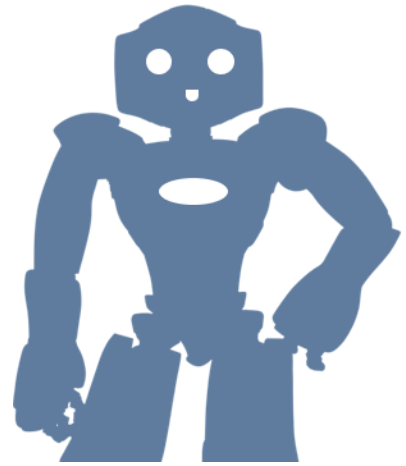
MATLAB EXPO 2019

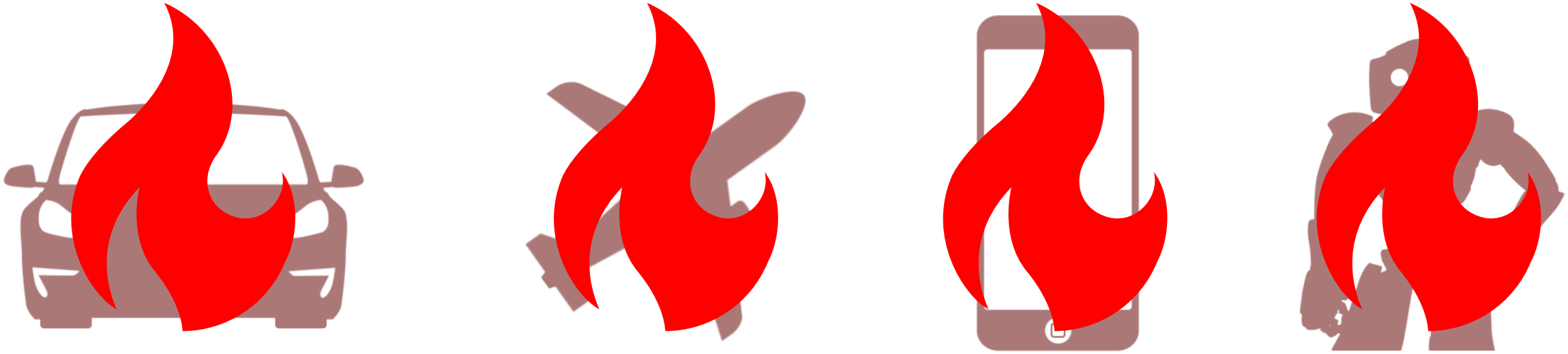
Developing Battery Management System using Simulink

Romain LACHAUX

Battery: a good answer to energy storage across industries…



… with some risks to keep under control



CX LIVE

At Least Three Electric Vehicles Have Spontaneously Exploded in China This Week

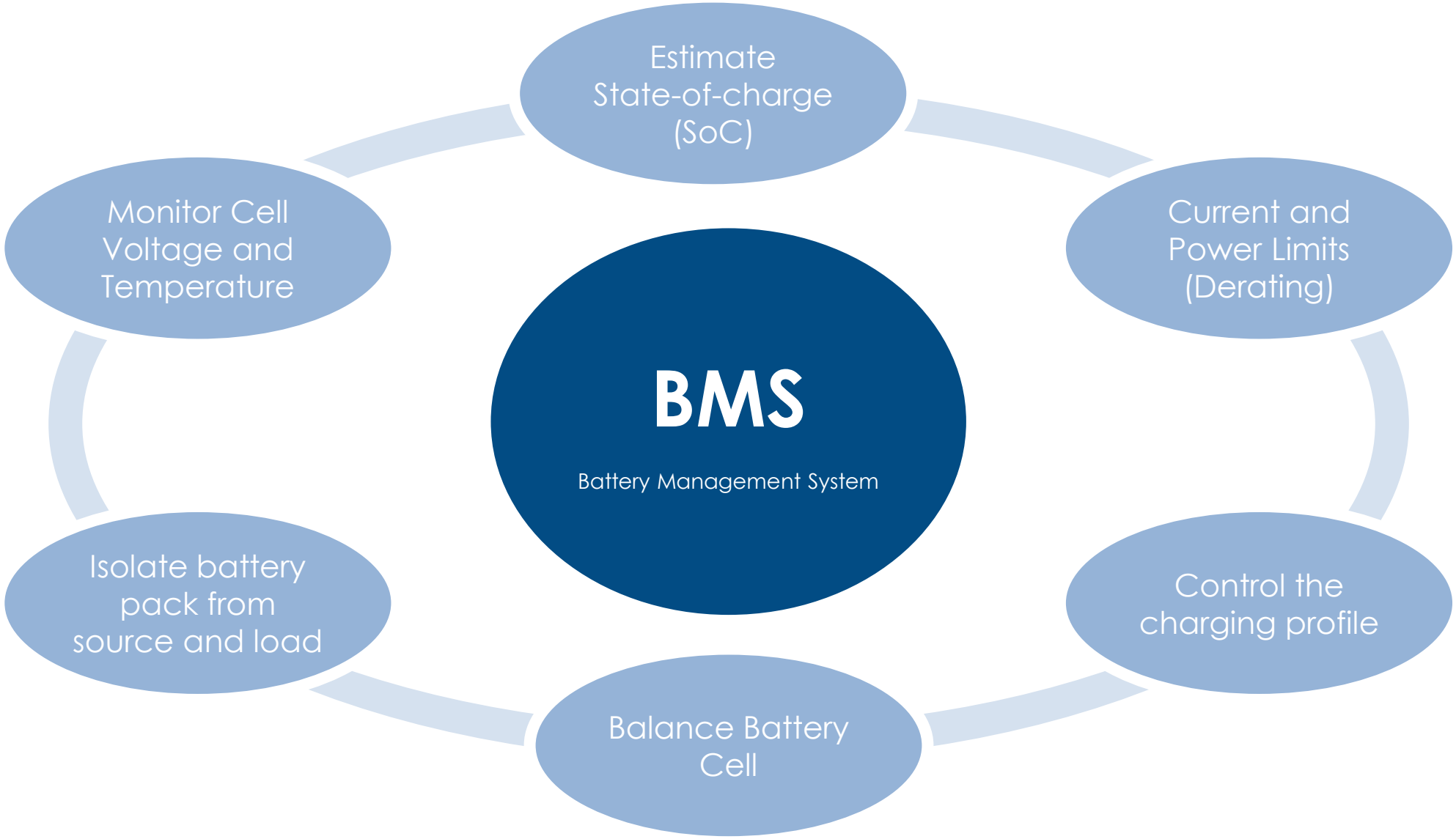
By Zhao Runhua / Apr 25, 2019 08:28 PM / Business & Tech

Newsweek

RYANAIR FLIGHT EVACUATED AFTER PHONE CATCHES FIRE ON PLANE

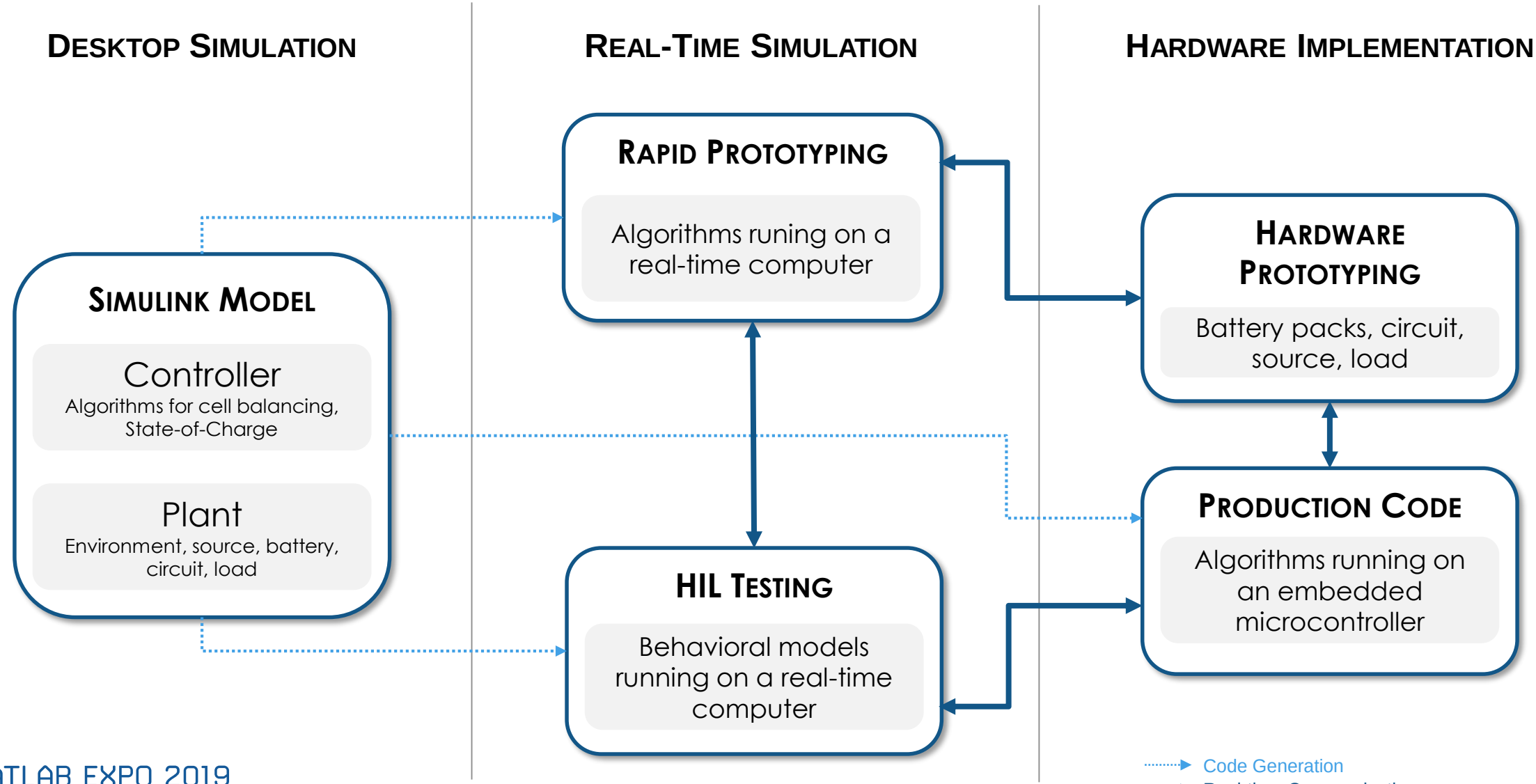
BY JENNI FINK ON 8/1/18 AT 11:00 AM EDT

CHALLENGE: Design and verify **battery management functions**


Estimate State-of-charge (SoC)
Current and Power Limits (Derating)
Control the charging profile
Balance Battery Cell
Isolate battery pack from source and load
Monitor Cell Voltage and Temperature
BMS
Battery Management System

SOLUTION: Perform **system-level simulations** with **Simulink**

BMS Development Workflow with Simulink and MBD



Agenda

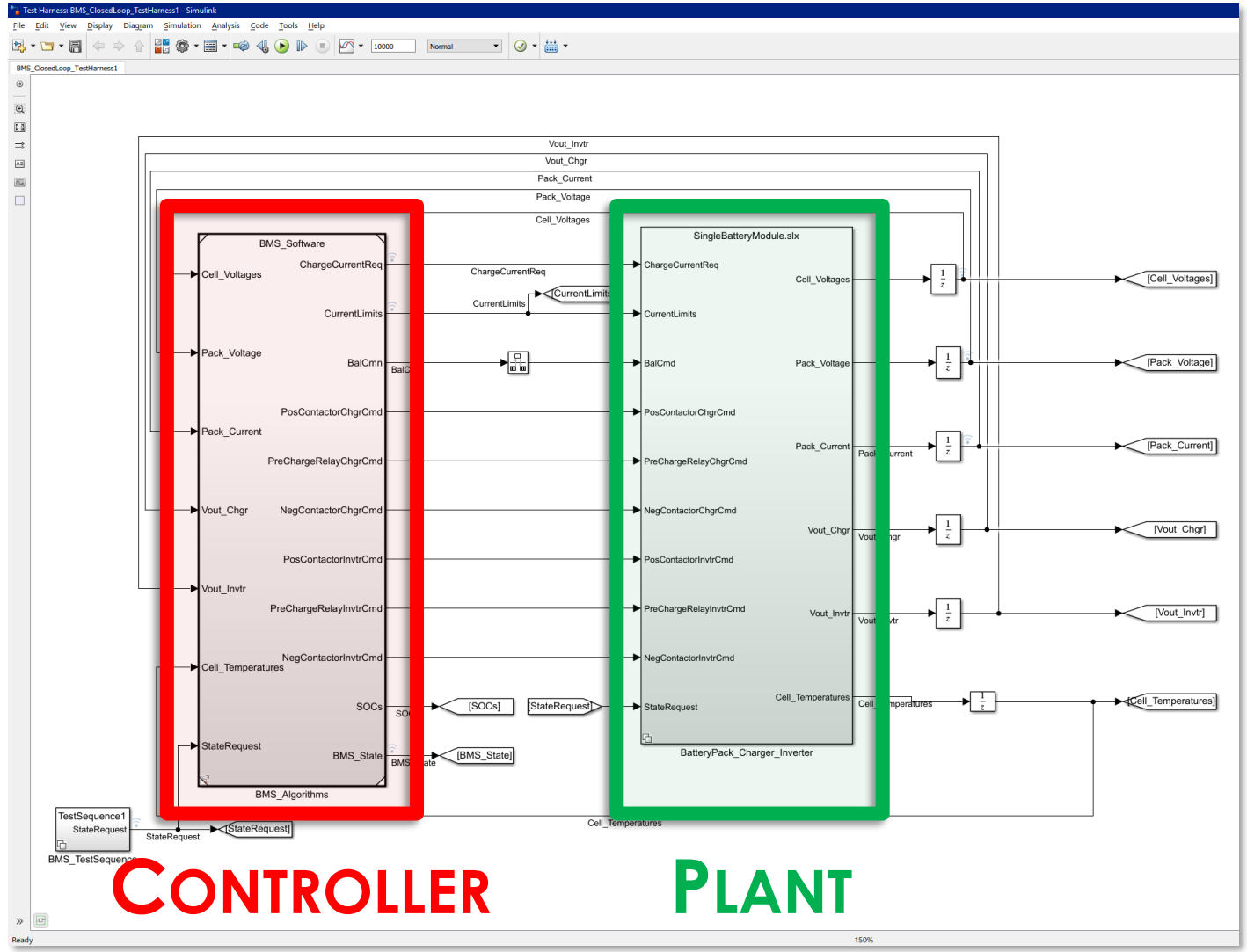
- **BMS Model Demo**
  - **Physical Modeling**
  - **BMS Algorithms**

- Deployment on Hardware
  - Code Generation
  - Real-Time Testing

BMS Model Overview

- System Model
  - Controller: BMS Algorithms
  - Plant : Physical Modeling

- Advantages of System-Level Simulation:
  - Quick design iterations
  - Early results in the development workflow
  - Possible to test each part alone or together in the same model (**Closed-loop testing**)



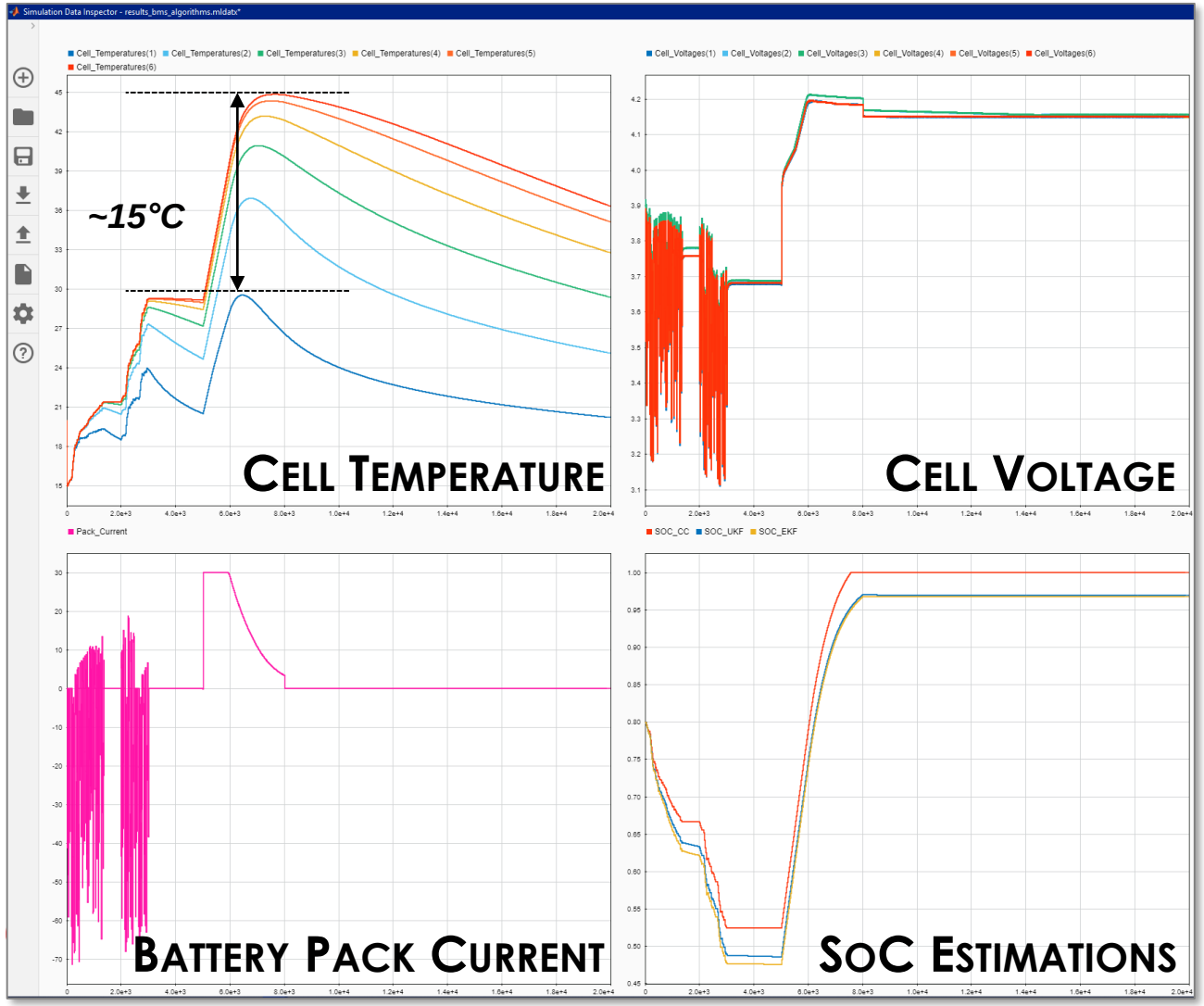
Simulation Results Overview

- **Early results** during design process

  ➔ Possible to refine or add missing requirements

- Example:
  - Temperature differences
  - Potential impact on cells ageing

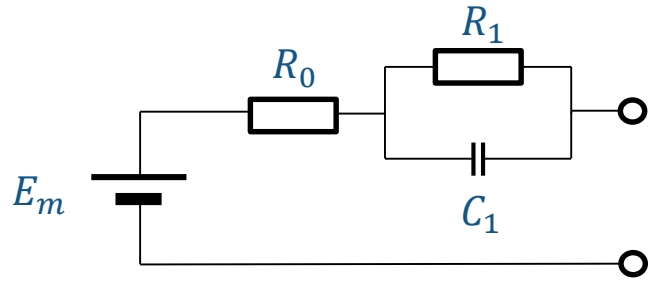
  ➔ Need of a cooling system?



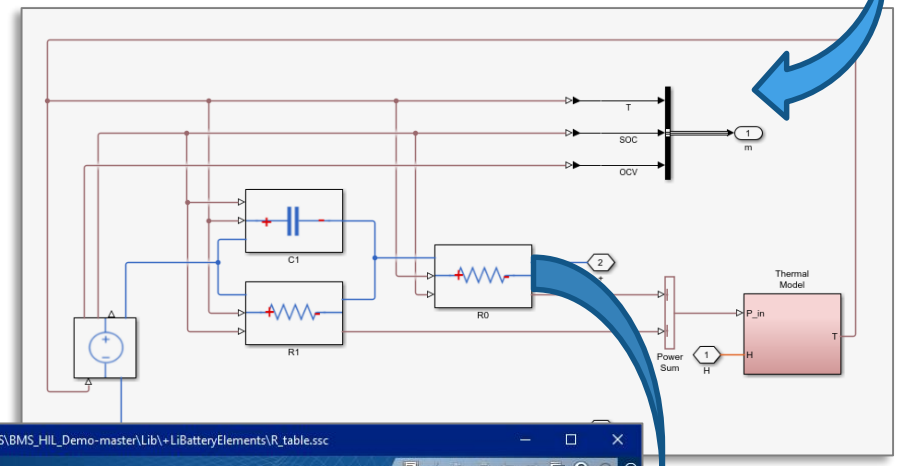
PLANT: Battery Physical Modeling with **Simulink** and **Simscape**

Battery Cell Modeling

- **Thevenin Model** (1st Order) to represent electrical behavior of battery cell
- Model based on Simscape Foundation Library components…
- ... with dependance upon SoC and temperature by **modifying source code**

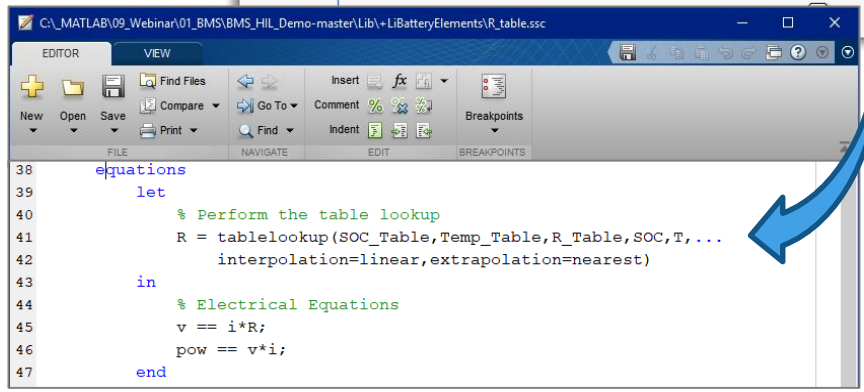


Thevenin Model

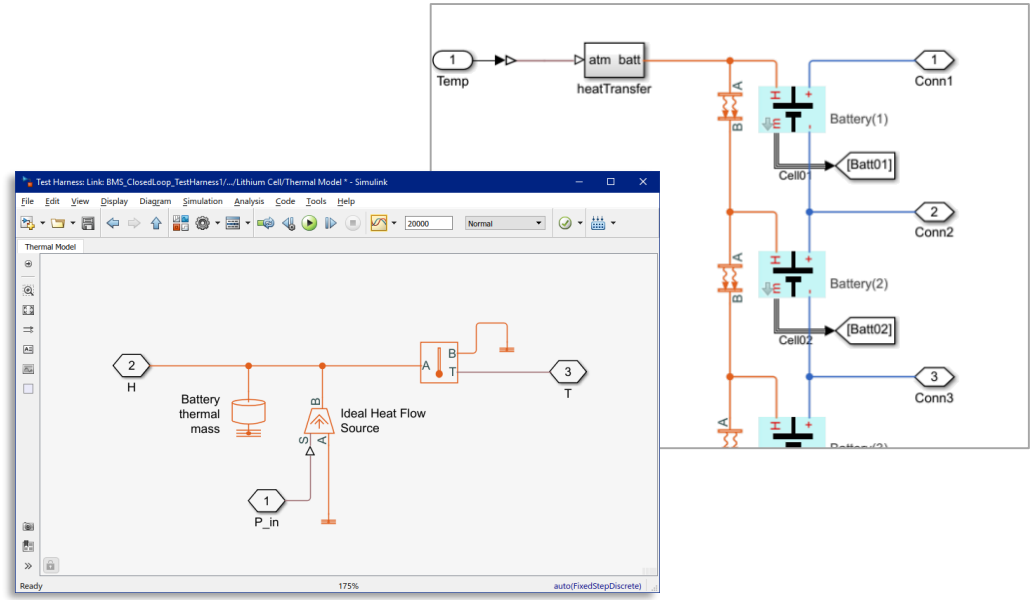



```
equations
    if v > Vf
      i == (v - Vf*
    else
      i == v*Goff;
```

Custom Component

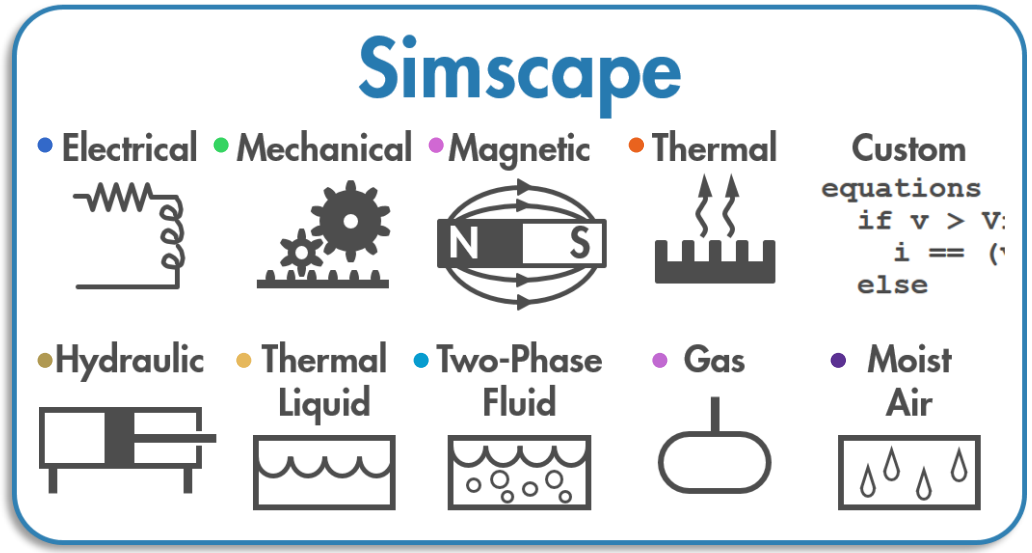


Battery Cell Modeling

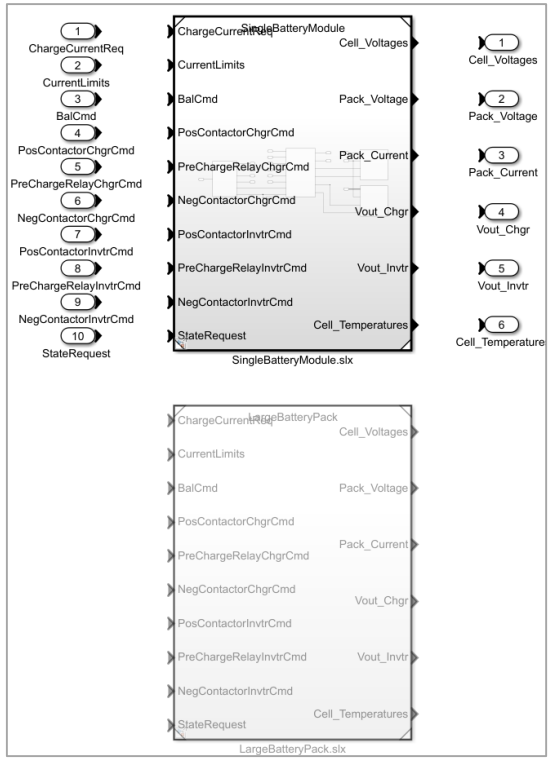


- What about **thermal behavior**?
  - Cell heat up under load
  - Convection heat flux between cells
  - Thermal exchange with environment

➔ **Thermal component** from Foundation Library



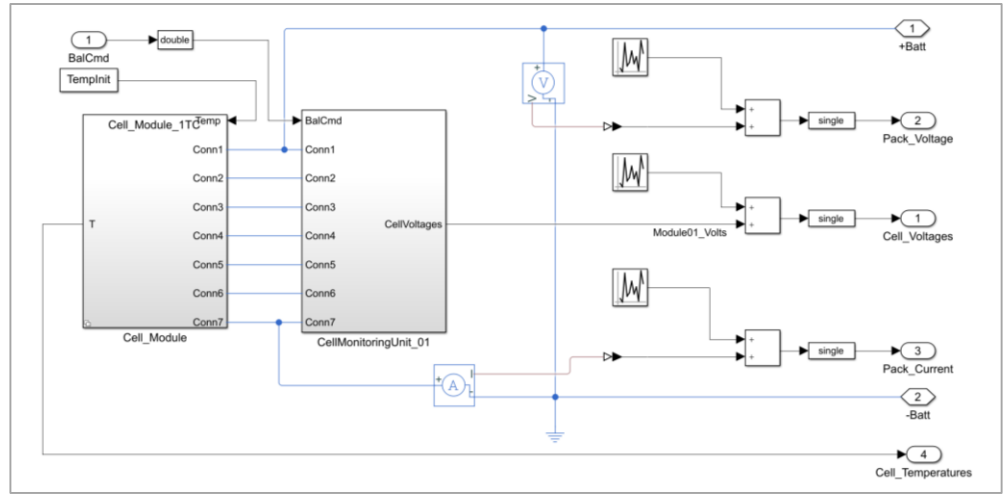
Battery Pack

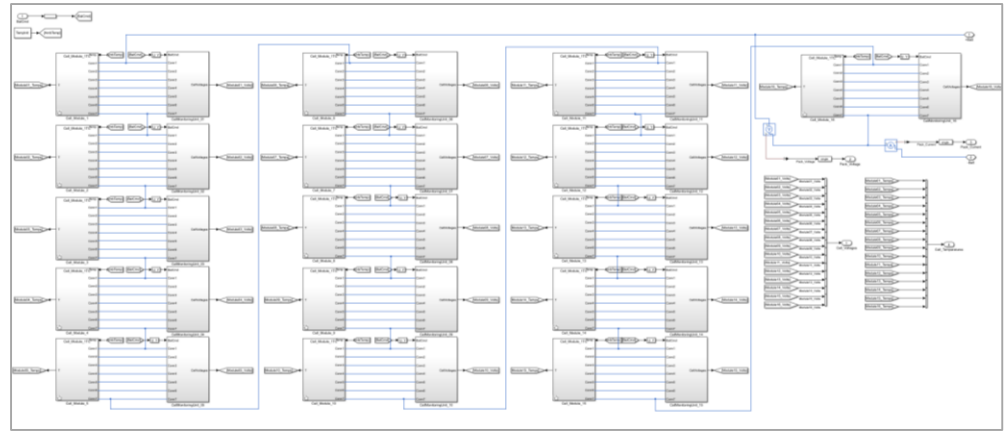


1 Module of 6 cells

OR

16 Module of 6 cells



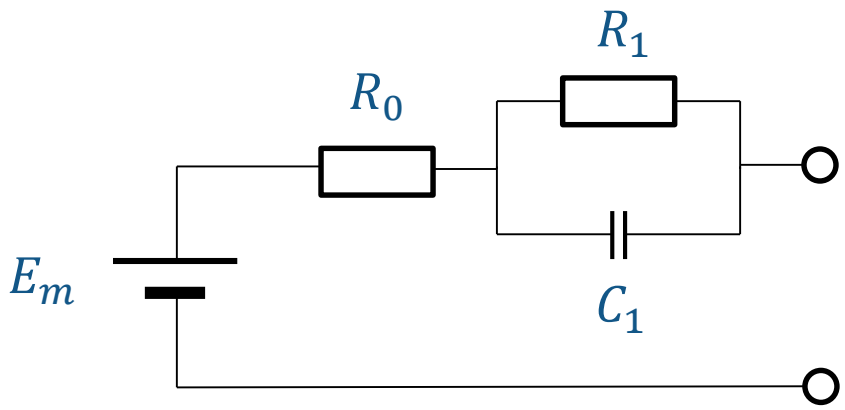


Variant Model

Possible to model **different architecture or technological choice** of a subsystem in the **same model**

Tuning a Lithium Battery Model to Match Measured Data

Cell Physical Model:



Thevenin Model

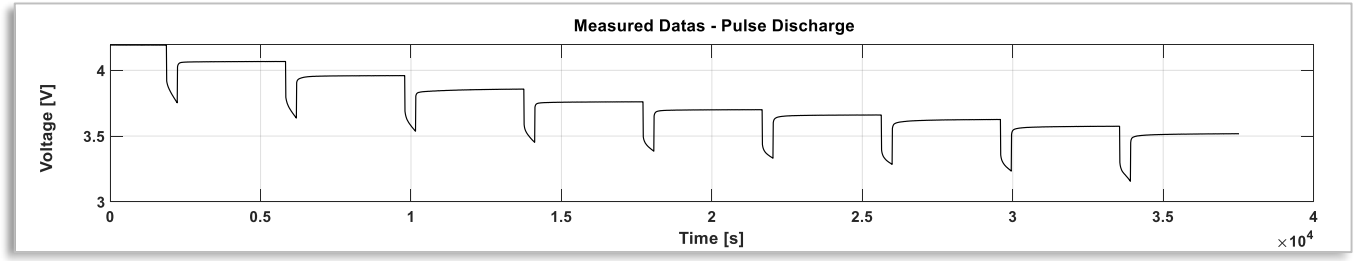
Lithium Cell Characteristic Measurement:

E_m	SOC 1	SOC 0.9	SOC 0.8	…	SOC 0
5°C	4.20 V	4.12 V	4.05 V	…	3.09 V
20°C	4.18 V	4.09 V	4.01 V	…	3.05 V
40°C	4.15 V	4.02 V	3.97 V	…	3.01 V

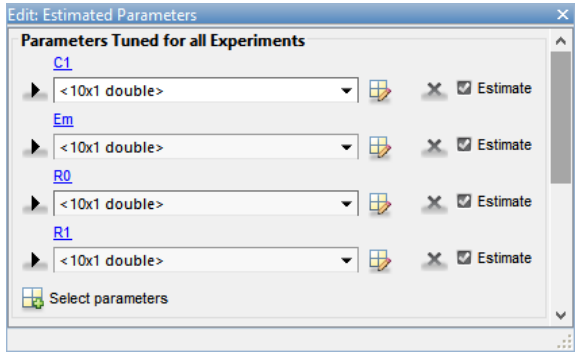
OBJECTIVE: Match model behavior to tests measurements

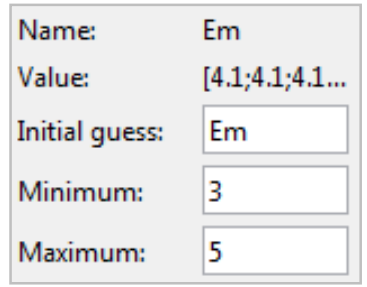
Estimating Parameters Using Measured Data

1. **Import** measurement datas

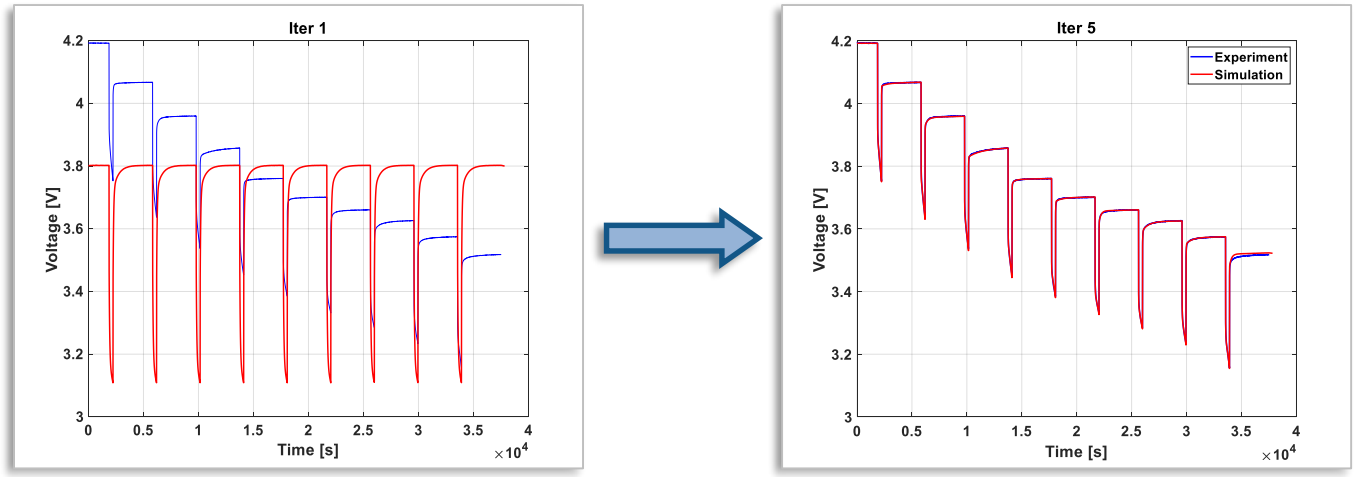


2. **Identify** parameters and set range



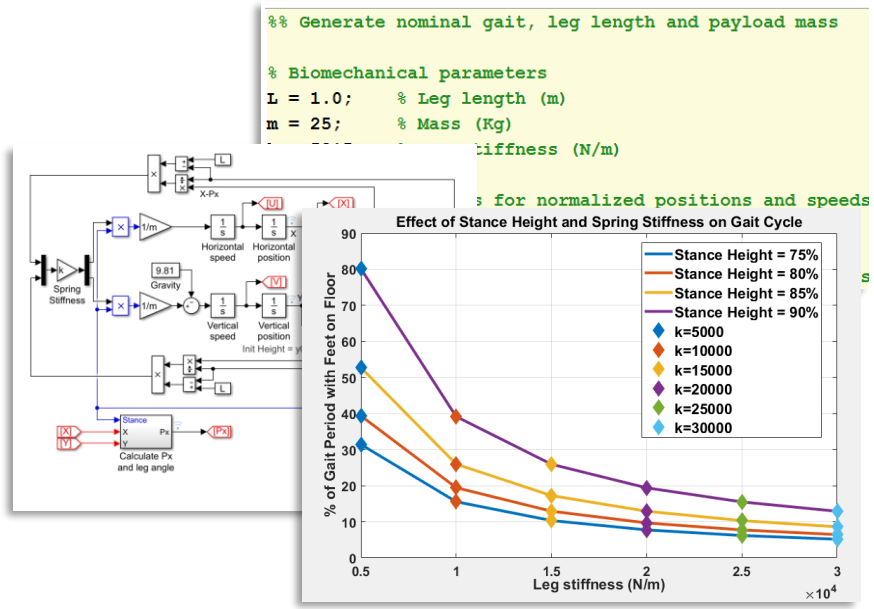


3. **Perform estimation**



Advantages of Physical Modeling

- With this physical model, you can
  - **Evaluate** your architecture
  - **Optimize** your design
  - **Refine** and **Validate** your requirements

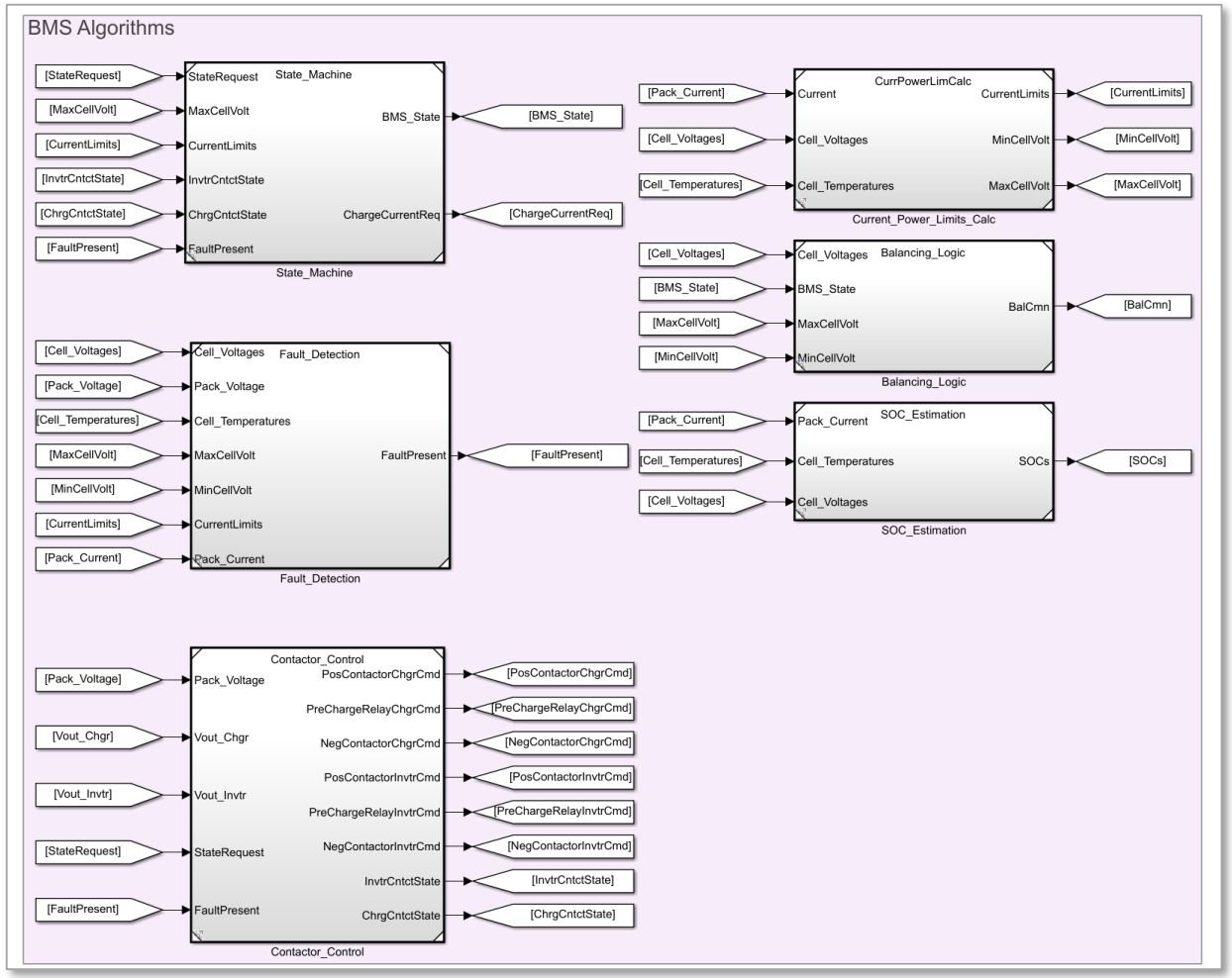


→ **Facilitate parameters sweep and iterations**
→ **Limit oversizing of components**

+ Enable **Closed-loop testing of your control algorithms** to verify and validate it

CONTROLLER: BMS Algorithms with **Simulink** and **Stateflow**

Battery Management System Functions

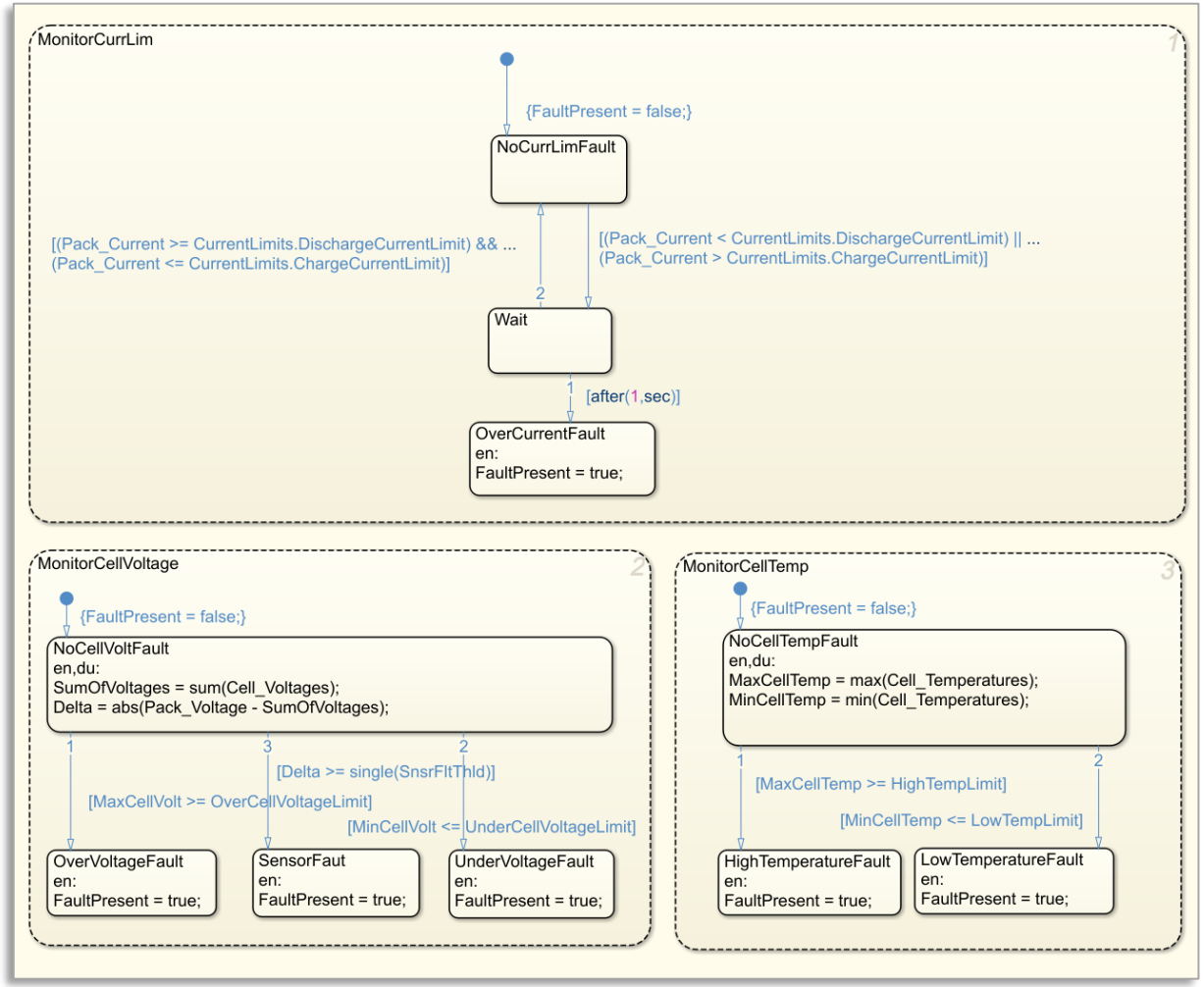


- **Battery State**
- **Fault Management**
- **Battery isolation control**
- **Derating Calculations**
- **State-of-Charge Estimation**

Fault Management

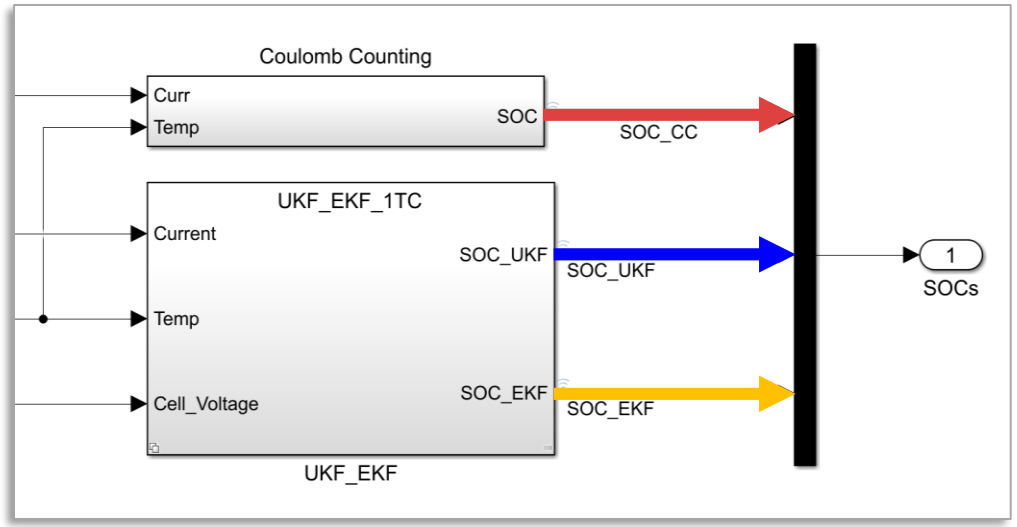
- Monitoring three physical channels:
  - Battery Pack **Current**
  - Cell **Voltage**
  - Cell **Temperature**

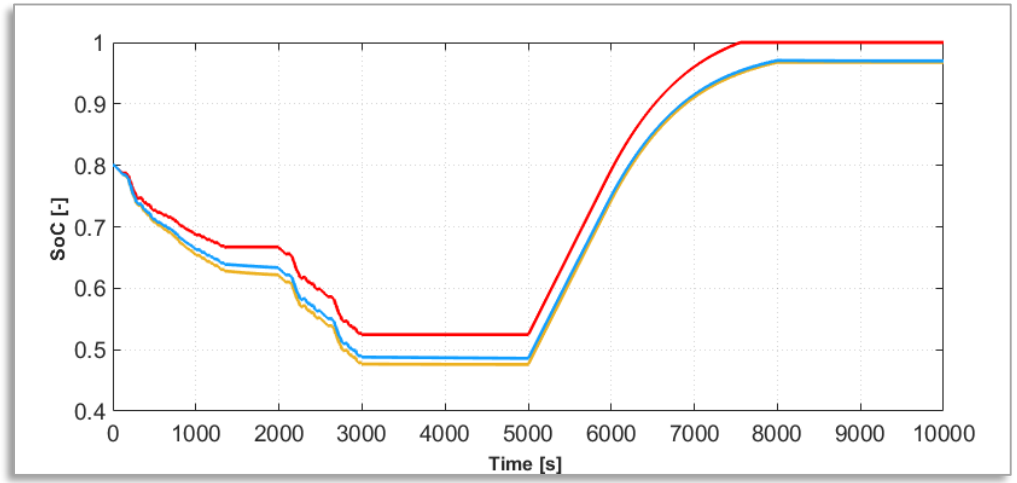
- Broadcasting Fault Presence to other BMS subsystems
  - → Contactor Opening (SAFETY)
  - → BMS_State == FAULT



State-Of-Charge Estimation

- Two methods:
  - **Coulomb Counting**
    - ✓ : Simple to implement / **low computational needs**
    - ✗ : Accuracy and robustness
  - **Kalman Filtering***
    - ✓ : **High accuracy** by including a nonlinear battery model which uses current and voltage measurement
    - ✗ : Slightly higher computational effort



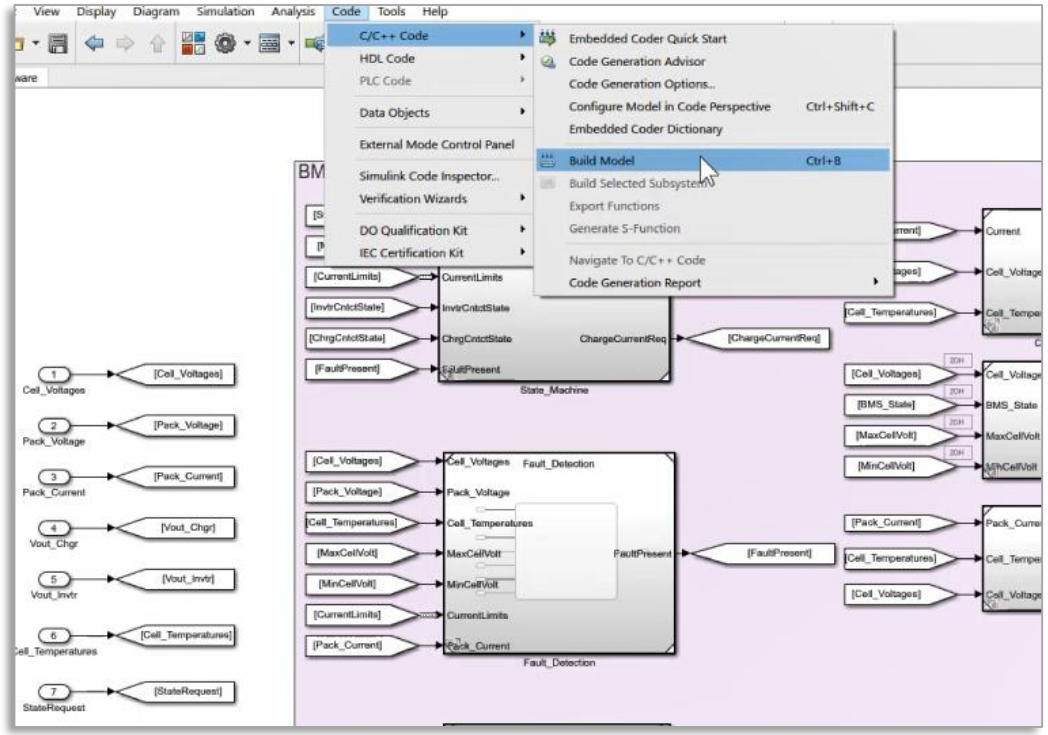


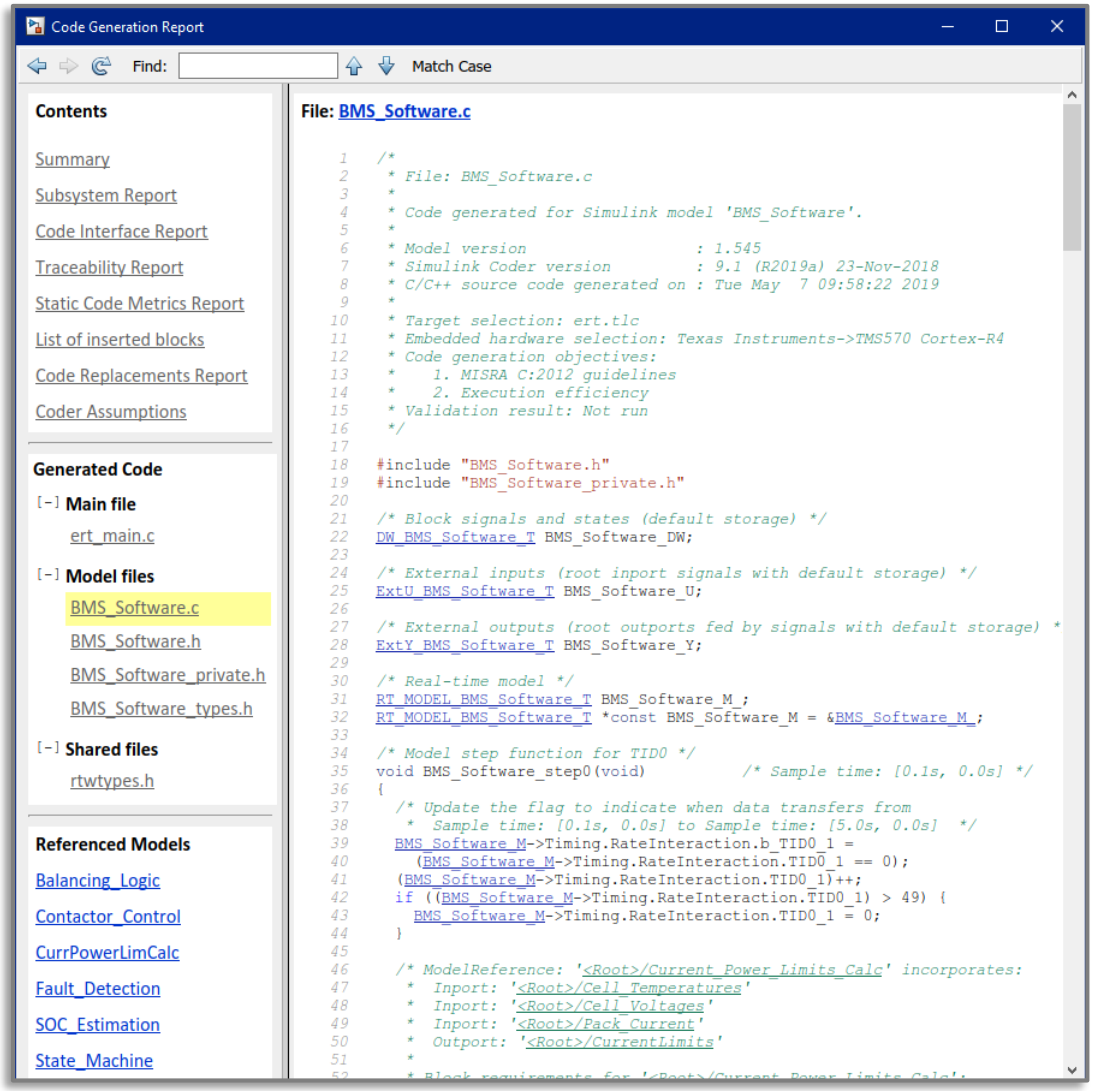
** : ready to use block available in Control System Toolbox or System Identification Toolbox*

Agenda

- BMS Model Overview
  - Physical Modeling
  - BMS Algorithms
- **Deployment on Hardware**
  - **Code Generation**
  - **Real-Time Testing**

Generate C/C++ Code From BMS Models



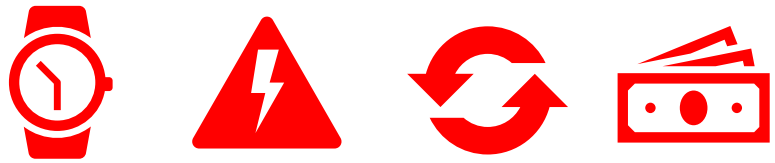


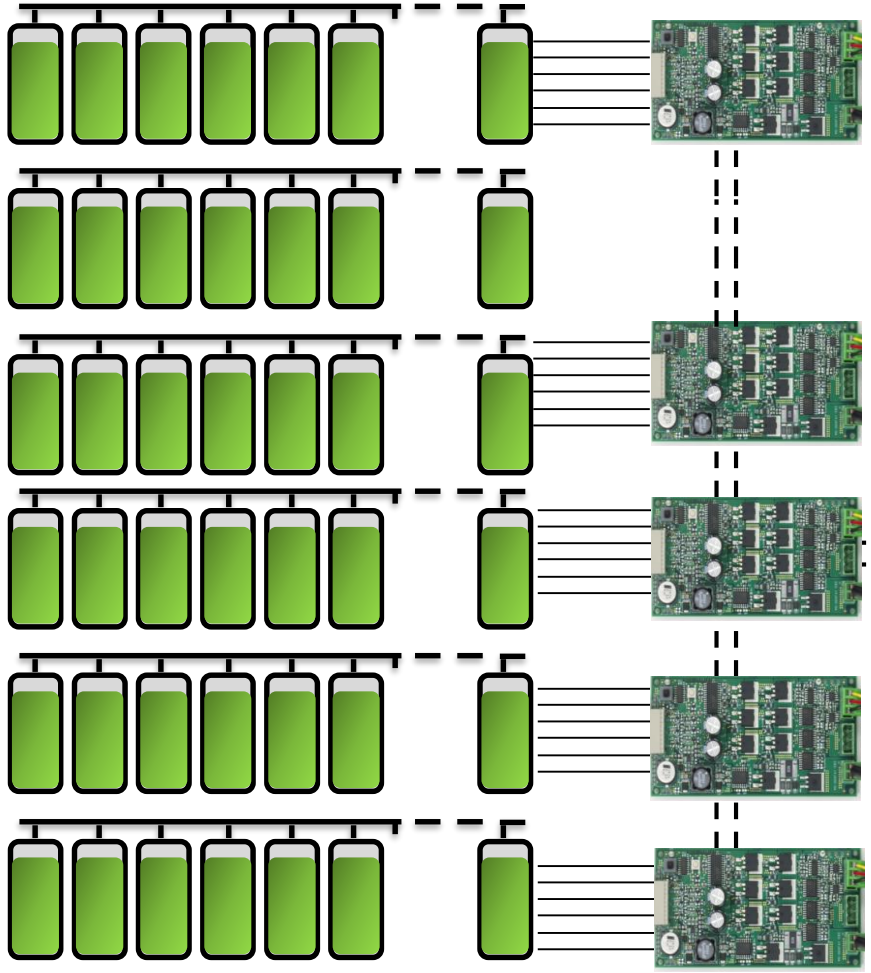
- Generate target optimized C/C++ code
- Fine-tune code optimizations, package and build generated code

Real-Time Testing of Battery Management System

- Testing BMS with Battery Cells
  - Longer test cycles
  - Difficult to test fault conditions
  - Difficult to reproduce results
  - Limited test automation

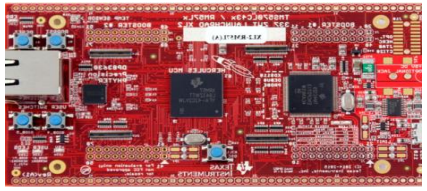
➔ Costs (Hardware prototype, possible failure, several people to perfom tests, etc)





Battery Pack

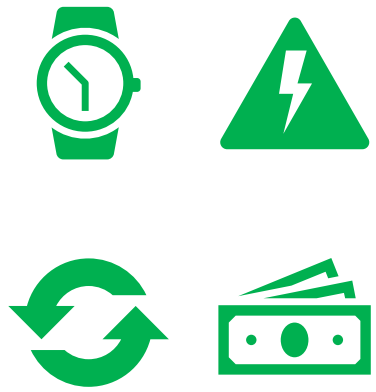
Measurement & Diagnostics

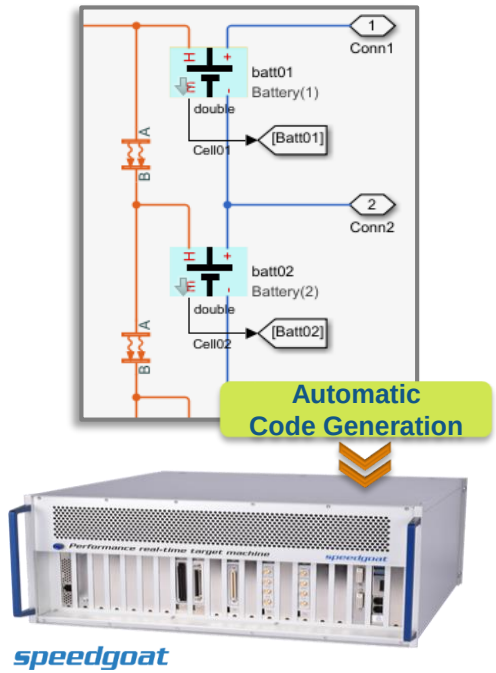


Main Controller

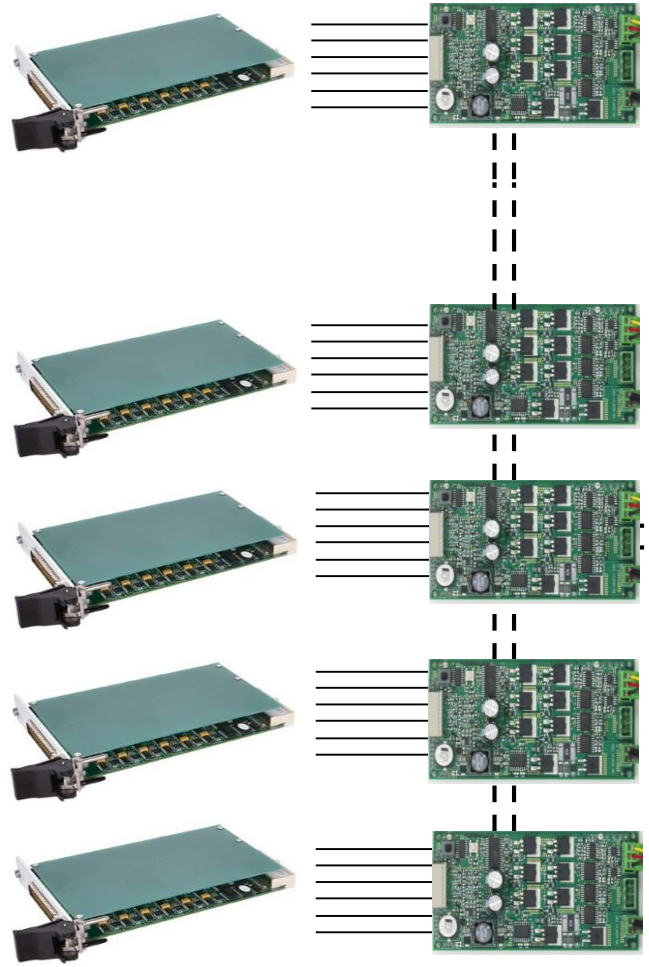
Hardware-In-Loop Testing of Battery Management System

- Testing BMS with Emulated Battery Cells
  - Reduce testing time
  - Test fault conditions safely
  - Automate testing

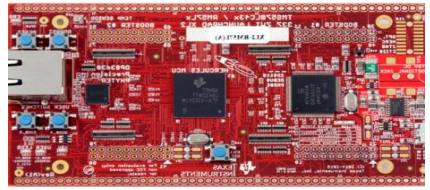




Battery Emulation

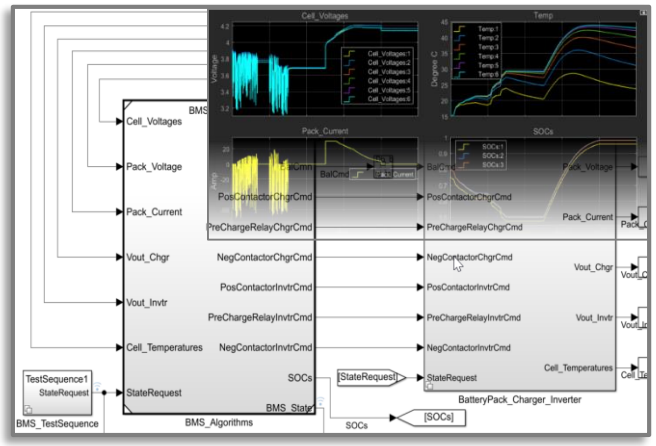


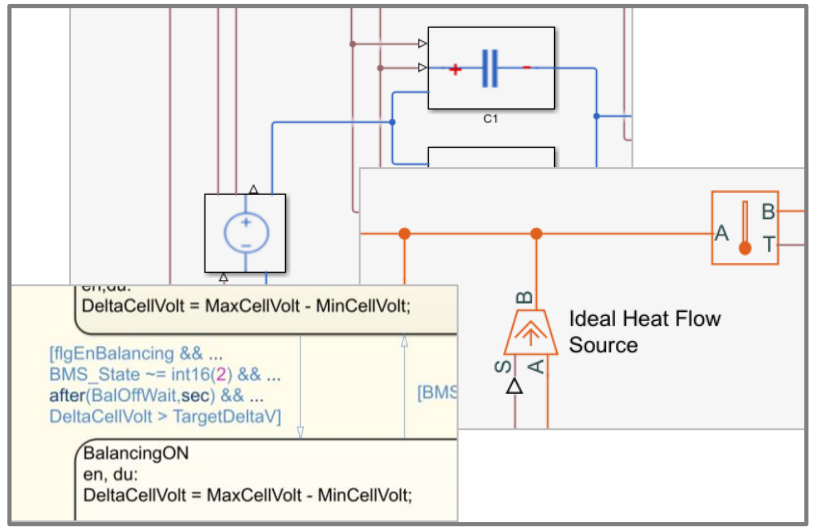
Measurement & Diagnostics

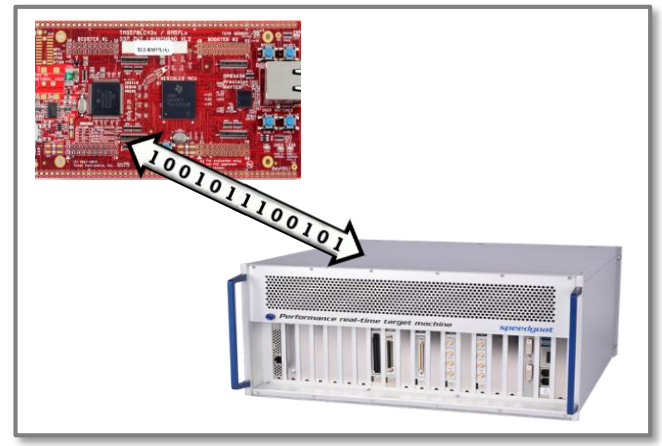


Main Controller

BMS Development with Simulink







Reduce Design Iteration Time

Collaborate Across Domains

Gain Confidence in Design