



Hybrid Battery–Supercapacitor Energy Management System for Electric Vehicles with Integrated AI

Dr. B. S. Dani¹, Mr. Sagar Tetu², Mr. Rohan Hattimare³, Mr. Harsh Markand⁴, Mr. Rajat Lonare⁵, Mr. Ayush Bhorkar⁶

¹ Assistant Professor (Guide) Department of Electrical Engineering Priyadarshini College of Engineering, Nagpur
bhagyashreedani@gmail.com

² Student Department of Electrical Engineering Priyadarshini College of Engineering, Nagpur
sagartetu@gmail.com

³ Student Department of Electrical Engineering Priyadarshini College of Engineering, Nagpur
rohanhattimare19@gmail.com

⁴ Student Department of Electrical Engineering Priyadarshini College of Engineering, Nagpur
tonymarkand@gmail.com

⁵ Student Department of Electrical Engineering Priyadarshini College of Engineering, Nagpur
rajatlonare18@gmail.com

⁶ Student Department of Electrical Engineering Priyadarshini College of Engineering, Nagpur
ayushbhorkar5@gmail.com

ABSTRACT:

This paper describes the hardware of a hybrid battery–supercapacitor system for a small electric vehicle. The load keeps changing: cruising needs little power, but starting, overtaking and climbing ask for short, heavy current pulses. A lead-acid battery handles the steady part well, but repeated spikes heat it and shorten its life. A supercapacitor gives and takes high current easily, though it stores much less energy. So the battery carries the average load and a supercapacitor bank is kept ready for peaks and braking energy.

The prototype has a 12 V lead-acid battery, a series bank of 2.7 V, 500 F supercapacitor cells with balancing, a 250 W, 12 V, 1300 rpm BLDC motor, a motor driver, relay modules, voltage and current sensors, a gyrometer for slope, a small DC–DC supply and an Arduino controller. Each component is described with a picture, its rated parameters and its job in the project, and the output of the monitoring dashboard is shown at the end.

I. HARDWARE OVERVIEW

Fig. 1 shows the whole prototype. Energy comes from the lead-acid battery and the supercapacitor bank, which meet on one DC bus. The motor driver takes power from that bus and runs the BLDC motor. During braking the motor acts as a generator and current returns to the bus (dashed path). The Arduino reads the voltage sensor, current sensor, gyrometer and throttle potentiometer, decides which relay should close, and sends the PWM signal to the driver. The power path is kept simple on purpose: the two sources are joined through relays and fuses, with no DC–DC converter in the energy-management stage.

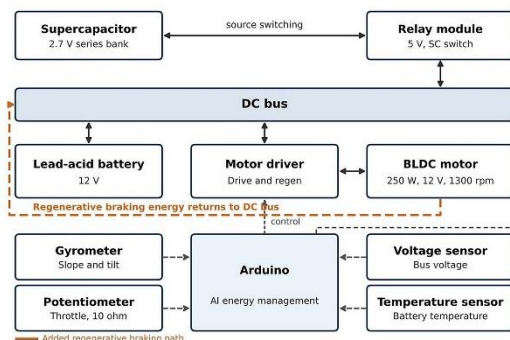


Fig. 1. Block diagram of the proposed hybrid battery–supercapacitor EV system.

Table I lists the hardware. Values marked † are typical manufacturer datasheet values for this class of part and should be checked against the label of the

actual unit; this applies to the † entries in all later tables.

TABLE I. Summary of the hardware used in the prototype.

Component	Key rating	Job in the system
Lead-acid battery	12 V	Average energy
Supercapacitor bank	5×2.7 V, 500 F in series (13.5 V, 100 F)	Peaks and regeneration
BLDC motor	250 W, 12 V, 1300 rpm	Traction load
Motor driver	PWM, 8-bit (0–255)	Drive and regen
Arduino controller	10-bit ADC, 5 V logic†	Decisions, logging
Voltage sensor	0–25 V in, 0–5 V out†	Source voltages
Current sensor	± 30 A, 66 mV/A†	Motor current
Relay module	5 V coil, 10 A contacts†	Source switching
DC–DC converter	12 V in, 5 V out, 3 A†	Logic supply
Gyrometer	3-axis gyro + accel†	Road slope

II. HARDWARE COMPONENTS

2.1 Lead-Acid Battery

The battery is the main energy source. We used a small sealed 12 V lead-acid unit (Fig. 2). It is cheap, easy to find, simple to charge and almost fully recyclable [3], which matters for a student-level vehicle. Six cells of about 2.1 V give the nominal 12 V, and a rested, fully charged battery shows 12.6 to 12.8 V, so voltage can serve as a rough guide to state of charge.



Fig. 2. Lead-acid battery on the prototype board.

Its weak point is high current. The heat lost in the internal resistance is $P = I^2R$, so doubling the current quadruples the loss, the terminal voltage sags, and the usable capacity falls [2], [3]. This is the reason a supercapacitor is added. Capacity is not listed in Table II because it must be copied from the label of the installed battery; the state-of-charge calculation depends on it.

TABLE II. Rated parameters of the lead-acid battery.

Parameter	Value
Nominal voltage	12 V
Full charge, at rest	12.6–12.8 V†
Discharge cut-off	About 10.5 V†
Internal resistance	15–30 mΩ†
Cycle life (50 % depth)	200–300 cycles†
Capacity	Printed on the label

The dashboard shows battery voltage, a state-of-charge estimate and a state-of-health estimate (from internal resistance). None is exact, but together they give a fair idea of battery condition. A reading of 11.8 V under load does not mean the battery is nearly empty, so the controller should combine voltage

with integrated current. Deep discharge below about 10.5 V causes sulphation, and the battery sits behind a fuse and a relay. Lithium-ion would be lighter and longer-lived, but needs a battery management system and costs far more; for a low-cost prototype whose purpose is to study how a supercapacitor relieves the battery, lead-acid was the practical choice [2], [8].

In operation the battery supplies the normal-driving load and the slow part of every other mode, while the supercapacitor is added during acceleration so the battery current stays near its average.

2.2 Supercapacitor Bank

The bank is the second energy source and the reason the project exists. It stores energy in an electric field, so it charges and discharges very quickly and survives a very large number of cycles [4], [7], at the price of low energy density.

Each cell is rated 2.7 V and 500 F; Fig. 3 shows five joined in series with balancing. Series connection adds voltages and divides capacitance, so the bank is $5 \times 2.7 = 13.5$ V and $500/5 = 100$ F. The energy is $E = \frac{1}{2}CV^2 \approx \frac{1}{2} \times 100 \times 13.5^2 \approx 9.1$ kJ, or about 2.5 Wh. Letting the voltage fall from 13.5 V to 6.75 V uses 75 % of this, about 1.9 Wh (Fig. 4).



Fig. 3. Series-connected supercapacitor cells on the prototype board.

TABLE III. Rated and calculated parameters of the supercapacitor bank.

Parameter	Value
Cell rating	2.7 V, 500 F
Cells in series	5 (as built)
Bank voltage, maximum	13.5 V
Bank capacitance	100 F
Stored energy, full	≈ 9.1 kJ (2.5 Wh)
ESR per cell	1–2 m Ω †
Cycle life	500 000 or more†

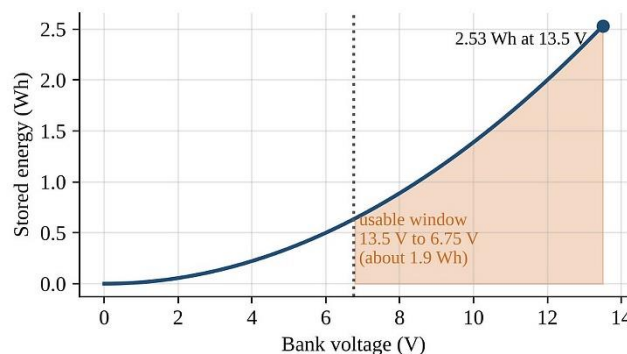


Fig. 4. Energy stored in the 100 F bank against its voltage.

Balancing is needed because no two cells are identical: the cell with the lowest capacitance reaches 2.7 V first and its life drops quickly. A resistor or Zener across each cell shares the voltage evenly [7].

Two cautions apply. First, an empty bank is almost a short circuit; switched straight onto a 12 V battery it could draw hundreds of amperes, so a pre-charge resistor or soft-start is needed before the main contact closes. Second, a charged bank can deliver very high current if shorted, so terminals are covered, a fuse sits close to the bank, and it is discharged through a resistor before anyone works on the wiring.

Because there is no DC–DC converter, the bank helps the motor only while its voltage is close to the bus voltage. This is a limitation we accept for low cost. During regeneration it is an advantage: a half-empty bank sits below battery voltage and can accept current at a lower back-EMF. Compared with the battery, the supercapacitor stores about two orders of magnitude less energy per kilogram but handles far more power, cycles and efficiency [4], [5]. It handles three jobs here: acceleration peaks, sharing the load on a steep slope, and absorbing braking energy. Its voltage is also the fuel gauge of the bank.

2.3 BLDC Traction Motor

The prototype uses a 12 V, 250 W BLDC motor with rated speed 1300 rpm, coupled by a chain to a heavy steel disc (Fig. 5). The disc plays the role of vehicle mass, giving the motor inertia to accelerate and something to brake against, so regeneration can be studied without a real vehicle. A BLDC motor has better efficiency, less maintenance and no brush noise, which would otherwise enter the analog sensor readings; it also works well as a generator for regenerative braking [1]. The price is that the driver must know the rotor position, from Hall sensors or back-EMF.

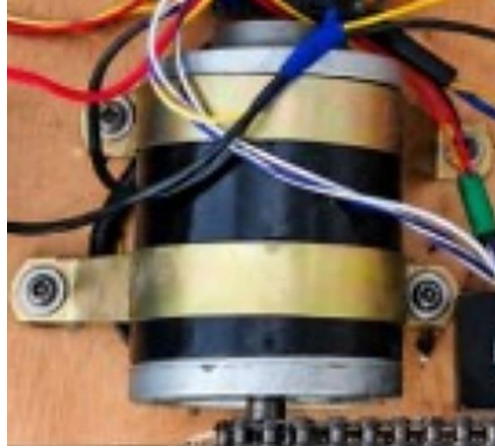


Fig. 5. Motor mounted on the board, with the chain drive below it.

From the rating, $1300 \text{ rpm} \approx 136 \text{ rad/s}$, so rated torque is $250/136 \approx 1.8 \text{ N}\cdot\text{m}$, and rated current at 12 V is at least $250/12 \approx 21 \text{ A}$ (a little more in reality). The back-EMF constant is $k_e \approx 12/136 \approx 0.09 \text{ V}\cdot\text{s/rad}$, so the back-EMF reaches the bus voltage only near rated speed; below that, regeneration relies on the driver's regenerative mode and a supercapacitor sitting at a lower voltage. The dashboard speed (Fig. 6) is calculated from the PWM value, not measured; a Hall or encoder feedback would be a useful next step. Everything rated above 21 A (wiring, driver, relays) must be chosen accordingly.

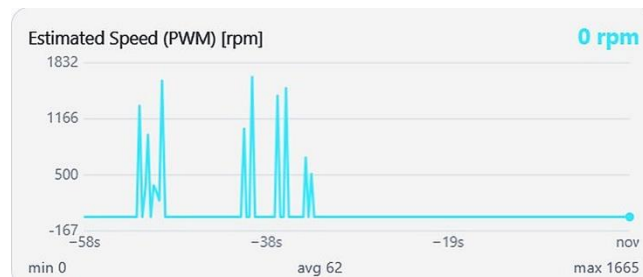


Fig. 6. Estimated motor speed (from PWM) during two test runs.

TABLE IV. Rated and calculated parameters of the motor.

Parameter	Value
Rated power / voltage	250 W / 12 V
Rated speed	1300 rpm
Rated torque	$\approx 1.8 \text{ N}\cdot\text{m}$
Rated current (ideal)	$\approx 21 \text{ A}$
Back-EMF constant	$\approx 0.09 \text{ V}\cdot\text{s/rad}$
Drive	PWM through driver

2.4 Motor Driver

The Arduino gives only a 5 V signal at a few milliamperes, while the motor takes over twenty amperes. The driver bridges the two, using power transistors or MOSFETs in a bridge arrangement (Fig. 7). It works by pulse width modulation: the motor inductance smooths the pulses, so the motor sees the duty cycle times the bus voltage; at 50 % duty a 12 V bus gives about 6 V on average (Fig. 8). The Arduino writes the duty as an 8-bit number (0–255), and 128 is about 50 %.



Fig. 7. Driver stage on the prototype board.

The default Arduino PWM frequency is about 490 Hz, which is audible as a hum; a final version should use a few kilohertz up to about 20 kHz, trading lower ripple against higher switching loss. The driver needs dead time so the upper and lower transistors of a leg are never on together, and a heatsink with some airflow, since the loss is $I^2R(\text{on})$ plus switching loss. In regeneration the driver returns current to the bus, and the source receiving it must accept it; a battery near full pushes back, which is where the supercapacitor helps. The bus voltage is not fixed (about 12 V on the battery, up to 13.5 V with a charged bank, falling as the bank discharges), so the under-voltage lock-out must be set well below the lowest planned voltage. Protection is a bus fuse and the controller current reading; a hardware over-current trip is a worthwhile future addition because the Arduino software loop is slow against a short circuit.

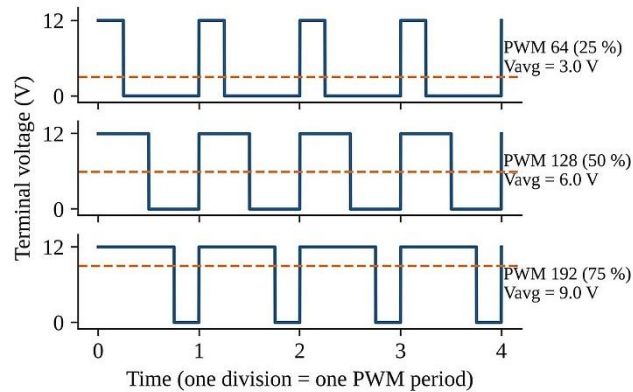


Fig. 8. PWM waveforms at three duty values and their average voltage.

TABLE V. Rated and required parameters of the driver stage.

Parameter	Value
Supply	12 V from the DC bus
Continuous current	Above 21 A, we suggest $\geq 25 \text{ A}^\dagger$
Control input	PWM, 5 V logic
Duty resolution	8-bit (0–255)
PWM frequency	$\approx 490 \text{ Hz}$ (default timer) †
Regeneration	Supported (as designed)

2.5 Arduino Controller

The Arduino is the brain of the prototype: an ATmega328P at 16 MHz with 5 V logic [11] (Fig. 9). It was chosen because it is cheap, familiar to every student in the group, and has the needed inputs and outputs without extra chips. The voltage and current sensors and the throttle potentiometer (Fig. 11) are read by the 10-bit ADC (one count = $5/1024 \approx 4.9 \text{ mV}$); the gyrometer is read over I2C. Fig. 10 summarises the inputs and outputs: the outputs are the PWM signal for the driver, the relay lines, the LCD and the USB serial port.



Fig. 9. Arduino board and the 16×2 LCD on the prototype.

TABLE VI. Rated parameters of the controller.

Parameter	Value
Microcontroller / clock	ATmega328P† / 16 MHz†
Logic level	5 V
ADC	10-bit, 0–1023
PWM outputs	6 channels, 8-bit†
Current per pin	20 mA continuous†
Serial link	9600 baud, JSON text

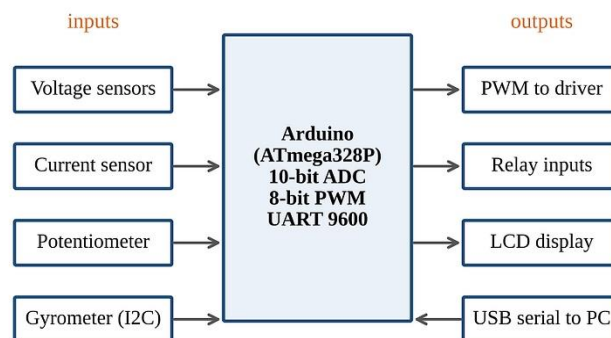


Fig. 10. Inputs and outputs of the controller.

The throttle potentiometer reads about 1023 at rest and falls as the throttle is turned, so the PWM value is computed from the inverted reading. The controller runs a rule-based program that compares voltages, current and slope with thresholds and chooses battery-only, battery plus supercapacitor, or regenerative mode. The same data could later train a small prediction model, but the present hardware does not depend on it. The dashboard can send SRC:SC or SRC:BAT to change the relays.

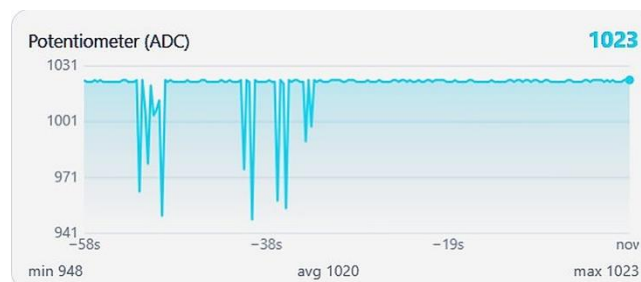


Fig. 11. Throttle potentiometer reading logged by the dashboard (two test runs).

Telemetry leaves as a JSON line at 9600 baud, which gives only about 3.4 samples per second. This is fine for voltage and tilt but too slow for millisecond current spikes; a shorter message or 115200 baud would raise the rate many times without hardware changes. An Arduino is enough for this prototype because the rules are simple and timing is not critical, but a real vehicle needs a faster controller with hardware safety features. Practical points: the ADC reference is the 5 V supply, which moves with USB and motor load, and the first reading after a channel switch can be wrong, so it should be read twice.

Motors and relays add noise, so short or twisted signal wires, a common ground point and decoupling capacitors are used; the present wiring is untidy and a cleaner board is advisable before long tests.

2.6 Voltage Sensor

An Arduino pin accepts only 0 to 5 V, while the sources reach 12–13.5 V. Both are measured through a voltage-sensor module, which is a resistor divider with 30 k Ω and 7.5 k Ω [9] (Fig. 12). Then $V_{out} = V_{in} \times R2/(R1 + R2) = V_{in}/5$, so the module measures up to 25 V and the full bank (13.5 V) becomes 2.7 V at the pin. One ADC count is $4.9 \text{ mV} \times 5 \approx 24 \text{ mV}$ at the input, ample for tracking 0.1 V; accuracy is limited more by the 1 % resistors and the reference voltage. Calibration against a multimeter at one low and one high point gives a correction factor for the code.

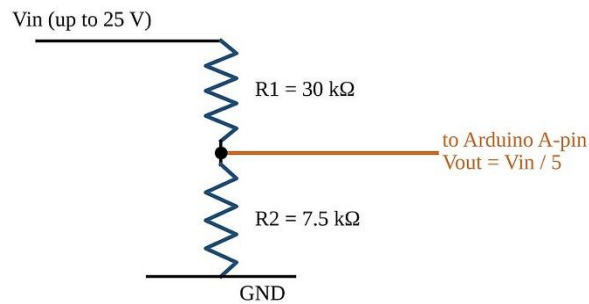


Fig. 12. Resistor-divider circuit of the voltage sensor module.

TABLE VII. Rated parameters of the voltage sensor.

Parameter	Value
Input range / divider	0–25 V† / 30 k Ω and 7.5 k Ω (ratio 5)†
Output range	0–5 V
Resolution (10-bit ADC)	$\approx 24 \text{ mV}$ per count
Resistor tolerance	1 %†

The divider draws $12/37\,500 \approx 0.32 \text{ mA}$, negligible for the battery but an added drain on an idle bank left connected overnight. The sensor is not isolated and shares ground with the Arduino, so its ground should be taken from a quiet point near the battery negative terminal and not from a wire carrying motor current. This is acceptable at 12 V; a higher-voltage vehicle would need isolated measurement. The voltage readings decide when the supercapacitor connects, when it is full enough to accept regeneration, and when the battery is too low.

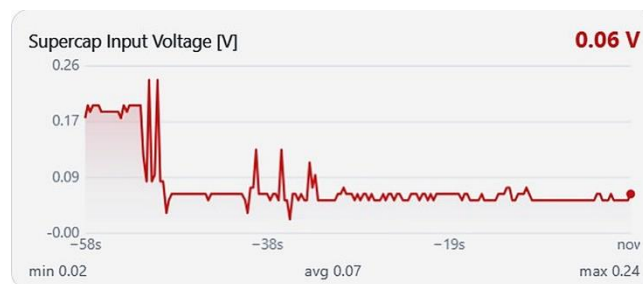


Fig. 13. Supercapacitor input voltage as shown on the dashboard.

Fig. 13 shows the bank voltage during a test. The bank was nearly empty, between about 0.05 V and 0.25 V, and the trace moves slightly each time the motor runs. The dashboard also draws a headroom signal (bank voltage minus a configured limit); here the limit appears to be 12.5 V, since 0.06 V gives -12.44 V .

2.7 Current Sensor

Motor current is measured by a Hall-effect module of the ACS712 type [10]. The current flows through a copper path inside the chip, and the Hall element senses its magnetic field, so the measuring side is isolated from the high-current side. The output sits at half the supply (about 2.5 V) at zero current and moves by 66 mV per ampere for the 30 A version, so $V_{out} = 2.5 + 0.066 \times I$ (Fig. 14). This allows negative readings, which indicate regeneration. One ADC count is $4.9/66 \approx 0.074 \text{ A}$, fine for a 20 A motor but not for tiny currents.

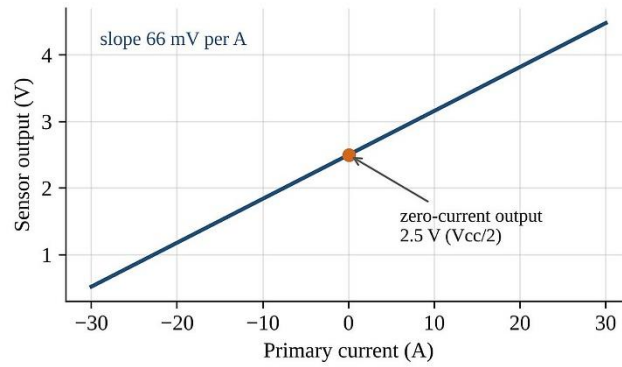


Fig. 14. Transfer curve of the 30 A Hall-effect current sensor.

TABLE VIII. Rated parameters of the current sensor.

Parameter	Value
Measuring range	$\pm 30 \text{ A}^\dagger$
Sensitivity	$66 \text{ mV/A}^\dagger$
Zero-current output	$2.5 \text{ V (Vcc/2)}^\dagger$
Total error	about $\pm 1.5 \%$ [†]
Isolation	$2.1 \text{ kV rms}^\dagger$
Resolution (10-bit ADC)	$\approx 0.074 \text{ A per count}$

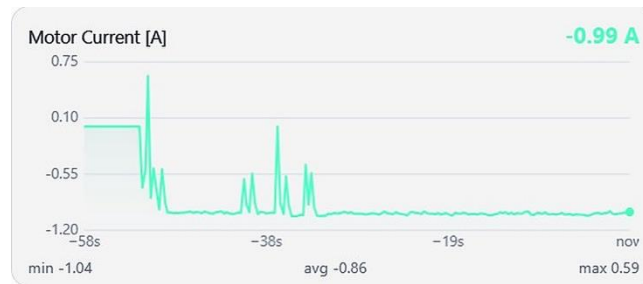


Fig. 15. Motor current as logged by the dashboard.

Fig. 15 shows the logged current. At standstill the dashboard reads about -1 A instead of 0 A . Because the zero level is $V_{cc}/2$, a USB-powered board at 4.87 V instead of 5.00 V shifts the offset by about 65 mV , which is about 1 A . We expect that measuring the offset at start-up with the motor stopped and subtracting it in the code will remove most of the error. The current also only rises to about 0.6 A , so the motor ran with almost no load; tests with the flywheel load must confirm that the relay and driver tolerate the real current. The lowest values (about -1 A) cannot yet be separated into offset and a first trace of regenerative current; a loaded run with the offset removed should answer that. Magnetic fields from the motor and relay coils shift the output, so the sensor is kept a few centimetres away. For protection, a fast hardware threshold is better than averaging, and since the sensor saturates at $\pm 30 \text{ A}$, the fuse is what actually protects the system in a short circuit.

2.8 Relay and Contactor Module

The relays connect the sources. An Arduino pin cannot close a power contact directly, so a small control signal energises a coil that closes a separate contact. Our modules are the common blue 5 V single-channel type (Fig. 16) [14]. They have an optocoupler between the Arduino pin and the coil driver, a flyback diode across the coil (Fig. 17), and are active-low (the relay closes when the input is pulled to ground).



Fig. 16. Relay modules on the prototype board.

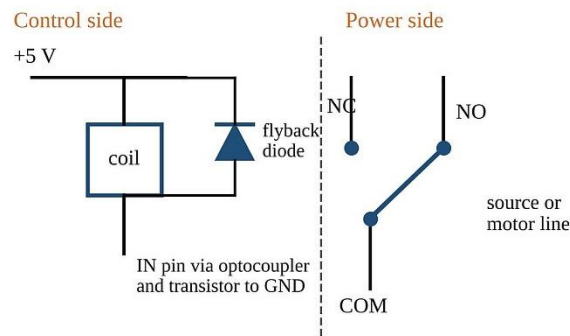


Fig. 17. Relay coil, flyback diode and changeover contact.

TABLE IX. Rated parameters of the relay module.

Parameter	Value
Coil voltage / current	5 V DC / 70–90 mA†
Contact rating	10 A at 30 V DC†
Operate / release time	≤ 10 ms / ≤ 5 ms†
Mechanical / electrical life	10 million / 100 000 operations†
Control	Active-low input, opto-isolated†

An honest limitation: the 10 A contact rating is below the rated motor current of about 21 A, and DC is harder on contacts than AC. For the full-power path these modules should be replaced by a proper DC contactor, or used only to drive its coil. For the present low-current bench tests they are acceptable.

The sources must never be closed onto the bus in a way that shorts one into the other, especially with an empty bank and a full battery, so the controller changes source in a fixed order. The dashboard shows Relay 1 (voltage) and Relay 2 (current/tilt). Closing takes about 10 ms, short compared with a vehicle manoeuvre, so the controller looks at slope and throttle in advance. Relays were chosen over MOSFETs for simplicity and complete isolation when open. Two relays take about 180 mA, a significant part of what the supply can give. A relay welded shut would leave the controller believing a source is disconnected, so the relays sit behind fuses, and a contact-feedback check should be added; the dashboard currently shows only the command, not the real contact position.

2.9 DC–DC Converter (Logic Supply)

There is no converter in the energy path; the sources are joined directly through relays. A small LM2596-type buck converter [13] only makes a steady 5 V for the Arduino, relay coils, sensors and LCD from the 12 V bus. Without it, the voltage dip when the motor starts could reset the Arduino and drop the relays at the worst time. A buck converter steps down by chopping the input with a switch, an inductor and capacitor, and a catch diode; in steady state $V_{\text{out}} = D \times V_{\text{in}}$, so the duty for 12 V to 5 V is $5/12 \approx 42\%$ (Fig. 18). Compared with a 7805 linear regulator burning $(12 - 5) \times 0.5 = 3.5$ W at 0.5 A, a buck converter at 85 % efficiency loses about 0.45 W.

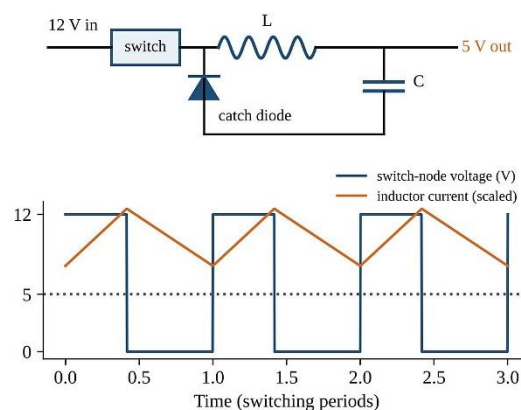


Fig. 18. Buck converter circuit and its switching waveforms for 12 V to 5 V.

TABLE X. Rated parameters of the DC–DC converter (LM2596 type) [13].

Parameter	Value
Input voltage	4–35 V†
Output voltage	5 V (adjustable 1.25–35 V)†
Output current	3 A maximum†

Parameter	Value
Switching frequency	150 kHz†
Efficiency, 12 V to 5 V	80–88 %†
Duty at 12 V to 5 V	≈ 42 %

The converter is kept away from the current sensor because its inductor field and 150 kHz switching noise disturb the Hall sensor and analog readings. Its input limit should be at least 20 V because the battery and bank can reach 13.5 V or more. A bidirectional DC–DC converter between the bank and the bus would let the system use almost all of the bank energy and control power sharing exactly [6], but adds cost, volume, losses and controller complexity; we leave it for a second version.

2.10 Gyrometer (Tilt Sensor)

The gyrometer tells the controller whether the vehicle is on a slope and by how much, so that the supercapacitor can be connected before the current peak arrives. A module with the MPU-6050 chip (3-axis gyroscope plus 3-axis accelerometer) is typical [12]; the exact module on the vehicle should be checked. The accelerometer gives the tilt angle directly but is noisy; the gyroscope is smooth but drifts when integrated. A complementary filter combines them: $\theta = \alpha(\theta + \omega \cdot dt) + (1 - \alpha) \cdot \theta_{acc}$ with $\alpha \approx 0.98$, giving the pitch angle shown on the dashboard (Fig. 19).

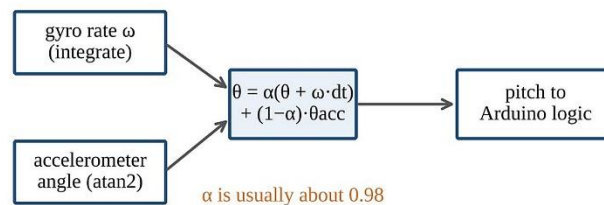


Fig. 19. Complementary filter combining gyroscope and accelerometer.

TABLE XI. Rated parameters of the gyrometer module.

Parameter	Value
Sensor	3-axis gyro + 3-axis accelerometer†
Interface	I2C, up to 400 kHz†
Accelerometer / gyro range	±2 to ±16 g† / ±250 to ±2000 °/s†
ADC resolution	16-bit†
Module supply	3.3–5 V†
Output used	Pitch angle (degrees)

Slope in per cent is $\tan \theta \times 100$, so 5° is about 8.7 % (Fig. 20). For a 100 kg vehicle on a 5° slope the extra force is about 85 N, which at 2 m/s is an extra 170 W, a lot for a 250 W motor. In the dashboard captures the pitch stays between about -0.4° and 2.2° on a flat bench, which gives the noise level; the steep-slope threshold must be set well above it. The sensor needs gyro-bias calibration at power-up, a level-surface check, a loop of 50–100 Hz using the measured time step, and a firm, flat mounting away from the motor. In the project the gyrometer selects the steep-slope mode (both sources connected) and helps recognise a descent where regenerative braking can be used. This looking ahead is what separates the controller from a simple voltage switch and is what the later AI extension would build on.

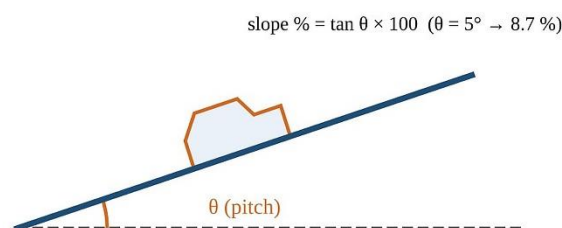


Fig. 20. Pitch angle and slope percentage ($5^\circ \approx 8.7\%$).

IV. WORKING PRINCIPLE

The system splits the job according to what each source is good at. The lead-acid battery supplies the average energy for normal driving; the supercapacitor bank is a high-power buffer for acceleration, steep slopes and braking, and its low internal resistance lets it handle large currents with little heating [4], [5]. The Arduino reads the voltages, current and tilt, and the relays connect whichever combination of sources is needed. The loss inside the battery is

$$P_{\text{loss}} = I^2 R_{\text{int}} \quad (1)$$

so if the current doubles the heating is four times larger. Letting the supercapacitor take the short peaks keeps the battery current close to its average. The energy in the supercapacitor is

$$E = \frac{1}{2} C V^2 \quad (2)$$

so the bank voltage falls while it discharges and rises while it charges, and the controller uses it as a fuel gauge to decide when to connect or release the bank.

4.1 Operating Sequence

In normal driving only the battery is connected. When acceleration suddenly raises the demand, the supercapacitor is connected to supply the extra current. When the gyrometer sees a steep slope, both sources are connected and share the heavy load. In regenerative braking the motor acts as a generator and the controller sends the recovered current to the supercapacitor, which stores it quickly and gives it back in the next high-power event. The battery thus keeps a calm, steady role while the supercapacitor reacts to fast changes.

V. PROTOTYPE

All components are mounted on one wooden board (Fig. 23): the battery, supercapacitor bank, motor with chain and flywheel, relay modules, driver stage, Arduino with LCD, and the sensors. It is a bench platform and not a full vehicle, but it lets us watch how electrical sensing and source switching interact and compare with a battery-only run later.

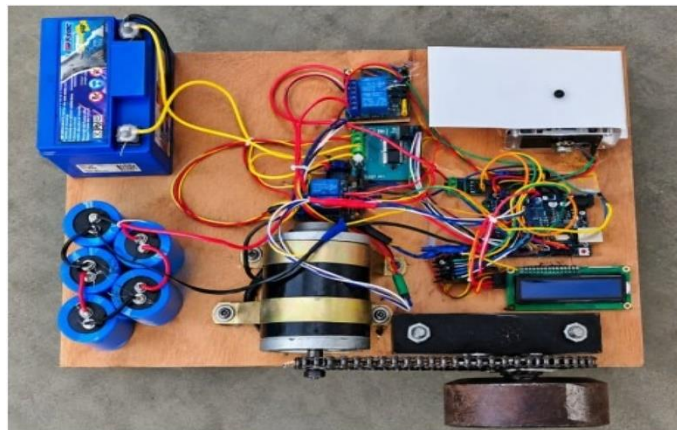


Fig. 23. Experimental prototype of the hybrid battery-supercapacitor EV system.

VI. CONCLUSION

This paper described the hardware of a hybrid battery-supercapacitor system for a small electric vehicle. The battery gives the steady energy, the 100 F, 13.5 V supercapacitor bank handles peaks and braking, the 250 W motor is the load, and the driver, Arduino, sensors, relays and small DC-DC supply let the controller see what is happening and act on it.

Working through the ratings also exposed a few things to fix: the small relay modules are rated below the motor's rated current, the empty supercapacitor bank needs a pre-charge, and the current sensor needs a zero calibration. The dashboard also shows that the battery voltage channel must be connected before battery readings can be trusted. The next steps are to fix these, run the vehicle with a real load, and compare the hybrid with a battery-only configuration.

REFERENCES

- [1] S. S. Patil and S. B. Kadu, "An Efficient Regenerative Braking System Based on Battery/Supercapacitor for Electric, Hybrid and Plug-In Hybrid Electric Vehicles with BLDC Motor," *Int. J. Engineering and Technology*, 2015.
- [2] "Hybrid energy storage unit fed motoring and regenerative braking control of electric vehicle drivetrain," *Journal of Energy Storage*, 2024.
- [3] "Modeling and analysis of a regenerative braking system with a battery-supercapacitor energy storage," in *Proc. IEEE Conference*, 2017.
- [4] "Energy Recovery and Power Management in EVs Using Supercapacitors and Regenerative Braking," *Conference Paper*, 2025.

-
- [5] “Energy Management in Electric Vehicles with Regenerative Braking using Super Capacitors and Batteries,” *Int. J. Vehicle Structures & Systems*, vol. 18, no. 1, 2026, doi: 10.4273/ijvss.18.1.23.
- [6] “Energy Management and Power Distribution for Battery/Ultracapacitor Hybrid Energy Storage System in Electric Vehicles with Regenerative Braking Control,” *Mathematical and Computational Applications*, vol. 30, no. 4, 2026, doi: 10.3390/mca30040082.
- [7] J. M. Miller, *Ultracapacitor Applications*, IET, London.
- [8] H. Ibrahim, A. Ilinca, and J. Perron, “Energy storage systems—Characteristics and comparisons,” *Renewable and Sustainable Energy Reviews*.
- [9] R. C. Dorf and J. A. Svoboda, *Introduction to Electric Circuits*, Wiley.
- [10] Allegro MicroSystems, “ACS712: Fully Integrated, Hall Effect-Based Linear Current Sensor IC,” datasheet.
- [11] Microchip Technology, “ATmega328P 8-bit AVR Microcontroller,” datasheet.
- [12] InvenSense, “MPU-6000 and MPU-6050 Product Specification,” datasheet.
- [13] Texas Instruments, “LM2596 SIMPLE SWITCHER Power Converter 150-kHz 3-A Step-Down Voltage Regulator,” datasheet.
- [14] Songle Relay, “SRD Series 5 V DC Miniature Power Relay,” datasheet.