

MATLAB EXPO

Developing Electrified Propulsion Systems for a Sustainable Future

Rahul Choudhary, MathWorks



Ramanuja Jagannathan, MathWorks





MathWorks ✓

@MathWorks

Share the EXPO experience
#MATLABEXPO



www.linkedin.com/in/rch193



<https://www.linkedin.com/in/jrvinayak/>

From Grid to Vehicle

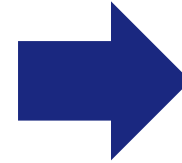
Clean Energy



Low-carbon fuels

**Energy efficiency
and electrification**

Battery storage



Design Challenges



Every bit of energy wasted reduces efficiency/ range



Improperly sized components can add weight or waste energy



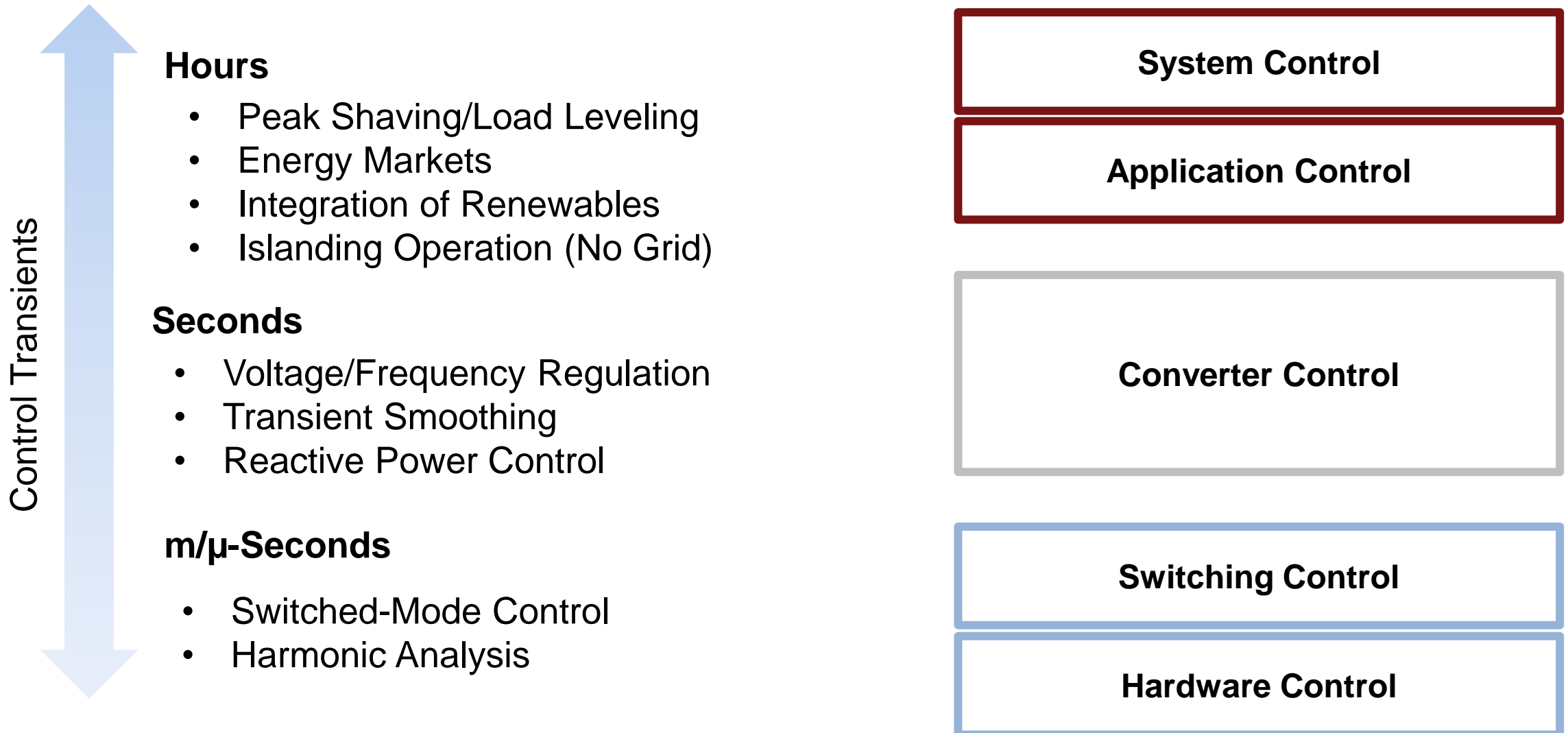
Overly complex solutions may reduce efficiency if not optimized



Analysis becomes important to predict issues

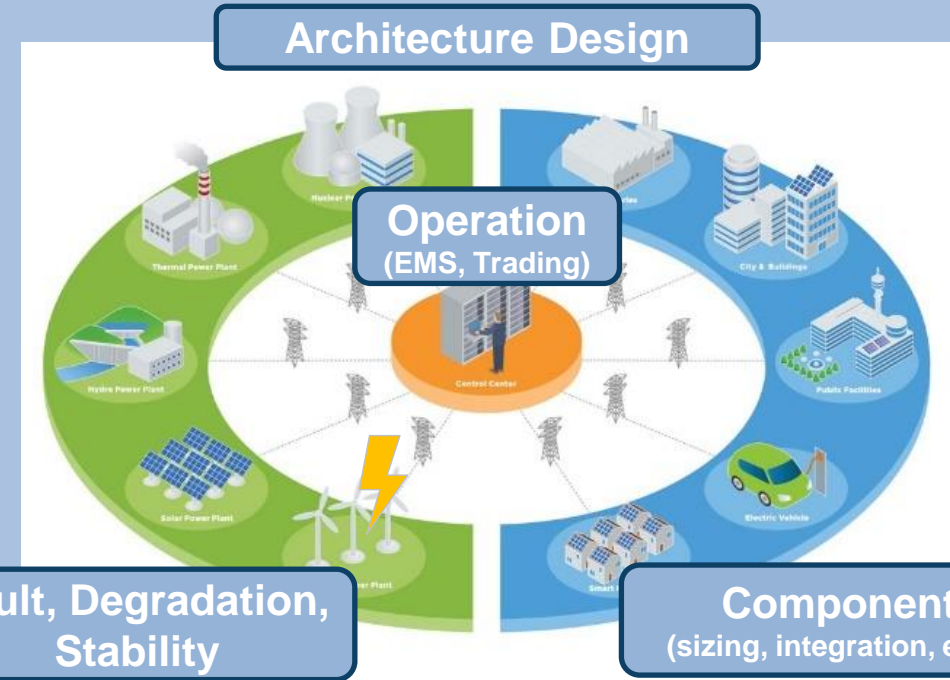
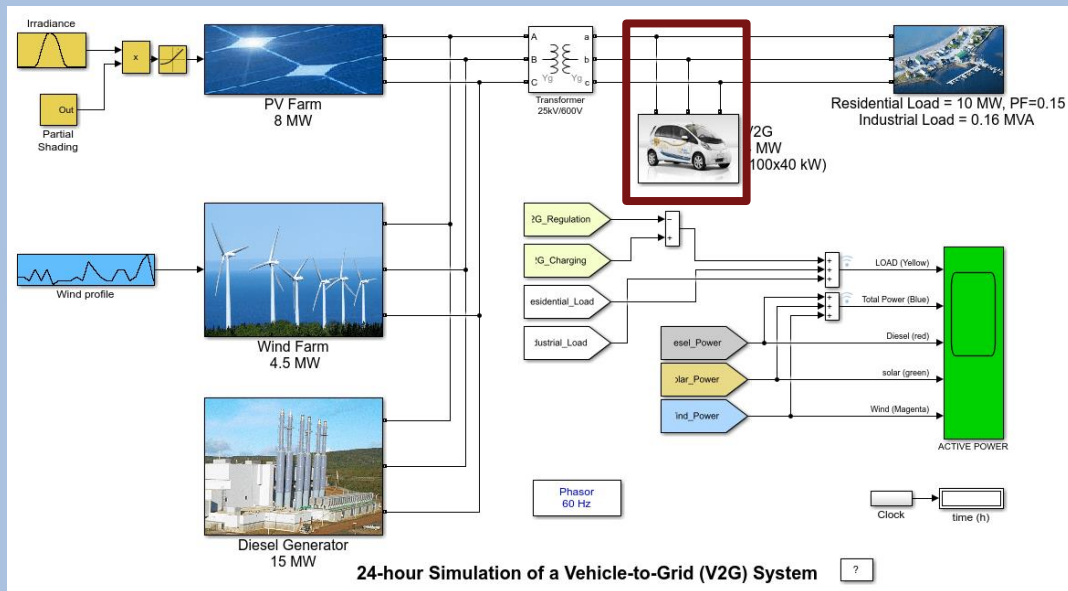
On the Grid Side

IEEE1676 – Control Architecture for High Power Electronic



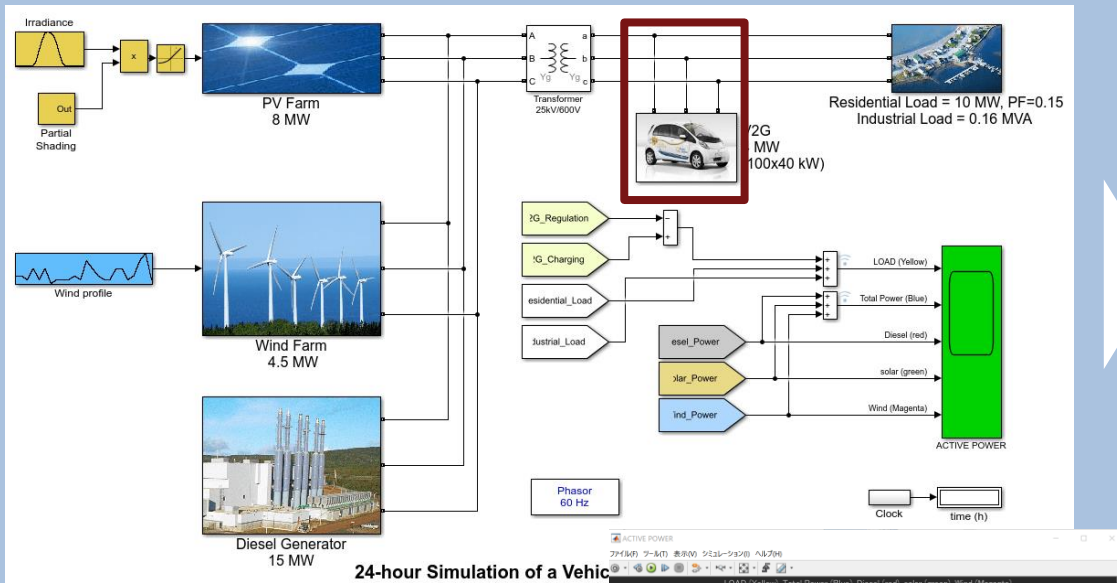
Design Flexibility

Low Fidelity

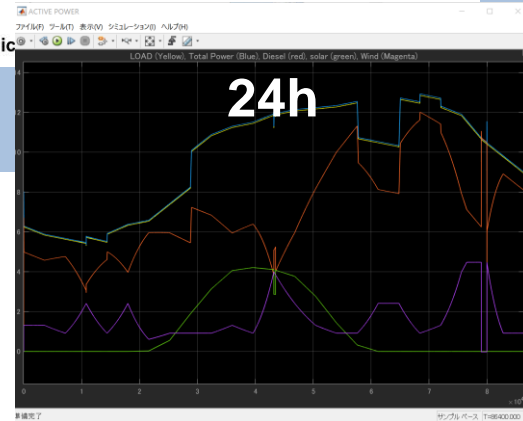


Design Flexibility

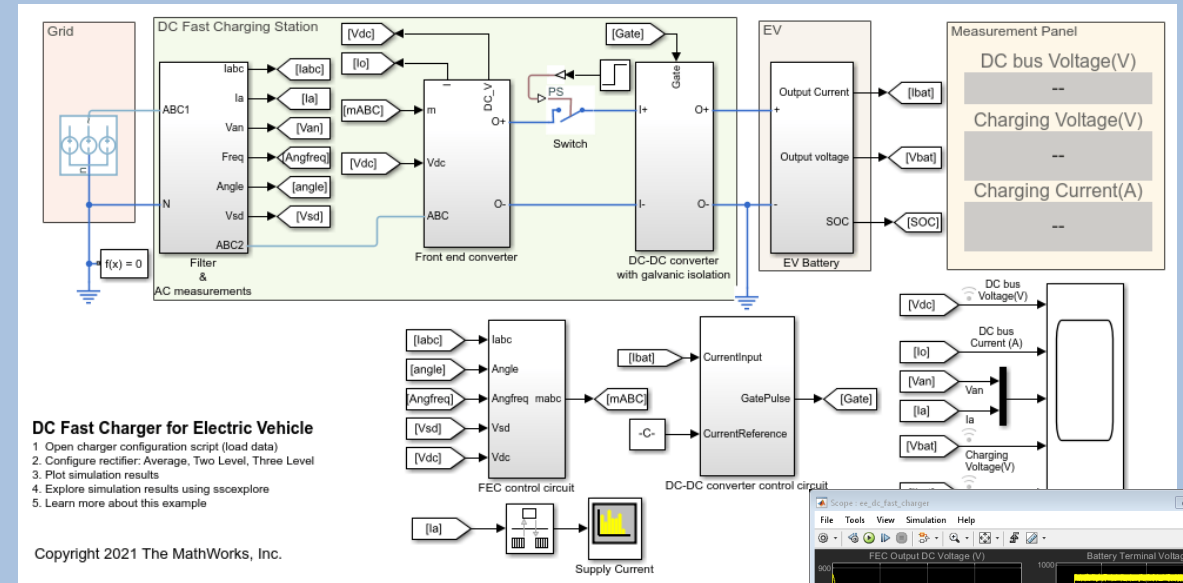
Low Fidelity



24-hour Simulation of a Vehicle



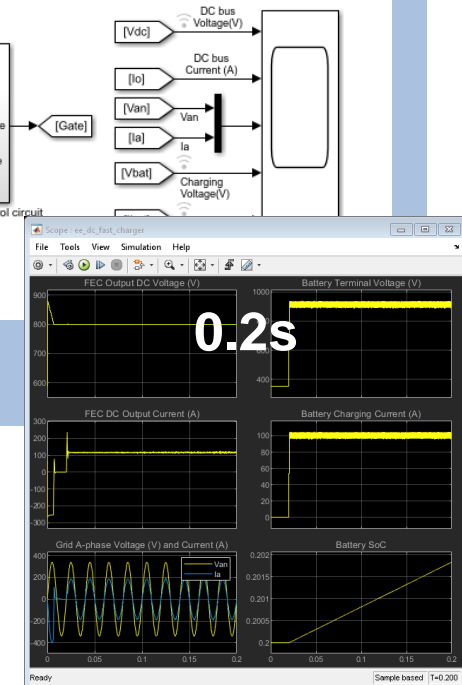
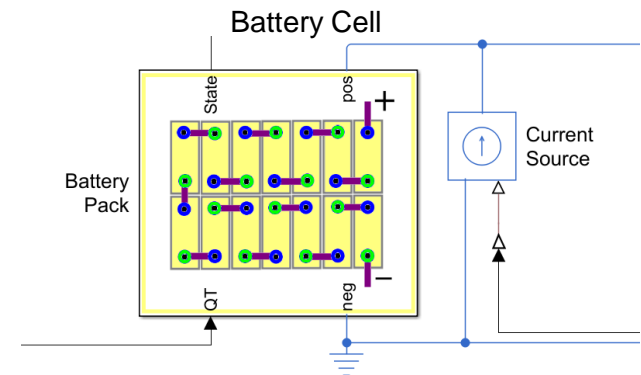
High Fidelity



DC Fast Charger for Electric Vehicle

1. Open charger configuration script (load data)
2. Configure rectifier: Average, Two Level, Three Level
3. Plot simulation results
4. Explore simulation results using sscope
5. Learn more about this example

Copyright 2021 The MathWorks, Inc.



Systems Engineering: System Composer

Requirement

HYBRID ELECTRIC VEHICLE POWERTRAIN REQUIREMENT
 ©Copyright 2020 The MathWorks, Inc.

1 HYBRID ELECTRIC VEHICLE OVERVIEW - updated

This document describes a requirement specification for the Hybrid Electric Vehicle, focused on the Powertrain part.

The ECU's of the full HEV must work together to control the following systems:

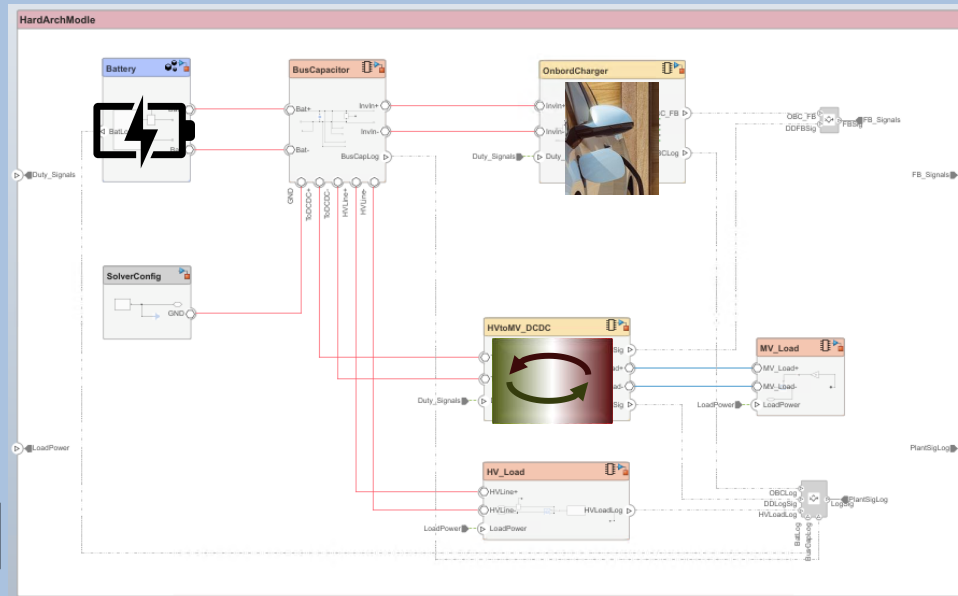
- Internal combustion engine
- Transmission
- Battery
- Electric Machine (Motor)

2 POWERTRAIN SYSTEMS

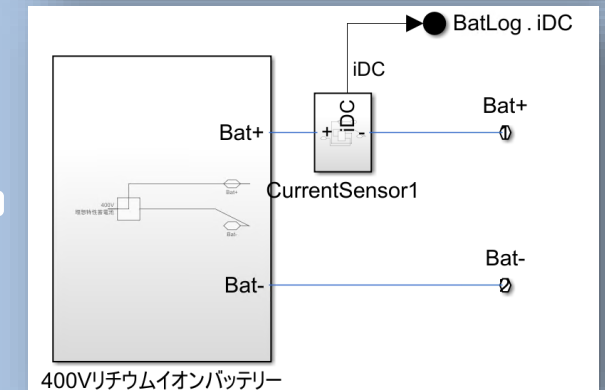
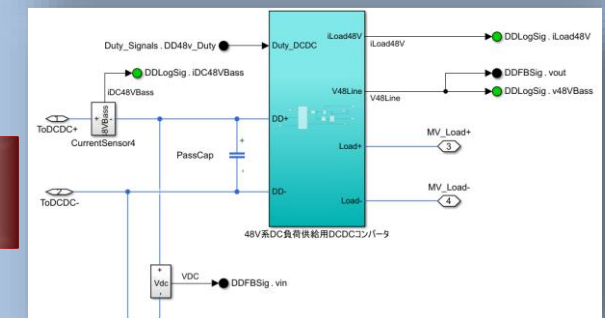
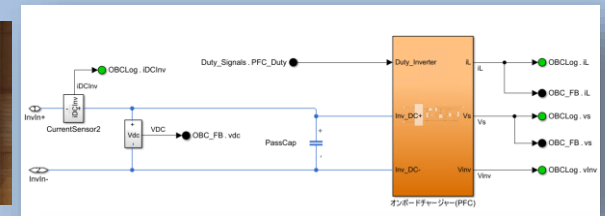
2.1 OVERVIEW AND CONFIGURATION
 The HEV Powertrain is a P4 configuration:



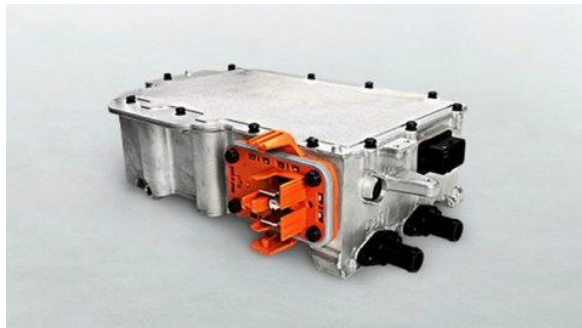

System Architecture



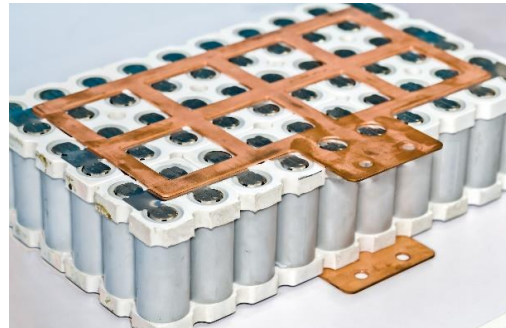
Component



On e-Mobility Side



Power Electronics

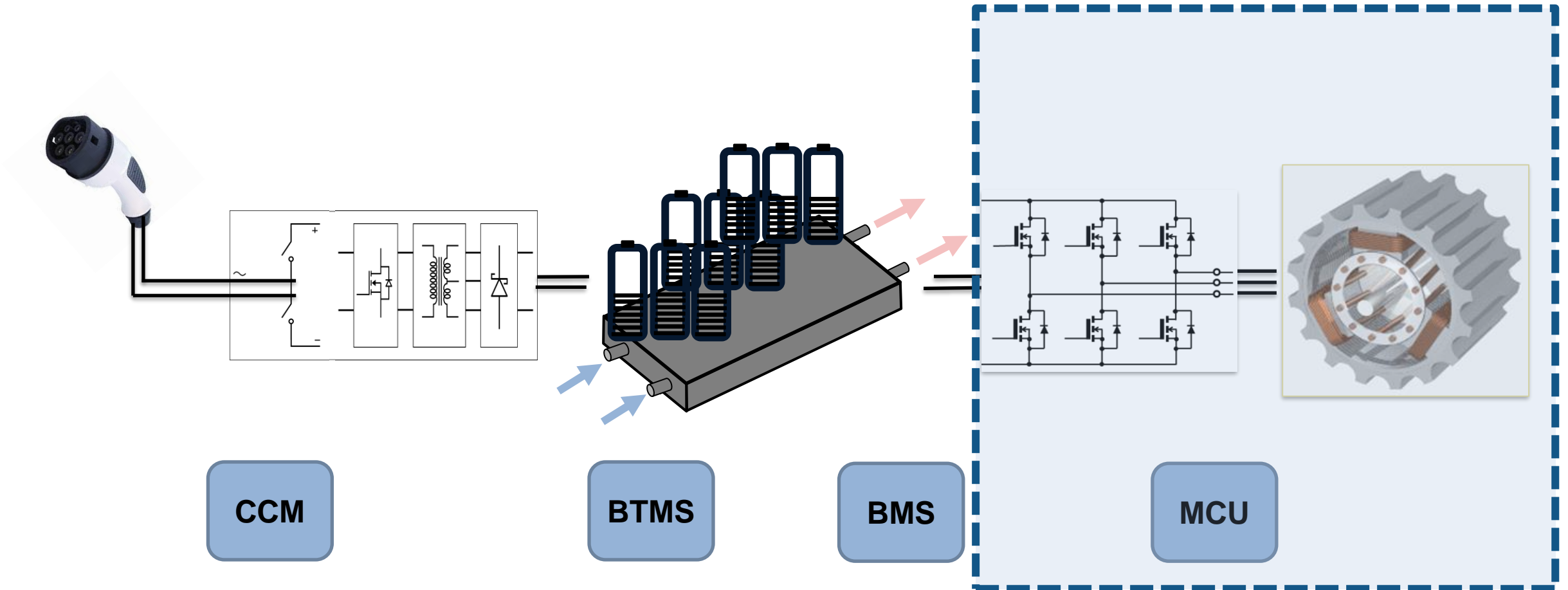


Battery Management



Motor Control

Electric Vehicle System



Common Challenges with Motor Control Development

- Developing accurate motor models with varying model fidelity
- Motor constraint curves and Identifying optimal operating condition
- Tune Control loop gains to get the desired performance
- Code generation and Deployment to a target hardware

ebm-papst Develops Electric Auxiliary Oil Pump for Automatic Transmissions Using Model-Based Design

Challenge

Develop, verify, and calibrate an automotive auxiliary oil pump without a pressure sensor

Solution

Use Model-Based Design to model and simulate the controller, and use Simulink Real-Time to validate the design and automate system identification and calibration

Results

- Overall development time halved
- System investigation time cut by 60%
- Deployment on specified microcontroller supported

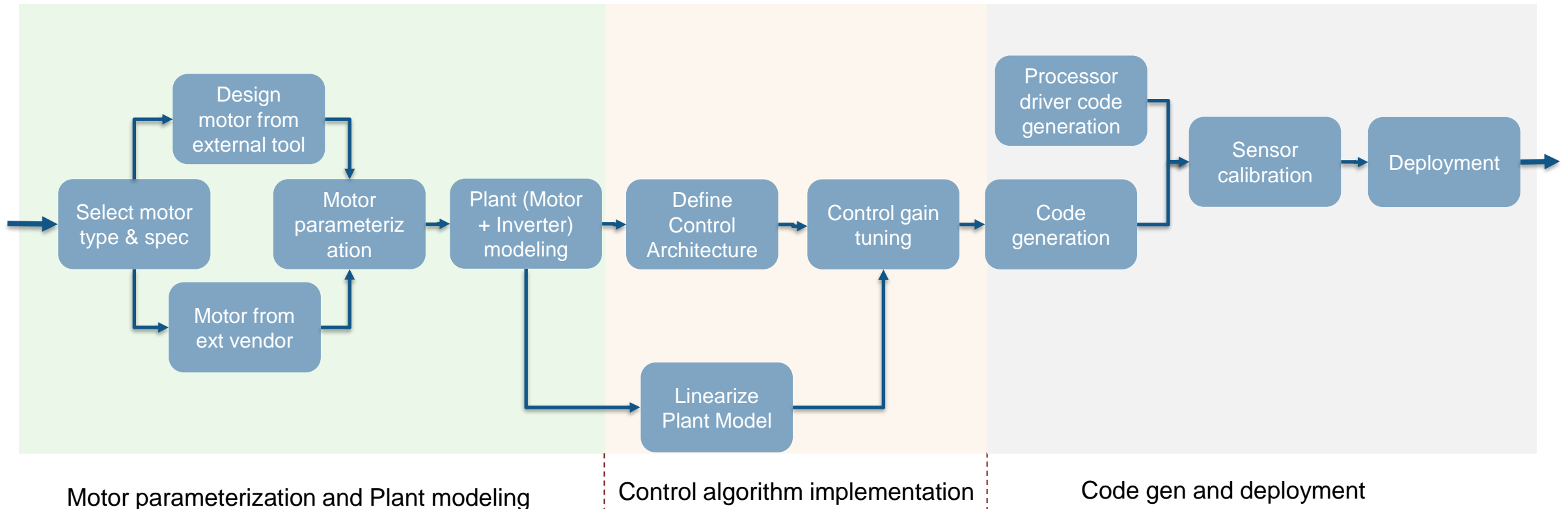
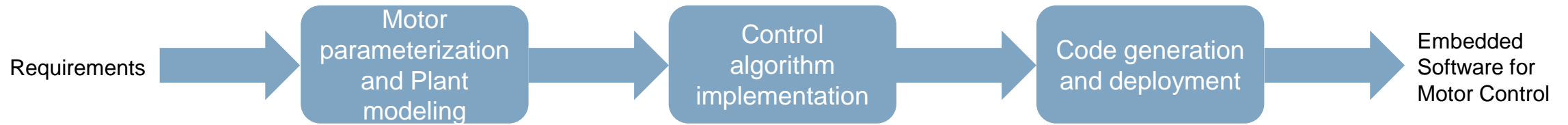


ebm-papst's automotive auxiliary oil pump without a pressure sensor.

“Given the time pressure we faced, Model-Based Design was our only chance to design a controller that satisfied our customer’s pressure regulation requirements, build a test rig to automate labor-intensive tests, and rapidly set up a calibration process in production to maximize performance.”

– Jens Löffler, ebm-papst

Workflow for Motor Control Development

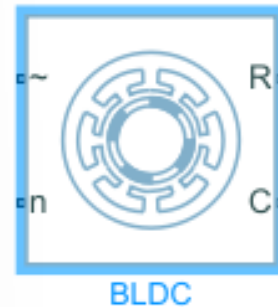


Motor Parametrization

Block Parameters: BLDC

BLDC Auto Apply

Settings	Description
NAME	VALUE
Modeling option	No thermal port
Selected part	<click to select>
▼ Rotor	
Electrical connection	Composite three-phase ports
Winding type	Wye-wound
Back EMF profile	Perfect trapezoid - specify maximum
> Maximum permanent magnet flu...	0.03 Wb
> Rotor angle over which back em...	$\pi / 12$ 0.2618 rad
> Number of pole pairs	6
Rotor angle definition	Angle between the a-phase magne
> Stator	
> Iron Losses	
> Mechanical	
> Initial Targets	
> Nominal Values	



Block Parameterization Manager: BLDC

SELECT | FORMAT

Apply all | Reset all

Manufacturer: All

PARAMETERIZE

Select part

Part number	Manufa	N*m	
BLY171D_24V_1400	Anaheim	67.4943	
BLY171D_24V_2800	Anaheim	86.0097	
BLY171D_24V_6000	Anaheim	39.2481	
BLY171S_15V_8000	Anaheim	30.7601	
BLY172D_24V_4000	Anaheim_Automation	53.0000	124.3186
BLY172S_24V_2000	Anaheim_Automation	41.0000	223.2015
BLY172S_24V_4000	Anaheim_Automation	53.0000	143.5967
BLY173D_24V_4000	Anaheim_Automation	77.0000	186.1566
BLY174D_24V_12000	Anaheim_Automation	113.0000	75.1208
BLY174S_24V_12000	Anaheim_Automation	113.0000	75.1208

From Motor Datasheet

From Pre-parametrized Motors

Parameter Estimation by Running Instrumented Tests

- Instrumented tests running on the target
- Sensor-based and Sensorless modes available
- Supports PMSM and Induction Motor

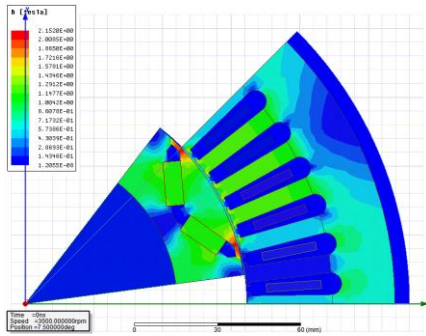
The screenshot displays the MATLAB/Simulink interface for parameter estimation. Key sections include:

- Board Selection:** DRV8305 and F28379D Launchpad.
- Communication Port:** Serial Setup.
- Required Inputs:**
 - Nominal Voltage: 24 V
 - Nominal Current: 7.1 A (rms value)
 - Nominal Speed: 4000 rpm
 - Pole pairs: 4
 - Input DC Voltage: 20 V
 - Hall Offset: 0.28 Per Unit Position
- Test Status:** Run (checked), Stop.
- Estimated Motor Parameters:**
 - R_s : 0.4775 ohm
 - L_d : 2.1052e-04 H
 - L_q : 2.0117e-04 H
 - B_{emf} : -- V/krpm
 - Motor Inertia: -- kg.m²
 - Friction constant: -- N.m.s
- Fault Status:** Over Current, Under Voltage, Serial communication.
- Signal Conditioning and Scaling:** Algorithm section.

A photograph of the TI C2000 hardware, consisting of a red PCB and a motor, is overlaid on the bottom right of the screenshot.

Figure : gif showing parameter estimation capability with TI C2000 hardware

Motor Parametrization using FEA tools



```

Editor - C:\Program Files\MATLAB\R2021a\toolbox\physmod\elec\eedemos\ee_ece_table.m
ee_ece_table.m
13
14 N = 8/2; % Number of pole pairs
15
16 %B_PhaseImp 3
17 % PhaseA 1.0000000000e-003 1.0
18 % PhaseB 1.0000000000e-003 1.0
19 % PhaseC 1.0000000000e-003 1.0
20 %E_PhaseImp
21
22 %B_Sweepings
23 idVec = [-300 -270 -240 -210 -180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180 210 240 270 300];
24 iqVec = [-300 -270 -240 -210 -180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180 210 240 270 300];
25 angleVec = [0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31];
26 %E_Sweepings
27
28 %B_OutputMatrix DQ0
29 data = [
30 % index fluxD
31 0 -9.2992778243e-002 -3.02

```

```

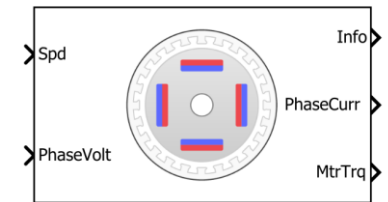
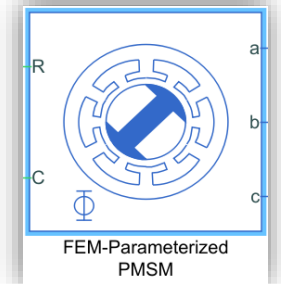
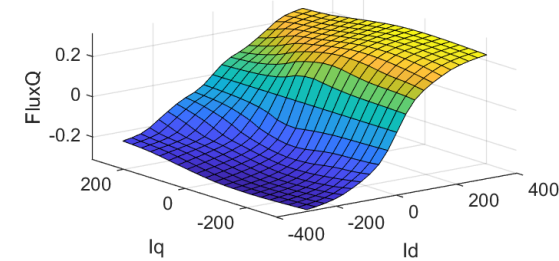
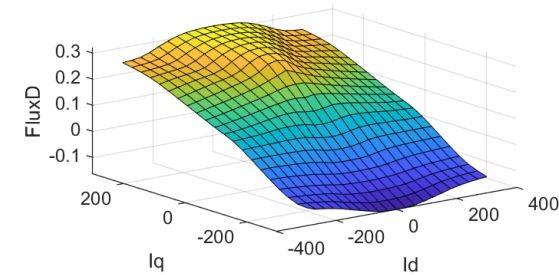
*****
* Copyright 2017-2019 ANSYS, Inc.
* ANSYS and all other ANSYS, Inc.
* of ANSYS, Inc. or its subsidiary
* is reproduced with permission of
*****
B_BasicData
  Version 1.0
  Poles 8
E_BasicData

B_PhaseImp 3
  PhaseA 1.0000000000e-003 1
  PhaseB 1.0000000000e-003 1
  PhaseC 1.0000000000e-003 1
E_PhaseImp

B_Sweepings
  Id_Iq (21: -300 -270 -240 -210 -180 -150 -120
        (21: -300 -270 -240 -210 -180 -150 -120
  Rotate (31: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14
E_Sweepings

B_OutputMatrix DQ0

```



Parameterizing the motor models

PMSM Parameterization from Datasheet

1. Pick design and calculate efficiency at rated load (see code)
2. Copy to the clipboard
3. Paste into the model

Two test harnesses that add confidence that a PMSM is correctly parameterized from a datasheet. It also calculates motor efficiency at

[Open Model](#)

From datasheet

Block Parameterization Manager: BLDC

SELECT FORMAT

Apply all Reset all Manufacturer: All

PARAMETERIZE

Part number	Manufa		N*m
BLY171D_24V_1400	Anaheim	Faulhaber	67.4943
BLY171D_24V_2800	Anaheim	Fulling_Motor	86.0097
BLY171D_24V_6000	Anaheim	Maxon	39.2481
BLY171S_15V_8000	Anaheim	Nanotec	30.7601
BLY172D_24V_4000	Anaheim_Automation	Transmotec	124.3186
BLY172S_24V_2000	Anaheim_Automation		223.2015
BLY172S_24V_4000	Anaheim_Automation		143.5967
BLY173D_24V_4000	Anaheim_Automation		186.1566
BLY174D_24V_12000	Anaheim_Automation		75.1208
BLY174S_24V_12000	Anaheim_Automation		75.1208

From Pre-parametrization

Lumped Parameters

Estimate Motor Parameters Using Motor Control Blockset Parameter Estimation Tool

Motor Control Blockset™ provides a parameter estimation tool that estimates the motor parameters accurately. Use the estimated motor parameters to simulate the motor model and design the control system. Therefore, the simulation response with the estimated parameters for the motor model is close to the behavior of the motor under test.

The parameter estimation tool determines these motor parameters for a Permanent Magnet Synchronous Motor:

Motor parameters	Units
Phase resistance (R_s)	Ohm
d and q axis inductances (L_d and L_q)	Henry
Back-EMF constant (K_e)	Vpk_LL/krpm (where Vpk_LL is the peak voltage line-to-line measurement)
Motor inertia (J)	Kg.m ²
Friction constant (F)	N.m.s

The parameter estimation tool accepts the minimum required inputs, runs tests on the target hardware, and displays the estimated parameters.

From Instrumented tests running on the Hardware

Import IPMSM Flux Linkage Data from ANSYS Maxwell

Import a motor design from ANSYS® Maxwell® into a Simscape™ simulation.

[Open Model](#)

Import IPMSM Flux Linkage Data from Motor-CAD

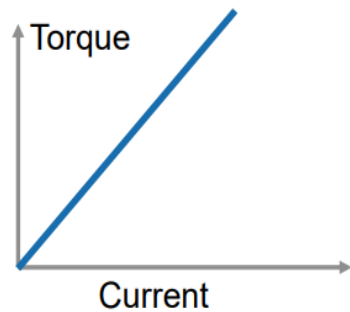
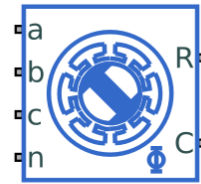
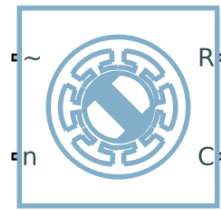
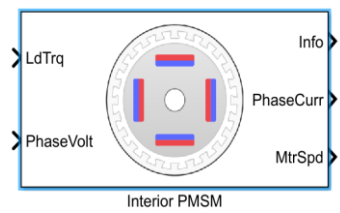
Import a motor design from Motor-CAD into a Simscape™ simulation.

[Open Model](#)

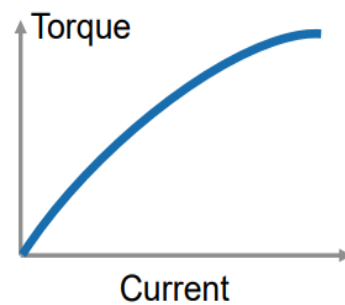
From FEA tools such as ANSYS Maxwell, JMAG, Motor-CAD

Saturation + Spatial Harmonics

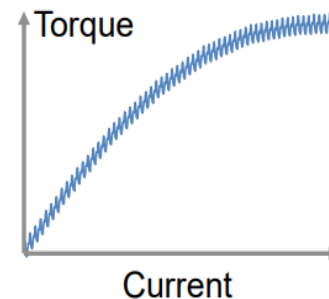
Right model fidelity for Motor & Inverter Model



Lumped Parameter

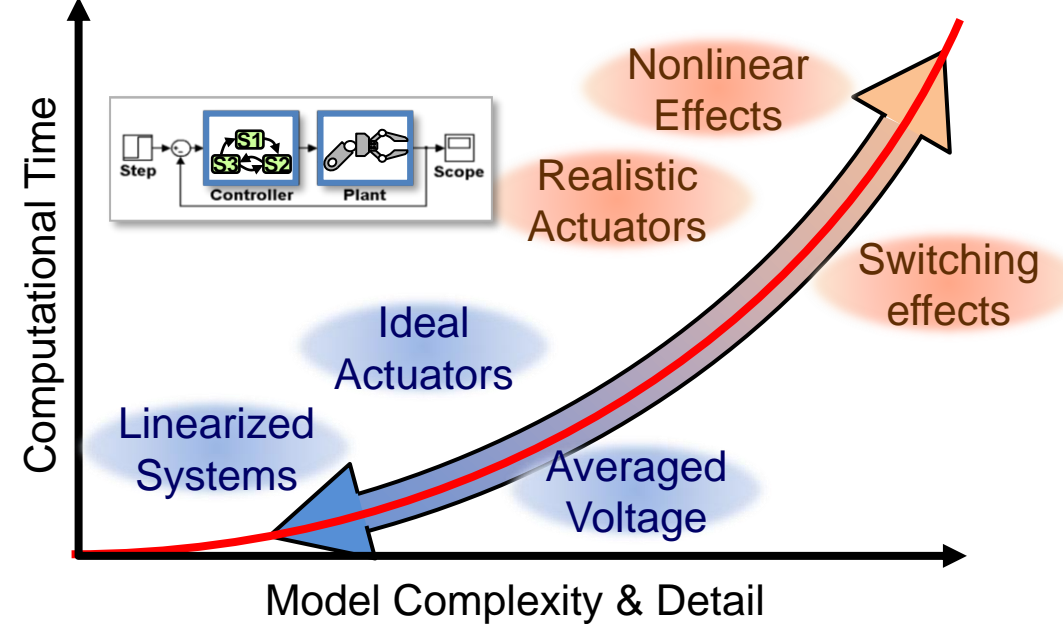


Saturation



Saturation + Spatial Harmonics

Computational Time vs. Model Complexity



Configure your model to balance simulation speed and model fidelity.

Motor Constraint Curves and Characteristics

- Display Motor Constraints such as
 - MTPA curve (Maximum Torque per Ampere)
 - MTPV curve (Maximum Torque per Voltage)
 - Voltage Limit curve
 - Current Limit

- Exploration Motor Characteristics

```

inverter = mcb_SetInverterParameters('BoostXL-DRV8305');
pmsm = mcb_SetPMSMMotorParameters('BLY171D');
w_rpm= 500 ;
T_load= 0 ;
pmsm.Rs= 0.05 ;
inverter.V_dc= 24 ;
pmsm.Lq=pmsm.Ld* 1 ;
pmsm.I_rated= 1.5 ;
mcbPMSMCharacteristics(pmsm,inverter,'speed',w_rpm,'torque',T_load);
xlim([-10.9 6.0])

```

```

Run this section
inverter.V_dc= 24 ;
pmsm.p=4;
pmsm.Rs= 0.45 ;
pmsm.Ld=1e-3;
pmsm.Lq=pmsm.Ld* 1.4 ;
pmsm.FluxPM=5.2e-3;
pmsm.B=1.16e-5;
pmsm.I_rated= 2 ;
w_rpm= 7000 ;
T_load= 0.01 ;
FWCMethod= 'vcimt' ;
mcbPMSMCharacteristics(pmsm,inverter,'speed',w_rpm,'torque',T_load,'FWCMethod'

```

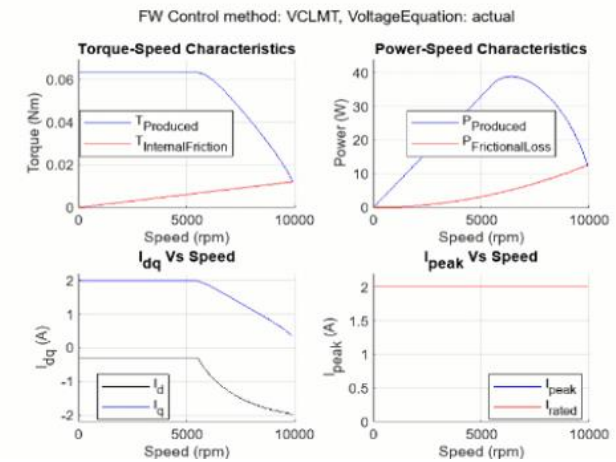
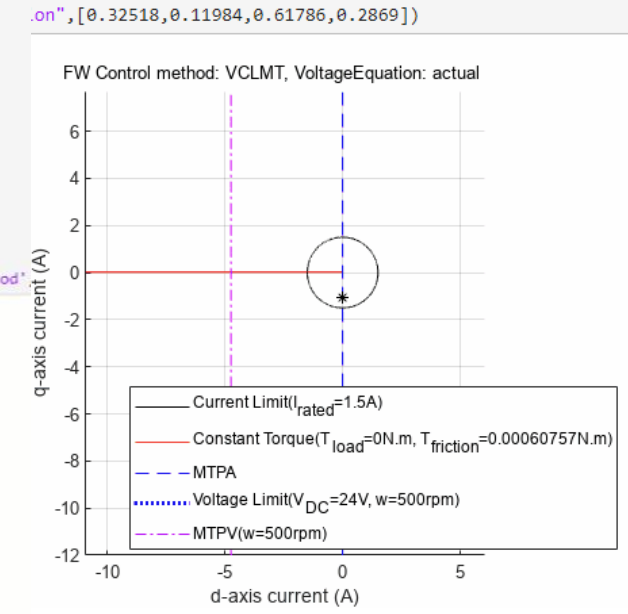


Figure : Dependency of curves and characteristics on parameters changes

Control loop gain tuning

```

%% Set PWM Switching frequency
PWM_frequency = 20e3; %Hz // converter s/w freq
T_pwm = 1/PWM_frequency; %s // PWM switching time period

%% Set Sample Times
Ts = T_pwm; %sec // sample time for controller
Ts_simulink = T_pwm/2; %sec // simulation time step for model simulation
Ts_motor = T_pwm/2; %sec // simulation sample time
Ts_inverter = T_pwm/2; %sec // simulation time step for average value inverter
Ts_speed = 10*Ts; %sec // sample time for speed controller

%% Set data type for controller & code-gen
% dataType = fixdt(1,32,17); % Fixed point code-generation
dataType = 'single'; % Floating point code-generation

%% System Parameters // Hardware parameters
pmsm = mcb_SetPMSMMotorParameters('BLY171D');

%% Parameters below are not mandatory for offset computation
inverter = mcb_SetInverterParameters('DRV8312-C2-KIT');
inverter.ADCOffsetCalibEnable = 1; % Enable: 1, Disable:0
target = mcb_SetProcessorDetails('F28069M', PWM_frequency);

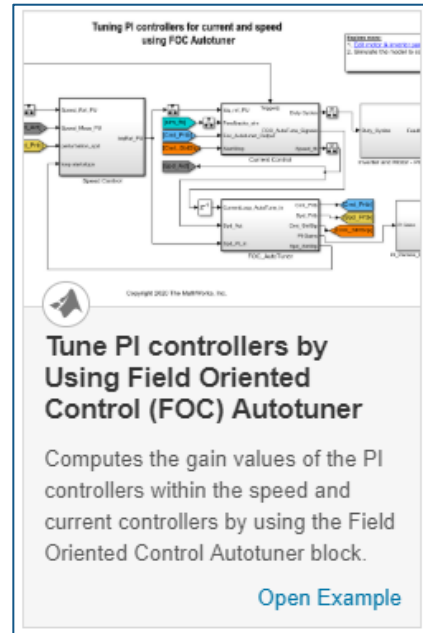
%% Derive Characteristics
pmsm.N_base = mcb_getBaseSpeed(pmsm, inverter); %rpm // Base speed of motor at given Vdc
% mcb_getCharacteristics(pmsm, inverter);

%% PU System details // Set base values for pu conversion
PU_System = mcb_SetPUSystem(pmsm, inverter);

%% Controller design // Get ballpark values!
PI_params = mcb.internal.SetControllerParameters(pmsm, inverter, PU_System, T_pwm, Ts, Ts_speed);
    
```

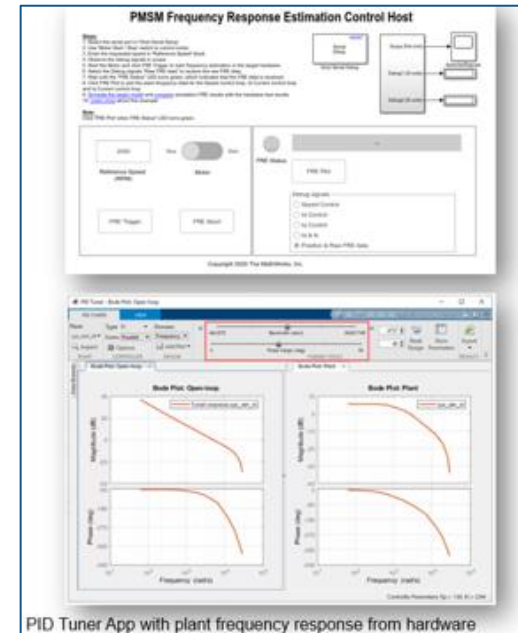
Empirical Computation

Motor Control Blockset

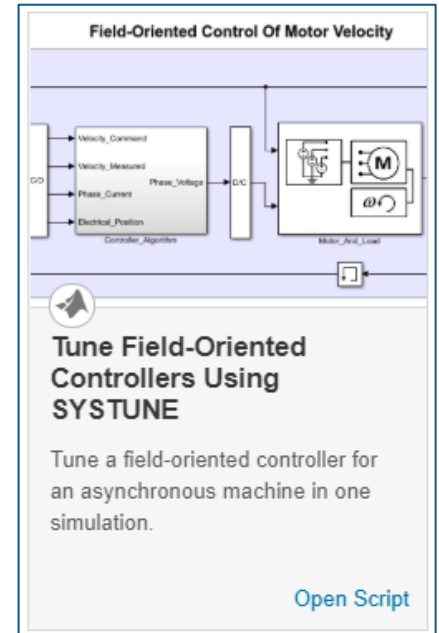


FOC Autotuner

Motor Control Blockset and Simulink Control Design



Online frequency estimation and PID tuner app

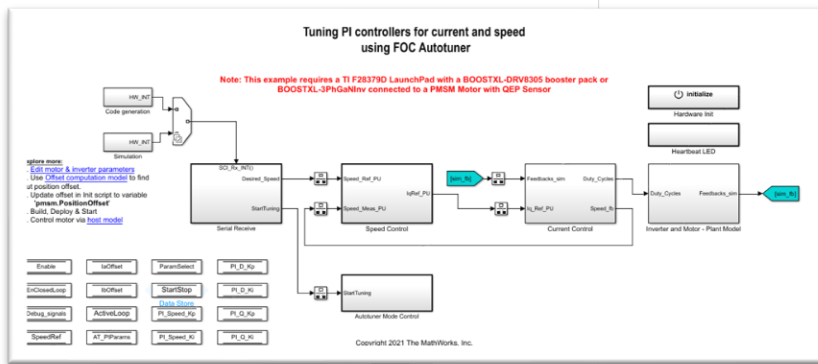
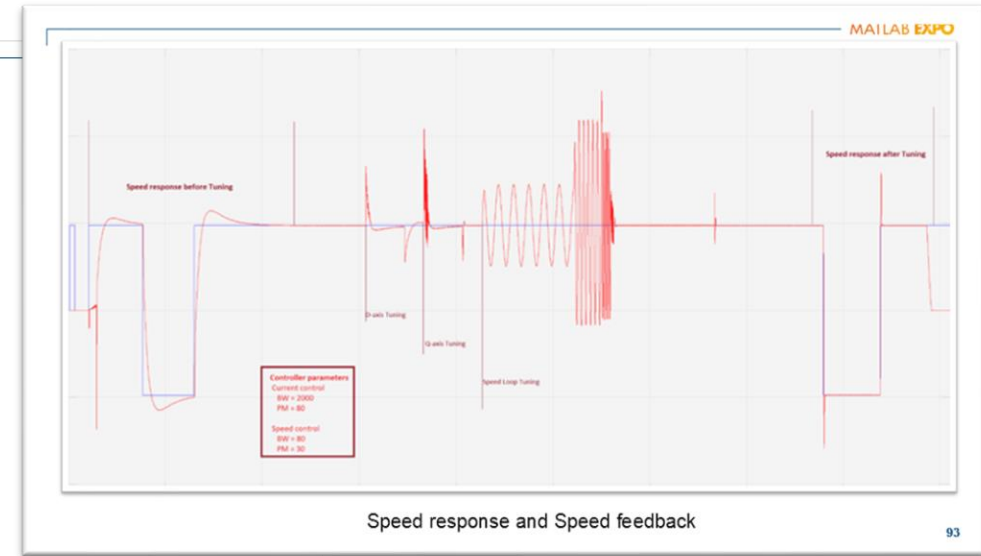


Classical Control Theory

Simulink Control Design

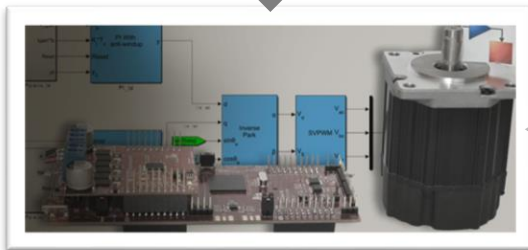
Control loop gain tuning with FOC Autotuner

Target model deployed to the hardware

Speed response and Speed feedback

Deployed in hardware

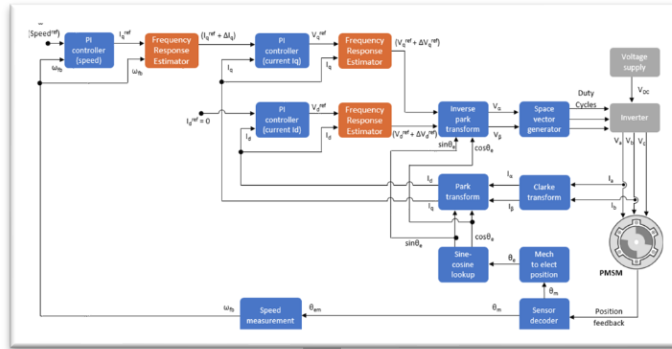


Motor control kit with evaluation board

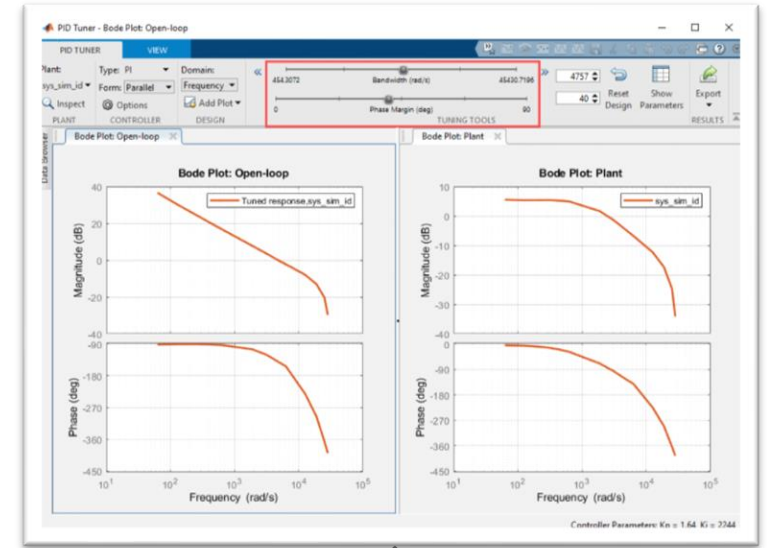
Host model reads the tuned gain values from hardware

Control loop gain tuning Online Frequency Estimation

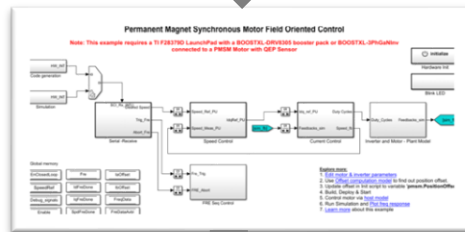
Online plant frequency response



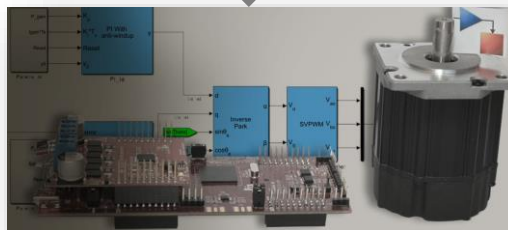
Input frequency response data to PID Tuner App



Target model deploy in target

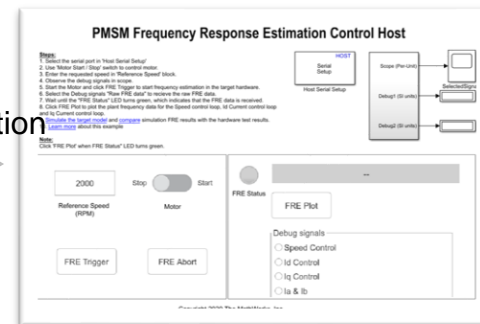


Deployed in hardware

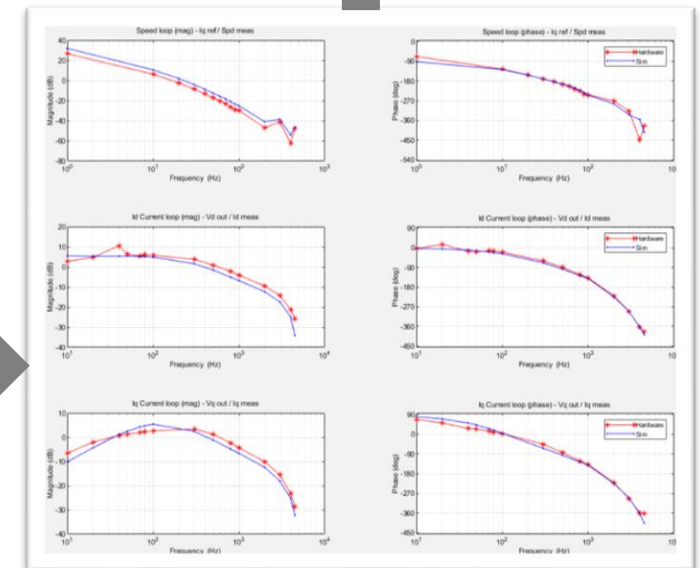


Motor control kit with evaluation board

Serial communication



Host model reads the frequency spectrum data from hardware



Compare frequency response data in Sim and hardware testing

Code Generation and Deployment on embedded device

Hardware Support Package for Driver code




Embedded Coder Support Package for Texas Instruments C2000 Processors by MathWorks Embedded Coder Team **STAFF**

Generate code optimized for C2000 MCU.


Embedded Coder® Support Package for Texas Instruments C2000™ Processors enables you to run Simulink® models on TI C2000 MCUs. Embedded Coder automatically generates C code for your algorithms and




Field-Oriented Control of PMSM Using NXP™ S32K144 Kit version 1.0 by Shivaprasad Narayan **STAFF**

The workflow demonstrates Field Oriented Control of a Permanent Magnet Synchronous Motor using NXP™ MCSPT1AK144: S32K144 Development Kit

FOC-of-PMSMField-Oriented Control of Permanent Magnet Synchronous Motor Using NXP™ S32K144 Development kitThis example implements a motor control system using the NXP™ MCSPT1AK144 hardware. The

Demo for Motor Control Deployment on Microchip Controllers version 1.0.0 by Brian McKay **STAFF**

Demo used in MathWorks-Microchip joint webinar: Deploying Motor Control Algorithms on Microchip dsPIC, PIC32, and SAM Controllers.

which includes a dsPIC33E Digital Signal Controller. The demo also require you to download and install the free add-on MPLAB Device Blocks for Simulink: dsPIC, PIC32, and SAM MCU's. View the webinar for a



Hand-written Driver code for peripherals




Algorithm-Export Workflows for Custom Hardware

Enables you to use any custom motor-control hardware (hardware not used in the Motor Control Blockset™ examples) to run a three-

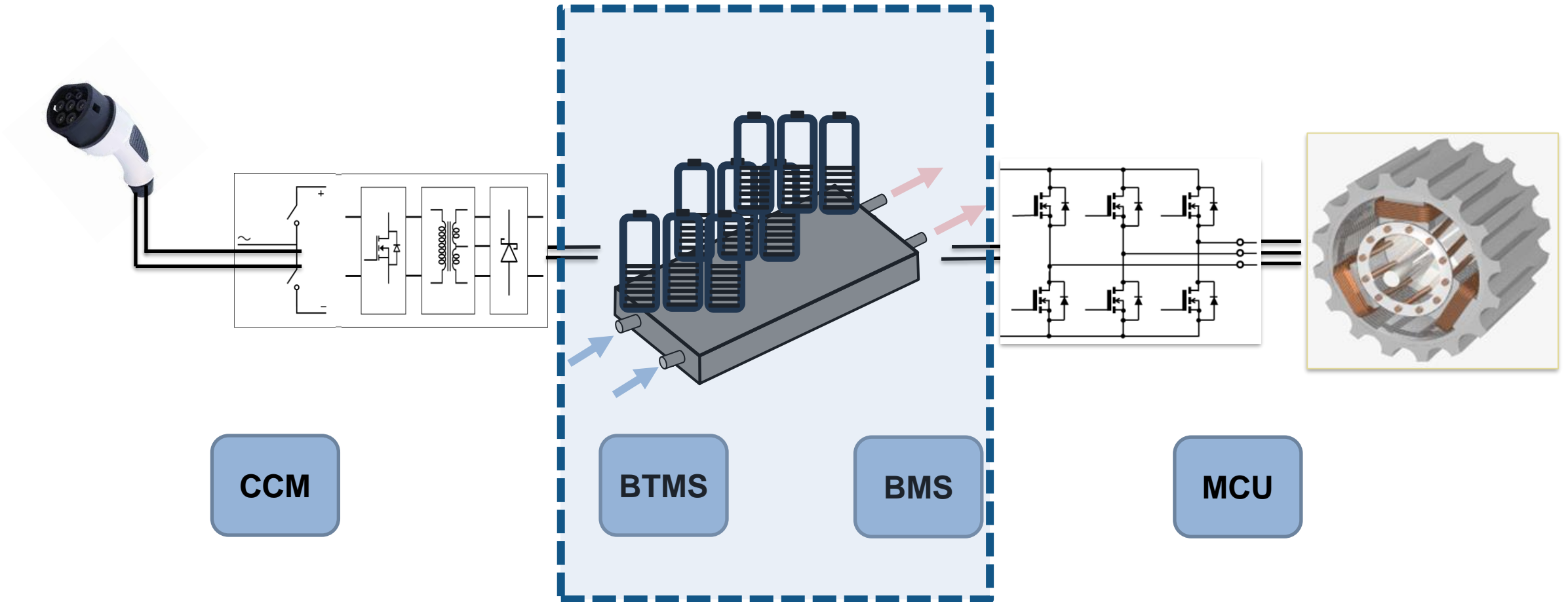
[Open Example](#)

Summary



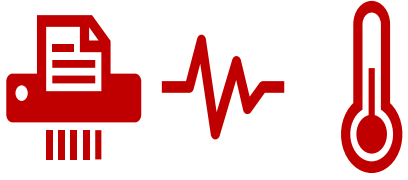
- ✓ Discussed various methods to parameterize the motor and achieve different fidelity levels
- ✓ Utility to display constraints curves as a function of motor parameters
- ✓ Discussed systematic approaches to tune loop gains
- ✓ Deploy to the hardware and validate

Electric Vehicle System

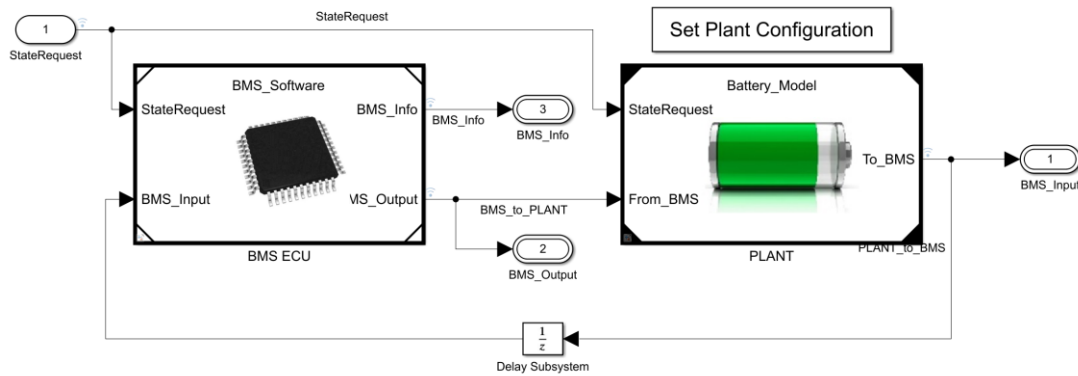


Challenges in developing a BMS

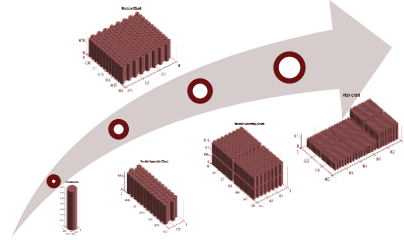
Collaboration Gap



Multi-Domain Modeling Environment



System Analysis



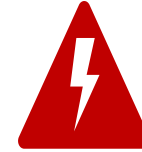
Scaling up from cell to pack using automation

Long Iteration Cycles

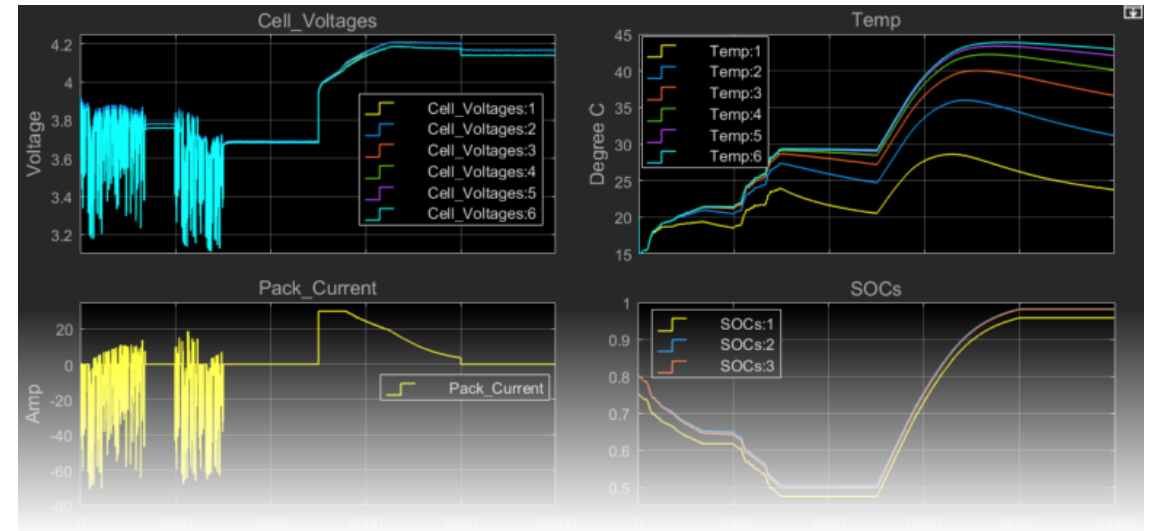


Simulations and Code Generation

Safety Critical System



Model V&V and Hardware-In-Loop Testing

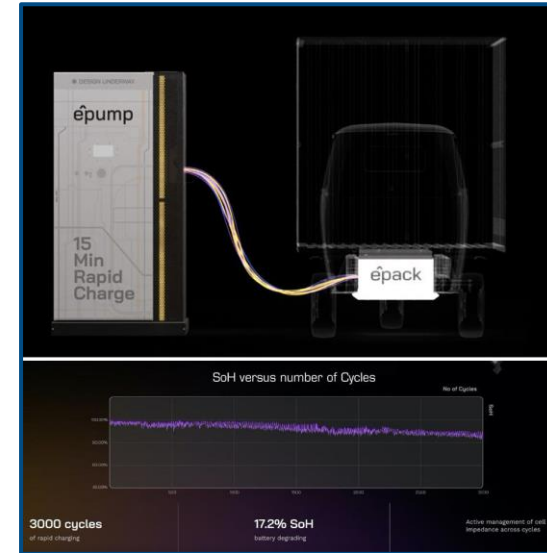


Exponent Energy Develops 15-Minute Fast-Charging Battery System for Electric Vehicles Using Model-Based Design

Using Simulink, Stateflow, and Embedded Coder, Exponent Energy designed a battery system with a fast charger that accounts for state of charge, state of health, and impedance faster and more accurately than existing systems. After 3,000 cycles, its residual capacity was still over 80%

Key Outcomes/Advantages

- Accelerated the development, testing, and verification of the entire solution by at least five times using Model-Based Design
- Accelerated product prototyping through code generation with Embedded Coder and TSP for TI C2000
- Developed an accurate and responsive BMS with improved thermal management, resulting in more than 80% state of health after 3,000 fast-charging cycles

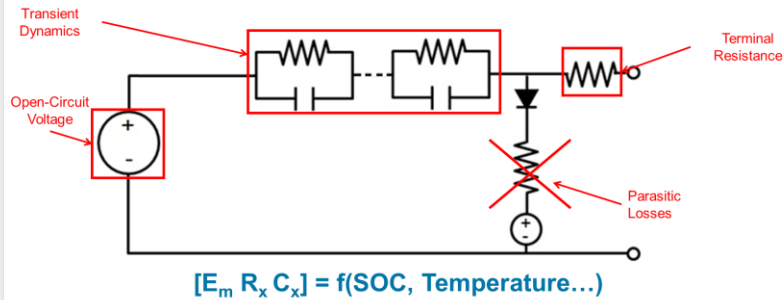


(Above) Exponent Energy's flexible energy stack with the e^pump fast charger and e-pack battery in the vehicle. (Below) A graph measuring battery health across fast charging cycles.

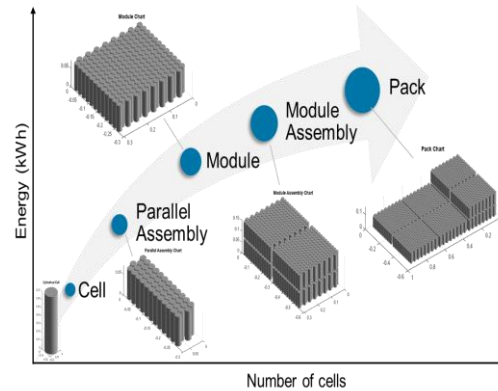
"In just a month, Exponent Energy successfully integrated the battery with the vehicle and conducted multiple charging and discharging tests on the road. We reduced development time significantly using Model-Based Design and MATLAB and Simulink products."
 - Richard Davis, lead system architect, Exponent Energy

Battery Pack and BMS Design Workflow Tasks

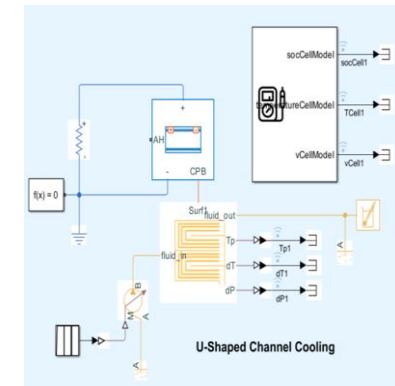
1. Cell Modeling



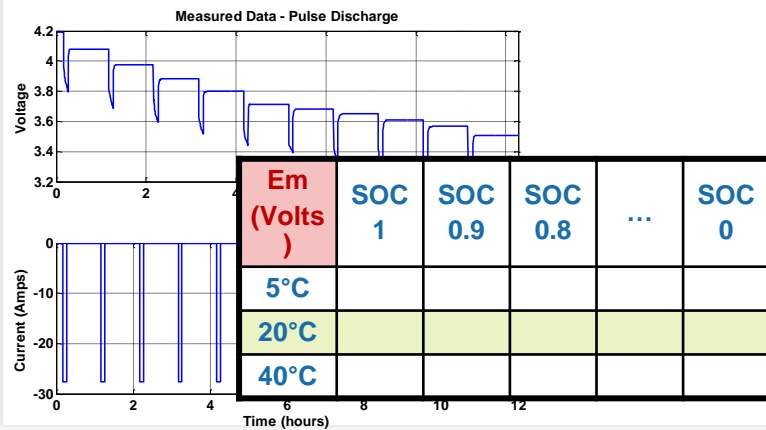
3. Battery Pack Design



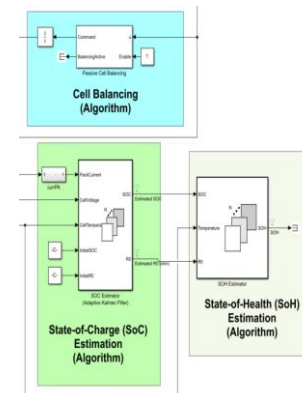
5. Thermal Management System Design



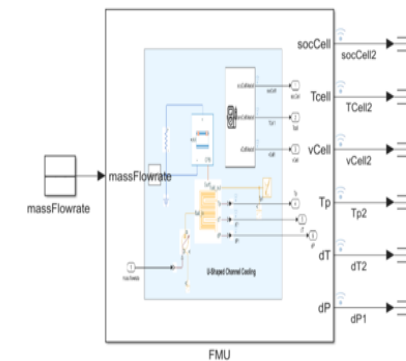
2. Cell Parametrization



4. Battery Management System Design & Deployment



6. Deployment and HIL



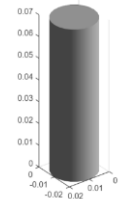
Where do we start?



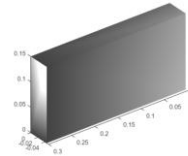
Gain insight into cell behavior and model it

Cell Geometry

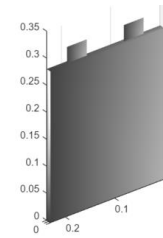
Available geometries



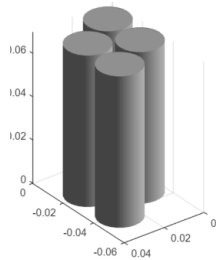
Cylindrical Cell



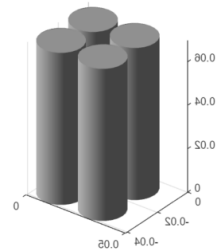
Prismatic Cell



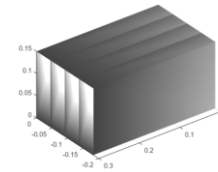
Pouch Cell



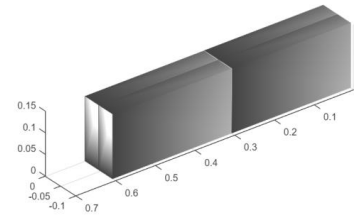
Hexagonal



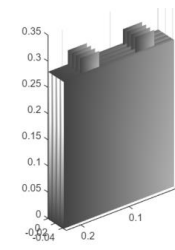
Square



Single Stack



Multiple Stack



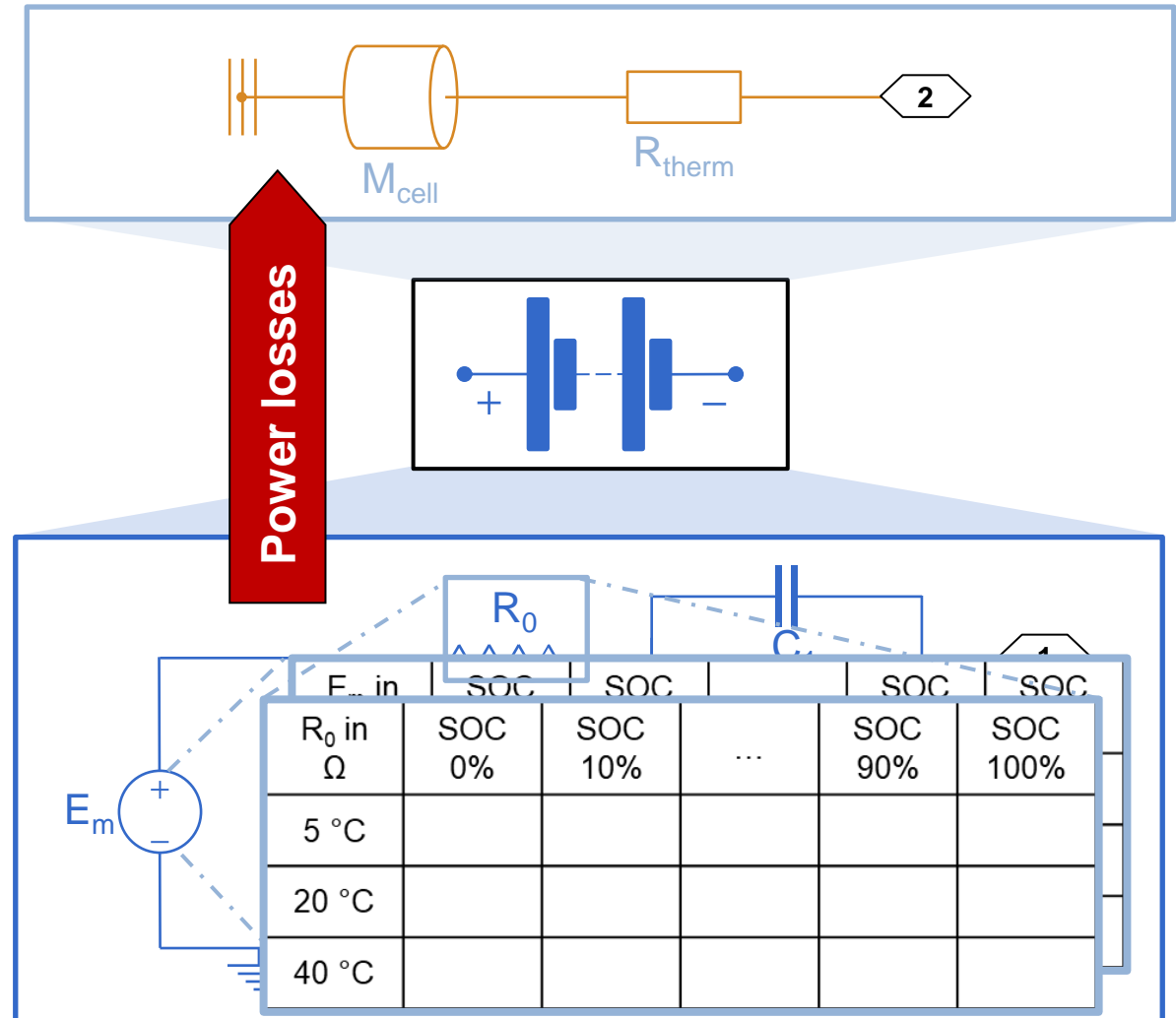
Single Stack

- The cells can be disposed in different topologies, depending on their geometry
- Battery pack volume, mass, and dimensions are scaled automatically as new cells are added

Understanding the Cell Model

Electrical and thermal model

- Electrical model for RC circuit
 - Look-up table based
 - Dependency from SOC and Temperature
- Thermal lumped model
- Thermal resistances that account for different thermal paths
- Power losses calculated from Ohmic losses

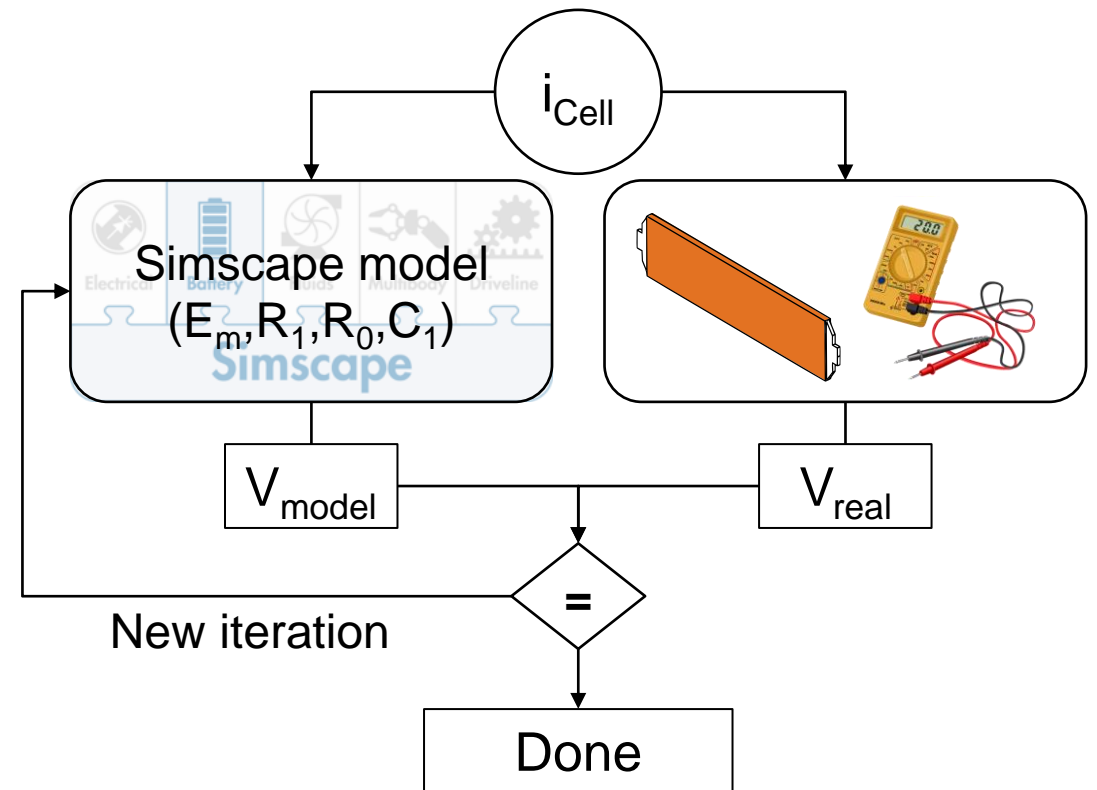
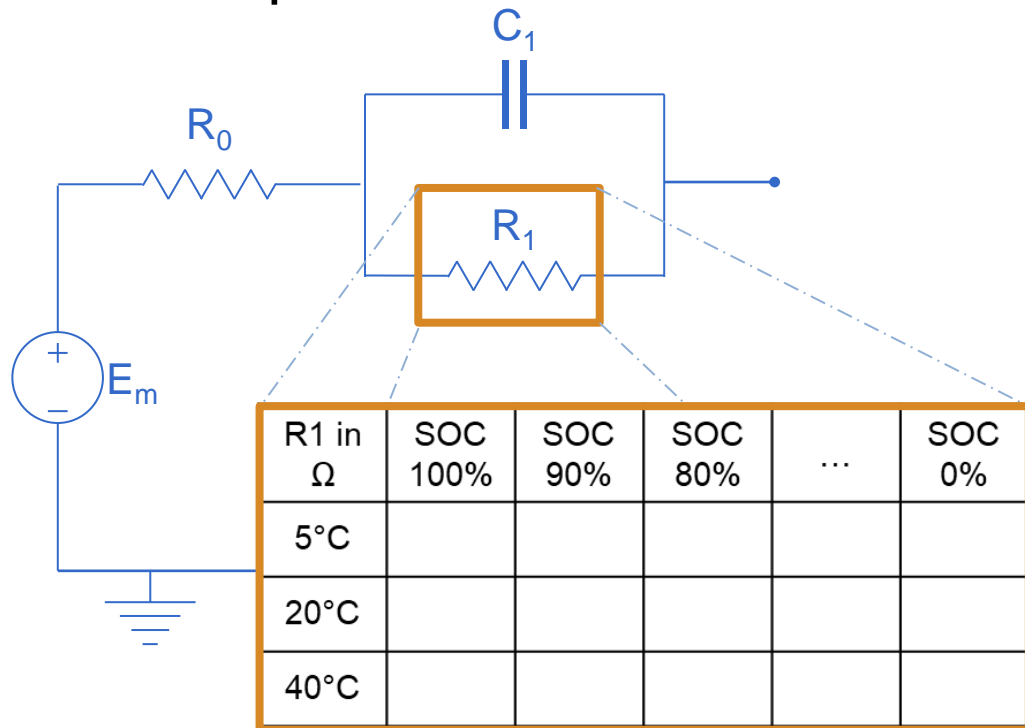


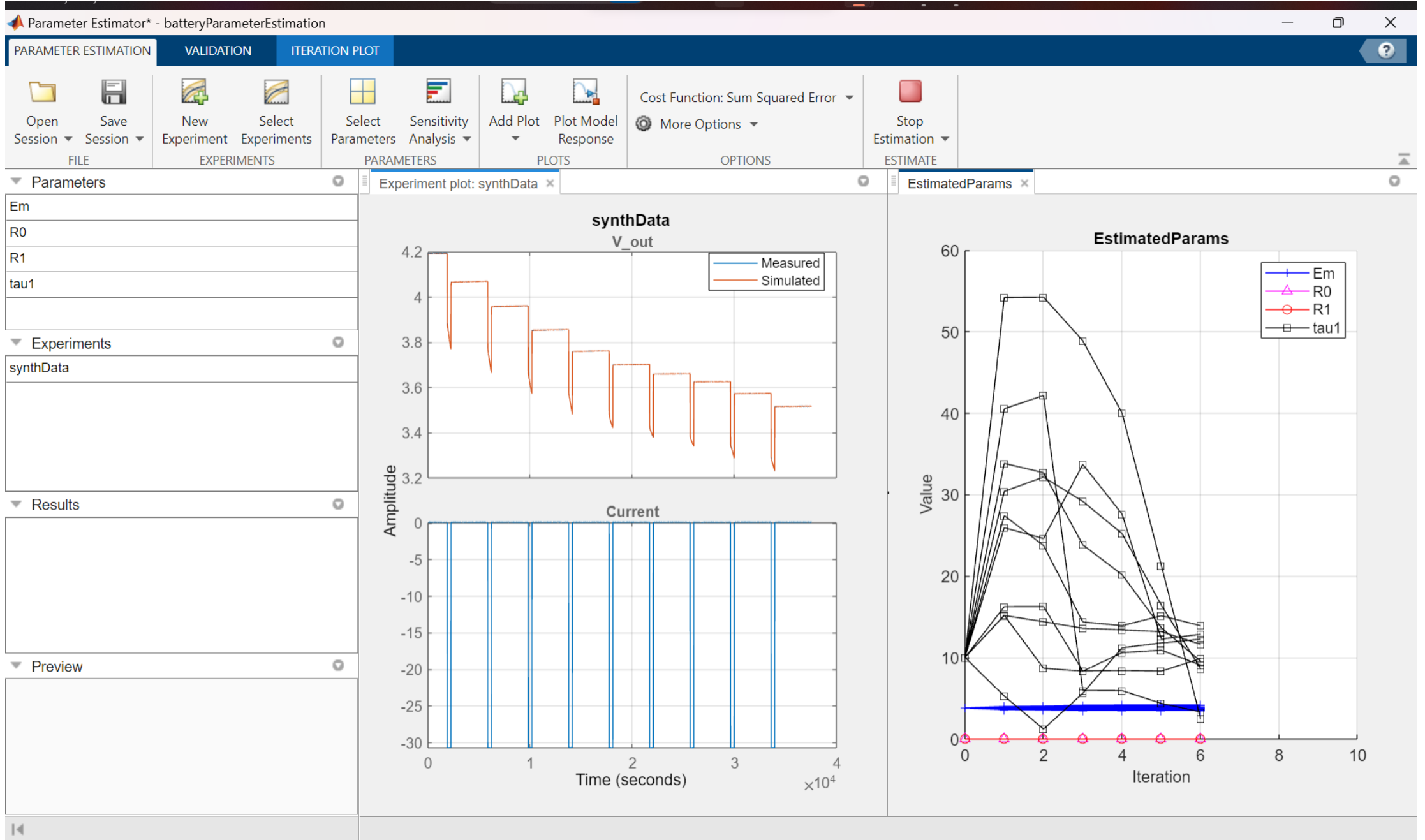
Parametrizing the Cell Model

RC circuit parametrization

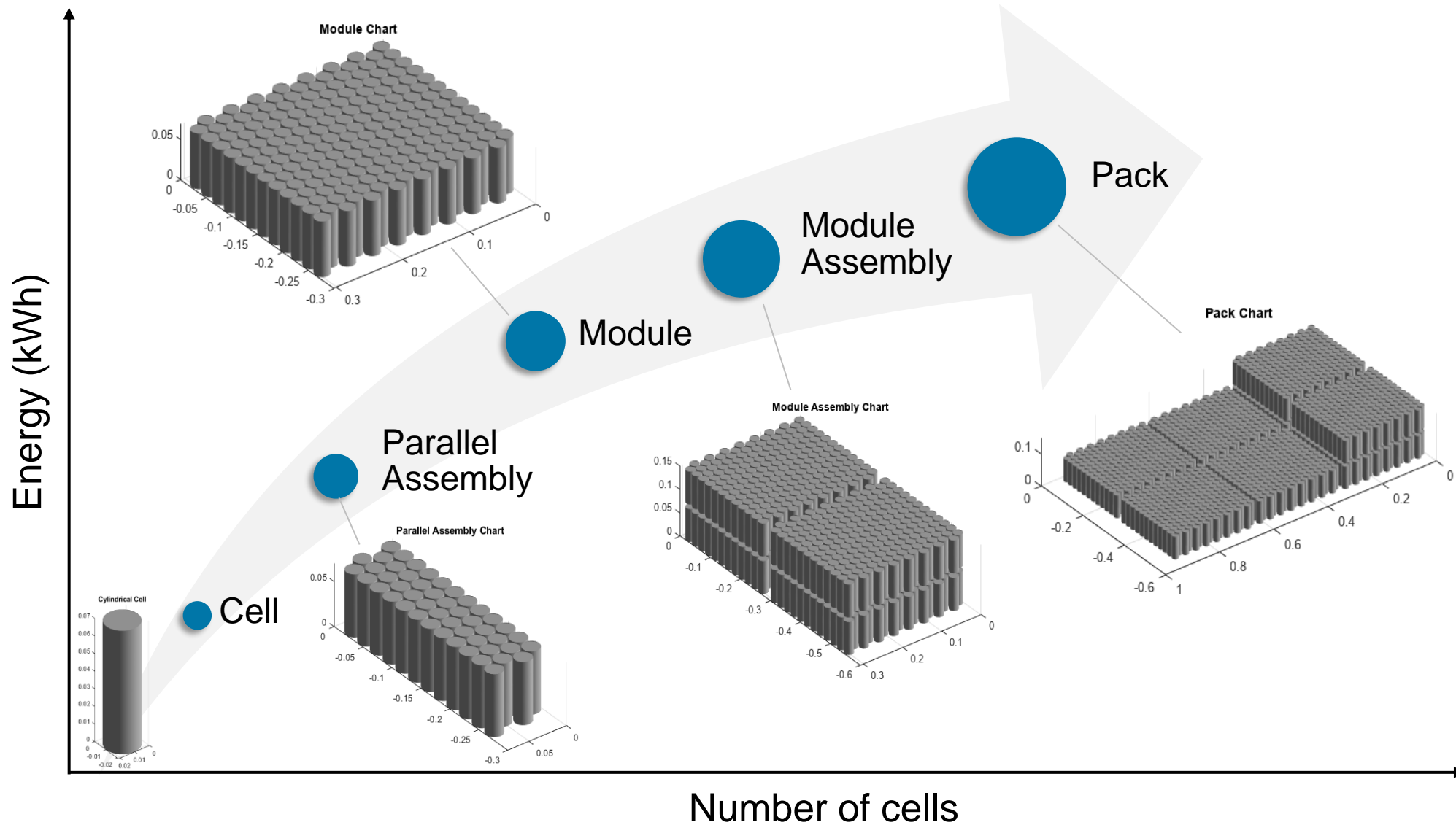
Two possible cases for RC circuit (E_m , R_0 , $R_1 \dots$) parametrization:

- Look-up tables are known
- Look-up tables must be estimated



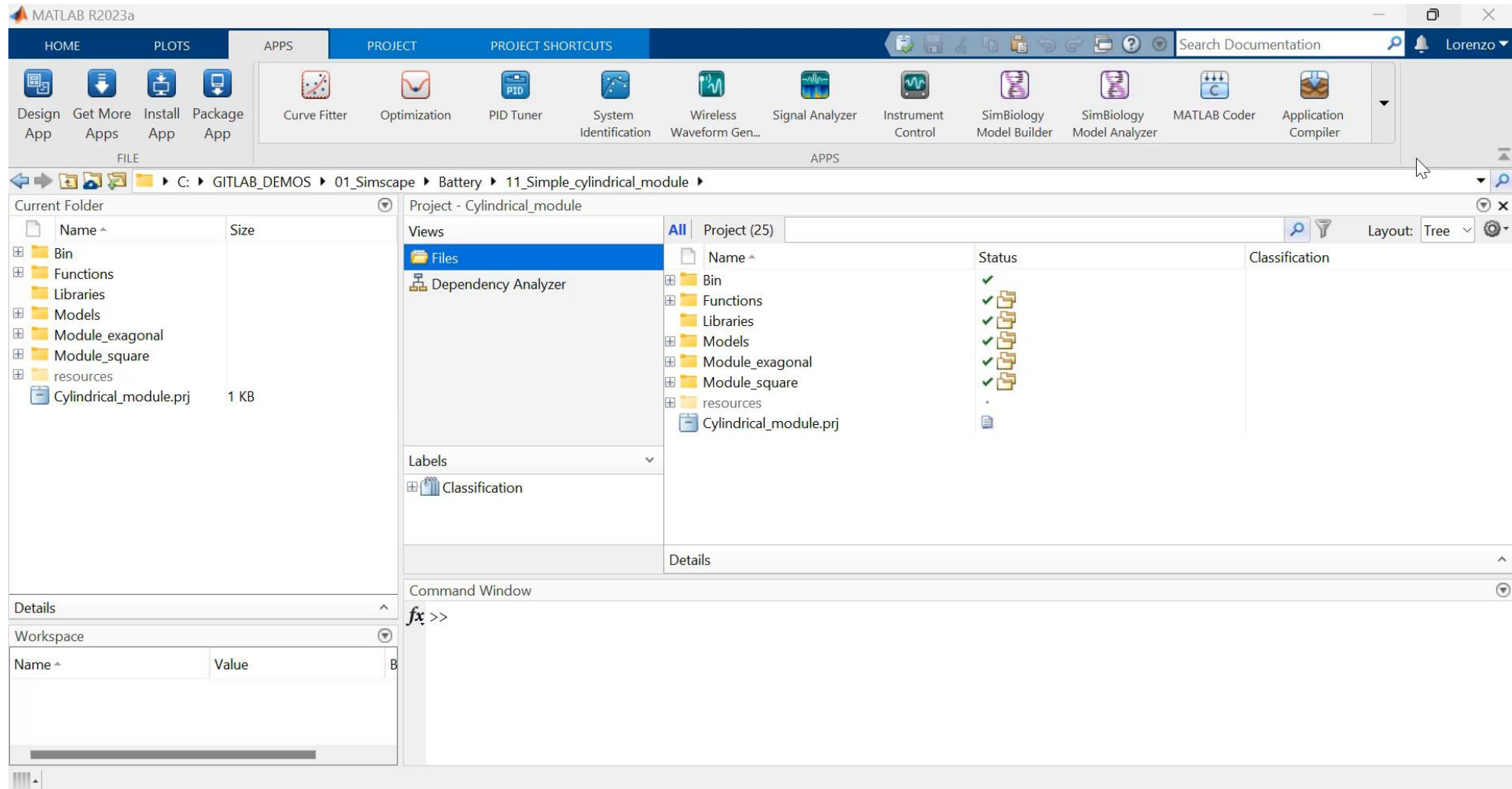


Electric Cell to Battery Pack



Battery Builder

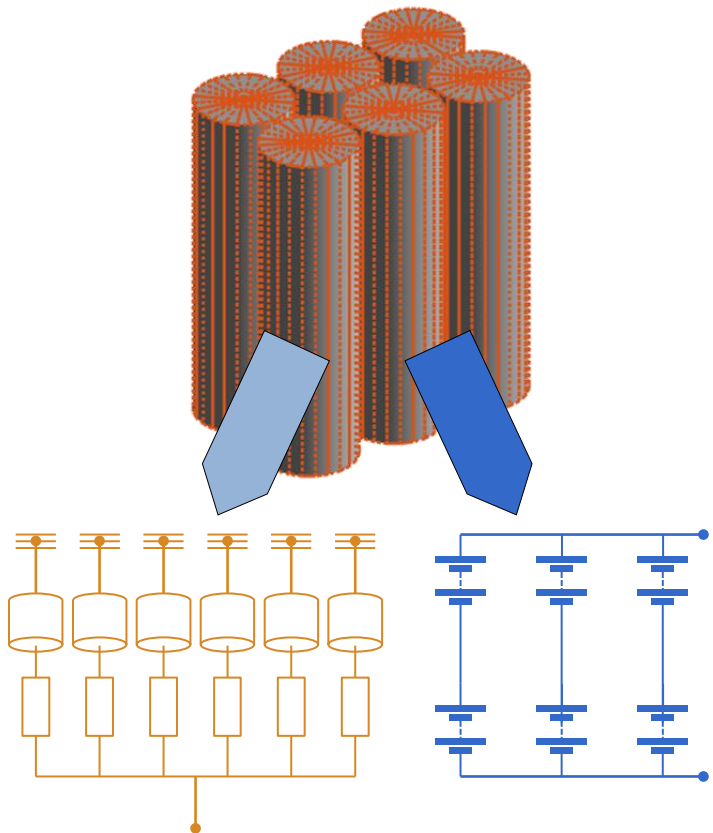
App-based battery pack generation



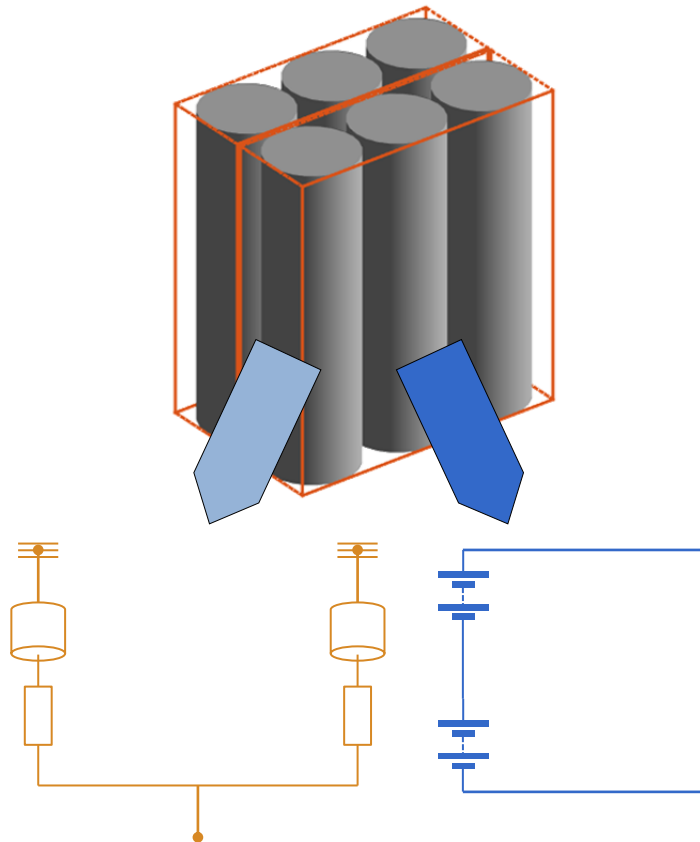
Choose Model Fidelity

Find tradeoff between calculation speed and precision

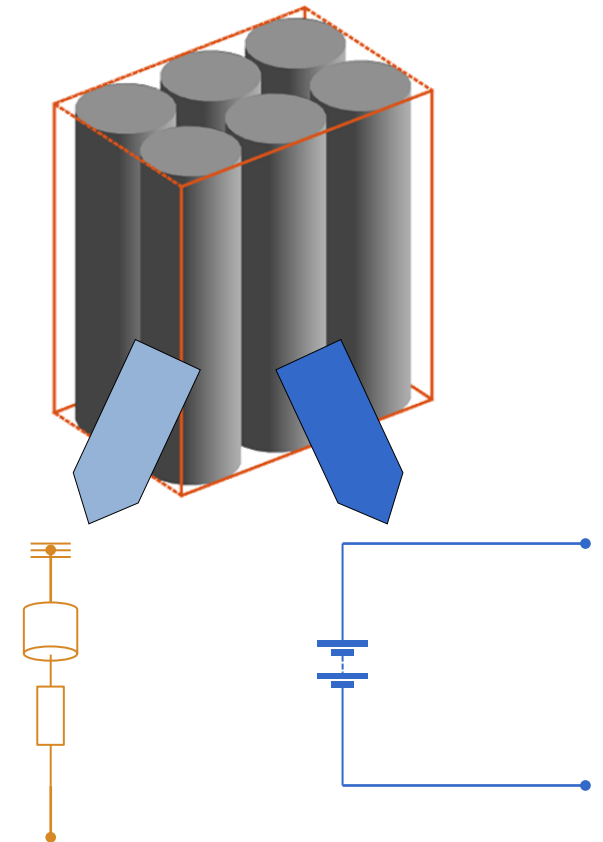
Detailed



Grouped



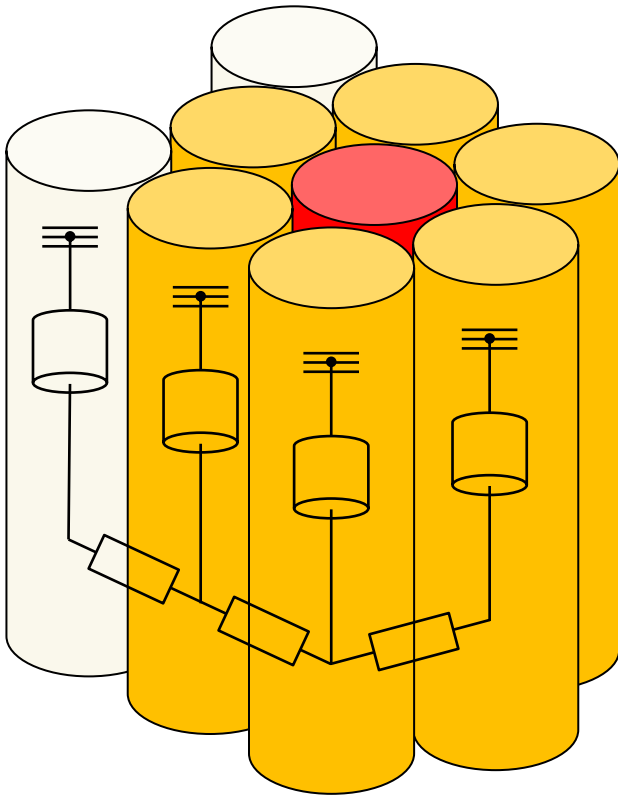
Lumped



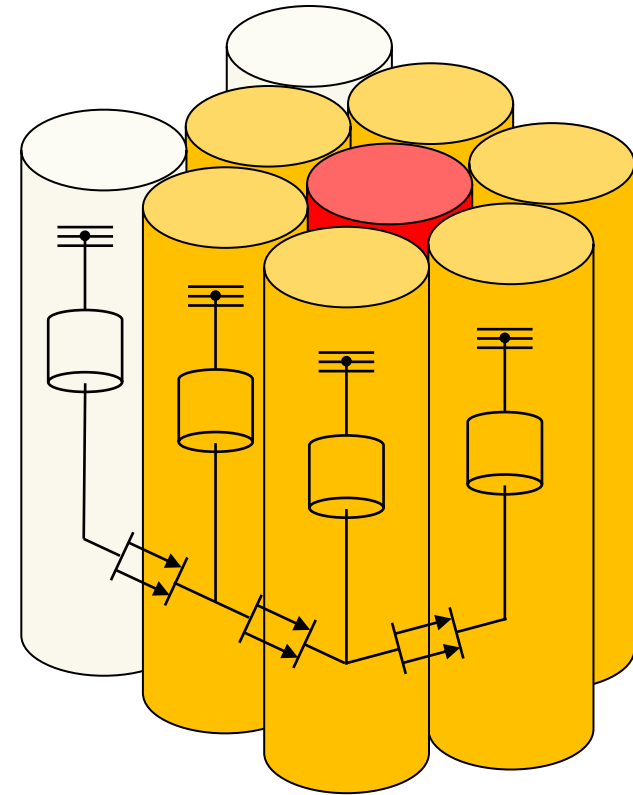
Battery pack: Thermal Paths WITHIN the Pack

Model conduction, convection, and radiation between cells

Inter cell path



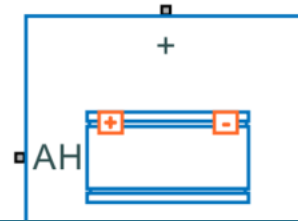
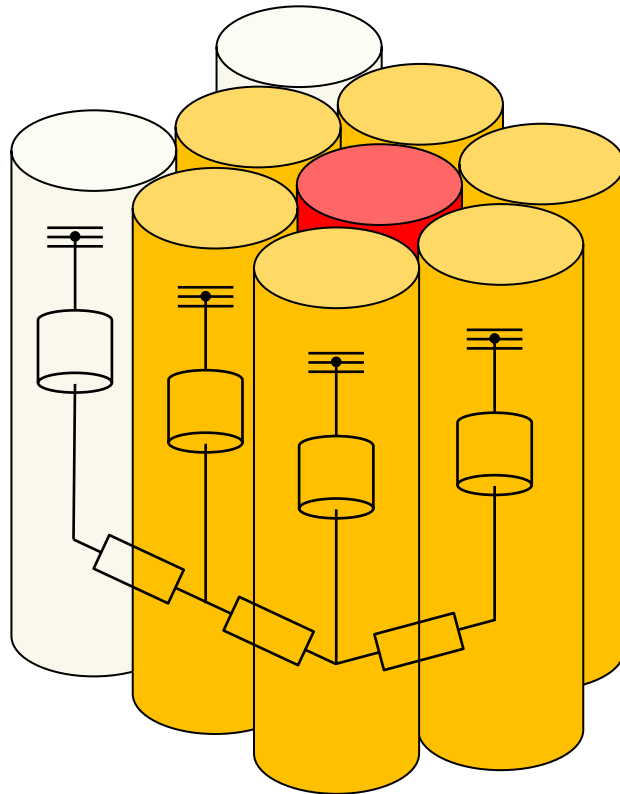
Inter cell radiative path



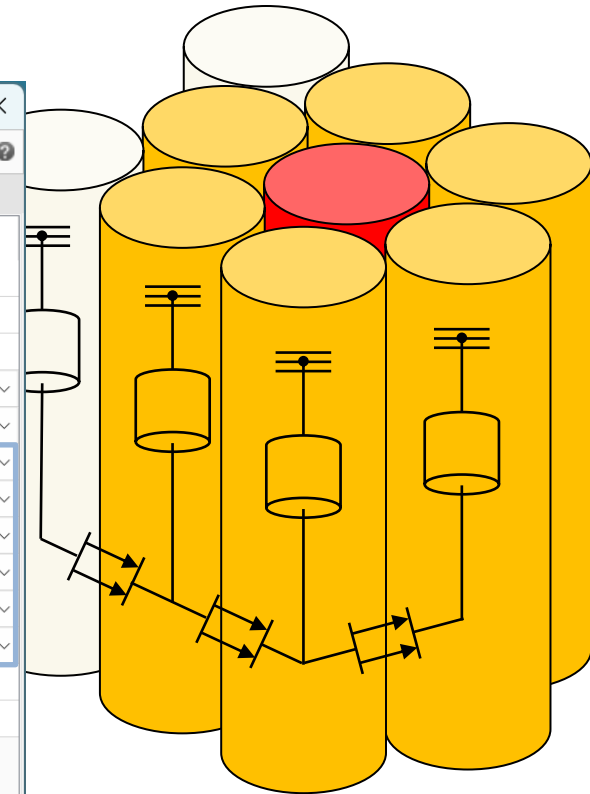
Battery pack: Thermal Paths WITHIN the Pack

Model conduction, convection, and radiation between cells

Inter cell path



Inter cell radiative path



Block Parameters: Module1

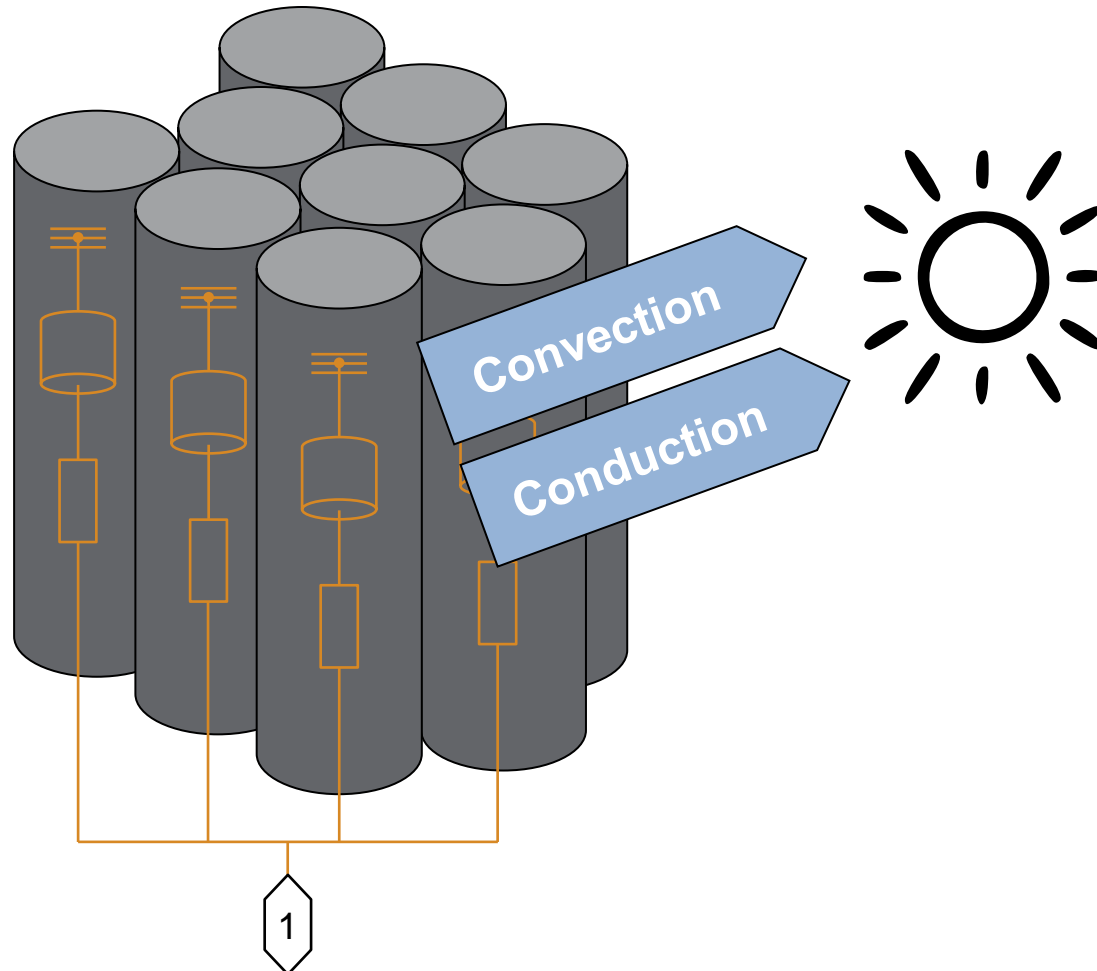
Module1 Auto Apply

NAME	VALUE	
> Main		
> Dynamics		
> Thermal		
> Thermal mass	100	J/K
> Cell level ambient thermal ...	25	K/W
> Inter-cell thermal path resi...	1	K/W
> Inter-cell radiation heat tr...	1e-3	m ²
> Inter-cell radiation heat tr...	1e-6	W/(K ⁴ *m ²)
> Inter-parallel assembly the...	1	K/W
> Inter-parallel assembly are...	1e-3	m ²
> Inter-parallel assembly co...	1e-6	W/(K ⁴ *m ²)
> Initial Targets		
> Nominal Values		

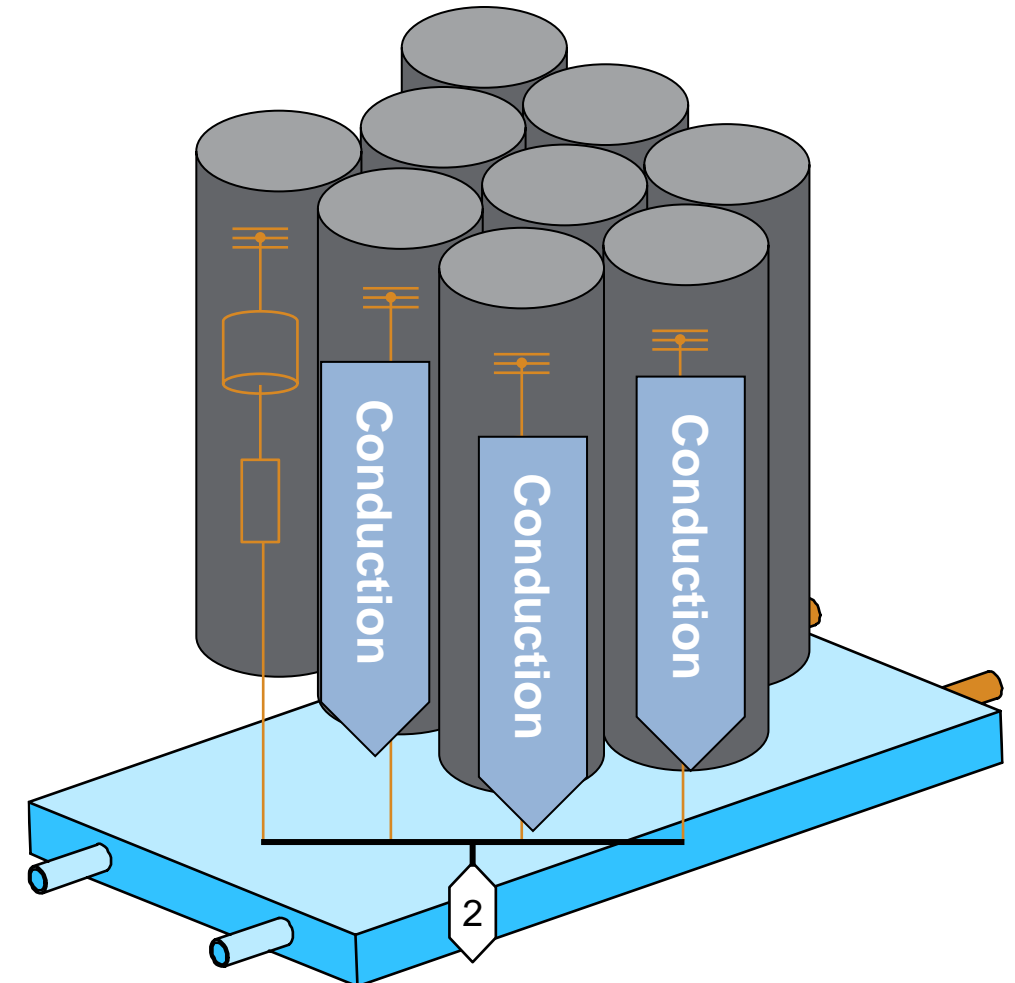
Battery pack: Thermal Paths OUTSIDE of the Pack

Model conduction, convection between pack and environment

Thermal path to ambient



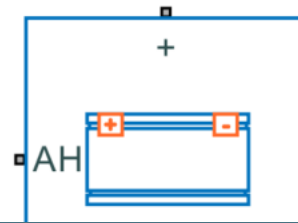
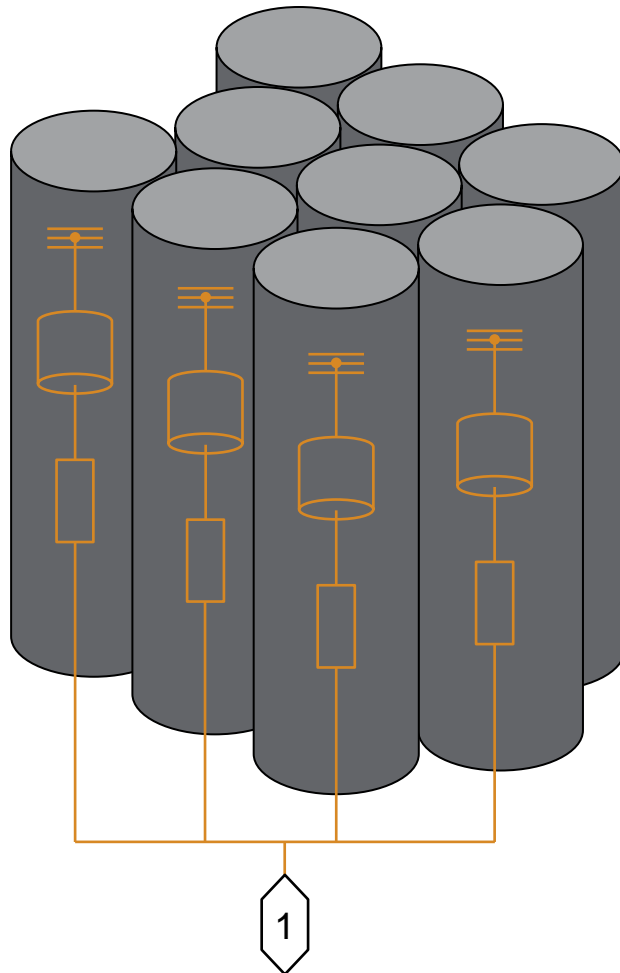
Thermal path to cooling plate



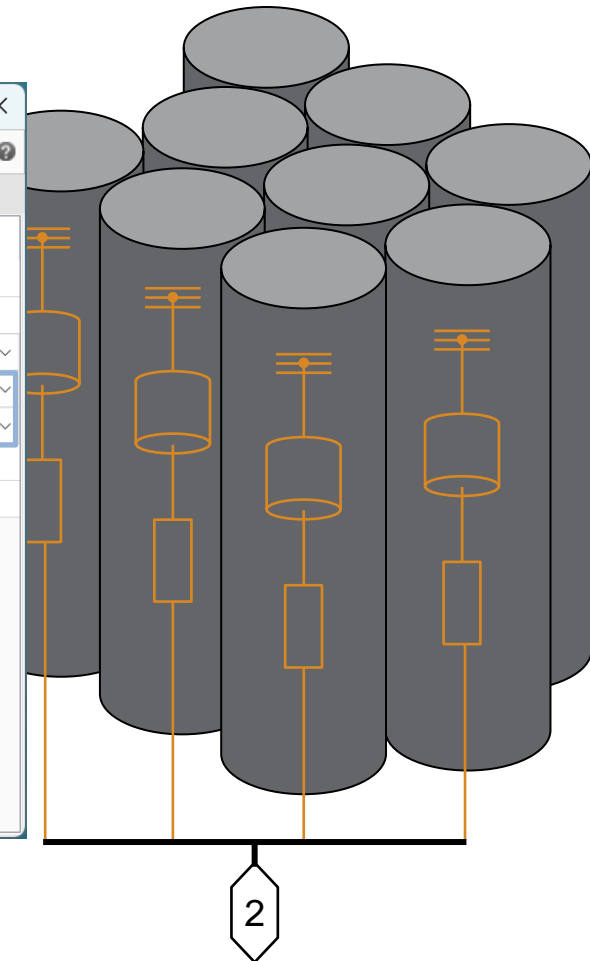
Battery pack: Thermal Paths OUTSIDE of the Pack

Model conduction, convection between pack and environment

Thermal path to ambient



Thermal path to cooling plate



Block Parameters: ModuleType1

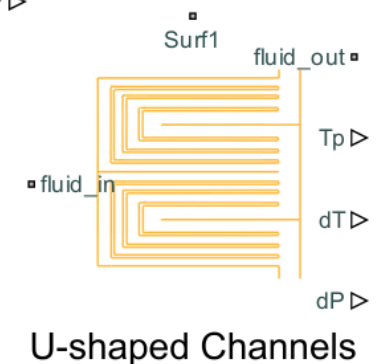
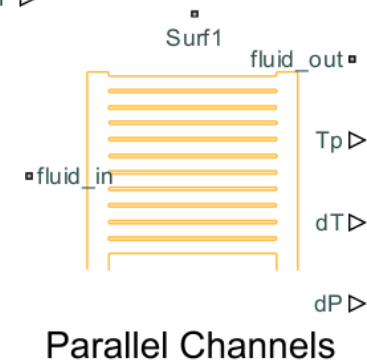
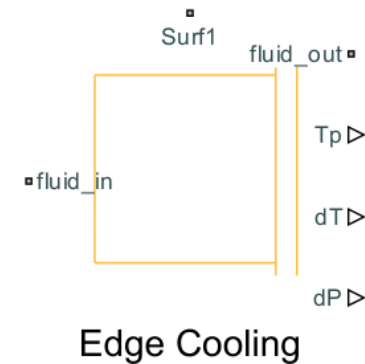
ModuleType1 Auto Apply

NAME	VALUE
> Main	
> Thermal	
> Thermal mass	100 J/K
> Cell level coolant thermal ...	1.2 K/W
> Cell level ambient thermal ...	25 K/W
> Initial Targets	
> Nominal Values	

Battery pack active cooling

Cooling plate topologies

- Model heat transfer between battery, liquid cooling system, and environment
 - Control cell-to-cell temperature variation
 - Tradeoff of pumping costs and cooling efficiency
- Different cooling plate topologies
 - Edge, parallel channel, U-shaped channel
 - Single- and double-sided plates
- Adjust resolution of thermal model
 - Define quantity and placement of nodes



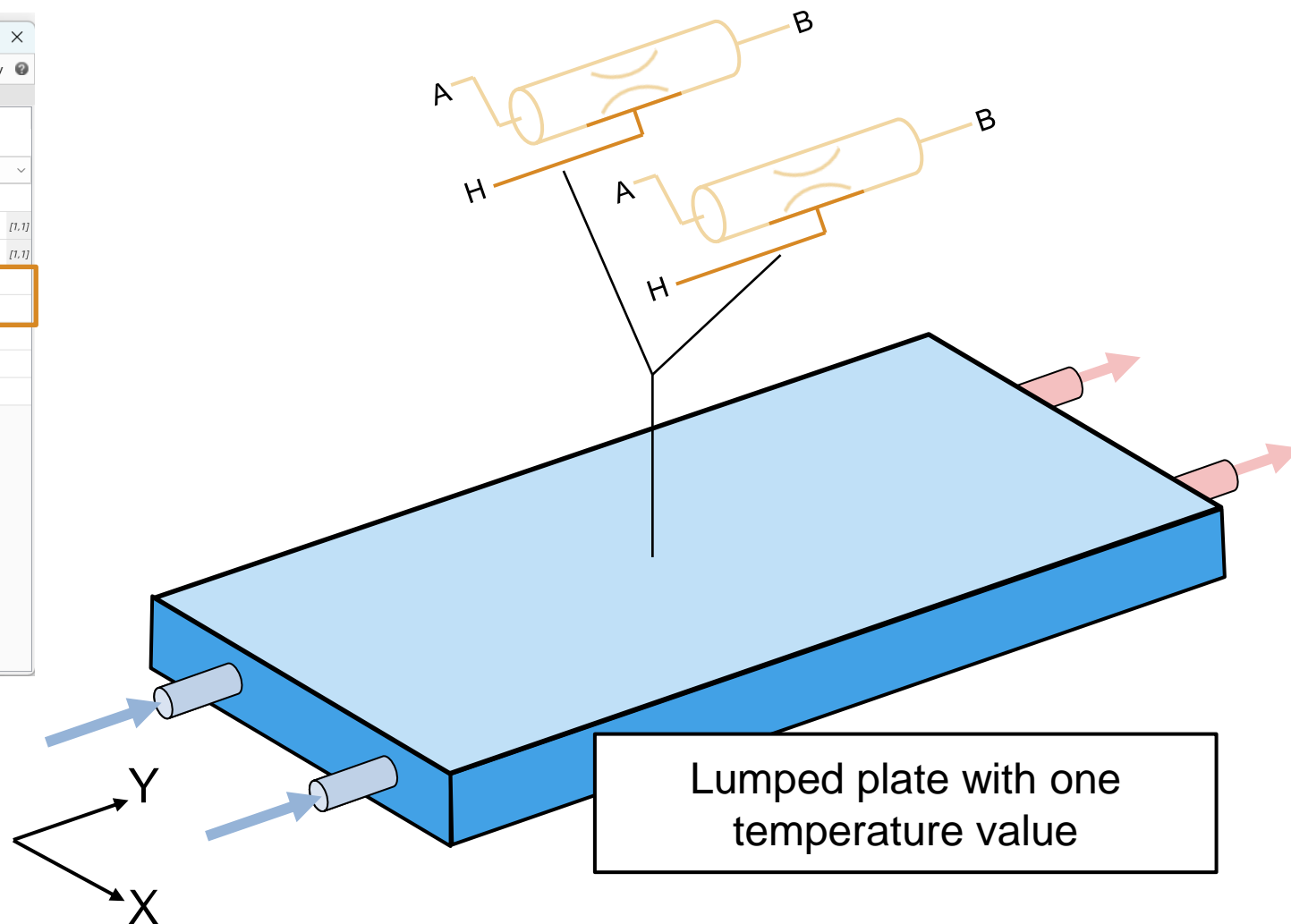
Cooling plate model

Parallel Channels plate with two channels, lumped

NAME	VALUE
Interface	
Battery Connectivity	Single sided
Number of battery thermal nodes (surfa...	1
> Dimension of battery thermal nodes (surfa...	ones(1, 2)
> Coordinates of battery thermal nodes (sur...	ones(1, 2)
Number of partitions in X dimension for t...	1
Number of partitions in Y dimension for t...	1
Plate Material	
Fluid Properties	
Design	

Parallel Channels

Number of partitions in X: 1
Number of partitions in Y: 1



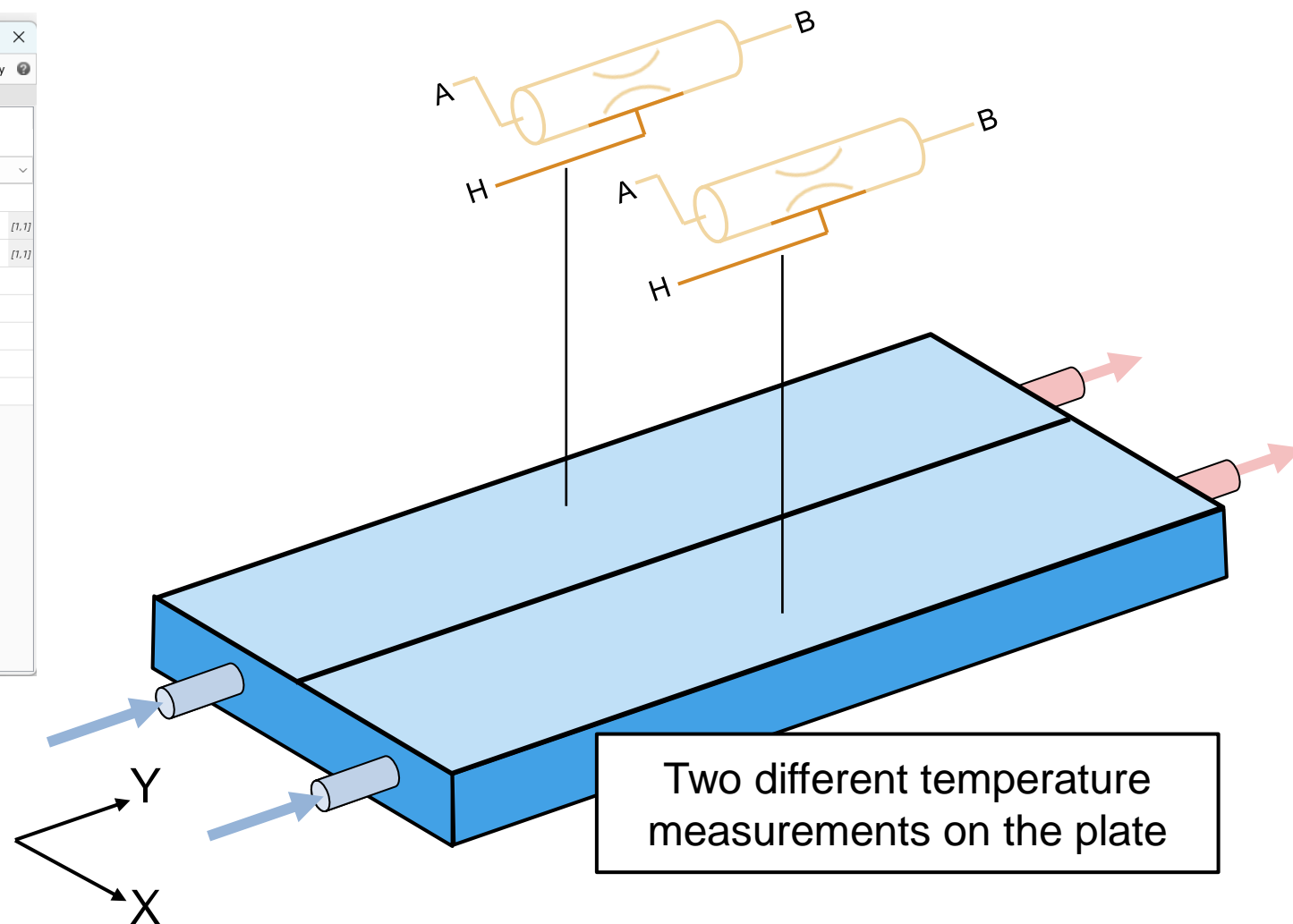
Lumped plate with one temperature value

Cooling plate model

Parallel Channels plate with two channels, two plate elements

NAME	VALUE
Interface	
Battery Connectivity	Single sided
Number of battery thermal nodes (surface...)	1
> Dimension of battery thermal nodes (surfa...)	ones(1, 2) [1, 1]
> Coordinates of battery thermal nodes (sur...)	ones(1, 2) [1, 1]
Number of partitions in X dimension for t...	1
Number of partitions in Y dimension for t...	1
Plate Material	
Fluid Properties	
Design	

Number of partitions in X: 2
 Number of partitions in Y: 1

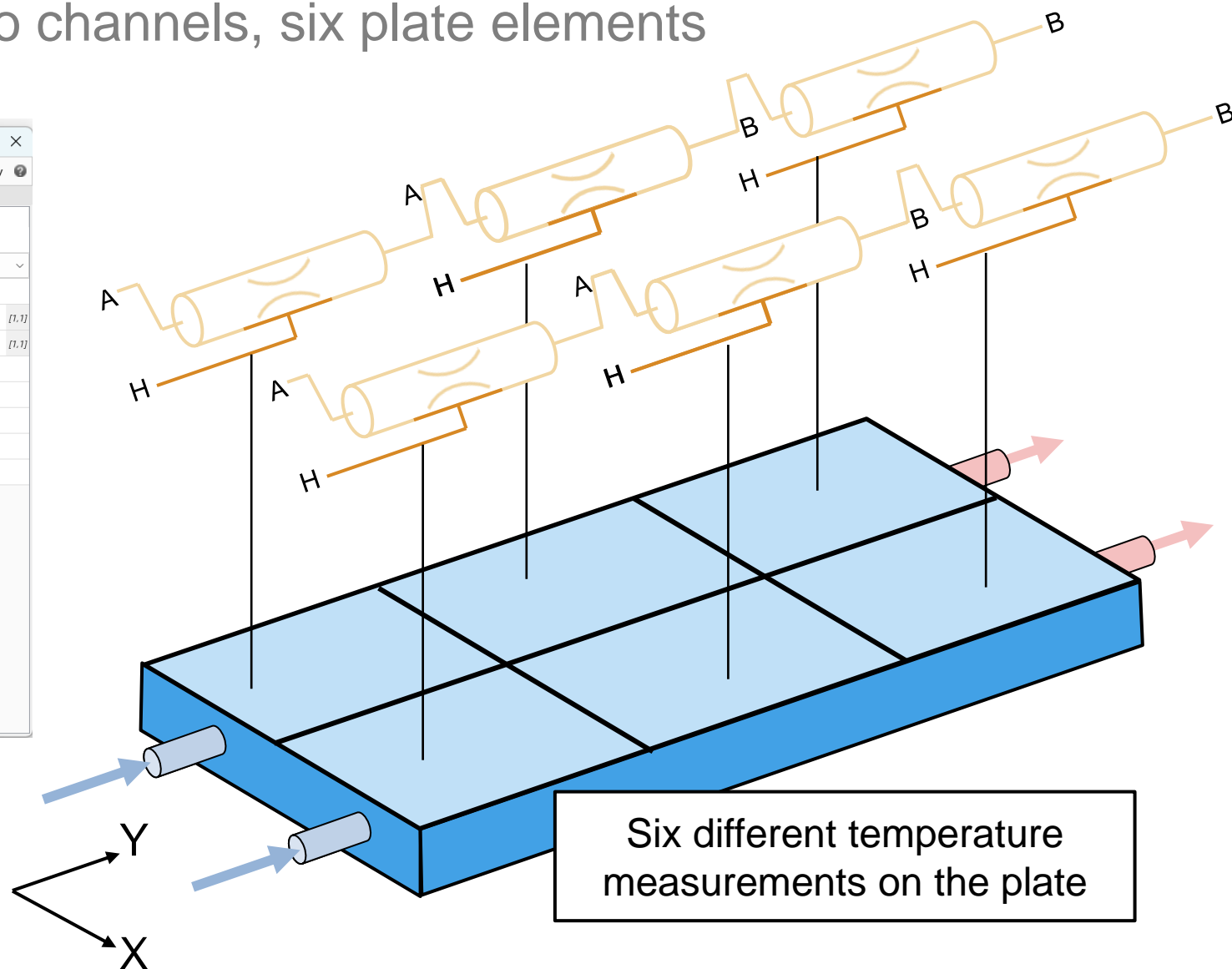


Two different temperature measurements on the plate

Cooling plate model

Parallel Channels plate with two channels, six plate elements

NAME	VALUE
Interface	
Battery Connectivity	Single sided
Number of battery thermal nodes (surface...)	1
> Dimension of battery thermal nodes (surfa...)	ones(1,2) [1,1]
> Coordinates of battery thermal nodes (sur...)	ones(1,2) [1,1]
Number of partitions in X dimension for t...	1
Number of partitions in Y dimension for t...	1
Plate Material	
Fluid Properties	
Design	

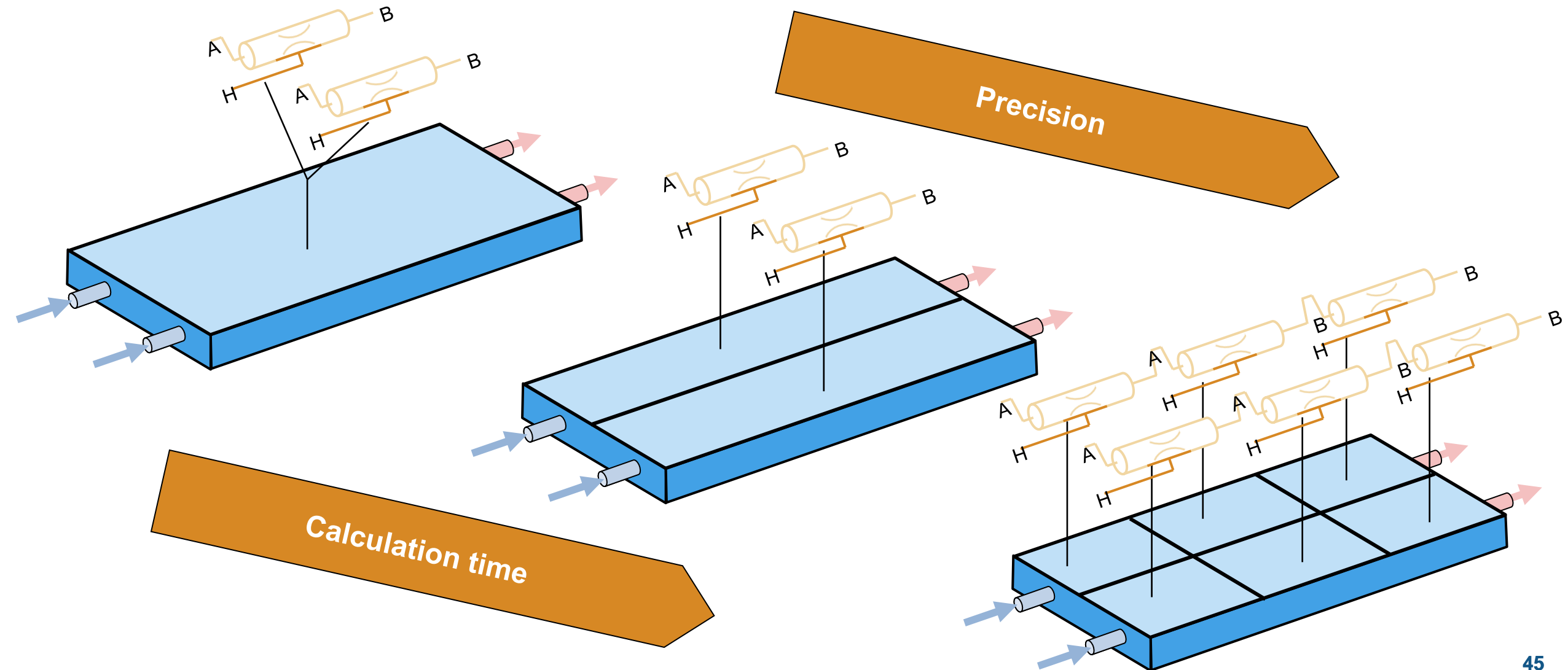


Number of partitions in X: 2
 Number of partitions in Y: 3

Six different temperature measurements on the plate

Cooling plate model

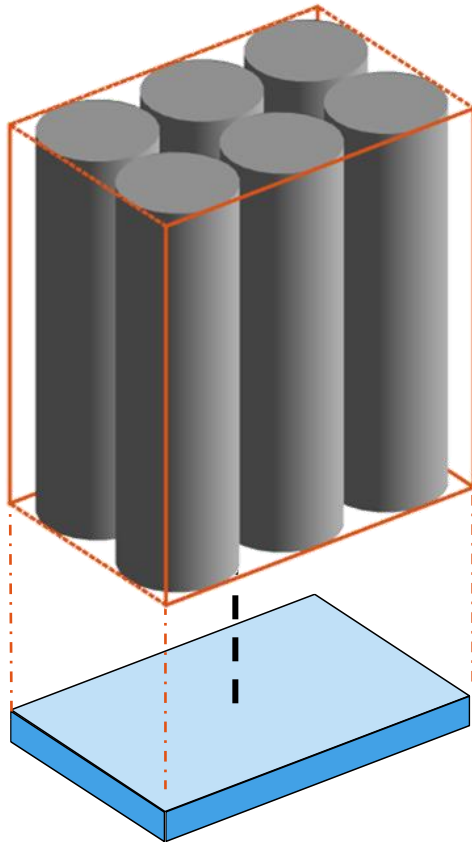
Find tradeoff between calculation speed and precision



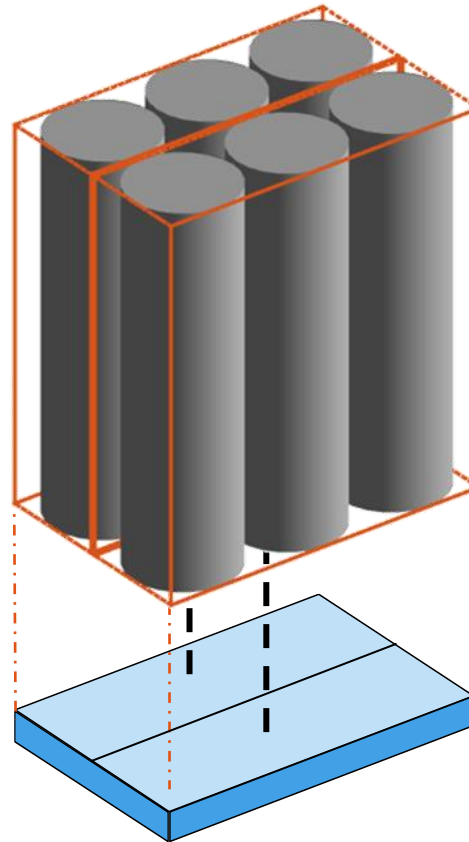
Connecting Pack and Plate

Combine plate and pack module fidelities

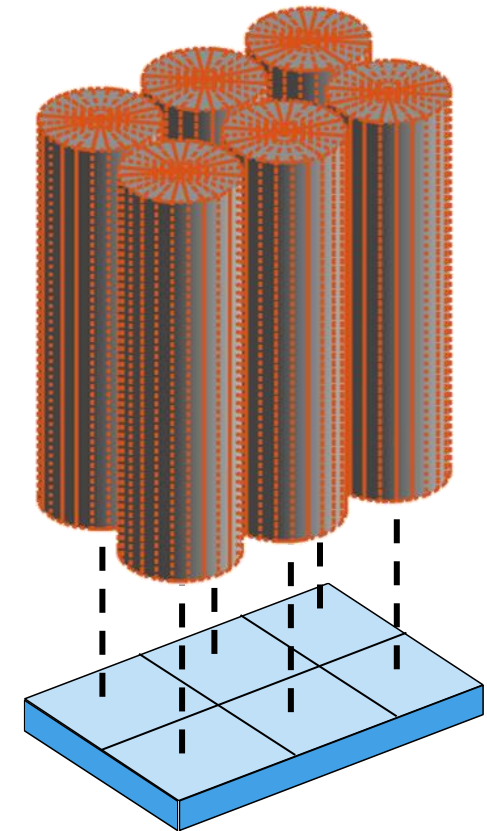
**Lumped module
Lumped plate**



**Grouped module
Discretized plate**



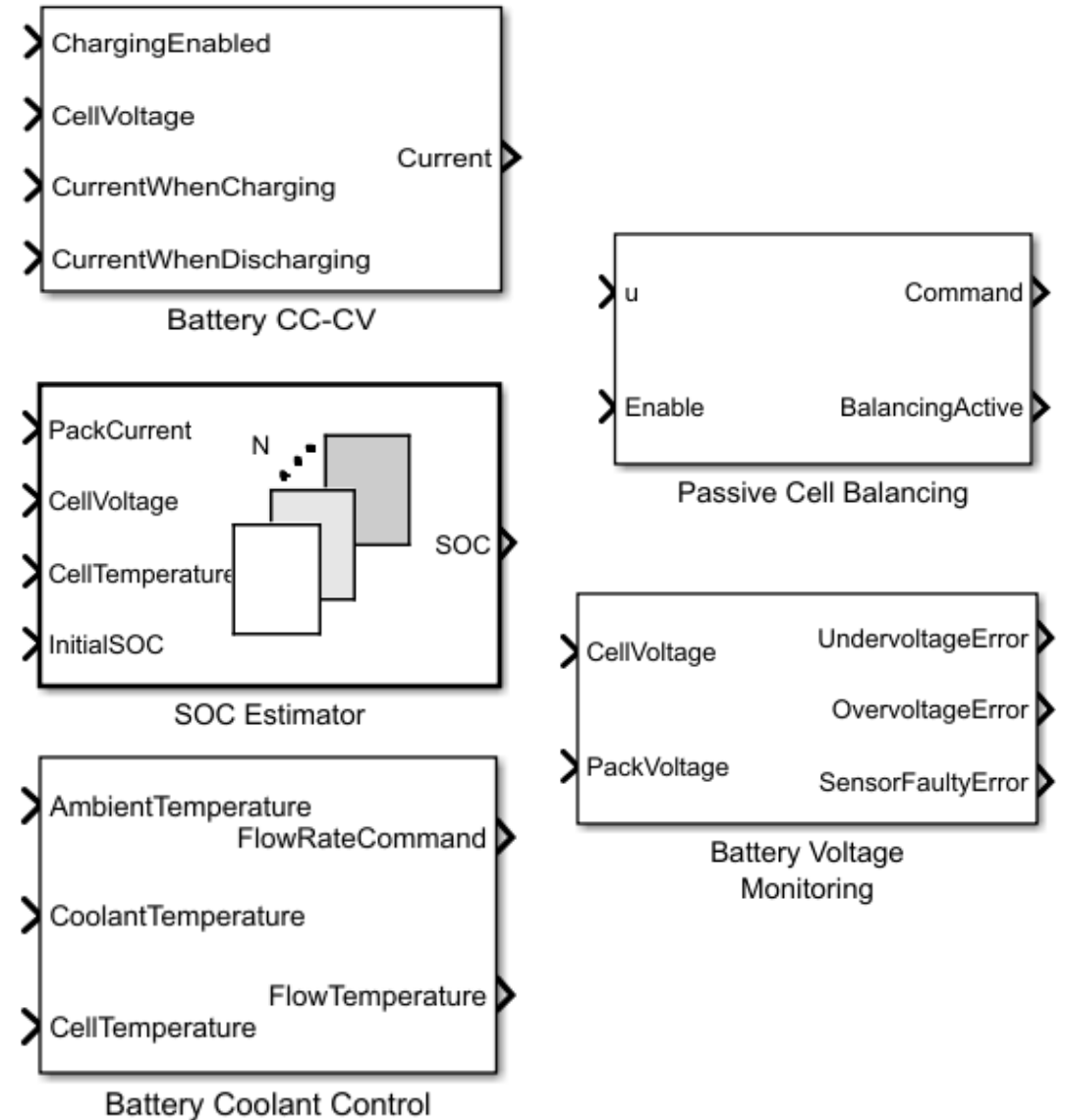
**Detailed module
Discretized plate**



Battery Management Algorithms

Block overview

- Charge and discharge
 - CC-CV, current limits
- Passive cell balancing
- Estimators
 - SOC, SOH
- Protection
 - Current, voltage, temperature monitor
 - Fault qualification
- Thermal management
 - Coolant and heater control



Pouch Module with Cooling Plate & BMS Blocks

Demo implementation

Simple module with cooling plate & BMS

1. Battery module with 12 pouch cells ([plot module](#))
2. The module has a detailed model resolution
3. The module is coupled with a cooling plate with two channels
4. The plate is discretized along the X and Y direction (2x2)
5. The SOC of the module is estimated with a Kalman Filter
6. The estimated SOC is used to tune a CC-CV profile
7. The module is actively cooled by battery coolant control
8. Analyze results with [Simscape Result Explorer](#)

Module and plate visualization

Module and Plate

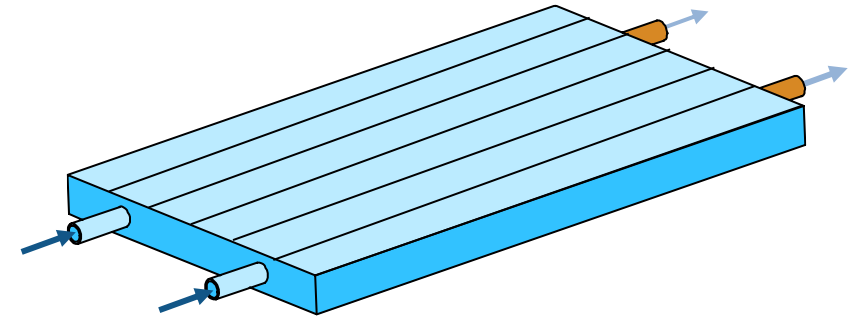
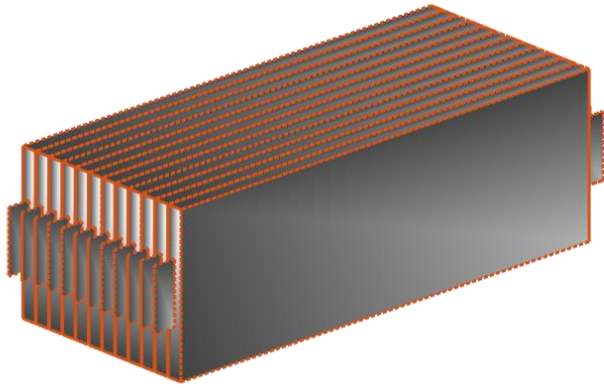
Coolant control

SOC Estimator

Battery CC-CV

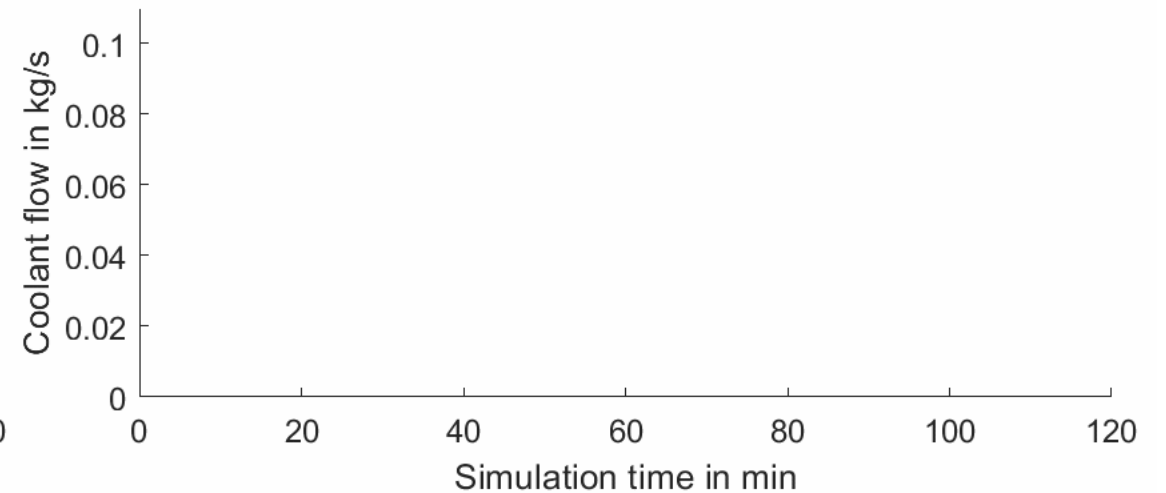
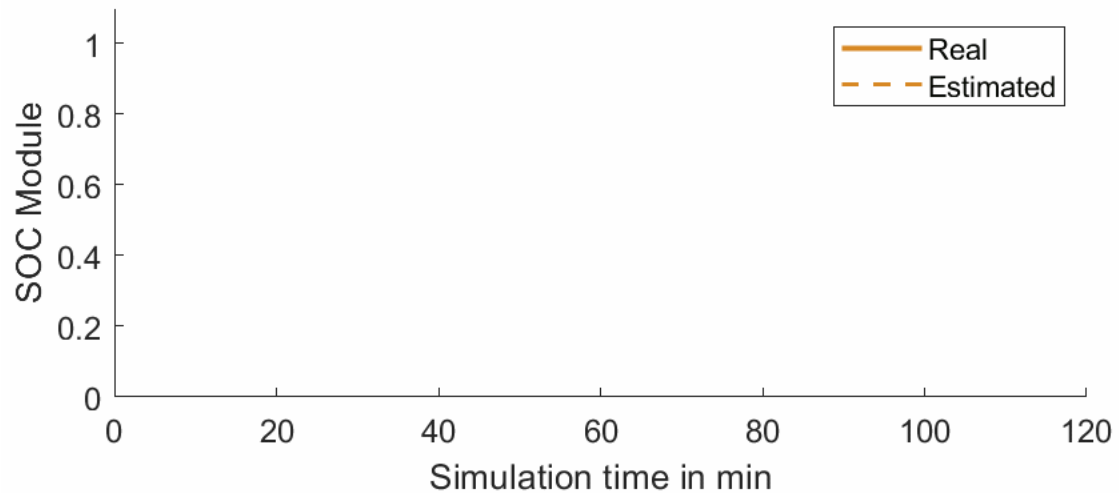
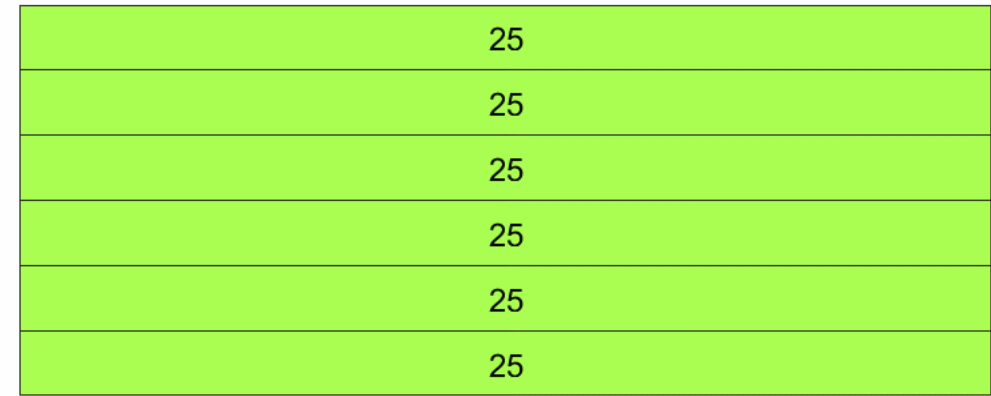
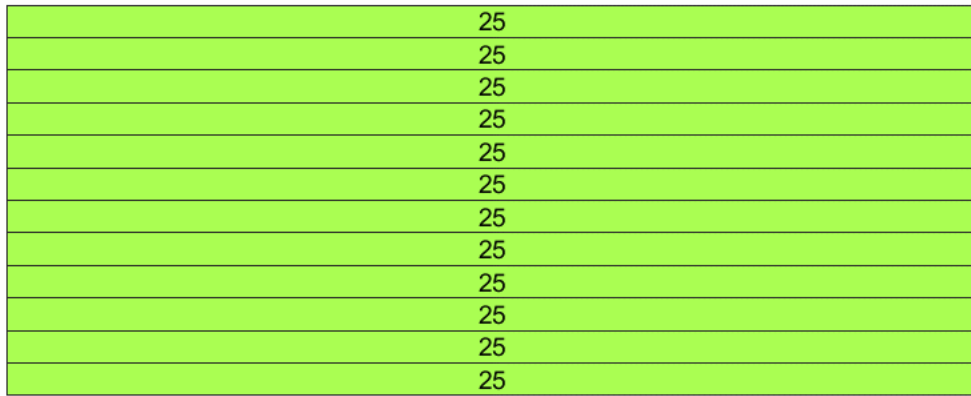
Pouch Module with Cooling Plate & BMS Blocks

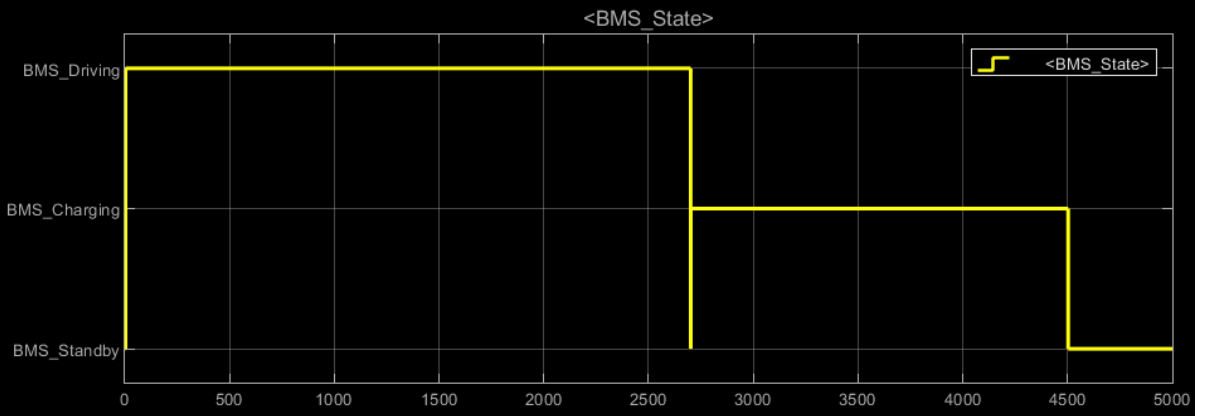
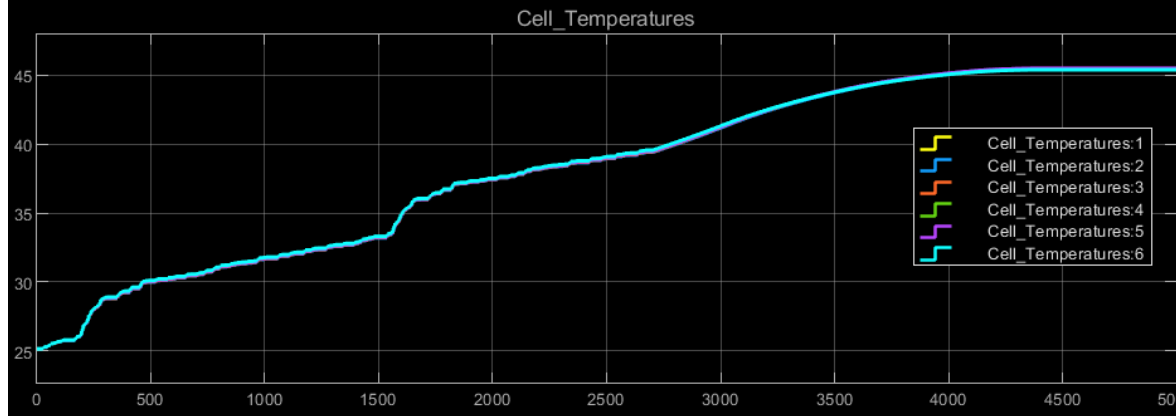
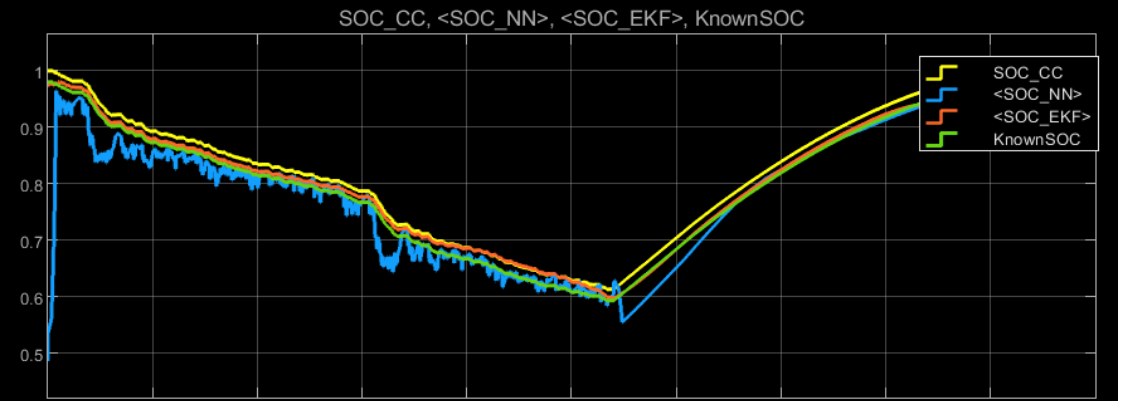
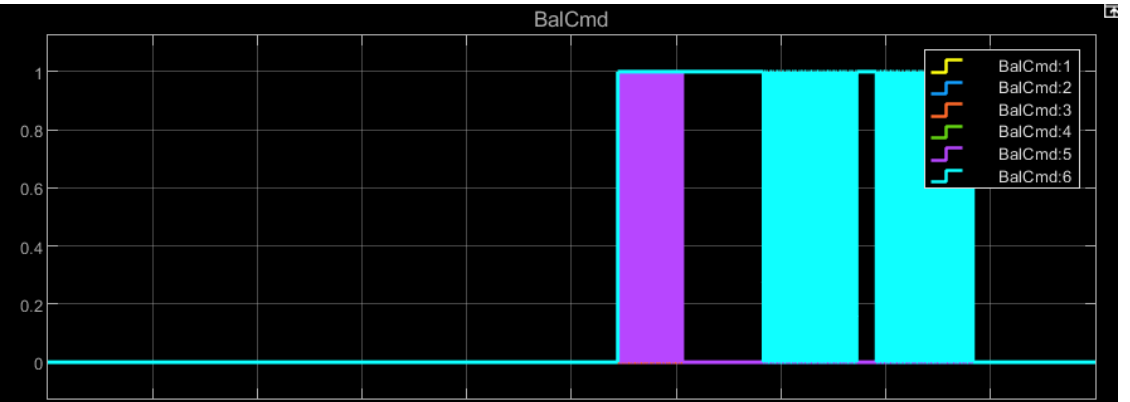
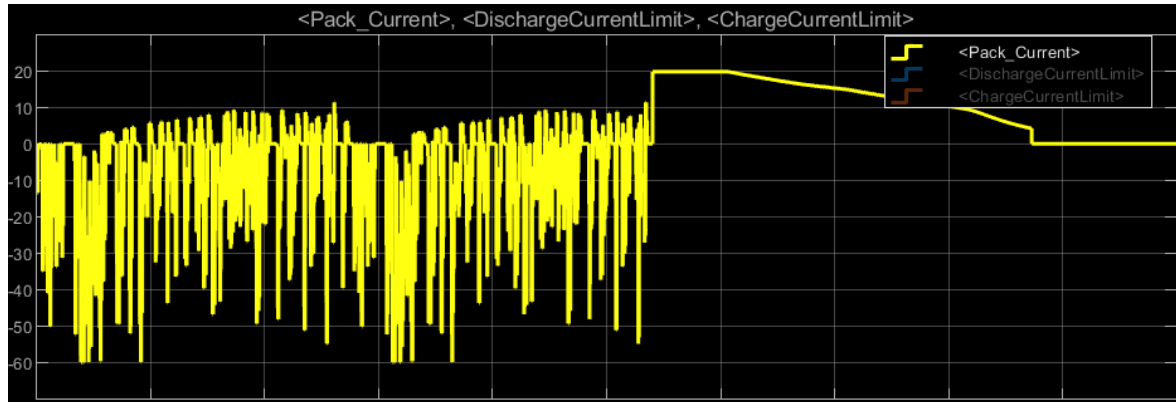
Simulation results



Pouch Module with Cooling Plate & BMS Blocks

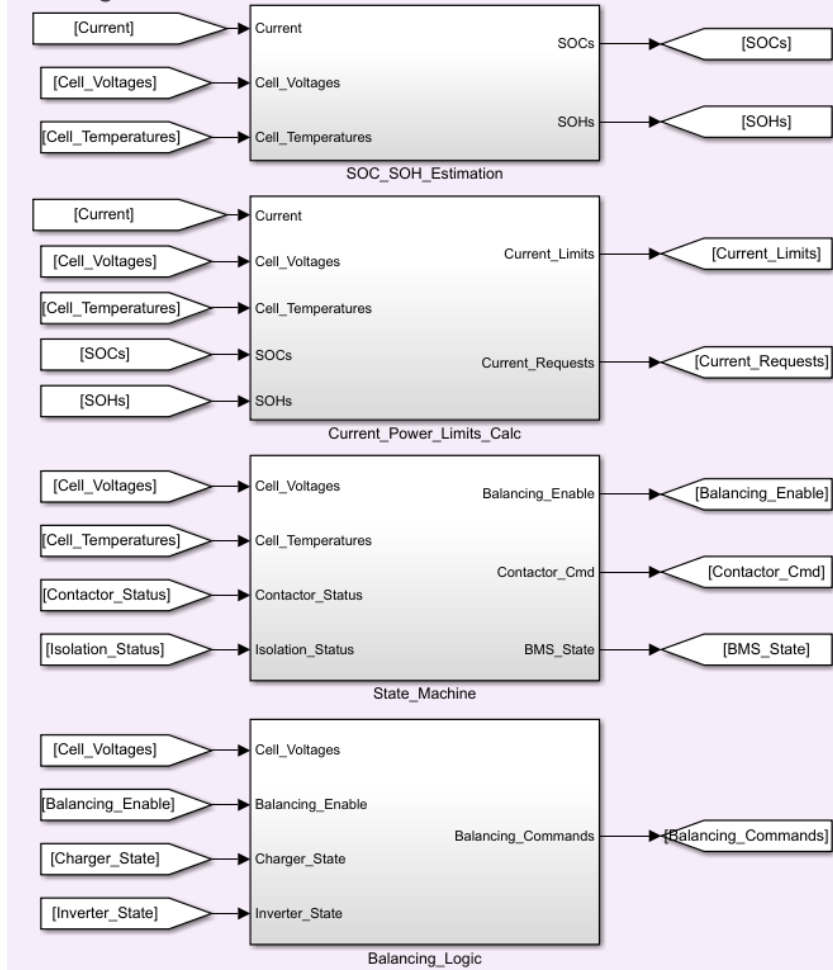
Simulation results





Generate C/C++ Code From BMS Algorithm Models

BMS Algorithms



Find:
Match Case

Contents

[Summary](#)

[Subsystem Report](#)

[Traceability Report](#)

[Static Code Metrics Report](#)

[Code Replacements Report](#)

Highlight Navigation

Previous
Next

Generated Code

[-] Model files

State_Machine.c (16)

[State_Machine.h](#)

[State_Machine_private.h](#)

[State_Machine_types.h](#)

[+] Shared files (3)

```

387
388     if (((uint32_T)State_Machine_DW.temporalCounter_i3) < 15U) {
389         State_Machine_DW.temporalCounter_i3 = (uint8_T)((int32_T)((int32_T)
390             State_Machine_DW.temporalCounter_i3) + 1));
391     }
392
393     if (((uint32_T)State_Machine_DW.is_active_c2_State_Machine) == 0U) {
394         State_Machine_DW.is_active_c2_State_Machine = 1U;
395         State_Machine_DW.is_MainStateMachine = State_Machine_IN_Standby;
396         *rtu_BMS_State = 0;
397         State_Machine_DW.MonitorCurrLimMode = MonitorCurrLimModeType_NoCurrLimFault;
398         State_Machine_DW.MonitorCellVoltageMode =
399             MonitorCellVoltageModeType_NoCellVoltFault;
400         State_Machine_DW.Delta = (real32_T) fabs((real_T)((real32_T)
401             ((*rtu_Pack_Voltage) - sum_gyOCKAG3(rtu_Cell_Voltages))));
402         State_Machine_DW.FaultPresent = false;
                    
```

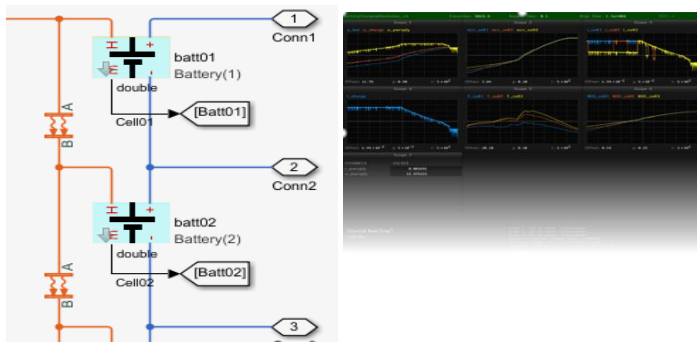
State_Machine View All

State_Machine ▶ State_Machine

Hardware-In-Loop Testing of Battery Management System

Testing BMS with Emulated Battery Cells

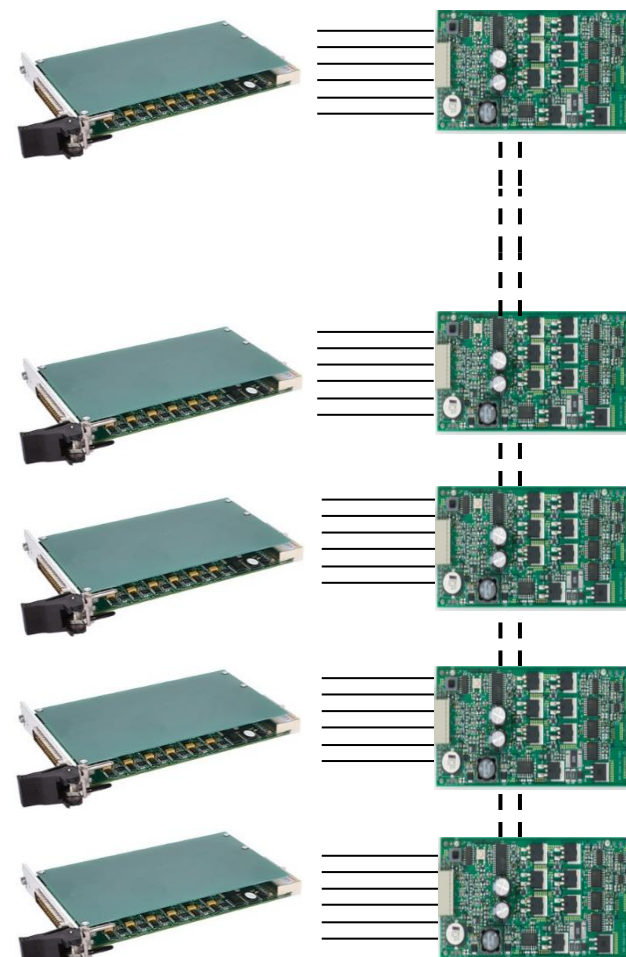
- Reduce testing time
- Test fault conditions safely
- Automate testing



Automatic Code Generation



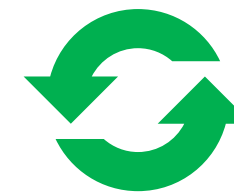
Battery Emulation



Measurement & Diagnostics

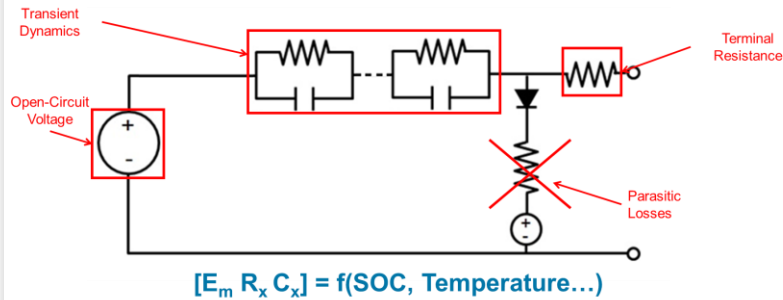


Main Controller

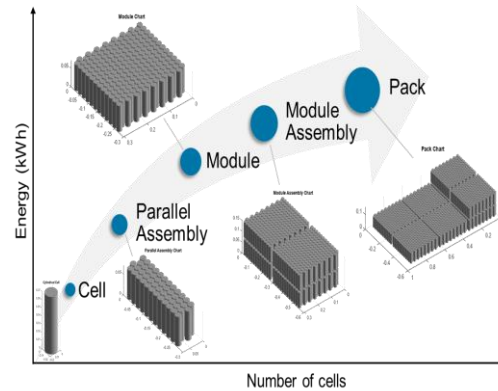


Battery Pack and BMS Design Workflow Tasks

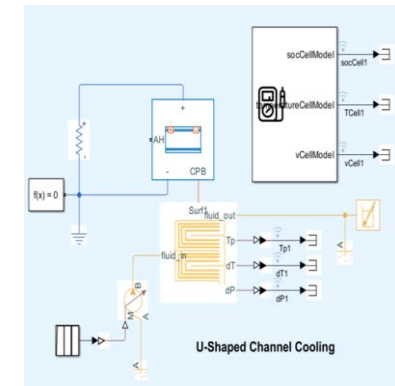
1. Cell Modeling



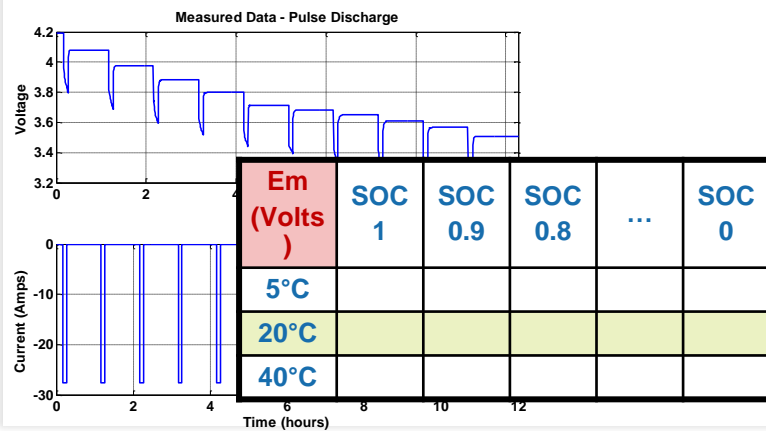
3. Battery Pack Design



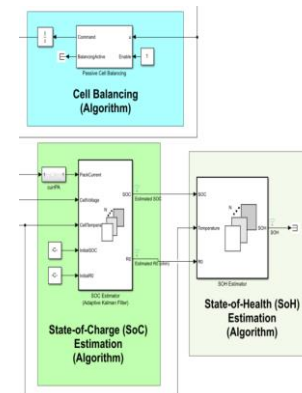
5. Thermal Management System Design



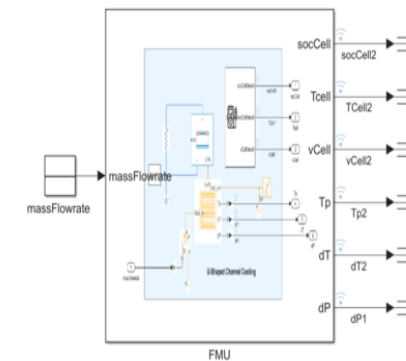
2. Cell Parametrization



4. Battery Management System Design & Deployment



6. Deployment and HIL



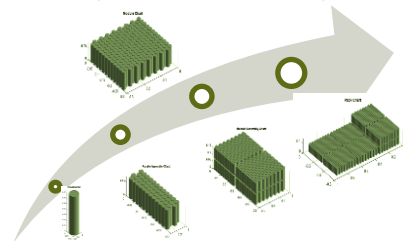
Challenges in developing a BMS

Collaboration Gap



Leverage models to communicate technical specifications, design implementation, results and maintain traceability

System Analysis



Use Simscape Battery framework that is assembled specifically to create a bridge between cell and system.

Long Iteration Cycles



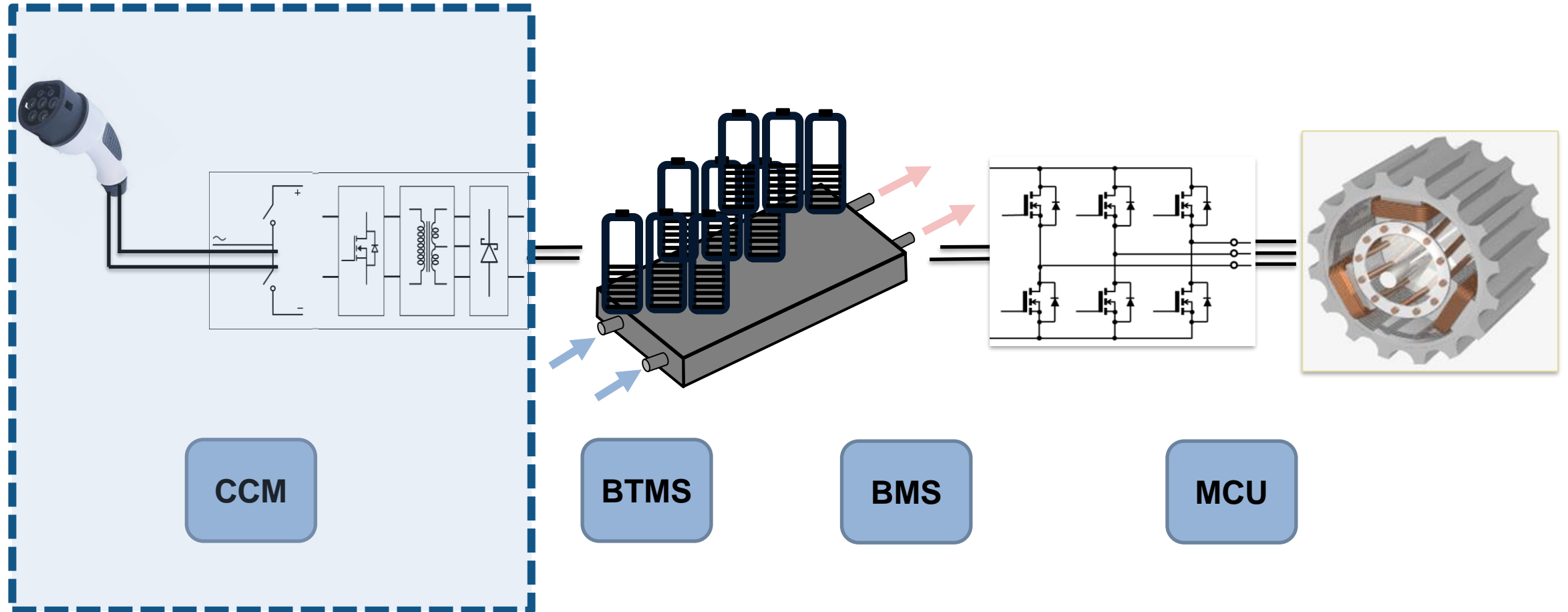
Test your design iterations every step of the way through simulations and Hardware-In-Loop testing

Safety Critical System



Gain confidence in design and work towards safety certification

Electric Vehicle System



Challenges for Power Electronics Engineer

- Understand the impact of the power source and load
- Testing for a complete range of operating and fault conditions
- Designing and implementing digital controls using *only* SPICE simulator tools
- Catching errors during software-hardware integration testing
- Compliance to industry standards
- Development Time



VONSCH Speeds the Development of Control Systems for Solar Inverters and Battery Chargers

Challenge

- Develop solar inverter and battery charger control systems amid frequently shifting market requirements

Solution

- Use Model-Based Design with MATLAB and Simulink to model power electronics and control systems, run simulations, and generate embedded code for a TI microcontroller

Results

- Product development time reduced by one year
- New product R&D accelerated via model reuse
- Number of hardware prototypes reduced



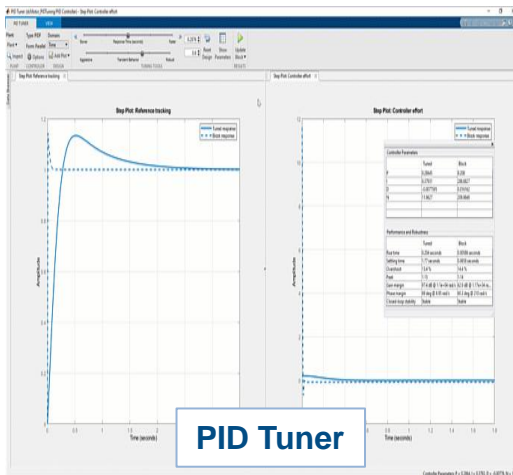
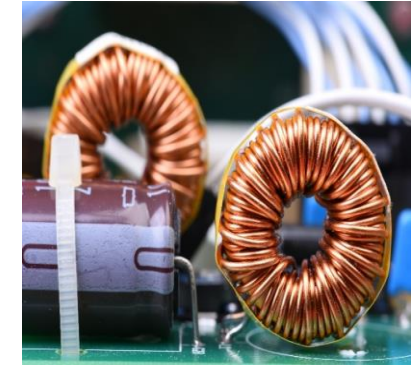
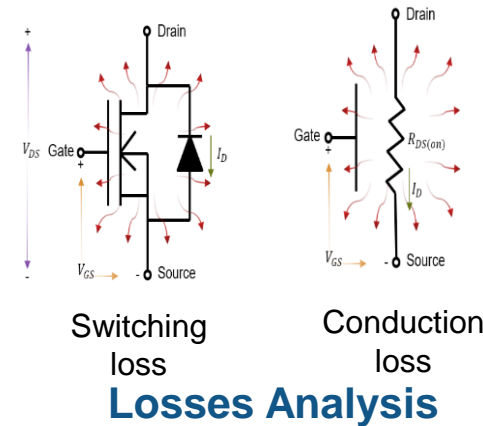
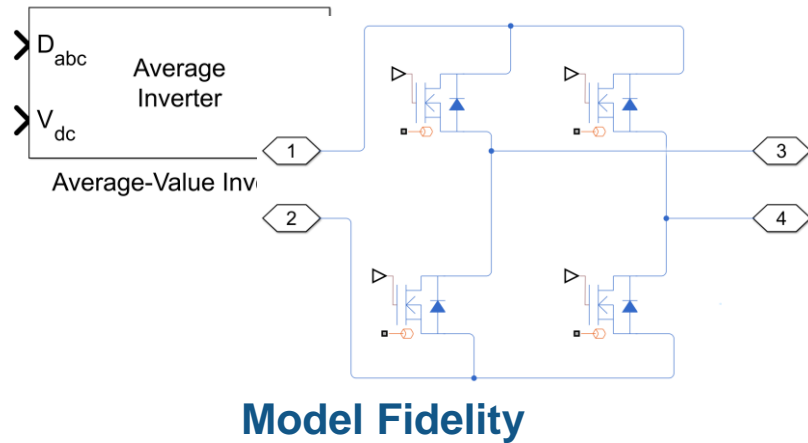
Development and testing of FOTO CONTROL 1f and FOTO CHARGER products.

“Model-Based Design enabled us to quickly adapt to changing legislation and requirements. Before prototype hardware was available, we designed and simulated the entire system in Simulink and generated embedded code for the controller, which was working on the prototype within a day or two after the hardware was available.”

Dr. Jakub Vonkomer
VONSCH

[Link to user story](#)

Power Converter Control Design Workflow Tasks



Controller Design

Model

CONTROLLER

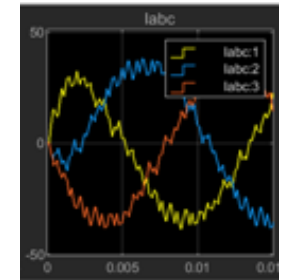
```

1  /* Model step function */
2  void Controleur_step(void)
3  {
4  /* local block I/O variables */
5  real_T rtb_Gainreglage;
6  real_T rtb_Saturation;
7
8  /* Outputs for Atomic Subsystem: '<root>/Controleur numerique'
9  /* DiscreteStateSpace: '<sl>/Filtre numerique' incorporates:
10 * Inports: '<root>/Heart position'
11 */
12
13 static const int_T colCidsRow[] = { 0, 1, 2, 3, 4 };
14
15 const int_T *pCids = &colCidsRow[0];
16 const real_T *pCO = Controleur_F.Filtre;
17 const real_T *sd = &Controleur_IM.Filtre;
18 real_T *y0 = &rtb_Gainreglage;
19 int_T numNonZero = 0;
20 *y0 = (*pCO++)*sd[*pCids++];
21 while (numNonZero-->0) {
22 *y0 += (*pCO++)*sd[*pCids++];
23 }
24 }
    
```

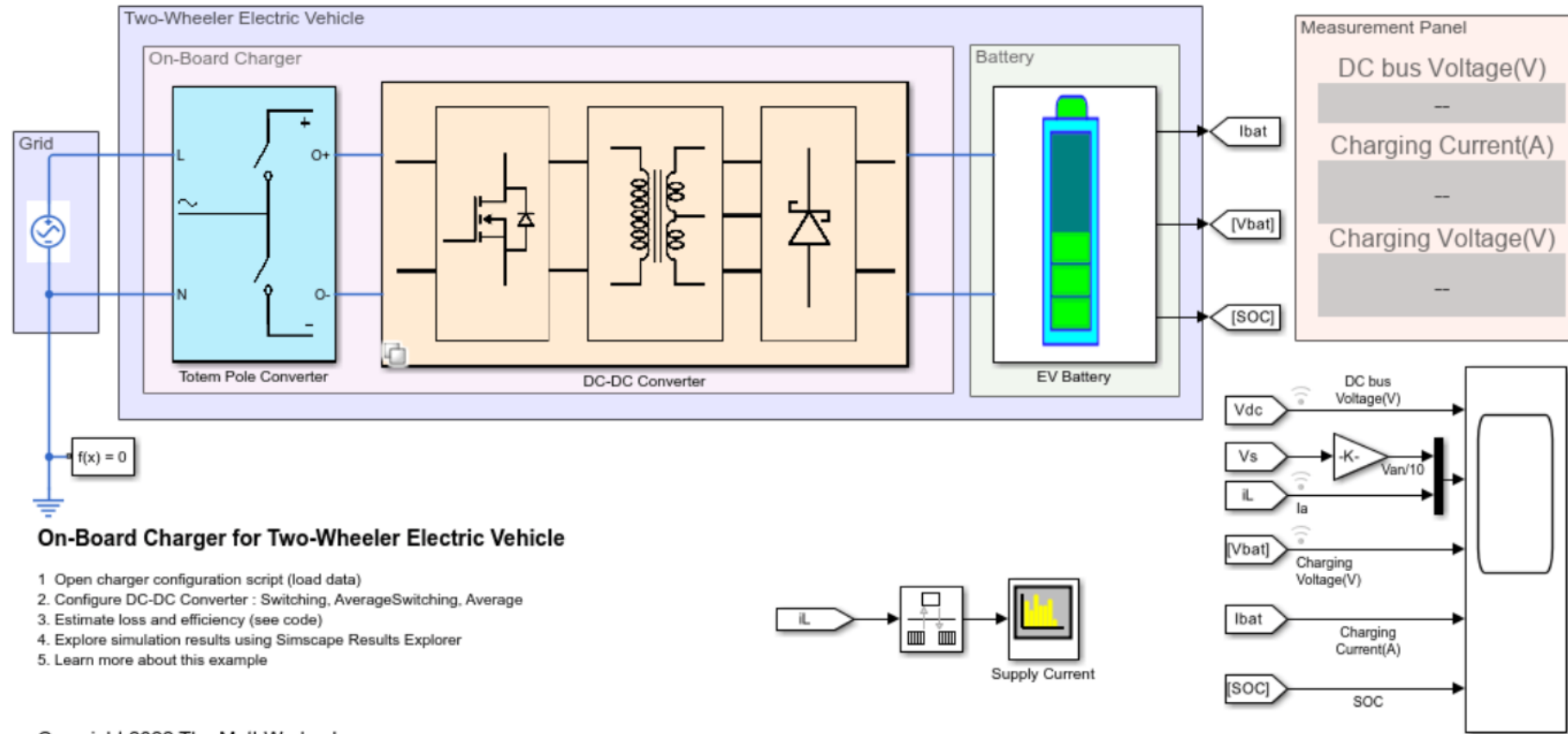
C64x™ DSP

TEXAS INSTRUMENTS

One-Click Embedded Deployment

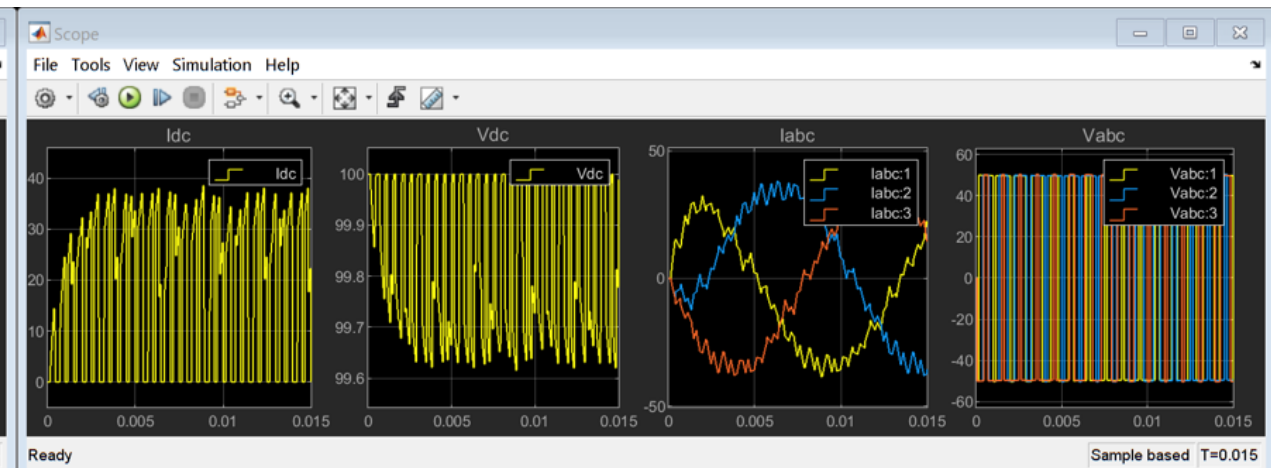
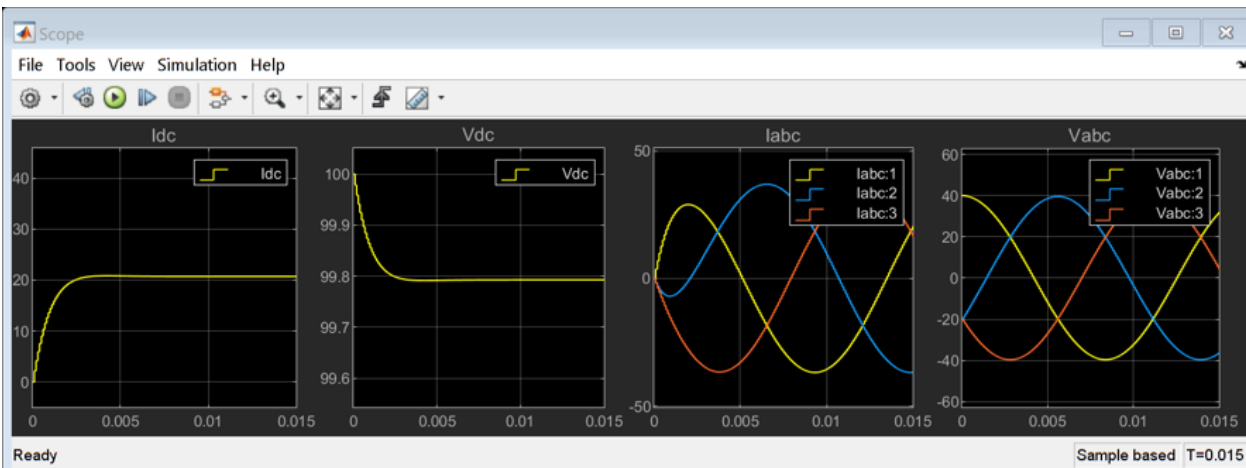
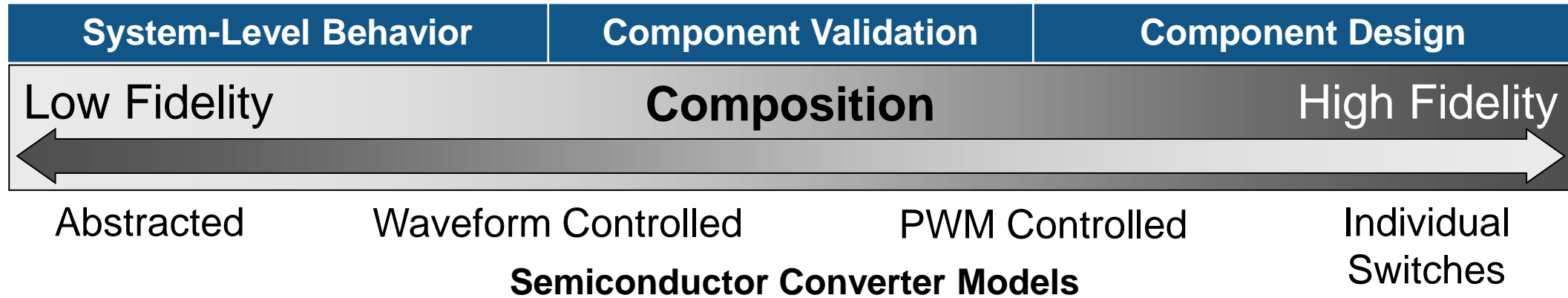


On-Board Charger for Two-Wheeler Electric Vehicle

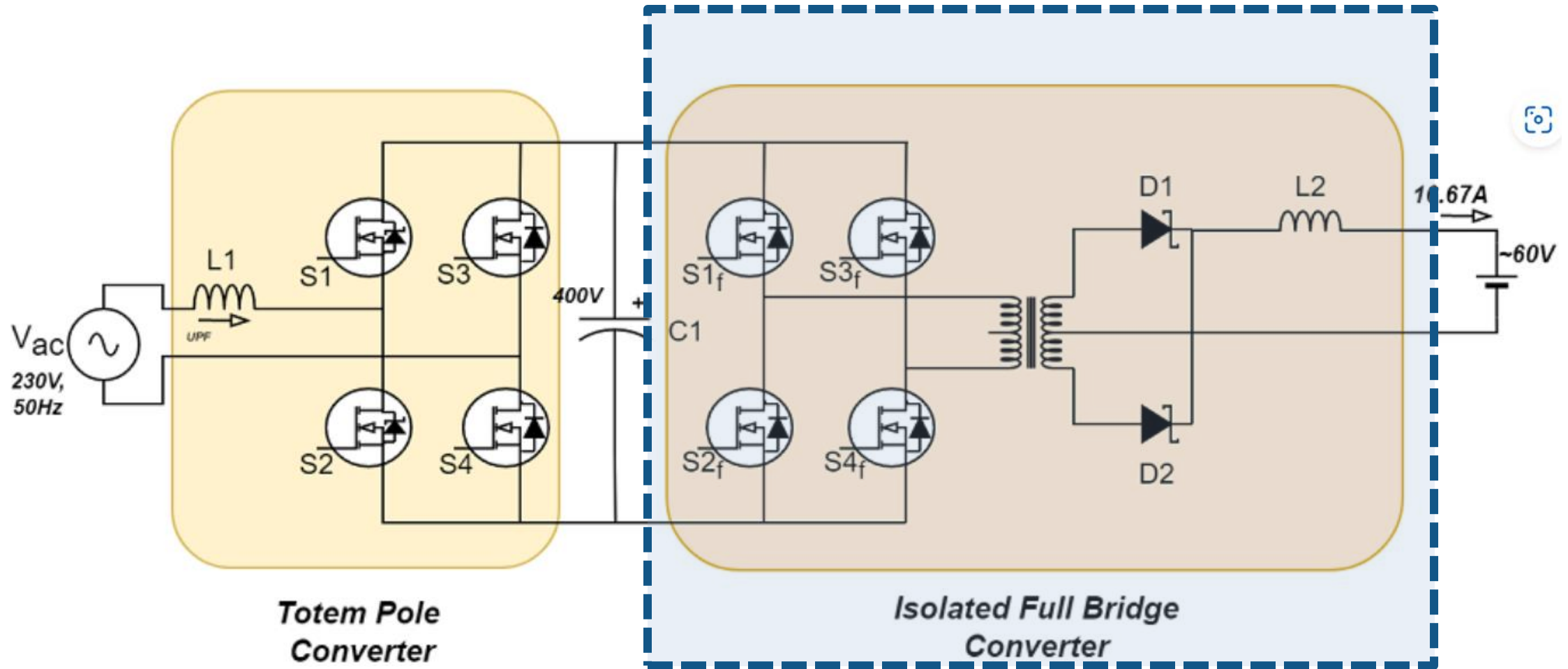


>> ee_onboard_charger_two_wheeler

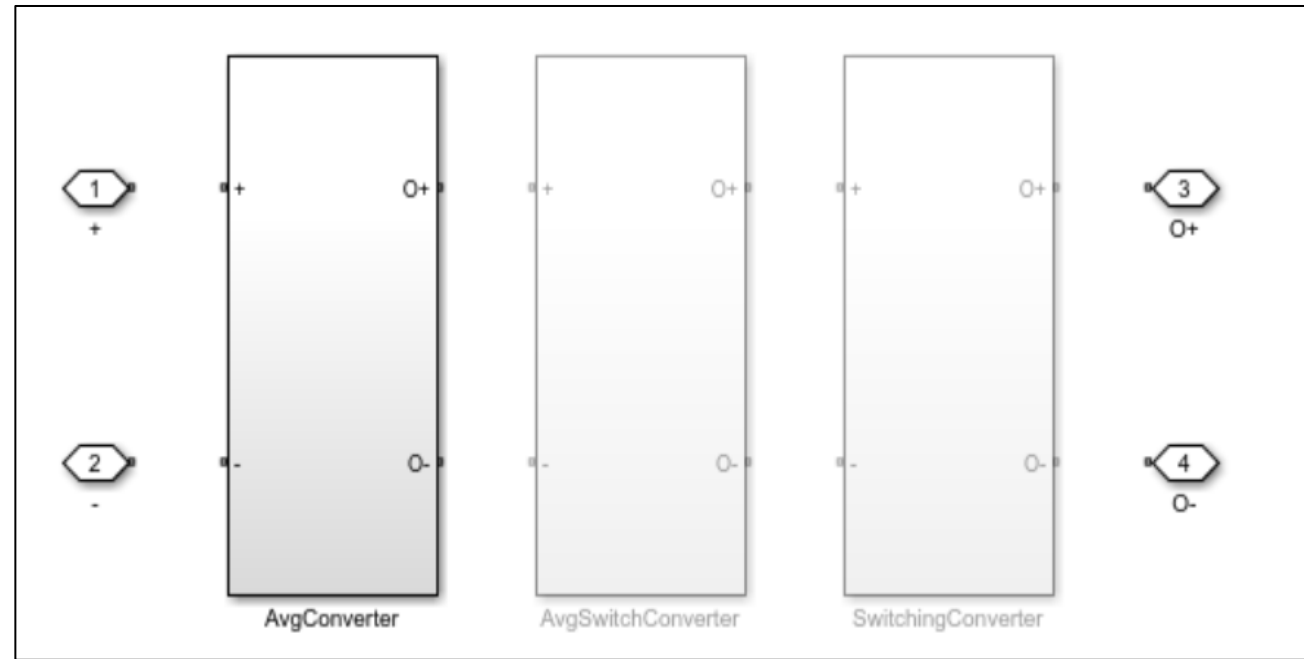
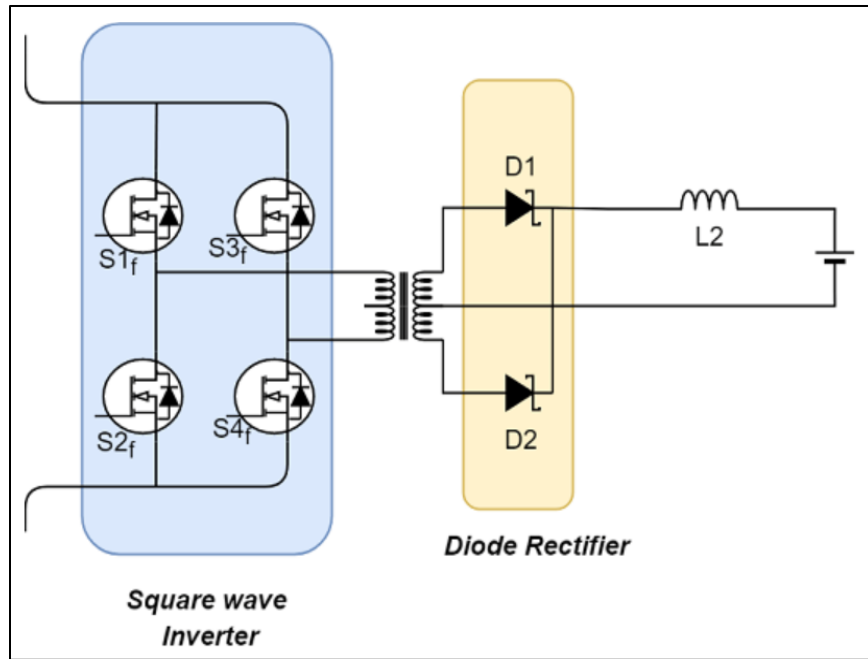
Power Converter Model Fidelity



On-Board Charger for Two-Wheeler Electric Vehicle



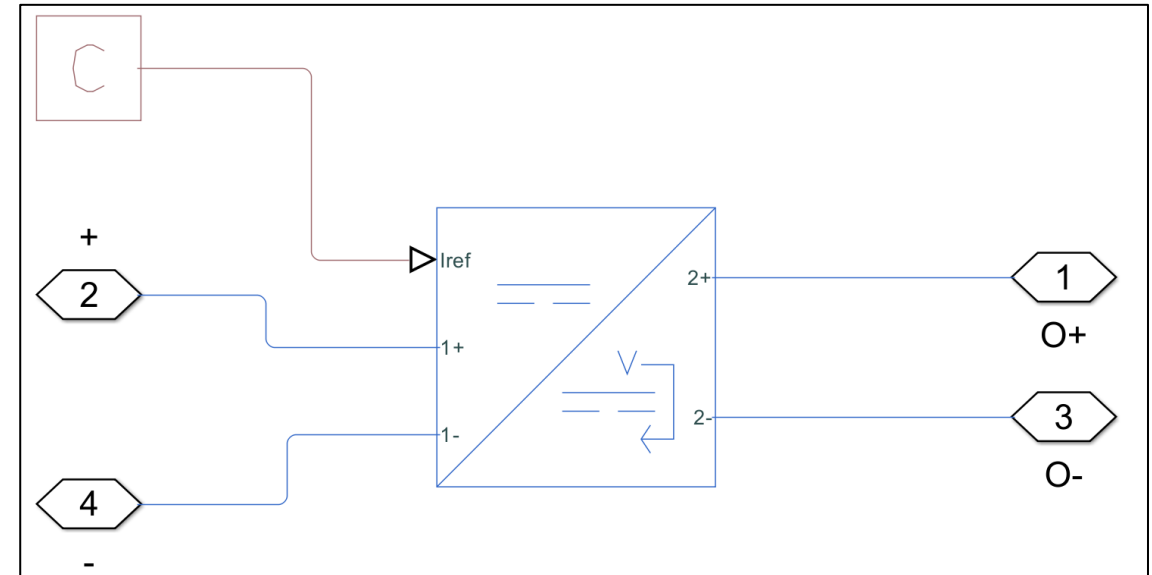
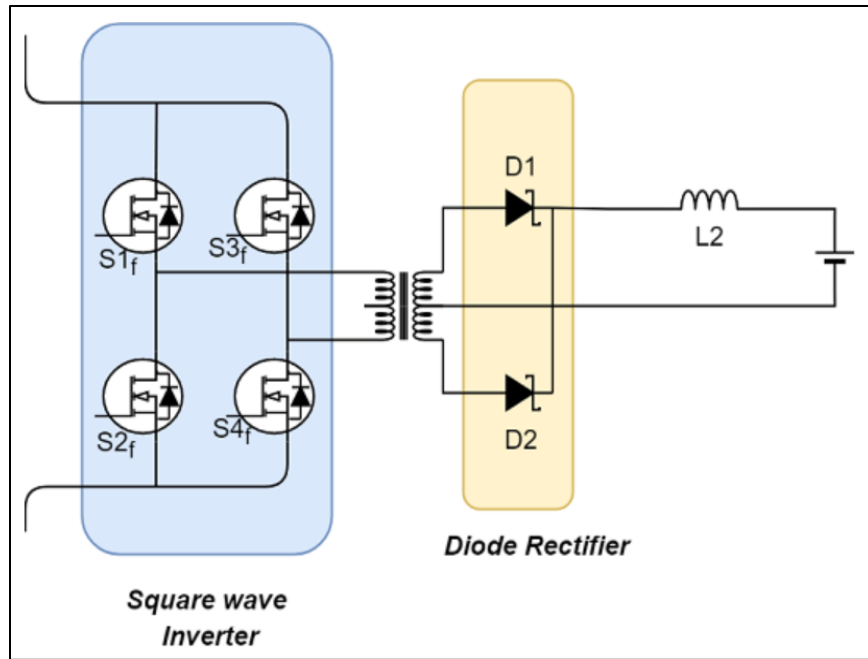
DC-DC Converter - Variant Subsystem



Three Variant Implementation:

- Average Converter
- Averaged Switching Converter
- Switching Converter

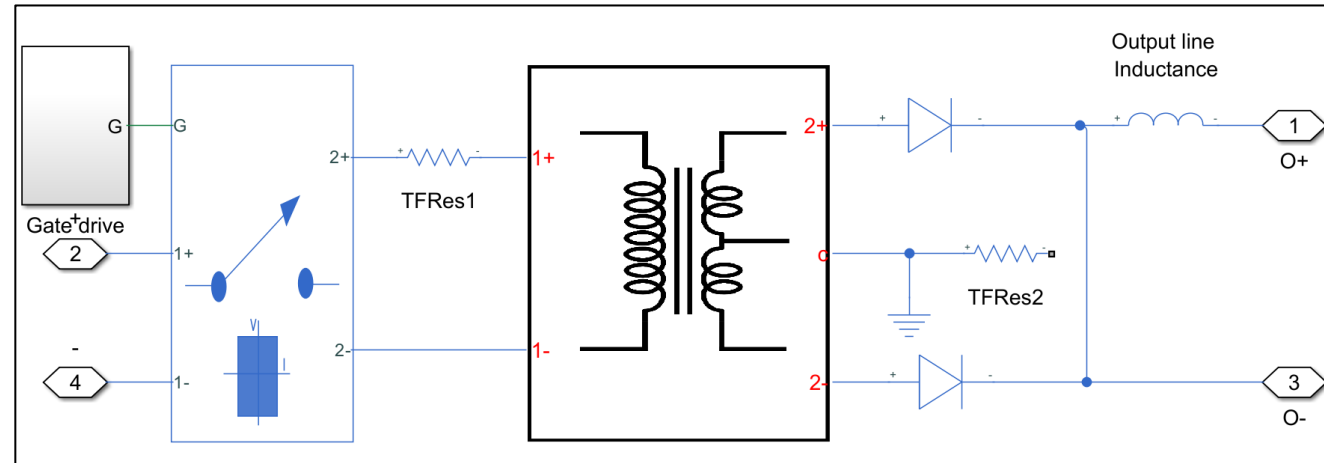
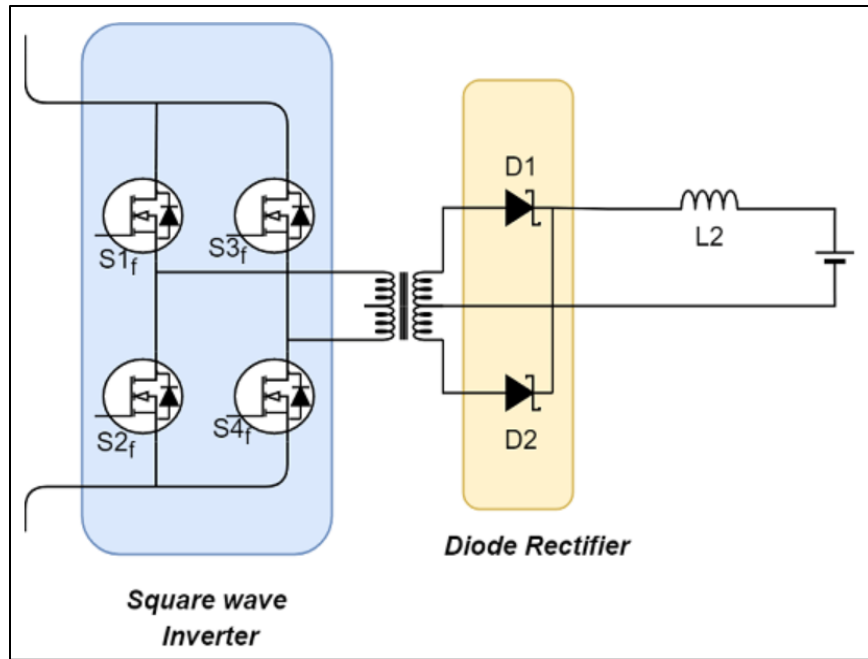
DC-DC Converter - Variant Subsystem



Average Converter:

- Fastest simulation
- Slight reduction in accuracy during transient
- Capture system level behavior

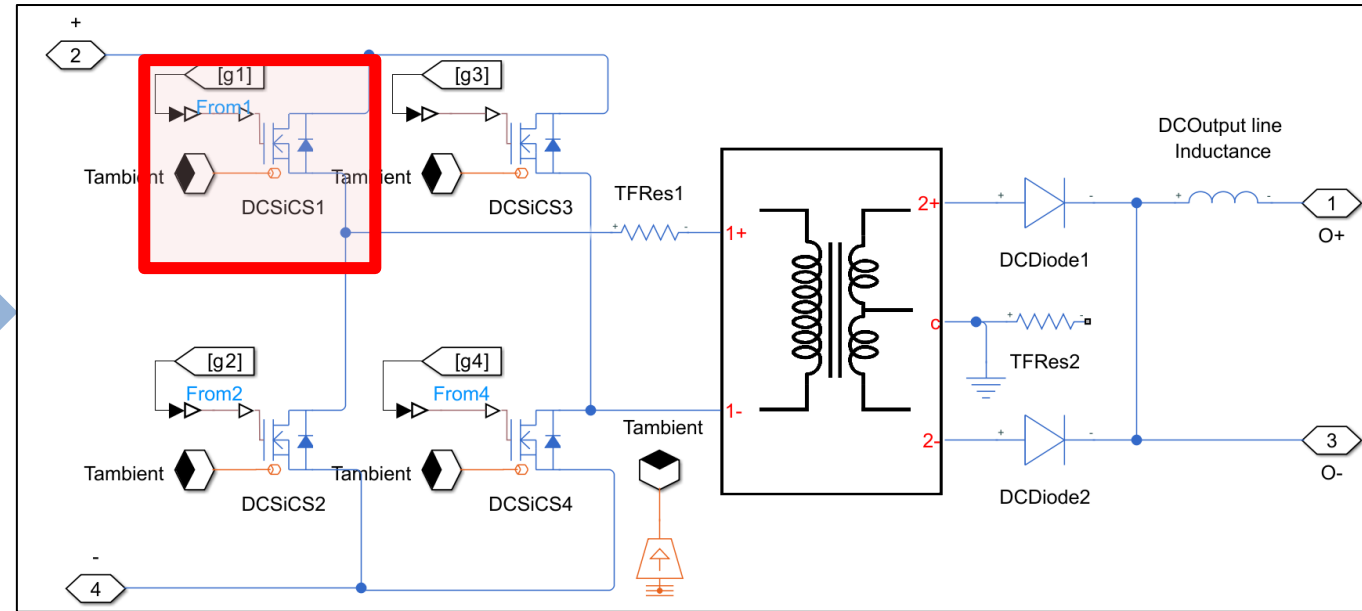
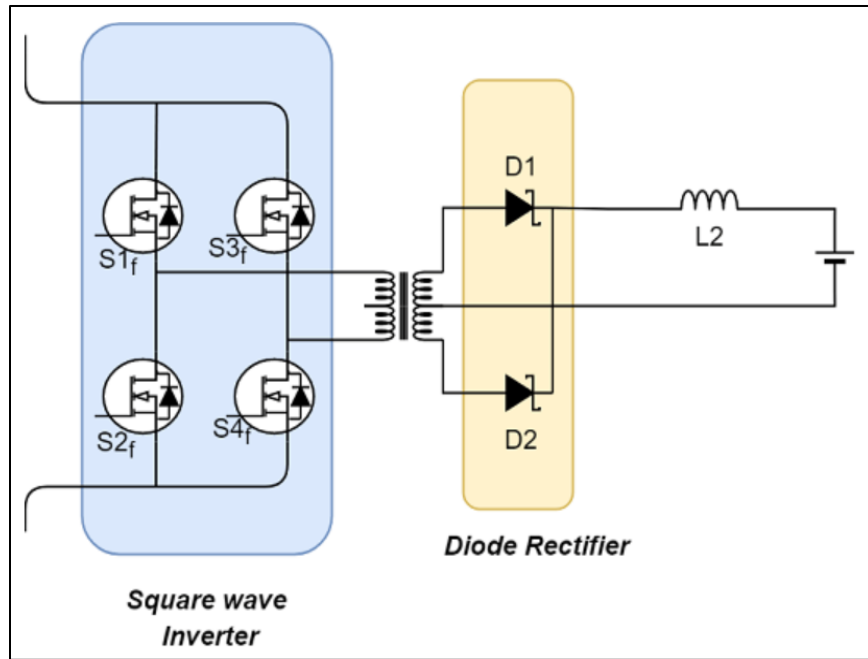
DC-DC Converter - Variant Subsystem



Averaged Switching Converter:

- Faster simulation time
- Better accuracy and capture switching events
- Verify the operation of the converter

DC-DC Converter - Variant Subsystem



Fully Switching Converter:

- Slower simulation
- Capture complete switching details
- Calculate converter losses and to estimate its efficiency

MOSFET Parametrization

Manufacturer Specific SiC MOSFETs

- Database with parameter values to match vendor-specific motors

Block Parameterization Manager: MOSFET(Ideal, Switching)

SELECT FORMAT

Apply all Reset all

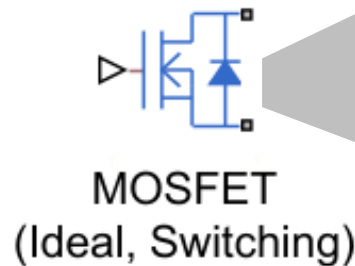
Manufacturer: Infineon

PARAMETERIZE

Select part

Part number	Manufact	IdsMaxPulsed,A
AIMW120R035M1H	Infineon	52
AIMW120R045M1	Infineon	52
AIMW120R060M1H	Infineon	36
AIMW120R080M1	Infineon	33
IMRG120R030M1H	Infineon	56

Manufacturer dropdown menu items: All, IXYS, Infineon, ROHM Semiconductor, STMicroelectronics, Wolfspeed



Block Parameters: MOSFET (Ideal, Switching)

MOSFET (Ideal, Switching) Auto Apply

Settings Description

NAME	VALUE
Modeling option	No thermal port
Selected part	Infineon:AIMW120R035M1H
▼ Main	
Gate-control port	PS
> Drain-source on resistance...	0.095 % Datasheet ... Ohm
> Off-state conductance	2e-08 % Datasheet ... 1/Ohm
> Threshold voltage, Vth	4.5 % Datasheet de... V
> Integral Diode	
> Initial Targets	
> Nominal Values	

Import Spice Subcircuit

Buck Converter Subcircuit Import

>>subcircuit2ssc (BUCK_SUBCKT.CIR)

The image shows a MATLAB code editor with two tabs: 'BUCK_SUBCKT.CIR' and 'buck.ssc'. A blue arrow labeled 'subcircuit2ssc' points from the left tab to the right tab. The 'BUCK_SUBCKT.CIR' tab contains the following code:

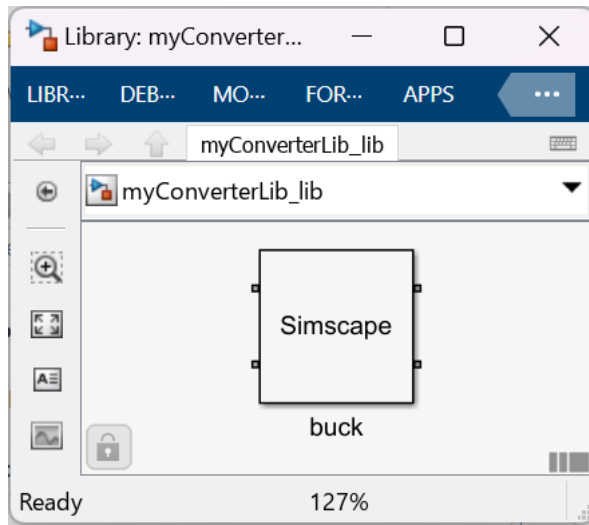
```

1  .subckt buck Vcc Gate Out Gnd
2
3  M1  Vcc Gate 2 2 N1
4  D1  Gnd 2  D
5  L1  2  Out 50UH
6  C1  Out Gnd 25UF
7
8  .MODEL N1 NMOS LEVEL=1 L=1e-6 W=1
9  .MODEL D D IS=0.0002 RS=0.05 CJO=5e-10
10
11 .ends
  
```

The 'buck.ssc' tab contains the following code:

```

11 components(ExternalAccess=observe)
12
13 M1 = ee.additional.spice_semiconductors.spice_nmos(AD={0, 'm^2'},AS={0, 'm^2'},CBD=
14 CGBO={0, 'F/m'},CGDO={0, 'F/m'},CGSO={0, 'F/m'},CJ={0, 'F/m^2'},CJSW={0, 'F/m'},DE
15 FC={0.5, '1'},GAMMA={nan, 'V^0.5'},IS={1e-14, 'A'},JS={0, 'A/m^2'},KAPPA={0.2, '1'
16 LD={0, 'm'},LENGTH={1e-06, 'm'},LEVEL={1, '1'},SCALE={1, '1'},MJ={0.5, '1'},MJSW={
17 NFS={0, '1/cm^2'},NSS={nan, '1/cm^2'},NSUB={nan, '1/cm^3'},NRD={0, '1'},NRS={0, '1'
18 PHI={nan, 'V'},PS={0, 'm'},RD={nan, 'Ohm'},RS={nan, 'Ohm'},RSH={nan, 'Ohm'},THETA=
19 TPG={1, '1'},UCRIT={10000, 'V/cm'},UEXP={0, '1'},U0={600, 'cm^2/(V*s)'},VMAX={0, '
20 WIDTH={1, 'm'},Ci_param={1, '1'},Cov_param=2,C_param=2);
21
22 D1 = ee.additional.spice_semiconductors.spice_diode(AREA={1, 'm^2'},SCALE={1, '1'},
23 IKF={Inf, 'A/m^2'},ISR={0, 'A/m^2'},N={1, '1'},NR={2, '1'},M={0.5, '1'},VJ={1, 'V'}
24 CJO={5e-10, 'F/m^2'},FC={0.5, '1'},TT={0, 's'},RevBrk={2, '1'},BV={Inf, 'V'},IBV={
25 NBV={1, '1'},NBVL={1, '1'},XTI={3, '1'},EG={1.11, 'eV'},TIKF={0, 'K^-1'},TRS1={0, '
26 TBV2={0, 'K^-2'});
27
28 L1 = foundation_electrical_elements.inductor(1/(50*1e-06), 'H', rs=0, 'Ohm', gain
  
```



Parametrizing from datasheet

IGBT Module

F Fuji Electric

<http://www.fujielectric.com/products/semiconductor/>

6MBI450V-170-50

IGBT Modules

IGBT MODULE (V series)

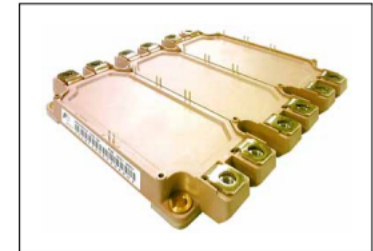
1700V / 450A / 6 in one package

■ Features

- Compact Package
- P.C.Board Mount
- Low $V_{CE(sat)}$

■ Applications

- Inverter for Motor Drive
- AC and DC Servo Drive Amplifier
- Uninterruptible Power Supply
- Industrial machines, such as welding machines



■ Maximum Ratings and Characteristics

● Absolute Maximum Ratings (at $T_c=25^\circ\text{C}$ unless otherwise specified)

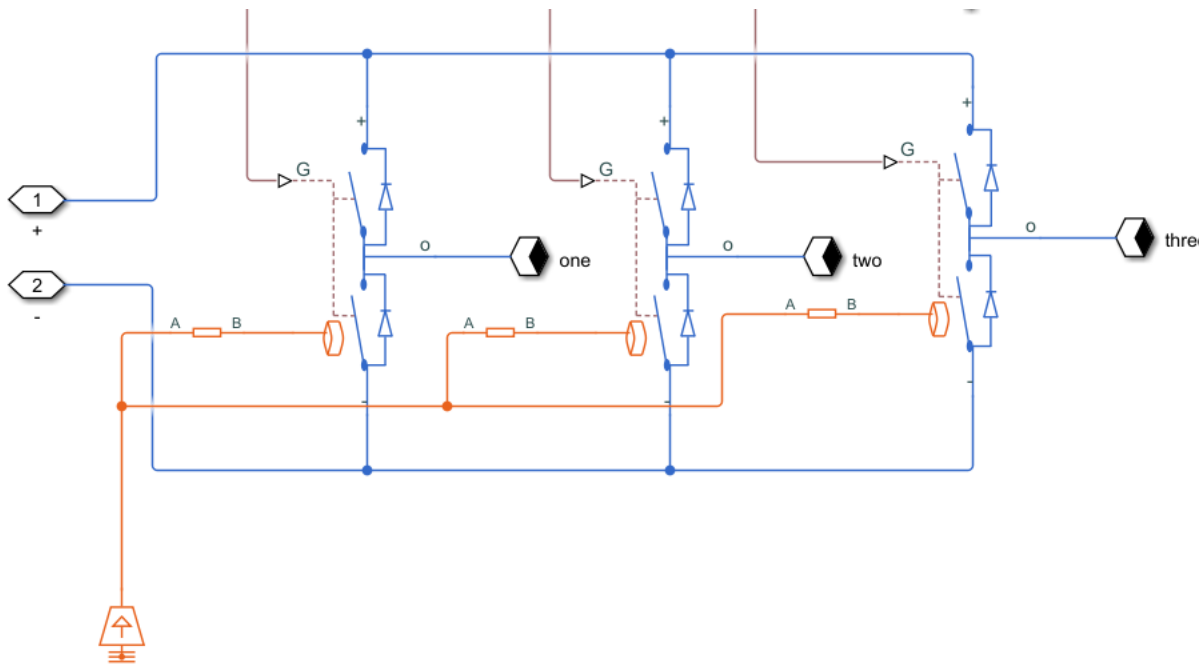
Items	Symbols	Conditions	Maximum ratings	Units
Collector-Emitter voltage	V_{CES}		1700	V
Gate-Emitter voltage	V_{GES}		± 20	V
Inverter Collector current	I_c	Continuous	600	A
	$I_{c\ pulse}$	1ms	900	
	$-I_c$		450	
	$-I_{c\ pulse}$	1ms	900	
Collector power dissipation	P_c	1 device	2500	W
Junction temperature	T_j		175	°C
Operating junction temperature (under switching conditions)	T_{op}		150	
Case temperature	T_c		125	
Storage temperature	T_{stg}		-40 ~ 125	
Isolation voltage	V_{iso}	Between terminal and copper base (*1)	AC : 1min.	3400
		Between thermistor and others (*2)		
Screw torque	Mounting (*3)		3.5	N m
	Terminals (*4)		4.5	

Note *1: All terminals should be connected together during the test.

Note *2: Two thermistor terminals should be connected together, other terminals should be connected together and shorted to base plate during the test.

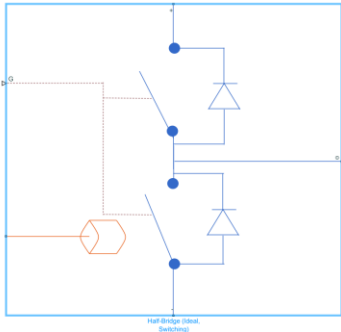
Note *3: Recommendable Value : 2.5-3.5 Nm (M5)

Note *4: Recommendable Value : 3.5-4.5 Nm (M6)

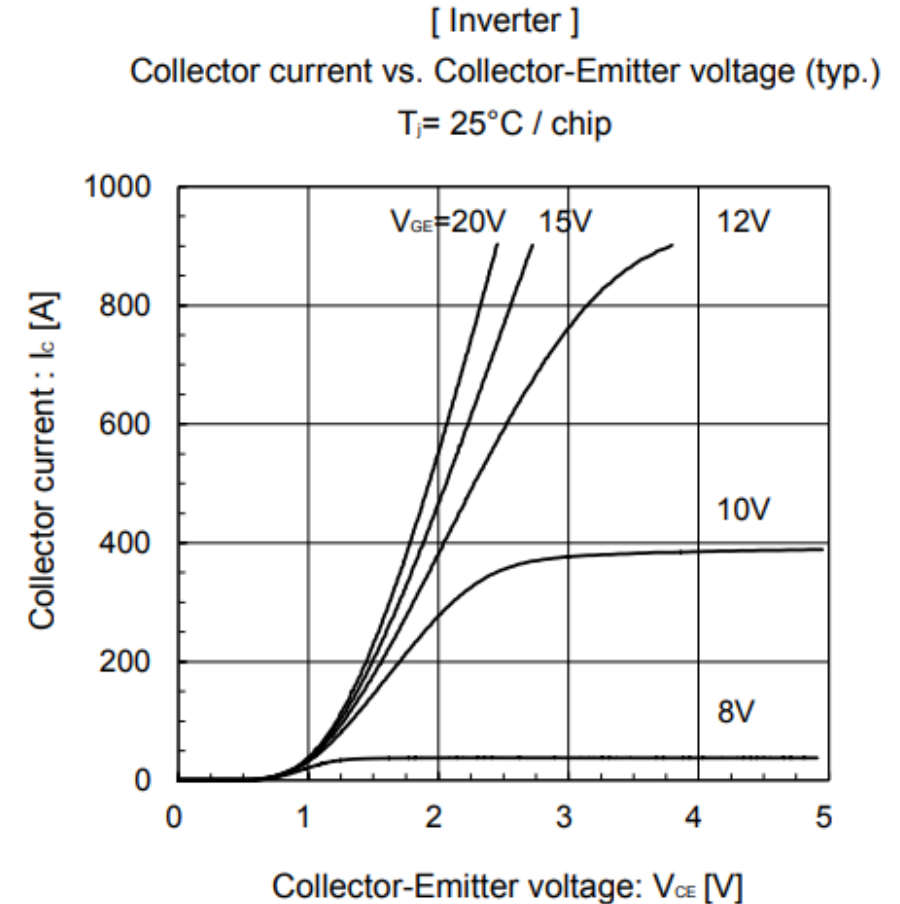


Parametrizing from datasheet

IGBT Module

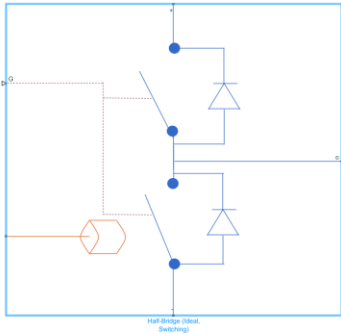


Main		
Gate-control port	PS	▼
Switching device	IGBT	▼
> Threshold voltage, V_{th}	V_{ge_th} 6.5 V	▼
On-state behavior and losses	Tabulate with temperature and current	▼
> On-state voltage, $V_{ce}(T_j, I_{ce})$	V_{ce} <2x9 double> V	▼
> Temperature vector, T_j	[25 150] degC	▼
Collector-emitter current vector, I_{ce}	I_{ds} <1x9 double> A	▼
> Off-state conductance	1e-4 1/Ohm	▼

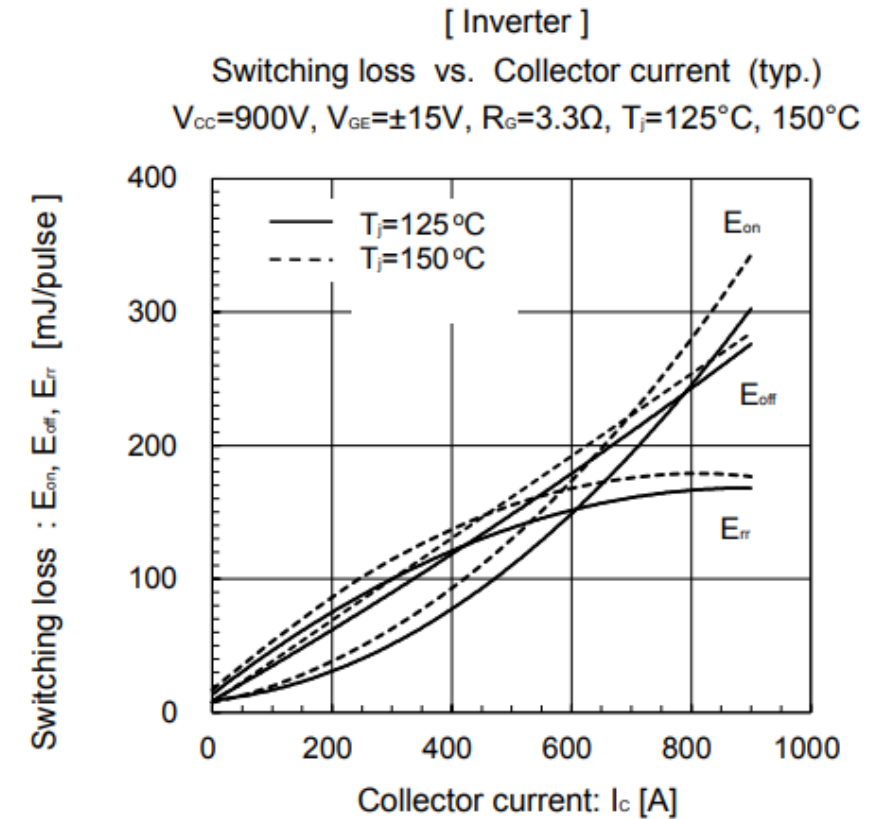


Parametrizing from datasheet

IGBT Module

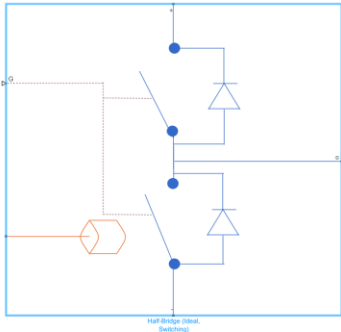


Losses				
> Switch-on loss, $E_{on}(T_j, I_{ce})$	Eon	<2x9 double>	mJ	∨
> Switch-off loss, $E_{off}(T_j, I_{ce})$	Eoff	<2x9 double>	mJ	∨
> Diode reverse recovery loss, $E_{rr}(T_j, I_{ce})$	Err	<2x9 double>	mJ	∨
> Temperature vector for losses, T_j	[125 150]		degC	∨
> Collector-emitter current vector for losses,...	Ids	<1x9 double>	A	∨
> Off-state voltage for loss data	900		V	∨



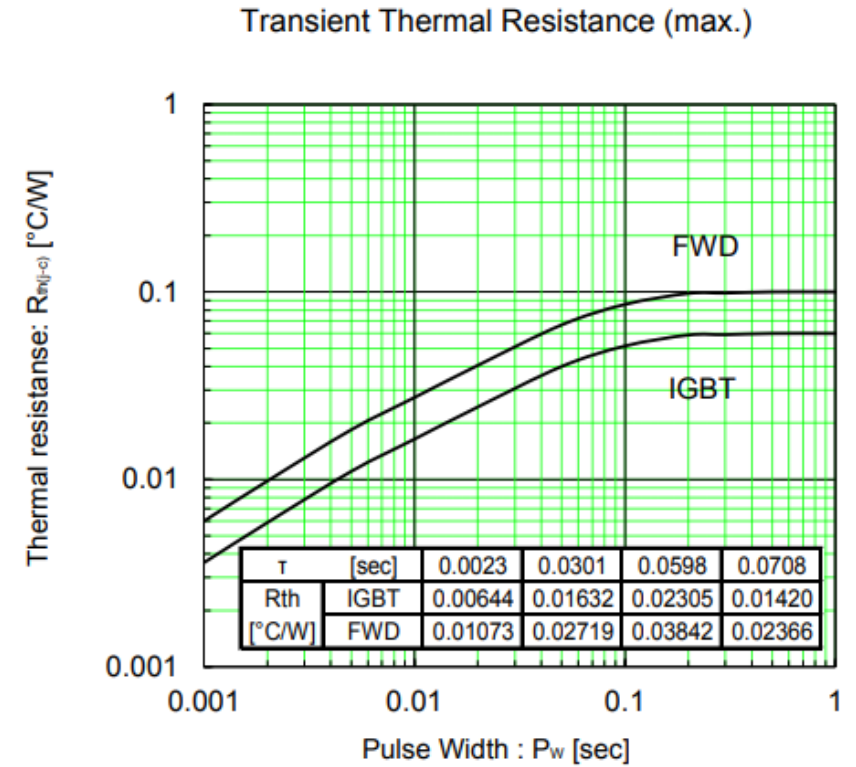
Parametrizing from datasheet

IGBT Module



▼ Thermal Port

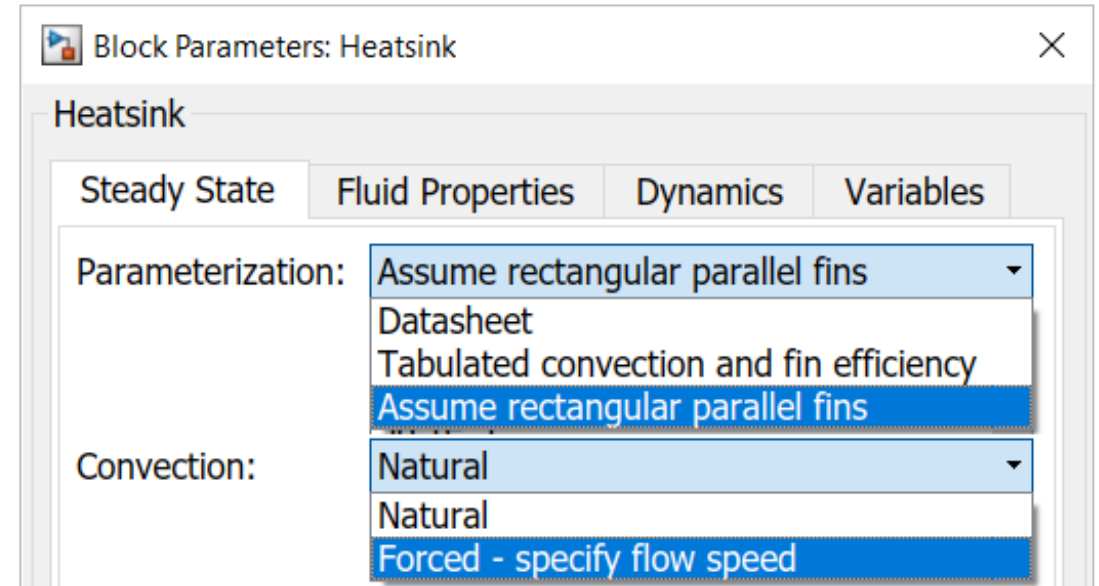
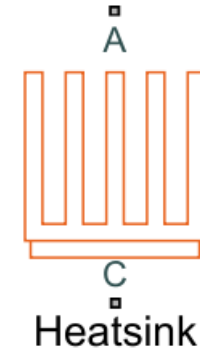
Thermal network	Cauer model	▼
Thermal resistances, [R1 R2 ... Rn]	Rt [0.00644,0.01632,0.02305,0.0...	K/W ▼
Thermal mass parameterization	By thermal time constants	▼
Thermal time constants, [t1 t2 ... tn]	tau [0.0023,0.0301,0.0598,0.0708]	s ▼
> Thermal masses initial temperatures, [T1 T...	[25, 25, 25,25]	degC ▼



FWD safe operating area (max.)
 $T_j=150^\circ\text{C}$

Heatsink Block

- Model a heatsink used to dissipate heat from power electronics
 - Model conduction through fins and convection to environment
- Flexible parameterization
 - Datasheet
 - Tabulated heat transfer properties
 - Geometry assuming an empirical convection correlation



Calculate Power Loss and Efficiency

```
>> ee_getEfficiency
>> ee_getPowerLossSummary
>> ee_getPowerLossTimeSeries
```

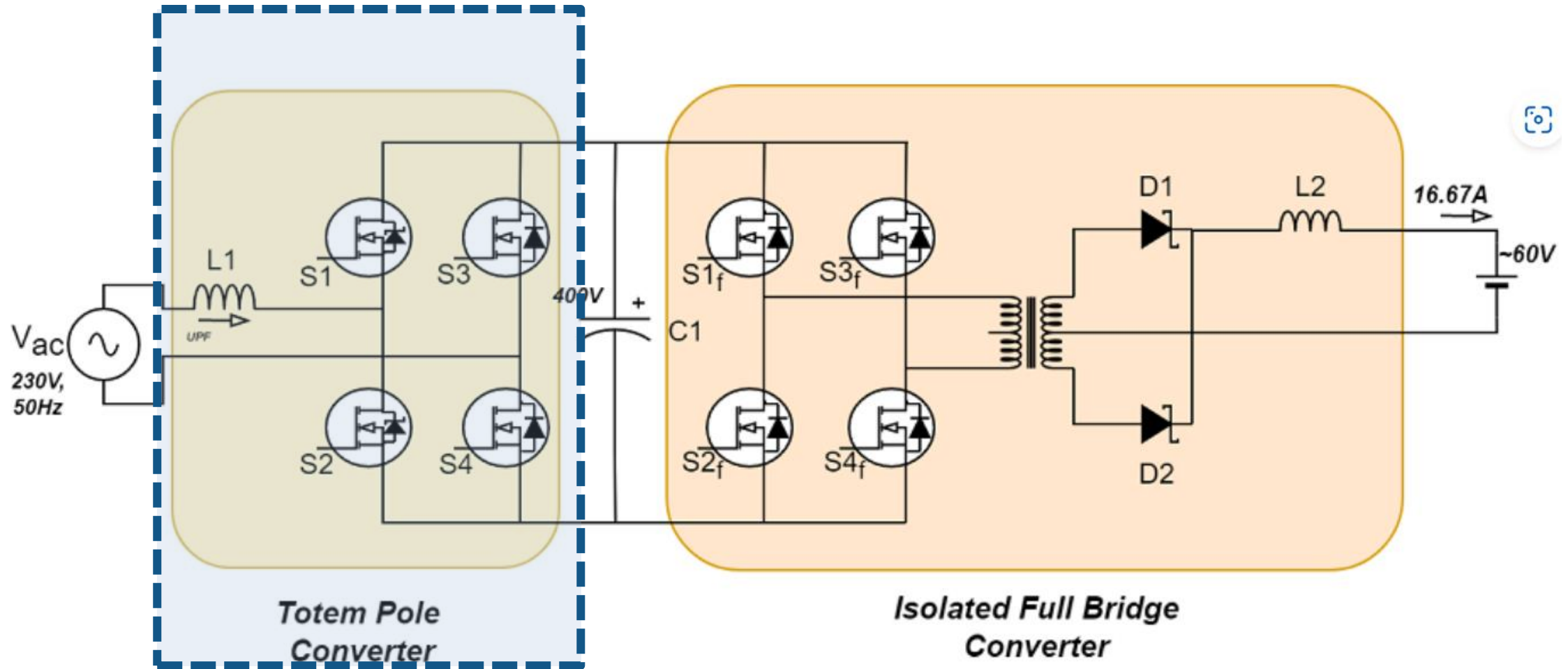
```
>> ee_getPowerLossSummary(simlog_ee_onboard_charger_two_wheeler)
```

```
ans =
```

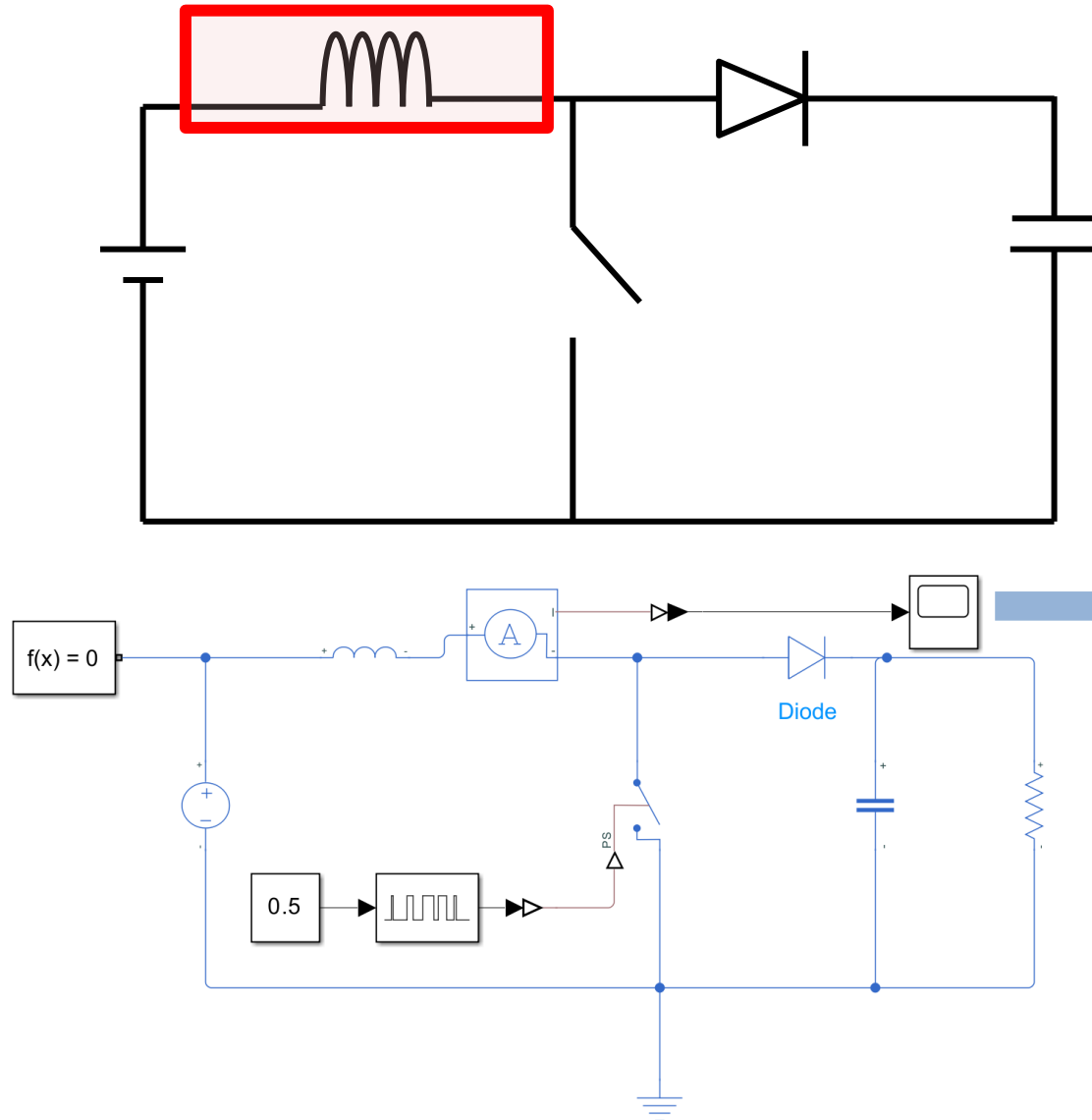
```
16×3 table
```

LoggingNode	Power	SwitchingLosses
{'ee_onboard_charger_two_wheeler.DC_DC_Converter.SwitchingConverter.TFRes1' }	74.369	0
{'ee_onboard_charger_two_wheeler.DC_DC_Converter.SwitchingConverter.DCDiode2' }	27.44	0
{'ee_onboard_charger_two_wheeler.DC_DC_Converter.SwitchingConverter.DCDiode1' }	23.925	0
{'ee_onboard_charger_two_wheeler.DC_DC_Converter.SwitchingConverter.TFRes2' }	19.237	0
{'ee_onboard_charger_two_wheeler.Totem_Pole_Converter.Converter.TPSiMOSL' }	5.2336	0
{'ee_onboard_charger_two_wheeler.Totem_Pole_Converter.Converter.TPSiMOSH' }	5.1749	0
{'ee_onboard_charger_two_wheeler.Totem_Pole_Converter.Converter.TPSiCMOSL' }	4.7752	0.56326
{'ee_onboard_charger_two_wheeler.Totem_Pole_Converter.Converter.TPSiCMOSH' }	4.6683	0.56565
{'ee_onboard_charger_two_wheeler.DC_DC_Converter.SwitchingConverter.SiC_Inverter.DCSiCS3' }	2.8173	0.63248
{'ee_onboard_charger_two_wheeler.DC_DC_Converter.SwitchingConverter.SiC_Inverter.DCSiCS1' }	2.5285	0.63496
{'ee_onboard_charger_two_wheeler.DC_DC_Converter.SwitchingConverter.SiC_Inverter.DCSiCS2' }	1.7666	0.6308
{'ee_onboard_charger_two_wheeler.DC_DC_Converter.SwitchingConverter.DCOutput_line_Inductance' }	1.6031	0
{'ee_onboard_charger_two_wheeler.DC_DC_Converter.SwitchingConverter.SiC_Inverter.DCSiCS4' }	1.4778	0.00088757
{'ee_onboard_charger_two_wheeler.Totem_Pole_Converter.Converter.TPInductor' }	0.37979	0
{'ee_onboard_charger_two_wheeler.EV_Battery.EV_Battery.Battery_Table_Based1' }	0.074153	0
{'ee_onboard_charger_two_wheeler.Totem_Pole_Converter.C' }	0.0022549	0

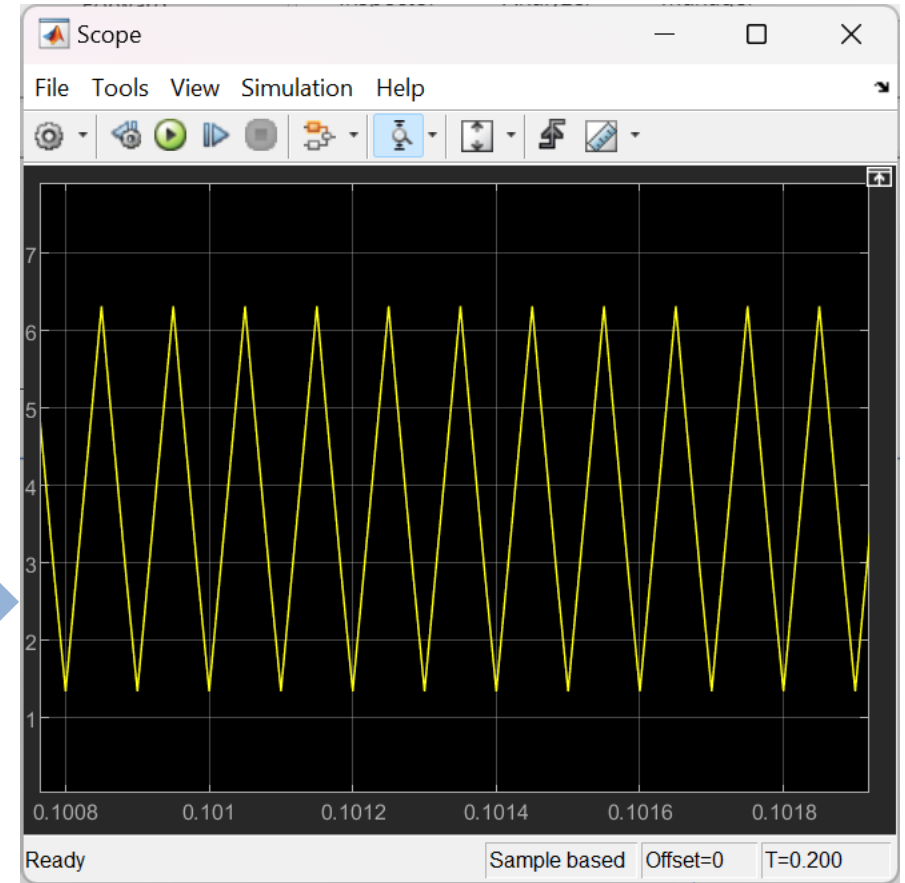
On-Board Charger for Two-Wheeler Electric Vehicle



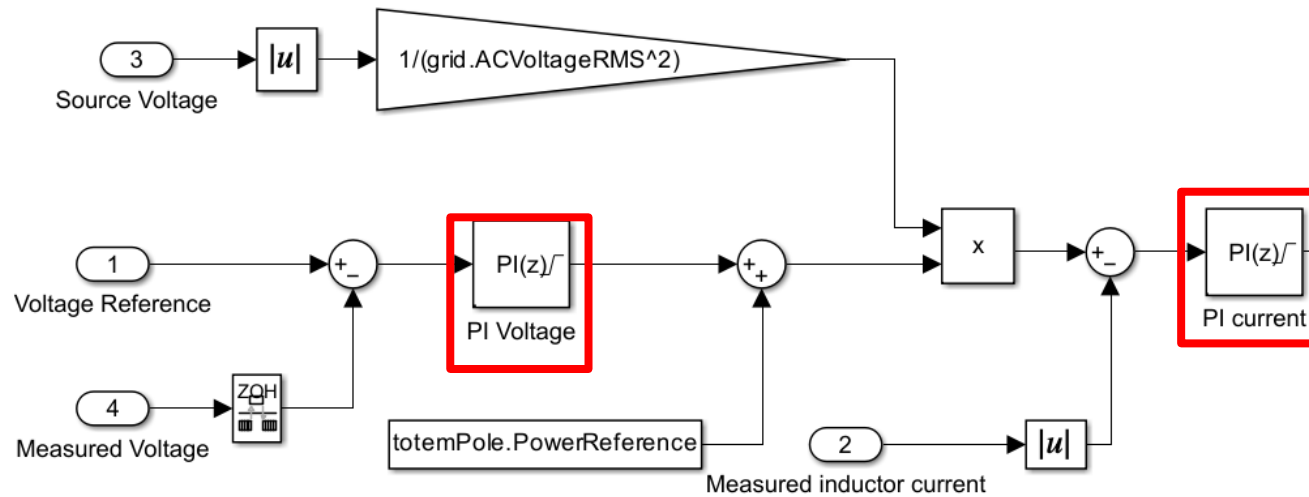
Component Sizing using Simulation



Discontinuous Conduction Mode



Totem Pole Converter for Power Factor Correction



Function Block Parameters: PID Controller1

PID Controller
 This block implements continuous- and discrete-time PID control algorithms and includes advanced features such as anti-windup, external reset, and signal tracking. You can tune the PID gains automatically using the 'Tune...' button (requires Simulink Control Design).

Controller: PID Form: Parallel

Time domain:
 Continuous-time
 Discrete-time

Discrete-time settings
 Integrator method: Forward Euler
 Filter method: Forward Euler
 Sample time (-1 for inherited): 0.001

Main PID Advanced Data Types State Attributes

Controller parameters

Proportional (P): Kp [Compensator formula](#)
 Integral (I): Ki
 Derivative (D): Kd
 Filter coefficient (N): N

$$P + I \cdot T_s \frac{1}{z-1} + D \frac{N}{1 + N T_s \frac{1}{z-1}}$$

Tune...

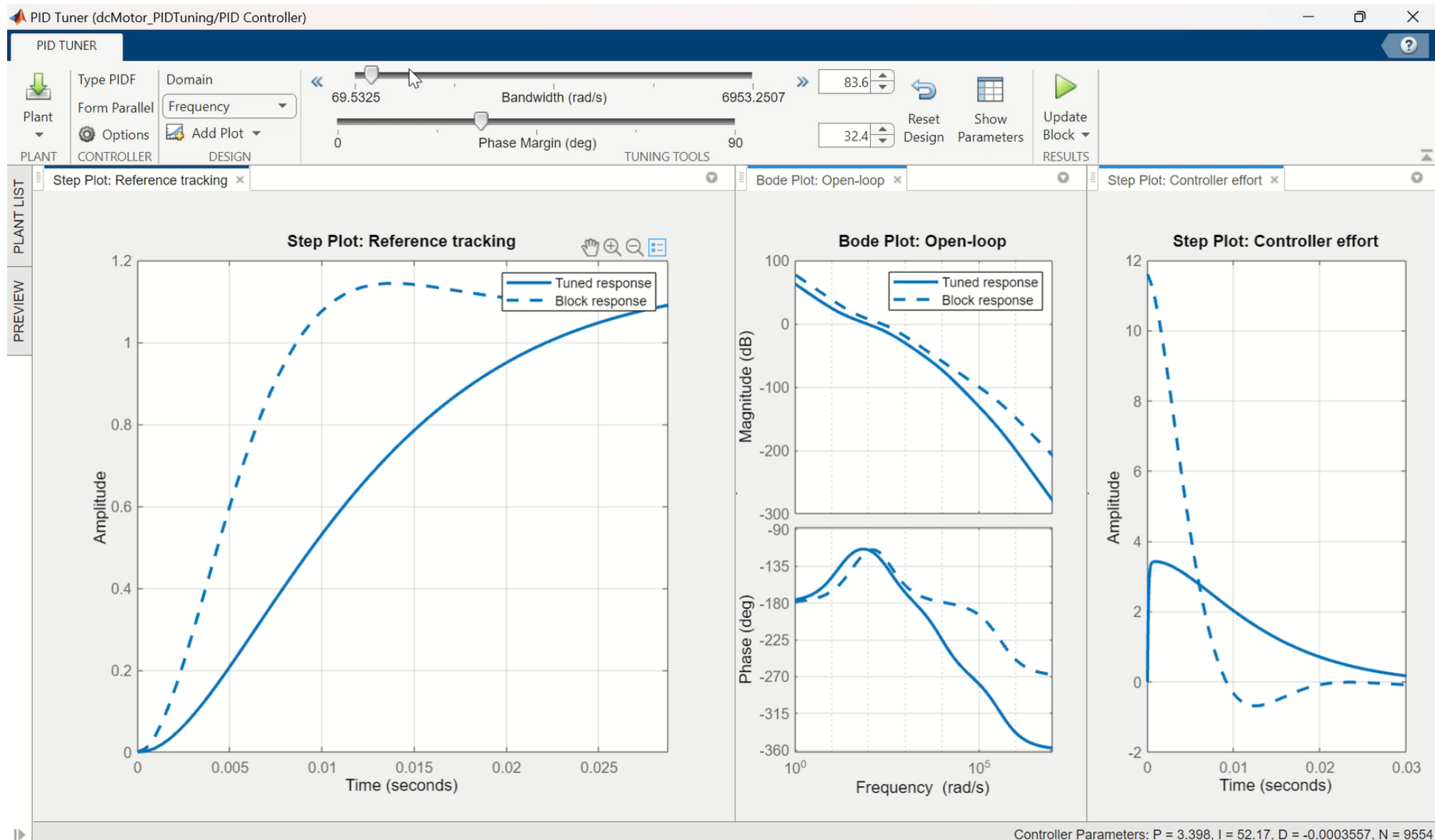
Initial conditions
 Source: internal
 Integrator: 0
 Filter: 0

External reset: none

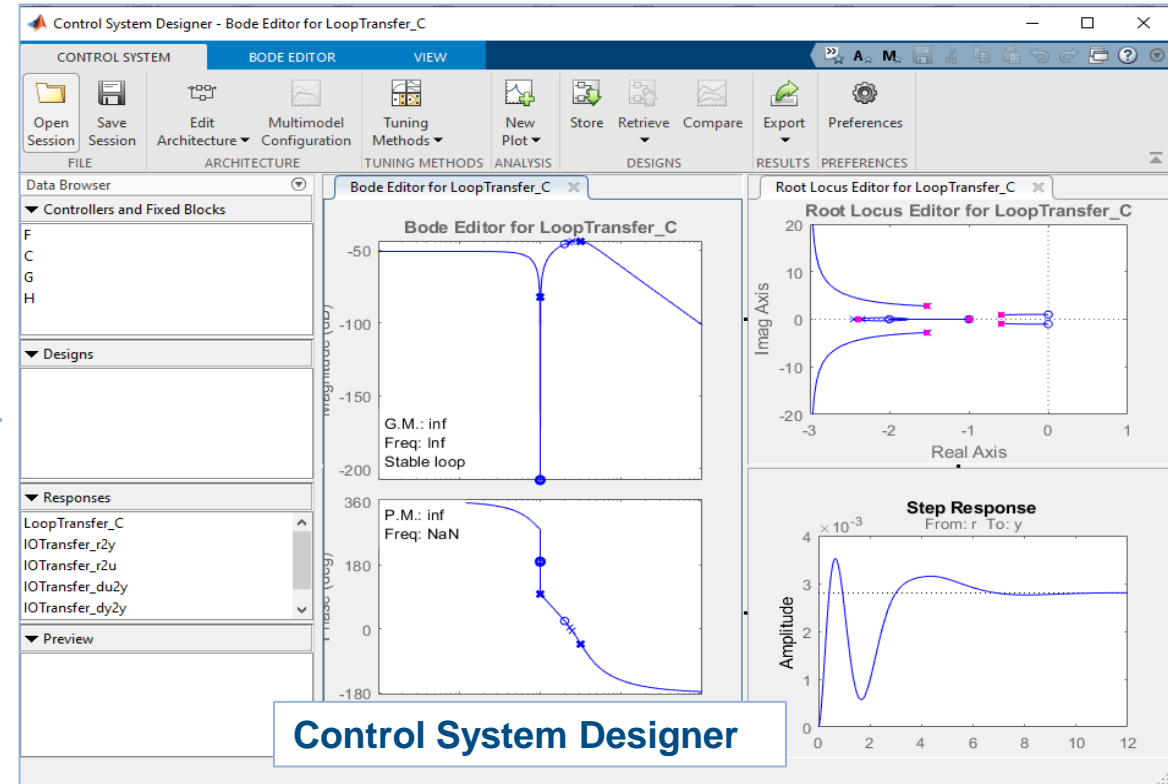
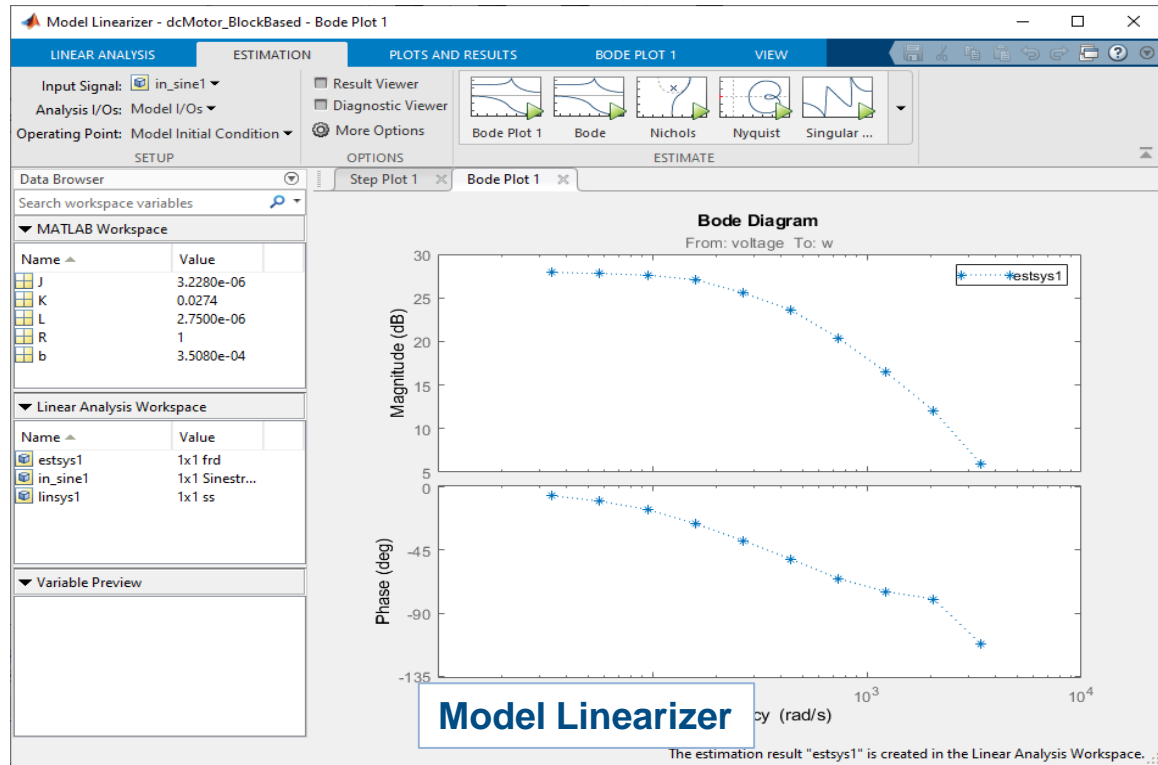
Ignore reset when linearizing
 Enable zero-crossing detection

OK Cancel Help Apply

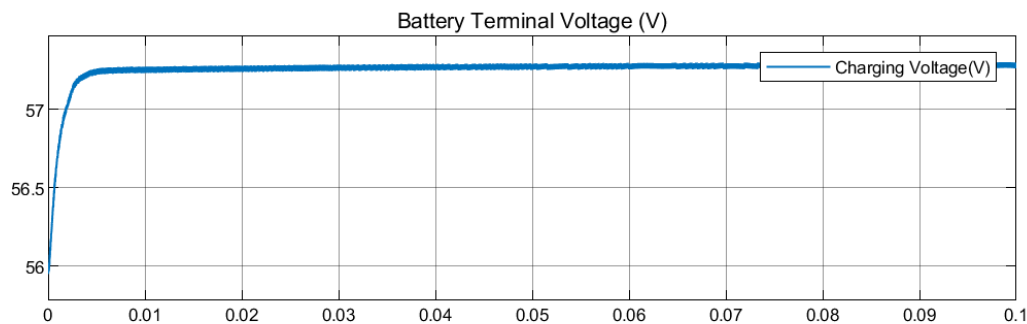
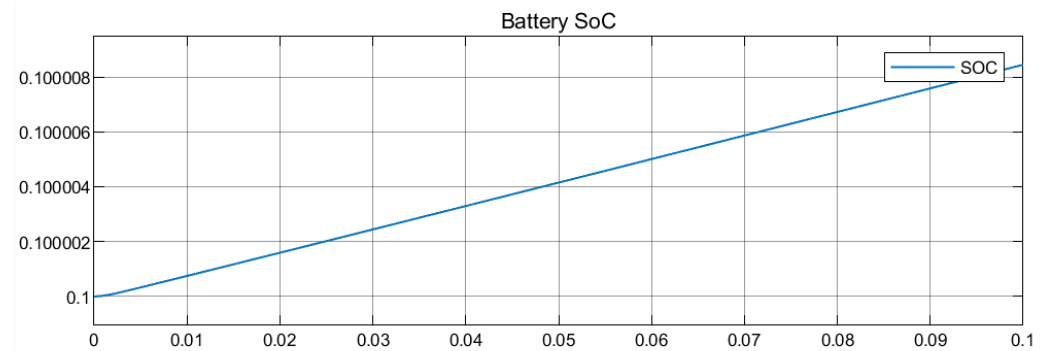
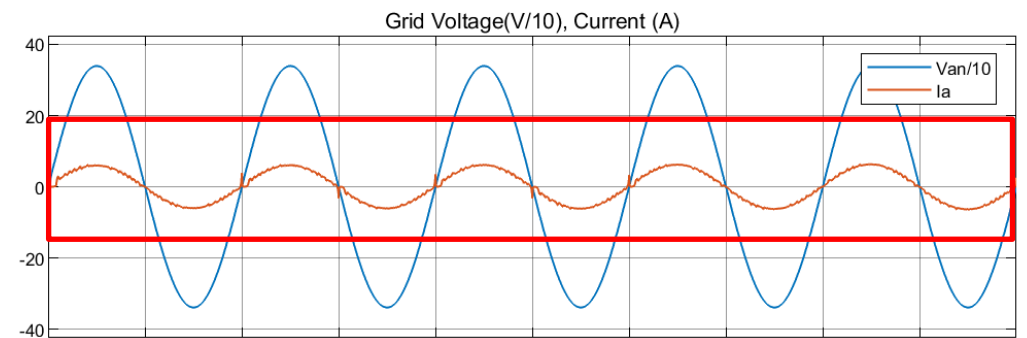
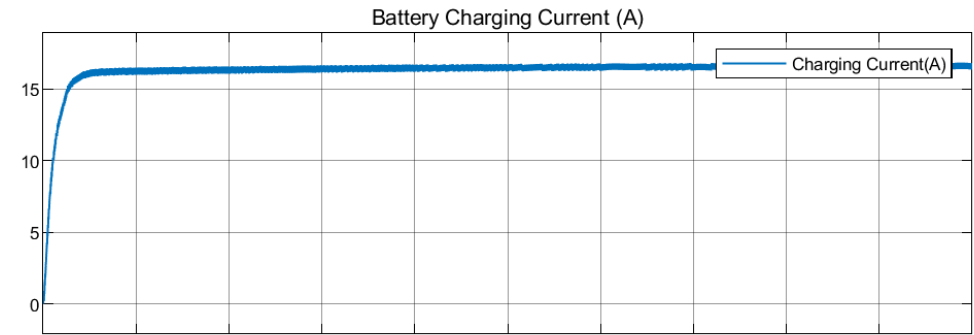
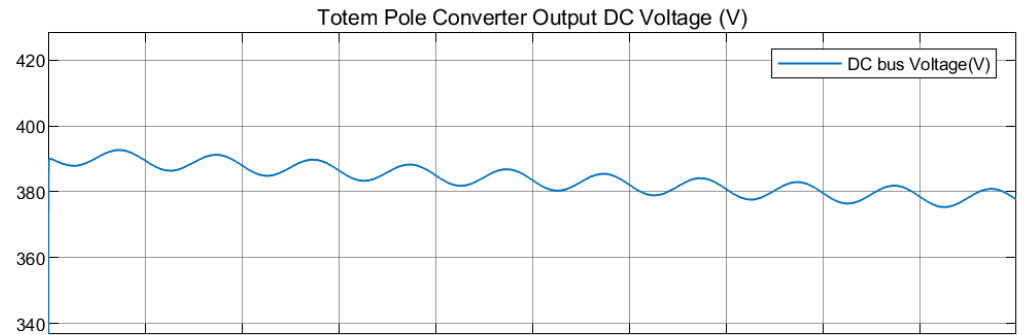
PID Tuning



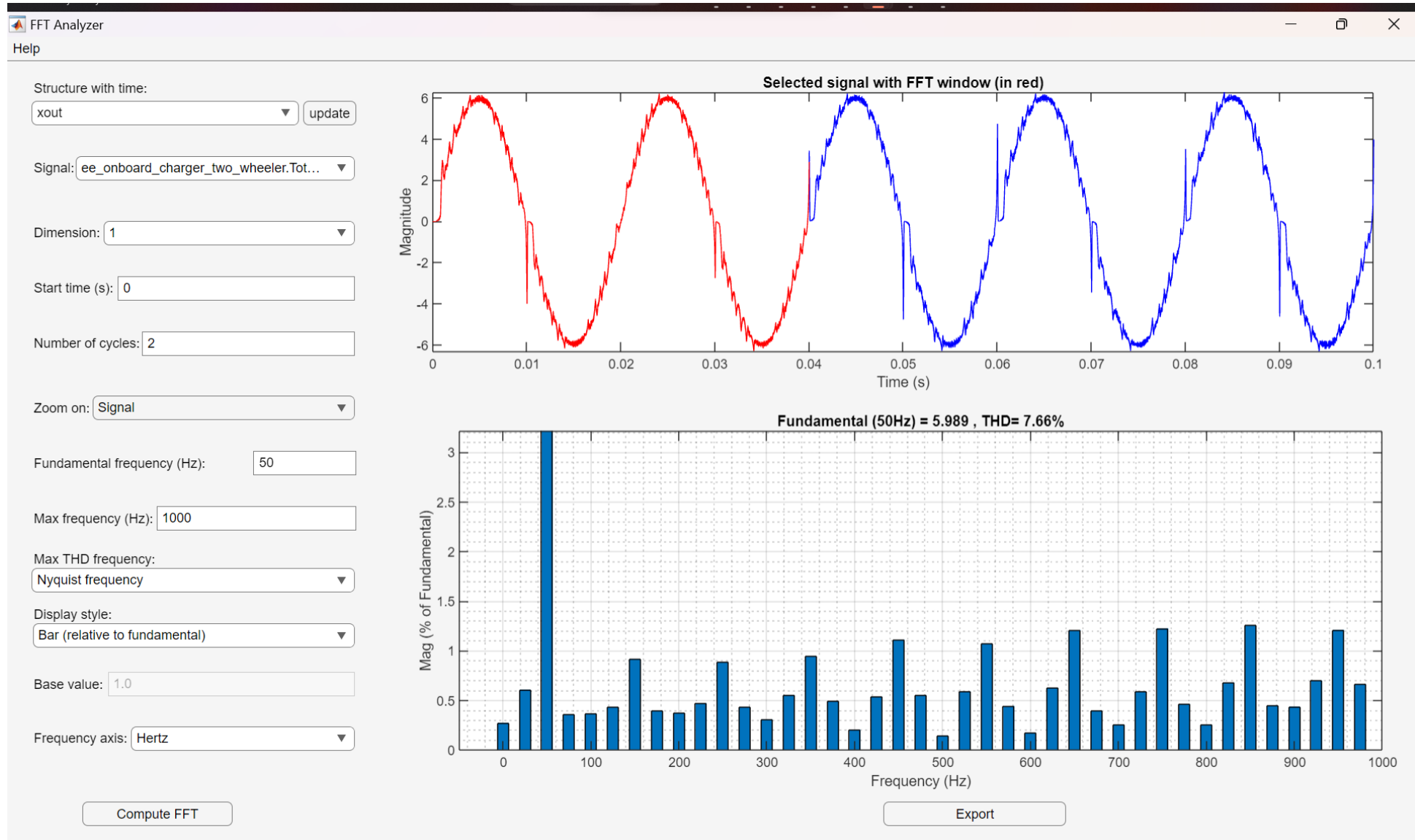
Compensator Design



Simulation Output



FFT Analyzer

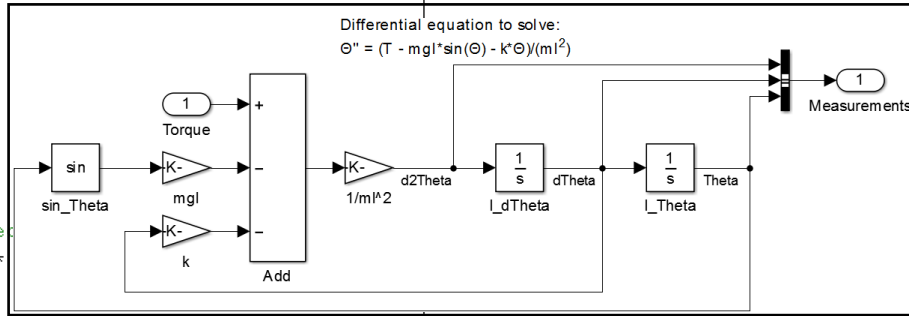


Replace Hand Coding with Code Generation

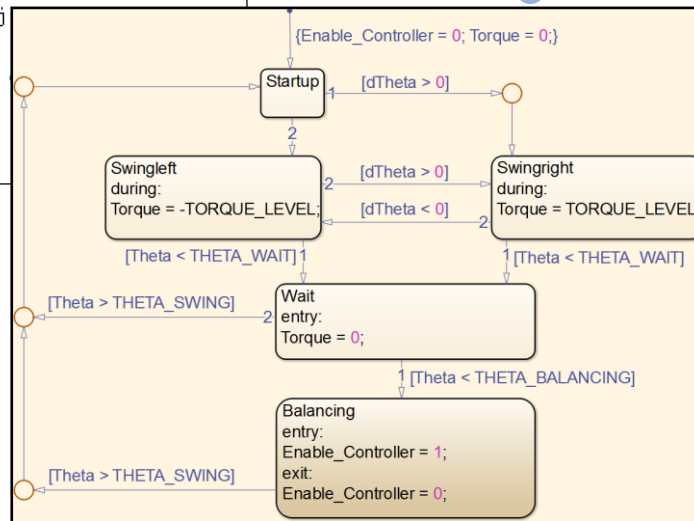
```
function [symbols, weights] = gainctrl(rxsig, train)
% 1-tap adaptive equalizer using LMS or RLS algorithm

% Equalizer settings
lambda = 0.99;
Delta = 0.1+0i;
weights = 0+0i;

for n = 1:length(rxsig)
    u = rxsig(n); % reference signal
    y = conj(weights) * u; % equalized signal
    if n<=length(train)
        d = train(n); % desired signal
    else
        d = detect(real(y)) + 1j*detect(imag(y)); % error signal
    end
    % Single-tap RLS
    Delta = 1/(lambda/Delta + u*conj(u));
    G = Delta * u;
    e = d - y; % symbol estimation error
    weights = weights + G*conj(e);
    symbols(n) = y;
end
```



Simulink



Stateflow

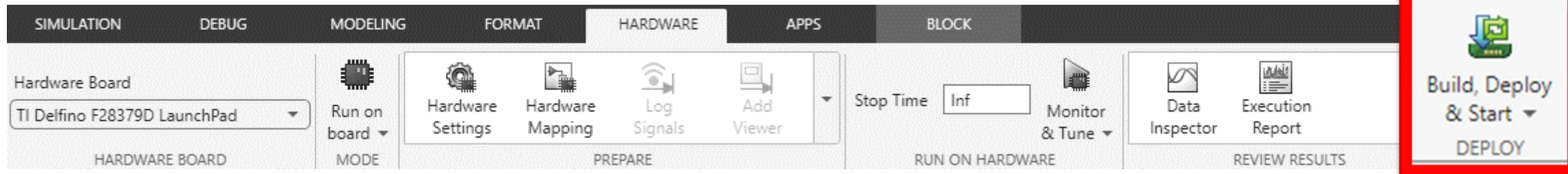
MATLAB

Optimizations



C Code

Include TI C2000 Driver Blocks

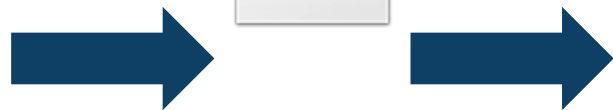
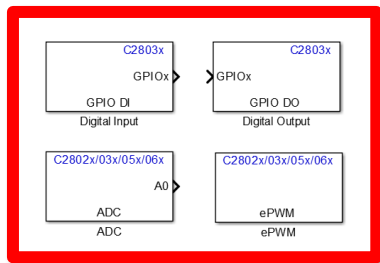
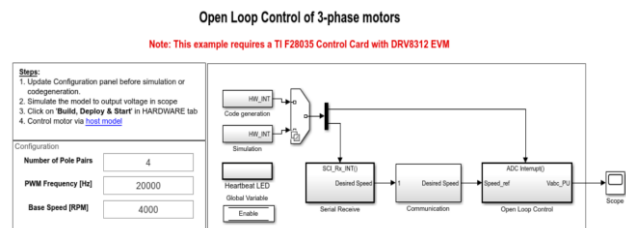


Simulink Model

Embedded Coder

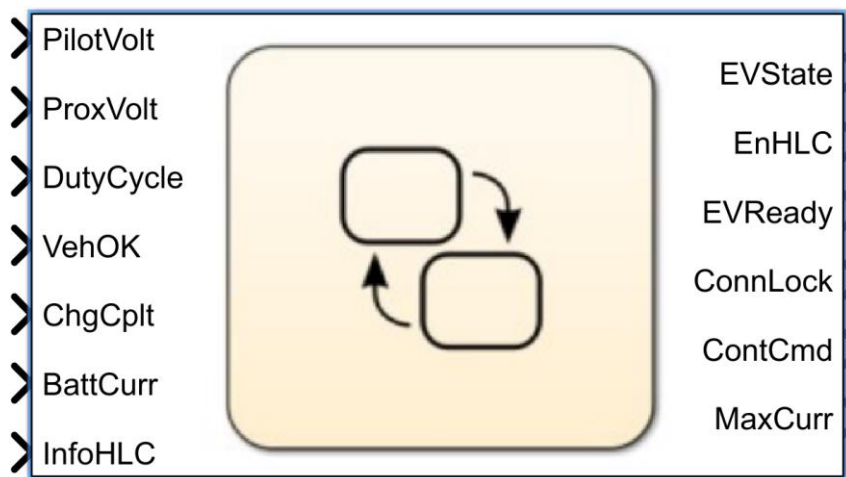
CCS Project

Run on C2000

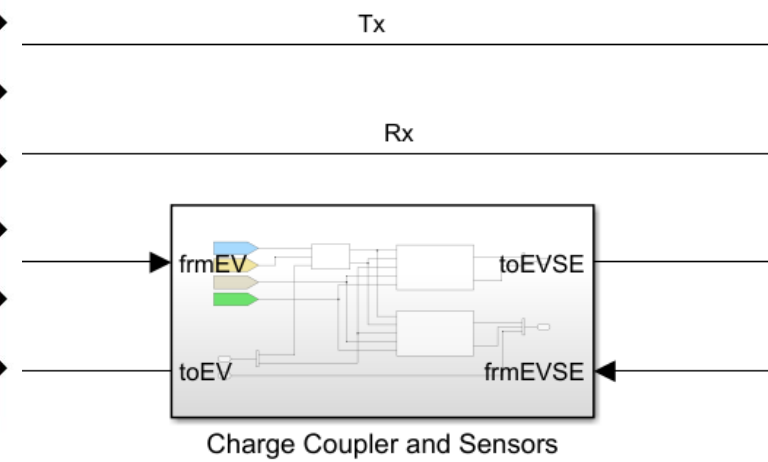


Test Digital Communication Protocols

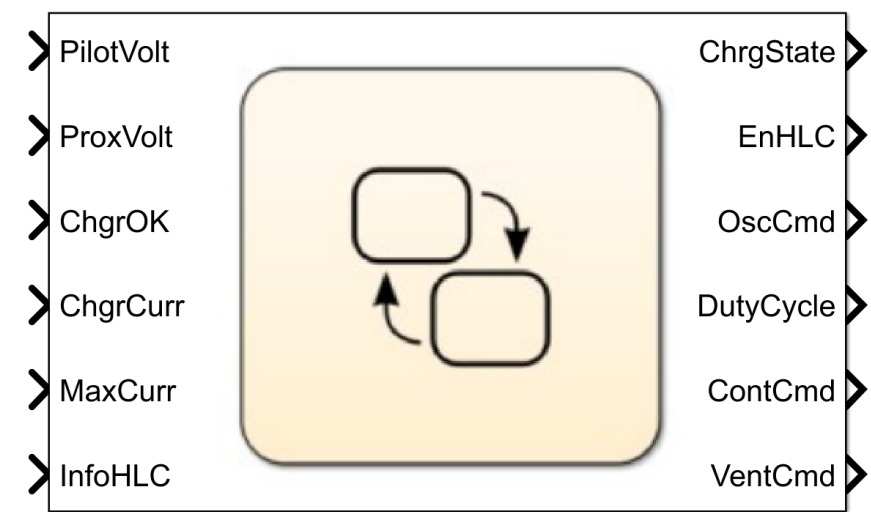
SAE J1772 protocol for basic charging and ISO15118 protocol for high level communication (HLC)



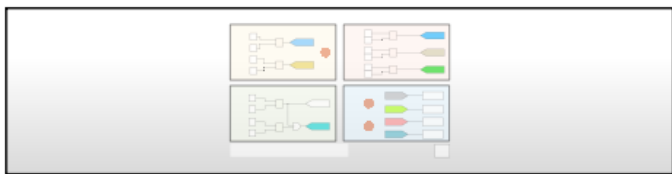
SAE J1772 DC Charging (EV)



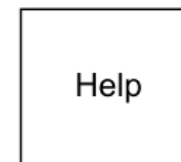
Charge Coupler and Sensors



SAE J1772 DC Charging (EVSE)

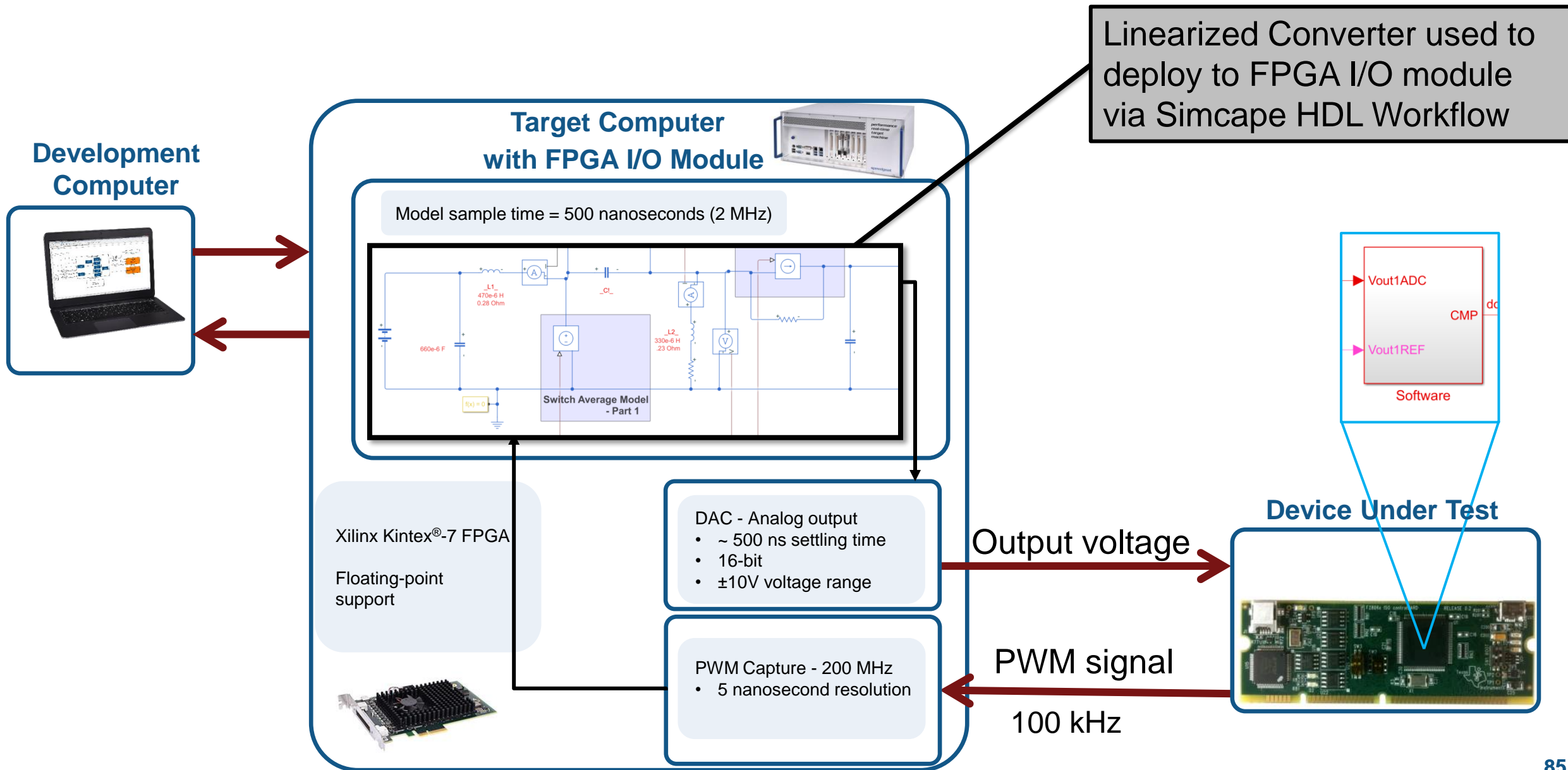


Dashboard

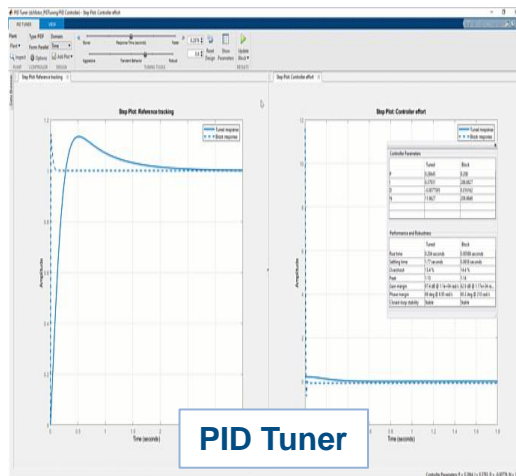
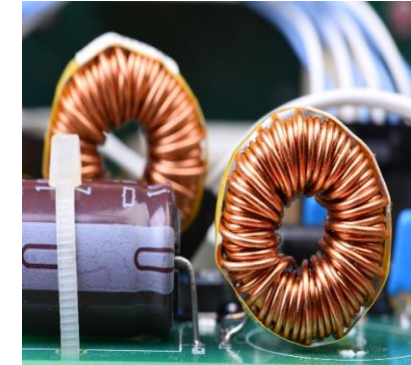
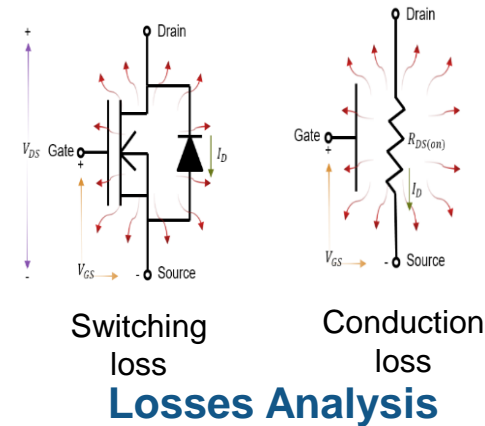
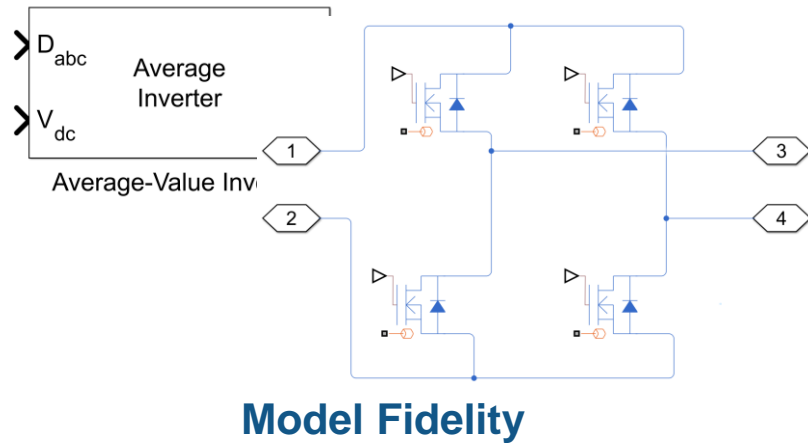


Help

Hardware-in-the-Loop Simulation of Power Converter



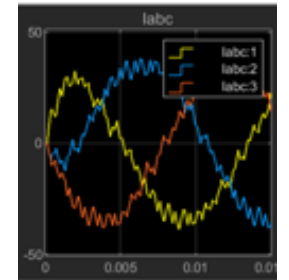
Power Converter Control Design Workflow Tasks



Controller Design

The screenshot shows a software interface for embedded deployment. It features a 'Model' window with a 'CONTROLLER' block. To the right, there is a code editor showing C code for a controller. The code includes comments and function definitions for a step function and a controller. The code is for a C64x DSP, as indicated by the 'C64x DSP' logo and the 'TEXAS INSTRUMENTS' logo.

One-Click Embedded Deployment

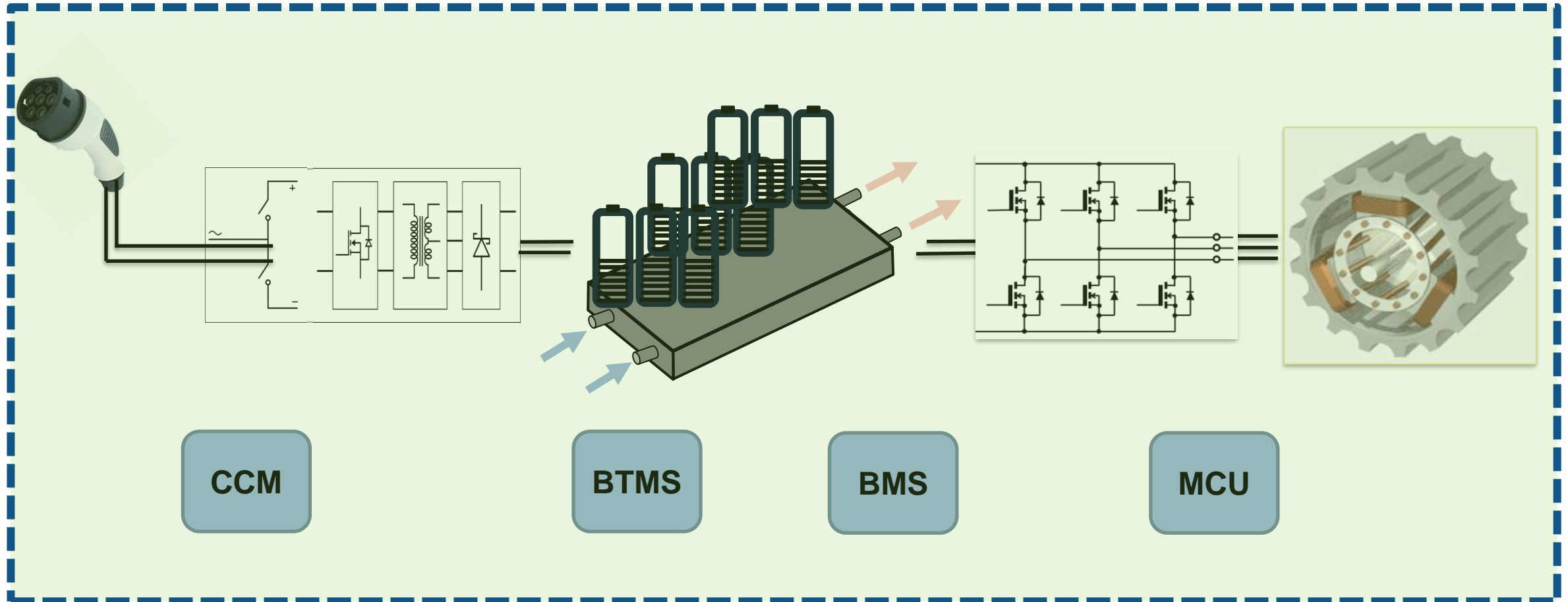


Challenges for Power Electronics Engineer

- Understand the impact of the power source and load
- Testing for a complete range of operating and fault conditions
- Designing and implementing digital controls using *only* SPICE simulator tools
- Catching errors during software-hardware integration testing
- Compliance to industry standards
- Development Time



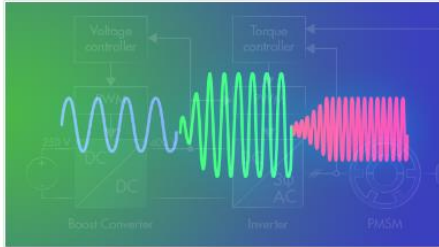
Electric Vehicle System



Key Takeaways

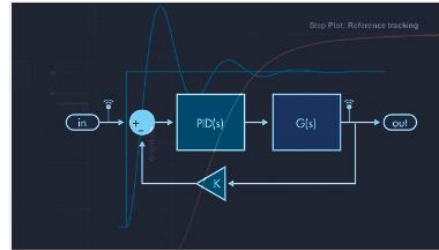
- **Speeding up journey from an idea to implementation!**
- **Iterating on new ideas faster using Model Based Design**
 - Design with simulation
 - Prototype on real-time hardware
 - Generate code for production
- **Varying model fidelity based on your needs**

Enable Your Team For Electrification



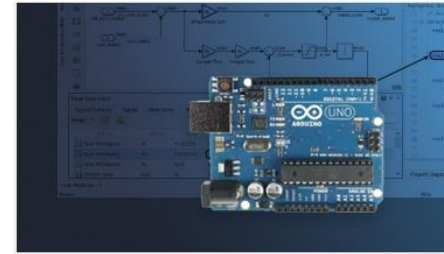
Power Electronics Control Design with Simulink and Simscape

Learn to model power electronic systems in the Simulink environment using Simscape Electrical™ and to design control with Simulink Control Design.



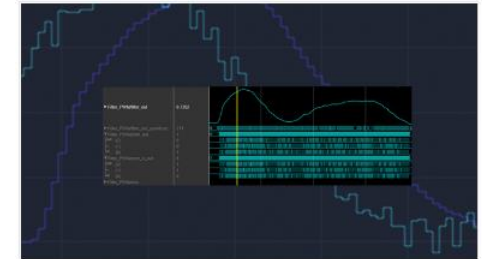
Control System Design with MATLAB and Simulink

Learn to design and model control systems with Simulink. Topics include system identification, parameter estimation, control system analysis, and response optimization.



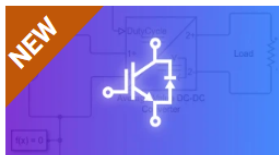
Embedded Coder for Production Code Generation

Develop Simulink models for deployment in embedded systems. Topics include code structure and execution, code generation options and optimizations, and deploying code to target hardware.



Generating HDL Code from Simulink

Learn to prepare Simulink models for HDL code generation, generate HDL code and testbench for a compatible Simulink model, and perform speed and area optimizations.



Power Electronics Simulation Onramp

5 modules | 1 hour | [Languages](#)

Learn the basics of simulating power electronics converters in Simscape.



Circuit Simulation Onramp

7 modules | 2 hours | [Languages](#)

Learn the basics of simulating electrical circuits in Simscape.

Visit the Power Electronics Control Community on MATLAB Central to find Models, Answers, and How-to Videos

<https://www.mathworks.com/matlabcentral/topics/power-electronics-control.html>



MathWorks Videos

Suggest a video

<



Understanding Field-Oriented Control | Motor Control, Part 4



Reinforcement Learning for Developing Field-Oriented Control



Motor Control, Part 3: BLDC Speed Control Using PWM

>

» View all MathWorks videos

Answers

Ask a question

AD How to generate first value for Kp, Ki of PID in current controller
 Latest activity by adhavan d on 7 Jul 2023 at 5:09
 Tags: power_electronics_control, electric_motor_control, control system
 25 views 0 votes 1 comment

1
answer

Community Videos






Welcome to the Power Electronics Control Community

Moderator:
Tony Lennon

The MathWorks community for engineers using Simulink to apply power electronics control to Electric Vehicles, Renewable Energy, Battery Systems, Power Conversion, and Motor Control. Learn from examples and videos submitted by your peers and MathWorks engineers.

Files



Electric Aircraft Model in Simscape

by Steve Miller on 5 Jul 2023 at 17:40

Tags: emissions, more electric aircraft, physical modeling, battery_system_management, power_electronics_control


21 Downloads (30 days)

★★★★★


Discussions

PR Electric vehicle thermal management


Top Community Contributors




Mohsen Aleenejad




Dakai Hu




Jonathan LeSage



Graham Dudgeon



Gernot Schrabberger



Joel Van Sickle

Additional Resources

- Hardware Support
- Hardware Support for Texas Instruments Microcontrollers
- Hardware Support for NXP
- Hardware Support for ARM
- Topics in Power Electronics
- Power Electronics Control Design with Simulink

Start a discussion

MATLAB EXPO

Thank you



© 2023 The MathWorks, Inc. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See [mathworks.com/trademarks](https://www.mathworks.com/trademarks) for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.