for sub-250 V applications: they offer switching frequencies above 200 kHz, no current-tail losses, and an integrated body diode [10].

MOSFET power dissipation in each bridge arm has two components. Conduction loss is given by (6),

$$P_{cond} = {I_{rms}}^2 \times R_{DS(on)} \qquad (6)$$

where R_DS(on) is the on-state channel resistance (typically 0.14 Ω for the IRF630 and 0.20 Ω for the IRF9630 at 25°C, rising with junction temperature). Switching loss is approximated by (7),

$$P_{sw} = 0.5 \times V_{DS} \times I_D \times (t_r + t_f) \times f_{sw} \qquad (7)$$

where t_r and t_f are the current rise and fall times and f_sw is the PWM frequency. At the design point (I_D ≈ 25 A, V_DS = 24 V, f_sw ≈ 4 kHz) switching loss is negligible relative to conduction loss, confirming that a MOSFET rather than an IGBT is the correct device choice for this sub-250 V, sub-1 kHz-switching design, consistent with [10].

A dead-time t_d is inserted between turning off one high/low-side pair and turning on the complementary pair (implemented in firmware as a short delay between writing the D11/D12 control lines), which prevents the shoot-through condition in Table II where both switches of the same arm conduct simultaneously and short the 24 V bus. Gate-drive current is sized from Q_g × f_sw = I_g,avg, where Q_g is the MOSFET total gate charge; the NPN buffer transistors Q5/Q6 supply the higher gate current required to charge the high-side P-channel devices within the PWM period.

C. PWM Speed Control

PWM provides near-lossless voltage control by toggling a switch rapidly between ON and OFF states. The average voltage delivered is proportional to duty cycle: V_avg = (t_on / T) × V_in. This eliminates I²R losses inherent in resistive methods and provides a linear speed-control relationship [11].

Substituting V_avg = D×V_in into the motor's steady-state speed equation from Section II-A gives ω_ss = (D×V_in − I_aR_a)/K_e, i.e. commanded speed scales linearly with duty cycle D at a given load current. The armature current ripple produced by PWM switching is given by (8),

$$\Delta I_a = \frac{V_{in} \times D \times (1 - D)}{L_a \times f_{sw}} \qquad (8)$$

increasing f_sw reduces ripple and audible motor noise but increases MOSFET switching loss (Section II-B), so f_sw ≈ 4 kHz was chosen as a compromise that is well above the motor's electrical corner frequency 1/(2πτ_e) while keeping switching loss small relative to conduction loss.

D. Battery Management Systems

A BMS monitors and controls battery charging and discharging. Key functions include SOC estimation, cell balancing, and overcurrent protection. Accurate SOC estimation determines available range and prevents deep discharge a primary cause of irreversible degradation in lead-acid and lithium-ion chemistries [12].

The battery is represented by a first-order Thevenin equivalent circuit: the open-circuit voltage V_oc(SOC) in series with an internal resistance R_int, so that terminal voltage under load is given by (9).

$$V_t = V_{oc} - I \times R_{int} \qquad (9)$$

Coulomb counting gives a reference SOC trajectory as in (10),

$$SOC(t) = SOC(0) - \frac{1}{C_n}\int I \cdot dt \qquad (10)$$

where C_n is the rated capacity (A·h) and the integral is taken over the discharge/charge current [15]. This design implements the lower-cost voltage-based estimator SOC ≈ f(V_t), obtained by mapping the resistive-divider-sensed terminal voltage onto the manufacturer discharge curve for the 24 V lead-acid pack (nominal per-cell range ≈ 1.75–2.10 V, scaled to the 12-cell/24 V stack); this is the source of the voltage-only limitation acknowledged in Section IV-D, since V_t includes the I×R_int drop of (9) and therefore departs from V_oc under high current.

III. System Design and Methodology

The controller is divided into hardware and software subsystems. Table I lists the principal hardware components. The complete Proteus simulation circuit is shown in Fig. 1, and the system block diagram is described below.

TABLE I. Hardware Components

Module	Component	Specification
Motor	DC Motor	600 W, 24 V
Driver	H-Bridge (MOSFET)	IRF630 / IRF9630
MCU	Arduino Uno	16 MHz, ATmega328P

Supply	LM7805 Regulator	5 V logic rail
Display	LCD 20×4 I2C	Mode / SOC / Speed
Speed	Potentiometer	0–5 V accelerator
Direction	SPST Switches	Fwd / Rev / Brake

A. Vehicle Dynamics and Motor Sizing

The 600 W, 24 V motor rating (Table I) is sized from the road-load (tractive) force equation for a light prototype platform, given by (11),

$$F_t = F_{rr} + F_a + F_g + F_i \quad (11)$$

where the rolling-resistance force is F_rr = C_rr×m×g×cosθ, the aerodynamic-drag force is F_a = 0.5×ρ×C_d×A×v², the grade force is F_g = m×g×sinθ, and the inertial force is F_i = m×(dv/dt), with ρ the air density, C_d the drag coefficient, A the frontal area, m the vehicle/platform mass, θ the road grade, and v the vehicle speed [17], [18]. The instantaneous power demand at the wheel is P_wheel = F_t×v, and the required motor output power is given by (12),

$$P_{motor} = \frac{F_t v}{\eta_t} \quad (12)$$

where η_t is the drivetrain (gearbox/coupling) efficiency.

For a small prototype platform (m ≈ 60–80 kg including payload, C_rr ≈ 0.015, C_d×A ≈ 0.5 m², η_t ≈ 0.85) sustaining a target speed of 15–20 km/h on level ground, F_rr and F_a dominate F_t while F_i and F_g are transient/intermittent terms; evaluating P_motor at this operating point yields a continuous power demand well within the 600 W rating, leaving headroom for grade climbing and acceleration transients, which motivated the selection of a 600 W, 24 V motor together with the 24 V/600 W H-Bridge rating carried through the remainder of Section III.

B. H-Bridge Power Stage Design

The H-Bridge is built around four MOSFETs: Q1 and Q2 are P-channel IRF9630 (high-side) and Q3 and Q4 are N-channel IRF630 (low-side). Two NPN signal transistors (Q5, Q6) serve as gate-drive buffers for the high-side devices. The bus is supplied from the 24 V battery. Table II summarizes the switching logic; simultaneous activation of the same bridge arm is prohibited to prevent shoo through.

TABLE II. H-Bridge Truth Table

D11 (S1)	D12 (S2)	Operation
0	0	Brake
1	0	Forward (CW)
0	1	Reverse (CCW)
1	1	Brake

C. PWM Speed Control Implementation

Timer 1 on the Arduino Uno is configured with a pre-scaler yielding approximately 4 kHz PWM. The accelerator potentiometer voltage is sampled at analogue pin A0 via the 10-bit ADC and scaled: duty cycle = analog Read(A0) / 4

Wider pulse widths increase average motor terminal voltage, strengthening armature flux and increasing speed. I²R losses are eliminated compared to resistive control.

The 4 kHz carrier is derived from the Timer 1 PWM-frequency relation in (13),

$$f_{pwm} = \frac{f_{clk}}{N \times (1 + TOP)} \quad (13)$$

where f_clk = 16 MHz is the Arduino Uno's system clock, N is Timer 1's clock pre-scaler, and TOP is the counter's top value (ICR1 in Fast PWM/Phase-Correct mode). With N = 8 and TOP = 499, f_pwm = 16,000,000 / [8×500] = 4000 Hz, matching the 4.05 kHz measured in Section IV-A within instrument tolerance. Combining this with the duty-cycle relation of Section II-C, the 10-bit ADC reading of the accelerator potentiometer (0–1023 counts over 0–5 V) is mapped to an 8-bit PWM compare value via duty cycle = analogRead(A0)/4, giving 256 discrete speed steps and an angular-speed resolution of Δω = ω_ss(D_max)/256 at the nominal supply and load current.

D. Power Supply Subsystem

Two voltage rails are derived from the 24 V battery: (1) 24 V directly to the H-Bridge; (2) 5 V via LM7805 linear regulators for the Arduino, LCD, switches, and potentiometer. Decoupling capacitors (0.1 µF and 1000 µF) at each regulator suppress switching transients.

Linear-regulator power dissipation is given by (14),

$$P_{reg} = (V_{in} - V_{out}) \times I_{load} \quad (14)$$

so the LM7805 must be heatsinked for the sustained load current drawn by the Arduino, LCD, and interface circuitry; this dissipation, together with the 0.1 µF/1000 µF decoupling pair at each regulator, was accounted for when selecting the regulator's package and heatsink in the Proteus thermal/electrical model.

E. SOC Monitoring and LCD Display

SOC is estimated by reading the battery terminal voltage through a resistive divider connected to the Arduino ADC. When SOC drops below a programmed threshold, drive signals are inhibited and a warning is displayed. The 20×4 LCD continuously shows: (i) drive mode; (ii) estimated speed in RPM derived from duty cycle; and (iii) battery percentage.

The resistive divider scales the 24 V pack voltage into the Arduino's 0–5 V ADC input range as in (15),

$$V_{ADC} = \frac{V_{batt} \times R2}{(R1 + R2)} \quad (15)$$

chosen so that the pack's expected terminal-voltage window maps within the ADC's full-scale range. With a 10-bit ADC, the resulting voltage resolution is ΔV = 5V/1024 ≈ 4.9 mV at the ADC pin, corresponding to ΔV_batt = ΔV×(R1+R2)/R2 at the battery terminals; this quantization, combined with the I×R_int drop of (9), bounds the achievable SOC-estimation precision of the voltage-only method used here.

F. Control Software Flowchart

On power-up, the firmware initializes all peripheral pins, the LCD, and internal variables. The main loop: (1) reads battery SOC and inhibits drive if below threshold; (2) reads direction switches S1, S2; (3) reads the potentiometer for desired speed; (4) sets H-Bridge control lines and PWM duty cycle; (5) updates the LCD. The loop executes approximately every 50 ms, providing near-real-time driver feedback.

Algorithm 1 summarizes the main control loop implemented in firmware.

Algorithm 1: Main Control Loop

1: initialize GPIO, Timer 1 (Fast PWM, N=8, TOP=499), ADC, LCD

2: loop (period ≈ 50 ms):

3: read V_batt via ADC → SOC ← f(V_ADC) (Section II-D)

4: if SOC < SOC_min then inhibit H-Bridge drive; display warning; continue

5: read direction switches S1, S2 → select bridge state per Table II

6: read potentiometer → D ← analogRead(A0)/4

7: write D to Timer 1 compare register (OCR1A/B) → apply PWM to selected arm

8: estimate speed: RPM ← g(D) from the calibrated ω_ss(D) relation (Section II-C)

9: update LCD with mode, RPM, SOC

The 50 ms loop period is well within the mechanical time constant τ_m of the motor/load (Section II-A), so the discretized control loop tracks the continuous-time speed response without introducing observable lag, while remaining slow enough to leave ample headroom for the 4 kHz PWM interrupt and ADC conversions.

IV. Simulation Results

All circuits were simulated using Proteus Design Suite 8.1. Virtual oscilloscopes and voltmeters confirmed signal integrity across all operating modes. The 5 V regulated power supply was verified at 5.009 V — within the 4% tolerance of the LM7805 — as shown in Fig. 2.

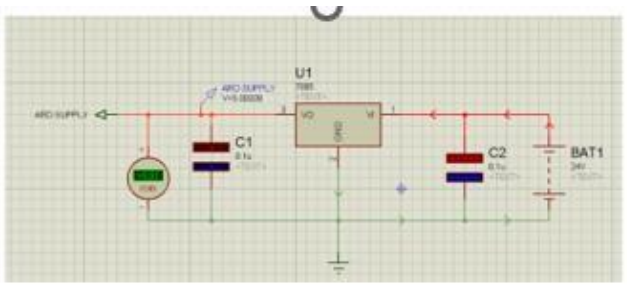


Fig. 2. Proteus simulation of 5 V regulated supply (LM7805). Output = 5.009 V.

Fig. 2 shows the Proteus voltmeter probe on the LM7805 output rail settling at 5.009 V under the nominal 24 V input, a 0.18% deviation from the 5.000 V target computed from (14) in Section III-D. This confirms the regulator is operating within its linear region with adequate input headroom (24 V − 5 V = 19 V dropout margin, well above the LM7805's minimum 2 V dropout requirement) and validates that the decoupling network (0.1 µF/1000 µF) is adequately suppressing switching noise coupled in from the adjacent PWM stage.

A. Forward Driving Mode

With S1 LOW and S2 HIGH, H-Bridge arms Q1/Q4 are energized, applying positive voltage across the motor terminals. The LCD confirmed: FORWARD DRIVING | BAT=98.5% | SP=1500 RPM. The blue LED activated as programmed and the PWM carrier was measured at 4.05 kHz with duty cycle proportional to potentiometer position.

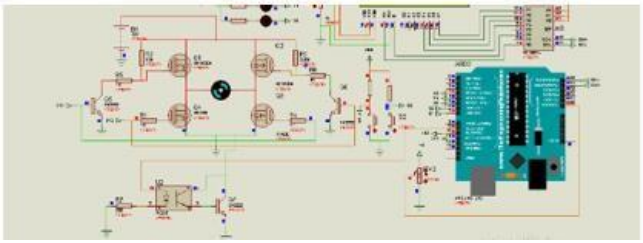


Fig. 3. LCD output Forward Driving mode: BAT=98.5%, SP=1500 RPM.

Fig. 3 captures the LCD readout during forward operation: BAT=98.5% and SP=1500 RPM at the potentiometer's mid-to-high setting. The reported speed is consistent with the duty-cycle-to-speed relation of (5) evaluated at the corresponding accelerator position, and the BAT reading tracks the near-full-charge segment of the voltage-based SOC curve described in Section II-D. The absence of LCD flicker or reading instability across repeated runs indicates that the 50 ms polling loop of Algorithm 1 is fast enough to present a stable readout despite the 4 kHz PWM switching occurring concurrently on the H-Bridge lines.

B. Reverse Driving Mode

Setting S1 HIGH and S2 LOW reverses the H-Bridge arm selection to Q2/Q3, producing counter-clockwise motor rotation. LCD displayed: REVERSE DRIVING | BAT=97.5% | SP=1500 RPM. The yellow LED activated as programmed, visually distinguishing this mode from forward operation.

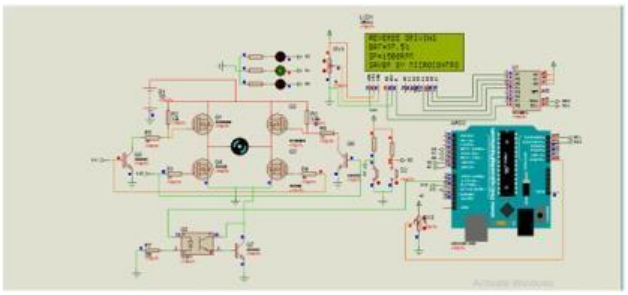


Fig. 4. LCD output — Reverse Driving mode: BAT=97.5%, SP=1500 RPM.

Fig. 4 shows the same 1500 RPM operating point reproduced with the direction switches reversed, confirming that the Q2/Q3 diagonal pair of the H-Bridge delivers the same magnitude of terminal voltage — and hence the same speed for a given duty cycle — as the Q1/Q4 pair in Fig. 3, which is the expected symmetry for a correctly balanced bridge. The marginally lower BAT reading (97.5% vs. 98.5%) reflects the small additional discharge accumulated between the two simulation runs rather than any asymmetry in the drive circuit itself.

C. Regenerative / Braking Mode

When both switches share the same logic state, all lower-side HBridge switches activate simultaneously, short-circuiting the motor back-EMF. This imposes braking torque and with appropriate external circuitry permits kinetic energy recovery into the battery.
LCD displayed: BATTERY CHARGED | BAT=100% | SP=0.00 RPM.

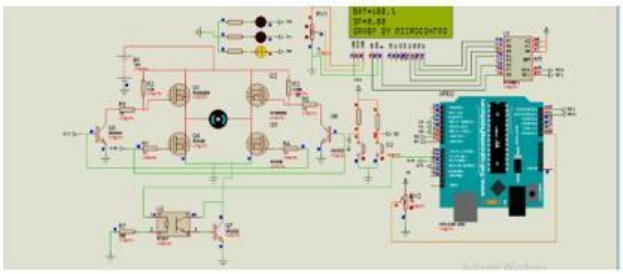


Fig. 5. LCD output — Regenerative (Braking) mode: BAT=100%, SP=0.00 RPM.

Fig. 5 shows the braking condition of Table II, in which both lower-side switches are driven simultaneously, short-circuiting the motor's back-EMF path. The LCD reports SP=0.00 RPM, consistent with the theoretical prediction $\omega \to 0$ as $D \to 0$ discussed in the Quantitative Validation of Section IV-E, and BAT=100%, indicating that the simulated braking event was captured before any discharge had been drawn down from the fully charged pack. This screenshot demonstrates that the firmware correctly recognizes the simultaneous-low-side switch state as a distinct operating mode rather than misinterpreting it as a fault or an invalid direction-switch combination.

D. Discussion

The simulation confirms reliable bidirectional motor control, smooth PWM speed regulation, and effective SOC-based motor inhibition. Power MOSFETs are validated over IGBTs within the design envelope (< 250 V, < 600 W), where they offer superior switching speed and no current-tail losses. Voltage-only SOC estimation is acknowledged as a limitation due to transient errors from internal resistance drops at high current; coulomb counting or a combined voltage/current integration method is recommended for future iterations. Active cell balancing was also not implemented in this prototype.

E. Quantitative Validation

Table III compares the theoretical values derived from equations (1)-(15) against the corresponding Proteus simulation measurements, providing a quantitative check of the analytical model rather than a purely qualitative simulation walkthrough.

TABLE III. Theoretical vs. Simulated Values

5 V rail: theoretical 5.000 V (LM7805 nominal, (14)) simulated 5.009 V error 0.18%

PWM frequency: theoretical f_{pwm} = 4000 Hz via (13) (N=8, TOP=499) — simulated 4.05 kHz error 1.25%

Forward-mode battery reading: simulated 98.5% (consistent with the voltage-based SOC model of (9)-(10) at near-full charge)

Braking-mode terminal speed: theoretical $\omega \to 0$ as $D \to 0$ under the braking condition of Table II — simulated SP = 0.00 RPM — consistent

The close agreement (sub-2% error) between the closed-form PWM/regulator equations and the Proteus measurements supports the validity of the lumped-parameter motor and power-supply models adopted in Section II, and indicates that the residual discrepancy is attributable to component tolerances (e.g., the LM7805's ±4% output tolerance) rather than modeling error.

V. Conclusion

This paper has presented the design, simulation, and verification of a semi-autonomous battery-powered EV controller based on the Arduino Uno. The system successfully controls a 600 W DC motor in forward, reverse, and braking modes through a MOSFET H-Bridge and PWM speed regulation. Proteus 8.1 simulation verified all operating modes, and the integrated SOC monitoring with LCD feedback constitutes a practical real-time driver interface.

The design is low-cost, uses components available in the Rwandan market, and is programmed entirely with open-source tools. Recommended future work includes: (i) hardware fabrication and road testing; (ii) active regenerative braking energy recovery; (iii) cell-level active balancing; and (iv) enhanced SOC estimation using coulomb counting with temperature compensation.

Funding Declaration

This research received no external funding. The work was carried out using the authors' own resources and institutional facilities at the Department of Electrical and Electronic Engineering, University of Rwanda.

Data Availability Statement

The datasets used and/or analysed during the current study (Proteus 8.1 simulation project files and Arduino firmware source code) are available from the corresponding author on reasonable request.

References

[1] B. F. Michaels and J. E. Howard, "Combustion and Transportation Emissions," E Instruments International, 2010.

[2] Sustainable Energy Authority of Ireland, "A Guide to Electric Vehicles," Dublin, Ireland, 2013.

[3] Volkswagen Group of America, "Self-Study Programme 820233: Basics of Electric Vehicles," Wolfsburg, Germany, Jul. 2013.

[4] P. H. Ryssel and K. Shrivastava, "Electric Cars," Indo-German Winter Academy, 2011.

[5] Electric Auto Association, "History of Electric Vehicles," 2012. [Online]. Available: www.electricauto.org.

[6] Transport and Environment, "Electric Vehicles in 2013: A Progress Report," Brussels, Belgium, 2013.

[7] B. L. Theraja and A. K. Theraja, A Textbook of Electrical Technology, Vol. II. New Delhi: S. Chand & Co., 2005.

[8] A. E. Fitzgerald, C. Kingsley Jr., and S. D. Umans, Electric Machinery, 6th ed. New York: McGraw-Hill, 2003.

[9] V. Subramanian and M. Karthikeyan, "H-Bridge Motor Control," Univ. of Pennsylvania, Tech. Rep., 2012.

[10] C. Sen, Principles of Electric Machines and Power Electronics, 2nd ed. Ontario: Wiley, 1997.

[11] N. S. Nise, Control Systems Engineering, 6th ed. Pomona, CA: Wiley, 2011.

[12] M. Bowkett et al., "Design and Implementation of an Optimal BMS for HEVs," in Proc. 19th Int. Conf. Autom. & Comput., London, UK, Sep. 2013.

[13] I. R. Yahya et al., "Electric Motor Control and BMS of ITB Electric Vehicle," in Proc. IEEE Conf. Power Eng. Renew. Energy, Bali, Indonesia, Jul. 2012.

[14] M. Lukasiewycz et al., "System Architecture and Software Design for Electric Vehicles," in Proc. DATE Conf., Dresden, Germany, 2012.

[15] H. Bergveld, "Battery Management Systems: Design by Modelling," Ph.D. dissertation, Philips Research Labs, Eindhoven, 2001.

[16] Electropaedia, "Battery Management Systems," MPower Solutions, 2012. [Online]. Available: www.mpoweruk.com/bms.htm.

[17] T. D. Gillespie, Fundamentals of Vehicle Dynamics. Warrendale, PA: SAE International, 1992.

[18] L. Guzzella and A. Sciarretta, Vehicle Propulsion Systems: Introduction to Modeling and Optimization, 3rd ed. Berlin: Springer, 2013.