

Microgrid Battery Manager: A Labview-Based Virtual-Instrumentation System for Multi-Cell Li-Ion Monitoring, State-Of-Charge Estimation and Protection

Amishi Pandey¹ Kushagra Shukla² Ilanganiyan Rajkumar³ Sourya Pradhan⁴

School Of Electrical Engineering (Select) Vit University, Vellore, India

Abstract—Li-ion packs in microgrid energy-storage units need continuous supervision, because cell imbalance, overcurrent and overheating shorten pack life and create safety risks. This paper presents a Microgrid Battery Manager built as a LabVIEW virtual instrument on an Arduino platform. A four-cell 18650 series pack (14.8 V nominal, 2.6 Ah) is scanned through a CD4051 analog multiplexer that carries four cell-voltage taps, two NTC thermistors and one shunt-based current channel. The host VI estimates State of Charge (SoC) by Coulomb counting, corrected by open-circuit voltage (OCV) whenever the pack is at rest, and uses nested Case Structures to classify overcurrent, over-temperature and cell-imbalance faults. Solid-state switches perform passive balancing and automatic load disconnect, and a front-panel dashboard shows SoC, cell voltages, temperatures and protection status. The system is validated against four targets: ± 10 mV cell-voltage accuracy, SoC error within ± 5 %, cell spread below 20 mV after balancing, and an overcurrent trip in under 100 ms.

Keywords—Battery management system, Li-ion battery, LabVIEW, virtual instrumentation, state of charge, Coulomb counting, cell balancing, microgrid, data acquisition.

I. INTRODUCTION

Microgrid Combines Local Generation Such As Solar And Wind With Loads And An Energy-Storage Unit That Absorbs The Mismatch Between Them. Li-Ion Packs Are Widely Used For This Storage, And Their Interconnection And Evaluation Are Covered By Standards Such As Ieee Std 1547-2018 [1] And Ieee Std 1679.1-2017 [2]. A Li-Ion Pack Cannot Be Run Without Supervision. Cells In A Series String Drift Apart In Voltage Because Of Manufacturing Tolerance And Unequal Self-Discharge, So The Weakest Cell Sets The Safe Charge And Discharge Limit Of The Whole Pack. Overcurrent, Over-Temperature And Overcharge Are The Main Failure Modes, And Safety Requirements For Industrial Lithium Batteries Are Set Out In Iec 62619 [3]. A Battery Management System (Bms) Measures Cell Voltage, Temperature And Current, Estimates State Of Charge (Soc), Balances The Cells And Disconnects The Pack When A Limit Is Violated [4]. This Work Builds The Bms As A Virtual Instrument: A Labview Program Running On A Pc With An Arduino Board Replaces A

Dedicated Controller, So Thresholds And Logic Can Be Changed Without Rewiring. The Objectives Are To (1) Acquire Individual Cell Voltages Through A Multiplexer-Based Front End; (2) Acquire Cell And Pack Temperatures With Ntc Thermistors; (3) Estimate Soc By Coulomb Counting With Periodic Ocv Correction; (4) Classify Faults With Nested Case Structures; (5) Drive Solid-State Switches For Passive Balancing And Load Disconnect; (6) Present A Real-Time Front-Panel Dashboard; And (7) Validate The System Under Charge/Discharge Cycles And Fault Injection. The Scope Of The Project Covers The Complete Signal Chain Of A Battery Manager, From The Cells To The Operator Display. On The Hardware Side It Includes Assembly Of A Four-Cell 18650 Pack, Design Of The Multiplexed Sensing Front End, And A Mosfet Power Stage That Can Interrupt Charge And Discharge Current Independently. On The Software Side It Covers The Labview Acquisition Loop, Calibration, The Soc Estimator, The Fault-Classification Logic And The Dashboard. The Report Is Organised As Follows. Section I Gives The Background And The Theory Behind The Methods Used. Section Ii Describes The Hardware, The Labview Implementation,

The Step-By-Step Build Procedure And The Validation Plan. Section Iii Presents The Prototype And The Results, And Section Iv Concludes With The Future Scope.

Literature Review

Andrea [4] Describes The Monitoring, Balancing And Protection Functions Of Bmss For Large Lithium-Ion Packs, And Plett [5] Gives A Detailed Treatment Of Cell Modelling, Which Underlies Model-Based Soc Estimation. Among The Soc Methods Reviewed By Chang [6] Are Coulomb Counting, Which Is Simple But Accumulates Drift, And The Ocv Method, Which Is Accurate Only When The Pack Is At Rest. This Motivates The Hybrid Estimator Used Here. Daowd Et Al. [7] Compare Passive And Active Balancing In Matlab Simulation; Passive Balancing Is Chosen In This Work Because It Needs Only A Resistor And One Switch Per Cell. Integrated Monitor-And-Protect Devices Such As The Bq76920 [8] Implement These Functions In Fixed Hardware. The Virtual-Instrumentation Approach Used Here Keeps The Same Functions In Software On A General-Purpose Arduino And Pc Platform [9], So Thresholds, Scan Rate And Logic Are Easy To Change During Development. This Project Combines Multiplexed Multi-Cell Acquisition, Hybrid Soc Estimation, Case-Structure Fault Classification And Solid-State Protection In One Low-Cost, Reconfigurable Labview System.

Theoretical Background

V nominal, and a 4S2P pack doubles the capacity by paralleling two such strings. The main pack quantities a BMS tracks are State of Charge (SoC), State of Health (SoH), Depth of Discharge (DoD) and internal resistance. In a series string the cells drift apart in voltage over repeated cycles because of manufacturing tolerance and unequal self-discharge. This cell imbalance reduces the usable capacity, because the weakest cell reaches its voltage limit first and stops the whole string, and it risks overcharging or over-discharging that cell. In a microgrid the pack buffers the mismatch between intermittent solar or wind generation and the load, so safe and efficient cycling is essential.

1. Multiplexed sensing and thermistors: An analog multiplexer connects each cell tap in turn to one ADC channel, so N cell voltages can be scanned with a single

Arduino analog input instead of N inputs. This lowers the input count and wiring complexity, at the expense of scan time across the channels. The multiplexer address lines are driven as digital outputs from LabVIEW inside a timed acquisition loop, and the scan rate is a trade-off against the settling time of each channel. An NTC thermistor falls in resistance as temperature rises, and the resistance is converted to temperature with the Steinhart–Hart equation (1). Placing thermistors on the cell surface and on the switch heatsink gives a spatial thermal profile rather than a single average reading. Pack current is measured with a shunt resistor or a Hall-effect sensor and is the essential input to Coulomb counting.

2. SoC estimation: Coulomb counting integrates the measured current over time. It is simple and works in real time under load, but sensor offset and integration error make it drift over long operation. The OCV method maps a rested terminal voltage to SoC using a pre-characterised OCV–SoC curve. It does not drift, but it is valid only when the pack carries no load. The hybrid estimator used here combines the two: Coulomb counting provides continuous tracking, and an OCV reading corrects the running value whenever the pack is detected at rest.

3. Virtual instrumentation and Case Structures:

Virtual instrumentation replaces a dedicated controller with software-defined logic on a general-purpose Arduino and PC platform, so the behaviour can be changed without rewiring. The LabVIEW Case Structure is the graphical equivalent of an if-else or switch-case construct, and its selector terminal is driven by Boolean or enumerated fault flags computed on each scan. Four cases are used: Normal, Overcurrent, Over-temperature and Cell imbalance. Each case wires a distinct set of outputs to the driver digital lines, so exactly one control action runs per scan cycle. The structure sits inside a timed loop synchronised to the Arduino acquisition rate, which gives a deterministic response.

1. Protection and thermal management: In overcurrent protection the measured current is compared with a threshold, and on violation a solid-state switch (MOSFET) opens the load path within milliseconds, which is faster and more durable than an electromechanical fuse or relay. Passive balancing

bleeds charge from higher-voltage cells through a resistor switched in by a solid-state device, which equalises the string over successive cycles. For thermal management, temperature thresholds trigger dedicated branches that derate or shed load, and sustained over-temperature escalates to a full pack disconnect. Together these three loops define a safe operating envelope against the principal Li-ion failure modes of overcurrent, over-temperature and cell overcharge or imbalance.

II. METHODOLOGY

1. System Overview

Fig. 1 shows the system. Cell voltages, temperatures and pack current pass through the sensing front end and signal conditioning to the Arduino board (forward path). The LabVIEW host VI estimates SoC, decides on a protection or balancing action, and sends it through a switch driver to the solid-state switches (control path). The development followed four stages. In requirement analysis, the aim and objectives were translated into concrete acquisition channels, safety thresholds and dashboard requirements before any hardware was built. In hardware front-end design, the multiplexer, resistive dividers, thermistors and current shunt were selected and wired, and each channel was bench-calibrated against reference instruments. In LabVIEW VI development, the acquisition, SoC-estimation and case-structure fault logic were built as modular sub-VIs inside a timed loop.

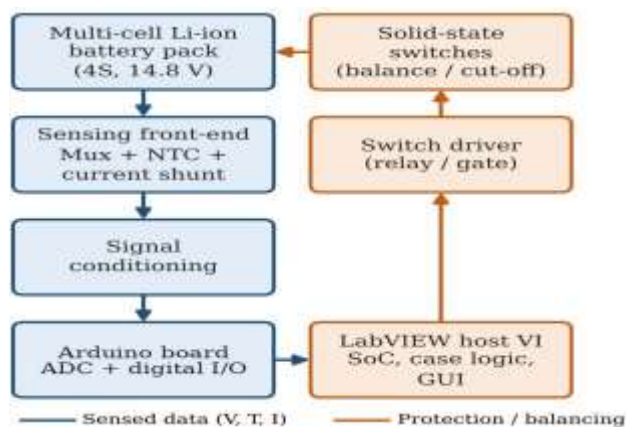


Fig. 1. Block diagram of the Microgrid Battery Manager. Blue: sensed data path. Orange: protection and balancing path.

acquisition channels, safety thresholds and dashboard requirements before any hardware was built. In hardware front-end design, the multiplexer, resistive dividers, thermistors and current shunt were selected and wired, and each channel was bench-calibrated against reference instruments. In LabVIEW VI development, the acquisition, SoC-estimation and case-structure fault logic were built as modular sub-VIs inside a timed loop. In integration and calibration, the Arduino board was connected to the sensing front end, per-channel offsets were tuned, and the dashboard was checked against live pack readings before testing began.

2. Hardware Front End

The pack has four 18650 Li-ion cells in series (14.8 V nominal, 2.6 Ah) with nickel-strip tabs and a fused main lead. A CD4051 8:1 analog multiplexer [10] with three digital address lines connects seven signals, one at a time, to a single Arduino analog input (Table I). Each cell tap passes through a precision resistive divider and a unity-gain buffer so that the ADC sees a low source impedance. Two 10 k Ω NTC thermistors ($B = 3950$ K) [11] are bonded to the cell body and to the switch heatsink. Pack current is measured by a low-side shunt and a differential amplifier scaled for ± 15 A. Back-to-back N-channel MOSFETs with an isolated gate driver give independent charge and discharge cut-off.

The multiplexer is simply a switch that lets one measuring channel read all seven signals one at a time. The three address lines select channels 0 to 6 in sequence, and the common output is fed to a single Arduino analog input. Because the analog switch has a finite on-resistance and the ADC input has a sampling capacitor, a short settling delay is inserted after each address change before the sample is taken. The delay is the main limit on the scan rate, and it is kept short enough that a full seven-channel scan is well inside the 100 ms protection budget.

A series string places the upper cell taps at high common-mode voltage, with the top tap at about 14.8 V, which is above the 5 V input range of the Arduino analog input. For this reason every cell tap passes through its own precision resistive divider, which scales the tap into the ADC range, and then through a unity-gain buffer. The buffer isolates the divider from the

multiplexer and the ADC sampling capacitor, so the source impedance seen by the ADC is low and the reading is not disturbed by the switching. The per-cell voltage is recovered in software by subtracting adjacent tap readings and applying the divider ratio.

The two thermistors are bonded to the cell body and to the switch heatsink. The first tracks the temperature of the energy store itself, and the second tracks the power stage, which is the part most likely to heat during a sustained overload. Each thermistor forms a divider with a fixed resistor so that the multiplexer sees a voltage that falls as temperature rises. The low-side shunt in the negative lead produces a small voltage proportional to pack current, and the differential amplifier scales it to the ADC range for a full-scale range of ± 15 A, with the sign showing the direction of current (charge or discharge). The power stage uses back-to-back N-channel MOSFETs driven by an isolated gate driver. One device blocks charging current and the other blocks discharging current, so the VI can cut one direction while leaving the other available. For example, an over-voltage cell can stop charging while still allowing the load to discharge the pack. The isolated driver keeps the Arduino digital outputs electrically separate from the pack. Balancing uses one resistor and one solid-state switch per cell, so any single cell can be bled independently.

Table I
Signals Scanned Through The Multiplexer

Signal conditioning	Qty	Sensing	and
Cell voltage	4	Resistive divider	+ unity-gain buffer
Temperature	2	10 k Ω NTC (B = 3950 K): cell body,	
switch			
heatsink			
Pack current	1	Low-side shunt	+ differential amplifier, ± 15 A

2. Signal Conversion and Calibration

Each channel is compared with a multimeter and a small correction is stored, so the VI reports true voltage and temperature instead of raw counts. Thermistor resistance is converted to temperature with the Steinhart–Hart equation:

In practice the stored corrections are one offset and one gain term per channel, found by comparing the VI reading with the multimeter at two or more points. They are saved in a calibration array that the acquisition sub-VI applies before any other processing. This keeps the later stages simple, since every downstream value (cell voltage, temperature, current) is already in engineering units.

$$1/T = A + B \ln(R) + C (\ln R)^3 \quad (1)$$

where T is the absolute temperature (K), R is the thermistor resistance (Ω), and A , B and C are device-specific constants.

3. State-of-Charge Estimation

SoC is tracked by Coulomb counting:

$$\text{SoC}(t) = \text{SoC}_0 - (1/\text{Crated}) \int I \, dt \quad (2)$$

where $\text{Crated} = 2.6$ Ah and I is the discharge current (positive). Integration error and sensor offset make the estimate drift. When the measured current stays near zero, the pack is at rest, so its OCV is read and mapped through a pre-characterised OCV–SoC curve to reset the running total. The estimator is a sub-VI that integrates current and voltage arrays in a timed loop.

In the VI the estimator runs as a sub-VI called once per scan. It receives the calibrated current and the cell-voltage array, and keeps the running SoC in a shift register of the timed loop so that the value carries from one scan to the next. The integration step is the loop period, and a rest condition is declared when the measured current stays below a small threshold for a set time. When this holds, the pack OCV is read, looked up in the stored OCV–SoC table with interpolation, and written back into the shift register. The estimate is

clamped between 0 and 100 %, and the same value is passed to the dashboard gauge.

5. Fault Logic and Protection

Nested Case Structures inside a timed loop evaluate the fault flags on every scan in priority order: overcurrent, over-temperature, cell imbalance, then normal operation (Fig. 2). Exactly one action runs per scan, so commands never conflict. Table II lists the conditions and actions. A tripped state is latched, the relevant switch is opened and an alarm is raised. The priority order reflects the severity and speed of each hazard. Overcurrent can damage the cells and the switches within milliseconds and is therefore tested first. Over-temperature develops over seconds to minutes and is tested next. Imbalance is a slow condition handled by balancing and is lowest. Because an overcurrent trip is latched, it stays active until it is cleared deliberately and does not reset by itself when the current drops to zero once the load is disconnected. The over-temperature case first derates the load and escalates to a full disconnect only if the temperature stays above the 55 °C threshold, which avoids unnecessary trips from short thermal excursions. The cell-voltage limit case cuts charging when a cell is too high and cuts discharging when it is too low.

Signals Scanned Through The Multiplexer

Fault	Condition	Action
Overcurrent	$I > I_{max}$ (test step > 10 A)	Open charge and discharge MOSFETs; latch trip
Over-temperature	$T > 55\text{ }^{\circ}\text{C}$	Derate, then trip if sustained
Cell imbalance	Cell spread above limit	Passive balancing on the highest cell
Cell voltage limit	Cell voltage out of limits	Cut charging (high) or discharging (low)

The VI is modular: acquisition, SoC estimation, fault logic and the dashboard are separate sub-VIs inside one timed loop synchronised with the Arduino sampling and data-transfer rate, which gives deterministic fault response. An Arduino board reads the multiplexed signal through its on-board ADC and exchanges data with the LabVIEW host over a USB serial link. Its digital output pins drive the multiplexer address pins and the switch driver. The front panel shows SoC, the four cell voltages, the two temperatures, pack current and protection status. The block diagram is built around a single timed loop. Inside the loop, the digital-output lines set the multiplexer address, a short wait allows settling, and one analog sample is read, so that

each iteration of an inner sequence acquires one channel. After seven channels the raw values are assembled into an array and passed through the calibration, divider-ratio and Steinhart–Hart conversions. The converted arrays then go to the SoC sub-VI and to the fault logic. All the sub-VIs share the same loop timing, so a fault flag raised in one scan is acted on in the same scan. The fault logic uses nested Case Structures with Boolean selector terminals. The outer structure checks for overcurrent, and its false case contains a second structure that checks over-temperature, followed in turn by the imbalance test. Each case wires a fixed set of digital-output values to the switch driver: the charge MOSFET, the discharge MOSFET, the per-cell balancing switches and the alarm line. Because every case assigns every output, the outputs are always in a defined state and no two cases can issue conflicting commands.

The front-panel dashboard is the operator interface. It shows the SoC as a gauge, the four cell voltages as indicators with the cell spread, the two temperatures, the pack current, and Boolean LEDs for each protection state. A latched alarm indicator and a reset button allow the operator to acknowledge a trip. A waveform chart shows pack current and cell voltages over time. Data are also logged to a file with a time stamp on every scan, so each test can be reviewed afterwards and checked against its acceptance criterion. The threshold values, scan period and calibration constants are front-panel controls, so they can be changed during testing without editing the block diagram.

Validation Procedure

Five tests are carried out (Table III). (1) Channel accuracy: all seven multiplexed channels are compared with a 4½-digit bench meter over the full range. (2) SoC: a fully charged pack is discharged at constant current and the estimate is compared with the integrated reference. (3) Imbalance: one cell is deliberately discharged by 60 mV and the balancing scheduler. Each test has a numerical acceptance criterion, summarised in Table III. The channel-accuracy test aims at $\pm 10\text{ mV}$ on cell voltage, the SoC test at an error within $\pm 5\%$, and the imbalance test at a cell spread below 20 mV after balancing. The overcurrent test requires the trip within 100 ms of the load step, and the thermal test requires a clean derate followed by a

trip when the 55 °C threshold is crossed and held. Every result is logged against its criterion, and thresholds are refined where necessary before the final report.

TABLE III

TABLE III
VALIDATION TESTS AND ACCEPTANCE CRITERIA

Test	Method	Target
Channel accuracy	All 7 channels vs. 4½-digit bench meter	±10 mV (cell voltage)
SoC validation	Constant-current discharge vs. integrated reference	Error within ±5 %
Imbalance	One cell discharged by 60 mV; observe balancing	Spread below 20 mV
Overcurrent trip	Step load above 10 A; capture gate signal	Trip under 100 ms
Thermal	Hot-air warming through 55 °C and back	Clean derate, then trip

Step-by-Step Build Procedure

(1) Build the pack from four Li-ion cells in series and run a thin wire from each cell junction, from the thermistors and from the current sensor to the multiplexer. (2) Feed the multiplexer output into the Arduino analog input so that the VI can read the values. (3) Compare each reading with a multimeter and store a correction for each channel, so that the software reports true voltage and temperature instead of raw counts. (4) Keep integrating the current into and out of the pack to track charge, and when the pack is idle use the resting voltage to reset the running total. (5) Write the rules that compare every reading against its safe limit: too high a voltage, too low a voltage, too much current and too high a temperature. Each condition points to one action: cut charging, cut discharging, or trip both. Add the balancing rule that bleeds the highest cell when it runs ahead of the others. (6) Wire the MOSFET switches into the power path of the pack so that the controller can actually cut the current, then latch the trip, open the relevant switch and raise an alarm on the panel. (7) Deliberately cause each fault, by overcharging one cell, overloading the pack and heating it, confirm that it trips correctly, and log and report the results.

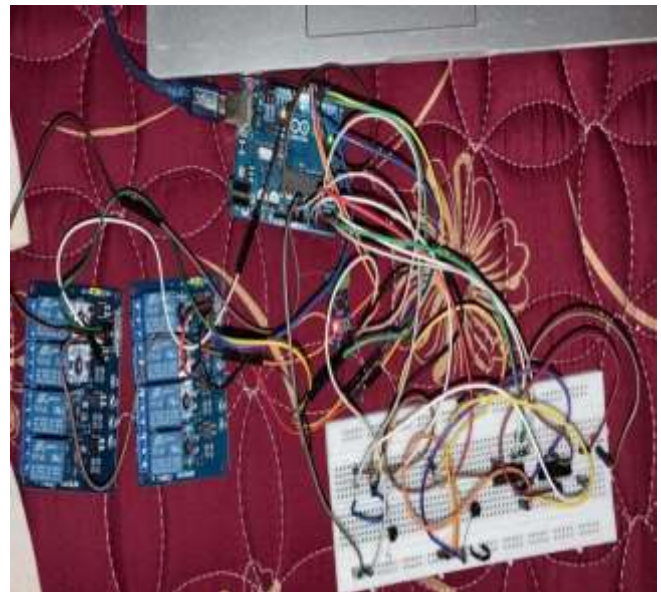
I. Project Status and Schedule

Component selection and pack assembly are complete, and the step-by-step procedure has been mapped. Work in progress includes tuning the balancing thresholds on a live pack, the thermal derating profile and heatsink sizing, the dashboard layout and logging format, full hardware–software integration on the pack, fault-injection and endurance cycling, and the final report. The project runs over twelve weeks with four milestones: a calibrated acquisition demonstrated in Week 5, SoC validated against the reference in Week 8, protection trips verified in Week 11, and the final review in Week 12.

III. RESULTS AND DISCUSSION

All signals are acquired live from the prototype pack, so no separate dataset is used. Each step is shown below as input, intermediate signal and output.

1. Hardware Prototype



The prototype places the four-cell pack, the sensing front end with the CD4051 multiplexer, the Arduino board and the MOSFET switch board on one bench setup. The cell taps, thermistor leads and shunt leads are wired to the multiplexer board, and the Arduino digital outputs go to the address pins and the isolated gate driver. The thermistors are fixed to a cell and to the switch heatsink, so the thermal readings correspond to the points used in the fault logic.

2. LabVIEW Front Panel and Block Diagram



Fig. 4. LabVIEW front-panel dashboard during operation.

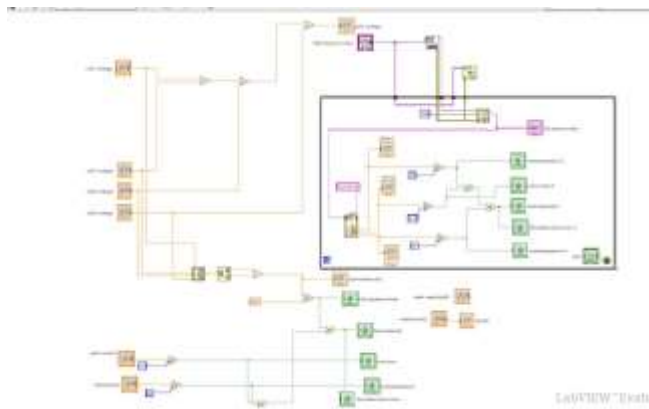


Fig. 5. LabVIEW block diagram showing the acquisition loop and case-structure fault logic.

The front panel in Fig. 4 gives the operator a single view of the pack state, with the SoC gauge, the four cell voltages, the temperatures, the pack current and the protection LEDs. The block diagram in Fig. 5 shows the timed acquisition loop with the multiplexer-scan sequence and the nested Case Structures that select one control action per scan. Because the thresholds are front-panel controls, the same VI was used for all of the validation tests without modification.

3. Acquisition: Input and Intermediate Signals

The input is the raw multiplexed ADC stream. After the stored per-channel correction and the divider and Steinhart–Hart conversions, the intermediate signals are the four cell voltages, two temperatures and the pack current. Table III reports the error of each channel against the bench meter.

4. State-of-Charge Estimation

The input is pack current and, at rest, terminal voltage. The Coulomb-counting output drifts slowly; each OCV reset pulls it back to the reference. The figure compares the pure Coulomb count, the OCV-corrected estimate and the integrated reference during a constant-current discharge.

5. Cell Balancing

One cell is discharged by 60 mV and the balancing scheduler bleeds the highest cell until the spread falls below 20 mV.

During the imbalance test the VI continuously computes the spread between the highest and lowest cell voltages. When the spread is above the limit, the balancing switch of the highest cell is closed and its resistor bleeds charge, and the switch is opened again once the spread falls below the 20 mV target. The cell voltages and the switch state are logged, so the reduction of the spread over time can be read directly from the recorded data.

6. Overcurrent and Thermal Protection

For a step load beyond 10 A, the output is the gate signal of the MOSFET switches, captured against the load step to measure trip time. In the thermal test, temperature rises through 55 °C, the VI derates and then trips.

IV. CONCLUSION AND FUTURE SCOPE

A LabVIEW-based Microgrid Battery Manager has been designed for a four-cell Li-ion pack. One multiplexed Arduino analog channel acquires four cell voltages, two temperatures and the pack current. A hybrid Coulomb-counting and OCV estimator tracks SoC, nested Case Structures classify overcurrent, over-temperature and imbalance faults, and solid-state switches carry out balancing and load disconnect.

Future work includes: replacing Coulomb counting with a model-based estimator such as a Kalman filter [5]; adding State-of-Health estimation from internal-resistance trends; trying active balancing to reduce energy lost in resistors [7]; moving the logic to a real-time target so the unit runs without a PC; extending the

design to larger series-parallel packs such as 4S2P; and logging data to the microgrid energy-management system.

REFERENCES

1. IEEE Standard for Interconnection and Interoperability of Distributed Energy Resources with Associated Electric Power Systems Interfaces, IEEE Std 1547-2018, 2018.
2. IEEE Guide for the Characterization and Evaluation of Lithium-Based Batteries in Stationary Applications, IEEE Std 1679.1-2017, 2017.
3. Secondary cells and batteries containing alkaline or other non-acid electrolytes — Safety requirements for secondary lithium cells and batteries, for use in industrial applications, IEC 62619:2022, 2022.
4. D. Andrea, Battery Management Systems for Large Lithium-Ion Battery Packs. Norwood, MA, USA: Artech House, 2010.
5. G. L. Plett, Battery Management Systems, Volume I: Battery Modeling. Norwood, MA, USA: Artech House, 2015.
6. W.-Y. Chang, "The state of charge estimating methods for battery: A review," *ISRN Appl. Math.*, vol. 2013, Art. no. 953792, 2013.
7. M. Daowd et al., "Passive and active battery balancing comparison based on MATLAB simulation," in *Proc. IEEE Vehicle Power and Propulsion Conf. (VPPC)*, 2011.
8. Texas Instruments, "bq76920 Battery Monitor and Protector Datasheet," 2019.
9. National Instruments, *LabVIEW Data Acquisition and Signal Conditioning Reference Manual*, 2021.
10. Texas Instruments, "CD4051B CMOS Analog Multiplexer Datasheet," 2017.
11. Vishay Intertechnology, "NTCLE100E3 NTC Thermistor Datasheet," 2020. APPROVAL Microgrid Battery Manager (BMS) — Virtual Instrumentation Project
12. Approval sheet for the project report.