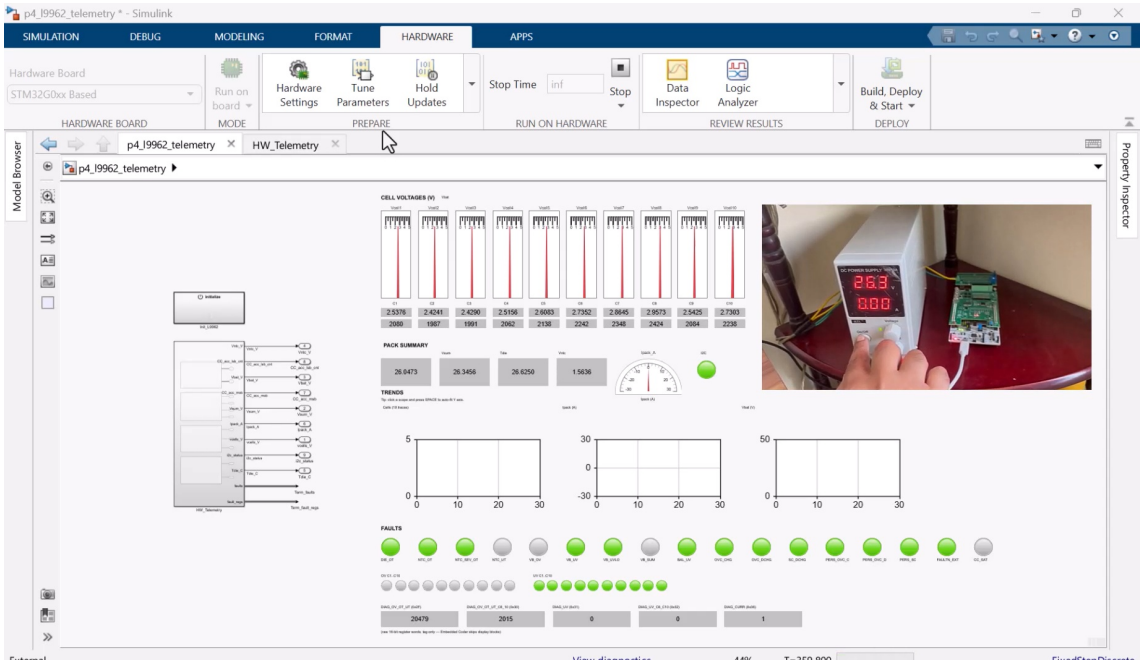
Battery Monitoring and State-of-Charge Estimation on STM32 with the STMicroelectronics L9962

Introduction

Mathworks and STMicroelectronics | Battery Management Systems

A model-based design workflow from a simulated battery pack to state-of-charge estimation running on an STM32G071, reading live cells over I²C.

Figure 1. External mode running on the NUCLEO-G071RB. Simulink model streams live cell voltages, pack summary values, and decoded L9962 fault flags during hardware operation.



This document describes a complete, reproducible development path: We modeled a battery pack in Simulink®. We developed and verified a state-of-charge estimator in simulation. We generated C code and profiled it on the target processor. Finally, the deployed model reads live measurements from an STMicroelectronics L9962 battery monitoring device over I²C. Each stage is verified before the next one begins.

Table 1. Solution at a glance

Environment	MATLAB® and Simulink
Physical modeling	Simscape™ Battery™
Code generation	Simulink Coder™, Embedded Coder®
Hardware support	STM32 Microcontroller Blockset
Target processor	STM32G071RB on the STM32 NUCLEO-G071RB board
Monitoring device	STMicroelectronics L9962 on the EVL9962 evaluation board
Device interface	PC
Battery pack	6 cells in series, 1 parallel (6S1P), Panasonic NCR18650BD
Release	MATLAB® R2026a

1 Model-based design with Simulink

The model-design with Simulink replaces the hand-off points with a continuous, verified chain. Simulate the pack and develop the estimator against it. Generate code from the same model. Then confirm that the code behaves identically on the processor and fits the available time budget. Nothing is rewritten between stages, so nothing has to be reverified from scratch.

The STM32 Microcontroller Blockset extends that chain onto ST hardware. It provides peripheral driver blocks, automatic code generation, and on-target verification, so the algorithm moves from model to running firmware without hand-written code.

2 System overview

The reference configuration splits the system across two devices. The L9962 measures; the STM32G071RB decides.

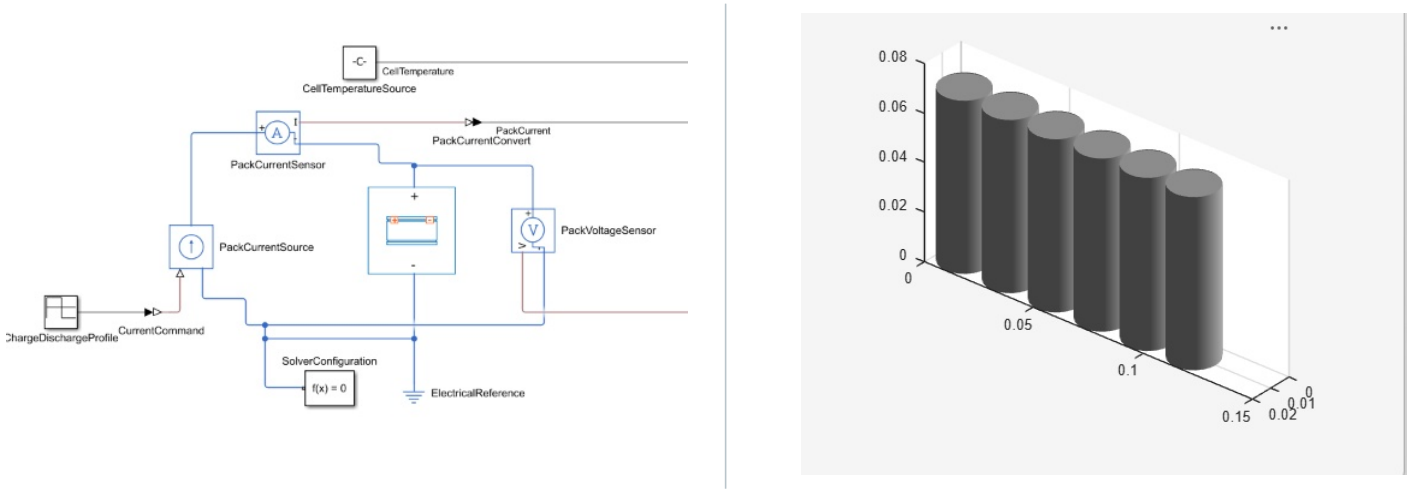
Element	Role in the system
Battery pack (6S1P)	Six Panasonic NCR18650BD cells in series. Represented in simulation by Simscape battery; emulated on the bench by a cell emulator or bench supply.
L9962 monitoring device	Measures individual cell voltages, stack voltage, and temperature via a high-precision ADC. A high-accuracy current-sense amplifier monitors stack current and performs Coulomb counting. Supports 4- to 10-cell series configurations and passive balancing on all cells.
EVL9962 evaluation board	A ready carrier for the L9962, connecting to the Nucleo board through the morpho headers.
STM32G071RB	Runs the deployed application, including I²C transactions, register decoding, scaling, fault decoding, and the state-of-charge estimator.
NUCLEO-G071RB board	Hosts the STM32G071RB, provides ST-LINK programming and power, and carries the serial link used for external mode.
Host workstation	It runs MATLAB® and Simulink for modeling, simulation, code generation, and live monitoring.

3 Stage 1 - Model the battery pack

Everything downstream depends on the cell, so the plant model is built first. Rather than deriving a cell from scratch, the example starts from the parameterized Panasonic NCR18650BD in the Simscape battery cell library. The starting point therefore traces to a real part.

The plant contains the pack, a current source that drives a charge or discharge profile, voltage sensing, and current sensing. Logged signals are routed to a single scope through tags, which keeps the model readable as it grows.

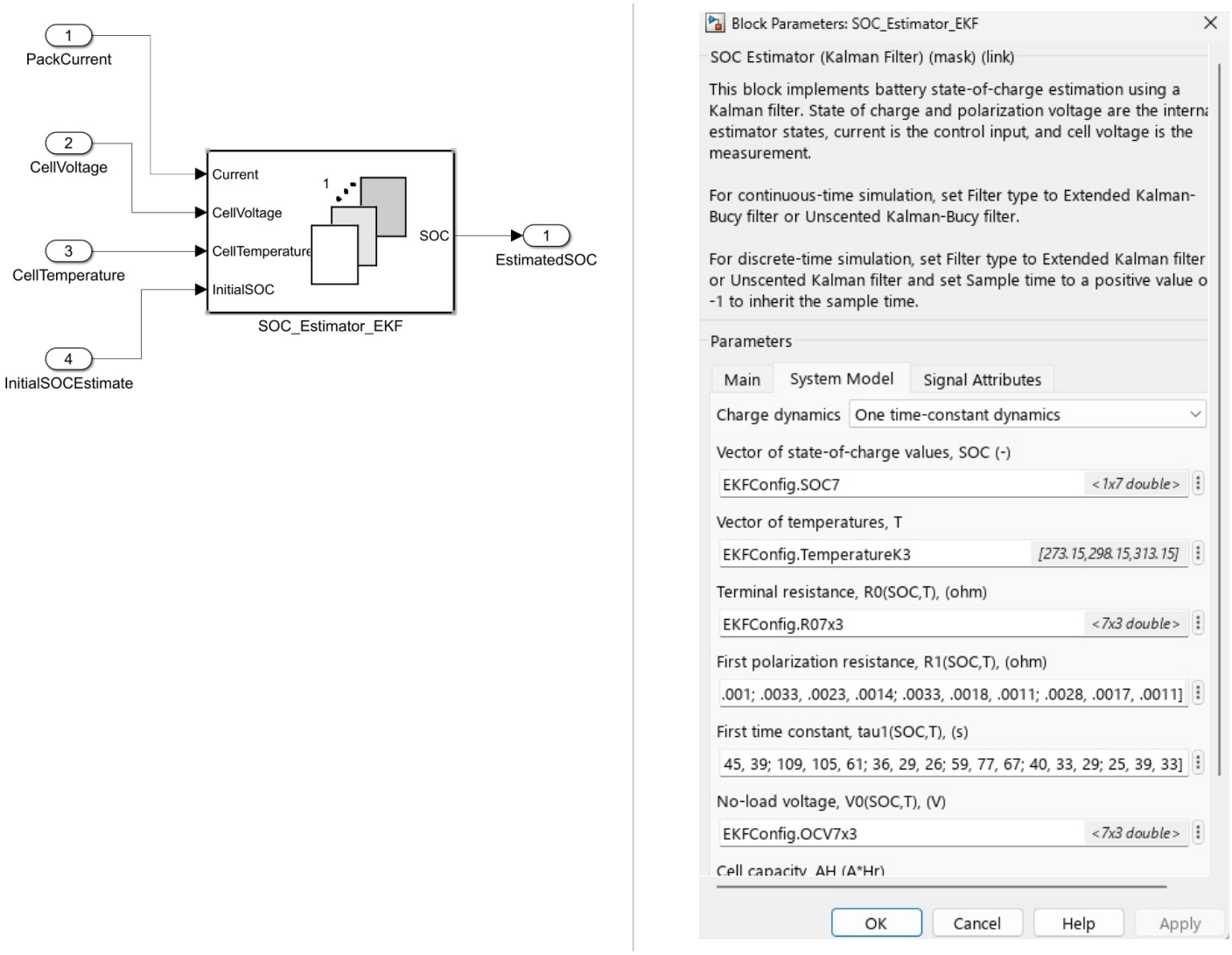
Figure 2. Left: the Simscape battery plant: pack, controlled current source, and voltage and current sensors. Right: the 6S1P pack built from the parameterized NCR18650BD cells



4 Stage 2 — Estimate state of charge

The estimator is an extended Kalman filter, implemented with the SoC estimator (Kalman filter) block that ships in Simscape battery. It works in two stages: it uses the cell model from stage 1 to predict what the terminal voltage should be, then corrects its estimate against what was actually measured.

Figure 3. Left: The estimator takes pack current, cell voltage, cell temperature, and an initial estimate. It returns the estimated state of charge. Right: the block parameters, where the cell model from stage 1 is supplied as lookup data over SoC and temperature.



The estimator sits in its own model, referenced by the integration model that contains the plant. In simulation, the plant drives it. In stage 4, that same model is built for the processor—unchanged. One algorithm is exercised in simulation and then deployed.

5 Stage 3 — Verify in simulation

Various verification test cases can be simulated, such as starting the estimator well below the true state of charge and verifying whether it recovers.

Figure 4. Left: estimated state of charge against true state of charge, the estimate begins low and pulls in. Right: estimation error settling toward zero over the cycle.

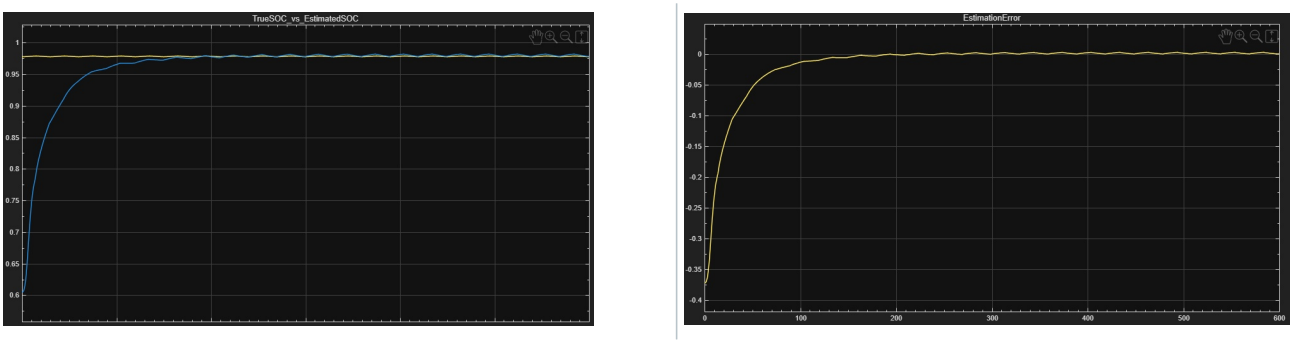


Table 2. Results from a 600-second validation run

Initial estimate	Time to reach within 2% of true SoC	Mean absolute error after convergence
Equal to true SoC	Within 2% immediately	Approximately 0.2%
60% (below true SoC)	86 s	Comparable to the case above

6 Stage 4 — Generate code and deploy

- Software-in-the-loop (SIL) asks whether the generated C code produces the same results as the model. Embedded Coder generates the code and it runs on the host against the same stimulus.
- Processor-in-the-loop (PIL) asks whether it still behaves the same way on the actual device, and the execution metrics. The same code runs on the STM32G071RB, still driven by the simulation.

Figure 5. The PIL harness. The plant runs in simulation while generated code runs on the STM32G071RB. We compare signals at the boundary.

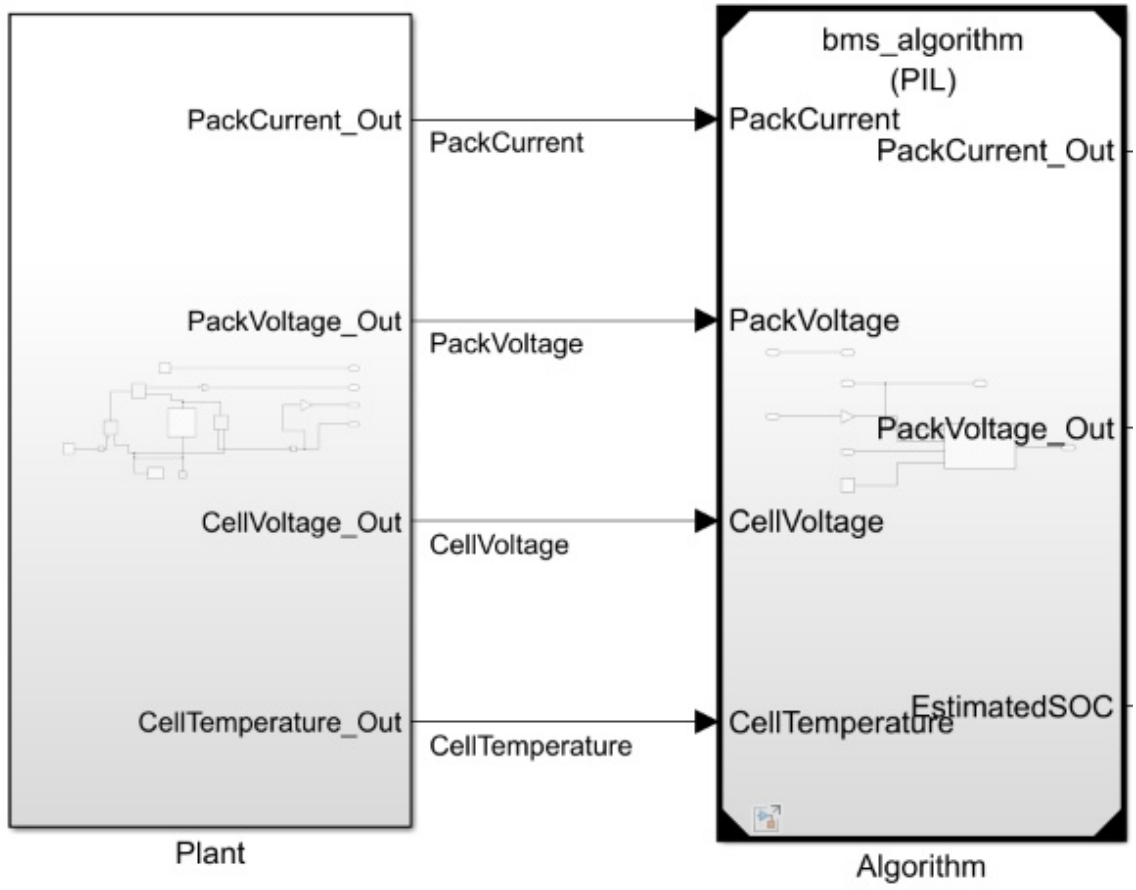
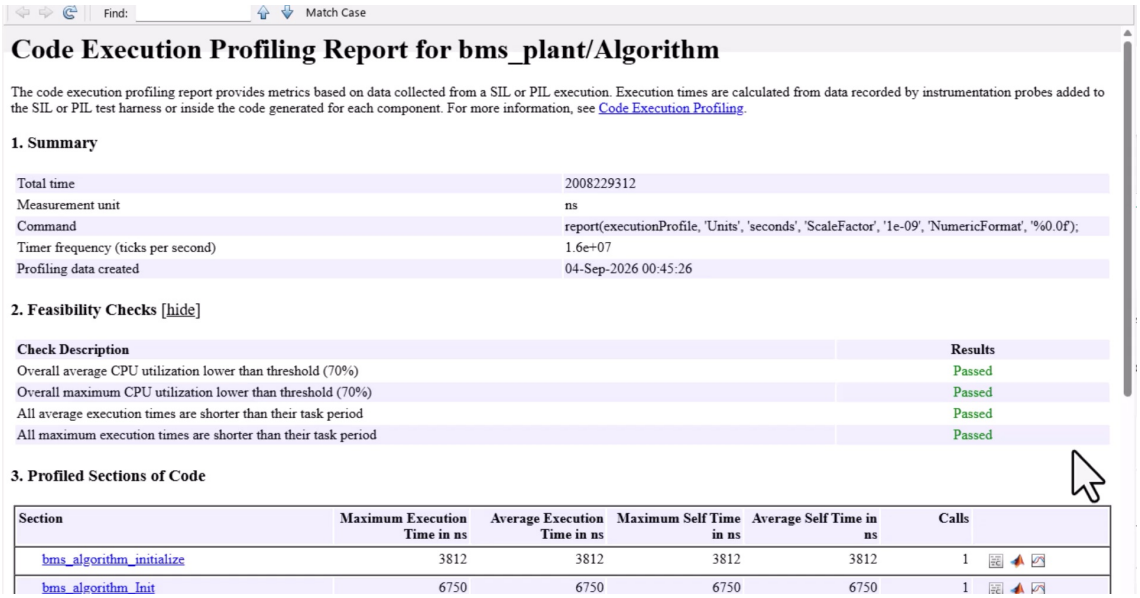


Table 3. Measured execution time on the STM32G071RB

Profiled section	Average execution time	Calls
bms_algorithm_initialize	3.8 µs	1
bms_algorithm_Init	6.8 µs	1
bms_algorithm step function	3.34 ms	601

Figure 6. The code execution profiling report retrieved from the board, showing the feasibility checks and per-section timing.



Find: Match Case

Code Execution Profiling Report for bms_plant/Algorithm

The code execution profiling report provides metrics based on data collected from a SIL or PIL execution. Execution times are calculated from data recorded by instrumentation probes added to the SIL or PIL test harness or inside the code generated for each component. For more information, see Code Execution Profiling.

1. Summary

Total time	2008229312
Measurement unit	ns
Command	report(executionProfile, 'Units', 'seconds', 'ScaleFactor', '1e-09', 'NumericFormat', '%0.0f');
Timer frequency (ticks per second)	1.6e+07
Profiling data created	04-Sep-2026 00:45:26

2. Feasibility Checks [hide]

Check Description	Results
Overall average CPU utilization lower than threshold (70%)	Passed
Overall maximum CPU utilization lower than threshold (70%)	Passed
All average execution times are shorter than their task period	Passed
All maximum execution times are shorter than their task period	Passed

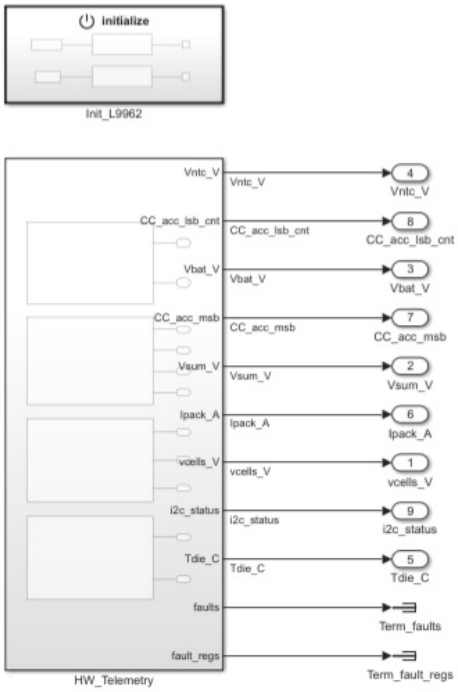
3. Profiled Sections of Code

Section	Maximum Execution Time in ns	Average Execution Time in ns	Maximum Self Time in ns	Average Self Time in ns	Calls	
bms_algorithm_initialize	3812	3812	3812	3812	1	
bms_algorithm_Init	6750	6750	6750	6750	1	

7 Stage 5 — Read the L9962 and run on live cells

The model has to read the monitoring device. That means configuring I²C for the L9962, the STM32 Microcontroller Blockset provides the I²C blocks. External mode is a workflow that enables to close the last gap: the model runs on the STM32G071RB while Simulink displays the signals live, so values are confirmed during development rather than after deployment.

Figure 7. Left: the telemetry model. An initialization subsystem configures the L9962. HW_Telemetry reads data cyclically and sends decoded signals to root outports. Right: the dashboard during an External mode run. It shows live cell voltages, pack summary values, fault lamps, and the raw 16-bit diagnostic register words displayed alongside the decoded flags.



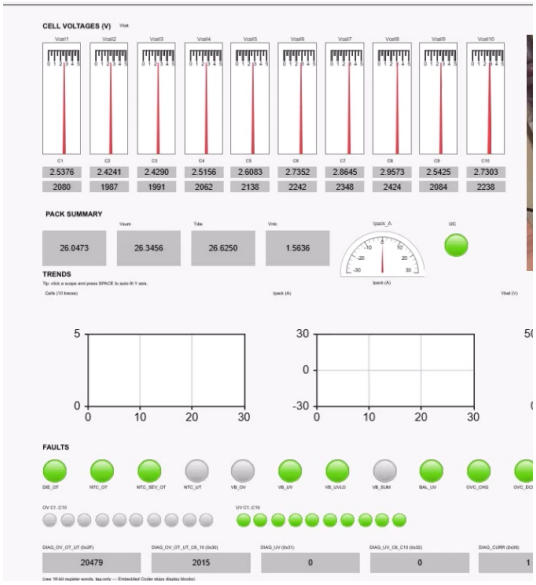


Table 4. Signals read from the device

Signal	Source on the L9962	Units
vcells_V	Individual cell voltage registers (up to 10 channels)	V
Vbat_V	Battery stack monitor	V
Vsum_V	Sum of enabled cell voltages, used for the plausibility check against stack voltage	V
Ipack_A	Current-sense amplifier	A
Tdie_C	Internal die temperature	°C
Vntc_V	External NTC temperature channel	V
CC_acc_lsb_cnt, CC_acc_msb	Coulomb counter accumulator	counts

Signal	Source on the L9962	Units
faults, fault_regs	DIAG_OV_OT_UT (0x2F), DIAG_OV_OT_UT_C6_10 (0x30), DIAG_UV (0x31), DIAG_UV_C6_C10 (0x32), DIAG_CURR (0x36)	Bitfields and raw 16-bit words
i2c_status	Transaction status returned by the I²C read blocks	0 = success

8 Getting started

The example is published on MATLAB® Central File Exchange.

Required products

- MATLAB® R2026a
- Simulink
- Simscape battery (for the plant model and the SoC estimator block)
- Simulink coder and Embedded coder
- STM32 Microcontroller Blockset

Required hardware and software

- STMicroelectronics NUCLEO-G071RB development board
- STMicroelectronics EVL9962 evaluation board
- USB cable for ST-LINK programming, power, and serial External mode
- An EVL9962-compatible cell emulator or a safe bench supply for telemetry testing
- An STM32CubeMX installation compatible with the STM32 Microcontroller Blockset release

Table 5. Resource links

Download the example	mathworks.com/matlabcentral/fileexchange/184339-stm32-blockset-support-for-stmicroelectronics-l996x-bms
L9962 product page	st.com/en/power-management/l9962.html
NUCLEO-G071RB	st.com/en/evaluation-tools/nucleo-g071rb.html
STM32 Microcontroller Blockset	mathworks.com/products/stm32-microcontroller.html
Simscape Battery	mathworks.com/products/simscape-battery.html
Embedded coder	mathworks.com/products/embedded-coder.html

9 Extending the example

The workflow generalizes in several directions without changing its shape.

- Other devices in the L996x family. The same structure. Initialization model, cyclic read subsystem, decode, and scale, dashboard — applies to other ST battery monitoring devices. The register map changes; the workflow does not.
- Larger packs. The L9962 covers 4 to 10-cell series configurations. Extending beyond the 6S1P pack used here is a matter of enabling additional cell channels and updating the pack definition in Simscape Battery.
- Passive cell balancing. The L9962 provides passive balancing by discharging cells through the Cx pins. Adding balancing control to the deployed algorithm is a natural next increment.
- Hardware-in-the-loop testing. Running the battery plant on a real-time target with the STM32 enables repeatable fault injection before connecting live cells.
- State of health and remaining useful life. The Coulomb counter data already read from the device is the input to capacity-fade estimation.

Revision history

Table 6. **Document revision history**

Date	Version	Changes
30-Sep-2026	1	Initial release.

Contents

1 Model-based design with Simulink 2
2 System overview 3
3 Stage 1 - Model the battery pack 4
4 Stage 2 — Estimate state of charge 5
5 Stage 3 — Verify in simulation 6
6 Stage 4 — Generate code and deploy 7
7 Stage 5 — Read the L9962 and run on live cells 9
8 Getting started 11
9 Extending the example 12
Revision history 13
List of tables 15
List of figures 16

List of tables

Table 1. Solution at a glance . . . 1
Table 2. Results from a 600-second validation run . . . 6
Table 3. Measured execution time on the STM32G071RB . . . 7
Table 4. Signals read from the device . . . 9
Table 5. Resource links . . . 11
Table 6. Document revision history . . . 13

List of figures

Figure 1. External mode running on the NUCLEO-G071RB. Simulink model streams live cell voltages, pack summary values, and decoded L9962 fault flags during hardware operation. 1
Figure 2. Left: the Simscape battery plant: pack, controlled current source, and voltage and current sensors. Right: the 6S1P pack built from the parameterized NCR18650BD cells 4
Figure 3. Left: The estimator takes pack current, cell voltage, cell temperature, and an initial estimate. It returns the estimated state of charge. Right: the block parameters, where the cell model from stage 1 is supplied as lookup data over SoC and temperature. 5
Figure 4. Left: estimated state of charge against true state of charge, the estimate begins low and pulls in. Right: estimation error settling toward zero over the cycle. 6
Figure 5. The PIL harness. The plant runs in simulation while generated code runs on the STM32G071RB. We compare signals at the boundary. 7
Figure 6. The code execution profiling report retrieved from the board, showing the feasibility checks and per-section timing. 8
Figure 7. Left: the telemetry model. An initialization subsystem configures the L9962. HW_Telemetry reads data cyclically and sends decoded signals to root outports. Right: the dashboard during an External mode run. It shows live cell voltages, pack summary values, fault lamps, and the raw 16-bit diagnostic register words displayed alongside the decoded flags. 9

IMPORTANT NOTICE – READ CAREFULLY

STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice.

In the event of any conflict between the provisions of this document and the provisions of any contractual arrangement in force between the purchasers and ST, the provisions of such contractual arrangement shall prevail.

The purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST’s terms and conditions of sale in place at the time of order acknowledgment.

The purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of the purchasers’ products.

No license, express or implied, to any intellectual property right is granted by ST herein.

Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product.

If the purchasers identify an ST product that meets their functional and performance requirements but that is not designated for the purchasers' market segment, the purchasers shall contact ST for more information.

ST and the ST logo are trademarks of ST. For additional information about ST trademarks, refer to www.st.com/trademarks. All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

© 2026 STMicroelectronics – All rights reserved