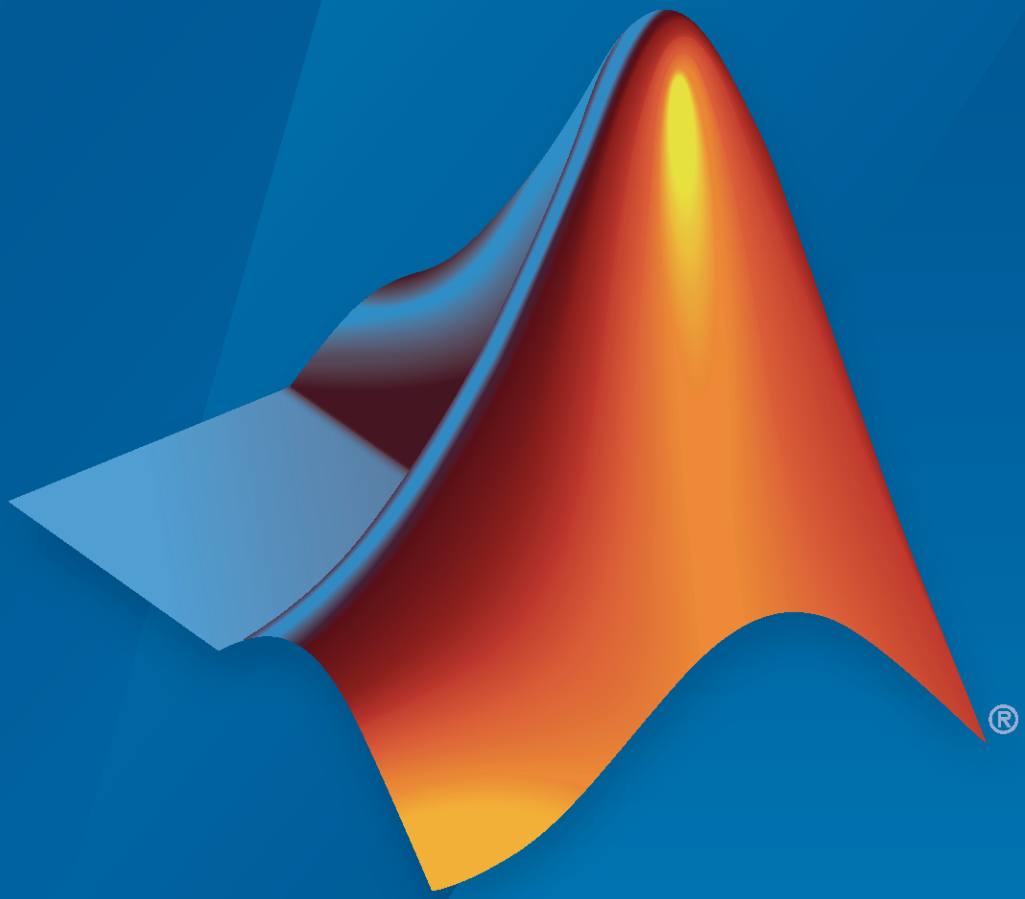
# Simscape™ Electrical™

User's Guide



MATLAB® & SIMULINK®

R2022b

MathWorks®

# How to Contact MathWorks

Latest news: `www.mathworks.com`

Sales and services: `www.mathworks.com/sales_and_services`

User community: `www.mathworks.com/matlabcentral`

Technical support: `www.mathworks.com/support/contact_us`

Phone: 508-647-7000

The MathWorks, Inc.
1 Apple Hill Drive
Natick, MA 01760-2098

*Simscape™ Electrical™ User's Guide*

© COPYRIGHT 2013–2022 by The MathWorks, Inc.

The software described in this document is furnished under a license agreement. The software may be used or copied only under the terms of the license agreement. No part of this manual may be photocopied or reproduced in any form without prior written consent from The MathWorks, Inc.

FEDERAL ACQUISITION: This provision applies to all acquisitions of the Program and Documentation by, for, or through the federal government of the United States. By accepting delivery of the Program or Documentation, the government hereby agrees that this software or documentation qualifies as commercial computer software or commercial computer software documentation as such terms are used or defined in FAR 12.212, DFARS Part 227.72, and DFARS 252.227-7014. Accordingly, the terms and conditions of this Agreement and only those rights specified in this Agreement, shall pertain to and govern the use, modification, reproduction, release, performance, display, and disclosure of the Program and Documentation by the federal government (or other entity acquiring for or through the federal government) and shall supersede any conflicting contractual terms or conditions. If this License fails to meet the government's needs or is inconsistent in any respect with federal procurement law, the government agrees to return the Program and Documentation, unused, to The MathWorks, Inc.

**Trademarks**

MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See `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.

**Patents**

MathWorks products are protected by one or more U.S. patents. Please see `www.mathworks.com/patents` for more information.

**Revision History**

| | | |
|---|---|---|
| September 2013 | Online only | New for Version 6.0 (Release 2013b) |
| March 2014 | Online only | Revised for Version 6.1 (Release 2014a)<br>(Renamed from *SimPowerSystems™ User's Guide (Third Generation)*) |
| October 2014 | Online only | Revised for Version 6.2 (Release 2014b) |
| March 2015 | Online only | Revised for Version 6.3 (Release 2015a) |
| September 2015 | Online only | Revised for Version 6.4 (Release 2015b) |
| March 2016 | Online only | Revised for Version 6.5 (Release 2016a)<br>(Renamed from *SimPowerSystems™ User's Guide (Simscape™ Components)*) |
| September 2016 | Online only | Revised for Version 6.6 (Release 2016b) |
| March 2017 | Online only | Revised for Version 6.7 (Release 2017a) |
| September 2017 | Online only | Revised for Version 6.8 (Release 2017b) |
| March 2018 | Online only | Revised for Version 6.9 (Release 2018a) |
| September 2018 | Online only | Revised for Version 7.0 (Release 2018b)<br>(Renamed from *Simscape™ Power Systems™ User's Guide (Simscape™ Components)* and *Simscape™ Electronics™ User's Guide* ) |
| March 2019 | Online only | Revised for Version 7.1 (Release 2019a)<br>(Renamed from *Simscape™ Electrical™ User's Guide (Electronics, Mechatronics, and Power Systems)*) |
| September 2019 | Online only | Revised for Version 7.2 (Release 2019b) |
| March 2020 | Online only | Revised for Version 7.3 (Release 2020a) |
| September 2020 | Online only | Revised for Version 7.4 (Release 2020b) |
| March 2021 | Online only | Revised for Version 7.5 (Release 2021a) |
| September 2021 | Online only | Revised for Version 7.6 (Release 2021b) |
| March 2022 | Online only | Revised for Version 7.7 (Release 2022a) |
| September 2022 | Online only | Revised for Version 7.8 (Release 2022b) |

# Contents

## 1 Getting Started


**Simscape Electrical Product Description** . . . . . 1-2
Key Features . . . . . 1-2

**Simscape Electrical Block Libraries** . . . . . 1-3
Libraries Compatible with Simscape Technology . . . . . 1-3
Specialized Power Systems Library . . . . . 1-4
Access the Simscape Electrical Block Libraries . . . . . 1-4

**Modeling Analog Circuit Architectures, Mechatronic Systems, and Electrical Power Systems Using Simscape Electrical** . . . . . 1-6
Simulink Templates for Modeling with Simscape Electrical . . . . . 1-6
Simscape Electrical Blocks and Ports . . . . . 1-7
Machine and Transformer Source Code Examples . . . . . 1-7
Plotting and Display Options for Asynchronous and Synchronous Machines . . . . . 1-7
Choosing the Right Simscape Electrical Technology . . . . . 1-7
Assumptions and Limitations . . . . . 1-8

**Per-Unit System of Units** . . . . . 1-9
What Is the Per-Unit System? . . . . . 1-9
Example 1: Three-Phase Transformer . . . . . 1-10
Example 2: Asynchronous Machine . . . . . 1-11
Base Values for Instantaneous Voltage and Current Waveforms . . . . . 1-12
Why Use the Per-Unit System Instead of the Standard SI Units? . . . . . 1-12


## 2 Tutorials


**Build and Simulate Composite and Expanded Three-Phase Models** . . . . . 2-2
Select System Component Blocks and Build a Resistive Three-Phase Model . . . . . 2-2
Specify Simulation Parameters . . . . . 2-4
Load Impedance Parameters . . . . . 2-4
Specify Display Parameters . . . . . 2-5
Simulate and Analyze the Resistive Three-Phase Model . . . . . 2-5
Simulate and Analyze a Reactive Three-Phase Model . . . . . 2-6
Create an Expanded Balanced Three-Phase Model . . . . . 2-6
Create an Expanded Unbalanced Three-Phase Model . . . . . 2-7
Simulate the Expanded Balanced and Unbalanced Models and Analyze the Results . . . . . 2-7

**DC Motor Model** . . . . . **2-10**
Select Blocks to Represent System Components . . . . . **2-10**
Build the Model . . . . . **2-10**
Specify Model Parameters . . . . . **2-12**
Configure the Solver Parameters . . . . . **2-14**
Run the Simulation and Analyze the Results . . . . . **2-14**

**Build and Simulate a Simple Circuit** . . . . . **2-16**
Select Blocks to Represent System Components . . . . . **2-16**
Build the Model . . . . . **2-17**
Specify Model Parameters . . . . . **2-18**
Configure the Solver Parameters . . . . . **2-20**
Simulate Model and Analyze Results . . . . . **2-21**

**Build and Simulate a Single-Phase Half-Bridge Inverter with Ideal Switches** . . . . . **2-22**
Build the Model . . . . . **2-22**
Specify Model Parameters . . . . . **2-26**
Configure the Solver Parameters . . . . . **2-28**
Simulate the Model and Analyze the Results . . . . . **2-29**

**Build and Simulate Single-Phase Half-Bridge Inverter With Ideal Switches and Thermal Port** . . . . . **2-33**
Build the Model . . . . . **2-33**
Specify Model Parameters . . . . . **2-37**
Configure the Solver Parameters . . . . . **2-40**
Simulate the Model and Analyze the Results . . . . . **2-40**

**Multi-Domain Examples on GitHub** . . . . . **2-46**

## 3 Modeling and Simulation Basics

**Essential Electrical Modeling Techniques** . . . . . **3-2**
Overview of Modeling Rules . . . . . **3-2**
Required Blocks . . . . . **3-3**
Creating a New Model . . . . . **3-3**
Modeling Instantaneous Events . . . . . **3-3**
Using Simulink Blocks to Model Physical Components . . . . . **3-4**

**Three-Phase Ports** . . . . . **3-5**
About Three-Phase Ports . . . . . **3-5**
Expand and Collapse Three-Phase Ports on a Block . . . . . **3-6**

**Switch Between Physical Signal and Electrical Ports** . . . . . **3-7**

**Simulating an Electronic, Mechatronic, or Electrical Power System** . . . . **3-8**
Selecting a Solver . . . . . **3-8**
Specifying Simulation Accuracy/Speed Tradeoff . . . . . **3-8**
Avoiding Simulation Issues . . . . . **3-9**
Running a Time-Domain Simulation . . . . . **3-9**
Running a Small-Signal Frequency-Domain Analysis . . . . . **3-9**

**Selecting the Output Model for Logic Blocks** . . . . . **3-10**
Available Output Models . . . . . **3-10**
Quadratic Model Output and Parameters . . . . . **3-11**

**Parameterizing Blocks from Datasheets** . . . . . **3-13**

**Parameterize a Piecewise Linear Diode Model from a Datasheet** . . . . . **3-14**

**Parameterize an Exponential Diode from a Datasheet** . . . . . **3-17**

**Parameterize an Exponential Diode from SPICE Netlist** . . . . . **3-21**

**Parameterize an Op-Amp from a Datasheet** . . . . . **3-25**

**Additional Parameterization Workflows** . . . . . **3-27**
Validation Using Data from SPICE Tool . . . . . **3-27**
Parameter Tuning Against External Data . . . . . **3-27**
Building an Equivalent Model of a SPICE Netlist . . . . . **3-27**

**Simulating Thermal Effects in Rotational and Translational Actuators** . . . . . **3-28**
Using the Thermal Ports . . . . . **3-28**
Thermal Model for Actuator Blocks . . . . . **3-29**

**Simulating Thermal Effects in Semiconductors** . . . . . **3-31**
Using the Thermal Ports . . . . . **3-31**
Specify junction and thermal case parameters . . . . . **3-32**
Cauer Model . . . . . **3-33**
Cauer Model Parameterized with Foster Coefficients . . . . . **3-34**
External Model . . . . . **3-35**
Thermal Mass Parameterization . . . . . **3-35**
Electrical Behavior Depending on Temperature . . . . . **3-36**
Improving Numerical Performance . . . . . **3-36**
Model Thermal Losses for a Rectifier . . . . . **3-37**

**Plot Basic Characteristics for Battery Blocks** . . . . . **3-43**

**Plot Basic Characteristics for Semiconductor Blocks** . . . . . **3-45**

**MOSFET Characteristics Viewer** . . . . . **3-48**
Suggested Workflow . . . . . **3-48**
Add and Manage Characteristics . . . . . **3-50**
Choose Parameters and Generate Plots . . . . . **3-52**
Save the Results . . . . . **3-54**

**Converting a SPICE Netlist to Simscape Blocks** . . . . . **3-56**
Commands . . . . . **3-56**
Numeric Suffixes . . . . . **3-57**
Mathematical Functions . . . . . **3-57**
Symbols . . . . . **3-61**
Components . . . . . **3-61**
Performing Manual Conversions . . . . . **3-64**
Limitations . . . . . **3-64**

**Photovoltaic Thermal (PV/T) Hybrid Solar Panel** . . . . . **3-66**

**List of Pre-Parameterized Components** . . . . . **3-75**
Load Predefined Parameterizations . . . . . **3-75**
List of Pre-Parameterized Components . . . . . **3-77**
Additional Parameterizations . . . . . **3-102**
Request a Pre-Parameterization . . . . . **3-103**

## 4 Modeling Machines

**Machine Parameterization** . . . . . **4-2**

**Per-Unit Conversion for Machine Parameters** . . . . . **4-3**
Impedance Conversion Equations . . . . . **4-3**
Magnetic Flux Linkage Conversion Equations . . . . . **4-3**

**Machine Plotting and Display Options** . . . . . **4-4**
Asynchronous Machine Options . . . . . **4-4**
Induction Machine Options . . . . . **4-4**
Machine Inertia Block Options . . . . . **4-5**

**Initialize Synchronous Machines and Controllers** . . . . . **4-6**

## 5 Customization

**Build Custom Blocks Using the Three-Phase Electrical Domain** . . . . . **5-2**

**Custom Synchronous Machine** . . . . . **5-4**

## 6 Control

**Tune an Electric Drive** . . . . . **6-2**
Cascade Control Structure . . . . . **6-2**
Equations for PI Tuning Using the Pole Placement Method . . . . . **6-2**
Equations for DC Motor Controller Tuning . . . . . **6-4**
Tune the Electric Drive in the Example Model . . . . . **6-6**

# 7 Simulation and Analysis of Power Engineering Systems


**Optimize Block Settings for Simulating with the Partitioning Solver** . . . **7-2**
Update Solver and Zero-Sequence Settings Using the ee_solverUpdate Function . . . **7-2**
Limitations of the ee_updateSolver Function . . . **7-9**

**Phasor-Mode Simulation Using Simscape Components** . . . **7-11**

**Examine the Simulation Data Logging Configuration of a Model** . . . **7-15**

**Perform a Power-Loss Analysis** . . . **7-17**
Prerequisite . . . **7-17**
Calculate Average Power Losses for the Simulation . . . **7-17**
Analyze Power Dissipation Differences Using Instantaneous Power Dissipation . . . **7-18**
Mitigate Transient Effects in Simulation Data . . . **7-21**

**Choose a Simscape Electrical Function for an Offline Harmonic Analysis** . . . **7-24**
Harmonic Distortion . . . **7-24**
Harmonic Analysis Functions . . . **7-24**
Evaluate Relative Overall Harmonic Distortion . . . **7-25**
Compare Harmonic Distortion to Standard Limits . . . **7-25**
Minimize Harmonic Distortion with Passive Filters . . . **7-25**
Verify the Results of an Online Harmonic Analysis . . . **7-26**

**Perform an Online Harmonic Analysis Using the Simscape Spectrum Analyzer Block** . . . **7-27**
Harmonic Distortion . . . **7-27**
Prerequisite . . . **7-27**
Perform an Offline Harmonic Analysis . . . **7-27**
Perform an Online Harmonic Analysis . . . **7-30**

**Perform a Load-Flow Analysis Using Simscape Electrical** . . . **7-35**
Network Requirements for a Simscape Electrical Load-Flow Analysis . . . **7-35**
Essential Blocks for a Load-Flow Analysis . . . **7-36**
Performing a Load-Flow Analysis . . . **7-36**
Machine Parameterization and Variable Initialization . . . **7-38**
Load-Flow Analyzer App . . . **7-38**
Troubleshooting Load-Flow Analysis and Initialization Issues . . . **7-39**


# 8 Examples


**Peak Shaving with Battery Energy Storage System** . . . **8-2**

**Measure Temperature During Quenching of Steel Using Thermocouple Block** . . . **8-11**

**Protect Components During Fault Using Current Limiter Block** . . . . . **8-17**

**Optimize an Inverter Liquid Cooling System** . . . . . **8-20**

**DC Fast Charging Station for Electric Vehicles with Solar Cogeneration** . . . . . **8-34**

**SiC MOSFET Parameterization Using Simulation Results from SPICE** . **8-36**

**On-Board Charger for Two-Wheeler Electric Vehicle** . . . . . **8-41**

**Fault Detection of Electric Vehicle Charger** . . . . . **8-50**

**Fuel Cell I-V Characteristic** . . . . . **8-57**

**Externally Excited Synchronous Motor Field Oriented Control** . . . . . **8-60**

**Microgrid Resynchronization with Main Grid** . . . . . **8-62**

**System-Level AC Drive** . . . . . **8-72**

**Simple Induction Hob Simulation** . . . . . **8-74**

**Marine Power System Deployment to HIL** . . . . . **8-82**

**Three-Phase High-Power Converter Design and Analysis Workflow** . . . . **8-88**

**Test Harness to Generate High-Power IGBT Device Characteristics** . . **8-115**

**Test Harness to Generate IV Characteristics of N-Channel IGBT** . . . . . **8-124**

**Manage Model Fidelity Using Variants** . . . . . **8-127**

**Power Converter Voltage Stabilizer** . . . . . **8-131**

**Microgrid with Electric Vehicles V2G (Vehicle-to-Grid) Support** . . . . . **8-133**

**Wind Turbine** . . . . . **8-136**

**Green Hydrogen Microgrid** . . . . . **8-149**

**Dual Active Bridge Control** . . . . . **8-151**

**Four-Phase PMSM Torque Control** . . . . . **8-153**

**Four-Phase PMSM Velocity Control** . . . . . **8-155**

**Mixed AC/DC System Loadflow** . . . . . **8-157**

**Six-Phase Switched Reluctance Machine Control** . . . . . **8-158**

**Washing Machine Fault Analysis** . . . . . **8-161**

**Calculate Losses in a Power Converter** . . . . . **8-164**

**Import IGBT Parameters from Hitachi** . . . . . **8-165**

**Analysis of Synchronous Buck Converter with Self Turn-On** . . . . . **8-167**

**DC Fast Charger for Electric Vehicle** . . . . . **8-180**

**Smartphone Battery Charging** . . . . . **8-187**

**Mars Helicopter System-Level Design** . . . . . **8-190**

**Overcurrent Relay Protection in AC Microgrid** . . . . . **8-195**

**Distance Relay Protection in AC Microgrid** . . . . . **8-202**

**Two-Phase DC-DC Converter Current Control** . . . . . **8-215**

**Fuel Cell System** . . . . . **8-218**

**Totem-Pole PFC** . . . . . **8-220**

**Import PMSM Flux Linkage Data from JMAG-RT** . . . . . **8-222**

**Import a Dual Three-Phase SPMSM from Motor-CAD** . . . . . **8-224**

**Bidirectional DC-DC Converter Current Control** . . . . . **8-226**

**Battery Pack Fault Protection** . . . . . **8-229**

**Analysis of Solar Photovoltaic System Shading** . . . . . **8-236**

**Faulted PMSM** . . . . . **8-245**

**Parameterize a Lookup Table-Based MOSFET from SPICE** . . . . . **8-250**

**Import Efficiency Map Data from Motor-CAD** . . . . . **8-256**

**Permanent Magnet Synchronous Generator Battery Charging** . . . . . **8-259**

**Three-Phase Matrix Converter with Venturini Modulation** . . . . . **8-261**

**Scalar Control in Matrix Converter-Fed Induction Machine Drive** . . . . **8-265**

**Piezo Bender Energy Harvester** . . . . . **8-267**

**Battery Pack DC Fast Charging** . . . . . **8-272**

**SPICE Conversion of a MOSFET Subcircuit and Validation** . . . . . **8-281**

**Three-Phase Grid-Tied Inverter Optimal Current Control** . . . . . **8-296**

**Compound Motor Design Optimization** . . . . . **8-298**

**Battery Pack Cell Balancing** . . . . . **8-302**

**Battery Pack Thermal Management** . . . . . **8-308**

**Battery Pack Short Circuit** . . . . . **8-315**

**Input Admittance Response of RLC Ladder Network with Mutual Coupling Between Multiple Coils** . . . . . **8-320**

**Tolerance Study Using Monte Carlo Simulations in Resonant LLC DC-DC Converter** . . . . . **8-323**

**Six-Phase PMSM Torque Control** . . . . . **8-330**

**Six-Phase PMSM Velocity Control** . . . . . **8-332**

**Three-Phase Grid-Connected Rectifier Control** . . . . . **8-334**

**BLDC Position Control with Thermal Model** . . . . . **8-336**

**Three-Phase PMSM Drive with Thermal Model** . . . . . **8-339**

**Single-Phase Inverter Current Control** . . . . . **8-342**

**Three-Phase Grid-Tied Inverter** . . . . . **8-344**

**Three-Phase Inverter Voltage Control** . . . . . **8-346**

**SPICE Conversion of a CMOS Voltage Comparator** . . . . . **8-348**

**Troubleshoot a SPICE Conversion of an IGBT Subcircuit** . . . . . **8-354**

**Calculate Performance Curves of BLDC** . . . . . **8-359**

**Single-Phase PMSM Control** . . . . . **8-361**

**Five-Phase PMSM Velocity Control** . . . . . **8-363**

**Five-Phase PMSM Torque Control** . . . . . **8-365**

**Three-Phase PMLSM Drive** . . . . . **8-367**

**DC-DC Converter Model Fidelity Comparison** . . . . . **8-369**

**Chopper Model Fidelity Comparison** . . . . . **8-372**

**Single-Phase Grid-Connected Solar Photovoltaic System** . . . . . **8-375**

**Three-Phase Grid-Connected Solar Photovoltaic System** . . . . . **8-380**

**Solar PV System with MPPT Using Boost Converter** . . . . . **8-385**

**Stand-Alone Solar PV DC Power System with Battery Backup** . . . . . **8-389**

**Stand-Alone Solar PV AC Power System with Battery Backup** . . . . . **8-396**

**Pre-Parameterized Stepper Motor Block Validation** . . . . . **8-403**

**IEEE 9-Bus Loadflow** . . . . . **8-405**

**2-Bus Loadflow** . . . . . **8-407**

**Induction Motor Initialization with Loadflow** . . . . . **8-409**

**Synchronous Machine Initialization with Loadflow** . . . . . **8-411**

**Stepper Motor Pull-in Characteristics** . . . . . **8-413**

**Three-Phase Matrix Converter** . . . . . **8-416**

**Battery Parameter Extraction from Data** . . . . . **8-418**

**Power Converter Model Fidelity Comparison** . . . . . **8-427**

**Earthing Effects with Unbalanced Load** . . . . . **8-430**

**Clocked Set-Dominant SR-Latch** . . . . . **8-432**

**Clocked Reset-Dominant SR-Latch** . . . . . **8-434**

**Frequency-Dependent Transmission Line** . . . . . **8-436**

**Load Side Converter Control** . . . . . **8-444**

**PMSM Position Control** . . . . . **8-446**

**Power Factor Correction for CCM Boost Converter** . . . . . **8-448**

**PWM-Controlled DC Motor** . . . . . **8-450**

**Microcontroller with GPIO, ADC and DAC Connections** . . . . . **8-452**

**Motor Torque-Speed Curves** . . . . . **8-456**

**Linear Electric Actuator (Motor Model)** . . . . . **8-459**

**Linear Electric Actuator with Control** . . . . . **8-462**

**Brushless DC Motor** . . . . . **8-467**

**Import IPMSM Flux Linkage Data from ANSYS Maxwell** . . . . . **8-470**

**Switched Reluctance Motor Parameterized with FEM Data** . . . . . **8-472**

**HEV PMSM Drive Test Harness** . . . . . **8-478**

**PMSM Iron Losses** . . . . . **8-483**

**PMSM with Thermal Model** . . . . . **8-485**

**Stepper Motor with Control** . . . . . **8-489**

**Stepper Motor Averaged Mode** . . . . . **8-493**

**Unipolar Stepper Motor with Control** . . . . . **8-498**

**Unipolar Stepper Motor Averaged Mode** . . . . . **8-502**

**Hybrid Linear Actuator** . . . . . **8-505**

**Solenoid Parameterized with FEM Data** . . . . . **8-511**

**Custom Solenoid with Magnetic Hysteresis** . . . . . **8-518**

**Torque Motor Parameterization** . . . . . **8-520**

**Automotive Alternator Charging a Battery** . . . . . **8-527**

**Digital Potentiometer Parameterized from Datasheet** . . . . . **8-529**

**Switched Capacitor Analog to Digital Converter** . . . . . **8-532**

**Delta Sigma ADC with Noise** . . . . . **8-535**

**Analog Anti-Aliasing Filter** . . . . . **8-537**

**Strain Gauge and Wheatstone Bridge** . . . . . **8-540**

**Thermistor-Controlled Fan** . . . . . **8-542**

**JFET Amplifier and Frequency Response Analysis** . . . . . **8-544**

**Op-Amp with Noise** . . . . . **8-547**

**AM Radio Receiver** . . . . . **8-549**

**Switching Audio Power Amplifier** . . . . . **8-552**

**Audio Power Amplifier with H-Bridge** . . . . . **8-555**

**Differential Pair Amplifier** . . . . . **8-558**

**Low-Noise Amplifier** . . . . . **8-562**

**Low-Pass Filter Using Operational Transconductance Amplifiers** . . . . . **8-565**

**Controllable Phase Shifter** . . . . . **8-568**

**Fourth-Order Sallen-Key Lowpass Filter** . . . . . **8-571**

**Interactive Generation of MOSFET Characteristics** . . . . . **8-574**

**Interactive Generation of LDMOS Characteristics** . . . . . **8-576**

**Interactive Generation of Thermal MOSFET Characteristics** . . . . . **8-578**

**MOSFET Characterization Using Y Parameters** . . . . . **8-580**

**Solar Cell Power Curve** . . . . . **8-583**

**Use of Peltier Device as Thermoelectric Cooler** . . . . . **8-585**

**IGBT Characteristics** . . . . . **8-587**

**IGBT Thermal Characteristics** . . . . . **8-589**

**IGBT Dynamic Characteristics** . . . . . **8-592**

**IGBT Behavioral Model** . . . . . **8-594**

**IGBT Loss Characteristics** . . . . . **8-596**

**MOSFET Characteristics** . . . . . **8-598**

**Nonlinear Inductor Characteristics** . . . . . **8-600**

**Inductor with Hysteresis** . . . . . **8-606**

**Nonlinear Transformer Characteristics** . . . . . **8-608**

**NPN Bipolar Transistor Characteristics** . . . . . **8-612**

**PNP Bipolar Transistor Characteristics** . . . . . **8-614**

**Schottky Barrier Diode Characteristics** . . . . . **8-616**

**Thyristor Static Behavior Validation** . . . . . **8-618**

**Thyristor Dynamic Behavior Validation** . . . . . **8-627**

**PWM Circuit Using 555 Timer** . . . . . **8-628**

**Triangle Wave Generator** . . . . . **8-630**

**Astable Oscillator** . . . . . **8-632**

**LC Oscillator Based on Colpitts Circuit** . . . . . **8-634**

**Voltage-Controlled Oscillator with PI Control** . . . . . **8-636**

**Modeling an Integrated Circuit** . . . . . **8-638**

**Multiplier Integrated Circuit** . . . . . **8-642**

**Solar Cell Parameter Extraction from Data** . . . . . **8-644**

**Synchronous J-K Flip-Flop** . . . . . **8-649**

**Conducted Emission of a Buck Converter** . . . . . **8-651**

**Electrical Transformer with Hysteresis** . . . . . **8-653**

**Linear Voltage Regulator with Thermal Effects** . . . . . **8-655**

**Linear Voltage Regulator with Feedback** . . . . . **8-657**

**Photovoltaic Generator** . . . . . **8-659**

**Solar Power Inverter** . . . . . **8-662**

**Buck Converter** . . . . . **8-664**

**Buck Converter with Thermal Dynamics** . . . . . **8-669**

**Buck Converter Thermal Model** . . . . . **8-674**

**Buck Converter with Faults** . . . . . **8-676**

**Flyback Converter** . . . . . **8-679**

**DC-DC LLC Converter** . . . . . **8-681**

**Class E DC-DC Converter** . . . . . **8-686**

**Linear LED Driver** . . . . . **8-689**

**TVS Diode Parameterization** . . . . . **8-692**

**Ultracapacitor with Converter** . . . . . **8-694**

**Energy Scavenger** . . . . . **8-696**

**MOSFET Fault in Buck Converter** . . . . . **8-698**

**Surge Protection in Buck Converter** . . . . . **8-701**

**ARINC 429 Communication Link** . . . . . **8-703**

**Band-Pass Filter Using Three Mutually-Coupled Inductors** . . . . . **8-707**

**Class-E RF Amplifier** . . . . . **8-710**

**Diode Ring Mixer** . . . . . **8-712**

**Phase-Locked Loop** . . . . . **8-716**

**LC Transmission Line and Test Bridge** . . . . . **8-719**

**Automotive Electrical System** . . . . . **8-724**

**Electric Vehicle Configured for HIL** . . . . . **8-728**

**Kinetic Energy Recovery System** . . . . . **8-733**

**Power Split Hybrid Vehicle Electrical Network** . . . . . **8-737**

**Import IPMSM Flux Linkage Data from Motor-CAD** . . . . . **8-741**

**Three-Phase PMSM Traction Drive** . . . . . **8-743**

**Solar Panel Parameterization Validation** . . . . . **8-745**

**Energy Balance in a 48V Starter Generator** . . . . . **8-748**

**Three-Phase Asynchronous Direct Online Motor Connected to Hydraulic Pump** . . . . . **8-750**

**Three-Phase Asynchronous Drive with Sensorless Control** . . . . . **8-751**

**Three-Phase Asynchronous Drive with Sensor Control** . . . . . **8-752**

**Asynchronous Machine Direct Torque Control** . . . . . **8-753**

**Asynchronous Machine Direct Torque Control with Space Vector Modulator** . . . . . **8-754**

**Three-Phase Asynchronous Wind Turbine Generator** . . . . . **8-756**

**Asynchronous Machine Scalar Control** . . . . . **8-758**

**Three-Phase Asynchronous Machine Starting** . . . . . **8-759**

**Torque Control in Three-Level Converter-Fed Asynchronous Machine Drive** . . . . . **8-760**

**Average-Value Chopper Control** . . . . . **8-762**

**Average-Value DC-DC Converter Control** . . . . . **8-764**

**BLDC Hysteresis Current Control** . . . . . **8-766**

**BLDC Position Control** . . . . . **8-768**

**BLDC Speed Control** . . . . . **8-770**

**Boost Converter Voltage Control** . . . . . **8-772**

**Buck Converter Voltage Control** . . . . . **8-774**

**Buck Converter Polynomial Voltage Control** . . . . . **8-776**

**AC Cable with Bonded Sheaths** . . . . . **8-778**

**Composite Three-Phase Rectifier** . . . . . **8-783**

**Custom Inductor (B-H Curve)** . . . . . **8-784**

**Three-Phase Custom PMSM** . . . . . **8-790**

**Three-Phase Custom Synchronous Machine** . . . . . **8-792**

**Three-Phase Custom Simplified Synchronous Machine** . . . . . **8-793**

**Custom Transformer (B-H Curve)** . . . . . **8-797**

**Three-Phase Custom Transforms** . . . . . **8-801**

**Three-Phase Custom Zigzag Transformer** . . . . . **8-802**

**DC Motor Control** . . . . . **8-805**

**DC Motor Control (Lead-Lag)** . . . . . **8-806**

**DC Motor Control (RST)** . . . . . **8-807**

**DC Motor Control (Smith Predictor)** . . . . . **8-808**

**DC Motor Control (State-Feedback and Observer)** . . . . . **8-809**

**Electric Engine Dyno** . . . . . **8-810**

**Electric Power Assisted Steering** . . . . . **8-811**

**Five-Phase Switched Reluctance Machine Control** . . . . . **8-813**

**Four-Phase Switched Reluctance Machine Control** . . . . . **8-815**

**Four-Quadrant Chopper Control** . . . . . **8-817**

**Four-Switch Buck-Boost Converter Control** . . . . . **8-819**

**Harmonic Analysis of a Three-Phase Rectifier** . . . . . **8-821**

**HESM Torque Control** . . . . . **8-827**

**HESM Velocity Control** . . . . . **8-828**

**HV Battery Charge/Discharge** . . . . . **8-829**

**Quantifying IGBT Thermal Losses** . . . . . **8-831**

**Inverting Topology Buck-Boost Converter Control** . . . . . **8-834**

**IPMSM Outer Loop Controller Evaluation** . . . . . **8-836**

**IPMSM Torque-Based Load Control** . . . . . **8-838**

**IPMSM Torque Control** . . . . . **8-840**

**IPMSM Torque Control in an Axle-Drive EV** . . . . . **8-842**

**IPMSM Torque Control in a Series HEV** . . . . . **8-844**

**IPMSM Torque Control in a Parallel HEV** . . . . . **8-846**

**IPMSM Torque Control in a Series-Parallel HEV** . . . . . **8-849**

**IPMSM Velocity Control** . . . . . **8-851**

**IPMSG Voltage Stabilization** . . . . . **8-853**

**Marine Full Electric Propulsion Power System** . . . . . **8-855**

**Three-Phase Modular Multilevel Converter** . . . . . **8-857**

**One-Quadrant Chopper Control** . . . . . **8-859**

**Three-Phase PMSM Drive** . . . . . **8-861**

**PMSM Field-Weakening Control** . . . . . **8-862**

**PMSM Parameterization from Datasheet** . . . . . **8-864**

**PMSM Parameterization from Measurements** . . . . . **8-868**

**Parameterize a Permanent Magnet Synchronous Motor** . . . . . **8-872**

**Push-Pull Buck Converter in Continuous Conduction Mode** . . . . . **8-878**

**Push-Pull Buck Converter in Discontinuous Conduction Mode** . . . . . **8-884**

**Three-Phase Three-Level PWM Generator** . . . . . **8-891**

**Three-Phase Two-Level PWM Generator** . . . . . **8-894**

**Power-Loss Analysis of a Three-Phase Rectifier** . . . . . **8-897**

**Single-Phase Asynchronous Machine Direct Torque Control** . . . . . **8-898**

**Single-Phase Asynchronous Machine Field-Oriented Control** . . . . . **8-901**

**Three-Phase Synchronous Machine Control** . . . . . **8-903**

**Three-Phase Synchronous Machine Drive** . . . . . **8-905**

**Three-Phase Synchronous Machine Governor Control Design** . . . . . **8-907**

**Synchronous Machine State-Space Control** . . . . . **8-910**

**SM Torque Control** . . . . . **8-912**

**SM Velocity Control** . . . . . **8-914**

**Switched Reluctance Machine Current Control** . . . . . **8-916**

**Switched Reluctance Machine Speed Control** . . . . . **8-918**

**Supercapacitor Charging and Discharging Behavior** . . . . . **8-919**

**Supercapacitor Parameter Identification** . . . . . **8-920**

**Synchronous Reluctance Machine Torque Control** . . . . . **8-929**

**Synchronous Reluctance Machine Velocity Control** . . . . . **8-930**

**Tap-Changing Transformer for Automatic Voltage Regulation** . . . . . **8-931**

**Three-Phase Bridge Cycloconverter** . . . . . **8-933**

**Comparison of Three-Phase Port Types** . . . . . **8-934**

**Thyristor-Based Rectifier** . . . . . **8-935**

**Twelve-Pulse Thyristor Rectifier** . . . . . **8-936**

**First and Second Quadrant Chopper Control** . . . . . **8-938**

**First and Fourth Quadrant Chopper Control** . . . . . **8-940**

**Vienna Rectifier Control** . . . . . **8-942**

**Three-Phase Voltage-Sourced Converter (FLB)** . . . . . **8-944**

**Three-Phase Voltage-Sourced Converter (SPWM)** . . . . . **8-945**

**HVDC-MMC Interconnection (1000-MW, +/- 320 kV)** . . . . . **8-947**

**Single-Phase Dual Active Bridge Converter DC/DC Converter** . . . . . **8-949**

**Running a Grid-Connected PV Array Model in Real-Time Using Decoupling Line Blocks** . . . . . **8-953**

**Impact of GIC on Autotransformer Saturation** . . . . . **8-960**

**Islanded Operation of an Inverter-based Microgrid Using Droop Control Technique** . . . . . **8-963**

**Speed Regulation of Brushless DC Motor Drive Using a Hysteresis-Based Current Controller** . . . . . **8-966**

**Speed Regulation of a Brushless DC Motor Drive Using a Variable DC Link Six-Step Inverter** . . . . . **8-968**

**Direct Torque Control of an Induction Motor Drive** . . . . . **8-971**

**Direct Torque Control with Space Vector Modulation of an Induction Motor Drive** . . . . . **8-974**

**Field-Oriented Control of an Induction Motor Drive Used in a Ground Transportation System** . . . . . **8-977**

**Field-Oriented Control of an Interior Permanent Magnet Synchronous Motor** . . . . . **8-982**

**Field-Oriented Control of a Surface Mounted Permanent Magnet Synchronous Motor** . . . . . **8-986**

**Speed Regulation of a 6/4 Switched Reluctance Motor** . . . . . **8-989**

**MMC-STATCOM Connected to a 735-kV Transmission System** . . . . . **8-992**

**PV Home On-Grid Solar System** . . . . . **8-994**

**2-MW PV Farm Connected to a 25-kV Distribution System** . . . . . **8-996**

**On Load Tap Changer (OLTC) Regulating Transformer** . . . . . **8-998**

**Three-Phase Three-Limb (Core-Type) Two-Winding Transformer** . . . . . **8-999**

**Three-Phase Five-Limb (Shell-Type) Three-Winding Transformer** . . . . **8-1001**

**Switching an Inductive Circuit Using a Breaker With No Snubber** . . **8-1003**

**Steady-State Analysis of a Linear Circuit** . . . . . **8-1004**

**Transient Analysis of a Linear Circuit** . . . . . **8-1006**

**Three-Winding Distribution Transformer** . . . . . **8-1008**

**Three-Phase Saturable Transformer** . . . . . **8-1010**

**Current Transformer Saturation** . . . . . **8-1012**

**Use of Surge Arresters in Transmission System** . . . . . **8-1014**

**Single Phase Line - Time and Frequency Domain Testing** . . . . . **8-1016**

**Three-Phase Line (DPL and PI Section) - Single-Phase Energization** **8-1018**

**Four Coupled 66-kV Cables** . . . . . **8-1020**

**Dynamic Load and Programmable Voltage Source** . . . . . **8-1022**

**Single Phase Dynamic Load Block** . . . . . **8-1024**

**Saturable Transformer with Hysteresis** . . . . . **8-1026**

**Cassie and Mayr Arc Models for a Circuit Breaker** . . . . . **8-1029**

**Variable Inductance Modeling** . . . . . **8-1032**

**Interfacing Simulink Models with Simscape Electrical Specialized Power Systems** . . . . . **8-1033**

**Interfacing Simscape Models with Simscape Electrical Specialized Power Systems** . . . . . **8-1039**

**Boost Converter** . . . . . **8-1043**

**Buck Converter** . . . . . **8-1045**

**Voltage-Controlled Buck Converter** . . . . . **8-1047**

**Buck Boost Converter** . . . . . **8-1048**

**Cuk Converter** . . . . . **8-1050**

**Forward Converter** . . . . . **8-1052**

**Flyback Converter with Transformer Leakage** . . . . . **8-1053**

**Resonant LLC Converter** . . . . . **8-1054**

**Series Load Resonant (SLR) Converter** . . . . . **8-1056**

**Two-Quadrant DC/DC Converter** . . . . . **8-1058**

**AC-DC-AC Converter** . . . . . **8-1059**

**Single-Phase PWM Inverter** . . . . . **8-1060**

**AC/DC Three-Level PWM Converter** . . . . . **8-1062**

**Regulated Resonant LLC Converter** . . . . . **8-1065**

**Watkins-Johnson Converter** . . . . . **8-1066**

**Three-Phase SV-PWM Converter** . . . . . **8-1068**

**Three-Level NPC Inverter Using Space-Vector PWM with Neutral-Point Voltage Control** . . . . . **8-1071**

**Five-Cell Multi-Level Converter** . . . . . **8-1073**

**Three-Phase 48-Pulse GTO Converter** . . . . . **8-1074**

**Three-Phase Matrix Converter** . . . . . **8-1077**

**Six-Pulse Cycloconverter** . . . . . **8-1079**

**Thyristor Rectifiers** . . . . . **8-1081**

**Three-Phase Rectifier** . . . . . **8-1082**

**Full-Wave Rectifier** . . . . . **8-1084**

**Current-Controlled Thyristor Rectifier** . . . . . **8-1085**

**Zener Diode Regulator** . . . . . **8-1087**

**Two- and Three-Level Voltage Source Controllers (VSC)** . . . . . **8-1089**

**Switching Function Converter Controlled by Averaged Firing Pulses** **8-1090**

**Buck Converter - Increased Accuracy and Simulation Speed Using Interpolation** . . . . . **8-1091**

**Speed Control of a DC Motor Using BJT H-Bridge** . . . . . **8-1093**

**Inductive Current Chopping** . . . . . **8-1095**

**Multilevel Multiphase Space-Vector PWM** . . . . . **8-1097**

**Two-Level PWM Converter and Dead Time** . . . . . **8-1099**

**Neutral Point Clamp Inverter and Dead Time** . . . . . **8-1102**

**Synchronous Buck Converter** . . . . . **8-1104**

**Loss Calculation in a Three-Phase 3-Level Inverter** . . . . . **8-1105**

**Modular Multi-Level Converter (MMC)** . . . . . **8-1108**

**Power Converters Modeling Techniques** . . . . . **8-1110**

**STATCOM (Detailed MMC Model with 22 Power Modules per Phase)** **8-1112**

**1.5-MVA Multicell Motor Drive** . . . . . **8-1114**

**Six-Phase Synchronous Machine with Post-Fault Operating Strategy** **8-1116**

**Stepper Motor Drive** . . . . . **8-1118**

**Simplified Synchronous Machine - Speed Regulation** . . . . . **8-1120**

**Synchronous Machine** . . . . . **8-1122**

**Starting a Synchronous Motor** . . . . . **8-1124**

**Mechanical Coupling of Synchronous Generator with Exciter System Using the Simscape Mechanical Rotational Port** . . . . . **8-1126**

**Mechanical Coupling of Synchronous Generator with Exciter System** **8-1127**

**Three-Phase Asynchronous Machine** . . . . . **8-1129**

**Saturation in Three-Phase Asynchronous Machine** . . . . . **8-1131**

**Single-Phase Asynchronous Machine** . . . . . **8-1133**

**Single-Phase Asynchronous Machine - Voltage Control of Auxiliary Winding** . . . . . **8-1135**

**Single-Phase Asynchronous Machine - Vector Control of AC Drive** . . **8-1137**

**Permanent Magnet Synchronous Machine** . . . . . **8-1138**

**Five-Phase Permanent Magnet Synchronous Machine** . . . . . **8-1139**

**Starting a DC Motor** . . . . . **8-1141**

**Steam Turbine and Governor System - Subsynchronous Resonance** . **8-1142**

**Emergency Diesel-Generator and Asynchronous Motor** . . . . . **8-1144**

**Performance of Three PSS for Interarea Oscillations** . . . . . **8-1146**

**Switched Reluctance Motor** . . . . . **8-1153**

**Brushless DC Motor Fed by Six-Step Inverter** . . . . . **8-1155**

**Performance of Frequency Measurement (Phasor)** . . . . . **8-1156**

**PMU (PLL-based, Positive-Sequence) Kundur's Two-Area System** . . . **8-1157**

**PMU (PLL-based, Positive-Sequence) Benchmark** . . . . . **8-1161**

**IEEE 13 Node Test Feeder** . . . . . **8-1163**

**Flickermeter Statistical Analysis Module** . . . . . **8-1164**

**Power Flow Control and Line Deicing Using a Bundle-Controlled Line Impedance Modulator (LIM)** . . . . . **8-1166**

**Initializing a 29-Bus, 7-Power Plant Network With the Load Flow Tool of Powergui** . . . . . **8-1169**

**Initializing a 5-Bus Network with the Load Flow Tool of Powergui** . . **8-1171**

**Flickermeter on a Distribution STATCOM** . . . . . **8-1173**

**Three-Phase Series Compensated Network** . . . . . **8-1175**

**Simple 6-Pulse HVDC Transmission System** . . . . . **8-1177**

**Three-Phase Harmonic Filters** . . . . . **8-1180**

**Three-Phase Active Harmonic Filter** . . . . . **8-1182**

**Three-Phase Line - Single-Pole Reclosing** . . . . . **8-1184**

**Three-Phase Programmable Source, V-I Measurement and Sequence Analyzer** . . . . . **8-1186**

**Three-Phase Programmable Source, PLL, Voltage and Power Measurement** . . . . . **8-1188**

**12.8 V, 40 Ah, Lithium-Ion (LiFePO4) Battery Aging Model (1000 h Simulation)** . . . . . **8-1190**

**Lithium-Ion Temperature Dependent Battery Model** . . . . . **8-1192**

**Shaft Coupling (Simscape model)** . . . . . **8-1194**

**Gear Speed Reducer (Simscape model)** . . . . . **8-1196**

**Energy Management Systems for a Hybrid Electric Source (Application for a More Electric Aircraft)** . . . . . **8-1198**

**Supercapacitor Model** . . . . . **8-1202**

**6 kW 45 Vdc Fuel Cell Stack** . . . . . **8-1203**

**Ni-MH Battery Model** . . . . . **8-1205**

**AC1 - Six-Step VSI Induction 3HP Motor Drive** . . . . . **8-1207**

**AC2 - Space Vector PWM VSI Induction 3HP Motor Drive** . . . . . **8-1209**

**AC3 - Field-Oriented Control Induction 200 HP Motor Drive** . . . . . **8-1211**

**AC3 - Comparison Between Detailed and Simplified Models** . . . . . **8-1213**

**AC3 - Sensorless Field-Oriented Control Induction Motor Drive** . . . . **8-1215**

**AC4 - DTC Induction 200 HP Motor Drive** . . . . . **8-1217**

**AC4 - Space Vector PWM-DTC Induction 200 HP Motor Drive** . . . . . **8-1219**

**AC5 - Self-Controlled Synchronous 200 HP Motor Drive** . . . . . **8-1221**

**AC5 - Comparison Between Detailed and Simplified Models** . . . . . **8-1223**

**AC6 - PM Synchronous 3HP Motor Drive** . . . . . **8-1225**

**AC6 - Comparison Between Detailed and Simplified Models** . . . . . **8-1227**

**AC6 - 100 kW Interior Permanent Magnet Synchronous Motor Drive** **8-1229**

**AC7 - Brushless DC Motor Drive During Speed Regulation** . . . . . **8-1231**

**AC7 - Comparison Between Detailed and Simplified Models** . . . . . **8-1233**

**AC7 - Sensorless Brushless DC Motor Drive During Speed Regulation** . . . . . **8-1235**

**AC8 - PM Synchronous 3HP Motor Drive** . . . . . **8-1237**

**AC8 - Comparison Between Detailed and Simplified Models** . . . . . **8-1239**

**AC9 - Single-Phase Induction Motor Drive** . . . . . **8-1241**

**DC1 - Two-Quadrant Single-Phase Rectifier 5 HP DC Drive** . . . . . **8-1242**

**DC1 - Two-Quadrant Single-Phase Rectifier 5 HP DC Drive with Regenerative Braking System** . . . . . **8-1244**

**DC2 - Four-Quadrant Single-Phase Rectifier 5 HP DC Drive** . . . . . **8-1246**

**DC3 - Two-Quadrant Three-Phase Rectifier 200 HP DC Drive** . . . . . **8-1248**

**DC4 - Four-Quadrant Three-Phase Rectifier 200 HP DC Drive** . . . . . **8-1250**

**DC4 - Four-Quadrant Three-Phase Rectifier 200 HP DC Drive with No Circulating Current** . . . . . **8-1252**

**DC5 - One-Quadrant Chopper 5 HP DC Drive** . . . . . **8-1255**

**DC5 - One-Quadrant Chopper 5 HP DC Drive with Hysteresis Current Control** . . . . . **8-1257**

**DC6 - Two-Quadrant Chopper 200 HP DC Drive** . . . . . **8-1259**

**DC7 - Four-Quadrant Chopper 200 HP DC Drive** . . . . . **8-1261**

**Mechanical Shaft** . . . . . **8-1263**

**Speed Reducer** . . . . . **8-1265**

**Mechanical Coupling of Two Motor Drives I** . . . . . **8-1267**

**Mechanical Coupling of Two Motor Drives II** . . . . . **8-1268**

**Winding Machine** . . . . . **8-1269**

**Robot Axis Control Using Brushless DC Motor Drives** . . . . . **8-1270**

**Aircraft Electrical Power Generation and Distribution** . . . . . **8-1271**

**Thyristor-Based HVDC Transmission System (Detailed Model)** . . . . . **8-1273**

**Thyristor-Based HVDC Transmission System (Average Model)** . . . . . **8-1276**

**VSC-Based HVDC Transmission System (Detailed Model)** . . . . . **8-1280**

**STATCOM (Phasor Model)** . . . . . **8-1283**

**STATCOM (Detailed Model)** . . . . . **8-1286**

**D-STATCOM (Detailed Model)** . . . . . **8-1289**

**D-STATCOM (Average Model)** . . . . . **8-1292**

**SSSC (Phasor Model)** . . . . . **8-1295**

**SVC (Phasor Model)** . . . . . **8-1298**

**SVC (Detailed Model)** . . . . . **8-1300**

**TCSC (Phasor Model)** . . . . . **8-1302**

**TCSC (Detailed Model)** . . . . . **8-1304**

**SVC and PSS (Phasor Model)** . . . . . **8-1306**

**UPFC (Phasor Model)** . . . . . **8-1309**

**UPFC (Detailed Model)** . . . . . **8-1311**

**OLTC Phase Shifting Transformer (Phasor Model)** . . . . . **8-1314**

**OLTC Regulating Transformer (Phasor Model)** . . . . . **8-1317**

**Simplified Model of a Small Scale Micro-Grid** . . . . . **8-1320**

**Single-Phase, 240 Vrms, 3500 W Transformerless Grid-Connected PV Array** . . . . . **8-1322**

**250-kW Grid-Connected PV Array** . . . . . **8-1324**

**400-kW Grid-Connected PV Farm (Average Model)** . . . . . **8-1326**

**Partial Shading of a PV Module** . . . . . **8-1328**

**24-hour Simulation of a Vehicle-to-Grid (V2G) System** . . . . . **8-1330**

**Detailed Model of a 100-kW Grid-Connected PV Array** . . . . . **8-1332**

**Average Model of a 100-kW Grid-Connected PV Array** . . . . . **8-1335**

**Wind Farm (IG)** . . . . . **8-1338**

**Wind-Turbine Asynchronous Generator in Isolated Network** . . . . . **8-1341**

**Wind Farm (DFIG Phasor Model)** . . . . . **8-1345**

**Wind Farm - DFIG Detailed Model** . . . . . **8-1349**

**Wind Farm - DFIG Average Model** . . . . . **8-1353**

**Wind Farm - Synchronous Generator and Full Scale Converter (Type 4) Detailed Model** . . . . . **8-1357**

**Wind Farm - Synchronous Generator and Full Scale Converter (Type 4) Average Model** . . . . . **8-1361**

**Solid-Oxide Fuel Cell Connected to Three-Phase Electrical Power System** . . . . . **8-1365**

# Getting Started


- “Simscape Electrical Product Description” on page 1-2
- “Simscape Electrical Block Libraries” on page 1-3
- “Modeling Analog Circuit Architectures, Mechatronic Systems, and Electrical Power Systems Using Simscape Electrical” on page 1-6
- “Per-Unit System of Units” on page 1-9

# Simscape Electrical Product Description

**Model and simulate electronic, mechatronic, and electrical power systems**

Simscape Electrical (formerly SimPowerSystems™ and SimElectronics®) provides component libraries for modeling and simulating electronic, mechatronic, and electrical power systems. It includes models of semiconductors, motors, and components for applications such as electromechanical actuation, smart grids, and renewable energy systems. You can use these components to evaluate analog circuit architectures, develop mechatronic systems with electric drives, and analyze the generation, conversion, transmission, and consumption of electrical power at the grid level.

Simscape Electrical helps you develop control systems and test system-level performance. You can parameterize your models using MATLAB® variables and expressions, and design control systems for electrical systems in Simulink®. You can integrate mechanical, hydraulic, thermal, and other physical systems into your model using components from the Simscape family of products. To deploy models to other simulation environments, including hardware-in-the-loop (HIL) systems, Simscape Electrical supports C-code generation.

Simscape Electrical was developed in collaboration with Hydro-Québec of Montreal.

## Key Features

- Libraries of electrical components including sensors, actuators, motors, machines, passive devices, and semiconductor devices
- Adjustable model fidelity, including nonlinear effects, operational limits, fault modeling, and temperature-dependent behavior
- SPICE netlist importer for converting SPICE subcircuits of discrete devices to Simscape models
- Application-specific models, including common AC and DC electric drives, smart grids, and renewable energy systems
- Ideal switching, discretization, and phasor simulation for faster model execution
- MATLAB based Simscape language for creating custom component models
- Support for C-code generation (with Simulink Coder™)

# Simscape Electrical Block Libraries

Simscape Electrical software includes twelve different top-level libraries. These libraries allow you to model mechatronic systems, analog circuit architectures, and single- and multi-phase electrical power systems. You can also develop control algorithms for these systems within the Simulink environment by using these libraries. All of these libraries, except Specialized Power Systems, contain blocks developed specifically for extending the Simscape Foundation domains and are fully compatible with the Simscape technology. Blocks in the Specialized Power Systems library function in their own domain.

## Libraries Compatible with Simscape Technology

All Simscape Electrical libraries, except Specialized Power Systems, contain blocks specifically developed to:

- Extend the Simscape Electrical domain, a single-phase electrical domain.
- Extend the Simscape Three-Phase Electrical domain, a three-phase electrical domain.

These library blocks are written in the Simscape language and are fully compatible with the Simscape technology, including local solvers, data logging, statistics and variable viewers, frequency analysis, and component and library customizations. To configure Simscape Electrical models composed of these library blocks for local-solver simulation, use the Solver Configuration block. Many of the blocks in these libraries also work with other Simscape Foundation domains, such as the Mechanical, Magnetic, and Thermal domains. When working with the Simscape technology compatible library blocks, you can use these capabilities:

- Partitioning Solver
- Simscape HDL Workflow Advisor
- Simscape Results Explorer

These libraries include models of high-fidelity, nonlinear, faultable, electrothermal power electronics. You can use these components to develop mechatronic systems and to build behavioral models for evaluating analog circuit architectures. The libraries also include low-fidelity models that are switched linear and optimized for fast simulation. There are also some models that contain optional ports for thermal analysis.

You can create single-line three-phase diagrams by using the three-phase blocks because the Three-Phase Electrical domain supports signals that contain all three phases as individual elements in a single vector. You can also model each phase individually, for example, to inject a single-line-to-ground fault into your circuit, by expanding the three-phase ports on these blocks into three separate single-phase electrical ports.

The Control library contains Simulink blocks for signal generation, mathematical transformation, and machine control. You can use these components to develop control systems for single- and multi-phase electrical power systems.

Through conserving ports of the same domain, you can directly connect the blocks from these Simscape Electrical libraries to Simscape blocks from:

- Simscape Foundation libraries
- Simscape add-on products, such as Simscape Driveline™, Simscape Multibody™, and Simscape Fluids™

Through physical signal ports, you can connect the physical blocks from these Simscape Electrical libraries to:

- Simulink blocks, including blocks from the Control library, by using converter blocks from the Simscape Utilities library
- Blocks from the Physical Signals library, which is in the Simscape Foundation library.

## Specialized Power Systems Library

The Simscape Electrical Specialized Power Systems library contains blocks that use their own, *specialized* electrical domain. The library contains models of typical power equipment such as transformers, electric machines and drives, and power electronics. It also contains control, measurement, and signal generation models that you can use for developing power system control algorithms. The Specialized Power Systems Fundamental Blocks library contains the powergui block, which provides tools for the steady-state analysis of electrical circuits. To configure Specialized Power Systems models for continuous-time, discrete-time, or phasor simulation, and to analyze simulation results, use the powergui block. The powergui block is in the Specialized Power Systems Fundamental Blocks library.

You can connect Specialized Power Systems blocks to Simulink blocks either:

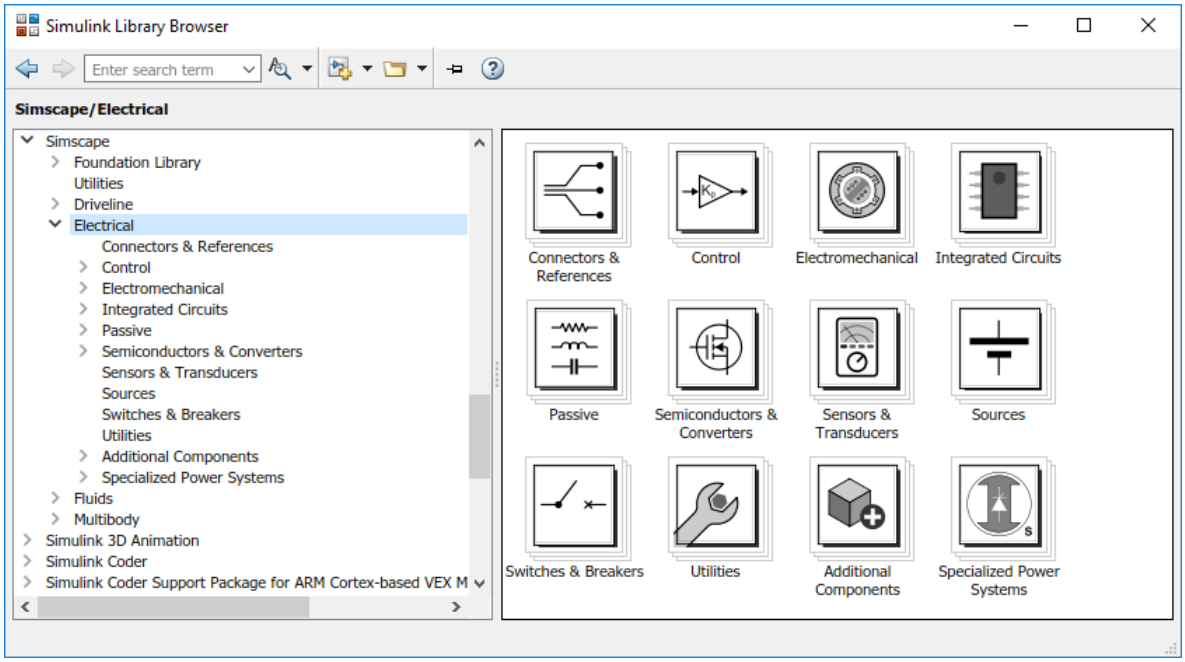
- Directly, through Simulink signal input and output ports.
- Through measurement blocks from the Measurements sublibrary of the Fundamental Blocks library.

## Access the Simscape Electrical Block Libraries

You can access the Simscape Electrical libraries from the Simulink Library Browser or from the MATLAB command prompt.

To display the **Electrical** library in the Simulink Library Browser, scroll to the **Simscape** node. Expand the **Simscape** node and then the **Electrical** node. Alternately, at the MATLAB command prompt, enter this command.

```
ee_lib
```




To access the sublibraries in the twelve top-level Simscape Electrical libraries, further expand the nodes. Alternately, use the `open_system` command at the MATLAB command prompt. For example, to access the sublibraries in the **Connectors & References** library, enter the commands:

```
ee_lib;
open_system('ee_lib/Connectors & References')
```

# Modeling Analog Circuit Architectures, Mechatronic Systems, and Electrical Power Systems Using Simscape Electrical

| In this section... |
| --- |
| “Simulink Templates for Modeling with Simscape Electrical” on page 1-6 |
| “Simscape Electrical Blocks and Ports” on page 1-7 |
| “Machine and Transformer Source Code Examples” on page 1-7 |
| “Plotting and Display Options for Asynchronous and Synchronous Machines” on page 1-7 |
| “Choosing the Right Simscape Electrical Technology” on page 1-7 |
| “Assumptions and Limitations” on page 1-8 |

When you model and analyze mechatronic systems, analog circuit architectures, or electrical power systems using Simscape Electrical, your workflow might include the following tasks:

1. Create a Simulink model that includes components from Simscape Electrical libraries.

   In most applications, it is most natural to model the physical system using Simscape Electrical blocks and other Simscape blocks, and then develop the controller or signal processing algorithm in Simulink.

   For more information about modeling the physical system, see “Essential Electrical Modeling Techniques” on page 3-2.
2. Define component data by specifying electrical or mechanical properties as defined on a datasheet.

   For more information about parametrization, see “Parameterizing Blocks from Datasheets” on page 3-13.
3. Configure the solver options.

   For more information about the settings that most affect the solution of a physical system, see “Setting Up Solvers for Physical Models”.
4. Run the simulation.

   For more information on how to perform time-domain simulation of an electrical system, see “Simulating an Electronic, Mechatronic, or Electrical Power System” on page 3-8.

## Simulink Templates for Modeling with Simscape Electrical

On the Simulink start page, the **Simscape** section has model templates that provide you with design patterns for modeling with Simscape Electrical:

- Electrical
- Electrical Three-Phase
- Mechanical Rotational
- Mechanical Translational

Models you create from these templates have the corresponding reference block, the required Solver Configuration block, and the frequently used Simscape-Simulink interfacing blocks already in the

Simulink canvas. The models also contain links that you can double-click to access other blocks in the corresponding Simscape libraries.

To create a model using one of these **Simscape** templates:

1. Open the Simulink Start page. In the MATLAB Home tab, select the **Simulink** button. Alternatively, at the command line, enter:

   ```
   simulink
   ```

2. In the **Simscape** section, locate the templates that are preconfigured for modeling with Simscape Electrical. Selecting a template opens a model in the Simulink Editor. To save the model, select **Simulation > Save > Save As**.

## Simscape Electrical Blocks and Ports

Simscape Electrical blocks that are written in the Simscape language are fully compatible with Simscape technology, including the local solver, code generation, and data logging.

Simscape Electrical blocks have single-phase, composite three-phase, thermal, magnetic, mechanical translational conserving, and mechanical rotational conserving ports. You can use composite three-phase ports to build models corresponding to single-line diagrams of three-phase electrical systems. Composite three-phase ports connect to other composite three-phase ports. Electrical and mechanical rotational conserving ports connect directly to Simscape Foundation library components and Simscape add-on products such as Simscape Driveline. You can use a Phase Splitter block to split a composite three-phase port into individual electrical conserving ports.

## Machine and Transformer Source Code Examples

Simscape Electrical software provides Simscape language source code examples for machines and transformers, which you can view and customize. To access the example blocks, type `ThreePhaseExamples_lib` at the MATLAB command prompt.

## Plotting and Display Options for Asynchronous and Synchronous Machines

For the Machine Inertia block and the asynchronous and synchronous machine blocks in Simscape Electrical software, you can perform some useful plotting and display actions using the **Electrical** menu on the block context menu. For example, to plot torque versus speed (both in SI units) for the Induction Machine Wound Rotor block, right-click the block. From the block context menu, select **Electrical > Plot Torque Speed (SI)**. The software plots the results in a figure window.

Using other options on the **Electrical** menu, you can plot values in per-unit or display base parameter values in the MATLAB Command Window. These options enable you to tune the performance of your three-phase machine quickly.

## Choosing the Right Simscape Electrical Technology

Simscape Electrical software includes two different technologies and corresponding libraries. For a comparison of the two technologies, see “Simscape Electrical Block Libraries” on page 1-3. Choose the Simscape Electrical technology most appropriate for your modeling needs and, if possible, build your model using blocks exclusively from that technology. However, if necessary, you can build a

model that uses blocks from both technologies. To do so, use blocks from the **Simscape > Electrical > Specialized Power Systems > Fundamental Blocks > Interface Elements** library to interface between them.

## Assumptions and Limitations

The Simscape Electrical blocks let you perform tradeoff analyses to optimize system design, for example, by testing various algorithms with different circuit implementations. The library contains blocks that use either high level or more detailed models to simulate components. Simscape Electrical does not have the capability to:

- Perform either layout (physical design) tasks, or the associated implementation tasks such as layout versus schematic (LVS), design rule checking (DRC), parasitic extraction, and back annotation.
- Model 3-D parasitic effects that are typically important for high-frequency applications.

For these types of requirements, you must use an EDA package specifically designed for the implementation of analog circuits.

## See Also

Phase Splitter

## More About

- “Simscape Electrical Block Libraries” on page 1-3
- “Essential Electrical Modeling Techniques” on page 3-2
- “Parameterizing Blocks from Datasheets” on page 3-13
- “Setting Up Solvers for Physical Models”
- “Simulating an Electronic, Mechatronic, or Electrical Power System” on page 3-8

# Per-Unit System of Units

| In this section... |
| --- |
| “What Is the Per-Unit System?” on page 1-9 |
| “Example 1: Three-Phase Transformer” on page 1-10 |
| “Example 2: Asynchronous Machine” on page 1-11 |
| “Base Values for Instantaneous Voltage and Current Waveforms” on page 1-12 |
| “Why Use the Per-Unit System Instead of the Standard SI Units?” on page 1-12 |

## What Is the Per-Unit System?

The per-unit system is widely used in the power system industry to express values of voltages, currents, powers, and impedances of various power equipment. It is typically used for transformers and AC machines.

For a given quantity (voltage, current, power, impedance, torque, etc.) the per-unit value is the value related to a base quantity.

$$\text{base value in p.u.} = \frac{\text{quantity expressed in SI units}}{\text{base value}}$$

Generally the following two base values are chosen:

- The base power = nominal power of the equipment
- The base voltage = nominal voltage of the equipment

All other base quantities are derived from these two base quantities. Once the base power and the base voltage are chosen, the base current and the base impedance are determined by the natural laws of electrical circuits.

$$\text{base current} = \frac{\text{base power}}{\text{base voltage}}$$

$$\text{base impedance} = \frac{\text{base voltage}}{\text{base current}} = \frac{(\text{base voltage})^2}{\text{base power}}$$

For a transformer with multiple windings, each having a different nominal voltage, the same base power is used for all windings (nominal power of the transformer). However, according to the definitions, there are as many base values as windings for voltages, currents, and impedances.

The saturation characteristic of saturable transformer is given in the form of an instantaneous current versus instantaneous flux-linkage curve: [i1 phi1; i2 phi2; ..., in phin].

When the per-unit system is used to specify the transformer R L parameters, the flux linkage and current in the saturation characteristic must be also specified in pu. The corresponding base values are

$$\text{base instantaneous current} = (\text{base rms current}) \times \sqrt{2}$$

$$\text{base flux linkage} = \frac{(\text{base rms voltage}) \times \sqrt{2}}{2\pi \times (\text{base frequency})}$$

where current, voltage, and flux linkage are expressed respectively in volts, amperes, and volt-seconds.

For AC machines, the torque and speed can be also expressed in pu. The following base quantities are chosen:

- The base speed = synchronous speed
- The base torque = torque corresponding at base power and synchronous speed

$$\text{base torque} = \frac{\text{base power (3 phases) in VA}}{\text{base speed in radians/second}}$$

Instead of specifying the rotor inertia in kg*m$^2$, you would generally give the inertia constant $H$ defined as

$$H = \frac{\text{kinetic energy stored in the rotor at synchronous speed in joules}}{\text{machine nominal power in VA}}$$

$$H = \frac{\frac{1}{2} \times J \cdot w^2}{Pnom}$$

The inertia constant is expressed in seconds. For large machines, this constant is around 3–5 seconds. An inertia constant of 3 seconds means that the energy stored in the rotating part could supply the nominal load during 3 seconds. For small machines, $H$ is lower. For example, for a 3-HP motor, it can be 0.5–0.7 seconds.

## Example 1: Three-Phase Transformer

Consider, for example, a three-phase two-winding transformer with these manufacturer-provided, typical parameters:

- Nominal power = 300 kVA total for three phases
- Nominal frequency = 60 Hz
- Winding 1: connected in wye, nominal voltage = 25-kV RMS line-to-line

  resistance 0.01 pu, leakage reactance = 0.02 pu
- Winding 2: connected in delta, nominal voltage = 600-V RMS line-to-line

  resistance 0.01 pu, leakage reactance = 0.02 pu
- Magnetizing losses at nominal voltage in % of nominal current:

  Resistive 1%, Inductive 1%

The base values for each single-phase transformer are first calculated:

- For winding 1:

| | |
|---|---|
| Base power | 300 kVA/3 = 100e3 VA/phase |
| Base voltage | 25 kV/sqrt(3) = 14434 V RMS |
| Base current | 100e3/14434 = 6.928 A RMS |
| Base impedance | 14434/6.928 = 2083 Ω |

| | |
|---|---|
| Base resistance | 14434/6.928 = 2083 Ω |
| Base inductance | 2083/(2π*60)= 5.525 H |

- For winding 2:

| | |
|---|---|
| Base power | 300 kVA/3 = 100e3 VA |
| Base voltage | 600 V RMS |
| Base current | 100e3/600 = 166.7 A RMS |
| Base impedance | 600/166.7 = 3.60 Ω |
| Base resistance | 600/166.7 = 3.60 Ω |
| Base inductance | 3.60/(2π*60) = 0.009549 H |

The values of the winding resistances and leakage inductances expressed in SI units are therefore

- For winding 1: R1= 0.01 * 2083 = 20.83 Ω; L1= 0.02*5.525 = 0.1105 H
- For winding 2: R2= 0.01 * 3.60 = 0.0360 Ω; L2= 0.02*0.009549 = 0.191 mH

For the magnetizing branch, magnetizing losses of 1% resistive and 1% inductive mean a magnetizing resistance Rm of 100 pu and a magnetizing inductance Lm of 100 pu. Therefore, the values expressed in SI units referred to winding 1 are

- Rm = 100*2083 = 208.3 kΩ
- Lm = 100*5.525 = 552.5 H

## Example 2: Asynchronous Machine

Now consider a three-phase, four-pole Asynchronous Machine block in SI units. It is rated 3 HP, 220 V RMS line-to-line, 60 Hz.

The stator and rotor resistance and inductance referred to stator are

- Rs = 0.435 Ω; Ls = 2 mH
- Rr = 0.816 Ω; Lr = 2 mH

The mutual inductance is Lm = 69.31 mH. The rotor inertia is $J$ = 0.089 kg.m$^2$.

The base quantities for one phase are calculated as follows:

| | |
|---|---|
| Base power | 3 HP*746VA/3 = 746 VA/phase |
| Base voltage | 220 V/sqrt(3) = 127.0 V RMS |
| Base current | 746/127.0 = 5.874 A RMS |
| Base impedance | 127.0/5.874 = 21.62 Ω |
| Base resistance | 127.0/5.874 = 21.62 Ω |
| Base inductance | 21.62/(2π*60)= 0.05735 H = 57.35 mH |
| Base speed | 1800 rpm = 1800*(2π)/60 = 188.5 radians/second |
| Base torque (three-phase) | 746*3/188.5 = 11.87 newton-meters |

Using the base values, you can compute the values in per-units.

Rs= 0.435 / 21.62 = 0.0201 pu Ls= 2 / 57.35 = 0.0349 pu

Rr= 0.816 / 21.62 = 0.0377 pu Lr= 2 / 57.35 = 0.0349 pu

Lm = 69.31/57.35 = 1.208 pu

The inertia is calculated from inertia $J$, synchronous speed, and nominal power.

$$H = \frac{\frac{1}{2} \times J \cdot w^2}{Pnom} = \frac{\frac{1}{2} \times 0.089 \times (188.5)^2}{3 \times 746} = 0.7065 \text{ seconds}$$

If you open the dialog box of the Asynchronous Machine block in pu units provided in the Machines library of the Simscape Electrical Specialized Power Systems Fundamental Blocks library, you find that the parameters in pu are the ones calculated.

## Base Values for Instantaneous Voltage and Current Waveforms

When displaying instantaneous voltage and current waveforms on graphs or oscilloscopes, you normally consider the peak value of the nominal sinusoidal voltage as 1 pu. In other words, the base values used for voltage and currents are the RMS values given multiplied by $\sqrt{2}$.

## Why Use the Per-Unit System Instead of the Standard SI Units?

Here are the main reasons for using the per-unit system:

- When values are expressed in pu, the comparison of electrical quantities with their "normal" values is straightforward.

  For example, a transient voltage reaching a maximum of 1.42 pu indicates immediately that this voltage exceeds the nominal value by 42%.
- The values of impedances expressed in pu stay fairly constant whatever the power and voltage ratings.

  For example, for all transformers in the 3–300 kVA power range, the leakage reactance varies approximately 0.01–0.03 pu, whereas the winding resistances vary between 0.01 pu and 0.005 pu, whatever the nominal voltage. For transformers in the 300 kVA to 300 MVA range, the leakage reactance varies approximately 0.03–0.12 pu, whereas the winding resistances vary between 0.005–0.002 pu.

  Similarly, for salient pole synchronous machines, the synchronous reactance $X_d$ is generally 0.60–1.50 pu, whereas the subtransient reactance $X'_d$ is generally 0.20–0.50 pu.

  It means that if you do not know the parameters for a 10-kVA transformer, you are not making a major error by assuming an average value of 0.02 pu for leakage reactances and 0.0075 pu for winding resistances.

The calculations using the per-unit system are simplified. When all impedances in a multivoltage power system are expressed on a common power base and on the nominal voltages of the different subnetworks, the total impedance in pu seen at one bus is obtained by simply adding all impedances in pu, without considering the transformer ratios.

# Tutorials


- “Build and Simulate Composite and Expanded Three-Phase Models” on page 2-2
- “DC Motor Model” on page 2-10
- “Build and Simulate a Simple Circuit” on page 2-16
- “Build and Simulate a Single-Phase Half-Bridge Inverter with Ideal Switches” on page 2-22
- “Build and Simulate Single-Phase Half-Bridge Inverter With Ideal Switches and Thermal Port” on page 2-33
- “Multi-Domain Examples on GitHub” on page 2-46

# Build and Simulate Composite and Expanded Three-Phase Models

| In this section... |
| --- |
| “Select System Component Blocks and Build a Resistive Three-Phase Model” on page 2-2 |
| “Specify Simulation Parameters” on page 2-4 |
| “Load Impedance Parameters” on page 2-4 |
| “Specify Display Parameters” on page 2-5 |
| “Simulate and Analyze the Resistive Three-Phase Model” on page 2-5 |
| “Simulate and Analyze a Reactive Three-Phase Model” on page 2-6 |
| “Create an Expanded Balanced Three-Phase Model” on page 2-6 |
| “Create an Expanded Unbalanced Three-Phase Model” on page 2-7 |
| “Simulate the Expanded Balanced and Unbalanced Models and Analyze the Results” on page 2-7 |

In this example, you build and analyze a simple Simscape Electrical model that simulates the behavior of a three-phase AC voltage source driving a purely resistive three-phase load. You then modify the load in this model to change it to:

- A reactive three-phase load
- A resistive three-phase load expanded into individual phases
- An expanded three-phase load that does not have equal resistance in each phase

For the completed initial model, see Simple Three-Phase Model.

## Select System Component Blocks and Build a Resistive Three-Phase Model

1. Open the Simulink Start page. In the MATLAB Home tab, select the **Simulink** button. Alternatively, at the command line, enter:

   ```
   simulink
   ```

2. In the **Simscape** section, find the templates that are preconfigured for modeling with Simscape Electrical. Select the Electrical Three-Phase template. A model that contains these blocks opens in the Simulink canvas.

| Block | Purpose | Library |
| --- | --- | --- |
| Scope | Display phase voltages and currents for the three-phase system. | **Simulink** > **Sinks** |
| Electrical Reference | Provide the ground connection for electrical conserving ports. | **Simscape** > **Foundation Library** > **Electrical** > **Electrical Elements** |
| PS-Simulink Converter | Convert the physical signals to Simulink signals. | **Simscape** > **Utilities** |
| Simulink-PS Converter | Convert Simulink signals to physical signals. | **Simscape** > **Utilities** |

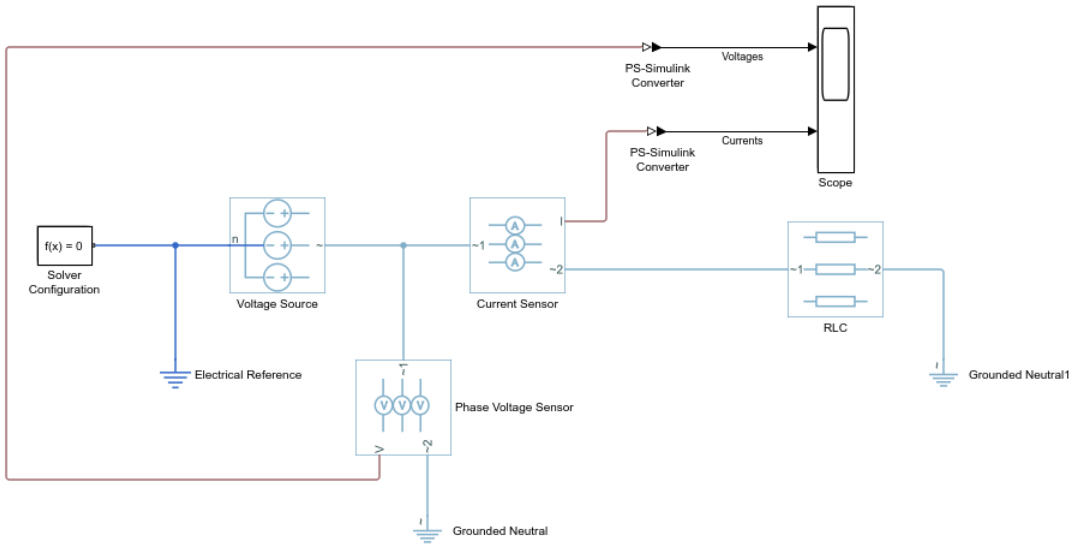
| Block | Purpose | Library |
|---|---|---|
| Solver Configuration | Define solver settings that apply to all physical modeling blocks. | **Simscape > Utilities** |
| Grounded Neutral (Three-Phase) | Provide an electrical ground connection for each phase of the three-phase system. | **Simscape > Electrical > Connectors & References** |
| Line Voltage Sensor (Three-Phase) | Measure the line-line voltages of a three-phase system and output a three-element physical signal vector. | **Simscape > Electrical > Sensors & Transducers** |

The model also contains two links that you can double-click to access blocks from Simscape and Simscape Electrical libraries. For more information on using templates for modeling with Simscape Electrical, see “Modeling Analog Circuit Architectures, Mechatronic Systems, and Electrical Power Systems Using Simscape Electrical” on page 1-6.

3. Delete the Simulink-PS Converter and Line Voltage Sensor (Three-Phase) blocks.
4. Add these blocks to the model.

| Block | Purpose | Library |
|---|---|---|
| RLC (Three-Phase) | Model the resistive, inductive, and capacitive properties of the three-phase load. | **Simscape > Electrical > Passive > RLC Assemblies** |
| Current Sensor (Three-Phase) | Convert the electrical current flowing in each phase of the three-phase load into a physical signal proportional to that current. | **Simscape > Electrical > Sensors & Transducers** |
| Phase Voltage Sensor (Three-Phase) | Convert the voltage across each phase of the three-phase system into a physical signal proportional to that voltage. | **Simscape > Electrical > Sensors & Transducers** |
| Voltage Source (Three-Phase) | Provide an ideal three-phase voltage source that maintains a sinusoidal voltage across its output terminals, regardless of the current flowing in the source. | **Simscape > Electrical > Sources** |

5. Copy the PS-Simulink Converter and Grounded-Neutral (Three-Phase) blocks by right-clicking them and dragging them to new locations on canvas.
6. Add a second input port to the Scope block by clicking its input side and selecting the interactive cue that appears.
7. Connect the blocks as shown.




8 Remove the on-canvas annotations titled Open Simscape Library and Open Simscape Electrical Library. Save the model using the name `simplethreephasemodel`.

The blocks in this model use composite three-phase ports. For more information, see “Three-Phase Ports” on page 3-5.

## Specify Simulation Parameters

As with Simscape models, you must include a Solver Configuration block in each topologically distinct physical network. This model has a single physical network, so use one Solver Configuration block.

1 In the Solver Configuration block, select **Use local solver** and set **Sample time** to `0.0001`.

In Simscape-based models, the local solver is a sample-based solver that represents physical network states as discrete states. For most Simscape Electrical models, the local solver is an appropriate first choice. The solver updates block states once per simulation time step, as determined by **Sample time**. For simulation of a 60-Hz AC system, an appropriate sample time is a value in the order of `1e-4`. For more information on solver options, see Solver Configuration.

If you prefer to use a continuous solver instead of a discrete solver, clear the **Use local solver** check box in the Solver Configuration block. The simulation then uses the Simulink solver specified in the model configuration parameters (**Modeling > Model Settings**). For Simscape Electrical models, an appropriate solver choice is the moderately stiff solver ode23t. For a 60 Hz AC system, specify a value for **Max step size** in the order of `1e-4`. For more information, see “Variable-Step Continuous Explicit Solvers”.

2 In the Simulink Editor, set the simulation **Stop time** to `0.1`.

## Load Impedance Parameters

The RLC block models resistive, inductive, and capacitive characteristics of the three-phase load. Using the **Component structure** parameter, you can specify a series or parallel combination of resistance, inductance, and capacitance.

In the RLC block, the defaults are:

- **Component structure** — R.
- **Resistance** — 1 Ω.

Using the default **Component structure** value, R, models a three-phase load that is purely resistive in nature. The resistance in *each* phase is 1 Ω.

## Specify Display Parameters

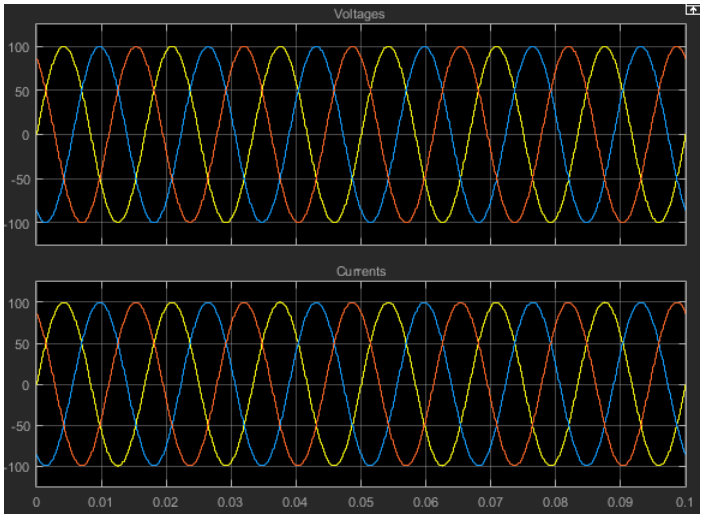
Sensor blocks in the model convert the current and voltage in each phase of the three-phase system to proportional physical signals. PS-Simulink Converter blocks convert the physical signals into Simulink signals for the Scope block to display.

1. Of these three types of blocks, only the converter blocks have parameters. For this example:
   - Set **Output signal unit** of the **PS-Simulink Converter1** block to V. This setting ensures that the block outputs a signal with the same magnitude as the voltage signal that enters it.
   - Set **Output signal unit** of the **PS-Simulink Converter2** block to A. This setting ensures that the block outputs a signal with the same magnitude as the current signal that enters it.
2. Label the input signals to the Scope block. Double-click each line, and type the appropriate label, `Voltages` or `Currents`, as shown in the model graphic.

You are ready to simulate the model and analyze the results.

## Simulate and Analyze the Resistive Three-Phase Model

1. Save the model.
2. Simulate the model.
3. View the phase currents and voltages. Double-click the Scope block.
4. From the scope menu, select **View > Configuration Properties**. Set **Layout** to 1-by-2 display.
5. To scale the scope axes to the data, click the **Autoscale** button .



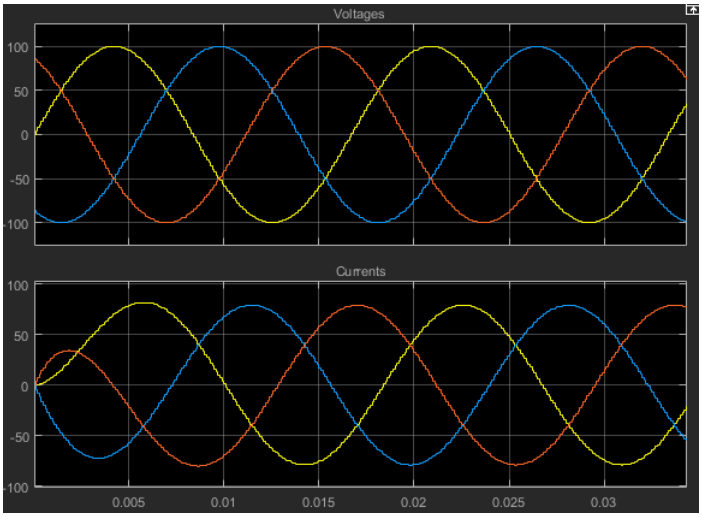

In this simulation, the **Component structure** parameter of the RLC (Three-Phase) block specifies that the electrical characteristics of the three-phase load are purely resistive. Therefore, for each

phase of the three-phase system, the voltage and current remain in phase with each other. Because the resistance in each phase is *1* Ω, the magnitude of the phase voltage is equal to the magnitude of the phase current.

## Simulate and Analyze a Reactive Three-Phase Model

You can modify the model to create a reactive load. A reactive load has inductive or capacitive characteristics.

1. Save this version of the model using the name `simplethreephasemodel_reactive`.
2. In the RLC (Three-Phase) block, set:
   - **Component structure** to `Series RL`
   - **Inductance** to `0.002`
3. Simulate the model.
4. View the simulation results. Autoscale the scope axes.
5. Examine the results in closer detail. For example, click the Zoom button and drag a box over the first third of one of the plots.




The electrical characteristics of three-phase load are no longer purely resistive. Because the load has an inductive characteristic, the current flowing in each phase lags the voltage.

## Create an Expanded Balanced Three-Phase Model

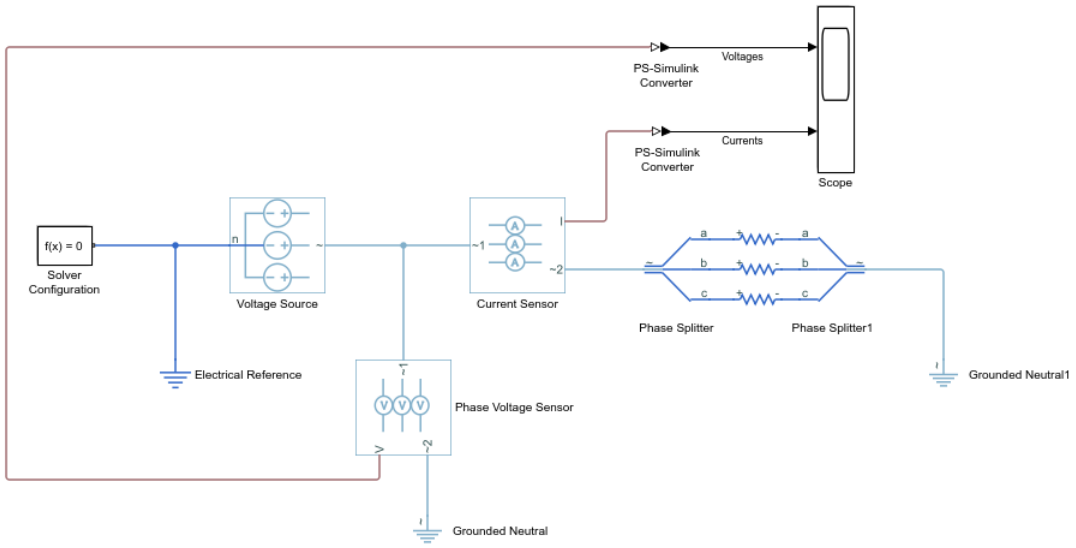
1. Open the resistive three-phase model `simplethreephasemodel` that you initially created.
2. Delete the RLC (Three-Phase) block.
3. Drag two copies of the Phase Splitter block into the model from the **Simscape** > **Electrical** > **Connections & References** library.
4. Flip one of the Phase Splitter blocks horizontally. Select the block. In the toolstrip, on the **Format** tab, click **Flip left-right** .

5 Drag a Resistor element into the model from the **Simscape > Foundation Library > Electrical > Electrical Elements** library.

6 To create space for more components, hide the Resistor element label. Right-click the resistor and select **Format > Show Block Name** to clear this option.

7 Make two more copies of the Resistor element.

8 Connect the components as shown.




9 Save this version of the modified model using the name `simplethreephasemodel_expanded_balanced`.

This model name reflects that the load previously modeled by the RLC block is now expanded into individual phases. The load is still balanced, that is, there is equal resistance in each phase.

## Create an Expanded Unbalanced Three-Phase Model

1 Unbalance the load in `simplethreephasemodel_expanded_balanced` by changing the resistance in one phase. Double-click the phase-c resistor element. Change **Resistance** to 2.

2 Save this version of the modified model using the name `simplethreephasemodel_expanded_unbalanced`.

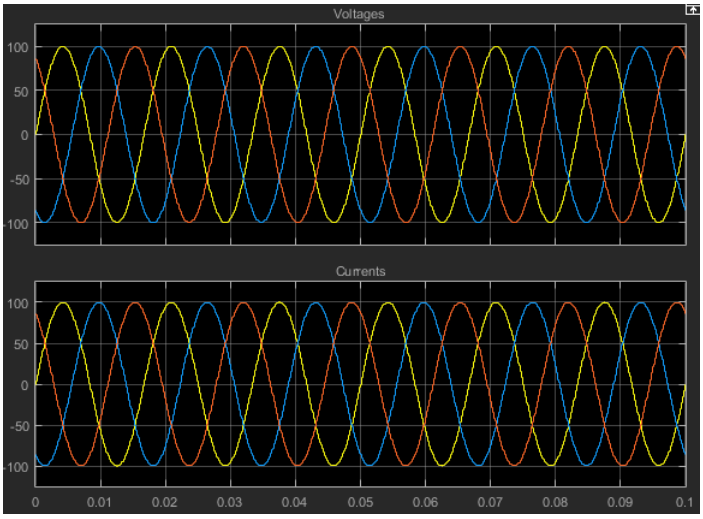
This model name reflects that the three-phase load previously modeled by the RLC block is expanded into individual phases. The load is unbalanced, that is, the resistance in one of the phases is higher than in the other two.

## Simulate the Expanded Balanced and Unbalanced Models and Analyze the Results

1 Simulate the `simplethreephasemodel_expanded_balanced` model. In the menu bar of the Simulink Explorer, click the **Run** button.

2 View the simulation results. Double-click the Scope block.

3 To scale the scope axes to the data, click the **Autoscale** button .

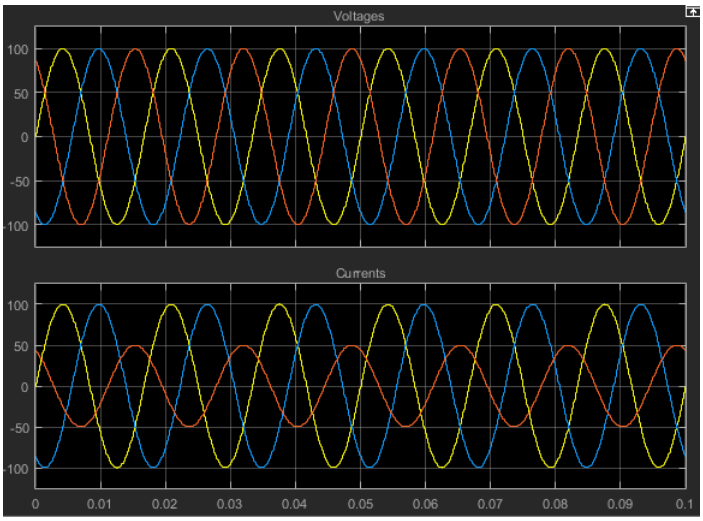



In the `simplethreephasemodel`, the **Component structure** parameter of the RLC (Three-Phase) block specifies that the three-phase load is purely resistive. In this version of the model, the load is expanded into an individual resistive element for each phase, but the resistance in each phase is unchanged. For each phase of the three-phase system, the voltage and current remain in phase with each other. Because the resistance in each phase is 1 Ω, the magnitude of the phase voltage is equal to the magnitude of the phase current.

Comparing these results with the results for the three-phase resistive model shows that a block with composite three-phase ports, the RLC (Three-Phase) block in the original model, produces results with the same fidelity as that of expanded phases.

4 Open the `simplethreephasemodel_expanded_unbalanced` model.

5 Simulate the model. Autoscale the scope axes.




In this version of the model, the c-phase of the three-phase load has twice the resistance of the other two. Therefore, half as much current flows in that phase, as the second plot shows. However, because the load remains purely resistive, the voltage and current remain in phase with each other.

## See Also

## Related Examples

- “Modeling Analog Circuit Architectures, Mechatronic Systems, and Electrical Power Systems Using Simscape Electrical” on page 1-6
- “Essential Electrical Modeling Techniques” on page 3-2

# DC Motor Model

| In this section... |
| --- |
| “Select Blocks to Represent System Components” on page 2-10<br>“Build the Model” on page 2-10<br>“Specify Model Parameters” on page 2-12<br>“Configure the Solver Parameters” on page 2-14<br>“Run the Simulation and Analyze the Results” on page 2-14 |

In this example, you model a DC motor driven by a constant input signal that approximates a pulse-width modulated signal and look at the current and rotational motion at the motor output.

To see the completed version of the model you create in this example, open the “PWM-Controlled DC Motor” on page 8-450 example.

## Select Blocks to Represent System Components

Select the blocks to represent the input signal, the DC motor, and the motor output displays.

The following table describes the role of the blocks that represent the system components.

| Block | Description |
| --- | --- |
| **Solver Configuration** | Defines solver settings that apply to all physical modeling blocks |
| **PS-Simulink Converter** | Converts the input physical signal to a Simulink signal |
| **Controlled PWM Voltage** | Generates the signal that approximates a pulse-width modulated motor input signal |
| **H-Bridge** | Drives the DC motor |
| **DC Motor** | Converts input electrical energy into mechanical motion |
| **Current Sensor** | Converts the electrical current that drives the motor into a measurable physical signal proportional to the current |
| **DC Voltage Source** | Generates a DC voltage |
| **Electrical Reference** | Provides the electrical ground |
| **Mechanical Rotational Reference** | Provides the mechanical ground |
| **Ideal Rotational Motion Sensor** | Converts the rotational motion of the motor into a measurable physical signal proportional to the motion |
| **Scope** | Displays motor current and rotational motion |

## Build the Model

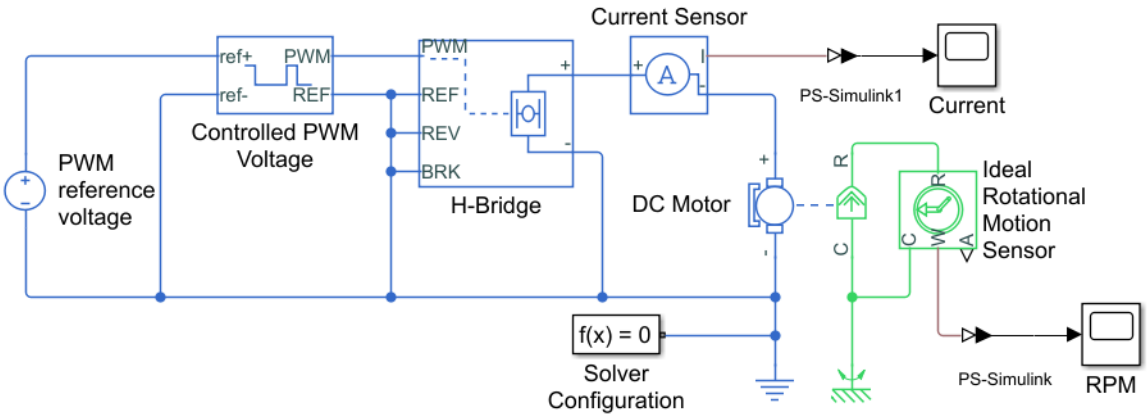
1. Create a new model.
2. Add the blocks in this table. The **Library Path** column specifies the hierarchical path to each block.

| Block | Library | Quantity |
|---|---|---|
| **Solver Configuration** | **Simscape > Utilities** | 1 |
| **PS-Simulink Converter** | **Simscape > Utilities** | 2 |
| **Controlled PWM Voltage** | **Simscape > Electrical > Integrated Circuits** | 1 |
| **H-Bridge** | **Simscape > Electrical > Semiconductors & Converters > Converters** | 1 |
| **DC Motor** | **Simscape > Electrical > Electromechanical > Brushed Motors** | 1 |
| **Current Sensor** | **Simscape > Foundation Library > Electrical > Electrical Sensors** | 1 |
| **DC Voltage Source** | **Simscape > Foundation Library > Electrical > Electrical Sources** | 1 |
| **Electrical Reference** | **Simscape > Foundation Library > Electrical > Electrical Elements** | 1 |
| **Mechanical Rotational Reference** | **Simscape > Foundation Library > Mechanical > Rotational Elements** | 1 |
| **Ideal Rotational Motion Sensor** | **Simscape > Foundation Library > Mechanical > Mechanical Sensors** | 1 |
| **Scope** | **Simulink > Commonly Used Blocks** | 2 |

**Note** You can use the Simscape function `ssc_new` with domain type `electrical` to create a Simscape model that contains these blocks:

- **Simulink-PS Converter**
- **PS-Simulink Converter**
- **Scope**
- **Solver Configuration**
- **Electrical Reference**

3 Rename and connect the blocks as shown in the diagram.




Now you are ready to specify block parameters.

## Specify Model Parameters

Specify these parameters to represent the behavior of the system components:

### Model Setup Parameters

These blocks specify model information that is not specific to a particular block:

- Solver Configuration
- Electrical Reference
- Mechanical Rotational Reference

As with Simscape models, you must include a Solver Configuration block in each topologically distinct physical network. This example has a single physical network, so use one Solver Configuration block with the default parameter values.

You must include an Electrical Reference block in each Simscape Electrical network. You must include a Mechanical Rotational Reference block in each network that includes electromechanical blocks. These blocks do not have any parameters.

For more information about using reference blocks, see “Grounding Rules”.

### Motor Input Signal Parameters

You generate the motor input signal using these blocks:

- The DC Voltage Source block (PWM reference voltage) generates a constant signal.
- The Controlled PWM Voltage block generates a pulse-width modulated signal.
- The H-Bridge block drives the motor.

In this example, all input ports of the H-Bridge block except the PWM port are connected to ground. As a result, the H-Bridge block behaves as follows:

- When the motor is on, the H-Bridge block connects the motor terminals to the power supply.
- When the motor is off, the H-Bridge block acts as a freewheeling diode to maintain the motor current.

In this example, you simulate the motor with a constant current whose value is the average value of the PWM signal. By using this type of signal, you set up a fast simulation that estimates the motor behavior.

1. In the DC Voltage Source block, set the **Constant voltage** parameter to `2.5 V`.
2. Set the parameters of the Controlled PWM Voltage block:
   - **PWM frequency** — `4000 Hz`
   - **Simulation mode** — `Averaged`

     This value tells the block to generate an output signal whose value is the average value of the PWM signal. Simulating the motor with an averaged signal estimates the motor behavior in the presence of a PWM signal. To validate this approximation, use value of `PWM` for this parameter.
3. In the H-Bridge block, set the **Simulation mode** parameter to `Averaged`.

   This value tells the block to generate an output signal whose value is the average value of the PWM signal. Simulating the motor with an averaged signal estimates the motor behavior in the presence of a PWM signal. To validate this approximation, use value of `PWM` for this parameter.

**Note** The simulation mode for both the Controlled PWM Voltage and H-Bridge blocks must be the same.

#### Motor Parameters

Configure the block that models the motor.

Set the parameters of the DC Motor block, leaving the unit settings at their default values where applicable:

- **Electrical Torque** section:
  - **Model parameterization** — `By rated load and speed`
  - **Armature inductance** — `0.01 H`
  - **No-load speed** — `4000 rpm`
  - **Rated speed (at rated load)** — `2500 rpm`
  - **Rated load (mechanical power)** — `10 W`
  - **Rated DC supply voltage** — `12 V`
- **Mechanical** section:
  - **Rotor inertia** — `0.0002 kg*m^2`
  - **Rotor damping** — `0.5e-5 N*m/(rad/s)`

#### Current Display Parameters

Specify the parameters of the blocks that create the motor current display:

- Current Sensor block
- **PS-Simulink Converter1** block

- **Current** scope

Of the three blocks, only the PS-Simulink Converter1 block has parameters. Set the PS-Simulink Converter1 block **Output signal unit** parameter to `A` to indicate that the block input signal has units of amperes.

#### Torque Display Parameters

Specify the parameters of the blocks that create the motor torque display:

- Ideal Rotational Motion Sensor block
- PS-Simulink Converter block
- **RPM** scope

Of the three blocks, only the PS-Simulink Converter block has parameters you need to configure for this example. Set the PS-Simulink Converter block **Output signal unit** parameter to `rpm` to indicate that the block input signal has units of revolutions per minute.

**Note** You must type this parameter value. It is not available in the drop-down list.

## Configure the Solver Parameters

Configure the solver parameters to use a continuous-time solver. Simscape Electrical models only run with a continuous-time solver when you clear the **Local Solver** parameter of a Solver Configuration block. Use the Configuration Parameters dialog box to change the simulation end time, tighten the relative tolerance for a more accurate simulation, and remove the limit on the number of simulation data points Simulink saves.

1. In the model window, select **Modeling** > **Model Settings** to open the Configuration Parameters dialog box.
2. In the left pane, click **Solver**:
   - Set **Stop time** to `10`.
   - Set **Solver** to `ode15s (Stiff/NDF)`.
   - Set **Max step size** to `1`.
   - Set **Relative tolerance** to `1e-3`.
3. Click **OK**.

For more information about configuring solver parameters, see “Simulating an Electronic, Mechatronic, or Electrical Power System” on page 3-8.

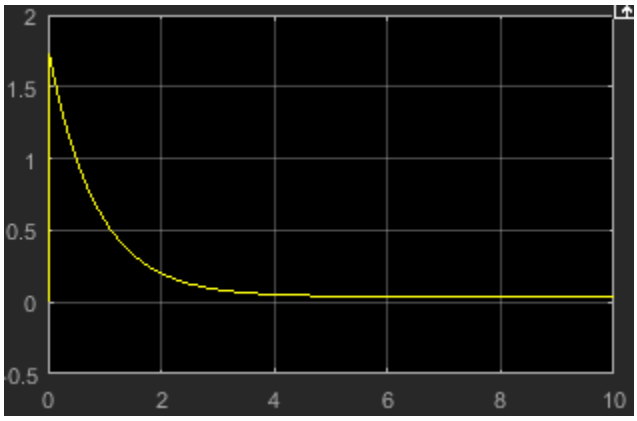
## Run the Simulation and Analyze the Results

Run the simulation and plot the results. In the model window, select **Simulation** > **Run**.

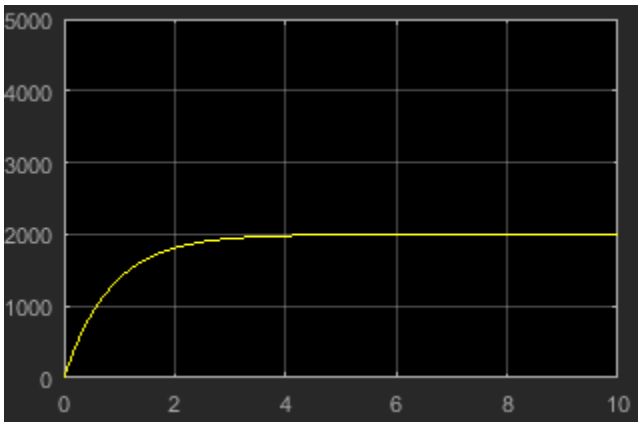
To view the motor current and torque, double-click the Scope blocks. You can do this before or after you run the simulation.

**Note** By default, the scope displays appear stacked on top of each other on the screen, so you can only see one of them. Click and drag the windows to reposition them.

This figure shows the motor current.




This figure shows the motor rpm.




As expected, the motor runs at about 2000 rpm when the applied DC voltage is 2.5 V.

## See Also

Controlled PWM Voltage | DC Motor | H-Bridge

## Related Examples

- “PWM-Controlled DC Motor” on page 8-450

## More About

- “Simulating an Electronic, Mechatronic, or Electrical Power System” on page 3-8
- “Build and Simulate a Simple Circuit” on page 2-16
- “Build and Simulate Single-Phase Half-Bridge Inverter With Ideal Switches and Thermal Port” on page 2-33

# Build and Simulate a Simple Circuit

| In this section... |
| --- |
| “Select Blocks to Represent System Components” on page 2-16<br>“Build the Model” on page 2-17<br>“Specify Model Parameters” on page 2-18<br>“Configure the Solver Parameters” on page 2-20<br>“Simulate Model and Analyze Results” on page 2-21 |

In this example, you model a triangle wave generator using Simscape Electrical blocks and custom Simscape Electrical blocks, and then look at the voltage at the wave generator output.

You use a classic circuit configuration consisting of an integrator and a noninverting amplifier to generate the triangle wave, and use datasheets to specify block parameters. For more information, see “Parameterizing Blocks from Datasheets” on page 3-13.

To see the completed model, open the “Triangle Wave Generator” on page 8-630 example.

## Select Blocks to Represent System Components

First, you select the blocks to represent the input signal, the triangle wave generator, and the output signal display.

You model the triangle wave generator with a set of physical blocks. The wave generator consists of:

- Two operational amplifier blocks
- Resistors and a capacitor that work with the operational amplifiers to create the integrator and noninverting amplifier
- Simulink-PS Converter and PS-Simulink Converter blocks whose function is to bridge the physical part of the model, which uses physical signals, and the rest of the model, which uses Simulink signals.

You have a manufacturer datasheet for the two operational amplifiers you want to model. Later in the example, you use the datasheet to parameterize the Simscape Electrical Band-Limited Op-Amp block.

The following table describes the role of the blocks that represent the system components.

| Block | Description |
| --- | --- |
| **Sine Wave** | Generates a sinusoidal signal that controls the resistance of the Variable Resistor block |
| **Scope** | Displays the triangular output wave |
| **Simulink-PS Converter** | Converts the sinusoidal Simulink signal to a physical signal |
| **Solver Configuration** | Defines solver settings that apply to all physical modeling blocks |
| **PS-Simulink Converter** | Converts the output physical signal to a Simulink signal |
| **Capacitor** | Works with an operational amplifier and resistor block to create the integrator |

| Block | Description |
|---|---|
| **Resistor** | Works with the operational amplifier and capacitor blocks to create the integrator and noninverting amplifier |
| **Variable Resistor** | Supplies a time-varying resistance that adjusts the gain of the integrator, which in turn varies the frequency and amplitude of the generated triangular wave |
| **DC Voltage Source** | Generates a DC reference signal for the operational amplifier block of the noninverting amplifier |
| **Voltage Sensor** | Converts the electrical voltage at the output of the integrator into a physical signal proportional to the current |
| **Electrical Reference** | Provides the electrical ground |
| **Band-Limited Op-Amp** | Works with the capacitor and resistor to create an integrator and a noninverting amplifier |
| **Diode** | Limits the output of the Band-Limited Op-Amp block, to make the output waveform independent of supply voltage |

## Build the Model

Create a Simulink model, add blocks to the model, and connect the blocks.

1. Create a new model.
2. Add to the model the blocks listed in this table. The Library Path column of the table specifies the hierarchical path to each block.

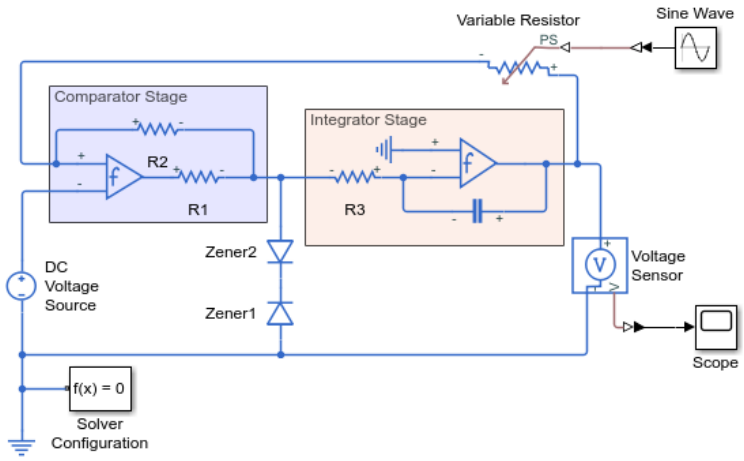
| Block | Library Path | Quantity |
|---|---|---|
| **Sine Wave** | **Simulink > Sources** | 1 |
| **Scope** | **Simulink > Commonly Used Blocks** | 1 |
| **Simulink-PS Converter** | **Simscape > Utilities** | 1 |
| **Solver Configuration** | **Simscape > Utilities** | 1 |
| **PS-Simulink Converter** | **Simscape > Utilities** | 1 |
| **Capacitor** | **Simscape > Foundation Library > Electrical > Electrical Elements** | 1 |
| **Resistor** | **Simscape > Foundation Library > Electrical > Electrical Elements** | 3 |
| **Variable Resistor** | **Simscape > Foundation Library > Electrical > Electrical Elements** | 1 |
| **Electrical Reference** | **Simscape > Foundation Library > Electrical > Electrical Elements** | 2 |
| **DC Voltage Source** | **Simscape > Foundation Library > Electrical > Electrical Sources** | 1 |
| **Voltage Sensor** | **Simscape > Foundation Library > Electrical > Electrical Sensors** | 1 |

| Block | Library Path | Quantity |
|---|---|---|
| **Band-Limited Op-Amp** | **Simscape > Electrical > Integrated Circuits** | 2 |
| **Diode** | **Simscape > Electrical > Semiconductor & Converters** | 2 |

**Note** You can use the Simscape function `ssc_new` with a domain type of `electrical` to create a Simscape model that contains these blocks:

- **Simulink-PS Converter**
- **PS-Simulink Converter**
- **Scope**
- **Solver Configuration**
- **Electrical Reference**

3 Rename and connect the blocks as shown in the diagram. The blocks in the triangle wave generator circuit are organized in two stages. The Comparator Stage contains a comparator constructed from a Band-Limited Op-Amp block and two Resistor blocks. The Integrator Stage contains an integrator constructed from another Band-Limited Op-Amp block, a Resistor, a Capacitor, and Electrical Reference.




## Specify Model Parameters

Specify these parameters to represent the behavior of the system components:


- “Model Setup Parameters” on page 2-18
- “Input Signal Parameters” on page 2-19
- “Triangle Wave Generator Parameters” on page 2-19
- “Signal Display Parameters” on page 2-20


### Model Setup Parameters

These blocks specify model information that is not specific to a particular block:

- Solver Configuration
- Electrical Reference

As with Simscape models, you must include a Solver Configuration block in each topologically distinct physical network. This example has a single physical network, so use one Solver Configuration block with the default parameter values.

You must include an Electrical Reference block in each Simscape Electrical network. This block does not have any parameters.

### Input Signal Parameters

Generate the sinusoidal control signal using the Sine Wave block.

Set the Sine Wave block parameters as follows:

- **Amplitude** — `0.5e4`
- **Bias** — `1e4`
- **Frequency** — `pi/5e-4`

### Triangle Wave Generator Parameters

Configure the blocks modeling the physical system that generates the triangle wave:

- Integrator stage — Band-Limited Op-Amp, Capacitor, and Resistor block R3
- Comparator stage — Band-Limited Op-Amp1, Resistor blocks R1 and R2
- Variable Resistor
- Diode and Diode1
- Simulink-PS Converter and PS-Simulink Converter blocks that bridge the physical part of the model and the Simulink part of the model.

1. Accept the default parameters for the Simulink-PS Converter block. These parameters establish the units of the physical signal at the block output such that they match the expected default units of the Variable Resistor block input.
2. Set the two Band-Limited Op-Amp block parameters for the LM7301 device with a +–20V power supply:
   - The datasheet gives the gain as 97 dB, which is equivalent to 10 ^ ( 97 / 20 ) = 7.1e4. Set the **Gain, A** parameter to `7.1e4`.
   - The datasheet gives input resistance as 39 Mohms. Set **Input resistance, Rin** to `39e6`.
   - Set **Output resistance, Rout** to `0` ohms. The datasheet does not quote a value for Rout, but the term is insignificant compared to the output resistor that it drives.
   - Set minimum and maximum output voltages to –20 V and +20 V, respectively.
   - The datasheet gives the maximum slew rate as 1.25 V/µs. Set the **Maximum slew rate, Vdot** parameter to `1.25e6` V/s.
   - Set the bandwidth to `4e6`.
3. Set the two Diode block parameters for a 4.3 V zener diode. To model a BZX384-B4V3, set block parameters as follows:

- On the **Main** tab, set **Diode model** to `Piecewise Linear`. This selects a simplified Zener diode model that is more than adequate to test the correct operation of this circuit.
- Leave the **Forward voltage** as 0.6 V — this is a typical value for most diodes.
- The datasheet gives the forward current as 250 mA when the forward voltage is 1V. So that the Diode block matches this, set the **On resistance** to (1 V – 0.6 V) / 250 mA = `1.6` ohms.
- The datasheet gives the reverse leakage current as 3 µA at a reverse voltage of 1 V. Therefore, set the **Off conductance** to 3 µA / 1 V = `3e-6` S.
- The datasheet gives the reverse voltage as 4.3 V. On the Breakdown tab, set the **Reverse breakdown voltage Vz** to `4.3` V.
- Set the **Zener resistance Rz** to a suitably small number. The datasheet quotes the Zener voltage for a reverse current of 5 mA. For the Diode block to be representative of the real device, the simulated reverse voltage should be close to 4.3V at 5mA. As Rz tends to zero, the reverse breakdown voltage tends to Vz regardless of current, as the voltage-current gradient becomes infinite. However, for good numerical properties, Rz must not be made too small. If, say, you allow a 0.01 V error on the Zener voltage at 5 mA, then Rz is 0.01 V / 5 mA = 2 ohms. Set the **Zener resistance** parameter to this value.

4. The Voltage Sensor block does not have any parameters.
5. Accept the default parameters for the Variable Resistor block. These parameters establish the units of the physical signal at the block output such that they match the expected default units of the Variable Resistor block input.
6. Set the Capacitor block parameters as follows:
   - **Capacitance** — `2.5e-9`
   - **Capacitor voltage** — `0.08`

     This value starts the oscillation in the feedback loop. It is found in the **Variables** tab.
   - **Series resistance** — `0`
7. Set the DC Voltage Source block **Constant voltage**parameter to `0`.
8. Set the Resistor R3 block **Resistance** parameter to `10000`.
9. Set the Resistor R1 block **Resistance** parameter to `1000`.
10. Set the Resistor R2 block **Resistance** parameter to `10000`.
11. Accept the default parameters for the PS-Simulink Converter block. These parameters establish the units of the physical signal at the block output such that they match the expected default units of the Scope block input.

#### Signal Display Parameters

Specify the parameters of the Scope block to display the triangular output signal.

Double-click the Scope block and then click the **View > Configuration Properties** to open the Scope Configuration Properties dialog box. On the **Logging** tab, clear the **Limit data points to last** check box.

## Configure the Solver Parameters

Configure the solver parameters to use a continuous-time solver.Simscape Electrical models only run with a continuous-time solver when the Simscape Solver Configuration block has its **Local Solver**

parameter cleared. You also change the simulation end time, tighten the relative tolerance for a more accurate simulation, and remove the limit on the number of simulation data points Simulink saves.

1. In the model window, select **Modeling** > **Model Settings** to open the Configuration Parameters dialog box.
2. In the **Solver** category in the tree on the left side of the dialog box:
   - Enter `2000e-6` for the **Stop time** parameter value.
   - Select `ode23t (Mod. stiff/Trapezoidal)` from the **Solver** list.
   - Enter `4e-5` for the **Max step size** parameter value.
   - Enter `1e-6` for the **Relative tolerance** parameter value.
3. In the **Data Import/Export** category in the **Select** tree, clear the **Limit data points to last** check box.
4. Click **OK**.

For more information about configuring solver parameters, see “Simulating an Electronic, Mechatronic, or Electrical Power System” on page 3-8.

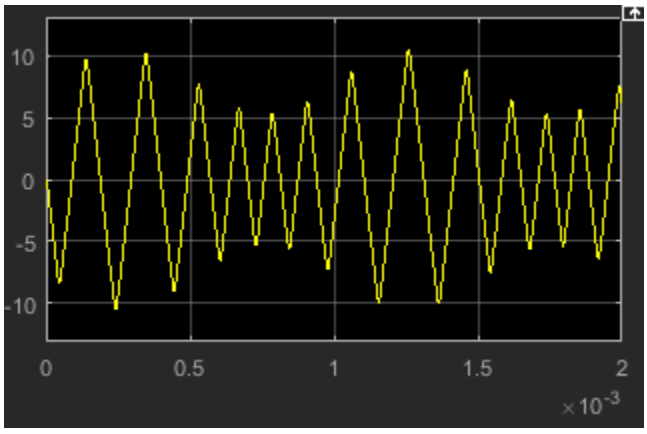
## Simulate Model and Analyze Results

Run the simulation and plot the results.

In the model window, select **Simulation** > **Run** to run the simulation.

To view the triangle wave in the Scope window, double-click the Scope block. You can do this before or after you run the simulation.

The following plot shows the voltage waveform. As the resistance of the Variable Resistor block increases, the amplitude of the output waveform increases and the frequency decreases.




**Triangle Waveform Voltage**

# Build and Simulate a Single-Phase Half-Bridge Inverter with Ideal Switches

| In this section... |
| --- |
| “Build the Model” on page 2-22<br>“Specify Model Parameters” on page 2-26<br>“Configure the Solver Parameters” on page 2-28<br>“Simulate the Model and Analyze the Results” on page 2-29 |

In this example, you build and analyze a single-phase, half-bridge inverter that you control by using sinusoidal pulse-width modulation (SPWM).

To see the completed version of the model you create in this example, at the MATLAB command prompt, enter `ee_half_bridge`.

## Build the Model

The single-phase, half-bridge inverter in this example consists of a power circuit and a control system. First, create both parts of the model by adding and connecting the blocks.

### Build the Power Circuit

The power circuit comprises a DC link, two semiconductor switches with their anti-parallel diodes, and an inductive load.

To build the power circuit:

1. Create a new model.
2. Add the blocks in this table. The **Library Path** column specifies the hierarchical path to each block.

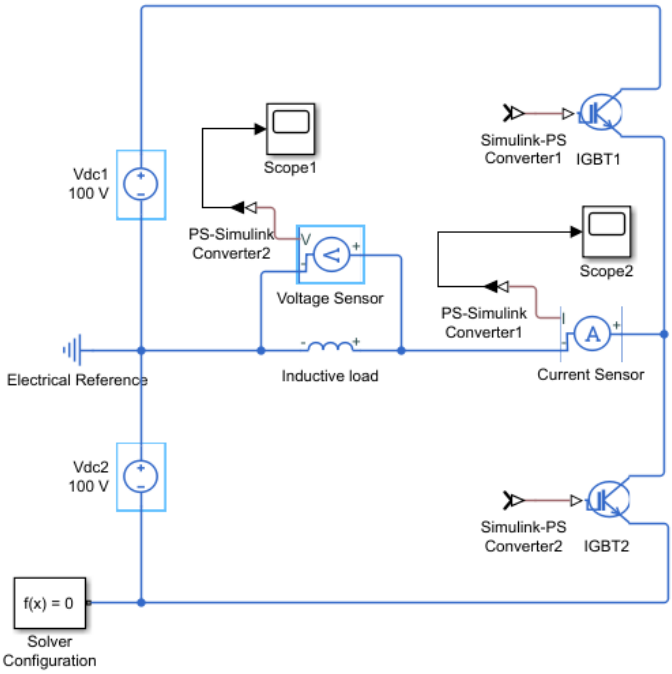
| Block | Purpose | Library Path | Quantity |
| --- | --- | --- | --- |
| **DC Voltage Source** | Provide a DC link | **Simscape > Foundation Library > Electrical > Electrical Sources** | 2 |
| **IGBT (Ideal, Switching)** | Model two semiconductor switches with their anti-parallel diodes and build the half-bridge | **Simscape > Electrical > Semiconductor & Converters** | 2 |
| **Inductor** | Model an inductive load | **Simscape > Foundation Library > Electrical > Electrical Elements** | 1 |
| **Electrical Reference** | Provide the ground connection for the electrical conserving ports | **Simscape > Foundation Library > Electrical > Electrical Elements** | 1 |

| Block | Purpose | Library Path | Quantity |
|---|---|---|---|
| **Voltage Sensor** | Measure the voltage across the load | **Simscape > Foundation Library > Electrical > Electrical Sensors** | 1 |
| **Current Sensor** | Measure the current that flows through the load | **Simscape > Foundation Library > Electrical > Electrical Sensors** | 1 |
| **Solver Configuration** | Define the solver settings that apply to all physical modeling blocks | **Simscape > Utilities** | 1 |
| **Simulink-PS Converter** | Convert an output physical signal to a Simulink signal | **Simscape > Utilities** | 2 |
| **PS-Simulink Converter** | Convert a Simulink input signal into a physical signal | **Simscape > Utilities** | 2 |
| **Scope** | Display the load voltage and the current wave | **Simulink > Commonly Used Blocks** | 2 |

**Note** You can use the Simscape function `ssc_new` with a domain type of `electrical` to create a Simscape model that contains these blocks:

- Simulink-PS Converter
- PS-Simulink Converter
- Scope
- Solver Configuration
- Electrical Reference

3 Rename and connect the blocks as shown in the diagram.




### Build the Control System

The control system generates two pulse-width modulated (PWM) gate signals that control the switching behavior of the two IGBT blocks. If the gate signal value is equal to `1`, the IGBT (Ideal, Switching) block turns on. If the gate signal value is equal to `0`, the block turns off.

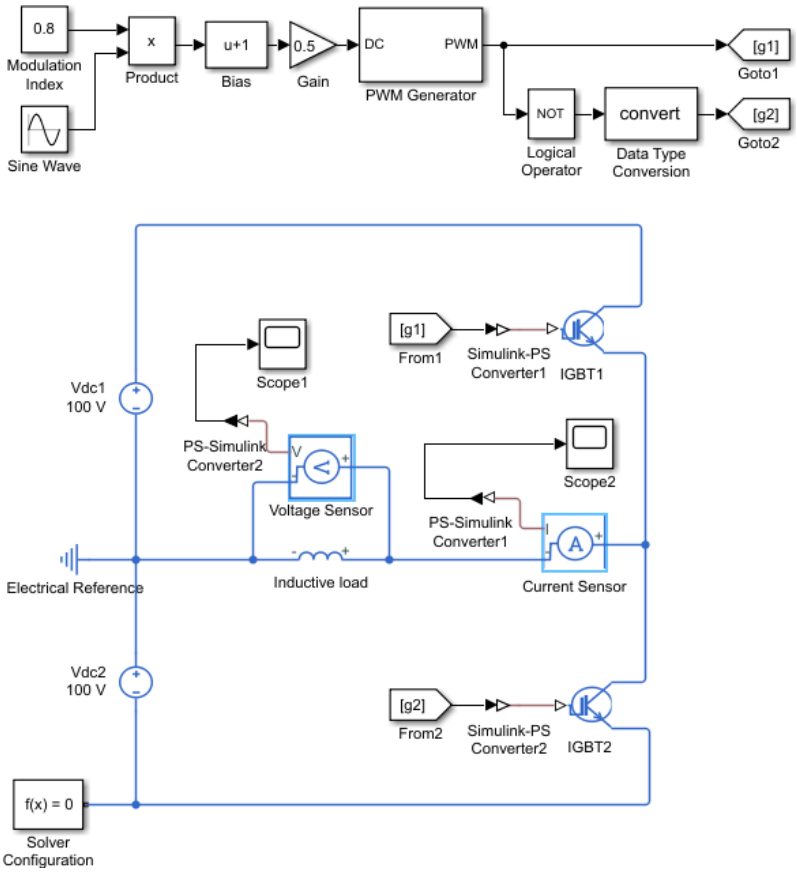
To build the control system:

1 Add the blocks in this table to the model. The **Library Path** column specifies the hierarchical path to each block.

| Block | Purpose | Library Path | Quantity |
|---|---|---|---|
| **Sine Wave** | Generate a sinusoidal signal that represents a unit sinusoidal modulation wave | **Simulink > Sources** | 1 |
| **Constant** | Define the sinusoidal pulse-width modulation index | **Simulink > Sources** | 1 |
| **Product** | Obtain the sinusoidal modulation wave by multiplying the unit sinusoidal wave by the modulation index | **Simulink > Math Operations** | 1 |
| **Bias** | Perform calculation to obtain the duty cycle signal | **Simulink > Math Operations** | 1 |

| Block | Purpose | Library Path | Quantity |
|---|---|---|---|
| **Gain** | | **Simulink > Math Operations** | 1 |
| **PWM Generator** | Generate the PWM gate signal for the `IGBT1` block | **Simscape > Electrical > Control > Pulse Width Modulation** | 1 |
| **Logical Operator** | Obtain the complementary PWM gate signal for the `IGBT2` block | **Simulink > Logic and Bit Operations** | 1 |
| **Data Type Conversion** | Convert the input Boolean signal to double signal | **Simulink > Signal Attributes** | 1 |
| **Goto** | Pass the input PWM gate signal to the corresponding From block | **Simulink > Signal Routing** | 2 |
| **From** | Receive a PWM gate signal from the corresponding Goto block and pass the signal as an output | **Simulink > Signal Routing** | 2 |

**2** Rename and connect the blocks as shown in the diagram.




## Specify Model Parameters

Specify these parameters to represent the behavior of the system components:

- “Model Setup Parameters” on page 2-26
- “Power Circuit Parameters” on page 2-27
- “Control System Parameters” on page 2-27
- “Signal Display Parameters” on page 2-28

### Model Setup Parameters

These blocks specify model information that is not specific to a particular block:

- Solver Configuration
- Electrical Reference

As with Simscape models, you must include a Solver Configuration block in each topologically distinct physical network. This example has a single physical network, so use one Solver Configuration block with the default parameter values.

You must include an Electrical Reference block in each Simscape Electrical network. This block does not have any parameters.

### Power Circuit Parameters

Specify these parameters to represent the behavior of the power circuit components:

1. Set the **Constant voltage** parameter of the two DC Voltage Source blocks to `100`.
2. Set the parameters of the two IGBT (Ideal, Switching) blocks:
   - **Main** settings
     - **Forward voltage, Vf** — `0.8`
     - **On-state resistance** — `0.001`
     - **Off-state conductance** — `1e-5`
     - **Threshold voltage, Vth** — `0.5`. With this value, the IGBT switches on when the value at the input port is equal to `1` and it switches off when the value at the input port is equal to `0`.
   - **Integral Diode** settings
     - **Integral protection diode** — `Protection diode with no dynamics`
     - **Forward voltage** — `0.8`
     - **On resistance** — `0.001`
     - **Off conductance** — `1e-5`
3. Set the parameters of the Inductor block:
   - **Inductance** — `0.005`
   - **Series resistance** — `1`
   - **Parallel conductance** — `0`

### Control System Parameters

Specify the following parameters to get the PWM gate signals for the `IGBT1` and `IGBT2` from a sinusoidal modulation wave:

1. In the Constant block, set the **Constant value** parameter to `0.8`.
2. To generate a unit sinusoidal modulation wave, set the parameters of the Sine Wave block:
   - **Amplitude** — `1`
   - **Bias** — `0`
   - **Frequency (rad/sec)** — `2*pi*60`
   - **Phase (rad)** — `0`
   - **Sample time** — `0`
3. In the Bias block, set the **Bias** parameter to `1`.
4. In the Gain block, set the **Gain** parameter to `0.5`.
5. Set the PWM Generator block parameters:
   - **Carrier counter** — `Up-Down`
   - **Timer period (s)** — `1/2/2000`
   - **Phase delay (s)** — `0`

- **Sample time** — `0`

6 In the Logical Operator block, set the **Operator** parameter to `NOT`.

The sinusoidal modulation wave is equal to $0.8 \cdot \sin(2\pi \cdot 60 \cdot t)$. To change the modulation wave, modify the modulation index, frequency, and phase values.

The PWM Generator block cannot receive the modulation wave as an input, because it must be a duty cycle. To obtain the duty cycle from the modulation wave, this example uses this equation:

$$D = \frac{M + 1}{2}$$

where $D$ is the duty cycle and $M$ is the modulation wave. The Bias and Gain blocks of the control system implement this calculation.

The output PWM gate signal from the PWM Generator block controls the `IGBT1` block. The Logical Operator block implements the complementary PWM gate signal that controls the `IGBT2` block. The value of the **Timer period (s)** parameter of the PWM Generator block is equal to `1/2/2000`, which means that the switching frequency is equal to `2 kHz`.

The Goto blocks and From blocks implement the connection between the generated PWM gate signals and the two IGBT (Ideal, Switching) blocks. Setting the **Goto tag** parameter in a Goto and From block to the same tag allows you to pass a signal from one block to another without connecting them.

#### Signal Display Parameters

Specify the parameters of the Scope block to display the output signals.

Double-click the Scope blocks and then click the **View > Configuration Properties** to open the Scope Configuration Properties dialog box. On the **Logging** tab, clear the **Limit data points to last** check box.

## Configure the Solver Parameters

Configure the solver parameters to use a continuous-time solver. Simscape Electrical models only run with a continuous-time solver when you clear the **Local Solver** parameter of a Solver Configuration block. Use the Configuration Parameters dialog box to change the simulation end time, tighten the relative tolerance for a more accurate simulation, and remove the limit on the number of simulation data points Simulink saves.

1 In the model window, select **Modeling > Model Settings** to open the Configuration Parameters dialog box.

2 In the left pane, click **Solver**:

- Set **Stop time** to `4/60`.
- Set **Solver** to `ode23t (Mod. stiff/Trapezoidal)`.
- Set **Max step size** to `1e-4`.
- Set **Relative tolerance** to `1e-4`.

3 Click **OK**.

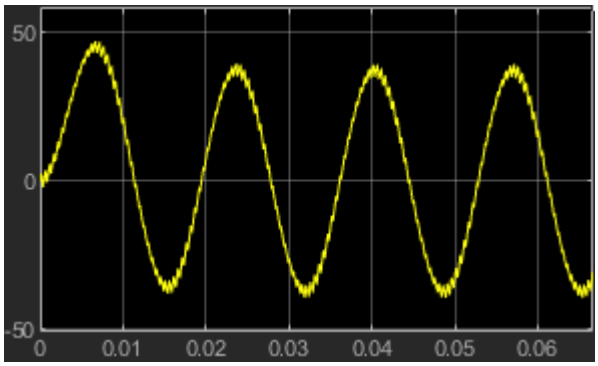
For more information about configuring solver parameters, see “Simulating an Electronic, Mechatronic, or Electrical Power System” on page 3-8.

## Simulate the Model and Analyze the Results

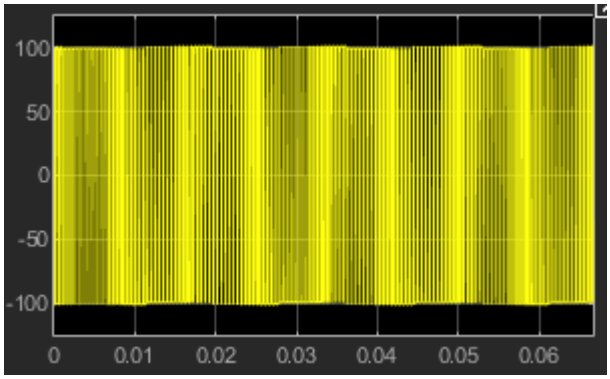
Run the simulation and plot the results. In the model window, select **Simulation > Run**.

To view the load voltage and current, double-click the Scope blocks. You can do this before or after you run the simulation.

This figure shows the load current.




The next figure shows the load voltage.

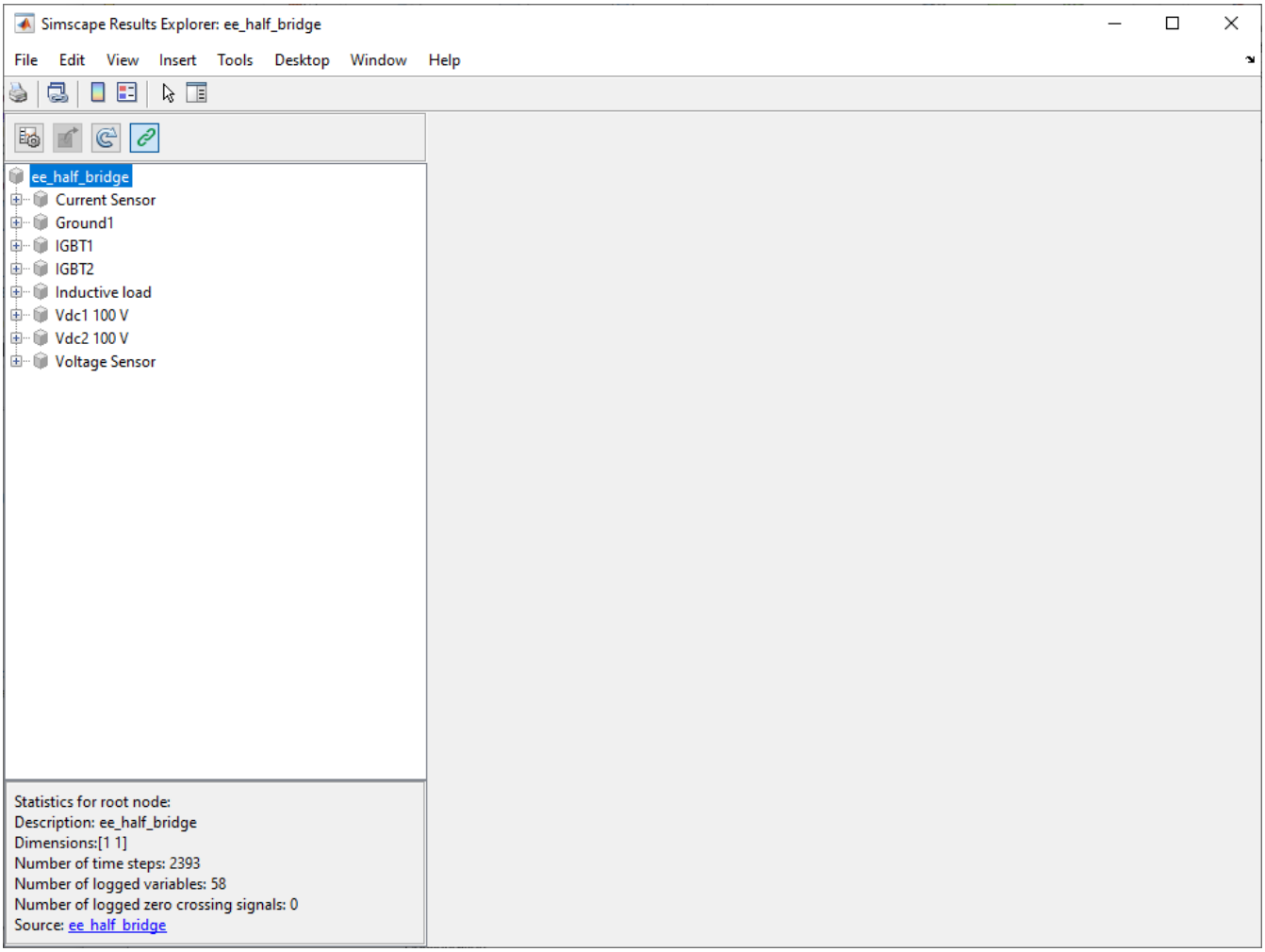



As expected, the load voltage is a PWM wave varying between -100 V and 100 V.

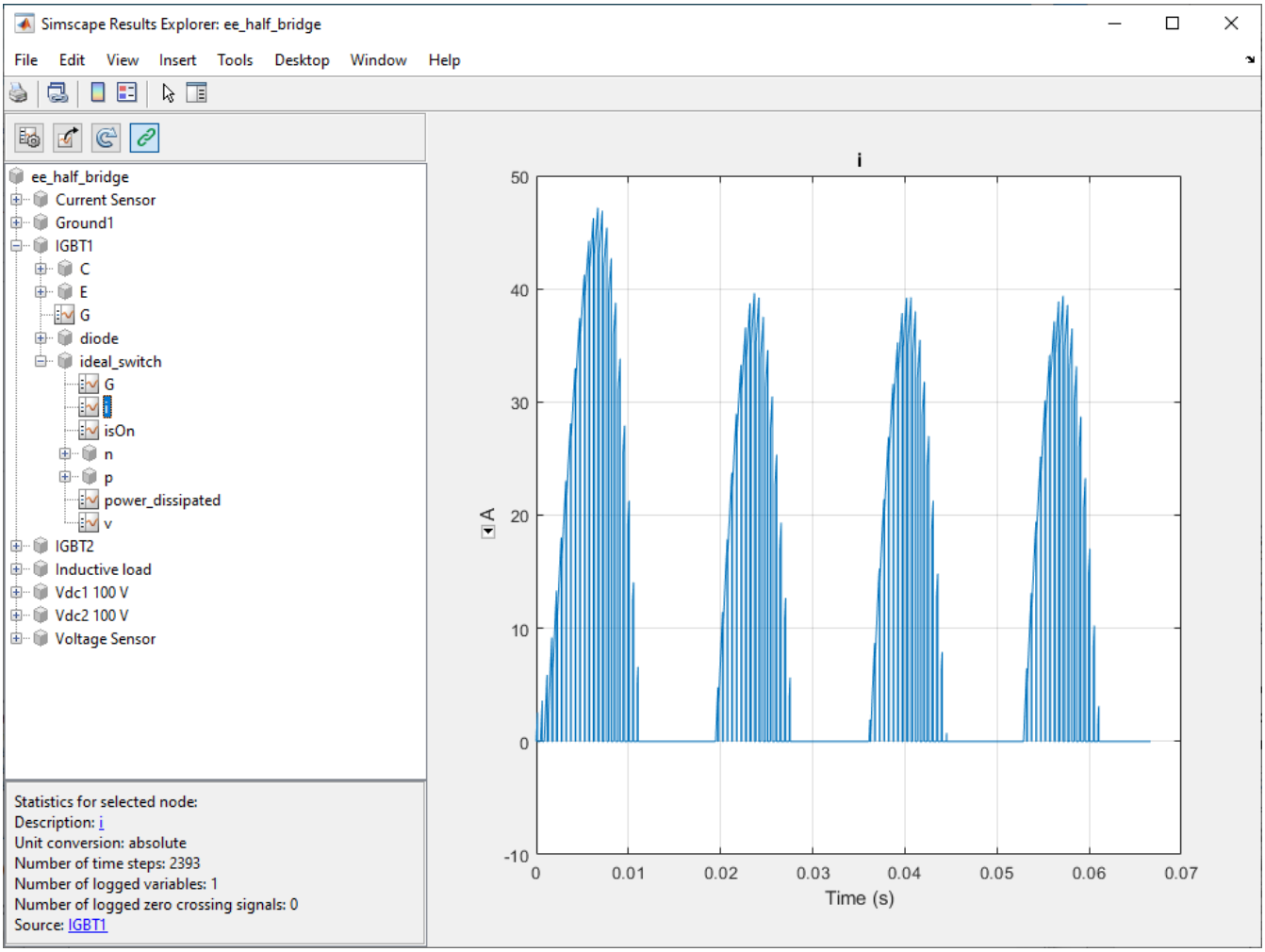
To view other simulation results, such as the current flowing through each IGBT (Ideal, Switching) block and its anti-parallel diode, use the Simscape Results Explorer. To enable the Simscape Results Explorer, first log the simulation data for all variables:

- In the model window, select **Modeling > Model Settings** to open the Configuration Parameters dialog box.
- In the left pane, click **Simscape**:
  - Set **Log simulation data** to `All`.
  - Select the **Open viewer after simulation** check box.
  - Clear the **Limit data points** check box.
- Click **OK**.

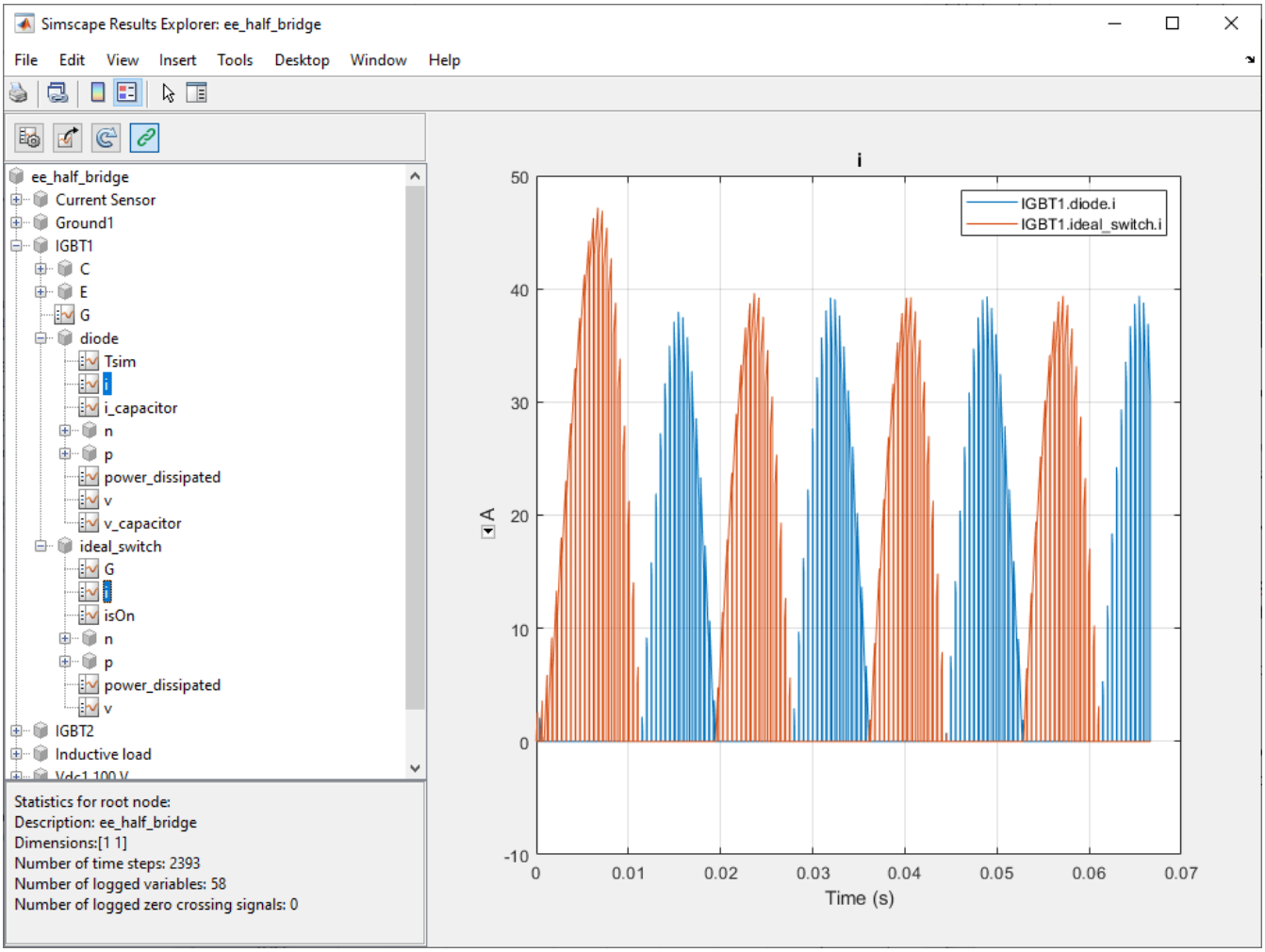
In the model window, select **Simulation > Run** to run the simulation. This figure shows the Simscape Results Explorer window that opens automatically.




When you click a node in the left pane, the corresponding plots appear in the right pane. For example, if you click the variable **i** in **IGBT1 > ideal_switch**, then the Simscape Results Explorer plots the current that flows through the `IGBT1` block in the half-bridge.




To select several variables for side-by-side plot comparison, press **Ctrl** and click multiple variables. For example, check the current that flows through the `IGBT1` block and the current that flows through the anti-parallel diode of the block:




For more information, see “About the Simscape Results Explorer”.

## See Also

IGBT (Ideal, Switching) | Inductor

# Build and Simulate Single-Phase Half-Bridge Inverter With Ideal Switches and Thermal Port

| In this section... |
| --- |
| “Build the Model” on page 2-33<br>“Specify Model Parameters” on page 2-37<br>“Configure the Solver Parameters” on page 2-40<br>“Simulate the Model and Analyze the Results” on page 2-40 |

In this example, you build and analyze a single-phase, half-bridge inverter that you control by using sinusoidal pulse-width modulation (SPWM) and simulate the effects of generated heat and device temperature.

To see the completed version of the model you create in this example, at the MATLAB command prompt, enter `ee_half_bridge_thermal`.

## Build the Model

The single-phase, half-bridge inverter in this example consists of a power circuit and a control system. First, create both parts of the model by adding and connecting the blocks.

### Build the Power Circuit

The power circuit comprises a DC link, two semiconductor switches with their anti-parallel diode, and an inductive load.

To build the power circuit:

1. Create a new model.
2. Add the blocks in this table. The **Library Path** column specifies the hierarchical path to each block.

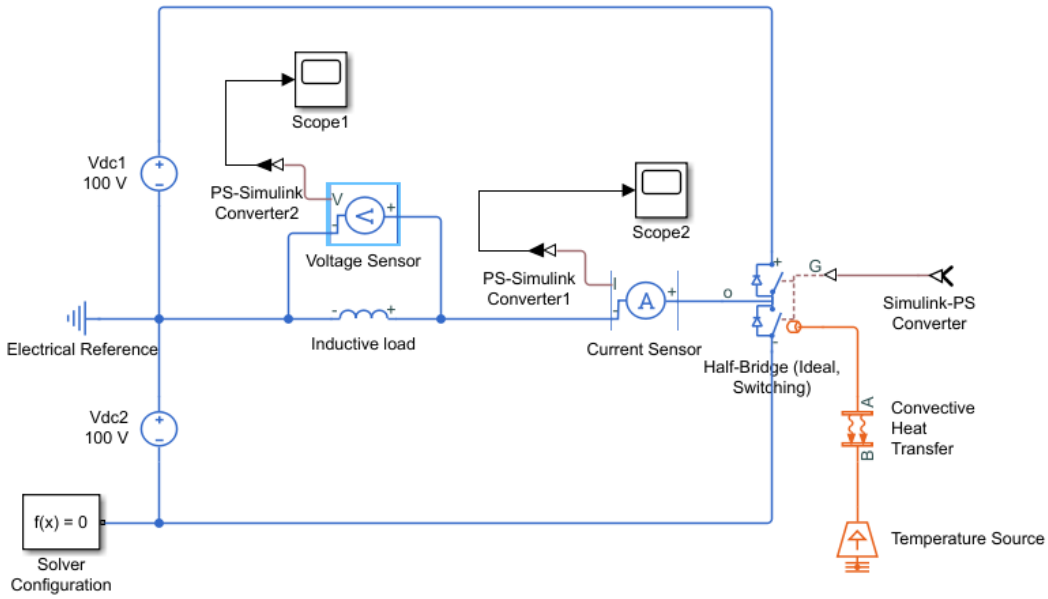
| Block | Purpose | Library Path | Quantity |
| --- | --- | --- | --- |
| **DC Voltage Source** | Provide a DC link | **Simscape > Foundation Library > Electrical > Electrical Sources** | 2 |
| **Half-Bridge (Ideal, Switching)** | Model two semiconductor switches with their anti-parallel diodes and build the half-bridge with thermal effects | **Simscape > Electrical > Semiconductor & Converters** | 1 |
| **Inductor** | Model an inductive load | **Simscape > Foundation Library > Electrical > Electrical Elements** | 1 |

| Block | Purpose | Library Path | Quantity |
|---|---|---|---|
| **Convective Heat Transfer** | Represent a heat transfer by convection from the half-bridge to the ambient | **Simscape > Foundation Library > Thermal > Thermal Elements** | 1 |
| **Temperature Source** | Define the ambient temperature | **Simscape > Foundation Library > Thermal > Thermal Sources** | 1 |
| **Electrical Reference** | Provide the ground connection for the electrical conserving ports | **Simscape > Foundation Library > Electrical > Electrical Elements** | 1 |
| **Voltage Sensor** | Measure the voltage across the load | **Simscape > Foundation Library > Electrical > Electrical Sensors** | 1 |
| **Current Sensor** | Measure the current that flows through the load | **Simscape > Foundation Library > Electrical > Electrical Sensors** | 1 |
| **Solver Configuration** | Define the solver settings that apply to all physical modeling blocks | **Simscape > Utilities** | 1 |
| **Simulink-PS Converter** | Convert an output physical signal to a Simulink signal | **Simscape > Utilities** | 2 |
| **PS-Simulink Converter** | Convert a Simulink input signal into physical signal | **Simscape > Utilities** | 2 |
| **Scope** | Display the load voltage and the current wave | **Simulink > Commonly Used Blocks** | 2 |

**Note** You can use the Simscape function `ssc_new` with a domain type of `electrical` to create a Simscape model that contains these blocks:

- Simulink-PS Converter
- PS-Simulink Converter
- Scope
- Solver Configuration
- Electrical Reference

3 Rename and connect the blocks as shown in the diagram.




To build the power circuit, this example uses a Half-Bridge (Ideal, Switching) block instead of an IGBT (Ideal, Switching) block. Similarly to the Half-Bridge (Ideal, Switching) block, you can calculate and parameterize the conduction loss of the IGBT and its anti-parallel diode, and the switching-on and switching-off losses of the IGBT. However, you cannot calculate the reverse recovery loss of the diode by using the same lookup table approach. To account for the reverse recovery loss, you need to specify the parameters associated with the diode charging dynamics. If you include the diode charging dynamics, the simulation requires more computational effort, compromising its performance.

### Build the Control

The control system generates two pulse-width modulated (PWM) gate signals that control the switching behavior of two IGBTs in the Half-Bridge (Ideal, Switching) block.

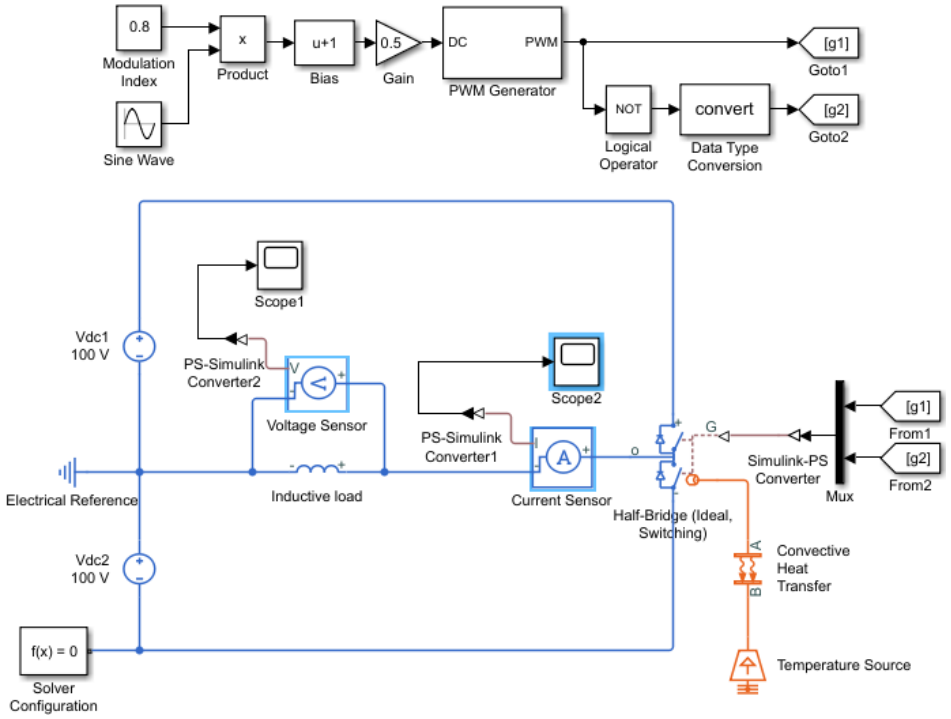
To build the control system:

1 Add the blocks in this table to the model. The **Library Path** column specifies the hierarchical path to each block.

| Block | Purpose | Library Path | Quantity |
|---|---|---|---|
| **Sine Wave** | Generate a sinusoidal signal that represents a unit sinusoidal modulation wave | **Simulink > Sources** | 1 |
| **Constant** | Define the sinusoidal pulse-width modulation index | **Simulink > Sources** | 1 |

| Block | Purpose | Library Path | Quantity |
|---|---|---|---|
| **Product** | Obtain the sinusoidal modulation wave by multiplying the unit sinusoidal wave by the modulation index | **Simulink > Math Operations** | 1 |
| **Bias** | Perform calculation to obtain the duty cycle signal | **Simulink > Math Operations** | 1 |
| **Gain** | | **Simulink > Math Operations** | 1 |
| **PWM Generator** | Generate PWM gate signal for the upper IGBT of the Half-Bridge (Ideal, Switching) block | **Simscape > Electrical > Control > Pulse Width Modulation** | 1 |
| **Logical Operator** | Obtain the complementary PWM gate signal for the lower IGBT of the Half-Bridge (Ideal, Switching) block | **Simulink > Logic and Bit Operations** | 1 |
| **Data Type Conversion** | Convert the input Boolean signal to double signal | **Simulink > Signal Attributes** | 1 |
| **Goto** | Pass the input PWM gate signal to the corresponding From block | **Simulink > Signal Routing** | 2 |
| **From** | Receive a PWM gate signal from the corresponding Goto block and pass the signal as an output | **Simulink > Signal Routing** | 2 |
| **Mux** | Combine two PWM gate signals | **Simulink > Signal Routing** | 1 |

2 Rename and connect the blocks as shown in the diagram.




## Specify Model Parameters

Specify these parameters to represent the behavior of the system components:

- “Model Setup Parameters” on page 2-37
- “Power Circuit Parameters” on page 2-37
- “Control System Parameters” on page 2-39
- “Signal Display Parameters” on page 2-40

### Model Setup Parameters

These blocks specify model information that is not specific to a particular block:

- Solver Configuration
- Electrical Reference

As with Simscape models, you must include a Solver Configuration block in each topologically distinct physical network. This example has a single physical network, so use one Solver Configuration block with the default parameter values.

You must include an Electrical Reference block in each Simscape Electrical network. This block does not have any parameters.

### Power Circuit Parameters

Specify these parameters to represent the behavior of the power circuit components:

1 Set the **Constant voltage** parameter of the two DC Voltage Source blocks to `100`.

2. In the Half-Bridge (Ideal, Switching) block, set the parameters as follows:
   - **Main** settings
     - **Gate-control port** — `PS`
     - **Switching device** — `IGBT`
     - **Threshold voltage, Vth** — `0.5`. With this value, the Half-Bridge (Ideal, Switching) switches on when the value at the input port is equal to `1` and switches off when the value at the input port is equal to `0`. If you set the threshold voltage to the actual threshold value, then you must change the value at the input gate ports accordingly.
     - **On-state behavior and losses** — `Tabulate with temperature and current`
   - **Integral Diode** settings
     - **Integral protection diode** — `Yes`
     - **Diode model** — `Tabulated I-V curve`
     - **Table type** — `Table in If(Tj,Vf) form`
   - **Thermal Port** settings
     - **Thermal network** — `External`
     - **Junction thermal mass** — `0.1`
   - **Variables** settings
     - Select **Override Junction temperature Target**.
     - **Priority** — `High`
     - **Value** — `25`
3. In the Inductor block, set these parameters:
   - **Inductance** — `0.005`
   - **Series resistance** — `1`
   - **Parallel conductance** — `0`
4. In the Convective Heat Transfer block, set these parameters:
   - **Area** — `1e-3`
   - **Heat transfer coefficient** — `20`
5. In the Temperature Source block, set the **Temperature** parameter to `298.15`.

For a semiconductor device, the generated heat, or thermal loss, includes:

- Conduction loss of the switch and its anti-parallel diode
- Switching-on and switching-off loss of the switch
- Reverse recovery loss of the anti-parallel diode

In this model, the Half-Bridge (Ideal, Switching) block models the semiconductor. The example models the thermal loses based on the voltage, current, and temperature. When the Half-Bridge (Ideal, Switching) block conducts current, the conduction loss is calculated by multiplying the on-state voltage across the half-bridge and the current that flows through itself.

A 2-D lookup table determines the value of the on-state voltage. To define the 2-D lookup table, in the **Main** settings, specify the values of the **On-state voltage Vce(Tj,Ice)**, **Temperature vector Tj**, and **Collector-emitter current vector Ic** parameters.

The conduction loss of the anti-parallel diode is calculated by using the **Forward currents If(Tj,Vf)**, **Junction temperatures, Tj**, and **Forward voltages, Vf** parameters in the **Integral Diode** settings.

A 2-D lookup table determines the value of the switching-on loss of the IGBT and then proportionates it to the **Off-state voltage for loss data** parameter. To define the 2-D lookup table, in the **Losses** settings, specify the values of the **Switch-on loss, Eon(Tj,Ice)**, **Temperature vector for losse,s Tj**, **Collector-emitter current vector for losses, Ice** parameters.

The switching-off loss of the IGBT is calculated by using the **Switch-off loss, Eoff(Tj,Ice)** parameter in the **Losses** settings and the reverse recovery loss of the diode is calculated by using the **Diode reverse recovery loss, Erec(Tj,Ice)** parameter.

The generated heat dissipates from the half-bridge to the ambient atmosphere. The Convective Heat Transfer and the Temperature Source blocks model this thermal behavior. For more options of thermal network modeling, see “Simulating Thermal Effects in Semiconductors” on page 3-31.

### Control System Parameters

Specify the following parameters to get the PWM gate signals for the upper and lower IGBTs from a sinusoidal modulation wave:

1. In the Constant block, set the **Constant value** parameter to `0.8`.
2. To generate a unit sinusoidal modulation wave, set the parameters of the Sine Wave block:
   - **Amplitude** — `1`
   - **Bias** — `0`
   - **Frequency (rad/sec)** — `2*pi*60`
   - **Phase (rad)** — `0`
   - **Sample time** — `0`
3. In the Bias block, set the **Bias** parameter to `1`.
4. In the Gain block, set the **Gain** parameter to `0.5`.
5. Set the PWM Generator block parameters:
   - **Carrier counter** — `Up-Down`
   - **Timer period (s)** — `1/2/2000`
   - **Phase delay (s)** — `0`
   - **Sample time** — `0`
6. In the Logical Operator block, set the **Operator** parameter to `NOT`.

The sinusoidal modulation wave is equal to $0.8 \cdot \sin(2\pi \cdot 60 \cdot t)$. To change the modulation wave, modify the modulation index, frequency, and phase values.

The PWM Generator block cannot receive the modulation wave as an input, because it must be a duty cycle. To obtain the duty cycle from the modulation wave, this example uses this equation:

$$D = \frac{M + 1}{2}$$

where *D* is the duty cycle and *M* is the modulation wave. The Bias and Gain blocks of the control system implement this calculation.

The output PWM gate signal from the PWM Generator block controls the upper IGBT. The Logical Operator block implements the complementary PWM gate signal that controls the lower IGBT. The value of the **Timer period (s)** parameter of the PWM Generator block is equal to `1/2/2000`, which means that the switching frequency is equal to `2 kHz`.

The Goto blocks and From blocks implement the connection between the generated PWM gate signals and the two IGBTs in the Half-Bridge (Ideal, Switching) block. Setting the **Goto tag** parameter in a Goto and From block to the same tag allows you to pass a signal from one block to another without connecting them.

#### Signal Display Parameters

Specify the parameters of the Scope block to display the output signals.

Double-click the Scope blocks and then click the **View > Configuration Properties** to open the Scope Configuration Properties dialog box. On the **Logging** tab, clear the **Limit data points to last** check box.

## Configure the Solver Parameters

Configure the solver parameters to use a continuous-time solver. Simscape Electrical models only run with a continuous-time solver when you clear the **Local Solver** parameter of a Solver Configuration block. Use the Configuration Parameters dialog to change the simulation end time, tighten the relative tolerance for a more accurate simulation, and remove the limit on the number of simulation data points Simulink saves.

1. In the model window, select **Modeling > Model Settings** to open the Configuration Parameters dialog box.
2. In the left pane, click **Solver**:
   - Set **Stop time** to `4/60`.
   - Set **Solver** to `ode23t (Mod. stiff/Trapezoidal)`.
   - Set **Max step size** to `1e-4`.
   - Set **Relative tolerance** to `1e-4`.
3. Click **OK**.

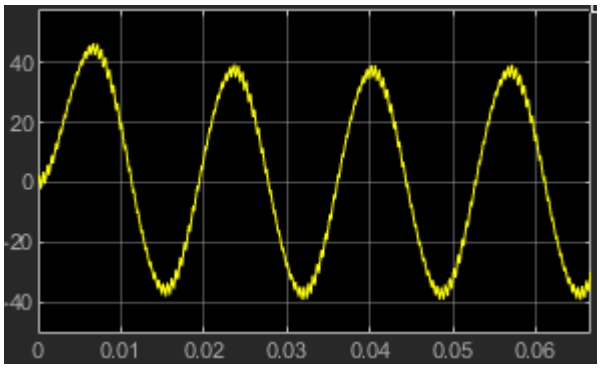
For more information about configuring solver parameters, see “Simulating an Electronic, Mechatronic, or Electrical Power System” on page 3-8.

## Simulate the Model and Analyze the Results

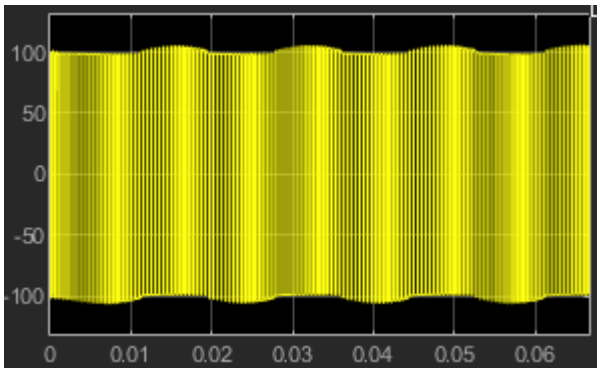
Run the simulation and plot the results. In the model window, select **SimulationRun**.

To view the load voltage and current, double-click the Scope blocks. You can do this before or after you run the simulation.

This figure shows the load current.




The next figure shows the load voltage.

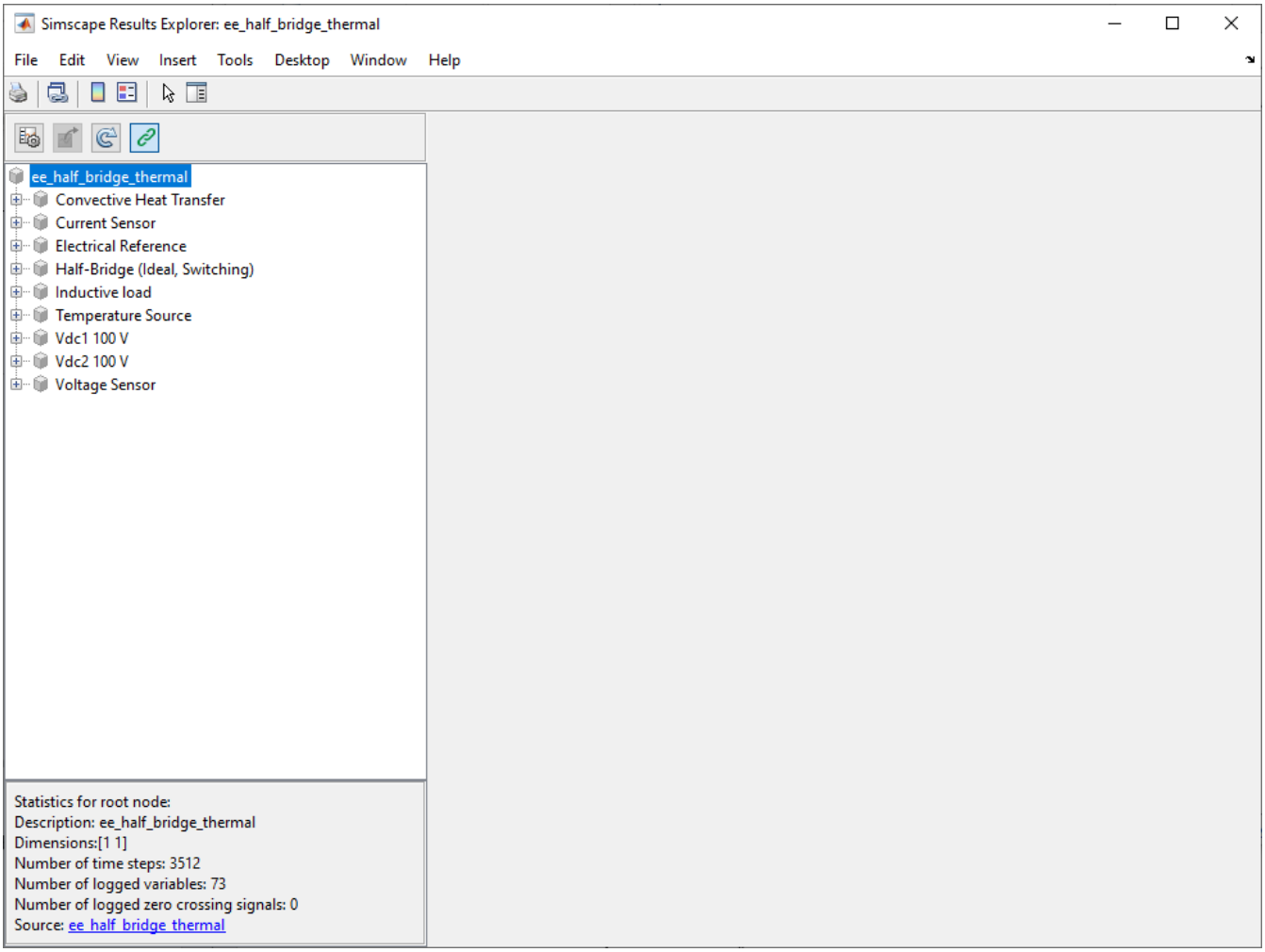



The load voltage is a PWM wave varying between -100 V and 100 V.

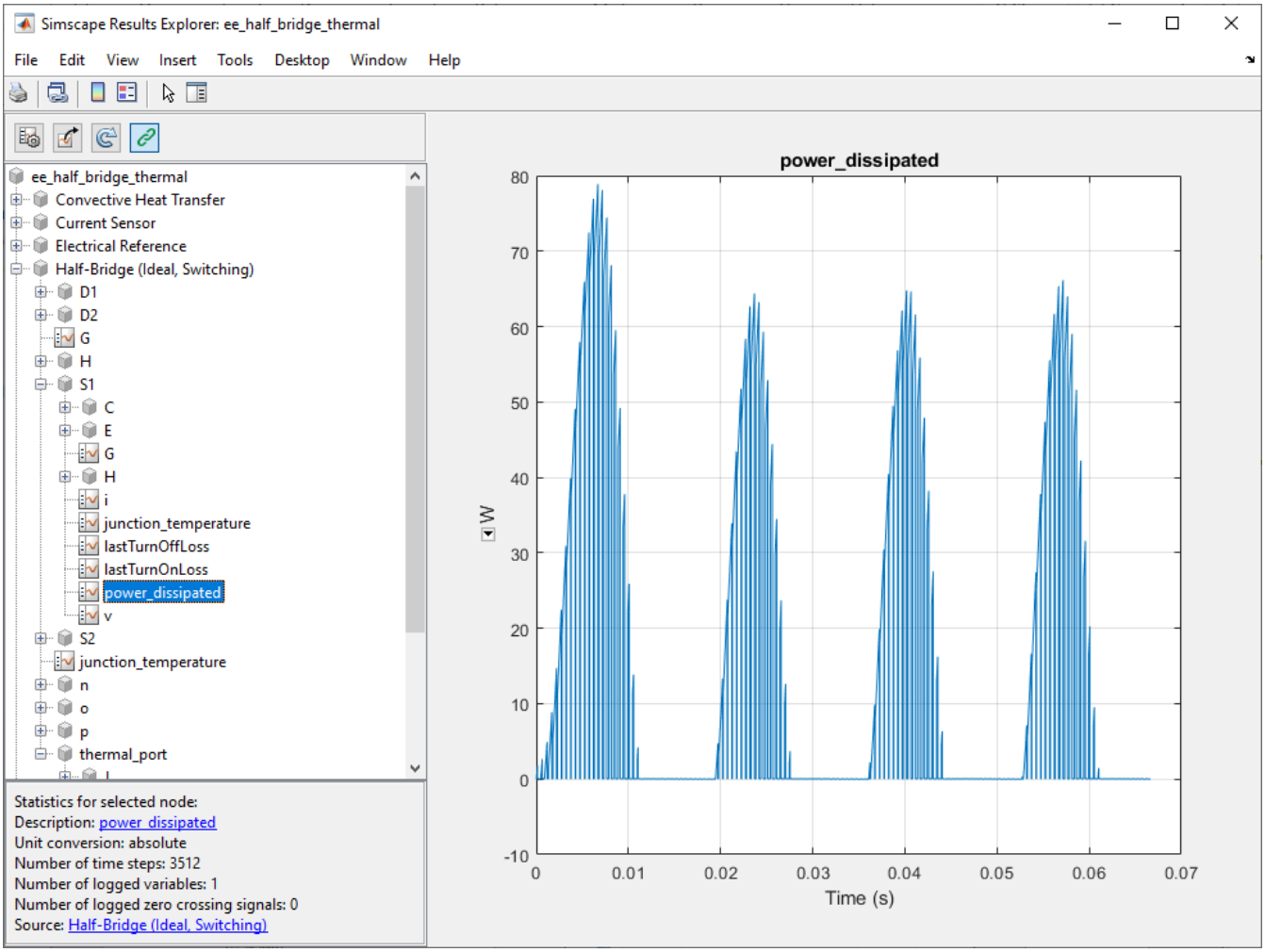
To view other simulation results, such as the conduction loss of the Half-Bridge (Ideal, Switching) block, use the Simscape Results Explorer. To enable the Simscape Results Explorer, first log the simulation data for all variables:

- In the model window, select **Modeling** > **Model Settings** to open the Configuration Parameters dialog box.
- In the left pane, click **Simscape**:
  - Set **Log simulation data** to `All`.
  - Select the **Open viewer after simulation** check box.
  - Clear the **Limit data points** check box.
- Click **OK**.

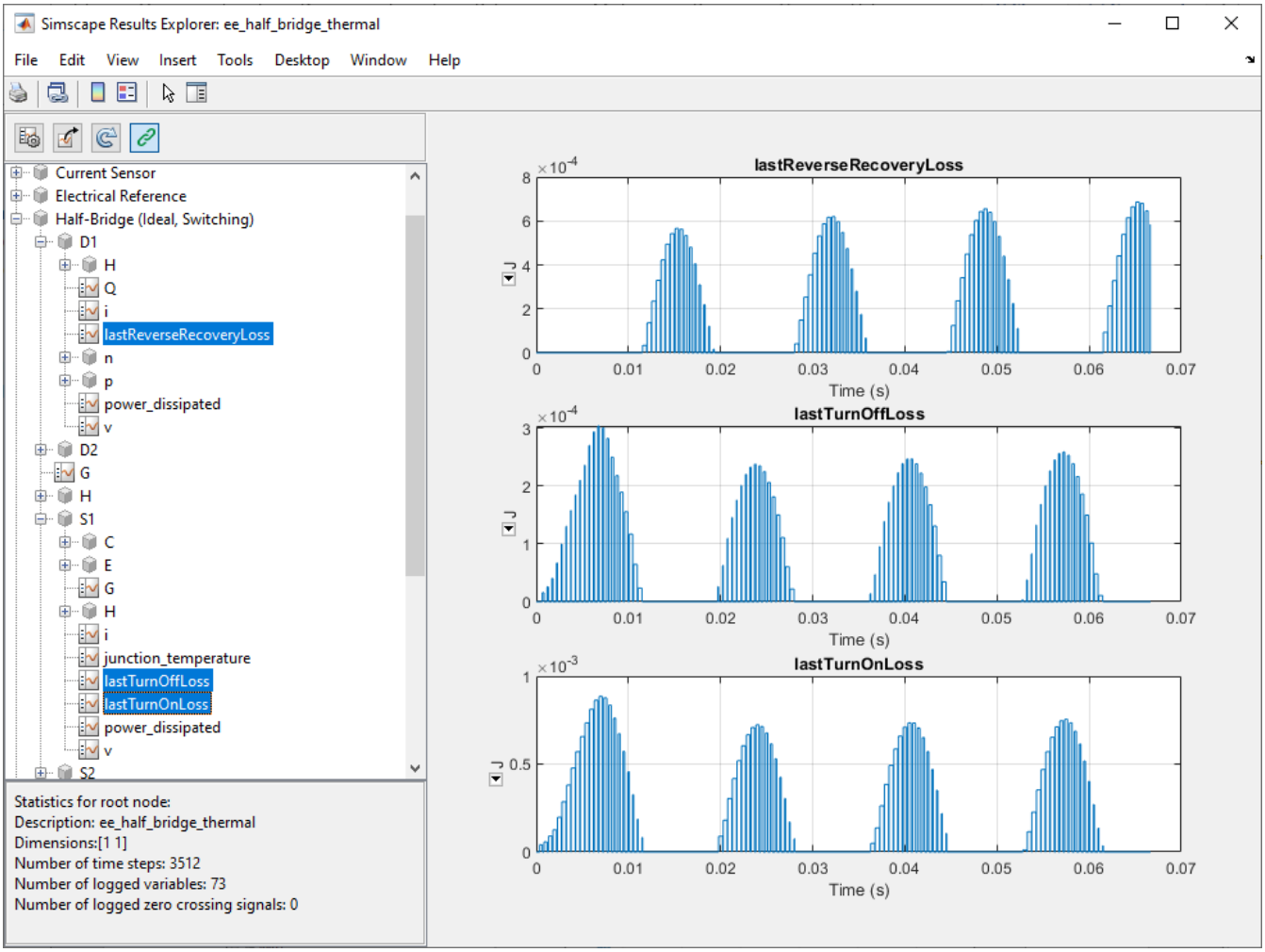
In the model window, select **Simulation** > **Run** to run the simulation. This figure shows the Simscape Results Explorer window that opens automatically.



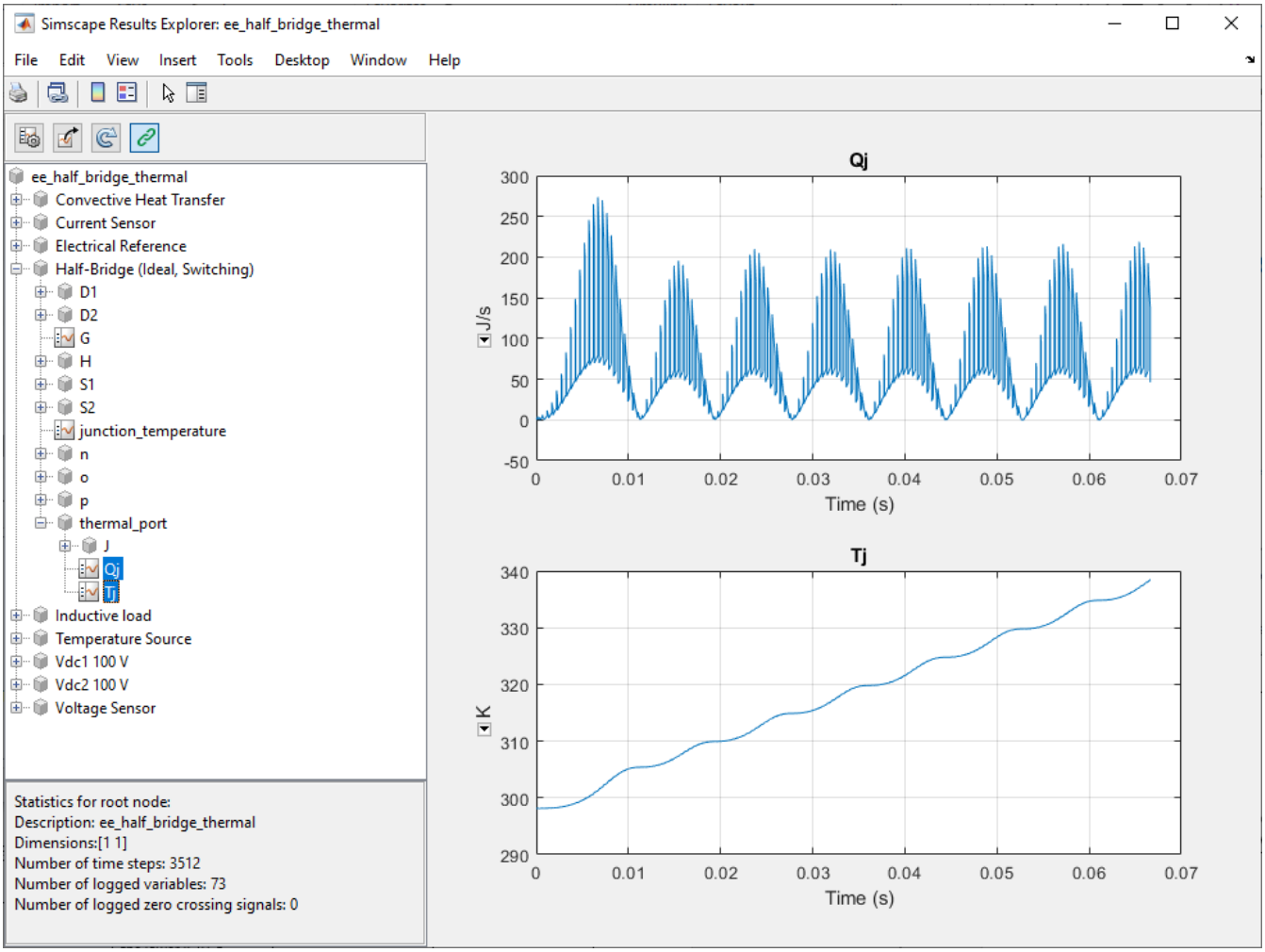

When you click a node in the left pane, the corresponding plots appear in the right pane. For example, if you click the variable **power_dissipated** in **Half-Bridge (Ideal, Switching) > S1**, then the Simscape Results Explorer plots the conduction loss of the upper IGBT in the half-bridge.




To select several variables for side-by-side plot comparison, press **Ctrl** and click multiple variables. For example, check the switching-on and switching-off loss of the upper IGBT and the reverse recovery loss of the anti-parallel diode:




You can also check the total power loss and the junction temperature of the half-bridge by clicking the **Qj** and **Tj** variables in **Half-Bridge (Ideal, Switching) > thermal_port**.




For more information, see “About the Simscape Results Explorer”.

## See Also

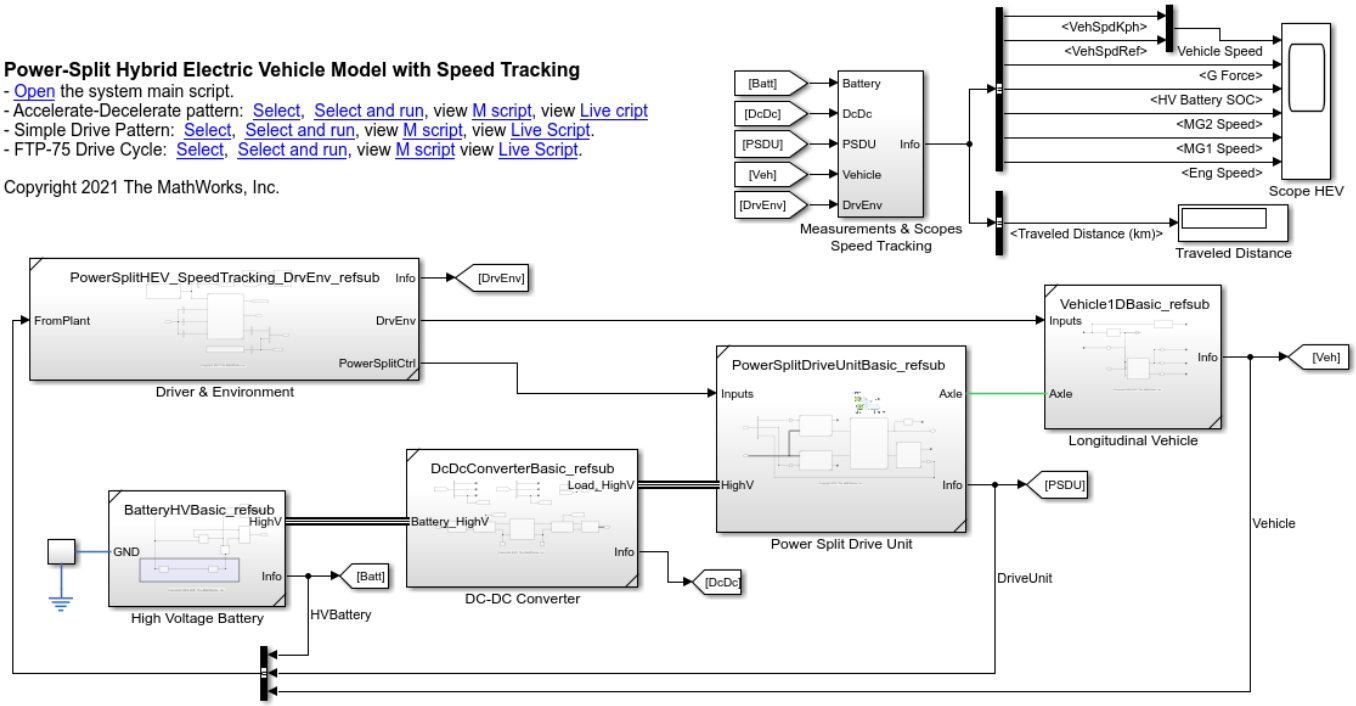
Half-Bridge (Ideal, Switching) | Inductor | Convective Heat Transfer | Temperature Source

# Multi-Domain Examples on GitHub

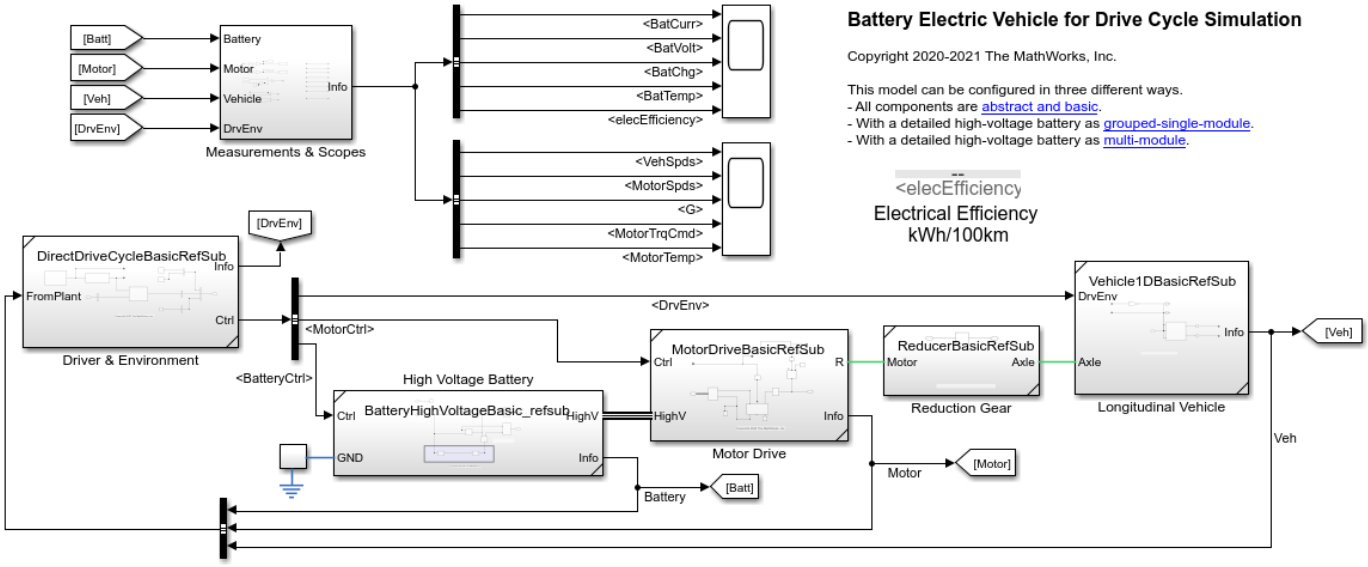
The MathWorks GitHub® page contains examples that illustrate modeling, control, and simulation of large systems using multiple MathWorks products. To find the latest models and examples, you can search for Simscape repositories in the MathWorks GitHub page: https://github.com/mathworks/?q=simscape.

For example:

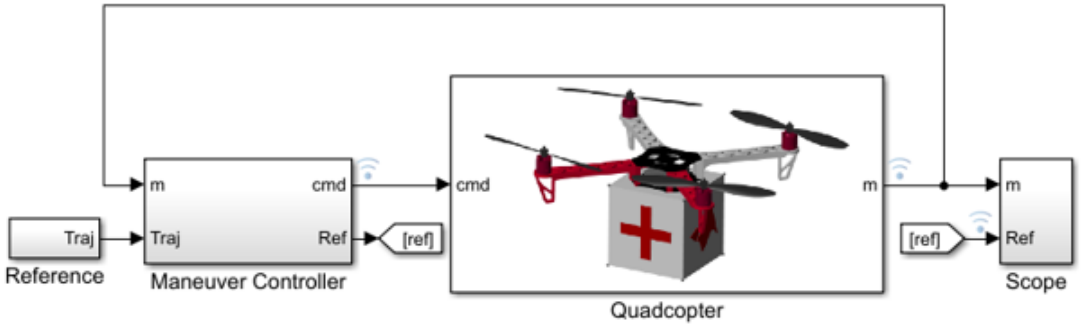
- For a power-split hybrid electric vehicle (HEV) model with a simple rule-based controller for speed tracking simulation, see the Hybrid Electric Vehicle Model in Simscape GitHub repository.




- For a battery electric vehicle (BEV) model and its components such as motor, high voltage battery, and longitudinal vehicle, see the Battery Electric Vehicle Model in Simscape GitHub repository.



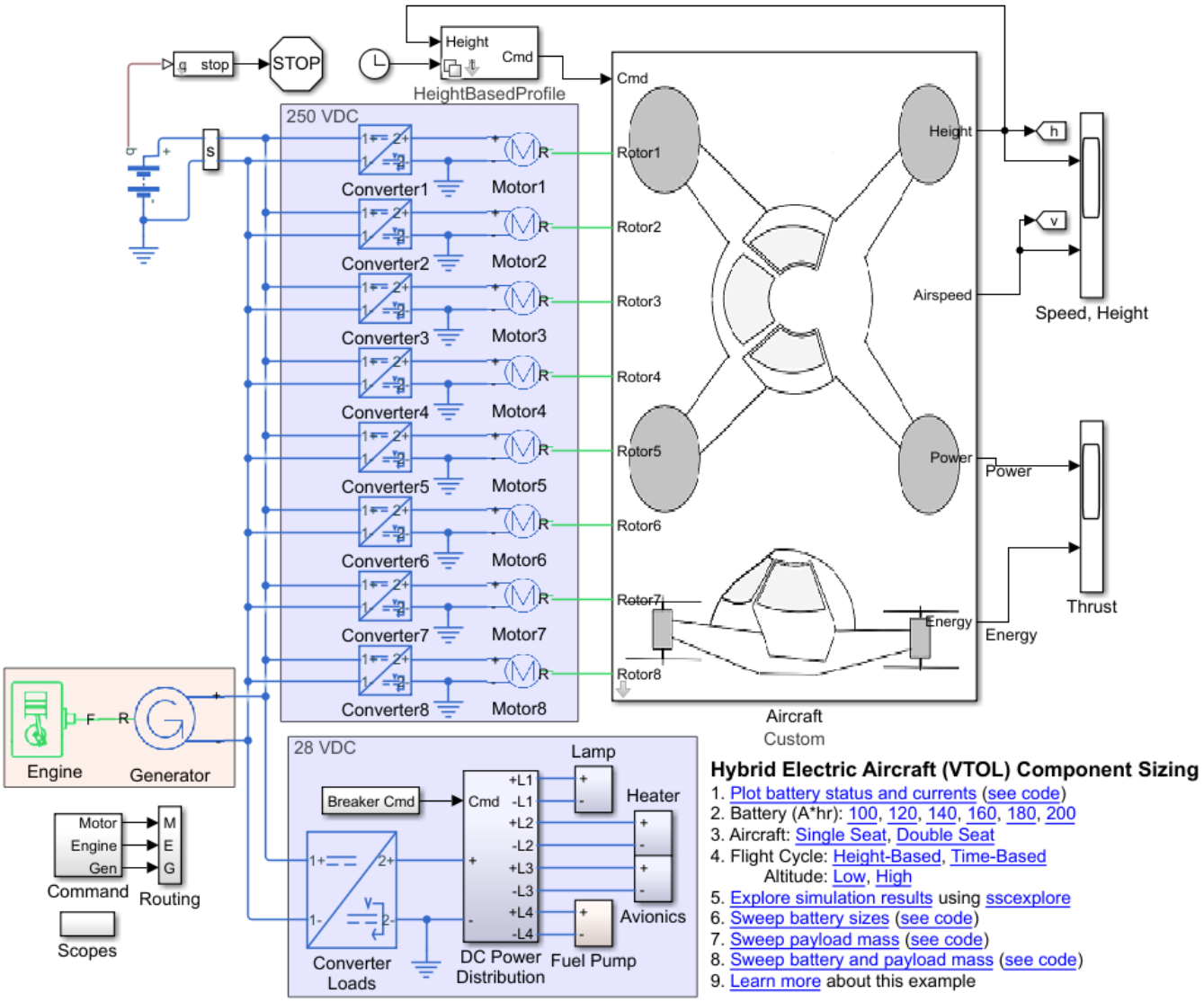

- For a quadcopter model that navigates a path to deliver a package, see the Quadcopter Drone Model in Simscape GitHub repository.




**Quadcopter Package Delivery**

1. Plot positions and velocities of chassis (see code)
2. Plot trajectory in 3D space (see code)
3. Define path: Path 1, Path 2, Path 3 (parameters, code)
4. Edit controller parameters
5. Parameter sweep: Package density, Trajectory speed
6. Share model: Convert to Protected Model Reference
7. Explore simulation results using sscexplore
8. Learn more about this example

- For aircraft models that evaluate the electric and hybrid electric architectures against design criteria such as flight range and flight duration, see the Electric Aircraft Model in Simscape GitHub repository.




## See Also

## More About

- “Applications”

# Modeling and Simulation Basics


- “Essential Electrical Modeling Techniques” on page 3-2
- “Three-Phase Ports” on page 3-5
- “Switch Between Physical Signal and Electrical Ports” on page 3-7
- “Simulating an Electronic, Mechatronic, or Electrical Power System” on page 3-8
- “Selecting the Output Model for Logic Blocks” on page 3-10
- “Parameterizing Blocks from Datasheets” on page 3-13
- “Parameterize a Piecewise Linear Diode Model from a Datasheet” on page 3-14
- “Parameterize an Exponential Diode from a Datasheet” on page 3-17
- “Parameterize an Exponential Diode from SPICE Netlist” on page 3-21
- “Parameterize an Op-Amp from a Datasheet” on page 3-25
- “Additional Parameterization Workflows” on page 3-27
- “Simulating Thermal Effects in Rotational and Translational Actuators” on page 3-28
- “Simulating Thermal Effects in Semiconductors” on page 3-31
- “Plot Basic Characteristics for Battery Blocks” on page 3-43
- “Plot Basic Characteristics for Semiconductor Blocks” on page 3-45
- “MOSFET Characteristics Viewer” on page 3-48
- “Converting a SPICE Netlist to Simscape Blocks” on page 3-56
- “Photovoltaic Thermal (PV/T) Hybrid Solar Panel” on page 3-66
- “List of Pre-Parameterized Components” on page 3-75

# Essential Electrical Modeling Techniques

| In this section... |
| --- |
| “Overview of Modeling Rules” on page 3-2 |
| “Required Blocks” on page 3-3 |
| “Creating a New Model” on page 3-3 |
| “Modeling Instantaneous Events” on page 3-3 |
| “Using Simulink Blocks to Model Physical Components” on page 3-4 |

## Overview of Modeling Rules

Simscape Electrical models are essentially Simscape block diagrams refined for modeling single- and multi-phase electronic, mechatronic, and electrical power systems. Simscape Electrical blocks feature these port types:

- Three-phase ports, which connect the phases of a three-phase electrical system between Simscape Electrical blocks.

  There are two three-phase port types in Simscape Electrical blocks, composite and expanded. You can connect a composite three-phase port only to another composite three-phase port. You can connect the individual electrical conserving ports of an expanded three-phase port only to other electrical conserving ports. For more information, see “Three-Phase Ports” on page 3-5.
- Electrical and mechanical rotational conserving ports, which connect directly to Simscape foundation blocks.

  Each port type has specific Across and Through variables associated with it. To learn about the rules to follow when building an electromechanical model, see “Basic Principles of Modeling Physical Networks”.
- Physical signal ports, which connect to Simulink blocks through Simulink-PS Converter and PS-Simulink Converter blocks from the Simscape Utilities library. These blocks convert physical signals to and from Simulink mathematical signals.

Keep these rules in mind when using each port type in Simscape Electrical blocks.

- You can connect physical conserving ports only to other conserving ports of the same type. Electrical conserving ports in Simscape Electrical blocks can connect directly to Simscape electrical components. Mechanical rotational conserving ports in Simscape Electrical blocks can connect directly to Simscape mechanical rotational components.
- The physical connection lines that connect conserving ports are nondirectional lines that carry physical variables (Across and Though variables) rather than signals. You cannot connect physical conserving ports to Simulink ports or to physical signal ports.
- You can branch physical connection lines. When you do so, directly connected components have the same Across variables. The value of any Through variable (e.g., current or torque) transferred along the physical connection line is divided among the multiple components connected by the branches.

  For each Through variable, the sum of the values flowing into a branch point equals the sum of the values flowing out.

- You can connect physical signal ports to other physical signal ports using regular connection lines, similar to Simulink signal connections. These connection lines carry physical signals between Simscape Electrical blocks.
- You can connect physical signal ports to Simulink ports through converter blocks. Use the Simulink-PS Converter block to connect Simulink outports to physical signal inports. Use the PS-Simulink Converter block to connect physical signal outports to Simulink inports.
- Unlike Simulink signals, physical signals can have units. In Simscape Electrical block dialog boxes, you can specify the units along with the parameter values, where appropriate. Use the converter blocks to associate units with an input signal and to specify the desired output signal units.

For an example of these rules applied to an electromechanical model, see “Three-Phase Asynchronous Machine Starting” on page 8-759.

## Required Blocks

Each topologically distinct physical network in a diagram requires exactly one Solver Configuration block from the Simscape Utilities library. The Solver Configuration block specifies global environment information for simulation and provides parameters for the solver that your model needs for simulation.

Each electrical network requires an Electrical Reference block. This block establishes the electrical ground for the circuit. Networks with electromechanical blocks also require a Mechanical Rotational Reference block. For more information about using reference blocks, see “Grounding Rules”.

## Creating a New Model

An easy way to start a new Simscape Electrical model, prepopulated with the required blocks, is to use the Simscape function `ssc_new`. For more information, see “Creating a New Simscape Model”.

Another way to start a new model is to use a Simscape template from the Simulink start page. The start page includes model templates that provide you with design patterns for modeling electrical, three-phase electrical, mechanical rotational, and mechanical translational networks using Simscape Electrical. For more information, see “Modeling Analog Circuit Architectures, Mechatronic Systems, and Electrical Power Systems Using Simscape Electrical” on page 1-6.

You can also use the “Creating a New Circuit” example as a template for a new electronic circuit model. This example opens a simple electrical model, prepopulated with some useful blocks, and also opens an Electrical Starter Palette, which contains links to the most often used electrical components. Open the example by typing `ssc_new_elec` in the MATLAB Command Window and use **File > Save As** to save the example model under the desired name. Then delete the unwanted blocks and add new ones from the Electrical Starter Palette and from the block libraries.

## Modeling Instantaneous Events

When working with Simscape Electrical, your model may include Simulink blocks that are associated with events or discrete sampling. Such blocks can create instantaneous changes to the physical system inputs through the Simulink-PS Converter block that connects them. When you build this type of model, make sure that the corresponding zero crossings are generated.

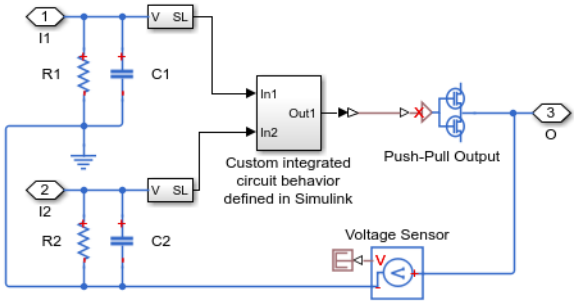
Many blocks in the Simulink library generate these zero crossings by default. For example, the Pulse Generator block produces a discrete-time output by default, and generates the corresponding zero

crossings. To generate zero crossings for all Simulink blocks that model instantaneous events, in the Solver Configuration Parameters for the model, expand **Solver details** and in the **Zero crossing options**, for the **Zero crossing control** option, select `Use local settings` or `Enable all`. For more information about zero crossing control, see “Zero-crossing control”.

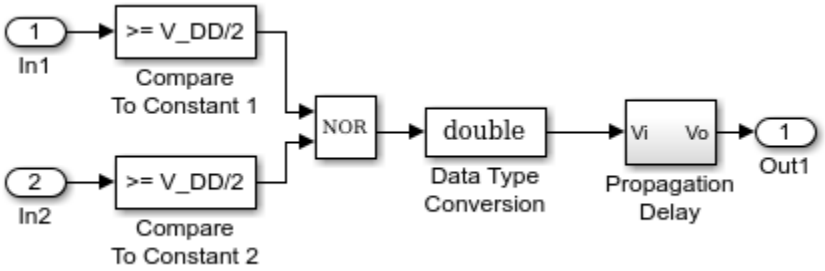
## Using Simulink Blocks to Model Physical Components

To run a fast simulation that approximates the behavior of the physical components in a system, you may want to use Simulink blocks to model one or more physical components.

The “Modeling an Integrated Circuit” on page 8-638 example uses Simulink to model a physical component. The 2-Input NOR (Behavioral Model) masked subsystem is a behavioral model, built using Simscape Foundation Library blocks.




This behavioral model contains a subsystem comprised of Simulink blocks, which implements the custom integrated-circuit behavior.




The Simulink Logical Operator block implements the behavioral model of the two-input NOR gate. Using Simulink in this manner introduces algebraic loops, unless you place a lag somewhere between the physical signal inputs and outputs. In this case, a first-order lag is included in the Propagation Delay subsystem to represent the delay due to gate capacitances. For applications where no lag is required, use blocks from the Physical Signals sublibrary in the Simscape Foundation Library to implement the desired functionality.

# Three-Phase Ports

| In this section... |
| --- |
| “About Three-Phase Ports” on page 3-5 |
| “Expand and Collapse Three-Phase Ports on a Block” on page 3-6 |

## About Three-Phase Ports

In Simscape Electrical software, you can connect the phases of a three-phase system between blocks using two types of ports.

- Composite three-phase port
- Expanded three-phase port

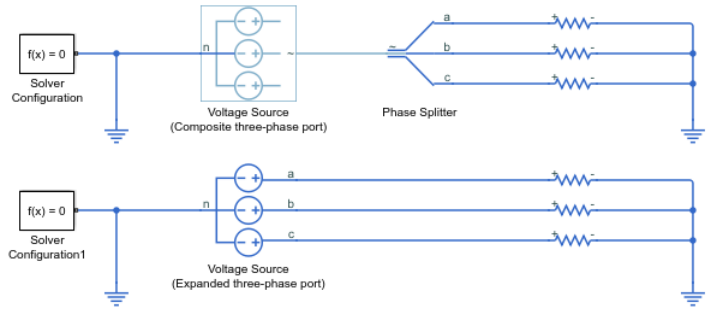
Composite three-phase ports represent three individual electrical conserving ports with a single block port. You can use composite three-phase ports to build models that correspond to single-line diagrams of three-phase electrical systems. Instead of explicitly connecting each phase of the three-phase system between blocks, you connect all three phases using a single port. You can connect composite three-phase ports only to other composite three-phase ports.

Expanded three-phase ports represent the individual phases of a three-phase system using three separate electrical conserving ports. You individually connect each phase of the three-phase system between blocks. Electrical conserving ports can connect directly to electrical components from the Simscape and Simscape Electrical libraries.

Composite three-phase ports produce results with the same fidelity as expanded three-phase ports. Both connection methods consider instantaneous phase voltages and currents and are suitable for modeling balanced and unbalanced three-phase electrical power systems. Each electrical conserving port in an expanded three-phase port has a Through variable of scalar current and an Across variable of scalar voltage. For a composite three-phase port, the Through variable is a three-element current, and the Across variable is a three-element voltage.

You can use the Phase Splitter block to expand a composite three-phase port into separate electrical conserving ports. The separate electrical ports can then connect to Simscape Electrical electrical components.

The figure shows two simple circuits that contrast the composite and expanded connection methods. The two circuits produce the same results.



The top circuit uses a Voltage Source block with a composite three-phase port ~. The bottom circuit uses a Voltage Source block with expanded electrical conserving ports **a**, **b**, and **c**. In each circuit, the instantaneous phase voltages and currents are the same.

## Expand and Collapse Three-Phase Ports on a Block

Simscape Electrical blocks that have composite three-phase ports have an option to switch between composite and expanded ports.

To switch between composite and expanded ports, double-click the block, then set the **Modeling option** or the **Electrical connection** parameter to either `Expanded three-phase ports` or `Composite three-phase ports`.

For blocks with a single composite port **~**, the expanded electrical ports are labeled **a**, **b**, and **c**. For blocks with more than one composite port **~1** and **~2**, the expanded electrical ports are labeled **a1**, **b1**, **c1** and **a2**, **b2**, **c2**.

# Switch Between Physical Signal and Electrical Ports

Some blocks in **Semiconductors and Converters** and **Switches and Breakers** Simscape Electrical libraries have an option to switch certain ports between physical signal and electrical conserving ports. An electrical conserving port is a Simscape physical conserving port that has a Through variable of current and an Across variable of voltage. For a comparison of Simscape physical signal and physical conserving ports, see “Connector Ports and Connection Lines”.

To switch a block connection port between an electrical conserving port ■ and a physical signal port ▷, double-click the block and set the **Modeling option** parameter to an option that includes the term **Electrical** for an electrical conserving port, or **PS** for a physical signal port.

# Simulating an Electronic, Mechatronic, or Electrical Power System

| In this section... |
| --- |
| “Selecting a Solver” on page 3-8 |
| “Specifying Simulation Accuracy/Speed Tradeoff” on page 3-8 |
| “Avoiding Simulation Issues” on page 3-9 |
| “Running a Time-Domain Simulation” on page 3-9 |
| “Running a Small-Signal Frequency-Domain Analysis” on page 3-9 |

## Selecting a Solver

Simscape Electrical software supports all of the continuous-time solvers that Simscape supports. For more information, see “Setting Up Solvers for Physical Models”.

You can select any of the supported solvers for running a simulation of an electronic model. The variable-step solvers, `ode23t` and `ode15s`, are recommended for most applications because they run faster and work better for systems with a range of both fast and slow dynamics. The `ode23t` solver is closest to the solver that SPICE traditionally uses.

To use Simulink Coder software to generate standalone C or C++ code from your model, you must use the `ode14x` or `ode1be` solvers. For more information about code generation, see “Code Generation”.

## Specifying Simulation Accuracy/Speed Tradeoff

To trade off accuracy and simulation time, adjust one or more of the following parameters:

- **Relative tolerance** in the Simulink Configuration Parameters dialog box
- **Absolute tolerance** in the Simulink Configuration Parameters dialog box
- **Max step size** in the Simulink Configuration Parameters dialog box
- **Consistency Tolerance** in the Solver Configuration block dialog box

In most cases, the default tolerance values produce accurate results without sacrificing unnecessary simulation time. The parameter value that is most likely to be inappropriate for your simulation is **Max step size**, because the default value, `auto`, depends on the simulation start and stop times rather than on the amount by which the signals are changing during the simulation. If you are concerned about the solver missing significant behavior, change the parameter to prevent the solver from taking too large a step.

The Simulink documentation describes the following parameters in more detail and provides tips on how to adjust them:

- “Relative tolerance”
- “Absolute tolerance”
- “Max step size”

The Solver Configuration block reference page in the Simscape documentation explains when to adjust the **Consistency Tolerance** parameter value.

## Avoiding Simulation Issues

If you experience a simulation issue, first read “Troubleshooting Simulation Errors” to learn about general troubleshooting techniques.

There are a few techniques you can apply to any Simscape Electrical model to overcome simulation issues:

- Add parasitic capacitors and/or resistors (specifically, junction capacitance and ohmic resistance) to the circuit to avoid numerical issues. The “Astable Oscillator” on page 8-632 example uses these devices.
- Adjust the current and voltage sources so they start at zero and ramp up to their final values rather than starting at nonzero values.

To learn about avoiding simulation errors in the presence of specific Simscape Electrical model configurations, see “Modeling Instantaneous Events” on page 3-3 and “Using Simulink Blocks to Model Physical Components” on page 3-4.

## Running a Time-Domain Simulation

When you run a time-domain simulation, Simscape Electrical software uses the Simscape solver to analyze the physical system in the Simulink environment. For more information, see “How Simscape Simulation Works”.

## Running a Small-Signal Frequency-Domain Analysis

You can perform small-signal analysis for Simscape and Simscape Electrical models using linearization capabilities of Simulink software. For more information, see “Linearize an Electronic Circuit”.

## See Also

Solver Configuration

## More About

- “How Simscape Simulation Works”
- “Trimming and Linearization”
- “Troubleshooting”

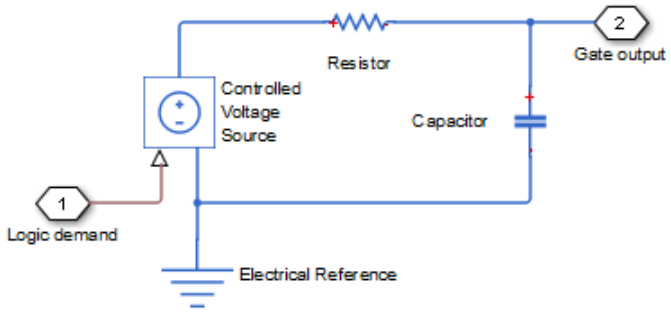
# Selecting the Output Model for Logic Blocks

| In this section... |
| --- |
| “Available Output Models” on page 3-10<br>“Quadratic Model Output and Parameters” on page 3-11 |

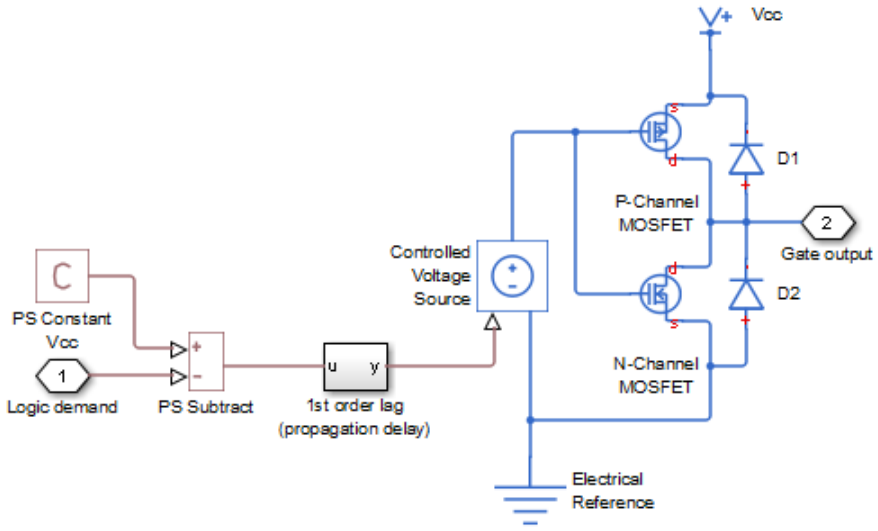
## Available Output Models

The blocks in the Logic sublibrary of the Integrated Circuits library provide a choice of two output models:

- `Linear` — Models the gate output as a voltage source driving a series resistor and capacitor connected to ground. This is suitable for logic circuit operation under normal conditions and when the logic gate drives other high-impedance CMOS gates. The block sets the value of the gate output capacitor such that the resistor-capacitor time constant equals the **Propagation delay** parameter value. The linear output model is shown in the following illustration.




- `Quadratic` — Models the gate output in terms of a complementary N-channel and P-channel MOSFET pair. This adds more fidelity, which becomes relevant if drawing higher currents from the gate output, or if exercising the gate under fault conditions. In addition, the gate input demand is lagged to approximate the **Propagation delay** parameter value. Default parameters are representative of the 74HC logic gate family. The quadratic output model is shown in the next illustration.




Use the **Output current-voltage relationship** parameter on the **Outputs** tab of the block dialog box to specify the output model.

For most system models, MathWorks recommends selecting the linear option because it supports faster simulation. If necessary, you can use the more detailed output model to validate simulation results obtained from the simpler model.

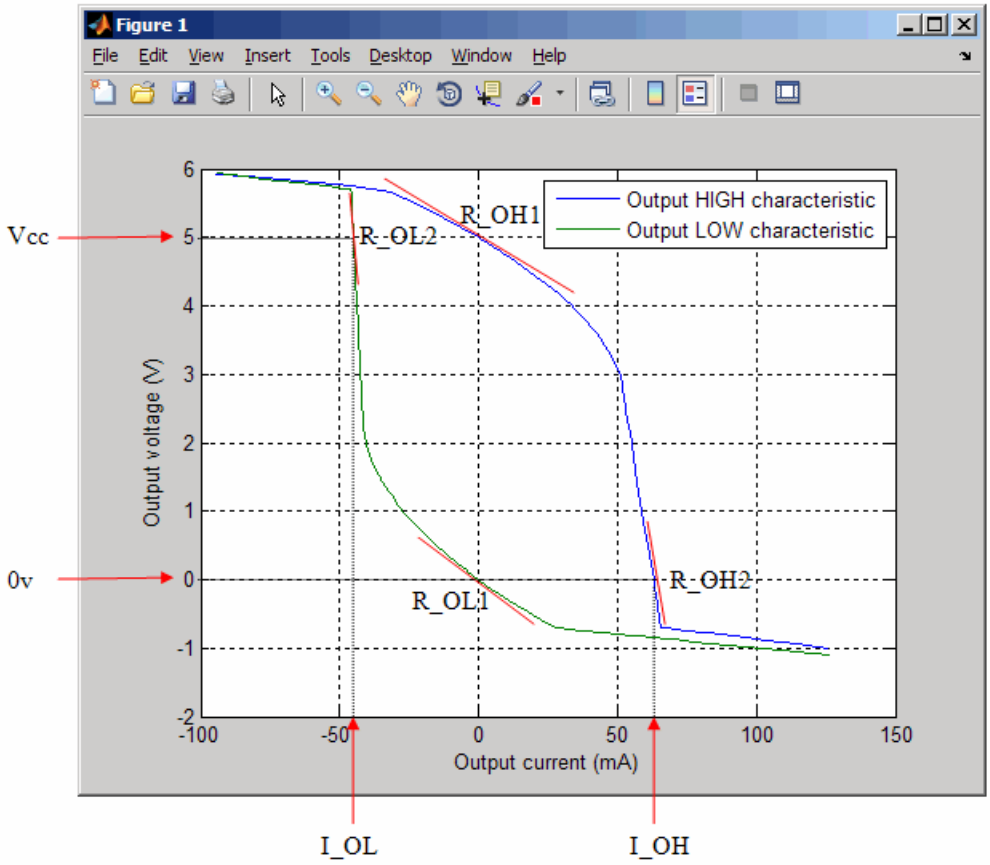
## Quadratic Model Output and Parameters

If you select the quadratic model, use the following parameters to control the block output:

- **Supply voltage** — Supply voltage value (Vcc) applied to the gate in your circuit. The default value is `5` V.
- **Measurement voltage** — The gate supply voltage for which mask data output resistances and currents are defined. The default value is `5` V.
- **Logic HIGH output resistance at zero current and at I_OH** — A row vector [ *R_OH1 R_OH2* ] of two resistance values. The first value *R_OH1* is the gradient of the output voltage-current relationship when the gate is logic HIGH and there is no output current. The second value *R_OH2* is the gradient of the output voltage-current relationship when the gate is logic HIGH and the output current is *I_OH*. The default value is `[ 25 250 ]` Ω.
- **Logic HIGH output current I_OH when shorted to ground** — The resulting current when the gate is in the logic HIGH state, but the load forces the output voltage to zero. The default value is `63` mA.
- **Logic LOW output resistance at zero current and at I_OL** — A row vector [ *R_OL1 R_OL2* ] of two resistance values. The first value *R_OL1* is the gradient of the output voltage-current relationship when the gate is logic LOW and there is no output current. The second value *R_OL2* is the gradient of the output voltage-current relationship when the gate is logic LOW and the output current is *I_OL*. The default value is `[ 30 800 ]` Ω.
- **Logic LOW output current I_OL when shorted to Vcc** — The resulting current when the gate is in the logic LOW state, but the load forces the output voltage to the supply voltage Vcc. The default value is `-45` mA.

- **Propagation delay** — Time it takes for the output to swing from `LOW` to `HIGH` or `HIGH` to `LOW` after the input logic levels change. For quadratic output, it is implemented by the lagged gate input demand. The default value is `25` ns.
- **Protection diode on resistance** — The gradient of the voltage-current relationship for the protection diodes when forward biased. The default value is `5` Ω.
- **Protection diode forward voltage** — The voltage above which the protection diode is turned on. The default value is `0.6` V.

The following graphic illustrates the quadratic output model parameterization, using the default parameter output characteristics for a +5V supply.



# Parameterizing Blocks from Datasheets

Simscape Electrical is a system-level simulation tool that provides blocks with a commensurate level of fidelity. Block parameters are designed, where possible, to match the data found on manufacturer datasheets. For example, the bipolar transistor blocks support parameterization in terms of the small-signal quantities quoted on a datasheet, and the underlying model is simpler than models typically used by specialist EDA simulation tools. The smaller number of parameters and simpler underlying models can support MATLAB system performance analysis better, and thus support design choices. Following system design, you can perform validation in hardware or more detailed modeling and validation using an EDA simulation tool.

The following parameterization examples illustrate various block parameterization techniques:

- Example 1: “Parameterize a Piecewise Linear Diode Model from a Datasheet” on page 3-14
- Example 2: “Parameterize an Exponential Diode from a Datasheet” on page 3-17
- Example 3: “Parameterize an Exponential Diode from SPICE Netlist” on page 3-21
- Example 4: “Parameterize an Op-Amp from a Datasheet” on page 3-25

Most of the time, datasheets should be a sufficient source of parameters for Simscape Electrical blocks (see Examples 1 on page 3-14, 2 on page 3-17, and 4 on page 3-25). Sometimes, there is need for more information than is available on the datasheet, and data can be augmented from a manufacturer SPICE netlist. For example, circuit performance may depend on one or two critical components, and increased accuracy is needed either for parameter values or the underlying model. Simscape Electrical libraries contain a SPICE-compatible sublibrary to support this case, as is illustrated by Example 3 on page 3-21. If you have many components that need to be modeled to a high level of accuracy, then Simulink cosimulation with a specialist circuit simulator may be a better option.

You can also use the SPICE conversion assistant to convert SPICE components into Simscape equivalents. For more information, see “Converting a SPICE Netlist to Simscape Blocks” on page 3-56

In mechatronic applications in particular, you may need to model input-output behavior of integrated circuits, such as PWM waveform generators and H-bridges. For these two examples, Simscape Electrical libraries contain abstracted-behavior equivalent blocks that you can use. Where you need to model other devices, possible options include creating your own abstracted model using the Simscape language, or using Simulink blocks. For an example of using Simulink blocks, see the “Modeling an Integrated Circuit” on page 8-638 example.

When looking for a datasheet, make sure that you have the originating manufacturer datasheet because some resellers abbreviate them.

For additional ways to parameterize and validate your model, see “Additional Parameterization Workflows” on page 3-27.

# Parameterize a Piecewise Linear Diode Model from a Datasheet

The “Triangle Wave Generator” on page 8-630 example model, also described in “Build and Simulate a Simple Circuit” on page 2-16, contains two zener diodes that regulate the maximum output voltage from an op-amp amplifier circuit. Each of these diodes is implemented with the Simscape Electrical Diode block, parameterized using the `Piecewise Linear` option. This simple model is sufficient to check correct operation of the circuit, and requires fewer parameters than the `Exponential` option of the Diode block. However, when specifying the parameters, you need to take into account the bias condition that will be used in the circuit. This example explains how to do this.

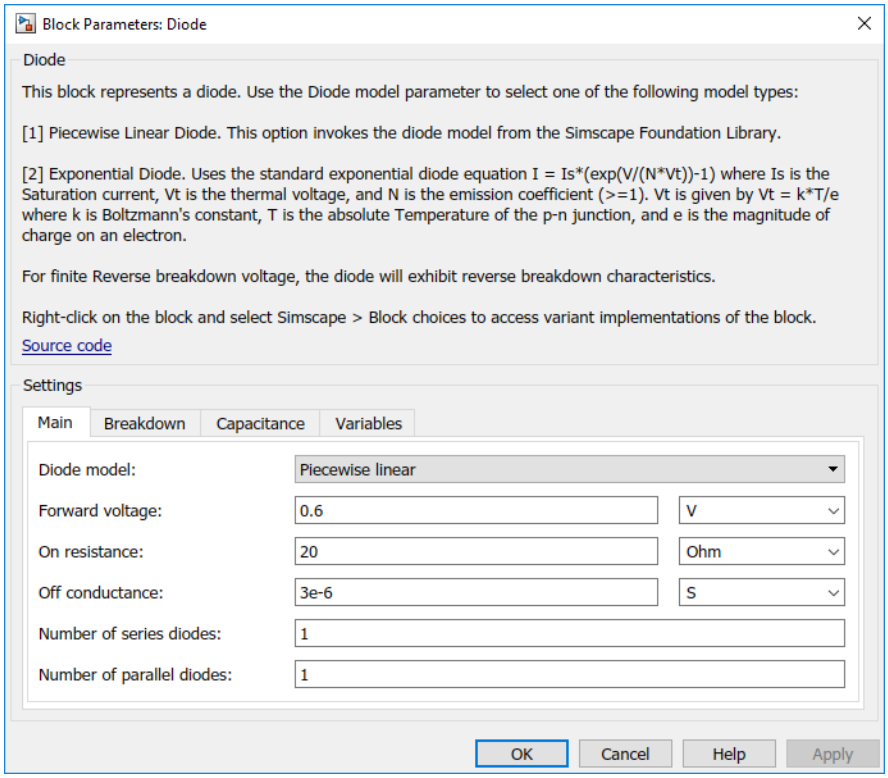
The Phillips Semiconductors datasheet for a BZX384–B4V3 gives the following data:

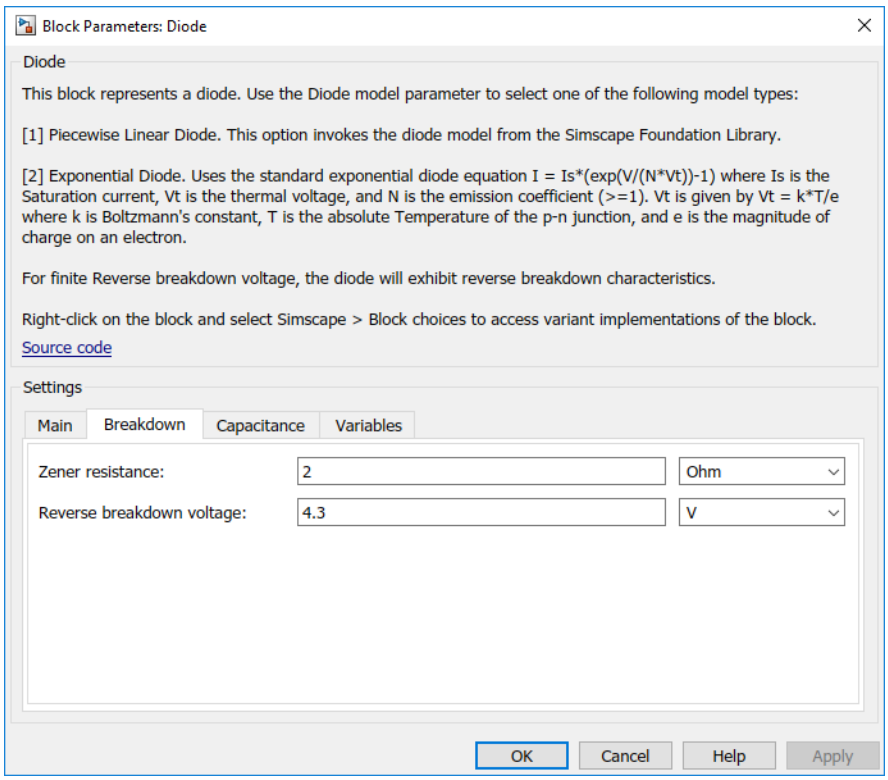
| | |
|---|---|
| Working voltage, $V_Z$(V) at $I_{Ztest}$ = 5 mA | 4.3 |
| Diode capacitance, $C_d$(pF) | 450 |
| Reverse current, $I_R$(µA) at $V_R$ = 1 V | 3 |
| Forward voltage, $V_F$(V) at $I_F$ = 5 mA | 0.7 |

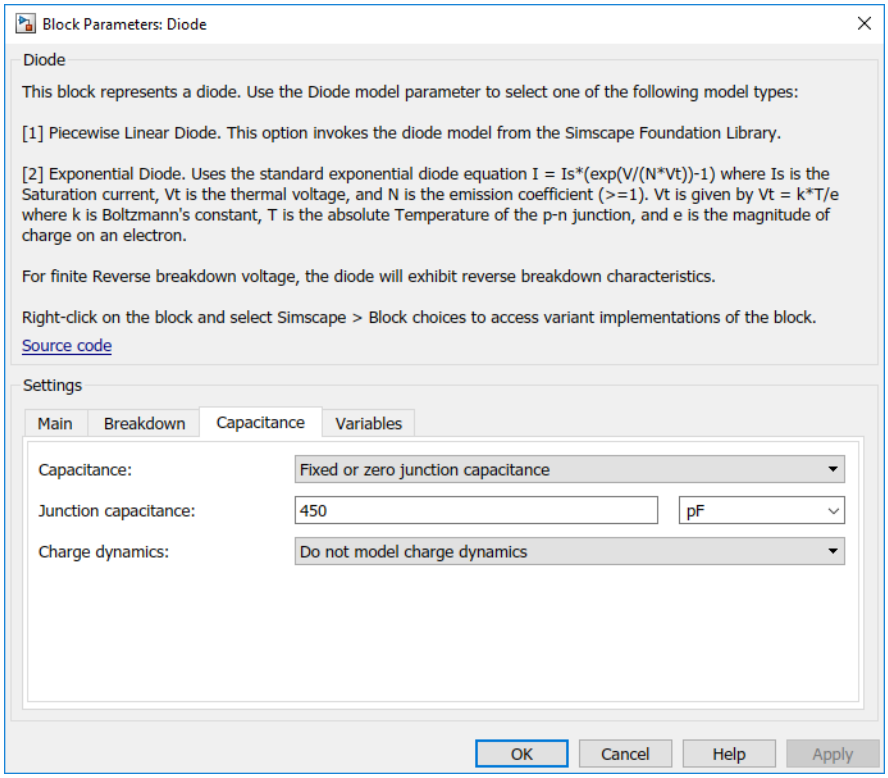
In the datasheet, the tabulated values for $V_F$ are for higher forward currents. This value of 0.7V at 5mA is extracted from the datasheet current-voltage curve, and is chosen as it matches the zener current used when quoting the working voltage of 4.3V.

To match the datasheet values, the example sets the piecewise linear diode block parameters as follows:

- **Forward voltage**. Leave as default value of 0.6V. This is a typical value for most diodes, and the exact value is not critical. However, it is important that the value set is taken into account when calculating the **On resistance** parameter.
- **On resistance**. This is set using the datasheet information that the forward voltage is 0.7V when the current is 5mA. The voltage to be dropped by the **On resistance** parameter is 0.7V minus the **Forward voltage** parameter, that is 0.1V. Hence the **On resistance** is 0.1V / 5mA = 20 Ω.
- **Off conductance**. This is set using the datasheet information on reverse current. The reverse current is 3µA for a reverse voltage of 1V. Hence the **Off conductance** should be set to 3µA / 1V = 3e-6 S.
- **Reverse breakdown voltage**. This parameter should be set to the datasheet working voltage parameter, 4.3V.
- **Zener resistance**. This needs to be set to a suitable small number. Too small, and the voltage-current relationship becomes very steep, and simulation convergence may not be as efficient. Too large, and the zener voltage will be incorrect. For the Diode block to be representative of the real device, the simulated reverse voltage should be close to 4.3V at 5mA (the reverse bias current provided by the circuit). Allowing a 0.01 V error on the zener voltage at 5mA, the zener resistance $R_Z$ will be 0.01V / 5mA = 2 Ω.
- **Junction capacitance**. This parameter is set to the datasheet diode capacitance value, 450 pF.


Block Parameters: Diode
Diode
This block represents a diode. Use the Diode model parameter to select one of the following model types:
[1] Piecewise Linear Diode. This option invokes the diode model from the Simscape Foundation Library.
[2] Exponential Diode. Uses the standard exponential diode equation I = Is*(exp(V/(N*Vt))-1) where Is is the Saturation current, Vt is the thermal voltage, and N is the emission coefficient (>=1). Vt is given by Vt = k*T/e where k is Boltzmann's constant, T is the absolute Temperature of the p-n junction, and e is the magnitude of charge on an electron.
For finite Reverse breakdown voltage, the diode will exhibit reverse breakdown characteristics.
Right-click on the block and select Simscape > Block choices to access variant implementations of the block.
Source code
Settings
Main
Breakdown
Capacitance
Variables
Diode model:
Piecewise linear
Forward voltage:
0.6
V
On resistance:
20
Ohm
Off conductance:
3e-6
S
Number of series diodes:
1
Number of parallel diodes:
1
OK
Cancel
Help
Apply


Block Parameters: Diode
Diode
This block represents a diode. Use the Diode model parameter to select one of the following model types:
[1] Piecewise Linear Diode. This option invokes the diode model from the Simscape Foundation Library.
[2] Exponential Diode. Uses the standard exponential diode equation I = Is*(exp(V/(N*Vt))-1) where Is is the Saturation current, Vt is the thermal voltage, and N is the emission coefficient (>=1). Vt is given by Vt = k*T/e where k is Boltzmann's constant, T is the absolute Temperature of the p-n junction, and e is the magnitude of charge on an electron.
For finite Reverse breakdown voltage, the diode will exhibit reverse breakdown characteristics.
Right-click on the block and select Simscape > Block choices to access variant implementations of the block.
Source code
Settings
Main
Breakdown
Capacitance
Variables
Zener resistance:
2
Ohm
Reverse breakdown voltage:
4.3
V
OK
Cancel
Help
Apply


Block Parameters: Diode
Diode
This block represents a diode. Use the Diode model parameter to select one of the following model types:
[1] Piecewise Linear Diode. This option invokes the diode model from the Simscape Foundation Library.
[2] Exponential Diode. Uses the standard exponential diode equation I = Is*(exp(V/(N*Vt))-1) where Is is the Saturation current, Vt is the thermal voltage, and N is the emission coefficient (>=1). Vt is given by Vt = k*T/e where k is Boltzmann's constant, T is the absolute Temperature of the p-n junction, and e is the magnitude of charge on an electron.
For finite Reverse breakdown voltage, the diode will exhibit reverse breakdown characteristics.
Right-click on the block and select Simscape > Block choices to access variant implementations of the block.
Source code
Settings
Main
Breakdown
Capacitance
Variables
Capacitance:
Fixed or zero junction capacitance
Junction capacitance:
450
pF
Charge dynamics:
Do not model charge dynamics
OK
Cancel
Help
Apply

# Parameterize an Exponential Diode from a Datasheet

Example 1 on page 3-14 uses a piecewise linear approximation to the diode's exponential current-voltage relationship. This results in more efficient simulation, but requires some thought to go into the setting of block parameter values. An alternative is to use a more complex model that is valid for a wider range of voltage and current values. This example uses the `Exponential` parameterization option of the Diode block.

This model either requires two data points from the diode current-voltage relationship, or values for the underlying equation coefficients, namely the saturation current IS and the emission coefficient N. The BZX384-B4V3 datasheet only provides values for the former case. Some datasheets do not give the necessary data for either case, and you must follow the processes in Example 1 on page 3-14 or Example 3 on page 3-21 instead.

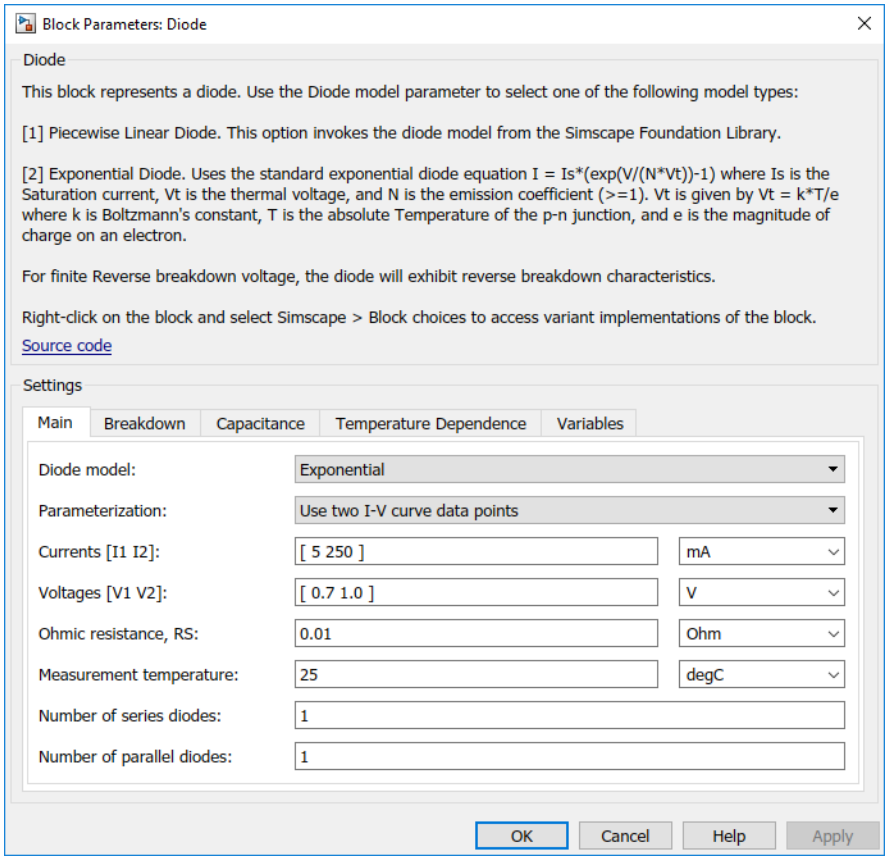
The two data points in the table below are from the BZX384-B4V3 datasheet current-voltage curve:

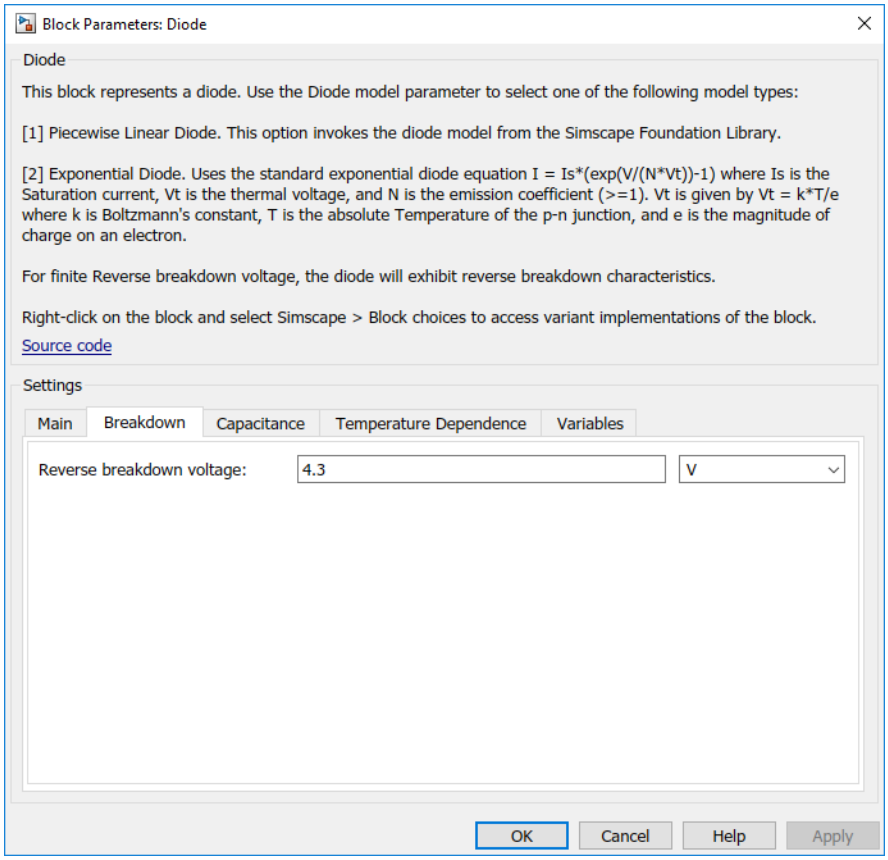
| | | |
|---|---|---|
| Diode forward voltage, $V_F$ | 0.7V | 1V |
| Diode forward current, $I_F$ | 5mA | 250mA |

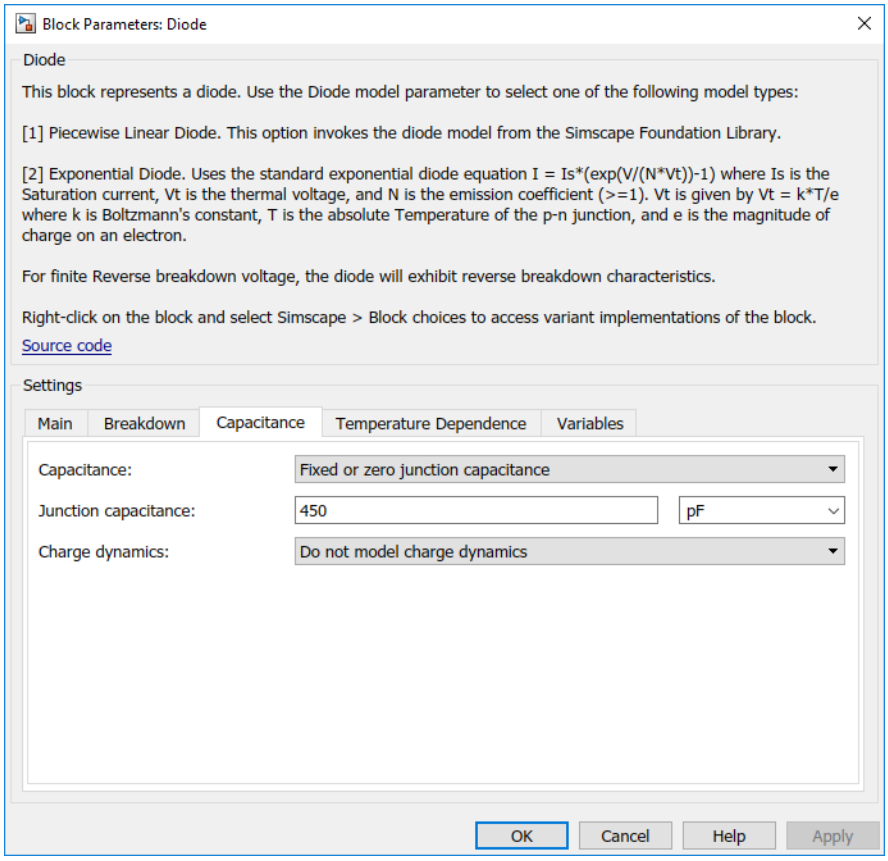
Set the exponential diode block parameters as follows:

- **Currents [I1 I2]**. Set to [5 250] mA.
- **Voltages [V1 V2]**. Set to [0.7 1.0] V.
- **Reverse breakdown voltage**. Set to the datasheet working voltage value, 4.3V.
- **Ohmic resistance, RS**. Set to 0.01 Ω. This is an example of a parameter that cannot be determined from the datasheet. However, setting its value to zero is not necessarily a good idea, because a small value can help simulation convergence for some circuit topologies. Physically, this term will not be zero because of the connection resistances.
- **Junction capacitance**. Set to the datasheet diode capacitance value, 450 pF.

  A more complex capacitance model is also available for the Diode component with the exponential equation option. However, the datasheet does not provide the necessary data. Moreover, the operation of this circuit is not sufficiently sensitive to voltage-dependent capacitance effects to warrant the extra detail.


Block Parameters: Diode
Diode
This block represents a diode. Use the Diode model parameter to select one of the following model types:
[1] Piecewise Linear Diode. This option invokes the diode model from the Simscape Foundation Library.
[2] Exponential Diode. Uses the standard exponential diode equation I = Is*(exp(V/(N*Vt))-1) where Is is the Saturation current, Vt is the thermal voltage, and N is the emission coefficient (>=1). Vt is given by Vt = k*T/e where k is Boltzmann's constant, T is the absolute Temperature of the p-n junction, and e is the magnitude of charge on an electron.
For finite Reverse breakdown voltage, the diode will exhibit reverse breakdown characteristics.
Right-click on the block and select Simscape > Block choices to access variant implementations of the block.
Source code
Settings
Main
Breakdown
Capacitance
Temperature Dependence
Variables
Diode model:
Exponential
Parameterization:
Use two I-V curve data points
Currents [I1 I2]:
[ 5 250 ]
mA
Voltages [V1 V2]:
[ 0.7 1.0 ]
V
Ohmic resistance, RS:
0.01
Ohm
Measurement temperature:
25
degC
Number of series diodes:
1
Number of parallel diodes:
1
OK
Cancel
Help
Apply


Block Parameters: Diode
Diode
This block represents a diode. Use the Diode model parameter to select one of the following model types:
[1] Piecewise Linear Diode. This option invokes the diode model from the Simscape Foundation Library.
[2] Exponential Diode. Uses the standard exponential diode equation I = Is*(exp(V/(N*Vt))-1) where Is is the Saturation current, Vt is the thermal voltage, and N is the emission coefficient (>=1). Vt is given by Vt = k*T/e where k is Boltzmann's constant, T is the absolute Temperature of the p-n junction, and e is the magnitude of charge on an electron.
For finite Reverse breakdown voltage, the diode will exhibit reverse breakdown characteristics.
Right-click on the block and select Simscape > Block choices to access variant implementations of the block.
Source code
Settings
Main
Breakdown
Capacitance
Temperature Dependence
Variables
Reverse breakdown voltage:
4.3
V
OK
Cancel
Help
Apply


Block Parameters: Diode
Diode
This block represents a diode. Use the Diode model parameter to select one of the following model types:
[1] Piecewise Linear Diode. This option invokes the diode model from the Simscape Foundation Library.
[2] Exponential Diode. Uses the standard exponential diode equation I = Is*(exp(V/(N*Vt))-1) where Is is the Saturation current, Vt is the thermal voltage, and N is the emission coefficient (>=1). Vt is given by Vt = k*T/e where k is Boltzmann's constant, T is the absolute Temperature of the p-n junction, and e is the magnitude of charge on an electron.
For finite Reverse breakdown voltage, the diode will exhibit reverse breakdown characteristics.
Right-click on the block and select Simscape > Block choices to access variant implementations of the block.
Source code
Settings
Main
Breakdown
Capacitance
Temperature Dependence
Variables
Capacitance:
Fixed or zero junction capacitance
Junction capacitance:
450
pF
Charge dynamics:
Do not model charge dynamics
OK
Cancel
Help
Apply

# Parameterize an Exponential Diode from SPICE Netlist

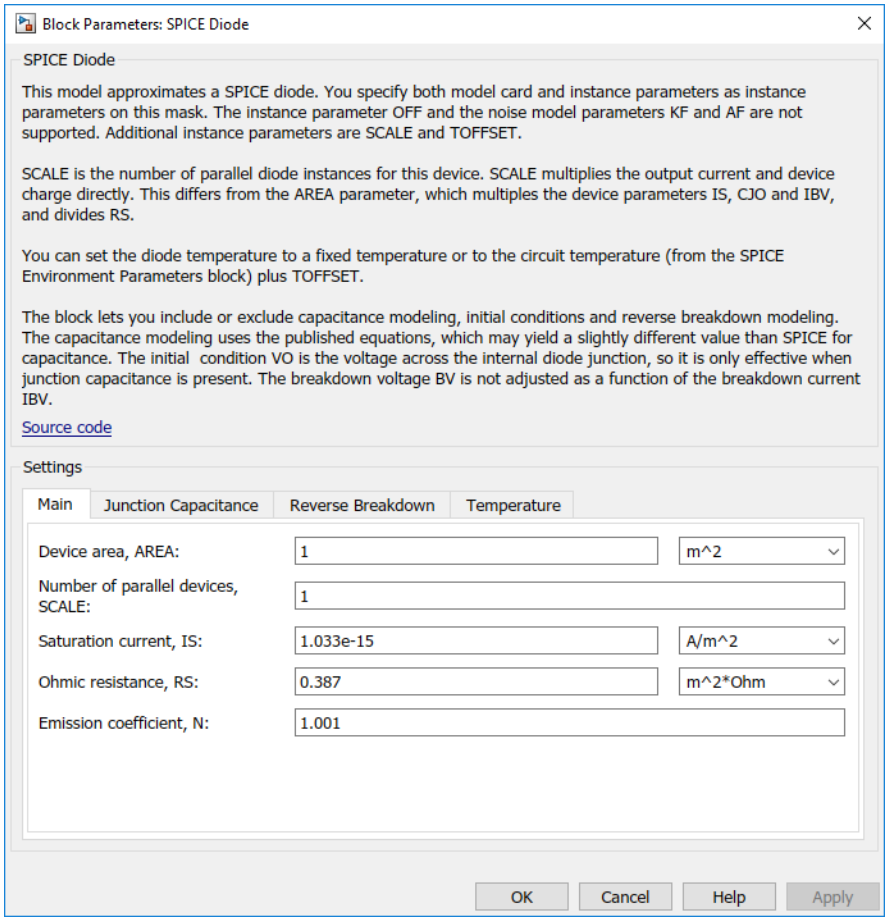
If a datasheet does not provide all of the data required by the component model, another source is a SPICE netlist for the component. Components are defined by a particular type of SPICE netlist called a subcircuit. The subcircuit defines the coefficients for the defining equations. Most component manufacturers make subcircuits available on their websites. The format is ASCII, and you can directly read off the parameters. The BZX384-B4V3 subcircuit can be obtained from Philips Semiconductors.

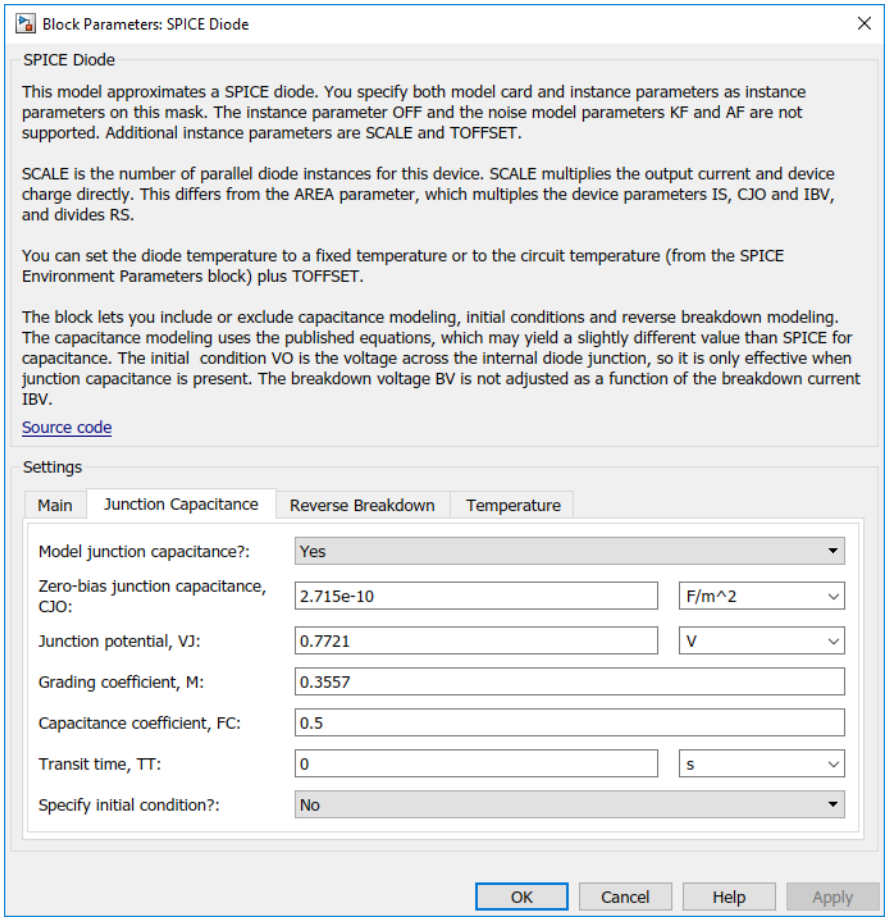
The subcircuit data can be used to parameterize the Simscape Electrical Diode block either in conjunction with the datasheet, or on its own. For example, the Ohmic resistance is defined in the subcircuit as RS = 0.387, thus providing the missing piece of information in Example 2 on page 3-17.

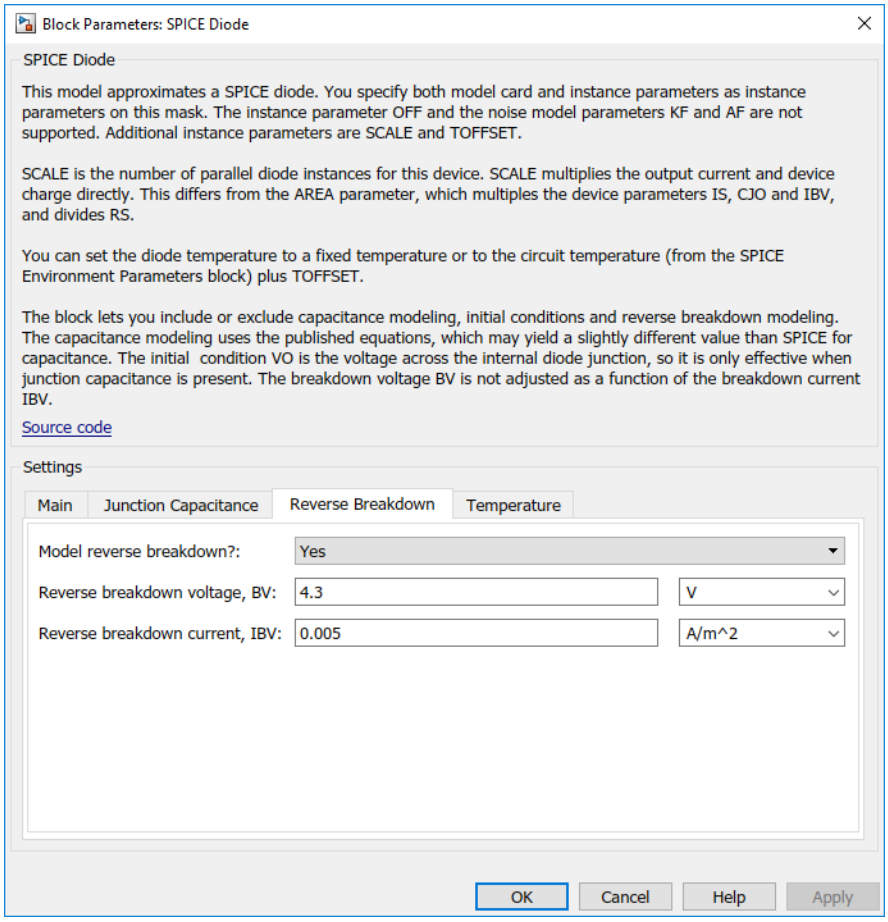
An alternative workflow is to use the Simscape Electrical Additional Components/SPICE Semiconductors sublibrary. The SPICE Diode block in this sublibrary can be directly parameterized from the subcircuit by setting:

- **Saturation current, IS** to `1.033e-15`
- **Ohmic resistance, RS** to `0.387`
- **Emission coefficient, N** to `1.001`
- **Zero-bias junction capacitance, CJO** to `2.715e-10`
- **Junction potential, VJ** to `0.7721`
- **Grading coefficient, M** to `0.3557`
- **Capacitance coefficient, FC** to `0.5`
- **Reverse breakdown current, IBV** to `0.005`
- **Reverse breakdown voltage, BV** to `4.3`

Note that where there is a one-to-one correspondence between subcircuit parameters and datasheet values, the numbers often differ. One reason for this is that datasheet values are sometimes given for maximum values, whereas subcircuit values are normally for nominal values. In this example, the CJO value of 271.5 pF differs from the datasheet capacitance of 450 pF at zero bias for this reason.


Block Parameters: SPICE Diode
SPICE Diode
This model approximates a SPICE diode. You specify both model card and instance parameters as instance parameters on this mask. The instance parameter OFF and the noise model parameters KF and AF are not supported. Additional instance parameters are SCALE and TOFFSET.
SCALE is the number of parallel diode instances for this device. SCALE multiplies the output current and device charge directly. This differs from the AREA parameter, which multiples the device parameters IS, CJO and IBV, and divides RS.
You can set the diode temperature to a fixed temperature or to the circuit temperature (from the SPICE Environment Parameters block) plus TOFFSET.
The block lets you include or exclude capacitance modeling, initial conditions and reverse breakdown modeling. The capacitance modeling uses the published equations, which may yield a slightly different value than SPICE for capacitance. The initial condition VO is the voltage across the internal diode junction, so it is only effective when junction capacitance is present. The breakdown voltage BV is not adjusted as a function of the breakdown current IBV.
Source code
Settings
Main
Junction Capacitance
Reverse Breakdown
Temperature
Device area, AREA:
1
m^2
Number of parallel devices, SCALE:
1
Saturation current, IS:
1.033e-15
A/m^2
Ohmic resistance, RS:
0.387
m^2*Ohm
Emission coefficient, N:
1.001
OK
Cancel
Help
Apply


Block Parameters: SPICE Diode
SPICE Diode
This model approximates a SPICE diode. You specify both model card and instance parameters as instance parameters on this mask. The instance parameter OFF and the noise model parameters KF and AF are not supported. Additional instance parameters are SCALE and TOFFSET.
SCALE is the number of parallel diode instances for this device. SCALE multiplies the output current and device charge directly. This differs from the AREA parameter, which multiples the device parameters IS, CJO and IBV, and divides RS.
You can set the diode temperature to a fixed temperature or to the circuit temperature (from the SPICE Environment Parameters block) plus TOFFSET.
The block lets you include or exclude capacitance modeling, initial conditions and reverse breakdown modeling. The capacitance modeling uses the published equations, which may yield a slightly different value than SPICE for capacitance. The initial condition VO is the voltage across the internal diode junction, so it is only effective when junction capacitance is present. The breakdown voltage BV is not adjusted as a function of the breakdown current IBV.
Source code
Settings
Main
Junction Capacitance
Reverse Breakdown
Temperature
Model junction capacitance?:
Yes
Zero-bias junction capacitance, CJO:
2.715e-10
F/m^2
Junction potential, VJ:
0.7721
V
Grading coefficient, M:
0.3557
Capacitance coefficient, FC:
0.5
Transit time, TT:
0
s
Specify initial condition?:
No
OK
Cancel
Help
Apply


Block Parameters: SPICE Diode
SPICE Diode
This model approximates a SPICE diode. You specify both model card and instance parameters as instance parameters on this mask. The instance parameter OFF and the noise model parameters KF and AF are not supported. Additional instance parameters are SCALE and TOFFSET.
SCALE is the number of parallel diode instances for this device. SCALE multiplies the output current and device charge directly. This differs from the AREA parameter, which multiples the device parameters IS, CJO and IBV, and divides RS.
You can set the diode temperature to a fixed temperature or to the circuit temperature (from the SPICE Environment Parameters block) plus TOFFSET.
The block lets you include or exclude capacitance modeling, initial conditions and reverse breakdown modeling. The capacitance modeling uses the published equations, which may yield a slightly different value than SPICE for capacitance. The initial condition VO is the voltage across the internal diode junction, so it is only effective when junction capacitance is present. The breakdown voltage BV is not adjusted as a function of the breakdown current IBV.
Source code
Settings
Main
Junction Capacitance
Reverse Breakdown
Temperature
Model reverse breakdown?:
Yes
Reverse breakdown voltage, BV:
4.3
V
Reverse breakdown current, IBV:
0.005
A/m^2
OK
Cancel
Help
Apply

# Parameterize an Op-Amp from a Datasheet

The “Triangle Wave Generator” on page 8-630 example model, also described in “Build and Simulate a Simple Circuit” on page 2-16, contains two op-amps, parameterized based on a datasheet for an LM7301. The National Semiconductor datasheet gives the following data for this device:

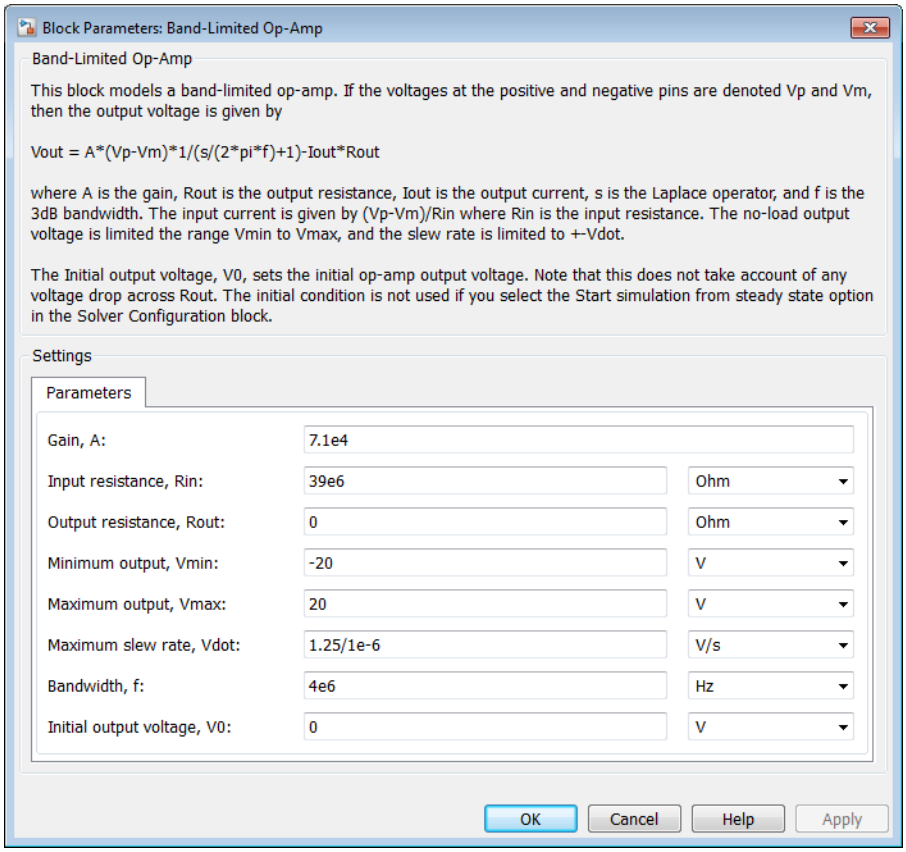
| | |
|---|---|
| Gain | 97dB = 7.1e4 |
| Input resistance | 39MΩ |
| Slew rate | 1.25V/µs |
| Bandwidth | 4MHz |

The Band-Limited Op-Amp and Finite-Gain Op-Amp blocks have been designed to work from manufacturer datasheets. Implementing detailed op-amp device models, derived from manufacturer SPICE netlist models, is not recommended, because it provides more accuracy than is typically warranted and slows down simulations. The simple parameterization of the Simscape Electrical op-amp blocks allows you to determine the sensitivity of your circuit to abstracted performance values, such as maximum slew rate and bandwidth. Because of this behavior-based parameterization, you can determine which specification of op-amp is required for a given application. A circuit designer can later match these behavioral parameters, determined from the model, against specific op-amp devices.

Based on the datasheet values above, set the Band-Limited Op-Amp block parameters as follows:

- **Gain** set to `7.1e4`
- **Input resistance, Rin** set to `39e6`Ω
- **Output resistance, Rout** set to zero. The value is not defined, but will be small compared to the 1000Ω load seen by the op-amp.
- **Minimum output, Vmin** set to the negative supply voltage, `-20`V in this model
- **Maximum output, Vmax** set to the positive supply voltage, `20`V in this model
- **Maximum slew rate, Vdot** set to `1.25/1e-6` V/s
- **Bandwidth, f** set to `4e6` Hz

Note that these parameters correspond to the values for +-5 volt operation. The datasheet also gives values for +-2.2V and +-30V operation. It is usually better to pick values for a supply voltage below what your circuit uses, because performance is worse at lower voltages; for example, the gain is less, and the input impedance is less. You can use the variation in op-amp parameters with supply voltage to suggest a typical range of parameter values for which you should check the operation of your circuit.


Block Parameters: Band-Limited Op-Amp
Band-Limited Op-Amp
This block models a band-limited op-amp. If the voltages at the positive and negative pins are denoted Vp and Vm, then the output voltage is given by
Vout = A*(Vp-Vm)*1/(s/(2*pi*f)+1)-Iout*Rout
where A is the gain, Rout is the output resistance, Iout is the output current, s is the Laplace operator, and f is the 3dB bandwidth. The input current is given by (Vp-Vm)/Rin where Rin is the input resistance. The no-load output voltage is limited the range Vmin to Vmax, and the slew rate is limited to +-Vdot.
The Initial output voltage, V0, sets the initial op-amp output voltage. Note that this does not take account of any voltage drop across Rout. The initial condition is not used if you select the Start simulation from steady state option in the Solver Configuration block.
Settings
Parameters
Gain, A:
7.1e4
Input resistance, Rin:
39e6
Ohm
Output resistance, Rout:
0
Ohm
Minimum output, Vmin:
-20
V
Maximum output, Vmax:
20
V
Maximum slew rate, Vdot:
1.25/1e-6
V/s
Bandwidth, f:
4e6
Hz
Initial output voltage, V0:
0
V
OK
Cancel
Help
Apply

# Additional Parameterization Workflows

| In this section... |
| --- |
| “Validation Using Data from SPICE Tool” on page 3-27 |
| “Parameter Tuning Against External Data” on page 3-27 |
| “Building an Equivalent Model of a SPICE Netlist” on page 3-27 |

## Validation Using Data from SPICE Tool

You can validate a parameterized Simscape Electrical component by comparing its behavior to the data from a specialist circuit simulation tool that uses a manufacturer SPICE netlist. Make sure to create a test harness for the component that validates the data across relevant operating points and frequencies.

## Parameter Tuning Against External Data

If you have lab measurements of the device, or data from another simulation environment, you can use this to tune the parameters of the equivalent Simscape Electrical component. For an example of parameter tuning, see “Solar Cell Parameter Extraction from Data” on page 8-644.

## Building an Equivalent Model of a SPICE Netlist

In “Parameterize an Exponential Diode from SPICE Netlist” on page 3-21, parameterization from a SPICE netlist is relatively straightforward because the netlist defines a single device (the diode) plus the corresponding model card (the parameters). Conversely, a netlist for an op-amp may have more than ten devices, plus supporting model cards. In principle, it is possible to build your own equivalent model of a more complex device by using the SPICE-compatible blocks in the **Simscape > Electrical > Additional Components**library. Connect the components together using the information in the netlist. Before embarking on this, make sure that the Additional Components sublibraries contain all the SPICE-compatible component models you need.

If the device models you wish to model are complex (hundreds of components), then cosimulation with an external circuit simulator may be a better approach.

## See Also

SPICE Diode

# Simulating Thermal Effects in Rotational and Translational Actuators

| In this section... |
| --- |
| “Using the Thermal Ports” on page 3-28<br>“Thermal Model for Actuator Blocks” on page 3-29 |

## Using the Thermal Ports

All blocks that represent rotational and translational actuators with electrical windings can optionally show a thermal port for each electrical winding. So, for example:

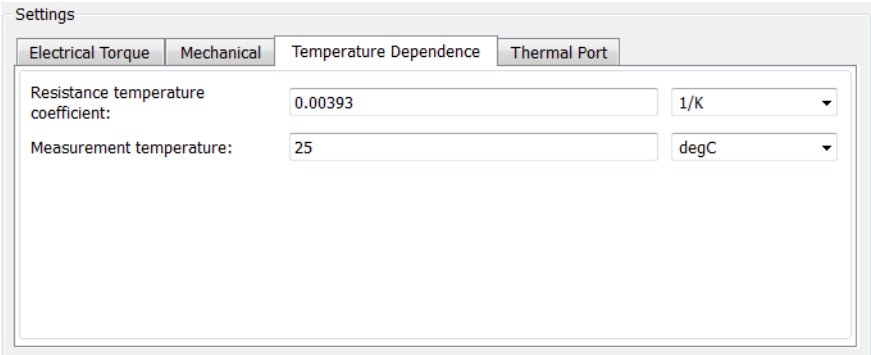
- A DC Motor block can optionally show a single thermal port corresponding to the armature
- A Shunt Motor block can optionally show two thermal ports, one for the stator winding and one for the field winding

The thermal port represents copper resistance losses which convert electrical power to heat. These losses are sometimes referred to as $i^2R$ losses. The thermal ports do not represent iron losses due to, for example, Eddy currents and hysteresis.

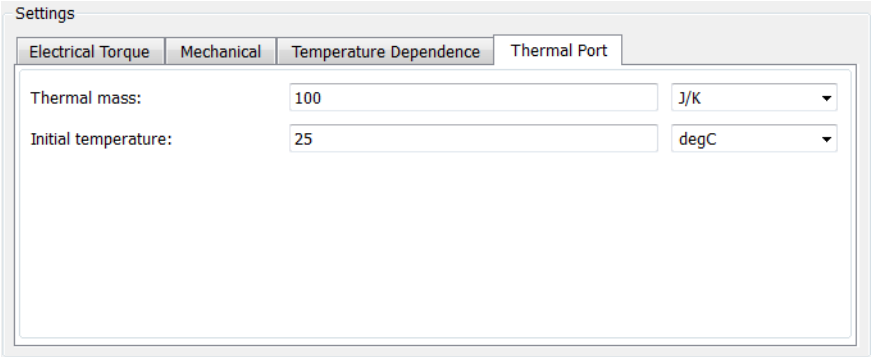
The thermal ports are hidden by default. To expose the thermal port on a particular block instance in your block diagram:

1. Double-click the block where you want to show the thermal port.
2. Set the **Modeling option** parameter to `Show thermal port`.

When the thermal port is exposed, the block dialog box contains two additional tabs, **Temperature Dependence** and **Thermal Port**. For actuator blocks with single winding, these tabs always contain the same set of parameters.




- **Resistance temperature coefficient** — Parameter α in the equation defining resistance as a function of temperature, as described in “Thermal Model for Actuator Blocks” on page 3-29. The default value is for copper, and is `0.00393` 1/K.
- **Measurement temperature** — The temperature for which motor parameters are defined. The default value is `25` °C.



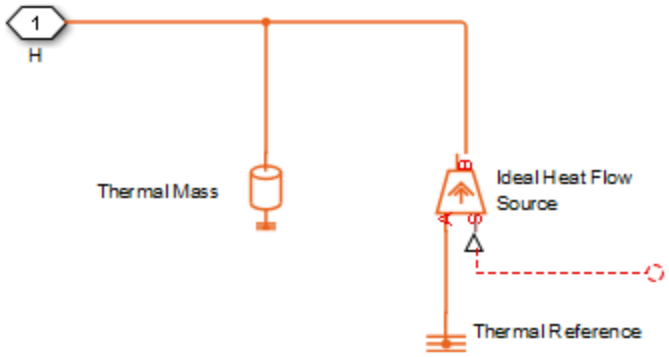

- **Thermal mass** — Thermal mass of the electrical winding, defined as the energy required to raise the temperature by one degree. The default value is `100` J/K.
- **Initial temperature** — The temperature of the thermal port at the start of simulation. The default value is 25 °C.

For more information on selecting the parameter values, see “Thermal Model for Actuator Blocks” on page 3-29.

Parameters for actuator blocks with two windings differ, and are described on the respective block reference pages.

## Thermal Model for Actuator Blocks

The following illustration shows the thermal port model used by the actuator blocks. The heat generated by the copper windings is provided as an input to the S physical signal input port of the Ideal Heat Flow Source. The thermal mass represents the lumped thermal mass of the copper winding where thermal mass is defined as the energy required to raise its temperature by one degree. If the mass is denoted $M$ and the specific heat capacity is $c_p$, then thermal mass is $M \cdot c_p$.




Winding resistance is assumed linearly dependent on temperature, and is given by:

$$R = R_0 \left(1 + \alpha \left(T - T_0\right)\right)$$

where:

- $R$ is the resistance at temperature $T$.

- $R_0$ is the resistance at the measurement (or reference) temperature $T_0$.
- α is the resistance temperature coefficient. A typical value for copper is 0.00393/K.

# Simulating Thermal Effects in Semiconductors

| In this section... |
| --- |
| “Using the Thermal Ports” on page 3-31 |
| “Specify junction and thermal case parameters” on page 3-32 |
| “Cauer Model” on page 3-33 |
| “Cauer Model Parameterized with Foster Coefficients” on page 3-34 |
| “External Model” on page 3-35 |
| “Thermal Mass Parameterization” on page 3-35 |
| “Electrical Behavior Depending on Temperature” on page 3-36 |
| “Improving Numerical Performance” on page 3-36 |
| “Model Thermal Losses for a Rectifier” on page 3-37 |

Thermal modeling provides data that helps you to estimate cooling requirements for your system by using the thermal ports. Some of the blocks in the Simscape Electrical Semiconductors & Converters library have thermal variants that allow you to determine device temperatures by simulating heat generation. For example, the IGBT (Ideal, Switching) block, which models a three-terminal semiconductor device, has thermal variants that can simulate the heat generated by switching events and conduction losses.

For more information on selecting the parameter values, see “Improving Numerical Performance” on page 3-36. For explanation of the relationship between the **Thermal Port** and **Temperature Dependence** settings in a block dialog box, see “Electrical Behavior Depending on Temperature” on page 3-36.

## Using the Thermal Ports

Certain Simscape Electrical blocks, such as the blocks in the Semiconductors & Converters library, contain an optional thermal port that is hidden by default. If you want to simulate the generated heat and device temperature, expose the thermal port by:

1. Double-click the block where you want to show the thermal port.
2. Set the **Modeling option** parameter to `Show thermal port`.

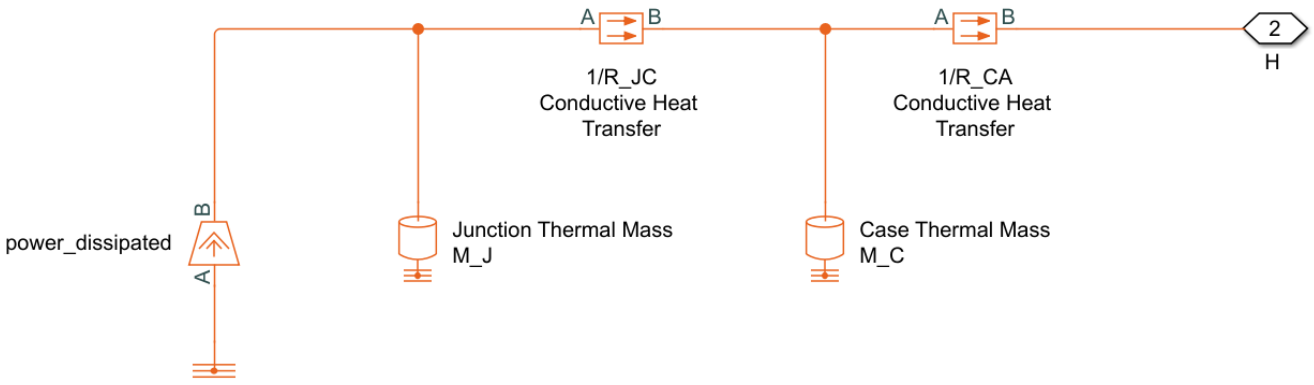
When the thermal port is exposed, the Block Parameters window for that block contains an additional setting, **Thermal Port**. Which parameters are visible depend on the value you set for the **Thermal network** parameter:

- “Specify junction and thermal case parameters” on page 3-32
- “Cauer Model” on page 3-33
- “Cauer Model Parameterized with Foster Coefficients” on page 3-34
- “External Model” on page 3-35

All blocks with optional thermal ports include an optional internal thermal model to keep your diagram uncluttered.

## Specify junction and thermal case parameters

This figure shows an equivalent model of the `Specify junction and thermal case parameters` model for semiconductor devices.




Port H corresponds to thermal port **H** of the block. The two Thermal Mass blocks represent the thermal mass of the device case and the thermal mass of the semiconductor junction, respectively. The Heat Flow Rate Source block (called power_dissipated in the diagram) inputs heat to the model with a value equal to the electrically generated heat from the device.

The two Conductive Heat Transfer blocks model the thermal resistances. Resistance *R_JC* (conductance 1/R_JC) represents the thermal resistance between junction and case. Because of this resistance, the junction will be hotter than the case under normal conditions. Resistance *R_CA* represents the thermal resistance between port **H** and the device case. If the device has no heat sink, then you should connect port **H** to a Temperature Source block with its temperature set to ambient conditions. If your device does have an external heat sink, then you must model the heat sink externally to the device and connect the heat sink thermal mass directly to port **H**.

If you choose to simulate the internal thermal network of the block through junction and thermal case parameters, these parameters are enabled:

- **Junction case and case-ambient (or case-heatsink) thermal resistances, [R_JC R_CA]** — A row vector [ *R_JC R_CA* ] of two thermal resistance values, represented by the two Conductive Heat Transfer blocks. The first value, *R_JC*, is the thermal resistance between the junction and the case. The second value, *R_CA*, is the thermal resistance between port **H** and the device case.
- **Thermal mass parameterization** — Select whether you want to parameterize the thermal masses in terms of thermal time constants (`By thermal time constants`), or specify the thermal mass values directly (`By thermal mass`). For more information, see “Thermal Mass Parameterization” on page 3-35. The default is `By thermal time constants`.
- **Junction and case thermal time constants, [t_J t_C]** — A row vector [ *t_J t_C* ] of two thermal time constant values. The first value, *t_J*, is the junction time constant. The second value, *t_C*, is the case time constant. To enable this parameter, set the **Thermal mass parameterization** to `By thermal time constants`.
- **Junction and case thermal masses, [M_J M_C]** — A row vector [ *M_J M_C* ] of two thermal mass values. The first value, *M_J*, is the junction thermal mass. The second value, *M_C*, is the case thermal mass. To enable this parameter, set the **Thermal mass parameterization** to `By thermal mass`.

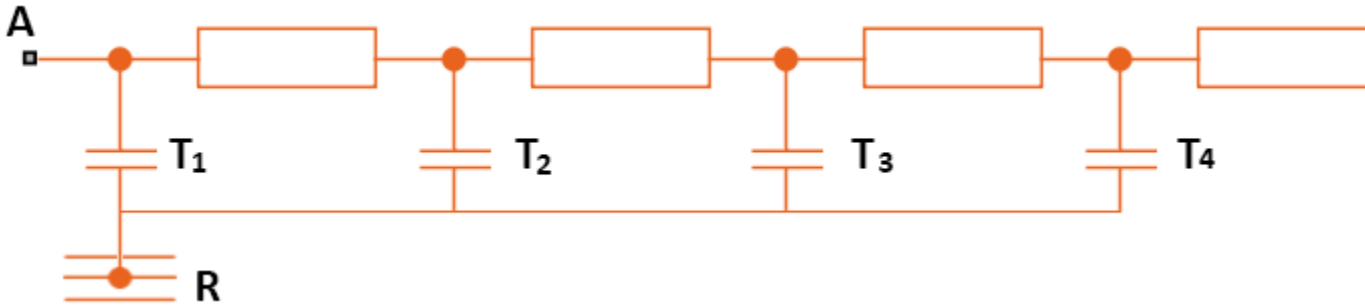
- **Junction and case initial temperatures, [T_J T_C]** — A row vector [ *T_J T_C* ] of two temperature values. The first value, *T_J*, is the junction initial temperature. The second value, *T_C*, is the case initial temperature.

The following rules apply:

- Case thermal mass must be greater than zero.
- Junction thermal mass can only be set to zero if the junction-case resistance is also set to zero.
- If both the case and junction thermal masses are defined, but the junction-case resistance is zero, then the initial temperatures assigned to the junction and case must be identical.

## Cauer Model

This figure shows an equivalent model of the internal Cauer thermal model for semiconductor devices.




If you choose to simulate the internal thermal network of the block through Cauer model, these parameters are enabled:

- **Thermal resistances, [R1 R2 ... Rn]** — A row vector of *n* thermal resistance values, represented by the Cauer elements used in the thermal network. These values must all be greater than zero.
- **Thermal mass parameterization** — Select whether you want to parameterize the thermal masses in terms of thermal time constants (`By thermal time constants`), or specify the thermal mass values directly (`By thermal mass`). For more information, see "Thermal Mass Parameterization" on page 3-35. The default is `By thermal time constants`.
- **Thermal time constants, [t1 t2 ... tn]** — A row vector of *n* thermal time constant values, where *n* is the number of Cauer elements used in the thermal network. The length of this vector must match the length of **Thermal resistances, [R1 R2 ... Rn]**. These values must all be greater than zero. With this parameterization, the thermal masses are computed as $M_i = t_i/R_i$, where $M_i$, $t_i$ and $R_i$ are the thermal mass, thermal time, and thermal resistance for the $i^{th}$ Cauer element. To enable this parameter, set **Thermal mass parameterization** to `By thermal time constants`.
- **Thermal masses, [M1 M2 ... Mn]** — A row vector of *n* thermal mass values, where n is the number of Cauer elements used in the thermal network. These values must all be greater than zero. To enable this parameter, set **Thermal mass parameterization** to `By thermal mass`.
- **Thermal masses initial temperatures, [T1 T2 ... Tn]** — A row vector of temperature values. It corresponds to the temperature drop across each thermal capacity in the model.

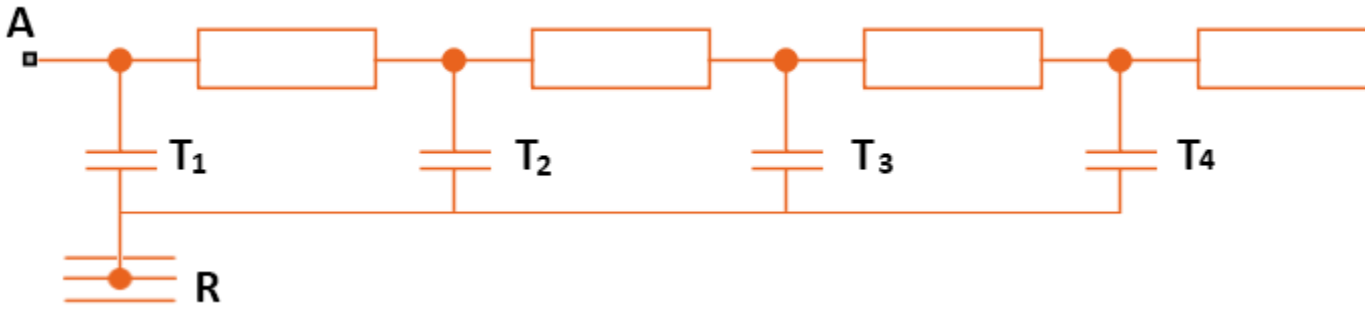
## Cauer Model Parameterized with Foster Coefficients

The datasheets of semiconductor devices often use Foster coefficients to specify the thermal model. However, Cauer thermal models are more useful as you can extend them with additional thermal components such as heatsinks, radiation, and convection elements.

If you want to model the internal thermal network of the semiconductor device with Foster coefficients, set the **Thermal network** parameter of the block to `Cauer model parameterized with Foster coefficients`.

**Note** This thermal network is implemented as a Cauer thermal network, but it behaves exactly as a Foster model implementation using the same parameters.

This figure shows the equivalent thermal model with a Cauer implementation, where each Cauer Thermal Model block has the **Parameterize Cauer with Foster coefficients** parameter selected.




This option allows you to connect a heatsink and other extensions to the thermal network while parameterizing the thermal model with Foster coefficients.

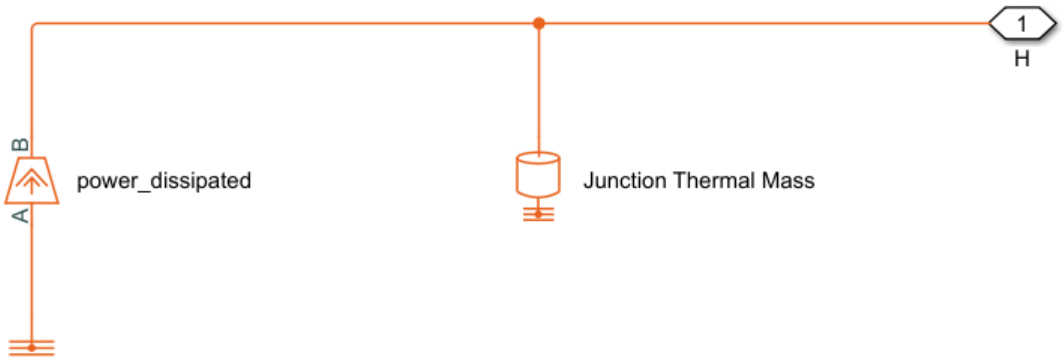
If you choose to simulate the internal thermal network of the block through a Cauer model parameterized with Foster coefficients, these parameters are enabled:

- **Thermal resistances, [R1 R2 ... Rn]** — A row of *n* thermal resistance values, represented by the Cauer elements used in the thermal network. These values must all be greater than zero.
- **Thermal mass parameterization** — Select whether you want to parameterize the thermal masses in terms of thermal time constants (`By thermal time constants`), or specify the thermal mass values directly (`By thermal mass`). For more information, see “Thermal Mass Parameterization” on page 3-35. The default is `By thermal time constants`.
- **Thermal time constants, [t1 t2 ... tn]** — A row vector of *n* thermal time constant values, where *n* is the number of Cauer elements used in the thermal network. The length of this vector must match the length of **Thermal resistances, [R1 R2 ... Rn]**. These values must all be greater than zero. With this parameterization, the thermal masses are computed as $M_i = t_i/R_i$, where $M_i$, $t_i$ and $R_i$ are the thermal mass, thermal time, and thermal resistance for the $i^{th}$ Foster element. To enable this parameter, set **Thermal mass parameterization** to `By thermal time constants`.
- **Thermal masses, [M1 M2 ... Mn]** — A row vector of *n* thermal mass values, where n is the number of Cauer elements used in the thermal network. These values must all be greater than zero. To enable this parameter, set **Thermal mass parameterization** to `By thermal mass`.

- **Initial node temperatures, [T1 T2 ... Tn]** — A row vector of absolute temperature values of each node starting from the junction.

## External Model

If you want to model the thermal network of a semiconductor block externally to the block itself, set the **Thermal network** parameter to `External`. This figure shows the equivalent model of the internal thermal model for semiconductor devices.




Port **H** corresponds to thermal port **H** of the block. The Heat Flow Rate Source block (called power_dissipated in the diagram) represents the total dissipated power in the block. The dissipated power is output as heat flow to the H node. Similar to the Foster thermal model, you need to connect a thermal source or additional thermal components to the H node so that the heat has somewhere to flow.

If you choose to simulate the internal thermal network of the block externally, the **Junction thermal mass** parameter is enabled.

## Thermal Mass Parameterization

If you need to estimate thermal masses, there are two parameterization options:

- `By thermal time constants` — Parameterize the thermal masses in terms of thermal time constants. This is the default.
- `By thermal mass` — Specify the thermal mass values directly.

For the `Cauer model (junction and case)`, the thermal time constants $t_J$ and $t_C$ are defined as follows:

$$t_J = M_J \cdot R_JC$$

$$t_C = M_C \cdot R_CA$$

where $M_J$ and $M_C$ are the junction and case thermal masses, respectively, $R_JC$ is the thermal resistance between junction and case, and $R_CA$ is the thermal resistance between port **H** and the device case.

For the `Foster model`, the thermal time constant, $t_i$, is defined as follows for the $i^{th}$ Foster element:

$$t_i = M_i \cdot R_i,$$

where $M_i$ and $R_i$ are the thermal mass and the thermal resistance of the $i^{th}$ Foster element, respectively.

You can determine the case-time constant by experimental measurement. If data is not available for the junction-time constant, you can either omit the constant and set the junction-case resistance to zero, or you can set the junction-time constant to a typical value of one-tenth of the case-time constant. Alternatively, you can estimate thermal masses based on the device dimensions and averaged material-specific heats.

## Electrical Behavior Depending on Temperature

For blocks with optional thermal ports, there are two simulation options:

- Simulate the generated heat, device temperature, and the effect of temperature on the electrical equations.
- Simulate the generated heat and device temperature, but do not include effect of temperature on the electrical equations. Use this option when the impact of temperature on the electrical equations is small for the temperature range that you are simulating, or where the primary task of the simulation is to capture the heat generated to support system-level design.

The thermal port and the **Thermal Port** setting of the Block Parameters window let you simulate the generated heat and device temperature. For blocks with a **Temperature Dependence** setting, it is possible to simulate the impact of the junction temperature on the electrical characteristics. The **Temperature Dependence** setting lets you model the effect that the temperature of the semiconductor junction has on the electrical equations. Therefore:

- To simulate all of the temperature effects, show the block's thermal port and, if the block has a **Temperature Dependence** setting, set the **Parameterization** parameter to one of the provided options, for example, `Use an I-V data point at second measurement temperature`.
- To simulate only the generated heat and device temperature, show the thermal port of the block and, in the **Temperature Dependence** settings, set **Parameterization** to `None — Simulate at parameter measurement temperature`.

## Improving Numerical Performance

Set realistic values for thermal masses and resistances. Otherwise, junction temperatures can become extreme, and out of range for valid results, which can manifest as numerical difficulties during simulation. You can test if numerical difficulties are a result of unrealistic thermal values by turning off the temperature dependence for the electrical equations. Open the Block Parameters window and, in the **Temperature Dependence** settings, set **Parameterization** to `None — Simulate at parameter measurement temperature`.

The thermal time constants are generally much slower than electrical time constants, so the thermal aspects of your model are unlikely to dictate the maximum fixed time step you can simulate at (for example, for hardware-in-the-loop simulations). However, if you need to remove detail (for example, to speed up simulation), the junction-thermal mass time constant is typically an order of magnitude faster than the case-thermal mass time constant. You can remove the effect of the junction-thermal mass by setting the junction-thermal mass and junction-case thermal resistance to zero.

## Model Thermal Losses for a Rectifier

### Model Heat Transfer for a Single Rectifier Diode

To model and measure heat transfer as a function of the thermal characteristics of a semiconductor, connect a Cauer-based thermal network parameterized with Foster coefficients, and a temperature sensor to a block with a thermal port.

1. Open the model. At the MATLAB command prompt, enter:

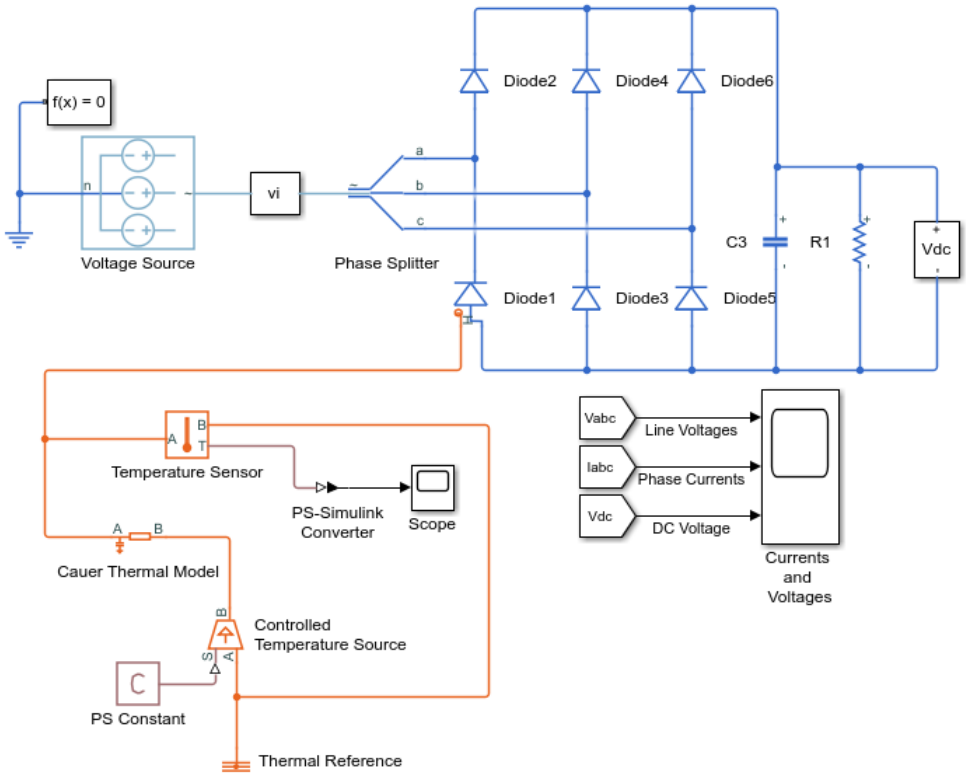
   ```
   ee_rectifier_diodes
   ```

   The model contains a three-phase rectifier that includes six Diode blocks.
2. Select a thermal modeling option for the Diode1 block by double-clicking the block and setting the **Modeling option** parameter to `Show thermal port`.
3. Open the Diode1 block.
   a. In the **Thermal Port** settings, set **Thermal network** to `External`.
   b. In the **Initial Targets** settings, select **Junction temperature**, and set **Priority** to `High` and **Value** to `300 K`.
4. Add a Simscape Electrical block that represents the heat flow between the diode and the environment. Open the Simulink Library browser, click **Simscape** > **Electrical** > **Passive** > **Thermal**, and add a Cauer Thermal Model block to the model.
5. Open the Cauer Thermal Model block and modify these parameters:
   a. In the **Parameters** settings:
      - Select the **Parameterize Cauer model using Foster coefficient data** parameter.
      - Set **Thermal resistance data (Foster)** to `[ 0.00311 0.008493 0.00252 0.00288 ] K/W`.
      - Set **Thermal time constant data (Foster)** to `[ 0.0068 0.0642 0.3209 2.0212 ] s`.
   b. In the **Initial Targets** settings:
      - Select **Vector of thermal mass temperatures**.
      - Set **Priority** to `High`.
6. Add these blocks to represent the ambient temperature as a constant by using an ideal temperature source.
   a. From the Simulink Library browser, open the **Simscape** > **Foundation Library** > **Thermal** > **Thermal Sources** library and add a Controlled Temperature Source block.
   b. From the **Simscape** > **Foundation Library** > **Thermal** > **Thermal Elements** library, add a Thermal Reference block.
   c. From the **Simscape** > **Foundation Library** > **Physical Signals** > **Sources** library, add a PS Constant block. For the **Constant** parameter, specify a value of `300` and select `K` for the unit.
7. Add these blocks to measure and display the temperature of Diode1:
   a. From the Simulink Library browser, open the **Simscape** > **Foundation Library** > **Thermal** > **Thermal Sensors** library and add a Temperature Sensor block.

b From the **Simscape > Utilities** library, add a PS-Simulink Converter block. For the **Output signal unit** parameter, select K.

c From the **Simulink > Sinks** library, and add a Scope block.

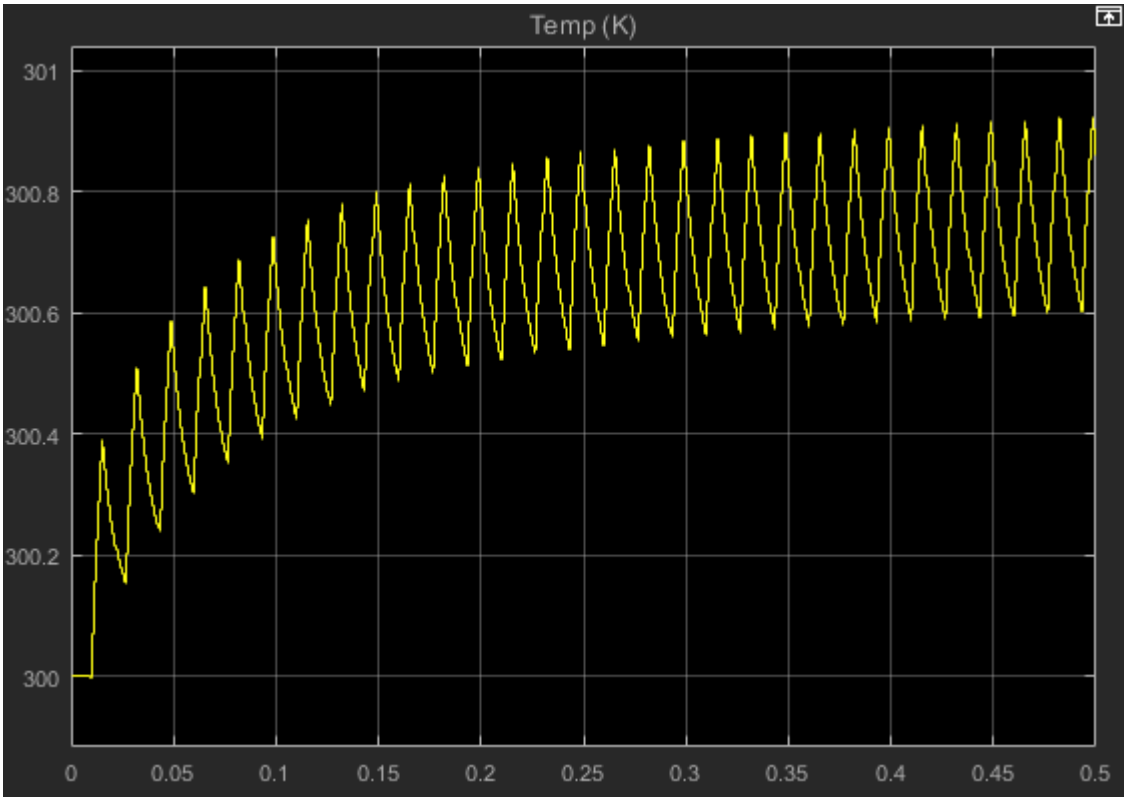
8 Arrange and connect the blocks as shown in the figure.




9 Label the signal from the PS-Simulink Converter block to the Scope block by double-clicking the line between the blocks and entering `Temp (K)`.

10 Simulate the model.

11 To see the temperature data, open the Scope block.



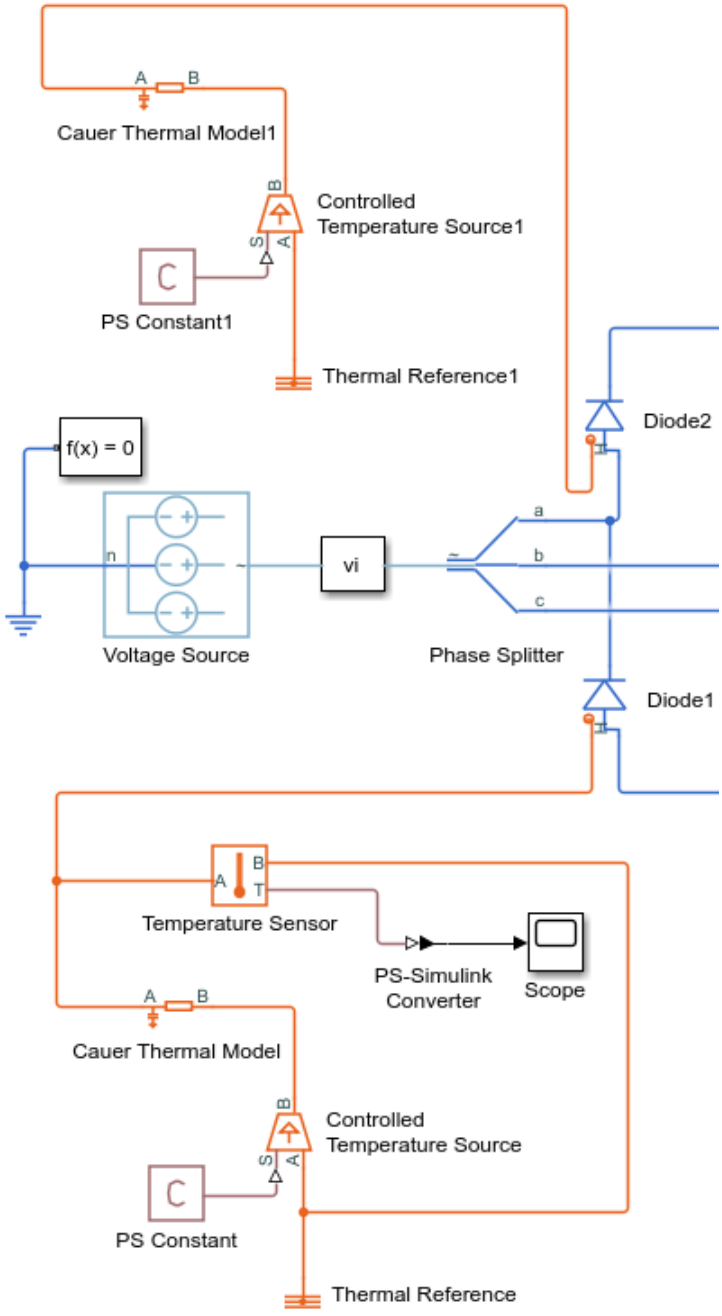

The temperature of Diode1 fluctuates over a temperature range of 0.3 K as it increases from the initial value of 300 K to a settling point of 300.6–300.9 K toward the end of the simulation.

### Model Heat Transfer for All Rectifier Diodes

To see the total heat generated by all of the semiconductors in the rectifier, use data logging and the Simscape Results Explorer.

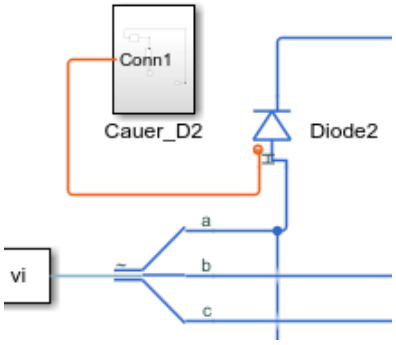
1. To enable the thermal ports on all the rectifier diodes, select thermal modeling options for the Diode2, Diode3, Diode4, Diode5, and Diode6 blocks by double-clicking the blocks and setting the **Modeling option** parameter to `Show thermal port`.
2. Open the Diode2, Diode3, Diode4, Diode5, and Diode6 blocks.
   a. In the **Thermal Port** settings, set **Thermal network** to `External`.
   b. In the **Initial Targets** settings, select **Junction temperature**, and set **Priority** to `High` and **Value** to `300` K.
3. Add blocks to measure the heat transfer for each diode by creating a Cauer thermal model subsystem.
   a. Make a copy of this group of blocks:
      - Cauer Thermal Model
      - Controlled Temperature Source
      - PS Constant
      - Thermal Reference

**b** Arrange and connect the copied blocks as shown in the figure.




**c** Create a subsystem from the copied blocks and rename the subsystem as Cauer_D2. For more information, see “Create Subsystems”.

**d** Open the Cauer_D2 subsystem. Open the Conn1 block, and for the **Port location on the parent subsystem** parameter, select `Right`.




e Make four copies of the Cauer_D2 subsystem. Attach one subsystem to each of the remaining Diode blocks and rename the subsystems as Cauer_D3 through Cauer_D6 to match the Diode3 through Diode6 block names.

4 Simulate the model.

5 View the results using the Simscape Results Explorer.

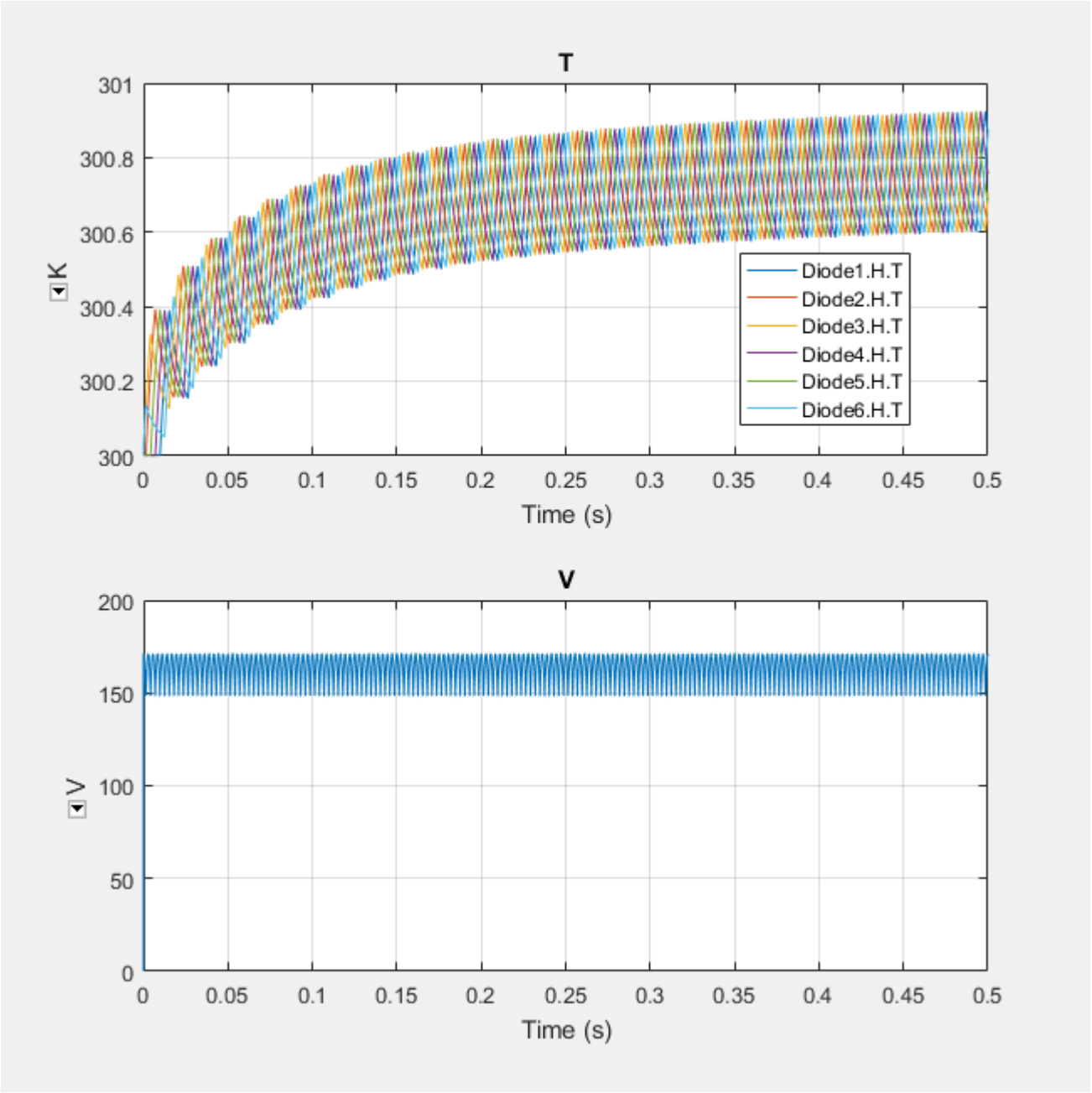
a In the model window, in the text under **Three-Phase Rectifier**, click **Explore simulation results**.

b To display the temperature data for **Diode1**, in the Simscape Results Explorer window, expand the **Diode1** > **H** node and click **T**.

c To display the DC voltage in a separate plot, expand the **Sensing Vdc** > **Voltage Sensor** node, press **CTRL**, and click **V**.

d To display the temperature data for all the diodes, expand the **Diode2** > **H** node, press **CTRL**, and click **T**. Repeat the process for Diode3 through Diode6.

e To overlay the temperature data in single plot, in the Simscape Results Explorer window, above the tree-node window, click the options button. In the Options dialog box, for **Plot signals**, select `Overlay`. To accept the change, click **OK**. Click and drag the legend down to see the temperature data clearly.




The temperature profile for each diode lags, in succession, behind the temperature profile of Diode1. For each diode, the temperature also rises and settles along the same values as the temperature profile for Diode1. The data indicate that, because of the lagging behavior of the individual diode temperatures, the temperature of the rectifier rises and settles along the same temperature profile as the diodes, but with less fluctuation.

## See Also

Cauer Thermal Model | Foster Thermal Model | Diode | `ee_getCauerFromFoster`

# Plot Basic Characteristics for Battery Blocks

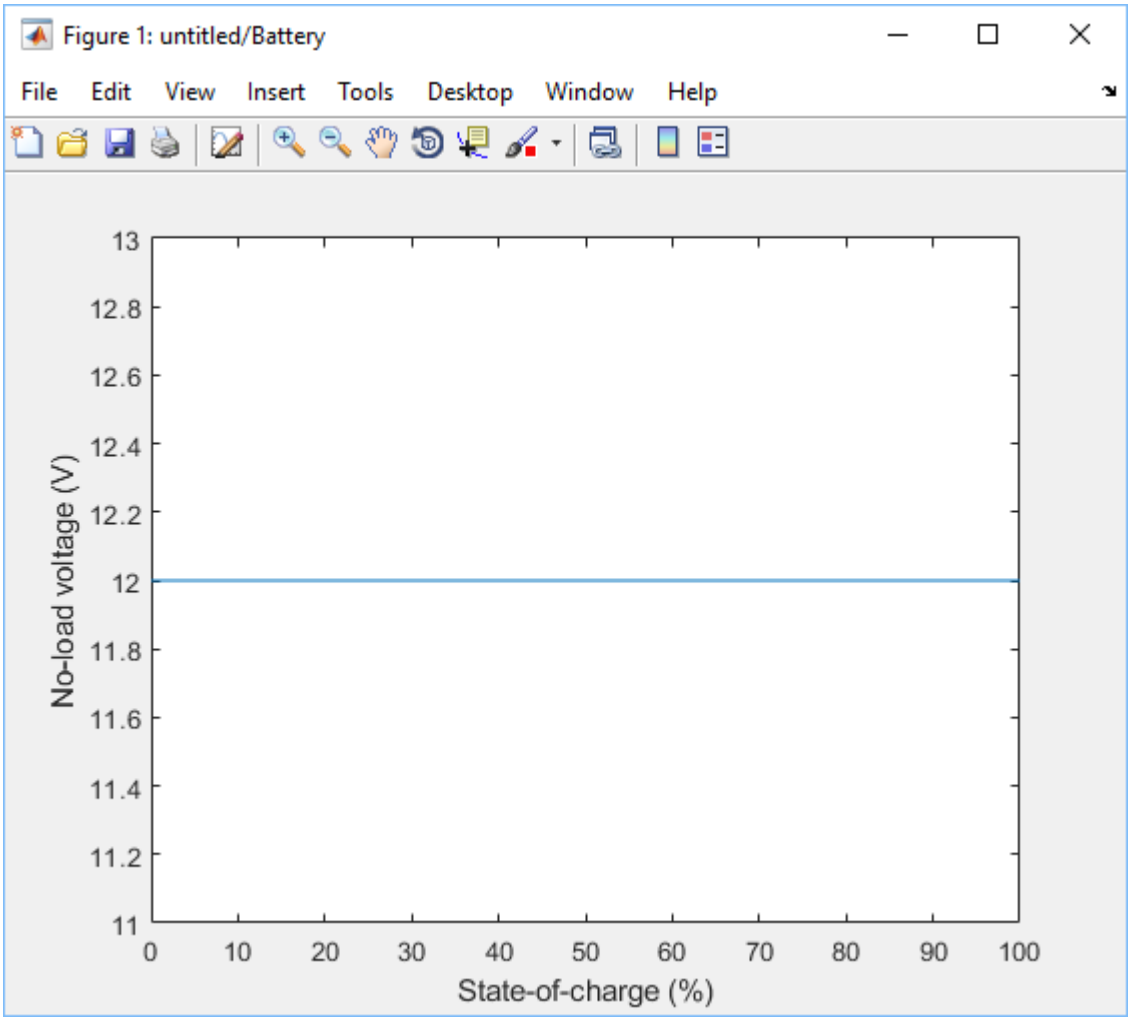
A quick plot feature lets you visualize the voltage-charge characteristic for battery blocks, based on the current block parameter values.

This feature is implemented for Battery and Battery (Table-Based) blocks.

To plot the battery voltage-charge characteristics:

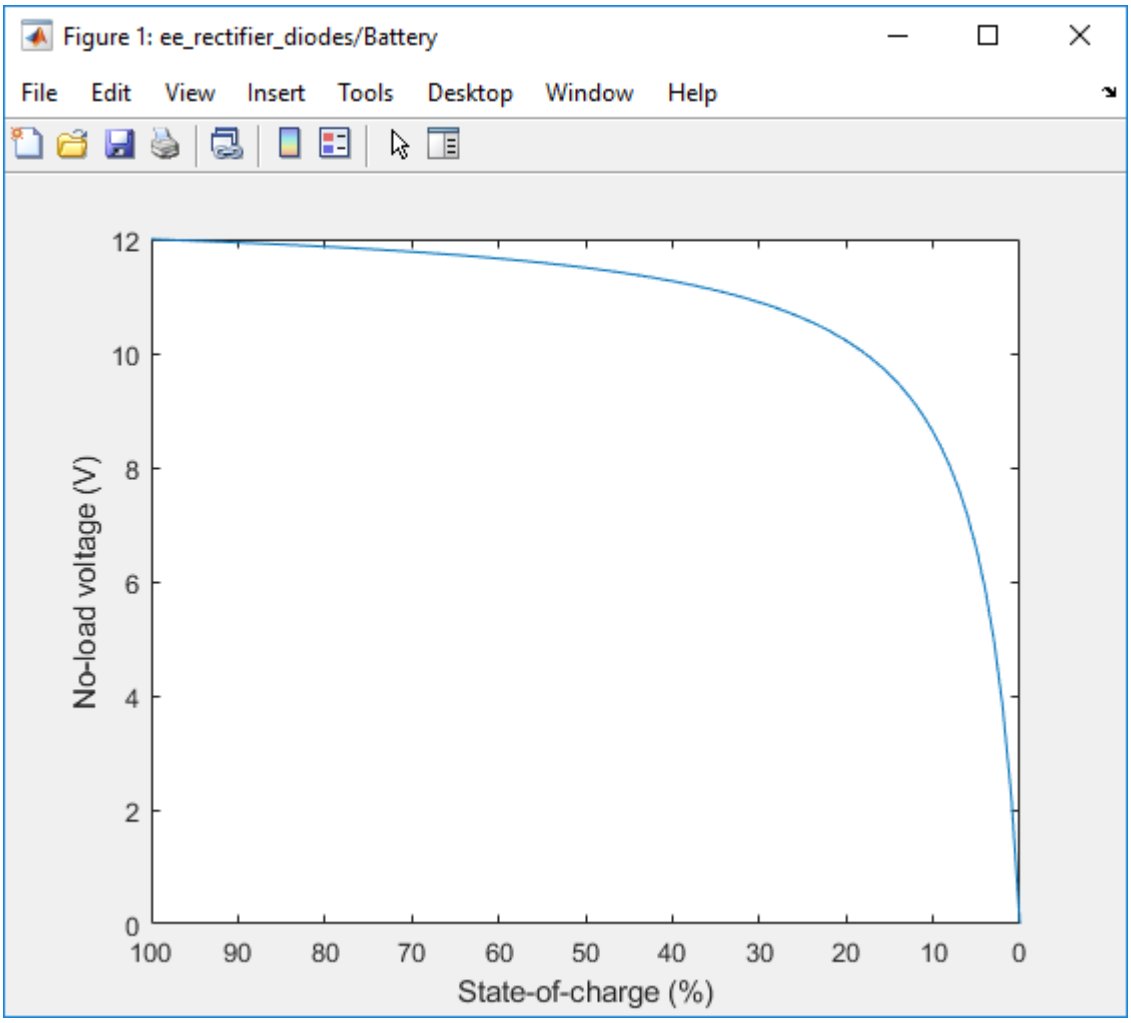
1. Right-click a battery block in your model and, from the context menu, select **Electrical > Basic characteristics**. The software automatically computes a set of bias conditions, based on the block parameter values, and opens a figure window containing a plot of no-load voltage versus the state-of-charge (SOC) for the block.

   For example, the following plot corresponds to the default parameter values of a Battery block with infinite charge.




2. If you change the block parameter values and plot the characteristics again, the plot opens in a new window. This way, you can compare the plots side-by-side and see how the parameter values affect the resulting voltage-charge characteristics for the block.

For example, if you change the **Battery charge capacity** parameter value to `Finite` and **Self-discharge** to `Enabled`, the new plot looks like this.




## See Also

Battery | Battery (Table-Based)

# Plot Basic Characteristics for Semiconductor Blocks

A quick plot feature lets you visualize the basic I-V characteristics for semiconductor switching devices, based on the current block parameter values.

This feature is implemented for nonthermal variants of the following blocks in the Semiconductors library:

- N-Channel IGBT
- N-Channel MOSFET (both threshold-based and surface-potential-based variants)
- P-Channel MOSFET (both threshold-based and surface-potential-based variants)
- N-Channel LDMOS FET
- P-Channel LDMOS FET
- N-Channel JFET
- P-Channel JFET
- NPN Bipolar Transistor
- PNP Bipolar Transistor

To plot the characteristics, right-click an appropriate semiconductor block in your model and, from the context menu, select **Electrical > Basic characteristics**.

---

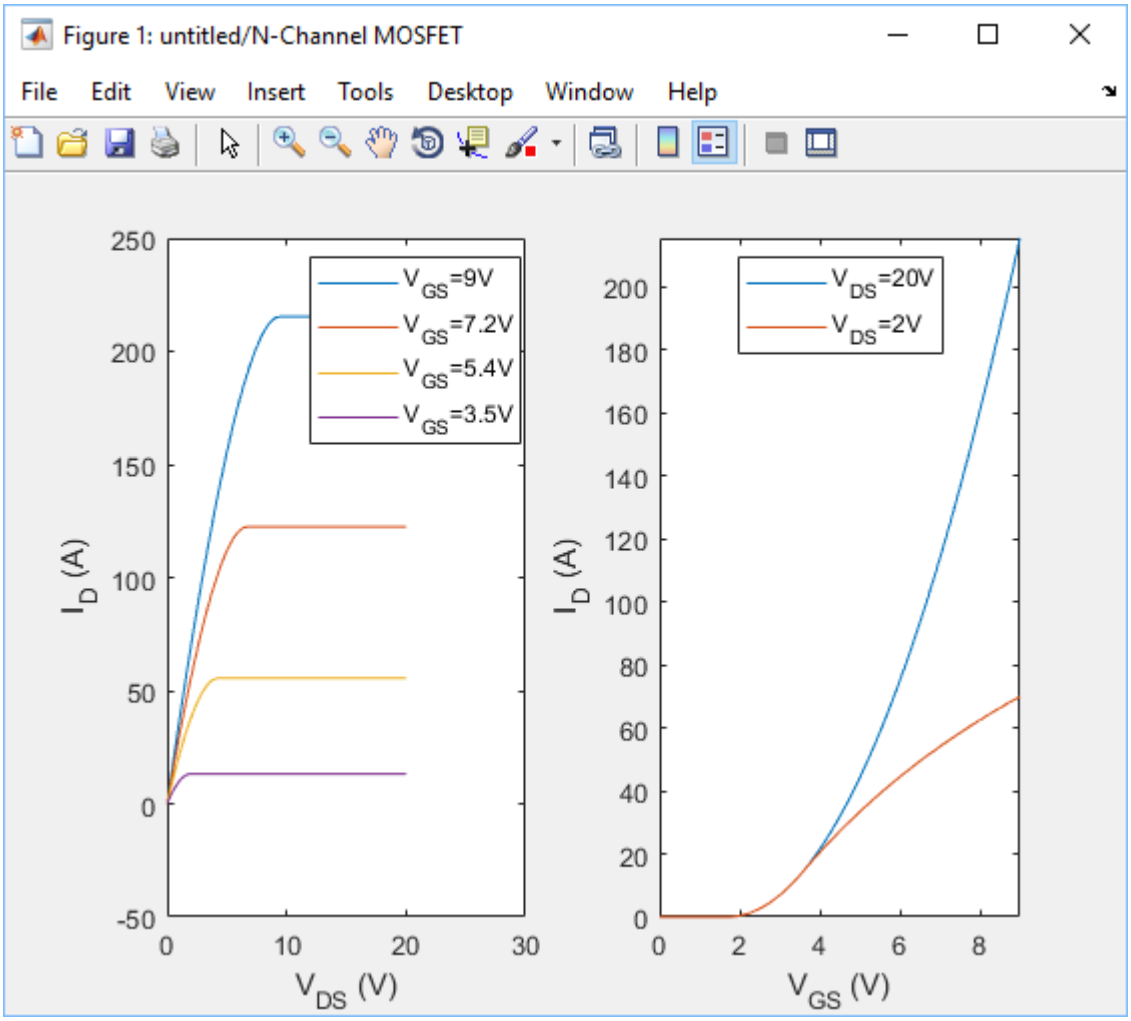
**Note** For surface-potential-based N-Channel MOSFET and P-Channel MOSFET blocks, the **Electrical > Explore characteristics** option is also available. This option opens the Characteristics Viewer tool, which lets you perform an in-depth study of block characteristics and match the block behavior to a set of target characteristics. For more information, see “MOSFET Characteristics Viewer” on page 3-48.

---

To plot the basic characteristics:

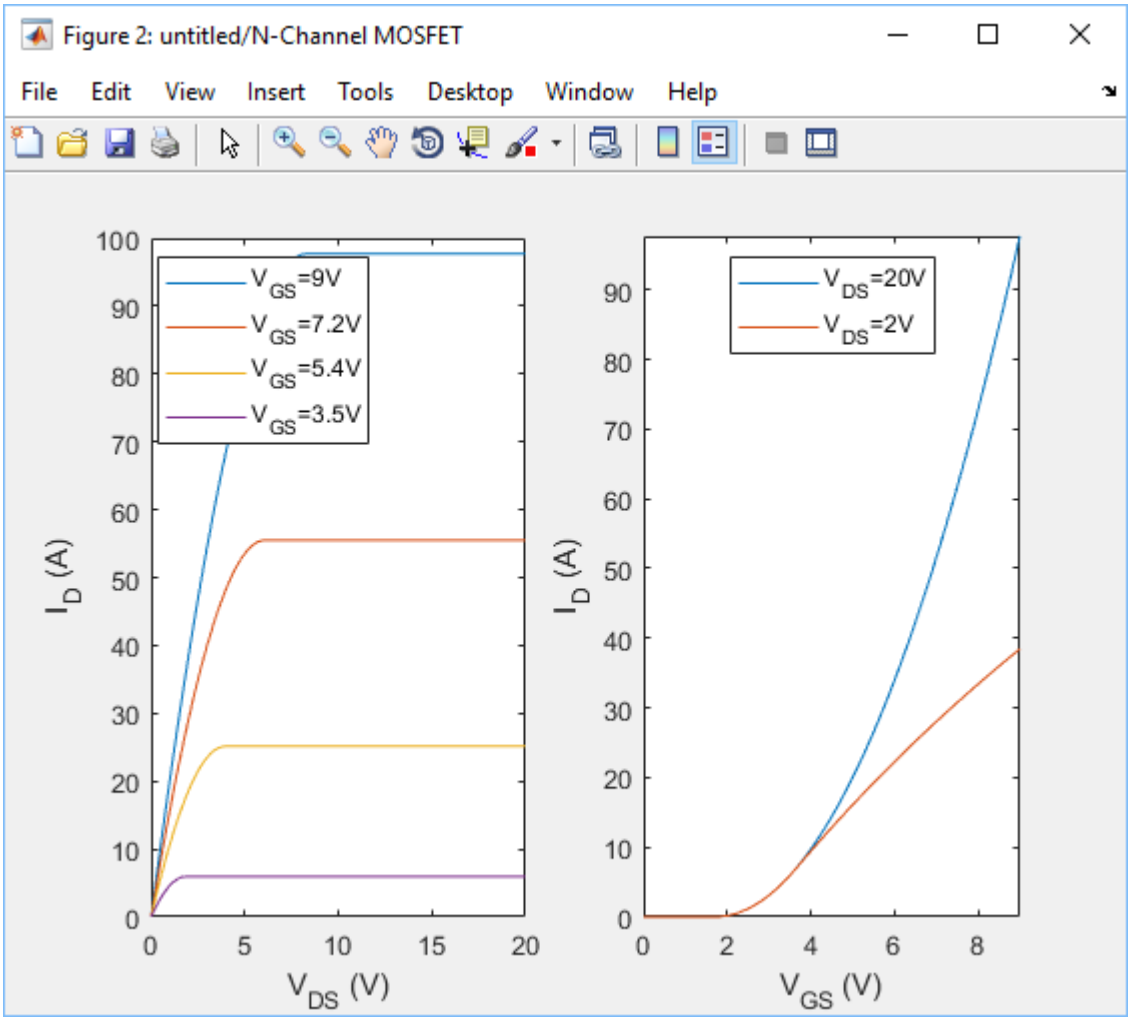
1. Right-click a semiconductor block in your model and, from the context menu, select **Electrical > Basic characteristics**. The software automatically computes a set of bias conditions, based on the block parameter values, and opens a figure window containing a plot of the DC I-V characteristics for the block.

   For example, the following plot corresponds to the default parameter values of a threshold-based N-Channel MOSFET block.




2 If you change the block parameter values and plot the characteristics again, the plot opens in a new window. This way, you can compare the plots side-by-side and see how the parameter values affect the resulting DC I-V characteristics for the block.

For example, if you change the **Gate-source voltage, Vgs, for R_DS(on)** parameter value to `20 V`, the new plot looks like this.




## See Also

## More About

- “MOSFET Characteristics Viewer” on page 3-48

# MOSFET Characteristics Viewer

| In this section... |
| --- |
| “Suggested Workflow” on page 3-48 |
| “Add and Manage Characteristics” on page 3-50 |
| “Choose Parameters and Generate Plots” on page 3-52 |
| “Save the Results” on page 3-54 |

The Characteristics Viewer tool lets you study characteristics of a particular parameterization of a surface-potential-based MOSFET block and match the block behavior to a set of target characteristics. The tool allows you to:

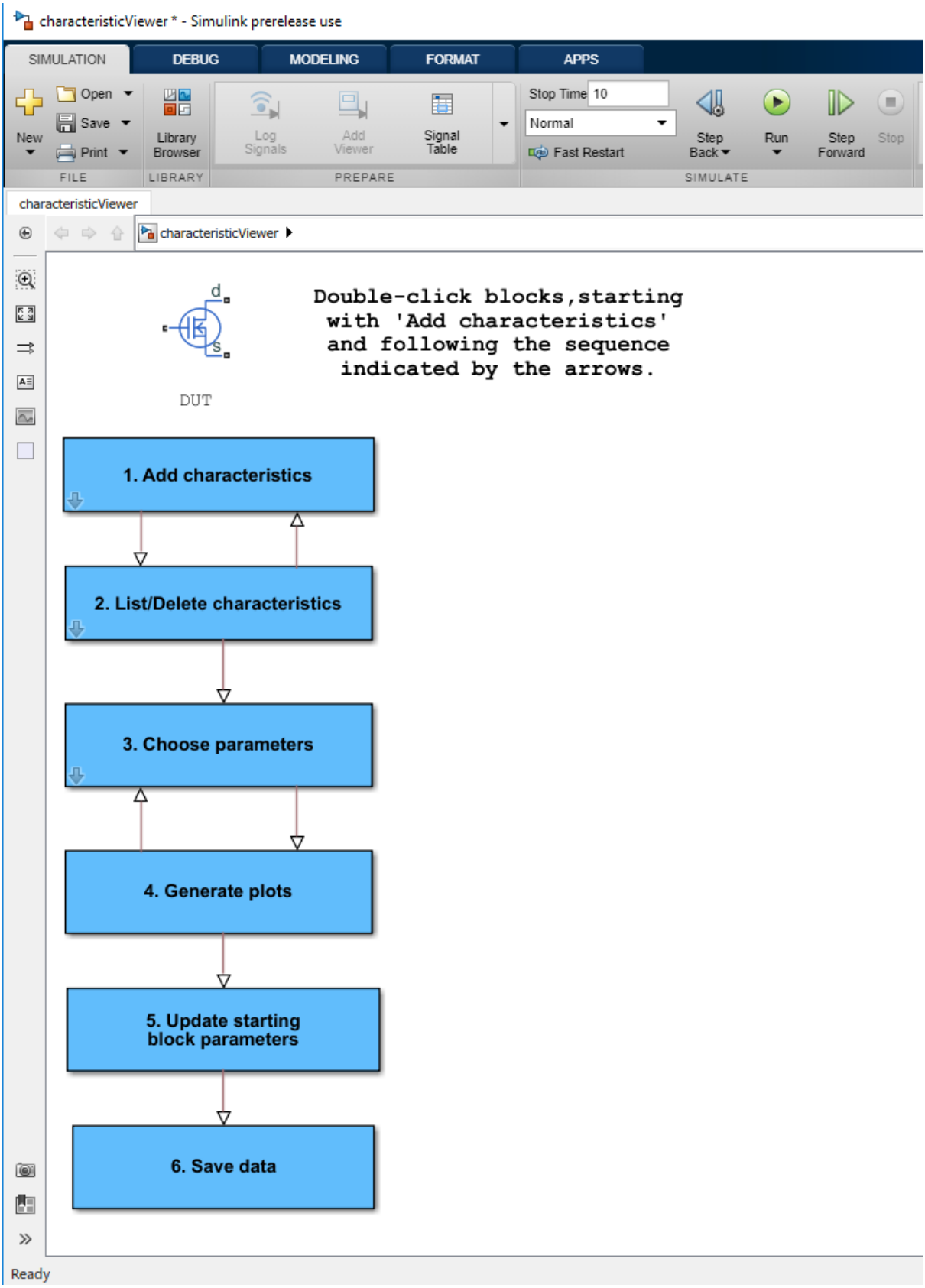
- Plot simulated data, using the current block parameters.
- Overlay simulated data plots over tabulated target data.
- Modify block parameters.
- When satisfied with the results of the parameters tuning in the Characteristics Viewer, update the block parameters in the model.
- Save generated parameter sets for future reuse in a different model.

## Suggested Workflow

The Characteristics Viewer tool is available for surface-potential-based N-Channel MOSFET or P-Channel MOSFET blocks only. To switch to a surface-potential-based modeling option when you add an N-Channel MOSFET or P-Channel MOSFET block from the library, double-click the block in your model and set the **Modeling option** parameter to `Surface-potential-based`. Then right-click the block and the context menu will contain the **Electrical** option, necessary to start the parameterization tool.

To use the MOSFET parameterization tool:

1. Right-click a surface-potential-based MOSFET block in your model and, from the context menu, select **Electrical** > **Explore characteristics**. A `characericViewer` window opens.


characteristicViewer * - Simulink prerelease use
SIMULATION
DEBUG
MODELING
FORMAT
APPS
New
Open
Save
Print
Library Browser
Log Signals
Add Viewer
Signal Table
Stop Time 10
Normal
Fast Restart
Step Back
Run
Step Forward
Stop
FILE
LIBRARY
PREPARE
SIMULATE
characteristicViewer
characteristicViewer
d
s
DUT
Double-click blocks,starting with 'Add characteristics' and following the sequence indicated by the arrows.
1. Add characteristics
2. List/Delete characteristics
3. Choose parameters
4. Generate plots
5. Update starting block parameters
6. Save data
Ready

2. Double-click **Add characteristics**. Specify the characteristics type (target, simulated, or both), and the desired values. Click **Add to plot**.

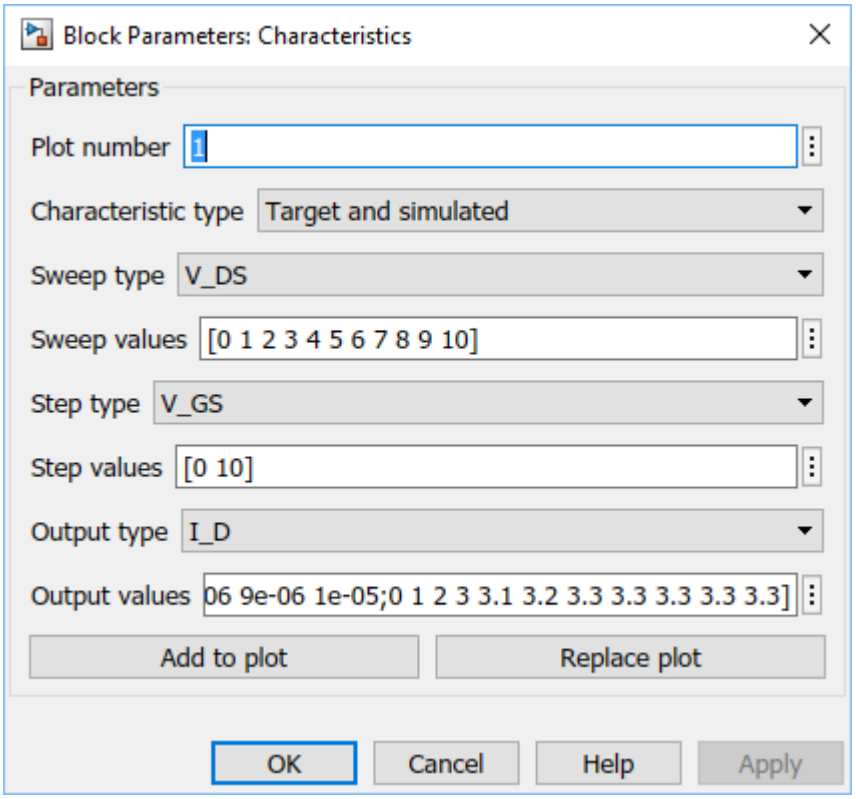
   Continue adding more characteristics, as needed. The **Replace plot** button lets you replace previously added plots. You can also use the **List/Delete characteristics** block, iteratively with **Add characteristics**, to configure your characteristics set.
3. Double-click **Choose parameters** and select the parameters of interest.
4. Double-click **Generate plots**.
5. Iterate between the previous two steps to tune the parameters by matching the simulation results to the target curves.
6. When satisfied with the results of the parameters tuning, double-click **Update starting block parameters** to update the block parameters in your model. Until you perform this step, the block in the original model is not affected.
7. You can double-click **Save data** to save the generated characteristics as a MAT-file, for future reuse in a different model.

## Add and Manage Characteristics

You start the MOSFET parameters tuning process by specifying the desired set of target characteristics:

1. In the `characticViewer` window, double-click **Add characteristics**.

   The Characteristics window opens.



2. Enter **Plot number**. This number defines the number of the figure that the characteristic will be plotted on. It allows you to add multiple characteristics to the same figure, for overlaying characteristics on top of each other. However, the figure will comprise one *xy*-axis only.
3. Specify the **Characteristic type**:
   - `Target only` — The plot will contain data that you specified, in terms of both input and output values. No simulation will be performed in this case. The data will simply be added to the appropriate plot.
   - `Simulated only` — The plot will contain data that is a result of a simulation over the input bias conditions that you specify.
   - `Target and simulated` — The plot will contain both types of data. This option is useful if you are trying to adjust parameters for the model to fit data that you have extracted from a datasheet.
4. Select **Sweep type**, which defines the *x*-axis variable for the resultant plot:
   - `V_GS` — Sweep over the gate-source voltage.
   - `V_DS` — Sweep over the drain-source voltage.
   - `I_D` — Sweep over the drain current. Normally, the drain current is not a typical input for a characteristic sweep.
5. If the **Characteristic type** is `Simulated only`, specify **Sweep range**. This is a vector of values indicating the range for the swept variable. Only the minimum and maximum values of this vector are utilized by the tool, since the exact sample points for the output data are determined by the variable-step simulation.
6. If the **Characteristic type** is `Target only` or `Target and simulated`, specify **Sweep values**. This is a vector of values for the swept variable at which the output is sampled for the target data. As an example, for an I_D-V_DS characteristic extracted from a datasheet, the vector would contain the V_DS values corresponding to the sampled I_D values in the target curve.
7. Select **Step type** to define the second independent input bias condition. The choices are the same as for **Sweep type**. For example, if an I_D-V_DS curve is defined as being at a constant V_GS, choose `V_GS` for **Step type**.
8. Use **Step values** to specify the values for the stepped variable. For example, if an I_D-V_DS curve is desired for V_GS values of 0 and 10V, set **Step type** to `V_GS` and **Step values** to `[0 10]`.
9. Select **Output type**, which defines the output measurement for the characteristic. This is the *y*-axis variable for the resultant plot. The available values are: `V_GS`, `V_DS`, `I_D`, `C_GG`, `C_GD`, `C_DG`, and `C_DD`. The capacitances C_GG, C_GD, C_DG, and C_DD are defined according to their terminals. To relate these quantities to the datasheet parameters of Ciss, Crss and Coss, note that C_GG = Ciss, C_DD = Coss, and C_GD = Crss at V_GS = 0.

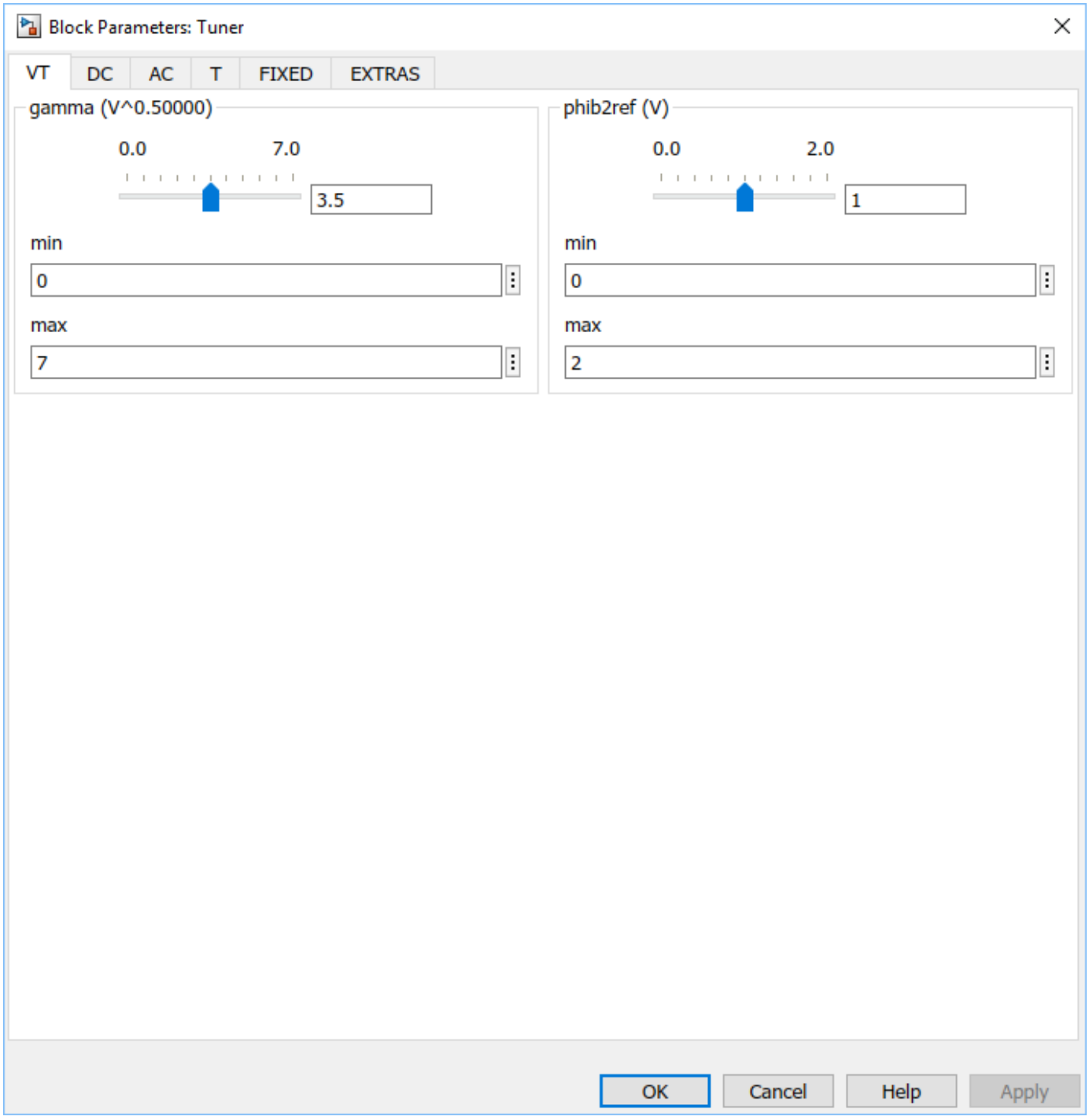
   `V_GS` is not a good choice as an output for the surface-potential-based MOSFET model. This value is provided in anticipation of using this tool for other device types.
10. If the **Characteristic type** is `Target only` or `Target and simulated`, specify **Output values**. This is the target data that you want to plot in the figures. Provide this data as an *m*-by-*n* matrix, where *m* is the size of **Step values** and *n* is the size of **Sweep values**.
11. Click **Add to plot** to add the characteristic specification to the appropriate **Plot number**.
12. Continue adding more characteristics, as needed.

The **Replace plot** button lets you replace previously added plots. You can also use the **List/Delete characteristics** block, iteratively with **Add characteristics**, to configure your characteristics set.

## Choose Parameters and Generate Plots

After you have specified the desired set of target characteristics, the next step is to define the parameters for the MOSFET block:

1 In the `characericViewer` window, double-click **Choose parameters**.




The Tuner window opens. It contains a series of sliders on different tabs, according to which feature of the MOSFET characteristics is most impacted by the specific parameter:

- The **VT** tab displays parameters that primarily impact the threshold voltage (**gamma** and **phib2ref**).
- The parameters on the **DC** tab primarily affect the DC characteristics.

- The parameters on the **AC** tab primarily affect the MOSFET dynamics.
- The parameters on the **T** tab affect temperature scaling.
- The parameters on the **FIXED** tab are generally fixed at some particular value that is not easy to derive from the displayed characteristics, such as the simulation temperature and the gate resistance (which is often indicated directly on datasheets).
- The **EXTRAS** tab contains other parameters, which impact the characteristics in ways similar to parameters that already appear on other tabs. For example, **Rsref** (the series resistance associated with the source) operates similarly to **betaref** from the **DC** tab. As a result, it is not always possible to disentangle these two effects.

2. Use the sliders on the appropriate tabs of the Tuner dialog.

   You can modify the `min` and `max` values, as needed, because they simply define the range over which the various sliders work. These values have no meaning for the underlying model parameters. Changing a `min` or `max` value automatically updates the slider range, without needing to click **OK** or **Apply**.

3. After adjusting the sliders, generate the plots to see how close the simulation data is to the target data. In the `characericViewer` window, double-click **Generate plots**.
4. Iterate between tuning the parameters and generating plots until the simulation results match the target curves.

## Save the Results

Once you are satisfied with the results of the parameters tuning:

- Double-click **Update starting block parameters** to update the block parameters in your model. Until you perform this step, the block in the original model is not affected.

  **Note** For this step to work, the original model must stay open while you are tuning the parameters.

- You can also double-click **Save data** to save the generated characteristics as a MAT-file, for future reuse in a different model. Specify the file name for saving the data. Inside the file, all the data is saved in an object named `parameterHelper`.

  To apply the parameters stored in this object to another MOSFET block, select the MOSFET block in a model and, in the MATLAB Command Window, type:

  ```
  parameterHelper.parameters.updateBlockParameters(gcbh)
  ```

  This command applies the parameter values to the block defined by the handle `gcbh`.

  You can also use a string instead of the block handle, for example:

  ```
  parameterHelper.parameters.updateBlockParameters(gcb)
  ```

  To inspect the parameters directly, type `parameterHelper.parameters.values` for the values (stored as character vectors) or `parameterHelper.parameters.names` for the names.

## See Also

## More About

- “Interactive Generation of MOSFET Characteristics” on page 8-574
- “Plot Basic Characteristics for Semiconductor Blocks” on page 3-45

# Converting a SPICE Netlist to Simscape Blocks

**In this section...**

“Commands” on page 3-56
“Numeric Suffixes” on page 3-57
“Mathematical Functions” on page 3-57
“Symbols” on page 3-61
“Components” on page 3-61
“Performing Manual Conversions” on page 3-64
“Limitations” on page 3-64

You can convert SPICE components into Simscape equivalents using the SPICE conversion assistant. Often this conversion is automatic. However, because SPICE is a rich language, it is not always possible to perform a full conversion without some manual intervention.

To convert SPICE subcircuits into equivalent Simscape components, follow these steps.

1. Use the `subcircuit2ssc` function to generate Simscape language component files from a SPICE netlist file. You can use the optional `subcircuit1,…,subcircuitN` input arguments to specify which subcircuits to convert.
2. Make any necessary manual conversions to the generated Simscape component files. To identify the required manual conversions, check the comments at the beginning of the generated Simscape component files. You can use the optional `unsupportedCommands` output argument to generate a `struct` array that lists unsupported SPICE commands for each subcircuit.
3. Build the library using `ssc_build` or add individual components to your model using Simscape Component blocks.

There are many different SPICE simulators with variations in syntax and syntax interpretation. The conversion assistant uses the same syntax as Cadence® PSpice and, where such differences exist, complies with PSpice.

## Commands

The SPICE conversion assistant supports these commands:

- `.FUNC` — Reusable function
- `.PARAM` — Definable parameter
- `.MODEL` — Set of reusable component parameters
- `.SUBCKT` — Subcircuit
- `.LIB` — Directive to include models from an external netlist
- `.INC` — Directive to include contents of external netlist

The conversion assistant implements `.FUNC` SPICE commands using Simscape functions. These functions are placed inside a package sublibrary named `+`*`subcircuit_name`*`_simscape_functions`, where *`subcircuit_name`* is the name of the subcircuit being converted.

Specify the `.MODEL` syntax for resistors, capacitors, and inductors, as

```
.MODEL <model name> res(r=<value>)
.MODEL <model name> cap(c=<value>)
.MODEL <model name> ind(l=<value>)
```

where the `r`, `c`, and `l` values are scaling factors for the value specified on the component declaration. This behavior complies with PSpice, but is not consistent across all simulators.

The conversion assistant does not automatically convert initial conditions specified using the `.IC` statement. However, you can specify initial conditions for capacitors and inductors using the syntax `IC=<value>`. Also, you can manually convert any `.IC` statements from the generated Simscape component files.

Because the purpose of the conversion assistant is to help convert SPICE subcircuits into Simscape blocks, simulation commands, such as `.TRAN`, are ignored.

## Numeric Suffixes

The conversion assistant supports these numeric SPICE suffixes:

| Suffix | Name | Scale |
|---|---|---|
| `T` | Tera | `1e12` |
| `G` | Giga | `1e9` |
| `MEG` | Mega | `1e6` |
| `K` | Kilo | `1e3` |
| `M` | Milli | `1e-3` |
| `MIL` | -- | `25.4e-6` |
| `U` | Micro | `1e-6` |
| `N` | Nano | `1e-9` |
| `P` | Pico | `1e-12` |
| `F` | Femto | `1e-15` |

## Mathematical Functions

The conversion assistant supports these basic mathematical functions used in SPICE and Simscape. These basic mathematical functions might not be continuous or smooth and can cause numerical problems during the simulation. To solve these issues, these functions might require smoothing.

The `subcircuit2ssc` function automatically turns on the smoothing parameters when converting a SPICE subcircuit. To disable the smoothing option, inside the block mask of the generated Simscape component block, set the **Specify function smoothing parameters** parameter to `No`.

## Elementary Math

| Name | SPICE Function | Simscape Function |
|---|---|---|
| Absolute value | `abs` | `simscape.function.abs(x,ZC)` |
| Smallest element | `min` | `simscape.function.minm(x,y,n, epsilon)` |

| Name | SPICE Function | Simscape Function |
|---|---|---|
| Largest element | `max` | `simscape.function.maxm(x,y,n,epsilon)` |
| Sign function | `sgn` | `simscape.function.sign(x,ZC)` |

## Trigonometry

| Name | SPICE Function | Simscape Function |
|---|---|---|
| Sine | `sin` | `sin` |
| Inverse sine | `asin` | `simscape.function.asinm(x,warn)` |
| Hyperbolic sine | `sinh` | `simscape.function.sinhm(x,maxAbsX,warn)` |
| Cosine | `cos` | `cos` |
| Inverse cosine | `acos` | `simscape.function.acosm(x,warn)` |
| Hyperbolic cosine | `cosh` | `simscape.function.coshm(x,maxAbsX,warn)` |
| Tangent | `tan` | `simscape.function.tanm(x,flag,epsilon,x0,warn)` |
| Inverse tangent | `atan` | `atan` |
| Four-quadrant inverse tangent | `atan2` | `atan2` |
| Hyperbolic tangent | `tanh` | `tanh` |

## Exponents and Logarithms

| Name | SPICE Function | Simscape Function |
|---|---|---|
| Power | `x**y`, `pwr(x,y)`, or `pwrs(x,y)` | `simscape.function.powerRational(x,y,flag,epsilon,warn)` |
| Exponential | `exp` | `simscape.function.expm(x,xl,xh,warn)` |
| Natural logarithm | `ln` or `log` | `simscape.function.logm(x,x0,warn)` |
| Base-10 logarithm | `log10` | `simscape.function.log10m(x,x0,warn)` |
| Square root | `sqrt` | `simscape.function.sqrtm(x,flag,epsilon,warn)` |

The conversion assistant interprets `log()` as the natural logarithm rather than the base-10 logarithm. Not all SPICE simulators are consistent in this regard, so ensure that this interpretation is congruent with your SPICE model.

## Other

In addition, the conversion assistant supports these SPICE and Simscape functions:

| Name | SPICE Function | Simscape Function |
|---|---|---|
| If condition | `if` | `if else` |
| Saturation | `limit` | `simscape.function.limitm(x,a,b,n,epsilon)` |
| Current through device | `i` | `i` |
| Voltage across device | `v` | `v` |
| Step function | `stp` | `if x>0, 1 else 0 end` |
| Derivative (see Limitations on page 3-64) | `ddt` | `der` |
| Table | `table` | `simscape.tablelookup([x1,x2,…xn],[y1,y2,…yn],x,interpolation=linear,extrapolation=nearest)` |

## List of Simscape Functions

#### simscape.function.hyp(x,epsilon) — Function with positive output

This function always outputs a positive value. Use this function to protect against issues such as division by zero. `epsilon` determines the smoothness of the transition and is the absolute error of the hyp function when `x=0`.

When `x>epsilon`, the output of this function tends to `x`.

#### simscape.function.abs(x,ZC) — Absolute function

This function returns the absolute value of the input argument, `x`.

You can enable or disable zero crossing by specifying the input argument `ZC` to either `true` or `false`.

#### simscape.function.expm(x,xl,xh,warn) — Exponential function

This function returns the exponential of `x`, `exp(x)`, when `x` ranges between `xl` and `xh`. Otherwise it uses linear extrapolation with gradient matching at the boundary.

Use this function to avoid numerical issues associated with `exp(x)` for very small or large input arguments.

#### simscape.function.coshm(x,maxAbsX,warn) — Hyperbolic cosine function

This function returns the hyperbolic cosine of `x`, `cosh(x)`, when `x` ranges between `-maxAbsX` and `maxAbsX`. Otherwise it uses linear extrapolation with gradient matching at the boundary.

Use this function to avoid numerical issues associated with `cosh(x)` for large negative or large positive input arguments.

**simscape.function.sinhm(x,maxAbsX,warn) — Hyperbolic sine function**

This function returns the hyperbolic sine of `x`, `sinh(x)`, when `x` ranges between `-maxAbsX` and `maxAbsX`. Otherwise it uses linear extrapolation with gradient matching at the boundary.

Use this function to avoid numerical issues associated with `sinh(x)` for large negative or large positive input arguments.

**simscape.function.acosm(x,warn) — Inverse cosine function**

This function returns the inverse cosine of `x`, `acos(x)`, for `-1≤x≤1`, `0` for `x>1`, and π for `x<-1`. This protects against small numerical incursions into the invalid input range for `acos(x)`.

**simscape.function.asinm(x,warn) — Inverse sine function**

This function returns the inverse sine of `x`, `asin(x)`, for `-1≤x≤1`, `0` for `x>1`, and π for `x<-1`. This protects against small numerical incursions into the invalid input range for `asin(x)`.

**simscape.function.limitm(x,a,b,n,epsilon) — Limit function**

This function returns `x` limited between `xl` and `xh`. The input argument, `n`, sets the order of the smoothing function.

Argument `epsilon` determines the start of the transition and is the absolute error of the function when *x* is equal to either `xl` or `xh`.

**simscape.function.logm(x,x0,warn) — Natural logarithm function**

This function returns the natural logarithm of `x`, `log(x)`, when `x` is greater than `x0`. Otherwise it uses linear extrapolation with gradient matching at the boundary.

Use this function when the value of `x` reaches close to `0` during simulation. Set `x0` between `0` and `1`.

**simscape.function.log10m(x,x0,warn) — Base-10 logarithm function**

This function returns the base-10 logarithm of `x`, `log10(x)`, when `x` is greater than `x0`. Otherwise it uses linear extrapolation with gradient matching at the boundary.

Use this function when the value of `x` reaches close to `0` during simulation. Set `x0` between `0` and `1`.

**simscape.function.maxm(x,y,n,epsilon) — Maximum function**

This function returns the largest argument between `x` and `y`. The input argument, `n`, sets the order of the smoothing function.

Argument `epsilon` determines the start of the transition and is the absolute error of the function when *x* is equal to `y`.

**simscape.function.minm(x,y,n,epsilon) — Minimum function**

This function returns the smallest argument between `x` and `y`. The input argument, `n`, sets the order of the smoothing function.

Argument `epsilon` determines the start of the transition and is the absolute error of the function when *x* is equal to `y`.

**simscape.function.powerRational(x,y,flag,epsilon,warn) — Power function**

Power function `x^y`, where `y` is rational, with protection for negative `x`.

If the `flag` argument is `0`, the function returns `x^y` for `x≥0` and `-(-x)^y` for `x<0`. If the `flag` argument is `1`, apply the power function to `simscape.function.hyp(x,epsilon)`.

The SPICE conversion assistant converts the `pwr(x,y)` SPICE function to `simscape.function.powerRational(x,y,1,0,warn)`

The SPICE conversion assistant converts the `pwrs(x,y)` SPICE function to `simscape.function.powerRational(x,y,0,0,warn)`

**simscape.function.sign(x,ZC) — Signum function**

This function returns the sign of the input argument, `x`.

You can enable or disable zero crossing by specifying the input argument `ZC` to either `true` or `false`.

**simscape.function.sqrtm(x,flag,epsilon,warn) — Square root function**

Square root function with input argument protection.

If the `flag` argument is `0`, this function returns `sqrt(x)` for positive `x` and `-sqrt(-x)` for negative `x`. If the `flag` argument is `1`, apply the square root function to `simscape.function.hyp(x,epsilon)`.

**simscape.function.tanm(x,flag,epsilon,x0,warn) — Tangent function**

Tangent function with input protection.

If the `flag` argument is `0`, this function returns the tangent of `x`, `tan(x)`, when `x` is equal to `2π` or `-π/2`. The `epsilon` argument determines the smoothness of the transition at these points.

If the `flag` argument is `1`, this function returns the tangent of `x`, `tan(x)`, when `x` ranges between `-x0` and `x0`. Otherwise it uses linear extrapolation with gradient matching at the boundary.

`1/epsilon` is the maximum allowed value of the output argument `y`.

## Symbols

The conversion assistant recognizes these SPICE symbols:

- `+` at the start of a line indicates line continuation from the previous line
- `*` at the start of a line indicates that the entire line is a comment
- `;` within a line indicates the beginning of an inline comment

## Components

The notation for SPICE commands in this section follows these rules:

- `<argument>` refers to a required item in a command line
- `<argument>*` refers to a required item in a command line that occur one or more times

- `[argument]` refers to an optional item in a command line
- `[argument]*` refers to an optional item in a command line that occur zero or more times

This list shows the full set of supported SPICE components, and their supported SPICE netlist notations. You can specify only the `.MODEL` parameters that differ from SPICE default values.

## Sources

- Independent voltage source

```
V<name> <+ node> <- node> [DC] <value>
V<name> <+ node> <- node> exp(<v1> <v2> <td1> <tc1> <td2> <tc2>)
V<name> <+ node> <- node> pulse(<v1> <v2> <td> <tr> <tf> <pw> <per>)
V<name> <+ node> <- node> pwl(<<tj> <vj>>*)
V<name> <+ node> <- node> sffm(<voff> <vampl> <fc> <mod> <fm>)
V<name> <+ node> <- node> sin(<voff> <vampl> <freq> <td> <df>)
```

- Independent current source

```
I<name> <+ node> <- node> [DC] <value>
I<name> <+ node> <- node> exp(<i1> <i2> <td1> <tc1> <td2> <tc2>)
I<name> <+ node> <- node> pulse(<i1> <i2> <td> <tr> <tf> <pw> <per>)
I<name> <+ node> <- node> pwl(<<tj> <ij>>*)
I<name> <+ node> <- node> sffm(<ioff> <iampl> <fc> <mod> <fm>)
I<name> <+ node> <- node> sin(<ioff> <iampl> <freq> <td> <df>)
```

- Current-controlled voltage source

```
H<name> <+ node> <- node> <voltage source name> <gain>
H<name> <+ node> <- node> VALUE={<expression>}
H<name> <+ node> <- node> POLY(<value>) <voltage source name>* <coefficient>*
H<name> <+ node> <- node> TABLE {<expression>}=< <input value>, <output value> >*
H<name> <+ node> <- node> <voltage source name> TABLE=< <input value>, <output value> >*
```

- Voltage-controlled voltage source

```
E<name> <+ node> <- node> <+ control node> <- control node> <gain>
E<name> <+ node> <- node> VALUE={<expression>}
E<name> <+ node> <- node> POLY(<value>) <<+ control node> <- control node>>* <coefficient>*
E<name> <+ node> <- node> TABLE {<expression>}=< <input value>, <output value> >*
E<name> <+ node> <- node> <+ control node> <- control node> TABLE=< <input value>, <output value> >*
```

- Current-controlled current source

```
F<name> <+ node> <- node> <voltage source name> <gain>
F<name> <+ node> <- node> VALUE={<expression>}
F<name> <+ node> <- node> POLY(<value>) <voltage source name>* <coefficient>*
F<name> <+ node> <- node> TABLE {<expression>}=< <input value>, <output value> >*
F<name> <+ node> <- node> <voltage source name> TABLE=< <input value>, <output value> >*
```

- Voltage-controlled current source

```
G<name> <+ node> <- node> <+ control node> <- control node> <gain>
G<name> <+ node> <- node> VALUE={<expression>}
G<name> <+ node> <- node> POLY(<value>) <<+ control node> <- control node>>* <coefficient>*
G<name> <+ node> <- node> TABLE {<expression>}=< <input value>, <output value> >*
G<name> <+ node> <- node> <+ control node> <- control node> TABLE=< <input value>, <output value> >*
```

- Behavioral source (The <expression> does not need to appear in braces {})

```
B<name> <+ node> <- node> V=<expression>
B<name> <+ node> <- node> I=<expression>
```

## Passive Devices

- Resistor

```
R<name> <+ node> <- node> [model name] <value>
.MODEL <model name> res(r=<value>)
```

- Capacitor

```
C<name> <+ node> <- node> [model name] <value> [IC=<value>]
.MODEL <model name> cap(c=<value>)
```

- Inductor

```
L<name> <+ node> <- node> [model name] <value> [IC=<value>]
.MODEL <model name> ind(l=<value>)
```

- Inductor coupling

```
K<name> <inductor name> <inductor name>* <value>
```

## Switches

- Voltage-controlled switch

```
S<name> <+ node> <- node> <+ control node> <- control node> <model name>
.MODEL <model name> sw(ron=<value>, roff=<value>, vt=<value>, vh=<value>)
```

- Current-controlled switch

```
W<name> <+ node> <- node> <voltage source name> <model name>
.MODEL <model name> csw(ron=<value>, roff=<value>, it=<value>, ih=<value>)
```

## Semiconductor Devices

- Diode

```
D<name> <+ node> <- node> <model name> [area]
.MODEL <model name> d(is=<value>, rs=<value>, n=<value>, cjo=<value>, vj=<value>,
    +m=<value>, fc=<value>, tt=<value>, revbrk=<value>, bv=<value>, ibv=<value>,
    +xti=<value>, eg=<value>)
```

- Bipolar junction transistor (BJT)

  NPN

```
Q<name> <collector node> <base node> <emitter node> [substrate node] <model name> <area>
.MODEL <model name> npn(bf=<value>, br=<value>, cjc=<value>, cje=<value>, cjs=<value>,
    +eg=<value>, fc=<value>, ikf=<value>, ikr=<value>, irb=<value>, is=<value>, isc=<value>,
    +ise=<value>, itf=<value>, mjc=<value>, mje=<value>, mjs=<value>, nc=<value>, ne=<value>,
    +nf=<value>, nr=<value>, rb=<value>, rbm=<value>, rc=<value>, re=<value>, tf=<value>,
    +tr=<value>, vaf=<value>, var=<value>, vjc=<value>, vje=<value>, vjs=<value>, vtf=<value>,
    +xcjc=<value>, xtb=<value>, xtf=<value>, xti=<value>)
```

  PNP

```
Q<name> <collector node> <base node> <emitter node> [substrate node] <model name> <area>
.MODEL <model name> pnp(bf=<value>, br=<value>, cjc=<value>, cje=<value>, cjs=<value>,
    +eg=<value>, fc=<value>, ikf=<value>, ikr=<value>, irb=<value>, is=<value>, isc=<value>,
    +ise=<value>, itf=<value>, mjc=<value>, mje=<value>, mjs=<value>, nc=<value>, ne=<value>,
    +nf=<value>, nr=<value>, rb=<value>, rbm=<value>, rc=<value>, re=<value>, tf=<value>,
    +tr=<value>, vaf=<value>, var=<value>, vjc=<value>, vje=<value>, vjs=<value>, vtf=<value>,
    +xcjc=<value>, xtb=<value>, xtf=<value>, xti=<value>)
```

- Junction field-effect transistor (JFET)

  N-Channel

```
J<name> <drain node> <gate node> <source node> <model name> [area]
.MODEL <model name> njf(beta=<value>, cgd=<value>, cgs=<value>, fc=<value>, is=<value>,
    +lambda=<value>, m=<value>, n=<value>, rd=<value>, rs=<value>, vto=<value>, xti=<value>)
```

  P-Channel

```
J<name> <drain node> <gate node> <source node> <model name> [area]
.MODEL <model name> pjf(beta=<value>, cgd=<value>, cgs=<value>, fc=<value>, is=<value>,
    +lambda=<value>, m=<value>, n=<value>, rd=<value>, rs=<value>, vto=<value>, xti=<value>)
```

- Metal-oxide-semiconductor field-effect transistor (MOSFET)

  N-Channel (only level-1 and level-3 are supported)

```
M<name> <drain node> <gate node> <source node> <bulk node> <model name>
    +[L=<value>] [W=<value>] [AD=<value>] [AS=<value>] [PD=<value>] [PS=<value>] [NRD=<value>]
```

```
    +[NRS=<value>] [M=<value>]
.MODEL <model name> nmos(cbd=<value>, cbs=<value>, cgbo=<value>, cgdo=<value>,
    +cgso=<value>, cj=<value>, cjsw=<value>, delta=<value>, eta=<value>, fc=<value>,
    +gamma=<value>, is=<value>, js=<value>, kappa=<value>, kp=<value>, lambda=<value>,
    +ld=<value>, level=<value>, mj=<value>, mjsw=<value>, n=<value>, neff=<value>, nfs=<value>,
    +nss=<value>, nsub=<value>, nrd=<value>, nrs=<value>, pb=<value>, phi=<value>, rd=<value>,
    +rs=<value>, rsh=<value>, theta=<value>, tox=<value>, tpg=<value>, ucrit=<value>,
    +uexp=<value>, uo=<value>, vmax=<value>, vto=<value>, xj=<value>)
```

P-Channel (only level-1 and level-3 are supported)

```
M<name> <drain node> <gate node> <source node> <bulk node> <model name>
    +[L=<value>] [W=<value>] [AD=<value>] [AS=<value>] [PD=<value>] [PS=<value>] [NRD=<value>]
    +[NRS=<value>] [M=<value>]
.MODEL <model name> pmos(cbd=<value>, cbs=<value>, cgbo=<value>, cgdo=<value>,
    +cgso=<value>, cj=<value>, cjsw=<value>, delta=<value>, eta=<value>, fc=<value>,
    +gamma=<value>, is=<value>, js=<value>, kappa=<value>, kp=<value>, lambda=<value>,
    +ld=<value>, level=<value>, mj=<value>, mjsw=<value>, n=<value>, neff=<value>, nfs=<value>,
    +nss=<value>, nsub=<value>, nrd=<value>, nrs=<value>, pb=<value>, phi=<value>, rd=<value>,
    +rs=<value>, rsh=<value>, theta=<value>, tox=<value>, tpg=<value>, ucrit=<value>,
    +uexp=<value>, uo=<value>, vmax=<value>, vto=<value>, xj=<value>)
```

## Subsystems

- Subcircuit

```
X<name> [node]* <subcircuit name> [PARAMS: < <name>=<value> >*]
```

## Performing Manual Conversions

After you generate the Simscape component files, inspect each file header for messages regarding unsupported SPICE commands. For example, the conversion assistant does not support the implementation of temperature coefficients for resistors:

```
R1 p n 1k TC=0.01,-0.002
```

The generated Simscape component file contains all the supported conversions, and this header, which identifies the temperature coefficients of the resistor for manual conversion:

```
component test
% test
% Component automatically generated from a SPICE netlist (11-Dec-2018 09:34:57).
% Users should manually implement the following SPICE commands in order to
% achieve a complete implementation:
%    R1: tc 0.01 -0.002
```

To complete the conversion, modify the Simscape component file to implement the missing components. For more information about performing manual conversions and this particular scenario, see `subcircuit2ssc`.

## Parasitics Values

For passive devices such as capacitors and inductors, to introduce parasitic values in the generated Simscape component files, set the **Specify parasitics values** parameter to **Yes**. Then specify the value of the **Capacitor parasitic series resistance** or **Inductor parasitic parallel conductance** parameters.

## Limitations

- The netlist must be written in PSpice format and be syntactically correct. The conversion assistant does not check for proper PSpice syntax.

- Only a subset of the PSpice netlist language is supported. However, unsupported PSpice commands are identified at the top of the corresponding Simscape component file to facilitate manual conversion.
- To build generated Simscape components into Simscape blocks, parameter values must conform to Simscape constraints. For example, capacitance of a fundamental capacitor and inductance of a fundamental inductor must be nonzero.
- The conversion assistant does not support the use of the derivative SPICE function, `ddt`, inside a function call.

## See Also

`subcircuit2ssc` | `ssc_build`

## More About

- “Building Custom Block Libraries”
- “Composite Components”

# Photovoltaic Thermal (PV/T) Hybrid Solar Panel

This example shows how to model the cogeneration of electrical power and heat using a hybrid PV/T solar panel. The generated heat is transferred to water for household consumption.

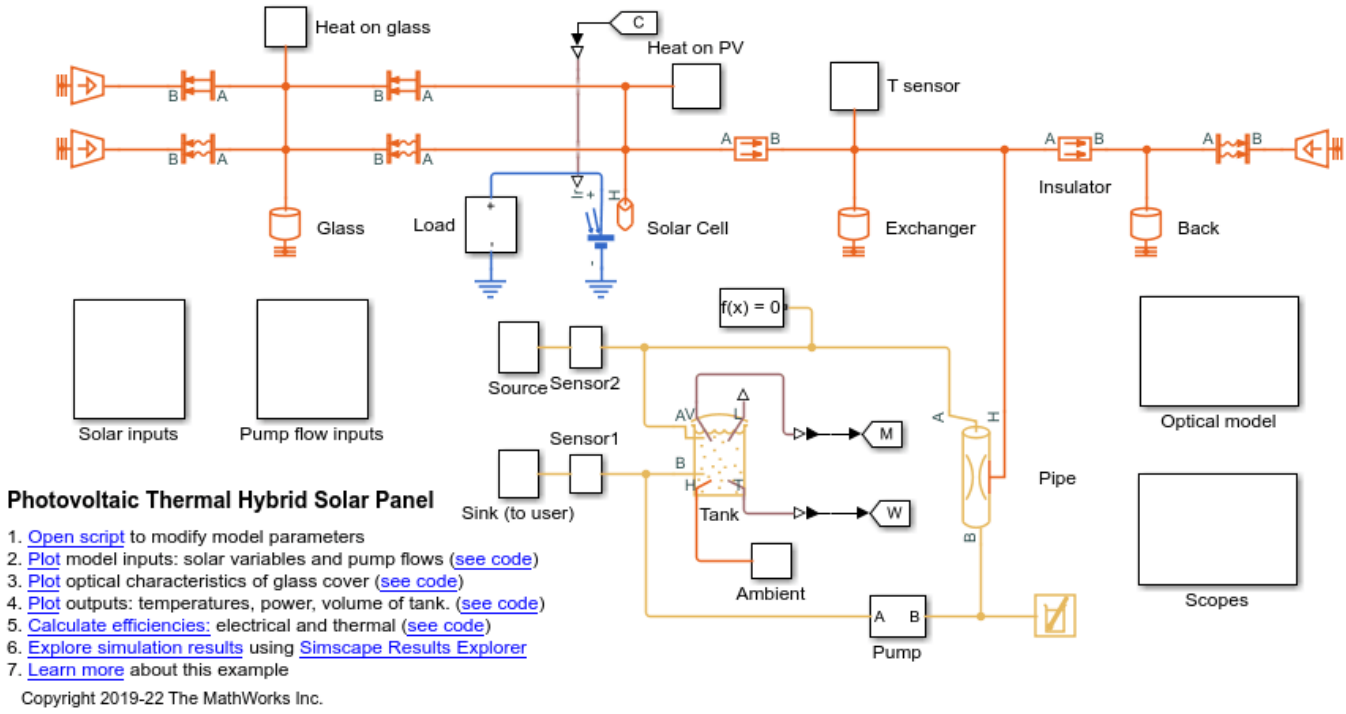
It uses blocks from the Simscape™ Foundation™, Simscape Electrical™, and Simscape Fluids™ libraries. The electrical portion of the network contains a Solar Cell block, which models a set of photovoltaic (PV) cells, and a Load subsystem, which models a resistive load. The thermal network models the heat exchange that occurs between the physical components of the PV panel (glass cover, heat exchanger, back cover) and the environment. Heat is exchanged through conduction, convection, and radiation. The thermal-liquid network contains a pipe, a tank, and pumps. The pumps control the flows of the liquids through the system.

To model the reflection, absorption and transmission of light in the glass cover, an optical model is embedded in a MATLAB® Function block.

### Model overview

Open the model to view its structure:

```
open_system('sscv_hybrid_solar_panel');
```




The thermal network is in red, the electrical network in blue and the thermal liquid network in yellow. There are subsystems for the solar and pump inputs. There is also a subsystem that contains scopes for visualizing the simulation results. Another subsystem contains the function for the optical model.

### Parameters

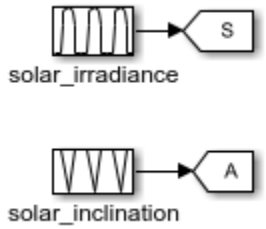
You can use the hybrid_solar_panel_data.m script to change the parameter values that this example uses for components such as the load, solar cell, pipe, and tank.

```
edit sscv_hybrid_solar_panel_data;
```

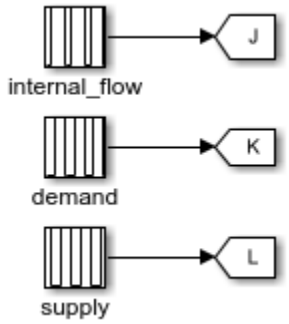
### Inputs

The inputs of the model are the pump flows and the solar variables for irradiance and incidence angle. A repeating sequence block is used to define the inputs because they follow a 24-hour periodic cycle.

```
open_system('sscv_hybrid_solar_panel/Solar inputs');
```




```
open_system('sscv_hybrid_solar_panel/Pump flow inputs');
```

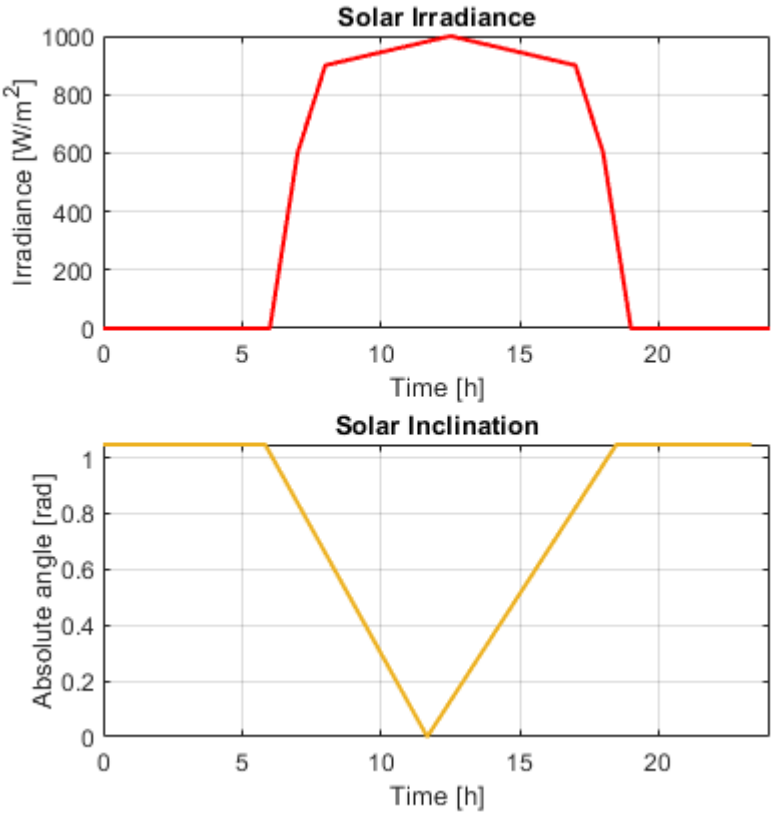


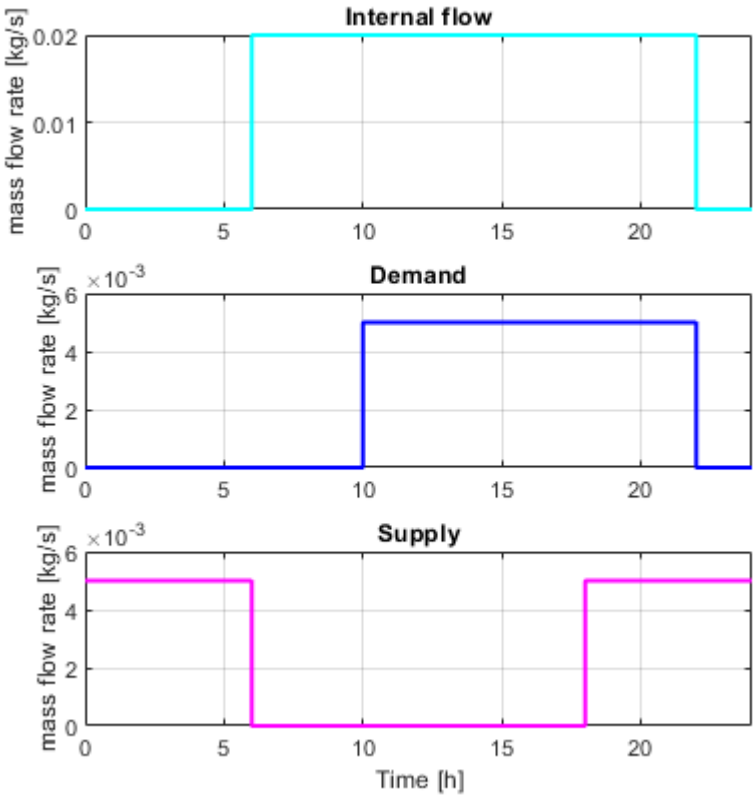

The sun rises at 6:00 and sets at 19:00. The irradiance follows a bell curve that peaks at 12:30. The incidence angle changes from pi/3 to 0.

There are three pumps. One pump models user demand, another models source supply, and a third models internal flow that forces convection in the pipe. The demand is constant and only non-zero from 10:00 to 22:00. The supply is constant and only non-zero from 18:00 to 6:00. The internal flow is also constant and only non-zero from 6:00 to 22:00. This model is used for the internal flow because it is not efficient to force heat exchange during the night when the ambient temperature is low.

You can use the hybrid_solar_panel_plot_inputs.m script to plot the inputs:

```
sscv_hybrid_solar_panel_plot_inputs;
```

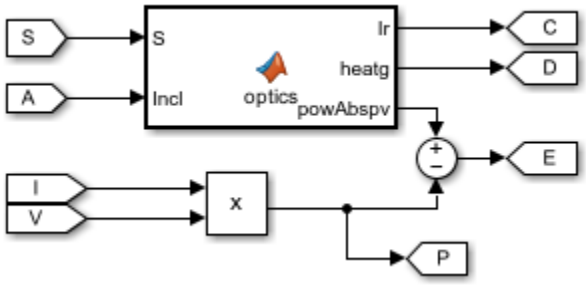

Solar Irradiance
Irradiance [W/m^2]
1000
800
600
400
200
0
0
5
10
15
20
Time [h]
Solar Inclination
Absolute angle [rad]
1
0.8
0.6
0.4
0.2
0
0
5
10
15
20
Time [h]




### Optical model for the glass cover

The optical model is inside a subsystem:

```
open_system('sscv_hybrid_solar_panel/Optical model');
```




It consists of a MATLAB® Function block, with the 2 solar inputs, and 3 outputs: the transmitted irradiance on the PV cells, the heat absorbed by the glass, and the radiative power absorbed by the PV cells. Part of it will be transformed into electrical power (V*I) and the rest will be heat absorbed by the PV cells.

From an optical point of view, the glass consists of 2 parallel boundaries (air-glass, glass-air), each one of those reflects and transmits light. The reflection coefficient in a boundary is obtained from the

**Fresnel equations**. $r_p$ is for P-polarization and $r_s$ for S-polarization. The total reflection is the average of both, and the transmittance is $1 - r$ as there is no absorption so far:

$$r_p = \left( \frac{n_{rel}^2 \cos(\theta_i) - \sqrt{n_{rel}^2 - \sin(\theta_i)^2}}{n_{rel}^2 \cos(\theta_i) + \sqrt{n_{rel}^2 - \sin(\theta_i)^2}} \right)^2$$

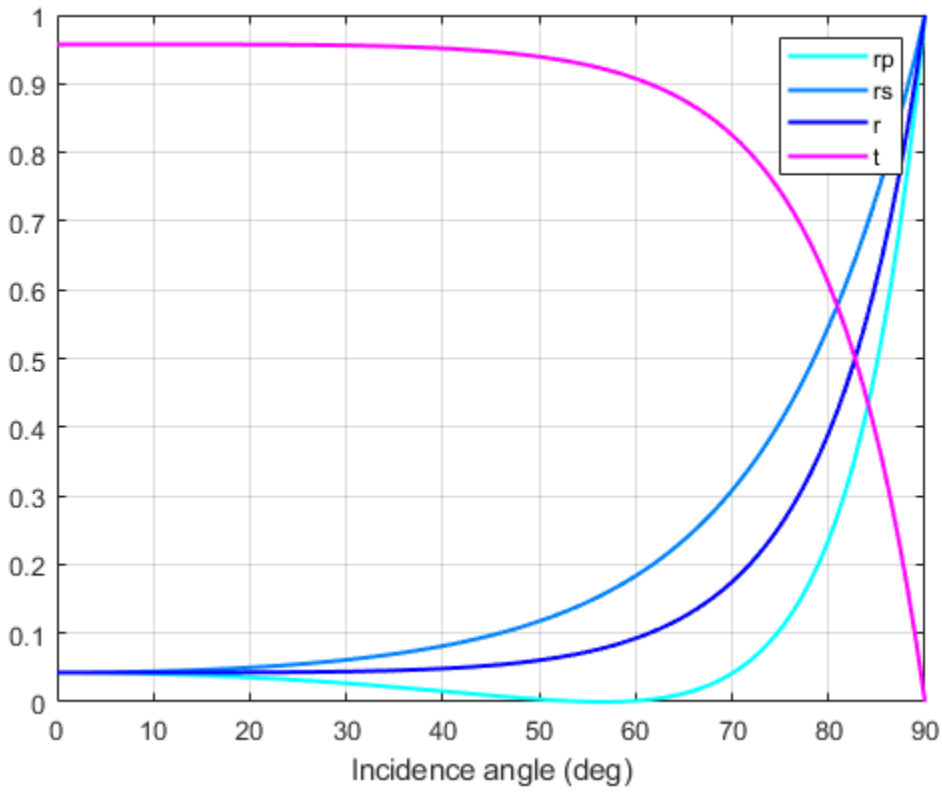
$$r_s = \left( \frac{\cos(\theta_i) - \sqrt{n_{rel}^2 - \sin(\theta_i)^2}}{\cos(\theta_i) + \sqrt{n_{rel}^2 - \sin(\theta_i)^2}} \right)^2$$

$$r = \frac{1}{2}(r_p + r_s)$$

$$t = 1 - r$$

This is an example of the optical coefficients rp, rs, r and t in function of incidence angle:

```
nrel = 1.52; %Optical index from air to glass
theta = linspace(0, pi/2, 100);
rp = ( nrel^2*cos(theta) - sqrt(nrel^2 - sin(theta).^2) ).^2./...
    ( nrel^2*cos(theta) + sqrt( nrel^2 - sin(theta).^2 ) ).^2 ;
rs = ( cos(theta) - sqrt(nrel^2 - sin(theta).^2) ).^2./...
    ( cos(theta) + sqrt( nrel^2 - sin(theta).^2 ) ).^2 ;
r = 0.5*(rp + rs);
t = 1 - r;

figure();
plot(theta*180/pi, rp, 'Color', [0 1 1], 'LineWidth', 1.5);
hold on
plot(theta*180/pi, rs, 'Color', [0 0.5 1], 'LineWidth', 1.5);
plot(theta*180/pi, r, 'Color', [0 0 1], 'LineWidth', 1.5);
plot(theta*180/pi, t, 'Color', 'm', 'LineWidth', 1.5);
legend('rp','rs','r','t');
xlabel('Incidence angle (deg)');
grid on
box on
```




This is what happens in one boundary, but the glass has 2 parallel boundaries separated by $d_g$. The angle after the 1st boundary is the incidence angle on the 2nd boundary and is calculated from **Snell's Law**:

$$n_1 \sin(\theta_1) = n_2 \sin(\theta_2)$$

When the light enters the glass, it absorbs part of it with a constant probability per unit length (alpha_g), resulting in an exponential decay from distance travelled for the transmittance coefficient in the glass:

$$\tau_g = \exp\left(\frac{-\alpha_g d_g}{\cos(\theta_2)}\right)$$

Then, when it arrives at the 2nd boundary, it reflects and transmits again with Fresnel equations. The reflected light is trapped inside the glass, reflecting infinite times between the 2 boundaries until completely absorbed. The total reflection and transmission coefficients of the system are then the sum of an infinite geometrical series, for which the result is:

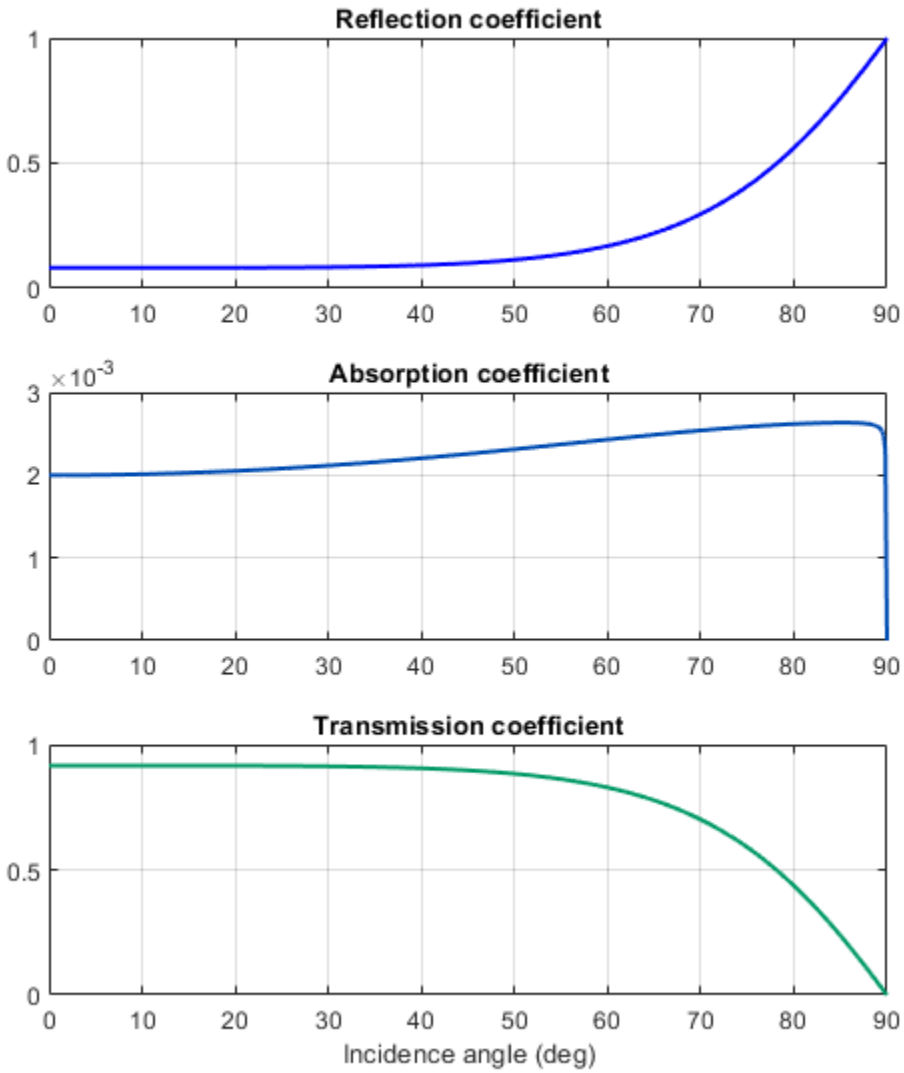
$$T_g = \frac{t_1 \tau_g t_2}{1 - r_1 r_2 \tau_g^2}$$

$$R_g = r_1 + \frac{t_1^2 \tau_g^2 r_2}{1 - r_1 r_2 \tau_g^2}$$

$$A_g = 1 - T_g - R_g$$

Finally, the total optical coefficients for the glass are:

```
sscv_hybrid_solar_panel_plot_optics;
```

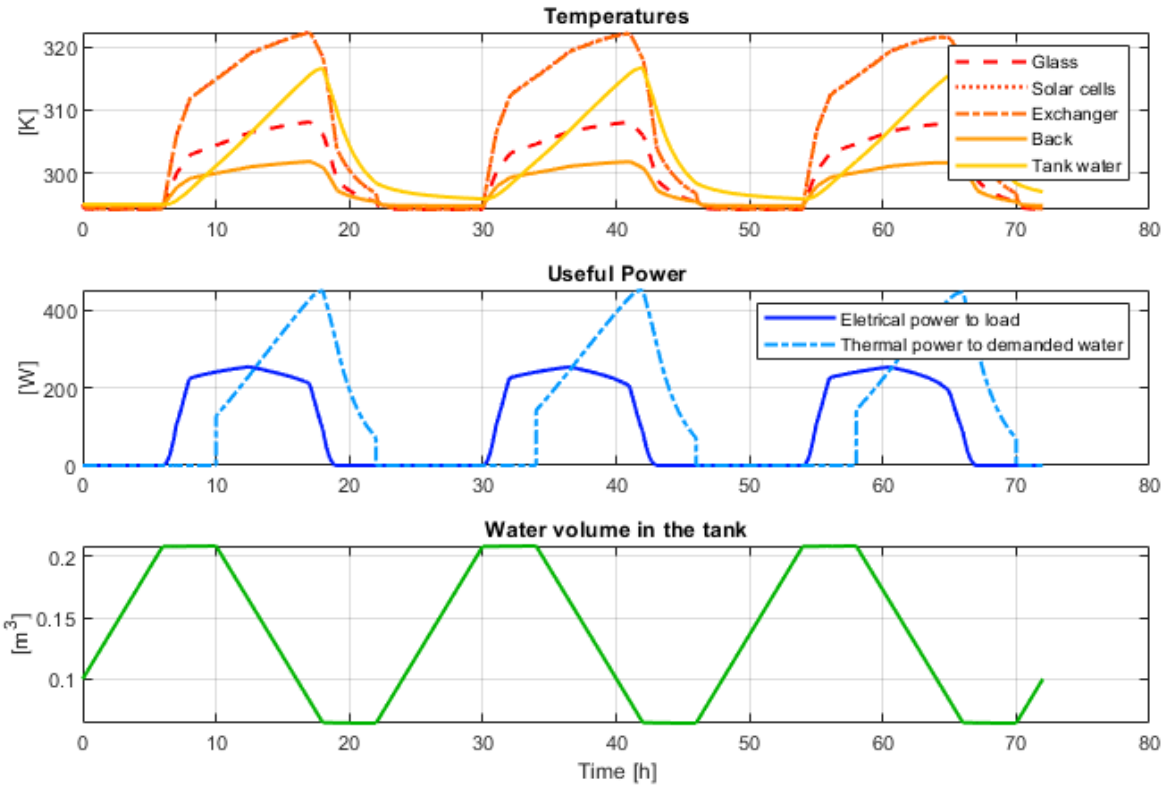



### Outputs

The outputs of the model are the temperatures of all components of the panel, the electrical and thermal power, and the volume in the tank.

You can use the script hybrid_solar_panel_plot_outputs to plot the solution:

```
sscv_hybrid_solar_panel_plot_outputs;
```




### Efficiency calculation

From the outputs it is possible to calculate the electrical, thermal, and total efficiency of the panel:

```
sscv_hybrid_solar_panel_efficiency;


****** Efficiency Calculation *********

Total input energy from the sun in the period: 43.7879 kWh
Average input energy from the sun per day: 14.596 kWh/day

Total electrical energy supplied to the load: 7.5325 kWh
Average electrical energy supplied per day: 2.5108 kWh/day

Total absolute thermal energy in the water supplied to the user: 26.4735 kWh
Total absolute thermal energy in the water extracted from the source: 16.5049 kWh

Total used thermal energy (sink - source): 9.9686 kWh
Average used thermal energy per day (sink - source): 3.3229 kWh/day

Electrical efficiency: 0.17202
Thermal efficiency: 0.22766
Total efficiency: 0.39968

**************************************
```

The electrical efficiency is on the order of standard PV cells, but adding the thermal efficiency the production of energy is significantly better, with a system efficiency on the order of a cogeneration plant.

A further analysis could use Simulink® Design Optimization™ or other optimization tools to find optimal values for certain parameters eligible for control, maximizing total efficiency.

Another improvement would be the addition of controllers to the pumps and the electrical load, in order to drive the system to different operating points and optimize the performance.

## See Also

Solar Cell

# List of Pre-Parameterized Components

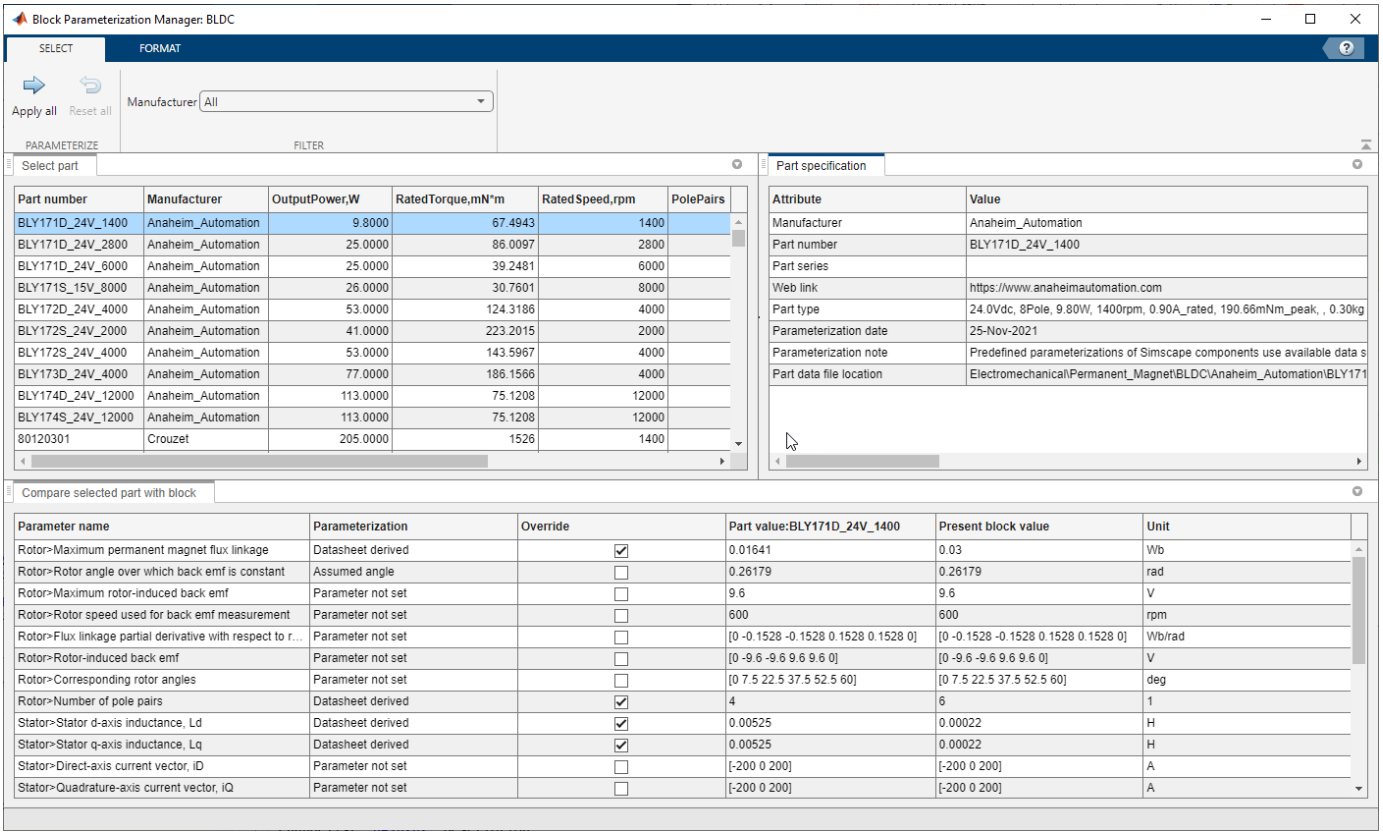
| In this section... |
| --- |
| “Load Predefined Parameterizations” on page 3-75 |
| “List of Pre-Parameterized Components” on page 3-77 |
| “Additional Parameterizations” on page 3-102 |
| “Request a Pre-Parameterization” on page 3-103 |

Many blocks in Simscape Electrical can be pre-parameterized to represent components by specific suppliers. These parameters match the manufacturer data sheets. You can pre-parameterize your block by using the Block Parameterization Manager.

**Note** The predefined parameterizations of Simscape components use available data sources for the parameter values. Engineering judgement and simplifying assumptions are used to fill in for missing data. As a result, expect deviations between simulated and actual physical behavior. To ensure accuracy, validate the simulated behavior against experimental data and refine component models as necessary.

## Load Predefined Parameterizations

To load a predefined parameterization, double-click a block, then click the **<click to select>** hyperlink of the **Selected part** parameter. The Block Parameterization Manager window opens.

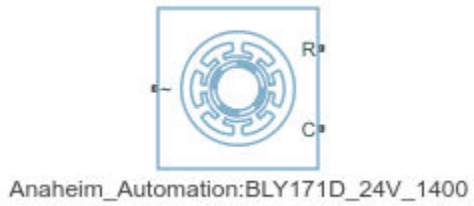



In the **Filter** settings of the **Select** tab, specify the supplier from the **Manufacturer** list. Then, in the **Select part** section, click the part from the list of available components.

When you click a specific part, the **Part specification** section contains the information about that specific component, and the **Compare selected part with block** section lists the parameter names of the block and compares their actual values with the datasheet values of the chosen pre-parameterized component.

You can also check what fields you can edit. In the **Format** tab, in the **Compare selected part with block** section, click **Highlight Editable fields**. If you click the **Highlight Differences** button, the app highlights the values of the parameters that differ from the those in the chosen pre-parameterized component.

To apply the changes and load the parameterization in this block, click **Apply all**.




The block in your model now shows the pre-parameterized component.

## List of Pre-Parameterized Components

This list shows the available pre-parameterization options for Simscape Electrical blocks.

| Block | Manufacturer | Component |
|---|---|---|
| BLDC | Anaheim_Automation | BLY171D_24V_1400 |
| | | BLY171D_24V_2800 |
| | | BLY171D_24V_6000 |
| | | BLY171S_15V_8000 |
| | | BLY172D_24V_4000 |
| | | BLY172S_24V_2000 |
| | | BLY172S_24V_4000 |
| | | BLY173D_24V_4000 |
| | | BLY174D_24V_12000 |
| | | BLY174S_24V_12000 |
| | Crouzet | 80120301 |
| | | 80140504 |
| | | 80140511 |
| | | 80140512 |
| | | 80180504 |
| | | 80180506 |
| | Faulhaber | 0515G006B |
| | | 1028S006B |
| | | 1028S012B |
| | | 1218S006B |
| | | 1218S012B |
| | | 1226S006B |
| | | 1226S024B |
| | | 1628T012B |
| | | 1628T024B |
| | | 2036U012B |
| | | 2036U024B |
| | | 2036U036B |
| | | 2036U048B |
| | | 2057S012B |
| | | 2057S024B |
| | | 2232S012BX4 |
| | | 2232S024BX4 |
| | | 2250S012BX4 |

| Block | Manufacturer | Component |
|---|---|---|
| | | 2250S012BX4S |
| | | 2250S018BX4 |
| | | 2250S018BX4S |
| | | 2250S024BX4 |
| | | 2250S024BX4S |
| | | 2444S024B |
| | | 2444S048B |
| | | 3056K012B |
| | | 3056K024B |
| | | 3056K036B |
| | | 3056K048B |
| | | 3216W012BXTH |
| | | 3216W012BXTR |
| | | 3216W024BXTH |
| | | 3216W024BXTR |
| | | 3242G012BX4 |
| | | 3242G018BX4 |
| | | 3242G024BX4 |
| | | 3242G036BX4 |
| | | 3242G042BX4 |
| | | 3242G048BX4 |
| | | 3268G018BX4 |
| | | 3268G024BX4 |
| | | 3268G030BX4 |
| | | 3268G036BX4 |
| | | 3268G042BX4 |
| | | 3268G048BX4 |
| | | 3564K012B |
| | | 3564K024B |
| | | 3564K036B |
| | | 3564K048B |
| | | 4221G018BXTH |
| | | 4221G018BXTR |
| | | 4221G024BXTH |
| | | 4221G024BXTR |
| | | 4221G048BXTH |

| Block | Manufacturer | Component |
|---|---|---|
| | | 4221G048BXTR |
| | Fulling_Motor | FL28BL38 |
| | | FL42BLS01_IE |
| | | FL86BLS98 |
| | Maxon | ECXSPEED_13L_18 |
| | | ECXSPEED_13L_24 |
| | | ECXSPEED_13L_36 |
| | | ECXSPEED_13L_48 |
| | | ECXSPEED_13M_18 |
| | | ECXSPEED_13M_24 |
| | | ECXSPEED_13M_36 |
| | | ECXSPEED_13M_48 |
| | | ECXSPEED_16L_18 |
| | | ECXSPEED_16L_24 |
| | | ECXSPEED_16L_36 |
| | | ECXSPEED_16L_48 |
| | | ECXSPEED_16M_18 |
| | | ECXSPEED_16M_24 |
| | | ECXSPEED_16M_36 |
| | | ECXSPEED_16M_48 |
| | | ECXSPEED_19L_18 |
| | | ECXSPEED_19L_24 |
| | | ECXSPEED_19L_36 |
| | | ECXSPEED_19L_48 |
| | | ECXSPEED_19M_24 |
| | | ECXSPEED_19M_36 |
| | | ECXSPEED_19M_48 |
| | | ECXSPEED_22L_24 |
| | | ECXSPEED_22L_36 |
| | | ECXSPEED_22L_48 |
| | | ECXSPEED_22M_18 |
| | | ECXSPEED_22M_24 |
| | | ECXSPEED_22M_36 |
| | | ECXSPEED_22M_48 |
| | | ECXSPEED_4L_3 |
| | | ECXSPEED_6M_6 |

| Block | Manufacturer | Component |
|---|---|---|
| | | ECXSPEED_8M_12 |
| | | ECXSPEED_8M_24 |
| | | ECXSQUARE_16L_12 |
| | | ECXSQUARE_16L_18 |
| | | ECXSQUARE_16L_6 |
| | | ECXSQUARE_16L_9 |
| | | ECXTORQUE_22L_12 |
| | | ECXTORQUE_22L_24 |
| | | ECXTORQUE_22L_48 |
| | | ECXTORQUE_22XL_24 |
| | | ECXTORQUE_22XL_48 |
| | Nanotec | DB22M01 |
| | | DB28L01 |
| | | DB28M01 |
| | | DB33S01 |
| | | DB41C024030_A |
| | | DB41L024030_A |
| | | DB41M024030_A |
| | | DB41S024030_A |
| | | DB43C048030_A |
| | | DB43L024030_A |
| | | DB43L048030_A |
| | | DB43M024030_A |
| | | DB43M048030_A |
| | | DB56C036030_A |
| | | DB56D036030_A |
| | | DB59C048035_A |
| | | DB59L024035R_A |
| | | DB59L024035_A |
| | | DB59L048035_A |
| | | DB80S048030_A |
| | | DBL36L024045_A |
| | | DBL36M024048_A |
| | | DBL42L024032_A |
| | | DF32M024027_A |
| | | DF45L024048_A |

| Block | Manufacturer | Component |
|---|---|---|
| | | DF45M024053_A |
| | | DF45S024050_A |
| | | DFA68M024035_ENEW |
| | | DFA68M024037_A |
| | | DFA90L048017_A |
| | | DFA90L048017_ENEW |
| | | DS28L024080_A |
| | | DS28M024080_A |
| | Transmotec | B4240_24 |
| | | B4260_24 |
| | | B4280_24 |
| | | B86112_48 |
| | | B8672_12 |
| | | B8672_48 |
| | | B8686_12 |
| | | B8686_48 |
| | | BR2268_12 |
| | | BR2877_24 |
| | | BR3337_12 |
| | | BR3337_24 |
| | | BR5743_12 |
| | | BR5753_24 |
| | | BR5773_24 |
| | | BR5793_12 |
| DC Motor | AMETEK | DC040A_2 |
| | | GM8224S028 |
| | | GM9234S023_R1 |
| | | GM9236S020_R1 |
| | | GM9236S024 |
| | | 9237S011_R1_SP |
| | Crouzet | 89_800_003 |
| | | 89_800_008 |
| | | 89_810_003 |
| | | 89_810_007 |
| | | 89_810_008 |
| | | 89_830_003 |

| Block | Manufacturer | Component |
|---|---|---|
| | | 89_830_004 |
| | | 89_830_011 |
| | | 89_830_012 |
| | | 89_850_003 |
| | | 89_850_007 |
| | | 89_850_008 |
| | | 89_890_003 |
| | | 89_890_004 |
| | | 89_890_005 |
| | | 89_890_011 |
| | Faulhaber | 0816K_003_SR |
| | | 0816K_006_SR |
| | | 0816K_009_SR |
| | | 0816K_012_SR |
| | | 1016K_003_SR |
| | | 1016K_006_SR |
| | | 1016K_009_SR |
| | | 1016K_012_SR |
| | | 1024K_003_SR |
| | | 1024K_006_SR |
| | | 1024K_009_SR |
| | | 1024K_012_SR |
| | | 1219N_006_G |
| | | 1219N_012_G |
| | | 1219N_015_G |
| | | 1219N_4p5_G |
| | | 1224N_006_SR |
| | | 1224N_012_SR |
| | | 1224N_015_SR |
| | | 1331T_006_SR |
| | | 1331T_012_SR |
| | | 1331T_024_SR |
| | | 1506N_003_SR |
| | | 1506N_006_SR |
| | | 1516T_1p5_S |
| | | 1524T_003_SR |

| Block | Manufacturer | Component |
|---|---|---|
| | | 1524T_006_SR |
| | | 1524T_009_SR |
| | | 1524T_012_SR |
| | | 1524T_018_SR |
| | | 1524T_024_SR |
| | | 1624T_003_S |
| | | 1624T_006_S |
| | | 1624T_012_S |
| | | 1624T_018_S |
| | | 1624T_024_S |
| | | 1717T_003_SR |
| | | 1717T_006_SR |
| | | 1717T_012_SR |
| | | 1717T_018_SR |
| | | 1717T_024_SR |
| | | 2224U_003_SR |
| | | 2224U_006_SR |
| | | 2224U_012_SR |
| | | 2224U_018_SR |
| | | 2224U_024_SR |
| | | 2224U_036_SR |
| | | 2230T_003_S |
| | | 2230T_006_S |
| | | 2230T_012_S |
| | | 2230T_015_S |
| | | 2230T_024_S |
| | | 2230T_040_S |
| | | 2232U_006_SR |
| | | 2232U_009_SR |
| | | 2232U_012_SR |
| | | 2232U_015_SR |
| | | 2232U_018_SR |
| | | 2232U_024_SR |
| | | 2233T_006_S |
| | | 2233T_012_S |
| | | 2233T_018_S |

| Block | Manufacturer | Component |
|---|---|---|
| | | 2233T_024_S |
| | | 2233T_030_S |
| | | 2233T_4p5_S |
| | | 2237S_006_CXR |
| | | 2237S_012_CXR |
| | | 2237S_018_CXR |
| | | 2237S_024_CXR |
| | | 2237S_036_CXR |
| | | 2237S_048_CXR |
| | | 2342S_006_CR |
| | | 2342S_012_CR |
| | | 2342S_018_CR |
| | | 2342S_024_CR |
| | | 2342S_036_CR |
| | | 2342S_048_CR |
| | | 2607T_006_SR |
| | | 2607T_006_SR_IE2_16 |
| | | 2607T_012_SR |
| | | 2607T_012_SR_IE2_16 |
| | | 2607T_024_SR |
| | | 2607T_024_SR_IE2_16 |
| | | 2642W_012_CR |
| | | 2642W_012_CXR |
| | | 2642W_015_CXR |
| | | 2642W_018_CR |
| | | 2642W_018_CXR |
| | | 2642W_024_CR |
| | | 2642W_024_CXR |
| | | 2642W_036_CR |
| | | 2642W_036_CXR |
| | | 2642W_048_CR |
| | | 2642W_048_CXR |
| | | 2657W_012_CR |
| | | 2657W_012_CXR |
| | | 2657W_018_CR |
| | | 2657W_018_CXR |

| Block | Manufacturer | Component |
|---|---|---|
| | | 2657W_024_CR |
| | | 2657W_024_CXR |
| | | 2657W_030_CXR |
| | | 2657W_036_CR |
| | | 2657W_036_CXR |
| | | 2657W_048_CR |
| | | 2657W_048_CXR |
| | | 2668W_018_CR |
| | | 2668W_024_CR |
| | | 2668W_036_CR |
| | | 2668W_048_CR |
| | | 3242G_012_CR |
| | | 3242G_018_CR |
| | | 3242G_024_CR |
| | | 3242G_036_CR |
| | | 3242G_048_CR |
| | | 3257G_012_CR |
| | | 3257G_018_CR |
| | | 3257G_024_CR |
| | | 3257G_036_CR |
| | | 3257G_048_CR |
| | | 3272G_012_CR |
| | | 3272G_018_CR |
| | | 3272G_024_CR |
| | | 3272G_036_CR |
| | | 3272G_048_CR |
| | | 3863H_012_CR |
| | | 3863H_018_CR |
| | | 3863H_024_CR |
| | | 3863H_036_CR |
| | | 3863H_048_CR |
| | | 3890H_018_CR |
| | | 3890H_024_CR |
| | | 3890H_036_CR |
| | | 3890H_048_CR |
| | Maxon | Amax_16_134844 |

| Block | Manufacturer | Component |
|---|---|---|
| | | Amax_16_139820 |
| | | Amax_16_139823 |
| | | Amax_16_220514 |
| | | Amax_16_231379 |
| | | Amax_16_260678 |
| | | Amax_16_304672 |
| | | Amax_16_352815 |
| | | Amax_16_352816 |
| | | Amax_16_352817 |
| | | Amax_16_352823 |
| | | Amax_16_352825 |
| | | Amax_16_352826 |
| | | Amax_16_352827 |
| | | Amax_16_352828 |
| | | Amax_16_352829 |
| | | Amax_16_352830 |
| | | Amax_16_352831 |
| | | Amax_16_352832 |
| | | Amax_16_352833 |
| | | Amax_19_139828 |
| | | Amax_19_202411 |
| | | Amax_19_202412 |
| | | Amax_19_233453 |
| | | Amax_19_235373 |
| | | Amax_19_238388 |
| | | Amax_19_267427 |
| | | Amax_19_352922 |
| | | Amax_19_352923 |
| | | Amax_22_134267 |
| | | Amax_22_134666 |
| | | Amax_22_137255 |
| | | Amax_22_137476 |
| | | Amax_22_139838 |
| | | Amax_22_202413 |
| | | Amax_22_218799 |
| | | Amax_22_238798 |

| Block | Manufacturer | Component |
|---|---|---|
| | | Amax_22_258367 |
| | | Amax_22_267423 |
| | | Amax_22_310003 |
| | | Amax_22_342390 |
| | | Amax_26_110181 |
| | | Amax_26_110182 |
| | | Amax_26_110183 |
| | | Amax_26_110184 |
| | | Amax_26_110185 |
| | | Amax_26_110186 |
| | | Amax_26_110187 |
| | | Amax_26_110188 |
| | | Amax_26_110189 |
| | | Amax_26_110190 |
| | | Amax_26_110191 |
| | | DCX_6M_1p5 |
| | | DCX_6M_3 |
| | | DCX_6M_4p5 |
| | | DCX_6M_6 |
| | | DCX_8M_12 |
| | | DCX_8M_1p4 |
| | | DCX_8M_4p2 |
| | | DCX_8M_6 |
| | | DCX_8M_7p2 |
| | | DCX_8M_9 |
| | | DC_max_16_S_12 |
| | | DC_max_16_S_24 |
| | | DC_max_16_S_6 |
| | | DC_max_22_S_12 |
| | | DC_max_22_S_24 |
| | | DC_max_22_S_6 |
| | | DC_max_26_S_12 |
| | | DC_max_26_S_24 |
| | | RE_65_388984_18 |
| | | RE_65_388985_24 |
| | | RE_65_388986_36 |

| Block | Manufacturer | Component |
|---|---|---|
| | | RE_65_388987_48 |
| | | RE_65_388988_60 |
| | | RE_65_388989_70 |
| | | RE_65_388990_70 |
| | | RE_65_388991_70 |
| | Portescap | 08GS61_105C_3 |
| | | 08GS61_105_3 |
| | | 08GS61_107_3 |
| | | 10NS61_107C_5 |
| | | 12G88_210E_1001 |
| | | 12G88_215E_1001 |
| | | 12GS88_208F_1007 |
| | | 12GS88_210E_1001 |
| | | 12GS88_210E_1007 |
| | | 12GS88_215E_1001 |
| | | 13N88_107E_1 |
| | | 13N88_110E_1 |
| | | 13N88_213E_1 |
| | | 16C18_115_67 |
| | | 16C18_204_67 |
| | | 16C18_205_67 |
| | | 16C18_207_67 |
| | | 16C18_210_67 |
| | | 16G88_205E_1 |
| | | 16G88_210E_1 |
| | | 16G88_211E_1 |
| | | 16G88_213E_1 |
| | | 16G88_214E_1 |
| | | 16G88_220P_1 |
| | | 16N78_135_1001 |
| | | 16N78_208E_1001 |
| | | 16N78_210E_1001 |
| | | 16N78_212E_1001 |
| | | 16N78_212P_1001 |
| | | 16N78_214E_1001 |
| | | 16S78_208P_1 |

| Block | Manufacturer | Component |
|---|---|---|
| | | 16S78_210E_1 |
| | | 17N78_207E_1 |
| | | 17N78_208E_1 |
| | | 17N78_210E_1 |
| | | 17N78_216E_1 |
| | | 17S78_208P_1 |
| | | 17S78_209E_1 |
| | | 17S78_210E_1 |
| | | 22N78_208E_1001 |
| | | 22N78_215E_1001 |
| | | 22N78_216E_1001 |
| | | 22N78_311P_1001 |
| | | 22N78_313P_1001 |
| | | 22N78_319P_1001 |
| | | 28DT12_219E_1 |
| | | 28DT12_219P_1 |
| | | 28DT12_222E_1 |
| | | 28DT12_222P_1 |
| | | 28L28_219_29 |
| | | 28L28_410E_29 |
| | | 28L28_413E_29 |
| | | 28L28_416E_29 |
| | | 28LT12_219_49 |
| | | 28LT12_416E_49 |
| | | 30GT2R82_234E_4 |
| | | 30GT2R82_234P_4 |
| | | 35GLT2R82_234E_1 |
| | | 35GLT2R82_326P_1 |
| | | 35GLT2R82_426E_1 |
| | | 35GLT2R82_426P_1 |
| | | 35GLT2R82_426SP_1 |
| | | 35NT2R32_228E_1 |
| | | 35NT2R32_228P_1 |
| | | 35NT2R32_416SP_1 |
| | | 5NT2R82_226E_1 |
| | | 5NT2R82_426E_1 |

| Block | Manufacturer | Component |
|---|---|---|
| | | 5NT2R82_426P_1 |
| | | 5NT2R82_426SP_1 |
| | RS | M9232_12 |
| | | M9232_24 |
| | | M9232_48 |
| | | M9234_12 |
| | | M9234_24 |
| | | M9234_48 |
| | | M9236_12 |
| | | M9236_24 |
| | | M9236_48 |
| | | M9237_12 |
| | | M9237_24 |
| | | M9237_48 |
| | ebmpaps | BCI_42_25_A00 |
| | | BCI_42_25_B00 |
| | | BCI_42_40_A00 |
| | | BCI_42_40_B00 |
| | | BCI_52_30_A00 |
| | | BCI_52_30_B00 |
| | | BCI_52_60_A00 |
| | | BCI_52_60_B00 |
| | | BCI_63_25_A00 |
| | | BCI_63_25_B00 |
| | | BCI_63_55_A00 |
| | | BCI_63_55_B00 |
| PMSM | ABB_BALDOR | BSM132C-8200AA |
| | | BSM33C-5177MHQ |
| | | BSM5ON-133 |
| | | BSM5ON-275 |
| | | BSM63N-133 |
| | | HDS100-0206A |
| | | HDS130-0817B |
| | | HDS130-1829B |
| | | HDS180-2540B |
| | | HDS180-4876B |

| Block | Manufacturer | Component |
|---|---|---|
| | | HDS65-0102A |
| | Allied_Motion | HMP04-002 |
| | | HMP06-007 |
| | | HMP06-015 |
| | | HMP08-028 |
| | | HMP10-056 |
| | | HMP13-055 |
| | Anaheim_Automation | BLK321S |
| | | BLY171D |
| | | BLY171S |
| | | BLY341D |
| | B_R_Automation | 8KSC82 |
| | | 8KSC84 |
| | | 8KSC85 |
| | | 8KSC86 |
| | | 8KSC96 |
| | | 8LSA23 |
| | | 8LSA35 |
| | | 8LSA43 |
| | | 8LSAA2 |
| | | 8LSAA3 |
| | | 8LSAA4 |
| | | 8MSA2S |
| | | 8MSA4L |
| | | 8MSA5S |
| | Electrocraft | RPP23-154V325 |
| | | RPP23-64V325 |
| | | RPP23-64V48 |
| | | RPX2204-V12 |
| | | RPX22-062V24 |
| | Parker_Motors | BE161C |
| | | BE161D |
| | | BE230D |
| | | BE232D |
| | | BE341F |
| | | BE342K |

| Block | Manufacturer | Component |
|---|---|---|
| | | BE343J |
| | | MB562502 |
| | | SMB40 |
| | SEM_Motors | HDM82A8-15S |
| | | HDM82C8-22S |
| | | HDM82E8-30S |
| | | HDM82J8-40S |
| | | HJ116C6-64S |
| | | HJ96C6-44 |
| | | HJ96G6-44 |
| | | HR70A4-22S |
| | Siemens | 1FK7032-2AK7 |
| | | 1FK7034-2AK7 |
| | | 1FK7060-2AC7 |
| | | 1FK7080-2AF7 |
| | | 1FK7086-4SF7 |
| | | 1FK7105-2AF7 |
| | | 1FT7034-5AK7 |
| | | 1FT7042-5AF7 |
| | | 1FT7046-5AH7 |
| | | 1FT7062-5AK7 |
| | | 1FT7108-5AC7 |
| | | 1FT7134-5AB71 |
| | | 1FT7136-5AB71 |
| | Swiss_Mekatronix | SMXN1-010 |
| | | SMXN4-510 |
| | | SMXN7-3000 |
| | | SMXQ1-225 |
| | | SMXQ2-650 |
| | | SMXQ3-950 |
| | | SMXQ4 |
| | | SMXQ5-280 |
| | | SMXQ6-600 |
| | Teknic_Motors | 2310P |
| Solar Cell | Amerisolar | AS_6M30_HC_320W |
| | | AS_6M_360W_PERC |

| Block | Manufacturer | Component |
|---|---|---|
| | | AS_6P30_HC_280W |
| | | AS_6P_HC_340W |
| | | AS_6P_HC_360W |
| | | ThinFilm_AS_100W |
| | | ThinFilm_AS_85W |
| | Astronergy | CHSM5001T_110W |
| | AzurSpace | 3C44_30sqmm |
| | | 3C44_9sqmm |
| | Calyxo | CX1_80_3 |
| | | CX3_85_3 |
| | | CX4_100_3 |
| | | CX4_102_3 |
| | | CX4_105_3 |
| | | CX4_107_3 |
| | | CX4_110_3 |
| | | CX4_97_3 |
| | CanadianSolar | CS1H330MS |
| | | CS1H345MS |
| | | CS3K_315P_AG |
| | | CS3W_420P |
| | | CS3W_435MS |
| | | CS6K270P |
| | | CS6K280P |
| | FirstSolar | FS_3100_PLUS |
| | | FS_390_PLUS |
| | | FS_395_PLUS |
| | | FS_397_PLUS |
| | | FS_4100A |
| | | FS_4102A |
| | | FS_4105A |
| | | FS_4112_3 |
| | | FS_4120_3 |
| | | FS_4122_3 |
| | | FS_492A |
| | | FS_495A |
| | | FS_497A |

| Block | Manufacturer | Component |
|---|---|---|
| | | FS_6420A |
| | | FS_6425A |
| | | FS_6430A |
| | | FS_6435A |
| | | FS_6440A |
| | | FS_6445A |
| | | FS_6450A |
| | GCL | GCL_M3_72GD_420W_Front |
| | | GCL_M3_72GD_420W_Rear |
| | | GCL_M3_72H_420W |
| | | GCL_M6_72H_415W |
| | | GCL_M8_72GDF_465W_Front |
| | | GCL_M8_72GDF_465W_Rear |
| | | GCL_MS_60H_335W |
| | | GCL_P6_60H_270W |
| | | GCL_P6_60H_305W |
| | | GCL_P6_72H_325W |
| | | GCL_P6_72H_360W |
| | Hanwha_Q_CELLS | QPEAK_DUO_BLK_G5_310 |
| | | QPEAK_DUO_BLK_G5_315 |
| | | QPEAK_DUO_BLK_G5_320 |
| | | QPEAK_DUO_BLK_G5_325 |
| | | QPEAK_DUO_BLK_G6_330 |
| | | QPEAK_DUO_BLK_G6_335 |
| | | QPEAK_DUO_BLK_G6_340 |
| | | QPEAK_DUO_BLK_G6_345 |
| | | QPEAK_DUO_BLK_G7_315 |
| | | QPEAK_DUO_BLK_G7_320 |
| | | QPEAK_DUO_BLK_G7_325 |
| | | QPEAK_DUO_BLK_G8_335 |
| | | QPEAK_DUO_BLK_G8_340 |
| | | QPEAK_DUO_BLK_G8_345 |
| | | QPEAK_DUO_BLK_G8_350 |
| | | QPEAK_DUO_G5_315 |
| | | QPEAK_DUO_G5_320 |
| | | QPEAK_DUO_G5_325 |

| Block | Manufacturer | Component |
|---|---|---|
| | | QPEAK_DUO_G5_330 |
| | | QPEAK_DUO_G5_335 |
| | | QPEAK_DUO_G6_340 |
| | | QPEAK_DUO_G6_345 |
| | | QPEAK_DUO_G6_350 |
| | | QPEAK_DUO_G6_355 |
| | | QPEAK_DUO_G7_325 |
| | | QPEAK_DUO_G7_330 |
| | | QPEAK_DUO_G7_335 |
| | | QPEAK_DUO_G8_345 |
| | | QPEAK_DUO_G8_350 |
| | | QPEAK_DUO_G8_355 |
| | | QPEAK_DUO_G8_360 |
| | | QPEAK_DUO_L_G5_380 |
| | | QPEAK_DUO_L_G5_385 |
| | | QPEAK_DUO_L_G5_390 |
| | | QPEAK_DUO_L_G5_395 |
| | | QPEAK_DUO_L_G5_400 |
| | | QPEAK_DUO_L_G6_405 |
| | | QPEAK_DUO_L_G6_410 |
| | | QPEAK_DUO_L_G6_415 |
| | | QPEAK_DUO_L_G6_420 |
| | | QPEAK_DUO_L_G6_425 |
| | | QPEAK_DUO_L_G7_385 |
| | | QPEAK_DUO_L_G7_390 |
| | | QPEAK_DUO_L_G7_395 |
| | | QPEAK_DUO_L_G7_400 |
| | | QPEAK_DUO_L_G7_405 |
| | | QPEAK_DUO_L_G8_415 |
| | | QPEAK_DUO_L_G8_420 |
| | | QPEAK_DUO_L_G8_425 |
| | | QPEAK_DUO_L_G8_430 |
| | Hyundai | HiE_S340SG |
| | | HiE_S345SG |
| | | HiE_S350SG |
| | | HiE_S390VG |

| Block | Manufacturer | Component |
|---|---|---|
| | | HiE_S395VG |
| | | HiE_S400VG |
| | | HiE_S405VG |
| | JA_Solar | JAM60S10_330PR |
| | | JAM60S20_380MR |
| | | JAM72S10_400PR |
| | | JAM72S20_455MR |
| | | JAP72S10_330_350SC |
| | JinKOSolar | Cheetah_JKM325M_60HB |
| | | Cheetah_JKM375M_66H |
| | | Cheetah_JKM390M_72H |
| | | Cheetah_JKM430M_78H |
| | | Swan_JKM390M_72H_TV |
| | | Swan_JKM395M_72H_BDVP |
| | | Tiger_JKM385M_6RL3 |
| | | Tiger_JKM395M_6RL3 |
| | | Tiger_JKM450M_7RL3_TV |
| | | Tiger_JKM460M_7RL3 |
| | | Tiger_JKM465N_7RL3 |
| | Kaneka | U_EA110 |
| | | U_EA120 |
| | | U_SA100 |
| | | U_SA110 |
| | LONGi | LR4_60HPH_365M |
| | | LR4_72HPH_440M |
| | Mitsubishi_Electric | PV_MLE270HD |
| | | PV_MLE275HD2 |
| | | PV_MLE280HD2 |
| | | PV_MLU250HC |
| | | PV_MLU255HC |
| | Philadelphia_Solar | PS_M60_305 |
| | | PS_M60_310 |
| | | PS_M60_315 |
| | | PS_M60_320 |
| | | PS_M60_325 |
| | | PS_M60_330 |

| Block | Manufacturer | Component |
|---|---|---|
| | | PS_M72_370 |
| | | PS_M72_375 |
| | | PS_M72_380 |
| | | PS_M72_385 |
| | | PS_M72_390 |
| | | PS_M72_395 |
| | | PS_M72_400 |
| | | PS_M72_405 |
| | | PS_M72_HC_425 |
| | | PS_M72_HC_430 |
| | | PS_M72_HC_435 |
| | | PS_M72_HC_440 |
| | | PS_M72_HC_445 |
| | | PS_M72_HC_450 |
| | | PS_P60_275 |
| | | PS_P60_280 |
| | | PS_P60_285 |
| | | PS_P60_290 |
| | | PS_P72_330 |
| | | PS_P72_335 |
| | | PS_P72_340 |
| | | PS_P72_345 |
| | REC | REC355TP4_Black |
| | | REC360AA |
| | | REC360TP4_Black |
| | | REC365A |
| | | REC365TP4_Black |
| | | REC370A |
| | | REC370TP4_Black |
| | | REC375A |
| | | REC375TP4 |
| | | REC380A |
| | | REC385AA_Pure |
| | | REC390AA_Pure |
| | | REC395AA_Pure |
| | | REC400AA_Pure |

| Block | Manufacturer | Component |
|---|---|---|
| | | REC405AA_Pure |
| | | REC430AA_72 |
| | | REC435AA_72 |
| | | REC440AA_72 |
| | | REC445AA_72 |
| | | REC450AA_72 |
| | Tata_Power_Solar | TP300 |
| | | TP305 |
| | | TP310 |
| | | TP315 |
| | | TP320 |
| | | TP325 |
| | | TP330 |
| | Telesun | TP6L60M_H_360 |
| | | TP6L60M_H_365 |
| | | TP6L60M_H_370 |
| | | TP6L60M_H_375 |
| | | TP6L60M_H_380 |
| | | TP6L72M_H_435 |
| | | TP6L72M_H_440 |
| | | TP6L72M_H_445 |
| | | TP6L72M_H_450 |
| | | TP6L72M_H_455 |
| | | TP7F72M_H_530 |
| | | TP7F72M_H_535 |
| | | TP7F72M_H_540 |
| | | TP7F72M_H_545 |
| | | TP7F72M_H_550 |
| | | TP7F78M_H_570 |
| | | TP7F78M_H_575 |
| | | TP7F78M_H_580 |
| | | TP7F78M_H_585 |
| | | TP7F78M_H_590 |
| | TrinaSolar | Allmax_TSM_DD05A_60C_280W |
| | | Allmax_TSM_DD05A_60C_315W |
| | | Allmax_TSM_DD06H_120C_320W |

| Block | Manufacturer | Component |
|---|---|---|
| | | Allmax_TSM_DD06H_120C_335 W |
| | | Duomax_TSM_DEG14_72C_340 W |
| | | Duomax_TSM_DEG14_72C_355 W |
| | | Duomax_TSM_DEG14_72C_375 W |
| | | Duomax_TSM_DEG15H_144C_380W |
| | | Duomax_TSM_DEG15H_144C_410W |
| | | Duomax_TSM_DEG5_60C_280W |
| | | Duomax_TSM_DEG5_60C_315W |
| | | Residential_TSM_DD06H_120C_310 |
| | | Residential_TSM_DD06H_120C_335 |
| | | Tallmax_TSM_DE14H_144C_345 W |
| | | Tallmax_TSM_DE14H_144C_395 W |
| | risensolar | RSM120_6_275PH_295P |
| | | RSM144_6_335P_355P |
| | | RSM72_6_325P_345P |
| | | RSM_72_6_370P |
| | | RSM_72_6_385P |
| Stepper Motor | Minebea | 08PM-K139BSTDCN |
| | | 17PM-KA053B |
| | | 17PM-KA39B |
| | | 23KY-K748B |
| | | 29SM-K149B |
| | | 29SM-K379B |
| | Precision_Motors | B-48-42-B17 |
| | | B-48-42-B18 |
| | Sonceboz | 6405 |
| SPICE-Imported MOSFET | Infineon OptiMOS6 40V | IAUC120N04S6N009 |
| | | IAUC120N04S6N010 |
| | | IAUC120N04S6N013 |

| Block | Manufacturer | Component |
|---|---|---|
| | | IAUC100N04S6N015 |
| | | IAUC100N04S6N022 |
| | | IAUC100N04S6N028 |
| | | IAUC80N04S6N036 |
| | | IAUC60N04S6N036 |
| | | IAUC120N04S6L008 |
| | | IAUC120N04S6L009 |
| | | IAUC120N04S6L012 |
| | | IAUC100N04S6L014 |
| | | IAUC100N04S6L020 |
| | | IAUC100N04S6L025 |
| | | IAUC80N04S6L032 |
| | | IAUC60N04S6L039 |
| | Infineon OptiMOS5 80V | IPB015N08N5 |
| | | IPB017N08N5 |
| | | IPP020N08N5 |
| | | IPT012N08N5 |
| | | IPB020N08N5 |
| | | IPP023N08N5 |
| | | IPB024N08N5 |
| | | IPP027N08N5 |
| | | IPB031N08N5 |
| | | IPP034N08N5 |
| | | IPB049N08N5 |
| | | IPP052N08N5 |
| | | IAUT300N08S5N012 |
| | | IAUT165N08S5N029 |
| | | IAUT300N08S5N014 |
| | | IAUT240N08S5N019 |
| | | IAUT200N08S5N023 |
| | | IAUS300N08S5N013T |
| | | IAUS300N08S5N015T |
| | | IAUC100N08S5N031 |
| | | IAUC100N08S5N043 |
| | | IAUC70N08S5N074 |
| | Infineon OptiMOS4 60V | IPB180N06S4_H1 |

| Block | Manufacturer | Component |
|---|---|---|
| | | IPB120N06S4_H1 |
| | | IPI120N06S4_H1 |
| | | IPP120N06S4_H1 |
| | | IPB120N06S4_02 |
| | | IPI120N06S4_02 |
| | | IPP120N06S4_02 |
| | | IPB120N06S4_03 |
| | | IPI120N06S4_03 |
| | | IPP120N06S4_03 |
| | | IPB90N06S4_04 |
| | | IPD90N06S4_04 |
| | | IPD100N06S4_03 |
| | | IPI90N06S4_04 |
| | | IPP90N06S4_04 |
| | | IPB80N06S4_05 |
| | | IPD90N06S4_05 |
| | | IPI80N06S4_05 |
| | | IPP80N06S4_05 |
| | | IPB80N06S4_07 |
| | | IPD90N06S4_07 |
| | | IPI80N06S4_07 |
| | | IPP80N06S4_07 |
| | | IPB45N06S4_09 |
| | | IPD50N06S4_09 |
| | | IPI45N06S4_09 |
| | | IPP45N06S4_09 |
| | | IPG20N06S4_15 |
| | | IPB90N06S4L_04 |
| | | IPD90N06S4L_03 |
| | | IPI90N06S4L_04 |
| | | IPP90N06S4L_04 |
| | | IPB80N06S4L_05 |
| | | IPD90N06S4L_05 |
| | | IPI80N06S4L_05 |
| | | IPP80N06S4L_05 |
| | | IPB80N06S4L_07 |

| Block | Manufacturer | Component |
|---|---|---|
| | | IPD90N06S4L_06 |
| | | IPI80N06S4L_07 |
| | | IPP80N06S4L_07 |
| | | IPB45N06S4L_08 |
| | | IPD50N06S4L_08 |
| | | IPI45N06S4L_08 |
| | | IPP45N06S4L_08 |
| | | IPD50N06S4L_12 |
| | | IPD30N06S4L_23 |
| | | IPD25N06S4L_30 |
| | | IPG20N06S4L_11 |
| | | IPG20N06S4L_14 |
| | | IPG20N06S4L_26 |
| | Infineon OptiMOST 120V | IPB100N12S3_05 |
| | | IPI100N12S3_05 |
| | | IPP100N12S3_05 |
| | | IPB70N12S3_11 |
| | | IPI70N12S3_11 |
| | | IPP70N12S3_11 |
| | | IPD70N12S3_11 |
| | | IPD70N12S3L_12 |
| | | IPB70N12S3L_12 |
| | | IPI70N12S3L_12 |
| | | IPP70N12S3L_12 |
| | | IPB50N12S3L_15 |
| | | IPD50N12S3L_15 |
| | | IPI50N12S3L_15 |
| | | IPP50N12S3L_15 |
| | | IPB35N12S3L_26 |
| | | IPD35N12S3L_24 |
| | | IPD30N12S3L_31 |

## Additional Parameterizations

To learn about additional ways to parameterize and validate your model, see the following examples.

| Task | Learn More |
| --- | --- |
| Tune parameters against external data. | “Solar Cell Parameter Extraction from Data” on page 8-644<br><br>“Import IPMSM Flux Linkage Data from ANSYS Maxwell” on page 8-470 |
| Build an equivalent model of a SPICE netlist. | “Parameterize an Exponential Diode from SPICE Netlist” on page 3-21<br><br>“Converting a SPICE Netlist to Simscape Blocks” on page 3-56 |
| Specify machine parameters using standard or fundamental parameterization methods. | “Machine Parameterization” on page 4-2 |
| Test harness and validation. | “Pre-Parameterized Stepper Motor Block Validation” on page 8-403<br><br>“PMSM Iron Losses” on page 8-483<br><br>“PMSM Parameterization from Datasheet” on page 8-864<br><br>“Stepper Motor Pull-in Characteristics” on page 8-413 |

## Request a Pre-Parameterization

To request a new component pre-parameterization, visit https://www.mathworks.com/products/simscape/model-request.html.

## See Also

Battery (Table-Based) | BLDC | PMSM | Solar Cell | SPICE-Imported MOSFET | Stepper Motor | `subcircuit2ssc`

# Modeling Machines


- “Machine Parameterization” on page 4-2
- “Per-Unit Conversion for Machine Parameters” on page 4-3
- “Machine Plotting and Display Options” on page 4-4
- “Initialize Synchronous Machines and Controllers” on page 4-6

# Machine Parameterization

In Simscape Electrical software, induction machines are parameterized using fundamental parameters. Each synchronous machine is parameterized using standard or fundamental parameters.

Machine fundamental parameters include the values of inductances and resistances of the stator and rotor *d*- and *q*-axis equivalent circuits. These parameters fully specify the electrical characteristics of the machine, but you cannot determine them directly from machine test responses. Hence, it is more common to parameterize a synchronous machine using a standard parameter set. You can obtain the standard parameters by observing responses at the machine terminals with suitable tests scenarios.

You can tell the parameter set a block uses because the block name includes the parameter set name, e.g. Induction Machine Squirrel Cage. The parameters you can set in the block dialog box correspond to the parameterization type.

If a machine block has standard and fundamental variants, base your block choice on the parameters you are most familiar with or you have available. Standard block variants use classical equations to convert standard parameter values that you enter to fundamental parameter values for use at run time.

If a machine block has an SI and a per-unit variant, base your block choice on the parameters you have available. For machine blocks that are SI variants, you enter the number of pole pairs and the SI values for the nominal voltage, power, and frequency on the main tab of the dialog box. You also enter SI values for the resistance and reactance parameters on the impedance tab, and for the magnetic flux linkage parameters on the initial condition tab. The block uses classical equations to calculate per-unit base values from the parameters on the main tab. It expresses the resistance, inductance, and magnetic flux linkage parameters as per-unit ratios of the SI values (resistance, reactance, and magnetic flux linkage) and the base values for use at run time.

The field circuit and rotational ports of machine blocks use SI units. However, the pu measurement port of machine blocks outputs a vector of physical signals in per-unit.

## See Also

## More About

- “Per-Unit System of Units” on page 1-9
- “Per-Unit Conversion for Machine Parameters” on page 4-3

# Per-Unit Conversion for Machine Parameters

| In this section... |
| --- |
| “Impedance Conversion Equations” on page 4-3 |
| “Magnetic Flux Linkage Conversion Equations” on page 4-3 |

## Impedance Conversion Equations

For machine impedance parameters (resistance, inductance, and reactance), the relationships between SI and per-unit values are defined by these equations:

$$R = \frac{R_{(SI)}}{R_{base}}$$

$$L = X = \frac{X_{(SI)}}{X_{base}}$$

where:

- $R_{(SI)}$ is the resistance, expressed in Ω.
- $R_{base}$ is the per-unit base resistance, expressed in Ω.
- $R$ is the per-unit resistance.
- $X_{(SI)}$ is the reactance, expressed in Ω.
- $X_{base}$ is the per-unit base reactance, expressed in Ω.
- $X$ is the per-unit reactance.
- $L$ is the per-unit inductance.

## Magnetic Flux Linkage Conversion Equations

For machine magnetic flux linkage parameters, the relationship between SI and per-unit values is defined by

$$\psi = \frac{\Psi_{(SI)}}{\Psi_{base}}$$

where:

- $\psi_{(SI)}$ is the magnetic flux linkage, expressed in Wb.
- $\psi_{base}$ is the per-unit base magnetic flux linkage, expressed in Wb.
- $\psi$ is the per-unit magnetic flux linkage.

## See Also

## More About

- “Per-Unit System of Units” on page 1-9

# Machine Plotting and Display Options

Use the **Electrical** menu on the block context menu to perform plotting and display actions for certain blocks in the Simscape ElectricalElectromechanical sublibrary. For example, you can plot torque versus speed for the Induction Machine Wound Rotor block, either in SI or per-unit units.

Using other options on the **Electrical** menu, you can display values in per-unit or display base parameter values in the MATLAB Command Window. These options enable you to initialize and tune your three-phase machine quickly.

## Asynchronous Machine Options

The context menus of certain asynchronous machine blocks contain some or all of these options:

- **Display Base Values** — Displays the machine per-unit base values in the MATLAB Command Window.
- **Plot Torque Speed (SI)** — Plots torque versus speed, both measured in SI units, in a MATLAB figure window using the present machine parameters.
- **Plot Torque Speed (pu)** — Plots torque versus speed, both measured in per-unit, in a MATLAB figure window using the present machine parameters.
- **Plot Open-Circuit Saturation** — Plots terminal voltage versus no-load stator current, both in per-unit, or, for SI blocks, in V and A, respectively, in a MATLAB figure window. The plot contains three traces:
  - Unsaturated
  - Saturated
  - Derived
- **Plot Saturation Factor** — Plots saturation factor applied to magnetic inductance versus magnetic flux linkage in per-unit, or for SI blocks, in Wb, in a MATLAB figure window.
- **Plot Saturated Inductance** — Plots magnetizing inductance versus per-unit magnetic flux linkage, both in per-unit, or, for SI blocks, in H and Wb, respectively, in a MATLAB figure window.

## Induction Machine Options

The context menus of certain induction machine blocks contain some or all of these options for displaying the associated values in the MATLAB Command Window:

- **Display Base Values** — Displays the machine per-unit base values in the MATLAB Command Window
- **Display Associated Base Values** — Displays the associated per-unit base values in the MATLAB Command Window.
- **Display Associated Initial Conditions** — Displays the associated intitial condition values in the MATLAB Command Window.
- **Plot Open-Circuit Saturation (pu)** — Plots air-gap voltage, $V_{ag}$, versus field current, $i_{fd}$, both measured in per-unit, in a MATLAB figure window. The plot contains three traces:
  - Unsaturated — **Stator d-axis mutual inductance (unsaturated), Ladu** you specify
  - Saturated — **Per-unit open-circuit lookup table (Vag versus ifd)** you specify

  - Derived — Open-circuit lookup table (per-unit) derived from the **Per-unit open-circuit lookup table (Vag versus ifd)** you specify. This data is used to calculate the saturation factor, $K_s$, versus magnetic flux linkage, $\psi_{at}$, characteristic.
- **Plot Saturation Factor (pu)** — Plots saturation factor, $K_s$, versus magnetic flux linkage, $\psi_{at}$, both measured in per-unit, in a MATLAB figure window using the present machine parameters. This value is derived from parameters you specify:
  - **Stator d-axis mutual inductance (unsaturated), Ladu**
  - **Per-unit field current saturation data, ifd**
  - **Per-unit air-gap voltage saturation data, Vag**

## Machine Inertia Block Options

For the Machine Inertia block, you can display the inertia parameters and base values using the **Electrical** menu on the block context menu. The block displays parameter values in the MATLAB Command Window.

# Initialize Synchronous Machines and Controllers

In Simscape Electrical software, you can specify steady-state power and voltage values for a synchronous machine. Based on the values you specify, the machine block calculates the initial field circuit and rotational input values required to achieve this steady state. Starting a machine at steady state prevents undesired transient effects in your simulation.

1. Calculate the required power and voltage characteristics of your load circuit.
2. In the **Initial Conditions** tab of the dialog box, set **Specify initialization by** to `Electrical power and voltage output`.
3. Enter the required power and voltage values and click **OK**.
4. Right-click the machine block and select **Electrical > Display Associated Initial Conditions**.

   Simscape Electrical Power Systems calculates the field circuit and rotational port values required to start the machine in steady state and displays them in the MATLAB Command Window.
5. Use these values to input parameters to the blocks connected to the field circuit and rotational ports of the synchronous machine.

**Note** If you set **Specify initialization by** to `Mechanical and magnetic states`, Simulink does not calculate the associated initial conditions fro the machine.

# Customization


- “Build Custom Blocks Using the Three-Phase Electrical Domain” on page 5-2
- “Custom Synchronous Machine” on page 5-4

# Build Custom Blocks Using the Three-Phase Electrical Domain

Simscape Foundation domains include a three-phase electrical domain. You can use this domain to develop your own custom three-phase blocks using Simscape language. To refer to this domain in your custom component declarations, use the following syntax:

```
foundation.electrical.three_phase
```

Additionally, the ThreePhaseExamples library, included in the Simscape Electrical product examples, contains a Fundamental library, a Transforms library, and the following custom three-phase components:

- Permanent Magnet Synchronous Motor
- Synchronous Machine
- Synchronous Machine (simplified)
- Zigzag Transformer

You can use these simplified example models to write your own custom component files.

To open the custom library, at the MATLAB command prompt, type `ThreePhaseExamples_lib`. Double-click any block in the library to open its dialog box, and then click the **Source code** link in the block dialog box to open the Simscape source file for this block in the MATLAB Editor.

To customize the block for your application, edit the source file and save it under another name.

For example, you can create a folder called `+MyMachines` and save the source files for your customized machines in this folder. Create this folder in your working directory, or in another directory that is on the MATLAB path. Running the `ssc_build` command on this package generates the `MyMachines_lib` library model. This library contains all your custom machine blocks and is located in the same directory where you have created the `+MyMachines` folder. Open the `MyMachines_lib` library by double-clicking it or by typing its name at the MATLAB command prompt.

For more information on packaging and deploying Simscape component files, see “Building Custom Block Libraries”.

Things to keep in mind when writing component files:

- If you create a custom component by modifying an existing one, do not forget to change the name of the component and the name of the resulting block.
- The component name must be the same as the name of the Simscape file. For example, if you plan to save your component in a file called `MyComponent.ssc`, change the declaration line in the file:

  ```
  component MyComponent
  ```
- The comment line immediately following the component declaration (that is, the first line beginning with the % character) defines the name of the block, as it appears in the custom library next to the block icon and at the top of the block dialog box. If you do not specify this comment, then the component name serves as the block name. The block name must be unique within the subpackage (sublibrary) where it resides.
- Additional comments, below the line specifying the block name, are interpreted as the block description. You do not have to modify them when copying an existing file, but if you change the way the component works, it makes sense to reflect the change in the block description. The block description is for informational purposes only.

- When modifying component equations, if you introduce additional terms, make sure to add the appropriate variables or parameters to the component declaration section. For example, if you add zero-sequence dynamics to the component equations, declare an additional parameter for stator zero-sequence inductance, L0, and an additional variable for the initial stator zero-sequence magnetic flux linkage.

The “Custom Synchronous Machine” on page 5-4 tutorial shows how you can modify the Synchronous Machine component file and customize it for use in your applications. For more information on writing customized component files, see “Custom Components”.

## See Also

## More About

- “Custom Synchronous Machine” on page 5-4
- “Custom Components”
- “Three-Phase Electrical Domain”

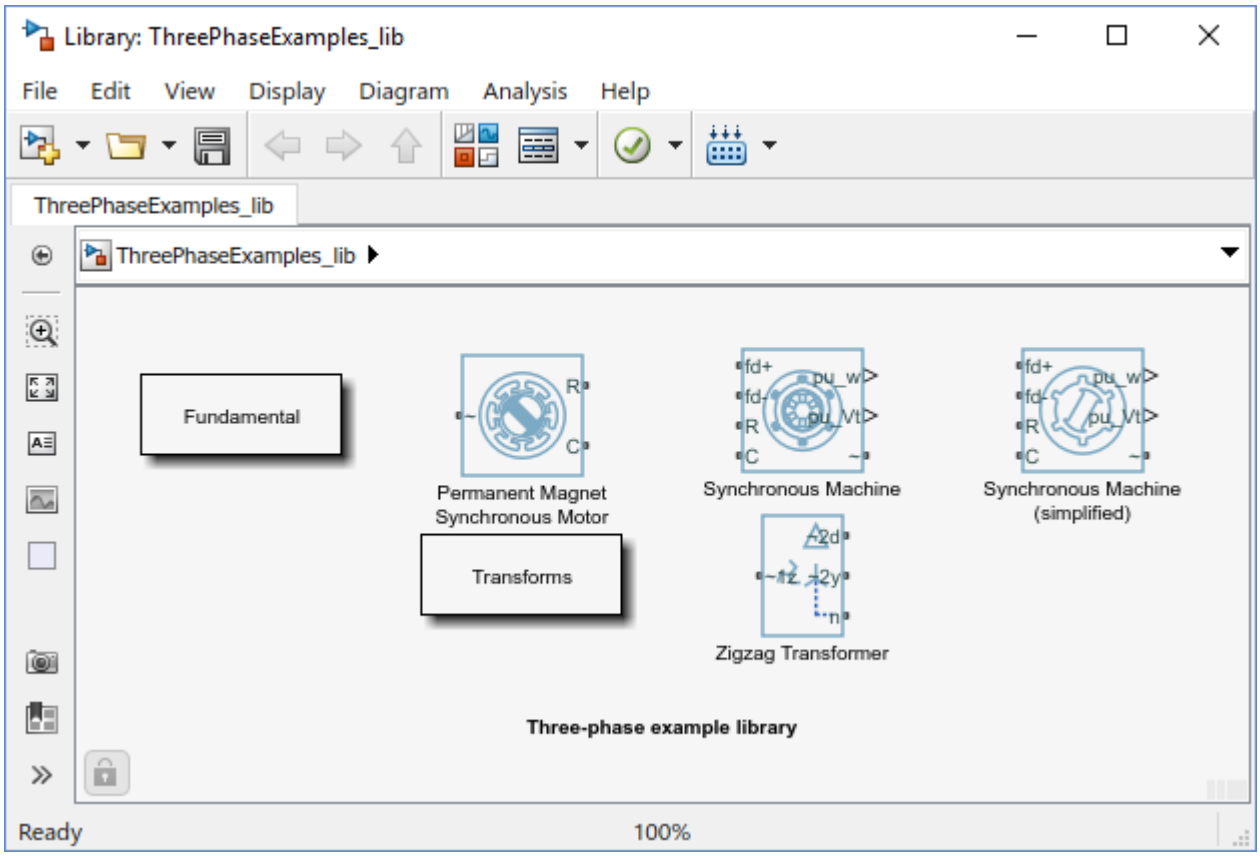
# Custom Synchronous Machine

The ThreePhaseExamples library, included in the product examples, contains simplified example models that you can use to write your own machine and transformer component files. The Synchronous Machine component in the ThreePhaseExamples library is similar to the Synchronous Machine Round Rotor block, but its equations have been simplified to omit zero-sequence dynamics. The Synchronous Machine block is therefore suitable for balanced operation only.

This example shows how you can further simplify the component file and make a custom machine block that does not account for the stator rate of change of flux.

1. In your working directory, create a folder called `+MyMachines`. This folder will contain the source files for your customized machines.
2. To open the library of simplified component examples, at the MATLAB command prompt, type:

   ```
   ThreePhaseExamples_lib
   ```




3. Double-click the Synchronous Machine block.
4. In the block dialog box, click the **Source code** link.

   The Simscape source file for this block opens in the MATLAB Editor.
5. Change the name of the component, the name of the block, and the block description by replacing these lines of the file:

   ```
   component sm
   % Synchronous Machine :1.5
   ```

```
% Synchronous machine (SM) with a round rotor parameterized
% using fundamental per-unit parameters. The defining equations are
% simplified by omitting the zero-sequence dynamics: the model is suitable
% for balanced operation.
% The model contains effect of rate of change of magnetic flux linkages
% on stator voltages, effect of speed variation on stator voltages, one
% damper winding on the d-axis and two damper windings on the q-axis.

% Copyright 2012-2018 The MathWorks, Inc.
```

with:

```
component sm1
% Simplified Synchronous Machine
% This synchronous machine does not include the stator d.psi/dt terms.
```

6 To remove the stator rate of change of flux terms, scroll down to the `equations` section and modify the stator voltage equations from:

```
% Per unit stator voltage equations
pu_ed == oneOverOmega*pu_psid.der - pu_psiq*pu_velocity - Ra*pu_id;
pu_eq == oneOverOmega*pu_psiq.der + pu_psid*pu_velocity - Ra*pu_iq;
```

to:

```
% Per unit stator voltage equations
pu_ed == -pu_psiq*pu_velocity - Ra*pu_id;
pu_eq ==  pu_psid*pu_velocity - Ra*pu_iq;
```

7 Save the file in the `+MyMachines` folder as `sm1.ssc`. The name of the Simscape file must match the component name.

8 To generate the custom library containing the new block, at the MATLAB command prompt, type:

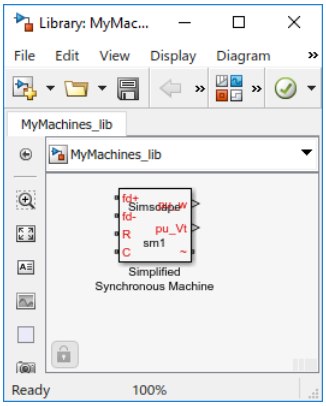
```
ssc_build(MyMachines)
```

This command generates the `MyMachines_lib` library model in your working directory.

9 To open the custom library, at the MATLAB command prompt, type:

```
MyMachines_lib
```

The library contains the Simplified Synchronous Machine block, which you can now use in your models.




## See Also

`ssc_build`

## More About

- “Build Custom Blocks Using the Three-Phase Electrical Domain” on page 5-2
- “Custom Components”
- “Customizing the Block Name and Appearance”
- “Component Equations”

# Control

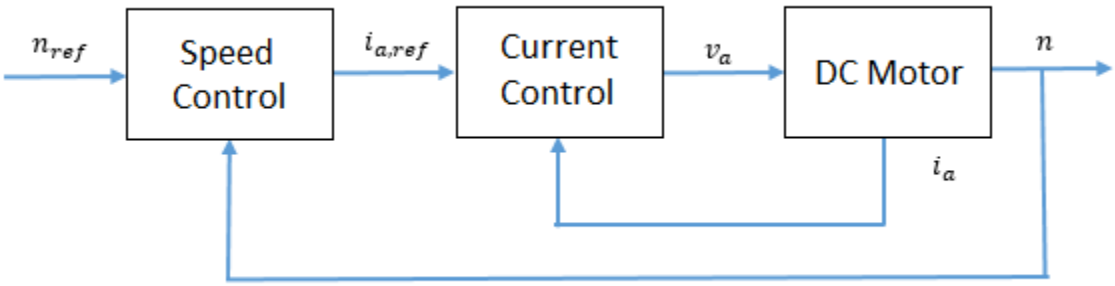
# Tune an Electric Drive

| In this section... |
| --- |
| “Cascade Control Structure” on page 6-2 |
| “Equations for PI Tuning Using the Pole Placement Method” on page 6-2 |
| “Equations for DC Motor Controller Tuning” on page 6-4 |
| “Tune the Electric Drive in the Example Model” on page 6-6 |

This example shows how to tune an electric drive using a cascade control structure.

## Cascade Control Structure

The figure shows a feedback control loop that uses a cascade control structure. The outer speed-control loop is slower acting than the inner current-control loop.




## Equations for PI Tuning Using the Pole Placement Method

To satisfy the required control performance for a simple discrete plant model, $G_f(z^{-1})$, use a closed loop PI control system $G_{PI}(z^{-1})$. The transient performance can be expressed in terms of the overshoot. The overshoot decreases relative to the damping factor:

$$\sigma = e^{\frac{-\pi\xi}{\sqrt{1-\xi^2}}}$$

where,

- $\sigma$ is overshoot.
- $\xi$ is the damping factor.

The response time, $t_r$, depends on the damping and the natural frequency, $\omega_n$, such that:

- If $\xi < 0.7$,

  $$t_r \cong \frac{4}{\omega_n \xi}.$$

- If $\xi \geq 0.7$,

$$t_r \cong \frac{6\xi}{\omega_n}.$$

The general workflow for designing a PI controller for a first-order system is:

1. Discretize the plant model using the zero-order hold (ZOH) discretization method. That is, given that the first-order equation representing the plant is

   $$G(s) = \frac{K_m}{T_m s + 1},$$

   where,

   - $K_m$ is the first-order gain.
   - $T_m$ is time constant of the first-order system.

   Setting

   $$s = \frac{1 - z^{-1}}{z^{-1} T_s},$$

   yields the discrete plant model,

   $$G(z^{-1}) = \frac{K_m\left(\frac{T_s}{T_m}\right)z^{-1}}{1 + \left(\frac{T_s - T_m}{T_m}\right)z^{-1}} = \frac{b_1 z^{-1}}{1 + a_1 z^{-1}},$$

   where$T_s$ is sample time for the discrete-time controller.
2. Write a discrete-time representation for the PI controller using the same transform. For

   $$G_{PI}(s) = K_P + K_I\left(\frac{1}{s}\right),$$

   setting

   $$s = \frac{1 - z^{-1}}{z^{-1} T_s},$$

   yields the discrete controller model,

   $$G_{PI}(z^{-1}) = \frac{K_P + (K_I T_s - K_P)z^{-1}}{1 - z^{-1}} = \frac{q_0 + q_1 z^{-1}}{1 - z^{-1}}.$$

   Combining the discrete equations for the plant and the controller yields the closed loop transfer function for the system,

   $$G_0(z^{-1}) = \frac{q_0 b_1 z^{-1} + q_1 b_1 z^{-2}}{1 + (a_1 - 1 + q_0 b_1)z^{-1} + (-a_1 + q_1 b_1)z^{-2}},$$

   The denominator of the transfer function is the characteristic polynomial. That is,

   $$P_{c0}(z^{-1}) = 1 + (a_1 - 1 + q_0 b_1)z^{-1} + (-a_1 + q_1 b_1)z^{-2}.$$

3 The characteristic polynomial for achieving the required performance is defined as

$$P_{cd}(z^{-1}) = 1 + \alpha_1 z^{-1} + \alpha_2 z^{-2},$$

where,

- $\alpha_1 = -2e^{-\xi\omega_n T_s}\cos\left(\omega_n T_s\sqrt{1-\xi^2}\right)$.
- $\alpha_2 = e^{-2\xi\omega_n T_s}$.

4 To determine the controller parameters, set the characteristic polynomial for the system equal to the characteristic polynomial for the required performance. If

$$P_{c0}(z^{-1}) = P_{cd}(z^{-1}),$$

then

$$\alpha_1 = a_1 - 1 + q_0 b_1$$

and

$$\alpha_2 = -a_1 + q_1 b_1 .$$

Solving for $q_0$ and $q_1$ yields

$$q_0 = \frac{\alpha_1 - a_1 + 1}{b_1}$$

and

$$q_1 = \frac{\alpha_2 + a_1}{b_1}.$$

Therefore, the general equations for the proportional and integral control parameters for the first-order system are

$$K_P = q_0$$

and

$$K_I = \frac{q_1 + K_p}{T_s}.$$

## Equations for DC Motor Controller Tuning

Assuming that, for the system in the example model, $K_b = K_t$, the simplified mathematical equations for voltage and torque of the DC motor are

$$v_a = L_a\frac{di_a}{dt} + R_a i_a + K_b\omega$$

and

$$T_e = J_m\frac{d\omega}{dt} + B_m\omega + T_{load} = K_b i_a,$$

where:

- $v_a$ is the armature voltage.
- $i_a$ is the armature current.
- $L_a$ is the armature inductance.
- $R_a$ is the armature resistance.
- $\omega$ is the rotor angular velocity
- $T_e$ is the motor torque.
- $T_{load}$ is the load torque.
- $J_m$ is the rotor moment of inertia.
- $B_m$is the viscous friction coefficient.
- $K_b$ is a constant of proportionality.

To tune the current controller, assume that the model is linear, that is, that the back electromotive force, as represented by $K_b\omega$, is negligible. This assumption allows for an approximation of the plant model using this first-order Laplace equation:

$$G_i(s) = \frac{\frac{1}{R_a}}{\left(\frac{L_a}{R_a}\right)s + 1}.$$

Given the system requirements, you can now solve for $K_P$ and $K_I$. The requirements for the current controller in the example model are:

- Sample time, $T_s$= 1 ms.
- Overshoot, σ = 5%.
- Response time, $t_r$ = 0.11 s.

Therefore, the proportional and integral parameters for the current controller are:

- $K_P$ = 7.7099.
- $K_I$ = 455.1491.

To tune the speed controller, approximate the plant model with a simple model. First assume that the inner loop is much faster than the outer loop. Also assume that there is no steady-state error. These assumptions allow for the use a first-order system by considering a transfer function of 1 for the inner current loop.

To output rotational velocity in revolutions per minute, the transfer function is multiplied by a factor of 30/π. To take as control input the armature current instead of the motor torque, the transfer function is multiplied by the proportionality constant, $K_b$. The resulting approximation for the outer-loop plant model is

$$G_n(s) = \frac{\frac{30K_b}{\pi B_m}}{\left(\frac{J_m}{B_m}\right)s + 1}.$$

The speed controller has the same sample time and overshoot requirements as the current controller, but the response time is slower, such that:

- Sample time $T_s$= 1 ms.
- Overshoot σ = 5%.
- Response time $t_r$ = 0.50 s.

Therefore, the proportional and integral parameters for the speed controller are:

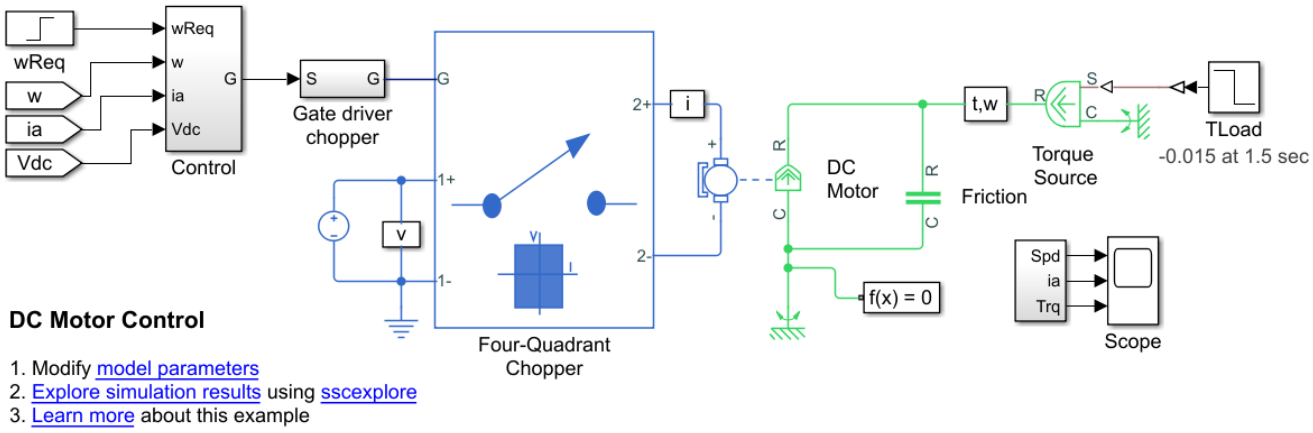
- $K_P$ = 0.0045
- $K_I$ = 0.0405

## Tune the Electric Drive in the Example Model

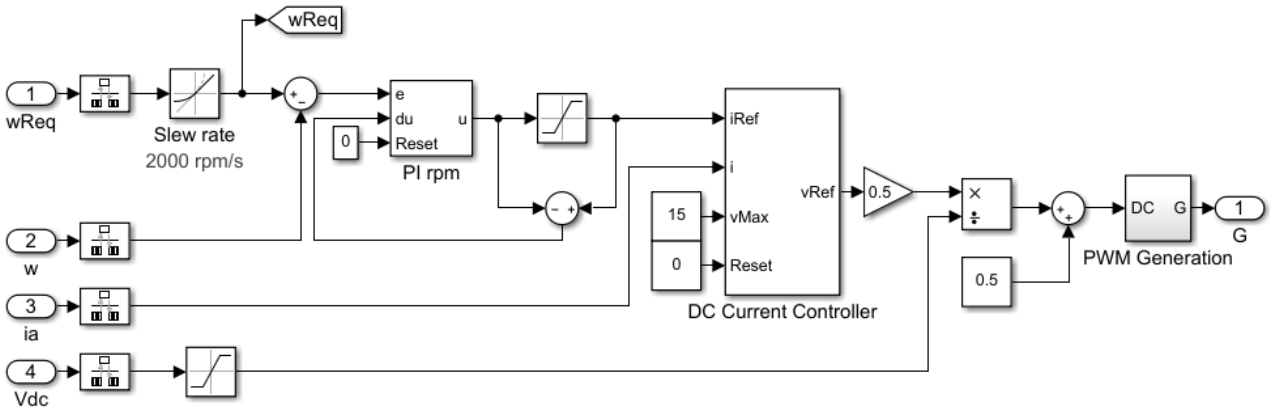
**1** Explore the components of the DC motor and the cascaded controller.

**a** Open the model. At the MATLAB command prompt, enter

```
model = 'ee_dc_motor_control'
open_system(model)
```




**b** The **Control** subsystem contains the model of the cascaded control system built using blocks from the Simulink library.



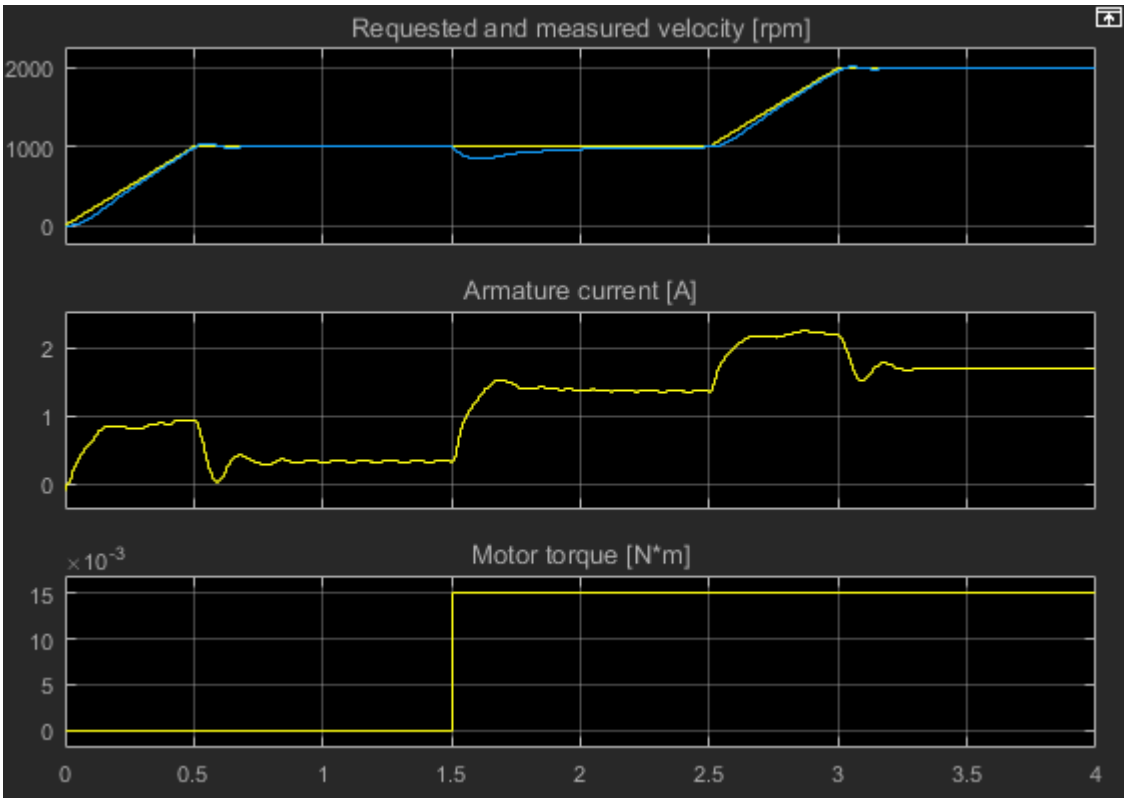

**c** The Four Quadrant Chopper block represents a four-quadrant DC-DC chopper that contains two bridge arms, each of which has two IGBT (Ideal, Switching) blocks. When the input

voltage exceeds the threshold of `0.5` V, the IGBT (Ideal, Switching) blocks behave like linear diodes with a forward-voltage of `0.8 V` and a resistance of `1e-4` ohm. When the threshold voltage is not exceeded, the IGBT (Ideal, Switching) blocks act like linear resistors with an off-state conductance of `1e-5` 1/ohm.

2. Simulate the model.

```
sim(model)
```

3. View the results. Open the **Scope** block.




At 1.5 seconds, there is a load torque that results in a steady-state error.

4. Tune the DC motor controller. The `ee_getDCMotorFirstOrderPIParams` function calculates the proportional gain, $K_P$, and the integral gain, $K_I$, for the first-order system in this example.

The function syntax is `[Kp, Ki] = getParamPI(Km,Tm,Ts,sigma,tr)`.

The input arguments for the function are the system parameters and the requirements for the controller:

- `Km` is the first-order gain.
- `Tm` is the time constant of the first-order system.
- $T_s$ is the sample time for the discrete-time controller.
- `sigma` is the desired maximum overshoot, $\sigma$.
- $t_r$ is the desired response time.

a. To examine the equations in the function, enter

```
edit ee_getDCMotorFirstOrderPIParams
```

**b** To calculate the controller parameters using the function, save these system parameters to the workspace:

```
Ra=4.67;              % [Ohm]
La=170e-3;               % [H]
Bm=47.3e-6;            % [N*m/(rad/s)]
Jm=42.6e-6;            % [Kg*m^2]
Kb=14.7e-3;            % [V/(rad/s)]
Tsc=1e-3;                % [s]
```

**c** Calculate the parameters for tuning the current controller as a function of the parameters and requirements for the inner controller:

- `Km = 1/Ra`.
- `Tm = La/Ra`.
- `Ts = Tsc`.
- `sigma = 0.05`.
- `Tr = 0.11`.

```
[Kp_i, Ki_i] = ee_getDCMotorFirstOrderPIParams(1/Ra,La/Ra,Tsc,0.05,0.11)
```

```
Kp_i =

    7.7099


Ki_i =

  455.1491
```

The gain parameters for the current controller are saved to the workspace.

**d** Calculate the parameters for tuning the speed controller based on the parameters and requirements for the outer controller:

- `Km = Kb*(30/pi)`.
- `Tm = Jm/Ra`.
- `Ts = Tsc`.
- `sigma = 0.05`.
- `Tr = 0.5`.

```
[Kp_n, Ki_n] = ee_getDCMotorFirstOrderPIParams((Kb*(30/pi))/Bm,Jm/Bm,Tsc,0.05,0.5)
```

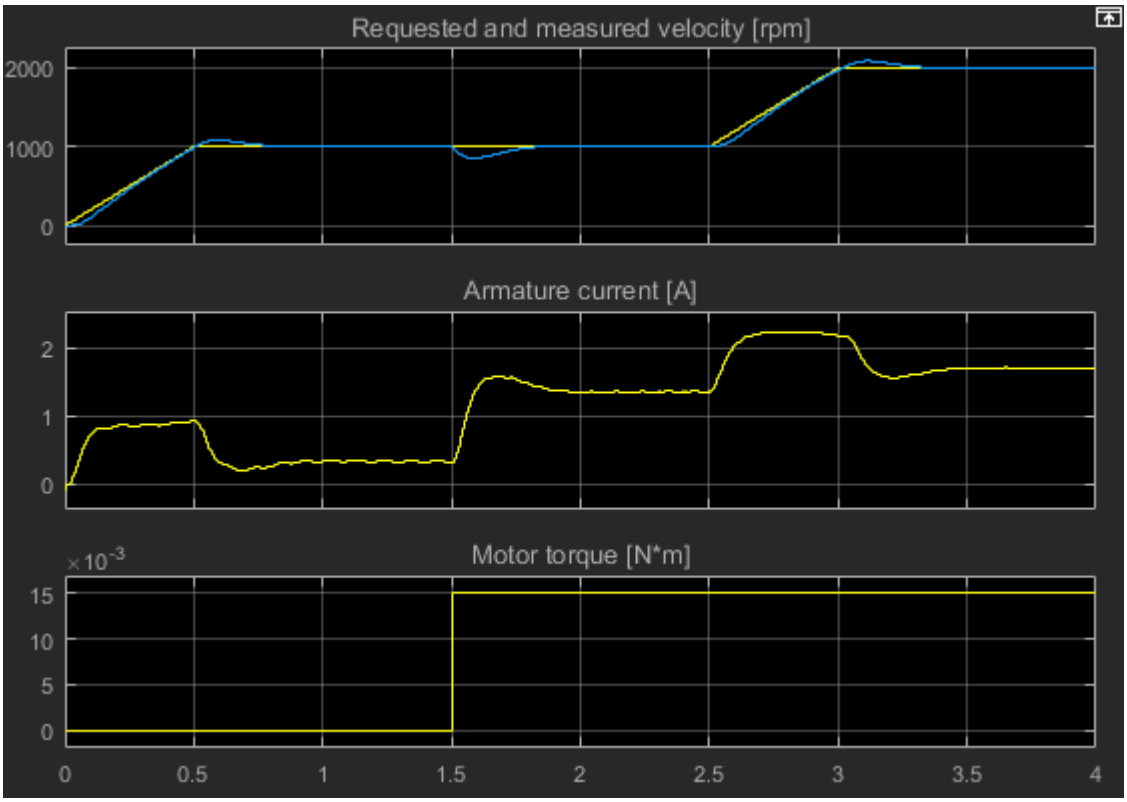
```
Kp_n =

    0.0045


Ki_n =

    0.0405
```

The gain parameters for the speed controller are saved to the workspace.

5 Simulate the model using the saved gain parameters for the speed and controllers.

```
sim(model)
```

6 View the results. Open the **Scope** block.




There is slightly more overshoot, however, the controller responds much faster to the load torque change.

## See Also

Rotational Electromechanical Converter | Inertia | Rotational Friction

## Related Examples

- “DC Motor Control” on page 8-805

# Simulation and Analysis of Power Engineering Systems


- “Optimize Block Settings for Simulating with the Partitioning Solver” on page 7-2
- “Phasor-Mode Simulation Using Simscape Components” on page 7-11
- “Examine the Simulation Data Logging Configuration of a Model” on page 7-15
- “Perform a Power-Loss Analysis” on page 7-17
- “Choose a Simscape Electrical Function for an Offline Harmonic Analysis” on page 7-24
- “Perform an Online Harmonic Analysis Using the Simscape Spectrum Analyzer Block” on page 7-27
- “Perform a Load-Flow Analysis Using Simscape Electrical” on page 7-35

# Optimize Block Settings for Simulating with the Partitioning Solver

| In this section... |
| --- |
| “Update Solver and Zero-Sequence Settings Using the ee_solverUpdate Function” on page 7-2<br>“Limitations of the ee_updateSolver Function” on page 7-9 |

The Partitioning solver is a Simscape fixed-step local solver that improves performance for certain models. However, not all networks can simulate with the Partitioning solver. Some models that use the Partitioning solver can produce errors and fail to initialize due to numerical difficulties. To resolve numerical difficulties preventing initialization with asynchronous, synchronous, and permanent magnet rotor machine blocks, you can exclude zero-sequence terms. Excluding parasitic conductance resolves numerical difficulties with the Floating Neutral (Three-Phase) and Neutral Connection block, which include such conductance by default.

To determine the best solver choice for your model, use the `ee_updateSolver` helper function, which is useful for iterating with various solvers. The function updates certain parameter values for every instance of these blocks in your model:

- Solver Configuration blocks
- Machine blocks that have a **Zero sequence** parameter
- Connection blocks that have a **Parasitic conductance to ground** parameter

The function syntax is `ee_updateSolver(solver,system)`. Specify both input arguments using character vectors. The table shows how the function updates the values, depending on the solver that you specify.

| **Input Argument** | **Solver Configuration Block (*Solver type*)** | **Solver Configuration Block (*Use local solver* and *Use fixed-cost runtime consistency iterations*)** | **Asynchronous,Synchronous, and Permanent Magnet Rotor Machine Blocks (*Zero sequence*)** | **Floating Neutral (Three-Phase) Block and Neutral Connection Block (*Parasitic conductance to ground*)** |
| --- | --- | --- | --- | --- |
| 'Partitioning' | Partitioning | Selected | `Exclude` | `0` |
| 'Backward Euler' or 'BackwardEuler' | Backward Euler | Selected | `Include` | `1e-12` |
| 'Trapezoidal' | Trapezoidal | Selected | `Include` | `1e-12` |
| 'Global' or 'Nonlocal' | No change | Cleared | `Include` | `1e-12` |

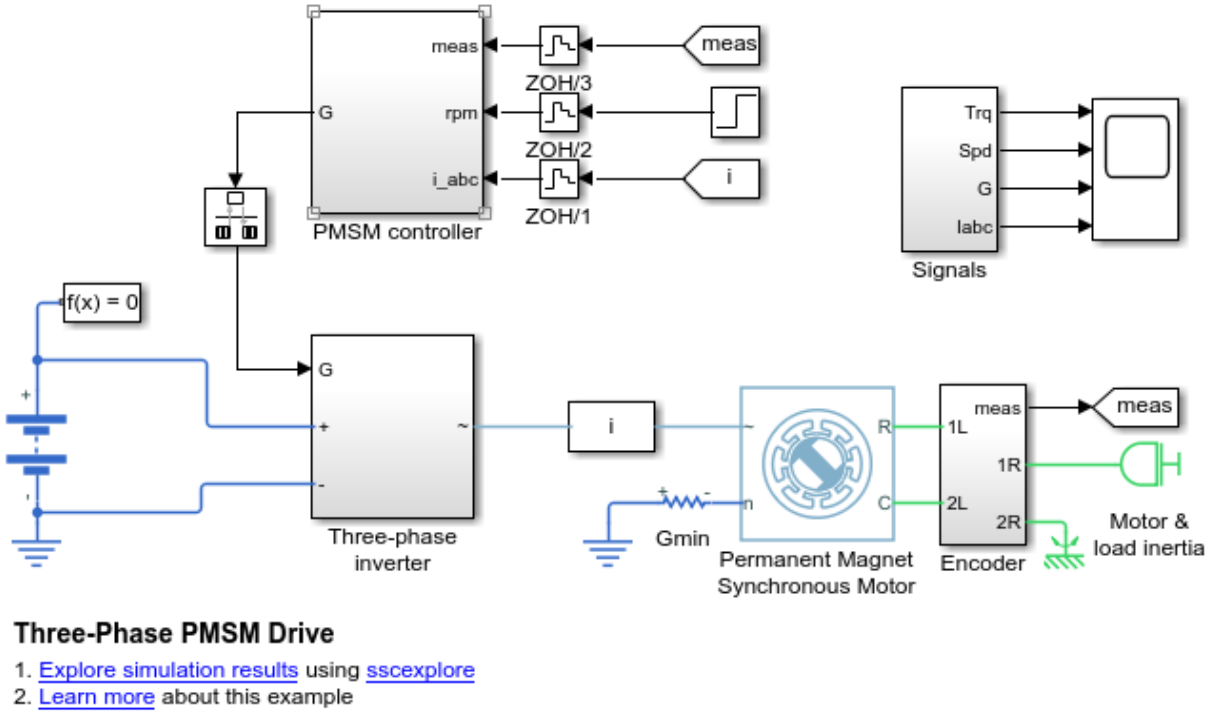
## Update Solver and Zero-Sequence Settings Using the ee_solverUpdate Function

This example shows how to use the `ee_solverUpdate` function to configure the Solver Configuration and PMSM blocks in a model for simulation with the Partitioning solver and the Backward Euler solver. It also shows how to compare the simulation duration times and the results.

1 Open the model. At the MATLAB command prompt, enter this code.

**See Code**

```
model = 'ee_pmsm_drive';
open_system(model)
```




Two blocks that the `ee_solverUpdate` function can update are the Solver Configuration block and PMSM block.

2 Save the parameter settings for the two blocks.

**See Code**

```
% Define the Solver Configuration block and the path
%    to it as variables
solvConfig = 'Solver Configuration';
solvConfigPath = [model,'/',solvConfig];

% Define the machine block and the path
%    to it as variables
machine = 'Permanent Magnet Synchronous Motor';
machinePath = [model,'/',machine];

% Create a cell array that contains configuration data
configBaseline = {'Block','Parameter','Value';
        'Solver Configuration','Use local solver',...
        get_param(solvConfigPath,'UseLocalSolver');
        'Solver Configuration','Solver type',...
```

```
get_param(solvConfigPath,'LocalSolverChoice');
'Solver Configuration','Sample time',...
get_param(solvConfigPath,'LocalSolverSampleTime');
'Solver Configuration',...
'Use fixed-cost runtime consistency iterations',...
get_param(solvConfigPath,'DoFixedCost');
'Machine','Zero sequence',...
get_param(machinePath,'zero_sequence')};
```

The settings are saved to `configBaseline` array in the MATLAB workspace.

configBaseline

6x3 cell

| | 1 | 2 | 3 |
|---|---|---|---|
| 1 | Block | Parameter | Value |
| 2 | Solver Configuration | Use local solver | off |
| 3 | Solver Configuration | Solver type | NE_BACKWARD_EULER_ADVANCER |
| 4 | Solver Configuration | Sample time | Ts |
| 5 | Solver Configuration | Use fixed-cost runtime consistency iterations | off |
| 6 | Machine | Zero sequence | pe.enum.park.zerosequence.include |

The settings of interest for the Solver Configuration block are:

- **Use local solver** — The option to use a local Simscape solver is cleared.
- **Solver type** — Backward Euler, a Simscape local fixed- cost solver, is specified. However, if you open the block dialog box, you can see that it is not enabled because the option to use a local solver is cleared.
- **Use fixed-cost runtime consistency iterations** — The option to use fixed-cost is cleared. This option is also disabled when the option to use a local solver is cleared.

For the machine, the **Zero sequence** parameter is set to `Include`. Zero-sequence equations can cause numerical difficulty when you simulate with the Partitioning solver.

3 To return all simulation outputs within a single `Simulink.SimulationOutput` object so that you can later compare simulation times, enable the single-output format of the `sim` command.

```
% Enable single-output format
set_param(model,'ReturnWorkspaceOutputs', 'on')
```

4 Mark the rotor torque signal, which connects the **trqMotor** From block to a Mux block, for Simulation Data Logging and viewing with the Simulation Data Inspector.

**See Code**

```
% Define the trqMotor From block and the path
%    to it as variables
torqueSensor = 'From6';
signalSubsystem = 'Signals';
torqueSensorPath = [model,'/',signalSubsystem,'/',torqueSensor];

% Mark the output signal from the trqMotor From block
%   for Simulink(R) data logging
```

```
phTorqueSensor = get_param(torqueSensorPath,'PortHandles');
set_param(phTorqueSensor.Outport(1),'DataLogging','on')
```

The logging badge marks the signal in the model.

5 Determine the results and how long it takes to simulate with the baseline settings.

**See Code**

```
% Run a timed simulation using the Baseline solver configuration
out = sim(model);
tBaseline  = out.SimulationMetadata.TimingInfo.ExecutionElapsedWallTime;
```

6 Use `ee_updateSolver` function to change to the Backward Euler solver configuration. Save the configuration settings, and compare the settings to the baseline settings.

**See Code**

```
% Configure for Backward Euler solver simulation
ee_updateSolver('Backward Euler',model)

% Save the new parameter settings and compare them to the baseline
%   configuration.

% Create a cell array that contains configuration data
configBackEuler = {'Block','Parameter','Value';
        'Solver Configuration','Use local solver',...
        get_param(solvConfigPath,'UseLocalSolver');
        'Solver Configuration','Solver type',...
        get_param(solvConfigPath,'LocalSolverChoice');
        'Solver Configuration','Sample time',...
        get_param(solvConfigPath,'LocalSolverSampleTime');
        'Solver Configuration',...
        'Use fixed-cost runtime consistency iterations',...
        get_param(solvConfigPath,'DoFixedCost');
        'Machine','Zero sequence',...
        get_param(machinePath,'zero_sequence')};

% Compare the Partitioning solver block settings to the Baseline settings
configDiff = setdiff(configBackEuler,configBaseline)
```

```
configDiff =

  1×1 cell array

    {'on'}
```

configBackEuler

{} 6x3 cell

| | 1 | 2 | 3 |
|---|---|---|---|
| 1 | Block | Parameter | Value |
| 2 | Solver Configuration | Use local solver | on |
| 3 | Solver Configuration | Solver type | NE_BACKWARD_EULER_ADVANCER |
| 4 | Solver Configuration | Sample time | Ts |
| 5 | Solver Configuration | Use fixed-cost runtime consistency iterations | on |
| 6 | Machine | Zero sequence | pe.enum.park.zerosequence.include |

The option to use the local solver, which is set to Backward Euler by default, and the option to use fixed-cost runtime consistency iterations are now both selected.

7 Run a timed simulation using the Backward Euler solver.

### See Code

```
out = sim(model);
tBackEuler = out.SimulationMetadata.TimingInfo.ExecutionElapsedWallTime;
```

8 If you change the local solver to the Partitioning solver and simulate the model now, an error occurs because of the zero-sequence terms. Use the `ee_updateSolver` function to configure the model for simulating with the Partitioning solver without generating an error. Save the configuration settings, compare the settings to baseline settings, and run a timed simulation.

### See Code

```
% Configure for Partioning solver simulation
ee_updateSolver('Partitioning', model)

% Create a cell array that contains configuration data
configPartitioning = {'Block','Parameter','Value';
        'Solver Configuration','Use local solver',...
        get_param(solvConfigPath,'UseLocalSolver');
        'Solver Configuration','Solver type',...
        get_param(solvConfigPath,'LocalSolverChoice');
        'Solver Configuration','Sample time',...
        get_param(solvConfigPath,'LocalSolverSampleTime');
        'Solver Configuration',...
        'Use fixed-cost runtime consistency iterations',...
        get_param(solvConfigPath,'DoFixedCost');
        'Machine','Zero sequence',...
        get_param(machinePath,'zero_sequence')};

% Compare the Partitioning solver block settings to the Baseline settings
configDiff = setdiff(configPartitioning,configBaseline)

%  Run a timed simulation using the Partitioning solver
out = sim(model);
tPartitioning = out.SimulationMetadata.TimingInfo.ExecutionElapsedWallTime;
```

```
configDiff =

  3×1 cell array

    {'NE_PARTITIONING_ADVANCER'         }
    {'ee.enum.park.zerosequence.exclude'}
    {'on'                               }

Warning: Initial conditions for nondifferential variables
not supported. The following states may deviate from
requested initial conditions:
   ['<a
   href="matlab:open_and_hilite_system('ee_pmsm_drive/Battery')"...
        >ee_pmsm_drive/Battery</a>']
   Battery.num_cycles
      o In ee.sources.battery_base
   ['<a
   href="matlab:open_and_hilite_system('ee_pmsm_drive/Permanent
   Magnet Synchronous Motor')">ee_pmsm_drive/Permanent
   Magnet Synchronous Motor</a>']
   Permanent_Magnet_Synchronous_Motor.angular_position
```

configPartitioning

6x3 cell

| | 1 | 2 | 3 |
|---|---|---|---|
| 1 | Block | Parameter | Value |
| 2 | Solver Configuration | Use local solver | on |
| 3 | Solver Configuration | Solver type | NE_PARTITIONING_ADVANCER |
| 4 | Solver Configuration | Sample time | Ts |
| 5 | Solver Configuration | Use fixed-cost runtime consistency iterations | on |
| 6 | Machine | Zero sequence | pe.enum.park.zerosequence.exclude |

The solver type is now set to the Partitioning solver and the machine is configured to exclude zero-sequence terms.

The simulation runs without generating an error. It does generate a warning because initial conditions for nondifferential variables are not supported for the Partitioning solver.

**9** Print tables that show:

- Simulation time for each solver
- Percent differences in speed for the local solvers versus the baseline global solver.

**See Code**

```
% Display the simulation times
compTimeDiffTable = table({'Baseline';...
    'Backward Euler';...
    'Partitioning'},...
    {tBaseline;tBackEuler;tPartitioning},...
'VariableNames', {'Solver','Sim_Duration'});

display(compTimeDiffTable);

% Compute and display the percent difference for the simulation times
spdBackEulerVsBaseline = 100*(tBaseline - tBackEuler)/tBaseline;
spdPartitionVsBaseline = 100*(tBaseline - tPartitioning)/tBaseline;

compPctDiffTable = table({'Backward Euler versus Baseline';...
    'Partitioning versus Baseline'},...
    {spdBackEulerVsBaseline;...
    spdPartitionVsBaseline},...
'VariableNames', {'Comparison','Percent_Difference'});

display(compPctDiffTable);
```

```
  3×2 table

        Solver         Sim_Duration
    _______________    ____________

    'Baseline'           [38.0255]
    'Backward Euler'     [23.4011]
    'Partitioning'       [ 9.2042]
```

```
compPctDiffTable =

  2×2 table

              Comparison                Percent_Difference
    ____________________________________    __________________

    'Backward Euler versus Baseline'          [38.4594]
    'Partitioning versus Baseline'            [75.7946]
```

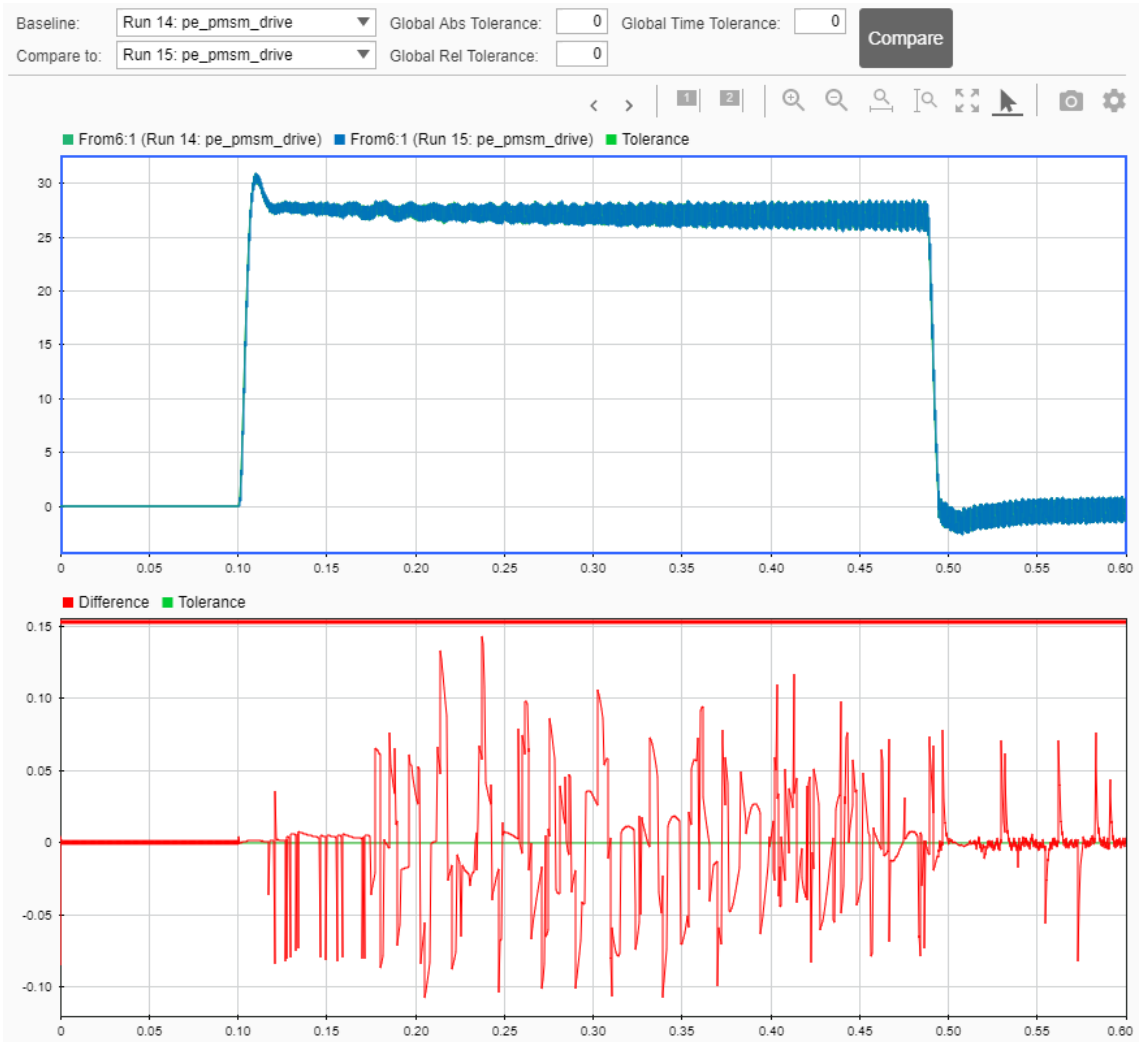
Simulation time on your machine may differ because simulation speed depends on machine processing power and the computational cost of concurrent processes. The local fixed-step Partitioning and Backward Euler solvers are faster than the baseline solver, which is a global, variable-step solver. The Partitioning solver is faster than the Backward Euler solver.

**10** To compare the results, open the Simulation Data Inspector.

### See Code

```
% Get Simulation Data Inspector run IDs for
%    the last three runs
runIDs = Simulink.sdi.getAllRunIDs;
runBackEuler = runIDs(end - 1);
runPartition = runIDs(end);

% Open the Simulation Data Inspector
Simulink.sdi.view

compBaselinePartition = Simulink.sdi.compareRuns(runBackEuler,...
    runPartition);
```

To see the comparison, click **Compare** and then click **From6**.




The first plot shows the overlay of the Backward Euler and Partitioning solver simulation results. The second plot shows how they differ. The default tolerance for differences is `0`. To determine if the accuracy of the results meets your requirements, you can adjust the relative, absolute, and time tolerances. For more information, see “Compare Simulation Data”.

You can also use the `ee_updateSolver` function to reset the model for simulation with a global solver.

**See Code**

```
% Configure for Global/Nonlocal solver simulation
ee_updateSolver('Global',model)
```

## Limitations of the ee_updateSolver Function

Using the `ee_updateSolver` function does not guarantee that a simulation does not generate an error or that a simulation produces accurate results. To ensure that simulation accuracy meets your requirements, it is a recommended practice to compare simulation results to baseline results whenever you change model or block settings.

## See Also

Solver Configuration | PMSM

## Related Examples

- “Increase Simulation Speed Using the Partitioning Solver”

# Phasor-Mode Simulation Using Simscape Components

You can run your model in phasor mode to speed up simulation. In Simscape, phasor mode is known as frequency-time equation formulation. In general, this formulation leads to accurate simulation of AC models using larger time steps than the traditional time formulation.

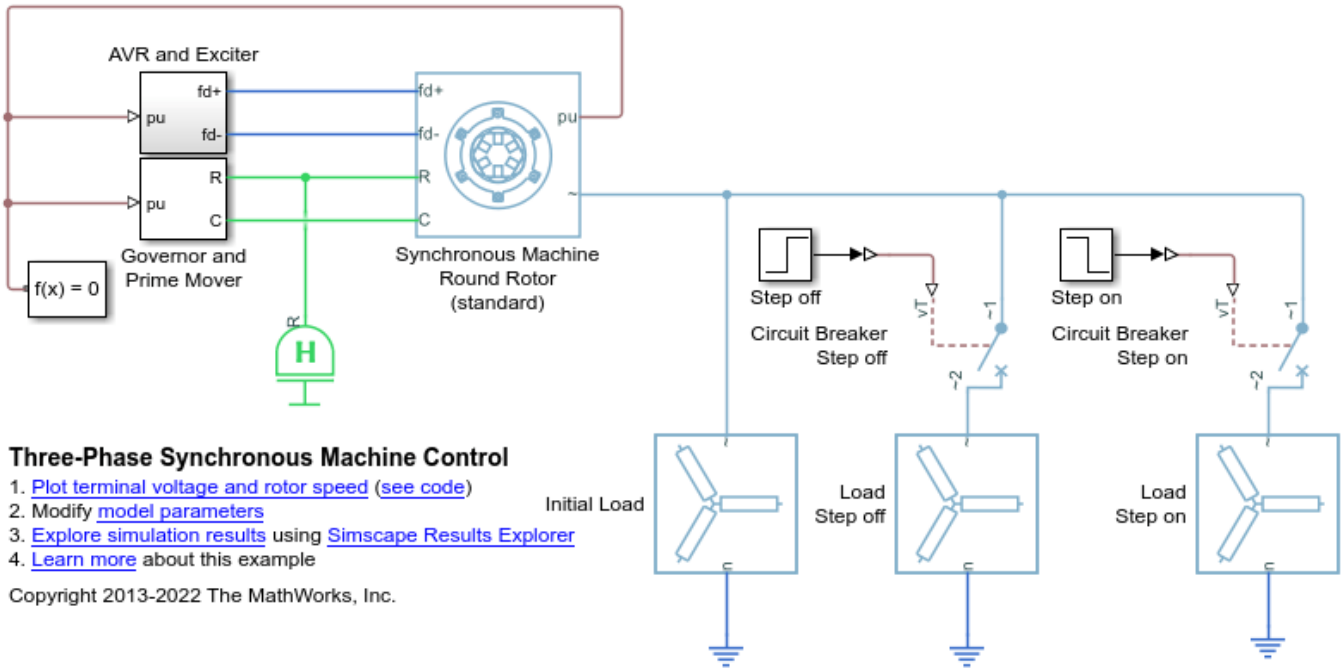
Use frequency-time equation forumulation to speed up your simulation when:

- Your simulation contains periodic AC signals with a common fundamental frequency
- You are interested in the slow-moving AC-related quantities, such as amplitude or phase, and the DC output signals

### Set up the model

To measure the time required to run a simulation, open the model *ee_sm_control* and create a model callback.

```
mdl = load_system('ee_sm_control');
open_system(mdl);
set_param(mdl,'StartFcn','tic;');
set_param(mdl,'StopFcn','tsim=toc;');
```




### Run a time-based simulation

Double-click the Solver Configuration block and apply the following configuration:

- Enable the local solver by checking the **Use local solver** check box
- Set the `Sample time` parameter to `1e-3`

- Set the `Equation formulation` parameter to `Time`

You can also run this code to configure the block.

```
blk = find_system(mdl,'MaskType','Solver Configuration');
set_param(blk,'UseLocalSolver','on');
set_param(blk,'LocalSolverSampleTime','1e-3');
set_param(blk,'EquationFormulation','NE_TIME_EF');
```

Simulate the model and save the run time and logging variable.

```
sim(get_param(mdl,'Name'));
tsim_time = round(tsim,2);
simlog_ee_sm_control_time = simlog_ee_sm_control;
```

**Run a phasor-mode simulation**

Double-click the Solver Configuration block and apply the following configuration:

- Enable the local solver by checking the `Use local solver` check box
- Set the `Sample time` parameter to `1e-2`
- Set the `Equation formulation` parameter to `Frequency and time`

You can also run this code to configure the block.

```
blk = find_system(mdl,'name','Solver Configuration');
set_param(blk,'UseLocalSolver','on');
set_param(blk,'LocalSolverSampleTime','1e-2');
set_param(blk,'EquationFormulation','NE_FREQUENCY_TIME_EF');
```

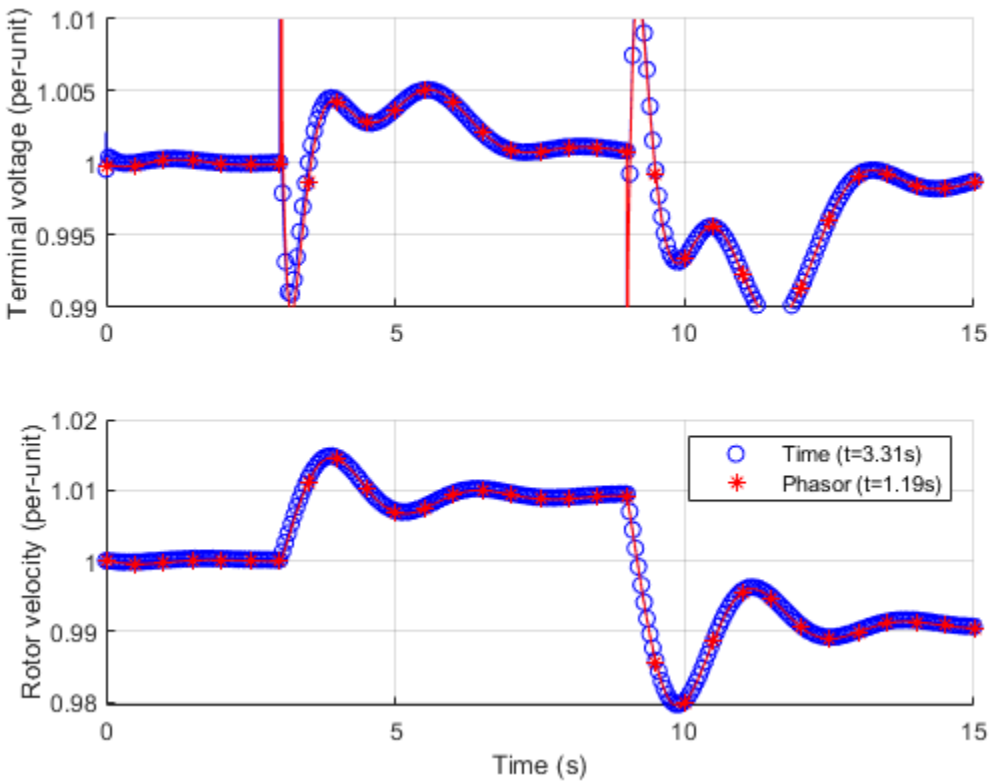
Simulate the model and save the run time and logging variable.

```
sim(get_param(mdl,'Name'));
tsim_phasor = round(tsim,2);
simlog_ee_sm_control_phasor = simlog_ee_sm_control;
```

**Compare DC results**

Plot the field voltage and rotor speed for both the time and frequency-time simulations. For each simulation mode, display markers at every 50 data points.

```
[hTime,hPhasor]=setup_figure(simlog_ee_sm_control_time,simlog_ee_sm_control_phasor,'dc');
legend([hTime,hPhasor],{['Time (t=',num2str(tsim_time),'s)'],['Phasor (t=',num2str(tsim_phasor),
```

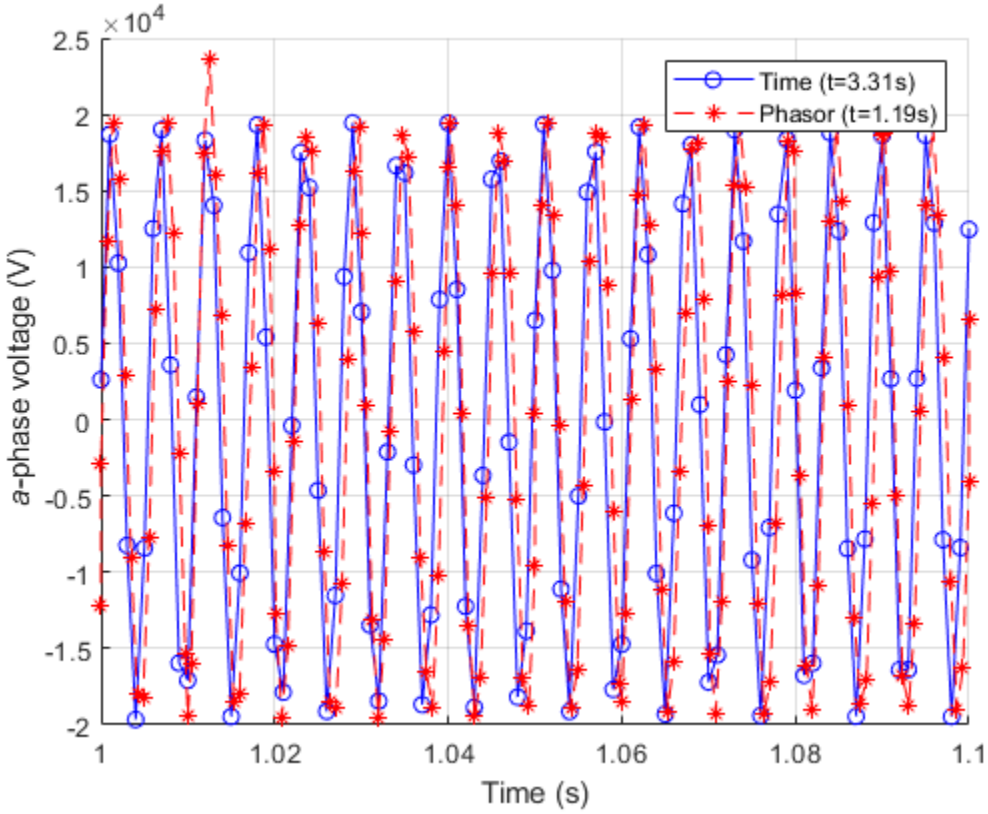



The phasor simulation reproduces near-identical results as the time-based simulation, despite using a time step that is 10 times larger. The measured simulation time is also shown for each of the simulation modes in the plot legend. This performance indicator is different on different machines, but the frequency-time simulation should be about two times faster than the time simulation. Note that the actual time required per step is higher in the frequency-time case, but the overal time is reduced.

### Compare AC results

Plot the *a*-phase voltage of the synchronous machine over the time period `1s` to `1.1s`. Because of the larger time steps in the frequency-time formulation, the resolution of the AC quantity is too small to make out the sine wave. The points that are available are undersampled, but still accurate.

```
[hTime,hPhasor]=setup_figure(simlog_ee_sm_control_time,simlog_ee_sm_control_phasor,'ac');
legend([hTime,hPhasor],{['Time (t=',num2str(tsim_time),'s)'],['Phasor (t=',num2str(tsim_phasor),
```




In general, use frequency-time formulation to speed up simulations where the outputs of interest are DC or slow-moving AC quantities. You can use periodic sensors to measure slow-moving properties of AC signals such as amplitude and phase in both time and frequency time formulations. For more information, see the PS Harmonic Estimator (Amplitude, Phase) block.

Sometimes there are small phase offsets between time- and frequency-time-generated AC signals. This difference is caused by the accumulated integration error of a slightly different signal frequency over time.

## See Also

Solver Configuration

## More About

- “Frequency and Time Simulation Mode”

# Examine the Simulation Data Logging Configuration of a Model

Many analyses that you can perform using Simscape Electrical require a simulation log variable in your MATLAB workspace. The model in this example is configured to log Simscape data for the whole model for the entire simulation time. To examine the data logging configuration of a model:

1. Open the model. At the MATLAB command prompt, enter

   ```
   model = 'ee_rectifier_power_dissipated';
   open(model)
   ```

2. Open the model configuration parameters and then, in the left pane, select **Simscape**. Relevant parameters are:

   - **Log simulation data** — Data logging is enabled for the whole model because this parameter is set to `All` so you can calculate the power dissipated by any of the semiconductors in the model.
   - **Workspace variable name** — This parameter, which is also referred to as the name of the simulation log variable, is specified as `simlog_ee_rectifier_power_dissipated`.
   - **Limit data points** — You can calculate the power dissipated for the entire simulation time because the option is not selected.

Alternatively, you can determine the Simscape data logging configuration without opening the model configuration parameters, by using the `get_param` function. For example, for the `ee_rectifier_power_dissipated` model, to determine:

- If all, some, or no data is logged, at the MATLAB command prompt, enter

  ```
  get_param(model,'SimscapeLogType')

  ans =

      'all'
  ```

- The name of the Simscape logging variable

  ```
  get_param(model,'SimscapeLogName')

  ans =

      'simlog_ee_rectifier_power_dissipated'
  ```

- If the option to limit data-points is on or off

  ```
  get_param(model,'SimscapeLogLimitData')

  ans =

      'off'
  ```

## See Also

**Functions**
`get_param`

## Related Examples

- “Data Logging”

# Perform a Power-Loss Analysis

| In this section... |
| --- |
| “Prerequisite” on page 7-17 |
| “Calculate Average Power Losses for the Simulation” on page 7-17 |
| “Analyze Power Dissipation Differences Using Instantaneous Power Dissipation” on page 7-18 |
| “Mitigate Transient Effects in Simulation Data” on page 7-21 |

This example shows how to analyze power loss and how to mitigate transient power dissipation behavior. Analyzing power loss, with and without transients, is useful for determining if components are operating within safety and efficiency guidelines.

## Prerequisite

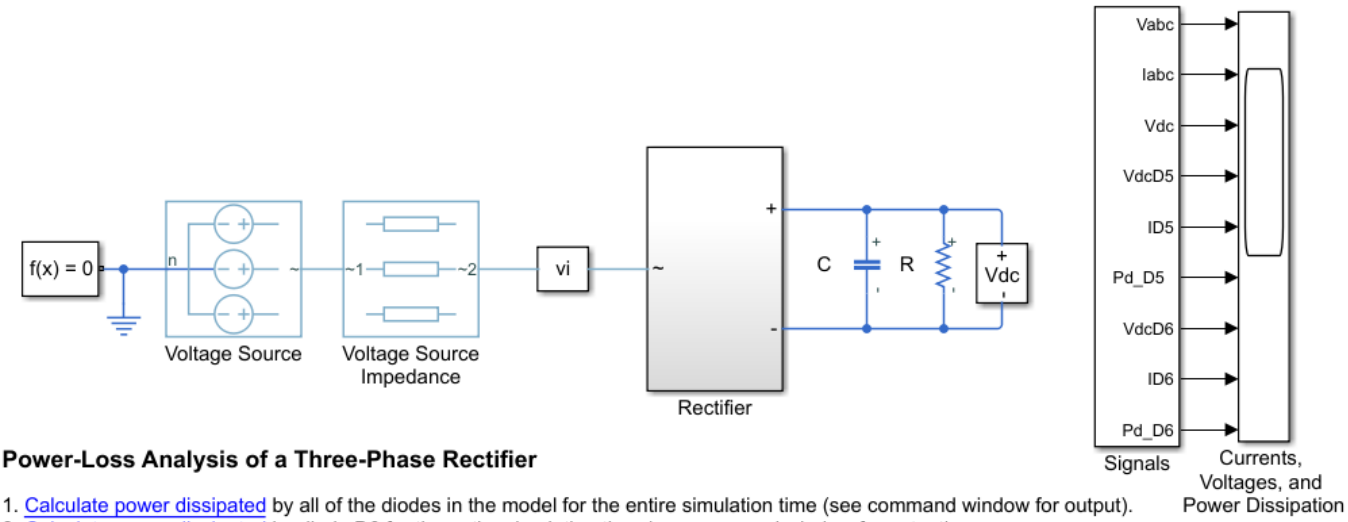
This example requires a simulation log variable in your MATLAB workspace. The model in this example is configured to log Simscape data for the whole model for the entire simulation time.

To learn how to determine if a model is configured to log simulation data, see “Examine the Simulation Data Logging Configuration of a Model” on page 7-15.

## Calculate Average Power Losses for the Simulation

**1** Open the model. At the MATLAB command prompt, enter

```
model = 'ee_rectifier_power_dissipated';
open(model)
```




**Power-Loss Analysis of a Three-Phase Rectifier**

1. Calculate power dissipated by all of the diodes in the model for the entire simulation time (see command window for output).
2. Calculate power dissipated by diode D6 for the entire simulation time (see command window for output).
3. Explore simulation results using sscexplore.
4. Calculate the maximum power dissipated by diode D6. (see command window for output).
5. Calculate power dissipated by diode D6 between simulation time, *t* = 1e-3 to 0.5 s (see command window for output).
6. Plot power dissipated by the diodes (see code).
7. Calculate power dissipated by all of the diodes between simulation time, *t* = 0.4 to 0.5 s (see command window for output).
8. Learn more about the ee_getPowerLossSummary function.

2. Simulate the model.

   ```
   sim(model)
   ```

   The simulation log variable, which is named `simlog_ee_rectifier_power_dissipated`, appears in the workspace.

3. Calculate the average losses for the entire simulation for each of the diodes in the model.

   ```
   rectifierLosses = ee_getPowerLossSummary(simlog_ee_rectifier_power_dissipated.Rectifier)

   rectifierLosses =

     6×2 table

                                    LoggingNode                                    Power
     ___________________________________________________________________________    _____

       'ee_rectifier_power_dissipated.Rectifier.D6'                                 52.222
       'ee_rectifier_power_dissipated.Rectifier.D3'                                 52.222
       'ee_rectifier_power_dissipated.Rectifier.D4'                                 52.194
       'ee_rectifier_power_dissipated.Rectifier.D5'                                 52.194
       'ee_rectifier_power_dissipated.Rectifier.D1'                                 52.194
       'ee_rectifier_power_dissipated.Rectifier.D2'                                 52.194
   ```

   On average, diodes D3 and D6 dissipate more power than the other diodes in the rectifier.

## Analyze Power Dissipation Differences Using Instantaneous Power Dissipation

The Diode blocks each have a *power_dissipated* variable, which measures instantaneous power dissipation. To investigate the differences in the average power dissipated by the diodes, view the simulation data using the Simscape Results Explorer.

1. Open the simulation data using the Results Explorer.

   ```
   sscexplore(simlog_ee_rectifier_power_dissipated)
   ```

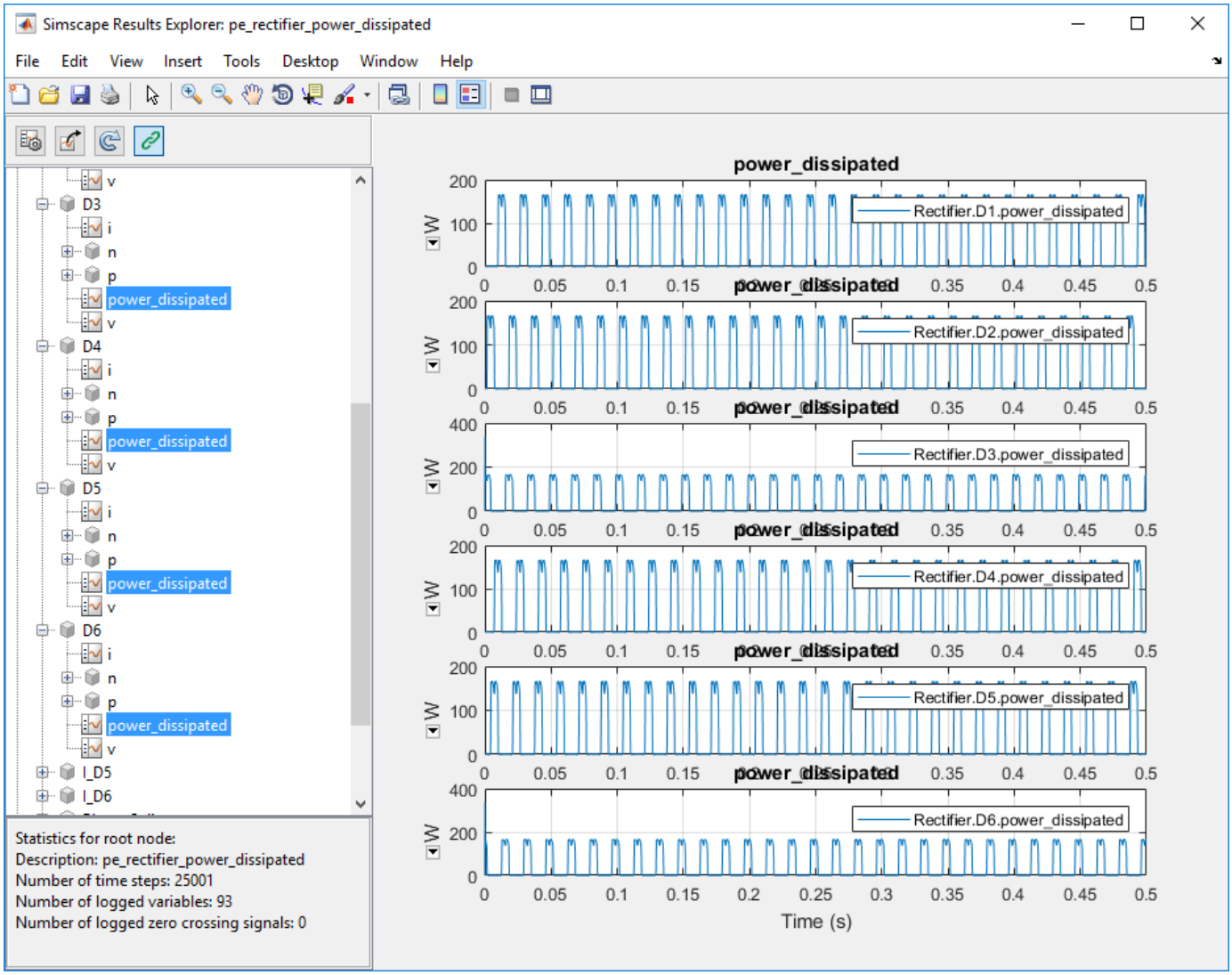
2. View the instantaneous power dissipated by the diodes.

   a. Expand the **Rectifier** node

   b. Expand the **D1** through **D6** nodes

   c. Click the `power_dissipated` nodes for diode **D1**, and then **Ctrl+click** the `power_dissipated` nodes for the other five diodes.

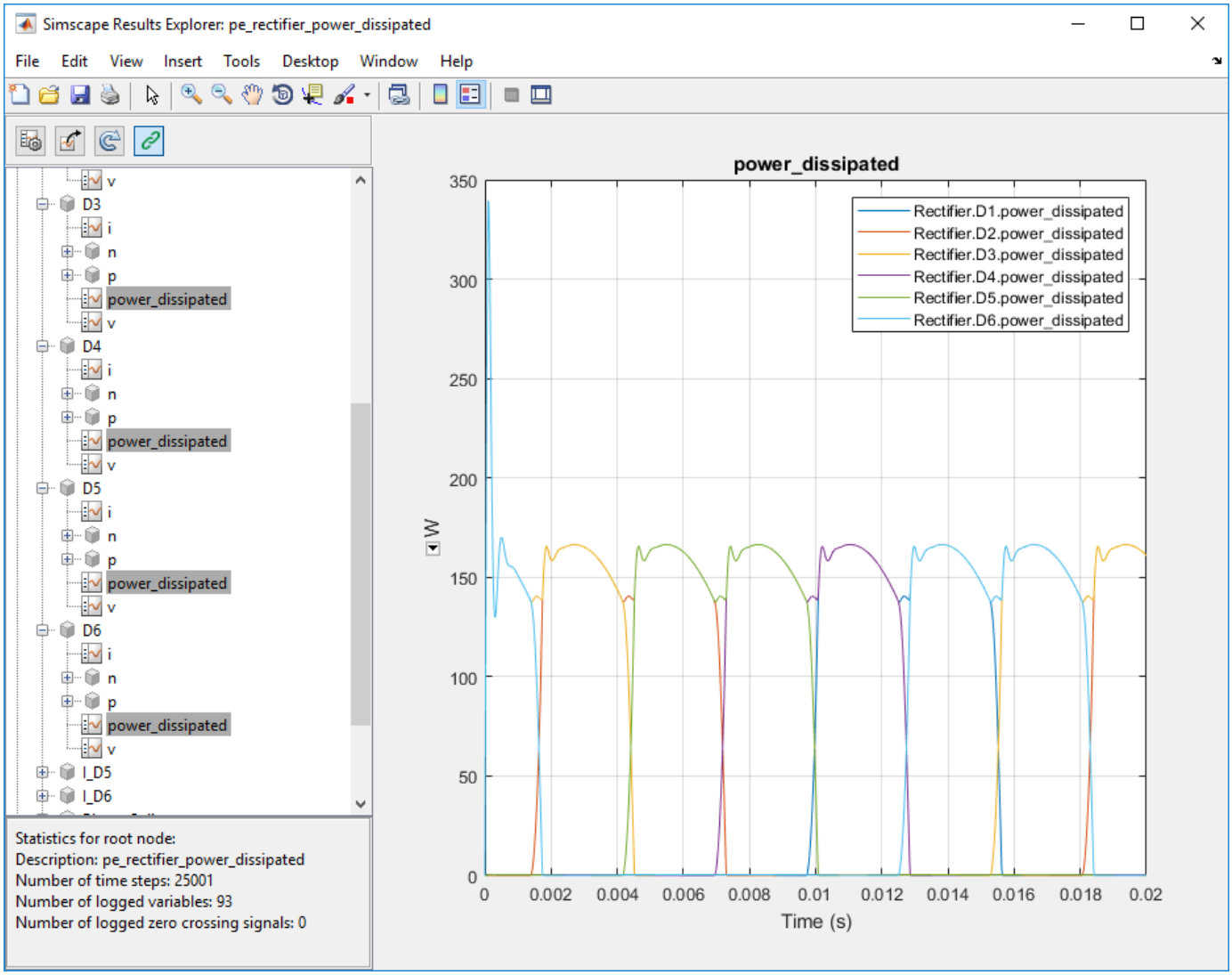
   d. In the Results Explorer window, click the plot options button and set **Plot signals** to `Separate`.




At the beginning of the simulation, there is a difference in the power dissipation for each diode.

3 Take a closer look at the differences. Overlay the plots and zoom to the beginning of the simulation.

   a In the Results Explorer window, click the plot options button.
   b Enable the **Limit time axis** option.
   c For **Stop time**, specify `0.02`.
   d Set **Plot signals** to `Overlay`.
   e Click **OK**.




The variation in power dissipation is due to transient behavior at the beginning of the simulation. The model reaches steady state at simulation time, $t \backsimeq 0.001$ seconds.

4 Determine the average power dissipation for only the diodes during the interval that contains transient behavior.

```
rectifierLosses = ee_getPowerLossSummary(simlog_ee_rectifier_power_dissipated.Rectifier,0,1e-3)
```

```
rectifierLosses =

  6×2 table

     LoggingNode        Power
    _______________    ________

    'Rectifier.D3'       174.88
    'Rectifier.D6'       174.88
    'Rectifier.D4'      0.27539
    'Rectifier.D5'      0.27539
```

```
    'Rectifier.D1'     0.12482
    'Rectifier.D2'    0.032017
```

The average power dissipated by diodes D3 and D6 exceeds the average for the other diodes.

5 Output a table of the maximum power dissipation for each diode, for the entire simulation time.

```
pd_D1_max = max(simlog_ee_rectifier_power_dissipated.Rectifier.D1.power_dissipated.series.values);
pd_D2_max = max(simlog_ee_rectifier_power_dissipated.Rectifier.D2.power_dissipated.series.values);
pd_D3_max = max(simlog_ee_rectifier_power_dissipated.Rectifier.D3.power_dissipated.series.values);
pd_D4_max = max(simlog_ee_rectifier_power_dissipated.Rectifier.D4.power_dissipated.series.values);
pd_D5_max = max(simlog_ee_rectifier_power_dissipated.Rectifier.D5.power_dissipated.series.values);
pd_D6_max = max(simlog_ee_rectifier_power_dissipated.Rectifier.D6.power_dissipated.series.values);

diodes = {'D1';'D2';'D3';'D4';'D5';'D6'};
PowerMax = [pd_D1_max;pd_D2_max;pd_D3_max;pd_D4_max;pd_D5_max;pd_D6_max];

T = table(PowerMax,'RowNames', diodes)
```

```
T =

  6×1 table

          PowerMax
          ________

    D1    166.45
    D2    166.45
    D3    339.54
    D4    166.45
    D5    166.45
    D6    339.54
```

The maximum instantaneous power dissipation for diodes D3 and D6 is almost double the maximum instantaneous power dissipation for the other diodes.

## Mitigate Transient Effects in Simulation Data

To mitigate the transient power dissipation at the beginning of the simulation, use the final simulation state to initialize a new simulation at steady-state conditions.

1 Configure the model to save the final state.

- **a** Open the model configuration parameters.
- **b** In the **Solver** pane, change the **Stop time** from `0.5` to 1e-3.
- **c** In the **Data Import/Export** pane, select these options:
  - **Final States**
  - **Save final operating point**
- **d** Click **Apply.**

2 Run the simulation.

The final state is saved as the variable *xFinal* in the MATLAB workspace.

3 Configure the model to initialize using *xFinal*, in the model configuration parameters.

- **a** In the Data Import/Export pane:

- Select the **Initial state** option.
- Change the **Initial state** parameter value from `xInitial` to `xFinal`.
- Clear the **Final states** option.

**b** In the **Solver** pane, change the **Stop time** to `0.5`.

**c** Click **OK**.

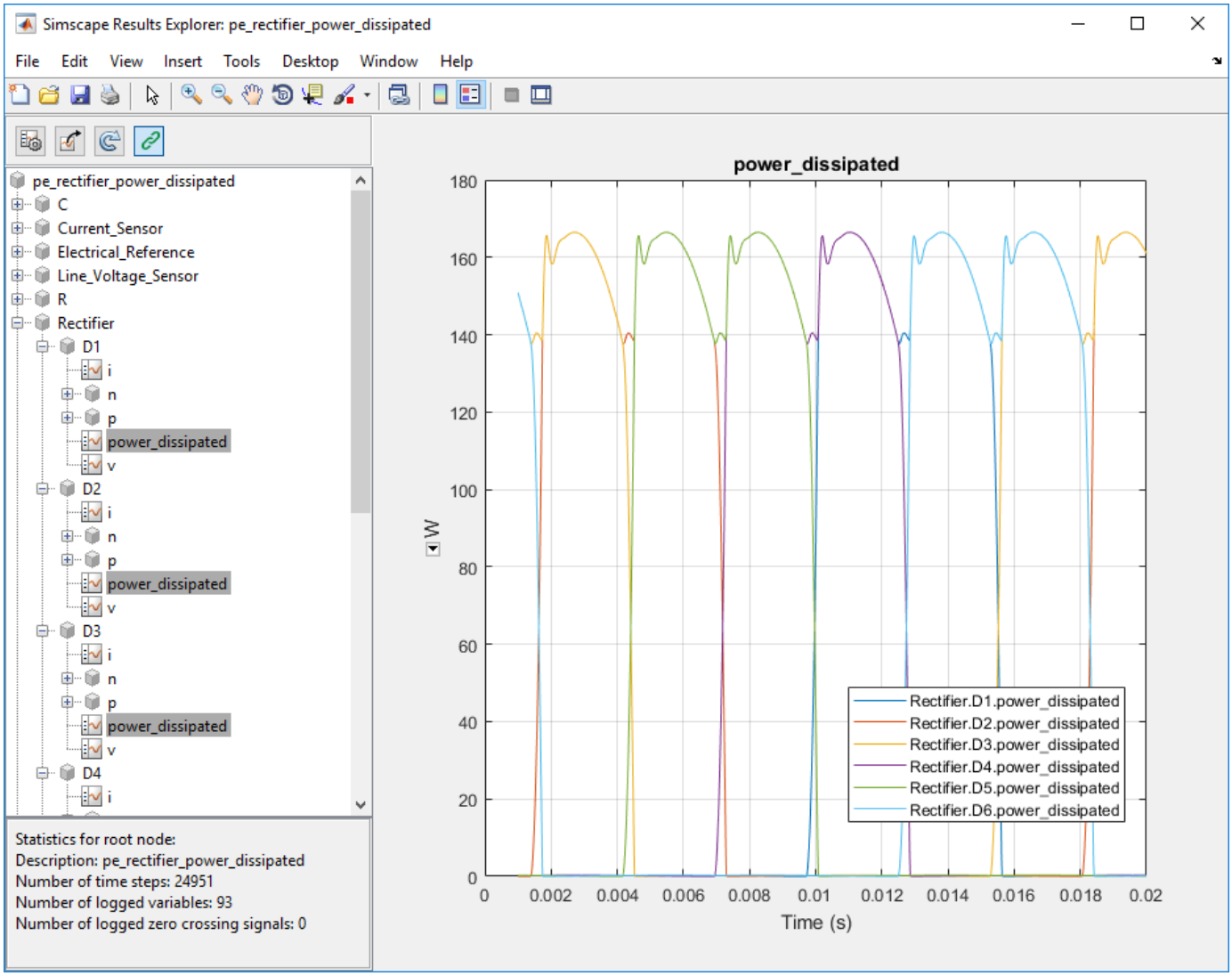
**4** Run the simulation.

**5** View the data from the new simulation.

**a** Click the **Reload logged data** button in the Simscape Results Explorer.

**b** Click **OK** to confirm that `simlog_ee_rectifier_power_dissipated` is the variable name that contains the logged data.

**c** To see the data more clearly, click and drag the legend away from the peak amplitudes.




The plot shows that the simulation no longer contains the transient.

6 Output a table of the maximum power dissipation for each diode, for the modified simulation.

```
pd_D1_max = max(simlog_ee_rectifier_power_dissipated.Rectifier.D1.power_dissipated.series.values);
pd_D2_max = max(simlog_ee_rectifier_power_dissipated.Rectifier.D2.power_dissipated.series.values);
pd_D3_max = max(simlog_ee_rectifier_power_dissipated.Rectifier.D3.power_dissipated.series.values);
pd_D4_max = max(simlog_ee_rectifier_power_dissipated.Rectifier.D4.power_dissipated.series.values);
pd_D5_max = max(simlog_ee_rectifier_power_dissipated.Rectifier.D5.power_dissipated.series.values);
pd_D6_max = max(simlog_ee_rectifier_power_dissipated.Rectifier.D6.power_dissipated.series.values);

diodes = {'D1';'D2';'D3';'D4';'D5';'D6'};
PowerMax = [pd_D1_max;pd_D2_max;pd_D3_max;pd_D4_max;pd_D5_max;pd_D6_max];

T = table(PowerMax,'RowNames', diodes)
```

```
T =

  6×1 table

          PowerMax
          ________

    D1    166.45
    D2    166.45
    D3    166.45
    D4    166.45
    D5    166.45
    D6    166.45
```

The maximum instantaneous power dissipation for diodes D3 and D6 is the same as the maximum instantaneous power dissipation for the other diodes.

## See Also

### Functions

`ee_getEfficiency` | `ee_getPowerLossSummary` | `ee_getPowerLossTimeSeries`

## Related Examples

- “Power-Loss Analysis of a Three-Phase Rectifier” on page 8-897
- “Examine the Simulation Data Logging Configuration of a Model” on page 7-15
- “Data Logging”
- “About the Simscape Results Explorer”

# Choose a Simscape Electrical Function for an Offline Harmonic Analysis

| In this section... |
| --- |
| “Harmonic Distortion” on page 7-24 |
| “Harmonic Analysis Functions” on page 7-24 |
| “Evaluate Relative Overall Harmonic Distortion” on page 7-25 |
| “Compare Harmonic Distortion to Standard Limits” on page 7-25 |
| “Minimize Harmonic Distortion with Passive Filters” on page 7-25 |
| “Verify the Results of an Online Harmonic Analysis” on page 7-26 |

## Harmonic Distortion

Nonlinear loads create power distortion in the form of harmonics, that is, voltages and currents that are multiples of the fundamental frequency. Harmonic waveforms can result in energy losses though heat dissipation and in reduced power quality. They can also cause equipment to malfunction or to become damaged. Standards development organizations such as the Institute of Electrical and Electronics Engineers (IEEE) and the International Electrotechnical Commission (IEC) define the recommended limits for harmonic content in electric power systems.

## Harmonic Analysis Functions

You can use the simulation and analysis functions in Simscape Electrical to perform an offline, that is post-simulation, analysis to examine harmonic distortion in your model. The `ee_plotHarmonics` function generates a bar chart. The `ee_getHarmonics` and `ee_calculateThdPercent` functions provide harmonic data in numerical form.

To decide which functions and workflows to use for your harmonic analysis, consider your goals. The table cross-references the harmonic functions with common harmonic analysis according to the data the function outputs and the task requires.

| Goal | `ee_plotHarmonics` | `ee_getHarmonics` | `ee_calculateThdPercent` |
| --- | --- | --- | --- |
| Evaluate the relative overall harmonic distortion | • Bar chart of the percentage of fundamental magnitude<br>• Fundamental peak value<br>• Total harmonic distortion (THD) percentage | | |

| Goal | ee_plotHarmonics | ee_getHarmonics | ee_calculateThdPercent |
|---|---|---|---|
| Compare the harmonic distortion to standard limits | | • Fundamental frequency<br>• Harmonic orders<br>• Harmonic magnitudes | Total harmonic distortion (THD) percentage |
| Determine the parameters for filtering harmonic distortion | | • Fundamental frequency<br>• Harmonic orders<br>• Harmonic magnitudes | |

## Evaluate Relative Overall Harmonic Distortion

Use this workflow for a high-level understanding of the waveform distortion in your power system.

1. Enable Simscape data logging.
2. Save the logged voltage or current data to a variable.
3. Use the `ee_plotHarmonics` function to generate a bar chart of harmonic percentages with the peak fundamental magnitude and the total harmonic distortion (THD) percentage displayed in the plot title.

## Compare Harmonic Distortion to Standard Limits

Use this workflow to obtain values for evaluating the IEEE or IEC suitability of your power system.

1. Enable Simscape data logging.
2. Save the logged voltage or current data to a variable.
3. Use the `ee_getHarmonics` function to obtain the harmonic orders, the magnitude for each order, and the fundamental frequency.
4. Save the fundamental peak to a new variable.
5. Calculate the RMS voltage or current for each order.
6. Calculate the harmonic distortion percentage for individual harmonics.
7. Use the `ee_calculateThdPercent` function to obtain the total harmonic distortion (THD).
8. Compare the percentage data for each order and the THD percentage to the standard limits.

## Minimize Harmonic Distortion with Passive Filters

Use this workflow to determine the parameters for filtering the distorted waveforms with passive filters. Use individual, series-tuned filters for specific harmonic orders. Use a single high-pass filter to filter higher orders.

1. Enable Simscape data logging.
2. Save the logged voltage or current data in a variable.
3. Use the `ee_getHarmonics` function to obtain the harmonic orders, the magnitude for each order, and the fundamental frequency.

4. Identify the harmonic orders that you want to filter.
5. For each filter:
    a. Specify the filter size, in terms of reactive power compensation, and specify the filter quality.
    b. Calculate the capacitor reactance at the tuned harmonic order.
    c. Calculate the filter capacitance.
    d. Calculate the inductor reactance at the tuned harmonic order.
    e. Calculate the filter inductance.
    f. Calculate the filter resistance.

## Verify the Results of an Online Harmonic Analysis

You can examine harmonic distortion in your model online, that is during simulation, using the Simscape Spectrum Analyzer block. To verify the results from the Spectrum Analyzer block:

1. To determine the THD in your model, perform an online analysis. For information, see “Perform an Online Harmonic Analysis Using the Simscape Spectrum Analyzer Block” on page 7-27.
2. Use the `ee_getHarmonics` and `ee_calculateThdPercent` functions to determine the THD in your model.
3. Compare the THD values for the online and offline analyses. If the results differ, reconfigure the Spectrum Analyzer block.

## See Also

**Blocks**
Spectrum Analyzer

**Functions**
`ee_calculateThdPercent` | `ee_getHarmonics` | `ee_plotHarmonics`

## Related Examples

- “Harmonic Analysis of a Three-Phase Rectifier” on page 8-821
- “Perform an Online Harmonic Analysis Using the Simscape Spectrum Analyzer Block” on page 7-27
- “Data Logging”

# Perform an Online Harmonic Analysis Using the Simscape Spectrum Analyzer Block

| In this section... |
| --- |
| “Harmonic Distortion” on page 7-27 |
| “Prerequisite” on page 7-27 |
| “Perform an Offline Harmonic Analysis” on page 7-27 |
| “Perform an Online Harmonic Analysis” on page 7-30 |

## Harmonic Distortion

Nonlinear loads create power distortion in the form of harmonics, that is, voltages and currents that are multiples of the fundamental frequency. Harmonic waveforms can result in energy losses through heat dissipation and in reduced power quality. They can also cause equipment to malfunction or to become damaged. Standards development organizations such as the Institute of Electrical and Electronics Engineers (IEEE) and the International Electrotechnical Commission (IEC) define the recommended limits for harmonic content in electric power systems.

This example shows how to examine harmonic distortion in your model using offline, that is after simulation, and online, that is during simulation, analyses. The offline analysis uses the Simscape Electrical harmonic analysis functions and helps you to determine configuration settings for, and verify the results of, the online analysis. The online analysis uses the Simscape Spectrum Analyzer block.

## Prerequisite

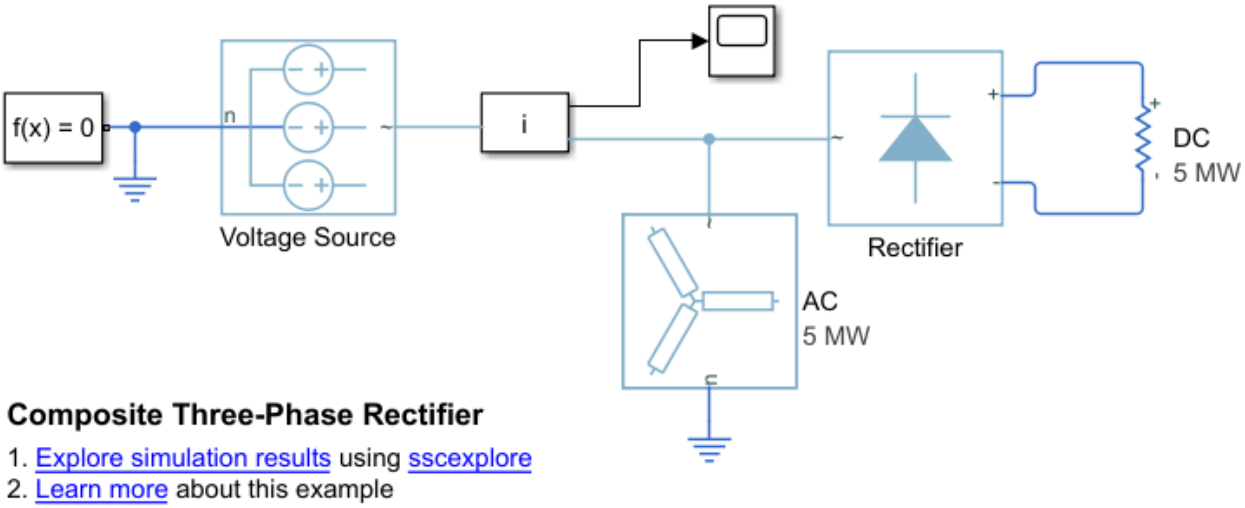
This example requires a simulation log variable in your MATLAB workspace. The model in this example is configured to log Simscape data for the whole model for the entire simulation time.

To learn how to determine if a model is configured to log simulation data, see “Examine the Simulation Data Logging Configuration of a Model” on page 7-15.

## Perform an Offline Harmonic Analysis

**1** Open the model. At the MATLAB command prompt, enter:

```
model = 'ee_composite_rectifier';
open_system(model)
```

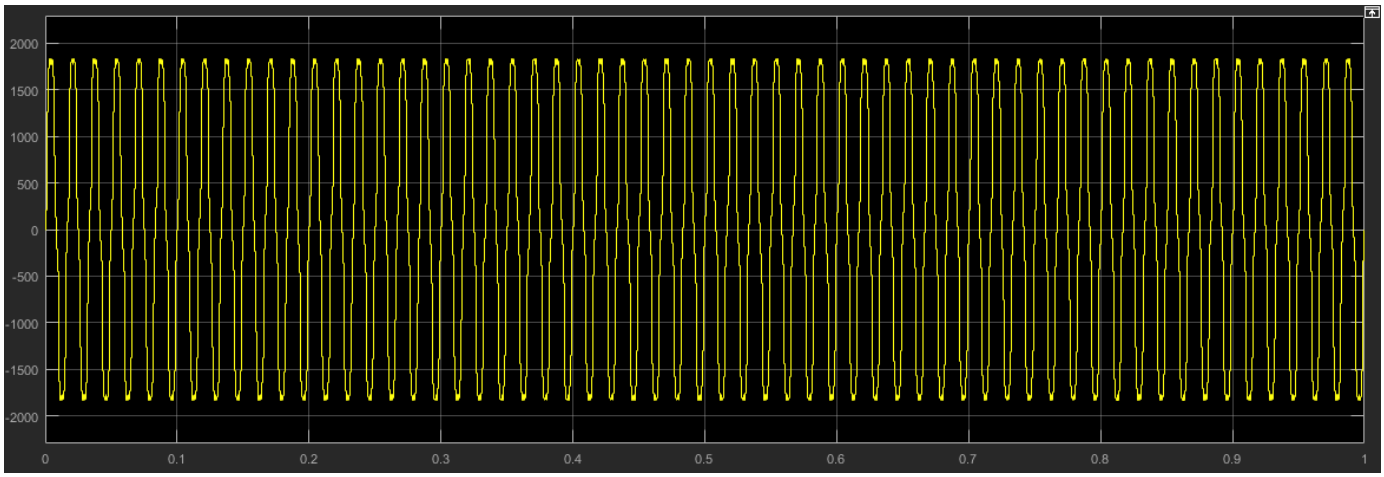



The example model contains a three-phase rectifier. The model also contains a Selector block that outputs only the *a*-phase from three-phase current signal that it receives from the PS-Simulink Converter block.

**2** Simulate the model.

```
sim(model)
```

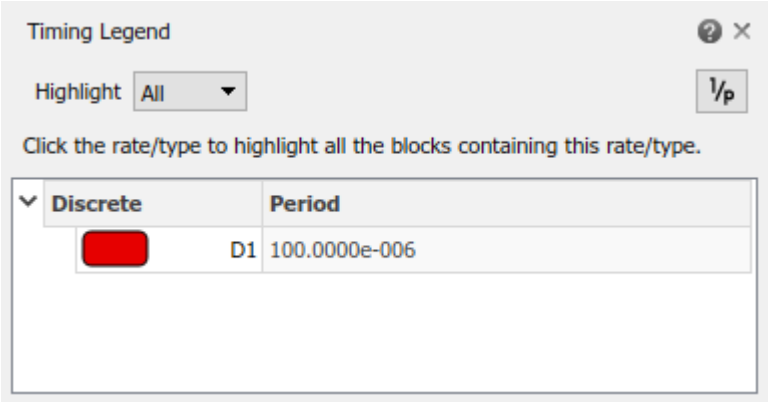
**3** View the time-domain results. Open the Scope block.




The time domain analysis shows that the rectifier is converting the voltage, but it does not include any information about the frequencies in the signal.

**4** Determine configuration settings and calculate the expected results for an online harmonic analysis. Perform an offline harmonic analysis.

**a** The Simscape Electrical harmonic analysis functions require that you use a fixed-step solver. Determine the solver type and sample time for the model. To turn on sample-time highlighting, in the Simulink editor menu bar, select **Debug > Information Overlays > Sample Time > Colors**.




The model is running at a discrete rate, therefore it is using a fixed-step solver, with a sample time of 1e-4 s.

**b** Use the `ee_getHarmonics` function to calculate the harmonic order, the harmonic magnitude, and the fundamental frequency based on the voltage source currents.

```
[harmonicOrder,harmonicMagnitude,fundamentalFrequency] = ...
ee_getHarmonics(simlog_ee_composite_rectifier.Voltage_Source.I);
```

**c** Performing an online harmonic analysis using the Spectrum Analyzer block requires that you specify a value for maximum harmonic order and the resolution bandwidth (RBW). The RBW depends on the fundamental frequency.

Extract and display the maximum harmonic order and the fundamental frequency:

```
disp(['Maximum Harmonic Order = ', num2str(max(harmonicOrder))])
disp(['Fundamental Frequency  = ', num2str(fundamentalFrequency)])
```

```
Maximum Harmonic Order = 30
Fundamental Frequency  = 60
```

**d** Determine the peak value of the fundamental frequency. This value is useful for filtering out negligible harmonics and for verifying the results of the offline analyses.

```
fundamentalPeak = harmonicMagnitude(harmonicOrder==1);
disp(['Peak value of fundamental = ', num2str(fundamentalPeak),' A']);
```

```
Peak value of fundamental = 1945.806 A
```

**e** Filter out small harmonics by identifying and keeping harmonics that are greater than one thousandth of the fundamental peak frequency.

```
threshold = fundamentalPeak ./ 1e3;
aboveThresold = harmonicMagnitude > threshold;
harmonicOrder = harmonicOrder(aboveThresold)';
harmonicMagnitude = harmonicMagnitude(aboveThresold)';
```

**f** Display the harmonic data in a MATLAB table.

```
harmonicRms = harmonicMagnitude./sqrt(2);
harmonicPct = 100.*harmonicMagnitude./harmonicMagnitude(harmonicOrder == 1);
harmonicTable = table(harmonicOrder,...
    harmonicMagnitude,...
    harmonicRms,...
    harmonicPct,...
    'VariableNames',{'Order','Magnitude','RMS','Percentage'});
display(harmonicTable);
```

```
harmonicTable =

  10×4 table

    Order    Magnitude     RMS      Percentage
    _____    _________    ______    __________

     1       1945.8       1375.9       100
     5       218.86       154.75    11.248
     7       105.83       74.835     5.439
    11       85.135         60.2    4.3753
    13       57.599       40.729    2.9602
    17       50.417        35.65    2.5911
    19       37.612       26.596     1.933
    23       33.859       23.942    1.7401
    25       26.507       18.743    1.3622
    29       23.979       16.955    1.2323
```

**g** Calculate the total harmonic distortion (THD) percentage using the `ee_calculate_ThdPercent` function.

```
thdPercent = ee_calculateThdPercent(harmonicOrder,harmonicMagnitude);
disp(['Total Harmonic Distortion Percentage = ' num2str(thdPercent),' %']);
```

```
Total Harmonic Distortion percentage = 14.1721 %
```

## Perform an Online Harmonic Analysis

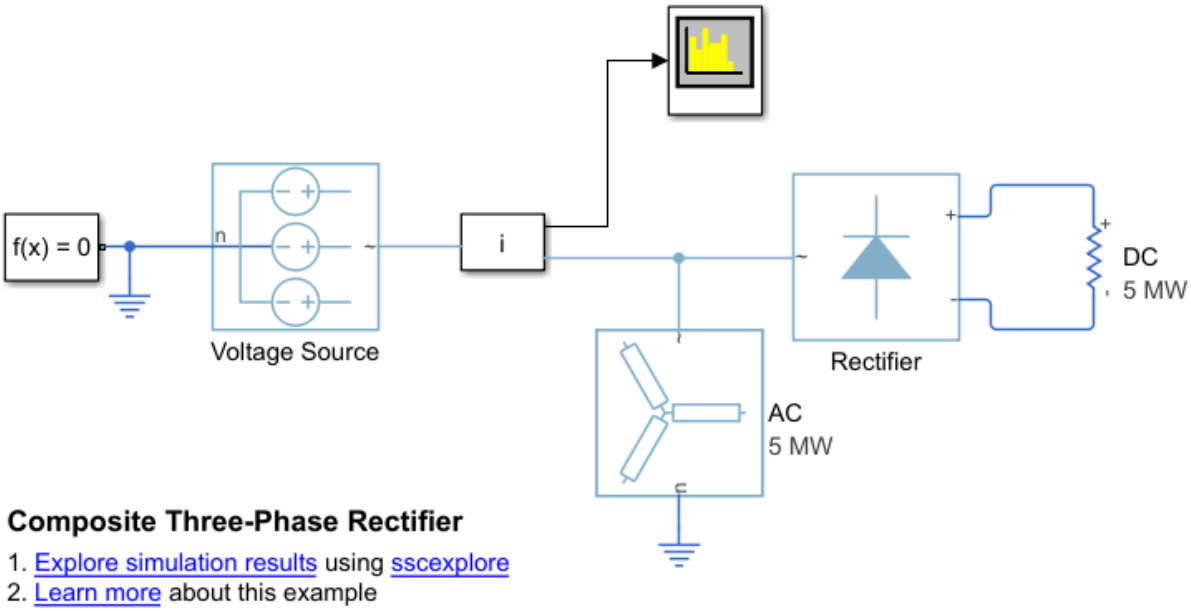
**1** In the Simulink editor that contains the `ee_composite_rectifier` model, replace the Scope block with a Spectrum Analyzer block from the Simscape Utilities Library:

**a** Delete the Scope block.

**b** Left-click within the block diagram.

**c** After the search icon appears, type `spec`, and then from the list, select the `Spectrum Analyzer` from the `Utilities` library.

**d** Connect the Spectrum Analyzer block to the output signal from the Subsystem i.




2 Configure the Spectrum Analyzer block using the Spectrum Settings panel.

**a** Open the Spectrum Analyzer.

**b** Open the Spectrum Settings panel. On the Spectrum Analyzer toolbar, click the **Spectrum Settings** button.

**c** Configure the parameters on the **Main Options** pane.

**i** Configure the block to display the root mean square (RMS) of the frequency. From the **Type** dropdown menu, select `RMS`.

**ii** Determine the value to specify for the resolution bandwidth (RBW) using this equation:

$$RBW = \frac{NENBW * f}{N},$$

where,

- *NENBW* is the normalized effective noise bandwidth, a factor of the windowing method used. The Hanning (Hann) window has an *NENBW* value of approximately 1.5.
- *f* is the fundamental frequency.
- *N* is the number of periods.
- *RBW* is the resolution bandwidth in Hz.

For a fundamental frequency of 60 Hz over 10 periods, using a Hann window,

$$RBW = \frac{1.5 * 60Hz}{10} = 9Hz$$

For **RBW (Hz)**, specify 9.

**d** Expand the **Windows Options** pane and specify an **Overlap (%)** of `90`.

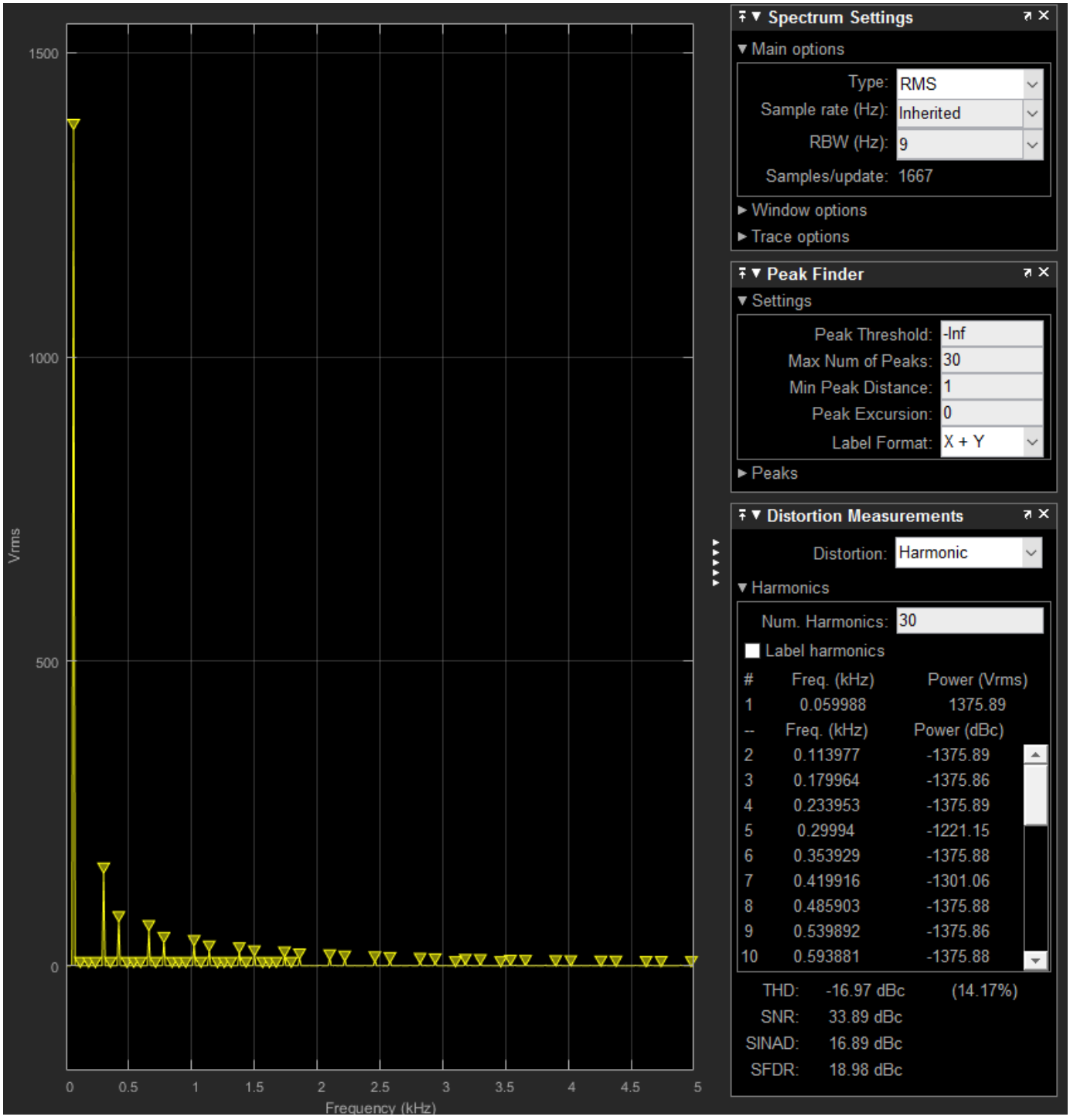
e Specify the maximum number of peaks for the analyzer to display. In the menu bar, select **Tools** > **Measurements** > **Peak Finder**. Alternatively, in the Spectrum Analyzer toolbar, select the Peak Finder button. In the **Peakfinder** pane, in the **Settings** section, for **Max Num of Peaks**, enter `30`. This value is based on the maximum harmonic order as indicated by the offline analysis.

f Set the number of harmonics to use for measuring harmonic distortion. Specify a number that captures the largest harmonic order that the offline analysis captures. In the menu bar, select **Tools** > **Measurements** > **Distortion Measurements**. Alternatively, in the Scope toolbar, click the Distortion Measurements button. Scroll as required to see the **Distortion Measurements** pane.

   In the **Distortion Measurements** pane, for **Num Harmonics**, again enter `30`.

3 Simulate the model.

```
sim(model)
```




The THD percentage is 14.17% and the fundamental peak power is 1375.89 Vrms at 0.06 kHz (60 Hz). These results agree with the results from the offline harmonic analysis.

## See Also

**Blocks**
PS-Simulink Converter | Spectrum Analyzer | Selector

**Functions**
`ee_calculateThdPercent` | `ee_plotHarmonics` | `ee_getHarmonics`

## Related Examples

- “Harmonic Analysis of a Three-Phase Rectifier” on page 8-821
- “Choose a Simscape Electrical Function for an Offline Harmonic Analysis” on page 7-24
- “Data Logging”

# Perform a Load-Flow Analysis Using Simscape Electrical

**In this section...**

"Network Requirements for a Simscape Electrical Load-Flow Analysis" on page 7-35
"Essential Blocks for a Load-Flow Analysis" on page 7-36
"Performing a Load-Flow Analysis" on page 7-36
"Machine Parameterization and Variable Initialization" on page 7-38
"Load-Flow Analyzer App" on page 7-38
"Troubleshooting Load-Flow Analysis and Initialization Issues" on page 7-39

Simscape Electrical can perform a power-flow, or load-flow, analysis for an AC, DC, or mixed AC and DC electrical power transmission system modeled using the Simscape three-phase electrical domain. A load-flow analysis allows you to determine the voltage magnitudes, voltage phase angles, active power, and reactive power of the electrical system in steady-state operation.

For a given steady-state operating point, the load-flow data reveals the:

- Voltage magnitude and voltage phase angle at each bus
- Active and reactive power generation for each generator that supplies the grid
- Active and reactive power that flows to each load that places demand on the grid

You can use the data to determine ideal operating conditions or estimate the response of your system to hypothetical situations. For example, if you know the active and reactive power in each transmission line, you can determine if the remaining lines can handle the extra load that occurs when one or more transmission lines go offline.

You can also use the data to calculate transmission line or system losses and examine the overall voltage profile of the network. Investigating these attributes can help you determine if the system needs reactive power compensation to overcome low voltage levels.

## Network Requirements for a Simscape Electrical Load-Flow Analysis

To determine the steady-state load-flow solution for a three-phase network using Simscape Electrical, your model must be:

- Compatible with and configured for the Simscape frequency and time simulation mode. For more information, see "Frequency and Time Simulation Mode" and Solver Configuration.
- Load balanced. The level of approximation of the load-flow analysis depends on how balanced the system is and the level of harmonics that are present.
- Enabled for Simscape data logging. For complex models or long simulation runs, you can improve simulation performance by enabling data logging for selected blocks by using local solver settings. For a load-flow analysis, data logging is required only for Busbar blocks. For more information, see "Enable Data Logging for the Whole Model" and "Log Data for Selected Blocks Only".

## Essential Blocks for a Load-Flow Analysis

### Bus Bar Connectors

In an electrical transmission system, a bus bar connector, or *bus*, is a vertical line that connects power components such as generators, loads, and transformers. To represent buses, the **Simscape > Electrical > Connectors & References** library provides the Busbar and Busbar (DC) blocks.

### Three-Phase Voltage Sources

You need to select the right three-phase voltage source for your model to conduct a load-flow analysis. Which source you choose depends on whether you want to prioritize simulation accuracy or performance. The balance between simulation accuracy and performance depends, in part, on the blocks that you use to represent the voltage sources in your analysis model. Simulation accuracy is a measure of model fidelity, that is, how closely the simulation results agree with mathematical and empirical models. As model fidelity increases, so does the computational cost of simulation. As computational cost increases, simulation speed, decreases. Conversely, as model fidelity decreases, simulation speed increases.

#### Prioritize Model Fidelity by Using Machine Blocks

To prioritize model fidelity over simulation speed, represent voltage sources by using induction or synchronous machine blocks. For modeling induction machines, the **Simscape > Electrical > Electromechanical > Asynchronous Machines** library provides both the Induction Machine Squirrel Cage and Induction Machine Wound Rotor blocks. For modeling synchronous machines, the **Simscape > Electrical > Electromechanical > Synchronous Machines** library provides the Synchronous Machine Model 2.1, Synchronous Machine Round Rotor, and Synchronous Machine Salient Pole blocks.

#### Prioritize Simulation Speed by Using Load Flow Source Blocks

For a faster simulating, but lower fidelity model, represent the voltage sources in your analysis model by using a Load Flow Source block from the **Simscape > Electrical > Sources** library. The Load Flow Source block supplies either an idealized or a current-dependent voltage source. The voltage can contain series impedance or can act as a source for a swing, PV, or PQ bus.

## Performing a Load-Flow Analysis

To examine load-flow data for a three-phase Simscape Electrical transmission system model that is compatible with frequency-time simulation mode:

1. Enable Simscape data logging.
2. Parameterize the voltage sources.

   At the beginning of a load-flow analysis, the equation variables for transmission line losses are unknown. While the unknown variables are being solved, the buses balance the losses by providing or absorbing active and reactive power. For each bus there are four variables:

   - $P$ — Active power
   - $Q$ — Reactive power
   - $V$ — Voltage
   - $\theta$ — Phase angle

Two of the variables are known and two are unknown. Which variables are known and which are unknown depends on the actively controlled three-phase sources and loads that are connected to the bus bar. The voltage source block configurations determine which bus types are used in load-flow analysis. You can include more than one bus type in your model. Bus type options are:

- Swing bus — A swing, slack, or reference, bus balances the active and reactive power in a system. The slack bus serves as an angular reference for other buses in the system. The phase angle of a swing bus is 0° and the voltage magnitude is specified. A typical value is `1 pu`. At the beginning of the load-flow analysis, $P$ and $Q$ are the unknown variables for this bus.
- PV bus — A PV (or generator) bus balances the active and reactive power in a system by supplying a constant, active power and voltage. At the beginning of the load-flow analysis, $\theta$ and $Q$ are the unknown variables for this bus.
- PQ bus — A PQ (or load) bus determines the amount of active and reactive power that is consumed. At the beginning of the load-flow analysis, $V$ and $\theta$ are the unknown variables for this bus.

If your model contains one or more:

- Load Flow Source blocks — For each block, for the **Source type** parameter, set the bus type to one of these options:

  - `Swing bus`
  - `PV bus`
  - `PQ bus`

  Specify the related parameters, which differ depending on which bus type you choose.

  To avoid a simulation issue due to a nonoptimal minimum for PV or PQ buses, in the **Expected Ranges** settings, specify minimum and maximum values for the **Internal source phase search range** parameter.
- Induction machine blocks — For each block, specify the priority and beginning values for the block using the **Variables** settings. In the **Main** settings, set the **Initialization option** parameter to `Set targets for load flow variables`. In the **Variables** settings, select a **Priority** and specify a **Beginning Value** for:

  - **Slip**
  - **Real power generated**
  - **Mechanical power consumed**

  For more on information on setting initial target values by using the **Variables** settings, see “Set Priority and Initial Target for Block Variables”.

  To fully specify the initial condition, you must include an initialization constraint in the form of a high-priority target value. For example, if your induction machine is connected to an Inertia block, the initial condition for the induction machine is completely specified if, in the **Variables** settings of the Inertia block, the **Priority** for **Rotational velocity** is set to `High`. Alternatively, you could set the **Priority** to `None` for the Inertia block **Rotational velocity**, and instead set the **Priority** for the induction machine block **Slip**, **Real power generated**, or **Mechanical power consumed** to `High`.
- Synchronous machine blocks — For each block, specify the bus type and beginning values using the **Initial Conditions** settings. The available parameter targets depend on whether the block is configured for a swing, PV, or PQ bus. In the **Initial Conditions** settings:

   a. Set the **Initialization option** parameter to `Set targets for load flow variables`.
   b. Select a bus for the **Source type** parameter.
   c. Specify values for the related bus parameter.
   d. To avoid a simulation issue due to a nonoptimal minimum, in the **Expected ranges** settings, specify minimum and maximum values for the **Internal source phase search range** parameter.

3. Configure each Busbar and Busbar (DC) block:
   a. Set the **Number of connections** to 2, 3, or 4.
   b. In the Busbar blocks, specify the voltage and frequency to match the specified values of the connected blocks. In the Busbar (DC) blocks, you only need to specify the rated DC voltage.
   c. To view the load-flow data using a Scope block, expose the optional measurement ports on the Busbar block:
      - To expose ports **Vt** and **ph**, set **Measurement ports** to `Yes`.
      - To expose ports **P** and **Q**, set **Measurement ports** to `Yes`.

      Connect the Busbar and Scope blocks.
4. Configure the Solver Configuration block. Set **Equation formulation** to `Frequency and time`.
5. Simulate the model.

After simulating, you can view the load-flow results in the Busbar and Busbar (DC) blocks annotation and in the Simscape logging data that the model outputs to the MATLAB workspace.

For examples that show how to perform a load-flow analysis, see:

- 2-Bus Loadflow on page 8-407
- IEEE 9-Bus Loadflow on page 8-405
- “Mixed AC/DC System Loadflow” on page 8-157

## Machine Parameterization and Variable Initialization

You can use the data from your load-flow analysis to correctly initialize three-phase induction and synchronous machine blocks. For examples, see:

- Induction Motor Initialization with Loadflow on page 8-409
- Synchronous Machine Initialization with Loadflow on page 8-411

## Load-Flow Analyzer App

If your model is configured for a power-flow analysis, you can also use the **Load-Flow Analyzer** to perform a power-flow, or load-flow, analysis for a three-phase AC or DC electrical power transmission system. The app generates three tables. The AC Nodes table contains data for the AC network nodes, as represented by AC busbar, load flow source, synchronous machine, induction machine, and three-phase load blocks. The AC & DC Busbars table contains data for all busbars. The AC Connections table contains data for the network connections, as represented by transmission line and transformer blocks. When you open the app, the tables are preloaded with the specified parameter values for the relevant blocks in the current or specified model. After you run the power-flow analysis, the tables

also display the steady-state voltage magnitudes, voltage phase angles, active power, and reactive power for the node and connection blocks.

The **Load-Flow Analyzer** app allows you to:

- Run a load-flow analysis.
- Highlight and update load-flow input block parameter values for busbar, load flow source, synchronous machine, induction machine, and three-phase load blocks.
- Change the bus type of load flow source, synchronous machine, and induction machine blocks.
- Select and highlight node and connection blocks in the model.
- Sort columns in the tables by increasing or decreasing values.
- Export the data to a spreadsheet, a MAT-file, or comma-separated variable (CSV) files.

## Troubleshooting Load-Flow Analysis and Initialization Issues

If you encounter issues when simulating a load-flow model, apply these troubleshooting measures. Testing your load-flow model incrementally can help you avoid specifying nonphysical load-flow requirements.

### Internal Load-Flow Source Impedance

Including internal source impedance for a Load Flow Source block when the **Source type** parameter of the block is set to `Swing bus`, `PV bus`, or `PQ bus` can prevent initialization convergence. To resolve any convergence issues, use one of these methods:

- Limit the solution range by specifying a value for the **Internal source phase search range** parameter.
- Neglect source impedance.
- Model the impedance externally from the Load Flow Source block.

### Field-Circuit Transient or Initial Rotor Acceleration

If you initialize a synchronous machine block for a load-flow analysis, the block solves all Park-transformed flux variables and mechanical variables for steady state. However, incorrect initialization of an automatic voltage regulator (AVR) or governor can result in a field-circuit transient or an initial rotor acceleration. To resolve these issues:

1. Determine the initialization values for the torque and field voltage.
    a. Run the load-flow analysis by using approximated values for the AVR and governor and settings.
    b. Make a note of these values in the load-flow results reported by the adjacent Busbar block:
        - Voltage magnitude
        - Phase angle
        - Generated real power
        - Generated reactive power
    c. For the synchronous machine block, in the **Initial Conditions** settings, set the **Initialization option** parameter to `Set real power, reactive power, terminal voltage, and terminal phase`.

d Specify these parameters using the values from the load flow results:

- **Terminal voltage magnitude**
- **Terminal voltage angle**
- **Terminal active power**
- **Terminal reactive power**

e Print the required initial conditions for the AVR and governor to the MATLAB workspace. Right-click the machine block and, from the context menu, select **Electrical > Display Associated Initial Conditions**. The relevant data are the field circuit voltage, *si_efd0*, and the mechanical torque, *si_torque0*.

2 Specify the AVR and governor initial conditions using the calculated initial condition values.

For example, the table shows the annotated data for the Busbar block that is next to the Synchronous Machine Salient Pole block in `ee_loadflow_sm_initialization`, the model for the "Synchronous Machine Initialization with Loadflow" on page 8-411 example. If you open the Synchronous Machine Salient Pole block, click the **Initial Conditions** settings, and set the **Initialization option** parameter to `Set real power, reactive power, terminal voltage, and terminal phase`, you can observe that the specified parameter values are equal to the load-flow simulation values.

**Note** The specified parameter values have already been entered to match the load flow results for this model.

| Physical Quantity | Load-Flow Simulation Value | Synchronous Machine Block Initial Conditions Parameter Name | Synchronous Machine Block Initial Conditions Parameter Value |
|---|---|---|---|
| Voltage magnitude | 1.020 pu | **Terminal voltage magnitude** | `13.8*1.02 kV` |
| Phase angle | 0.00 deg | **Terminal voltage angle** | `0 deg` |
| Generated real power | 31.2 MW | **Terminal active power** | `31.2e6 V*A` |
| Generated reactive power | 10.4 Mvar | **Terminal reactive power** | `10.4e6 V*A` |

If you print the data to the command line, the *si_torque0* and *si_efd0* data are printed under the `Initial conditions required for steady-state (SI)`:

```
Initial conditions required for steady-state (SI):
    si_efd0 =       85.4468     : V     % Field circuit voltage
    si_ifd0 =       1168.87     : A     % Field circuit current
    si_torque0 =    828709      : Nm    % Mechanical torque
    si_Pm0 =        3.12416e+07 : W     % Mechanical power
```

To initialize correctly, specify `85.4468 V` as the value for the field voltage source, and `828709 Nm` as the value for the **Shaft torque** Constant block that is connected to the Ideal Torque Source block.

### Multiple Load-Flow Simulation Solutions

There are often multiple solutions to the set of load-flow targets specified when initializing an AC electrical network. For example, for a PV bus source where you specify the active power and voltage,

there are two solutions for the reactive power. For the desired solution, the magnitude of the reactive power is typically less than the specified active power magnitude. For the undesired solution, the reactive power magnitude is much larger than the active power magnitude.

If the initialization returns the undesired solution, reconfigure the Load Flow Source or synchronous machine block and increase the value for the minimum boundary of the **Internal source range search range** parameter. For the Load Flow Source block, the parameter is in the **Expected Ranges** settings. For synchronous machine blocks, the parameter is in the **Initial Conditions** settings.

#### Nonoptimal Local Minimum

The simulation can stop and generate an error if, to satisfy the active and reactive power demands, the optimization decreases the Busbar block voltage, to the point where the solution is closer to an undesired local minimum around zero busbar voltage than to the desired load flow solution. To prevent this type of issue, reconfigure the Load Flow Source or synchronous machine blocks and increase the value of the **Minimum voltage (pu)** parameter. For the Load Flow Source block, the parameter is in the **Expected Ranges** settings. For synchronous machine blocks, the parameter is in the **Initial Conditions** settings.

#### Frequency and Time Simulation Mode Incompatibility

You can only perform a load-flow analysis by using the frequency and time simulation mode. Replace any blocks that are not compatible with the frequency and time simulation mode. For more information, see “Frequency and Time Simulation Mode”.

## See Also

**Simscape Blocks**
Busbar | Busbar (DC) | Induction Machine Squirrel Cage | Induction Machine Wound Rotor | Load Flow Source | Synchronous Machine Model 2.1 | Synchronous Machine Round Rotor | Synchronous Machine Salient Pole

**Apps**
**Load-Flow Analyzer**

## Related Examples

- 2-Bus Loadflow on page 8-407
- IEEE 9-Bus Loadflow on page 8-405
- Induction Motor Initialization with Loadflow on page 8-409
- Synchronous Machine Initialization with Loadflow on page 8-411
- “Mixed AC/DC System Loadflow” on page 8-157

# Examples

# Peak Shaving with Battery Energy Storage System

### Introduction

This example shows how to model a battery energy storage system (BESS) controller and a battery management system (BMS) with all the necessary functions for the peak shaving. The peak shaving and BESS operation follow the IEEE Std 1547-2018 and IEEE 2030.2.1-2019 standards.

In this example, an average converter, an output filter, and associated control model the BESS. The BESS can operate in grid-forming control and it receives setpoint from the operator control room for power dispatch. The BESS also receives the powerflow measurements from point of common coupling (PCC) and changes cotrol mode for peak shaving.

### Description of BESS controller

The BESS controller receives commands and setpoint from the control room operator as well as various measurements and status from different sources and loads connected to the feeder. The BESS in this model comprises these functions:

1. Reference frequency generation
2. Reference voltage generation
3. Receive setpoint and command from operator
4. Change control mode. According to the powerflow measurement at PCC, the BESS starts peak shaving or enables to charging mode

### Implementation of Photovoltaic (PV) model

The model represents a three-phase grid-connected photovoltaic (PV) system that injects power with unity power factor (UPF) without using an intermediate DC-DC converter. The transformerless configuration simulates leakage currents. To track the maximum power point (MPP), the example uses these maximum power point tracking (MPPT) techniques:

- Incremental conductance
- Perturbation and observation

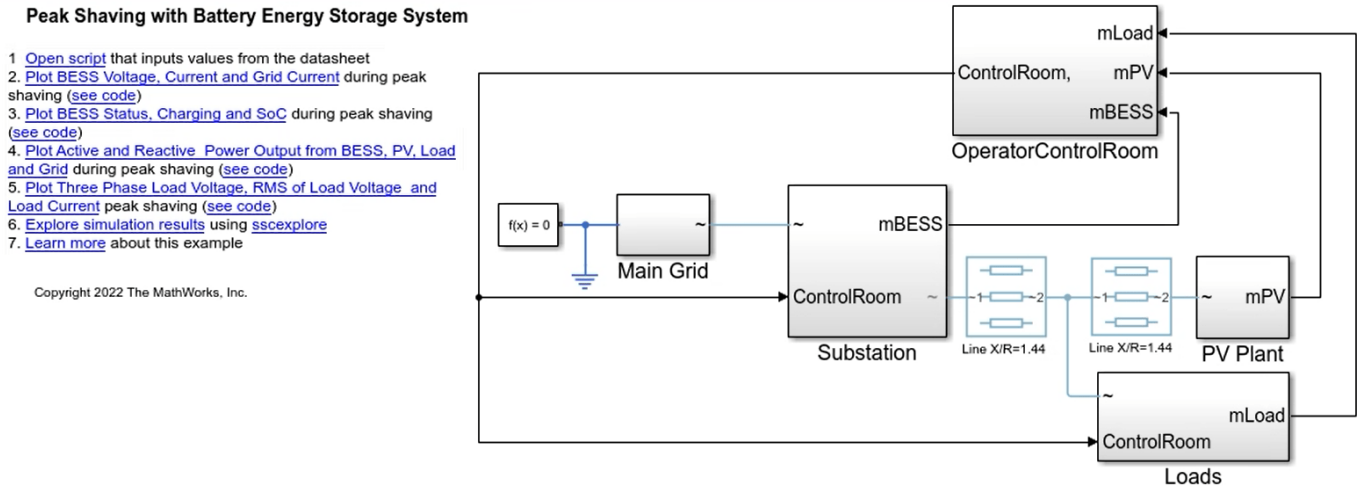
### Build Model for BESS Peak Shaving

### Model Overview

Open the model `sscv_peak_shaving.slx`.

```
mdl = "sscv_peak_shaving";
open_system(mdl)
```

The Substation subsystem connects the BESS and the feeder to the main grid. This subsystem comprises a connecting breaker, disconnectors, and transformers to connect the main grid to the BESS and the outgoing feeder. The substation also contains the BESS controller and the BMS.



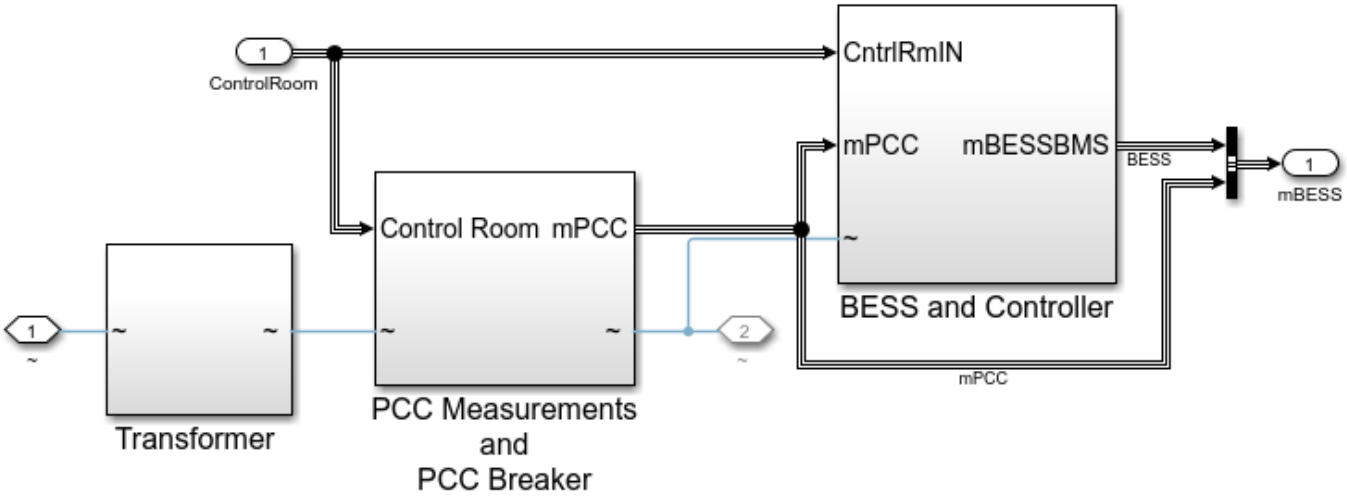

### Building Components for Peak Shaving with BESS

This example comprises these main components:

1. Substation
2. BESS System
3. Battery Management System (BMS)
4. Battery Module
5. Operator Control Room

### Substation

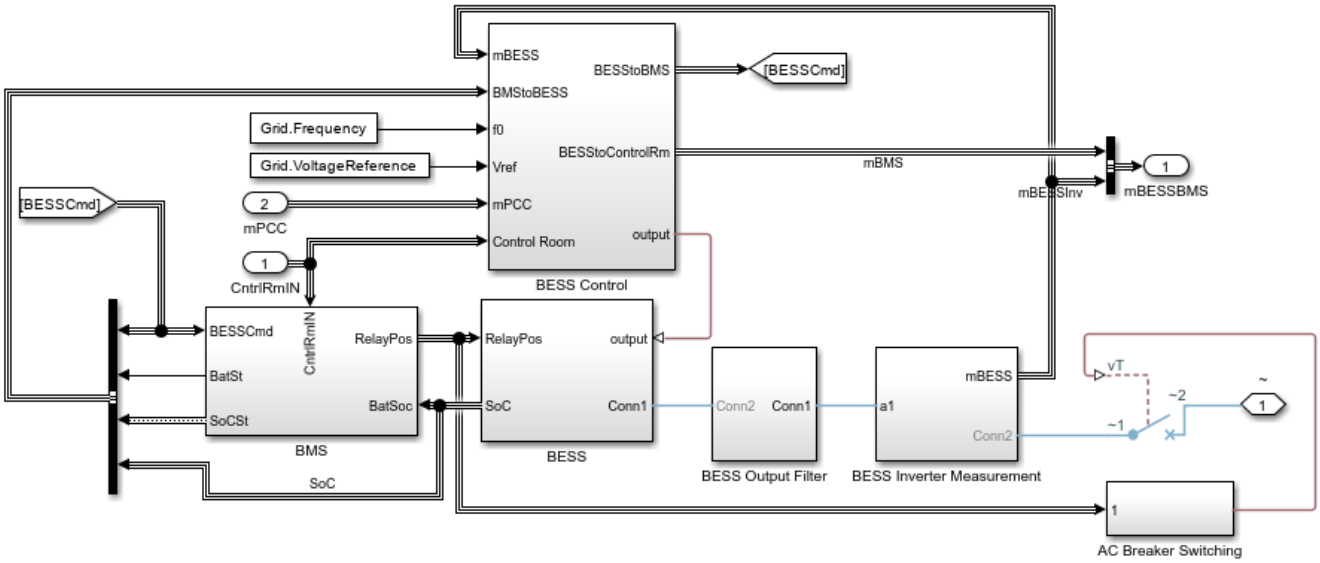
The Substation subsystem connects the BESS and the feeder to the main grid by using a connecting breaker, disconnectors, and transformers. The substation also contains the BESS controller and the BMS.



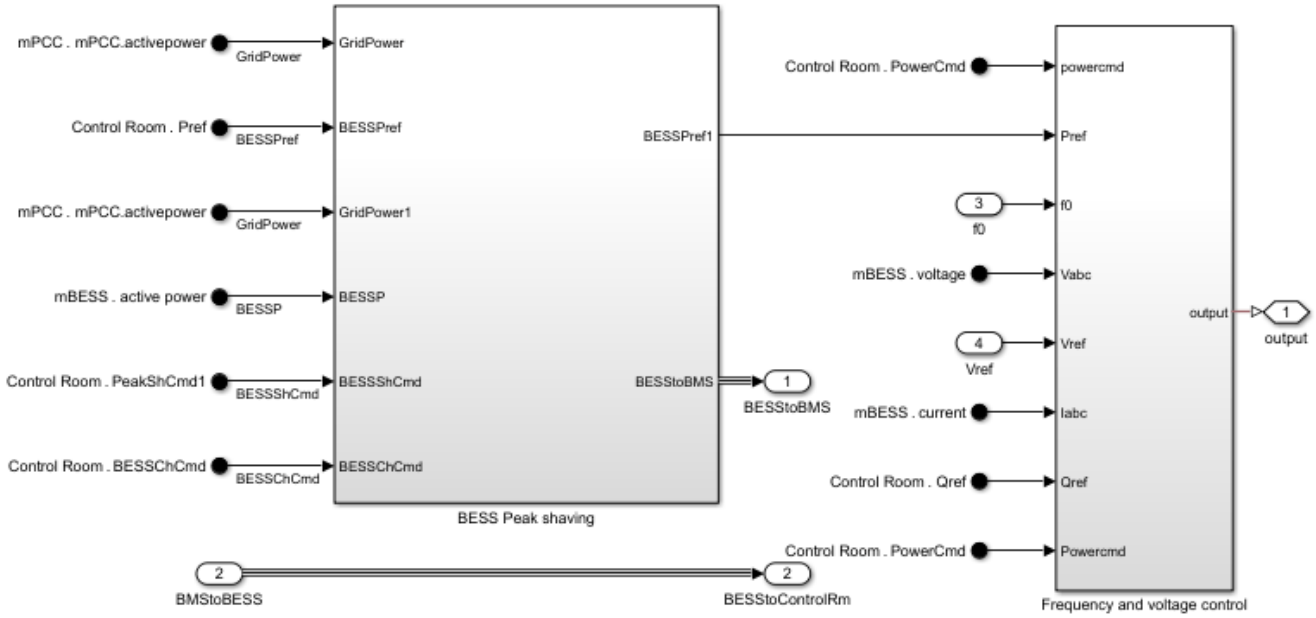
### BESS System

The BESS system comprises:

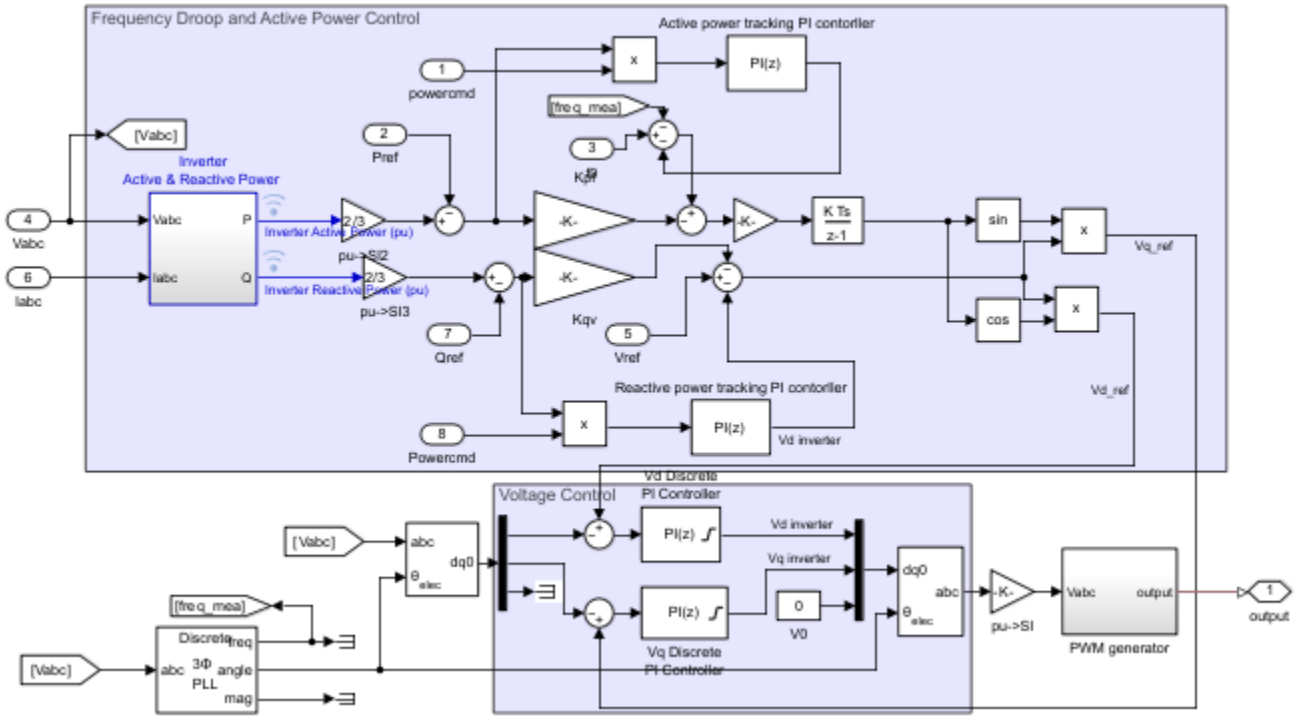
1. Grid side converter, filter, measurement, and control
2. Battery management system (BMS)
3. Battery Module




The BESS converter connects the battery modules to the grid and controls the power flow through the converter. The BESS controller implements the peak shaving function.

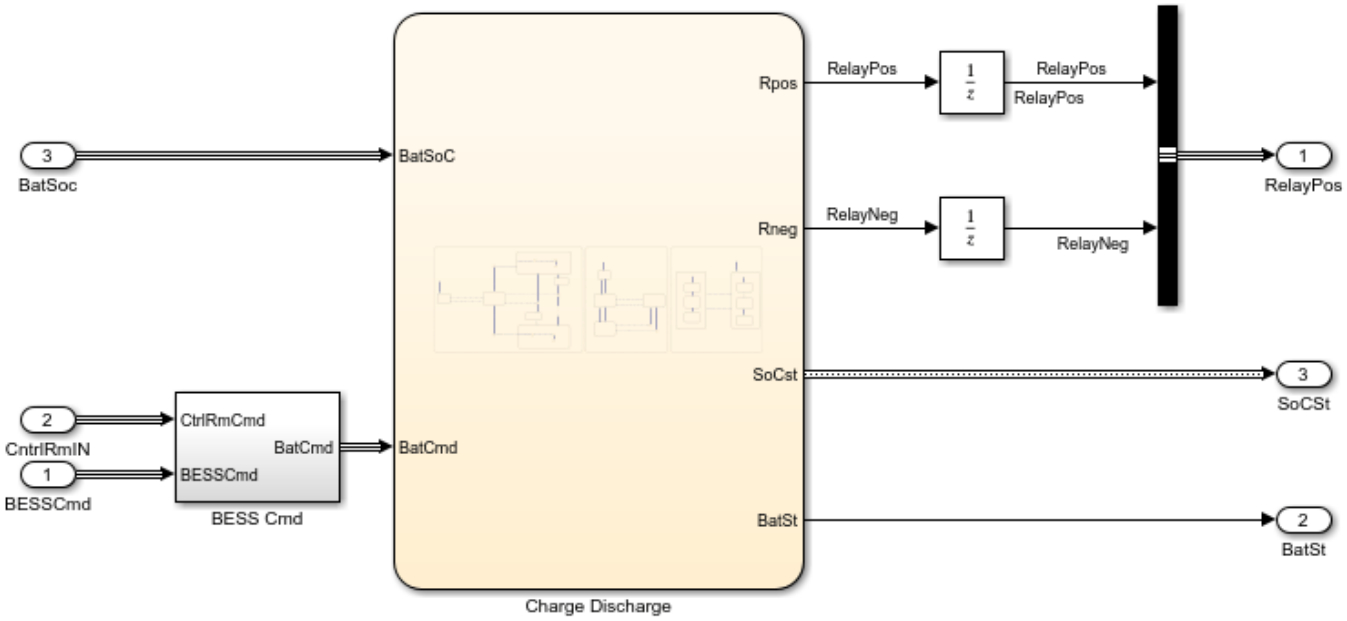


The power measurement at PCC detects high loading of the main grid at the substation and activates the peak shaving function. The peak shaving function limits the power from the main grid to the maximum rated power while the BESS system provides the rest of the power requirement.



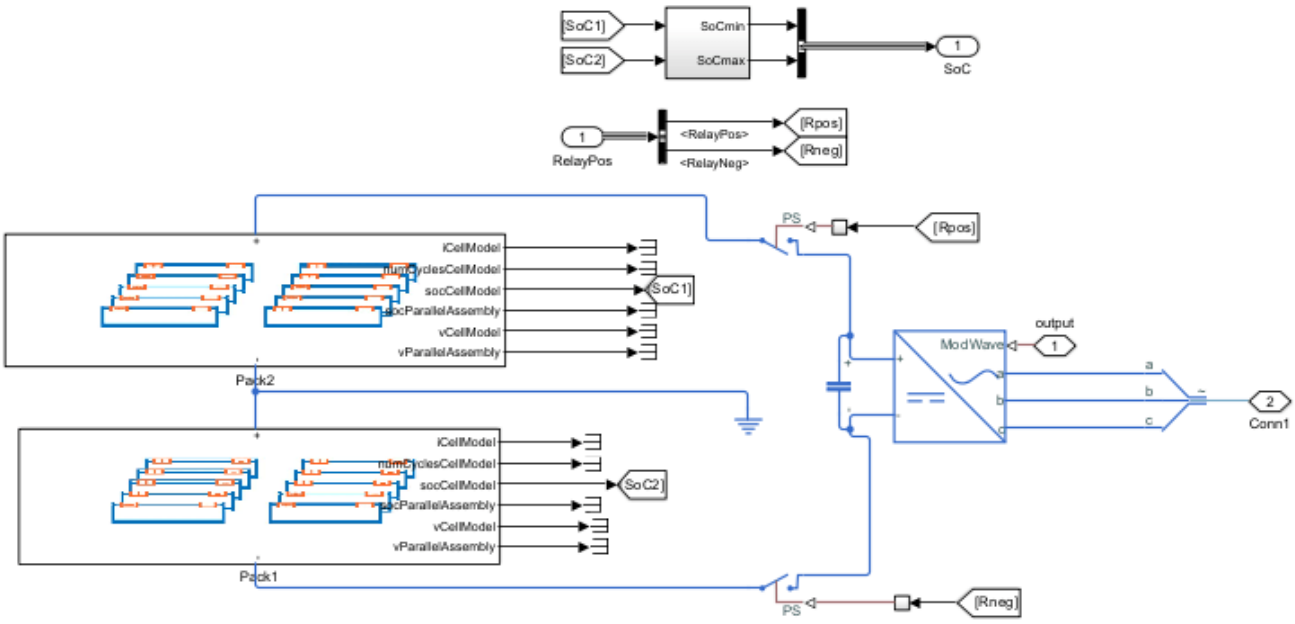

### Battery Management System (BMS)

The BMS receives the request from the grid-side converter on power requirement. The BMS also monitors the state-of-charge (SOC) of the battery module. In this example, the BMS disconnects the battery if the SOC is above the high SOC threshold and the battery is discharging. Similarly the BMS disconnects the battery if the SOC is below the low SOC threshold and the battery is charging. Once the battery opens from the DC side, the AC-side breaker also opens within one cycle.



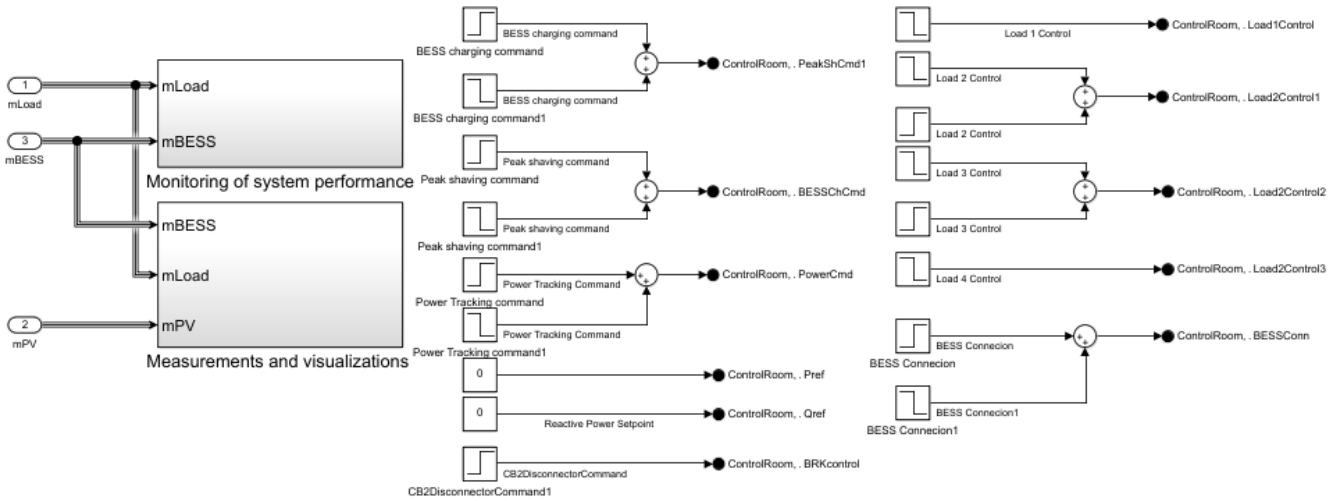

### Battery Module

The battery module is connected to the DC side of the BESS converter. Two battery packs are connected in series and grounded at the midpoint.The DC breakers can disconnect the battery module.



### Operator Control Room

The Operator Control Room subsystem sends all the setpoints and commands. It also plots the measured quantities and the system performance analysis.




### Define Parameters & Run Simulations

Initialize the BESS, grid, and PV parameters. At the MATLAB Command Window, enter:

```
run("sscv_peak_shaving_BESS_data.mlx");
```

### Initialize Battery parameters

The battery module in this example is generated by using the objects and functions in the Battery Pack Model Builder. For more information on how to build a battery pack, see the “Build a Simple Model of a Battery Pack in MATLAB and Simscape” (Simscape Battery) example.

```
run("sscv_peak_shaving_param.m");
Ns=1500/25;
Np=round(150*1000/(59*Ns*25));
load('sscv_peak_shaving_data.mat')
```

### Run Simulation

Simulate the model.

```
run("sscv_peak_shaving");
```

### Plot Simulation Results

These plots show:

1. Voltage and current of BESS.
2. Active and reactive power output of BESS, PV, load, and main grid.
3. Voltage, current, and power consumption of loads.
4. Status, discharge, charge, and SOC of BESS.

This plot shows the three-phase voltage and current output of the BESS, as well as the grid current during peak shaving and BESS disconnection.

```
run('sscv_peak_shaving_plot_BESS_VI.m')
```

The plot shows the measured values around the start of peak shaving around 3.0 s and the BESS disconnection at 4.97 s. A stable voltage and current output from BESS verifies a good peak shaving. The disconnection of BESS happens due to low SOC.

This plot shows the active and reactive power of BESS, PV, main grid, and loads.

```
run('sscv_peak_shaving_plot_PQ.m')
```

The stable active and reactive power output verifies the efficacy of the peak shaving method.

This plot shows the voltage and current at the loads.

```
run('sscv_peak_shaving_plot_Load_VI.m')
```

The load voltage and load current remain steady during peak shaving and BESS disconnection.

This plot shows the charge, discharge, BESS status, and SOC of the BESS.

```
run('sscv_peak_shaving_plot_BMS_SoC.m')
```

The discharge status during peak shaving and the disconnection of the BESS due to low SOC matches with

the results from the AC-side output. This also validates the BMS functions for BESS SOC monitoring.

**Evaluate System Performance**

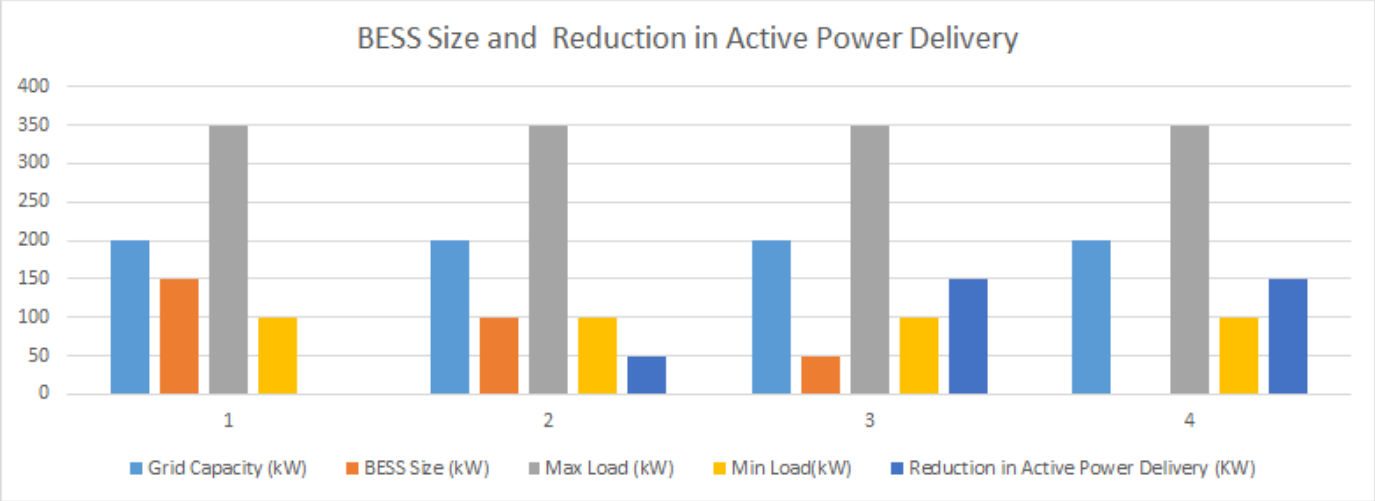
These plots show the results of the system performance and the impact of the peak shaving function.

These performance indices include:

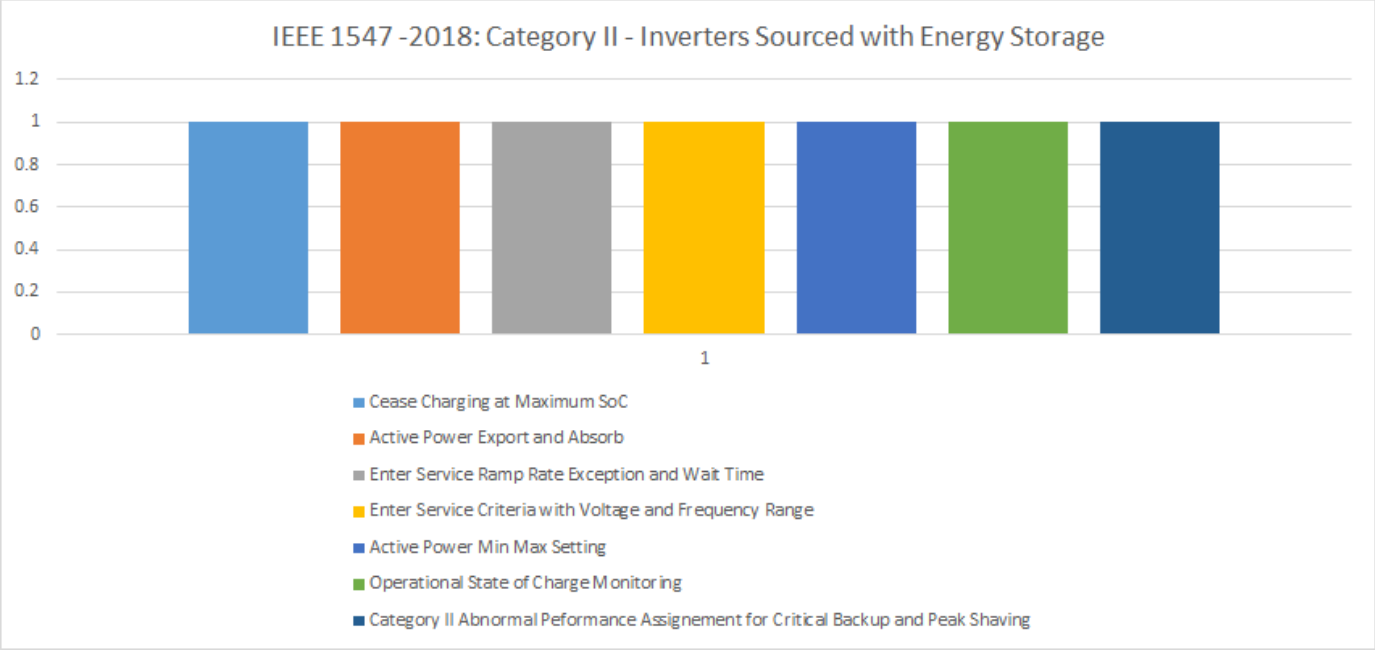
1. Active Power Delivery and BESS Sizing.
2. IEEE 1547 -2018: Category II - Inverters Sourced with Energy Storage Mapping.
3. IEEE 2030.2.1-2019 Guide for Design, Operation, and Maintenance of Battery Energy Mapping.
4. Impact of peak shaving function time delay

This plot shows the loss in active power delivery with variation in BESS sizes. The grid capacity and

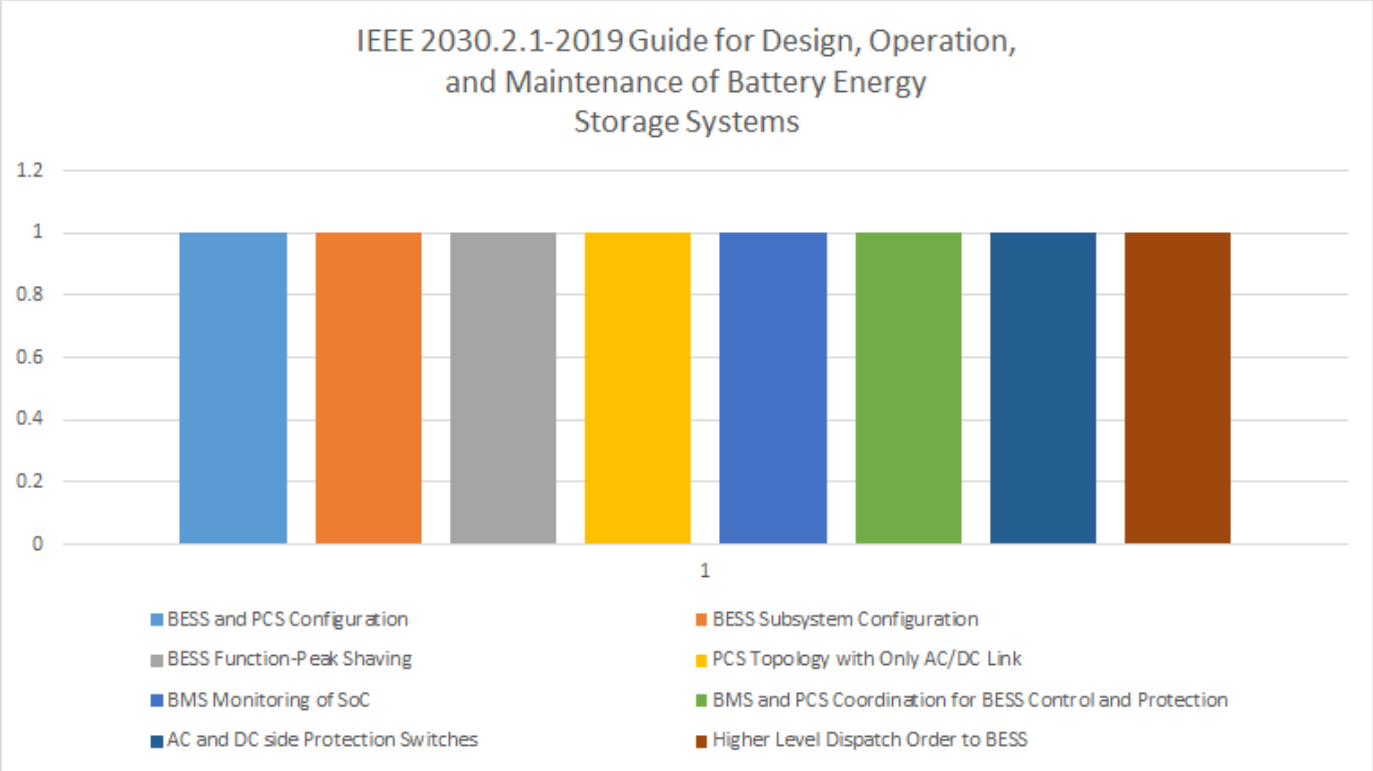
load variation are constant.



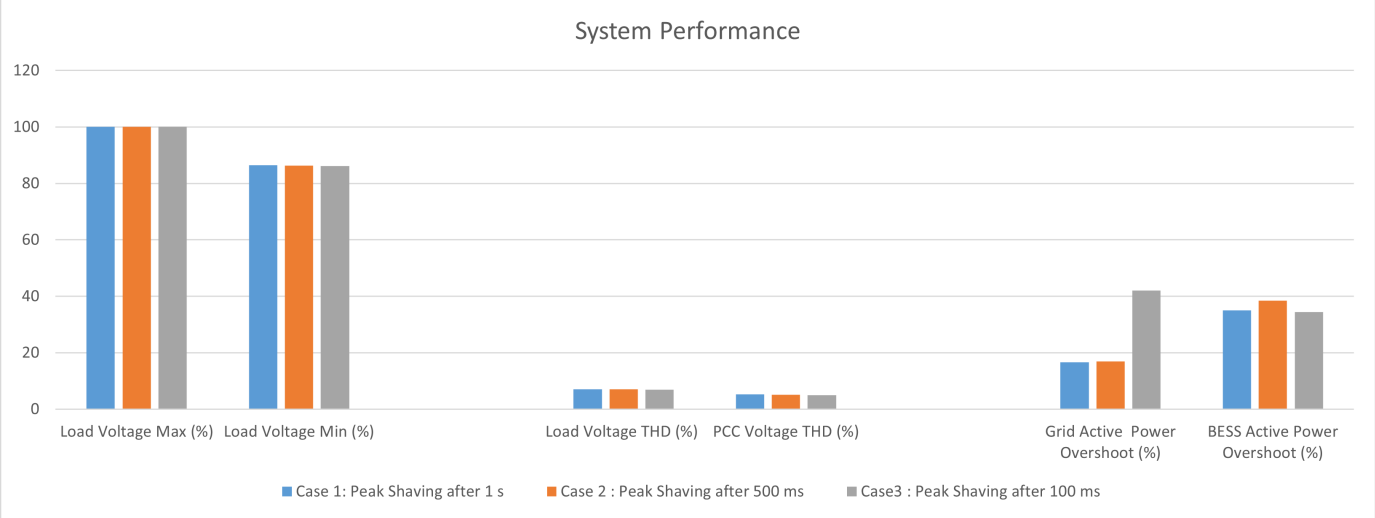

This plot shows the indices for the BESS system implemented in this model following the IEEE 1547 -2018: Category II - Inverters Sourced with Energy Storage standard.




This plot shows the indices for the BESS system operation and maintenance implemented in this model following the IEEE 2030.2.1-2019 Guide for Design, Operation, and Maintenance of Battery.




This plot shows the impact of peak shaving function time delay after grid power crosses the threshold.




The time delay of the peak shaving function has more impact on the overshoots of the active power from the grid and the BESS.

There are no significant impact on the load voltage and total harmonic distortion (THD) values.

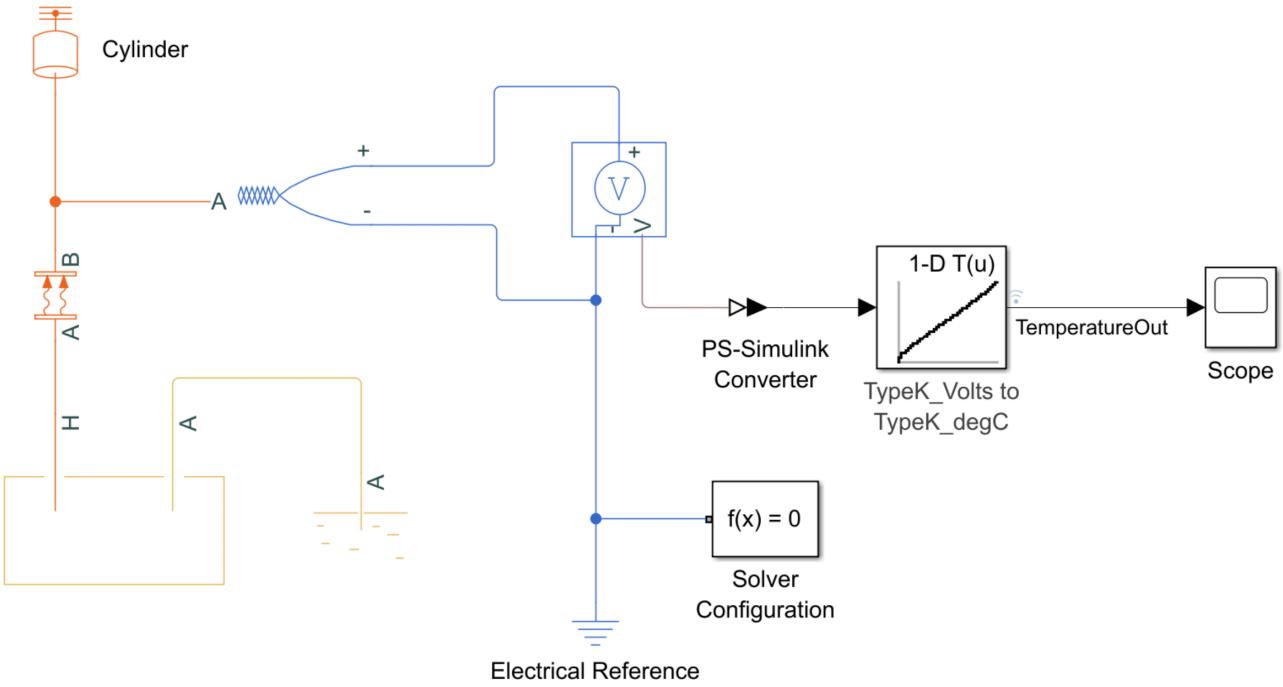
# Measure Temperature During Quenching of Steel Using Thermocouple Block

This example shows how you can use the Thermocouple block to measure the temperature in steel quenching. During quenching, you heat the steel to a temperature above its recrystallization temperature and then rapidly cool the steel by submerging it in liquid. Because of the high temperature, you cannot use the traditional temperature measurement techniques such as thermistor-based or semiconductor-based temperature sensors or liquid thermometers. To measure temperatures in such harsh environments, industries and laboratories employ thermocouples, which convert thermal potential difference into electrical potential difference.

### Model Overview

Open the model `ee_fault_current_limiter.slx`.

```
mdl = "ee_thermocouple_quenching";
open_system(mdl)
```




## Measure Temperature During Quenching of Steel Using Thermocouple Block

1. Explore simulation results using sscexplore
2. Learn more about this example
3. Switch between Type K and Type S Thermocouples and plot both results

Copyright 2022 The MathWorks, Inc.

A Thermal Mass block models a steel cylinder with an initial temperature of 850°C. A Constant Volume Chamber (TL) block models a quenching bath and cools the cylinder to 20°C through forced convection.

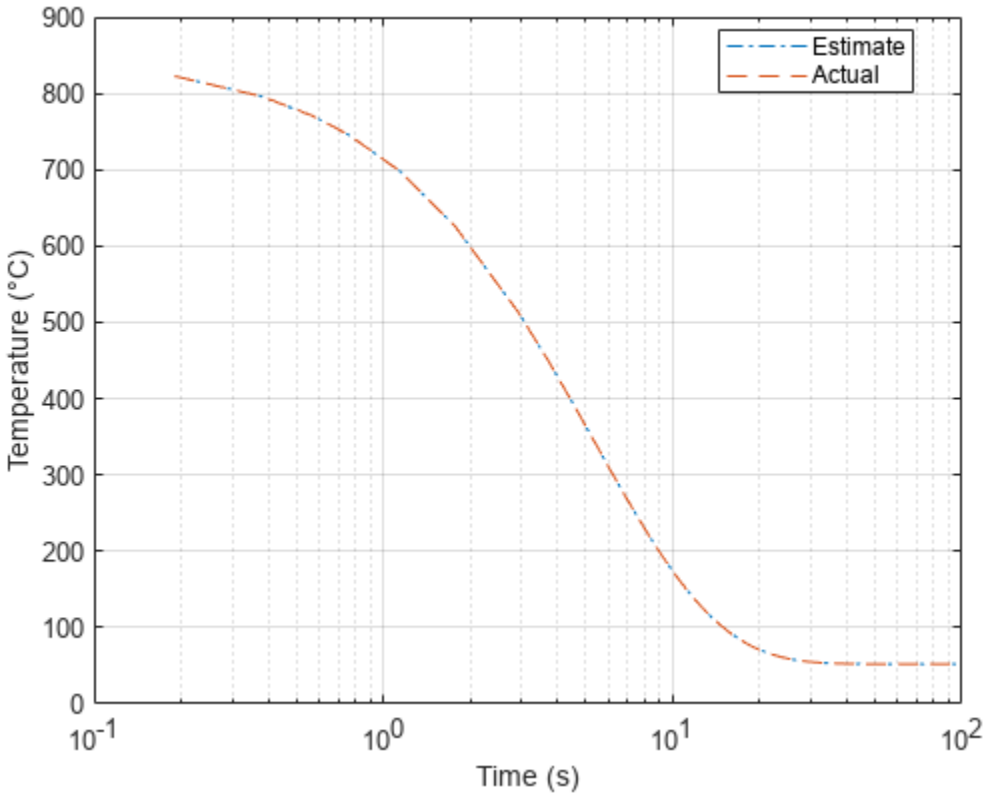
### Simulate the Model and Plot the Results

Simulate the model.

```
sim(mdl);
```

The 1-D Lookup Table block converts the output voltage from the Thermocouple block into temperature. Plot the temperature estimated from the thermocouple output voltage and the actual simulated temperature of the cylinder against time.

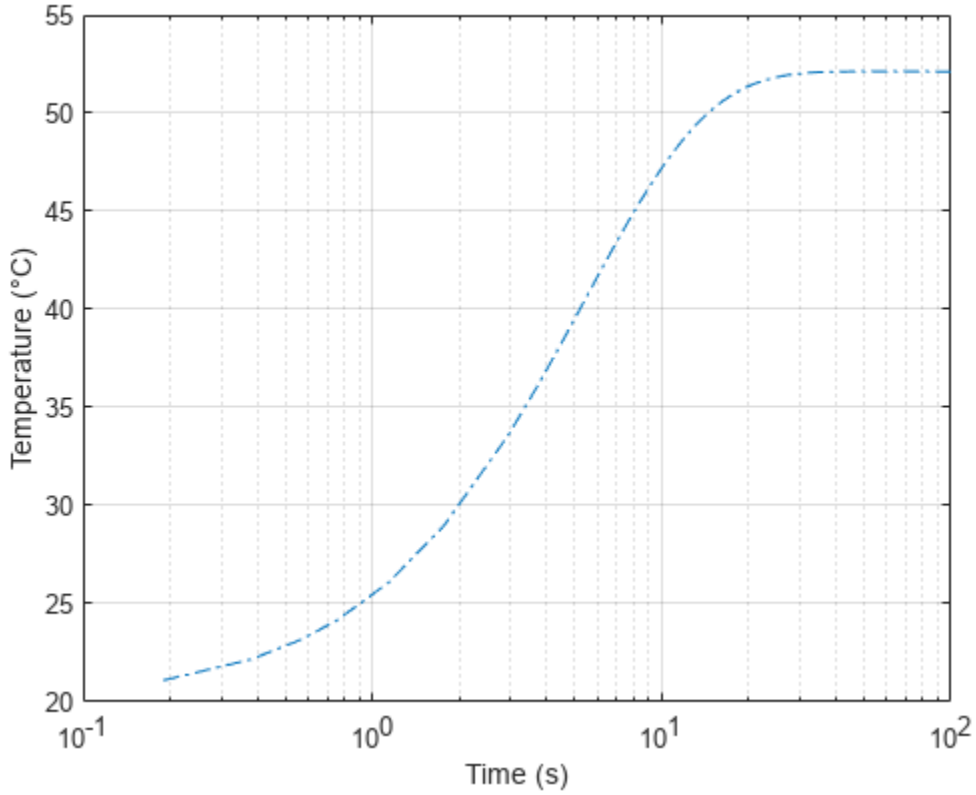
```
TemperatureOut = squeeze(logsout{1}.Values.Data);
time = simlog_ee_thermocouple_quenching.Cylinder.T.series.time;
temperature = simlog_ee_thermocouple_quenching.Cylinder.T.series.values;

ee_thermocouple_quenching_plot(tout,TemperatureOut,time,temperature,'Temperature (°C)','Estimate
```




As expected, the estimated and actual values are consistent.

Plot the temperature of the Constant Volume Chamber (TL) block.

```
time =  simlog_ee_thermocouple_quenching.Constant_Volume_Chamber_TL.T_I.series.time;
temperature =  simlog_ee_thermocouple_quenching.Constant_Volume_Chamber_TL.T_I.series.values;

ee_thermocouple_quenching_plot(time,temperature,'Temperature (°C)')
```




To ensure that the temperature is lower than the maximum valid temperature of the thermal liquid, the **Chamber volume** and **Cross-sectional area at port A** parameters of the Constant Volume Chamber (TL) block must be sufficiently high. In this example, the maximum temperature is equal to 52.1°C, which is within acceptable range.

### Change to Thermocouple Type S

The Thermocuple block parameters are specified for a Type K thermocouple by default. These values are obtained from the NIST ITS-90 Thermocouple Database [1] on page 8-16.

Save the results for the Type K thermocouple.

```
TypeK_tout = tout;
TypeK_TemperatureOut = squeeze(logsout{1}.Values.Data);
TypeK_vout = simlog_ee_thermocouple_quenching.Thermocouple.v.series.values;
```

In the thermocouple block, set the **Thermocouple type** parameter to `Type B, E, J, K (t<=0 degC), N, R, S or T`.

```
set_param(mdl + "/Thermocouple", "thermocouple_type", '1');
```

In the thermocouple block, set the values of the **Coefficients [c0 c1 ... cn]** parameter for a Type S thermocouple.

```
set_param(mdl + "/Thermocouple", "Cvec", 'TypeS_Cvec');
```

In the 1-D Lookup Table block, set the **Table data** and **Breakpoints 1** parameters, which convert from the Volts to degC, to the correct vectors for a Type S thermocouple.

For more information on how to obtain these vectors, see the “Approximating Nonlinear Relationships: Type S Thermocouple” example.

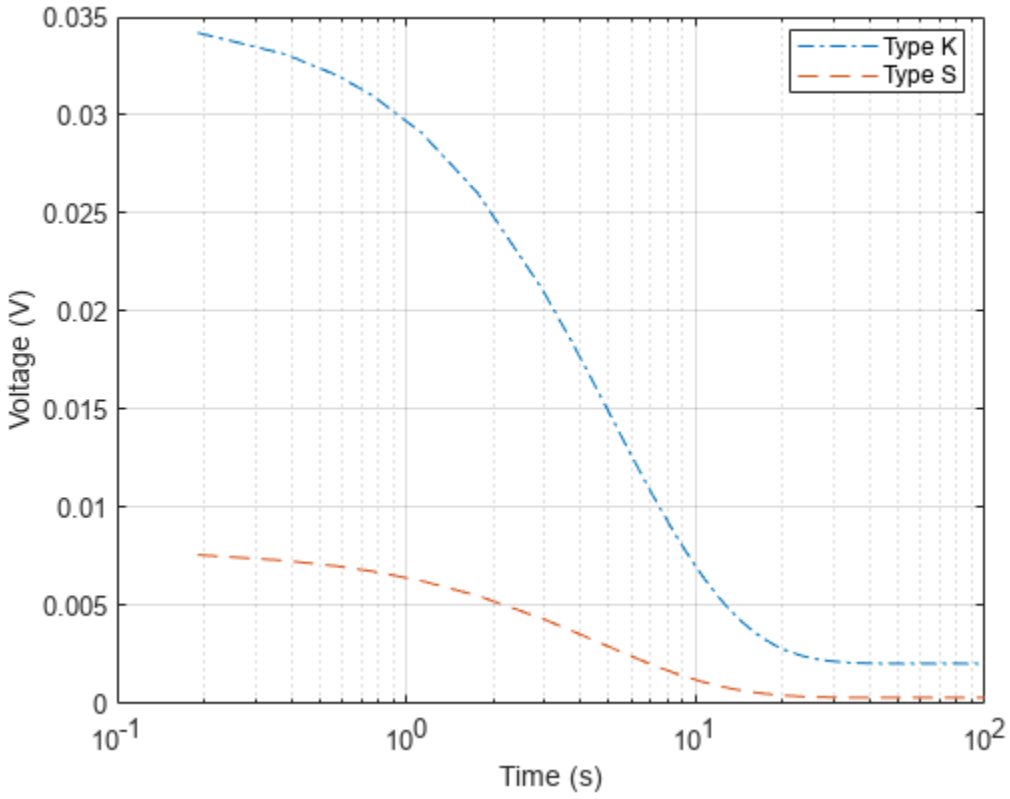
```
set_param(mdl + "/1-D Lookup Table", "Table", 'TypeS_degC');
set_param(mdl + "/1-D Lookup Table", "BreakpointsForDimension1", 'TypeS_Volts');
```

Simulate the model again and save the results.

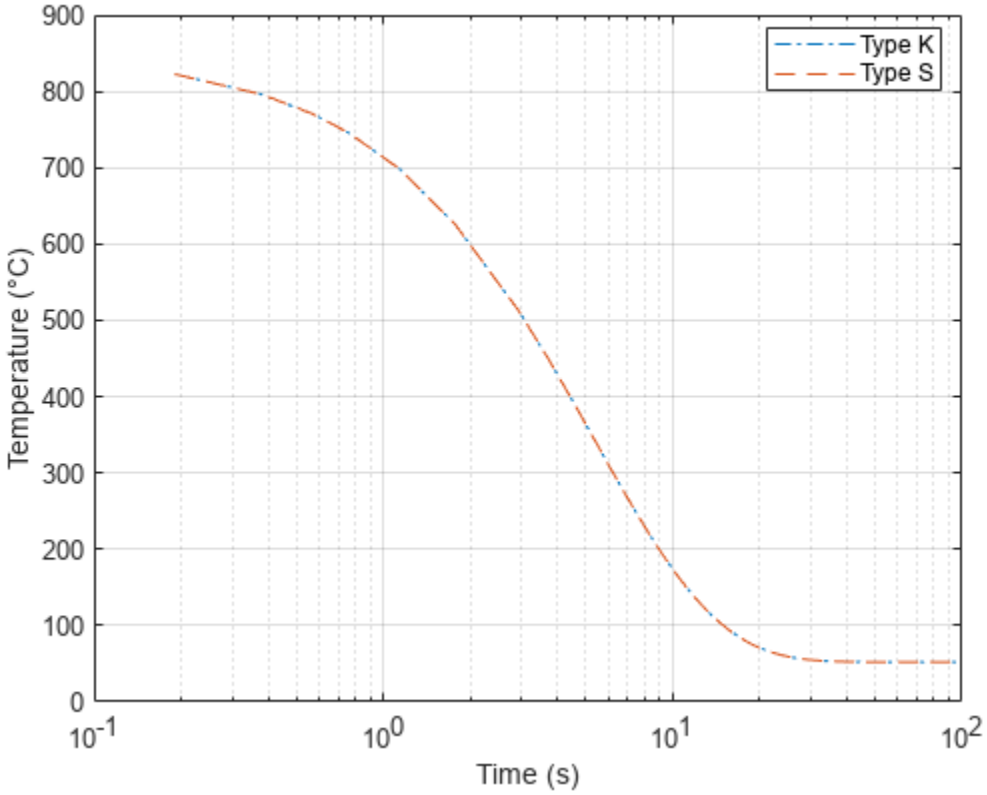
```
sim(mdl);

TypeS_tout = tout;
TypeS_TemperatureOut = squeeze(logsout{1}.Values.Data);
TypeS_vout = simlog_ee_thermocouple_quenching.Thermocouple.v.series.values;
```

Plot the output voltage and temperature for the Type K and Type S Thermocouples.

```
ee_thermocouple_quenching_plot(TypeK_tout,TypeK_vout,TypeS_tout,TypeS_vout,'Voltage (V)','Type K
```




```
ee_thermocouple_quenching_plot(TypeK_tout,TypeK_TemperatureOut,TypeS_tout,TypeS_TemperatureOut,'
```




As expected, the output voltage differs for the two thermocouple types but the temperatures are consistent.

### Change Heat Transfer Properties

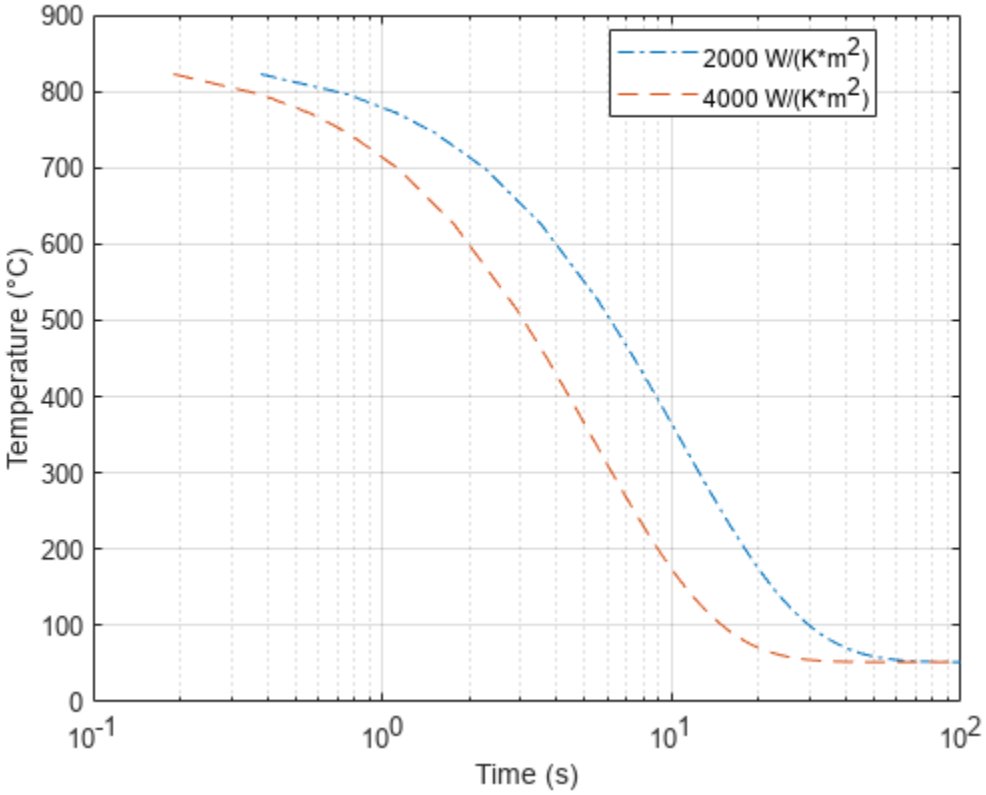
In this example, the value of the **Specific heat coefficient** parameter of the Thermal Mass block and the value of the **Heat transfer coefficient** of the Convective Heat Transfer block are estimates obtained from a scale-model laboratory study of industrial quenching of a steel cylinder in water [2] on page 8-16. For simplicity, these parameters are constant. For more accurate simulation results, specify these coefficients as a function of the temperature if the data are available. Alternatively, run simulations with different values of these parameters to obtain a range of estimates.

In the Convective Heat Transfer block, set the value of the **Heat transfer coefficient** parameter to 2000.

```
set_param(mdl + "/Convective Heat Transfer", "heat_tr_coeff", '2000');
```

Simulate the model and plot the results.

```
sim(mdl);
TemperatureOut = squeeze(logsout{1}.Values.Data);

ee_thermocouple_quenching_plot(tout,TemperatureOut,TypeS_tout,TypeS_TemperatureOut,'Temperature
```




With a **Heat transfer coefficient** parameter value of 2000, the steel cylinder cools much slower.

**References**

[1] Garrity, Karen. “NIST ITS-90 Thermocouple Database - SRD 60.” National Institute of Standards and Technology, July 1, 2000. https://doi.org/10.18434/T4S888.

[2] Hasan, H. S., M. J. Peet, J. M. Jalil, and H. K. D. H. Bhadeshia. “Heat Transfer Coefficients during Quenching of Steels.” *Heat and Mass Transfer* 47, no. 3 (March 2011): 315–21. https://doi.org/10.1007/s00231-010-0721-4.

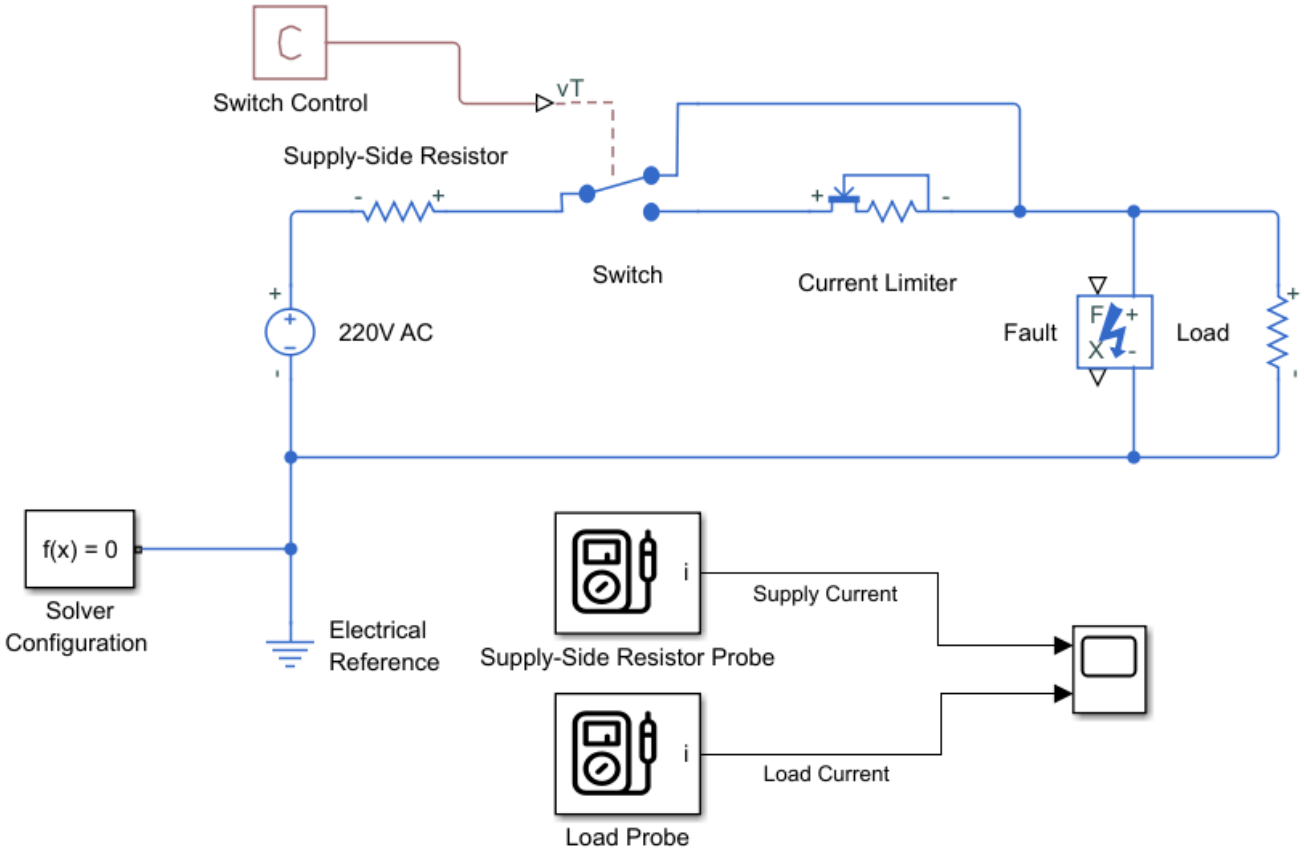
# Protect Components During Fault Using Current Limiter Block

This example shows how to use a Current Limiter block to protect a resistor from a short circuit or failure of the load.

### Model Overview

Open the model `ee_fault_current_limiter.slx`.

```
mdl = "ee_fault_current_limiter";
open_system(mdl)
```



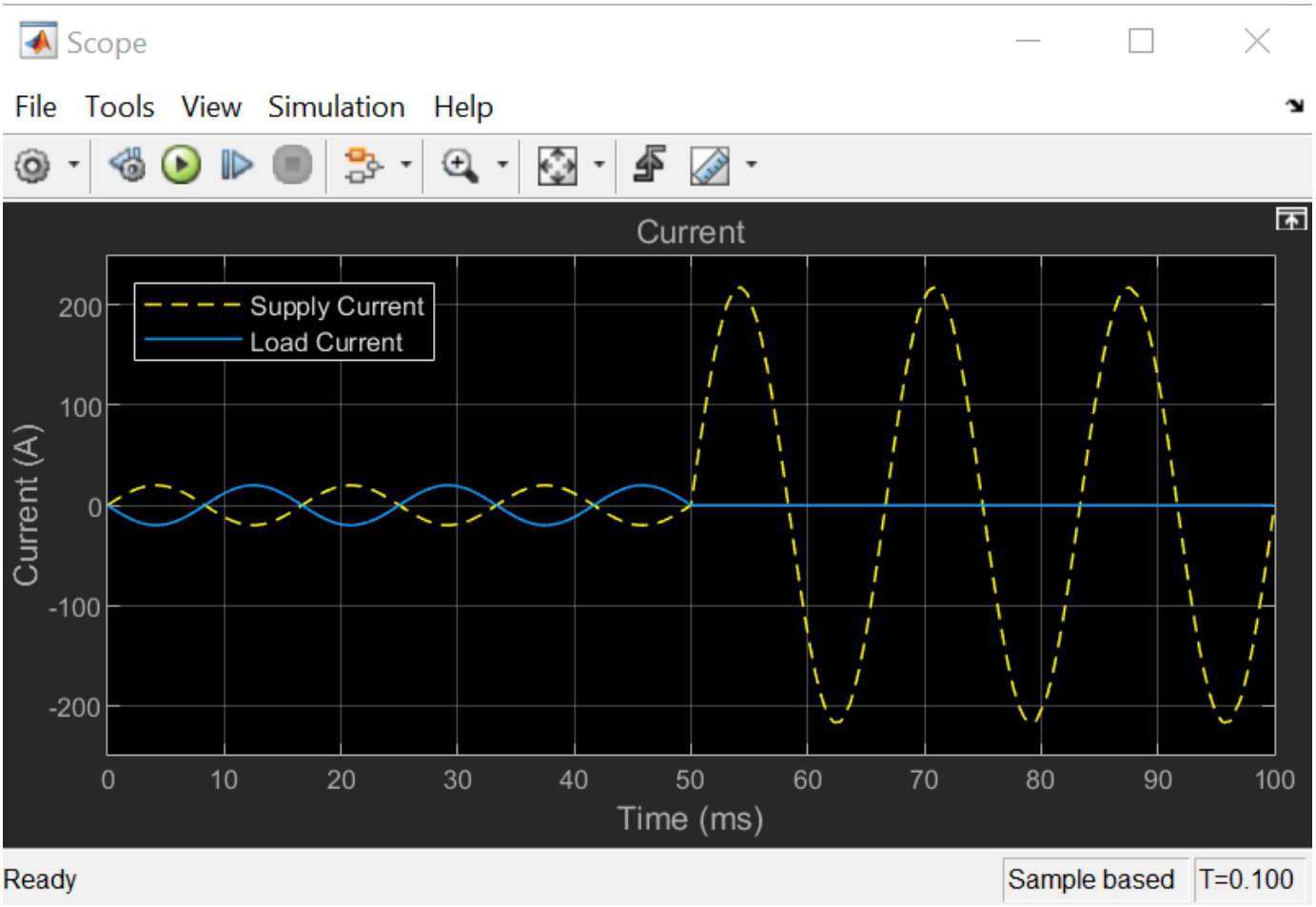

At t = 50 ms, the fault is triggered. A Switch block enables or disables the Current Limiter block. You can control the Switch block by using the Switch Control Constant block. The threshold value of the switch is equal to 0.5 V. To switch the current limiter on, set the value of the Constant block to 1 V. To switch the current limiter off, set the value of the Constant block to 0 V.

### Simulate the Model Without Limiting the Current and Check the Results

To disable the Current Limiter block in the model, set the value of the Switch Control block to 0 and simulate the model.

```
set_param( mdl + "/Switch Control", "constant",'0');
sim(mdl);
result1 = simlog_ee_fault_current_limiter;
```

Observe the supply-side current. In the model, open the Scope block.




When the fault triggers at t = 50ms, the load is in short circuit. This leads to a substantial increase in the current that the power supply requires as the resistance of the Supply-Side Resistor block is much higher than the resistance of the Load block.

### Set the Current Limiter Parameters

The Current Limiter block maintains the current on the supply side at values comparable to normal operation once the fault triggers. Set the **Current Limit** parameter, ilim, of the Current Limiter block to 20 A. This value is close to the peak current on the supply side during normal operation. This prevents the current from substantially exceeding this value during a fault without limiting the current during normal operation.

```
set_param( mdl + "/Current Limiter ", "ilim",'20');
```

Set the **Voltage drop when current starts to limit** parameter, vlim, of the Current Limiter block to 0.1 V. When under the current limit, the Current Limiter block behaves like a resistor with an average resistance roughly equal to vlim/ilim. In this example, this resistance is equal to 0.005 Ohm. This is much lower than the resistance of the Supply-Side Resistor, which is equal to 1 Ohm. This ensures the Current Limiter has negligible impact on the current during normal operation.

```
set_param( mdl + "/Current Limiter ", "vlim",'0.1');
```

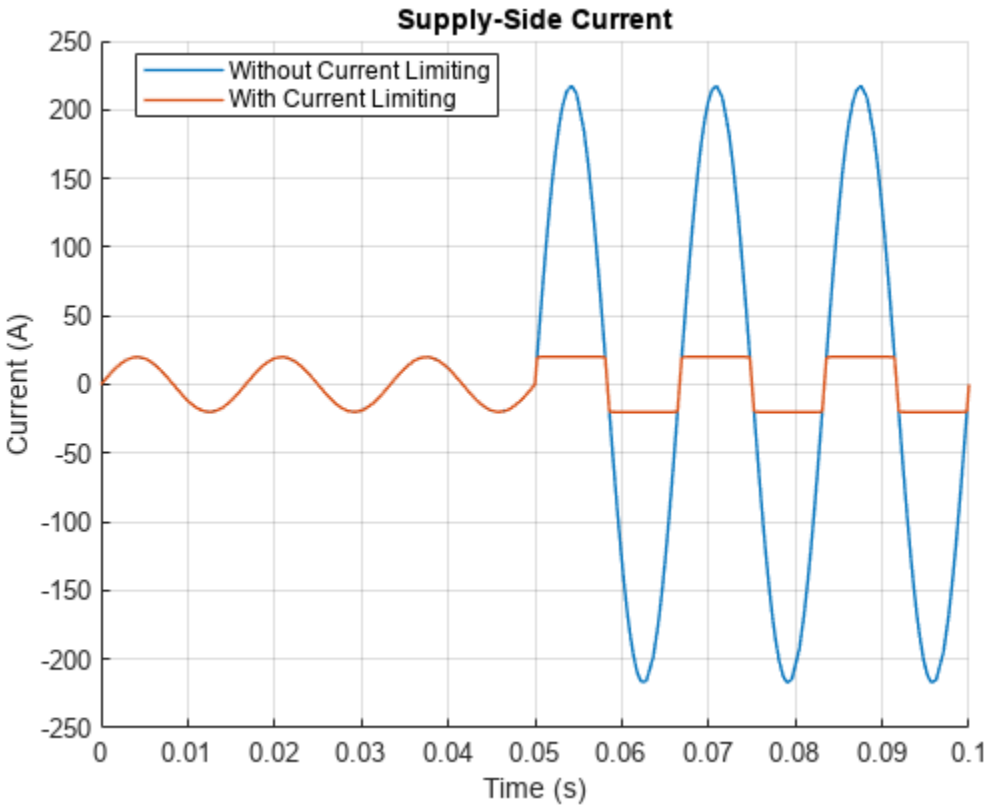
To enable the Current Limiter block in the model, set the value of the Switch Control block to 1 and then simulate the model.

```
set_param( mdl + "/Switch Control", "constant",'1');
sim(mdl);
result2 = simlog_ee_fault_current_limiter;
```

### Plot and Compare the Results

Plot the current on the supply side with and without current limiting.

```
ee_fault_current_limiter_plot_results(result1,result2)
```




During normal operation, the Current Limiter block has negligible impact on the current. When the fault triggers, the Current Limiter block maintains the current at values comparable to normal operation and protects the resistor from damage.

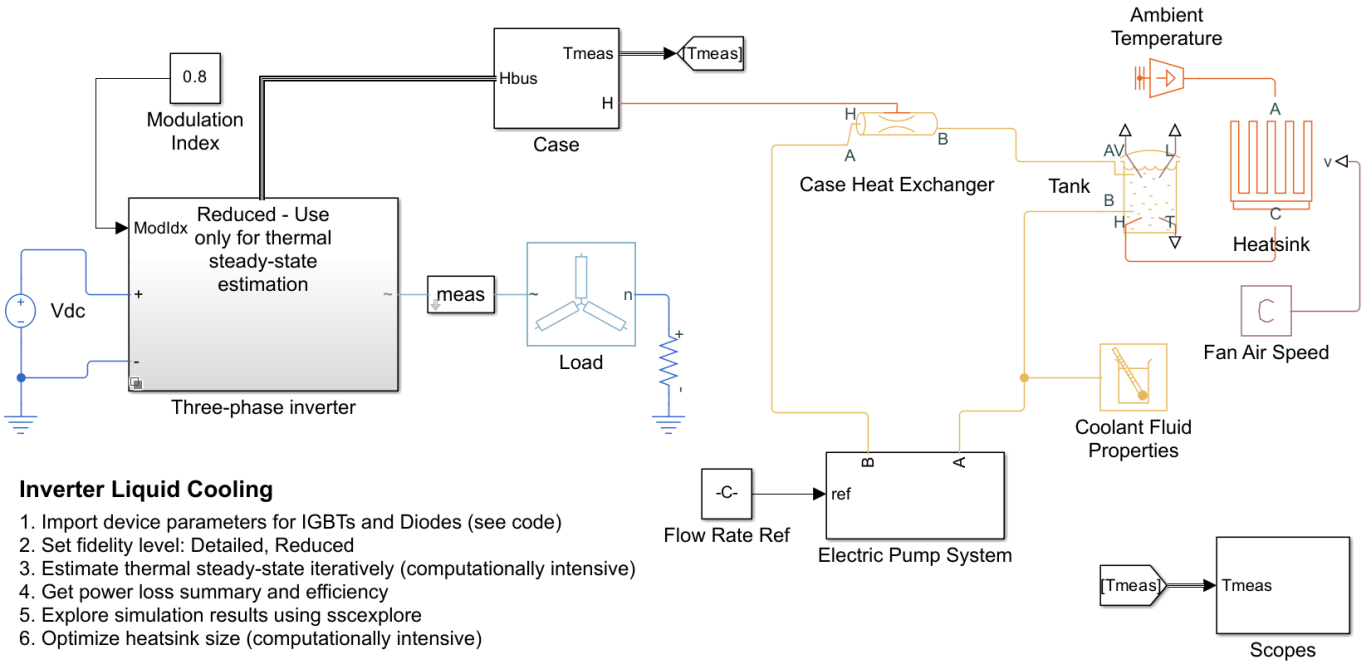
# Optimize an Inverter Liquid Cooling System

In this example you analyze the performance of a liquid cooling system for a three-phase inverter. You run detailed and reduced models (ROM) to find the steady-state temperatures and losses. You compute the optimal size of the heatsink that maximizes the inverter efficiency and minimizes lifetime cost.

### Model Overview

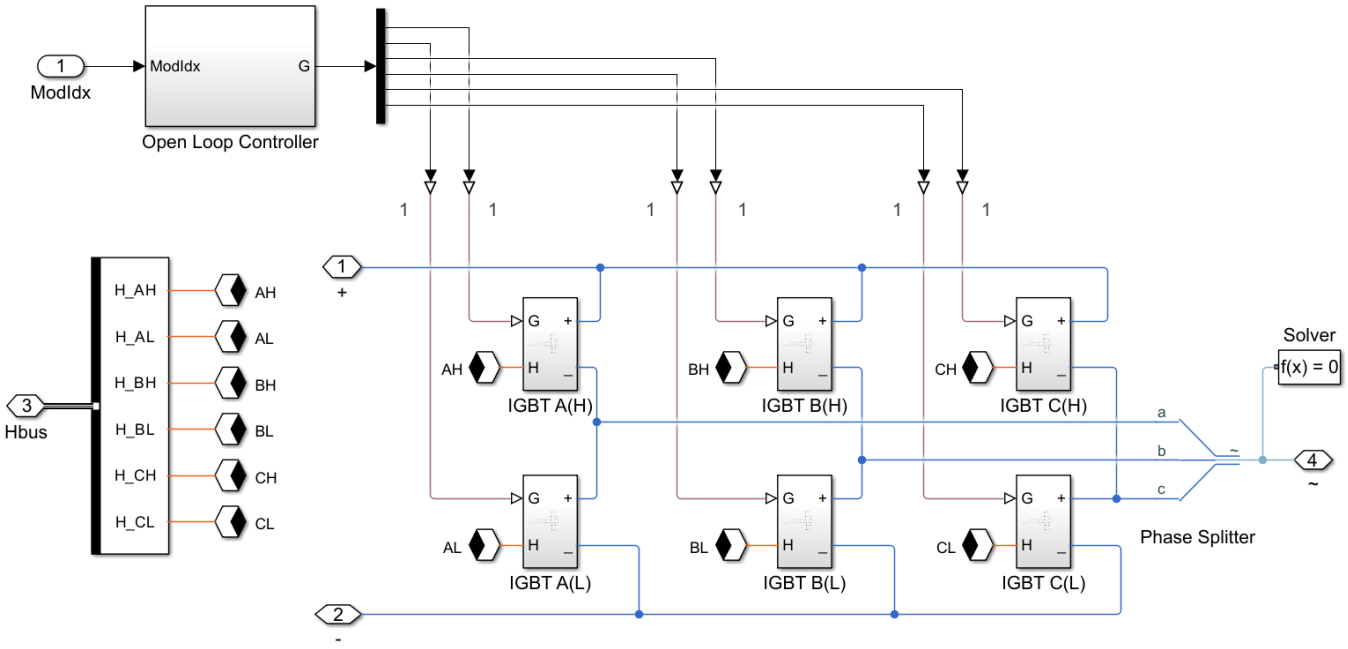
Open the model `sscv_inverter_liquid_cooling`.

```
model = 'sscv_inverter_liquid_cooling';
open_system(model)
sscv_inverter_liquid_cooling_params;
```

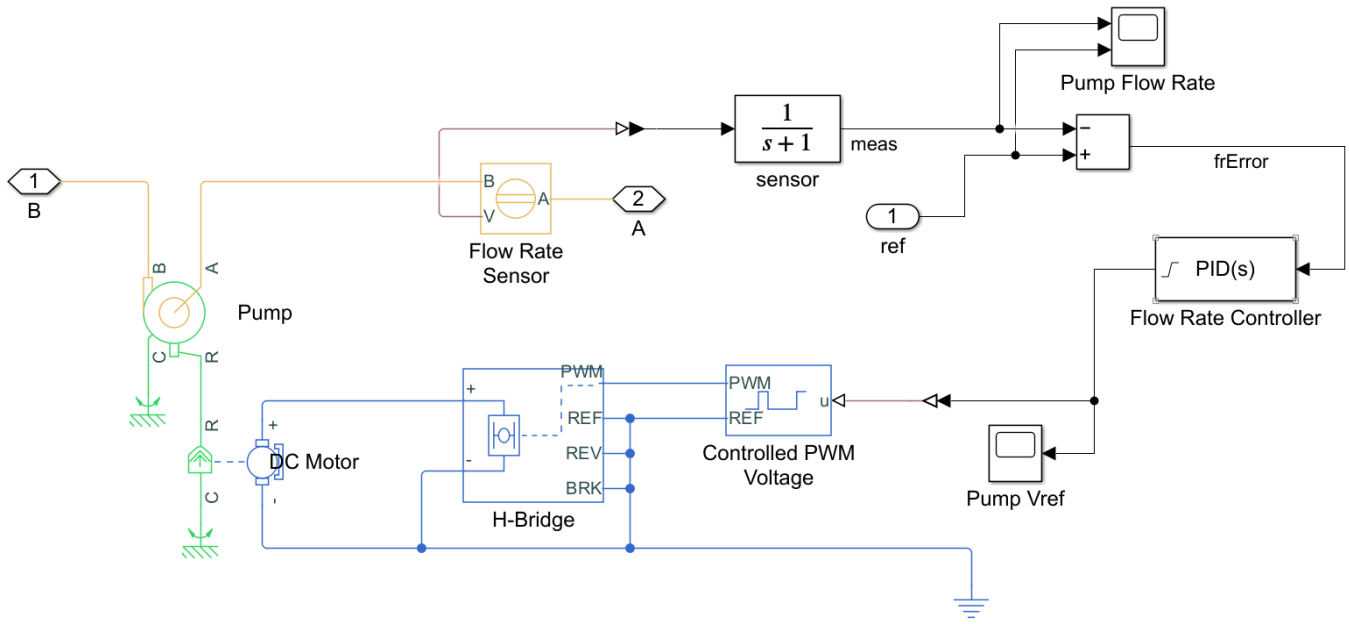



To drive the Load block, the inverter converts the DC power from the high-voltage battery into three-phase AC power. Conduction losses, switching losses, and reverse recovery losses generate heat in the case. Liquid cooling is effective to dissipate heat in the order of kilowatts. Liquid coolant flows in the inverter case to exchange heat. The Electric Pump System provides the hydraulic power to produce the flow. Then, the hot liquid flows into a heatsink to dissipate the heat. The heatsink is also called radiator. Fans blow air into the radiator to increase heat dissipation via forced convection.

The Three-phase Inverter has a Detailed Inverter variant that consists of six IGBT switching devices and the corresponding body diodes:



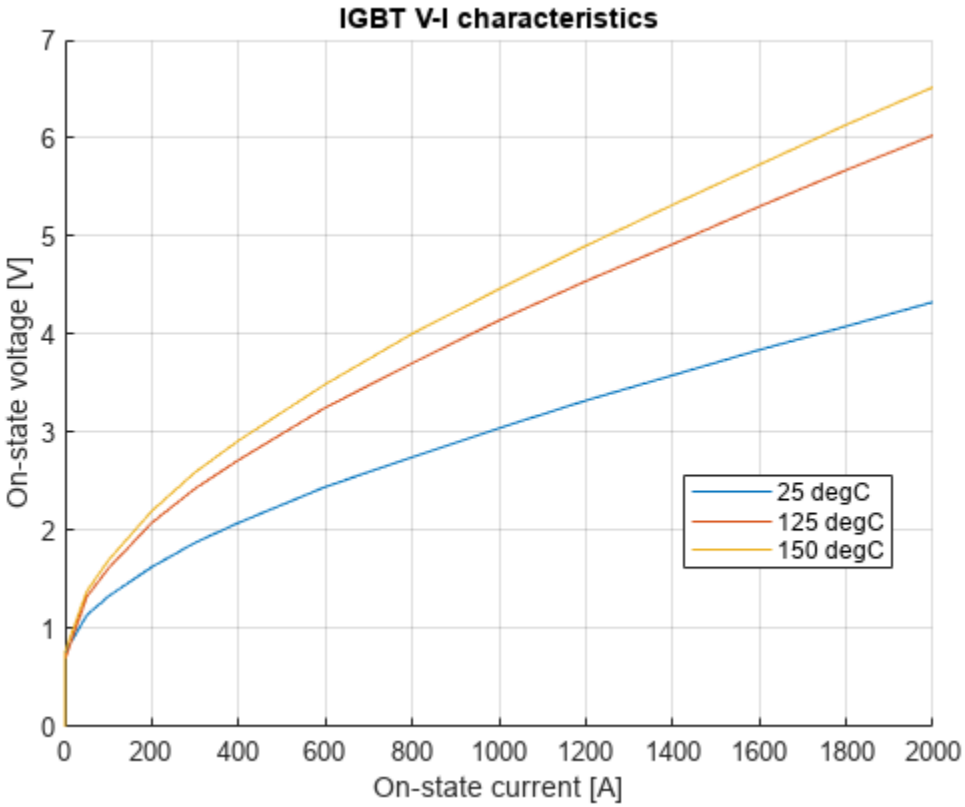

The Electric Pump System comprises a 12V DC motor and its H-bridge driver and a small centrifugal pump. The motor regulates the pump flow rate by using PWM switching and a PID controller:

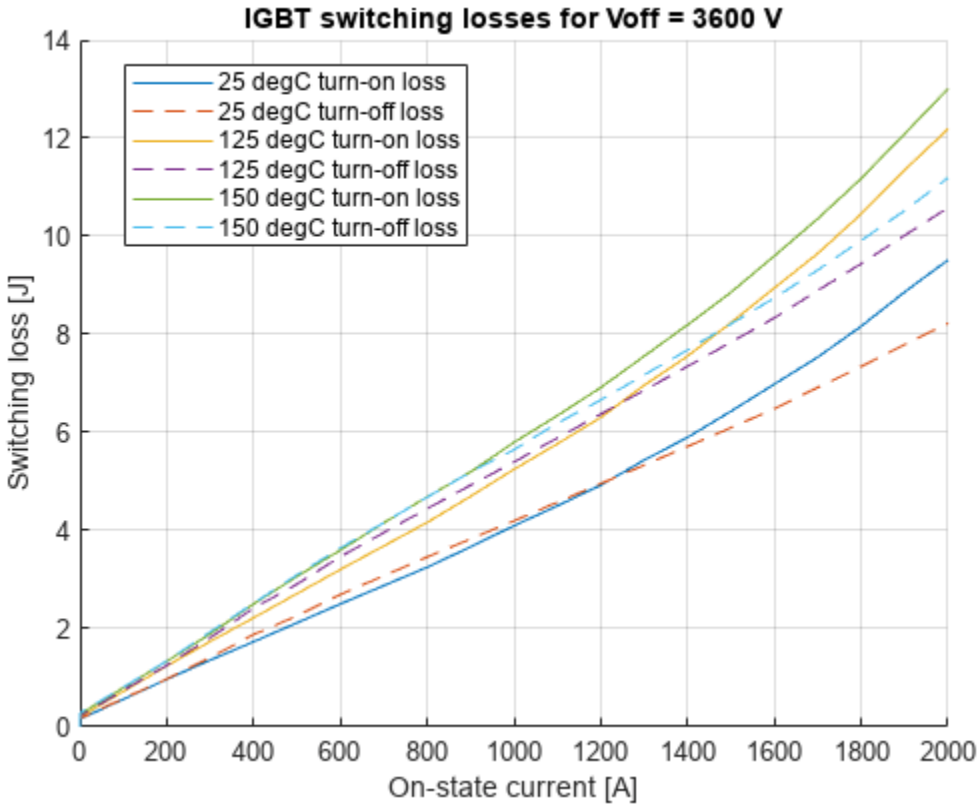


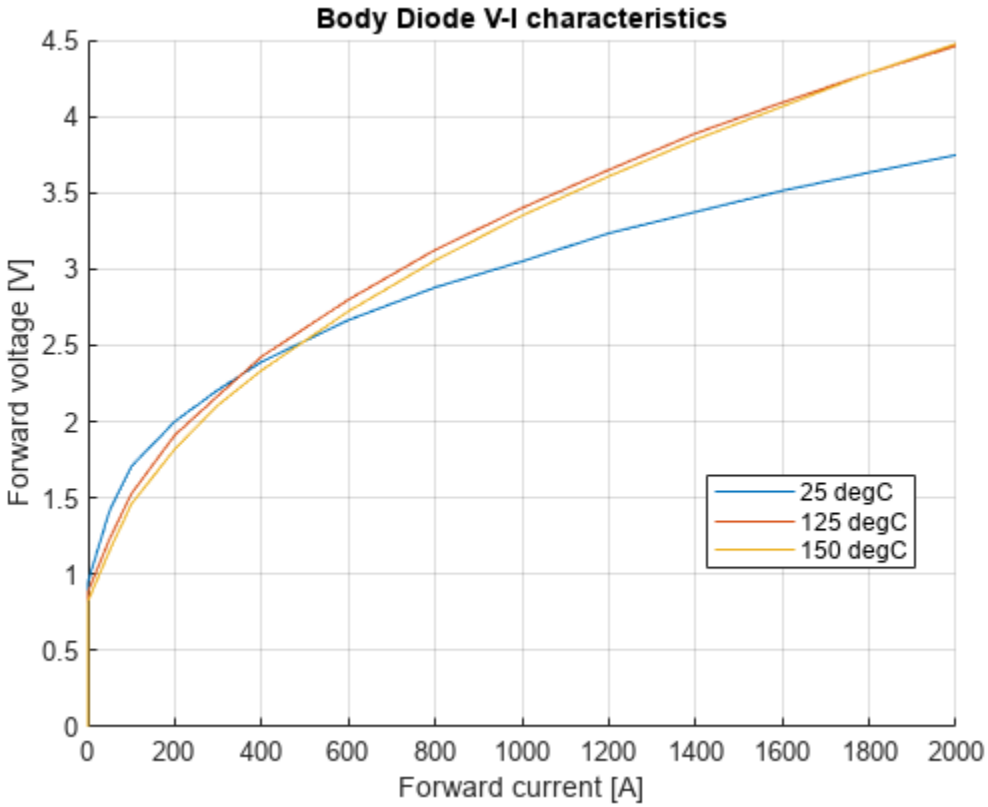

### Import Device Parameters

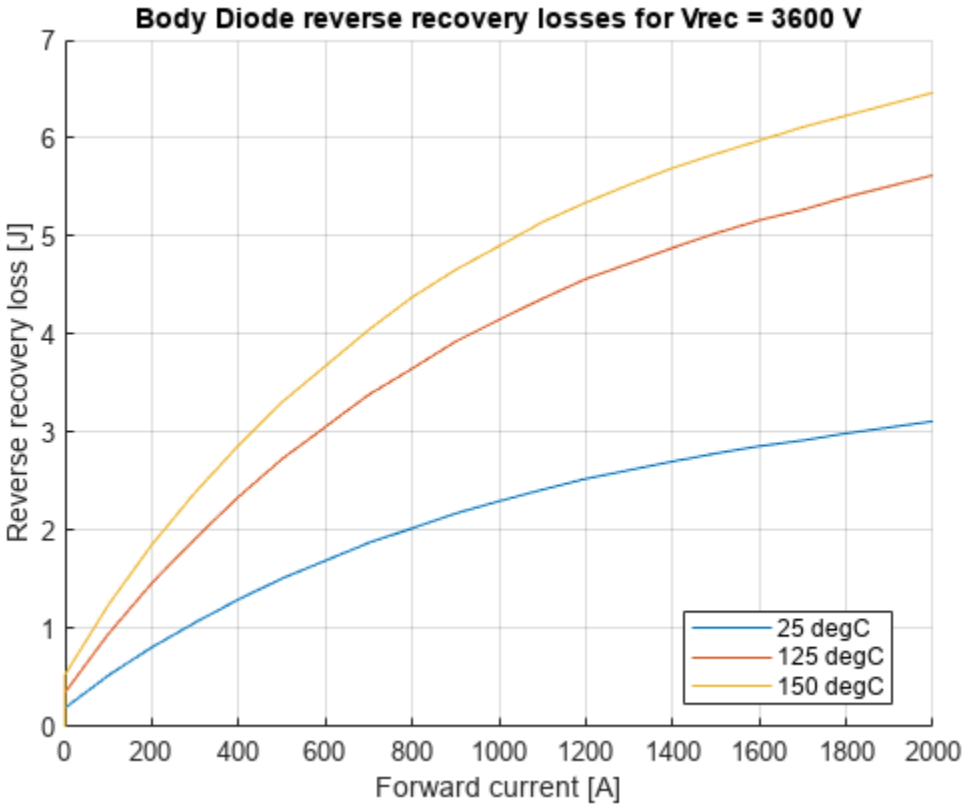
To import datasheet parameters for the IGBTs and Body Diodes in the Detailed inverter model, including turn-on and turn-off switching losses for the IGBT blocks, and reverse recovery losses for the Diode blocks, use the `ee_importDeviceParameters` utility function.

```
sscv_inverter_liquid_cooling_import
```


IGBT V-I characteristics
On-state voltage [V]
On-state current [A]
0
1
2
3
4
5
6
7
0
200
400
600
800
1000
1200
1400
1600
1800
2000
25 degC
125 degC
150 degC


IGBT switching losses for Voff = 3600 V
25 degC turn-on loss
25 degC turn-off loss
125 degC turn-on loss
125 degC turn-off loss
150 degC turn-on loss
150 degC turn-off loss
Switching loss [J]
On-state current [A]
0
2
4
6
8
10
12
14
0
200
400
600
800
1000
1200
1400
1600
1800
2000


Body Diode V-I characteristics
Forward voltage [V]
Forward current [A]
25 degC
125 degC
150 degC
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
0
200
400
600
800
1000
1200
1400
1600
1800
2000

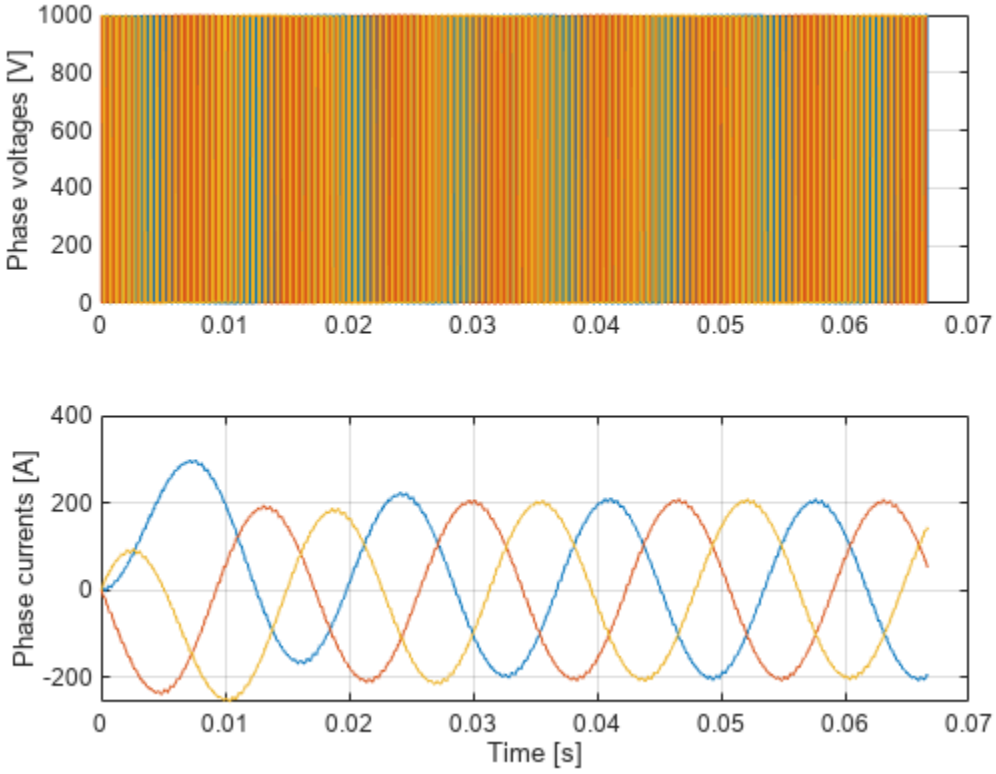



The plots above show the device characteristics for the imported IGBT and Diode blocks, including switching and reverse recovery losses.

### Run a Detailed Simulation

Simulate 4 AC cycles to observe load voltage and current.

```
set_param([model, '/Three-phase inverter'],'LabelModeActiveChoice', 'Detailed')
stopTime = 4/60; %#ok<NASGU>
out = sim(model);
figure();
Vabc = out.simlog.Load.N.V.series;
Iabc = out.simlog.Load.wye_impedance.I.series;
subplot(2,1,1);
plot(Vabc.time, Vabc.values('V'));
ylabel('Phase voltages [V]')
grid on
subplot(2,1,2);
plot(Iabc.time, Iabc.values('A'));
xlabel('Time [s]')
ylabel('Phase currents [A]')
grid on
```




Observe the inverter successfully produces a three-phase current at 60 Hz. It achieves electrical AC steady-state after an initial transient cycle.

### Run a Reduced Thermal Simulation

The electrical characteristic times in the inverter subsystem are in the order of milliseconds, but the thermal characteristic times are in the order of minutes. If you are only interested in the thermal steady-state of the system for a particular cooling setup and a fixed value of average conduction and switching losses, you can use the `Reduced - Use only for thermal steady-state estimation` variant of the Three-phase inverter subsystem.

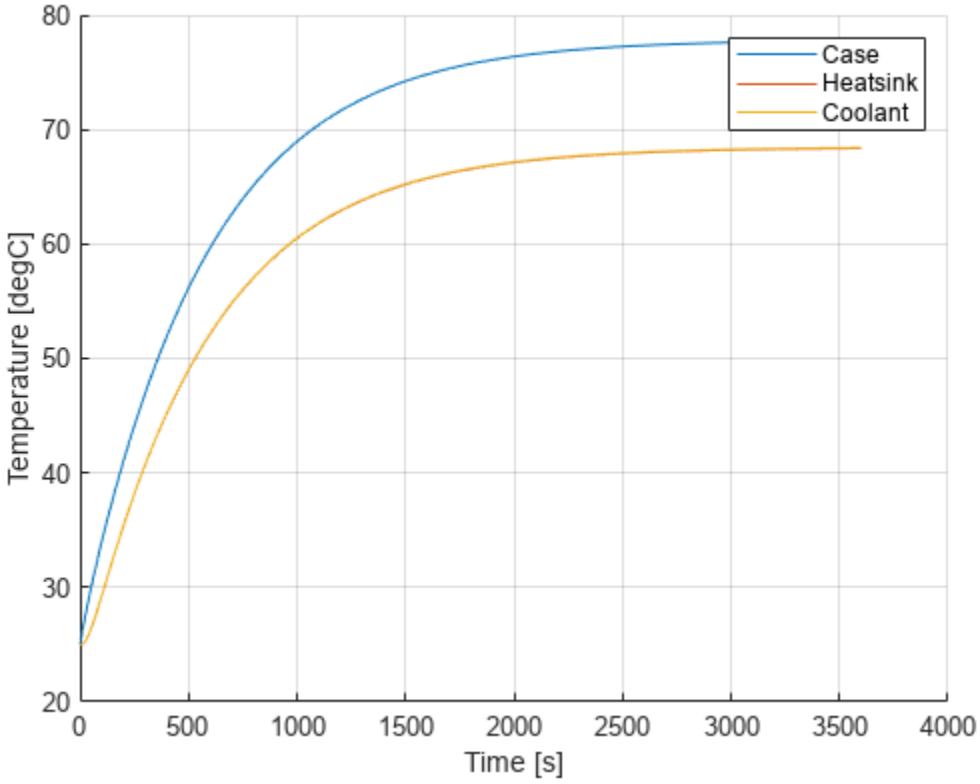
To run a 1 hour simulation of thermal model with constant conduction and switching losses, but without voltages or currents, at the MATLAB Command Window, enter:

```
set_param([bdroot, '/Three-phase inverter'],'LabelModeActiveChoice', 'Reduced - Use only for ther
stopTime = 3600;
out = sim(model);
```

Plot the temperature of the case, heatsink, and coolant parts:

```
figure();
Tcase = out.simlog.Case_Heat_Exchanger.H.T.series;
Theatsink = out.simlog.Heatsink.T_hs.series;
Tcoolant = out.simlog.Tank.T_I.series;
hold on
plot(Tcase.time, Tcase.values('degC'));
plot(Theatsink.time, Theatsink.values('degC'));
```

```
plot(Tcoolant.time, Tcoolant.values('degC'));
hold off
grid on
legend({'Case', 'Heatsink', 'Coolant'})
xlabel('Time [s]')
ylabel('Temperature [degC]')
```

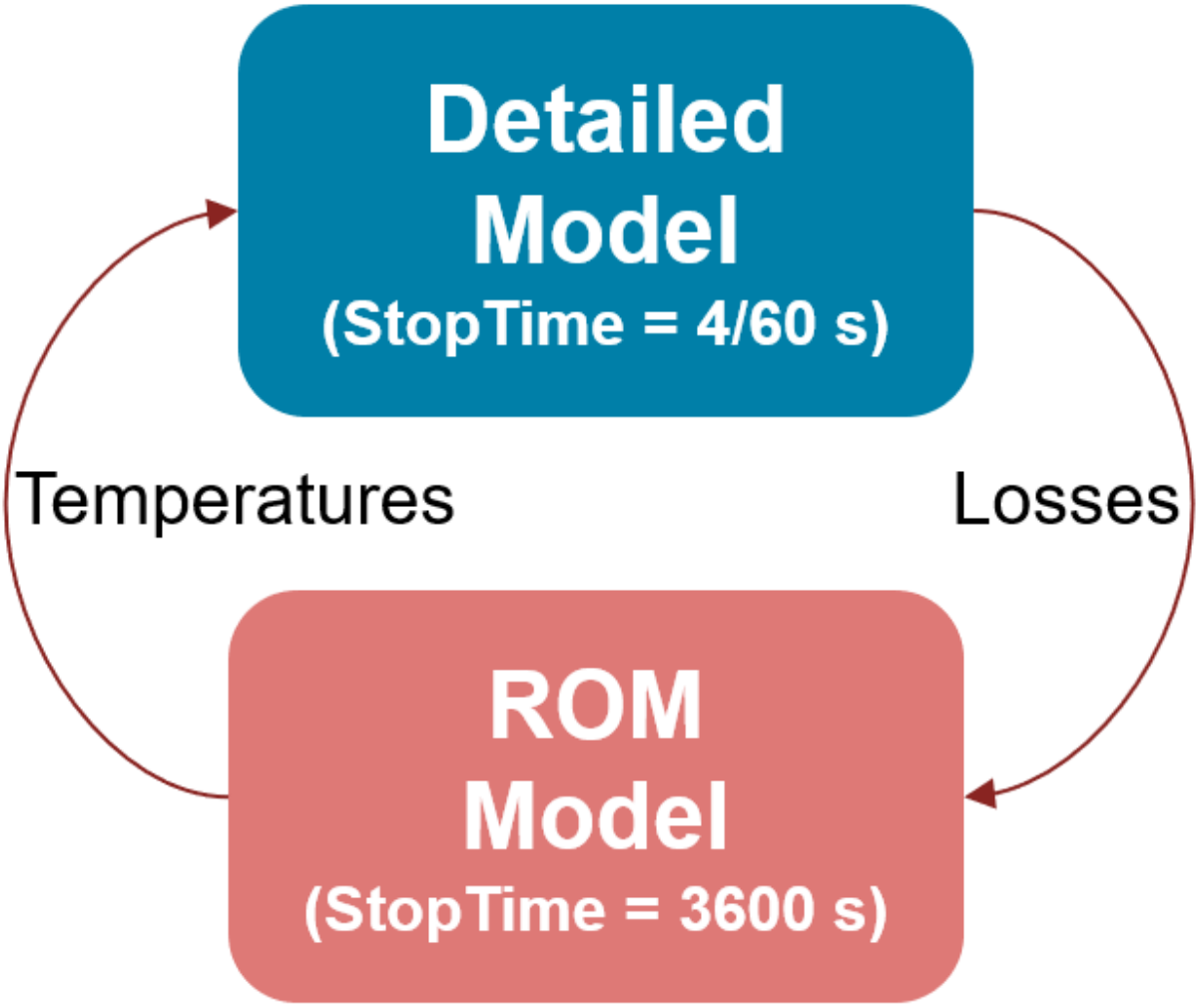



You observe that it takes almost one hour to reach thermal steady state.

**Estimate Thermal Steady-State Iteratively**

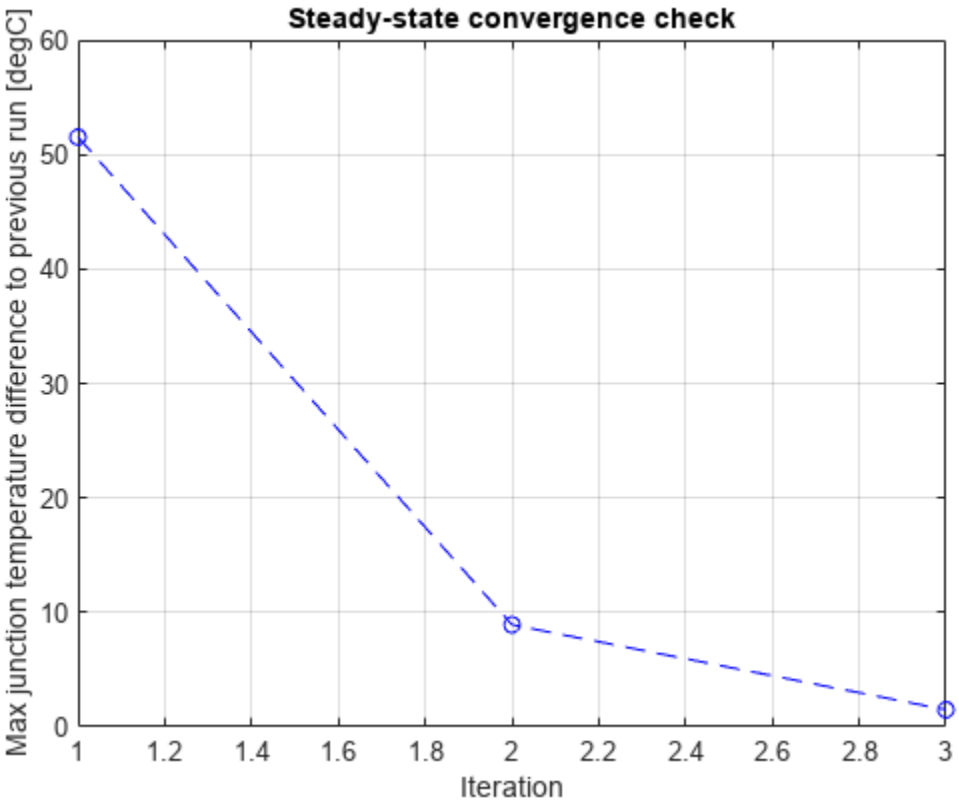
To find the accurate steady-state losses and temperatures faster, first compute the accurate losses at a given temperature by using the Detailed Inverter option of the Three-Phase Inverter subsystem. Then compute the accurate steady-state temperatures at a given loss by using the Reduced option of the Three-Phase Inverter subsystem.

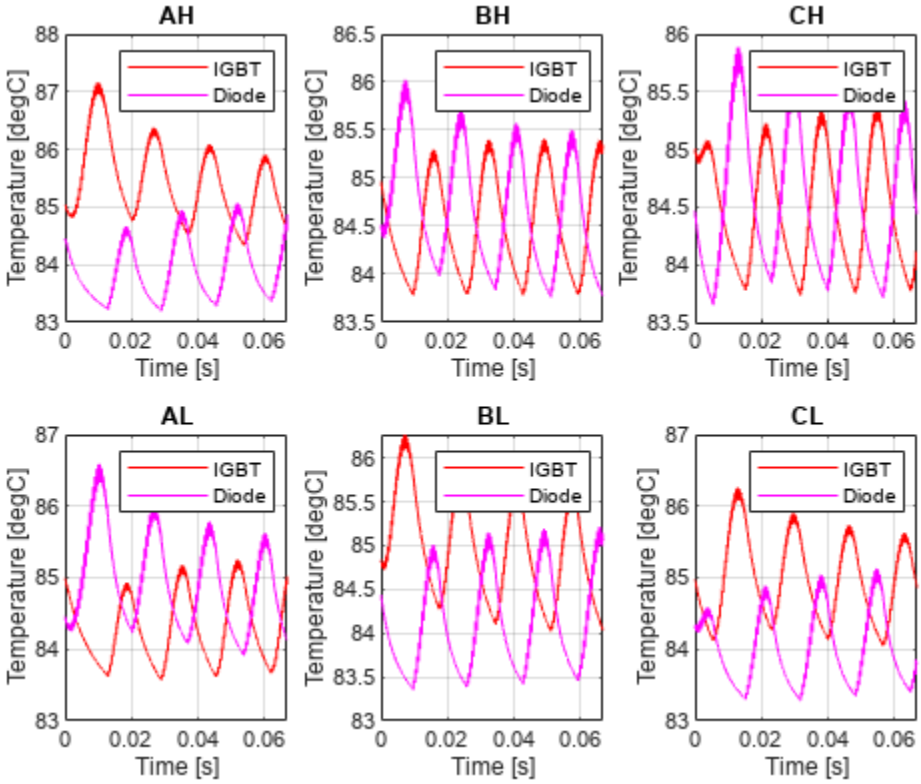
Finally, iterate between these two processes to estimate the thermal steady-state.

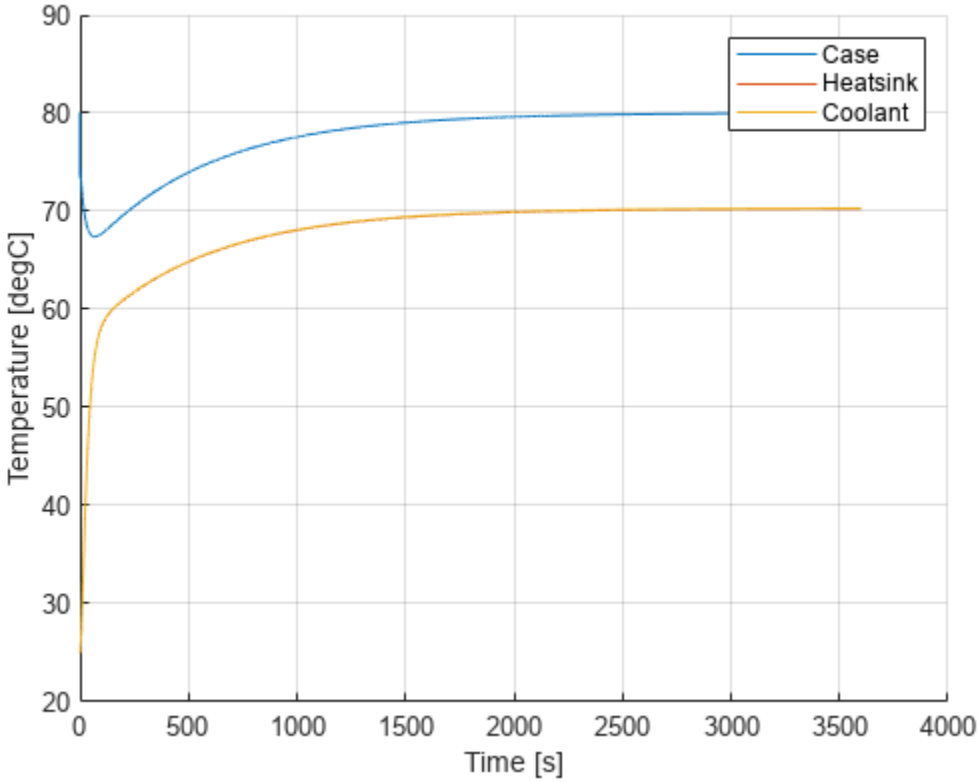



```
sscv_inverter_liquid_cooling_thermal_ss
```

```
Iteration 1 --------
Max Junction temperature difference to previous run = 51.5222 degC
Iteration 2 --------
Max Junction temperature difference to previous run = 8.9189 degC
Iteration 3 --------
Max Junction temperature difference to previous run = 1.4989 degC
```


Steady-state convergence check
Max junction temperature difference to previous run [degC]
Iteration
0
10
20
30
40
50
60
1
1.2
1.4
1.6
1.8
2
2.2
2.4
2.6
2.8
3


AH
BH
CH
AL
BL
CL
IGBT
Diode
Temperature [degC]
Time [s]




```
                                LoggingNode
_____________________________________________________________________________

    {'sscv_inverter_liquid_cooling.Load.wye_impedance'                                 }
    {'sscv_inverter_liquid_cooling.Three_phase_inverter.Detailed_Inverter.IGBT_AH.IGBT'      }
    {'sscv_inverter_liquid_cooling.Three_phase_inverter.Detailed_Inverter.IGBT_BL.IGBT'      }
    {'sscv_inverter_liquid_cooling.Three_phase_inverter.Detailed_Inverter.IGBT_CL.IGBT'      }
    {'sscv_inverter_liquid_cooling.Three_phase_inverter.Detailed_Inverter.IGBT_BH.IGBT'      }
    {'sscv_inverter_liquid_cooling.Three_phase_inverter.Detailed_Inverter.IGBT_CH.IGBT'      }
    {'sscv_inverter_liquid_cooling.Three_phase_inverter.Detailed_Inverter.IGBT_AL.IGBT'      }
    {'sscv_inverter_liquid_cooling.Three_phase_inverter.Detailed_Inverter.IGBT_AL.Body_Diode'}
    {'sscv_inverter_liquid_cooling.Three_phase_inverter.Detailed_Inverter.IGBT_BH.Body_Diode'}
    {'sscv_inverter_liquid_cooling.Three_phase_inverter.Detailed_Inverter.IGBT_AH.Body_Diode'}
    {'sscv_inverter_liquid_cooling.Three_phase_inverter.Detailed_Inverter.IGBT_CH.Body_Diode'}
    {'sscv_inverter_liquid_cooling.Three_phase_inverter.Detailed_Inverter.IGBT_BL.Body_Diode'}
    {'sscv_inverter_liquid_cooling.Three_phase_inverter.Detailed_Inverter.IGBT_CL.Body_Diode'}
    {'sscv_inverter_liquid_cooling.Electric_Pump_System.DC_Motor'                      }
    {'sscv_inverter_liquid_cooling.Electric_Pump_System.H_Bridge'                      }
    {'sscv_inverter_liquid_cooling.Gmin'                                               }

System efficiency = 86.0959%
```

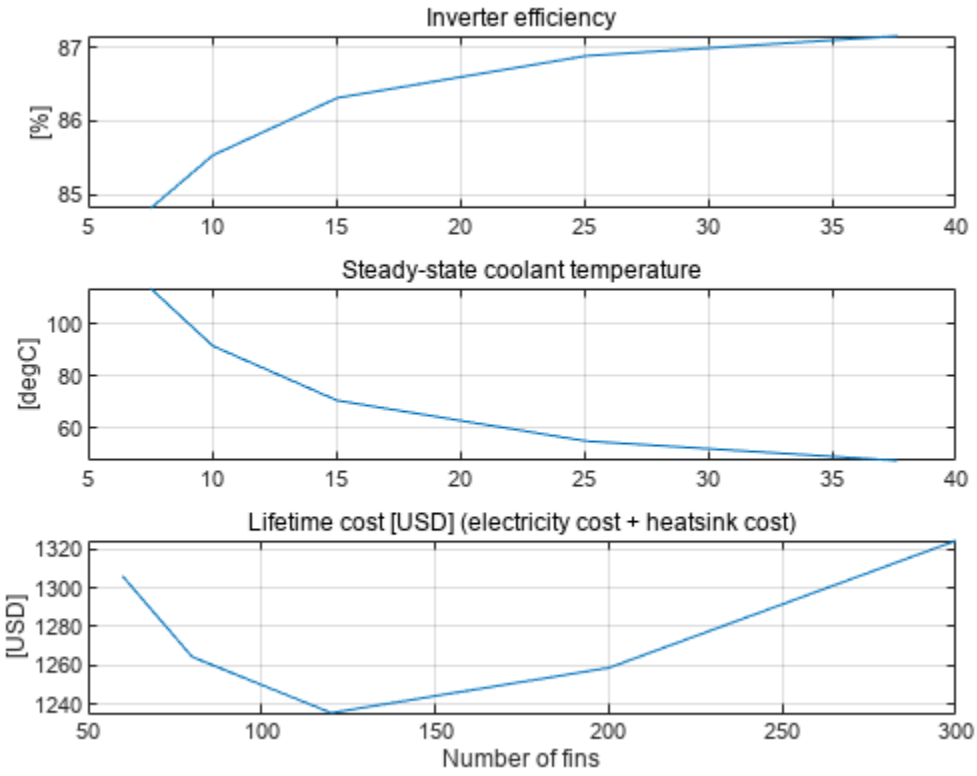
You observe that 3 iterations are enough to find a thermal steady-state with a reasonable accuracy of +-2 degC.

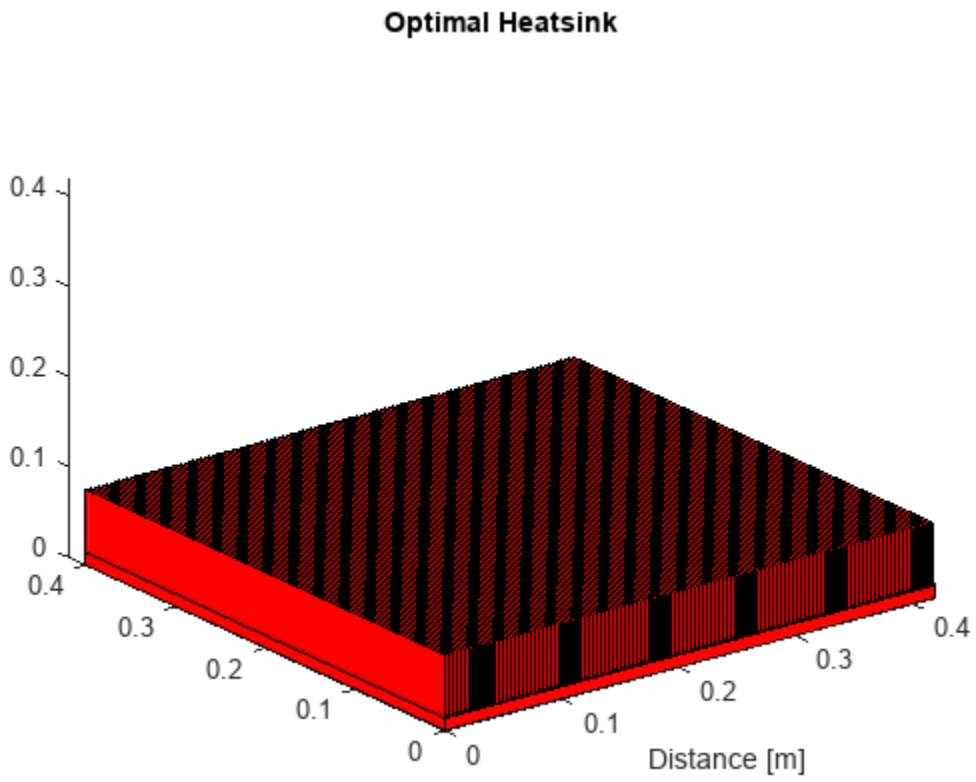
### Optimize Heatsink Size for Cost and Efficiency

To find the best option that minimizes the cost and maximizes the efficiency, repeat the steady-state iterations for several heatsink sizes. At the MATLAB Command Window, enter:

```
displayOutputs = false;
sscv_inverter_liquid_cooling_optimize
```

```
************ Case 1: Heatsink with NumFins = 60 ************
System efficiency = 84.8203%
************ Case 2: Heatsink with NumFins = 80 ************
System efficiency = 85.5376%
************ Case 3: Heatsink with NumFins = 120 ************
System efficiency = 86.3134%
************ Case 4: Heatsink with NumFins = 200 ************
System efficiency = 86.8824%
************ Case 5: Heatsink with NumFins = 300 ************
System efficiency = 87.1501%
-----------------------------------
Time to run 5 cases = 26.1305 min
***********************************
```






You observe that the option with a good blend of high efficiency and low coolant temperature, with a minimum lifetime cost, is a heatsink with around 120 fins.

# DC Fast Charging Station for Electric Vehicles with Solar Cogeneration

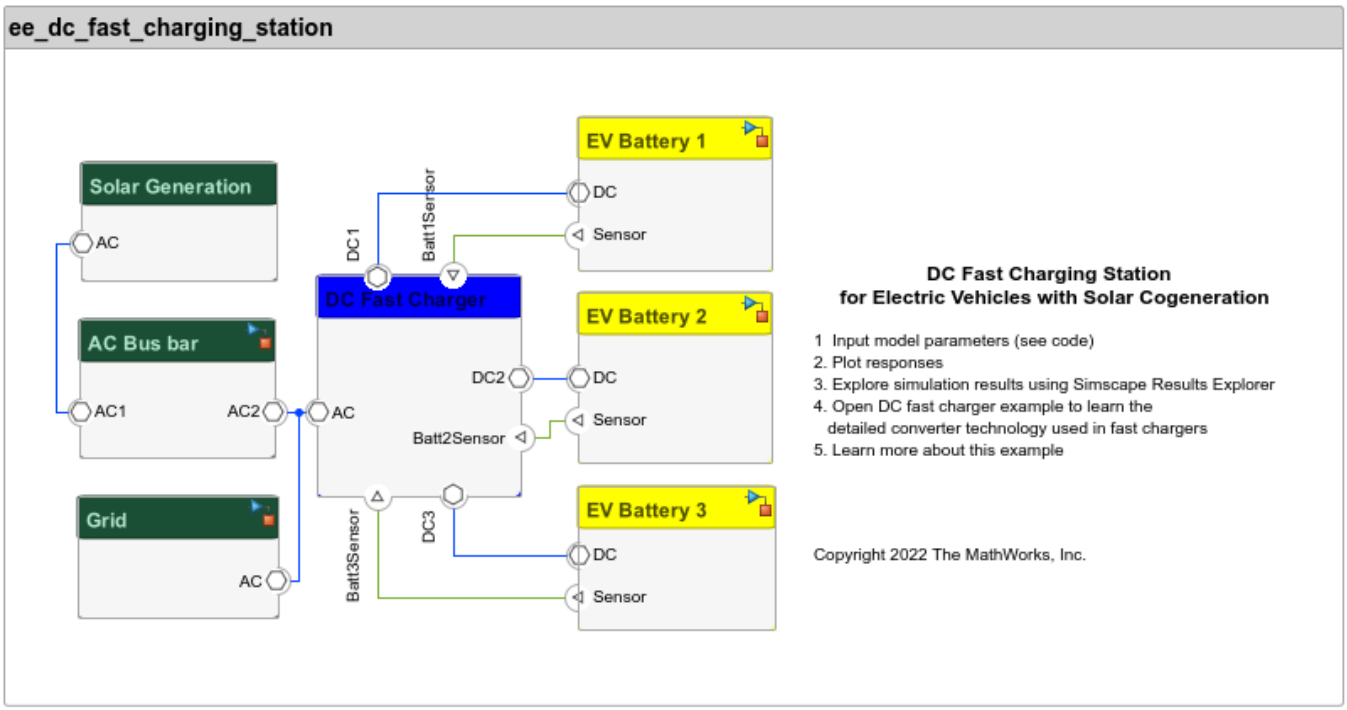
This example models a DC fast charging station with solar cogeneration connected with the three battery packs of electric vehicles (EV).

This example comprises four main components:

- Grid —— Model the AC supply voltage as a three-phase constant voltage source.
- Solar Generation —— Model the solar pack as parallel strings of series connected cells.
- DC Fast Charging Station —— Model the power electronic circuits to convert the AC supply voltage from the grid to the DC voltage level that the EV battery pack requires.
- EV battery pack —— Model the battery pack as series of battery cells.

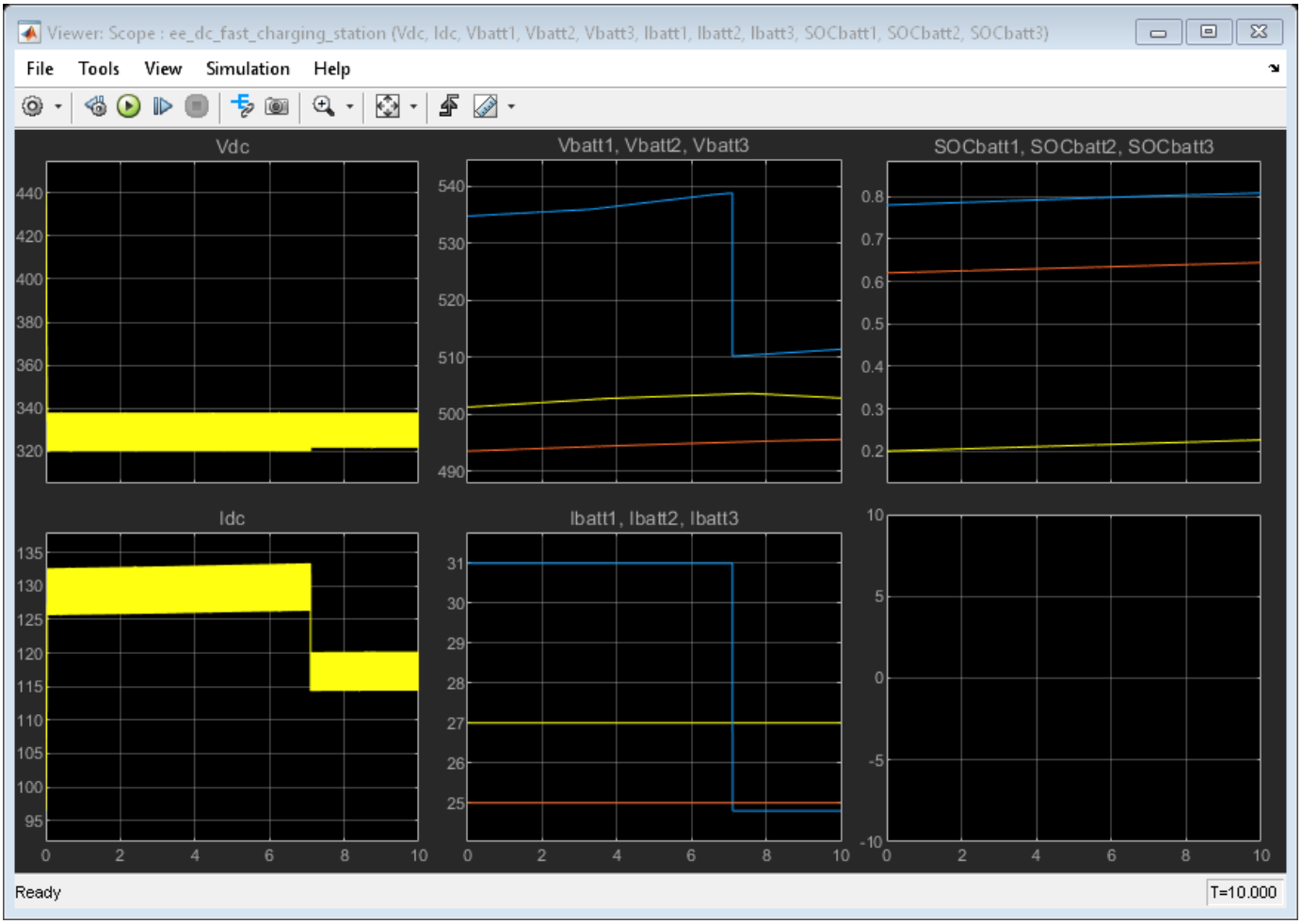
### Model Overview

The system composer overview of the example is shown below:




### Simulation Results from Simscape Logging

The plot below shows the DC bus voltage and current, battery terminal voltage, charging current and SOC. Each EV battery pack has different charging current as well as initial state of charge(SOC). As the battery is charged, it's SOC rises from specified initial SOC. As battery 2 SOC reaches trickle charge SOC (default value is 0.8) from its initial SOC value of 0.78, observe the battery's charging current drops to 80%.


Viewer: Scope : ee_dc_fast_charging_station (Vdc, Idc, Vbatt1, Vbatt2, Vbatt3, Ibatt1, Ibatt2, Ibatt3, SOCbatt1, SOCbatt2, SOCbatt3)
File
Tools
View
Simulation
Help
Vdc
Vbatt1, Vbatt2, Vbatt3
SOCbatt1, SOCbatt2, SOCbatt3
Idc
Ibatt1, Ibatt2, Ibatt3
Ready
T=10.000

# SiC MOSFET Parameterization Using Simulation Results from SPICE

This example shows how to generate lookup table data for a silicon carbide (SiC) metal-oxide-semiconductor field-effect transistor (MOSFET) from SPICE subcircuits by using the `ee.spice.semiconductorSubcircuit2lookup` function and parameterize an N-Channel MOSFET block in Simscape™ Electrical™.

### Open the SiC MOSFET Subcircuit

The subcircuit contains the SPICE model of the Infineon 1200V SiC Power MOSFET IMBG120R045M1. To open the subcircuit, at the MATLAB® command prompt, enter: edit Infineon-IMBG120R030M1.lib. You can download the SPICE library that contains the required model from the manufacturer website.

### Set the Required Lookup Table Parameters for the Function Input Arguments

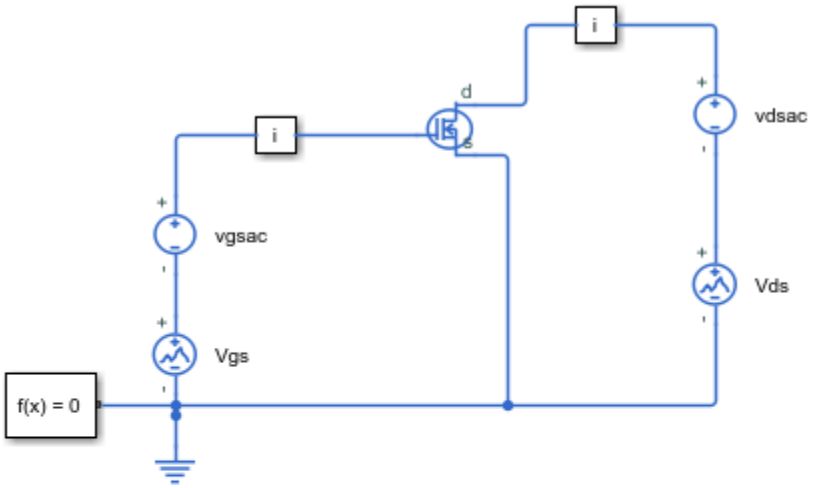
To run the `ee.spice.semiconductorSubcircuit2lookup` function, you need to characterize the device by specifying the corresponding subcircuit file path, the subcircuit name, and the name-value argument pair. For more information, see the `ee.spice.semiconductorSubcircuit2lookup` documentation page.

### Generate Lookup Table Data From SPICE Subcircuit File

Use the `ee.spice.semiconductorSubcircuit2lookup` function to generate the SiC MOSFET channel, diode, and capacitance characteristics.

### Open the N-Channel MOSFET (Lookup Table-Based) Model

Open the model `ee_SiC_mosfet_tabulated.slx`




**SiC MOSFET Parameterization Using Simulation Results from SPICE**

1. SPICE subcircuit (see SPICE netlist)
2. Set lookup table parameters (see code)
3. Compare results with SPICE  (see code)
4. Explore simulation results using Simscape Results Explorer
5. Learn more about this example

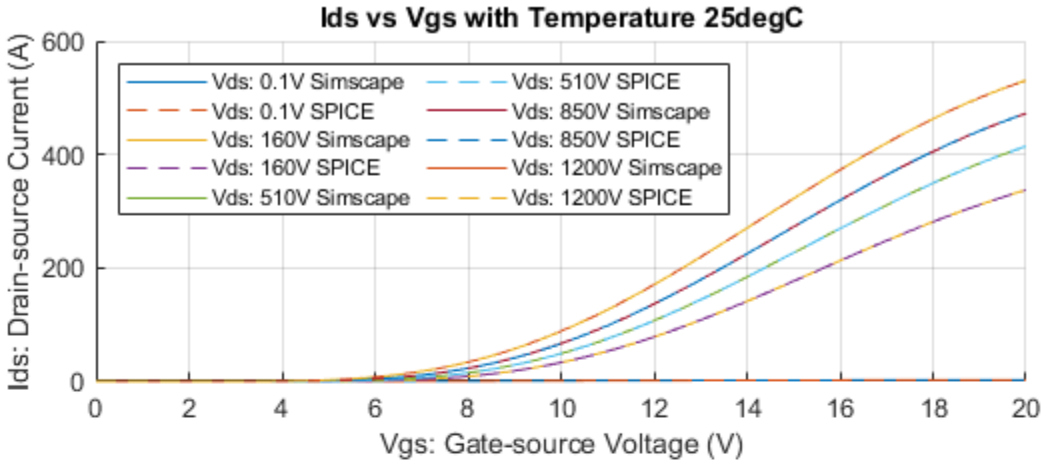
Copyright 2022 The MathWorks, Inc.

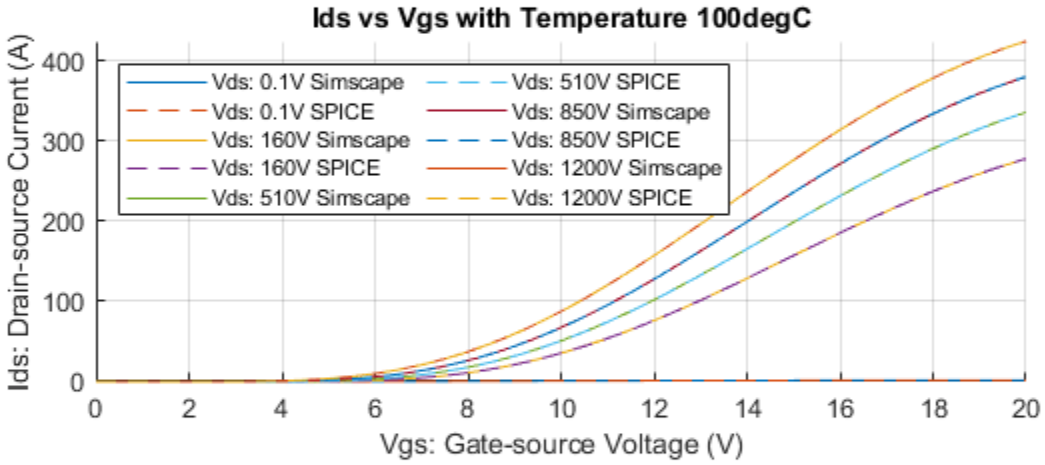
### Set Lookup Table Parameters for the N-Channel MOSFET (Lookup Table-Based) Model

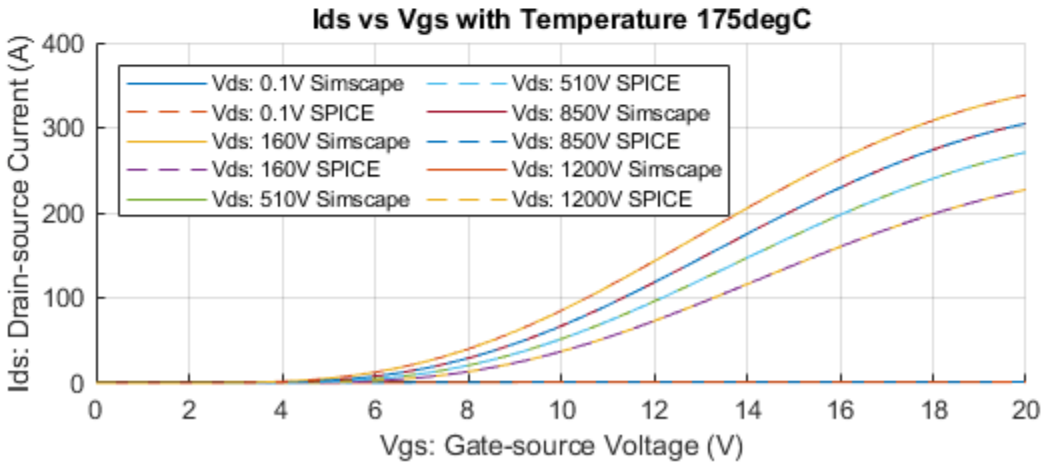
You can parameterize an N-Channel MOSFET (Lookup Table-Based) in your model using the generated lookup table data.

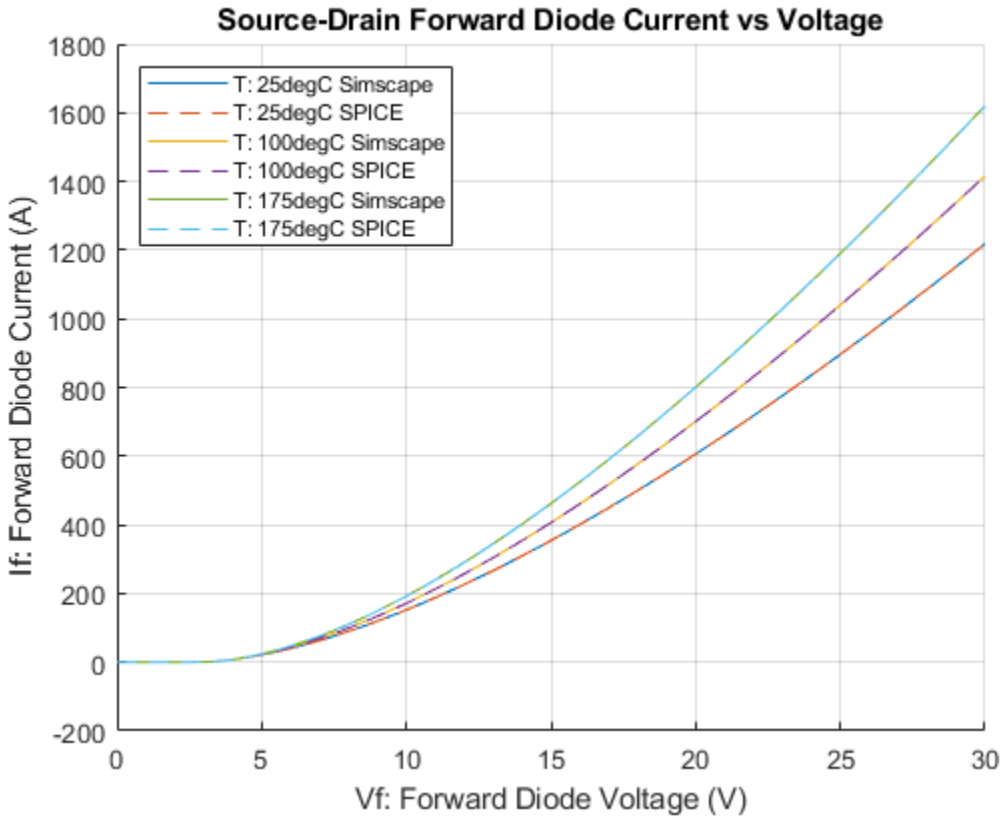
### Comparison of Characteristics

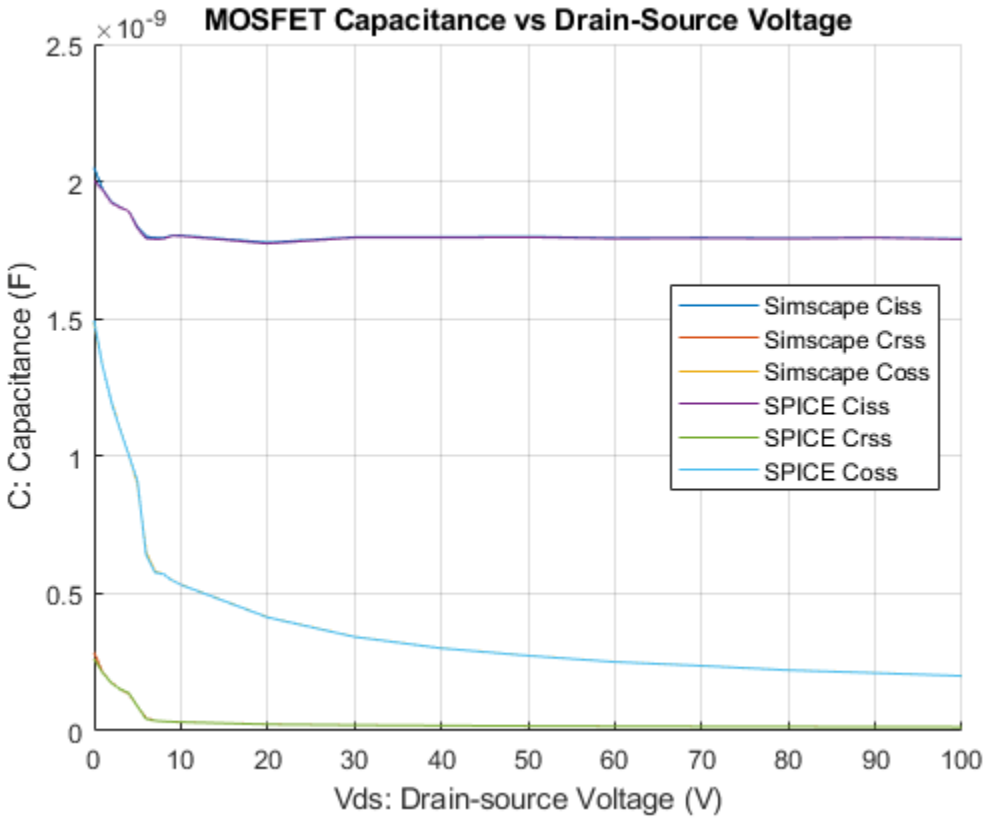
These plots compare the transfer characteristics, the diode characteristics, and the capacitance characteristics of the N-Channel MOSFET (Lookup table-based) with the SPICE subcircuit simulation results.


Ids vs Vgs with Temperature 25degC
Ids: Drain-source Current (A)
Vgs: Gate-source Voltage (V)
Vds: 0.1V Simscape
Vds: 0.1V SPICE
Vds: 160V Simscape
Vds: 160V SPICE
Vds: 510V Simscape
Vds: 510V SPICE
Vds: 850V Simscape
Vds: 850V SPICE
Vds: 1200V Simscape
Vds: 1200V SPICE


Ids vs Vgs with Temperature 100degC
Ids: Drain-source Current (A)
Vgs: Gate-source Voltage (V)
Vds: 0.1V Simscape
Vds: 0.1V SPICE
Vds: 160V Simscape
Vds: 160V SPICE
Vds: 510V Simscape
Vds: 510V SPICE
Vds: 850V Simscape
Vds: 850V SPICE
Vds: 1200V Simscape
Vds: 1200V SPICE


Ids vs Vgs with Temperature 175degC
Ids: Drain-source Current (A)
Vgs: Gate-source Voltage (V)
Vds: 0.1V Simscape
Vds: 0.1V SPICE
Vds: 160V Simscape
Vds: 160V SPICE
Vds: 510V Simscape
Vds: 510V SPICE
Vds: 850V Simscape
Vds: 850V SPICE
Vds: 1200V Simscape
Vds: 1200V SPICE


Source-Drain Forward Diode Current vs Voltage
T: 25degC Simscape
T: 25degC SPICE
T: 100degC Simscape
T: 100degC SPICE
T: 175degC Simscape
T: 175degC SPICE
If: Forward Diode Current (A)
Vf: Forward Diode Voltage (V)
1800
1600
1400
1200
1000
800
600
400
200
0
-200
0
5
10
15
20
25
30




The N-Channel MOSFET (Lookup Table-Based) model validates the power MOSFET IMBG120R045M1 against SPICE subcircuit results.

**Tidy Up**

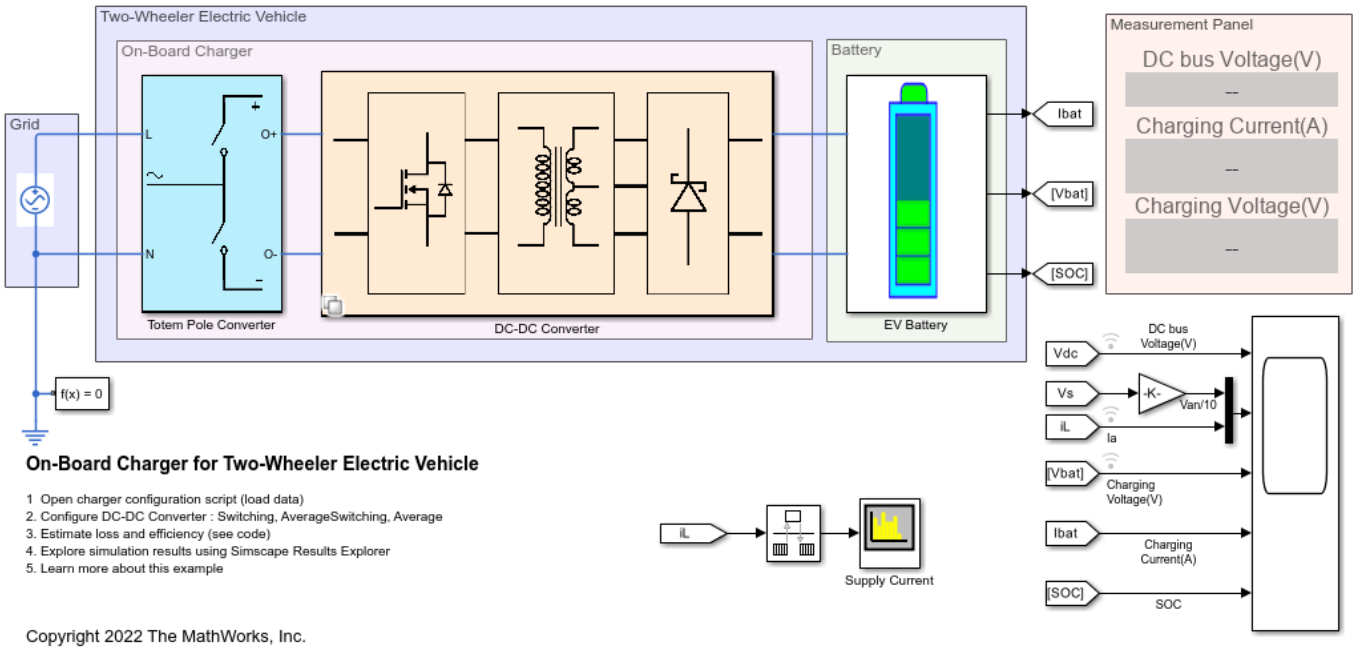
# On-Board Charger for Two-Wheeler Electric Vehicle

This example shows how to model an on-board charger for a two-wheeler vehicle by using an AC-DC converter and a DC-DC converter to achieve unity power factor (UPF) operation and constant current (CC) battery charging.

Efficiency plays a crucial role in the design of an on-board charger due to space constraints. This application requires specific converter topologies that allow the charger to achieve higher efficiency. The estimation of the harmonics injected into the grid helps to evaluate the grid compliance.

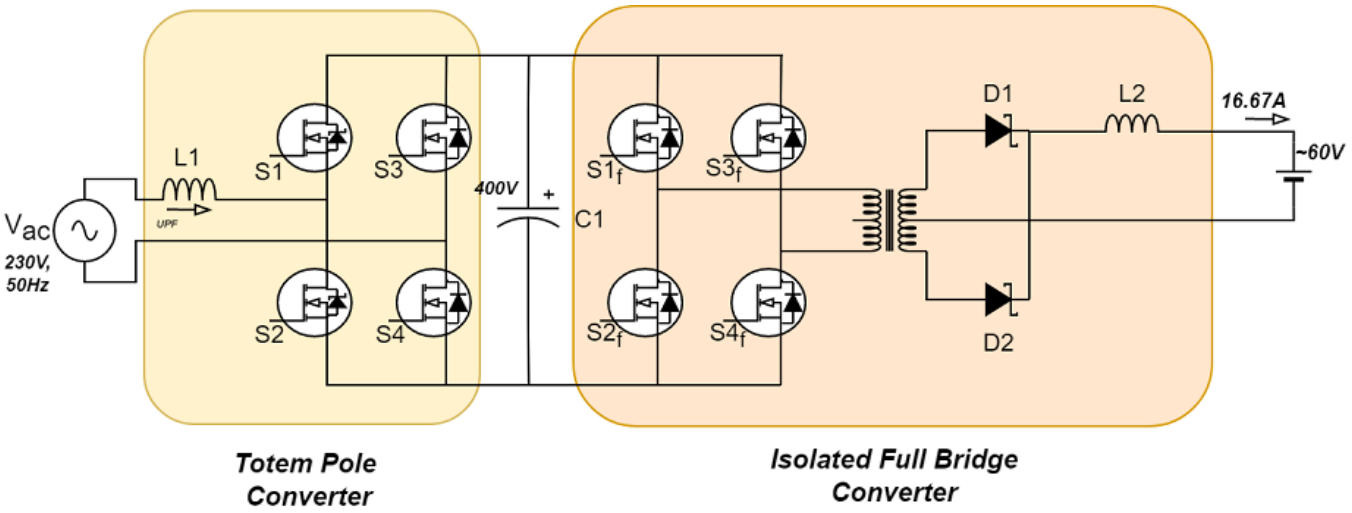
You can use this example to design the on-board charger and estimate its efficiency and harmonics.

### Model Overview




This example comprises three main components: the grid, the on-board charger, and the EV battery. The charger is rated at 1 kW, 230 V, and 50 Hz. The EV battery is a 60V, 50Ah/3kWh Li-ion battery. The intermediate capacitor and battery charging current are maintained at 400V and 16.67A respectively.

This figure shows the circuit for the on-board charger:

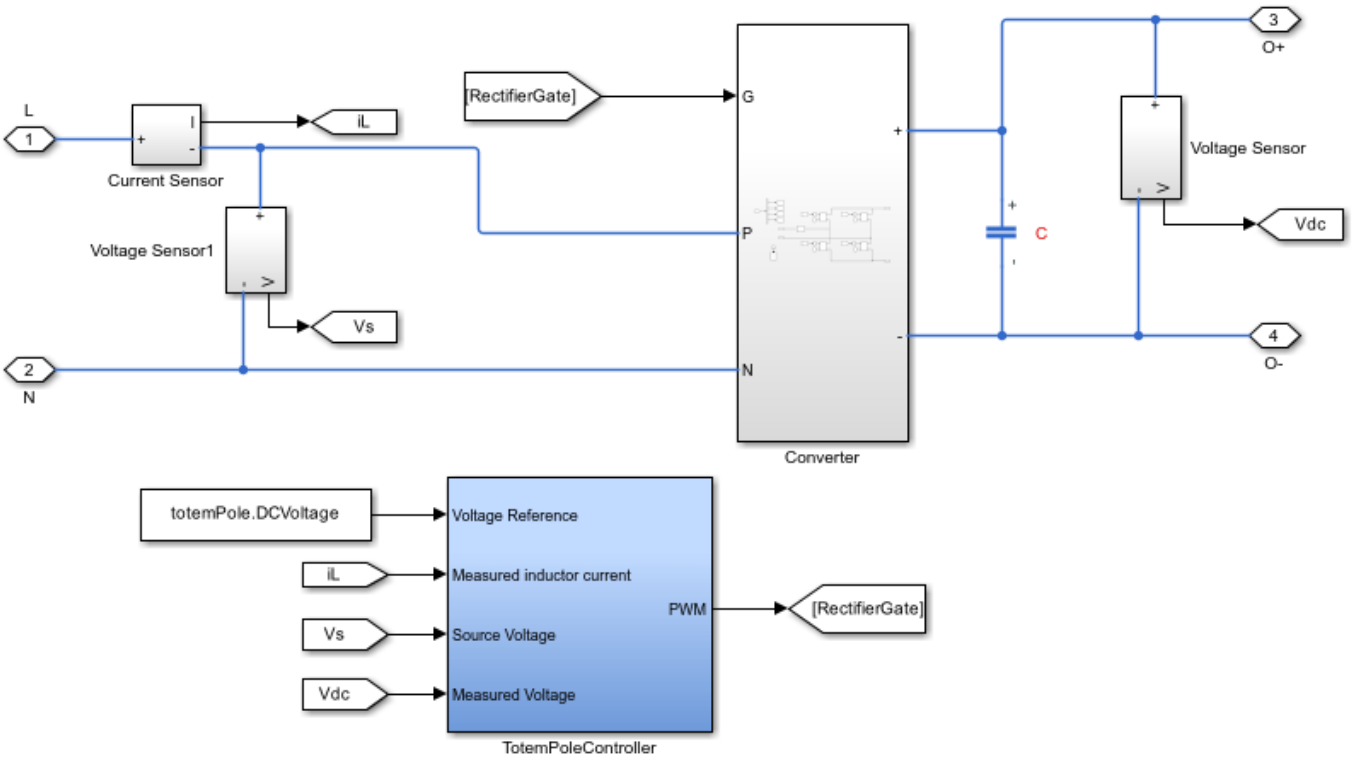



For a two wheeler vehicle, this example models the AC-DC converter by using a totem pole converter instead of a boost PFC converter due to its higher efficiency. The converter is rated at 400 V and 10 A.

To provide galvanic isolation between the battery and the grid, this example uses an isolated full bridge converter. The high-frequency inverter connected to the primary side of the transformer is rated at 400 V and 6 A. The diodes connected to the secondary side of the transformer are rated at 150 V and 20 A.

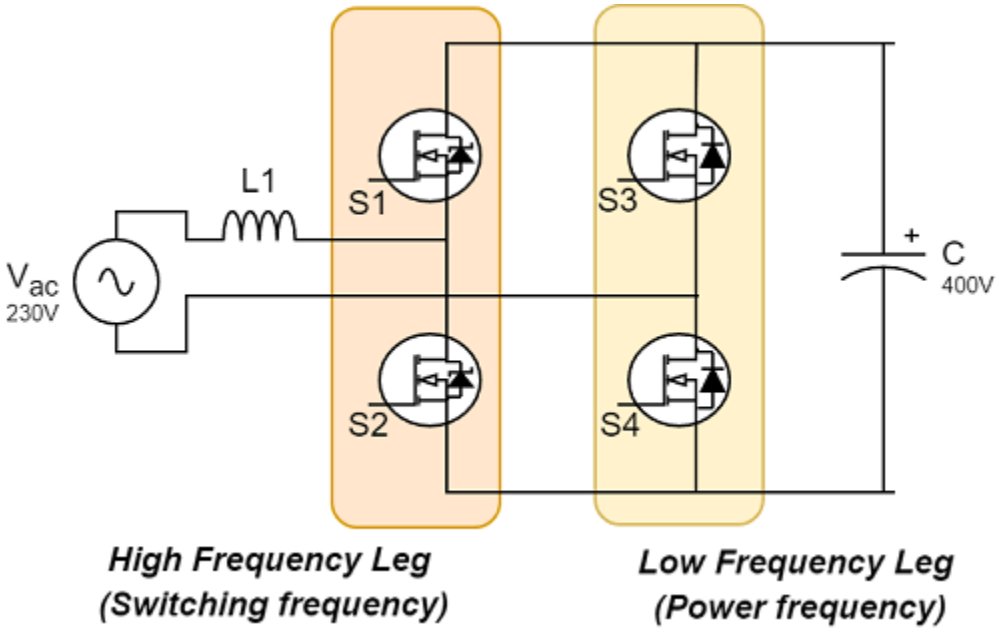
The I-V characteristics and the switching loss curves of the devices parameterize the Mosfet (Ideal, Switching) and Diode blocks. The `ee_onboard_charger_two_wheeler_SwitchData` MAT file contains the device parameters and the I-V curve data of the semiconductor devices in the model. You can obtain this data from the selected device datasheet.

### Totem Pole Converter




A totem pole converter is a derived topology of a boost-PFC converter and it achieves higher efficiency compared to all the other boost-PFC derived topologies.

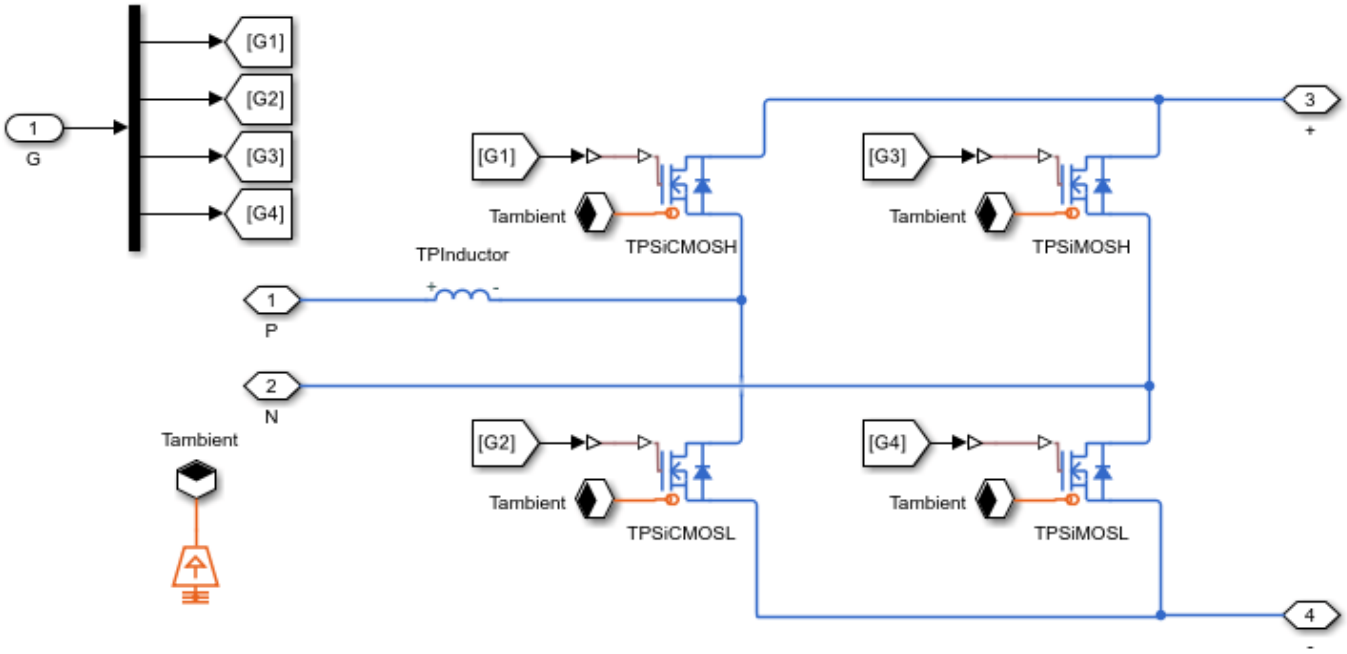
This figure shows the circuit for the totem pole converter:



A totem pole converter comprises a high-frequency leg and a low-frequency leg. The high-frequency leg operates at the switching frequency. The high-frequency legs use wide band gap (WBG) devices due to faster switching times and low switching energy.

The low-frequency leg operates at power frequency. Low-frequency legs use silicon-based diodes or MOSFETs for unidirectional or bidirectional power flow. The model in this example uses silicon carbide (SiC) and silicon (Si) based MOSFETs for the high-frequency and low-frequency legs.

The controller achieves the unity power factor (UPF) at supply terminals and maintains the intermediate capacitor voltage at a set value.

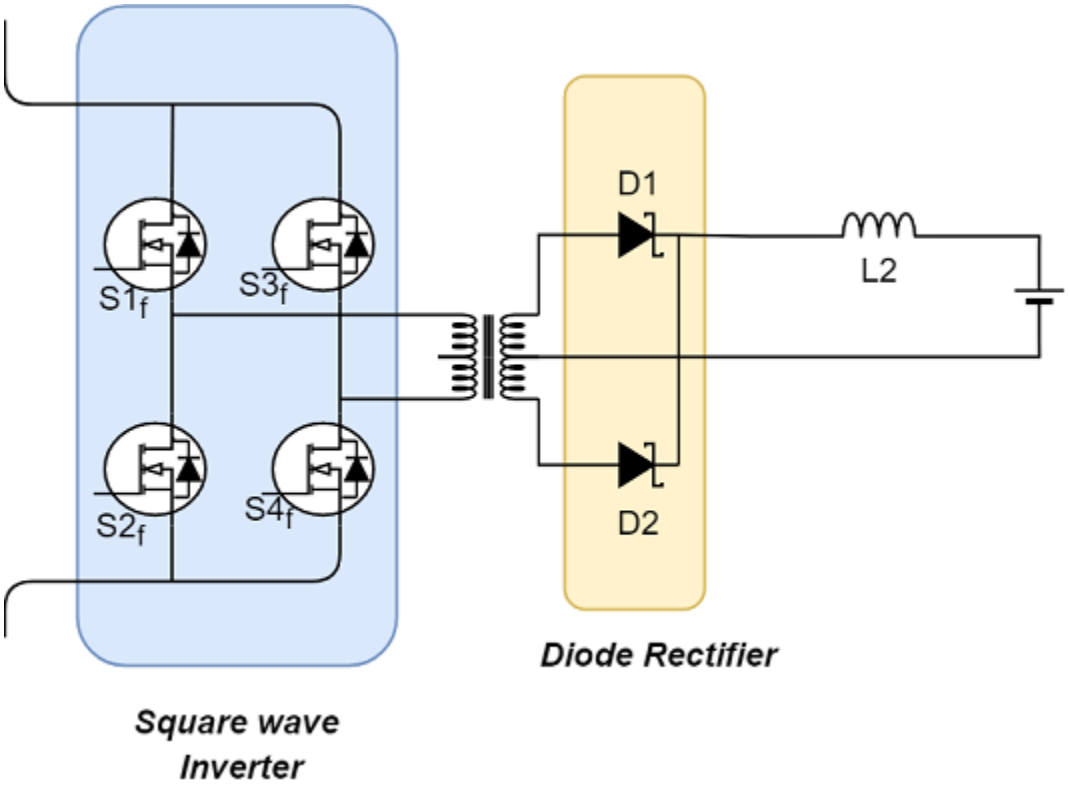



### Isolated Full Bridge Converter

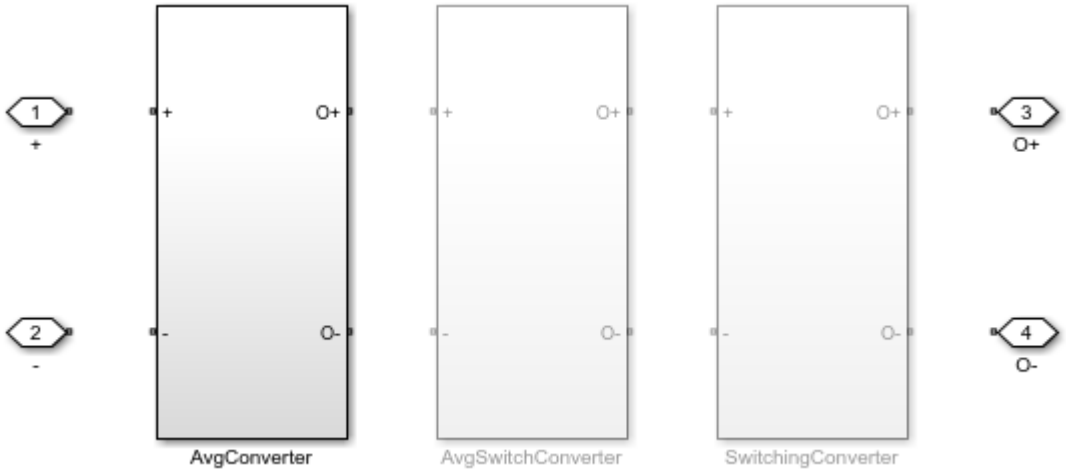
This example models the DC-DC converter by using an isolated full bridge converter topology. This is a double-ended topology and has higher power capability. It comprises three components: a square-wave inverter, a centre tapped transformer to improve the efficiency of the converter, and a diode rectifier. The diode rectifier uses Schottky diodes due to their lower reverse recovery losses.

The controller maintains the output current at a reference value to emulate the constant current mode of the battery charging.

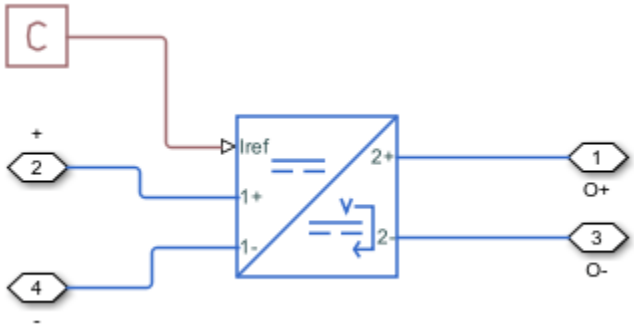
This figure shows the circuit for an isolated full bridge converter:



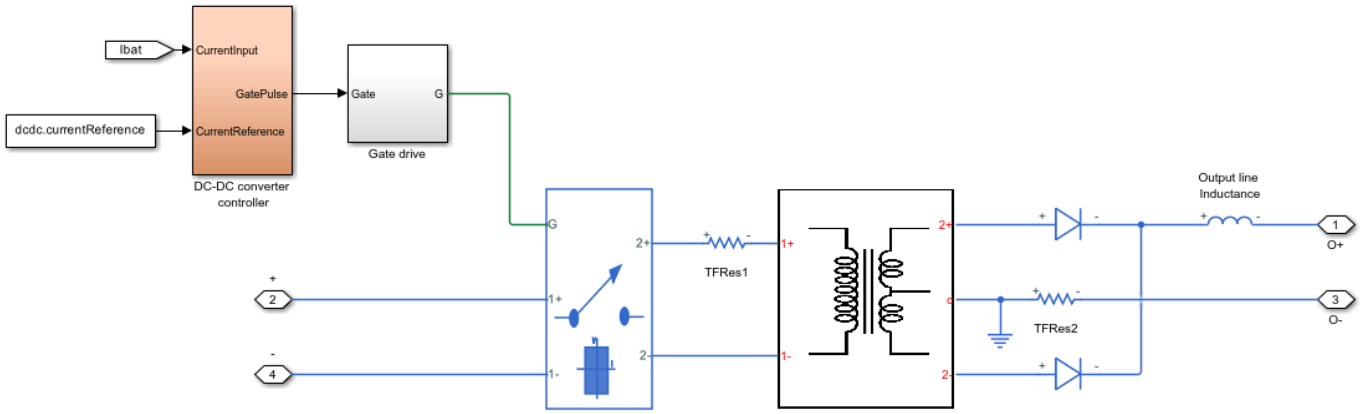

You can model the isolated DC-DC converter with three fidelity levels:



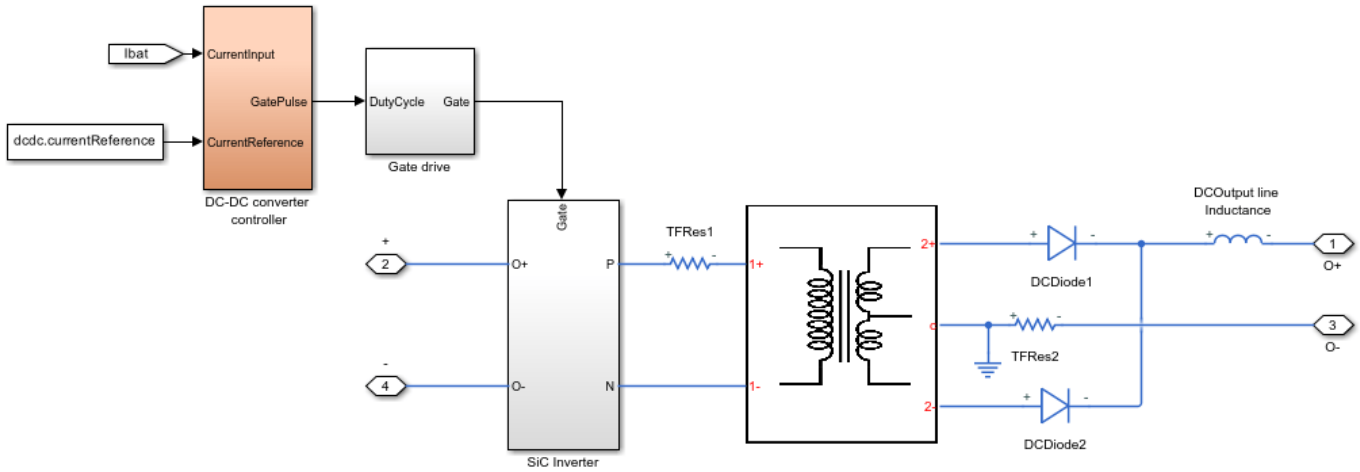

- Average fidelity — This variant achieves fastest simulation times with slight reduction in accuracy in transient condition. This is the default option.




- Averaged Switching — This variant uses averaged switching to achieve a faster simulation time. You can use this fidelity level to verify the operation of the converter.



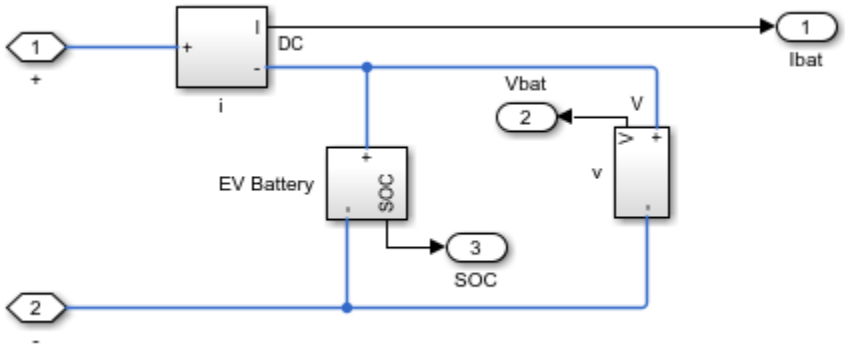

- Switching — Use this variant to calculate converter losses and to estimate its efficiency.




### EV Battery Pack Overview

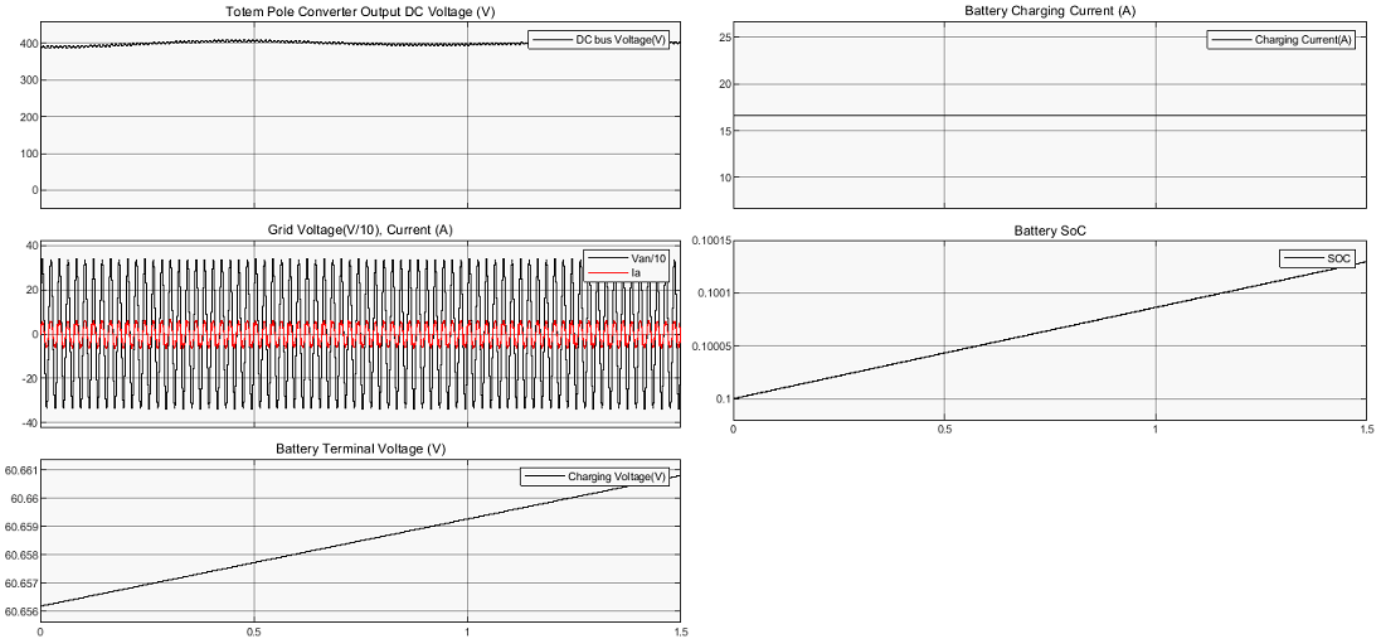
The EV battery pack models the battery cells connected in series and in parallel and the sensors that measure the battery terminal voltage and output current. A Battery (Table-based) block from the

Simscape Electrical (TM) library models the individual cell of the battery back. The Battery (Table-based) block uses the Panasonic NCRBD18650 pre-parameterization, which is rated at 3.6 V, 3.03 Ah. The configuration of the batteries in the battery pack meets the required voltage and ampere-hour specifications.

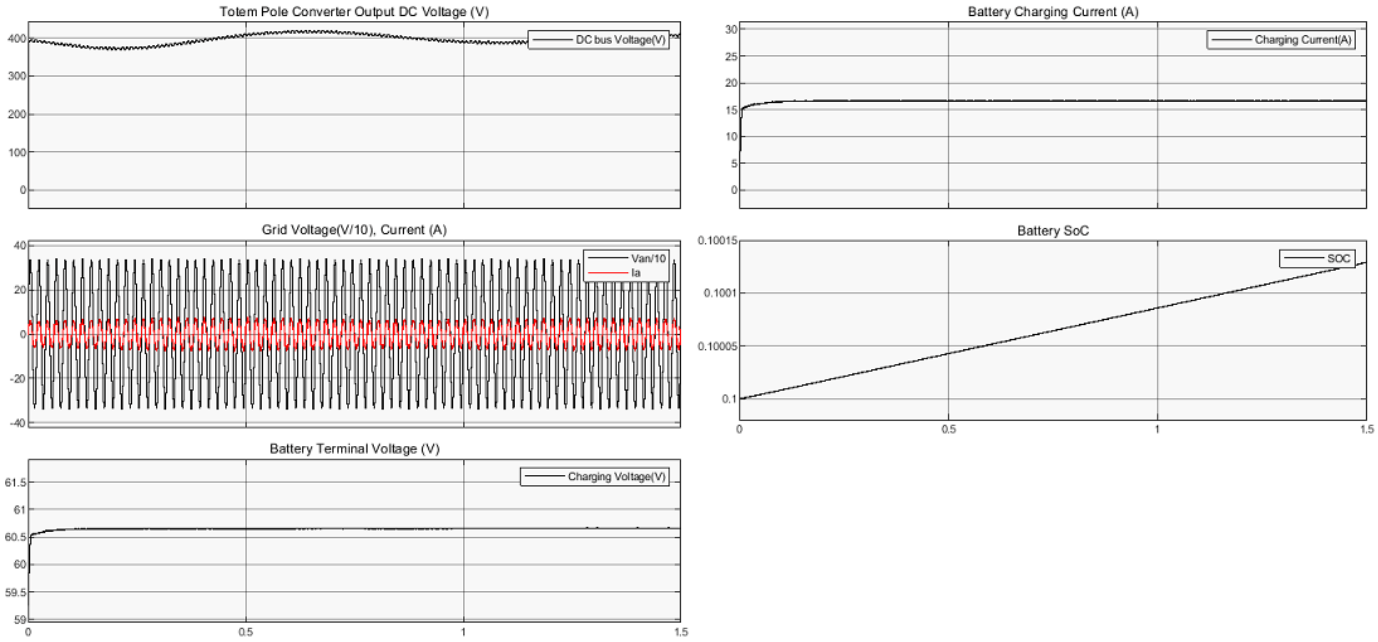



### Simulation Results from Simscape Logging

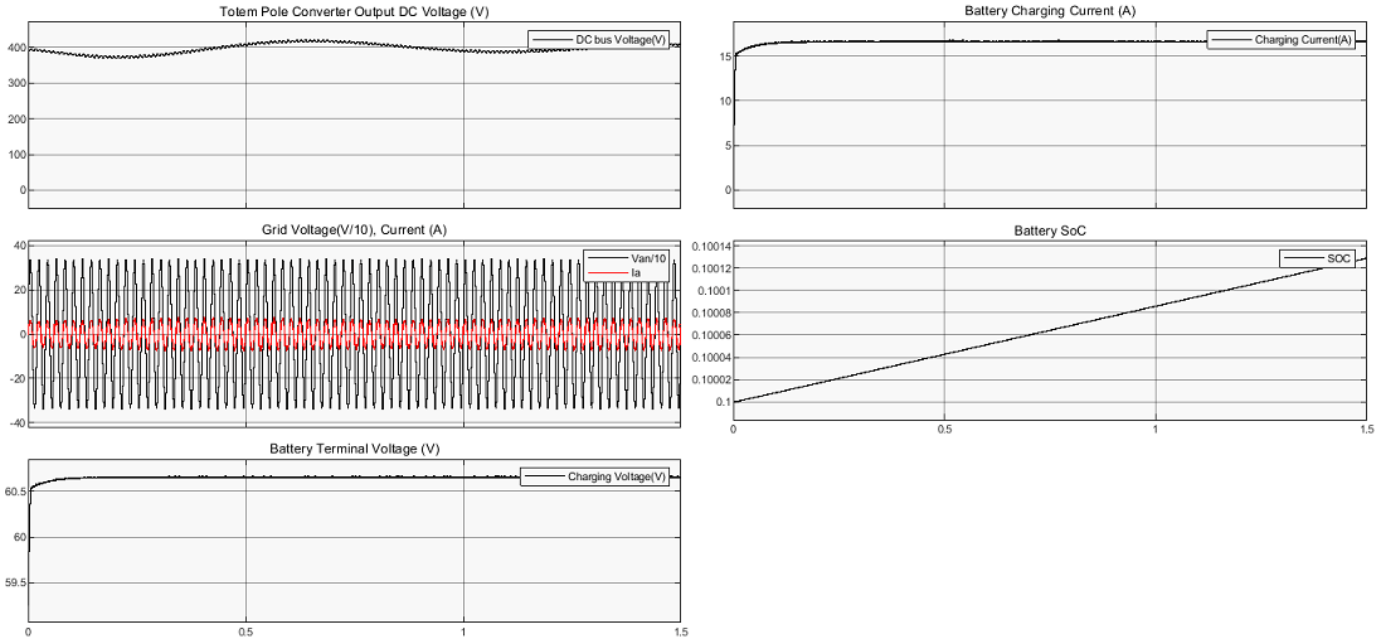
The plot belows shows the DC bus voltage, grid current, battery terminal voltage, and charging current for the ideal switching totem pole converter and for the average DC-DC converter. The intermediate capacitor voltage settles at 400V, the grid current is in phase with the grid voltage, and the battery charging is constant at 16.67A.




The plot belows shows the DC bus voltage, grid current, battery terminal voltage, and charging current for the ideal switching totem pole converter and for the average switching DC-DC converter. The average switching variants capture the switching events of the converters. During the transient state, due to the switching events, the voltage amplitude of the intermediate capacitor swings more than the average variant in the previous plot.




The plot belows shows the DC bus voltage, grid current, battery terminal voltage, and charging current for the ideal switching totem pole converter and for the switching DC-DC converter. This variant is mainly intended for loss and efficiency calculation.

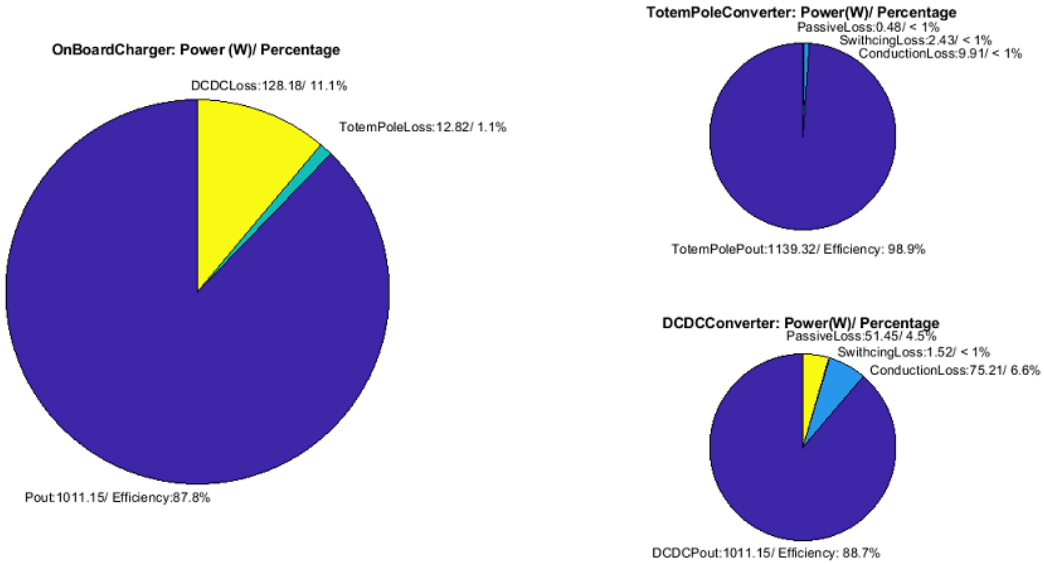



### Efficiency Calculation

An on-board charger for a two-wheeler vehicles requires a highly efficient converter due to the lack of space for heat dissipation. You can calculate the converter losses to analyze and modify the components responsible for higher losses according to your requirements. In the totem pole

converter, the calculation of the losses considers the device losses, such as conduction and switching losses, and the passive loss from the inductor. In the DC-DC converter, the calculation of the losses considers the device losses, such as conduction and switching losses, and the passive losses from the transformer and the inductor.

To estimate the losses, simulate the model in "Switching fidelity" for DC-DC converter until you achieve a steady state. Specify the start and end time at which the losses calculation considers the voltages and current values. These time values must be less than the simulated end time value. Run the script `ee_onboard_charger_two_wheeler_LossAnalysis` to analyse the losses and calculate the efficiency. This script utilizes the built-in function `ee_getPowerLossSummary` to calculate individual component losses.




These efficiency values vary according to the components you use in your model.

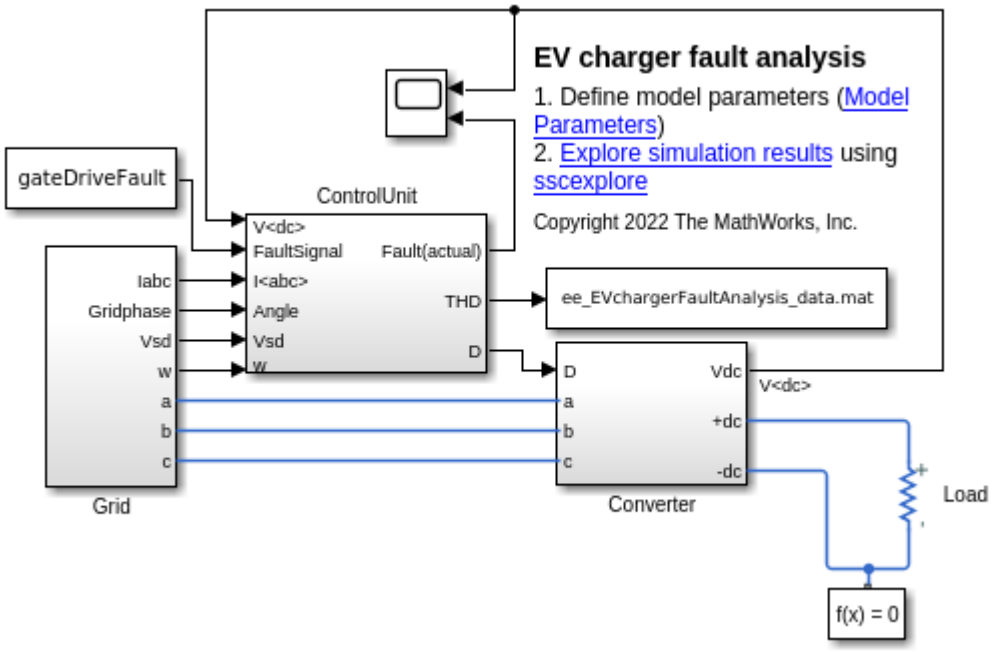
# Fault Detection of Electric Vehicle Charger

This example analyses the fault of an electric vehicle (EV) charger using Simscape Electrical™ to model the grid, the converter, and its control unit. The reliability of these chargers is an important factor in their adoption. In this example, you use measurements from the grid and the DC side of the converter to detect a gate driver fault in the converter. To analyze and detect a fault, first you generate synthetic data for different conditions with and without faults. Then you use this data to train a classification algorithm using the Classification Learner App. Finally, you use the trained model to identify or detect faults in any phase and to generate the code for deployment on hardware. You can extend this model for other system level variations or noises by having a much larger and comprehensive training dataset.

### Open EV Charger Model

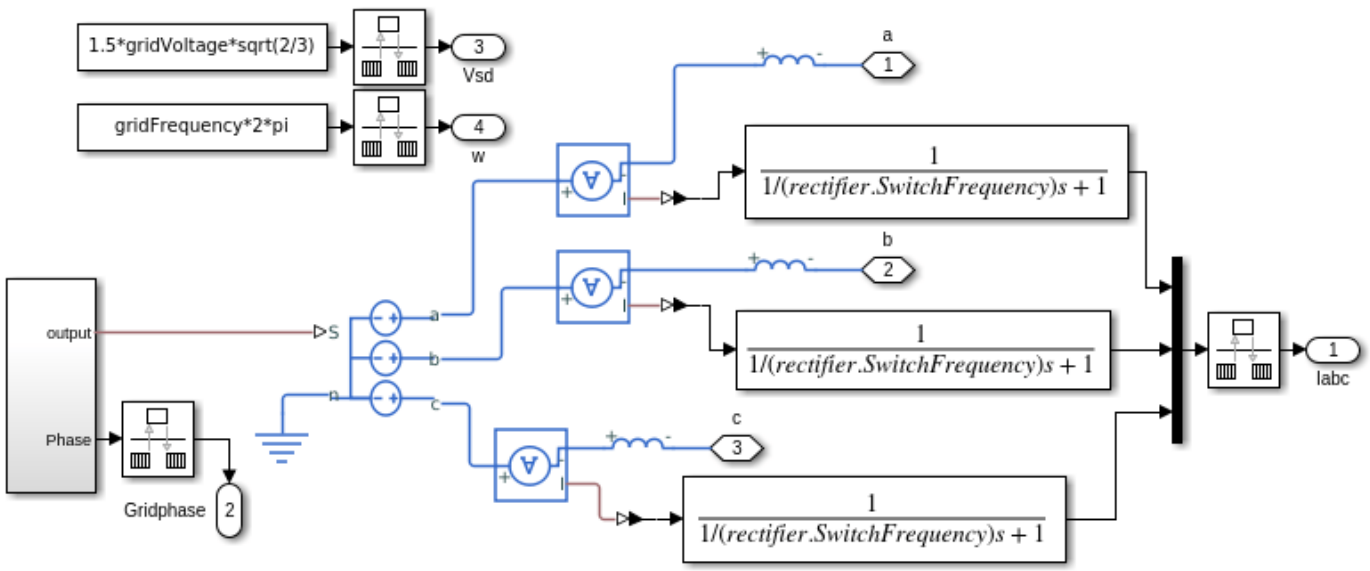
Open the model `ee_EVchargerFaultAnalysis`

```
open_system("ee_EVchargerFaultAnalysis.slx")
```

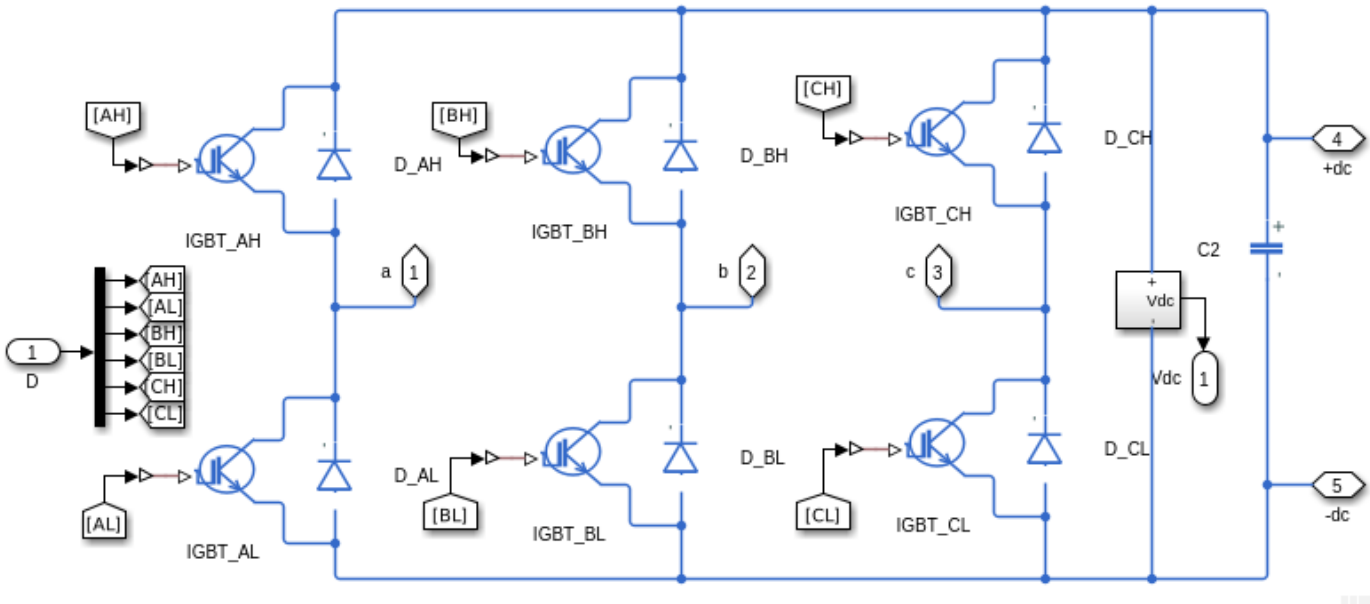



The model comprises three major subsystems: Grid, Converter, and ControlUnit. The resistor block, Load, represents the system load. In this example, you simulate the fault of gate drivers for one or more IGBT duty cycles. No other fault is considered. The gateDriveFault block defines a fault in the gate driver operation. The input to the Converter subsystem is a column vector of six elements, which represent the six IGBTs. Each element is either 0 (no fault) or 1 (fault). You can simulate uncertainties in system by setting the Resistance parameter of the Load block to different values. The `ee_EVchargerFaultAnalysis_data` MAT file stores the measured total harmonic distortion (THD) on the AC grid side.

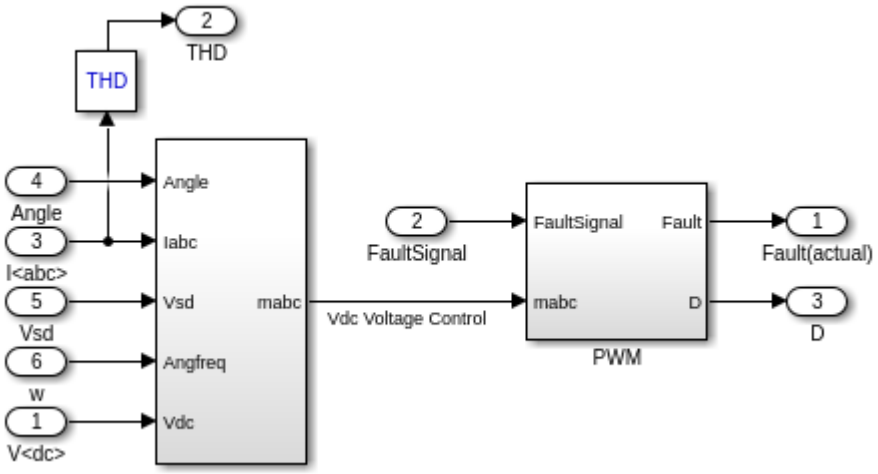
The Grid subsystem is modeled as an ideal voltage source. To model a grid with a different grid voltage and frequency, change the parameters in the `ee_EVchargerFaultAnalysis_ini` file.




The Converter subsystem is modeled as a three-phase, two-level converter using IGBT (Ideal, Switching) blocks. You can fault any of the six IGBTs by changing the duty cycle, D, to zero. To change the duty cycle, use the **gateDriveFault** parameter.




The ControlUnit subsystem controls the three-phase converter by using id-iq control, with unity power-factor operation, that enables a constant DC voltage of 800V.




In this example, you programmatically simulate multiple instances of fault behavior. The grid-side THD and the Load resistance values are used for training classification of faults. The classification aims to identify the gate driver on the faulted phases (a, b, c, or one of their combinations). To collect enough data to train the model, you run multiple simulations for different load conditions and generate synthetic data for fault and no fault scenarios.

### Generate Training Data

Define operating conditions, such as the grid voltage and the system load variation, for the simulation.

```
% Grid data
gridVoltOptions = 415;          % V,   Line-Line rms voltage
loadOptions = [8 9 10 11 12];  % Ohm, System load
```

To run simulations for all grid and load options and create the training data for the Classification Learner, at the MATLAB command prompt, enter `ee_EVchargerFaultAnalysis_training` to open the script.

This function generates the training data for fault and no fault scenarios. To generate the training data, the total harmonic distortion values from each simulation are averaged over a time period of [`tStart`, `tEnd`].

### Train and Validate Model

This example uses the MATLAB™ Classification Learner App to identify and train a classification learning algorithm for the system. For more information, see Train a Classification Learning Algorithm. First, load the training data in your workspace. The `ee_EVchargerFaultAnalysis_trainingData` MAT file already stores the pre-generated training data. The training labels denote faulted or unfaulted status (1 or 0) of the gate drives corresponding to the a, b, and c phases of the AC grid. It is denoted using a three-digit number, where the leftmost digit represents the a-phase of the gate driver, middle digit represents the b-phase of the gate driver, and the rightmost digit represents the c-phase of the gate driver. Each digit can be 0 or 1. Only one gate driver or a combination of two gate drivers might fault.

```
trainingData=load("ee_EVchargerFaultAnalysis_trainingData.mat").trainingData;
trainingData.gridDCvolt = [];
```

```
trainingData.driverFault = categorical(trainingData.driverFault);
trainingData.driverFault = categorical(trainingData.driverFault,...
    ["0","1","10","11","100","101","110"],"Ordinal",true);
```

The first 4 columns in `trainingData` represent the data for training and the fifth column denotes the labels marked against each set of data point.

Load `trainingData` workspace parameter in the Classification Learner App and train all available models on the data. The Ensemble Bagged Trees is the best trained model with an accuracy of over 90%.

Export the trained model from the App and save it in a `ee_EVchargerFaultAnalysis_trainClassifier.m` file. To obtain the trained classifier, at the MATLAB command prompt, enter:

```
[trainedClassifier, validationAccuracy] = ...
    ee_EVchargerFaultAnalysis_trainClassifier(trainingData)
```

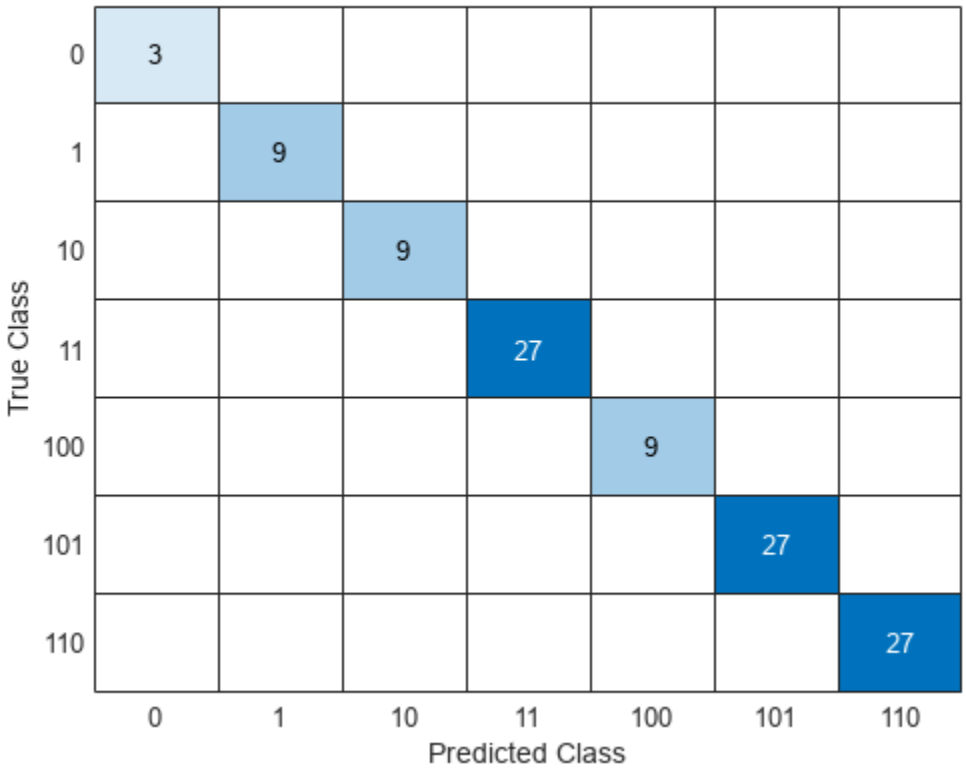
```
trainedClassifier = struct with fields:
                predictFcn: @(x)ensemblePredictFcn(predictorExtractionFcn(x))
          RequiredVariables: {1x4 cell}
    ClassificationEnsemble: [1x1 classreg.learning.classif.ClassificationBaggedEnsemble]
                     About: 'This struct is a trained model exported from Classification Learner
              HowToPredict: 'To make predictions on a new table, T, use: ...'


validationAccuracy = 0.9081
```

### Verify Model Predictions

To verify the model trained using the Classification Learner on an entirely new dataset, at the MATLAB command prompt, enter `ee_EVchargerFaultAnalysis_verify` to open the script.

The `ee_EVchargerFaultAnalysis_testData` MAT file contains pre-generated test data for this example. To verify the accuracy of the trained model, load the test data and run these commands.

```
testData=load("ee_EVchargerFaultAnalysis_testData.mat").testData;
testData.gridDCvolt = [];
testData.driverFault = categorical(testData.driverFault);
testData.driverFault = categorical(testData.driverFault,...
    ["0","1","10","11","100","101","110"],"Ordinal",true);
% Predicted labels
pred_fault = trainedClassifier.predictFcn(testData(:,1:4));
% Actual labels
real_fault = testData(:,5);
confusionchart(pred_fault,table2array(real_fault))
```




The Predicted Class and the True Class match for all the test data points. This shows a very good success rate in identifying the right kind of fault or no fault.

### Generate and Deploy Code

To deploy the `trainedClassifier` model trained with the Classification Learner App, run these steps.

```
saveLearnerForCoder(trainedClassifier.ClassificationEnsemble,...
    "ee_EVchargerFaultAnalysis_codegen");
```

For more information, see Code Generation and Classification Learner App. To simulate for a different system load and create a codegen model, run these commands.

```
% Simulation parameters
systemLoadR=11.2;
gridVoltage=415;
gateDriveFault=[1;0;0;0;0;0]; % Gate Driver for AH faulted (a phase)
realFault_a=min(1,gateDriveFault(1,1)+gateDriveFault(2,1));
realFault_b=min(1,gateDriveFault(3,1)+gateDriveFault(4,1));
realFault_c=min(1,gateDriveFault(5,1)+gateDriveFault(6,1));
%
% Check fault during time sample of 0.25 (= tEnd-tStart) seconds
tStart = 0.2;
tEnd = 0.45;
```

Run the `ee_EVchargerFaultAnalysis` SLX file to save the data in the `ee_EVchargerFaultAnalysis_data` MAT file. Use the measured data (THD, system load) to detect if there was one or more faults in the gate drivers.

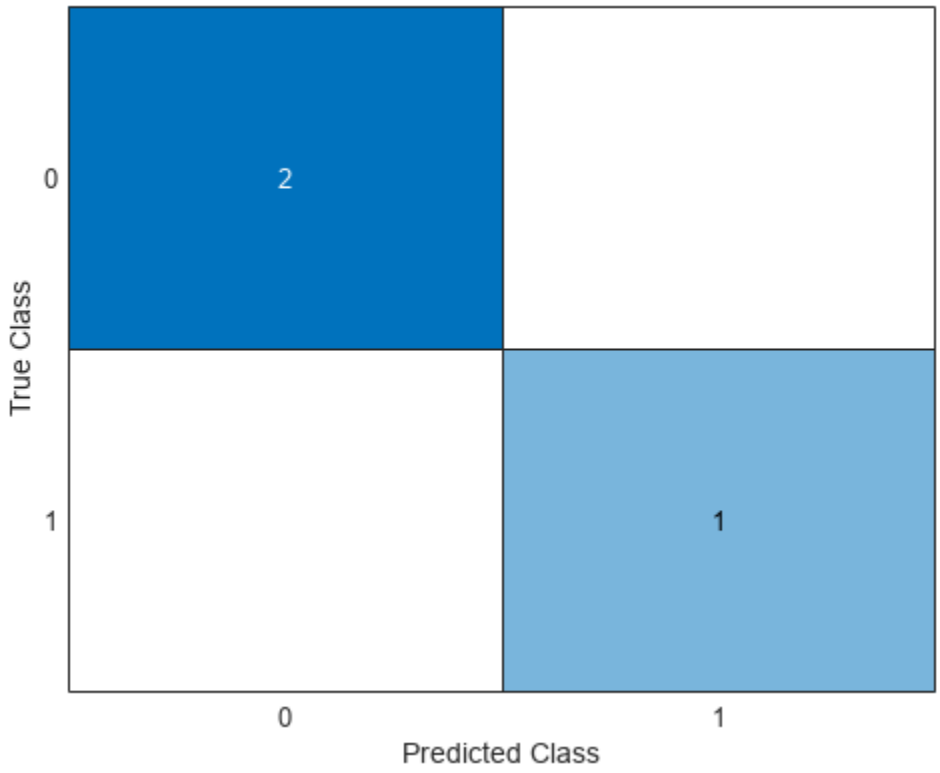
```
thd=load("ee_EVchargerFaultAnalysis_data.mat");
startData=find(thd.THD.Time==tStart);
endData=find(thd.THD.Time==tEnd);
```

The `ee_EVchargerFaultAnalysis_codegenFunc` function calls the `ee_EVchargerFaultAnalysis_codegen` MAT file that stores the trained classification suited for code generation.

```
gridTHD_a=thd.THD.Data(startData:endData,1);
gridTHD_b=thd.THD.Data(startData:endData,2);
gridTHD_c=thd.THD.Data(startData:endData,3);
lenData=length(gridTHD_a);
sysLoad=ones(lenData,1)*systemLoadR;
% Measured Data
measData=[sysLoad,gridTHD_a,gridTHD_b,gridTHD_c];
% Predict if there is any fault detected
[predFault_a,predFault_b,predFault_c] = ...
    ee_EVchargerFaultAnalysis_codegenFunc(measData);
%
if isequal(predFault_a,realFault_a) && ...
        isequal(predFault_b,realFault_b) && ...
        isequal(predFault_c,realFault_c)
    disp('*** Prediction : Correct !!')

else
    disp('*** Prediction : Incorrect !!')
end
```

```
*** Prediction : Correct !!
```

```
confusionchart(...
    [predFault_a,predFault_b,predFault_c],...
    [realFault_a,realFault_b,realFault_c]);
```

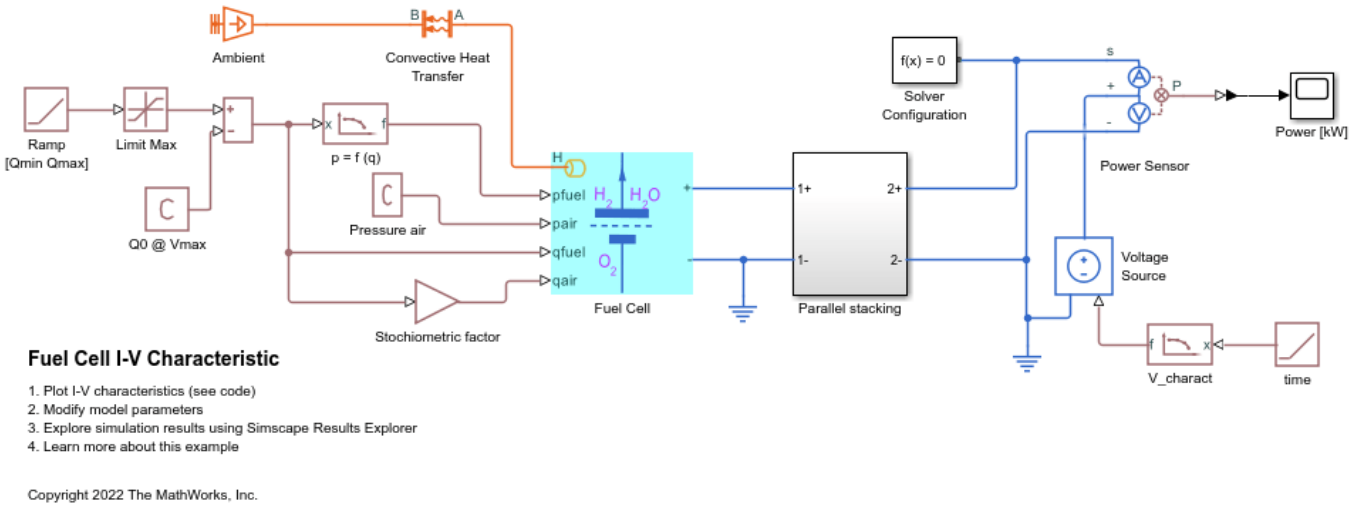



The `ee_EVchargerFaultAnalysis_codegenFunc` function is now ready for code generation. For more information, see Code Generation and Classification Learner App.

# Fuel Cell I-V Characteristic

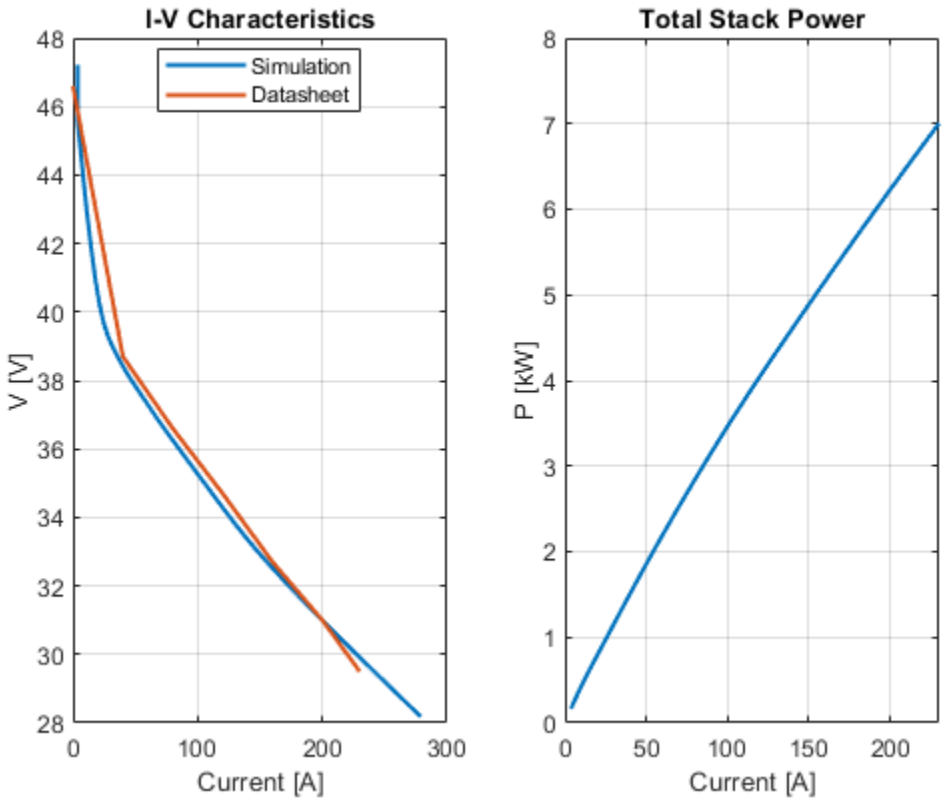
This model represents a harness for calibration of fuel cell (FC) parameters. This example extracts I-V and power characteristic curves from a datasheet. The datasheet also contains specific properties such as the minimum and maximum hydrogen and the air flows.

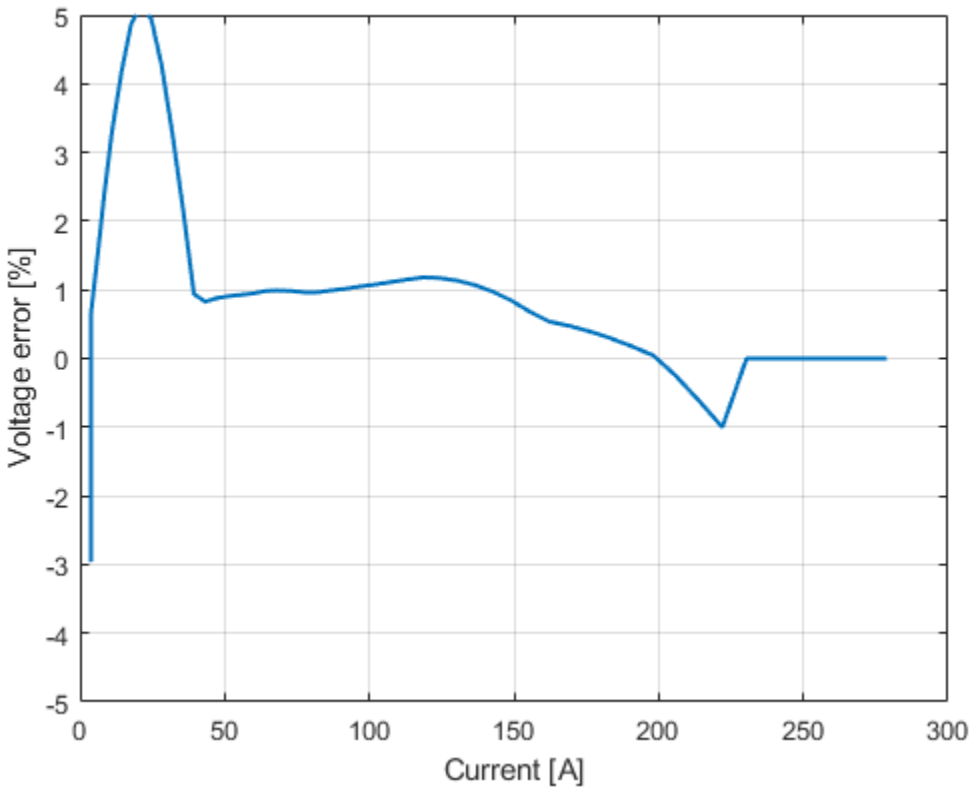
### Model




### Simulation Results from Simscape Logging

The plot below shows the simulated and datasheet I-V characteristic for the fuel cell, the total stack power, and the voltage error between the simulation and the datasheet.


I-V Characteristics
Simulation
Datasheet
V [V]
Current [A]
Total Stack Power
P [kW]
Current [A]



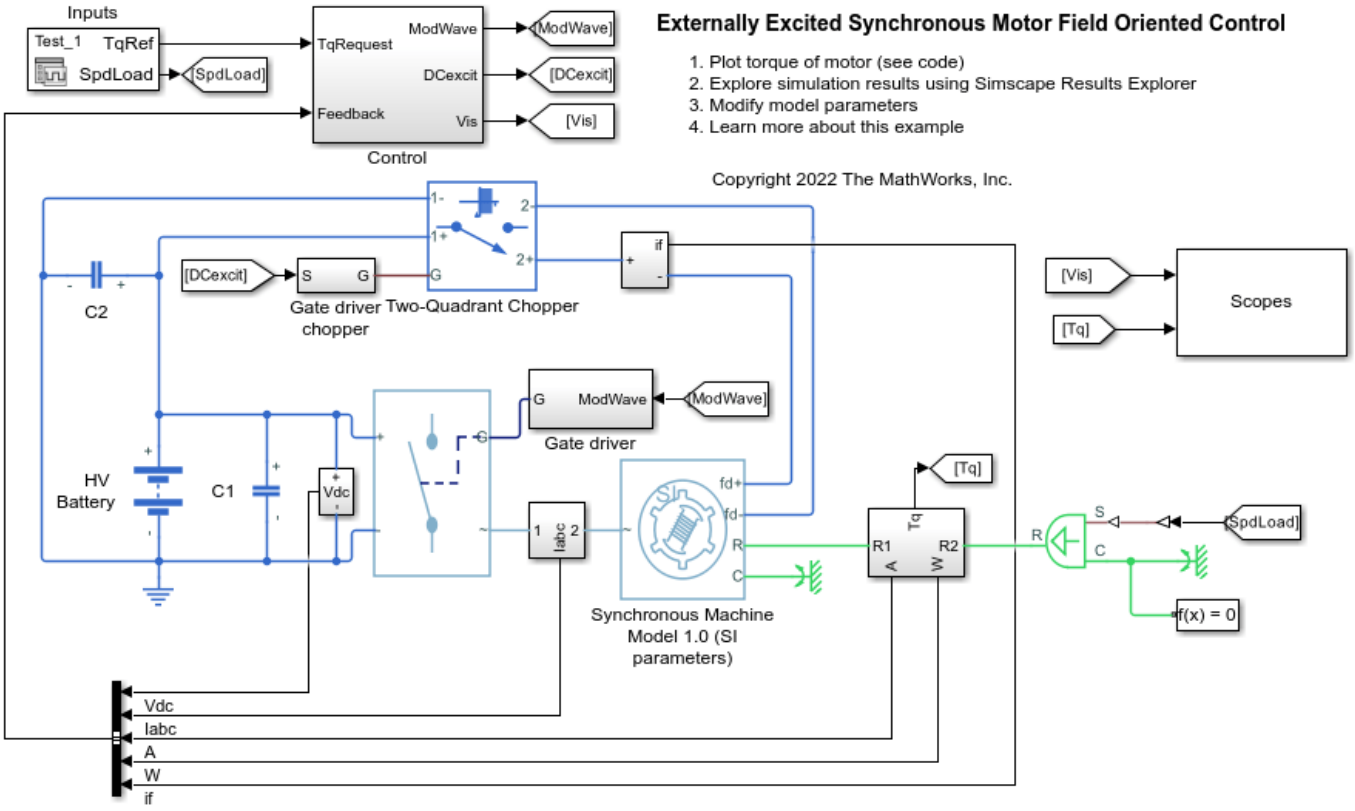

**Results from Real-Time Simulation**

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 50 microseconds.

# Externally Excited Synchronous Motor Field Oriented Control

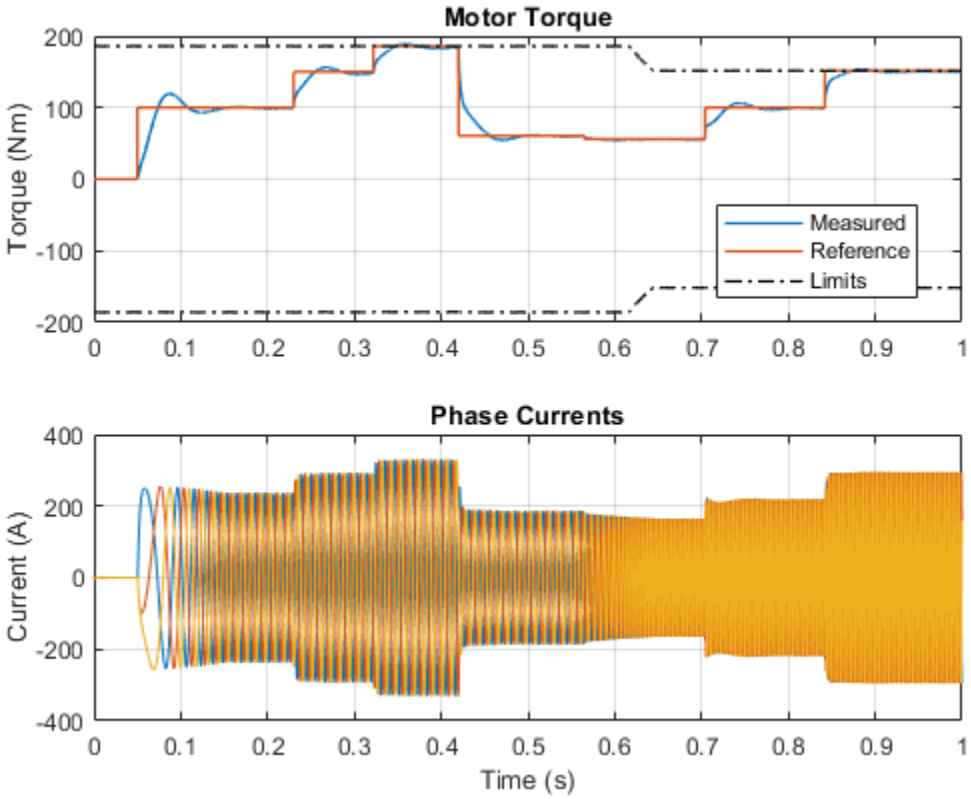
This example shows how to control the torque in an externally excited synchronous motor (SM) drive by using field-oriented control. A high-voltage battery feeds the SM through a controlled three-phase converter for the stator windings and through a controlled two-quadrant chopper for the rotor winding. Both converters are controlled using Averaged Switch and modulation waveforms for fast simulation. The implementation allows you to easily switch to PWM control signals. Current references for the d-axis, q-axis, and excitation are generated offline and implemented using 3-D lookup tables. Use the model to design and evaluate in real-time the SM control algorithm. For controller deployment on an embedded microcontroller, use optimized controllers from the Motor Control Blockset™ libraries. The Scopes subsystem contains scopes that allow you to see the simulation results.

### Model




### Simulation Results from Simscape Logging

The plot below shows the requested and measured torque for the test, as well as the phase currents in the electric drive.




### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 100 microseconds.

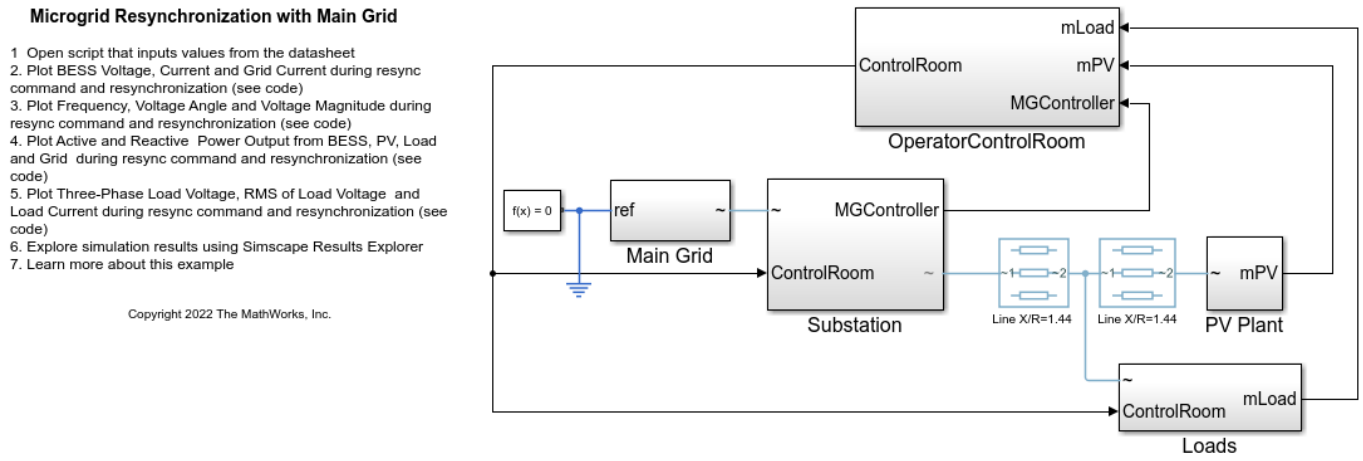
# Microgrid Resynchronization with Main Grid

This example shows how you can resynchronize an islanded microgrid with the main grid by using a battery energy storage system (BESS). The model in this example comprises a medium voltage (MV) microgrid model with a battery energy storage system, a photovoltaic solar park (PV), and loads. The microgrid can operate both autonomously (islanded) or in synchronization with the main grid. In this example, the microgrid is first in islanded mode. The resynchronization function then synchronizes the microgrid to the main grid. Finally, the breaker closes to connect the microgrid to the main grid. After the resynchronization, the battery system performs a power dispatch and the loads are changed.

The model includes a microgrid controller with all the necessary functions for the resynchronization. The resynchronization operation follows the IEEE Std 2030.7 standard.

### Model Overview

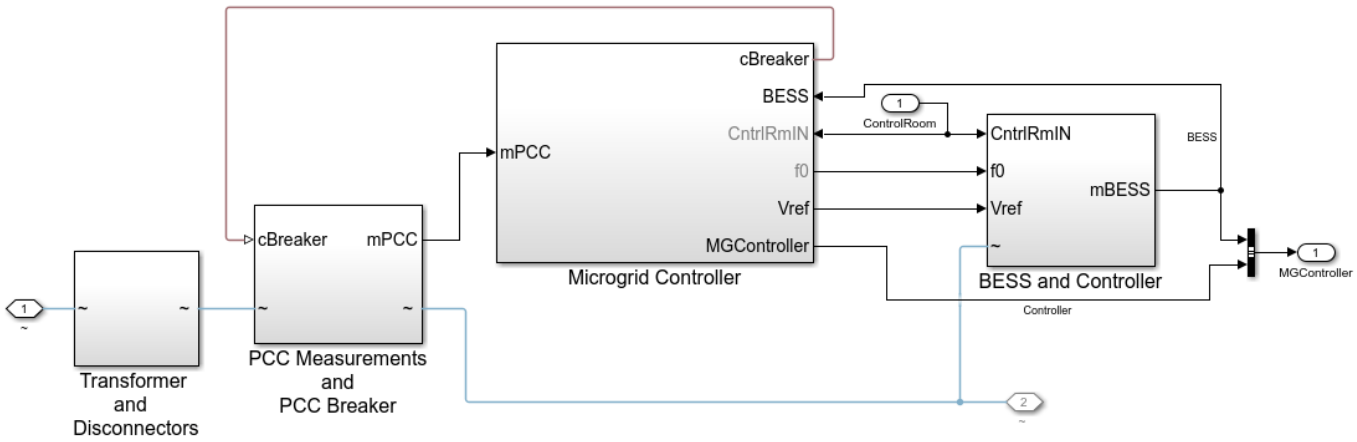
A substation connects the microgrid to the main grid. The loads and the PV are connected to the outgoing feeder. The model also contains a separate Operator Control Room subsystem. In a real system, the operator control room can be at the substation or somewhere distant to the microgrid.




### Components in Microgrid Resynchronization to Main Grid

These are the main components of the system:

*Substation * Subsystem that connects the microgrid to the main grid. It has a connecting breaker, disconnectors, and transformers to connect the main grid to the microgrid. The substation also contains the microgrid controller and the battery energy storage system.&

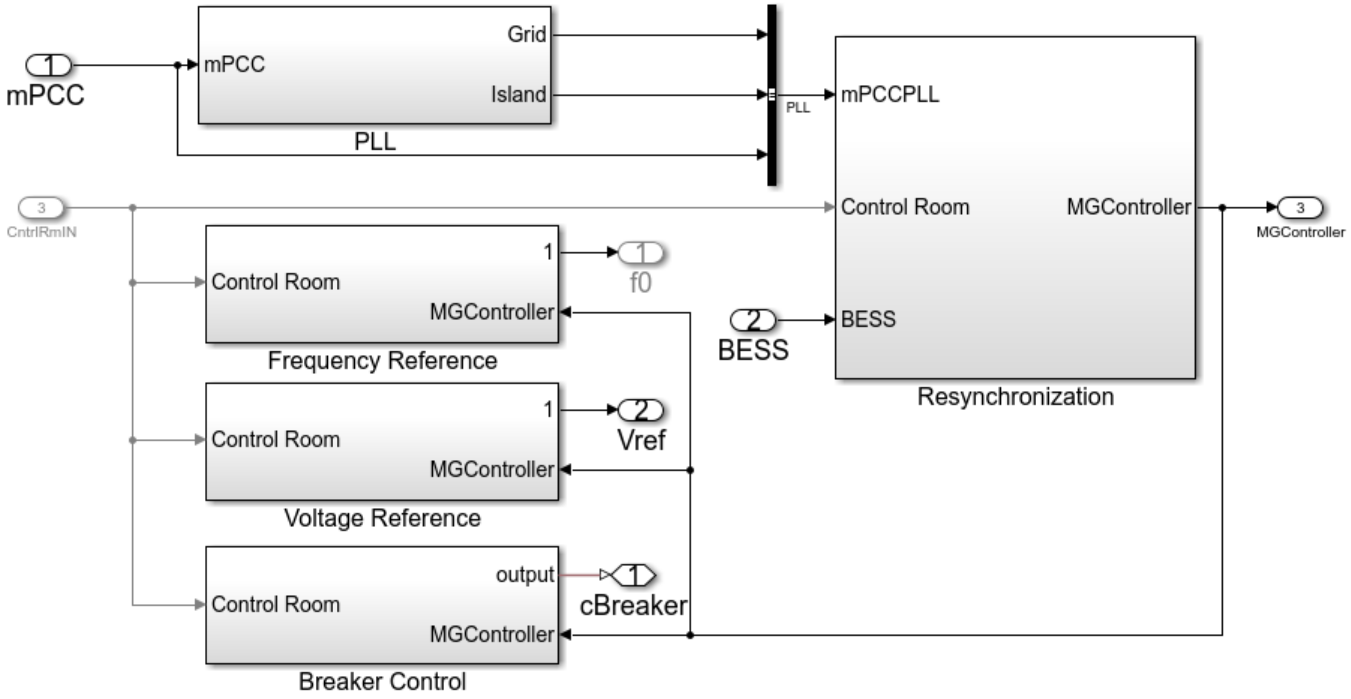



### Microgrid Controller

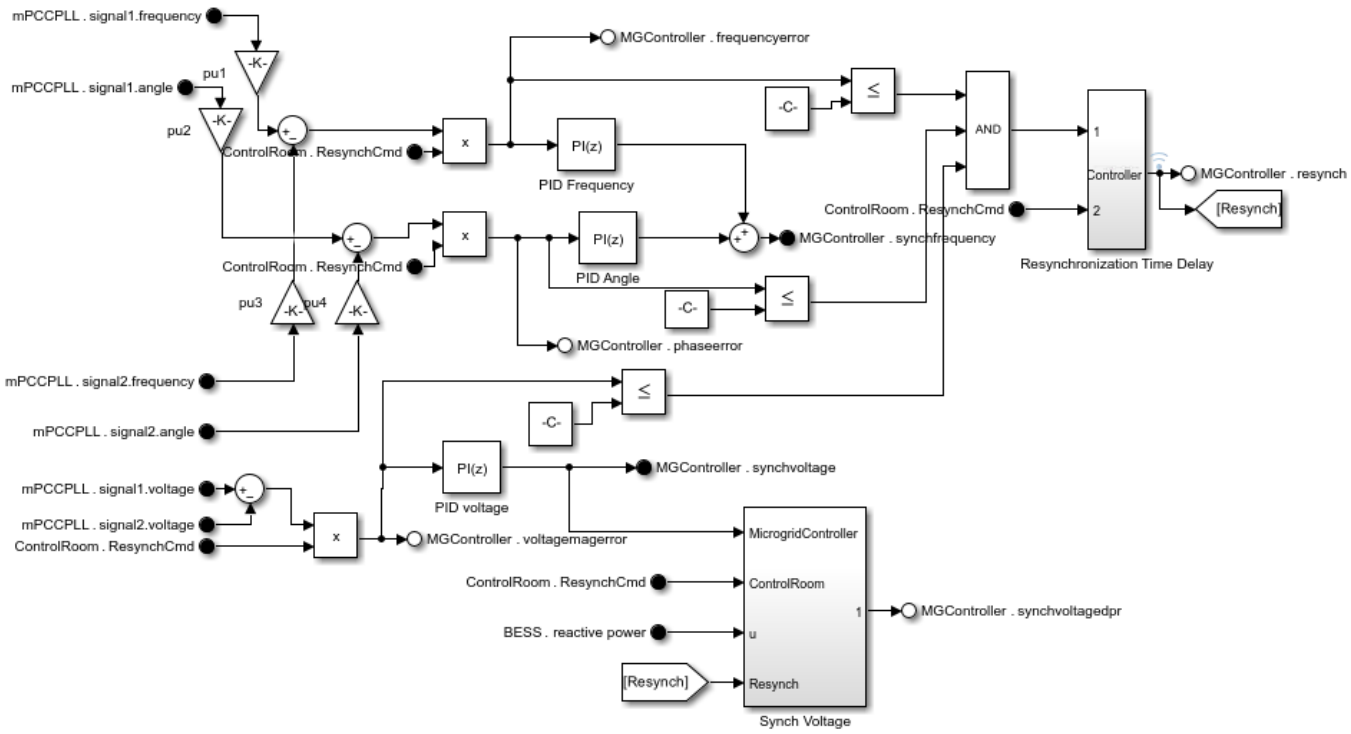
The microgrid controller receives commands and setpoints from the Operator Control Room subsystem, as well as measurements and status from other sources and loads in the microgrid. The microgrid controller has different microgrid functions. In this example, the microgrid controller comprises:

1. PLL measurements from grid side and microgrid side
2. Reference frequency generation for the microgrid
3. Reference voltage generation for the microgrid
4. Resynchronization function to provide setpoint for BESS during voltage and frequency matching
5. Activate main breaker control command
6. Receive setpoint and command from operator and communicate to the microgrid asset

For resynchronization, the resynchronization function matches the voltage and frequency of the microgrid to the main grid. To achieve this synchronization, the resynchronization function calculates the necessary voltage and frequency reference for the BESS system.

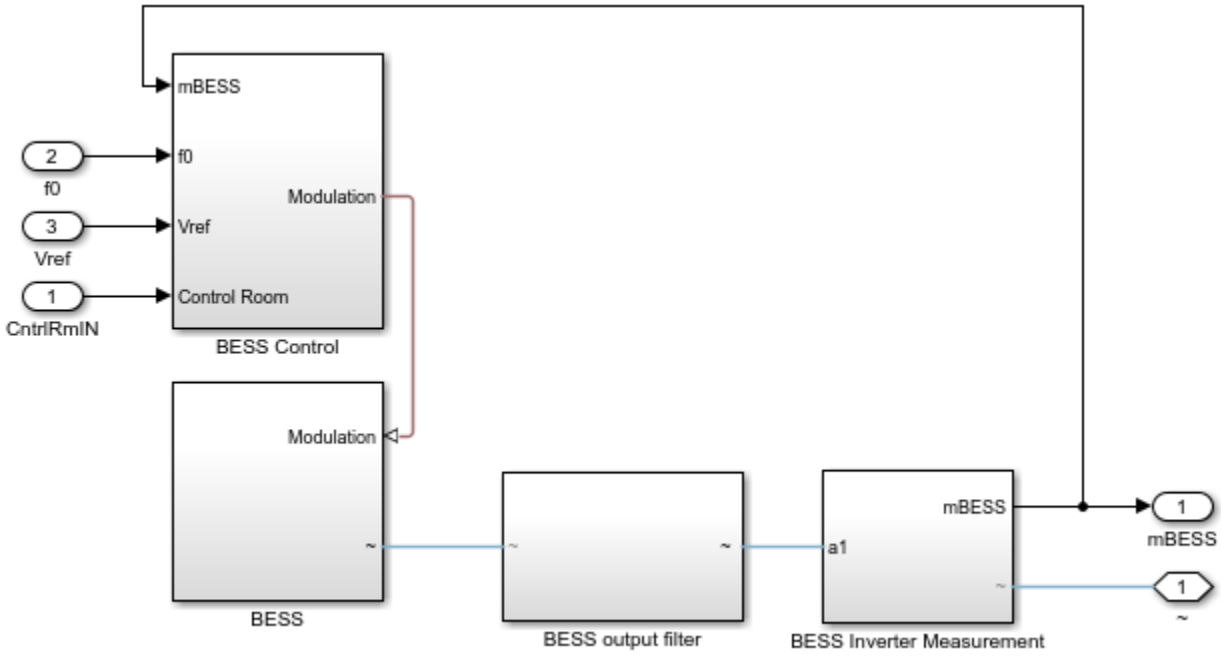



**Microgrid Resynchronization**



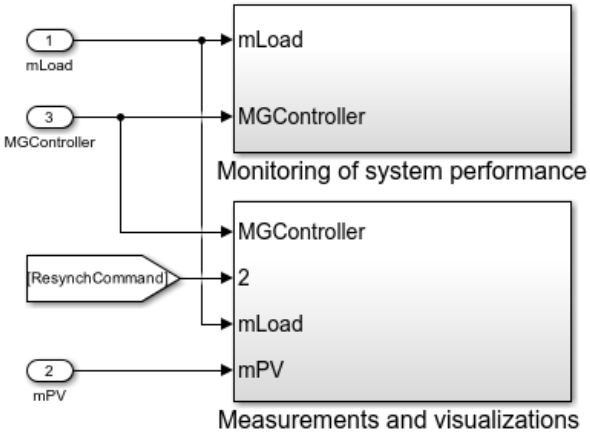

**BESS System**

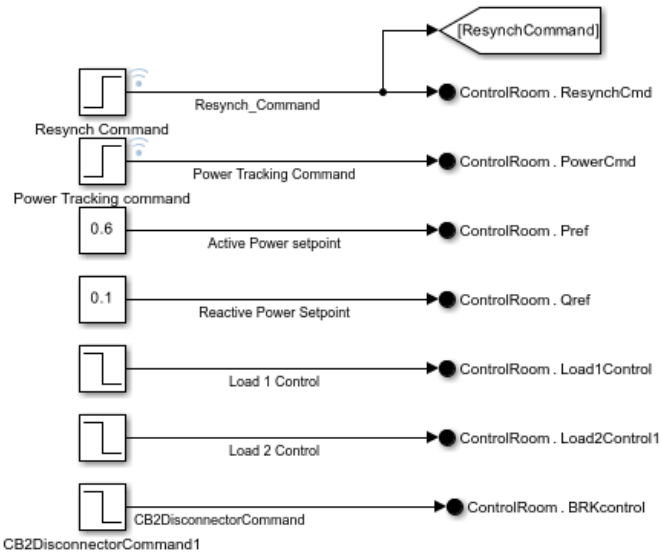
The BESS system connects at the point of common coupling and installed in the substation. The BESS is controlled to achieve the voltage and frequency setpoint received from the microgrid controller for the resynchronization. After the synchronization, the Operator Control Room orders the BESS to perform a power dispatch with setpoints.




### Operator Control Room

The operator control room sends all the setpoints and commands. It also plots the measured quantities and the system performance analysis.



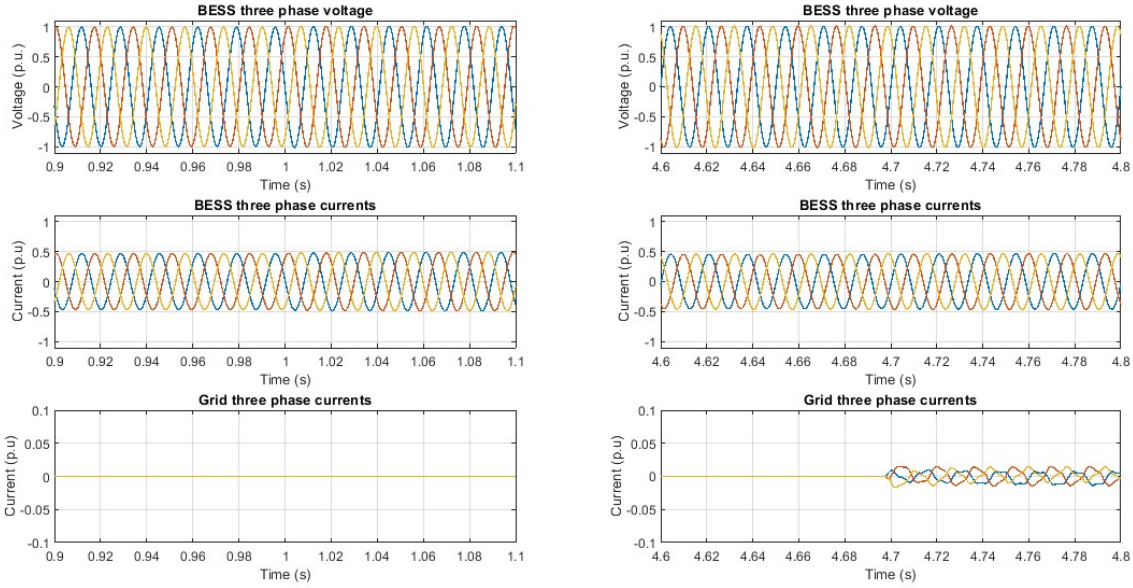



### Simulation Results

These plots show:

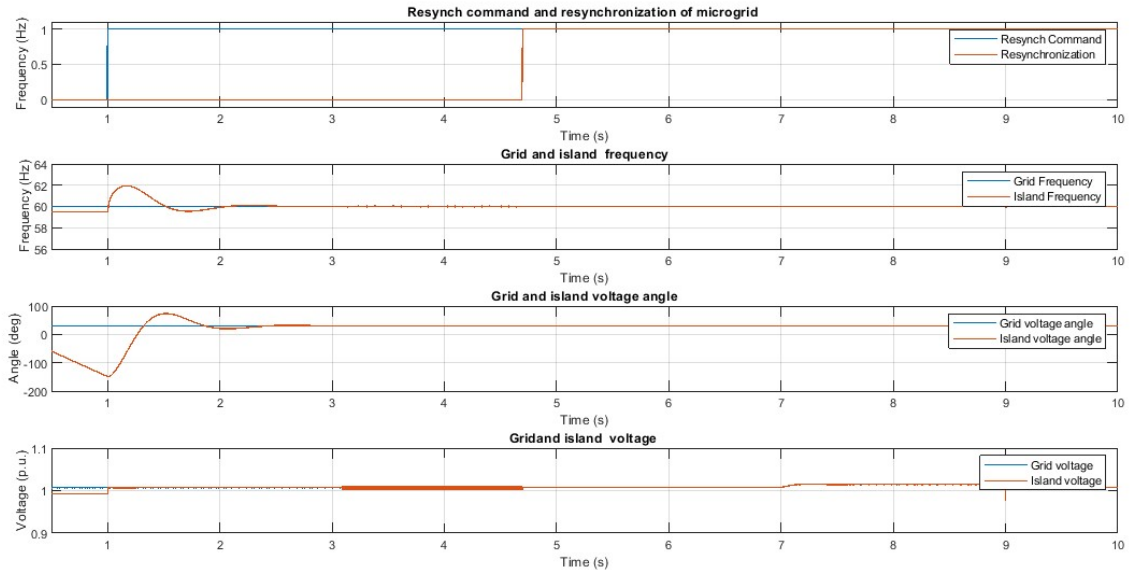
1. Voltage and current of BESS.
2. Synchronization of voltage, frequency, and phase angle.
3. Active and reactive power output of BESS, PV, load, and main grid.

This plot shows the three phase voltage and current output of the BESS, as well as the grid current during resynchronization.

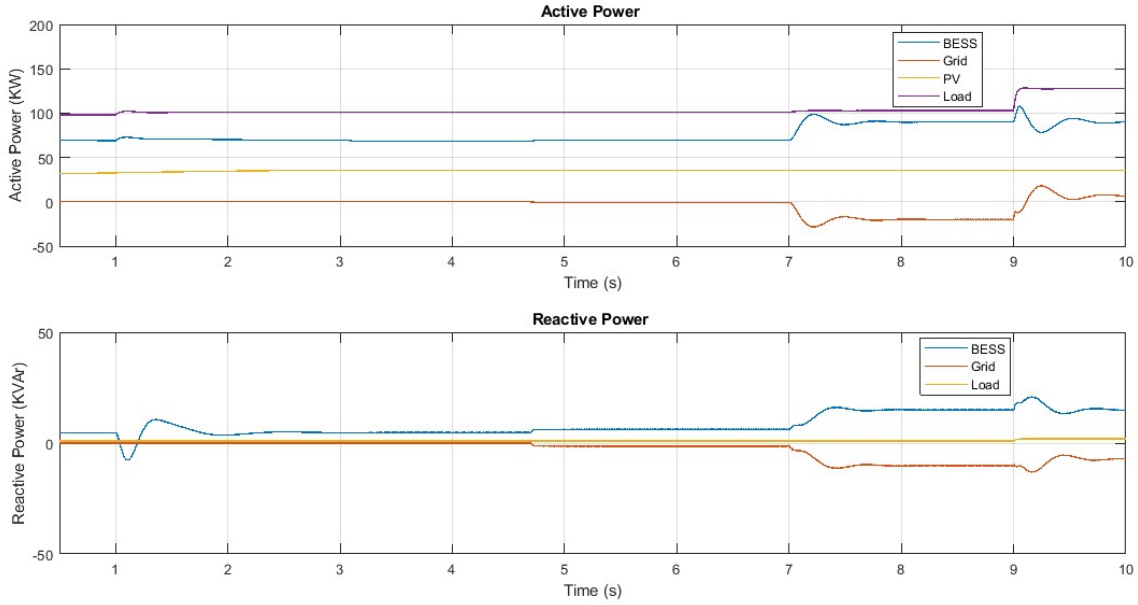



The plot shows the measured values around the resynchronization command at 1 s and the breaker closing at 4.7 s. A stable voltage and current output from BESS verifies a good resynchronization.The small grid current once the breaker closes shows a good voltage and frequency matching during resynchronization.

This plot shows resynchronization of voltage and frequency of the microgrid with the voltage and frequency of the main grid.

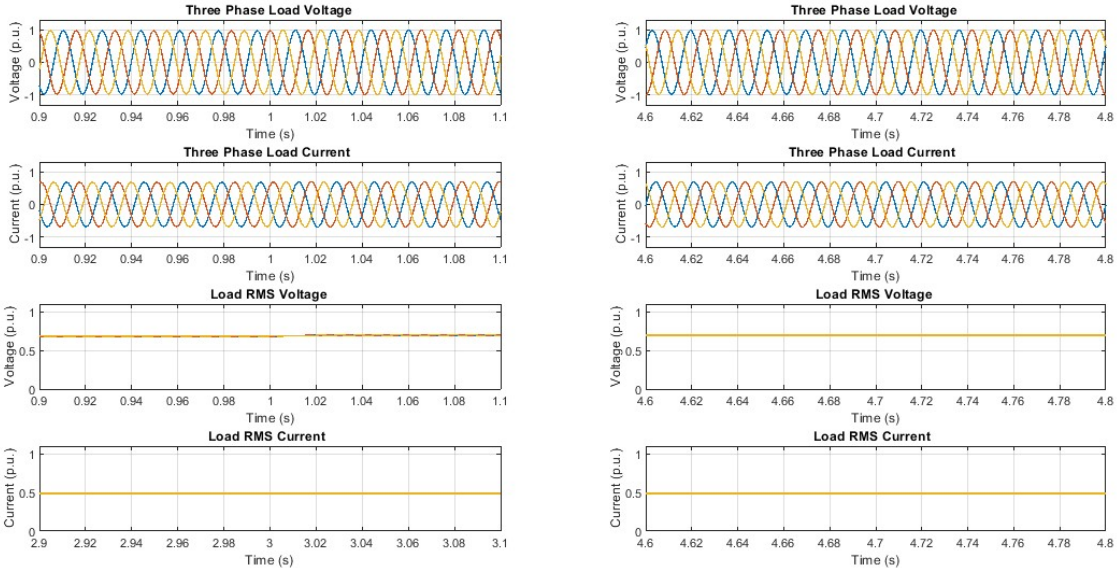



The quick matching of voltage, frequency, and phase angle after the resynchronization command validates the efficacy of the BESS control in resynchronization. This plot shows the active and reactive power of BESS, PV, main grid, and loads.




Stable power outputs confirm a stable resynchronization process. The uninterrupted load power verifies a smooth resynchronization of microgrid.

This plot shows the voltage and current of the microgrid loads.



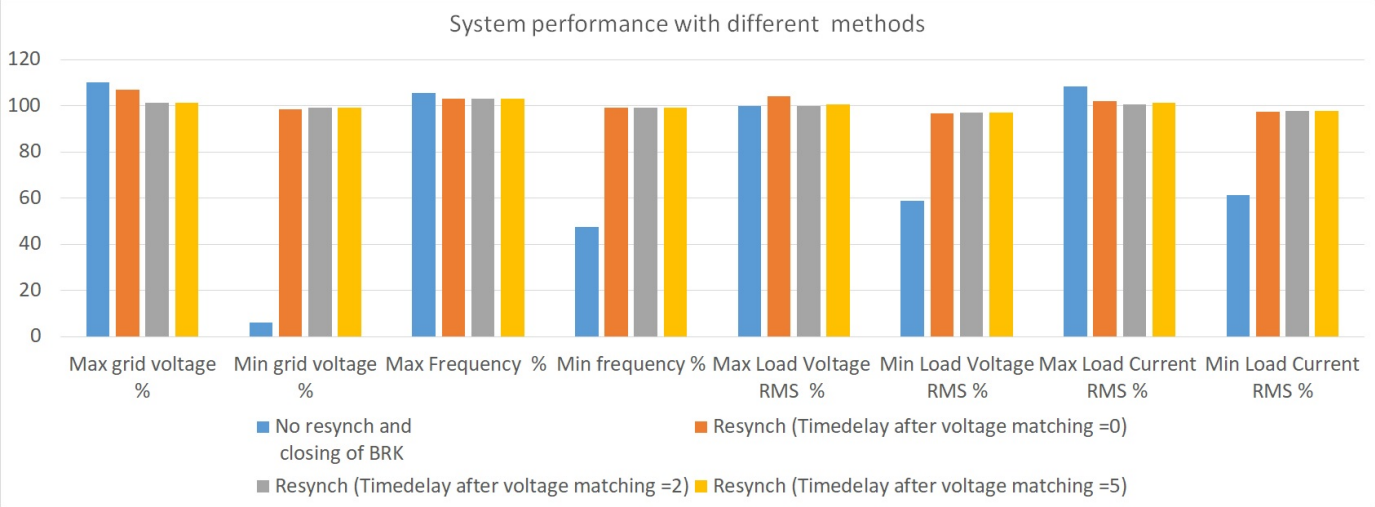

Uninterrupted stable voltage and current at microgrid loads verify effective resychronization of microgrid. The plot shows the measured values around the resynchronization command at 1 s and when the breaker closes at 4.7 s.

### System Performance

These plots show the results of the system performance and the impact of the resynchronization function. These performance indices include:

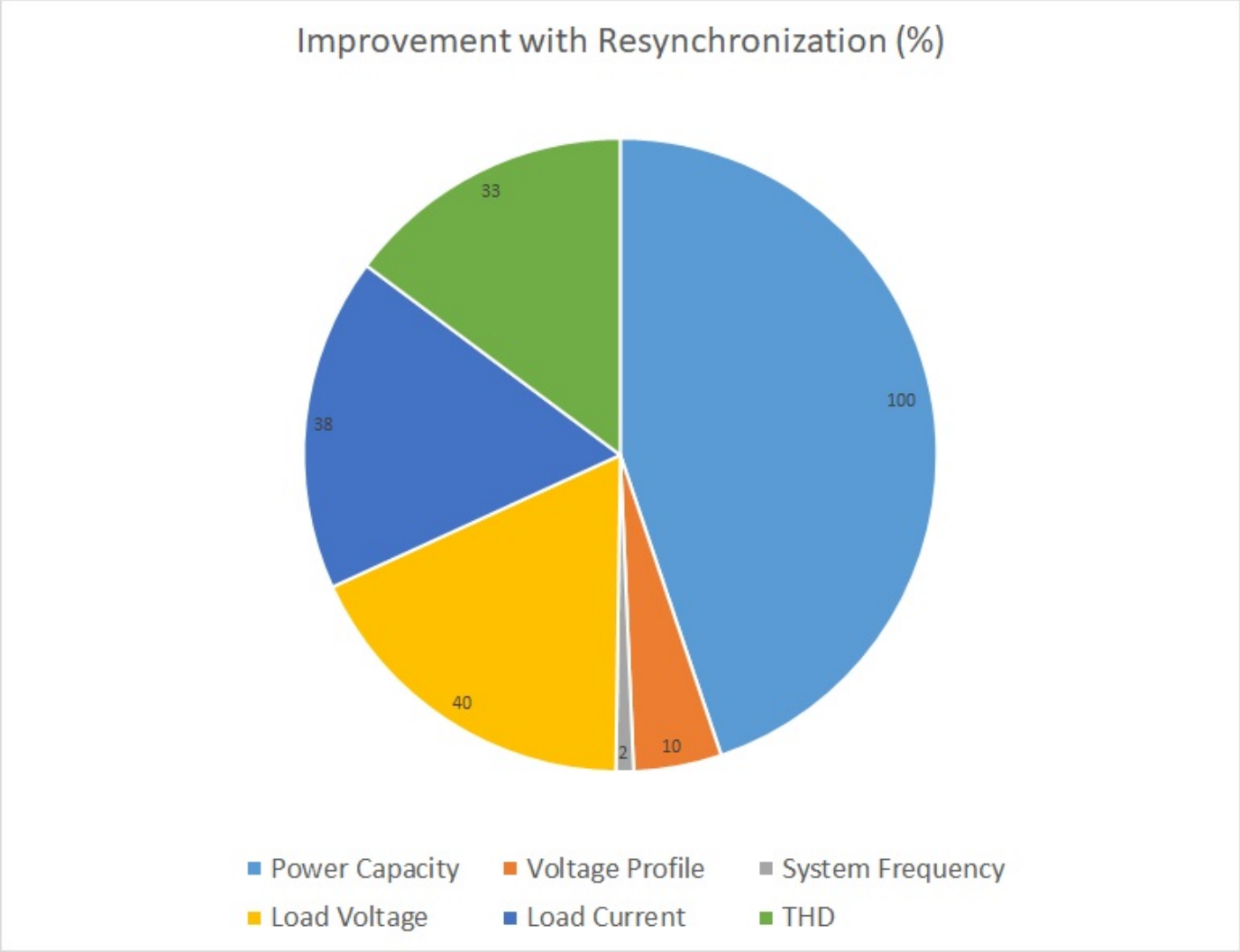
1. Maximum voltage, minimum voltage, and frequency measured at relevant grid location.
1. IEEE 2030.7 mapping on resynchronization process
1. Improvement of key performance indices with resynchronization process

This plot shows the minimum and maximum values of voltage and frequency at different locations of the microgrid.



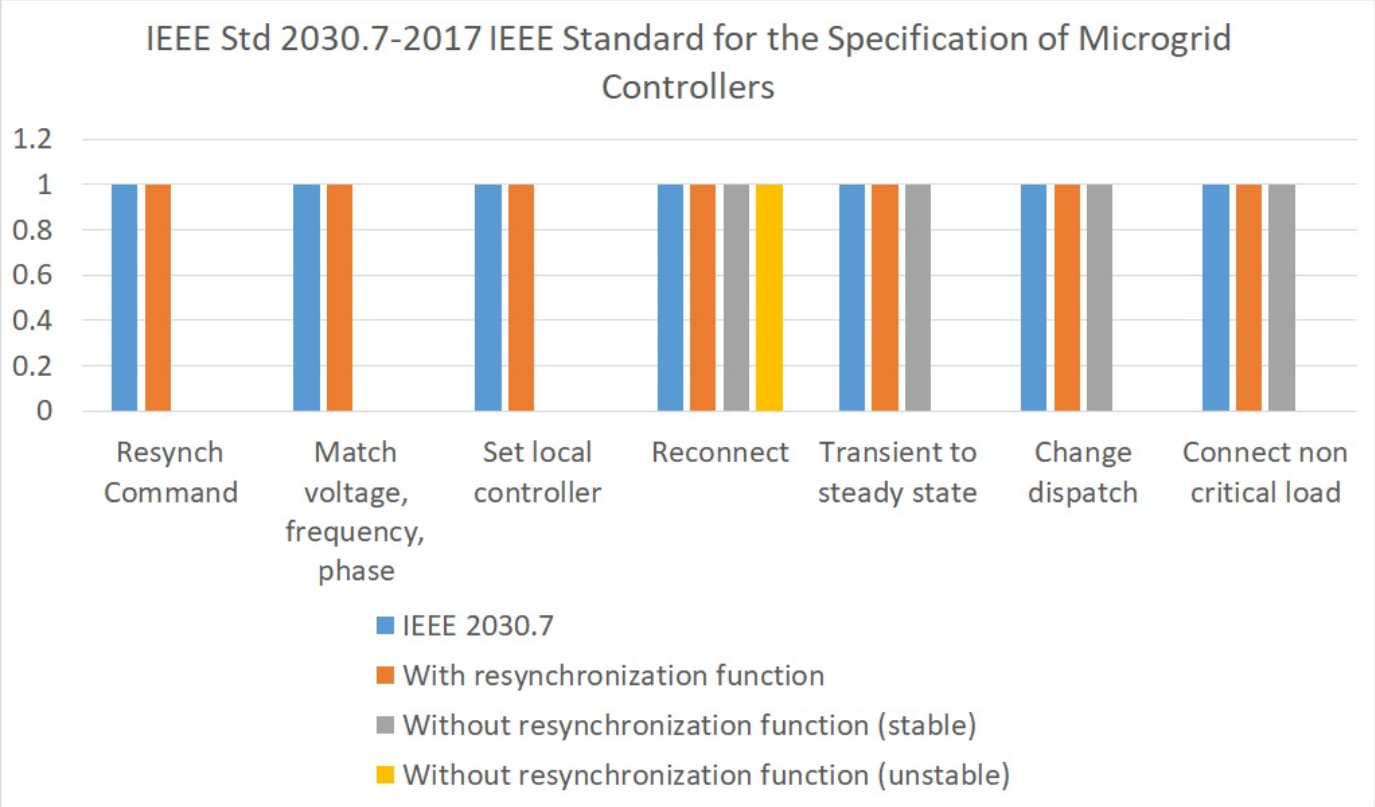

The resynchronization process maintains the deviations of the voltage and frequency within acceptable limits.

This plot shows the improvement in key performance indices with the resynchronization process compared to grid connection without resynchronization function.




The resynchronization process significantly improves the power capacity, load voltage, and load current fluctuations.

This plot shows the IEEE 2030.7 mapping on resynchronization process.

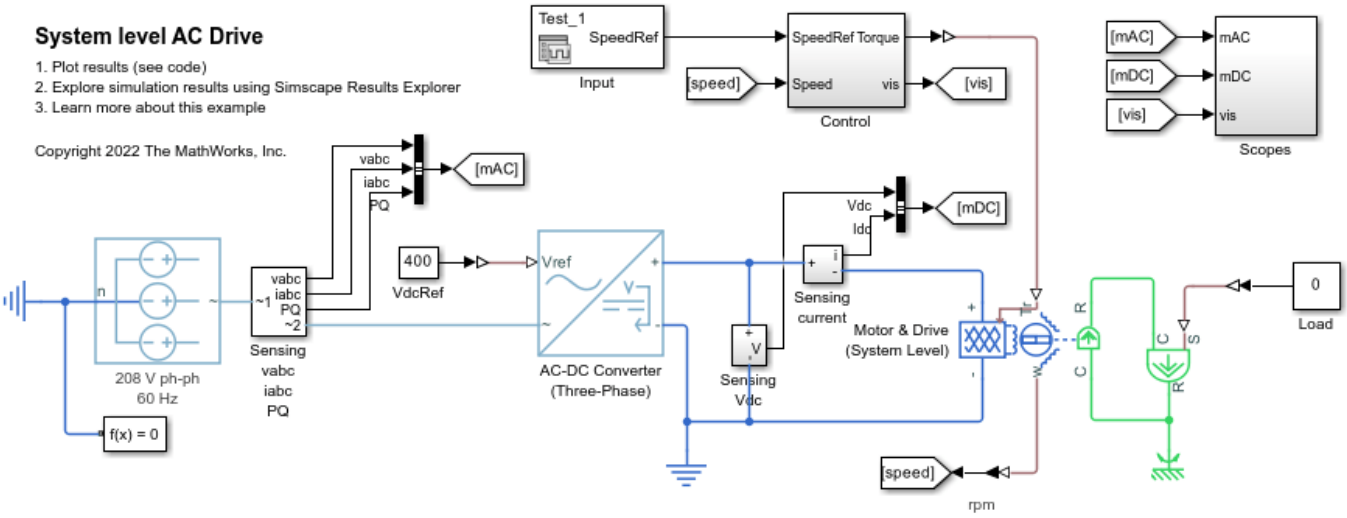



The resynchronization function executes all the steps from IEEE 2030.7. Grid connections without resynchronization functions cannot follow the IEEE standard.

# System-Level AC Drive

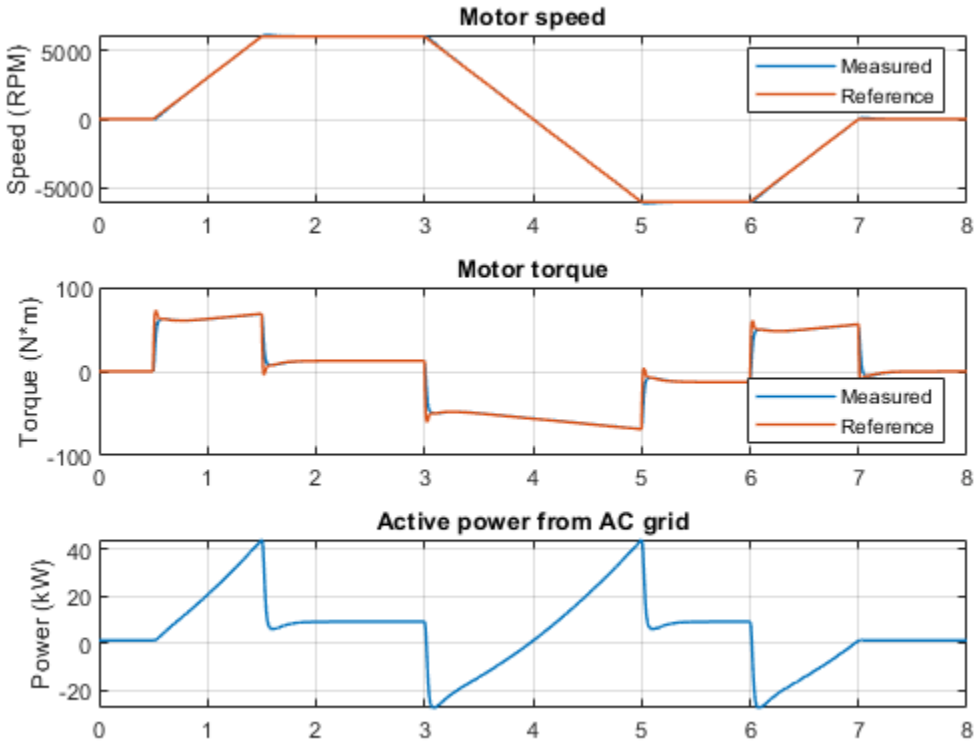
This example shows how to model a system-level AC drive for an electric machine by using the AC-DC Converter (Three-Phase) block and Motor & Drive (System Level) block. The AC-DC converter represents a grid-side converter which provides a constant DC-link voltage. The Motor & Drive block acts as a generic AC electric machine with its DC-AC converter. This modeling approach provides fast system-level simulation without the switching events of the power converters.

### Model Overview




### Simulation Results from Simscape Logging

The plot below shows the requested and measured motor speed, the requested and measured motor torque, and the active power that flows from the AC grid.


Motor speed
Speed (RPM)
5000
0
-5000
Measured
Reference
0
1
2
3
4
5
6
7
8
Motor torque
Torque (N*m)
100
0
-100
Measured
Reference
Active power from AC grid
Power (kW)
40
20
0
-20

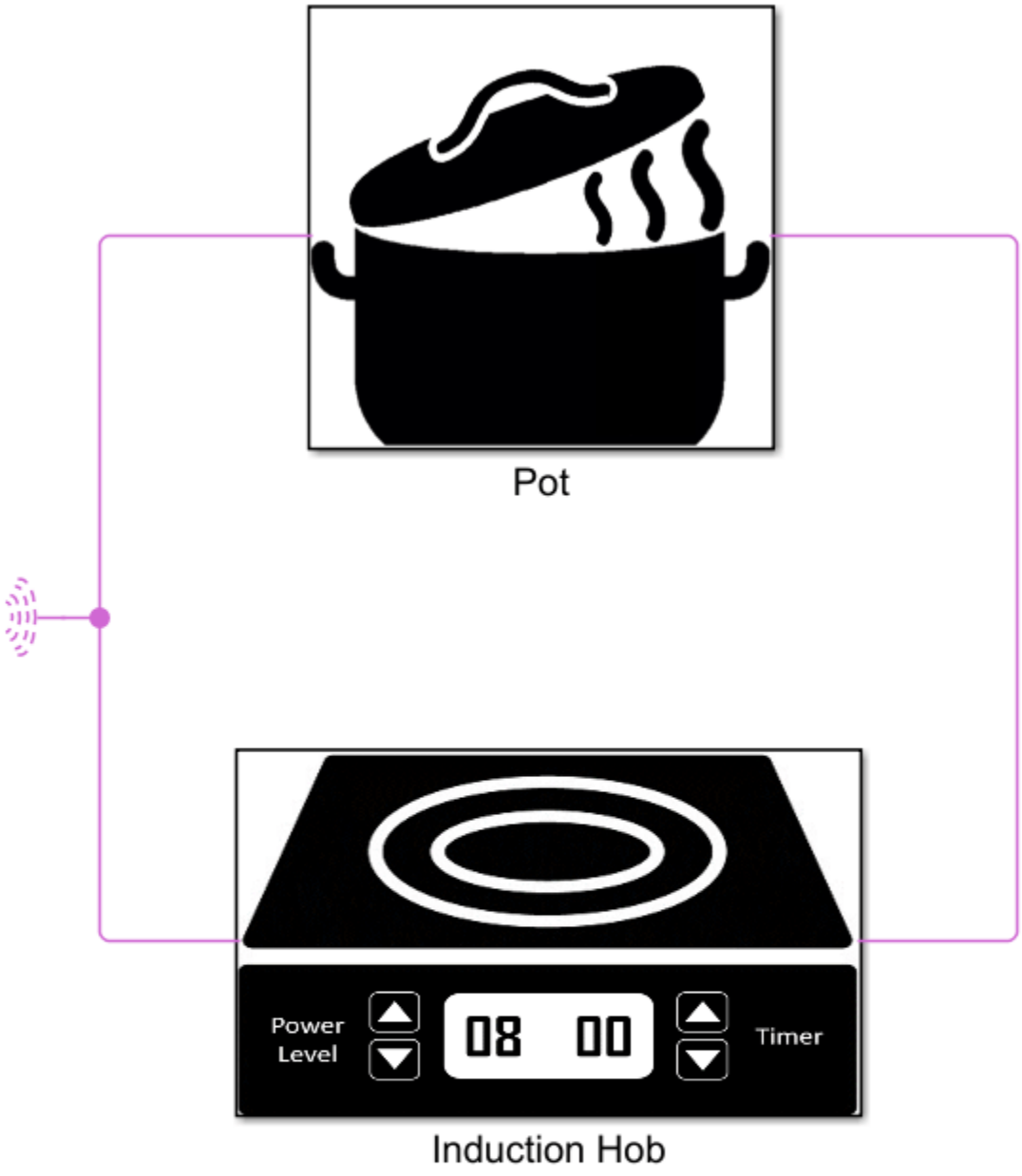
# Simple Induction Hob Simulation

This example shows how to model a simple induction hob system using Simscape Electrical™ libraries. This model focuses on the electromagnetic effect of the winding coils and the eddy current effect in the cooking pot.

### Model Overview

Open the model `ee_induction_hob.slx`.

```
mdl = "ee_induction_hob";
open_system(mdl)
```




# Induction Hob

1. Explore simulation results using sscexplore
2. Learn more about this example

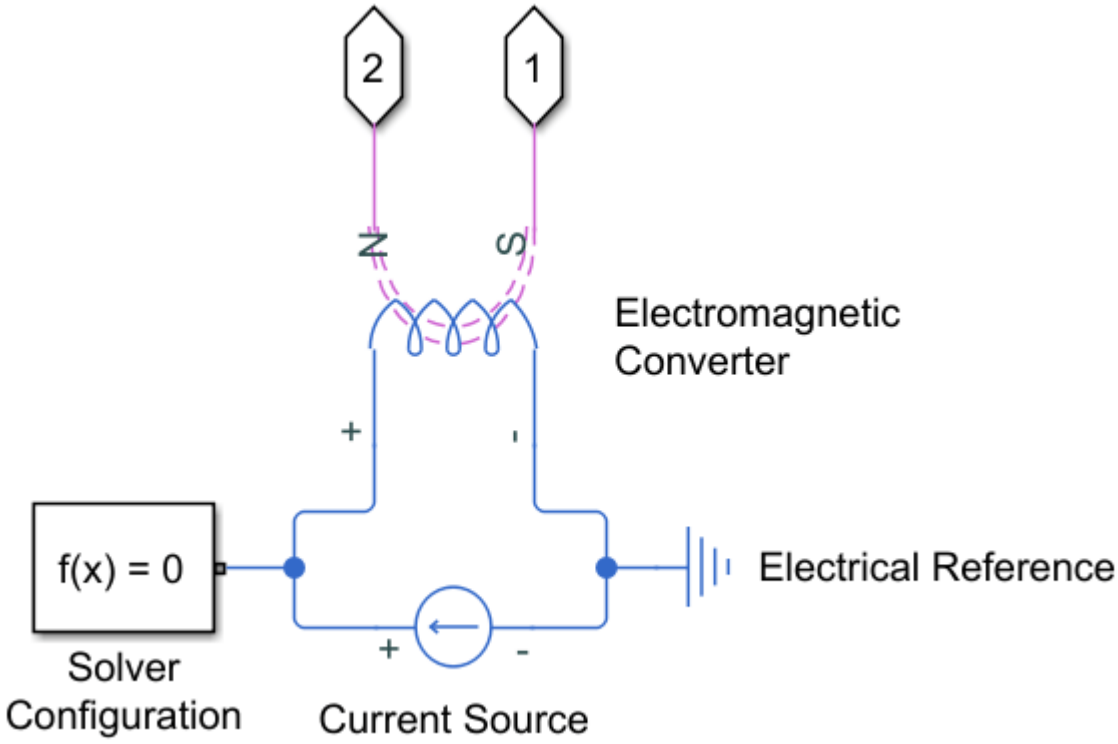
Copyright 2022 The MathWorks, Inc.

This example uses blocks from the Simscape™ Foundation™ and Simscape Electrical™ libraries. There are two subsystems for the pot and the induction hob.

### Open Induction Hob Subsystem

The Induction Hob subsystem represents the electromagnetic portion of the network. To open the subsystem, at the MATLAB command prompt, enter

```
open_system('ee_induction_hob/Induction Hob');
```

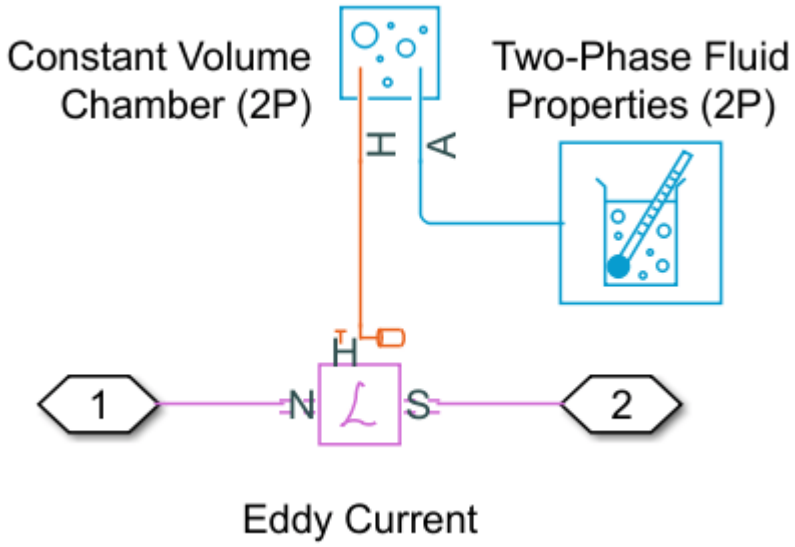



This subsystem contains a Current Source block that models the input current and an Electromagnetic Converter that represents the charging coil inside the inductor and models the electromagnetic induction that converts electrical energy to magnetic energy. The magnetic network is in purple and the electrical network is in dark blue.

### Open Pot Subsystem

The Pot subsystem models the cooking pot that contains the fluid you want to heat. To open the subsystem, at the MATLAB command prompt, enter

```
open_system('ee_induction_hob/Pot');
```




The subsystem contains an Eddy Current block that models the electrical-to-thermal energy conversion, a Constant Volume Chamber (2P) block that models the temperature change in the cooking fluid, and a Two-Phase Fluid Properties block that stores the information of the cooking fluid. The thermal network is in orange and the thermal liquid network is in light blue.

**Calculate the Conductance of the Pot**

To parameterize the model for different configurations of induction hobs and cooking pots, you must first calculate the conductance of the cooking pot. High frequencies and fast changing fields characterize the system. This means the magnetic field cannot penetrate completely in the inductive material at the bottom of the cooking pot. The penetration depth, $\delta$, is obtained from this equation:

$$\delta = \frac{1}{\sqrt{\pi f \mu \sigma}},$$

where

- $f$ is the operating frequency of the input currrent, in `Hz`
- $\mu$ is the magnetic permeability of the material of the cooking pot, in `H/m`
- $\sigma$ is the electrical conductivity of the material of the cooking pot, in `1/(Ωm)`

The equivalent resistance, `R`, and the conductance, `G`, are independent of the surface area of the cooking pot [1] on page 8-81. For a circular cooking pot, the conductance is equal to

$$G = \frac{1}{R} = \frac{\sigma \delta}{2\pi}.$$

In this example, you model an inductor with 15 winding coils that heats a cooking pot made of carbon steel, with a magnetic permeability of $1.26 \times 10^{-4}\,\mathrm{Hm}^{-1}$ and an electrical conductivity of $6.99 \times 10^{6}\,\Omega^{-1}\mathrm{m}^{-1}$. The Current Source block feeds the winding coils of the inductor with a 10 `A` and 24 `kHz` AC current. This example does not model the voltage input from the wall plug or the electrical converters that transform the electrical energy before reaching the inductor.

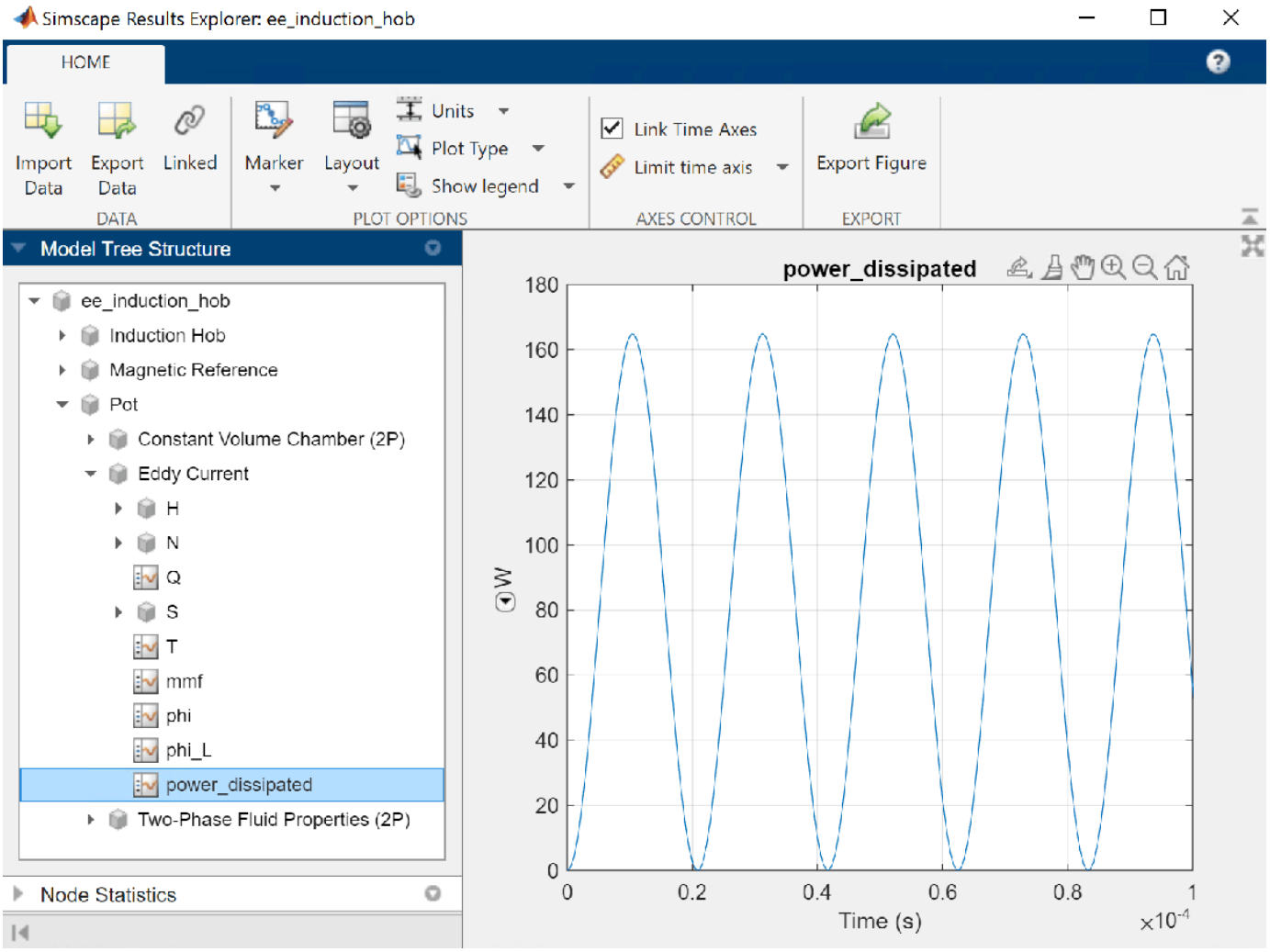
With these values, the penetration depth is equal to$1.227 \times 10^{-4}$ m and the conductance is equal to $136.55\,\Omega^{-1}$. Now set the model parameters with these values.

```
set_param( mdl + "/Induction Hob/Current Source", "ac_current", '10');
set_param( mdl + "/Induction Hob/Current Source", "ac_frequency", '24000');
set_param( mdl + "/Induction Hob/Electromagnetic Converter", "Nw", '15');
set_param( mdl + "/Pot/Eddy Current", "g", '136.55');
```

### Run Simulation

Simulate the model and explore the simulation results in the Simscape Results Explorer window.

```
result1 = sim(mdl);
sscexplore(result1.simlog)
```




When you click a node in the left pane, the corresponding plots appear in the right pane. For example, check the power output from the Eddy Current block by clicking the **power_dissipated** variable in **Pot > Eddy Current**.

### Change Parameters for Different Hobs and Cookers

To model different configurations of inductors and cooking pots, modify the parameters of the blocks in the model. For example, to model a different induction hob with 25 winding coils, in the Electromagnetic Converter block, set the **Number of winding turns** parameter to 25. Then run the simulation again and save the results.

```
set_param( mdl + "/Induction Hob/Electromagnetic Converter", "Nw", '25');
result2 = sim(mdl);
sscexplore(result2.simlog)
```

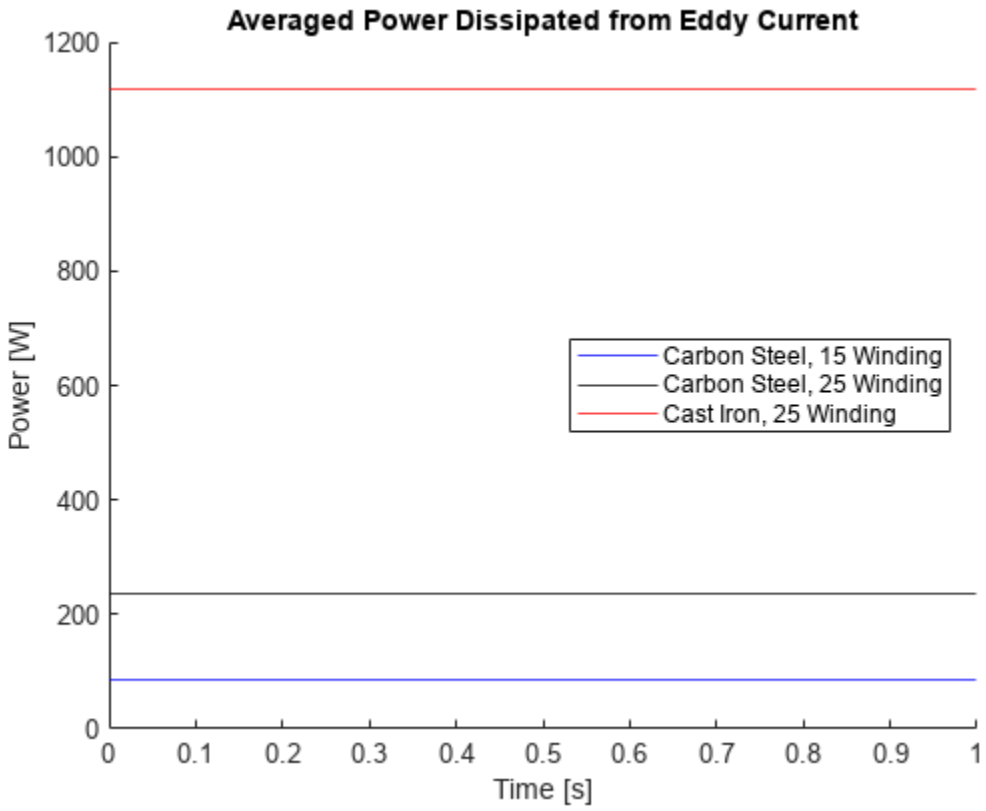
Now model a cooking pot made of grey cast iron, with a magnetic permeability of $3.96 \times 10^{-4}\,\mathrm{Hm}^{-1}$ and an electrical conductivity of $10^{6}\,\Omega^{-1}\mathrm{m}^{-1}$. With these values, the conductance is equal to $29.13\,\Omega^{-1}$. In the Eddy Current block, set the **Conductance of eddy current loop** parameter to 29.13. Again, run the simulation and save the new results.

```
set_param( mdl + "/Pot/Eddy Current", "g", '29.13');
result3 = sim(mdl);
sscexplore(result3.simlog)
```

### Plot and Compare the Results

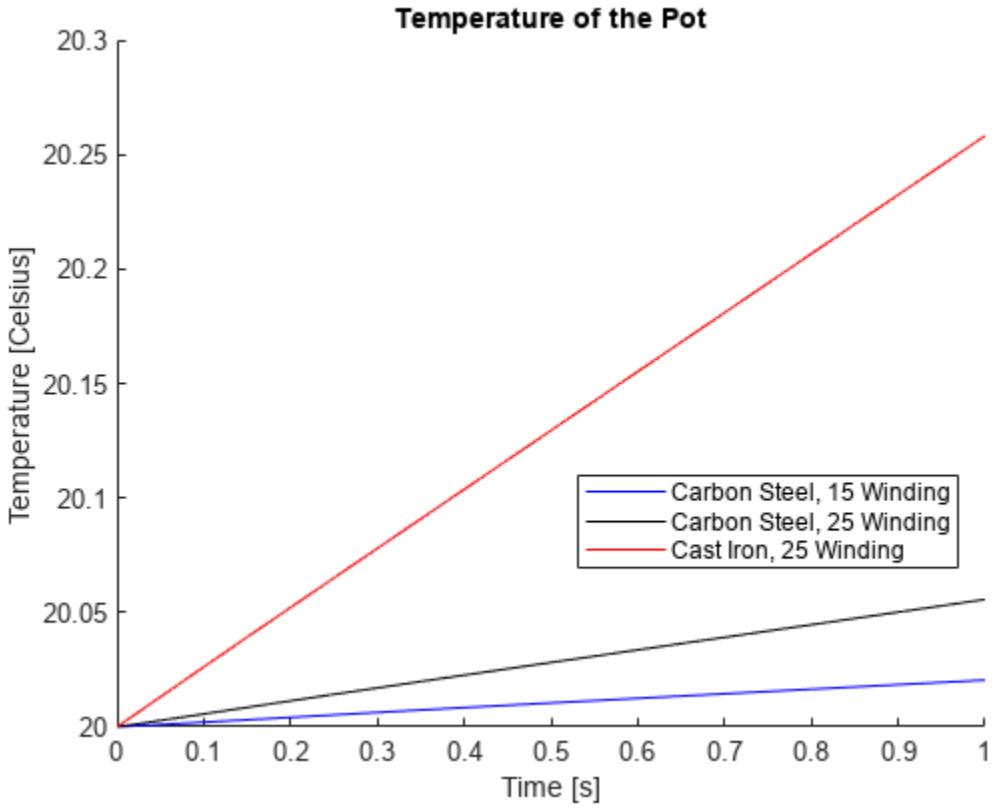
To extract, plot, and compare the results of the previous simulations, at the MATLAB Command Window, enter:

```
figure
hold all
yline(mean(result1.simlog.Pot.Eddy_Current.power_dissipated.series.values), "b")
yline(mean(result2.simlog.Pot.Eddy_Current.power_dissipated.series.values), "k")
yline(mean(result3.simlog.Pot.Eddy_Current.power_dissipated.series.values), "r")
simTime = result1.simlog.Pot.Constant_Volume_Chamber_2P.T_I.series.time;
xlim([0, simTime(end)])
xlabel("Time [s]")
ylabel("Power [W]")
title("Averaged Power Dissipated from Eddy Current")
legend("Carbon Steel, 15 Winding", "Carbon Steel, 25 Winding", "Cast Iron, 25 Winding", "Locatio
```




The first plot shows the averaged power dissipated from the Eddy Current block for each of the three cooking pots. The carbon steel cooking pot with 25 winding coils transforms more power into thermal than the carbon steel cooking pot with 15 winding coils. With the same number of winding coils, the cast iron pot has more cooking power due to its much lower conductance.

```
figure
hold all
plot(result1.simlog.Pot.Constant_Volume_Chamber_2P.T_I.series.time, result1.simlog.Pot.Constant_V
plot(result2.simlog.Pot.Constant_Volume_Chamber_2P.T_I.series.time, result2.simlog.Pot.Constant_V
plot(result3.simlog.Pot.Constant_Volume_Chamber_2P.T_I.series.time, result3.simlog.Pot.Constant_V
xlabel("Time [s]")
ylabel("Temperature [Celsius]")
title("Temperature of the Pot")
legend("Carbon Steel, 15 Winding", "Carbon Steel, 25 Winding", "Cast Iron, 25 Winding", "Locatio
```




The second plot compares the temperature rise for the three cooking pots. Similarly to the results of the first plot, a higher number of winding coils and a low-conductance material provide more power and quicker temperature rise.

**Reference**

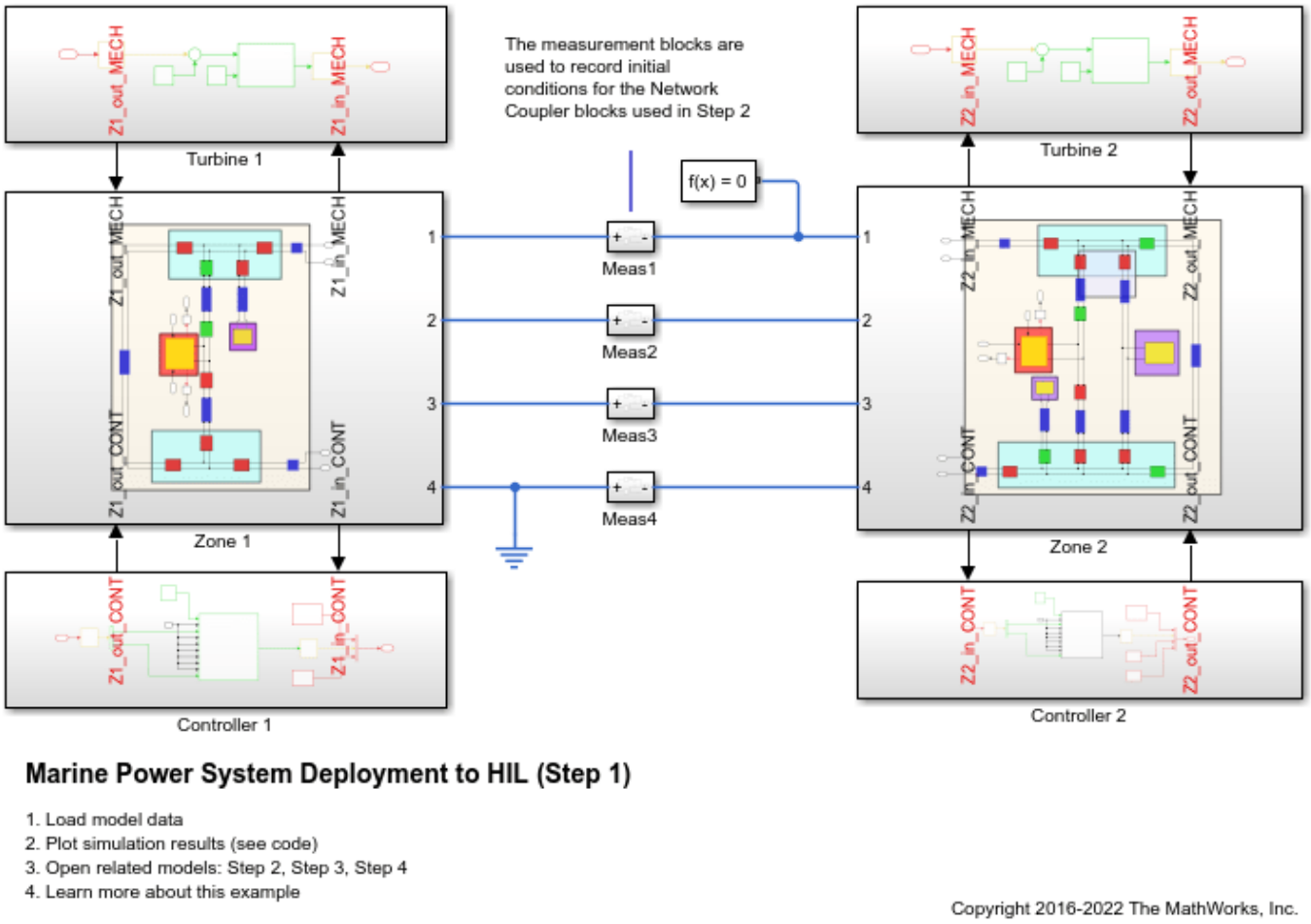
[1] Siakavellas, N. J. "Two simple models for analytical calculation of eddy currents in thin conducting plates," in *IEEE Transactions on Magnetics*, vol. 33, no. 3, pp. 2245-2257.

# Marine Power System Deployment to HIL

This example shows a marine power system model suitable for multirate Hardware-In-the-Loop (HIL) deployment. The example uses the Simscape™ Network Couplers Library to split the model into separate Simulink® subsystems that you can deploy at different sample rates. This allows you to run parts of the system (here, for example, the turbines) with a slower sample time and reduce overall computational cost.

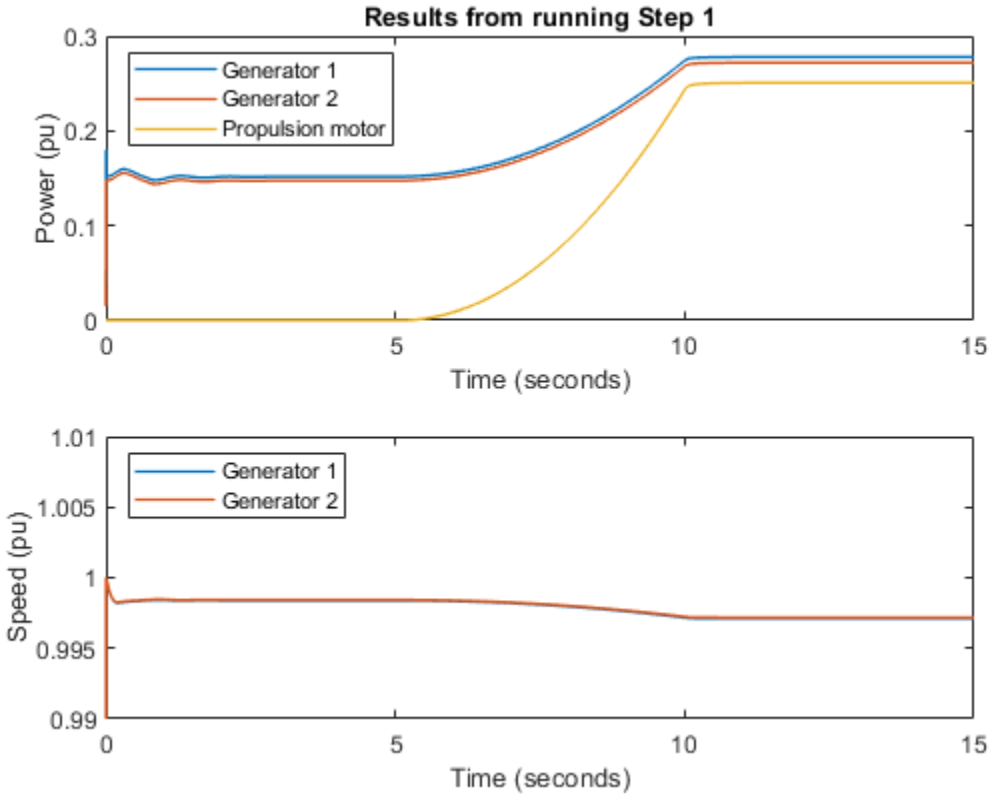
### Step 1: Design Model

This figure shows the top-level design model. The power system comprises two separate electrical zones, each with its own steam generator, controllers, and local loads. The model already has different sample times for the electrical, mechanical, and controller parts. However, it is configured to run as a single simulation thread, either for desktop simulation, or for deployment on a real-time target.




Now run the simulation to generate some validation data using the unmodified model.

```
Simulating ee_marine_power_system
```

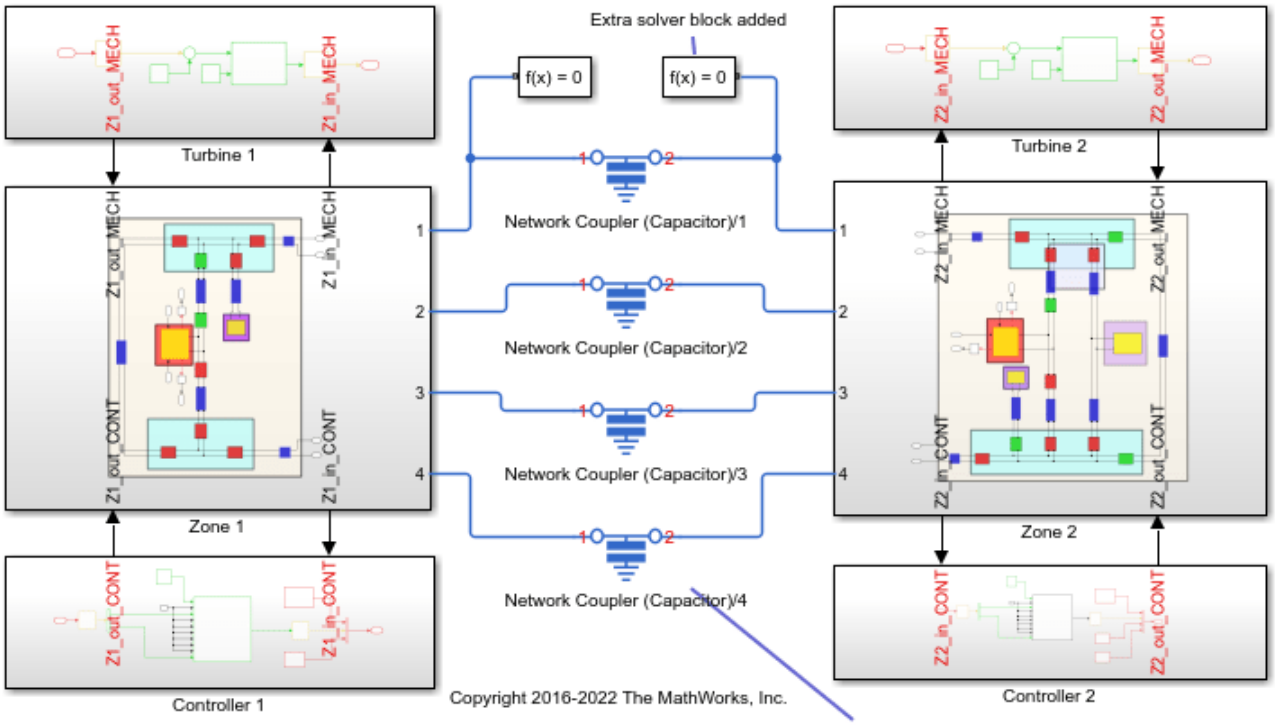



Depending on real-time hardware capability, a large model like this might not run in the available frame time. In this case, you can separate the model into smaller parts and distribute them across multiple processors. Here the model is split into four separate tasks that you can run on different cores.

**Step 2: Split the Electrical Network**

This figure shows how the Network Coupler (Capacitor) block separates the two zones by using the capacitor integrator to separate an algebraic voltage constraint. An alternative solution for this model is to use the Network Coupler (Voltage-Voltage) block, which uses voltage measurements from the last sample time to keep the two networks synchronized. This maintains numerical stability through careful choice of an internal impedance for each voltage source.

You must carefully set the initial conditions for the coupler blocks when splitting a physical network. This is because the decoupled networks are initialized separately by Simscape. The initial voltages and currents from Step 1 are used to set initial conditions on each of the Network Coupler blocks.



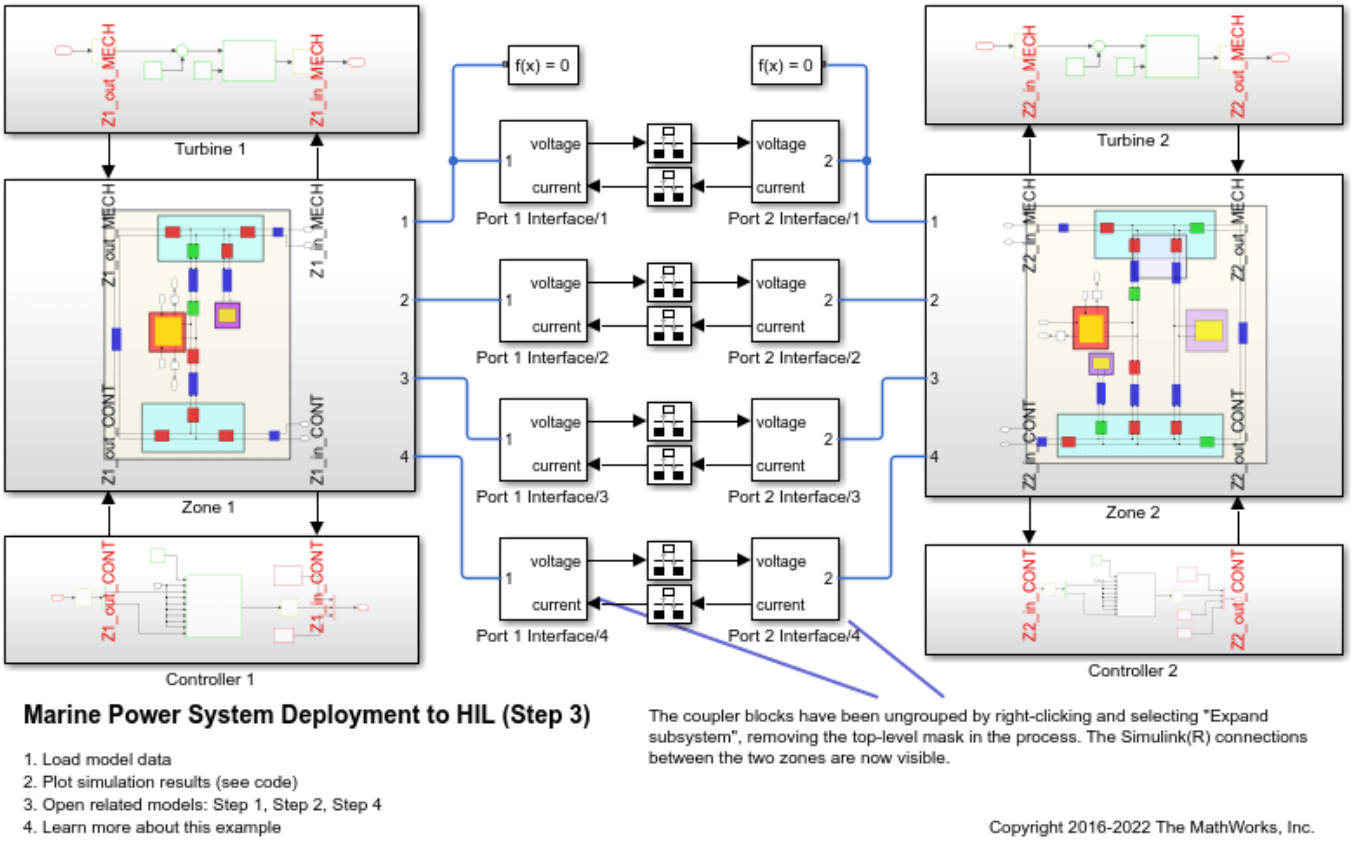

**Marine Power System Deployment to HIL (Step 2)**

1. Load model data
2. Plot simulation results (see code)
3. Open related models: Step 1, Step 3, Step 4
4. Open an alternative solution using voltage sources
5. Learn more about this example

Because Zone 1 and Zone 2 cables are represented with series inductances, the Network Coupler (Capacitor) block is chosen to avoid current sources driving inductances. A suitable value for the capacitance is determined using the Analysis tabs on the Port 1 Interface subsystem masks. The capacitance value was iterated on until the recommended max fixed step sample time was approximately equal to the electrical network sampling time, TsELEC. The approximate connected network impedance is set to the cable resistance, but could be relaxed to an impedance based on typical loads.

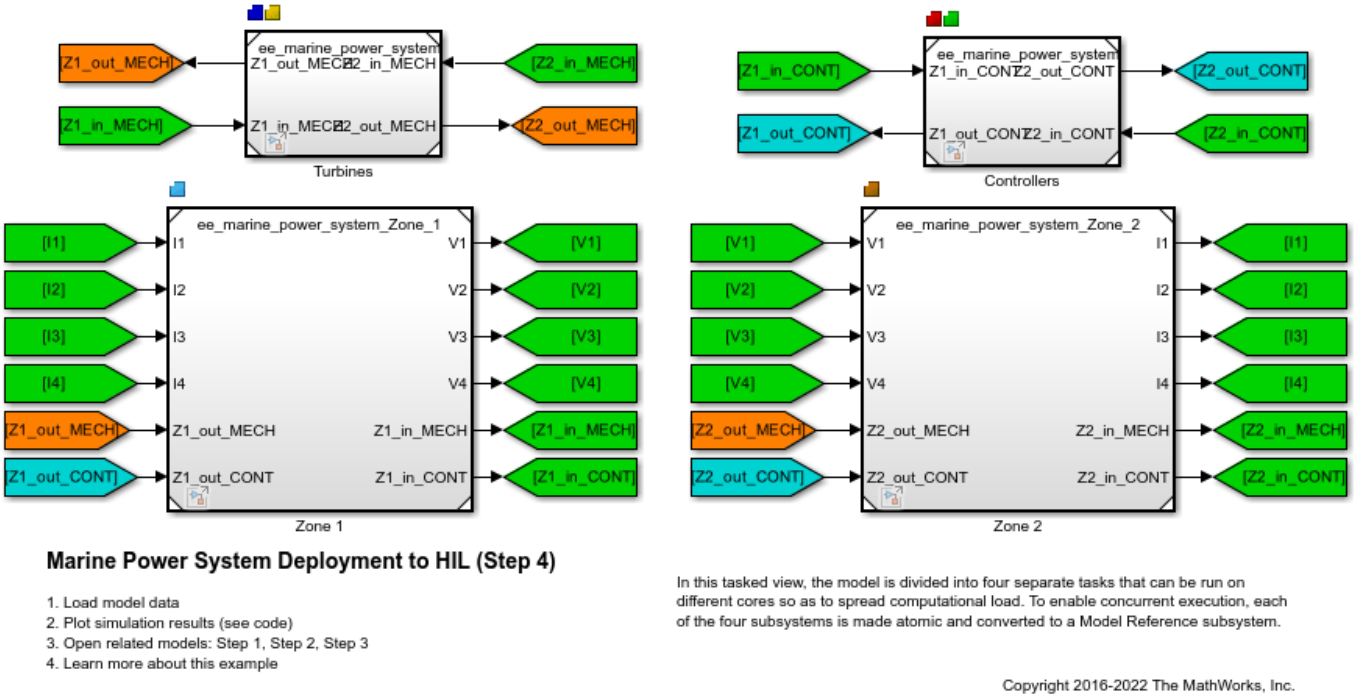
### Step 3: Ungroup and Redraw

In this step all of the Simscape Network Coupler Library blocks are first unmasked and then their contents moved up a level using the Subsystem && Model Reference -> Expand Subsystem option from the right-click context menu. This makes the Simulink connections visible at the top level, along with the rate transition blocks. You can then group the blocks and subsystems together by sampling time.



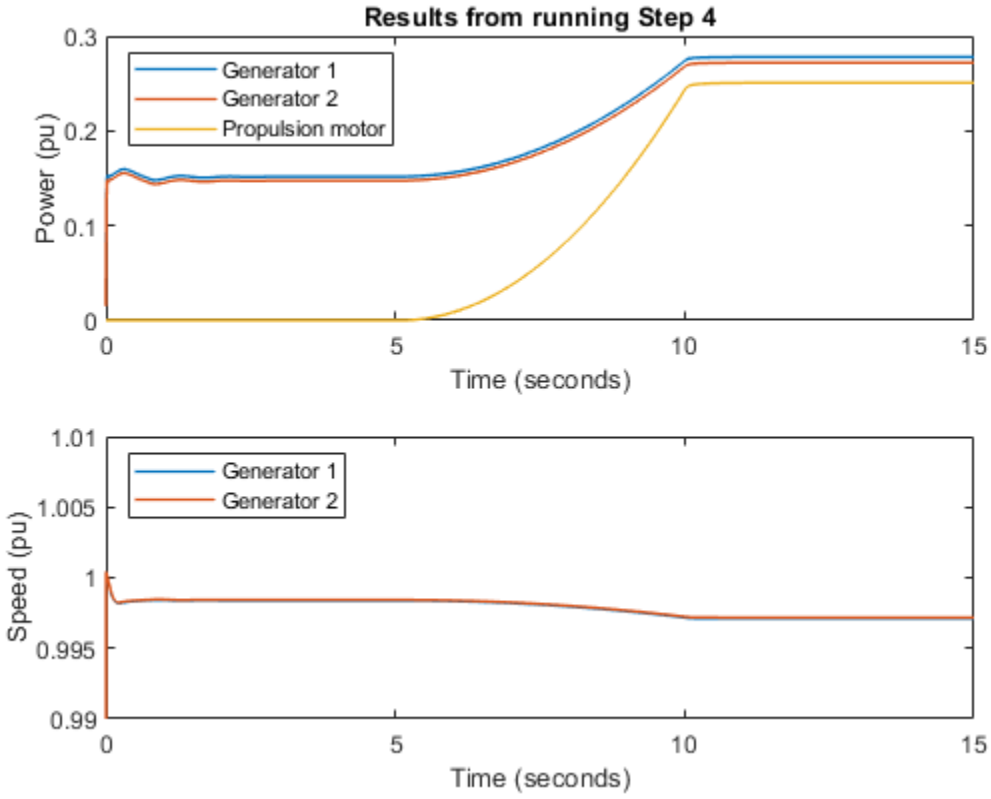

### Step 4: Deployment Model

In this final step the subsystems are converted to model reference blocks and six periodic tasks are defined. You can now simulate the model using multiple cores or by deploying to multiple microprocessors.




This figure shows the response of the two generators when the main propulsion system is started. You can compare the results against the same results generated in Step 1 and confirm that the deployment steps do not significantly change behavior.

```
Simulating ee_marine_power_system_step4
```

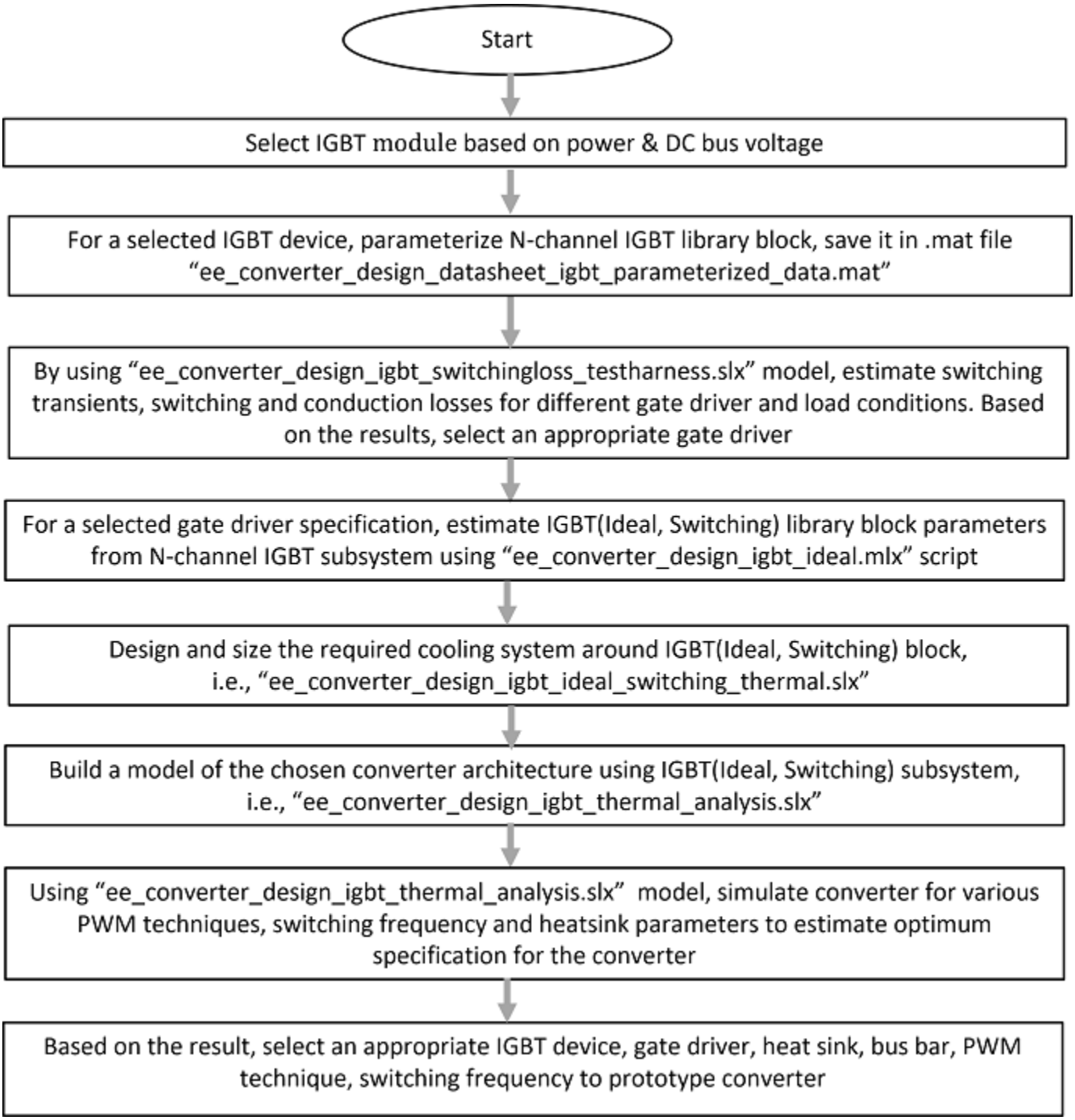

Results from running Step 4
Power (pu)
Time (seconds)
Generator 1
Generator 2
Propulsion motor
0.3
0.2
0.1
0
0
5
10
15
Speed (pu)
Time (seconds)
Generator 1
Generator 2
1.01
1.005
1
0.995
0.99
0
5
10
15

# Three-Phase High-Power Converter Design and Analysis Workflow

This example shows the main steps involved in designing a high-power converter. High power converters are important building blocks for future electric mobility and microgrid solutions. To design a cost effective, lightweight, efficient converter, you must perform detailed analysis of different converter design options and deployment scenarios. This example helps you to simulate the steady state, transient electrical, and thermal characteristics of a three-phase two-level converter that uses Insulated-Gate Bipolar Transistor (IGBT) devices.

### Converter Design and Analysis Workflow

This flowchart shows how to perform a converter analysis and how to study the effects of design decisions and component parameters. This analysis allows you to find an efficient and cost-effective design.



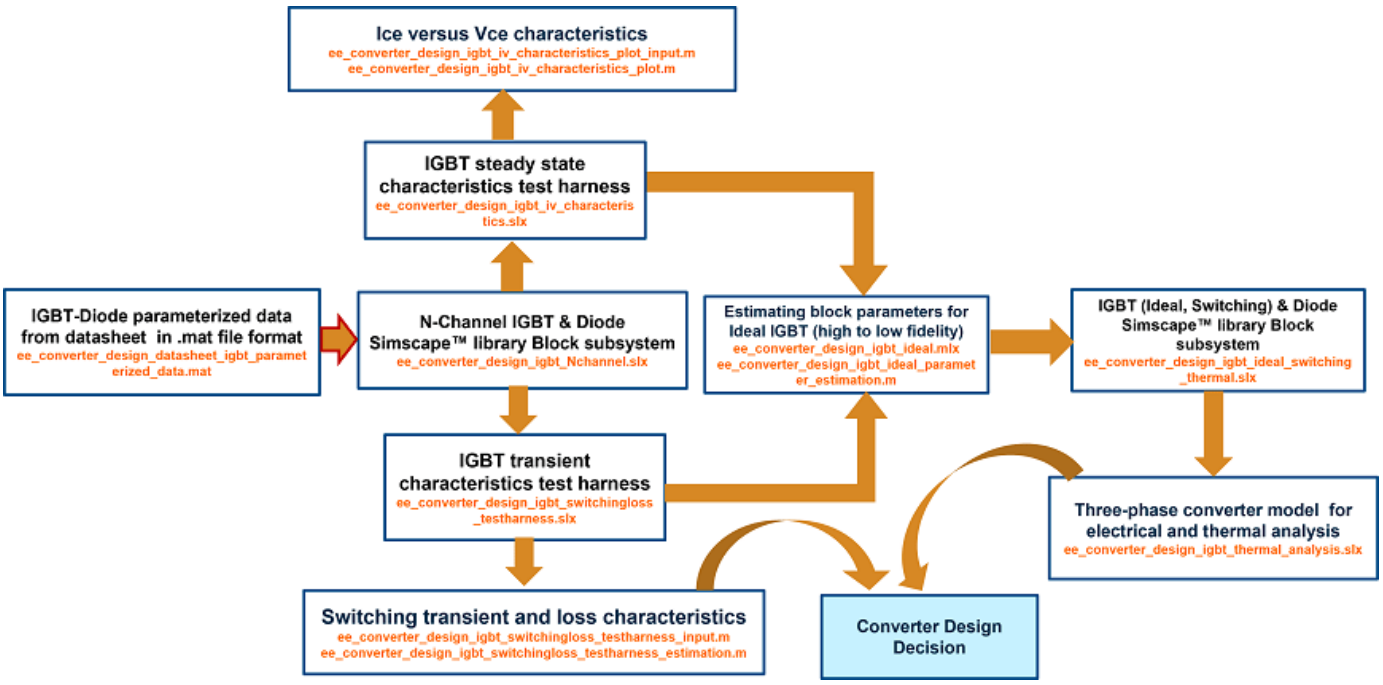

The N-Channel IGBT and Diode blocks of this example are parameterized to correctly captures the on-state losses and the switching losses. The **Modeling option** parameter of the N-Channel IGBT block is set to `Full I-V and capacitance characteristics`. The **Diode model** parameter of the Diode block is set to `Tabulated I-V curve`. The characteristics curves in the manufacturer datasheet provide the parameters for these blocks.

If you want to use this example for other IGBT and Diode devices, to carry out the trade-off studies and design optimization, you must provide proper parameterized data that is efficient to simulate.

The `ee_converter_design_datasheet_igbt_parameterized_data` MAT file contains the parameterized data for the N-Channel IGBT, Diode, Heatsink, and IGBT(Ideal, Switching) blocks. To load the device parameters in the workspace, at the MATLAB Command Window, enter:

### High-Power Converter Design Data Flow Diagram

This figure shows the data flow and the respective file names used in this example.




### Select the IGBT Device

This example is based on a 15 kW converter that operates in a three-phase grid with frequency of 60Hz and line voltage of 208 V. The RMS current of the converter is equal to 15e3/(sqrt(3)*208) = 41.6 Amperes and the current peak is equal to 59 Amperes. Because the minimum required DC bus voltage is equal to 2*208*sqrt(2/3) = 340 Volts, this example chooses a standard 400V DC bus.

Due to the required currents and voltage, the converter needs an IGBT rated at more than 450Vdc and 50A. The TO-247 package contains an IGBT device with a continuous current rating of 50 Amperes and 650 Volts rating. The parasitic inductance of the busbar that connects the IGBT device with the DC bus is assumed to be 30nH. The busbar inductance is calculated either through lab measurement or by 3-D electromagnetic simulation.

### Extract IGBT Parameters

IGBT devices are the most used semiconductor switches for high-power inverters thanks to their unique ability of better switching frequency, simpler gate driver, and available power rating. The N-Channel IGBT and the IGBT (Ideal, Switching) blocks in the Simscape™ Electrical™ library both model an IGBT device. The two blocks provide support for different modeling fidelity levels. The IGBT (Ideal, Switching) block is more suited to system-level simulations. The N-Channel IGBT block is more

suited to detailed designs, such as gate driver design and simulating peak switching voltage and current.

**N-Channel IGBT Block Overview**

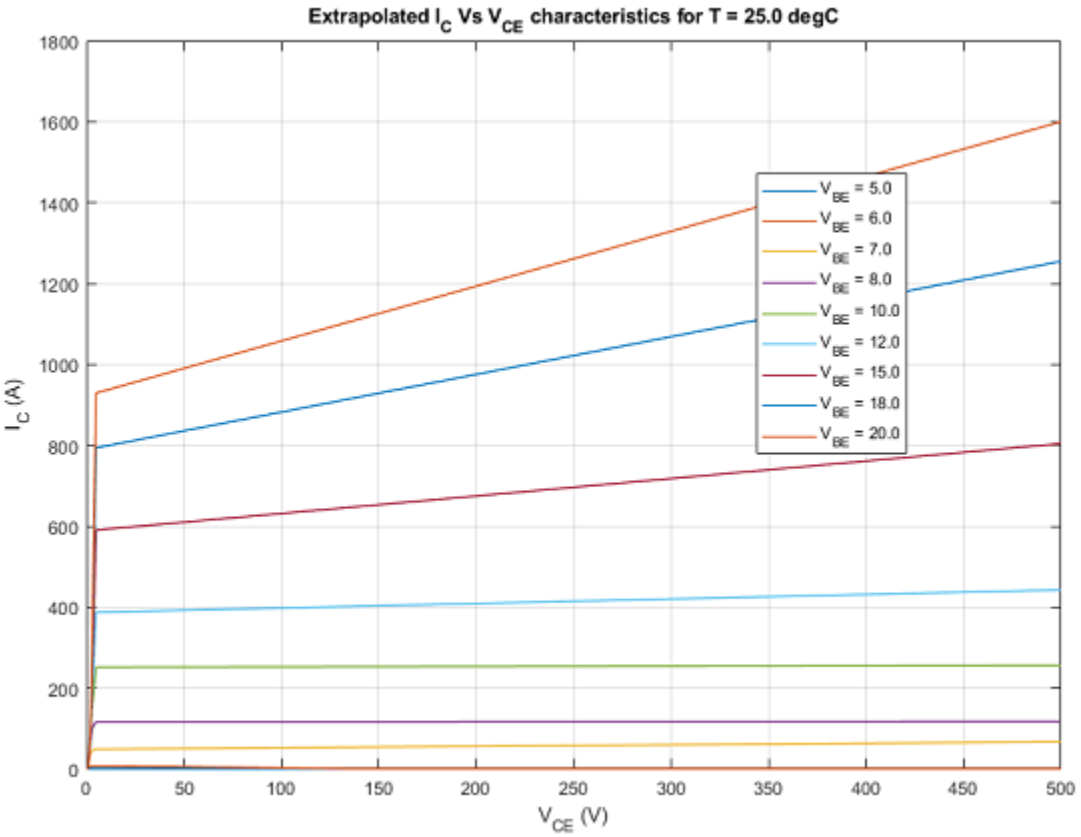
The N-Channel IGBT block provides a detailed model of an IGBT device by supporting the definition of non-linear I-V characteristics, either through physical equations or by lookup tables. It also supports the definition of nonlinear charge characteristics either through simple physical equations or lookup tables. The block parameterization matches the data in the manufacturer datasheet. The datasheet information usually includes:

1. $I_{ce}\ versus\ V_{ce}$ at two different junction temperatures.
2. Capacitance $(C_{ies},\ C_{oes},\ C_{res})$ as a function of $V_{ce}$.
3. Diode $I_F$ as a function of $V_F$ for two junction temperatures.
4. Parameters for thermal model (Foster, Cauer, or simple $R_{jc}\ and\ R_{ca}$).
5. Switching loss curves $E_{on}\ versus\ I_{ce}$ and $E_{off}\ versus\ I_{ce}$

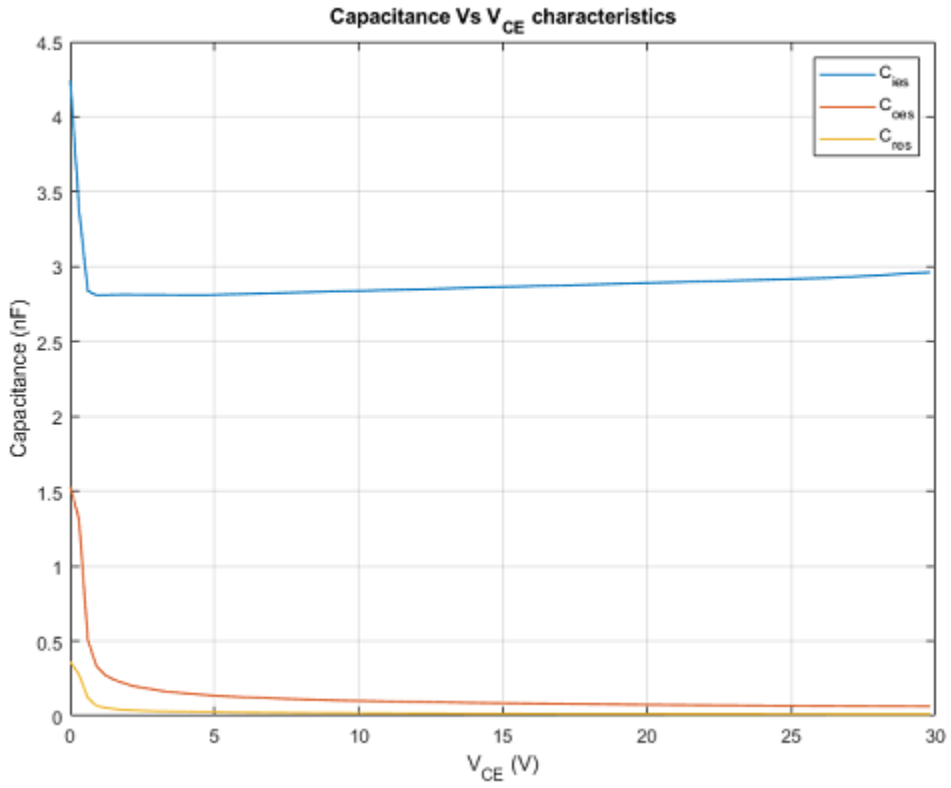
You can create the necessary tabulated data for the block from the datasheet characteristic plots. You can perform a more approximate parameterization by using the datasheet tabulated values only. Note that many datasheets only provide the $I_{ce}$$ versus $V_{ce}$$ data for the $V_{ce}$$ up to 5V, but a switching IGBT device goes through a transition from 0V to the DC bus voltage. In these cases, to avoid extrapolation to non-physical current values, provide some estimated values at higher voltage ranges.

In converters, an IGBT device operates in switching mode. For good simulation speed and accuracy, you must ensure accurate and smooth $I_{ce}(V_{ge},\ V_{ce},\ T_j)$$ values around the smaller collector-emitter voltages.

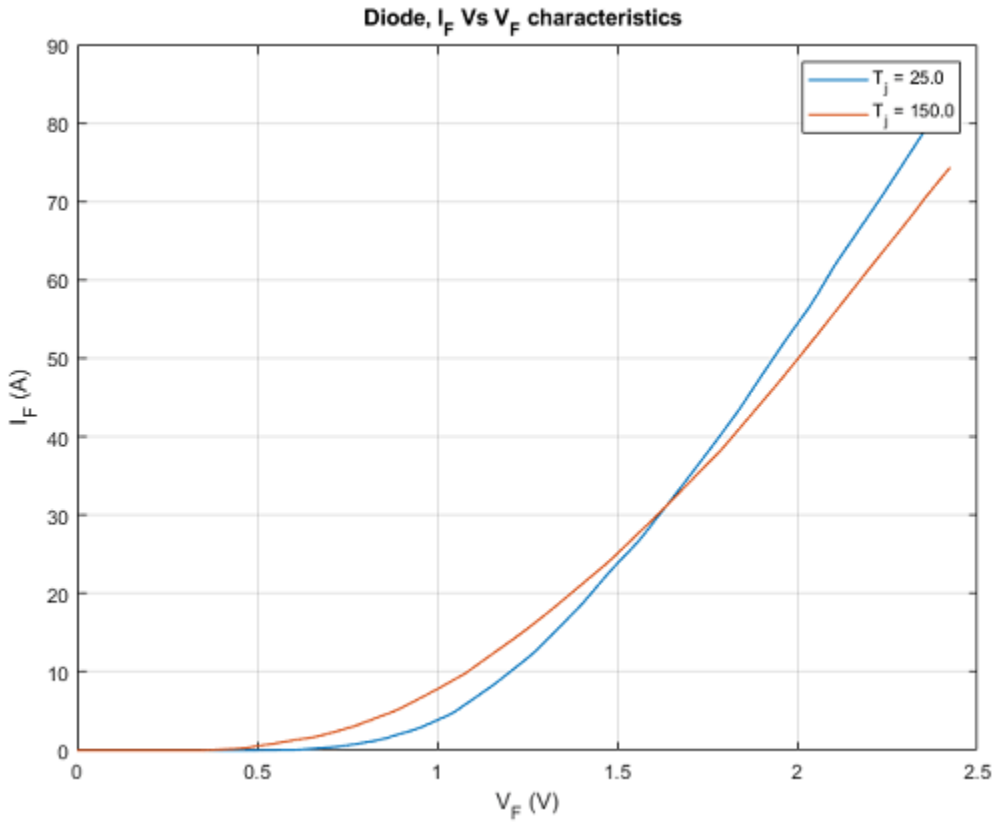
This plot shows the extension of the datasheet data to include points at the DC bus voltage for the device in this example.




% This plot shows the digitalized device capacitance characteristics.

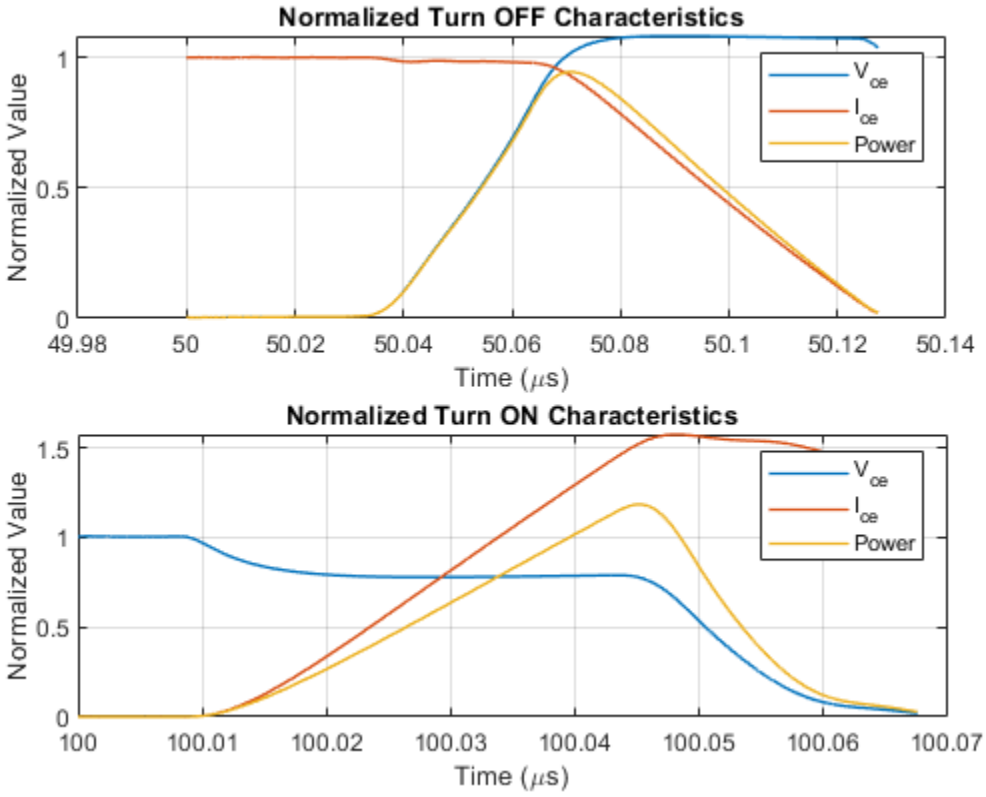



This plot shows the external diode forward current characteristics $I_F(T_j,\ VF)$, obtained from the IGBT datasheet.




**Simulated Switching Transient**

This figure shows the simulated N-Channel IGBT switching turn-on and turn-off characteristics.

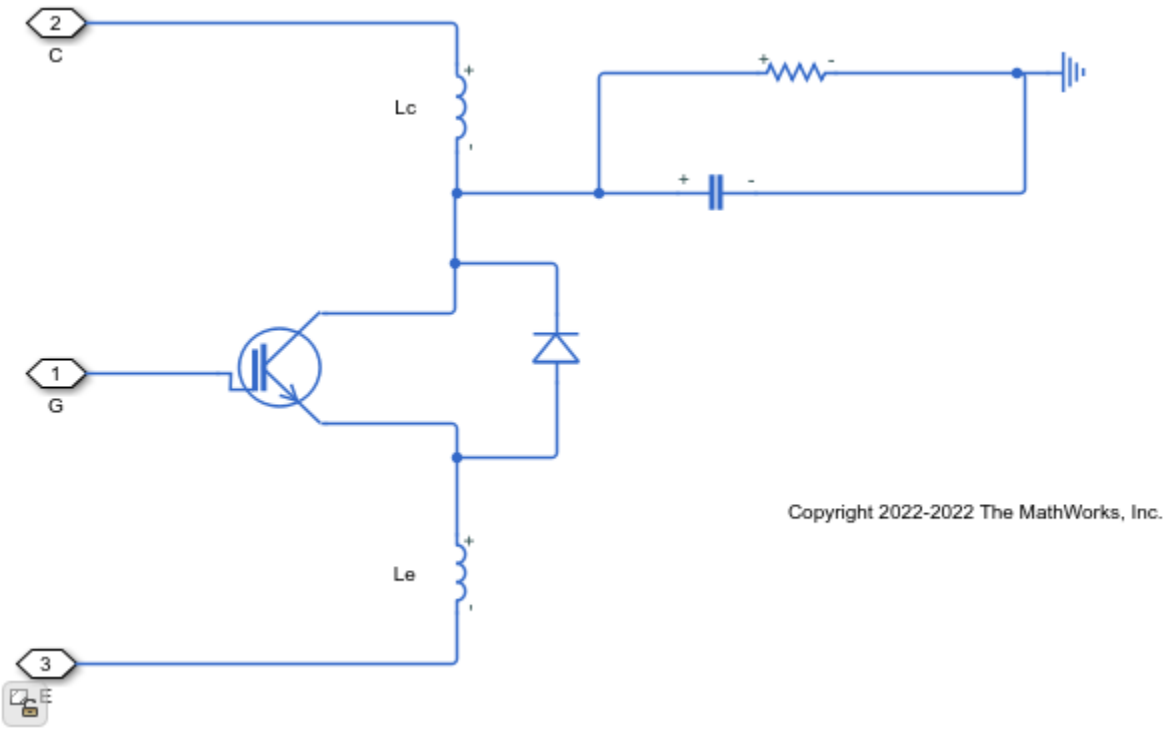



### IGBT (Ideal, Switching) Library Block

The detailed N-Channel IGBT block requires very low simulation time steps to capture the turn-on and turn-off transients. This slows down the simulation too much when this block is used in large models. To improve the simulation speed, you can use the IGBT(Ideal, Switching) block. This is ideal when you are looking at system-level harmonics or thermal simulation including heat sink dynamics and cooling. The IGBT(Ideal, Switching) block implements instantaneous junction temperature changes due to conduction, turn-on and turn-off losses.

#### N-Channel IGBT Device Model Subsystem

A Subsystem Reference model contains the parameterized N-Channel IGBT block. Use this subsystem to define the block once and then use it in multiple models. The Subsystem Reference includes the terminal lead parasitic inductance.

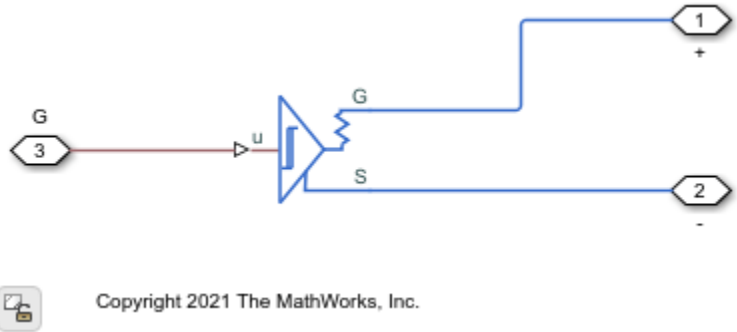



### Design the Gate Driver

Simscape Electrical also helps you to design a gate driver circuit for controlling power electronics devices. This example models three gate drivers to switch semiconductor devices.

#### Ideal Gate Driver

An ideal gate driver models the charging gate resistance and the discharging gate resistance and provides flexibility to define the turn-on and turn-off gate voltages $V_{ge}$.

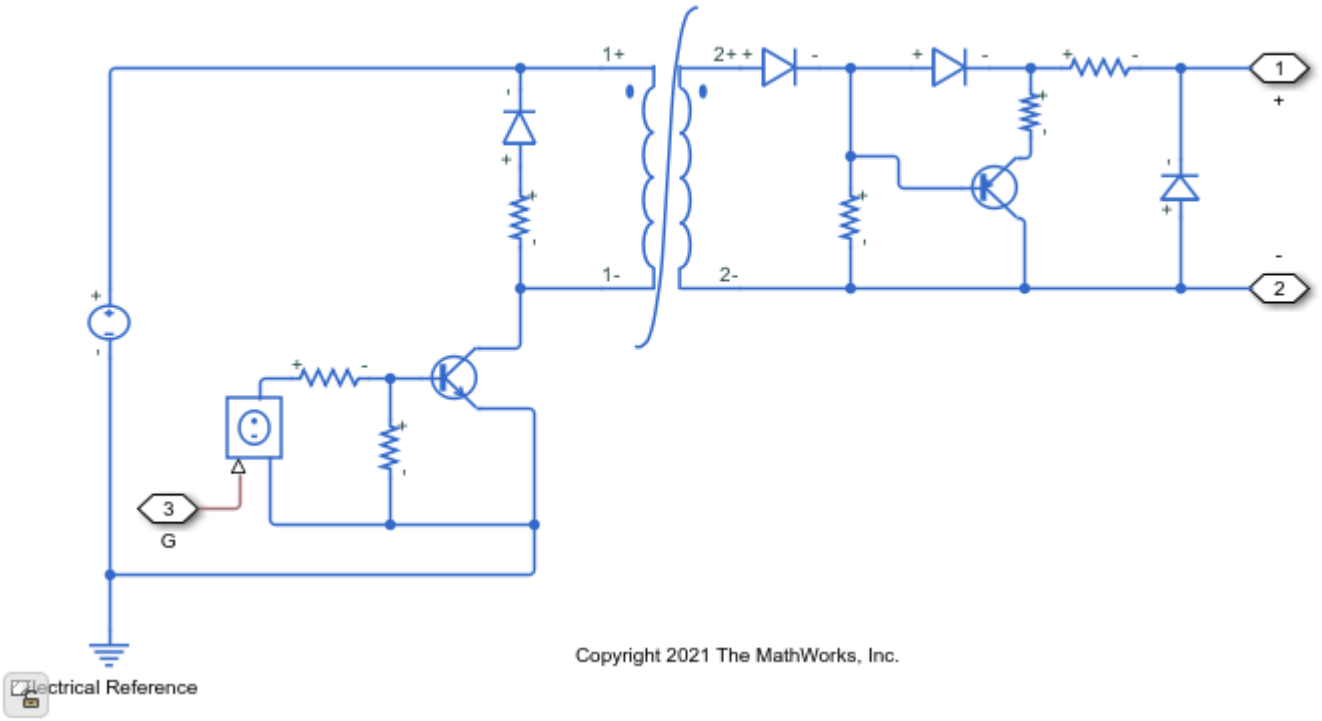



#### Gate Driver with Galvanic Isolation (Pulse Transformer)

A gate driver with galvanic isolation is a high frequency transformer-based gate driver that requires only a single gate driver power supply. This supply is isolated from the power circuit.

To simulate the gate driver with electromagnetic isolation with the N-Channel IGBT block, open the `ee_converter_design_igbt_switchingloss_testharness` SLX model and, in the masked

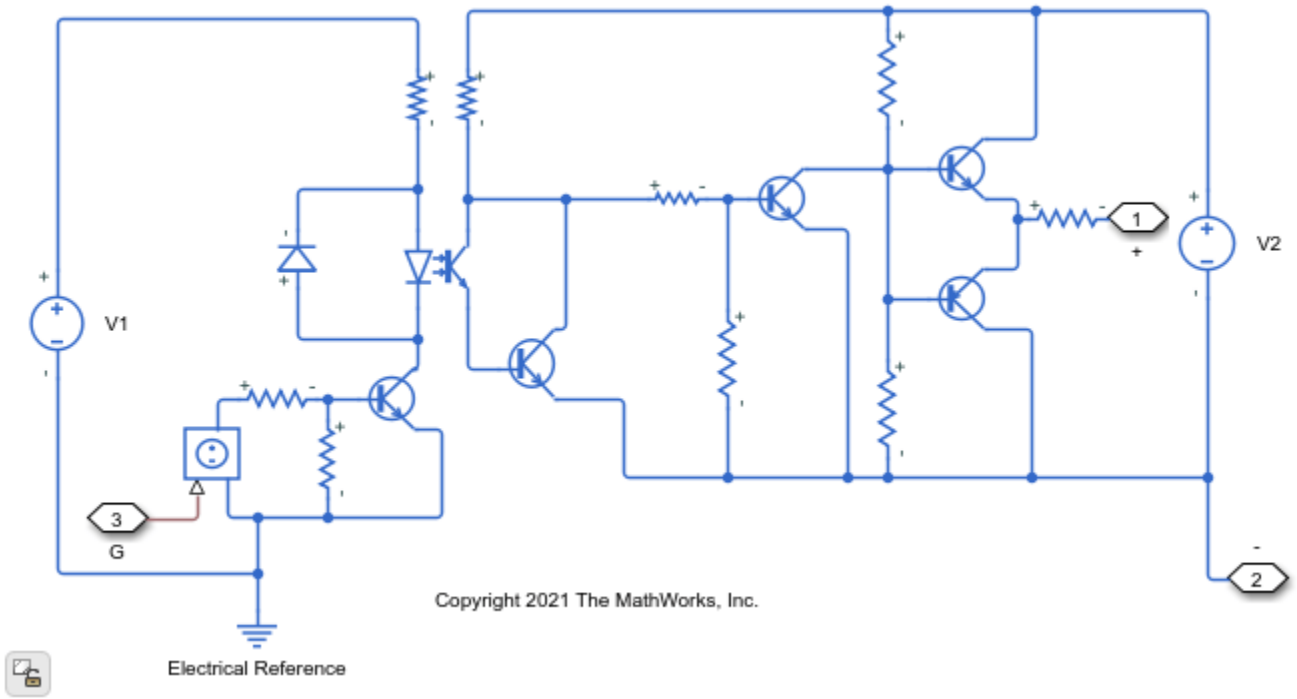
selection subsystem in the top-right corner, choose `Gate driver with electromagnetic isolation`.




**Gate Driver with Opto-Coupler Isolation**

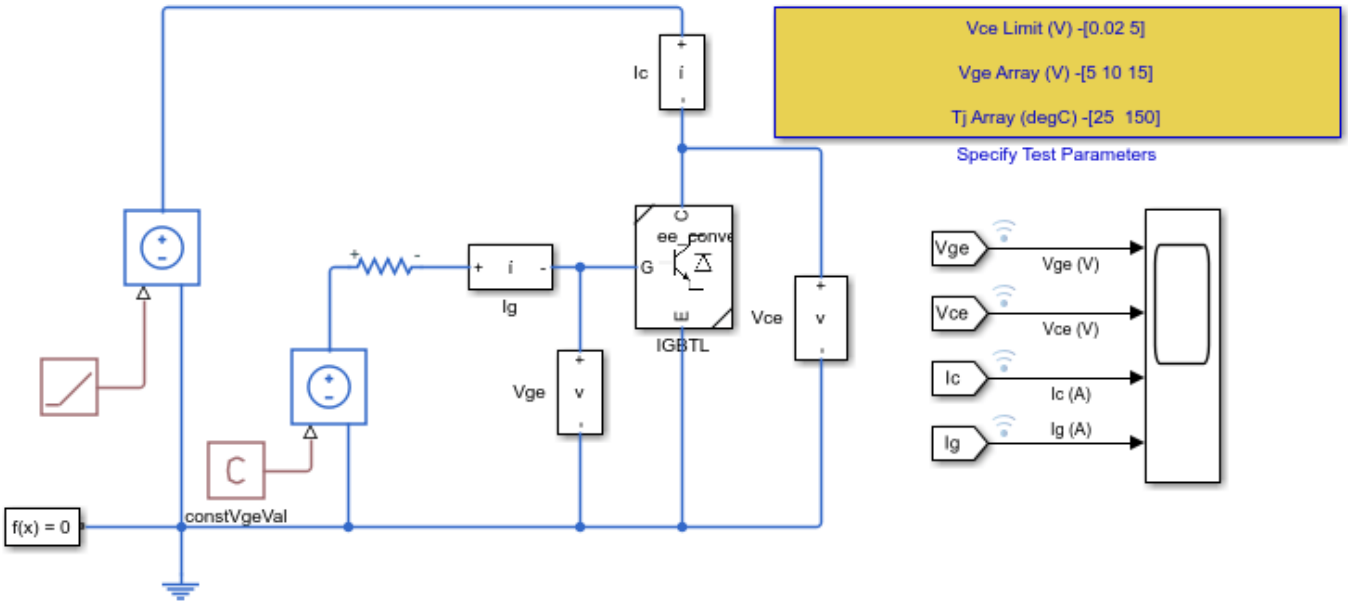
A gate driver with optocoupler isolation is a compact gate driver with lower losses. This gate driver requires two gate driver power supplies that are isolated from each other and from the power circuit.

To simulate the gate driver with optocoupler isolation with the N-Channel IGBT block, open the `ee_converter_design_igbt_switchingloss_testharness` SLX model and, in the masked selection subsystem in the top-right corner, choose `Gate driver with opto-coupler`.




**Test Harness for IGBT Steady State Characteristics**

The `ee_converter_design_igbt_iv_characteristics` SLX model provides test harness to plot the steady state $(I_{ce}\ versus\ V_{ce})$ curve for different junction temperature $(T_j)$ and gate-emitter voltage $(V_{ge})$.



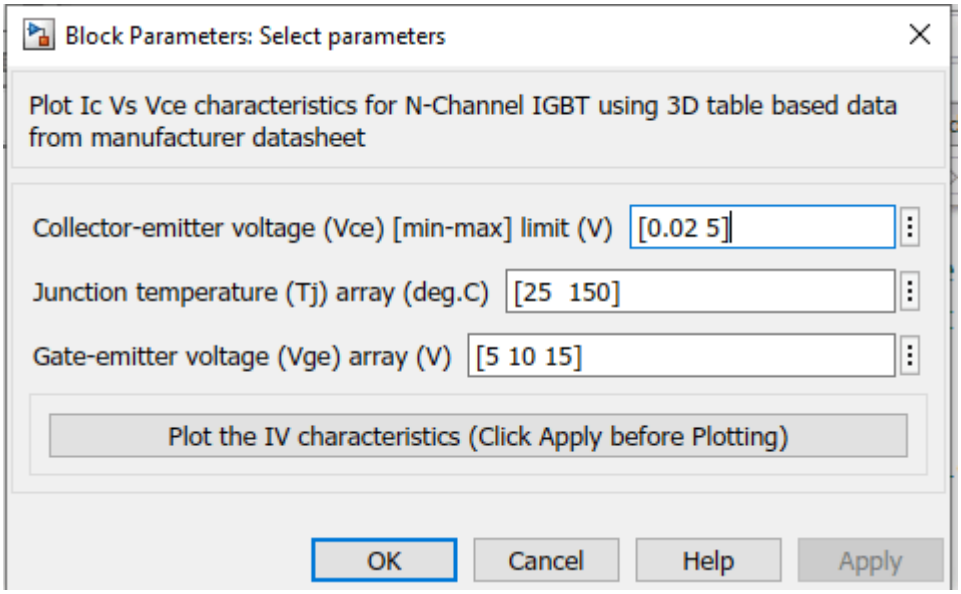

**Test Harness to Generate IV Characteristics of N-Channel IGBT**

1 Open test harness configuration script (load data )
2. Plot IV characteristics of N-Channel IGBT (see code)
3 Open script main converter design workflow
4. Explore simulation results using sscexplore
5. Learn more about this example

Copyright 2021-2022 The MathWorks, Inc.

### IGBT Steady State Characteristics Test Harness Parameter Selection Mask

To define simulation parameters, in the `ee_converter_design_igbt_iv_characteristics` SLX model, open the "Specify Test Parameters" mask in the top right corner. Then click on the "Plot iv-characteristics" button to get the required i-v characteristics curve.



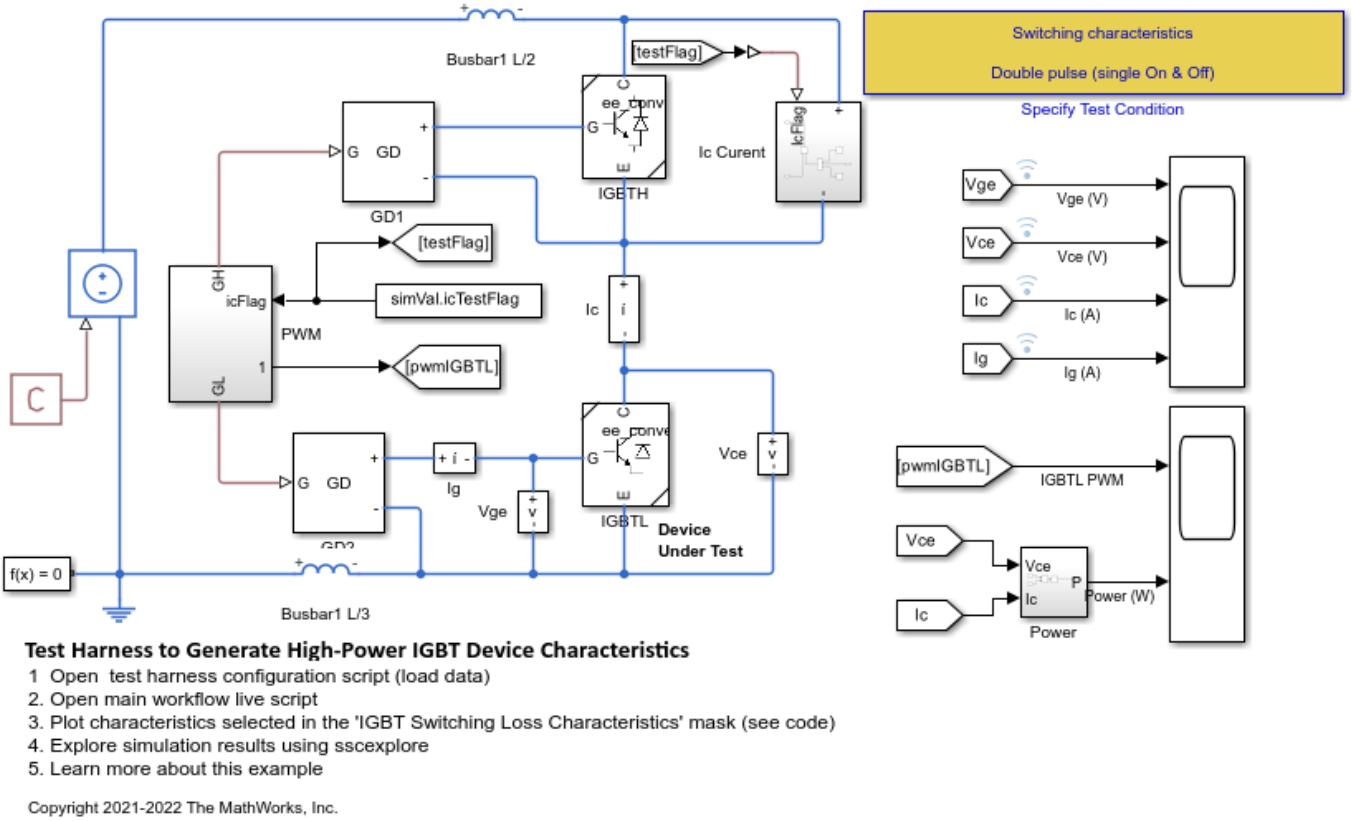
### Test Harness for High-Power Converter Switching Loss Estimation

The model `ee_converter_design_igbt_switchingloss_testharness`, provides test harness that plots the variation of switching loss characteristics for these IGBT simulation parameters:

1. Collector-emitter current $(I_{ce})$
2. PWM switching frequency $(f_{pwm})$
3. Gate-emitter voltage $(V_{ge})$
4. Collector-emitter voltage $(V_{ce})$
5. Gate resistance $(R_{gon}\ and\ R_{goff})$
6. Junction temperature $(T_j)$

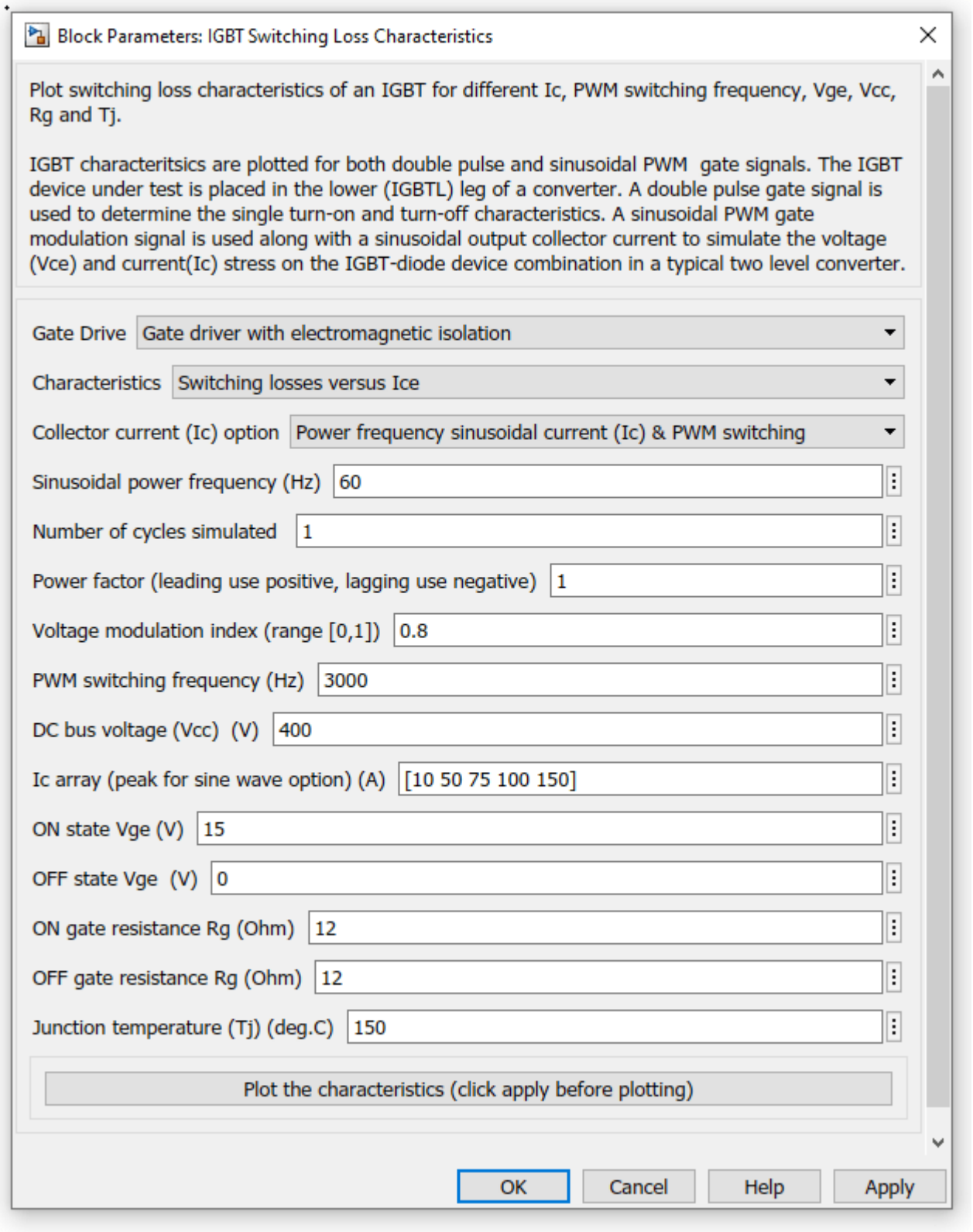
This model plots the IGBT characteristics for both double pulse and sinusoidal PWM gate signals. The IGBT device under test is placed in the lower (IGBTL) leg of a converter. A double pulse gate signal is used to determine the single turn-on and turn-off characteristics. A sinusoidal PWM gate modulation signal is used along with a sinusoidal output collector current to simulate the voltage $(V_{ce})$ and current $(I_{ce})$ stress on the IGBT-diode device combination in a typical two-level converter. A sinusoidal output load current with sinusoidal voltage modulation signal enables you to simulate for different power factor, power frequency, and switching frequency.

In the two-level converter, when bottom IGBT device is turned-on, the conduction transfer from top diode to bottom IGBT and vice versa during IGBT-turn-off. Busbar inductance influence the turn-off switching loss, it creates off state voltage spike across the device. Diode reserve recovery current affects the IGBT turn-on characteristics. It creates on-state current spike through the device.




### Switching Loss Estimation Test Harness Parameter Selection Mask

To define the simulation parameters and to plot the switching loss characteristics, open the "ee_converter_design_igbt_switchingloss_testharness" model and click the "Specify Test Condition" mask in the top right corner. Click on the "Plot the characteristics" button to get the required switching characteristics curve.


Block Parameters: IGBT Switching Loss Characteristics
Plot switching loss characteristics of an IGBT for different Ic, PWM switching frequency, Vge, Vcc, Rg and Tj.
IGBT characteritsics are plotted for both double pulse and sinusoidal PWM gate signals. The IGBT device under test is placed in the lower (IGBTL) leg of a converter. A double pulse gate signal is used to determine the single turn-on and turn-off characteristics. A sinusoidal PWM gate modulation signal is used along with a sinusoidal output collector current to simulate the voltage (Vce) and current(Ic) stress on the IGBT-diode device combination in a typical two level converter.
Gate Drive
Gate driver with electromagnetic isolation
Characteristics
Switching losses versus Ice
Collector current (Ic) option
Power frequency sinusoidal current (Ic) & PWM switching
Sinusoidal power frequency (Hz)
60
Number of cycles simulated
1
Power factor (leading use positive, lagging use negative)
1
Voltage modulation index (range [0,1])
0.8
PWM switching frequency (Hz)
3000
DC bus voltage (Vcc) (V)
400
Ic array (peak for sine wave option) (A)
[10 50 75 100 150]
ON state Vge (V)
15
OFF state Vge (V)
0
ON gate resistance Rg (Ohm)
12
OFF gate resistance Rg (Ohm)
12
Junction temperature (Tj) (deg.C)
150
Plot the characteristics (click apply before plotting)
OK
Cancel
Help
Apply

### Estimate the IGBT (Ideal, Switching) Block Parameters from N-Channel IGBT

IGBT (Ideal, Switching) blocks are better suited for simulating large model for harmonics and thermal analysis. Some datasheets directly provide the switching loss $E_{on}$ $E_{off}$ and $V_{ce}(T_j, I_{ce})$ parameters for a particular gate driver, busbar condition.

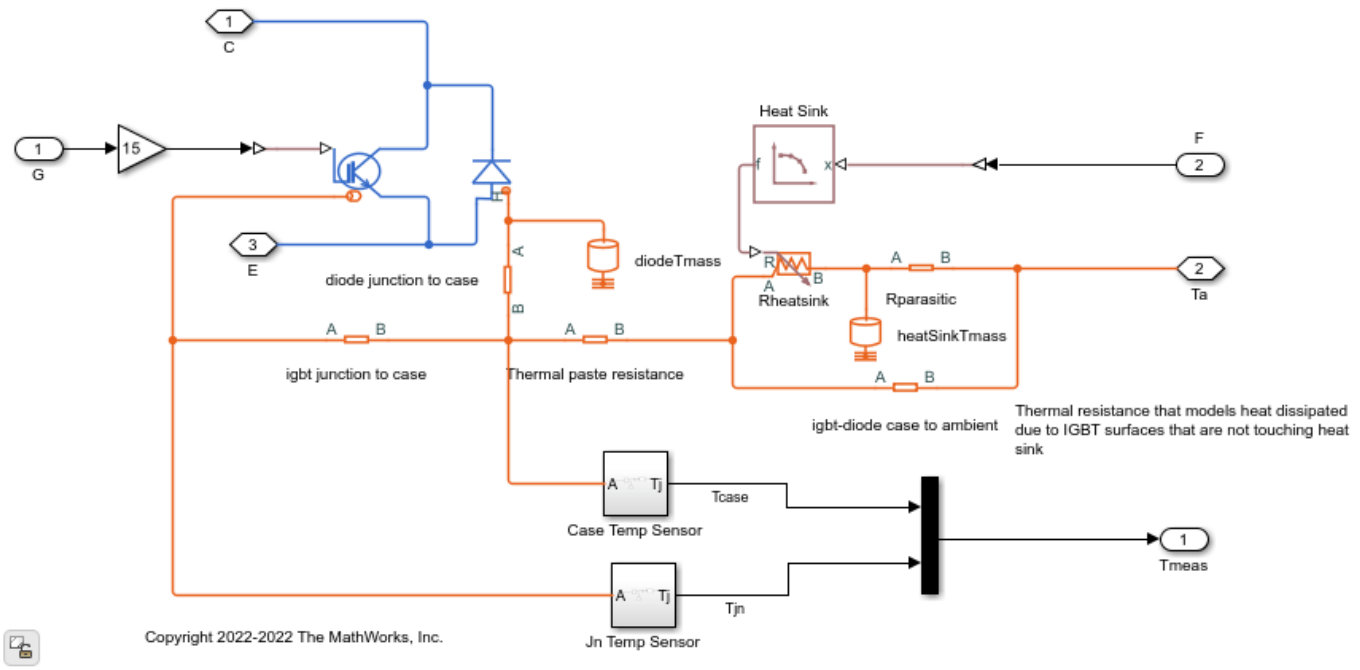
Switching loss and I-V curve are required for the chosen gate resistance $(R_{gon}\ and\ R_{off})$, gate-emitter voltage $(V_{ge})$ and bus bar parasitic inductance.

The `ee_converter_design_igbt_ideal` MLX live script file provides a method to estimate IGBT (Ideal, Switching) block parameters for user defined $R_{gon}$, $R_{off}$ $V_{ge}$, and circuit parasitic.

### IGBT (Ideal, Switching) Device Model Subsystem with Cooling Circuit

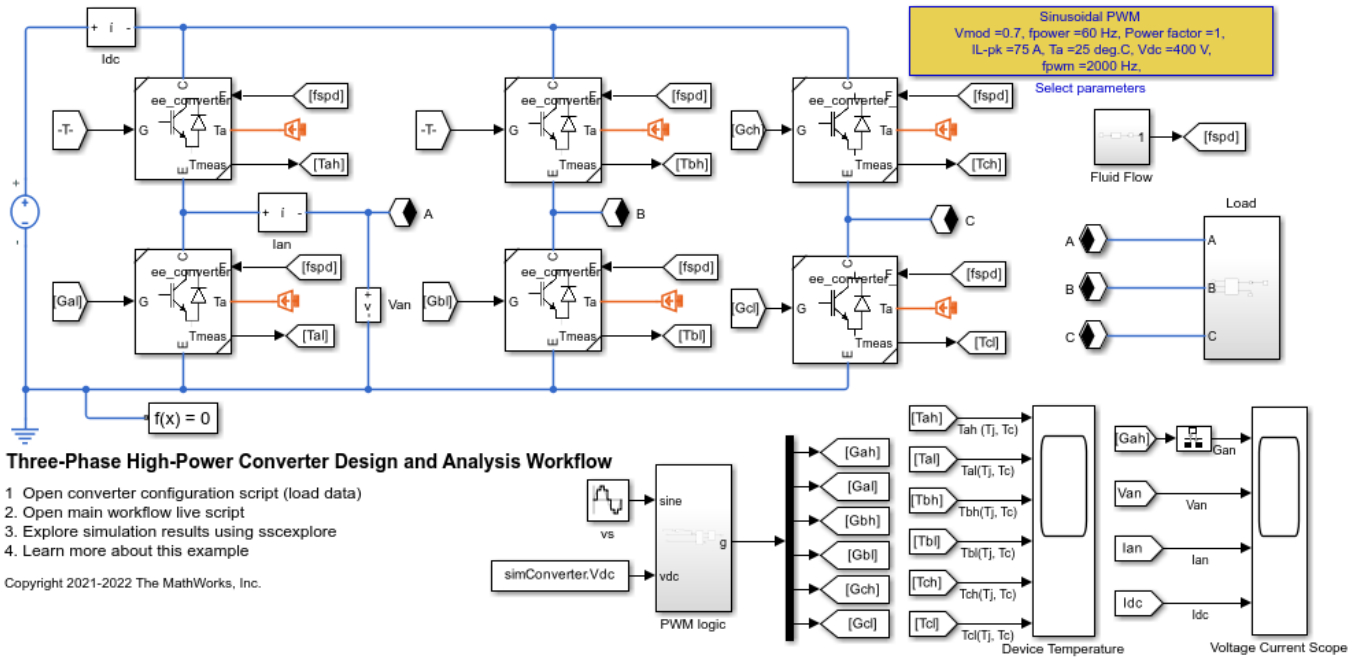
The subsystem reference model of the IGBT (Ideal, Switching) externally represents the thermal circuits by using a junction-to-case thermal resistance and a device thermal mass. A variable thermal resistance models the heat sink with forced cooling. The thermal resistance is fixed based on the cooling fluid flow rate across the heat sink fins.

To model the junction and case temperature rise, you need the IGBT and the diode thermal mass parameters. Some datasheets provide the Foster and Cauer thermal model parameters. Most of the blocks in the Semiconductors & Converters library in Simscape Electrical use these parameters directly. When possible, use Cauer thermal models as you can correctly connect them to any other thermal circuit.



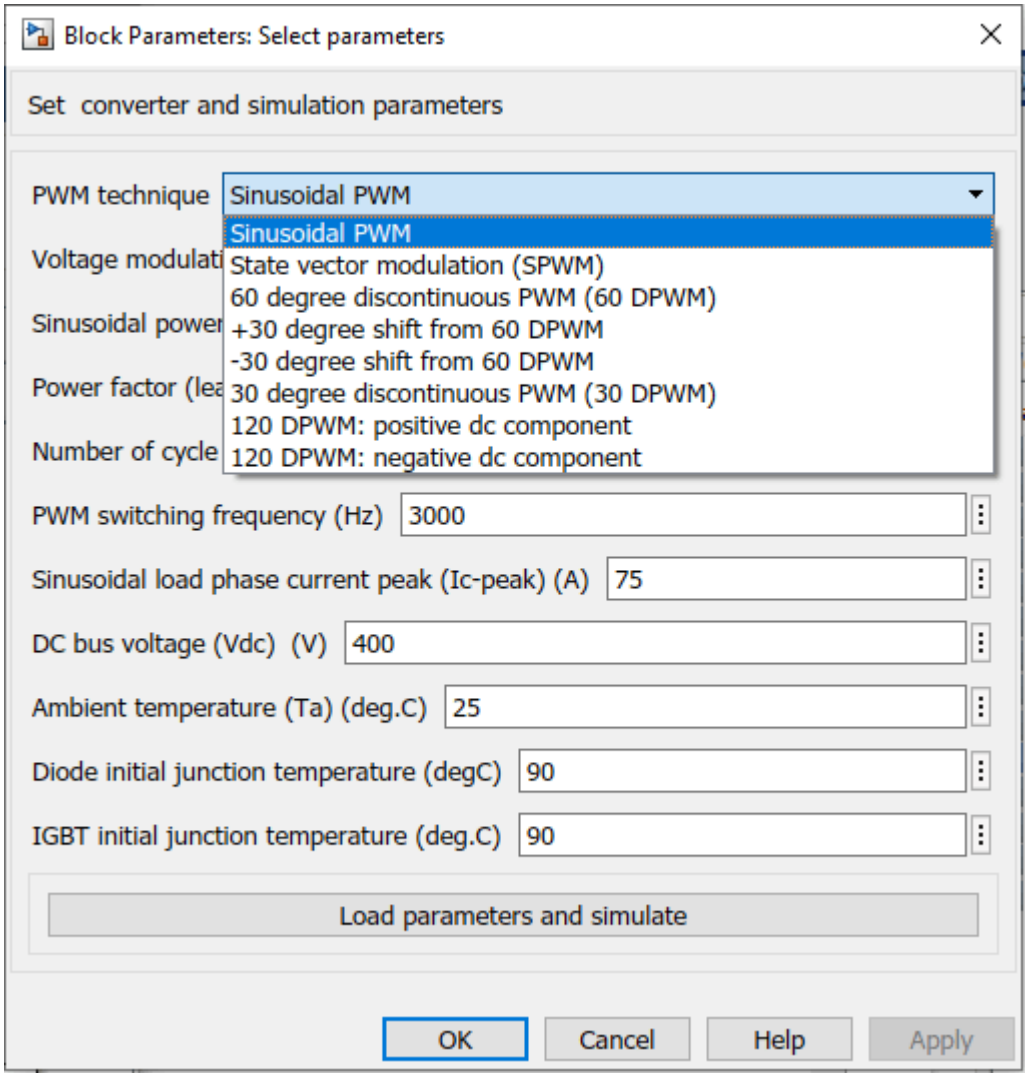

### Three-Phase High Power Converter with Cooling System

To study the temperature rise, use the `ee_converter_design_igbt_thermal_analysis` SLX model. This model helps you choosing the design parameters such as the required cooling system, the PWM technique, and the switching frequency. It also helps you selecting the various simulation parameters by clicking on the "Select parameters" mask on the top right corner of the model.




### Parameter Selection Mask for Three-Phase High Power Converter

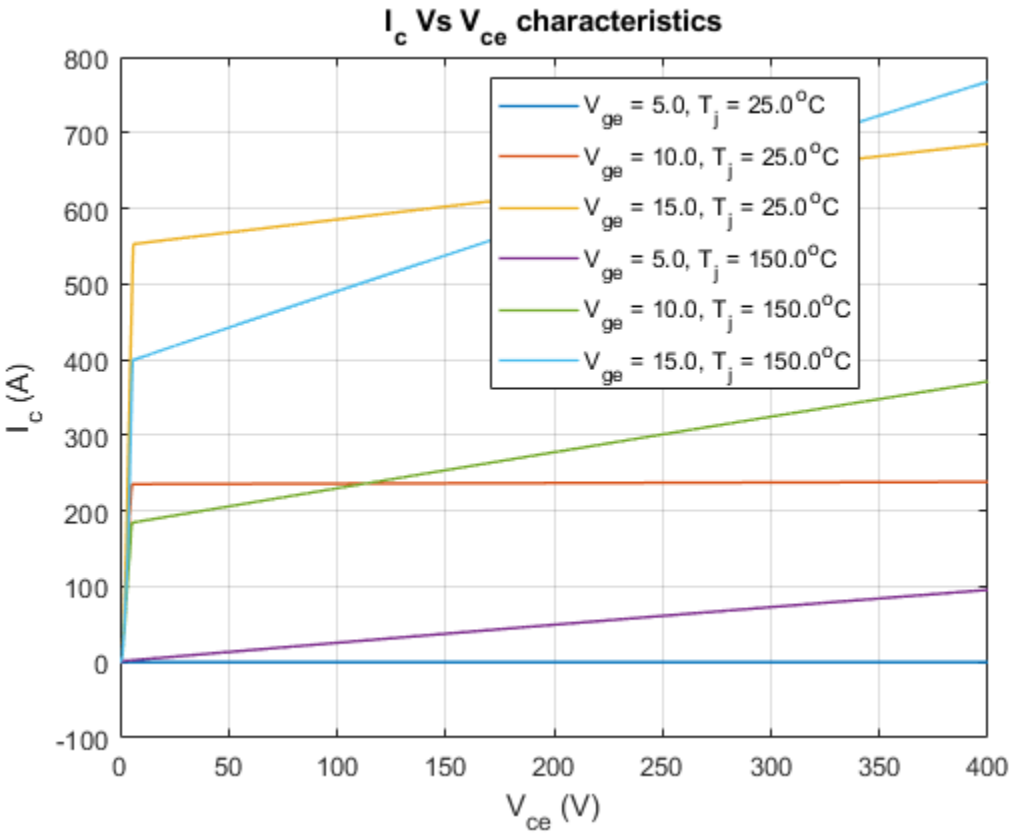
To define the simulation parameters and to simulate the IGBT junction and case temperature rise, open the `ee_converter_design_igbt_thermal_analysis` SLX model and click the "Select parameters" mask in the top right corner. To simulate the system for the given test condition, click on the "Load parameters and simulate" button.




### IV Characteristics Simulation Results

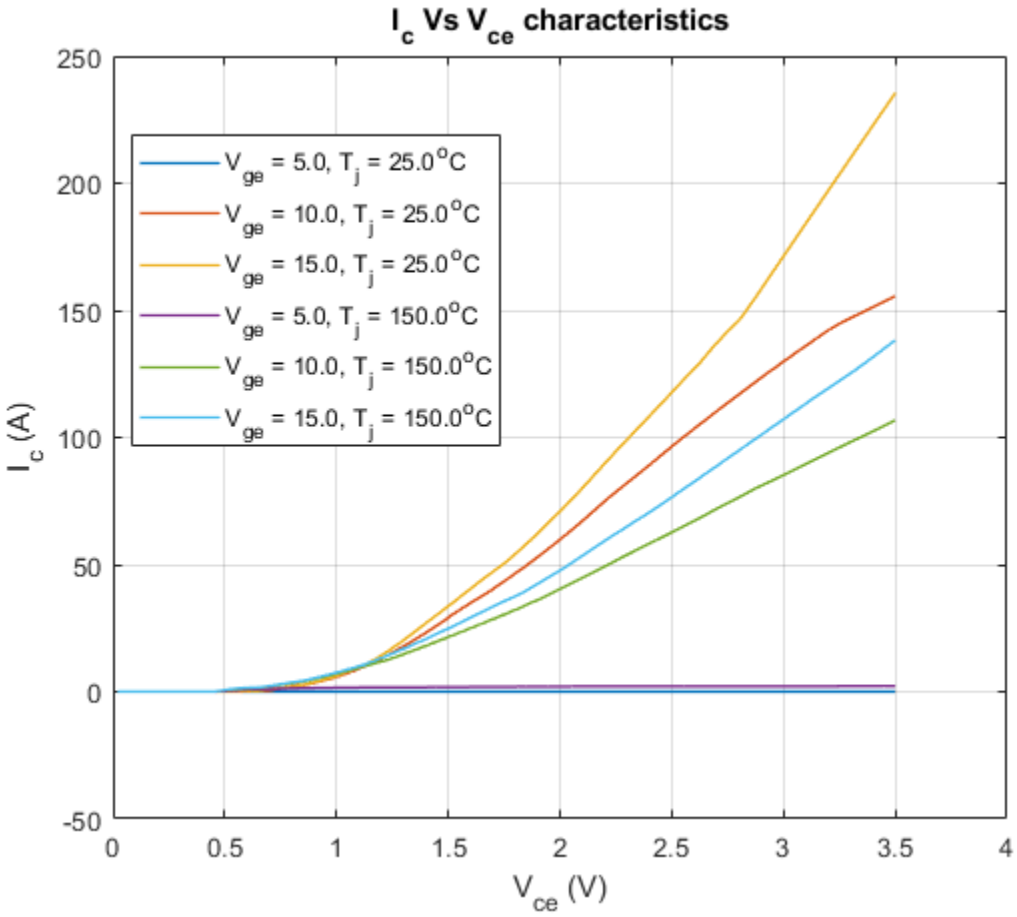
#### IV Characteristics at Full Range

This plot shows IGBT IV curve for $V_{ce}$ from 0 to $V_{dc}$




**IV Characteristics Near Vce(sat)**

This plot shows IV curve near IGBT on-state $V_{ce}$ voltage from 0 to $2V_{cesat}$

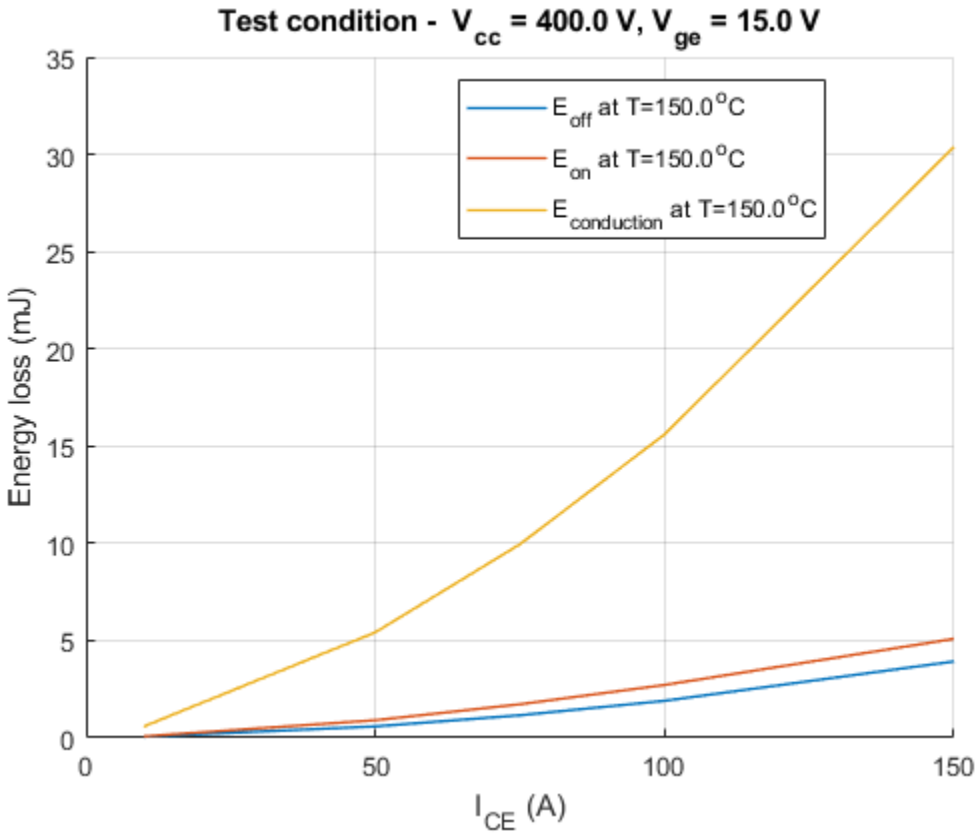



**Simulated Switching Loss Characteristics for Single Turn-On and Turn-Off**

The IGBT switching loss values change according to the definition of the switching region and reverse recovery of the freewheeling diode. In this example, some of the diode reverse recovery is part of the turn-on switching region and any power loss (Vce*Ice) greater than lossFactor*max(Vce*Ice) is a switching loss. This example uses lossFactor of 0.02 (2% of max switching power). During switching, any power loss (Vce*Ice) greater than lossFactor*max(Vce*Ice) is considered as a switching loss.

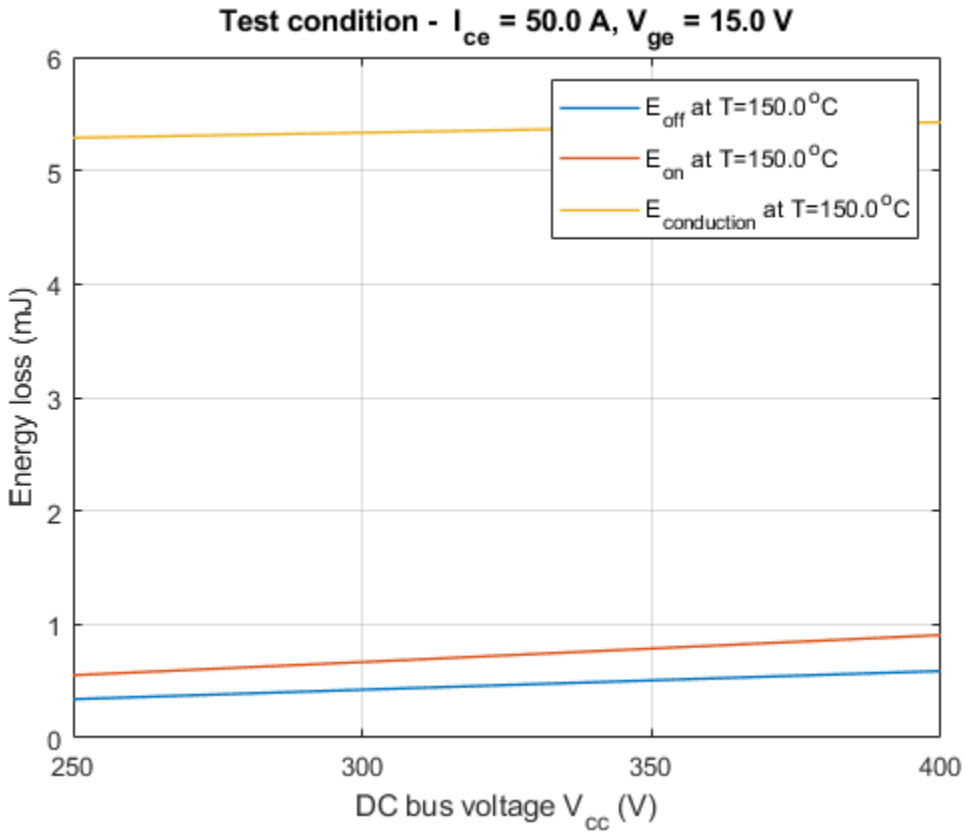
**Simulated Loss for Different Ice**

This plot shows the simulated loss for different Ice.



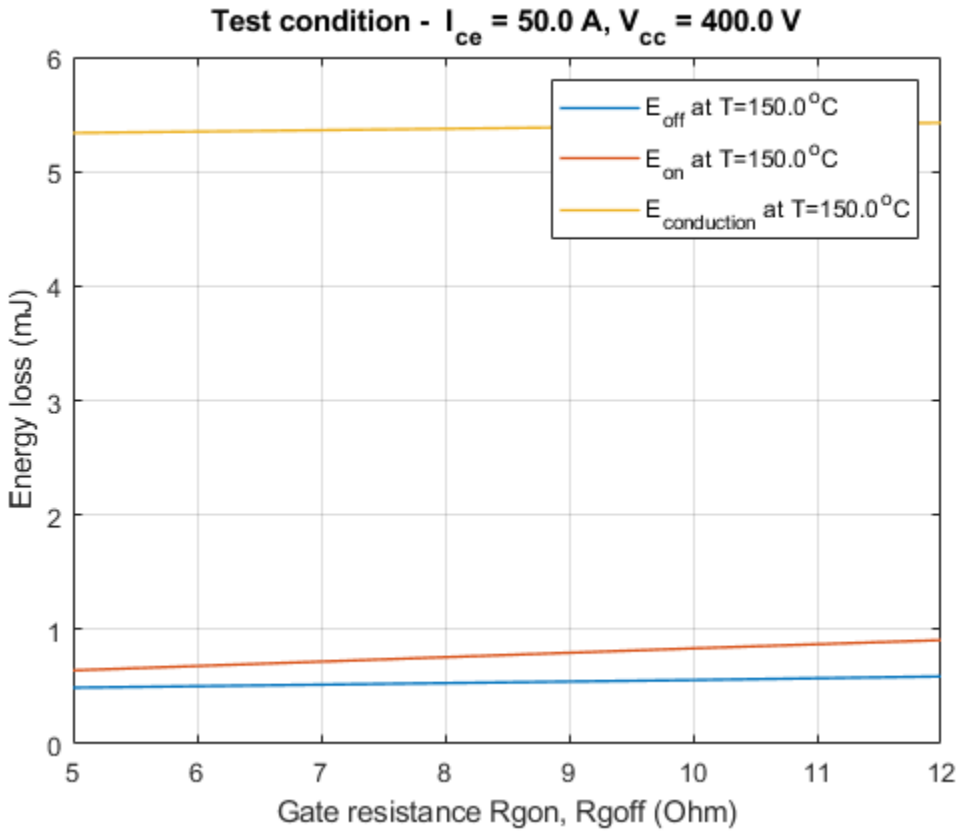

**Simulated Loss for Different Vce**

This plot shows the simulated loss for different Vce.




**Simulated Loss for Different Rg**

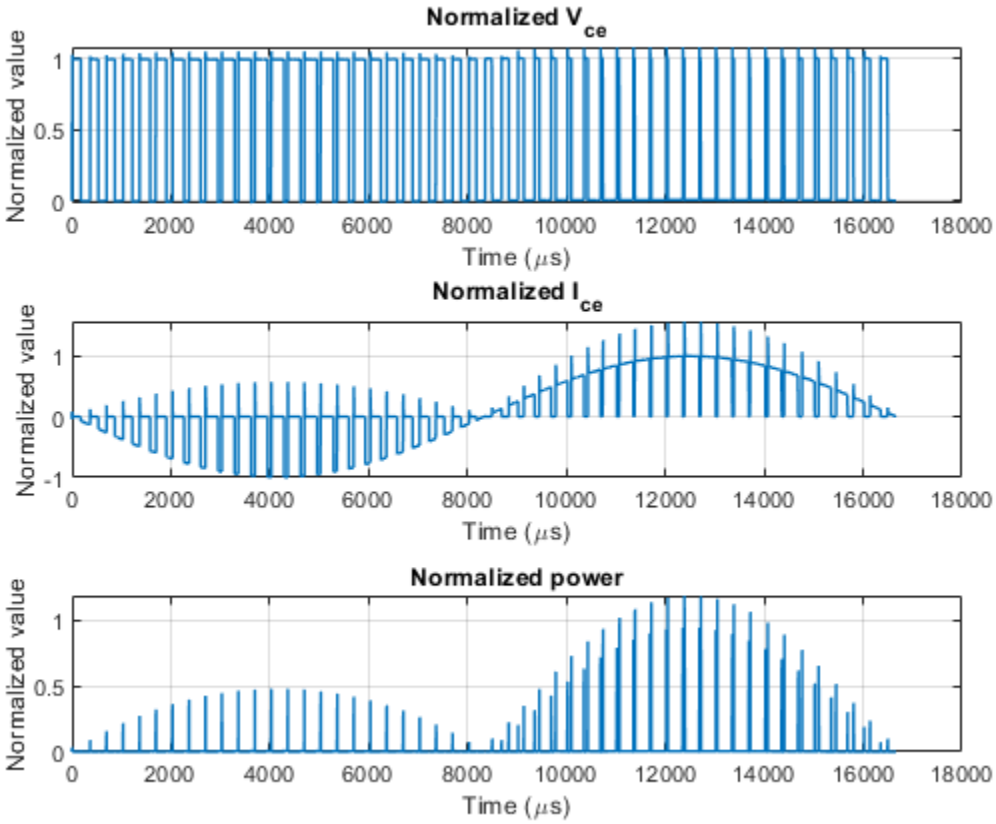
This plot shows the simulated loss for different Rg.




**Simulated Switching Loss Characteristics for Sinusoidal Load Current and Modulation Signal**

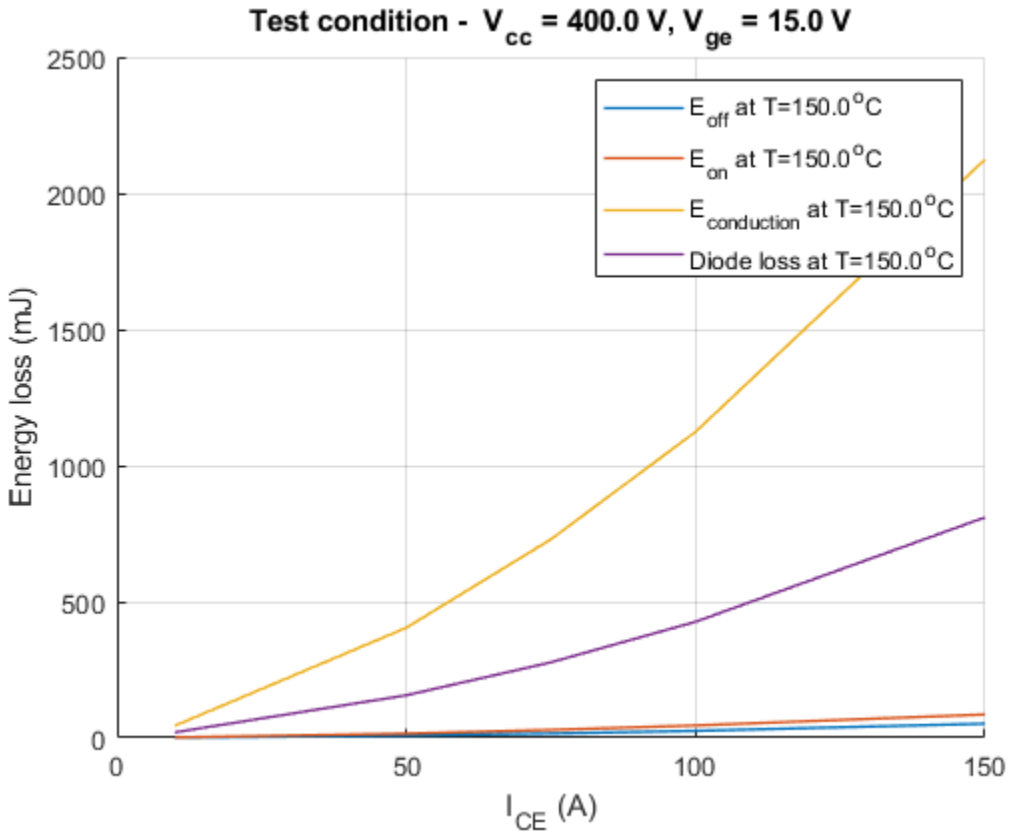
**Simulated Output for Sinusoidal Load Current**

This plot shows the simulated output for the sinusoidal load current when $f_{pwm} = 3000Hz$.



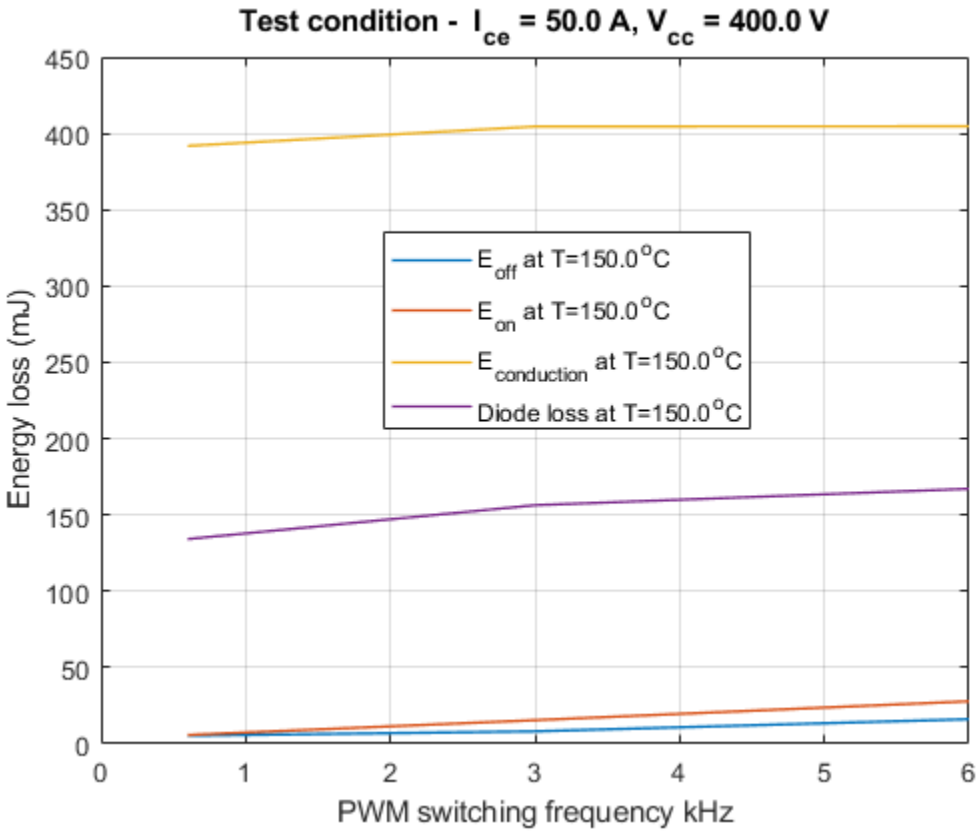

**Simulated Loss for Different Collector Current**

This plot shows the simulated loss for different Ice when $f_{pwm} = 3000Hz$.



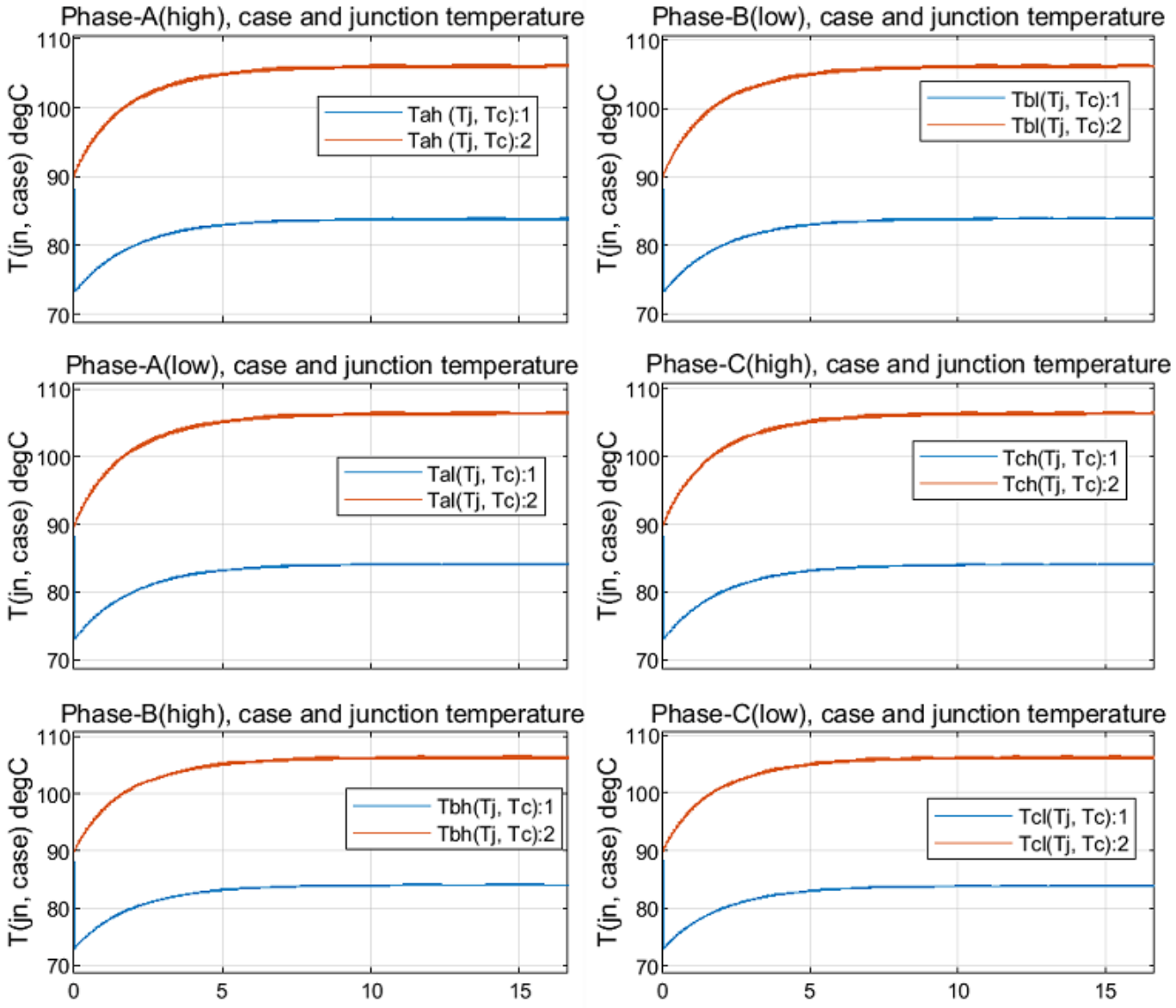

**Simulated Loss for Different PWM Frequency**

This plot shows the simulated loss for different PWM frequencies.




**IGBT Junction and Case Temperature Simulation Result**

Simulated IGBT case and junction temperature


Phase-A(high), case and junction temperature
T(jn, case) degC
Tah (Tj, Tc):1
Tah (Tj, Tc):2
Phase-B(low), case and junction temperature
Tbl(Tj, Tc):1
Tbl(Tj, Tc):2
Phase-A(low), case and junction temperature
Tal(Tj, Tc):1
Tal(Tj, Tc):2
Phase-C(high), case and junction temperature
Tch(Tj, Tc):1
Tch(Tj, Tc):2
Phase-B(high), case and junction temperature
Tbh(Tj, Tc):1
Tbh(Tj, Tc):2
Phase-C(low), case and junction temperature
Tcl(Tj, Tc):1
Tcl(Tj, Tc):2
110
100
90
80
70
0
5
10
15

# Test Harness to Generate High-Power IGBT Device Characteristics

This example provides test harness for estimating switching loss for different parameters of a N-Channel IGBT block.

For more information on designing and analyzing a three-phase converter, refer examples,

1. `ee_converter_design_igbt_iv_characteristics`.
2. `ee_converter_design_igbt_thermal_analysis`.

To access the overall workflow, click on the hyperlink in the model canvas or, at the MATLAB® Command Window, type:

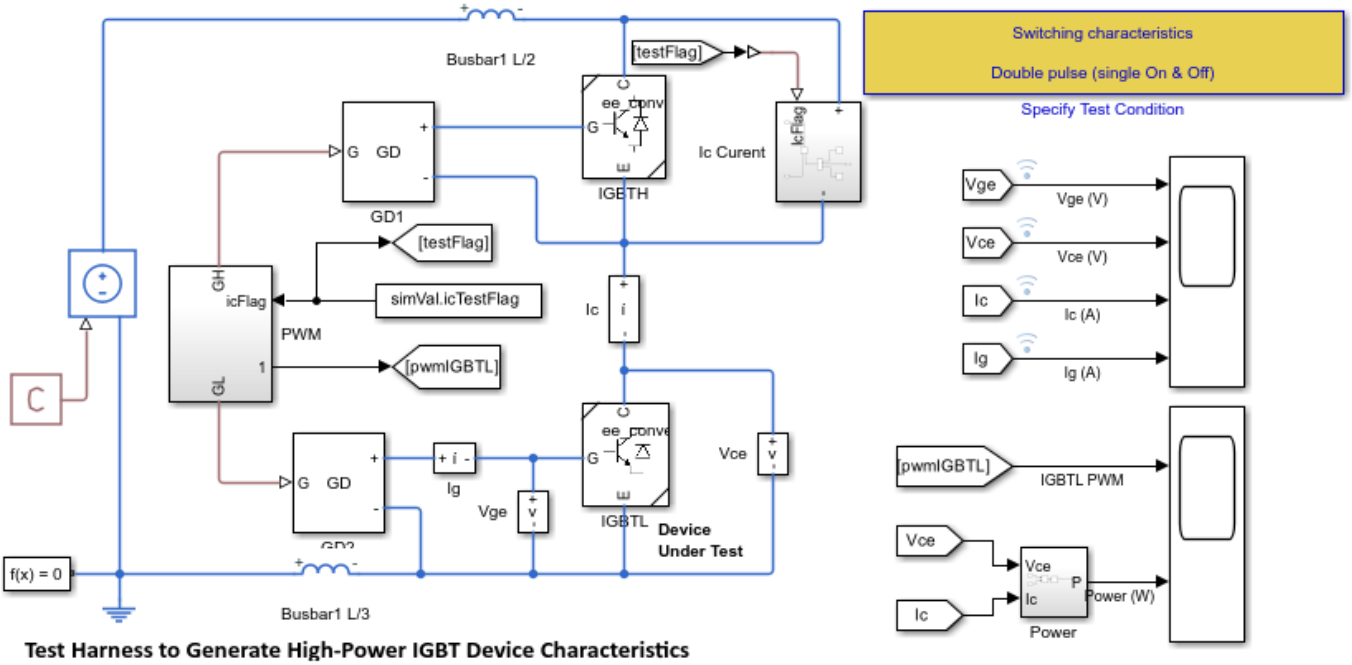
*edit ee_converter_design_igbt_workflow.mlx*

### Test Harness for High-Power Converter Switching Loss Estimation

This model plots the variation of switching loss characteristics for these IGBT simulation parameters:

1. Collector-emitter current $(I_{ce})$
2. PWM switching frequency $(f_{pwm})$
3. Gate-emitter voltage $(V_{ge})$
4. Collector-emitter voltage $(V_{ce})$
5. Gate resistance $(R_{gon}\ and\ R_{goff})$
6. Junction temperature $(T_j)$

This example plots the IGBT characteristics for both double pulse and sinusoidal PWM gate signals. The IGBT device under test is placed in the lower (IGBTL) leg of a converter. To determine the single turn-on and turn-off characteristics, this model uses a double pulse gate signal. To simulate the voltage $(V_{ce})$ and current $(I_{ce})$ stress on the IGBT-diode device combination in a typical two level converter, a sinusoidal PWM gate modulation signal is used along with a sinusoidal output collector current. Sinusoidal $(I_{ce})$ load current with sinusoidal voltage modulation signal option enables you to simulate for different power factor, power frequency, and switching frequency. IGBT characteristics are plotted for both double pulse and sinusoidal PWM gate signals.



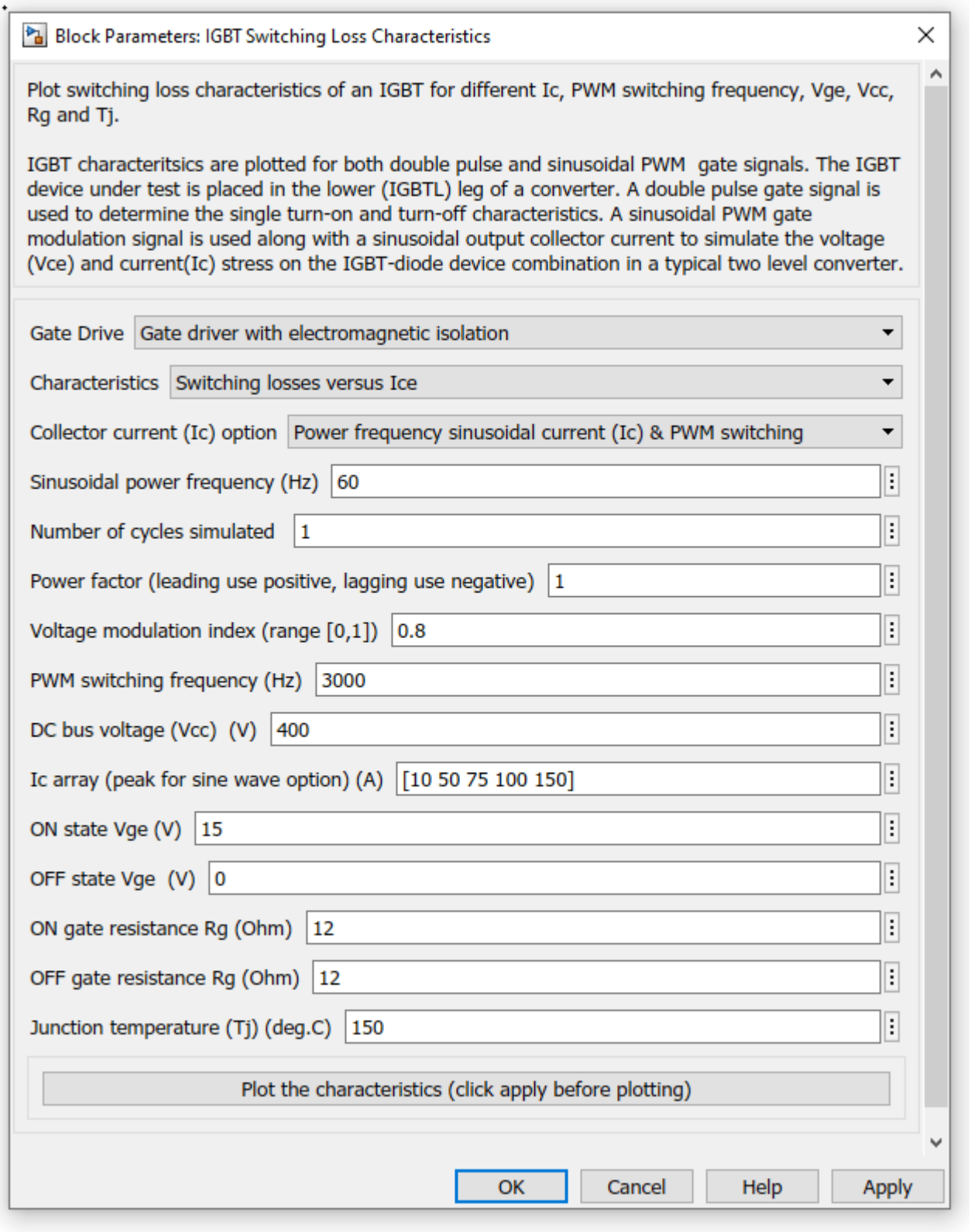

**Test Harness to Generate High-Power IGBT Device Characteristics**

1. Open test harness configuration script (load data)
2. Open main workflow live script
3. Plot characteristics selected in the 'IGBT Switching Loss Characteristics' mask (see code)
4. Explore simulation results using sscexplore
5. Learn more about this example

Copyright 2021-2022 The MathWorks, Inc.

### Parameter Selection Mask for Switching Loss Estimation Test Harness

To define the simulation parameters and plot the switching loss characteristics, open the right-hand top corner parameters selection mask.

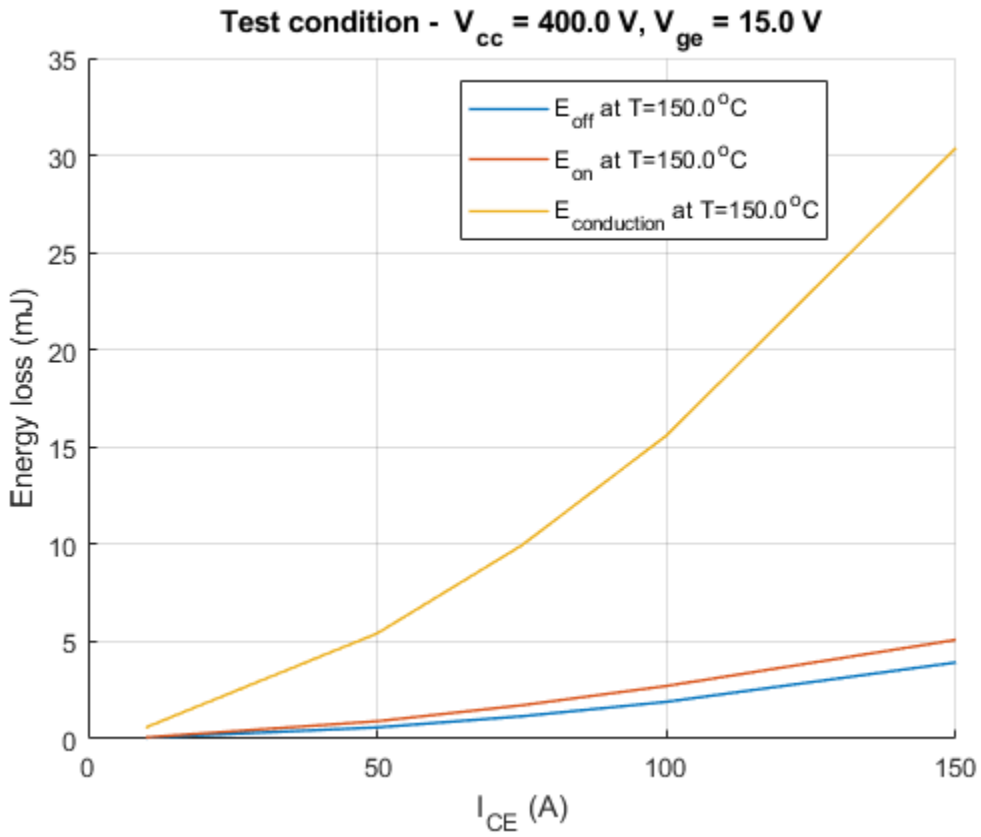

Block Parameters: IGBT Switching Loss Characteristics
Plot switching loss characteristics of an IGBT for different Ic, PWM switching frequency, Vge, Vcc, Rg and Tj.
IGBT characteritsics are plotted for both double pulse and sinusoidal PWM gate signals. The IGBT device under test is placed in the lower (IGBTL) leg of a converter. A double pulse gate signal is used to determine the single turn-on and turn-off characteristics. A sinusoidal PWM gate modulation signal is used along with a sinusoidal output collector current to simulate the voltage (Vce) and current(Ic) stress on the IGBT-diode device combination in a typical two level converter.
Gate Drive
Gate driver with electromagnetic isolation
Characteristics
Switching losses versus Ice
Collector current (Ic) option
Power frequency sinusoidal current (Ic) & PWM switching
Sinusoidal power frequency (Hz)
60
Number of cycles simulated
1
Power factor (leading use positive, lagging use negative)
1
Voltage modulation index (range [0,1])
0.8
PWM switching frequency (Hz)
3000
DC bus voltage (Vcc) (V)
400
Ic array (peak for sine wave option) (A)
[10 50 75 100 150]
ON state Vge (V)
15
OFF state Vge (V)
0
ON gate resistance Rg (Ohm)
12
OFF gate resistance Rg (Ohm)
12
Junction temperature (Tj) (deg.C)
150
Plot the characteristics (click apply before plotting)
OK
Cancel
Help
Apply

### Simulated Switching Loss Characteristics for Single Turn-On and Turn-Off

The IGBT switching loss values change according to the definition of the switching region and on the free wheeling diode. In this example, some of the diode reverse recovery is part of the turn-on switching region and any power loss (Vce*Ice) greater than lossFactor*max(Vce*Ice) is a switching loss. This example uses loss factor of 0.02 (2% of max switching power). During switching, any power loss (Vce*Ice) greater than lossFactor*max(Vce*Ice) is considered as a switching loss.

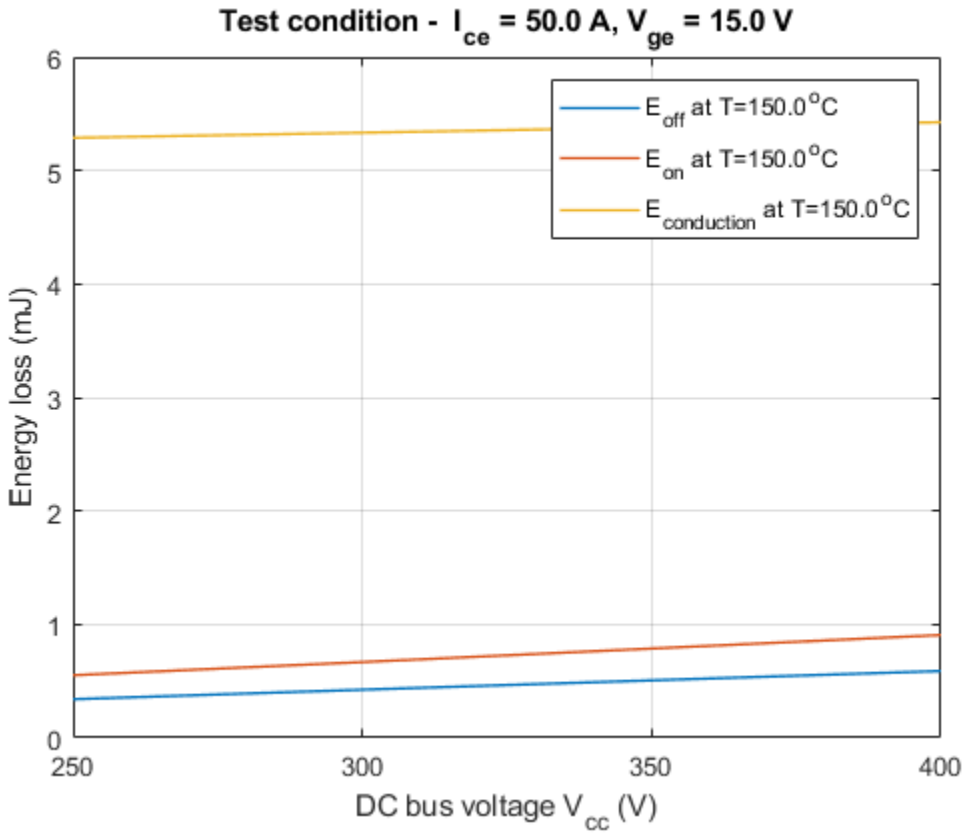
### Simulated Loss for Different Ice

This plot shows the simulated loss for different Ice.



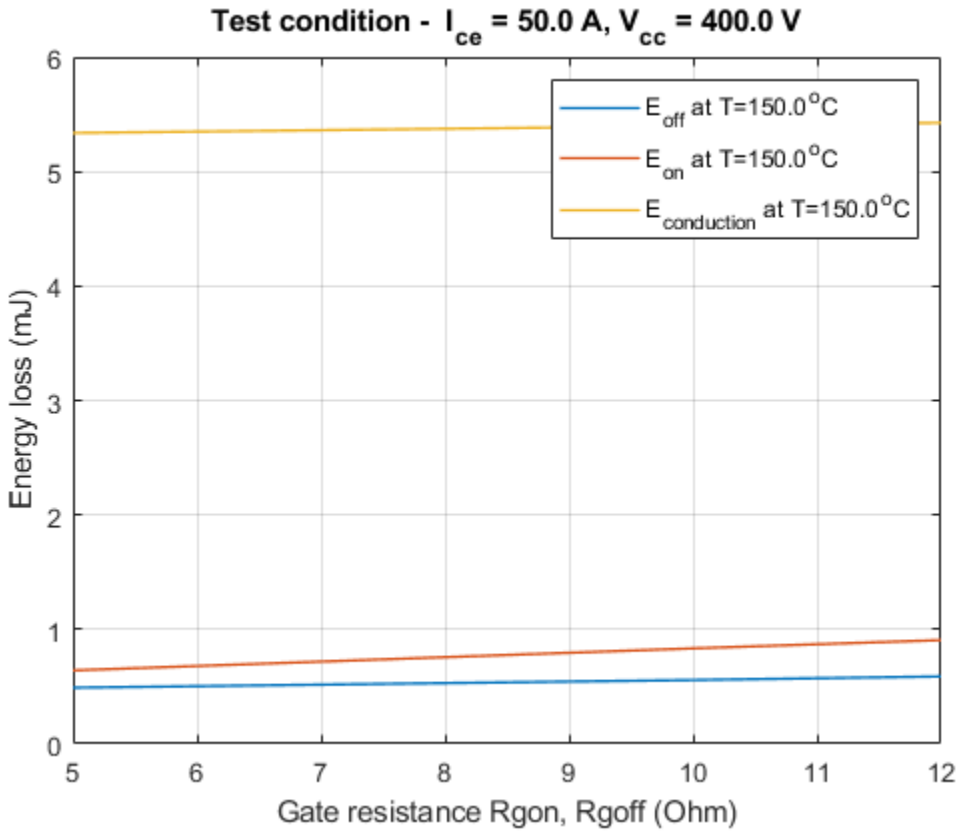

### Simulated Loss for Different Vce

This plot shows the simulated loss for different Vce.



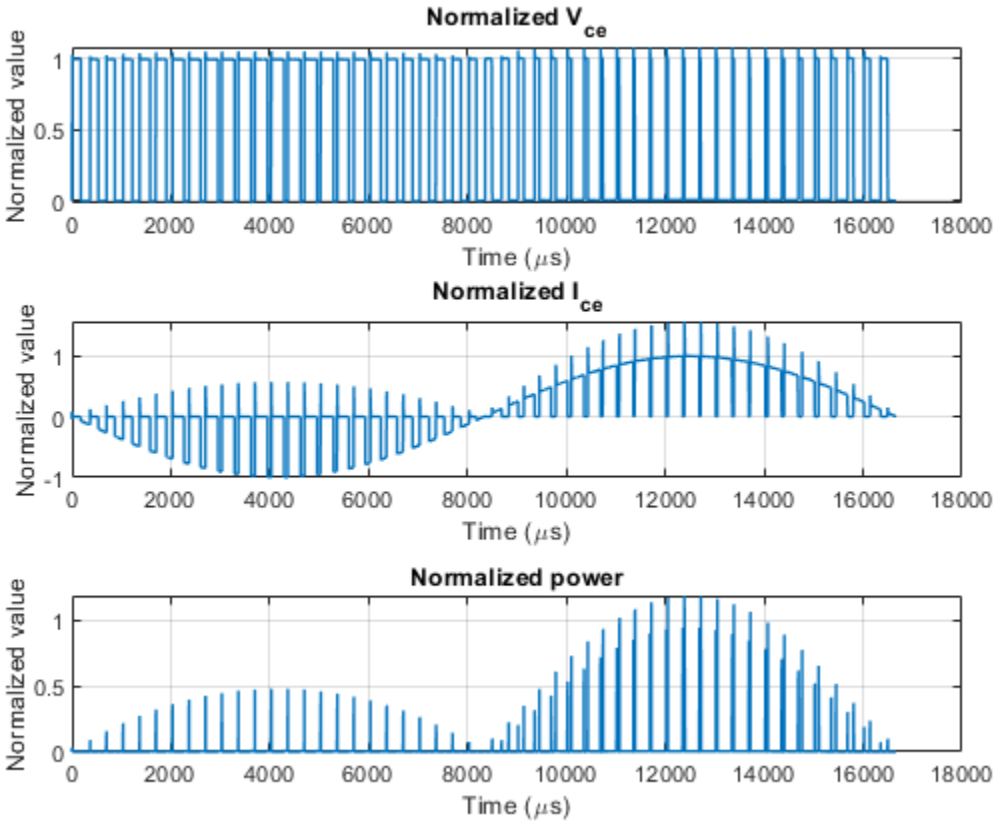

**Simulated Loss for Different Rg**

This plot shows the simulated loss for different Rg.



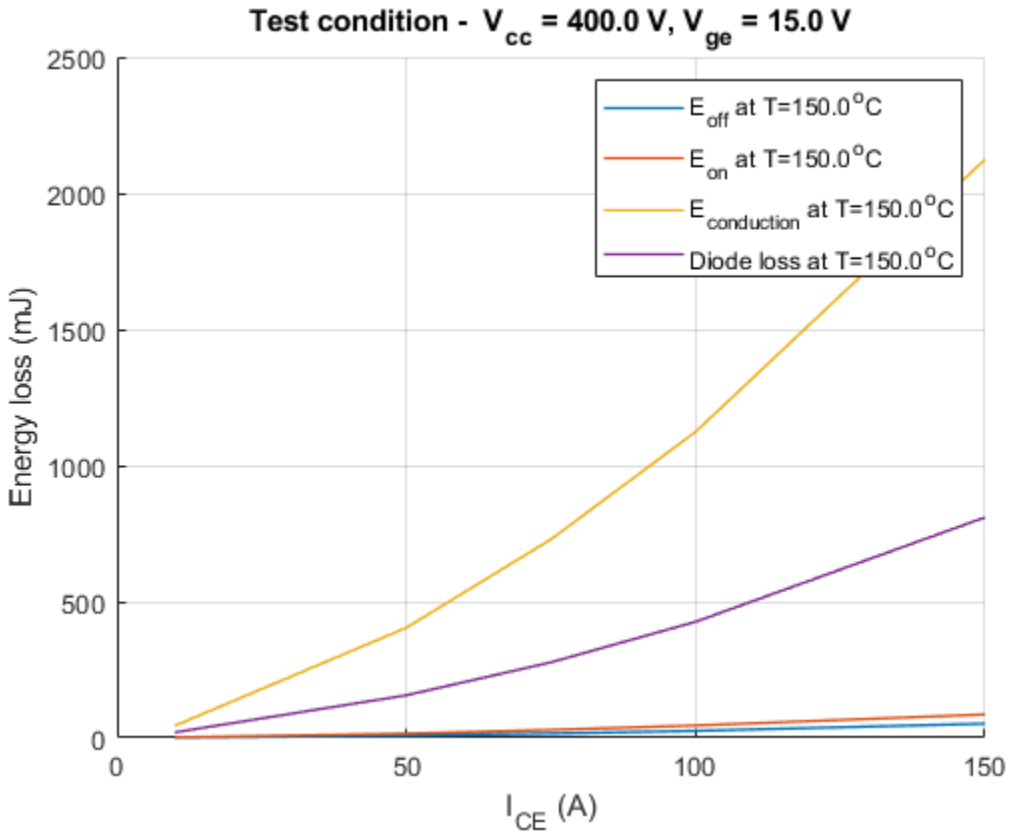

**Simulated Switching Loss Characteristics for Sinusoidal Load Current and Modulation Signal**

**Simulated Output for Sinusoidal Load Current** This plot shows the simulated output for the sinusoidal load current when $f_{pwm} = 3000Hz$.



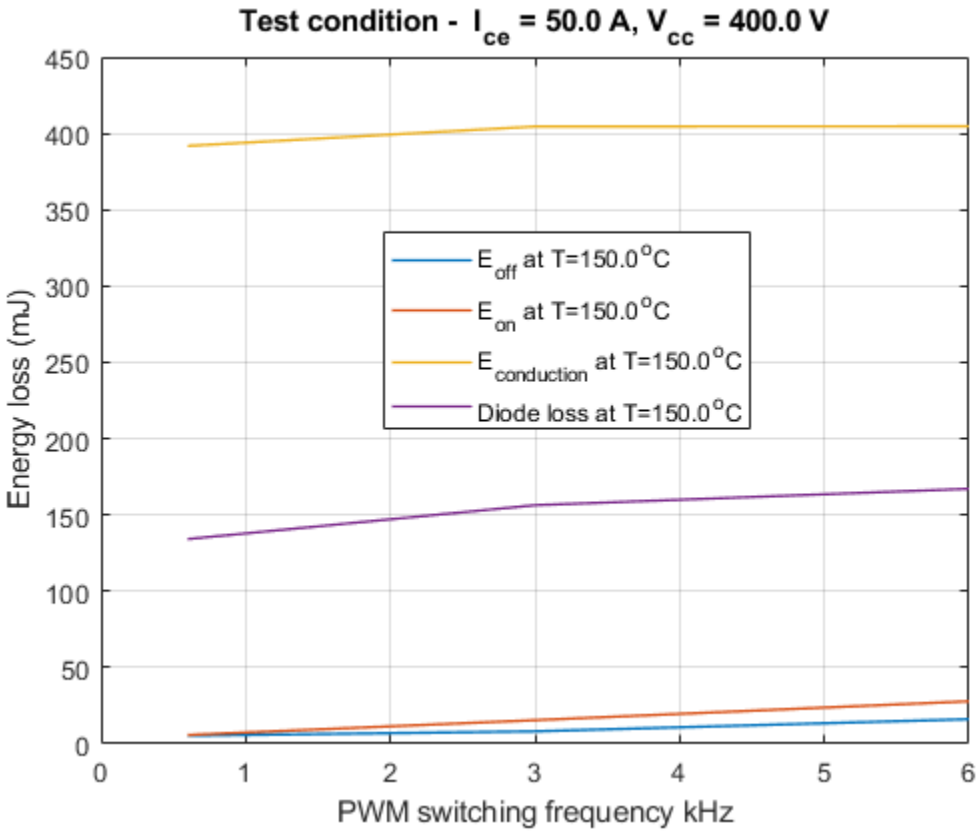

**Simulated Loss for Different Ice**

This plot shows the simulated loss for different Ice when $f_{pwm} = 3000Hz$.




**Simulated Loss for Different PWM Frequency**

This plot shows the simulated loss for different PWM frequencies.


Test condition - $I_{ce}$ = 50.0 A, $V_{cc}$ = 400.0 V
Energy loss (mJ)
PWM switching frequency kHz
$E_{off}$ at T=150.0°C
$E_{on}$ at T=150.0°C
$E_{conduction}$ at T=150.0°C
Diode loss at T=150.0°C
0
50
100
150
200
250
300
350
400
450
0
1
2
3
4
5
6

# Test Harness to Generate IV Characteristics of N-Channel IGBT

This example provides test harness for estimating current-voltage characteristics of a N-Channel IGBT.

For more information on designing and analyzing a three-phase converter, refer examples,

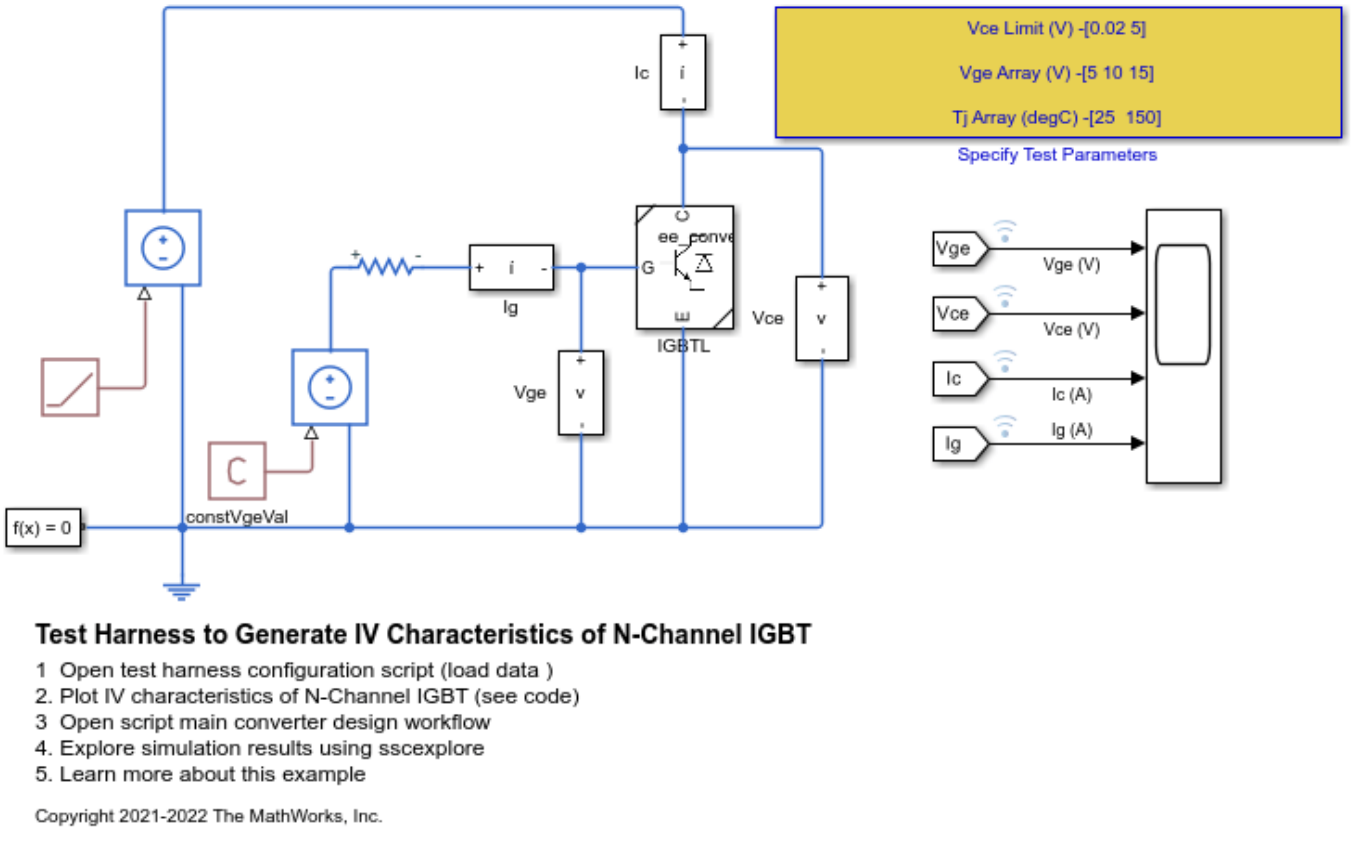
1. `ee_converter_design_igbt_switchingloss_testharness`.
2. `ee_converter_design_igbt_thermal_analysis`.

To access the overall workflow, click on the hyperlink in the model canvas or, at the MATLAB® Command Window, type:

*edit ee_converter_design_igbt_workflow.mlx*

### Test Harness for IGBT Steady State Characteristics

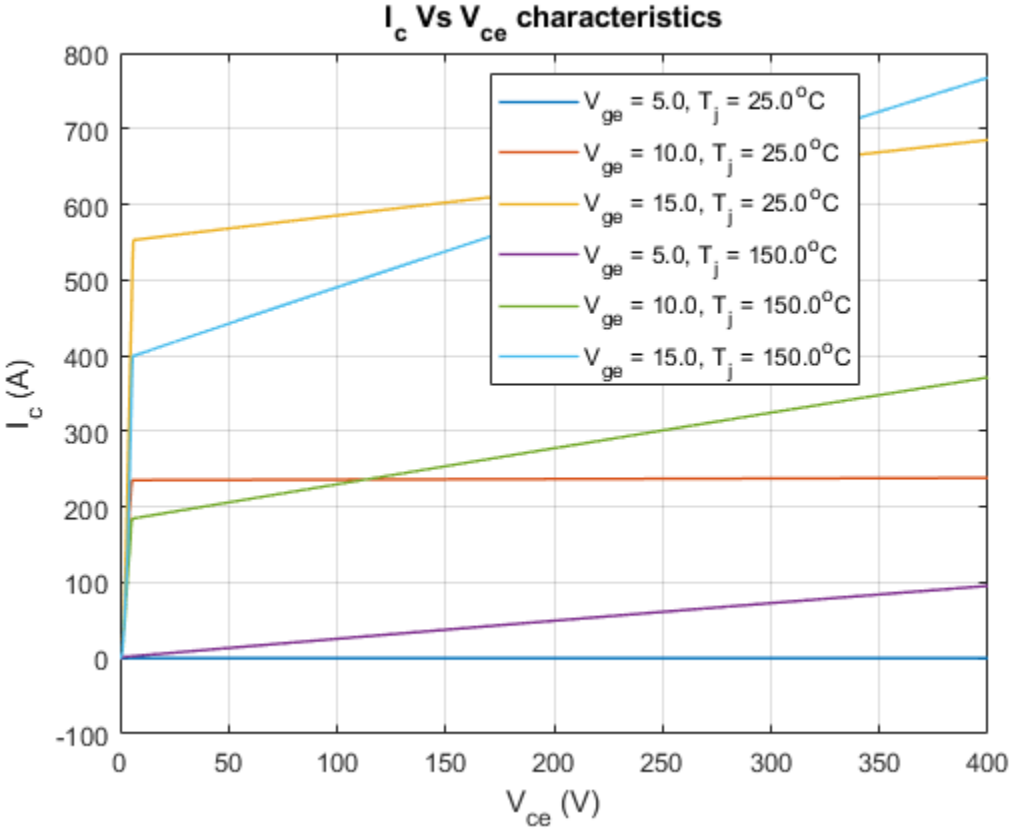
This model provides test harness to plot the steady state $(I_{ce}\ versus\ V_{ce})$ curve for different junction temperature $(T_j)$ and gate-emitter voltage $(V_{ge})$.




### IV Characteristics Simulation Results

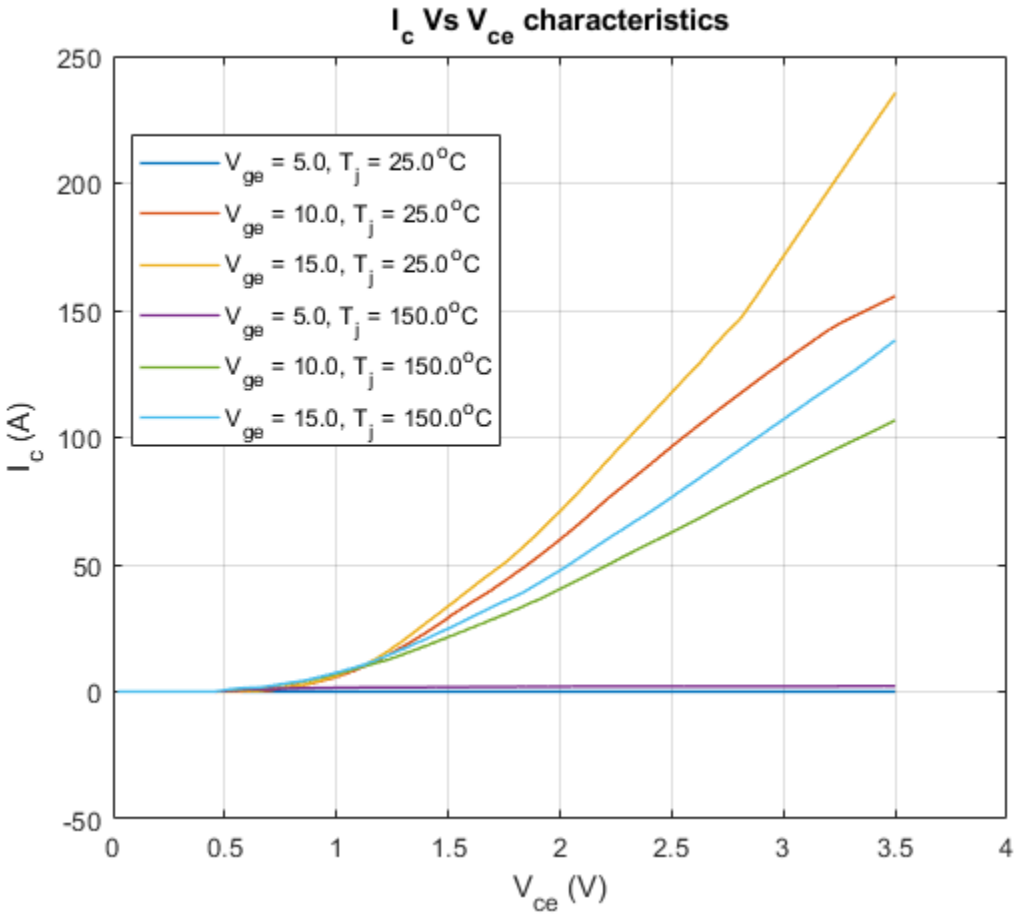
### IV Characteristics at Full range

This plot shows IGBT IV curve for $V_{ce}$ from 0 to $V_{dc}$




**IV Characteristics Near Vce(sat)**

This plot shows IV curve near IGBT on-state $V_{ce}$ voltage from 0 to $2V_{cesat}$


I_c Vs V_ce characteristics
250
200
150
100
50
0
-50
0
0.5
1
1.5
2
2.5
3
3.5
4
I_c (A)
V_ce (V)
V_ge = 5.0, T_j = 25.0°C
V_ge = 10.0, T_j = 25.0°C
V_ge = 15.0, T_j = 25.0°C
V_ge = 5.0, T_j = 150.0°C
V_ge = 10.0, T_j = 150.0°C
V_ge = 15.0, T_j = 150.0°C

# Manage Model Fidelity Using Variants

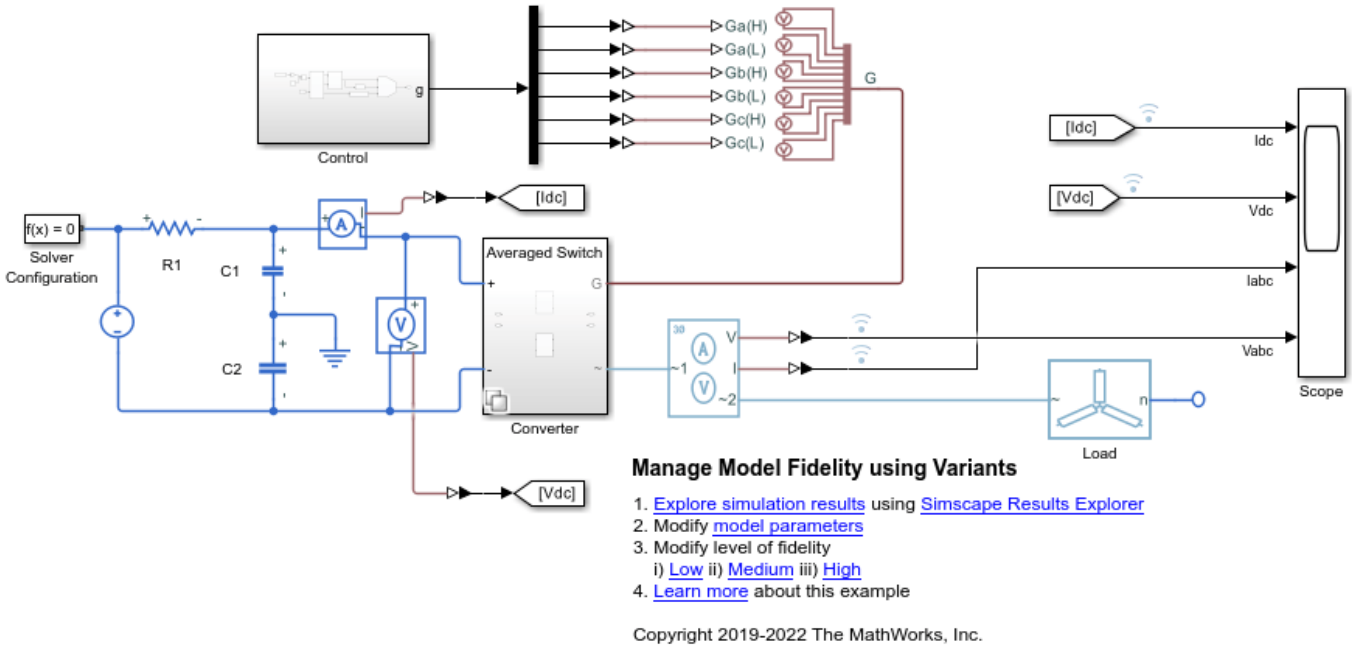
This example shows how to compare and contrast modeling different fidelity levels by using variants. The controller model uses a Variant Source block configured in Expression mode. The plant model uses a Variant Subsystem.

To permit easy interaction, this example demonstrates the commands you must type to configure the model from the command line.

### Define Model

Specify name of model, control subsystem, and scope.

### Open Model



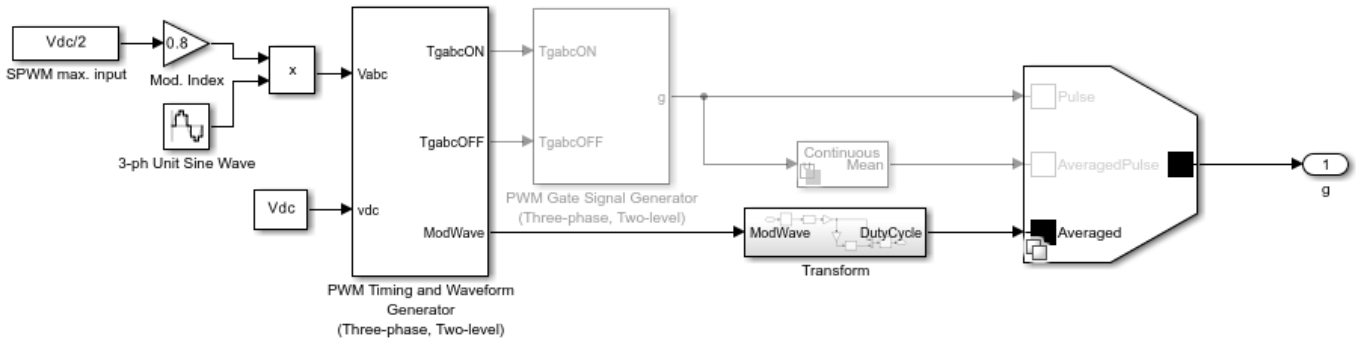

### Controller Variants

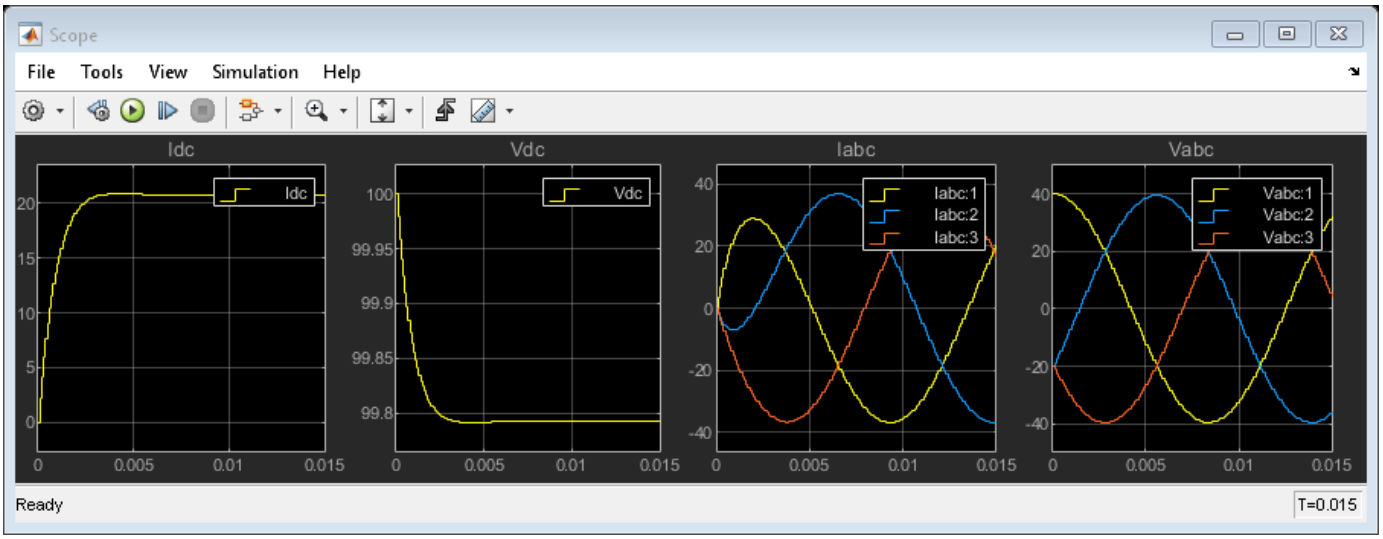
Create variant controls for converter controller. Controller has three different levels of fidelity: Averaged, AveragedPulse, Pulse

### Plant Variants

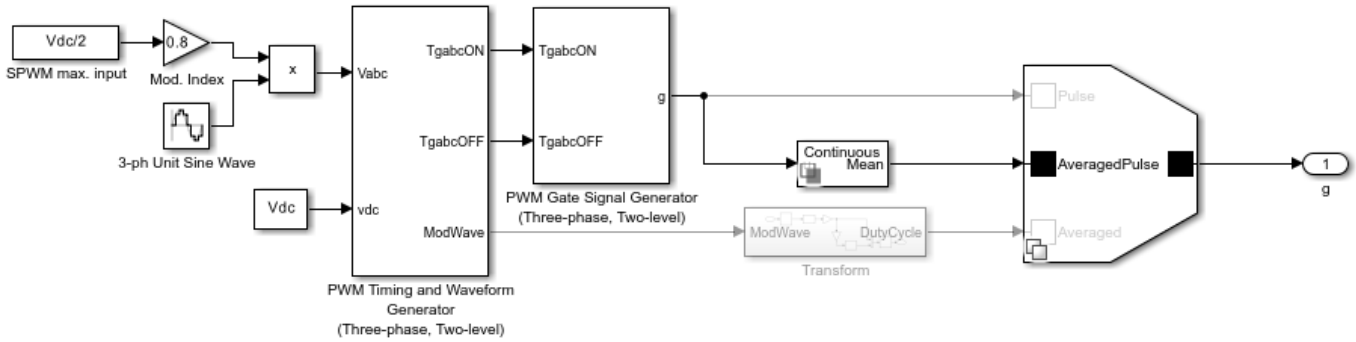
Create variant controls for converter plant. Plant has two different levels of fidelity.

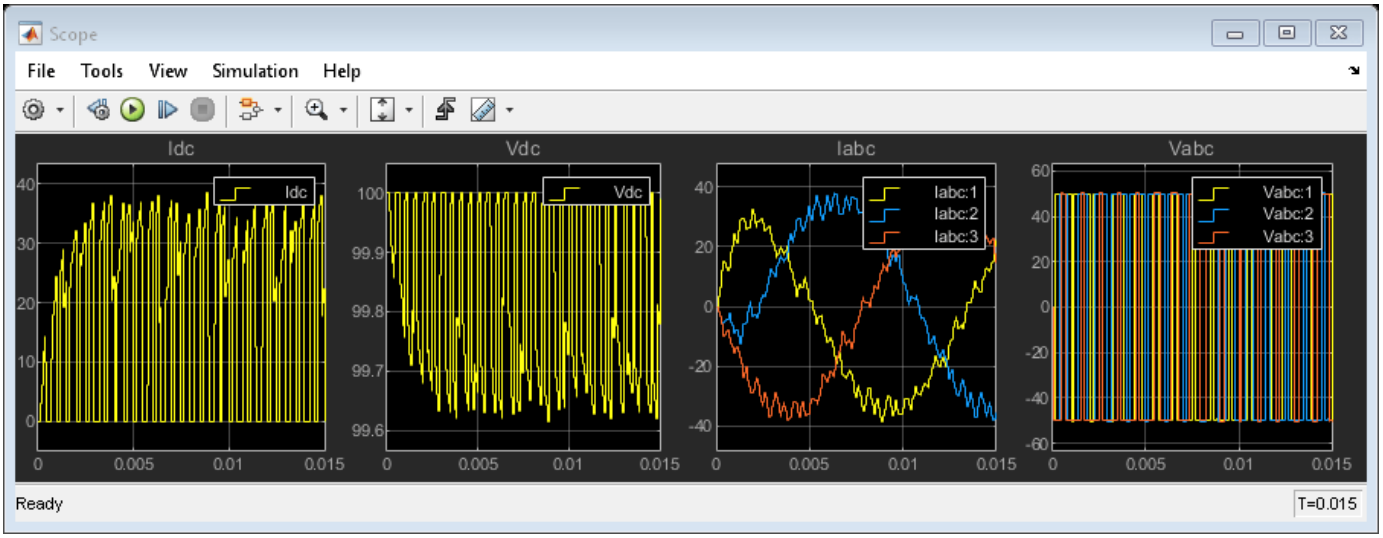
### Averaged Control Output, Averaged Switch Converter



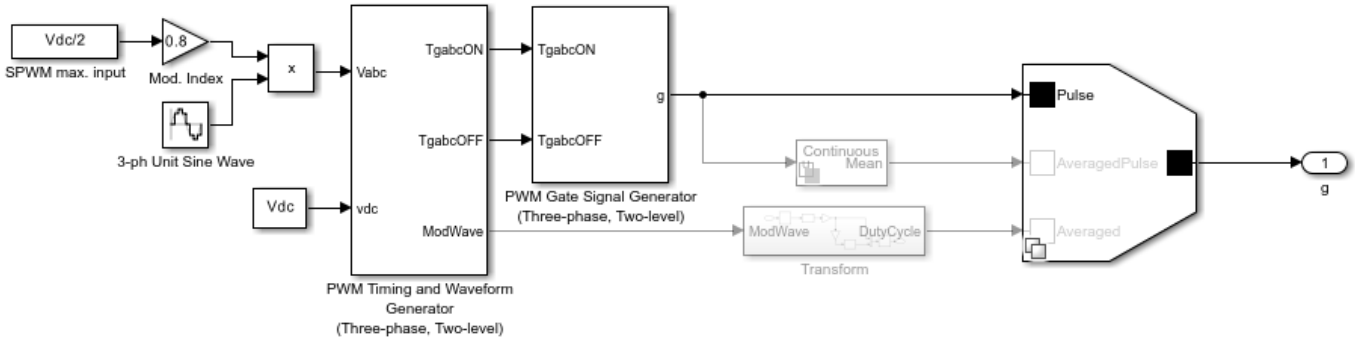



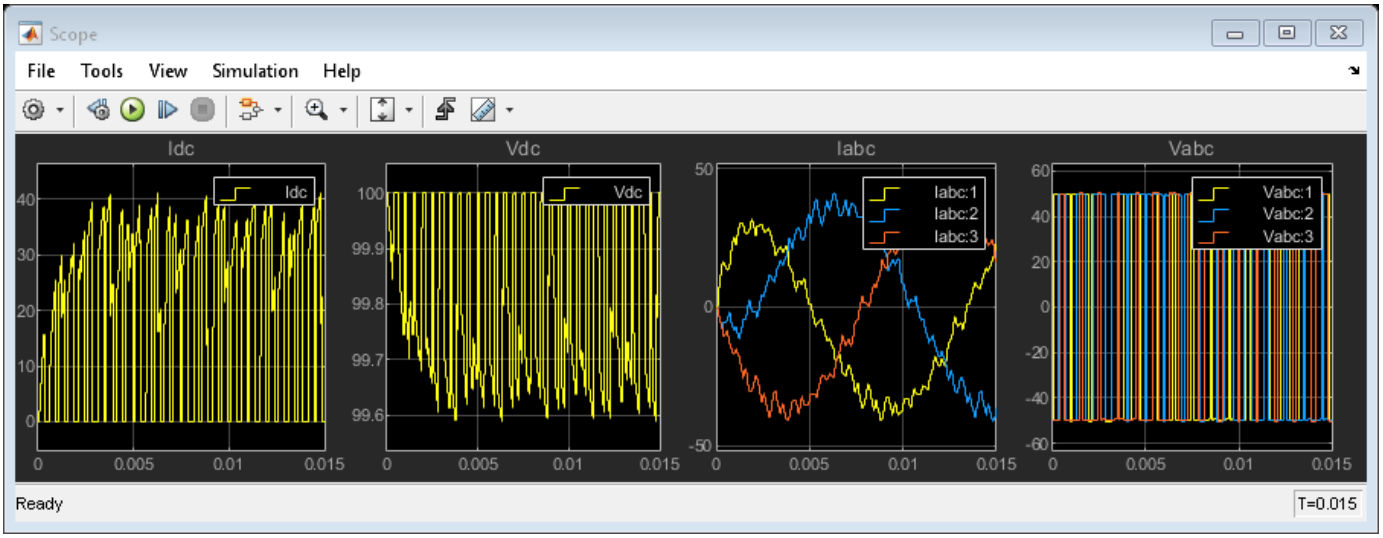

### Averaged Pulse Control Output, Averaged Switch Converter






**Pulse Control Output, Ideal Switch Converter**



### Results from Real-Time Simulation

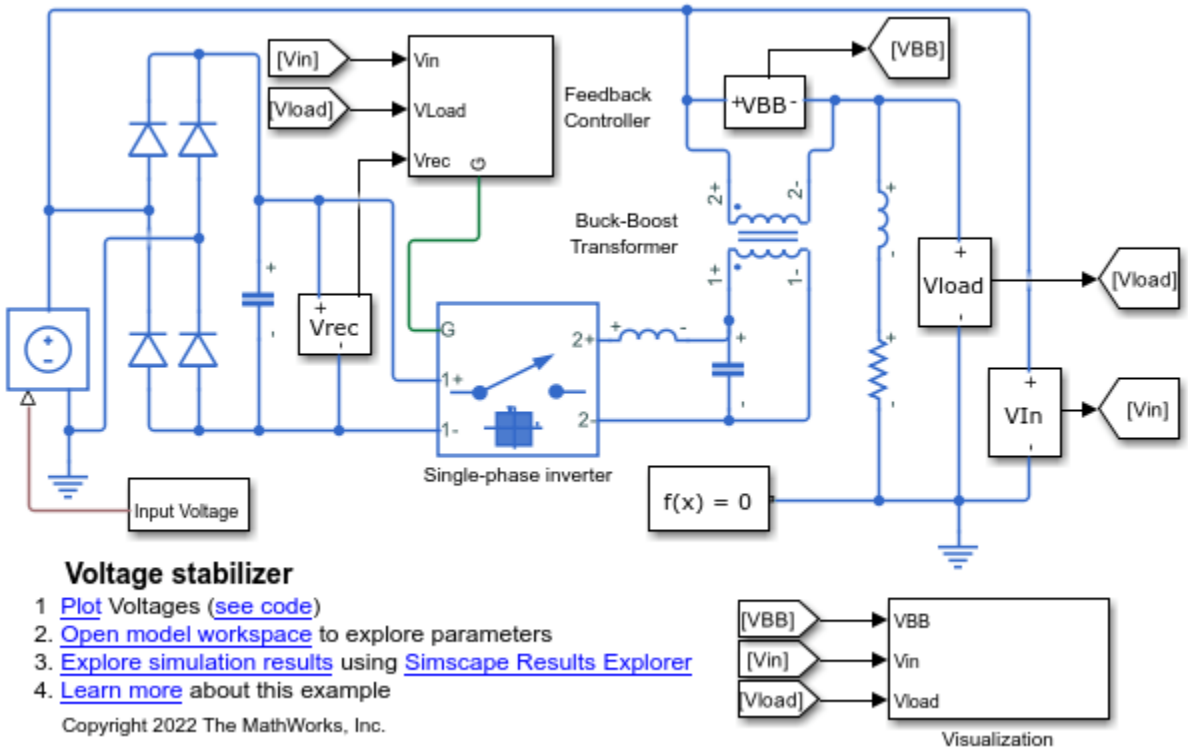
This example, with level of fidelity set to Low and Medium, has been tested on a Speedgoat Performance real-time target machine with an Intel® 3.5 GHz i7 multi-core CPU. This model can run in real time with a step size of 50 microseconds.

# Power Converter Voltage Stabilizer

This example shows a voltage stabilizer circuit. It uses a full-wave rectifier, a single-wave inverter, and a buck-boost transformer to achieve voltage regulation.

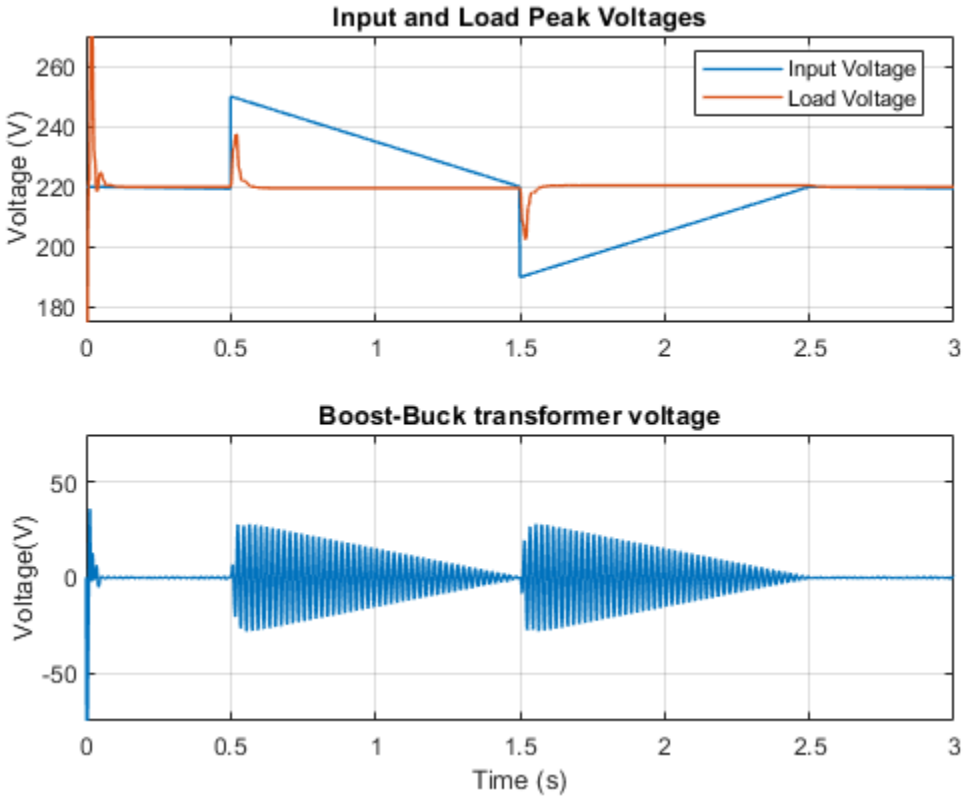
If the load sensor detects a voltage disparity, the feedback controller drives the four-quadrant chopper as a single-phase inverter and generates a voltage equal to the difference between the nominal value and the measured one. This output is either in phase or 180 degrees out of phase with the incoming voltage and is supplied to the primary winding of the transformer. The induced voltage in the secondary winding of the buck-boost transformer is added to the line, supplying the load with a constant voltage.

### Model Overview




### Simulation Results

The plot below shows the peak values of the input and load voltage. It also shows the voltage induced in the secondary winding of the buck-boost converter.




### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 50 microseconds.

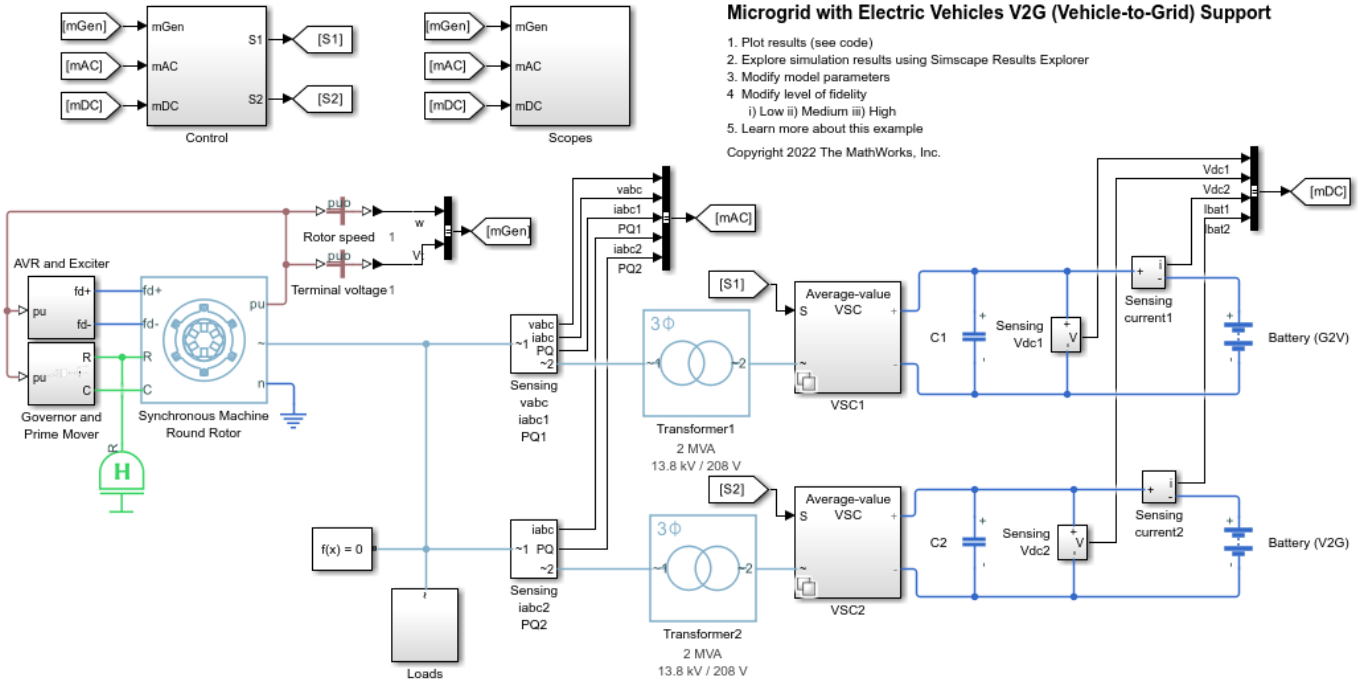
# Microgrid with Electric Vehicles V2G (Vehicle-to-Grid) Support

This example shows how to model a microgrid and how to regulate its frequency by using vehicle-to-grid (V2G) support from electric vehicles (EVs).

### Model Overview

The microgrid comprises one synchronous generator and two groups of electric vehicle batteries. Each battery block represents 20 EVs plugged into the grid. The range of grid frequency is set between 0.999 and 1.001 per unit. In this example, the Battery (G2V) block models EVs with a state-of-charge less than 80%. These EVs are always charged. The Battery (V2G) block models EVs with a SOC higher than 90% that provides V2G support during periods of peak demand in the grid.

Initially, two batteries are charged and obtain power from the grid. Their SOCs increase from an initial value of 70% and 98%, respectively. At time 10 s, the connection to the grid of a 1 MW load causes the grid frequency to drop below 0.999 per unit. The Battery (V2G) block starts to discharge and supply power to the grid. As a result, the grid frequency increases to the pre-set range (0.999 ~ 1.001 per unit). At time 30 s, the disconnection from the grid of a 2 MW load causes the grid frequency to rise above 1.001 per unit. The Battery (V2G) block goes back to charge status and the grid frequency decreases to the pre-set range. The Battery (G2V) block is being charged during the simulation since its SOC is less than 80%.




### Model Different Levels of Details Using Variant Controls

The `ee_microgrid_electric_vehicles_v2g_data` M file creates variant controls for voltage source converter and converter controller. Specifically, the file defines these Simulink.Variant objects:

- `AverageValueVSC` and `SwitchingVSC`, used with the average-value and switching voltage source converter.

- `ControllerTimeMode` and `ControllerFrequencyAndTimeMode`, used with the time and frequency-and-time simulation mode.

The file also defines a `fidelity` variable that specifies the level of fidelity.

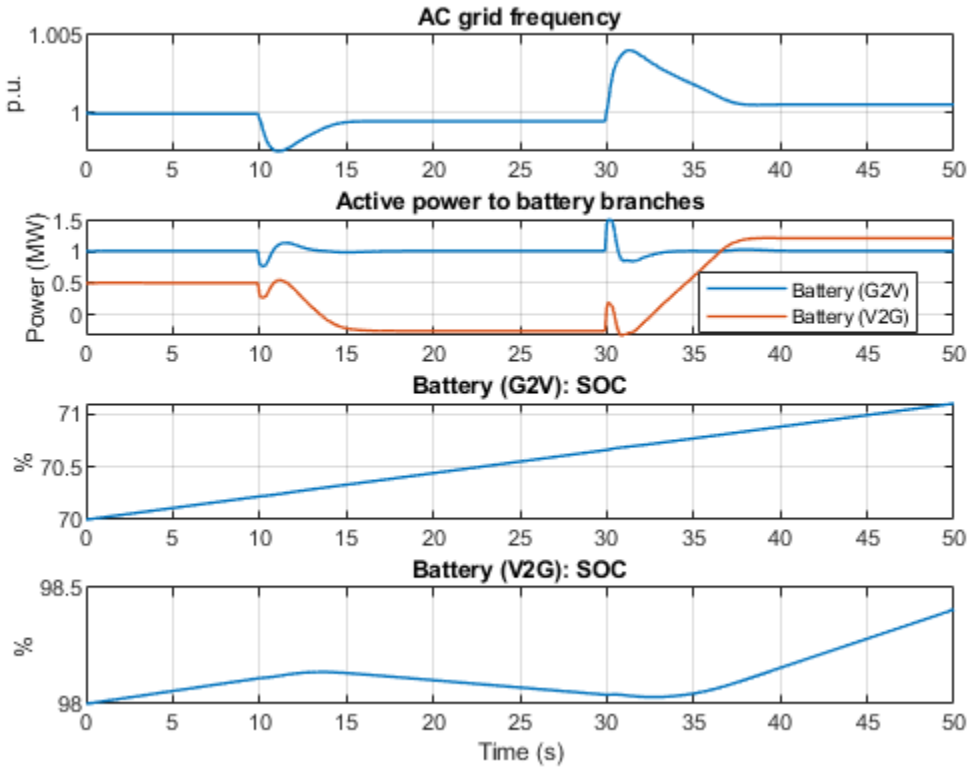
In this example, there are three different levels of fidelity:

- Low — The example runs in frequency-and-time simulation mode. An average-value voltage source converter models the converter. The sample time is 0.1 s.
- Medium — The example runs in time simulation mode. An average-value voltage source converter models the converter. The sample time is 200 us.
- High — The example runs in time simulation mode. Ideal power electronic devices model the converter The sample time is 2 us.

You can change the level of fidelity in the `ee_microgrid_electric_vehicles_v2g_variant_control` M file. The sample time and simulation mode are configured accordingly.

**Simulation Results from Simscape Logging**

The plot below shows the AC grid frequency, the active power that flows to two battery branches, and the state-of-charge of batteries in low level of fidelity



### Results from Real-Time Simulation

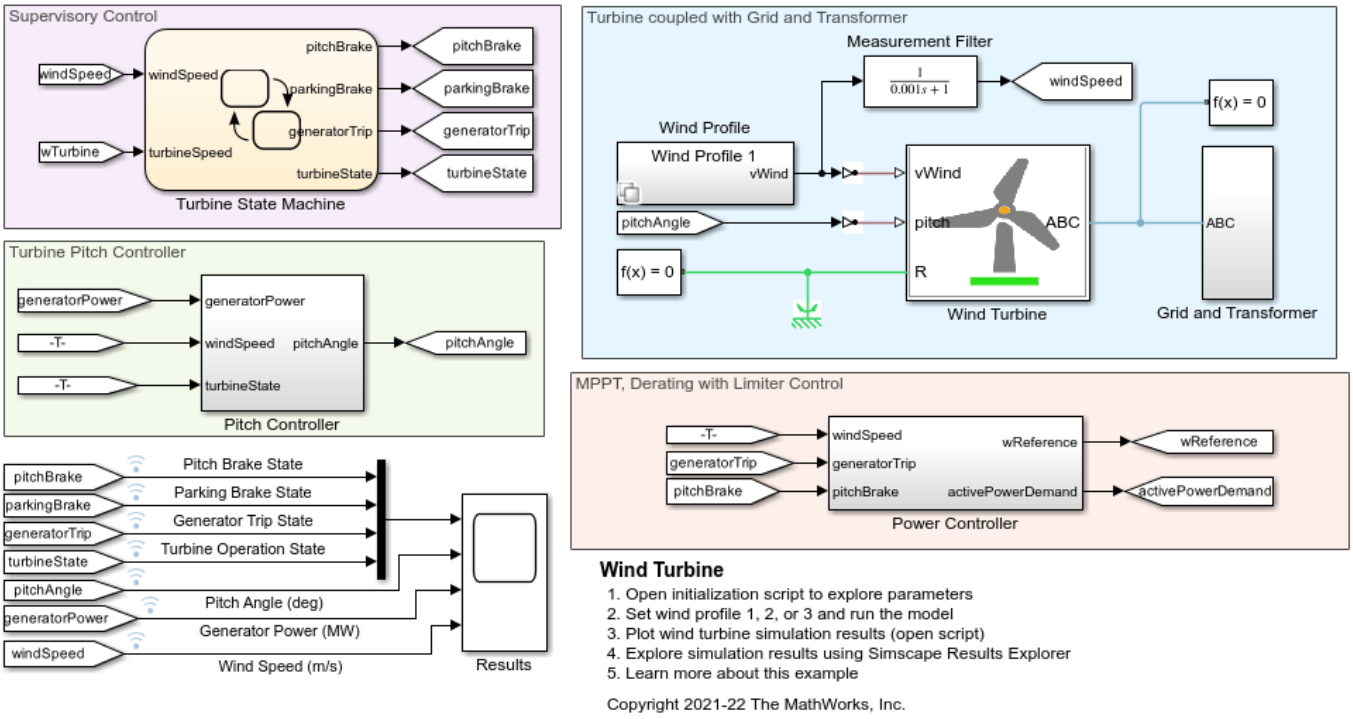
This example, with level of fidelity set to Low, has been tested on a Speedgoat Performance real-time target machine with an Intel® 3.5 GHz i7 multi-core CPU. This model can run in real time with a step size of 3 milliseconds.

# Wind Turbine

This example shows how to model, parameterize, and test a wind turbine with a supervisory, pitch angle, MPPT (maximum power point tracking), and derating control. When you run the plot function, it generates a plot of the state transitions, normalized physical quantities such as the wind speed, wind turbine rotation speed, generator power, and pitch angle.

### Model

The following figure shows the model of a wind turbine.The mechanical and electrical domains each require their own Solver Configuration block.




### Supervisory Control Subsystem

This subsystem demonstrates how to model the wind turbine state machine. The turbine state machine defines four wind turbine states.

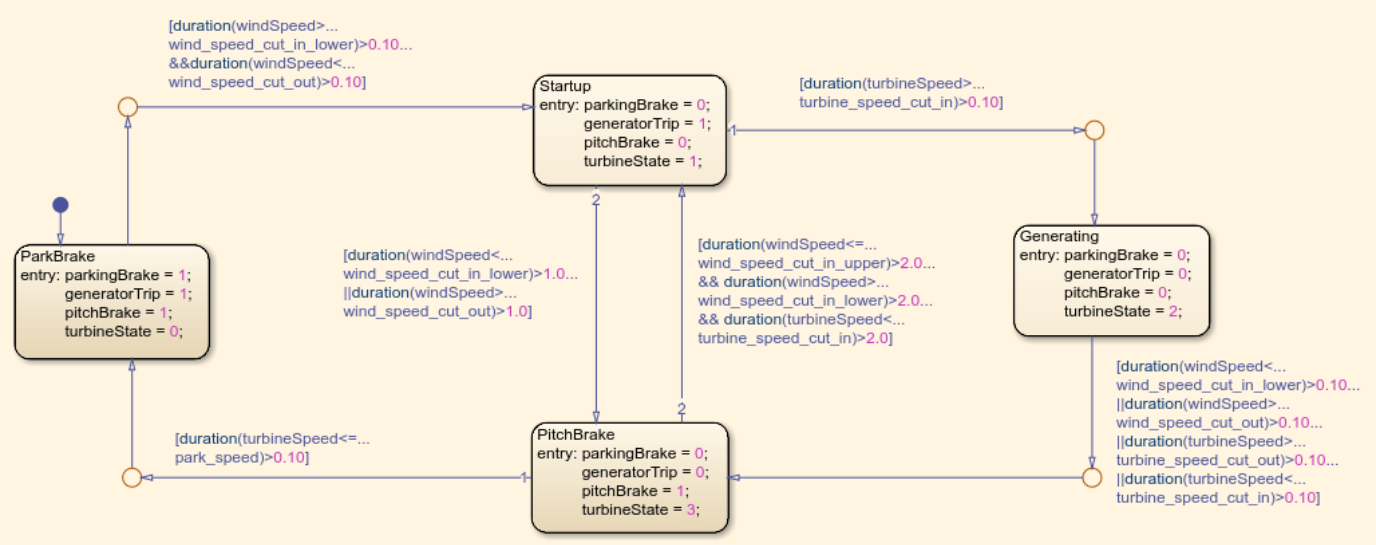
*Park brake mode*: This is the entry mode of the wind turbine operation. The wind turbine enters the park brake mode from the pitch brake mode when the turbine rotor speed is under the permissible limits for safe operation. During this mode, the generator is in the tripped state, the hydraulic park brake is engaged, and the wind turbine rotor blades are pitched to the braking angle for the aerodynamic braking. The hydraulic brake is the secondary method for braking the wind turbine.

*Startup mode*: The wind turbine enters the startup mode from the park brake mode when the wind speed is under the permissible limits for safe operation. The wind turbine enters this mode from the pitch brake mode when the wind speed and the turbine speed are under the permissible limits. During this mode, the generator is in the tripped state, the hydraulic park brake is released, and the

wind turbine rotor blades are pitched to the minimum angle for achieving the maximum turbine rotor acceleration.

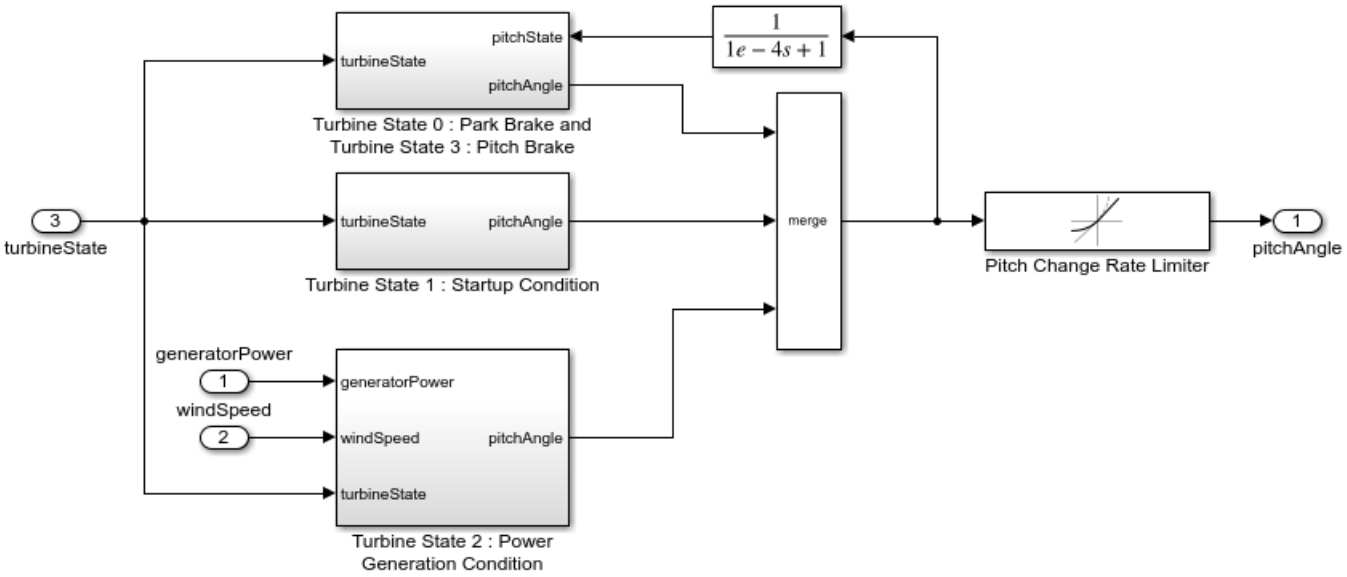
*Generating mode*: The wind turbine enters the generating mode from the startup mode when the wind turbine rotor speed goes above the turbine cut in speed. During this mode, the generator is connected to the transformer, the hydraulic park brake is released, and the wind turbine rotor blades are pitched to achieve the optimal electric power generation as per the operating conditions.

*Pitch brake mode*: The wind turbine enters pitch brake mode from generating mode when the wind speed and turbine rotation speed are not under the permissible limits. The wind turbine enters the pitch brake mode from the startup mode when the wind speed is not under the permissible limits. During this mode, the generator is connected to the transformer to consume the kinetic energy available in the rotor blade, the hydraulic park brake is released, and the wind turbine rotor blades are pitched to the braking angle for the aerodynamic braking. The aerodynamic braking is the primary method for braking the wind turbine.

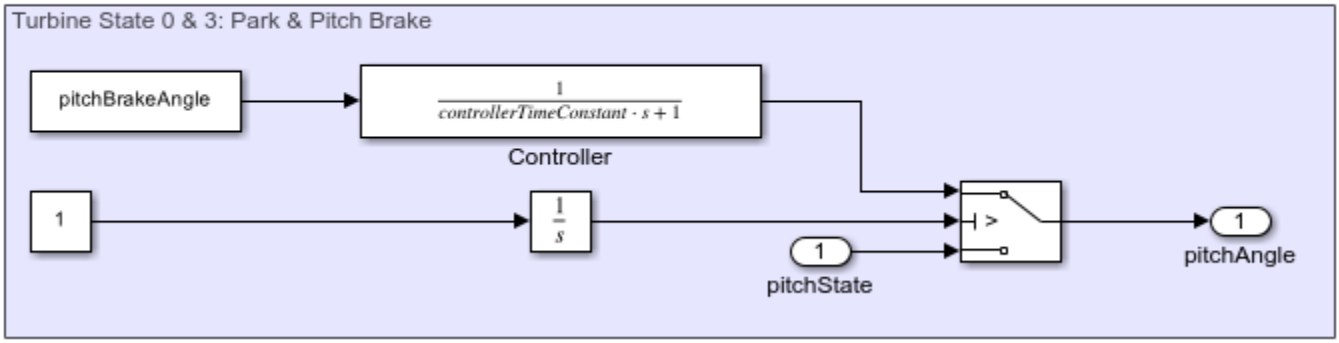



### Pitch controller Subsystem

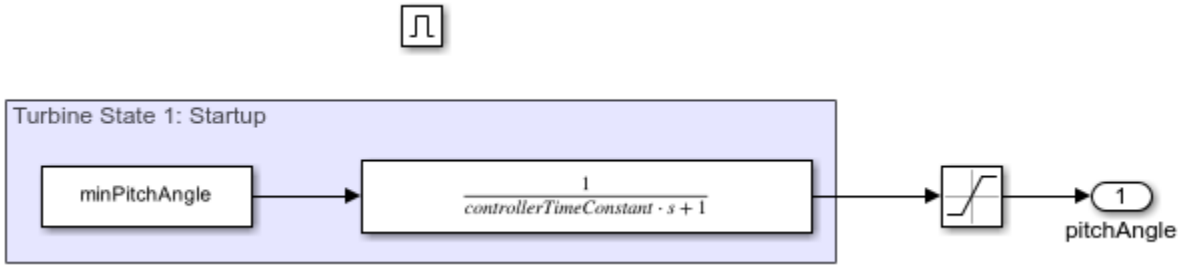
This subsystem demonstrates how to model the pitch angle controller of the wind turbine.




*Park brake and pitch brake mode*: The pitch angle is set to 95 degree for the aerodynamic braking during these modes.



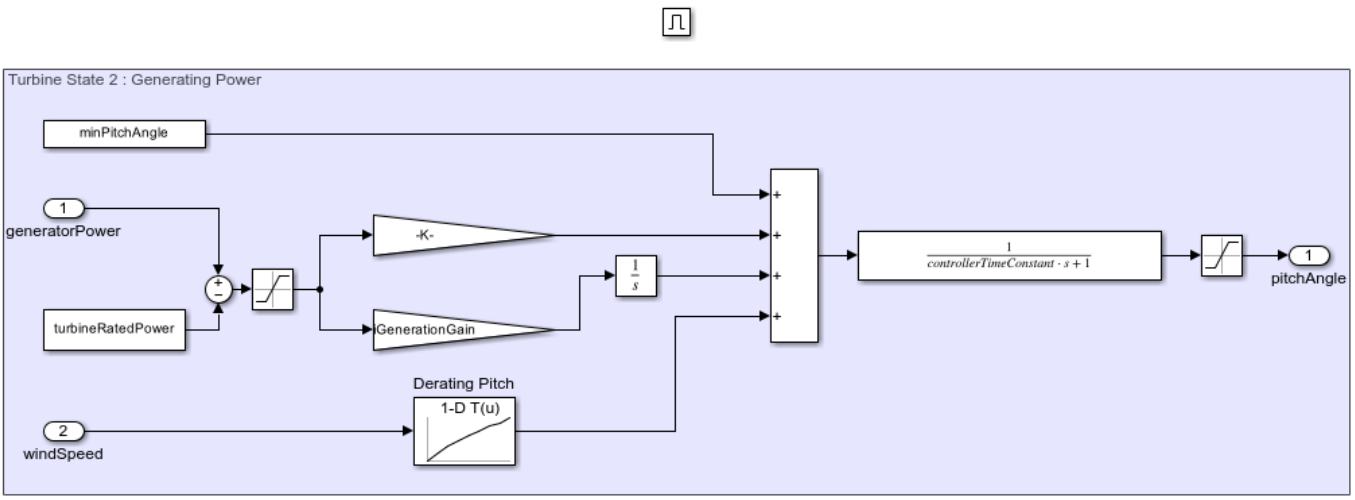

*Startup mode*: The pitch angle is set to 1 degree for achieving the maximum acceleration during this mode.



*Generating mode*: The pitch angle is set to achieve the optimal electric power generation as per the operating conditions.

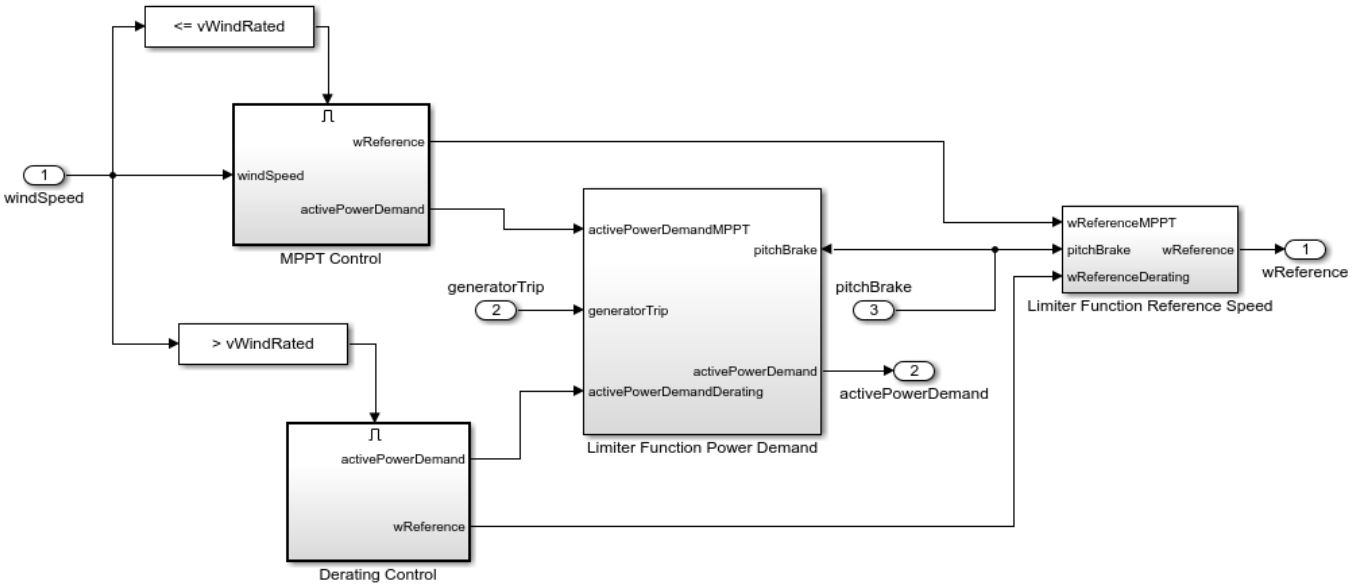
1. The pitch angle remains at the lowest setting at 1 degree up to rated wind speed. This control acts in synchronization with the wind turbine's MPPT power control.

2. The pitch angle changes when the wind speed is above the rated wind speed in synchronization with the wind turbine's derating power control.

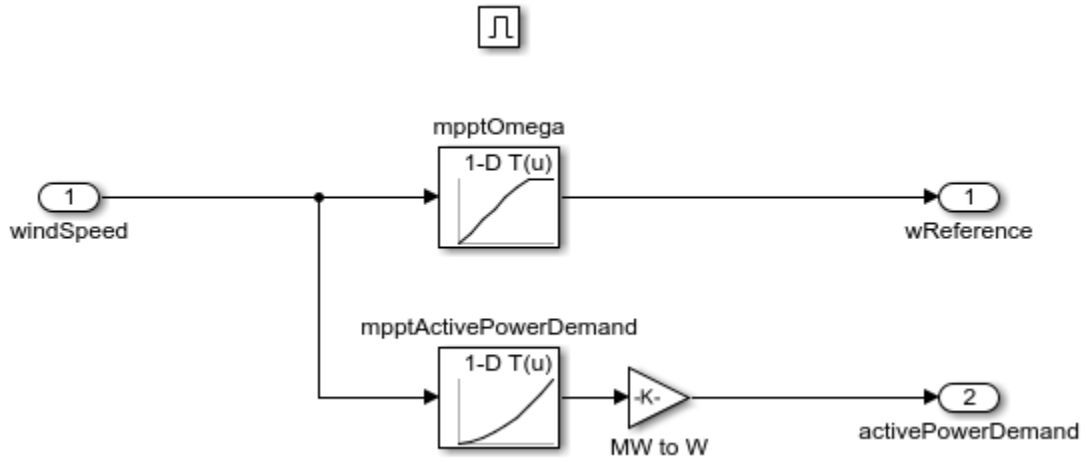



### Power Controller Subsystem

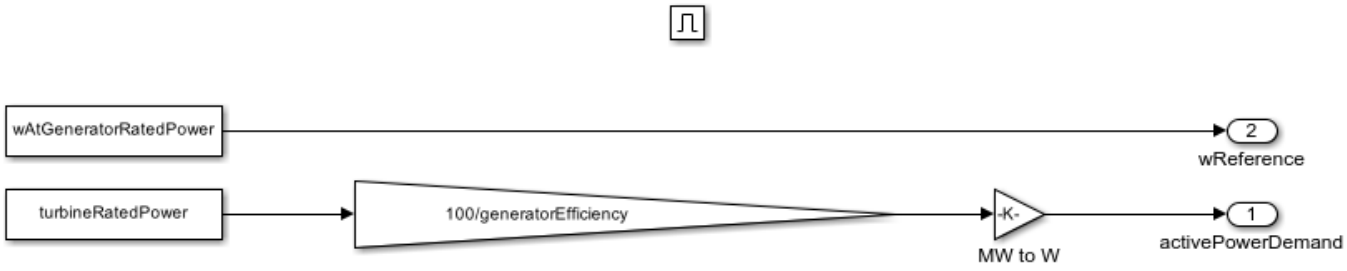
This subsystem demonstrates how to model the power demand and the generator input speed reference for the optimal torque loading on the wind turbine through the generator.



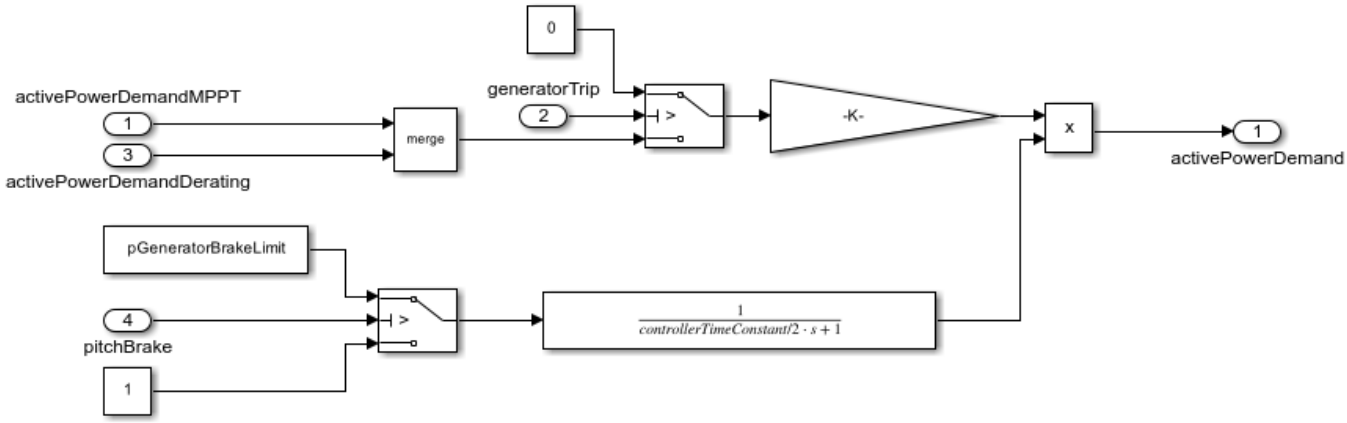

*MPPT control*: This mode is active up to the rated wind speed. The power demand and the generator reference speed is as per the wind turbine characteristic power performance curve.



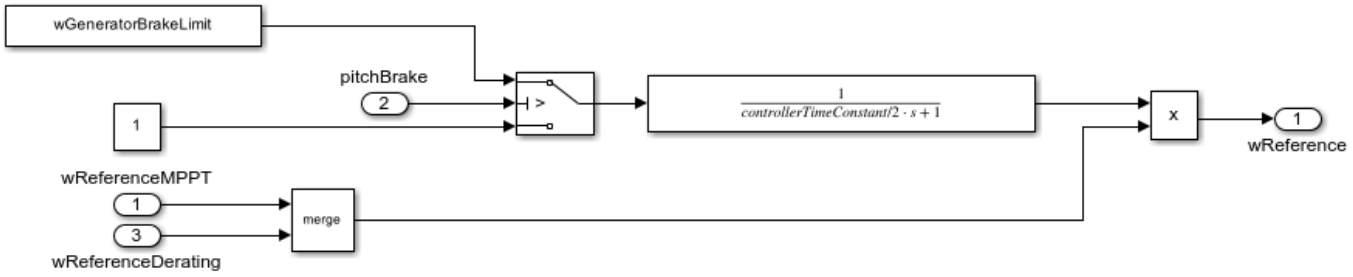

*Derating control*: This mode is active above the rated wind speed. The power demand and the generator reference speed is kept constant based on the rating of the wind turbine.




*Limiter Function Power Demand*: This function limits the power demand as per the wind turbine's mode of operation.

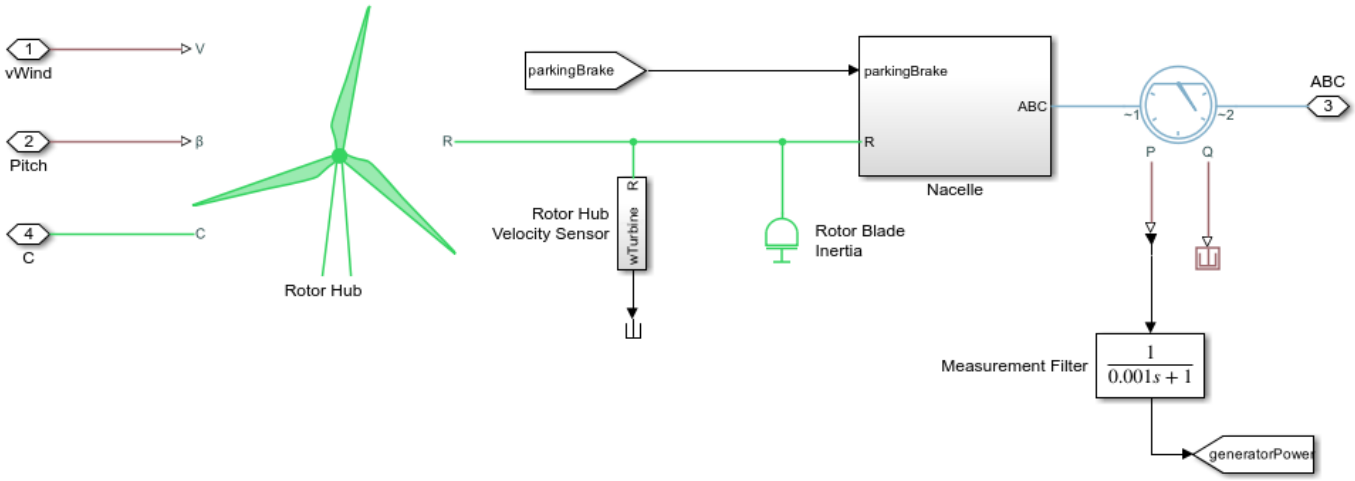



*Limiter Function Reference Speed*: This function limits the speed reference as per the wind turbine's mode of operation.




### Wind Turbine Subsystem

This subsystem demonstrates how to model the wind turbine.



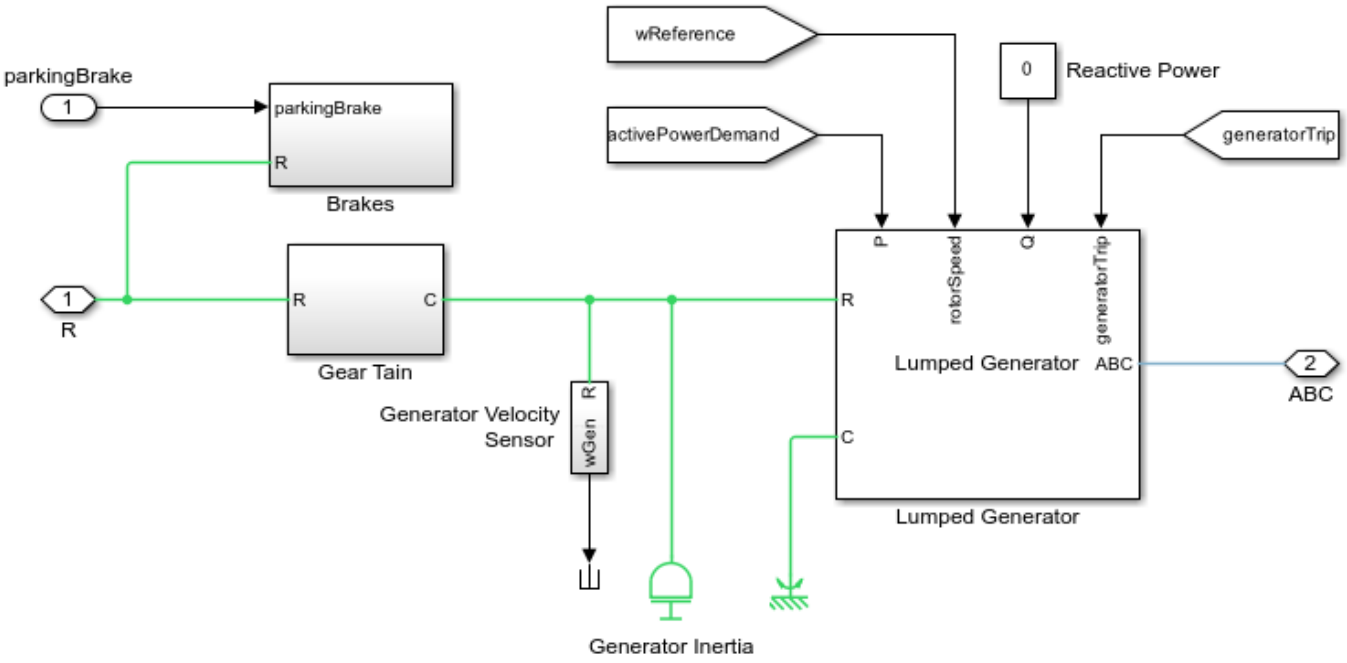

*Rotor Hub*: The rotor of a horizontal axis wind turbine is modeled using the Simscape™ Driveline™ Wind Turbine block. In this block, *mechanical power* extracted from the wind, P, is calculated as

$$P = 1/2 * rho * Ar * vWind^3 * cp(pitch, lambda)$$

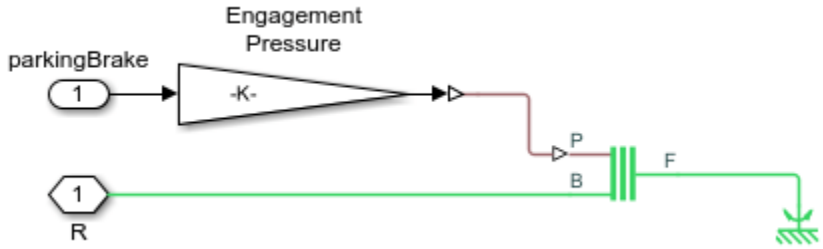
where,

1. *rho* is the air density
2. *Ar* is the area swept by the rotor blades
3. *vWind* is the wind speed
4. *cp* is the power coefficient as a function of lambda and pitch.
5. *lambda* is the ratio of the rotor blade tip speed and the wind speed, it is also called Tip Speed Ratio (TSR)
6. *pitch* is the rotor blade pitch angle

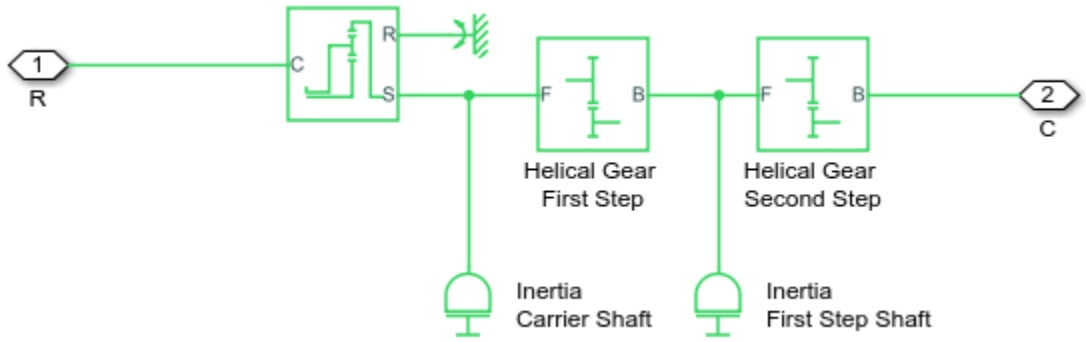
*Nacelle*: This subsystem demonstrates how to model the nacelle components of a wind turbine.



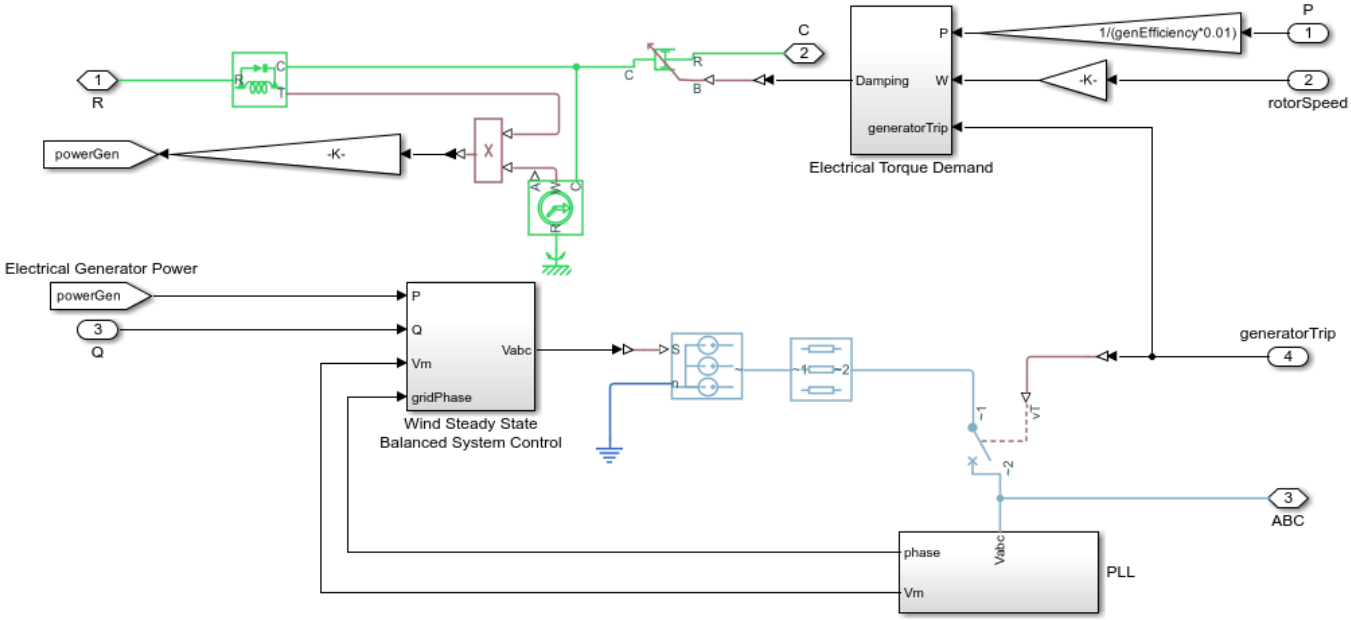

*Brakes*: This subsystem demonstrates how to model the brakes in the nacelle. The hydraulics brake is a secondary braking method in the wind turbine. The brakes are engaged when the wind turbine speed goes below the parking brake speed while the wind turbine is under the park brake mode or during the wind turbine maintenance operation. If they are applied above the parking brake speed, the brake can burn out or the nacelle can catch fire due to the excessive frictional heat generation.




*Gear train*: This subsystem demonstrates how to model the gear train in the nacelle. An epicyclic gear train represents the gear train system with the power losses. The gear train contributes to high power loss in the wind turbine operation.




*Lumped generator*: This subsystem demonstrates how to model the generator in the nacelle. This is a simplified model of a generator to increase the simulation speed.

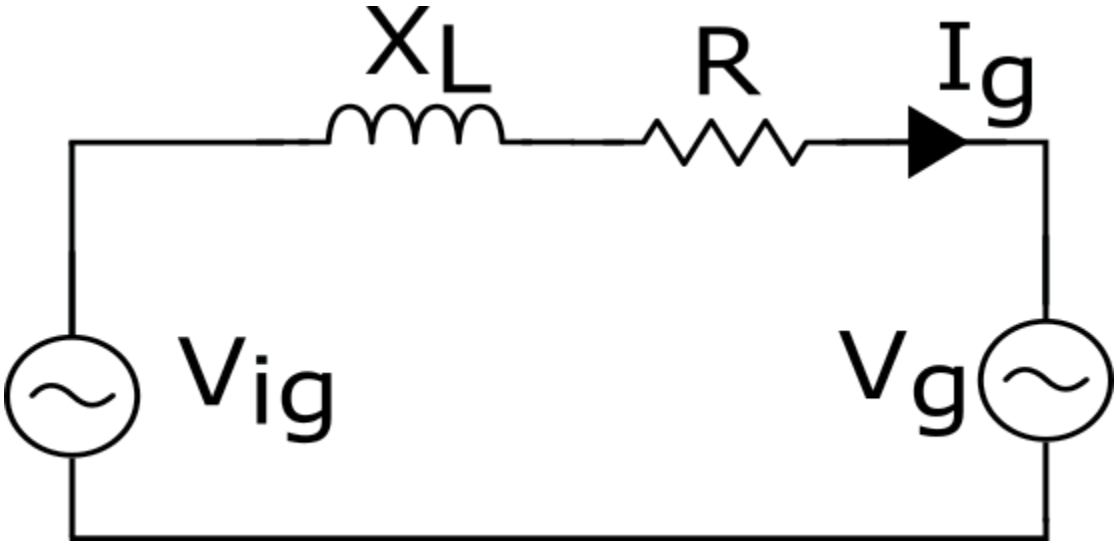



The wind turbine has a larger time constant and slower response than a traditional doubly-fed induction generator (DFIG) system. To simulate wind turbine control, you must run the simulation longer.

The closed-loop DFIG system is faster than wind turbine control systems such as pitch control. Therefore, a low fidelity lumped DFIG generator system is practical for improving simulation speed and providing flexibility. The lumped generator system integrates with the wind turbine system to simulate different aspects of the control algorithm.

The lumped generator model tracks the grid voltage and phase angle at the point of common coupling (PCC) using the phase locked loops (PLL). To improve the simulation speed, fast acting PLL is modelled using the grid voltage zero crossing detection. Lumped generator uses the grid voltage and phase obtained from the PLL and implements the power flow equation with the generator lumped leakage inductance and resistance. The lumped generator model takes the real power (P), reactive power (Q), and reference generator shaft speed as input.

In the wind turbine system, the lumped generator model gets the power reference and approximate speed reference input from the wind turbine power control system. Based on the reference input, the generator applies the load torque to the wind turbine shaft and supply the electrical power to the grid.




$$3V_g I_g^* = P + JQ$$

$$I_{gm}^* = \frac{2(P + JQ)}{3V_{gm}}$$

$$V_g = V_{gm} \angle 0^o$$

$$V_{ig} = V_{gm} + jX_L I_{gm} + RI_{gm} = V_{igm} \angle \phi$$

$$V_{iga}(t) = V_{igm} sin(\omega t + \phi)$$

$$V_{igb}(t) = V_{igm} sin(\omega t + \phi - 120^o)$$

$$V_{igc}(t) = V_{igm} sin(\omega t + \phi - 240^o)$$

where,

1. $V_g$ is grid rms phase voltage at PCC
2. $I_g$ is grid rms phase current at PCC
3. $V_{gm}$ is grid peak phase voltage at PCC
4. $I_{gm}$ is grid peak phase current at PCC
5. $V_{igm}$ is electrical generator peak induced phase voltage
6. $V_{ig}$ is electrical generator induced phase voltage
7. $X_L$ is lumped leakage reactance of induction generator (wrt to stator)

8 $R$ is lumped resistance of induction generator (wrt to stator)

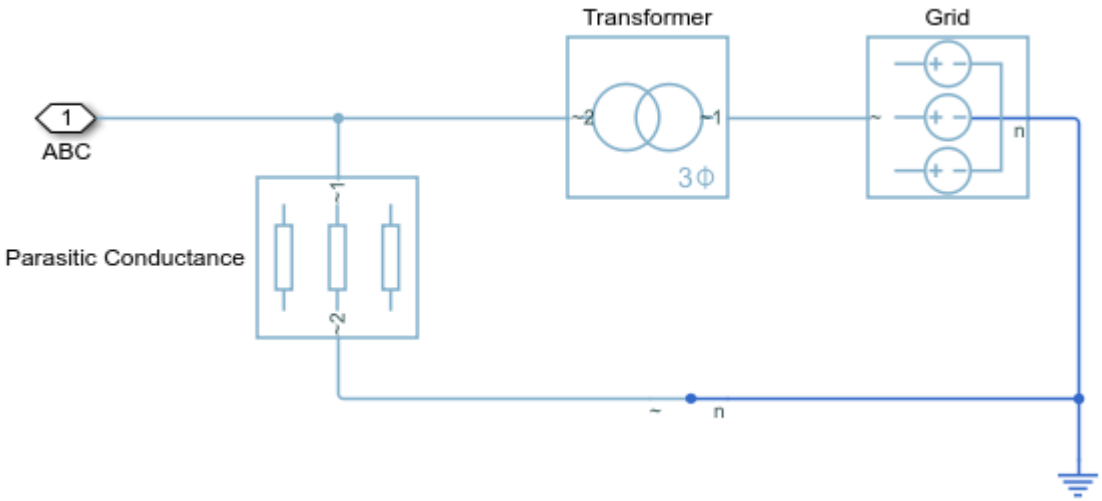
9 $\omega t$ is instantaneous grid voltage angle

Here, $V_{gm}$ and $\omega t$ are obtained from the PLL.

The lumped generator system model block should be used only in fixed frequency, full sinusoidal, and balanced three phase grid connected systems. This model is not suitable for the transient electrical stimulation such as the fault ride through and the grid frequency variation simulation.

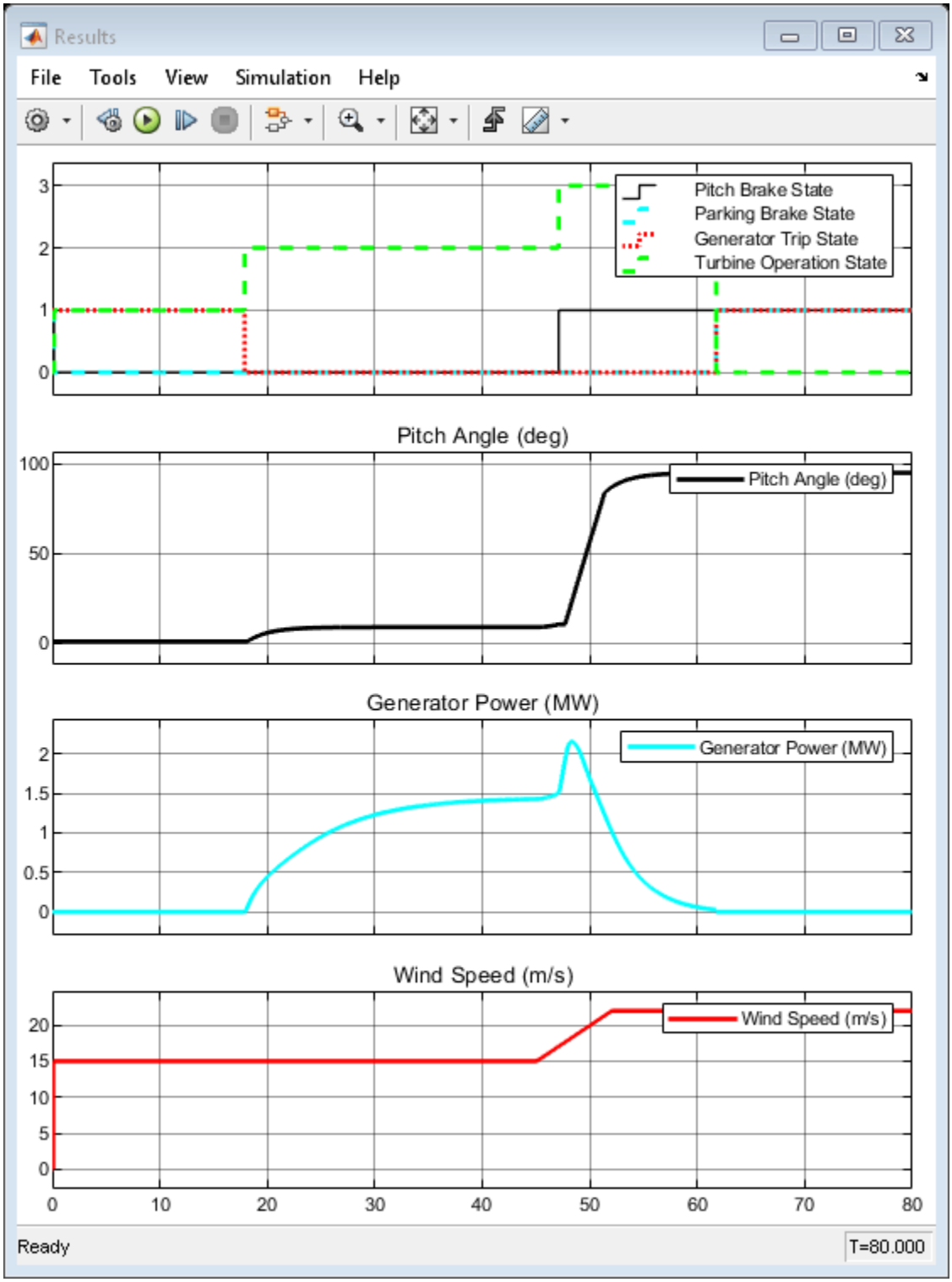
### Grid and Transformer Subsystem

This subsystem demonstrates how to model the grid and transformer. The parasitic conductance is added to increase the simulation speed.



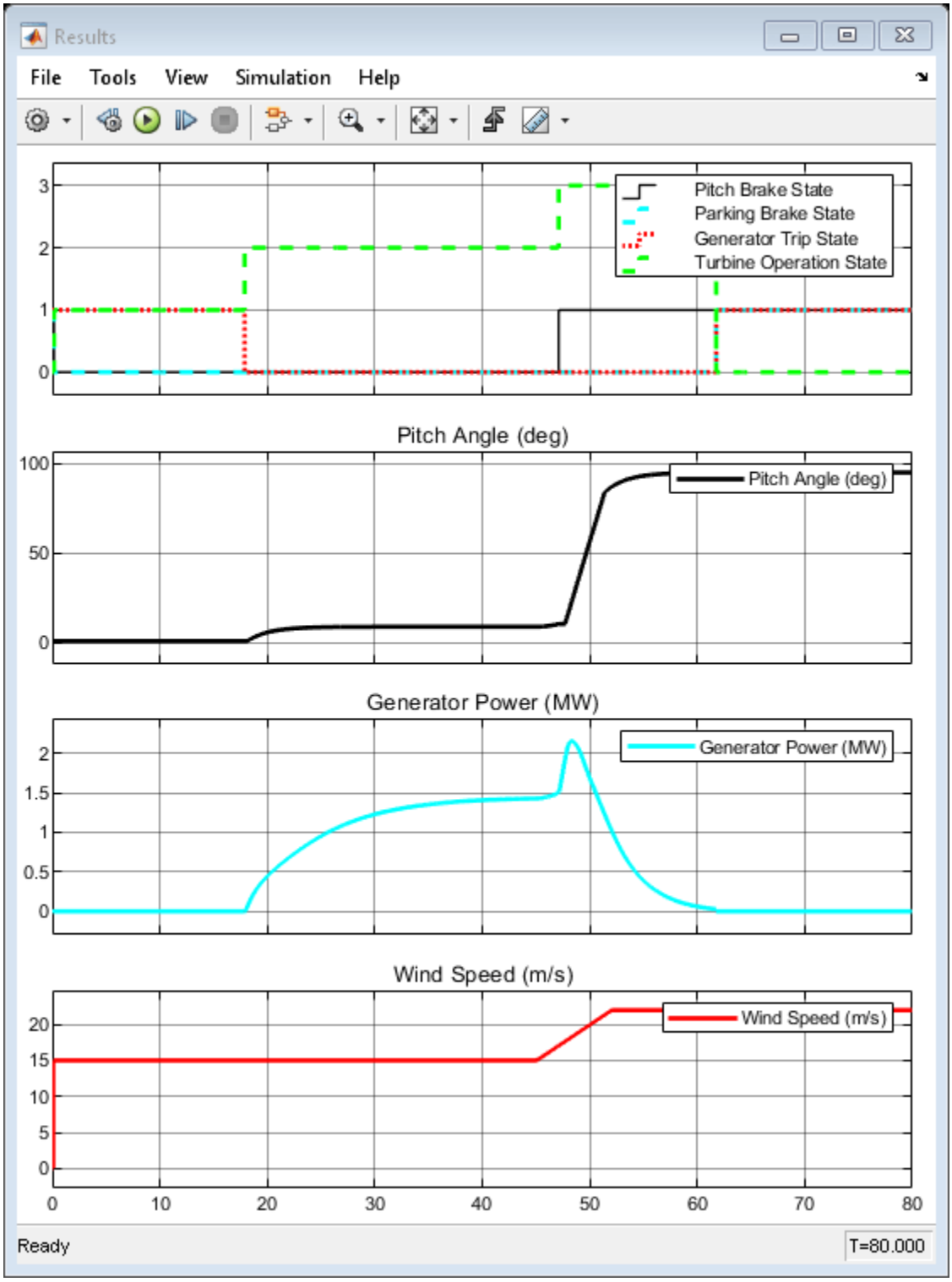

### Simulation Results from Scope

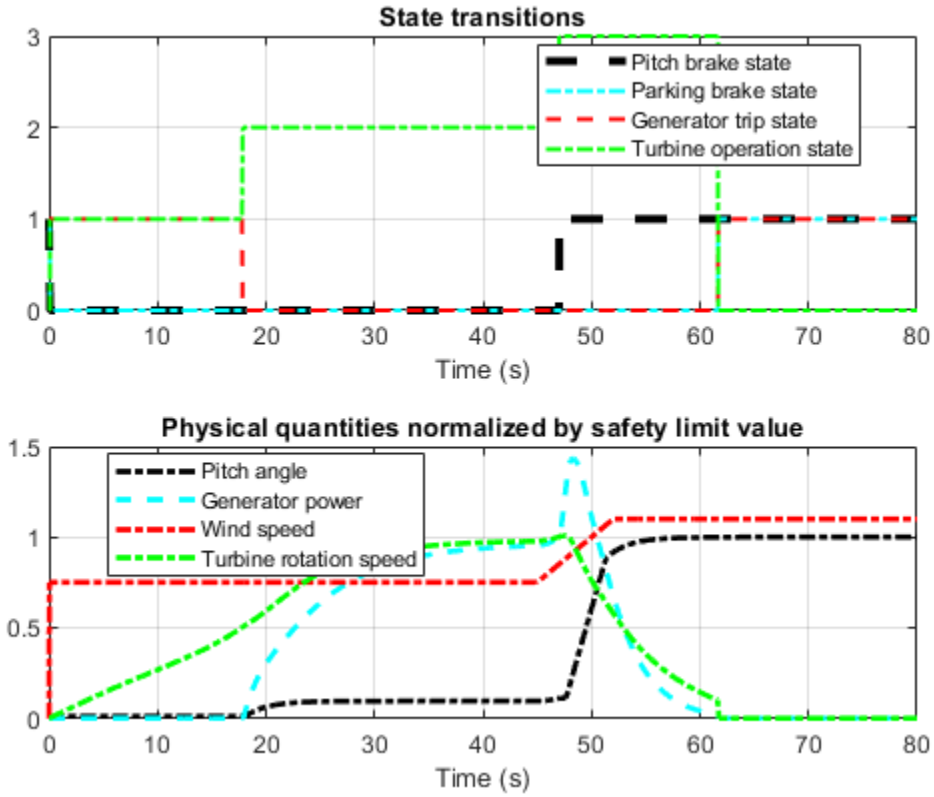
This figure shows the supervisory control states, turbine blade pitch angle, generator power, and wind speed.




**Simulation Results from Simscape Logging**

This model generates plot of the state transitions, the normalized physical quantities such as the wind speed, wind turbine rotation speed, generator power, and pitch angle.

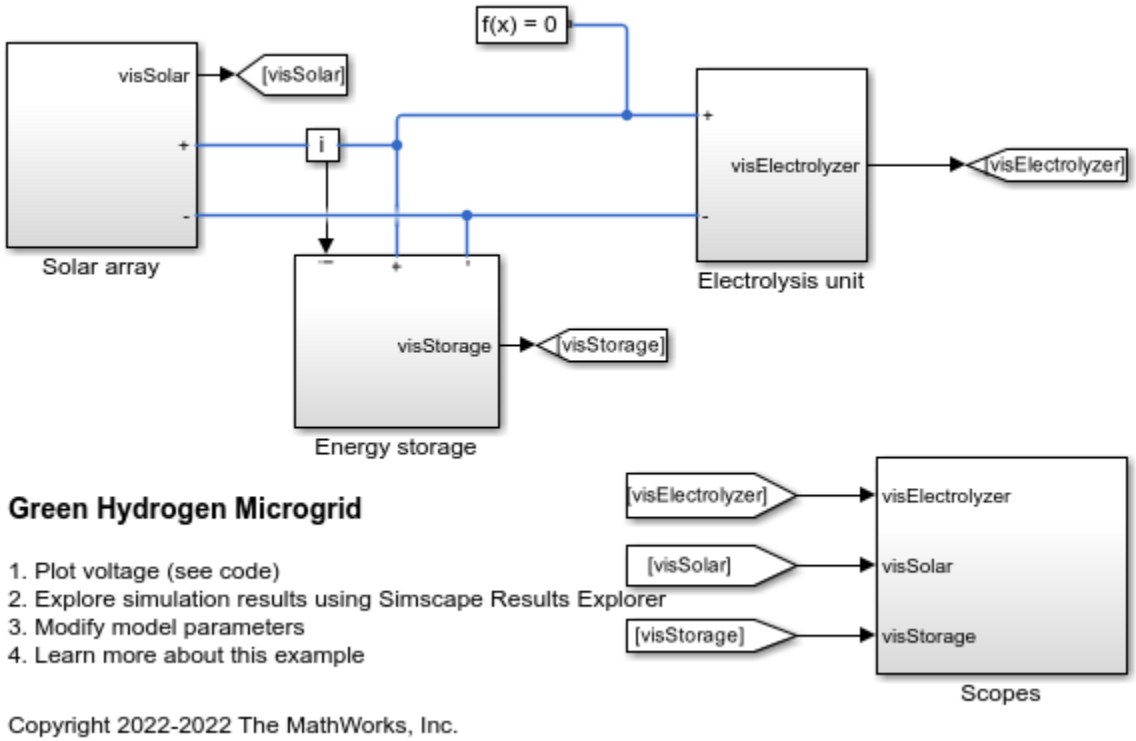

Results
File
Tools
View
Simulation
Help
Pitch Brake State
Parking Brake State
Generator Trip State
Turbine Operation State
Pitch Angle (deg)
Pitch Angle (deg)
Generator Power (MW)
Generator Power (MW)
Wind Speed (m/s)
Wind Speed (m/s)
Ready
T=80.000


State transitions
Pitch brake state
Parking brake state
Generator trip state
Turbine operation state
Time (s)
Physical quantities normalized by safety limit value
Pitch angle
Generator power
Wind speed
Turbine rotation speed
Time (s)

# Green Hydrogen Microgrid

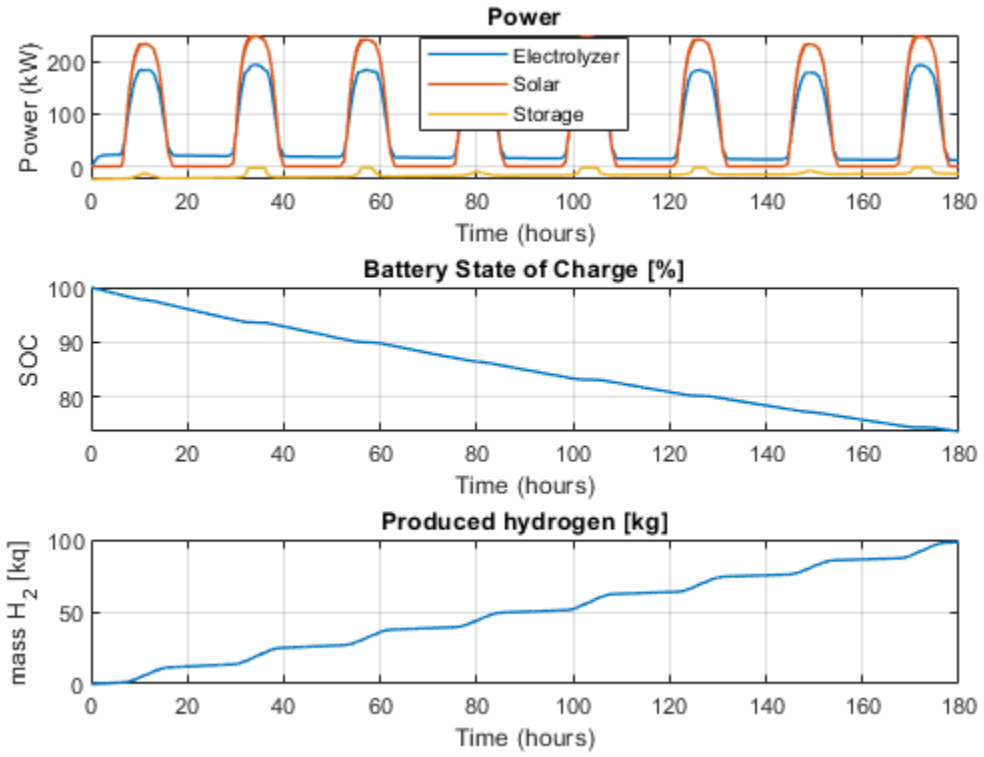
This example shows a DC islanded microgrid that provides power to an electrolyzer using a solar array and an energy storage system. You can use this model to evaluate the operational characteristics of producing green hydrogen over a 7-day period by power from a solar array, or from a combination of a solar array and an energy storage system. The model includes electrical, thermal liquid, and thermal gas domains.

### Model




### Simulation Results from Simscape Logging

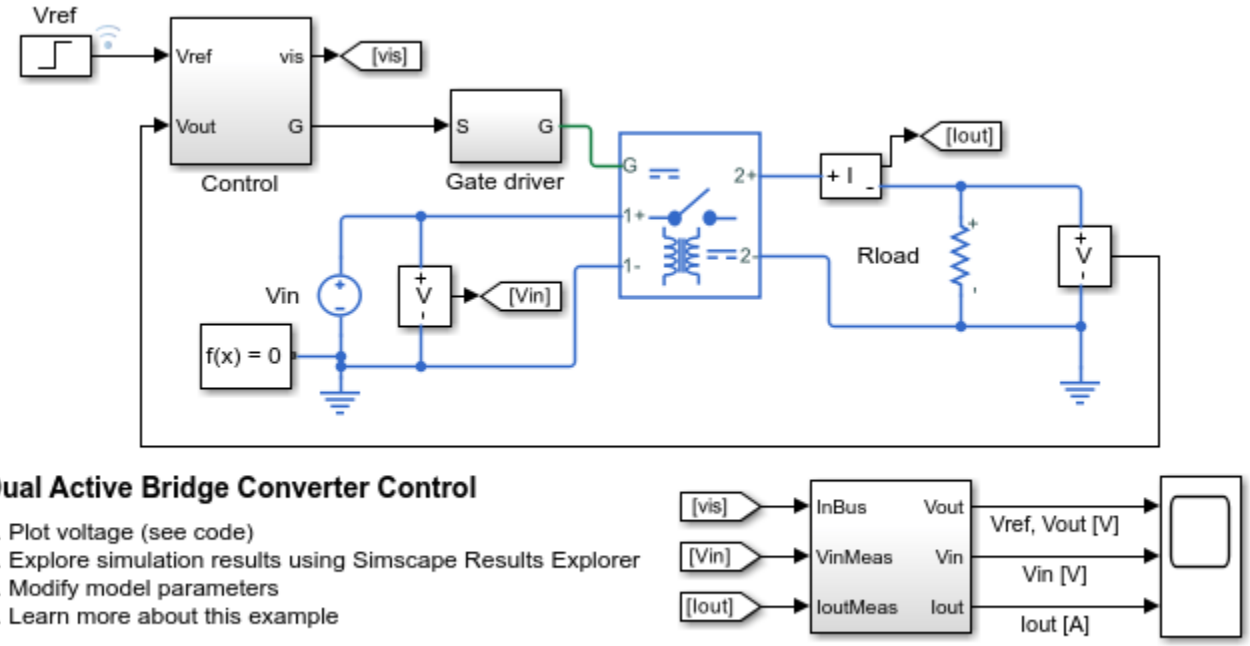
The plot below shows the power of the electrolyzer, solar and storage system, battery state of charge, and produced hydrogen.


Power
Power (kW)
200
100
0
Electrolyzer
Solar
Storage
0
20
40
60
80
100
120
140
160
180
Time (hours)
Battery State of Charge [%]
SOC
100
90
80
Time (hours)
Produced hydrogen [kg]
mass H2 [kg]
100
50
0
Time (hours)

# Dual Active Bridge Control

This example shows how to control the output voltage of a dual active bridge DC-DC converter. Each switch is on for 50% of its switching period. A phase shift controller introduces a variable phase shift in the output bridge and controls the output voltage. The input voltage and the system load are held constant throughout the simulation. The total simulation time (t) is 0.25 s. At t = 0.15 s, the voltage reference changes.

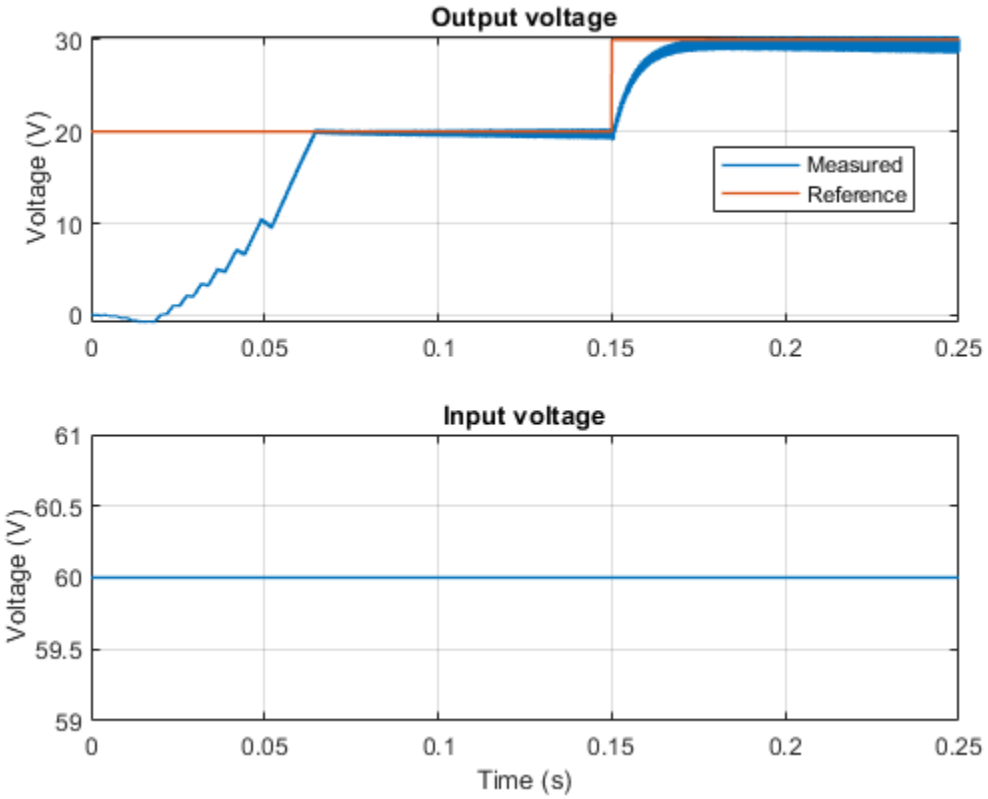
### Model




Copyright 2022 The MathWorks, Inc.

### Simulation Results from Simscape Logging

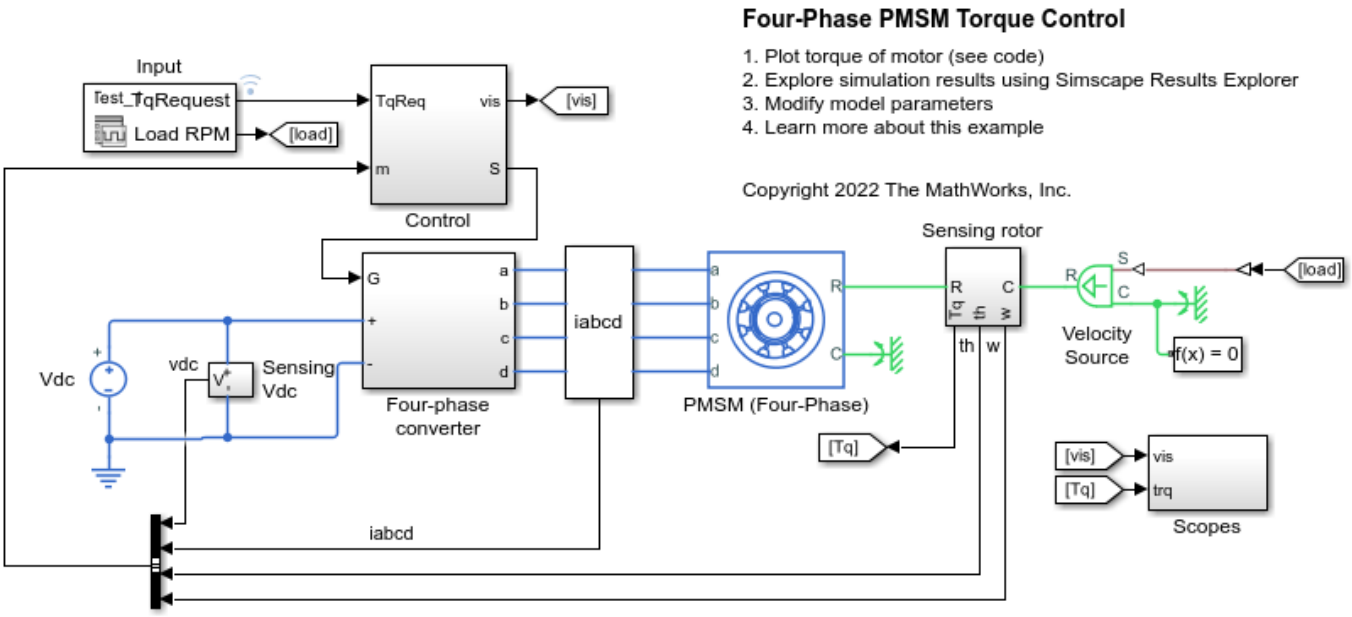
The plot below shows the requested and measured voltage for the test and the input voltage in the circuit.


Output voltage
Voltage (V)
30
20
10
0
0
0.05
0.1
0.15
0.2
0.25
Measured
Reference
Input voltage
Voltage (V)
61
60.5
60
59.5
59
0
0.05
0.1
0.15
0.2
0.25
Time (s)

# Four-Phase PMSM Torque Control

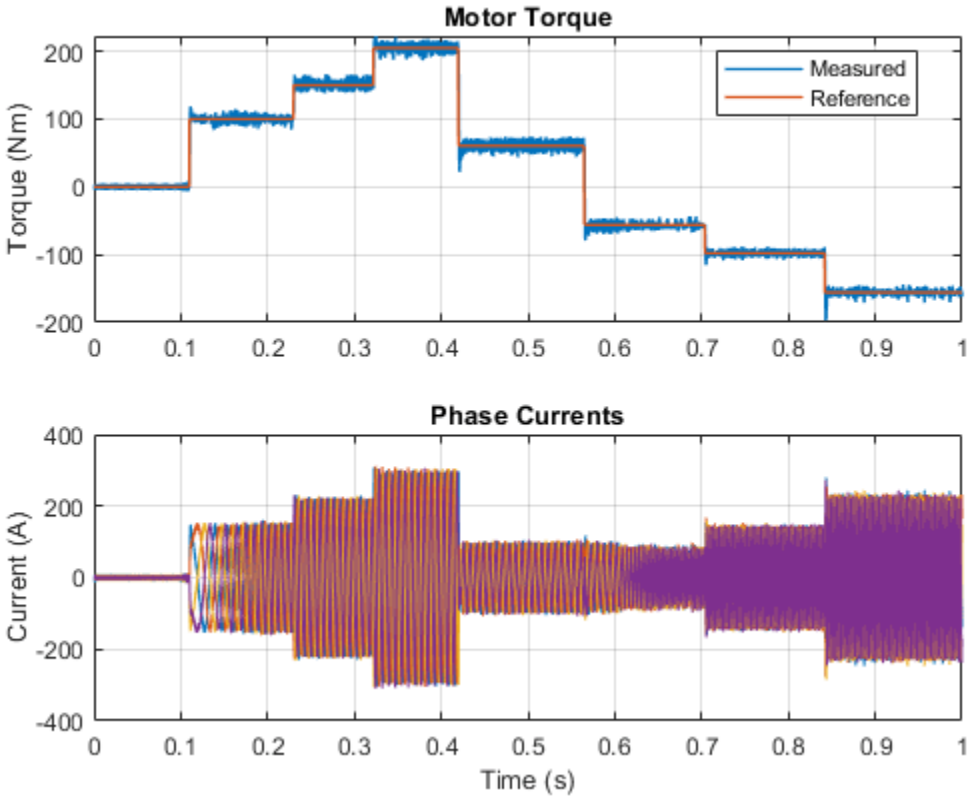
This example shows how to control the torque in an electrical-traction drive based on a four-phase permanent magnet synchronous machine (PMSM). A DC voltage source feeds the PMSM through a controlled four-phase converter. The PMSM operates in both motoring and generating modes according to the load. An ideal angular velocity source provides the load. The Control subsystem uses an open-loop approach to control the PMSM torque and a closed-loop approach to control the current. At each sample instant, the torque request is converted to a relevant q-axis current reference. The current control is PI-based. The simulation uses several torque steps in both motor and generator modes. The Scopes subsystem contains scopes that allow you to see the simulation results.

### Model




### Simulation Results from Simscape Logging

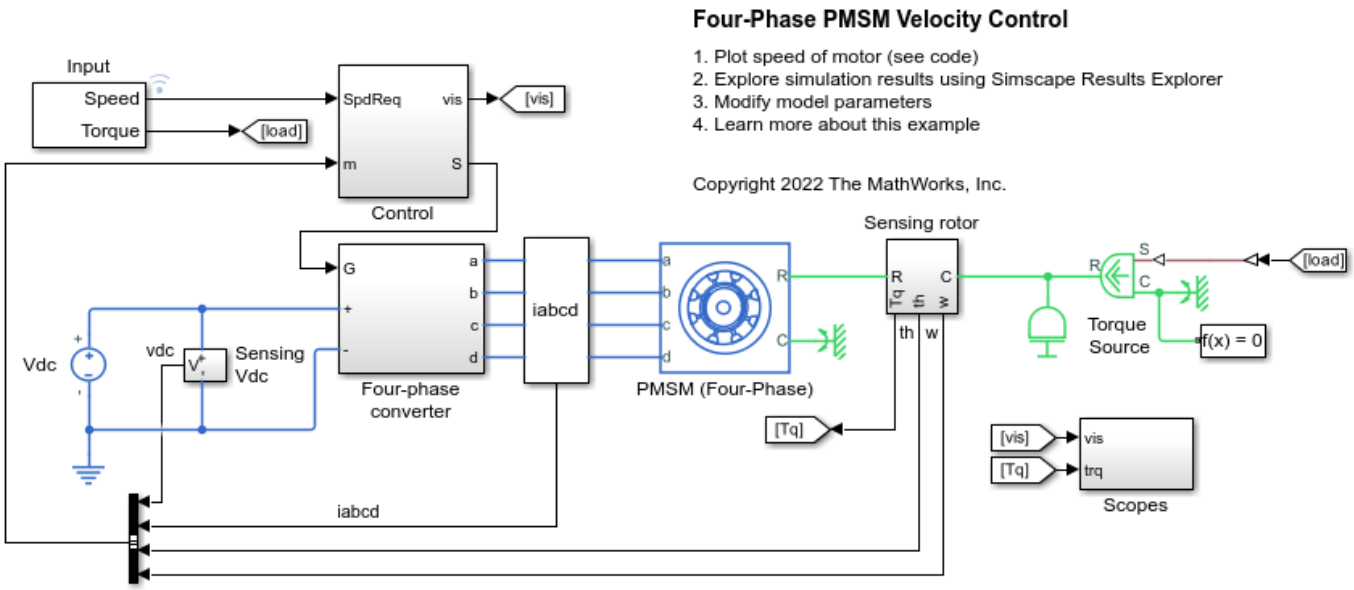
The plot below shows the requested and measured torque for the test, as well as the phase currents in the electric drive.


Motor Torque
Measured
Reference
Torque (Nm)
200
100
0
-100
-200
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Phase Currents
Current (A)
400
200
0
-200
-400
Time (s)

# Four-Phase PMSM Velocity Control

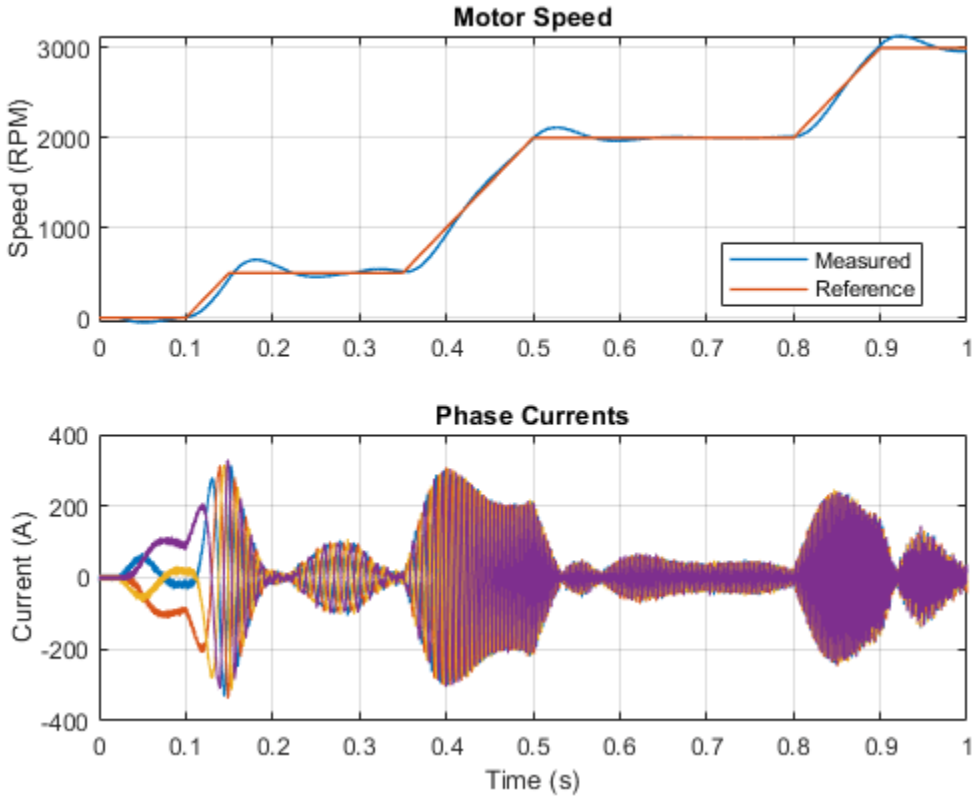
This example shows how to control the rotor angular velocity in an electrical-traction drive based on a four-phase permanent magnet synchronous machine (PMSM). A DC voltage source feeds the PMSM through a controlled four-phase converter. The PMSM operates in both motoring and generating modes according to the load. An ideal torque source provides the load. The Scopes subsystem contains scopes that allow you to see the simulation results. The Control subsystem includes a PI-based cascade control structure that has an outer angular-velocity-control loop and four inner current-control loops. During the one second simulation, the angular velocity demand is 0 rpm, 500 rpm, 2000 rpm, and then 3000 rpm.

### Model




### Simulation Results from Simscape Logging

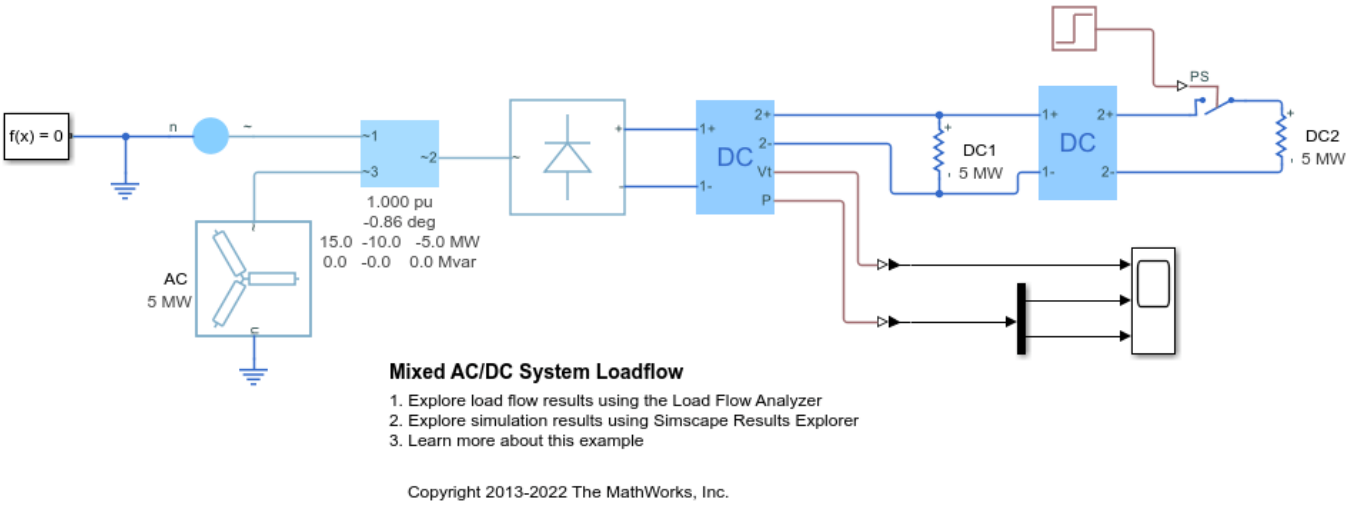
The plot below shows the requested and measured speed for the test, as well as the phase currents in the electric drive.


Motor Speed
Speed (RPM)
3000
2000
1000
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Measured
Reference
Phase Currents
Current (A)
400
200
0
-200
-400
Time (s)

# Mixed AC/DC System Loadflow

This example shows how to use the Load Flow Analyzer to review the load flow results of a mixed AC/DC system. The model to which this analysis is applied includes an AC load flow source, a three-phase rectifier, and three loads. One of the loads is AC, one is permanently connected on the DC side, and one is switched on the DC side.

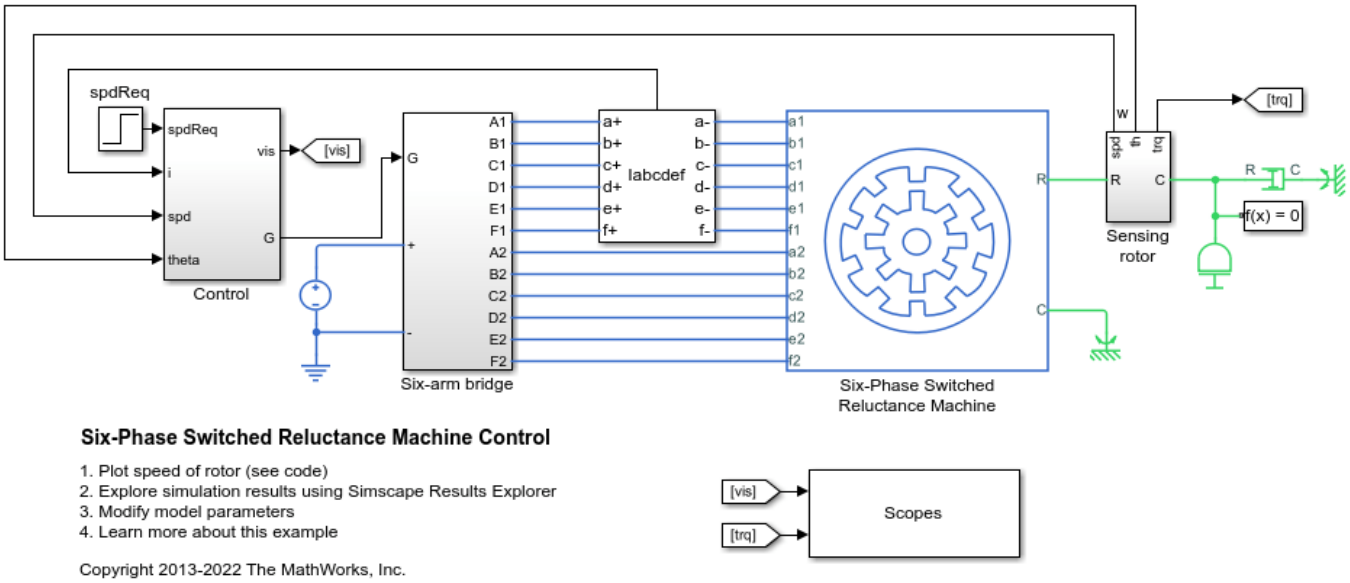
### Model



# Six-Phase Switched Reluctance Machine Control

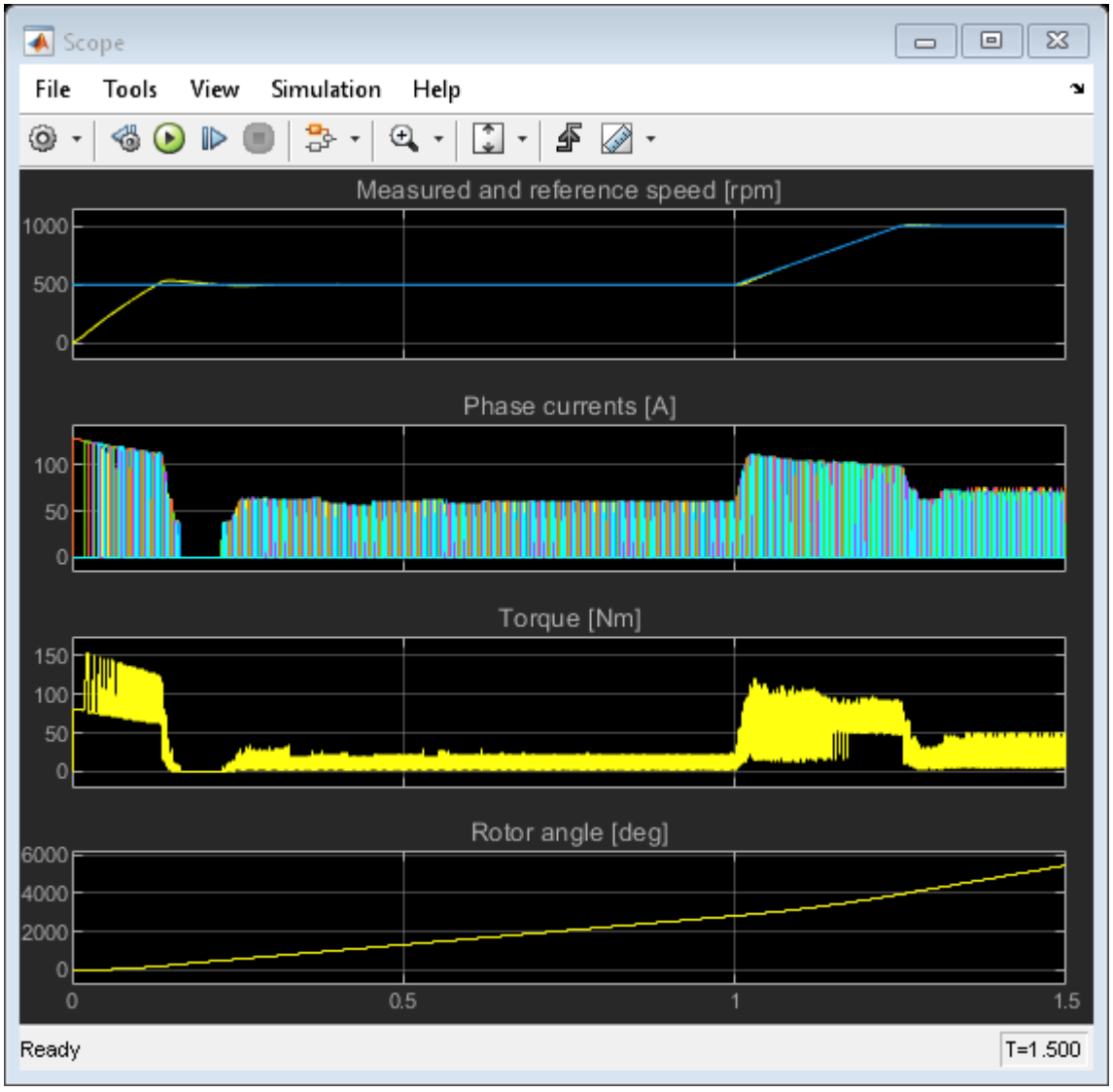
This example shows how to control the rotor speed in an electrical drive based on a six-phase switched reluctance machine (SRM). A DC voltage source feeds the SRM through a controlled six-arm bridge. The converter turn-on and turn-off angles are constant.

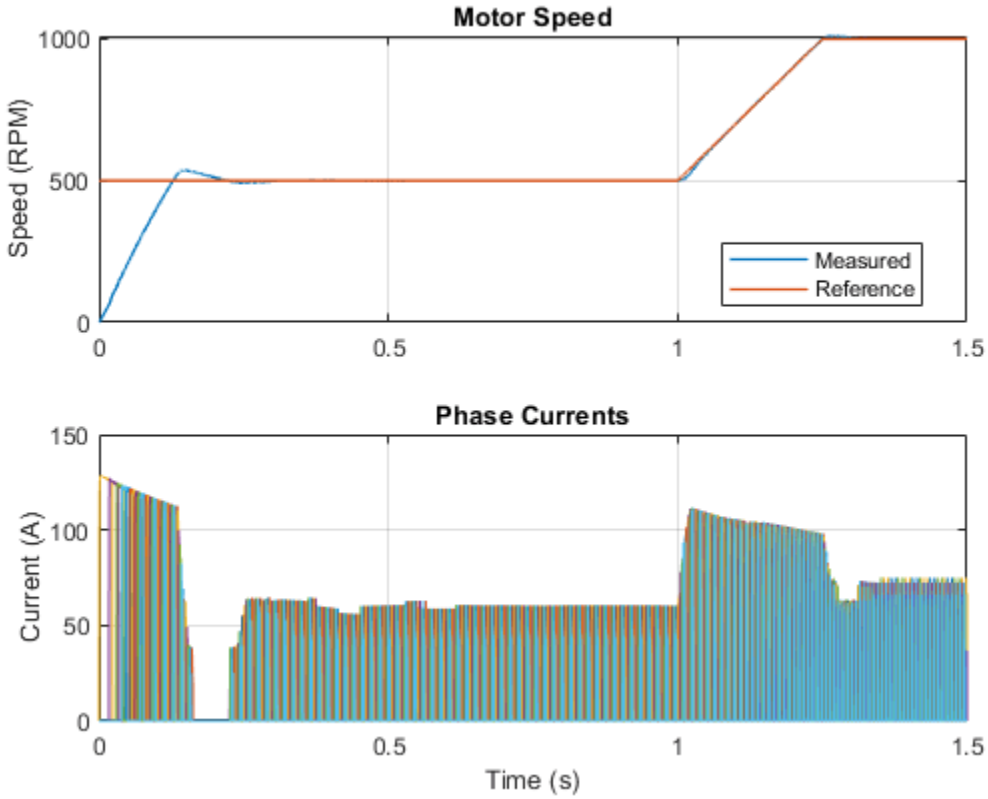
### Model Overview




### Simulation Results from Simscape Logging

The plot below shows the requested and measured speed for the test, as well as the phase currents in the electric drive.


Scope
File
Tools
View
Simulation
Help
Measured and reference speed [rpm]
1000
500
0
Phase currents [A]
100
50
0
Torque [Nm]
150
100
50
0
Rotor angle [deg]
6000
4000
2000
0
0
0.5
1
1.5
Ready
T=1.500




### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 100 microseconds.

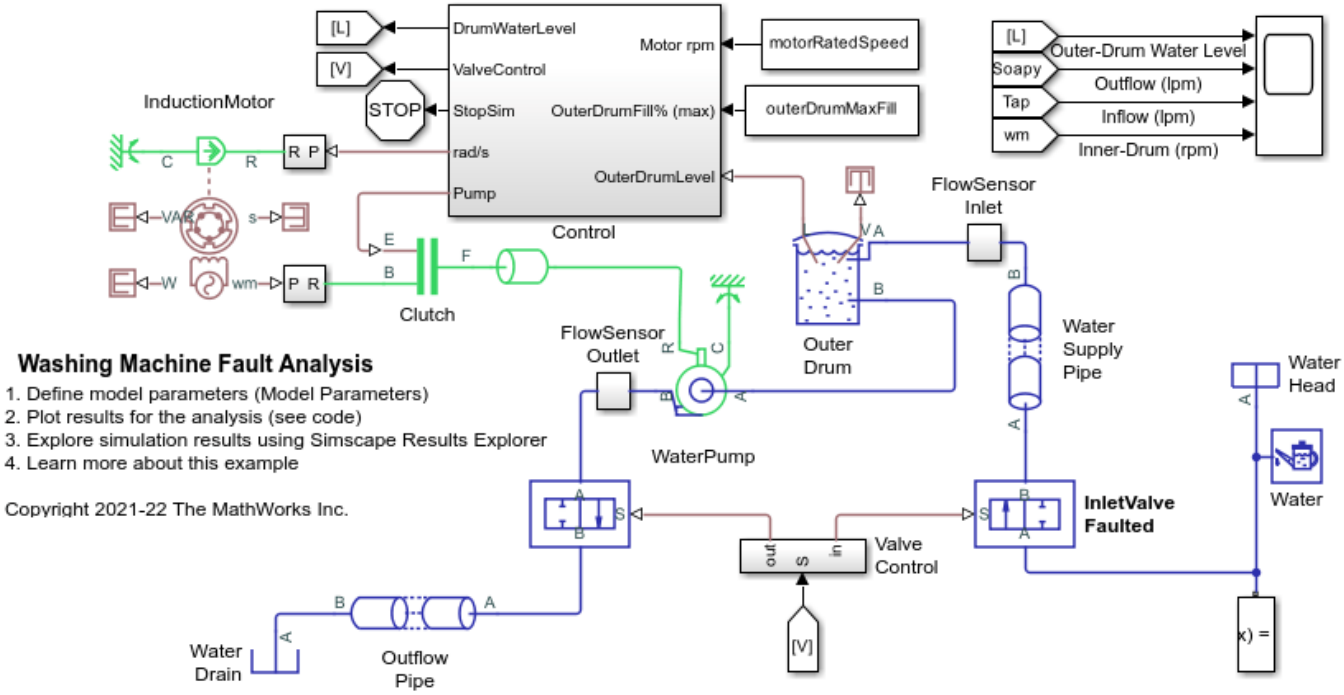
# Washing Machine Fault Analysis

This example shows how to model a washing machine and introduce a fault in its operation. The machine water supply faults at a specific time and this example models the system response under this scenario. The electric motor switches off and the machine operation is aborted by discharging the partially filled outer drum.

### Model Overview

```
open_system('sscv_washing_machine_fault_analysis.slx')
```

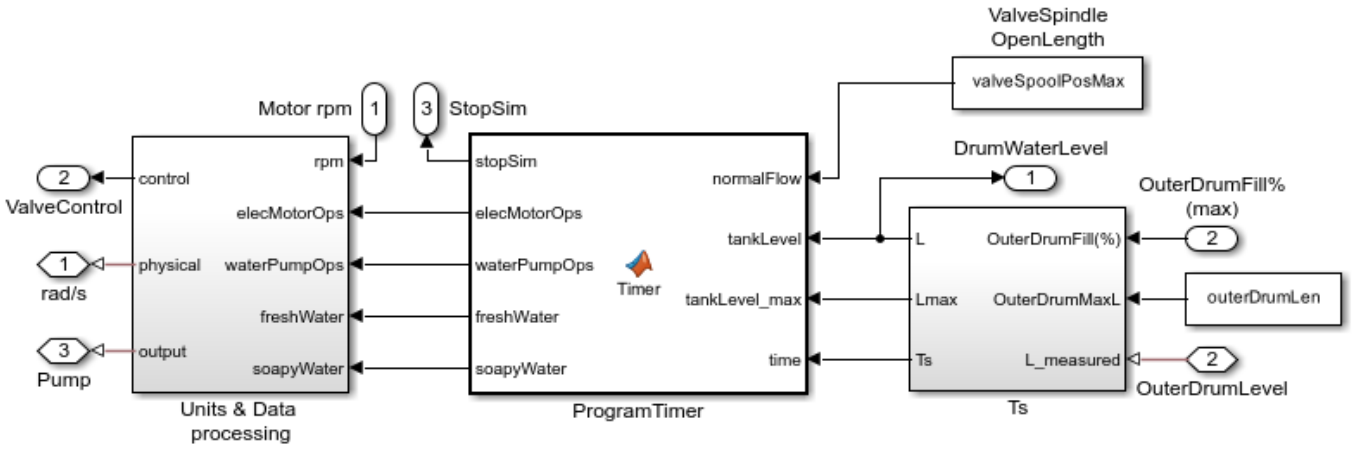
```
set_param(find_system('sscv_washing_machine_fault_analysis','FindAll', 'on','type','annotation',
```




### Control Overview

To modify or edit the machine wash cycle, use the ProgramTimer subsystem. The ProgramTime subsystem comprises the response to a fault.

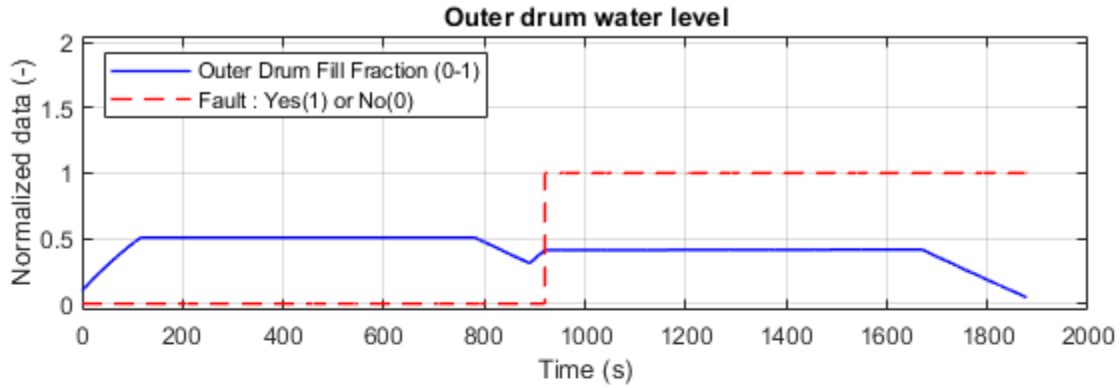
```
open_system('sscv_washing_machine_fault_analysis/Control','force')
```

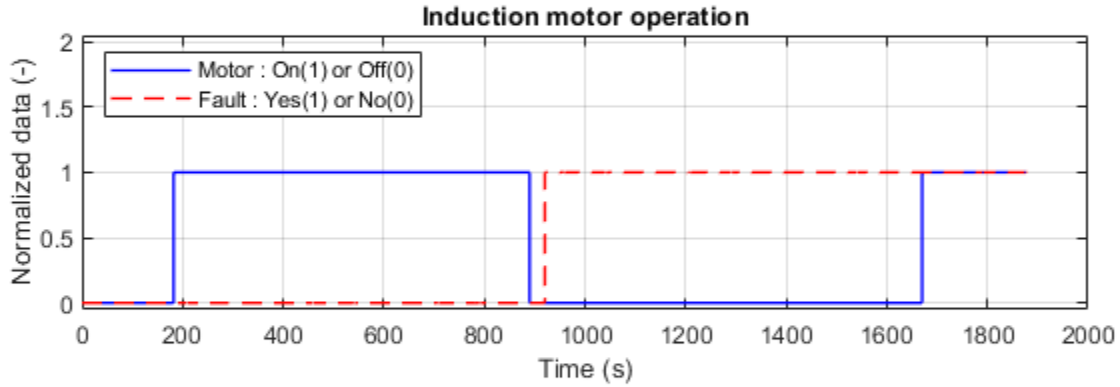


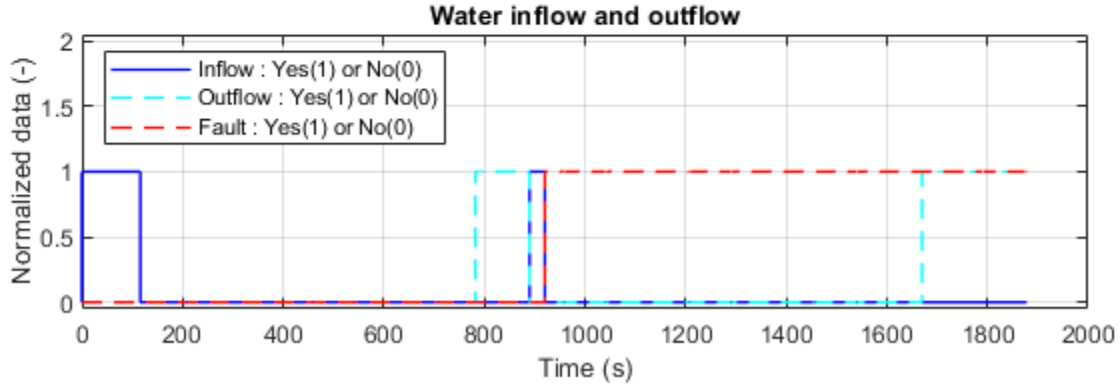

### Simulation Results from Simscape™ Logging

The plot below shows the water inlet and outlet valve operation (On/Off) and the pump operation with outer drum fill level. When the inlet valve faults, the pump switches off. After some time, the outlet valve opens to discharge the partially filled washing machine drum.

```
sscv_washing_machine_fault_analysis_plotResults;
```




```
clear all
close all
bdclose all
```

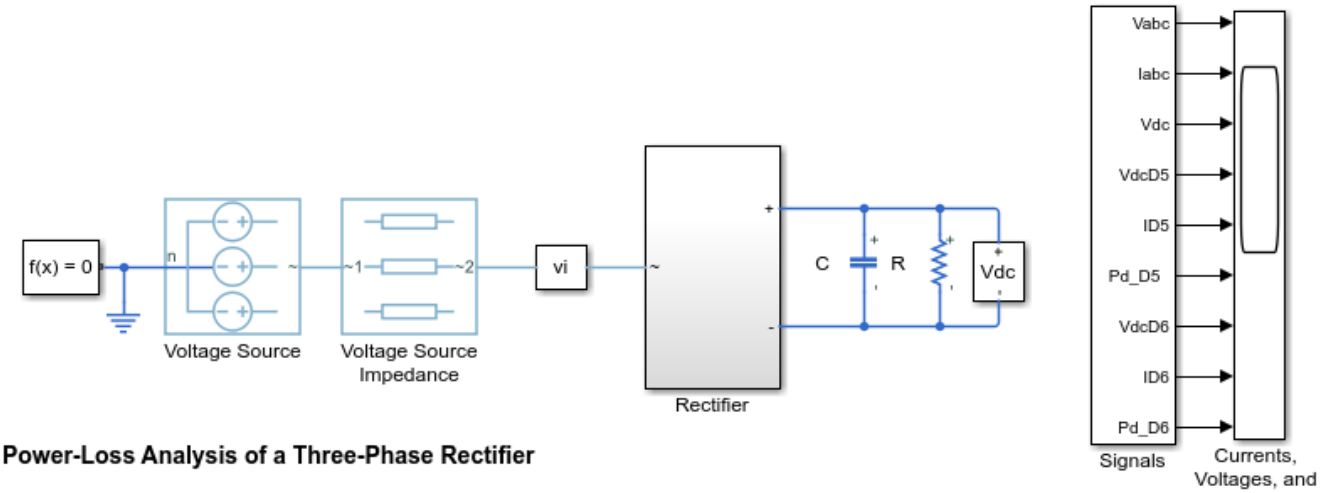
# Calculate Losses in a Power Converter

This example shows how to calculate the losses in a power converter using the ee_getPowerLossSummary function and the power_dissipated variable. The power converter contains six Diode blocks. Diodes D1, D2, D4, and D5 each loose, on average, 52.19 W over the course of the 0.5 second simulation. Diodes D3 and D6 each loose an average of 52.22 W for the same time period because they experience a 340 W transient power-dissipation spike due to DC-load capacitor charging before t = 1e-3 s. For each diode, the average power loss when the power converter is operating in steady-state, t = 0.4 to 0.5 s, is 52.19 W.

### Model

```
open_system('ee_rectifier_power_dissipated')
```

```
set_param(find_system('ee_rectifier_power_dissipated','FindAll', 'on','type','annotation','Tag',
```




**Power-Loss Analysis of a Three-Phase Rectifier**

1. Calculate power dissipated by all of the diodes in the model for the entire simulation time (see command window for output).
2. Calculate power dissipated by diode D6 for the entire simulation time (see command window for output).
3. Explore simulation results using Simscape Results Explorer.
4. Calculate the maximum power dissipated by diode D6. (see command window for output).
5. Calculate power dissipated by diode D6 between simulation time, t = 1e-3 to 0.5 s (see command window for output).
6. Plot power dissipated by the diodes (see code).
7. Calculate power dissipated by all of the diodes between simulation time, t = 0.4 to 0.5 s (see command window for output).
8. Learn more about the ee_getPowerLossSummary function.

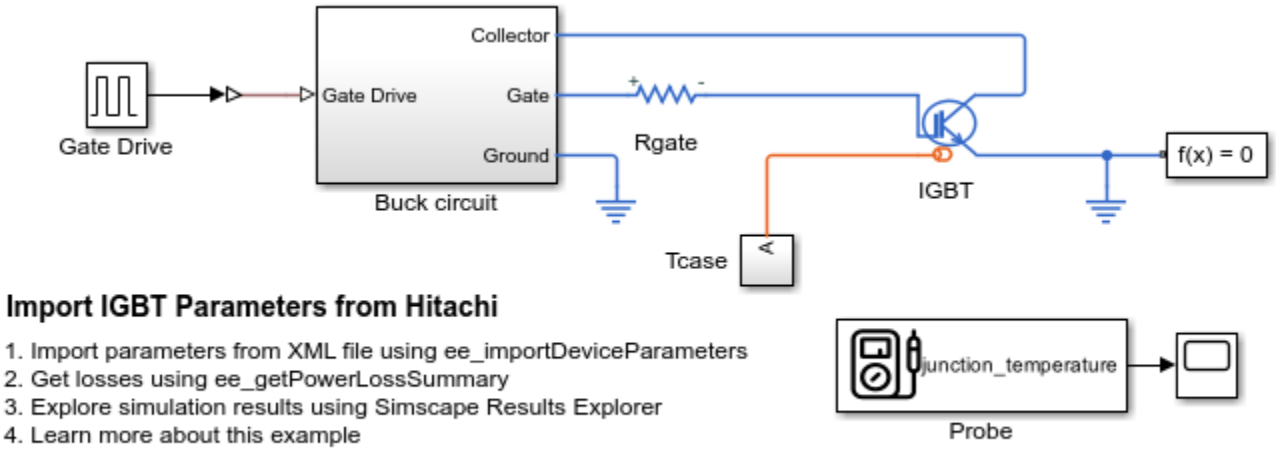
Copyright 2016-2022 The MathWorks, Inc.

# Import IGBT Parameters from Hitachi

This example shows how you can import parameters from a Hitachi IGBT module and save them in an IGBT(Ideal, Switching) block. The Hitachi IGBT parameters are stored in an XML file in the `hitachi` format.

Copyright 2021-2022 The MathWorks, Inc

### Model Overview

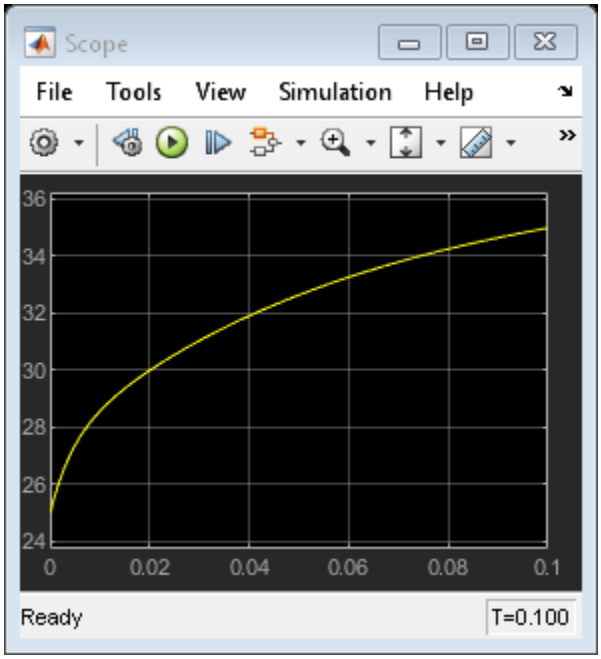



### Import Device Parameters

Use the utility function `ee_importDeviceParameters` to import the XML file that contains the IGBT parameters in the `hitachi` format. To find more IGBT devices to import, see Hitachi website.

The thermal network in the imported XML file comprises 4 elements. In the IGBT (Ideal, Switching) block, specify the appropriate value for the **Initial node temperatures [T1 T2 ... Tn]** parameter.

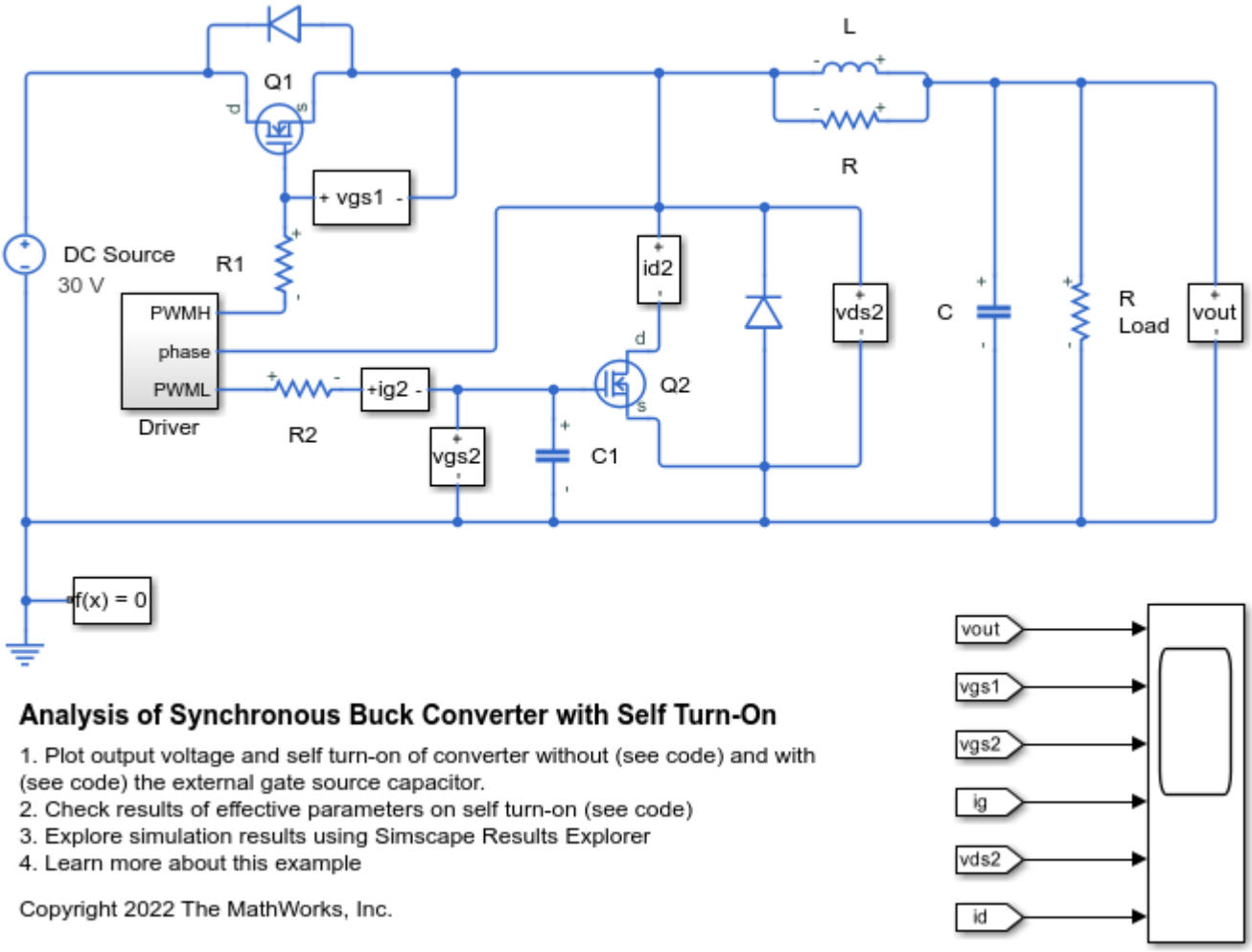
### Simulation Results



# Analysis of Synchronous Buck Converter with Self Turn-On

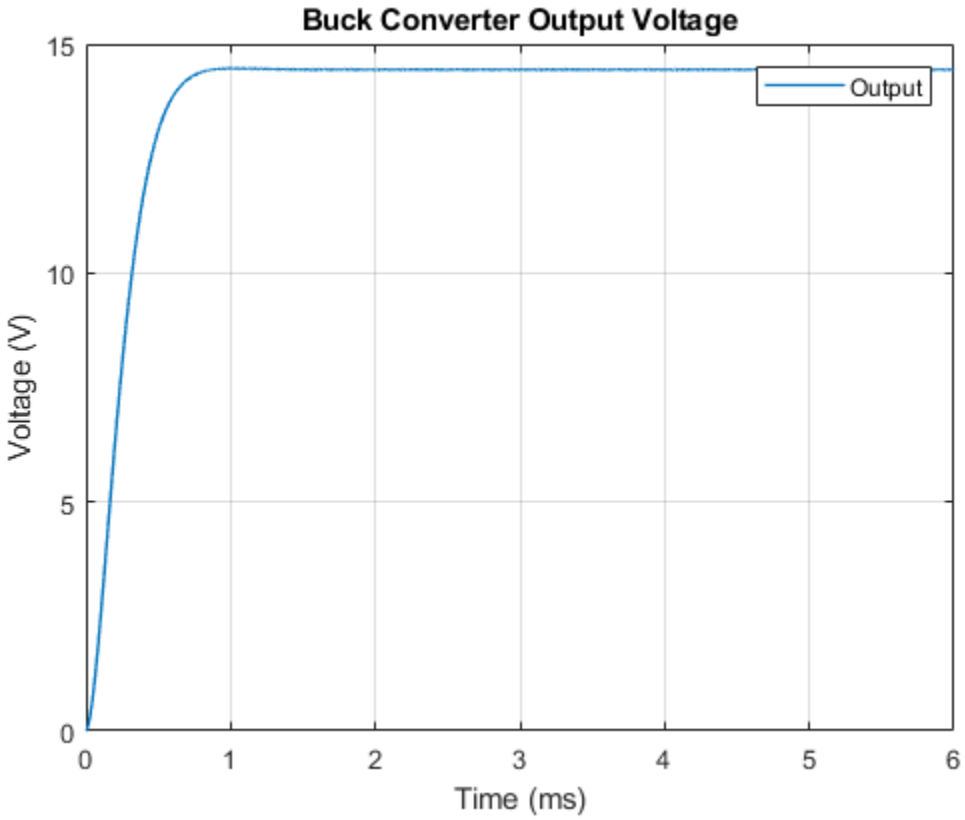
This example analyzes the self turn-on mechanism in a synchronous buck converter. When the low-side MOSFET Q2 is off and the high-side MOSFET Q1 turns on, a current flows through the gate-drain capacitance of Q2 due to the voltage ramp dv/dt between the drain and source of Q2. This current induces a voltage across the gate and source of Q2. If the voltage exceeds the threshold voltage, Q2 unintentionally turns on. This example shows how the MOSFET parameters affect the self turn-on mechanism and how you can prevent it.

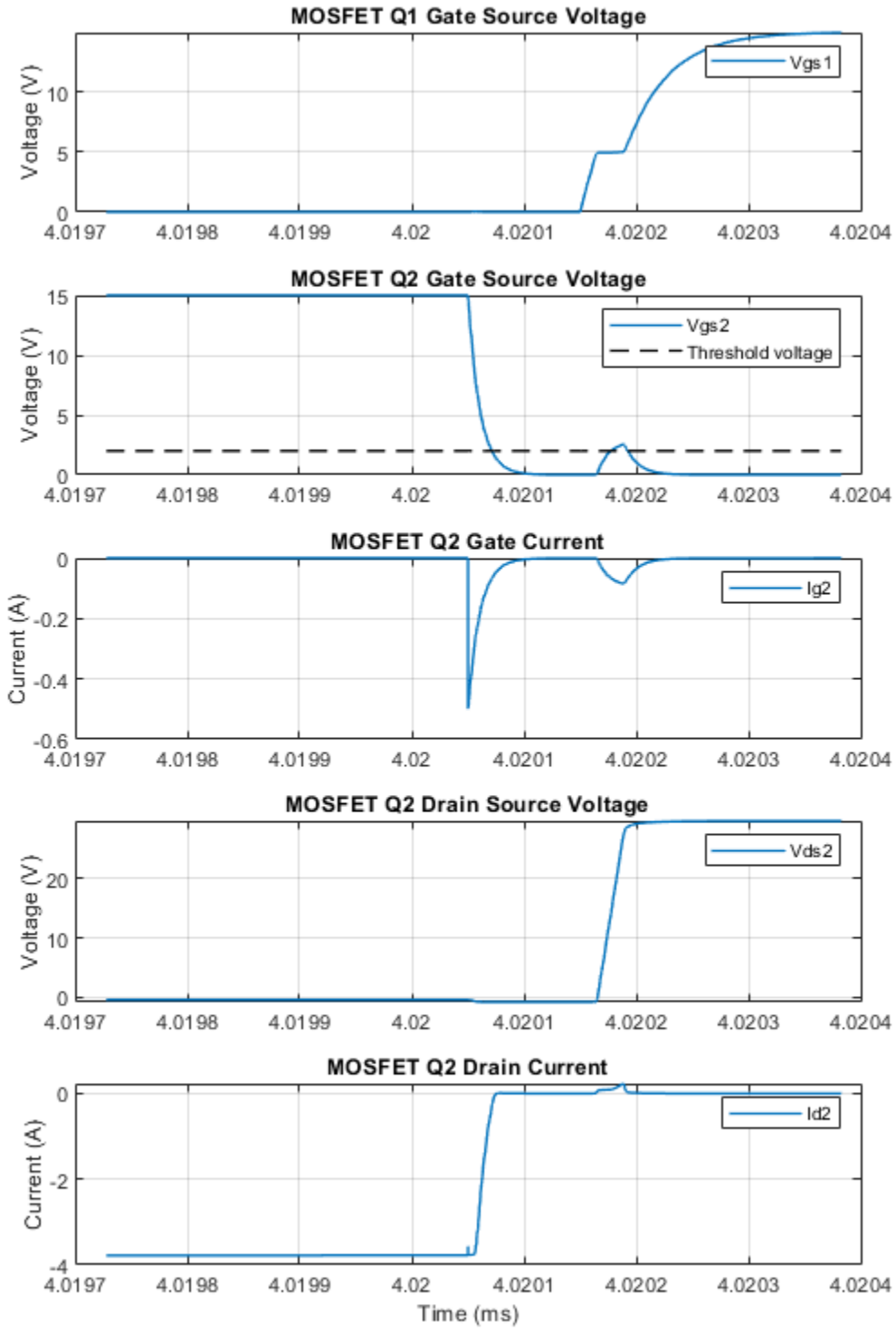
### Synchronous Buck Converter Overview




### Simulation of Output Voltage and Self Turn-On

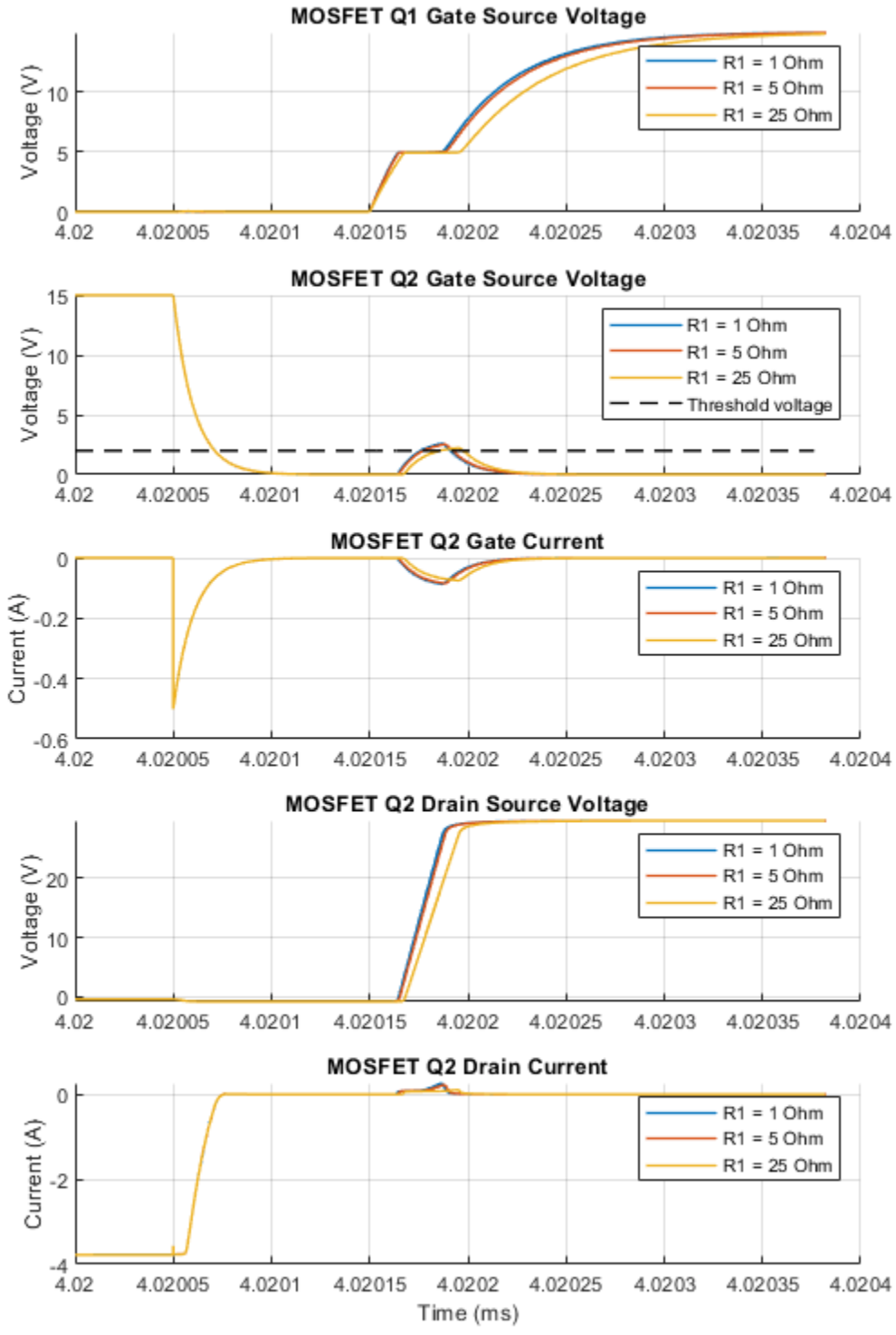
The plots below show the output voltage of the buck converter and illustrate the self turn-on mechanism of Q2. The duty cycle of the model is equal to 50%. In this example, to demonstrate the self turn-on phenomenon, the gate resistance is very high. Q2 self turns on at the rising edge of the gate source voltage of Q1.


Buck Converter Output Voltage
Output
Voltage (V)
Time (ms)
0
5
10
15
0
1
2
3
4
5
6


MOSFET Q1 Gate Source Voltage
Voltage (V)
Vgs1
MOSFET Q2 Gate Source Voltage
Voltage (V)
Vgs2
Threshold voltage
MOSFET Q2 Gate Current
Current (A)
Ig2
MOSFET Q2 Drain Source Voltage
Voltage (V)
Vds2
MOSFET Q2 Drain Current
Current (A)
Id2
Time (ms)

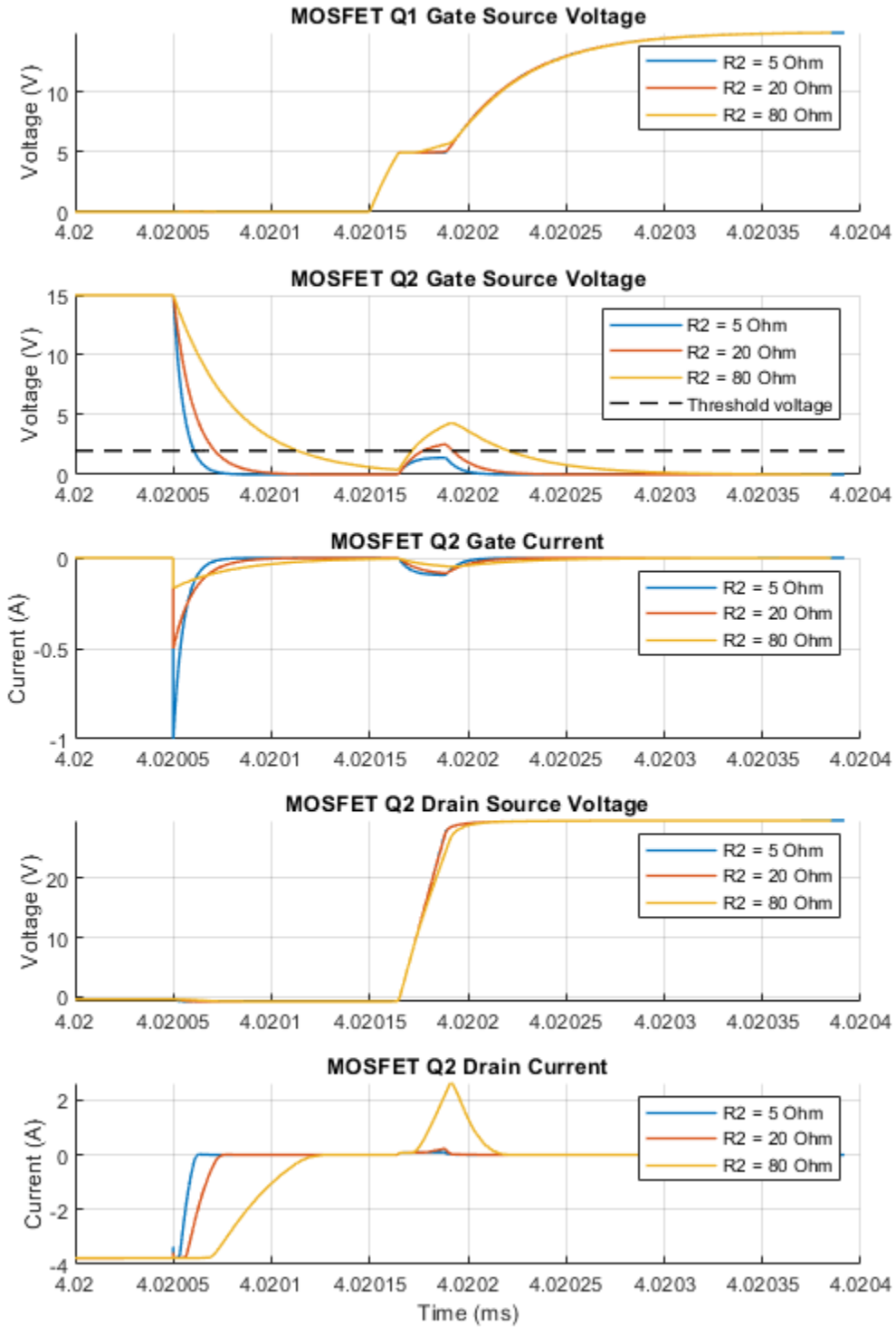
### Effect of Gate Resistance on High-Side MOSFET Q1

When Q1 turns on, its gate resistance affects the slope of the drain-source voltage of Q2. The higher gate resistance results in a lower slope rate of the drain-source voltage and reduces the occurrences of self turn-on for Q2.


MOSFET Q1 Gate Source Voltage
Voltage (V)
R1 = 1 Ohm
R1 = 5 Ohm
R1 = 25 Ohm
MOSFET Q2 Gate Source Voltage
Threshold voltage
MOSFET Q2 Gate Current
Current (A)
MOSFET Q2 Drain Source Voltage
MOSFET Q2 Drain Current
4.02
4.02005
4.0201
4.02015
4.0202
4.02025
4.0203
4.02035
4.0204
Time (ms)

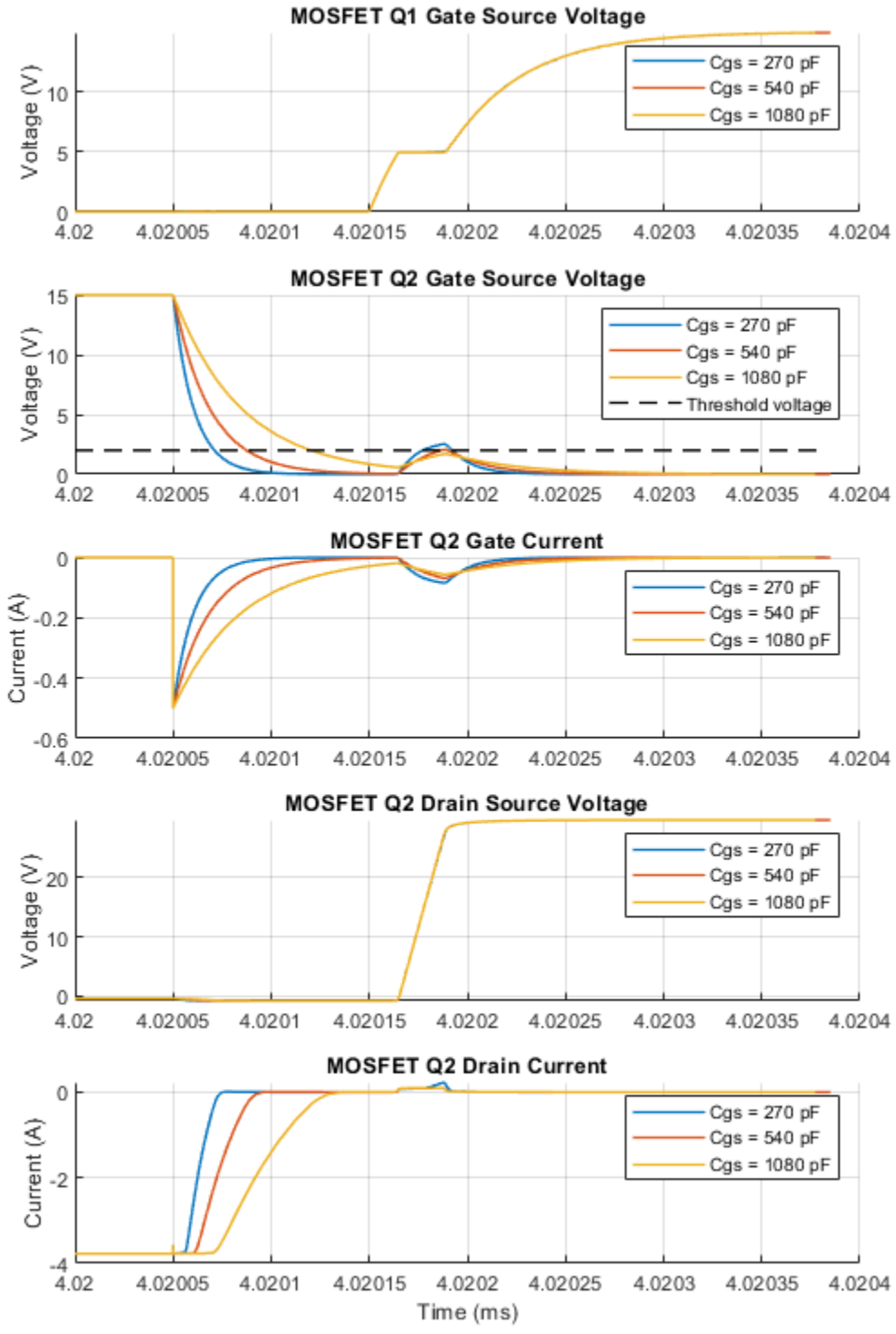
### Effect of Gate Resistance on Low-Side MOSFET Q2

The gate resistance of Q2 is directly proportional to its peak gate and to the time required to discharge the gate charge.


MOSFET Q1 Gate Source Voltage
Voltage (V)
R2 = 5 Ohm
R2 = 20 Ohm
R2 = 80 Ohm
MOSFET Q2 Gate Source Voltage
Voltage (V)
R2 = 5 Ohm
R2 = 20 Ohm
R2 = 80 Ohm
Threshold voltage
MOSFET Q2 Gate Current
Current (A)
R2 = 5 Ohm
R2 = 20 Ohm
R2 = 80 Ohm
MOSFET Q2 Drain Source Voltage
Voltage (V)
R2 = 5 Ohm
R2 = 20 Ohm
R2 = 80 Ohm
MOSFET Q2 Drain Current
Current (A)
R2 = 5 Ohm
R2 = 20 Ohm
R2 = 80 Ohm
4.02
4.02005
4.0201
4.02015
4.0202
4.02025
4.0203
4.02035
4.0204
Time (ms)

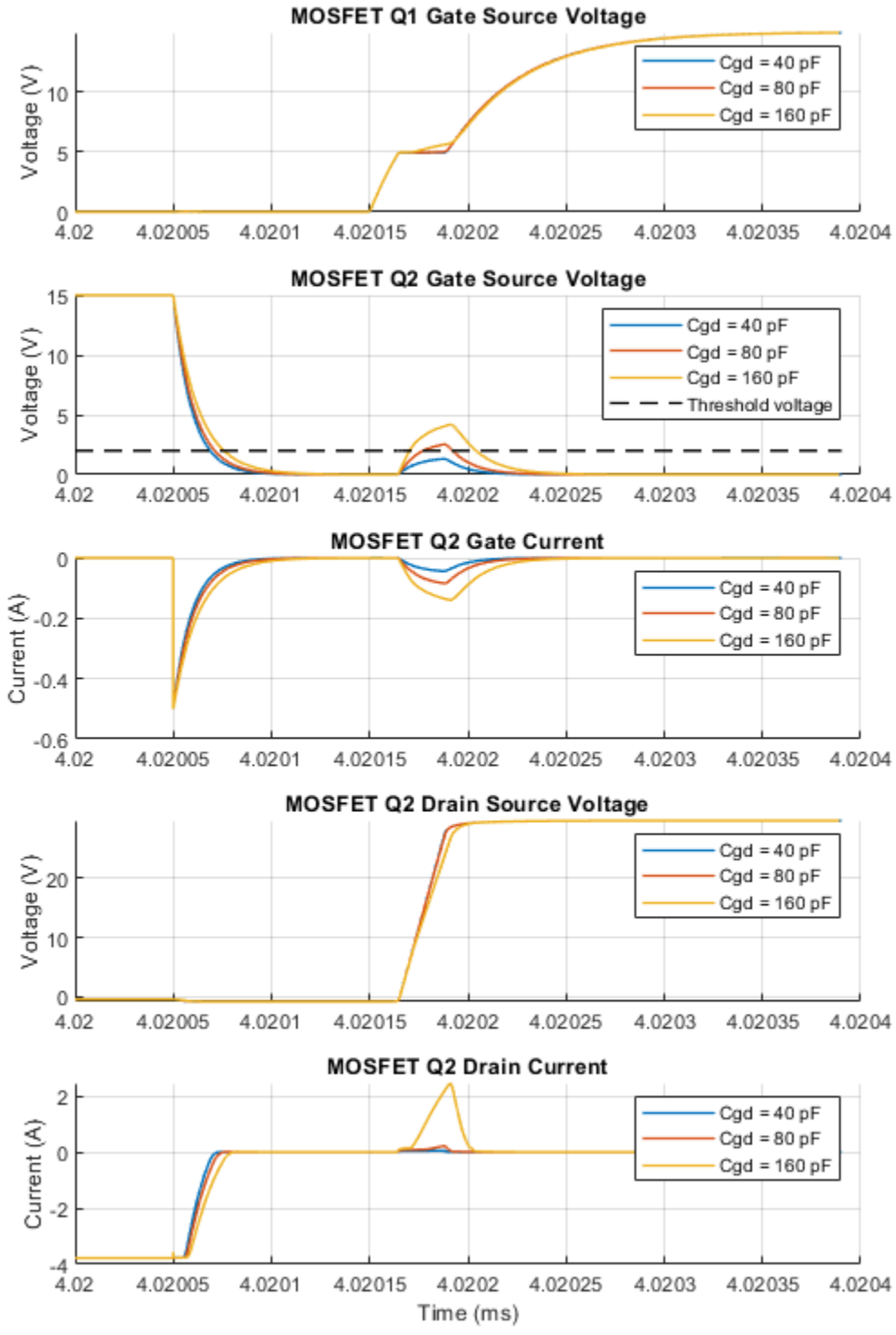
### Effect of Gate Drain Capacitance of Low-Side MOSFET Q2

The current that flows through the gate-drain capacitance, Cgd, of Q2 is proportional to the gate-drain capacitance value. A small Cgd helps preventing the self turn-on mechanism of a low-side MOSFET.


MOSFET Q1 Gate Source Voltage
Voltage (V)
Cgs = 270 pF
Cgs = 540 pF
Cgs = 1080 pF
MOSFET Q2 Gate Source Voltage
Voltage (V)
Cgs = 270 pF
Cgs = 540 pF
Cgs = 1080 pF
Threshold voltage
MOSFET Q2 Gate Current
Current (A)
Cgs = 270 pF
Cgs = 540 pF
Cgs = 1080 pF
MOSFET Q2 Drain Source Voltage
Voltage (V)
Cgs = 270 pF
Cgs = 540 pF
Cgs = 1080 pF
MOSFET Q2 Drain Current
Current (A)
Cgs = 270 pF
Cgs = 540 pF
Cgs = 1080 pF
4.02
4.02005
4.0201
4.02015
4.0202
4.02025
4.0203
4.02035
4.0204
Time (ms)

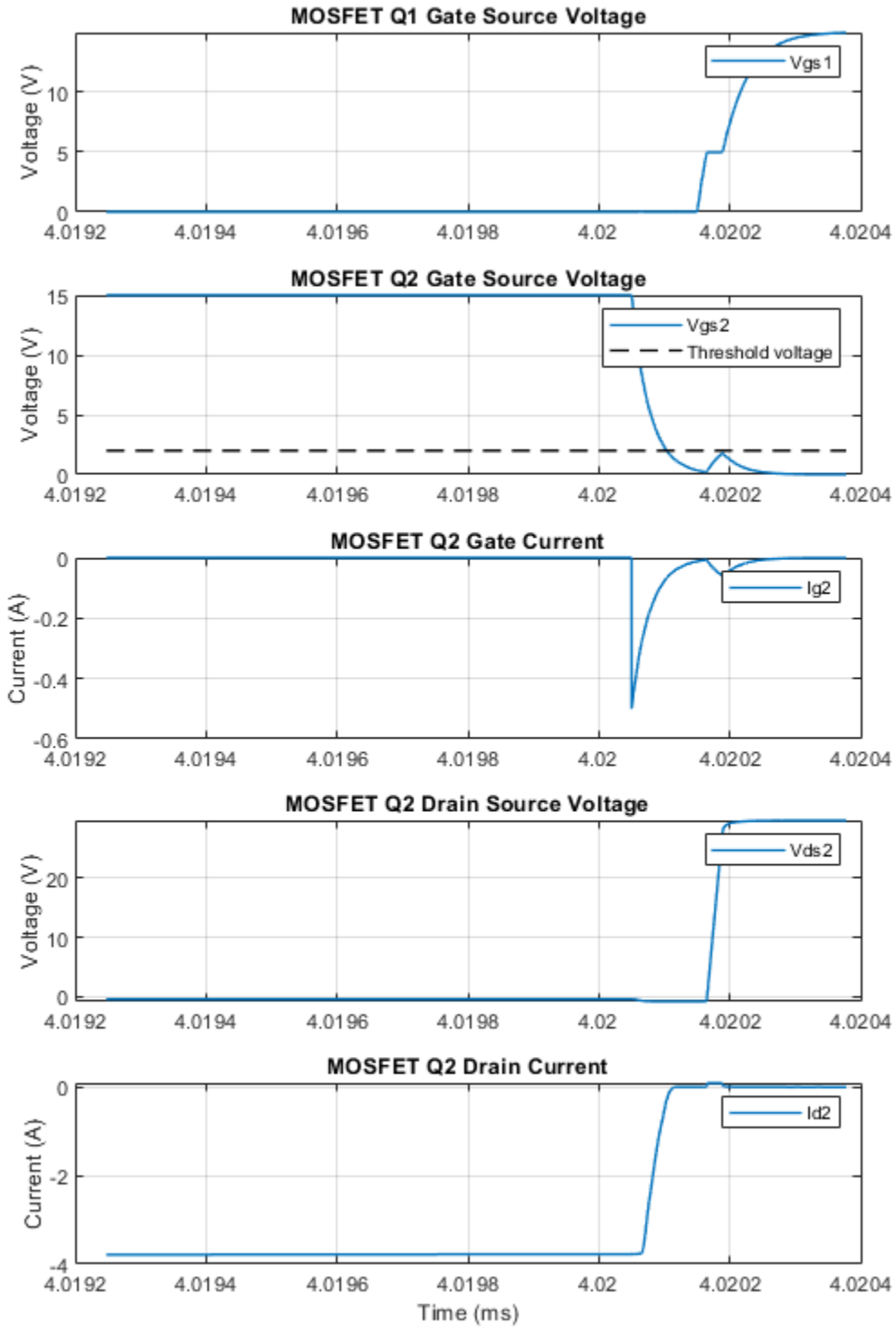
### Effect of Gate Source Capacitance of Low-Side MOSFET Q2

A high value of the gate-source capacitance, Cgs, decreases the peak gate source voltage of Q2 and increases the time required to discharge the gate charge. To prevent the self turn-on mechanism, as the gate-source voltage of Q2 is proportional to `Cgd/(Cgs+Cgd)`, use a MOSFET with low Cgd/Cgs ratio. However, high values of the gate-source capacitance reduce the switching speed.


MOSFET Q1 Gate Source Voltage
Voltage (V)
Cgd = 40 pF
Cgd = 80 pF
Cgd = 160 pF
MOSFET Q2 Gate Source Voltage
Voltage (V)
Cgd = 40 pF
Cgd = 80 pF
Cgd = 160 pF
Threshold voltage
MOSFET Q2 Gate Current
Current (A)
Cgd = 40 pF
Cgd = 80 pF
Cgd = 160 pF
MOSFET Q2 Drain Source Voltage
Voltage (V)
Cgd = 40 pF
Cgd = 80 pF
Cgd = 160 pF
MOSFET Q2 Drain Current
Current (A)
Cgd = 40 pF
Cgd = 80 pF
Cgd = 160 pF
4.02
4.02005
4.0201
4.02015
4.0202
4.02025
4.0203
4.02035
4.0204
Time (ms)

### Method of Preventing the Self Turn-On Mechanism

To prevent the MOSFET self turn-on mechanism, you should implement MOSFETs with high threshold voltage, low gate drain capacitance, and low gate drain to gate source capacitance ratio. You can add a capacitor between the gate and the source of the MOSFET to further reduce the Cgd/Cgs ratio.


MOSFET Q1 Gate Source Voltage
Voltage (V)
Vgs1
MOSFET Q2 Gate Source Voltage
Voltage (V)
Vgs2
Threshold voltage
MOSFET Q2 Gate Current
Current (A)
Ig2
MOSFET Q2 Drain Source Voltage
Voltage (V)
Vds2
MOSFET Q2 Drain Current
Current (A)
Id2
Time (ms)

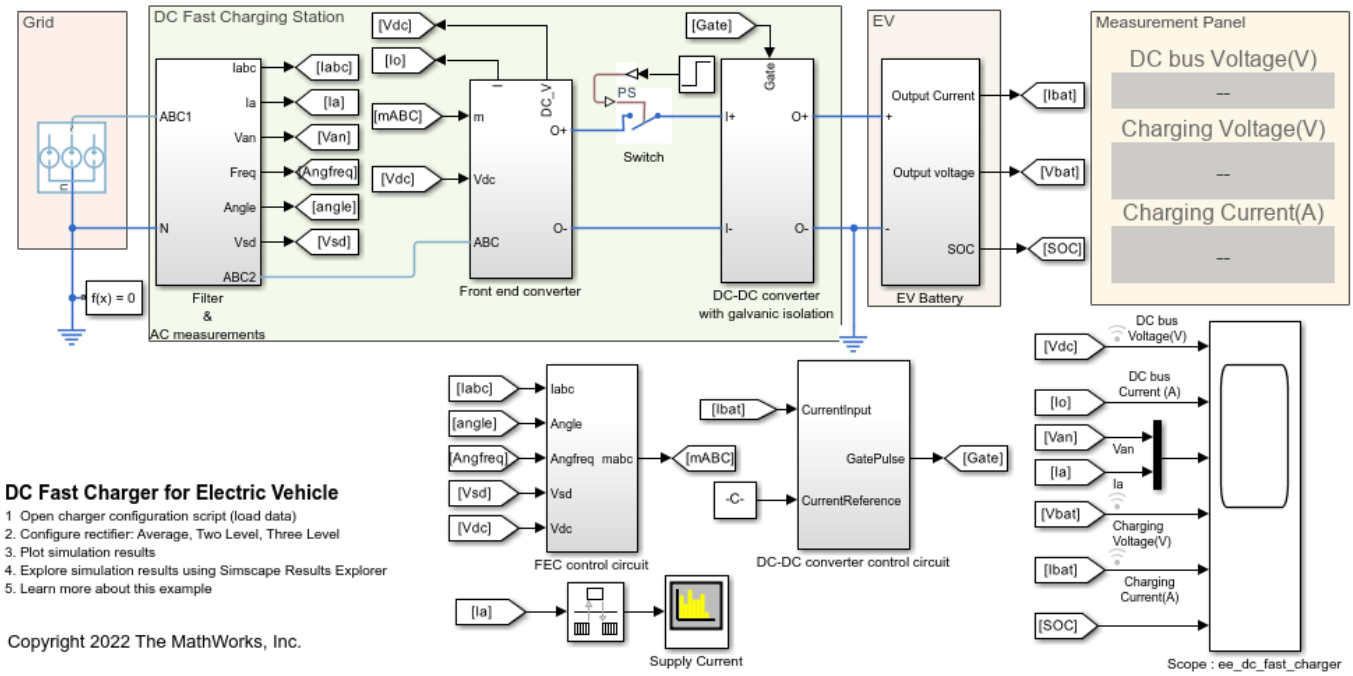
# DC Fast Charger for Electric Vehicle

This example models a DC fast charging station connected with the battery pack of an Electric Vehicle (EV).

The main components of the example are:

- Grid - Model the AC supply voltage as a three-phase constant voltage source.
- DC Fast Charging Station - Model the power electronic circuits to convert the AC supply voltage from the grid to the DC voltage level that the EV battery pack requires.
- EV battery pack - Model the battery pack as series of battery cells.

### Model Overview

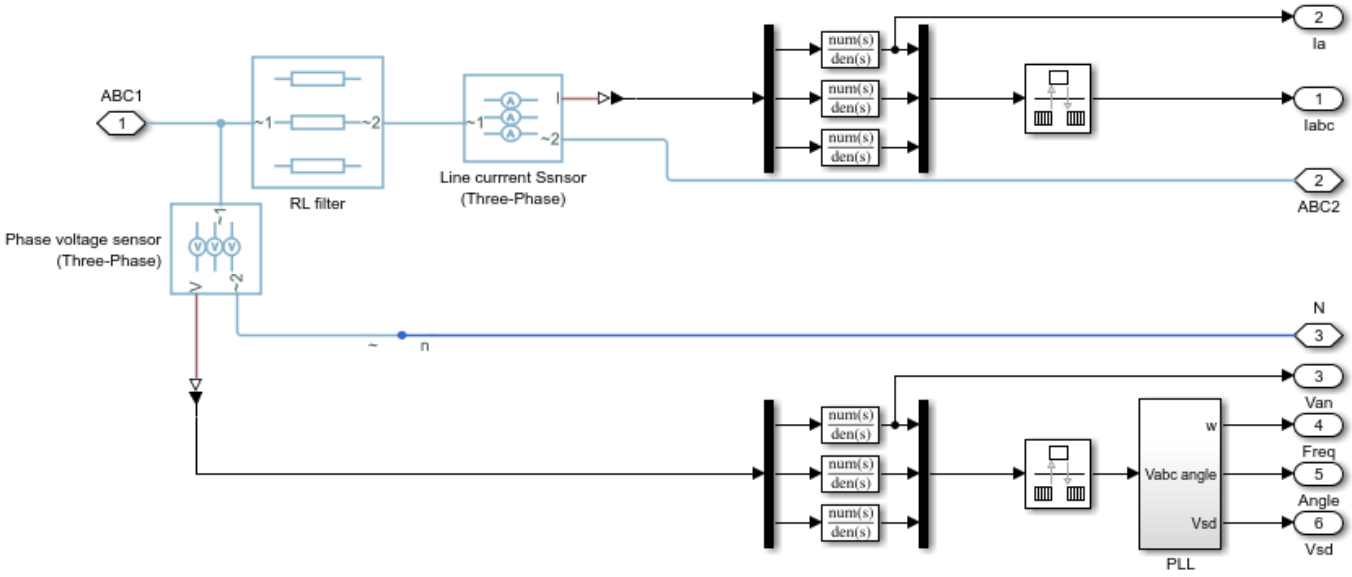



Copyright 2022-2022 The MathWorks, Inc.

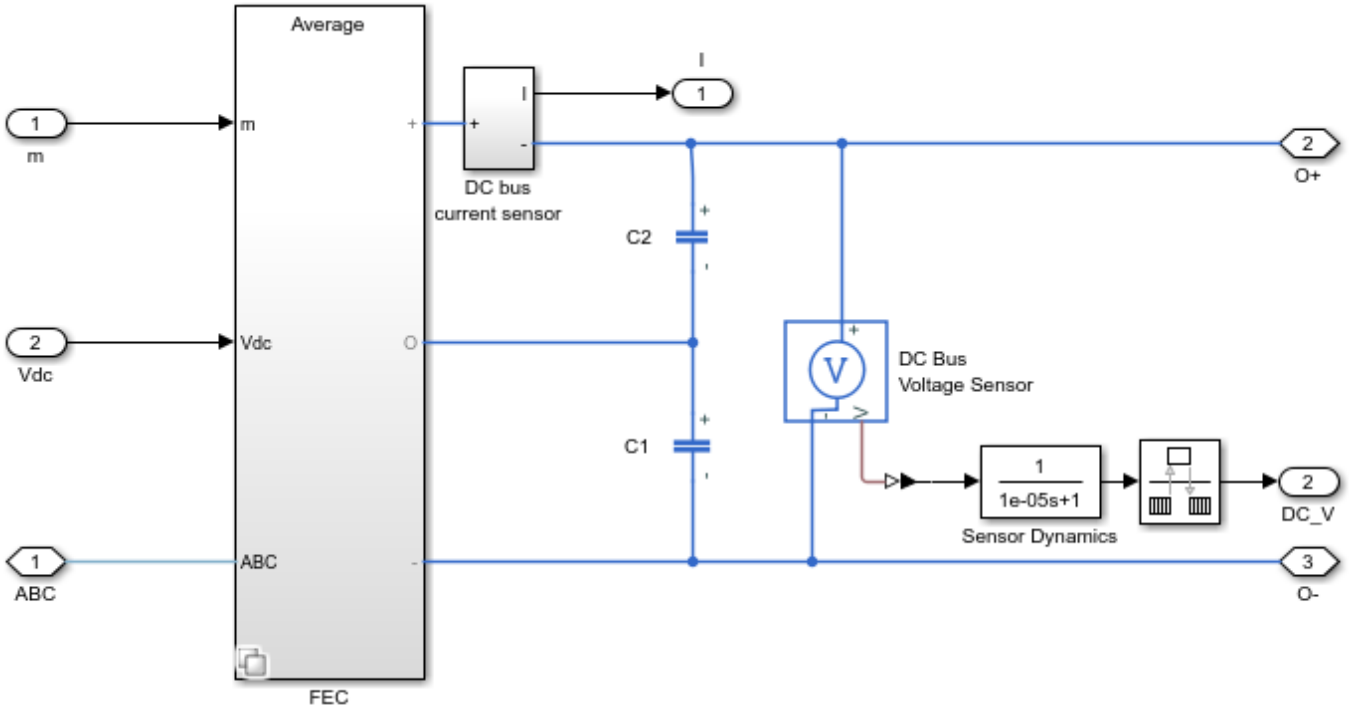
### Components in DC Fast Charging Station

These are the main components of the system:

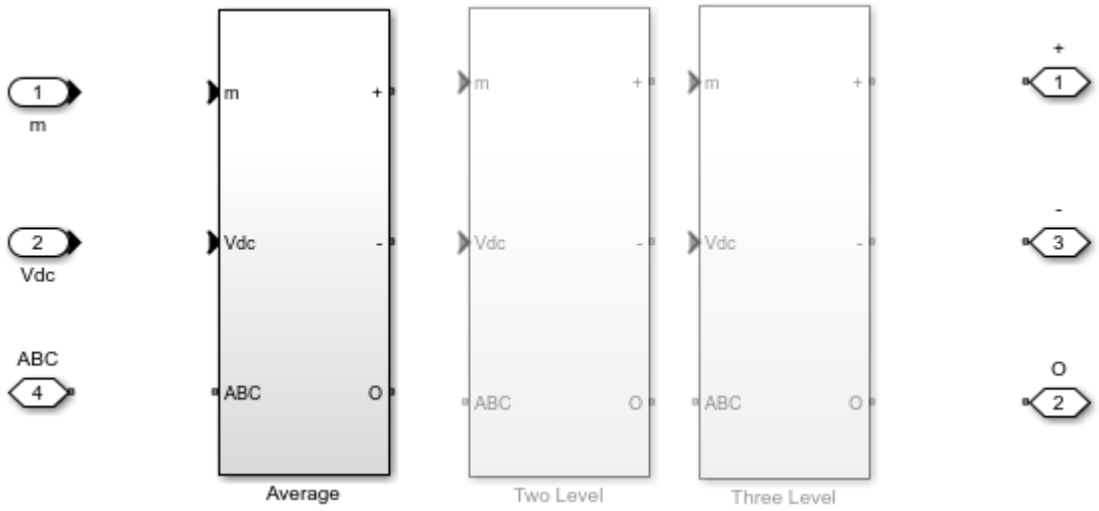
- Filter & AC Measurements to filter the harmonics in the line current and measure the three-phase supply voltage and line current.



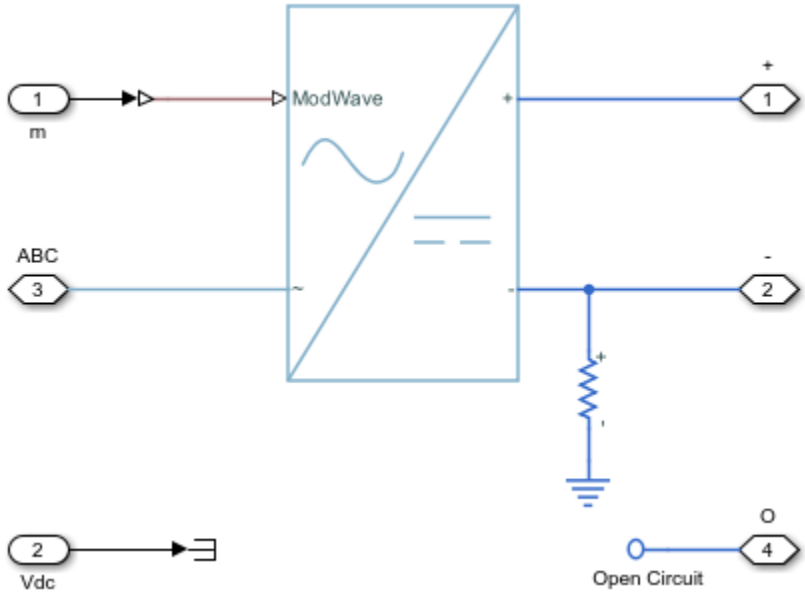

- Unity Power Factor (UPF) Front End Converter (FEC) to control output DC voltage at 800 V.



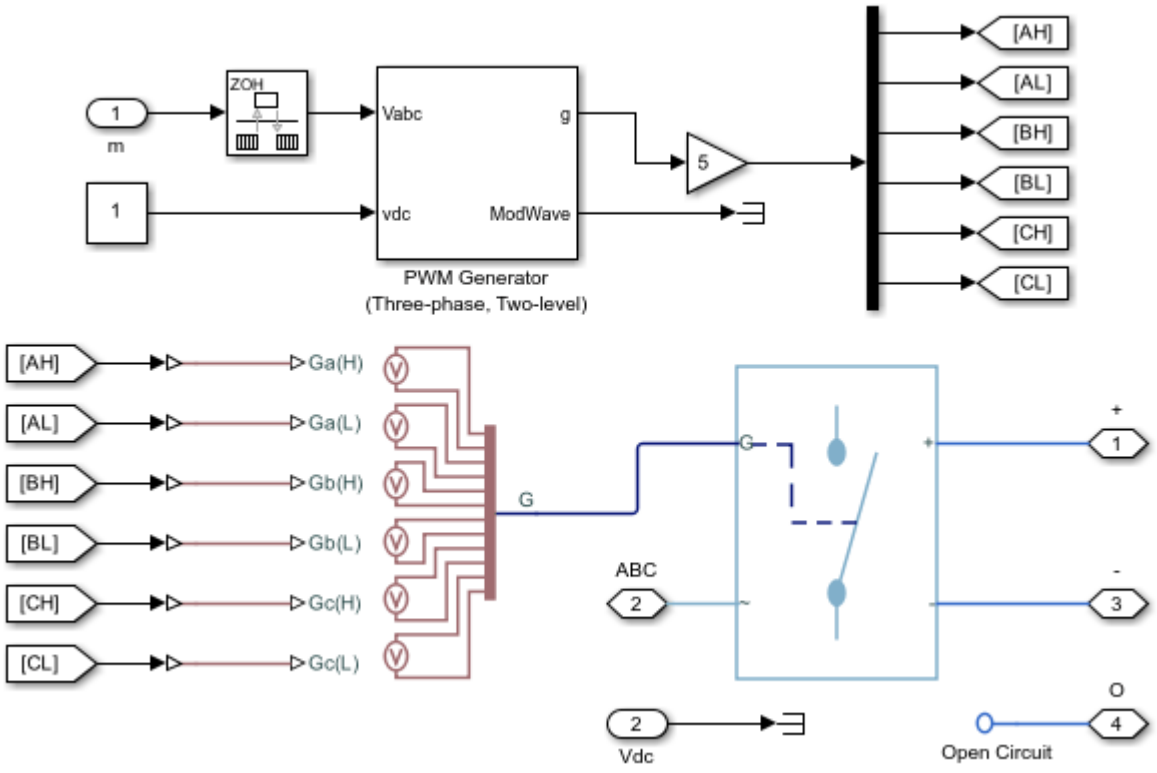
The converter circuit is modeled with three levels of fidelity:



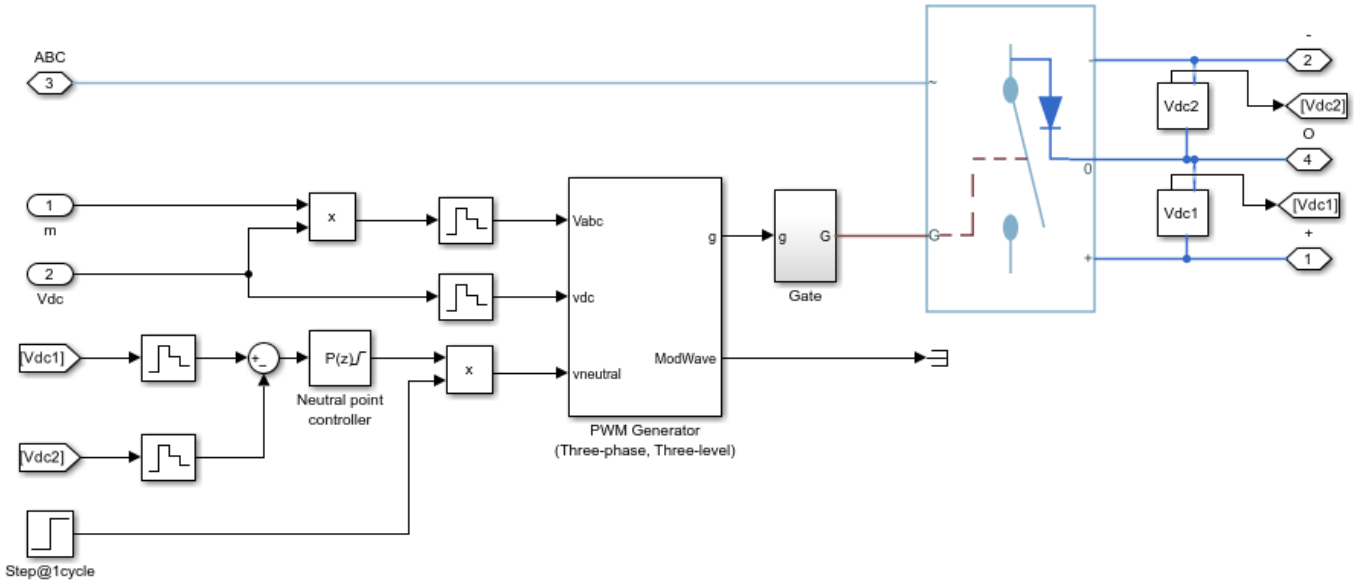

- Average Inverter Fidelity



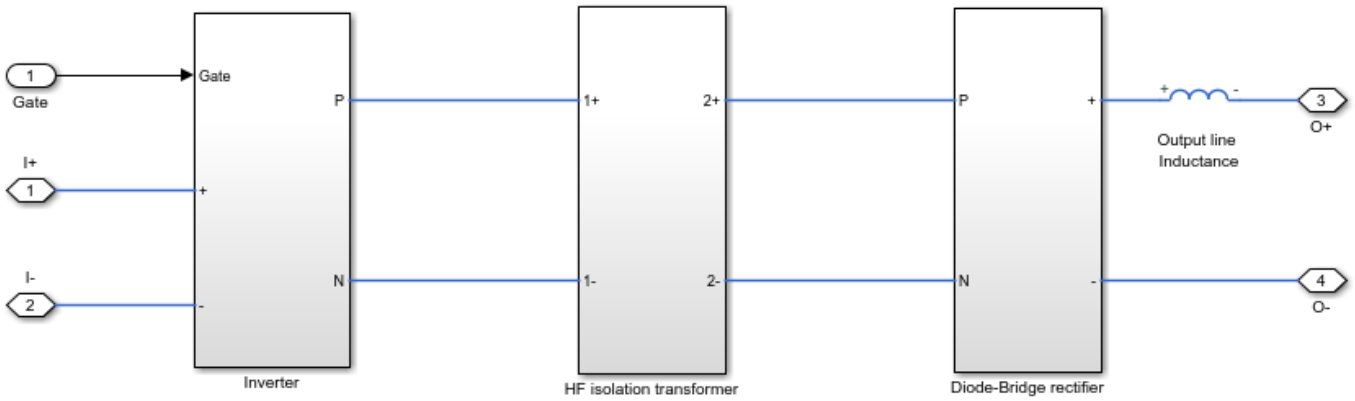

- Two Level Inverter Fidelity



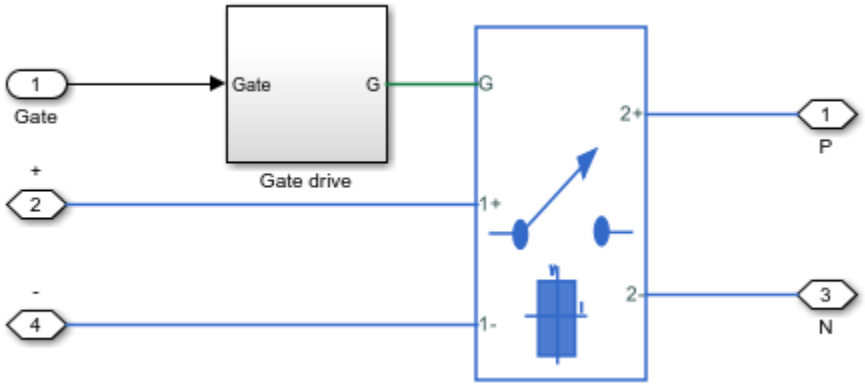

- Three Level Inverter Fidelity



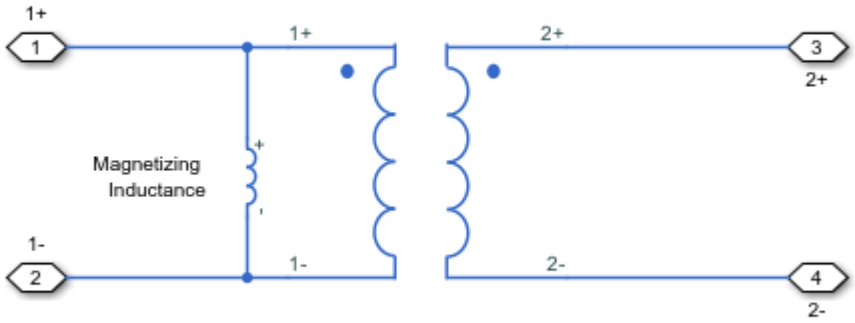

- Isolated DC-DC converter supply constant charging current to the EV battery.



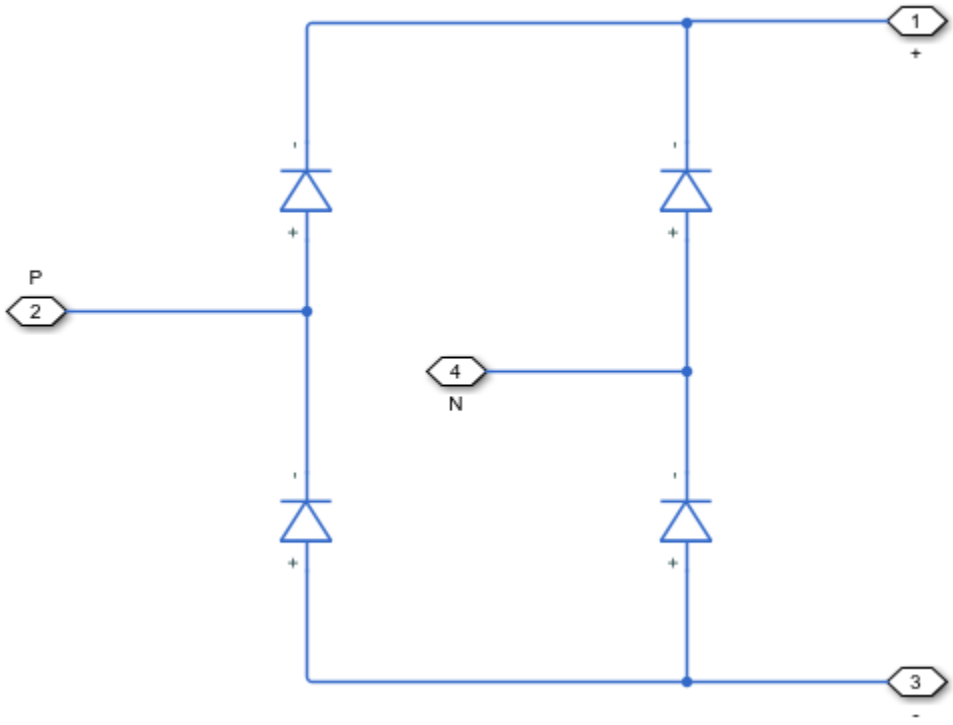

These are the main components of the isolated DC-DC Converter: * Inverter




- HF Isolation Transformer

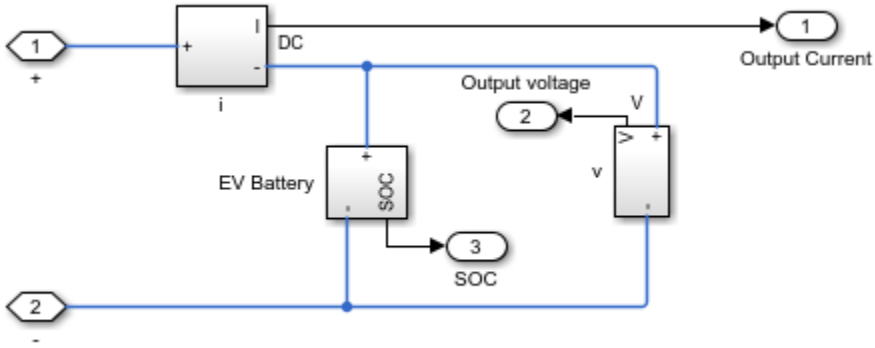



- Diode-Bridge Rectifier



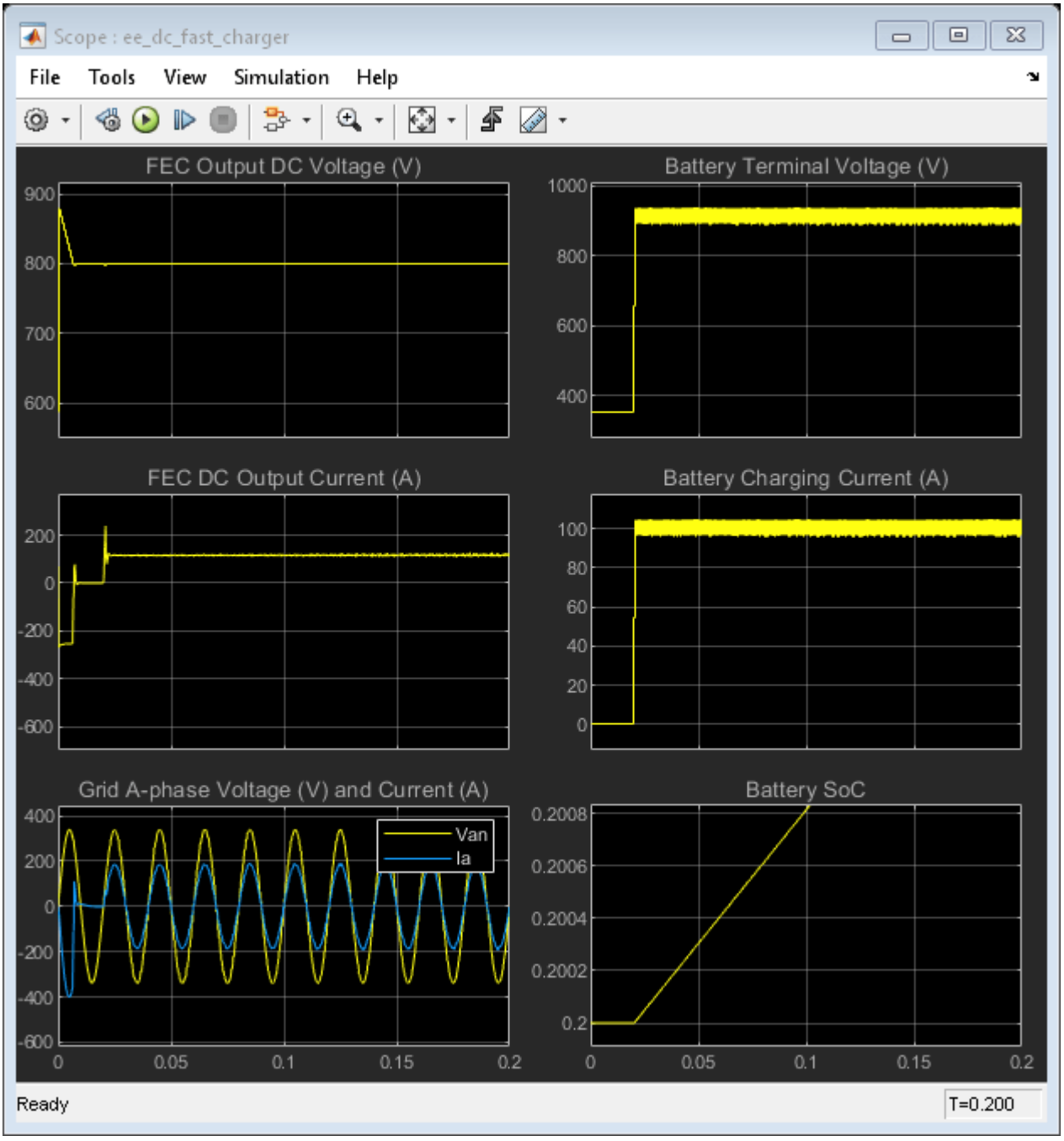

### EV Battery Pack Overview

The EV Battery Pack models the battery cells connected in series and the sensors to measure the battery terminal voltage and output current.




### Simulation Results from Simscape Logging

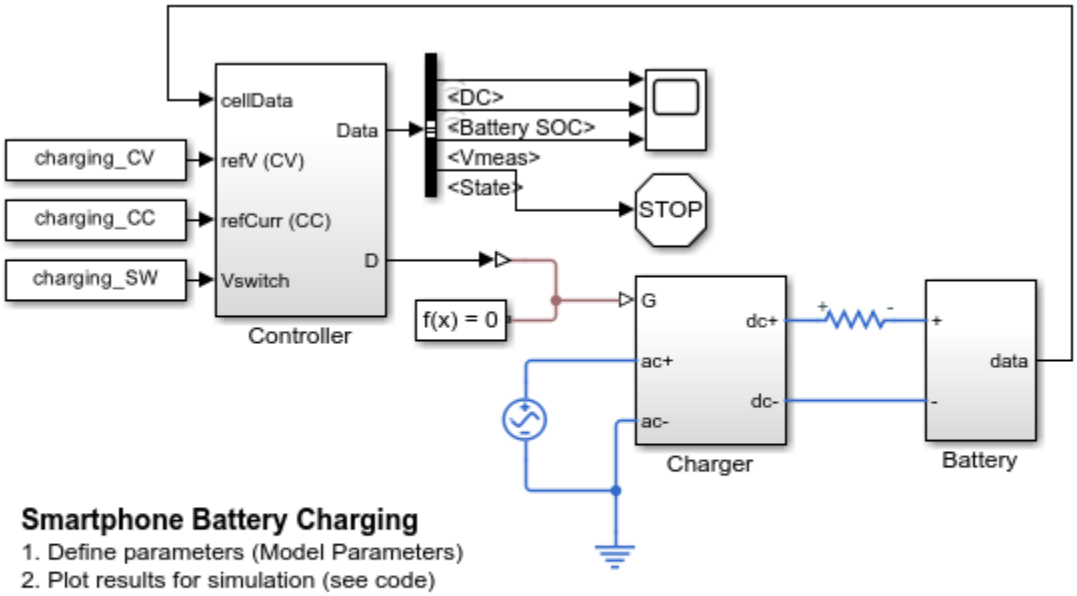
The plot below shows the DC bus voltage and current, battery terminal voltage, and charging current.


Scope : ee_dc_fast_charger
File Tools View Simulation Help
FEC Output DC Voltage (V)
900
800
700
600
Battery Terminal Voltage (V)
1000
800
600
400
FEC DC Output Current (A)
200
0
-200
-400
-600
Battery Charging Current (A)
100
80
60
40
20
0
Grid A-phase Voltage (V) and Current (A)
400
200
0
-200
-400
-600
Van
Ia
0
0.05
0.1
0.15
0.2
Battery SoC
0.2008
0.2006
0.2004
0.2002
0.2
0
0.05
0.1
0.15
0.2
Ready
T=0.200

# Smartphone Battery Charging

This example shows how to set up a constant current (CC) and constant voltage (CV) charging on a typical battery in a smartphone. A constant current starts charging the battery. When the battery voltage reaches a specific value, the constant voltage charging process starts.

### Model Overview

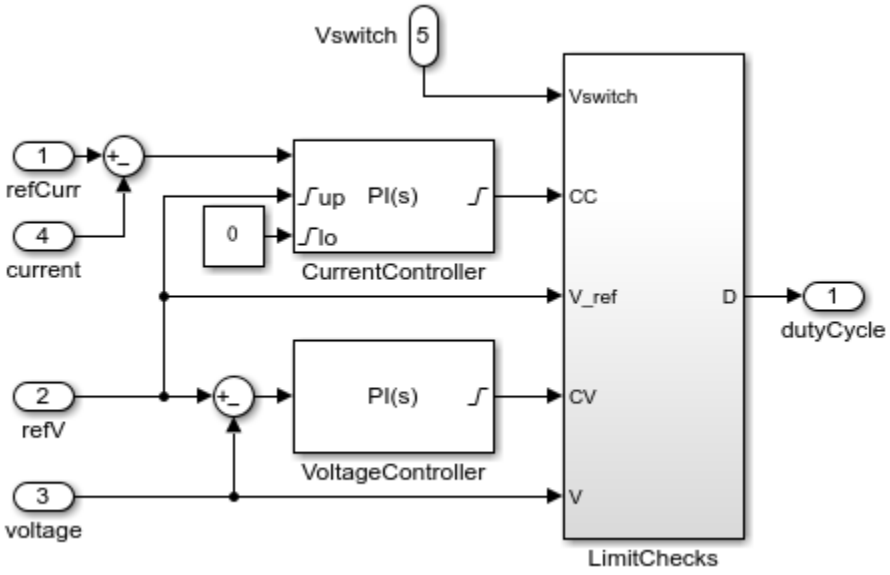



**Smartphone Battery Charging**

1. Define parameters (Model Parameters)
2. Plot results for simulation (see code)
3. Explore simulation results using Simscape Results Explorer
4. Learn more about this example

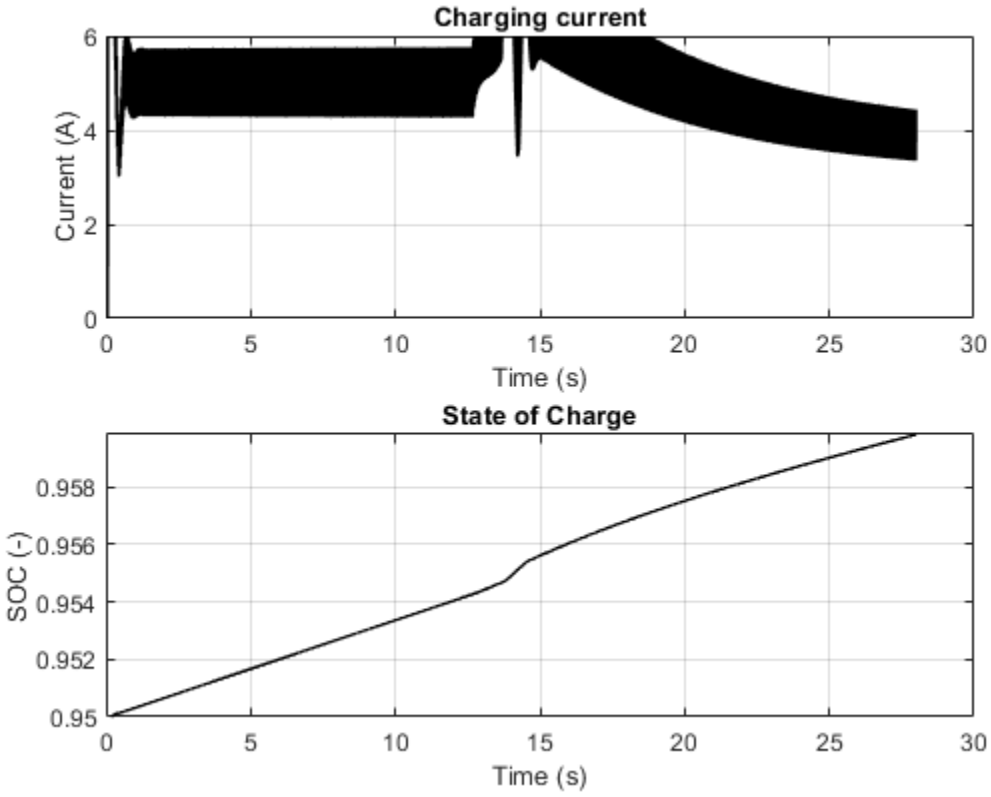
Copyright 2022 The MathWorks, Inc.

### Controller Subsystem




### Simulation Results from Simscape Logging

The plot below shows the battery charging current. First, the current is constant. Then it decreases as the constant voltage charging starts.


Charging current
Current (A)
Time (s)
State of Charge
SOC (-)
Time (s)

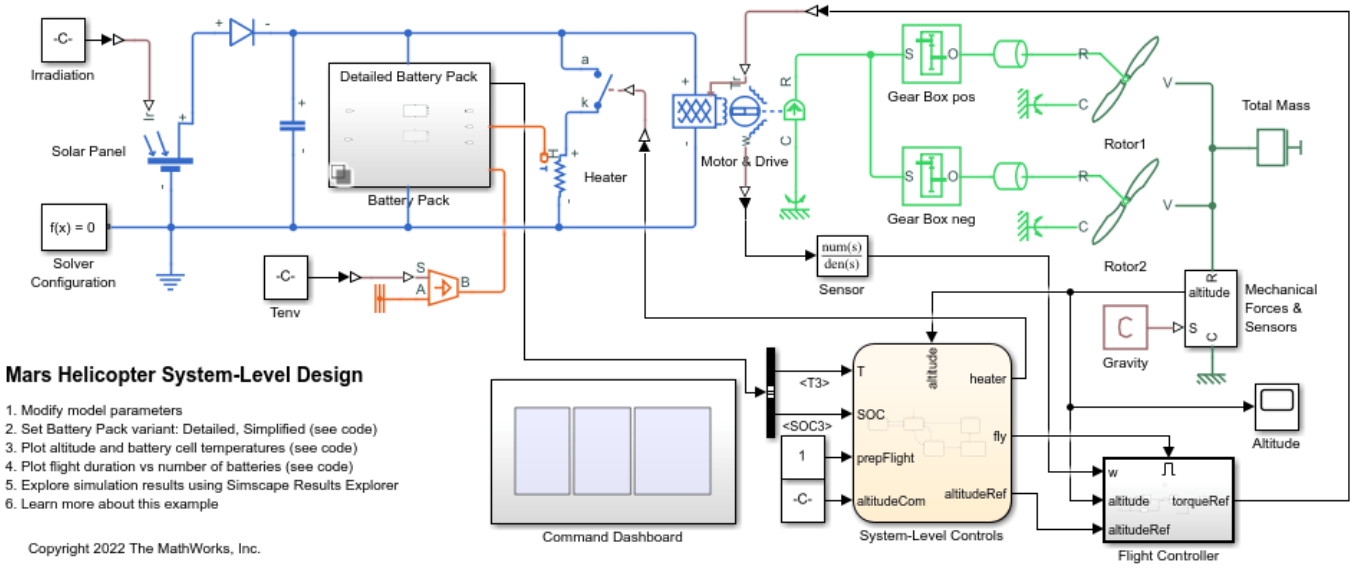
# Mars Helicopter System-Level Design

This example shows how to use Simscape™ Electrical™ to model a helicopter with coaxial rotors suitable to fly on Mars. This helicopter takes inspiration from `Ingenuity`, the robotic helicopter developed by NASA, which accomplished the first powered flight on another planet.

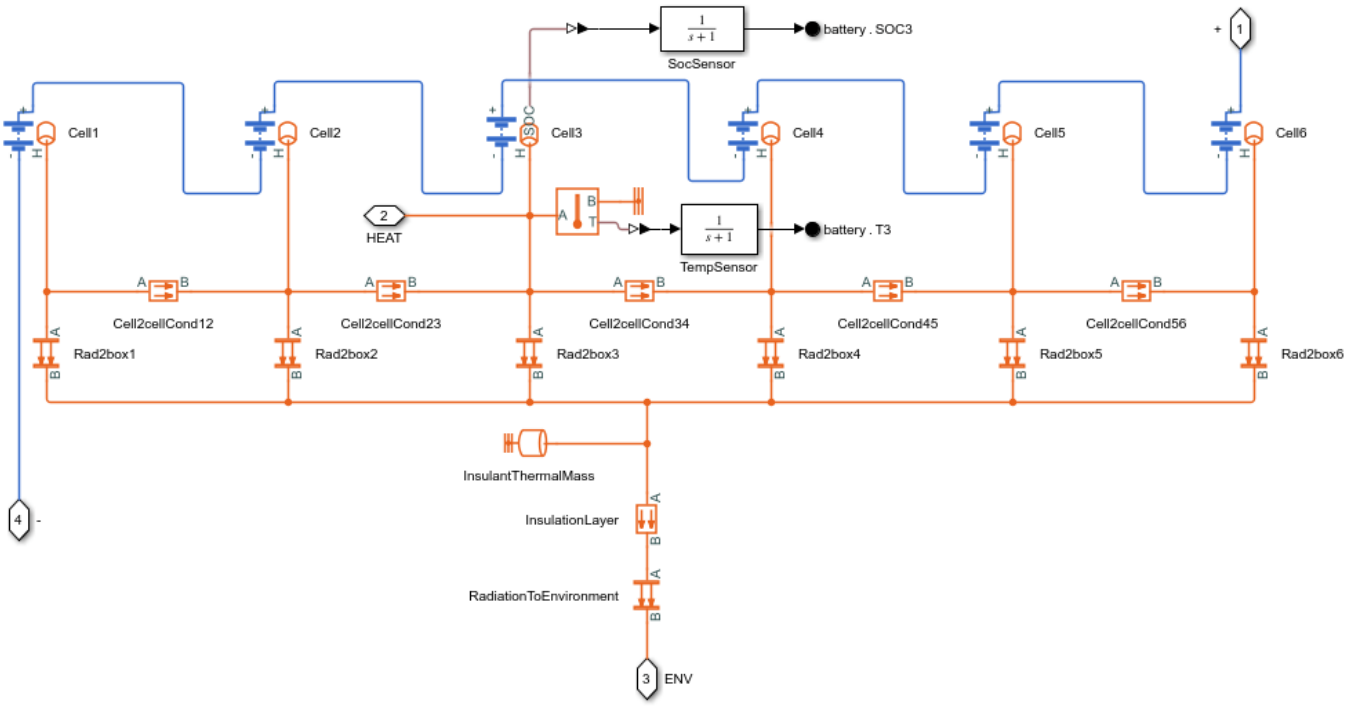
To control the helicopter altitude interactively, use the blocks in the Command Dashboard subsystem.

### Model Overview

The helicopter model comprises a Solar Panel, a Battery Pack, a Heater, a Motor & Drive, two gearboxes and two contra-rotating coaxial rotors, as well as a 1D mechanical model of the gravity, drag, mass, and ground contact forces.




The battery pack contains six Lithium-Ion cells connected in series. Only the central cell is instrumented. The Heater is attached to the central cell. The cells transfer heat to each other and to the insulating box, which dissipates heat to the martian environment.

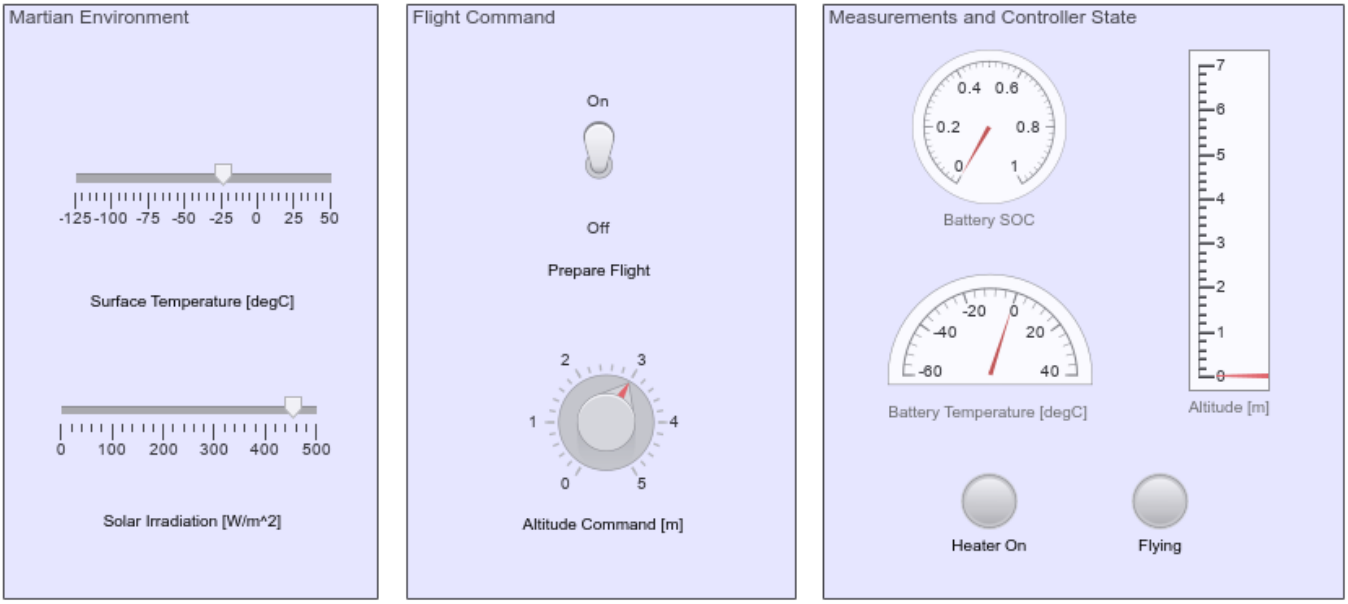



To correctly size the battery pack, you should consider the nominal voltage of the motor drive and the required nominal power, the nominal flight duration, the payload weight, and the battery energy to weight ratio. Additionally, the solar panel size and the average sun irradiation, together with the heat required to sustain the minimum temperature during the cold martian nights, determine the minimum and maximum energy that you can store during each daily cycle.

### Command Dashboard

To control the helicopter interactively during the simulation, use the Sliders, Switches, and Knob blocks in the Command Dashboard subsystem.

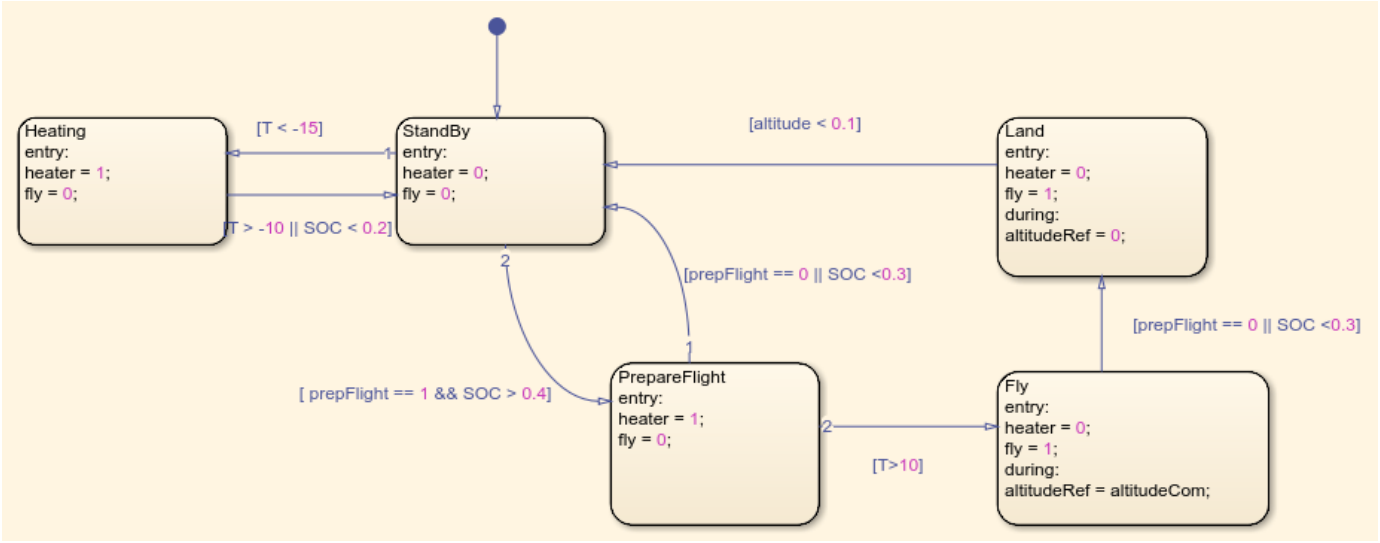
As the model solves the equations much faster than real time, the system enables simulation pacing to slow down the animations to only 30 times faster than real time.



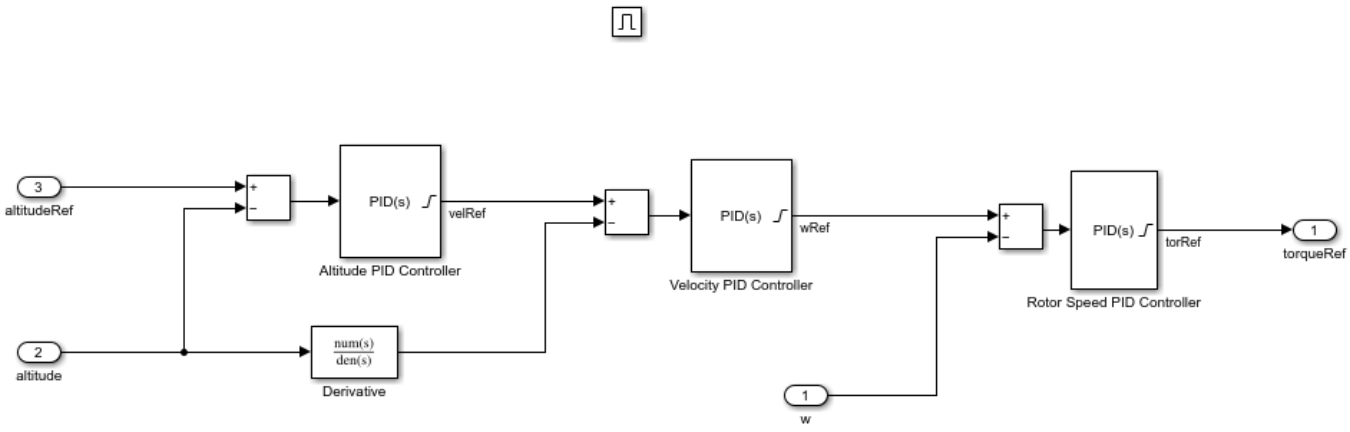

### Control System Overview

The helicopter needs to survive the cold martian nights, that can reach temperatures as low as -125 degC. The electronics and batteries cannot operate at such temperatures, so a heater keeps the temperatures in an acceptable range.

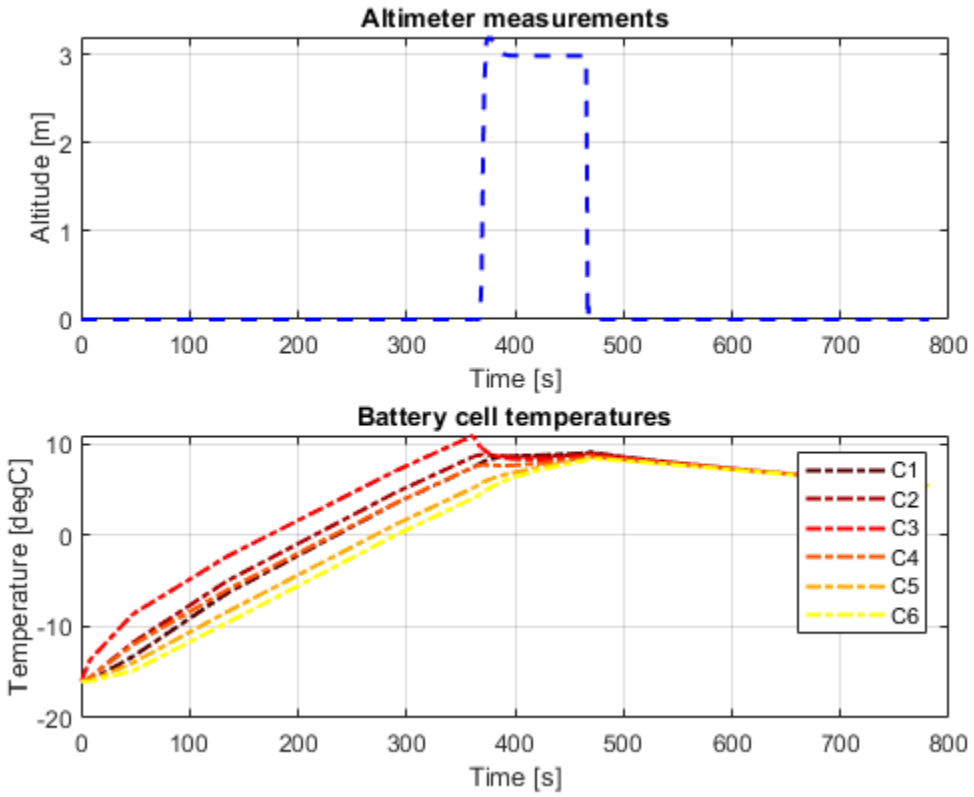
The rotors consume high amounts of power during flight, and the batteries can only provide such power above certain temperatures. The heater must be used to reach such temperatures before taking off.




Once the helicopter is ready to take off, the Flight Controller subsystem is enabled to control the altitude. It comprises three PIDs that control the rotor speed, the climb rate, and the altitude, respectively. The output is the reference torque for the Motor & Drive block.

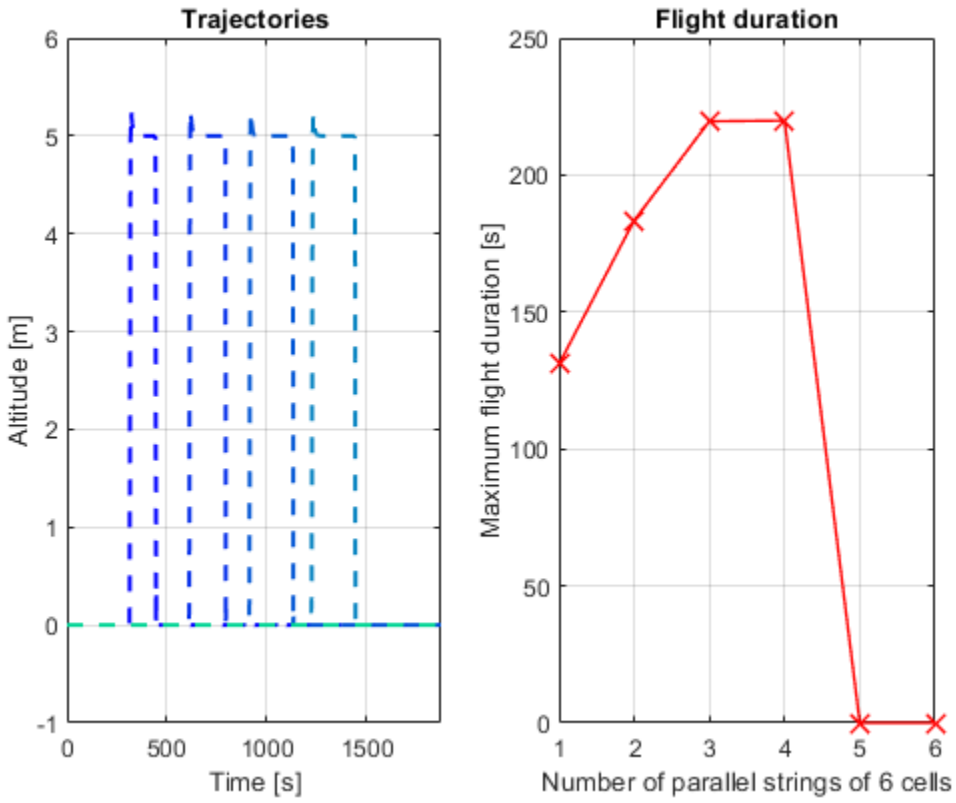



### Plot Altitude and Battery Cell Temperatures




### Plot Flight Duration Against Number of Battery Cells

The total number of batteries directly influences the maximum possible duration of the flight. You can add additional batteries to provide more energy storage, but this also increases the weight of the vehicle. Eventually, the motor is unable to provide enough power to lift the extra weight.


Trajectories
Altitude [m]
Time [s]
6
5
4
3
2
1
0
-1
0
500
1000
1500
Flight duration
Maximum flight duration [s]
Number of parallel strings of 6 cells
250
200
150
100
50
0
1
2
3
4
5
6

# Overcurrent Relay Protection in AC Microgrid

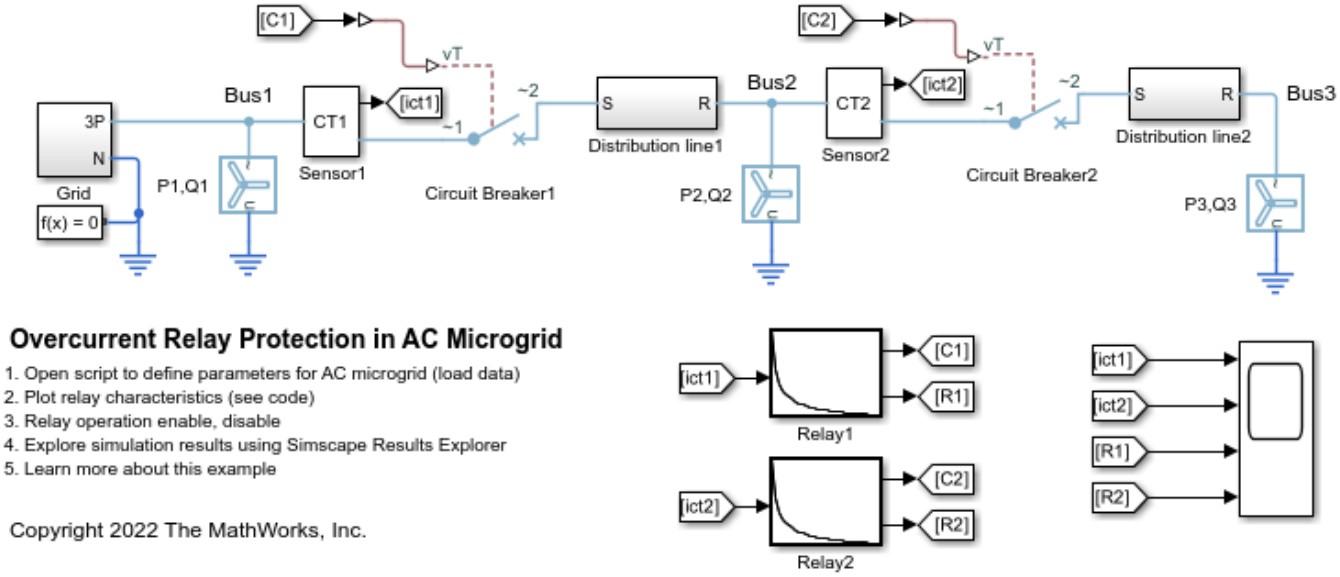
This example shows how to model an overcurrent relay in an AC microgrid. You can use this example to study overcurrent relay coordination in a microgrid. The Relay block comprises two protection units, phase protection and earth protection. The phase protection unit protects the microgrid from high phase currents. The earth protection unit protects the microgrid from high earth currents. In this example the **relay2** block protects the **distribution_line2** block. The **relay1** block protects the **distribution_line1** block and also acts a back-up for the **relay2** block. If a fault occurs on the **distribution_line2** block and the **relay2** block doesn't operate, the **relay1** block operates after a specified time and isolates the system.To avoid tripping of the system, the **relay1** and **relay2** blocks operate such that only one relay operates at any given time. You can specify either time multiplier setting or the desired operating time of **relay2** block.

### Microgrid Overview

The figure below shows an AC microgrid with a source, transformer, distribution lines, current transformers, circuit breakers, overcurrent relays, and loads.

The microgrid is connected to the grid at 132 kV. A three-phase transformer steps down the source voltage from 132 kV to 33 kV.

The current transformer blocks step down the current within the operating range of the overcurrent relays. The **relay1** and **relay2** blocks send trip signals to the **circuitBreaker1** and **circuitBreaker2** blocks, respectively.




### Current Transformer Overview

In this example, the current transformer comprises three single-phase transformer blocks. A 1 ohm resistor is placed in series with the current transformer secondary winding and a voltage sensor measures the voltage across the resistor. This voltage is equal to the current flowing through the resistor. This example determines the turns ratio of the current transformers based on the load currents flowing through them and the safety factor. If a load current of 60A flows through the

current transformer primary winding and the safety factor is equal to 0.25, the current transformer block ratio is equal to 100/1. You can specify the safety factor as an input. The secondary currents of the current transformer are inputs to the **relay1** and **relay2** blocks, respectively.

### Overcurrent Relay Block Overview

The relay block comprises the two protection units, phase protection unit and earth protection unit. When the value of the current in any of the phases is greater than the pick up value, the phase protection unit triggers the trip signal. When the value of the zero sequence current is greater than the pick up value, the earth protection unit triggers the trip signal. When one of the units triggers the trip signal, the relay sends the trip signal to the circuit breaker with a time delay, called relay operating time. The relay operating time depends on the pick up value and the time multiplier setting of the relay. The Relay block has a total of 8 operating characteristics including standard IEC and IEEE characteristics, Definite time, and IvsT. You can calculate the relay operating time for standard IEC and IEEE characteristics by using this equation:

$$Top = TMS * [\frac{a}{PSM^b - 1} + c]$$

where TMS is the time multiplier setting, PSM is the plug setting multiplier, and a, b, and c are constants. If you select Definite time, directly specify the relay operating time. If you select IvsT characteristic, specify the operating time of the relay as a two dimensional matrix for different current values and time multiplier settings. The relay algorithm calculates the operating time for the specified time multiplier setting and pick up value.

### Overcurrent Relay Settings

There is always a time delay between the operating time of the **relay1** and **relay2** blocks. Set the time delay by specifying the protection.tFactor in the `ee_relay_overcurrent_protection_input` MLX file. For example, if a fault occurs in the **distributionLine2** block and the **relay2** block fails to isolate it, the **relay1** block sends the trip signal to the **circuitBreaker1** block after a minimum time delay equal to the value you specified in the **protection.tFactor** parameter. If you select standard IEC or IEEE characteristics, the system estimates the **relay1** time settings based on the time settings of the **relay2** block. If you specify the time multiplier setting of the **relay2** block, the system estimates the time multiplier setting of the **relay1** block. If you specify the operating time of the **relay2** block, the system estimates both the time multiplier setting of the **relay2** and **relay1** blocks. A maximum short circuit current in the **distributionLine2** block happens when a fault occurs at the start of the distribution line.

In this example, the phase current value is maximum during phase-to-ground (L-G) fault and the earth current is maximum during two-phase-to-ground(L-L-G) fault. To estimate the relay time settings, L-G fault is considered for phase protection and L-L-G fault is considered for earth protection. This equation calculates the fault current in per-unit during the L-G fault:

$$I = \frac{3}{Z0 + Z1 + Z2}$$

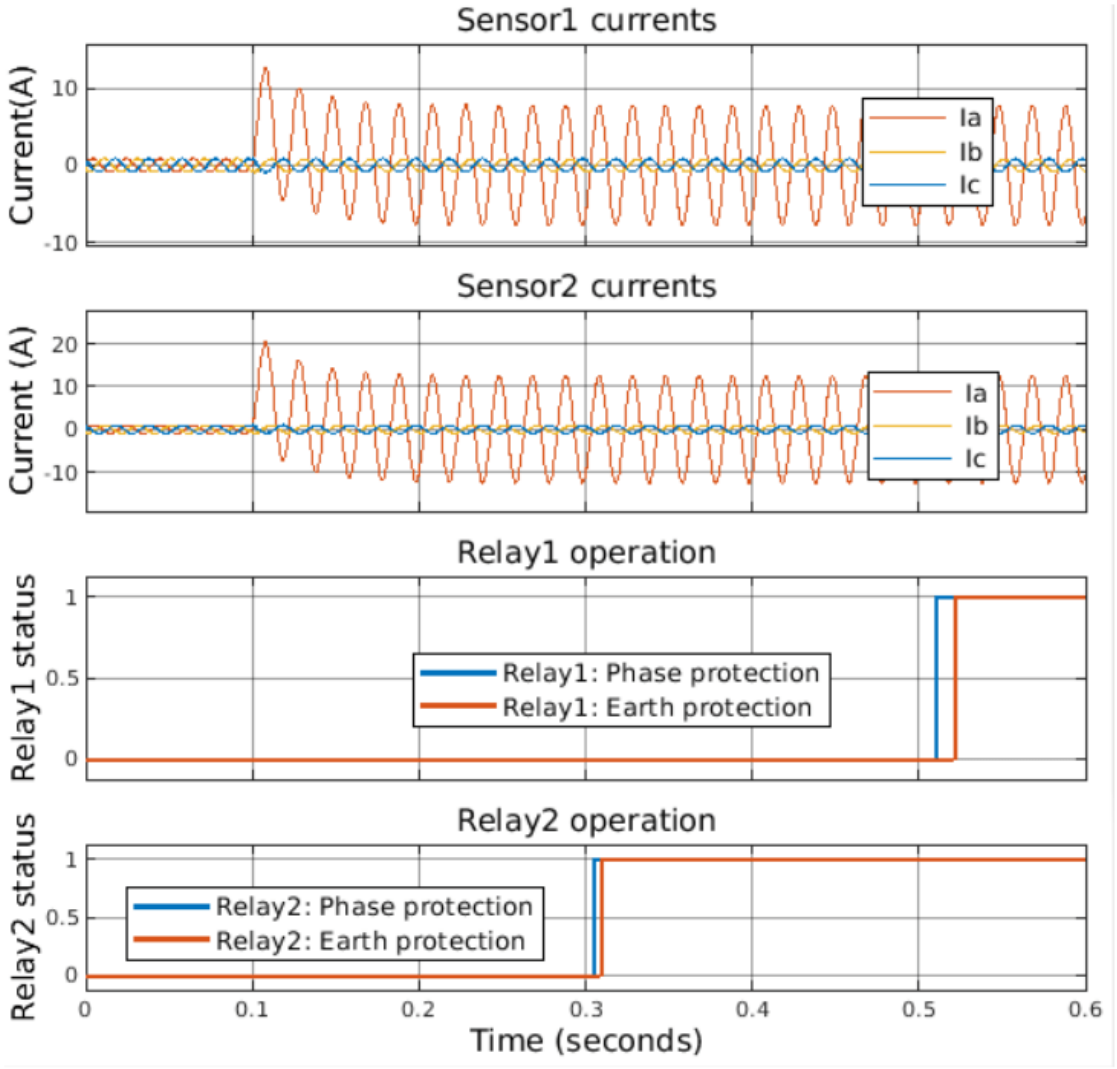
where Z0, Z1, and Z2 are the zero sequence, positive sequence, and negative sequence impedances. This equation calculates the zero sequence current during the L-L-G fault:

$$I = \frac{Z2}{Z1Z2 + Z1Z0 + Z2Z0}$$

where Z0, Z1, and Z2 are the zero sequence, positive sequence, and negative sequence impedances.

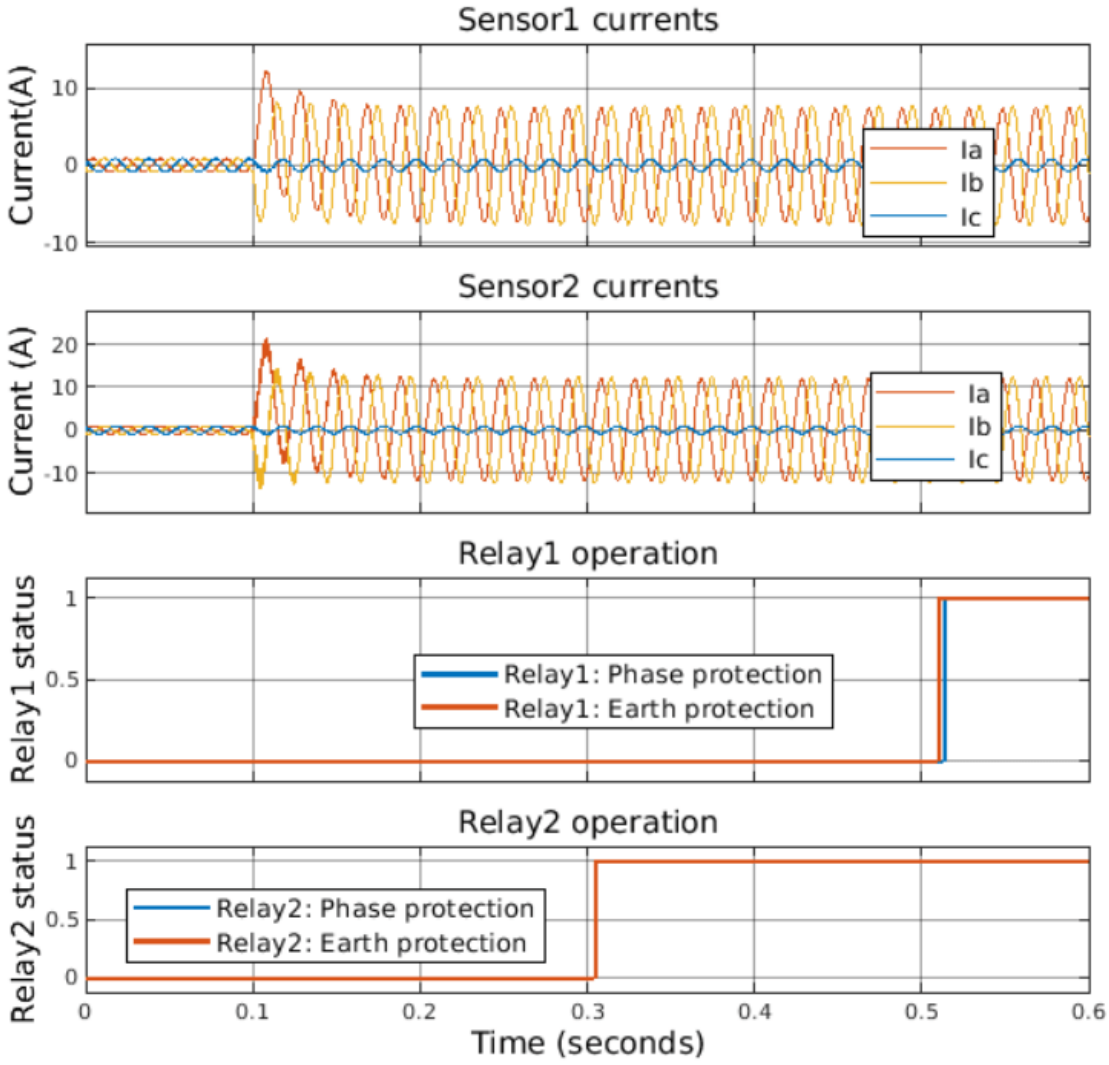
**Relay Operating Time for a Phase to Ground(L-G) Fault**

The figure below shows the current transformer secondary currents and the operating time of the **relay1** and **relay2** blocks for an L-G fault at 0.1 km on the **distributionline2** block. The fault occurs at 0.1 s. IEEE moderately inverse characteristics are selected for the phase protection and earth protection units for the **relay1** and **relay2** blocks. The operating time of the **relay2** block is equal to 0.2 s. The time grading factor between the **relay1** and **relay2** blocks is equal to 0.2 s. The pick up values for phase protection of the relays are equal to 1 A. The pick up values for earth protection of the relays are equal to 0.05 A. To observe the operating time of the **relay1** block, the circuit breaker protection is disabled.



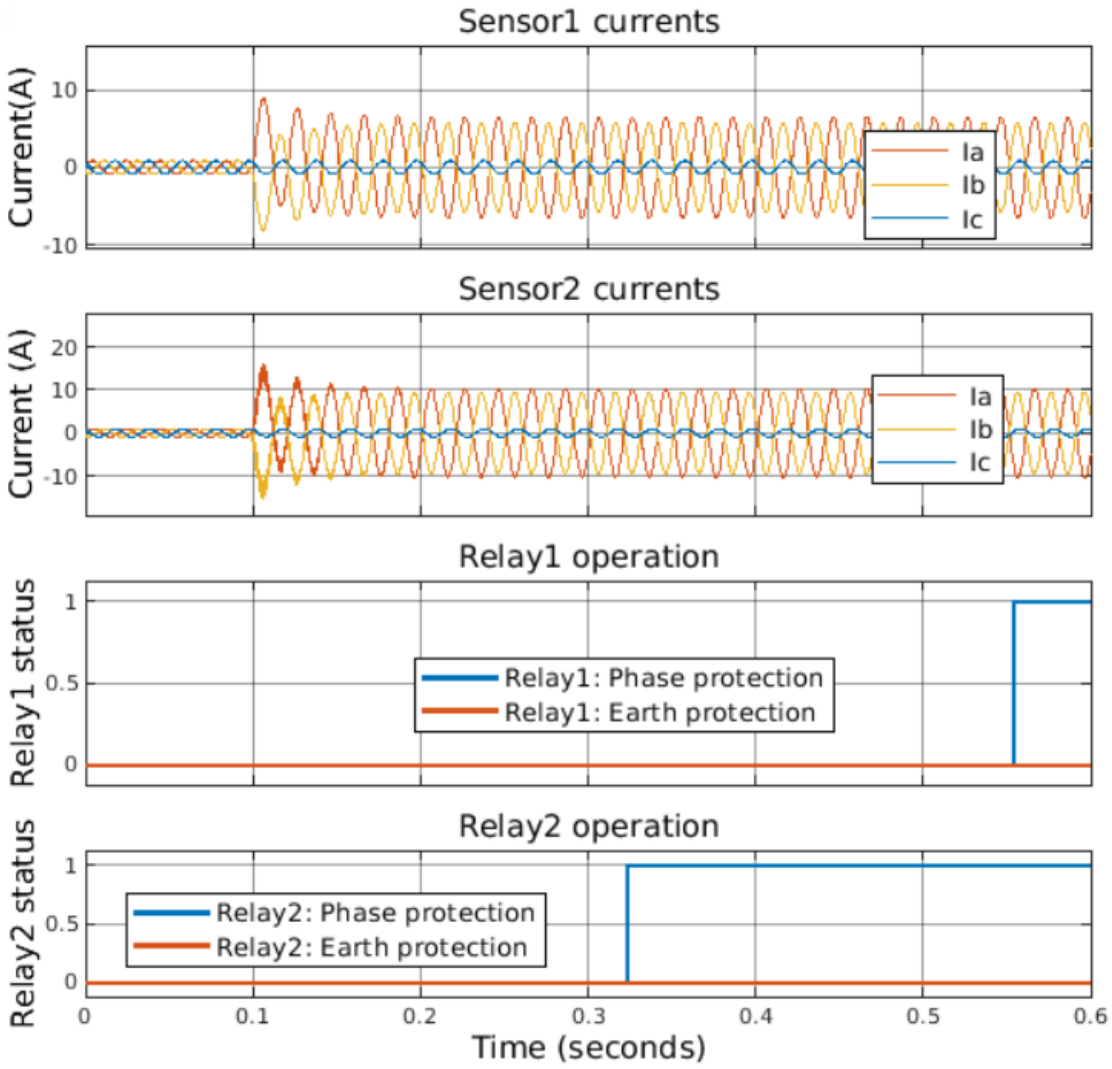
### Relay Operating Time for a Two-Phase to Ground(L-L-G) Fault

The figure below shows the current transformer secondary currents and the operating time of the **relay1** and **relay2** blocks for an L-L-G fault at 0.1 km on the **distributionline2** block. The fault occurs at 0.1 s. IEEE moderately inverse characteristics are selected for the phase protection and earth protection units for the **relay1** and **relay2** blocks. The operating time of the **relay2** block is equal to 0.2 s. The time grading factor between the **relay1** and **relay2** blocks is equal to 0.2 s. The pick up values for phase protection of the relays are equal to 1 A. The pick up values for earth protection of the relays are equal to 0.05 A. To observe the operating time of the **relay1** block, the circuit breaker protection is disabled.



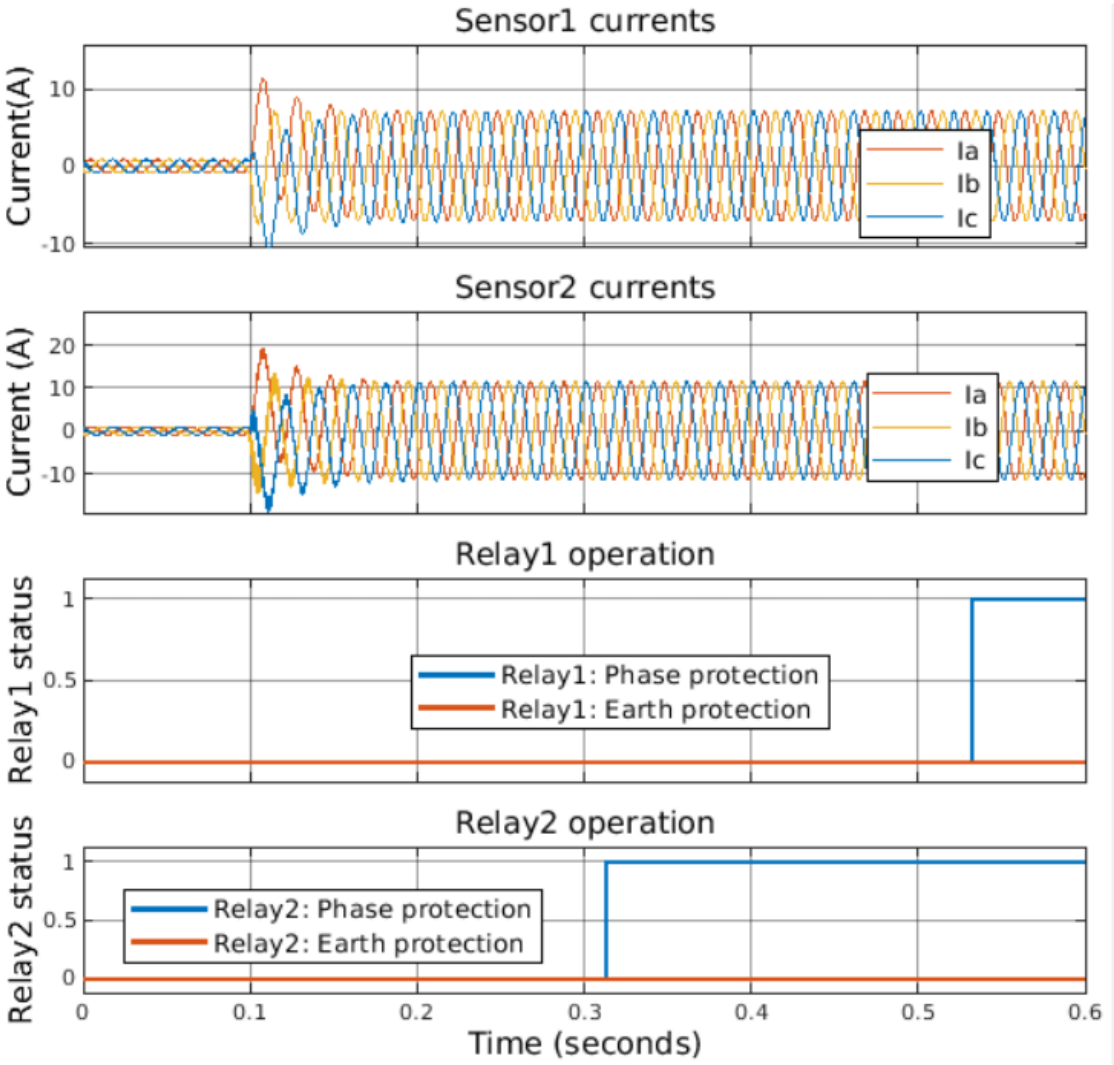
### Relay Operating Time for a Two-Phase(L-L) Fault

The figure below shows the current transformer secondary currents and the operating time of the **relay1** and **relay2** blocks for an L-L fault at 0.1 km on the **distributionline2** block. The fault occurs at 0.1 s. IEEE moderately inverse characteristics are selected for the phase protection and earth protection units for the **relay1** and **relay2** blocks. The operating time of the **relay2** block is equal to 0.2 s. The time grading factor between the **relay1** and **relay2** blocks is equal to 0.2 s. The pick up values for phase protection of the relays are equal to 1 A. The pick up values for earth protection of the relays are equal to 0.05 A. To observe the operating time of the **relay1** block, the circuit breaker protection is disabled.



### Relay Operating Time for Three-Phase(L-L-L) Fault

The figure below shows the current transformer secondary currents and the operating time of the **relay1** and **relay2** blocks for an L-L-L fault at 0.1 km on the **distributionline2** block. The fault occurs at 0.1 s. IEEE moderately inverse characteristics are selected for the phase protection and earth protection units for the **relay1** and **relay2** blocks. The operating time of the **relay2** block is equal to 0.2 s. The time grading factor between the **relay1** and **relay2** blocks is equal to 0.2 s. The pick up values for phase protection of the relays are equal to 1 A. The pick up values for earth protection of the relays are equal to 0.05 A. To observe the operating time of the **relay1** block, the circuit breaker protection is disabled.



### Relay Parameters Configuration

- **Relay characteristic for phase protection** — Standard IEC and IEEE operating characteristics are included in the **relay** block. Choose between IEC normal inverse, IEC extremely inverse, IEC very inverse, IEEE moderately inverse, IEEE extremely inverse, IEEE very inverse, Definite time, or IvsT.
- **Relay characteristic for earth protection** — Standard IEC and IEEE operating characteristics are included in the **relay** block. Choose between IEC normal inverse, IEC extremely inverse, IEC very inverse, IEEE moderately inverse, IEEE extremely inverse, IEEE very inverse, Definite time, or IvsT.
- **Fundamental frequency (Hz)** — Rated frequency, in Hz. This value must be greater than zero.
- **Relay sample time (s)** — Relay sampling time, in seconds.
- *Relay operation enable — Whether to enable or disable the circuit breaker operation. Set this value to 1 to enable relay operation or 0 to disable relay operation.
- **Pick up value for phase protection (A)** — Minimum current to trigger phase protection unit, in Ampere.
- **Pick up value for earth protection (A)** — Minimum current to trigger earth protection unit, in Ampere.
- **Time multiplier setting for phase protection (s)** — Time delay setting for phase protection unit, in seconds.
- **Time multiplier setting for earth protection (s)** — Time delay setting for earth protection unit, in seconds.
- **Relay operating time for phase protection (s)** — Time after which relay sends trip signal to circuit breaker, in seconds (To enable this parameter, set **Relay characteristic for phase protection** to Definite time).
- **Relay operating time for earth protection (s)** — Time after which relay sends trip signal to circuit breaker, in seconds (To enable this parameter, set **Relay characteristic for earth protection** to Definite time).
- **Current setting for phase protection (A)** — Current values for the lookup table, in Ampere (To enable these parameters, set **Relay characteristic for phase protection** to IvsT Characteristics).
- **Time multiplier setting for phase protection (s)** — Time settings for the lookup table, in seconds.
- **Operating time for phase protection (s)** — Relay operating time for phase protection, in seconds.
- **Current setting for earth protection (A)** — Current values for the lookup table, in Ampere (To enable these parameters, set **Relay characteristic for earth protection** to IvsT Characteristics).
- **Time multiplier setting for earth protection (s)** — Time settings for the lookup table, in seconds.
- **Operating time for earth protection (s)** — Relay operating time for earth protection, in seconds.

# Distance Relay Protection in AC Microgrid

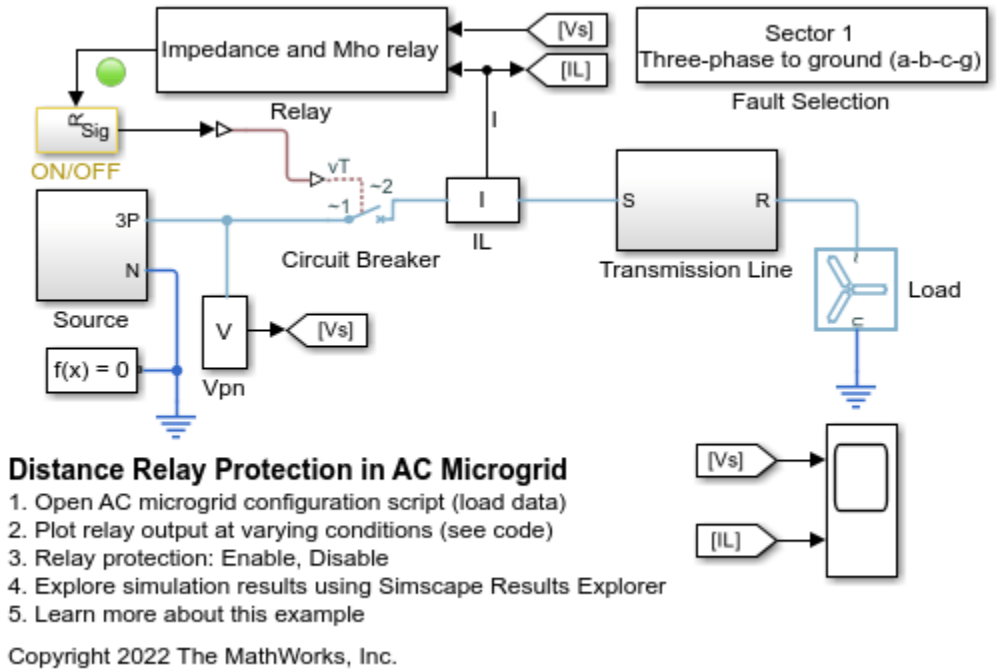
This example shows how to model a distance relay in an AC microgrid. The relay block comprises impedance relay characteristic and mho relay characteristic. You can use this example to study the performance of impedance relay and mho relay in various fault conditions. Both the relays have two types of relays for ground fault and phase-phase fault.

The impedance relay comprises a directional unit and a blinder unit. The directional unit ensures the tripping in a specified direction, and the blinder unit avoids malfunction of the relay during load encroachment.

### AC Microgrid Overview

The figure below shows an AC microgrid with a distance relay and a circuit breaker. The microgrid generates electricity at a voltage level of 11 kV. A three-phase transformer steps down the source voltage to 400 V.

To isolate the faulty section of the transmission line, relay and circuit breaker are connected next to the step-down transformer. An inductive load of 30 kW with a lagging power factor of 0.95 is connected at the receiving end of the transmission line.




### Relay Block Overview

The relay block comprises the two relay characteristics, impedance and mho relay. Relay characteristic has three zones based on the reach of the relay. A Fourier block estimates the fundamental voltage and current signals.

The relay estimates the line impedance from the circuit breaker to the fault points. When the estimated impedance is less than the reach of the first zone, the relay triggers a trip signal without

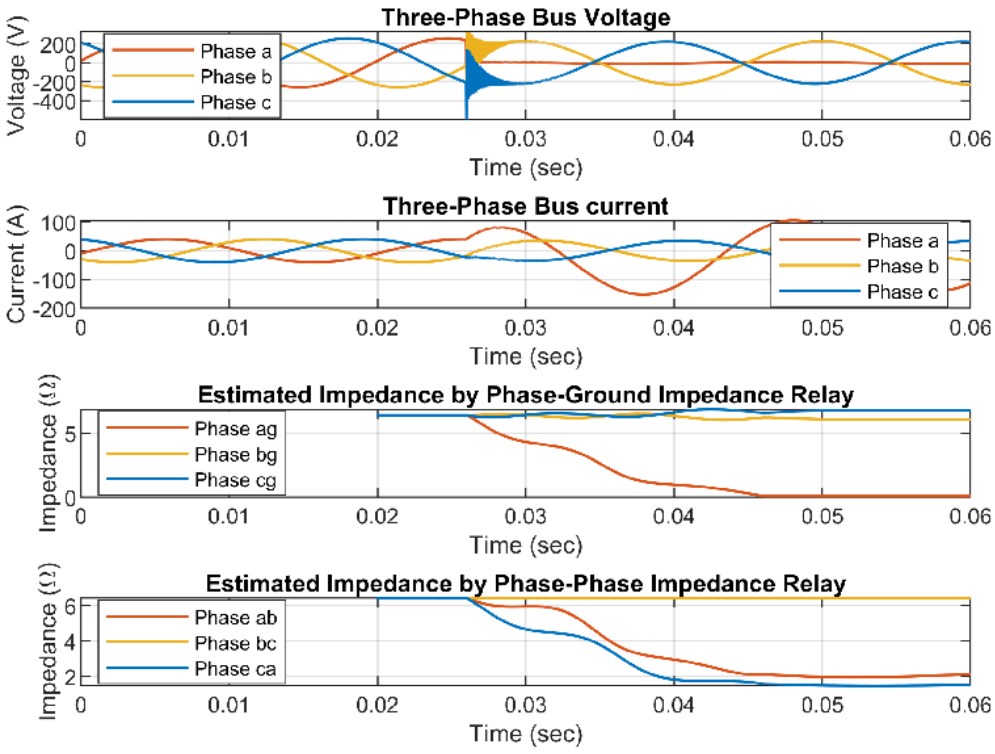
any time delay. If the estimated impedance is less than the reach of the second zone, the relay triggers a trip signal after a finite time delay. When the estimated impedance becomes less than the third zone reach, the relay triggers the trip signal after a time delay which must be greater than the time delay for the second zone.

To avoid tripping in the reverse direction of the fault, the impedance relay consists of a directional unit. You can set the direction of the impedance relay based on your requirements. The impedance relay does not trip if the estimated impedance falls in the blinder region.

This example also implements a mho relay with inherited directional unit. The mho relay uses self polarizing parameter like phase and line voltages.

### Plot for Voltage, Current, and Estimated Impedance

The plot below shows the variation of the estimated impedance with respect to time. A single-phase to ground (a-g) fault is initiated in Sector 1 of the transmission line with a fault initiation time of 0.056 sec. To estimate the fundamental component of voltage and current signal, this example uses full cycle Fourier analysis.

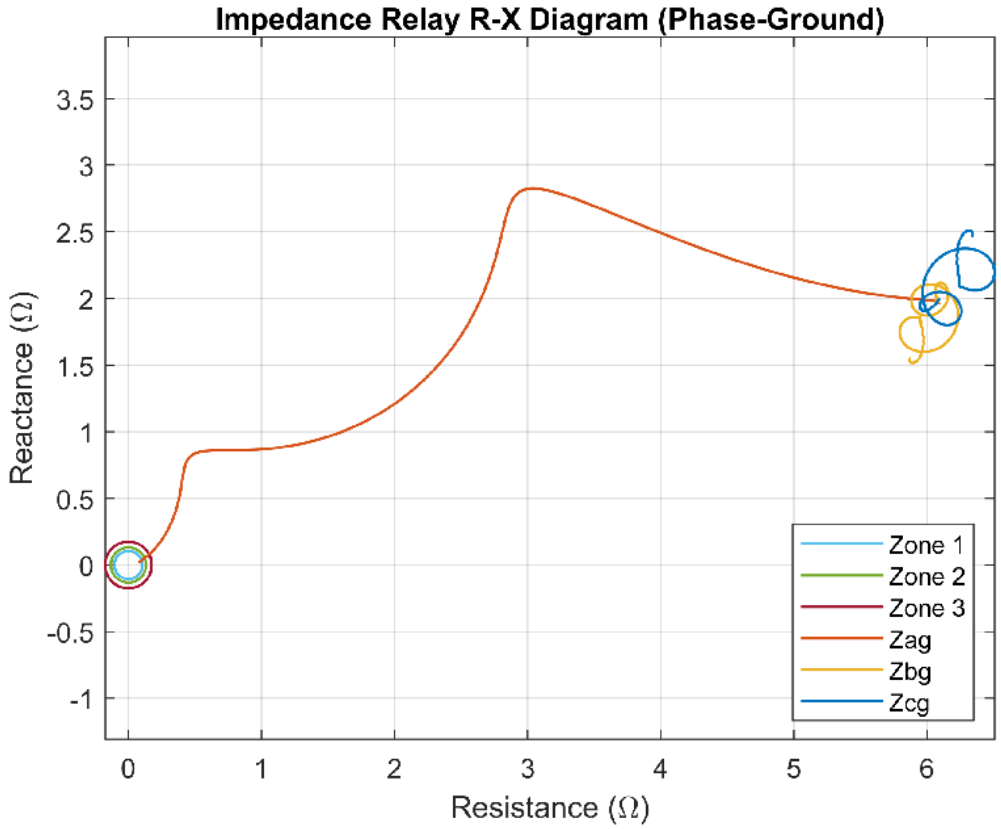



### Single-Phase to Ground (a-g) Fault

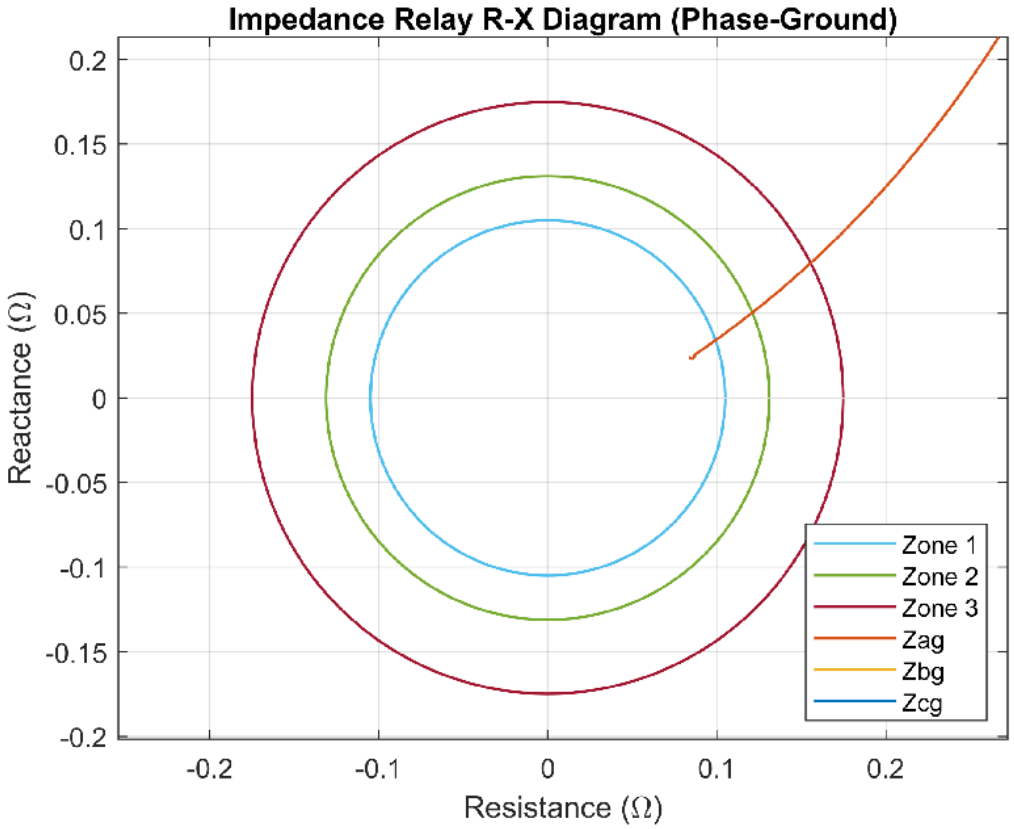
A single-phase to ground (a-g) fault is initiated in Sector 1 of the transmission line.

### R-X Diagram of Phase-Ground Impedance Relay

The plot below shows the R-X diagram of the phase-ground Impedance relay. This figure shows that the phase to ground (a-g) impedance decreases and reaches zone 1.

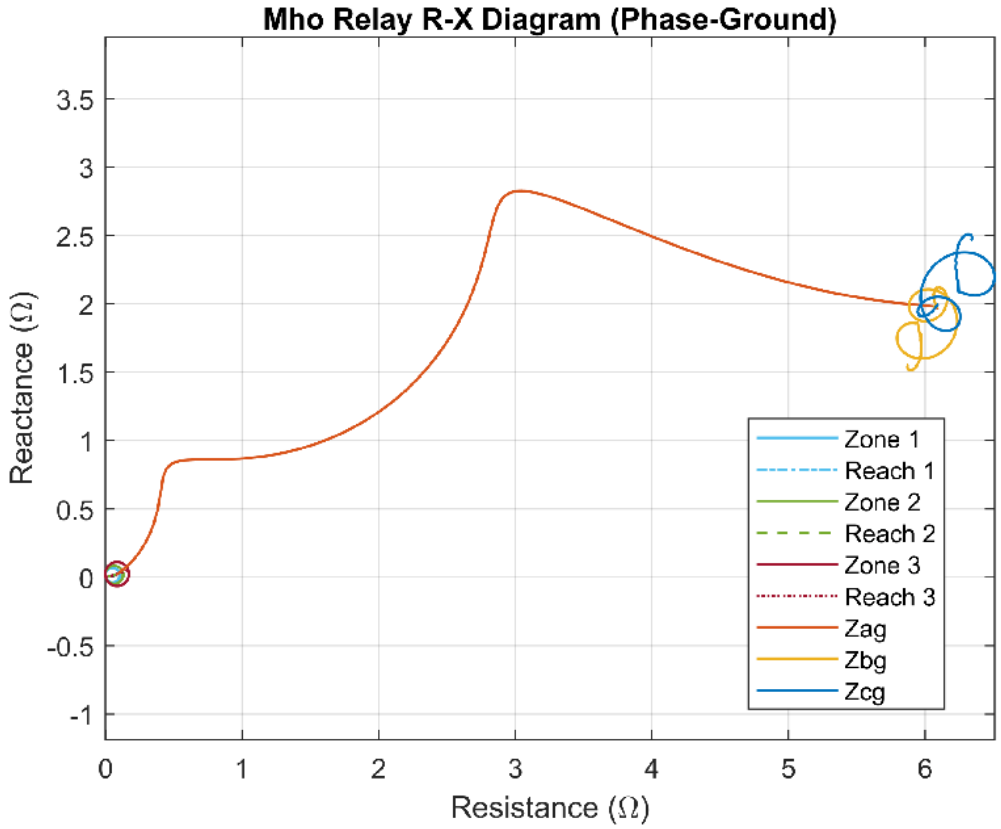



This figure shows the R-X diagram more in detail.

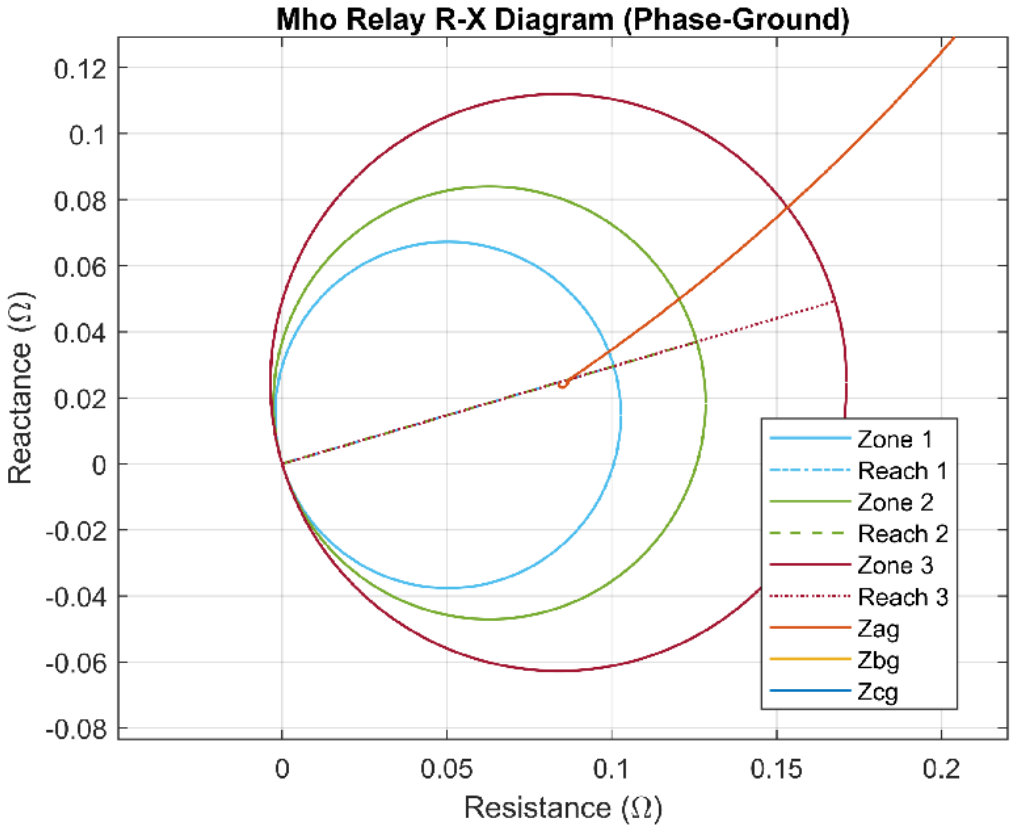



**R-X Diagram of Phase-Ground Mho Relay**

The plot below shows the R-X diagram of the phase-ground Mho relay. This figure shows that the phase to ground (a-g) impedance decreases and reaches zone 1 of the relay.




This figure shows the R-X diagram more in detail.

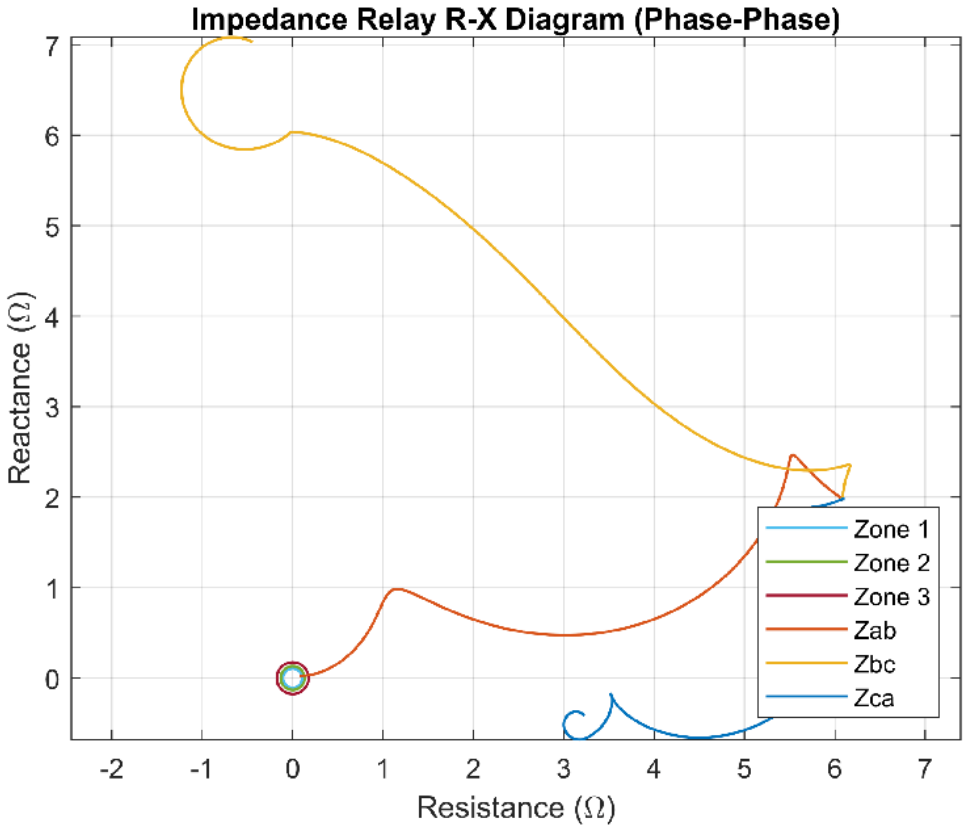



**Two-Phase Fault (a-b)**

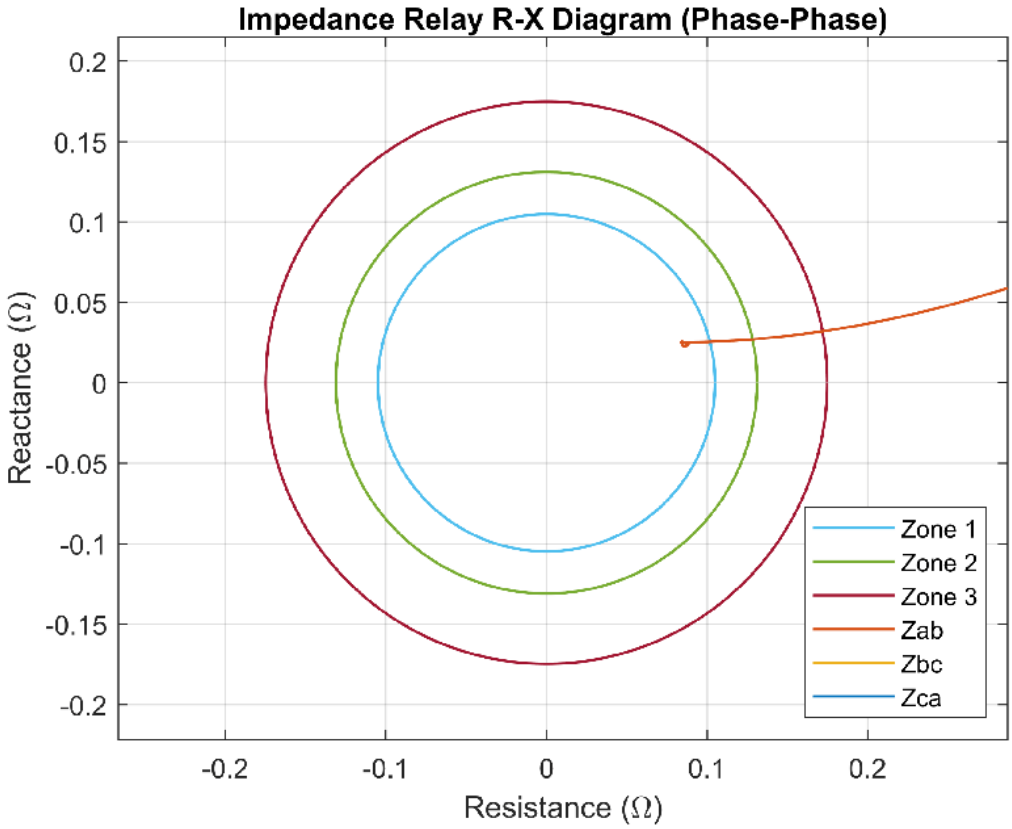
A phase-phase (a-b) fault is initiated in Sector 1 of the transmission line.

**R-X Diagram of Phase-Phase Impedance Relay**

The plot below shows the R-X diagram of the phase-phase Impedance relay. This figure shows that the phase-phase impedance (a-b) decreases and reaches zone 1 of the relay.

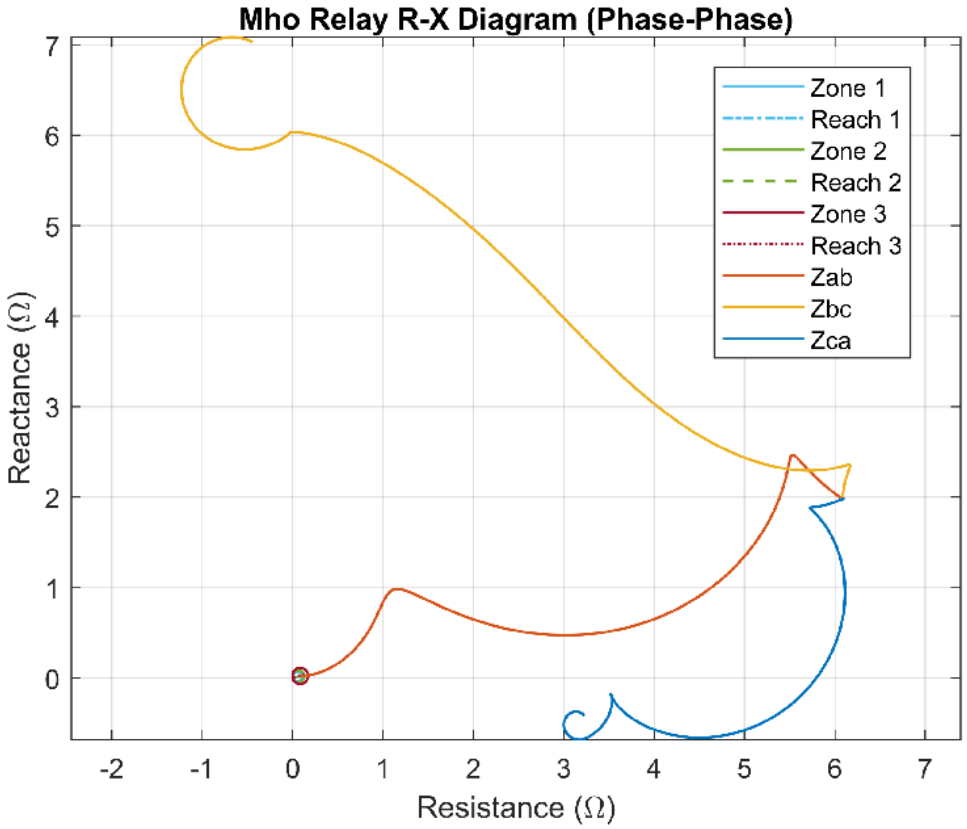



This figure shows the R-X diagram more in detail.

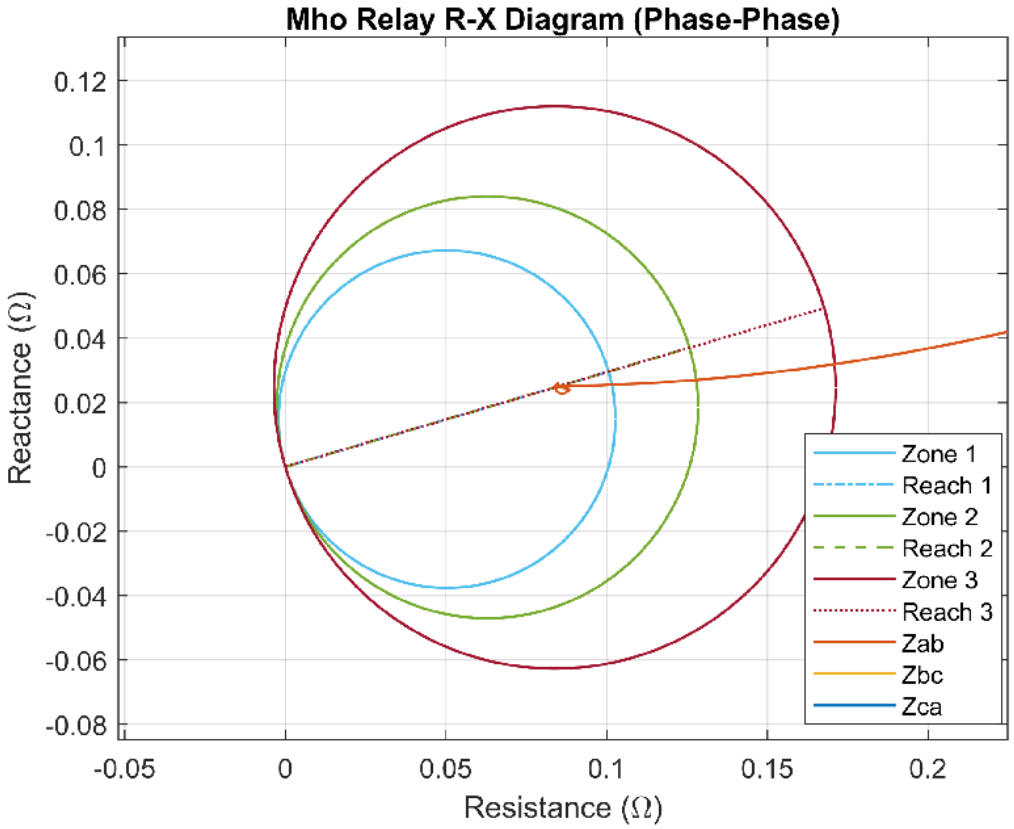



**R-X Diagram of Phase-Phase Mho Relay**

The plot below shows the R-X diagram of the phase-phase Mho relay. A two-phase to (a-b) fault is initiated in Sector 1 of the transmission line. This figure shows that the phase-phase impedance (a-b) decreases and reaches zone 1 of the relay.

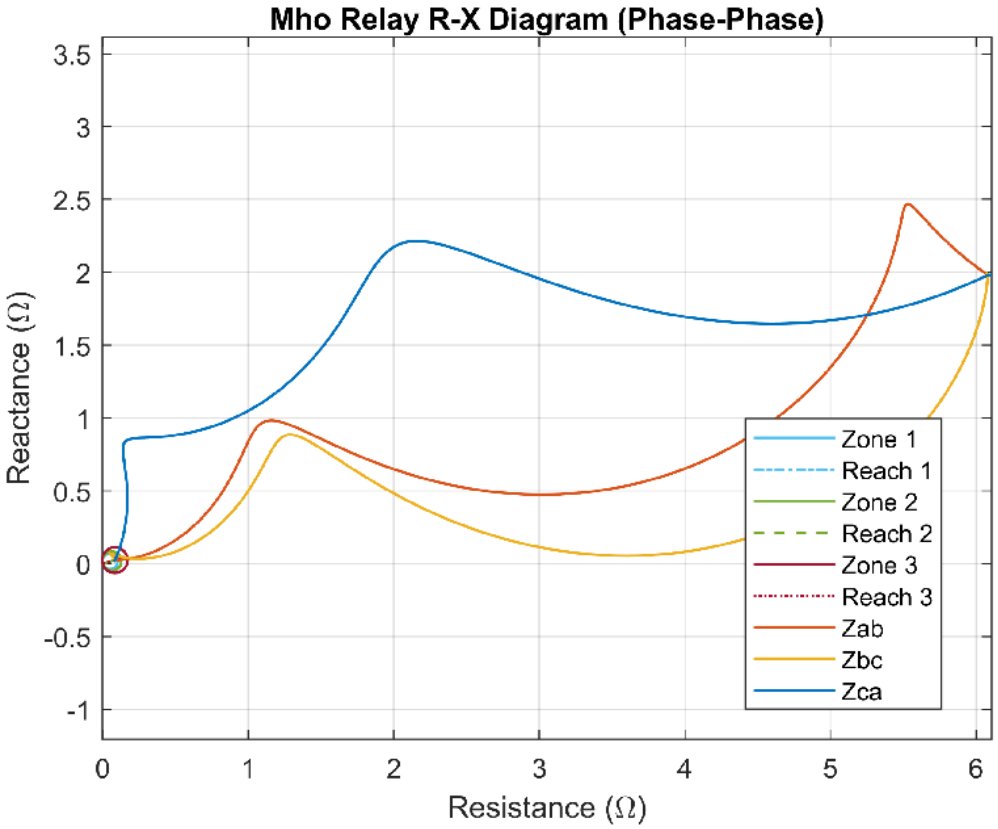



This figure shows the R-X diagram more in detail.

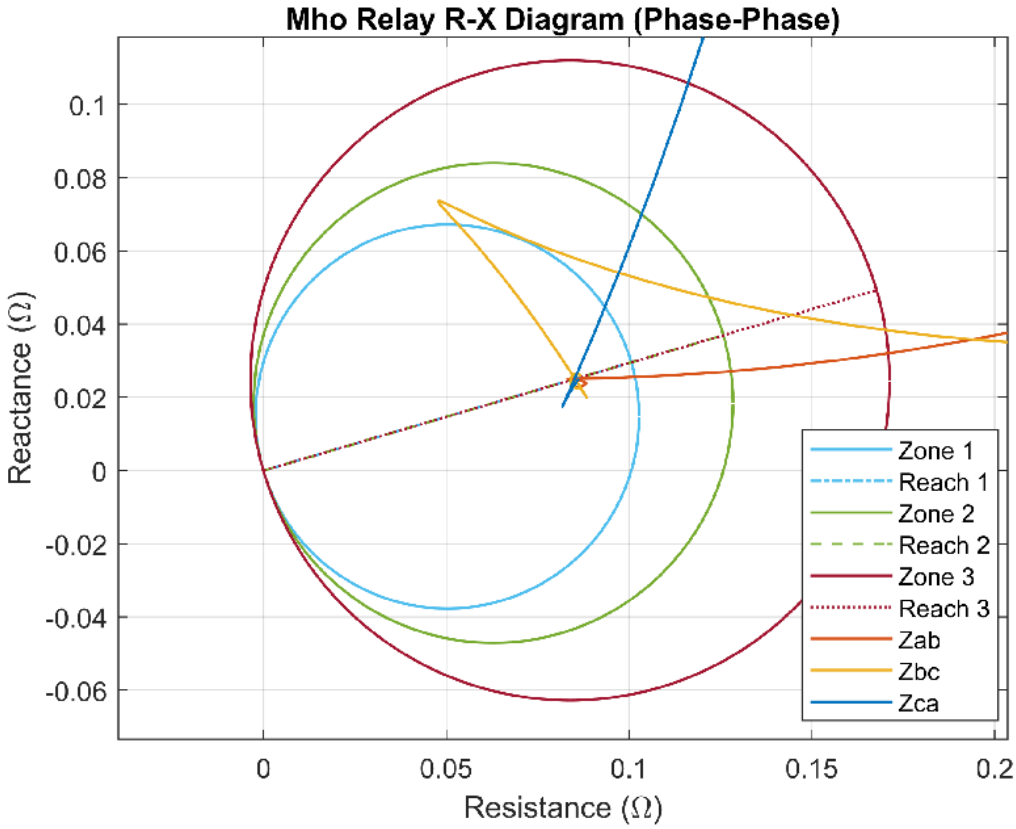



**Three-Phase to Ground Fault (a-b-c-g)**

A three-phase to ground fault (a-b-c-g) is initiated in Sector 1 of the transmission line. The plot below shows the estimated impedance for three-phase to ground fault on the R-X diagram of the phase-phase Mho relay. This figure shows that the phase-phase estimated impedance decreases and reaches zone 1 of the relay. imshow('ee_relay_distanceprotection_plot_MhoP2P_RX_Fabcg.png');




This figure shows the R-X diagram more in detail. You can see the final estimated impedance.




**Relay Parameters Configuration Overview**

- **Select the type of relay, Relay selection** — Impedance relay, Mho relay, or Impedance & Mho relay.
- **Fundamental frequency (Hz)** — Rated system frequency, in Hz, specified as a scalar. This value must be greater than zero.
- **Switching time step (s)** — Relay Sampling time, in sec, specified as a scalar.
- **Minimum threshold voltage (V)** — Minimum operating threshold voltage, in Volt.
- **Minimum threshold current (A)** — Minimum operating threshold current, in Amp.
- **Trip duration (s)** — Time duration for the trip signal, in sec.
- **Compensation factor (k0)** — Compensation factor for a ground fault.
- **Time delay zone2 (s)** — Relay trip signal delay time for the second zone, in sec.
- **Time delay zone3 (s)** — Relay trip signal delay time for the third zone, in sec.
- **Zone 1 reach (ohm)** — Mho relay reach for the first zone, in Ohm.
- **Zone 2 reach (ohm)** — Mho relay reach for the second zone, in Ohm.
- **Zone 3 reach (ohm)** — Mho relay reach for the third zone, in Ohm.

- **Zone 1 reach (ohm)** — Impedance relay reach for the first zone, in Ohm.
- **Zone 2 reach (ohm)** — Impedance relay reach for the second zone, in Ohm.
- **Zone 3 reach (ohm)** — Impedance relay reach for the third zone, in Ohm.
- **Blinder reach (ohm)** — Minimum blinder reach, in Ohm.
- **Blinder angle (deg)** — Blinder angle limit, in degree. The relay does not trip if the impedance angle is between **+Blinder angle** and **-Blinder angle** and the impedance magnitude is greater than the **Blinder reach**.
- **Directional unit minimum angle (deg)** — Minimum angle for directional unit, in degree.
- **Directional unit maximum angle (deg)** — Maximum angle for directional unit, in degree. The impedance relay trips only if the estimated impedance angle is greater than the **Directional unit minimum angle** and less than the **Directional unit maximum angle**.

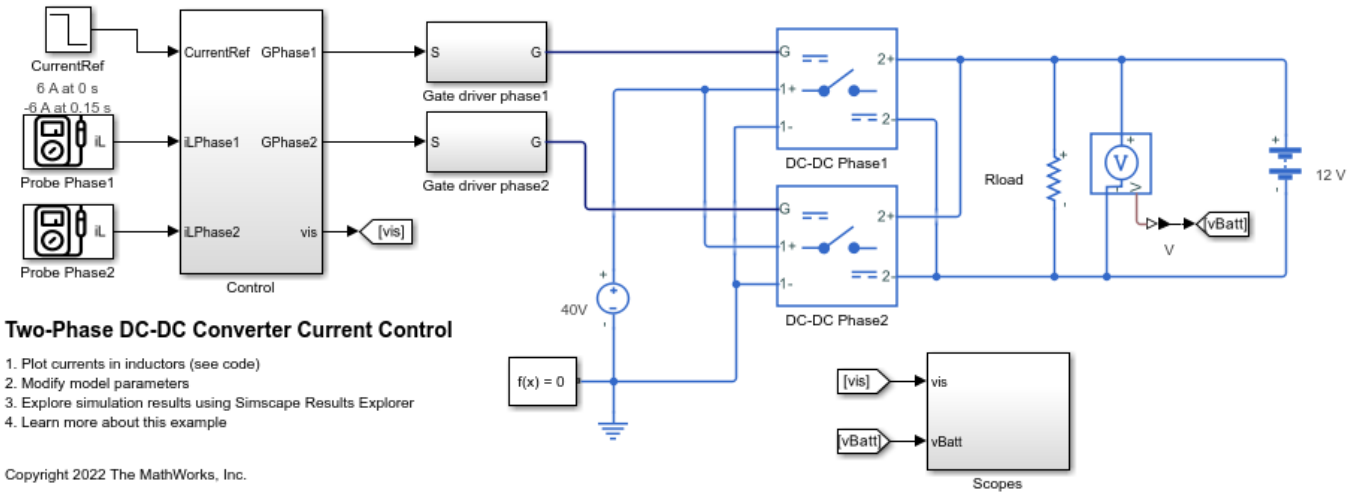
#### Fault Parameters Configuration Overview

- **Fault location** — Fault-point.
- **Fault type** — Fault type in the transmission line such as single-phase to ground, two phase, two-phase to ground, three-phase and three-phase to ground fault.
- **Faulted phase-neutral resistance (ohm)** — Resistance between phase and neutral, in Ohm.
- **Faulted neutral-ground resistance (ohm)** — Resistance between neutral and ground, in Ohm.
- **Fault start time (ms)** — Fault initiation time, in msec.
- **Fault duration (ms)** — Fault duration, in msec.

# Two-Phase DC-DC Converter Current Control

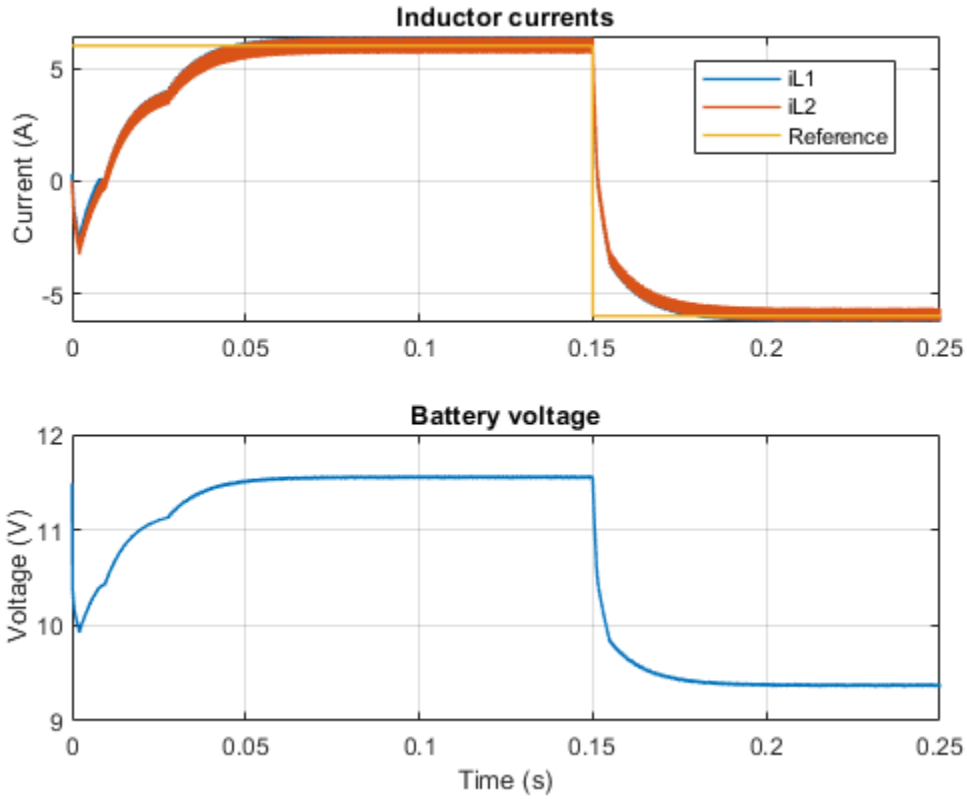
This example shows how to control the current of a two-phase interleaved bidirectional DC-DC converter. The two-phase converter comprises two Bidirectional DC-DC Converter with ideal IGBTs. To adjust the duty cycle, the Control subsystem uses a PI-based control algorithm. To reduce the ripple at the output port of the converter, the two phases are switched with the same duty ratio but with a relative phase shift of 180 degrees. The Scopes subsystem contains scopes that allow you to see the simulation results.

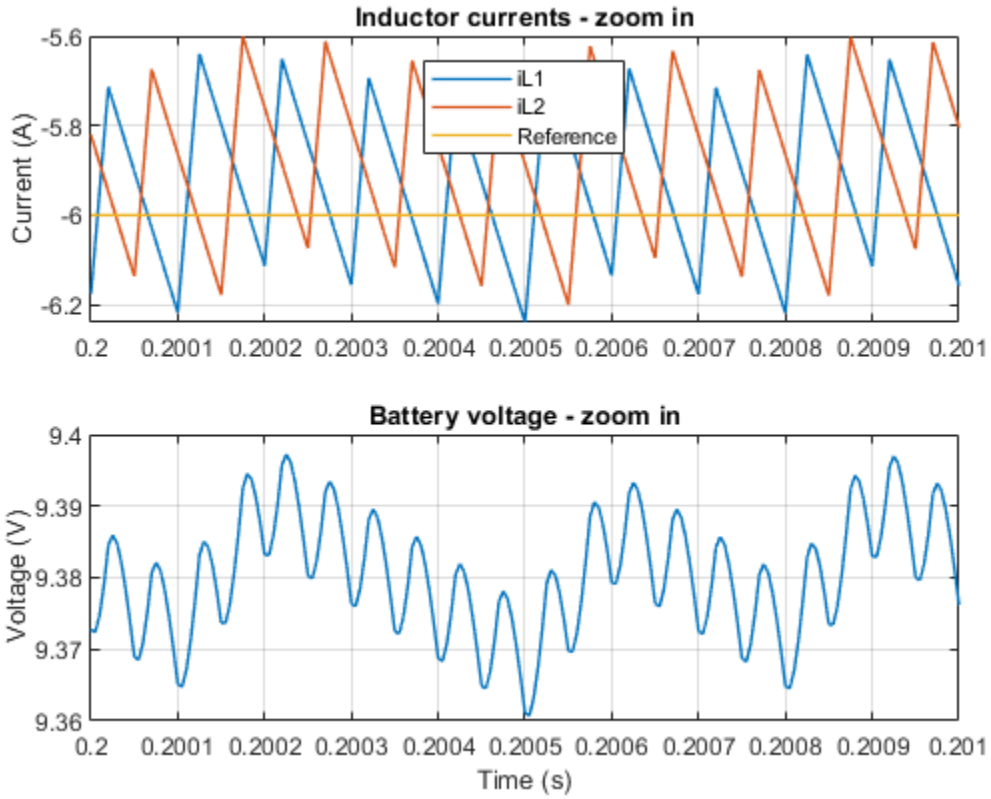
### Model




### Simulation Results from Simscape Logging

The plot below shows the requested and measured inductor currents for the test and the battery voltage in the 12V network.

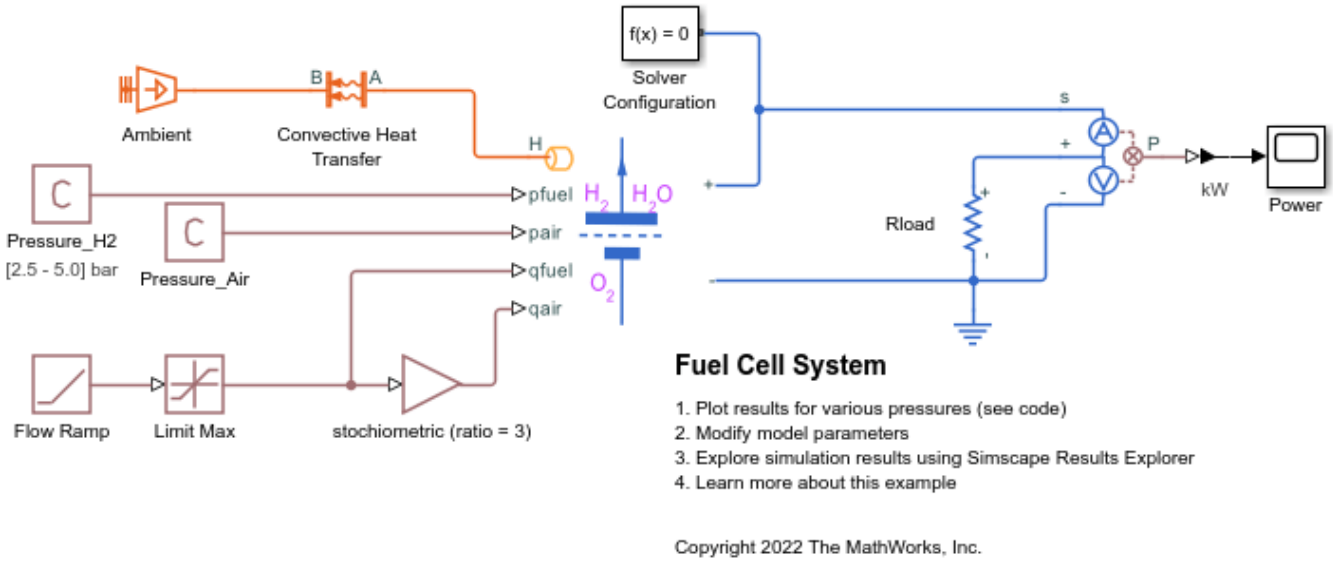

Inductor currents
Current (A)
5
0
-5
0
0.05
0.1
0.15
0.2
0.25
iL1
iL2
Reference
Battery voltage
Voltage (V)
12
11
10
9
0
0.05
0.1
0.15
0.2
0.25
Time (s)


Inductor currents - zoom in
Current (A)
-5.6
-5.8
-6
-6.2
iL1
iL2
Reference
0.2
0.2001
0.2002
0.2003
0.2004
0.2005
0.2006
0.2007
0.2008
0.2009
0.201
Battery voltage - zoom in
Voltage (V)
9.4
9.39
9.38
9.37
9.36
Time (s)

# Fuel Cell System

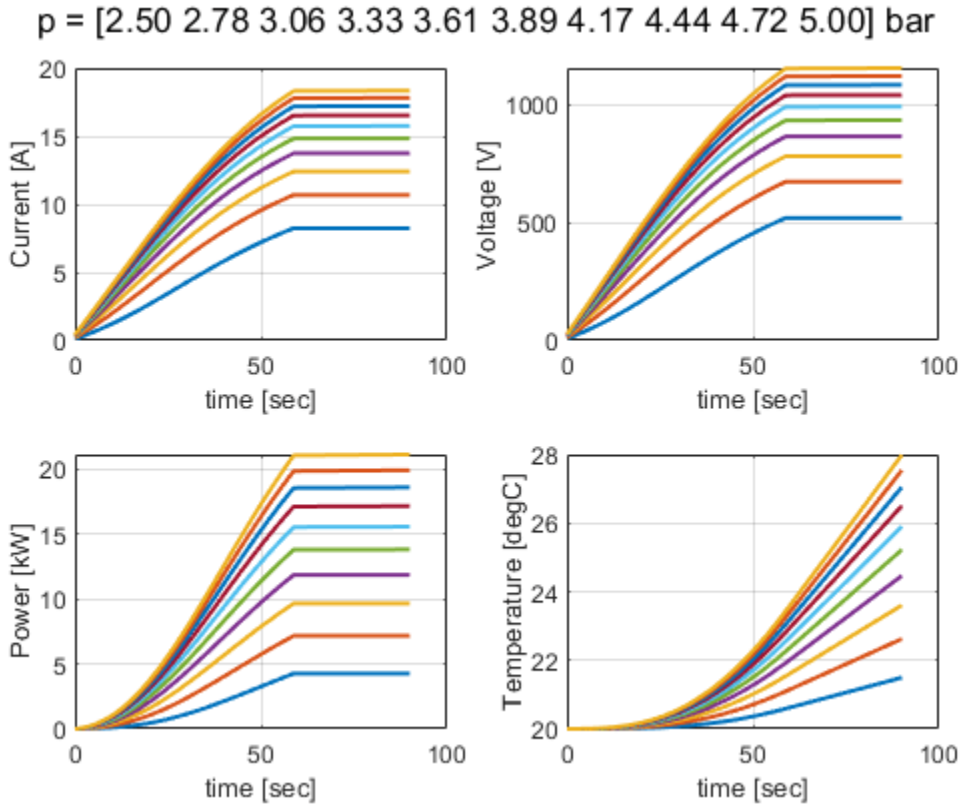
This example shows a Fuel Cell system that operates at stochiometric conditions and nominal parameters. The power delivered varies as a function of the hydrogen pressure.

### Model




### Simulation Results from Simscape Logging

The plot below shows the time evolution of the current, voltage, power, and temperature for pressures between 2.5 and 5 bar.



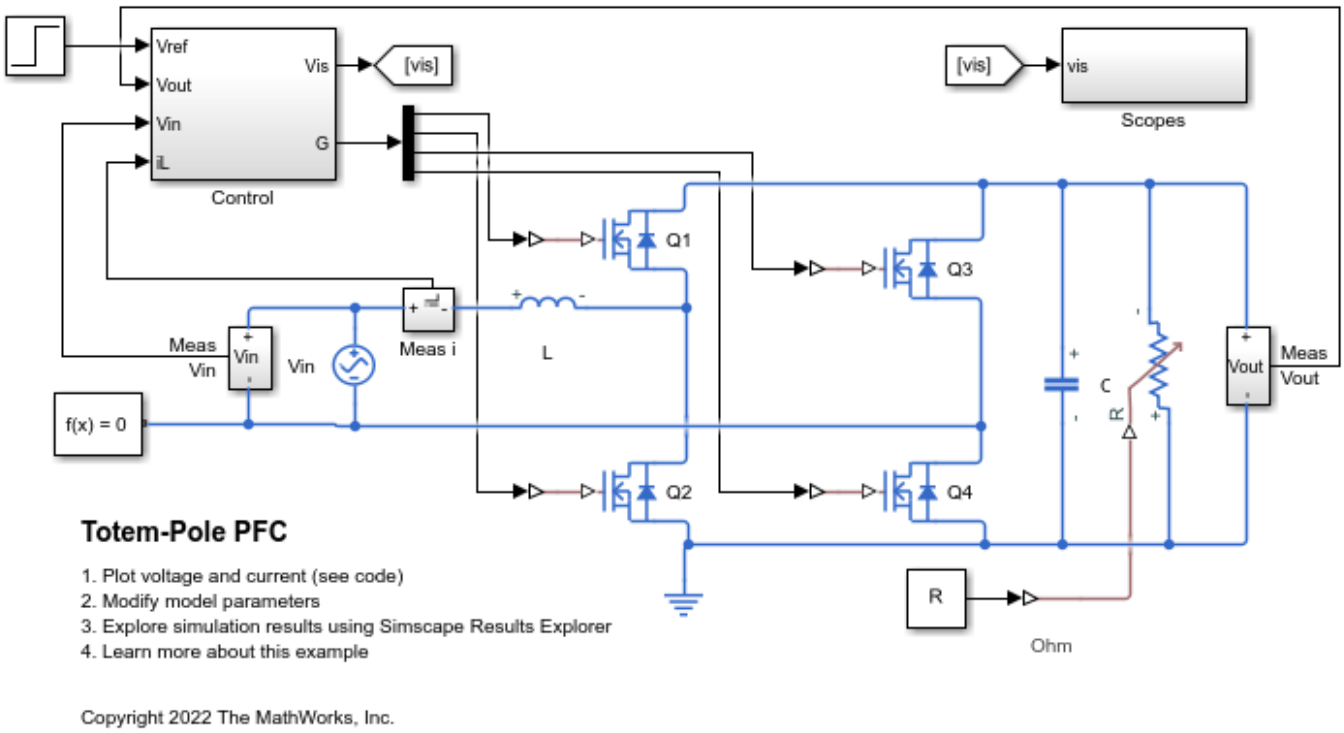

**Results from Real-Time Simulation**

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 50 microseconds.

# Totem-Pole PFC

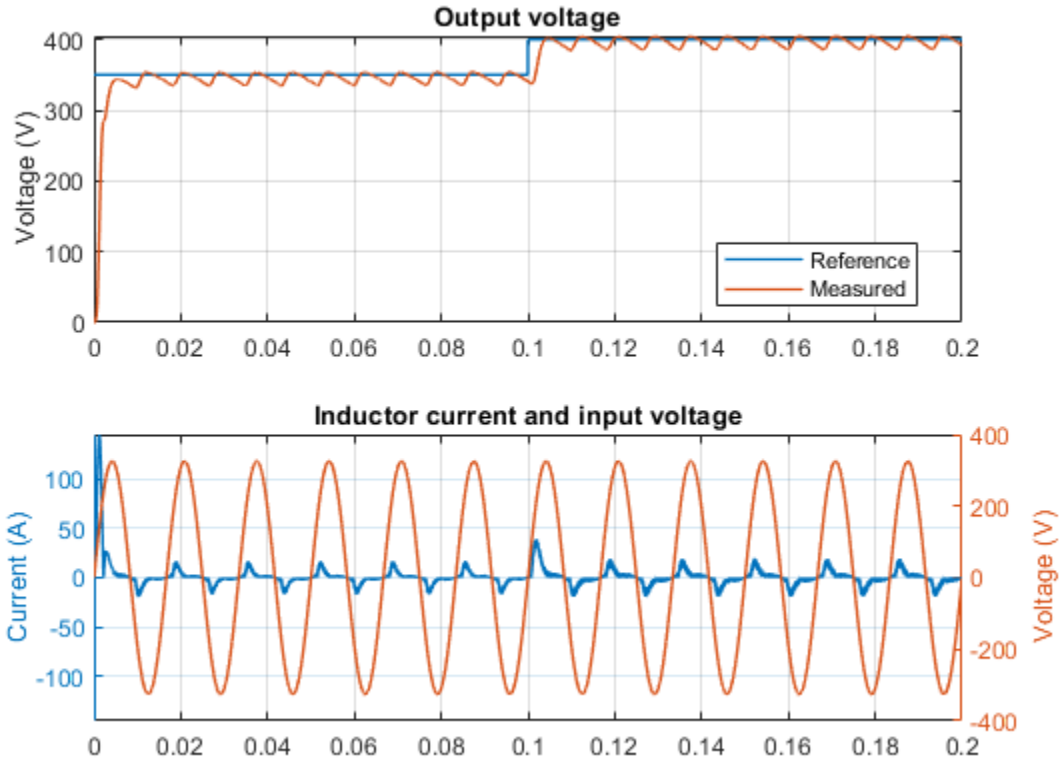
This example shows how to control the rectified voltage in a totem-pole power factor correction (PFC) circuit. MOSFETs Q1 and Q2 form the 50 kHz fast switching leg. MOSFETs Q3 and Q4 form the line frequency slow switching leg. The control subsystem uses a PI-based cascade control structure. The Scopes subsystem contains scopes that allow you to see the simulation results.

### Model




### Simulation Results from Simscape Logging

The plot below shows the requested and measured rectified voltage for the test and the inductor current.


Output voltage
Voltage (V)
400
300
200
100
0
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
0.18
0.2
Reference
Measured
Inductor current and input voltage
Current (A)
100
50
0
-50
-100
Voltage (V)
400
200
0
-200
-400

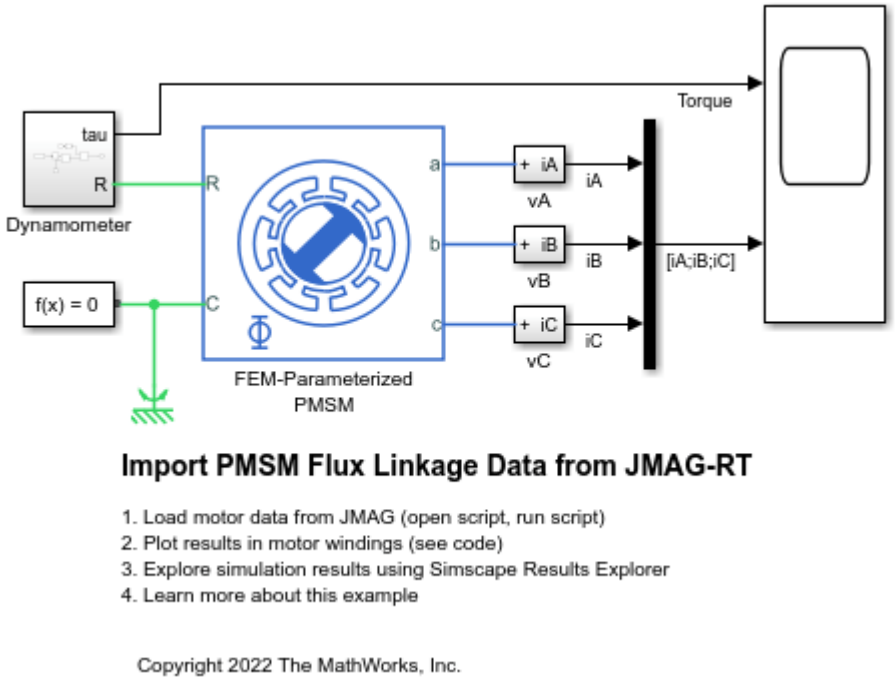
# Import PMSM Flux Linkage Data from JMAG-RT

This example shows how you can import a motor design from JMAG®-RT into a Simscape™ simulation.

### Flux Linkage Data

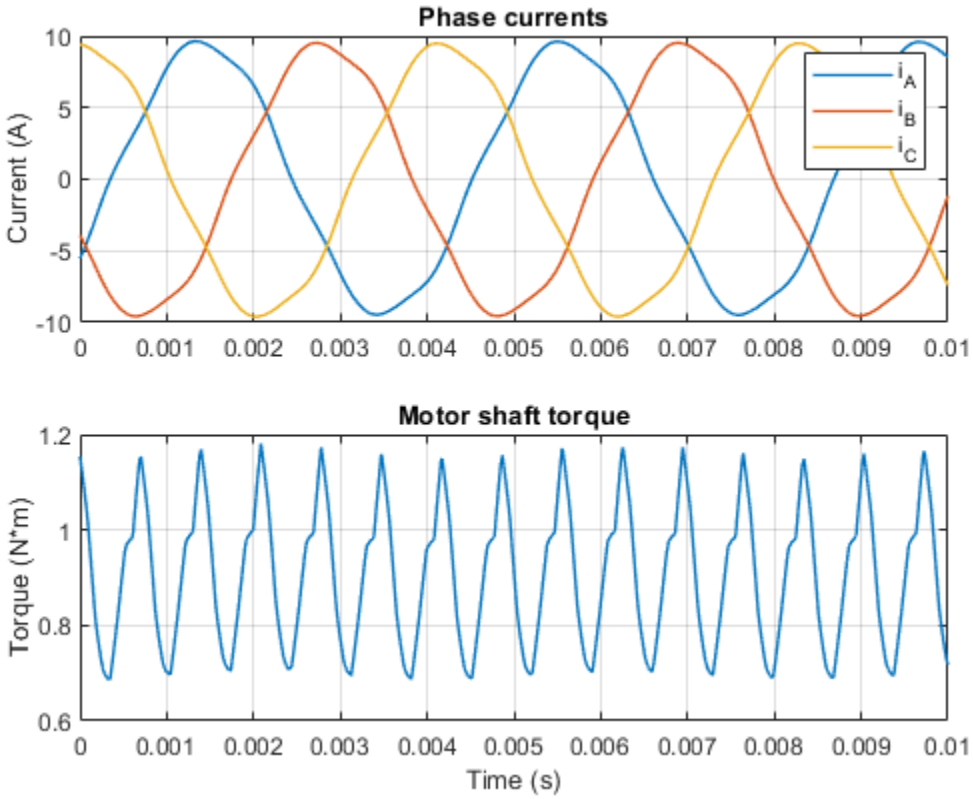
The data file defining the motor flux linkage and torque is converted from a JMAG-RT model data file, 1k_S_C_IV.rtt, and is included in this example with permission of JSOL Corporation. JMAG-RT model data files are lookup tables for the magnetic flux and torque calculated using JMAG-Designer. You can convert a JMAG-RT model data file into a MATLAB® binary (*.mat) data file using the JMAG-RT Viewer (https://www.jmag-international.com/products/rt-viewer/). The details of this example motor, such as dimensions and rated power, are also accessible from the JMAG-RT Viewer.

### Model




### Simulation Results from Simscape Logging

The plot below shows the simulated motor currents and shaft torque when the rotor electrical angle leads the stator angle by 90 degrees. Notice the torque contains the harmonic components due to the slots and the magnetic saturation.

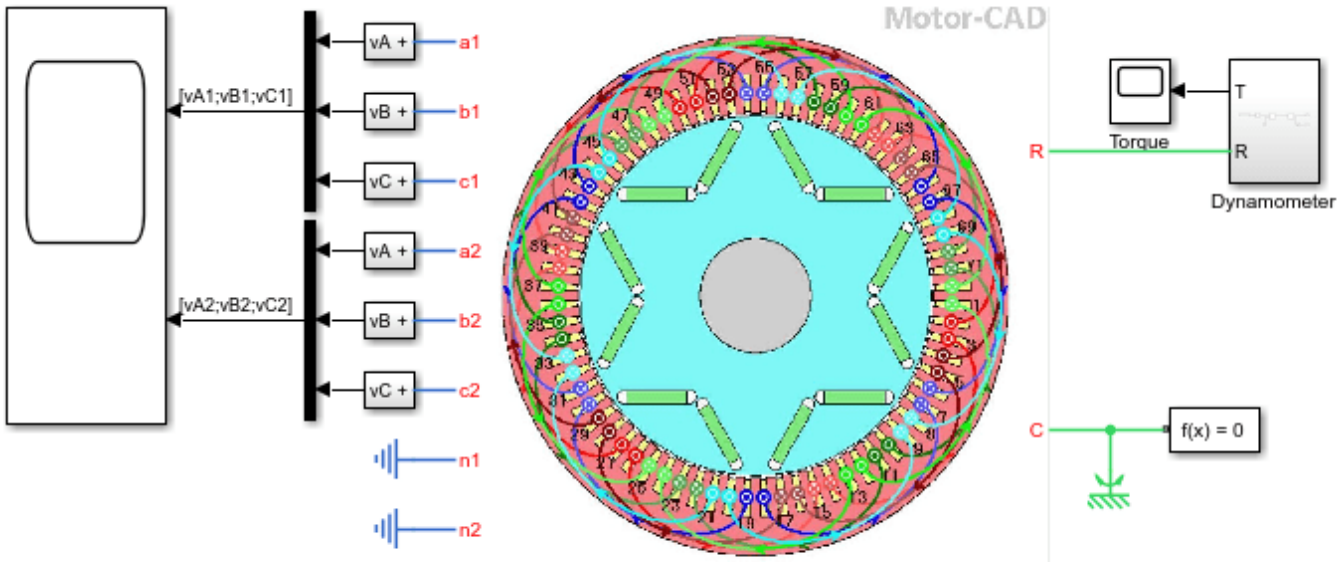

Phase currents
Current (A)
10
5
0
-5
-10
0
0.001
0.002
0.003
0.004
0.005
0.006
0.007
0.008
0.009
0.01
$i_A$
$i_B$
$i_C$
Motor shaft torque
Torque (N*m)
1.2
1
0.8
0.6
Time (s)

# Import a Dual Three-Phase SPMSM from Motor-CAD

This example shows how you can import a six-phase motor design from Motor-CAD into a Simscape™ simulation. This particular motor is configured with dual wye connections. The example sets out the derivation of the implementation and you can modify it to represent other winding configurations.

### Model Overview

To implement the imported behavior from Motor-CAD, this model uses a custom Simscape component. To view the source code, click on the Source code link located in the custom component's block help. The equations used are derived in the MATLAB® live script opened by clicking the Open MATLAB live script hyperlink.



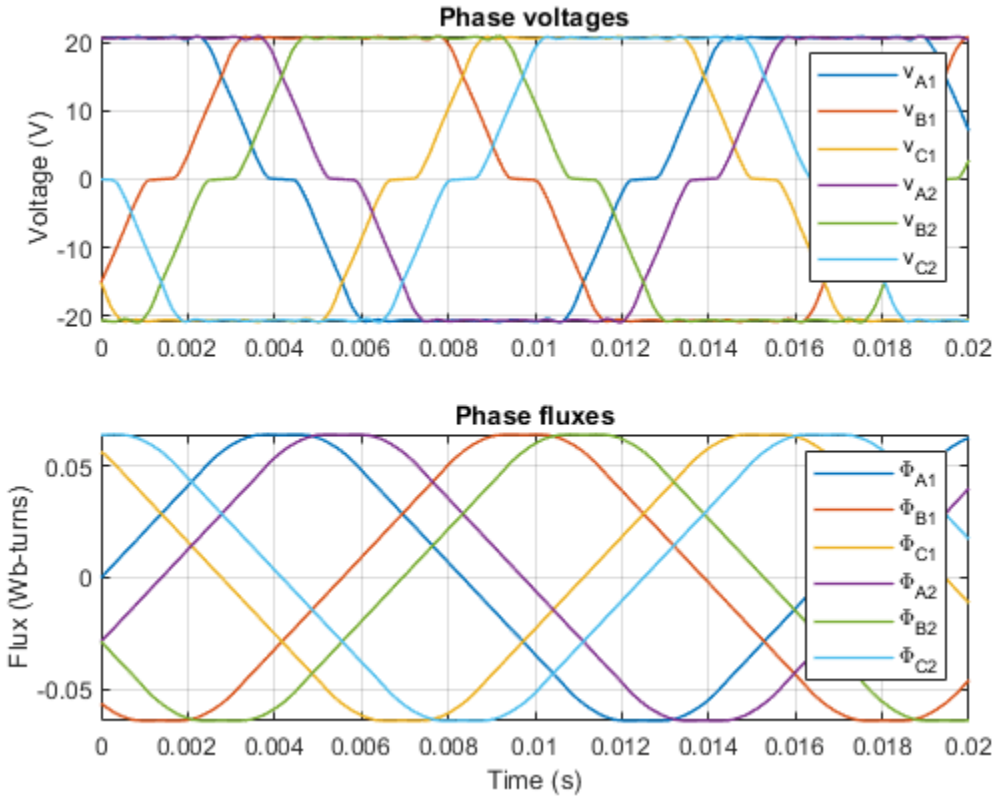

**Import a Dual Three-Phase SPMSM from Motor-CAD**

1. Theory and data import from Motor-CAD: Open MATLAB live script, run script
2. Plot flux linkages (see code)
3. Explore simulation results using Simscape Results Explorer
4. Learn more about this example

Copyright 2018-2022 The MathWorks, Inc.

### Flux Linkage and Back EMF Plots

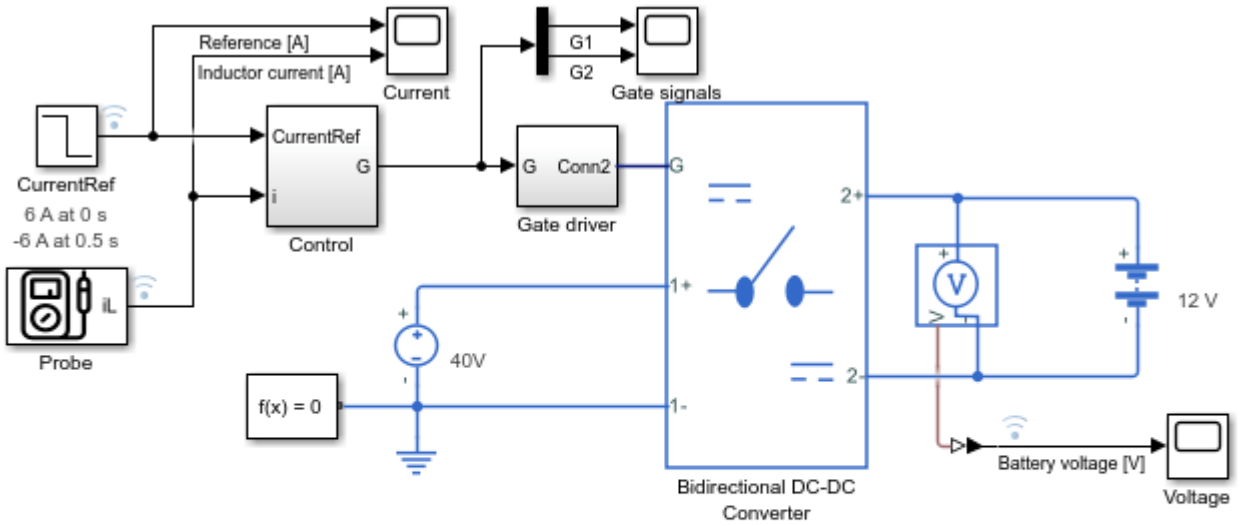
The plot below shows open-circuit flux linkages and associated back EMFs.


Phase voltages
Voltage (V)
20
10
0
-10
-20
0
0.002
0.004
0.006
0.008
0.01
0.012
0.014
0.016
0.018
0.02
$v_{A1}$
$v_{B1}$
$v_{C1}$
$v_{A2}$
$v_{B2}$
$v_{C2}$
Phase fluxes
Flux (Wb-turns)
0.05
0
-0.05
$\Phi_{A1}$
$\Phi_{B1}$
$\Phi_{C1}$
$\Phi_{A2}$
$\Phi_{B2}$
$\Phi_{C2}$
Time (s)

# Bidirectional DC-DC Converter Current Control

This example shows how to control the inductor current of a Bidirectional DC-DC Converter. To adjust the duty cycle, the Control subsystem uses a PI-based control algorithm. The Bidirectional DC-DC Converter uses averaged switches. To achieve different levels of fidelity, you can use either modulation waveforms, averaged gate pulses, or gate pulses.

### Model



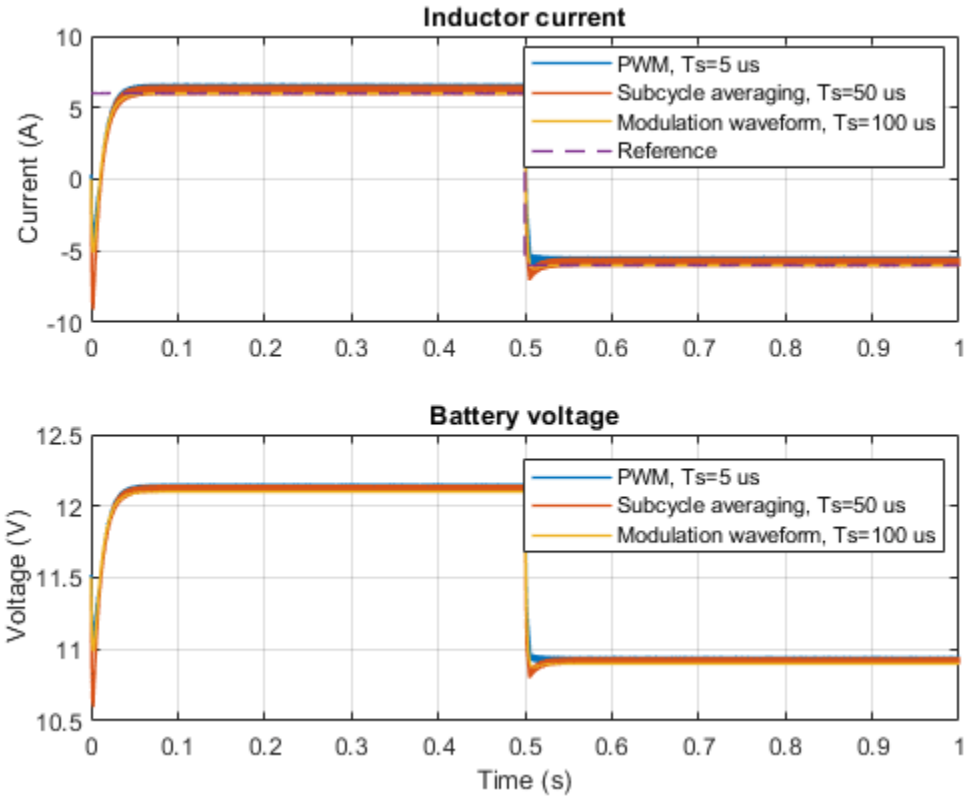

**Bidirectional DC-DC Converter Current Control**

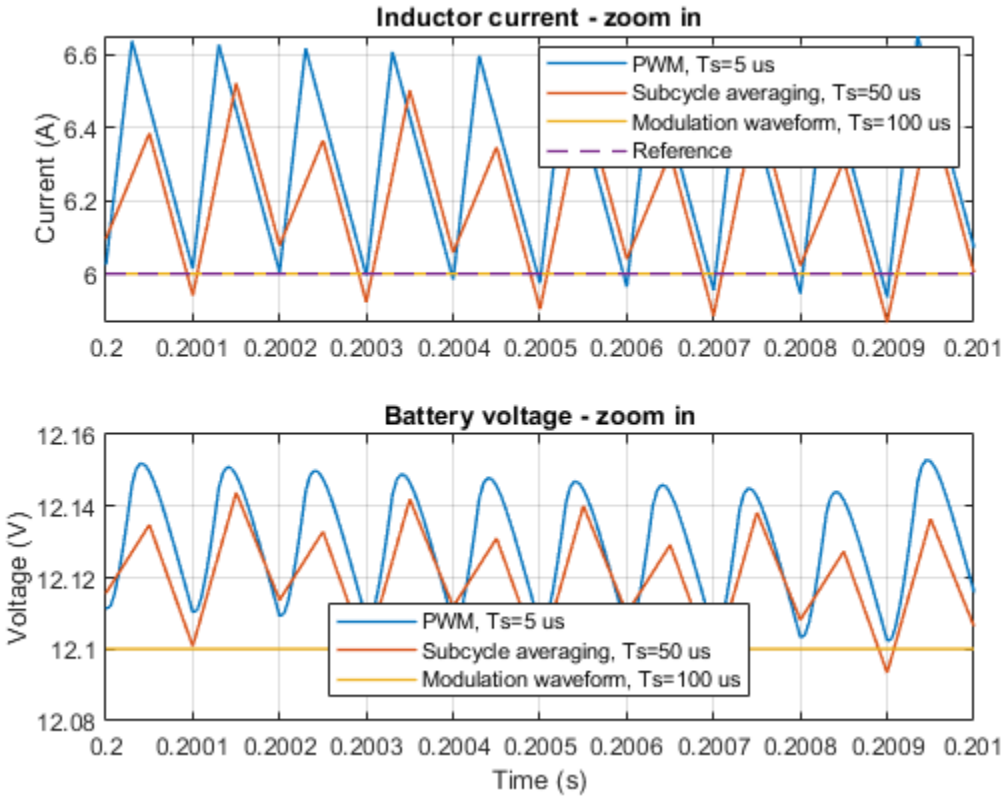
1. Plot current in inductor (see code)
2. Gate control signals
   i) Modulation waveform ii) Subcycle averaging iii) PWM
3. Modify model parameters
4. Learn more about this example

Copyright 2021 The MathWorks, Inc.

### Simulation Results from Simscape Logging

The plot below shows the requested and measured inductor current for the test and the battery voltage in the 12V network.


Inductor current
Current (A)
10
5
0
-5
-10
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
PWM, Ts=5 us
Subcycle averaging, Ts=50 us
Modulation waveform, Ts=100 us
Reference
Battery voltage
Voltage (V)
12.5
12
11.5
11
10.5
Time (s)



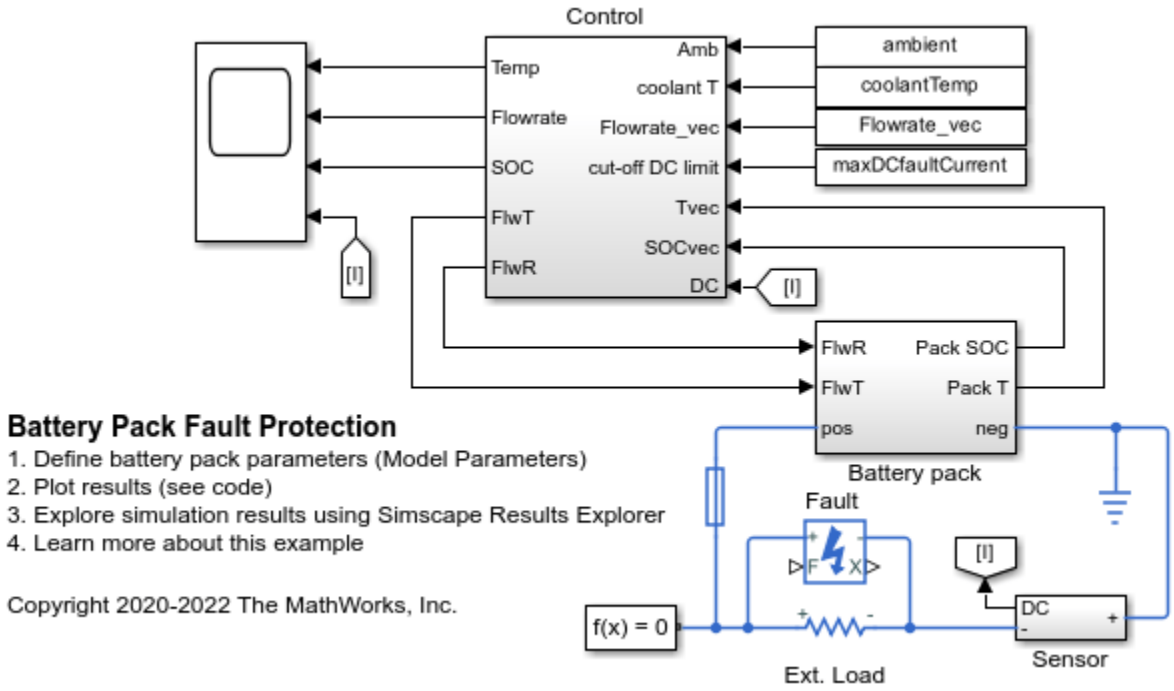

### Results from Real-Time Simulation

This example, with Bidirectional DC-DC Converter controlled using modulation waveform, has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 20 microseconds.

# Battery Pack Fault Protection

This example shows how to model fault and fault protection using a fuse in an automotive battery pack for thermal. The battery pack consists of several battery modules, which are combinations of cells in series and parallel. Each battery cell is modeled using the **Battery (Table-Based)** Simscape Electrical block. In this example, the initial temperature and the state of charge are the same for all cells. Four battery modules, three similar and one differing from the other three, are connected in series to simulate a battery pack. The results in this example assume an initial ambient temperature equal to 25 degree Celsius. The Control subsystem defines the logic used to determine the battery pack coolant flow rate. A fuse is placed inline to battery pack as a measure of fault protection.

### Model Overview

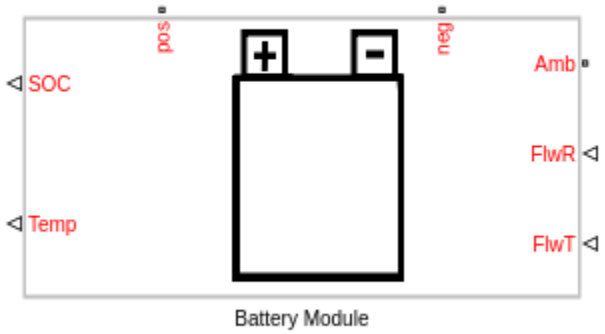



### Parameters and Input Overview

To use this model to create a unique battery module, first specify the number of series- and parallel-connected cells. Then specify the cell type for all individual cells by choosing one of these options for the **Choose cell type** parameter of the **Battery Module** block:

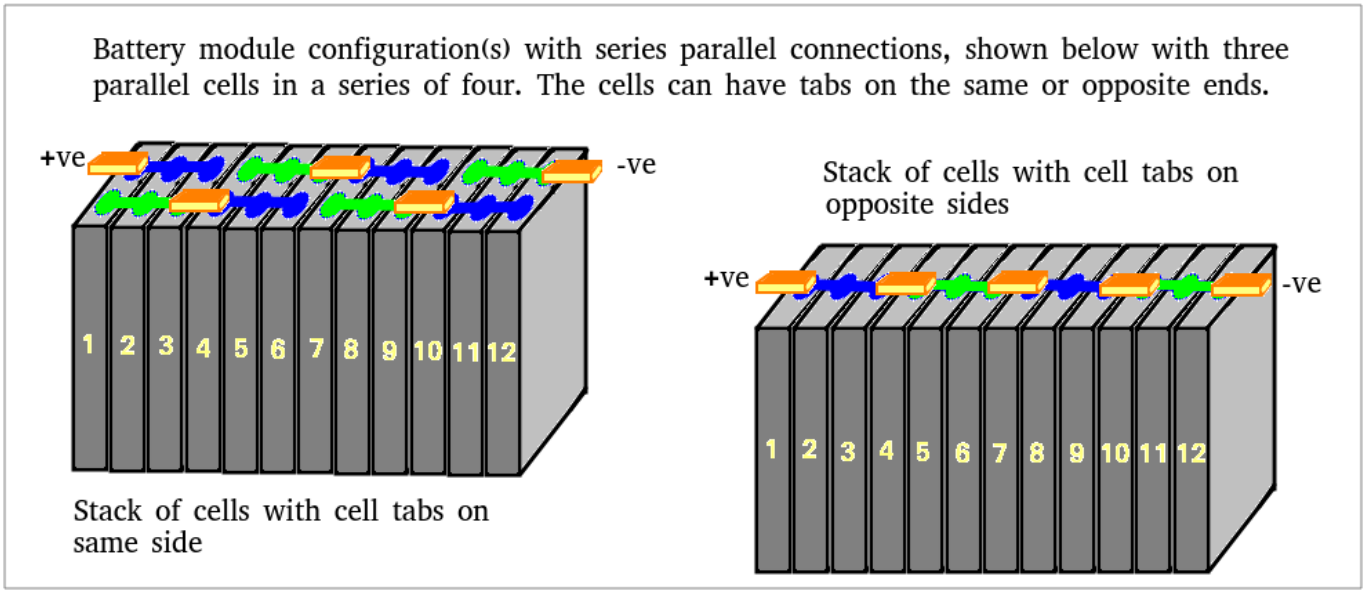
- `Pouch`
- `Can`
- `Compact cylindrical`
- `Regular cylindrical`

This example uses pouch-type cells. Module A, B, and C consist of five series-connected and two parallel-connected cells. Module D consists of six series-connected and two parallel-connected cells.

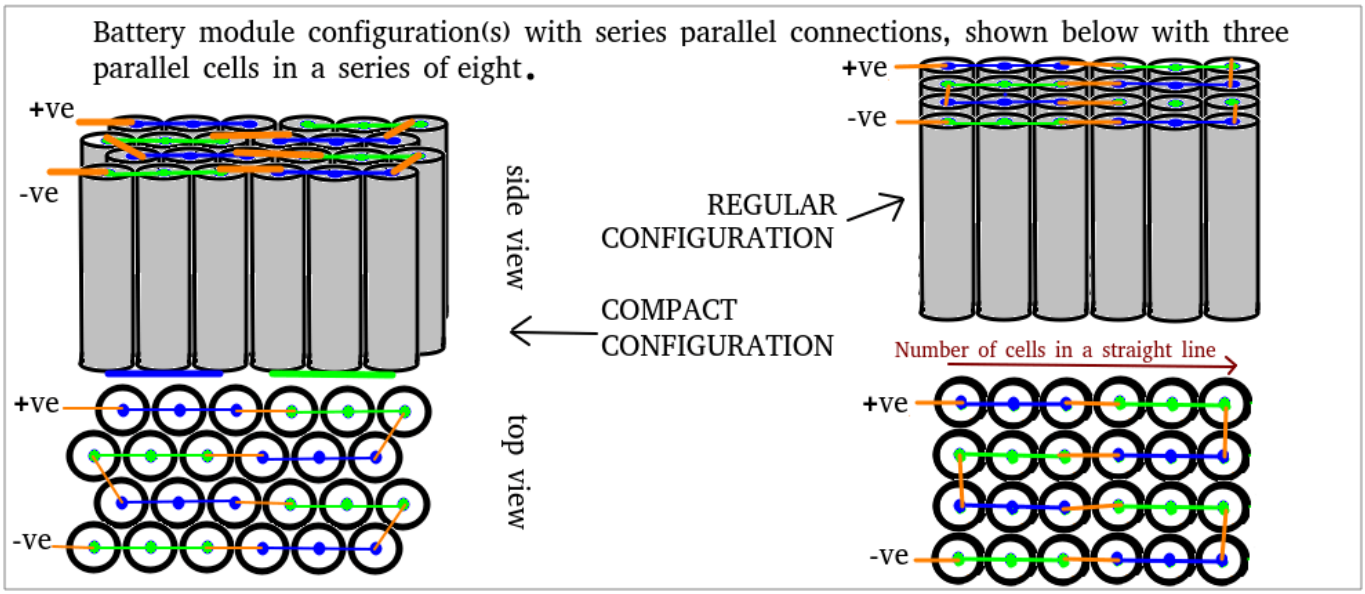



The two output ports, **SOC** and **Temp**, provide information regarding the state of charge and the temperature of each cell in the module. The thermal port, **Amb**, is used to define the ambient temperature in the simulation. The electrical ports, **pos** and **neg**, define the electrical positive and negative terminals, respectively. The two input ports, **FlwR** and **FlwT**, define the battery coolant flow rate control and inlet temperature into the module.

The figure below shows examples of battery cells in `Pouch` and `Can` configurations.




The figure below shows examples of battery cells in `Compact cylindrical` and `Regular cylindrical` configurations.




These are the parameters in the battery module:

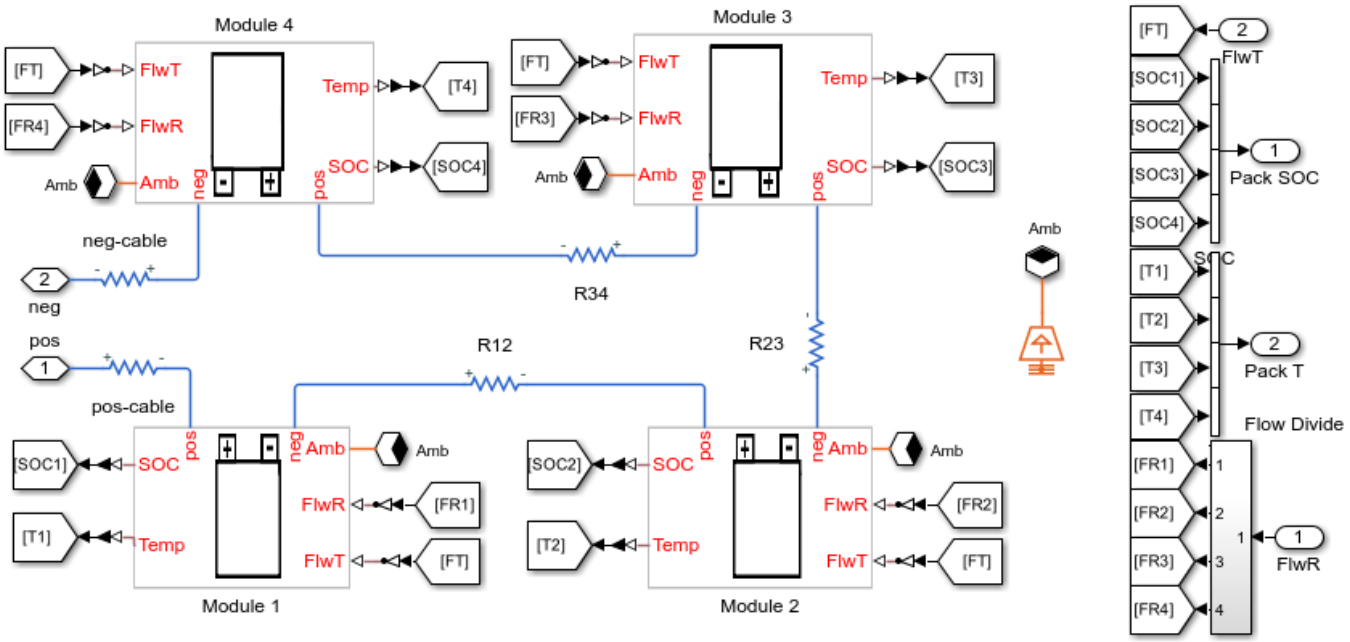
- **Vector of temperatures, T** — Temperatures at which the cell or module data for temperature-varying properties are tabulated, specified as a vector.
- **Single cell Ahr rating, baseline** — Cell capacity at the temperatures defined in the **Vector of temperatures, T** parameter, specified as a vector.
- **Vector of state of charge values, SOC** — Range of values between 0 and 1 at which the cell electrical parameters are defined, specified as a vector.
- **Vector of coolant flowrates, L** — Coolant mass flow rate values at which a lookup table for cell cooling is defined. This parameter needs to cover multiple points in the flow range of interest. This parameter defines the size of the **Effective rate of coolant heat transfer** parameter and is specified as a vector.
- **No load voltage, V0** — Cell open-circuit potential values at different **Vector of state of charge values, SOC** and **Vector of temperatures, T** points, specified as a matrix.
- **Terminal resistance, R0** — Cell ohmic resistance values at different **Vector of state of charge values, SOC** and **Vector of temperatures, T** points, specified as a matrix.
- **Polarization resistance** — Polarization resistance values at different **Vector of state of charge values, SOC** and **Vector of temperatures, T** points, specified as a matrix.
- **Time constant** — Time constant at different **Vector of state of charge values, SOC** and **Vector of temperatures, T** points, specified as a matrix.
- **Cell thermal mass** — Thermal mass of a single cell, specified as a scalar.
- **Cell thermal conductivity** — Cell through-plane conductivity for pouch and can cells, or the radial conductivity for cylindrical cells, specified as a scalar.
- **Heat transfer coefficient to ambient** — Heat transfer coefficient value, specified as a scalar.
- **Number of series connected cells Ns** — Number of strings in series, specified as an integer.
- **Number of parallel connected cells Np** — Number of parallel-cells in a string, specified as an integer.

- **Choose cell type** — Type of cell, specified as either `Pouch`, `Can`, `Compact cylindrical`, or `Regular cylindrical`.
- **Cell height** — Cell height, specified as a scalar.
- **Cell width** — Cell width for `Pouch` and `Can` cells, specified as a scalar.
- **Cell thickness** — Cell thickness for `Pouch` or `Can` cells, specified as a scalar.
- **Cell diameter** — Cell diameter for `Compact cylindrical` or `Regular cylindrical`, specified as a scalar.
- **Number of cylindrical cells in a straight line** — Number of cylindrical cells arranged in a straight line for packaging, specified as an integer.
- **Accessory total resistance** — Resistance that combines all inline resistance in a module, specified as a scalar. This resistance is the sum of cell tab, busbar, cable and/or weld resistances, specified as a scalar.
- **Cell balancing** — Cell balancing method, specified as either `none` or `passive`. In this example, this parameter is set to `none`.
- **Effective rate of coolant heat transfer from each cell** — Estimate of the thermal resistance (W/K) of heat transfer from battery cells to coolant, specified as a 3-D matrix of scalar values. The 3-D matrix size depends on the **Vector of temperatures, T**, **Vector of coolant flowrates, L** and **NsxNp** parameters. The **NsxNp** parameter is the total number of cells in the module. The battery cooling is represented as a lookup table or 3-D matrix of size [T,L,Ns*Np] and the values are calculated using detailed 3-D methods such as computational fluid dynamics. The values of the matrix depend on the actual hardware design of the cooling system or cold plates in the module. The performance of the cold plate is controlled using input values **FlwR** and **FlwT**.
- **External heat** — External heat input to each cell in a module due to a hot component placed near the module, specified as a vector.
- **Vector of initial cell temperature** — Cell initial temperature, specified as a vector.
- **Vector of initial cell state of charge** — Cell initial state of charge, specified as a vector.
- **Cell Ahr rating variation** — Cell-to-cell variations in cell capacity at all **Vector of temperatures, T** points for each cell, specified as a vector of scalar values. If you set this array to 1, all cell capacity is the same. The array values for a cell are multiplied with the value specified in the **Single cell Ahr rating, baseline** parameter to calculate the actual capacity or the Ahr rating of the cell.

To define the battery coolant flow rate and temperature, specify these inputs:

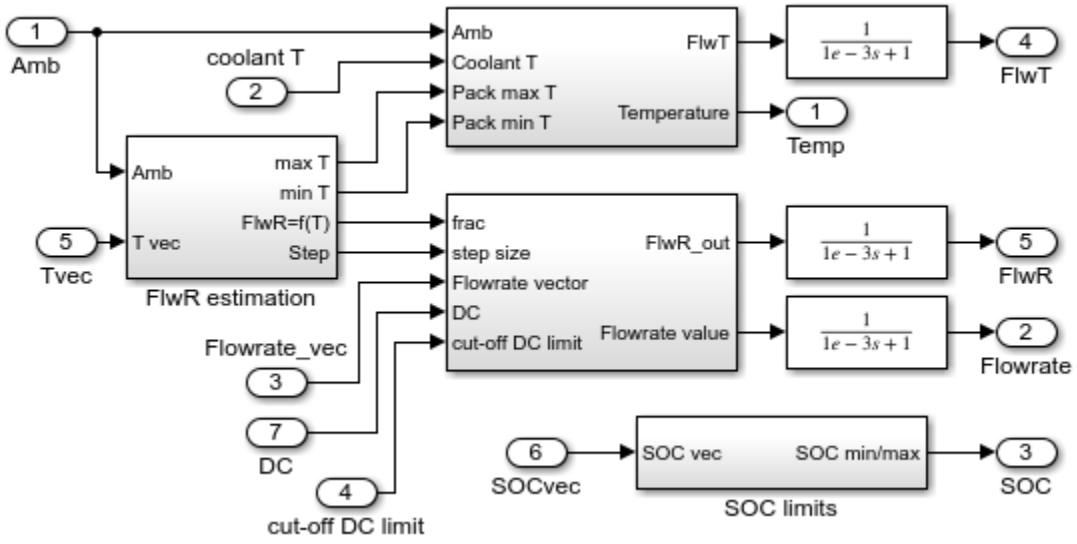
- **FlwR** — Value between 0 and 1, specified as a scalar. The **FlwR** input value is used to dynamically choose the right value of the flow rate during the simulations. The value of the **FlwR** input defines the actual flow rate in the module. In the **Vector of coolant flowrates L** parameter, **FlwR** equal to 0 means no flow, while **FlwR** equal to 1 means highest flow rate value.
- **FlwT** — Positive or negative value that, when summed to the ambient temperature, equals the coolant inlet temperature. A value of +15 for the **FlwT** input and 273.15 K at the **Amb** port makes the coolant inlet temperature equal to 273.15 + 15 = 288.15K. A value of -15 for the **FlwT** input and 273.15 K at the **Amb** makes the coolant inlet temperature equal to 273.15 - 15 = 258.15 K

In this example, the battery pack consists of four modules connected in series. The first three modules are identical. The fourth module has a different number of cells, `Ns4`, and cooling efficiency, `coolantQ4`, defined in the `ee_lithium_pack_cooling_ini.m` file. Cables, modeled as resistors, connect the modules to each other. All modules have a different flow rate of coolant. The Flow Divide block in the Battery Pack subsystem determines the flow rate that reaches each battery module.



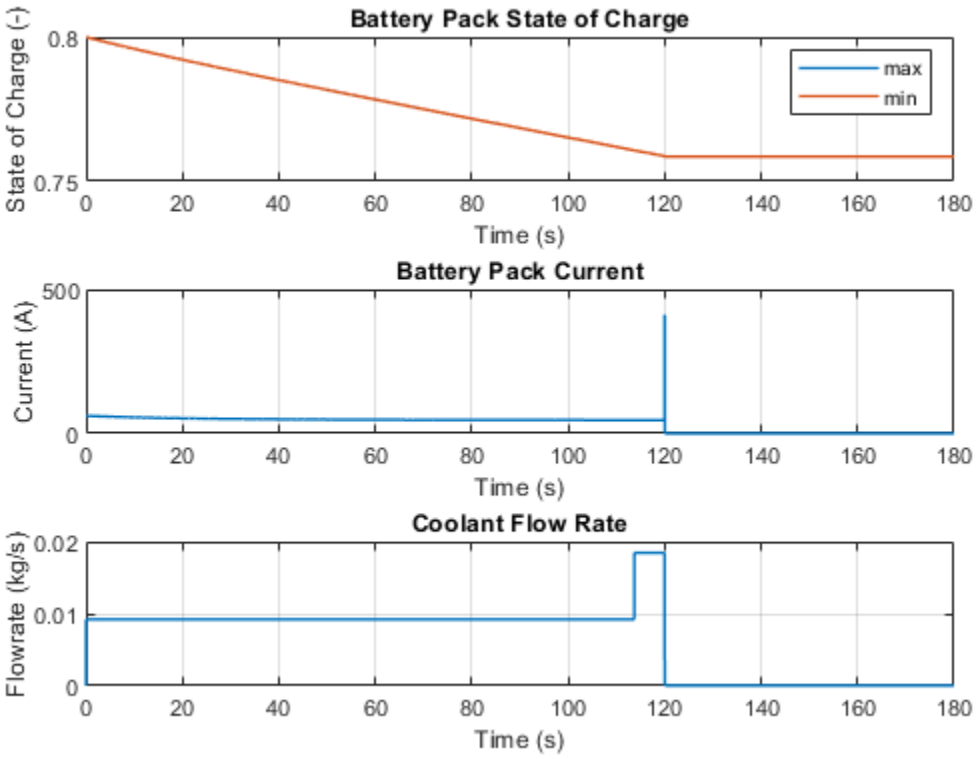

### Control Subsystem Overview

The Control subsystem tracks the minimum and maximum temperatures in the battery pack. This subsystem calculates the flow rate based on the largest value between the difference in the maximum and the minimum cell temperatures in the pack, and the difference in the maximum temperature in the pack and the value at the **Amb** port. For a difference of 10 degrees Celsius or more, *FlwR* is set to 1, else it is linearly scaled till zero, when there is no temperature difference between the different cells and the battery pack temperature is very close the the value set at the **Amb** port. In this example, the coolant inlet temperature, defined in the `coolantTemp` workspace variable in the `ee_lithium_pack_fault_protection_ini.m` file, is constant. The coolant flow is switched off if the battery current is zero or more than `maxDCfaultCurrent` in the file `ee_lithium_pack_fault_protection_ini.m`.




### Simulation Results from Simscape Logging

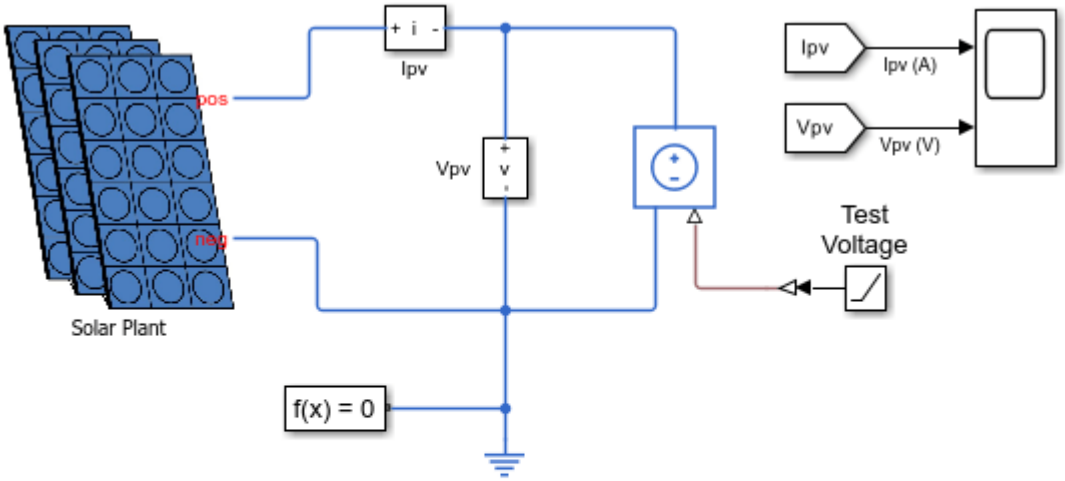
This example simulates a 180 seconds constant load profile. The flow rate increases along with the battery pack temperature, leading to a better cooling of the battery pack. As the fault happens at 120s, the battery current increases and the fuse blows. The coolant flowrate is also turned to zero.


Battery Pack State of Charge
State of Charge (-)
0.8
0.75
max
min
0
20
40
60
80
100
120
140
160
180
Time (s)
Battery Pack Current
Current (A)
500
0
Time (s)
Coolant Flow Rate
Flowrate (kg/s)
0.02
0.01
0
Time (s)

# Analysis of Solar Photovoltaic System Shading

This example shows how to implement shading effects in a solar photovoltaics (PV) plant or module. The solar plant block is created using Simscape™ language. Shading in a solar plant or module occurs when solar irradiance is not uniform across all solar PV modules or cells. You can use this example to study the effects of shading and PV cell junction temperature in a large interconnected solar plant or a single PV module. To improve the maximum power and to protect the solar panel from overheating, the Solar Plant block comprises bypass and blocking diodes. To define the shading, set the values of the **Irradiance** and **Temperature** parameters.

### Model Overview



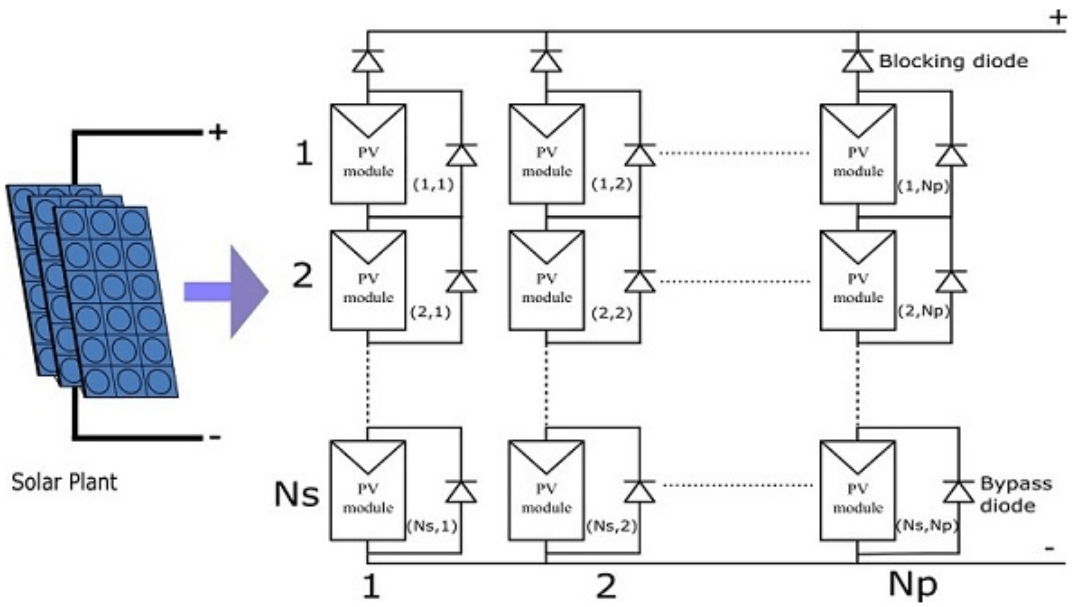

**Analysis of Solar Photovoltaic System Shading**

1 Open script that inputs values from the datasheet
2. Plot current, power curves at varying conditions (see code)
3. Explore simulation results using Simscape Results Explorer
4. Learn more about this example

Copyright 2018-2022 The MathWorks, Inc.

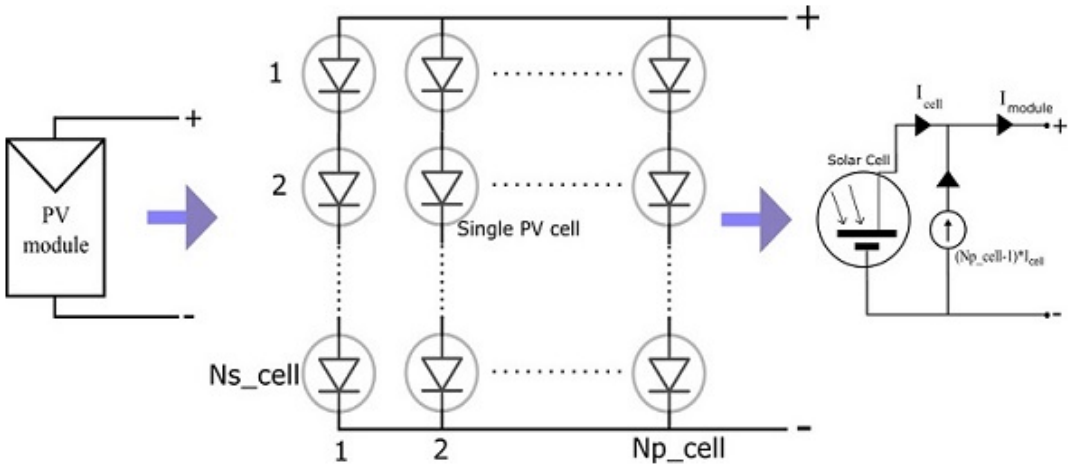
### Solar Plant Block Overview

This figure shows a Solar Plant block. The Solar Plant block comprises **Np** parallel-connected strings. Each string comprises **Ns** series-connected solar PV modules.




### Photovoltaic Solar PV Module Overview

The Solar Plant block comprises `Ns*Np` PV modules. Each solar PV module consists of **`Np_cell`** parallel-connected strings and each string comprises **`Ns_cell`** series-connected solar cells. A Solar Cell block from the Simscape™ Electrical™ library models the solar cell strings. To specify the size of the PV module, define the number of cells, **`Ns_cell`** and **`Np_cell`**, in the modules. To replicate a commercially available solar panel, the solar PV module parameters are directly obtained from a solar panel manufacturer datasheet. For more information about manufacturer datasheet-based parameterization, see the Simscape Electrical 'ee_solar_panel' example.




### Protection Diode

The Solar Plant block comprises both bypass and blocking diodes. A Diode block from the Simscape foundation library models the protection diodes. To bypass the solar PV module in a string that does not have enough irradiance to support the solar PV string current, bypass diodes are connected across PV modules. The blocking diodes isolates the solar PV string that has a lower string voltage. The protection diodes improve the output power and solar PV module lifetime.

Set the **Solar module protection type** parameter to specify the protection diodes in the solar plant: * No protection diode - The solar plant does not have bypass and blocking diodes. * Only bypass diode across modules - The solar plant has bypass diode across each PV modules in all solar strings but does not have blocking diodes between solar PV strings. * Both bypass diode and series module strings blocking diode - The solar plant has both protection diodes.

### Parameters Overview

- **Number of series connected modules, Ns** — Number of series-connected solar PV module in a string, specified as a positive integer. This value must be greater than 0.
- **Number of parallel connected strings of modules, Np** — Number of parallel-connected solar PV strings, specified as a positive integer. This value must be greater than 0.
- **Irradiance(Ns,Np)** — Solar irradiance across each solar PV module. The solar irradiance is assumed to be uniform across all the solar cells in the PV module. The matrix must have **Ns** rows and **Np** columns. Each element in the matrix must be greater than or equal to 0.
- **Cell temperature(Ns,Np)** — Solar cell junction temperature across each solar PV modules. The junction temperature is assumed to be uniform across all solar cells in the PV module. The matrix must have **Ns** rows and **Np** columns.
- **Number of series cells, Ns_cell** — Number of series-connected solar cells in a solar PV module, specified as a positive integer. This value must be greater than 0.
- **Number of parallel cell strings, Np_cell** — Number of parallel-connected solar cell string in a PV module, specified as a positive integer. This value must be greater than 0.

For more information on the other parameters, see the Diode and Solar Cell blocks documentation pages.

### Solar PV Plant Configuration

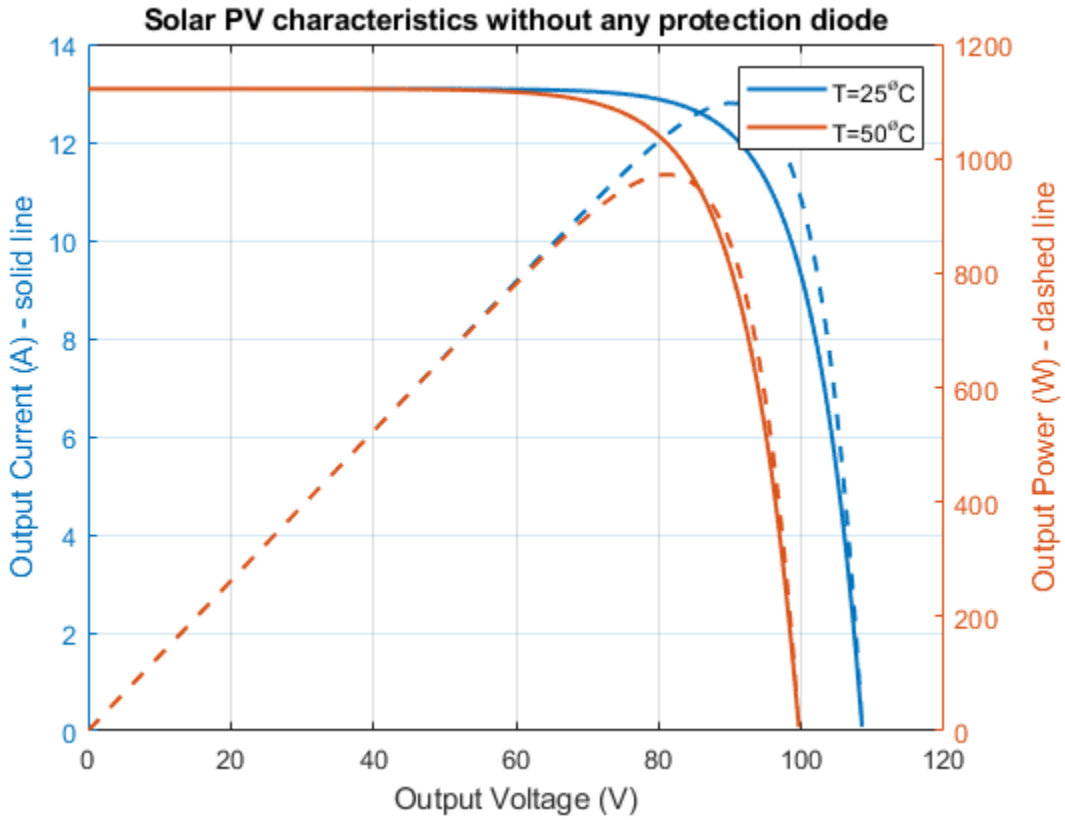
You can configure the Solar Plant block to study the shading effects in both solar PV plant and PV module. To study the shading effects in a single solar PV panel, set the **Number of series cells, Ns_cell** and **Number of parallel cell strings, Np_cell** parameters to 1. To define the number of solar cells in the solar panel, specify the values of the **Number of series connected modules, Ns** and **Number of parallel connected strings of modules, Np** parameters. The **Irradiance(Ns,Np)** matrix and the **Cell temperature (Ns,Np)** matrix parameters are used to define the irradiance and temperature values in each solar cell in the solar panel. To customize the shading field area, specify the proper Solar Plant block **Ns**, **Np**, **Ns_cell**, and **Np_cell** parameters. You can also use the **Ns_cell** and **Np_cell** parameters to define the cluster of PV cell that has uniform irradiance and temperature. You can use the **Ns** and **Np** parameters to define the number of cell clusters that has different irradiance and temperature.

All the solar cells are assumed to be similar across the solar plant.

### Solar Plant I-V Characteristics Without Shading

The plot below shows the I-V and P-V curve of the solar plant with uniform irradiance and temperature. Junction temperature is assumed to be uniform across solar plant.

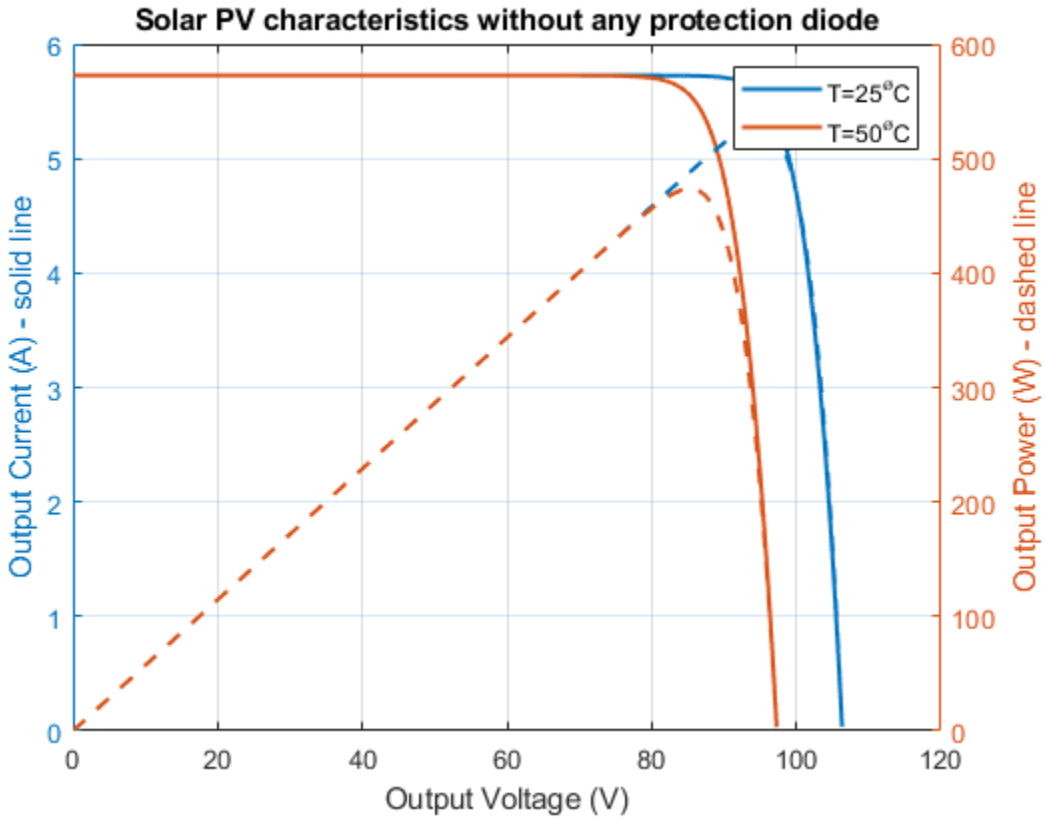
```
Cell junction temperature in degC, maximum power point current in A, voltage in V and Power in W
    JuntionTemperature    MaximumCurrent    MaximumVoltage    MaximumPower
    __________________    ______________    ______________    ____________

            25                12.156            90.226           1096.8
            50                11.981            81.117           971.88
```




**Solar Plant I-V Characteristics with Shading Without Protection Diodes**

The plot below shows the I-V and P-V curve of the solar plant with different irradiance (irradianceMat) across solar PV module without protection diodes. Junction temperature is assumed to be uniform across solar plant. There is a significant reduction in the solar plant maximum output power.

```
Cell junction temperature in degC, maximum power point current in A, voltage in V and Power in W
    JuntionTemperature    MaximumCurrent    MaximumVoltage    MaximumPower
    __________________    ______________    ______________    ____________

            25                5.5836            94.78           529.21
            50                5.5517            85.297          473.55
```

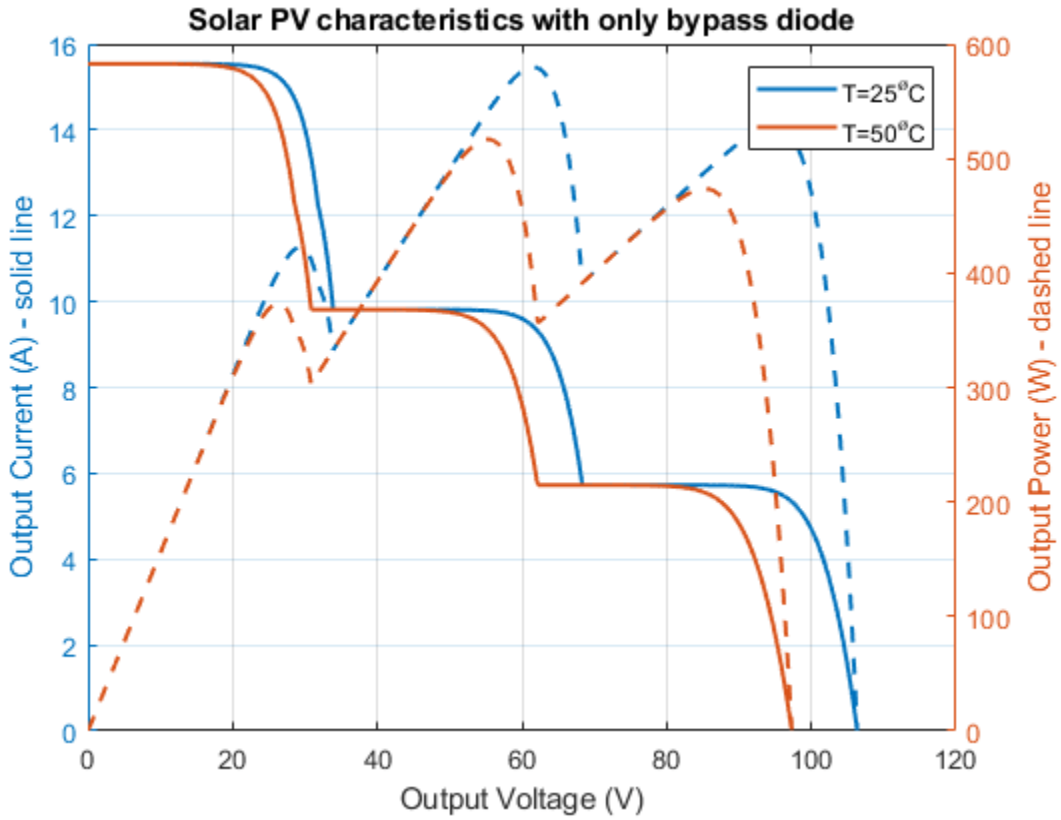



**Solar Plant I-V Characteristics with Shading and Bypass Protection Diodes**

The plot below shows the I-V and P-V curve of the solar plant with different irradiance (irradianceMat) across the solar PV modules with only bypass diodes. The junction temperature is assumed to be uniform across the solar plant. The plot and table below show the improvement in the maximum output power. Many local output power maxima are observed with bypass diode. This effect is very important while evaluating the performance of the Maximum Power Point Tracking (MPPT) algorithm.

The plot below shows that near the highest voltage (large resistive load), all three PV modules in the solar PV strings are supplying power. In the middle voltage range, the PV module with the lowest irradiance, the third PV module in the PV string, got bypassed. In the lower voltage range (low resistance load) only PV modules with highest irradiance supplies power. In this example, this is the first PV module that supplies power to the load, remaining two PV modules are bypassed by the diode.

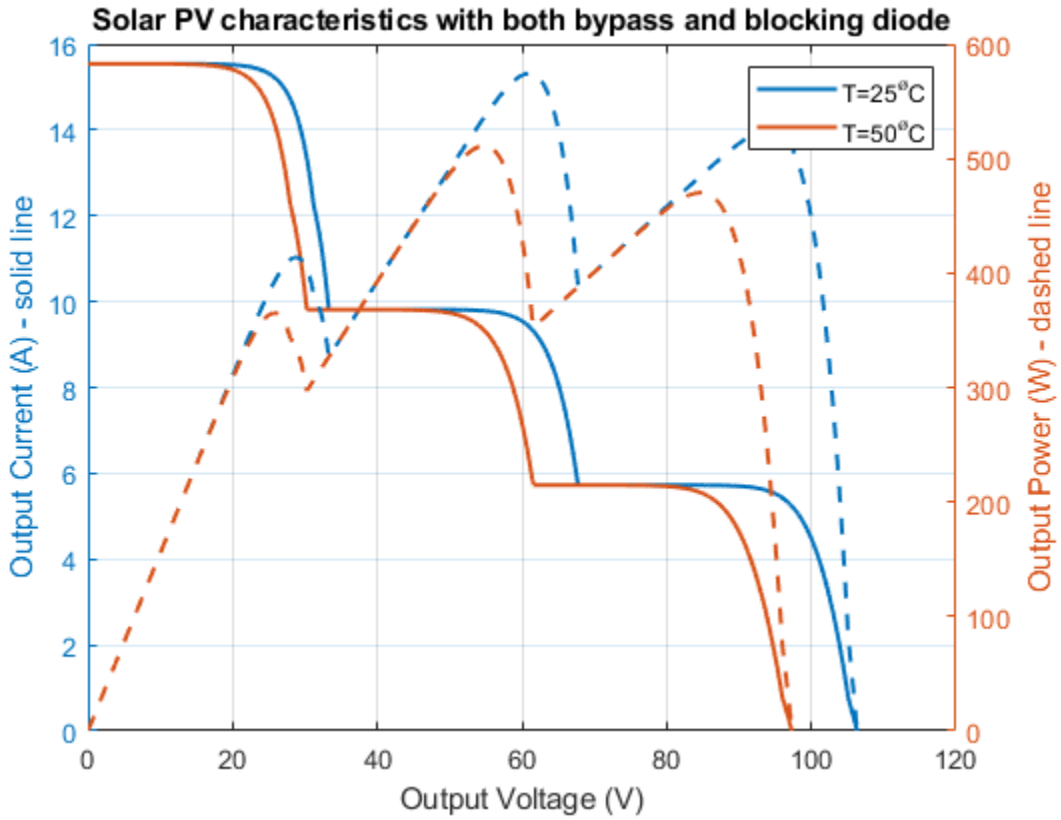
```
Cell junction temperature in degC, maximum power point current in A, voltage in V and Power in W
    JuntionTemperature    MaximumCurrent    MaximumVoltage    MaximumPower
    __________________    ______________    ______________    ____________

            25                9.4275            61.518          579.96
            50                9.3418            55.307          516.66
```




**Solar Plant I-V Characteristics with Shading and Both Protection Diodes**

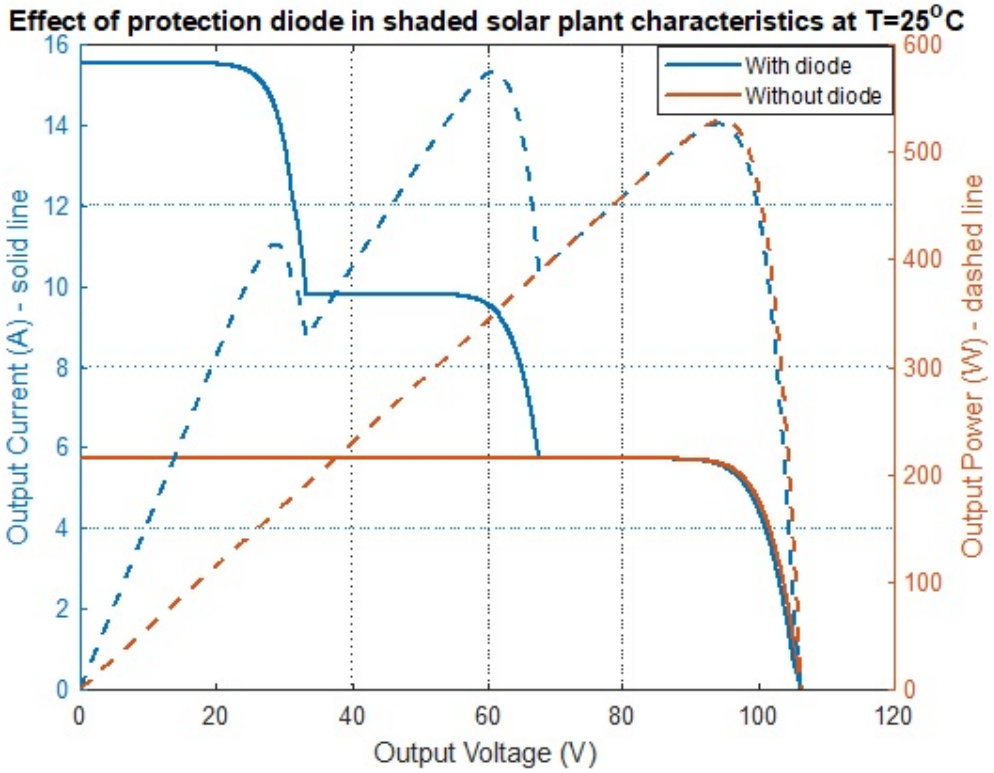
The plot below shows the I-V and P-V curve of the solar plant with different irradiance (irradianceMat) across the solar PV modules with both bypass and blocking protection diodes. The junction temperature is assumed to be uniform across the solar plant. In this case, there is a reduction in voltage due to voltage drop across the blocking diode. This effect also reduces the maximum output power.

```
Cell junction temperature in degC, maximum power point current in A, voltage in V and Power in W
    JuntionTemperature    MaximumCurrent    MaximumVoltage    MaximumPower
    __________________    ______________    ______________    ____________

            25                9.4227            60.944           574.26
            50                9.3358            54.737           511.01
```




**Shaded Solar Plant Characteristics with and Without Protection Diodes**

The plot below shows the I-V and P-V curve of the solar plant with and without protection diode. Protection diodes improves the output power but characteristics has many local output power maxima.


Effect of protection diode in shaded solar plant characteristics at T=25°C
With diode
Without diode
Output Current (A) - solid line
Output Power (W) - dashed line
Output Voltage (V)
0
2
4
6
8
10
12
14
16
0
100
200
300
400
500
600
0
20
40
60
80
100
120

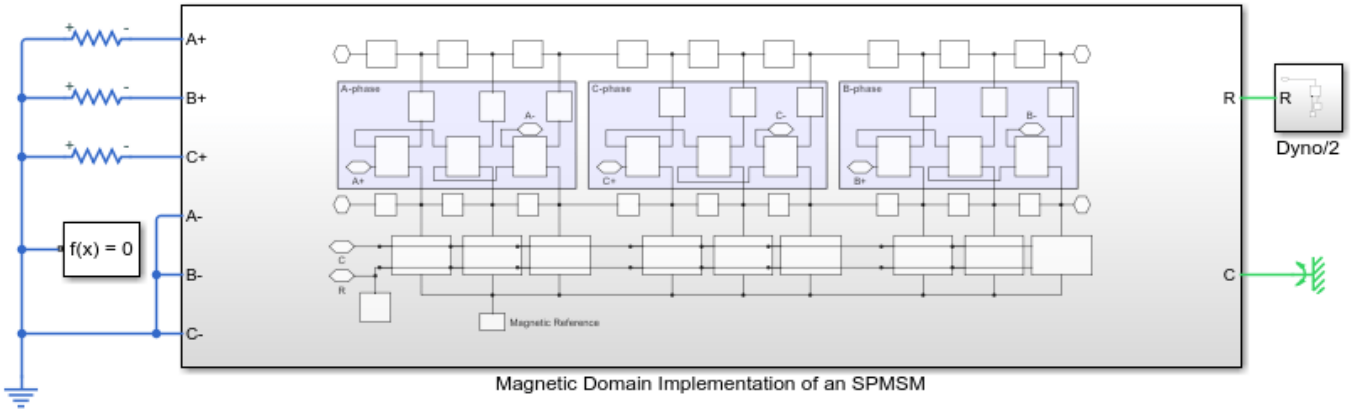
# Faulted PMSM

This example shows how to model a faulted PMSM using Simscape™ Electrical™. Normally when modeling a PMSM, you can represent each winding as a single entity with associated inductance, induced back EMF, and mutual inductive coupling to adjacent windings. However, when a winding fault occurs, the single entity assumption breaks down. To correctly capture the resulting dynamics, you have to model at a winding slot level. This requires modeling in the magnetic domain.

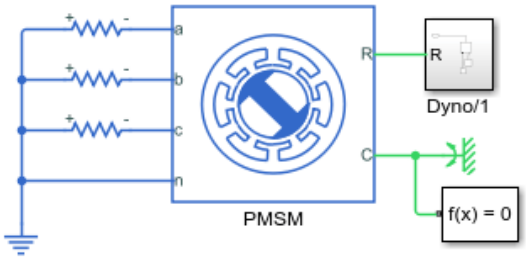
Initially, this example shows how to create a model of a PMSM in the magnetic domain using the Simscape Electrical fundamental blocks for a winding and for a rotor air gap. Then, it shows how to invoke and assess the impact of a number of different winding fault types.

### Model Overview

At the top level the model compares the standard library PMSM block with the custom magnetic-domain subsystem that represents an SPMSM. You can then use the magnetic-domain model to study different fault behaviors, once validated for the no-fault case.




Magnetic Domain Implementation of an SPMSM



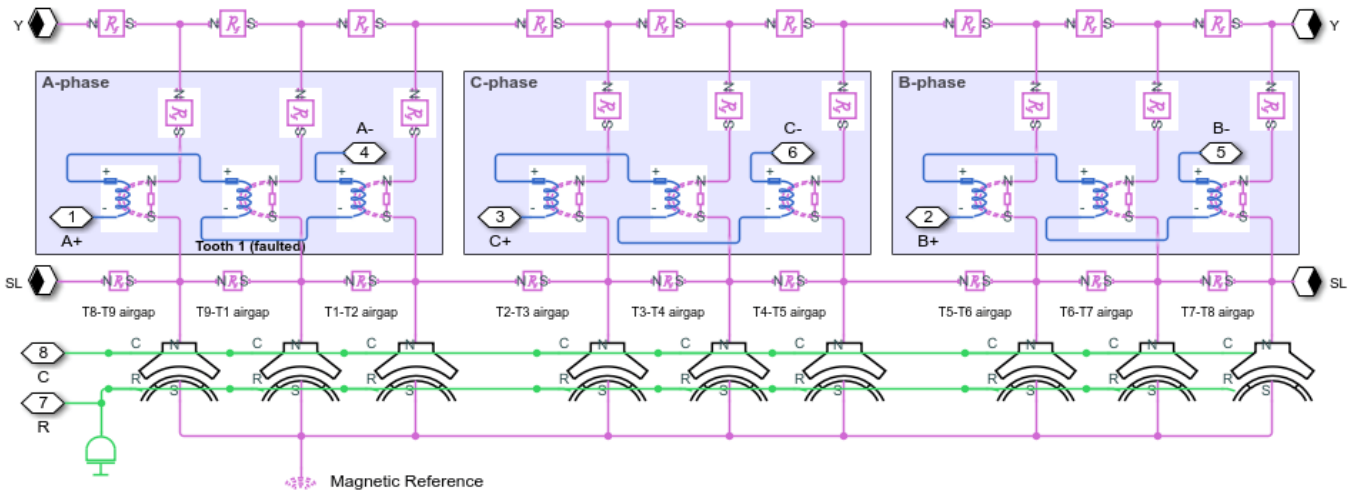

**Faulted PMSM**

1. Plot winding currents and torque (see code)
2. Configure stator tooth winding fault: (see code)
   None, Open-circuit, Ground, Isolated turns
3. Set parameters for model (see code)
4. Explore simulation results using Simscape Results Explorer
5. Learn more about this example

Copyright 2013-2022 The MathWorks, Inc.

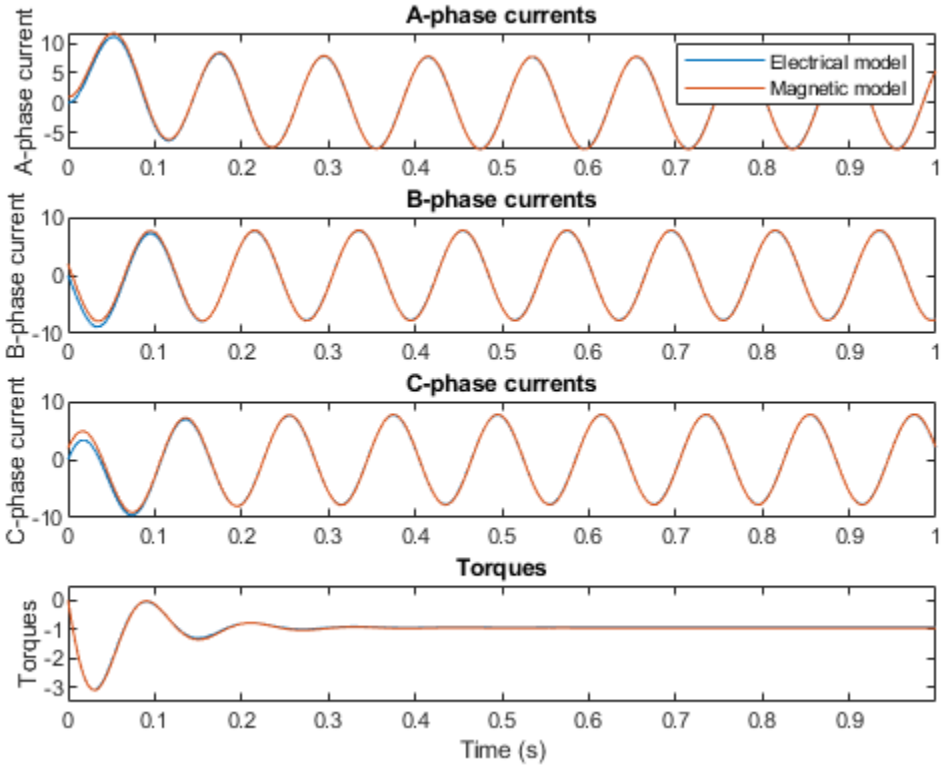
### Magnetic Domain SPMSM Subsystem Overview

The figure shows the model architecture of a surface mount permanent magnet synchronous motor (SPMSM) with ten rotor poles and nine stator winding slots. The reference slot, Slot 1, is the middle of the three A-phase slots. Faults are introduced in this slot.



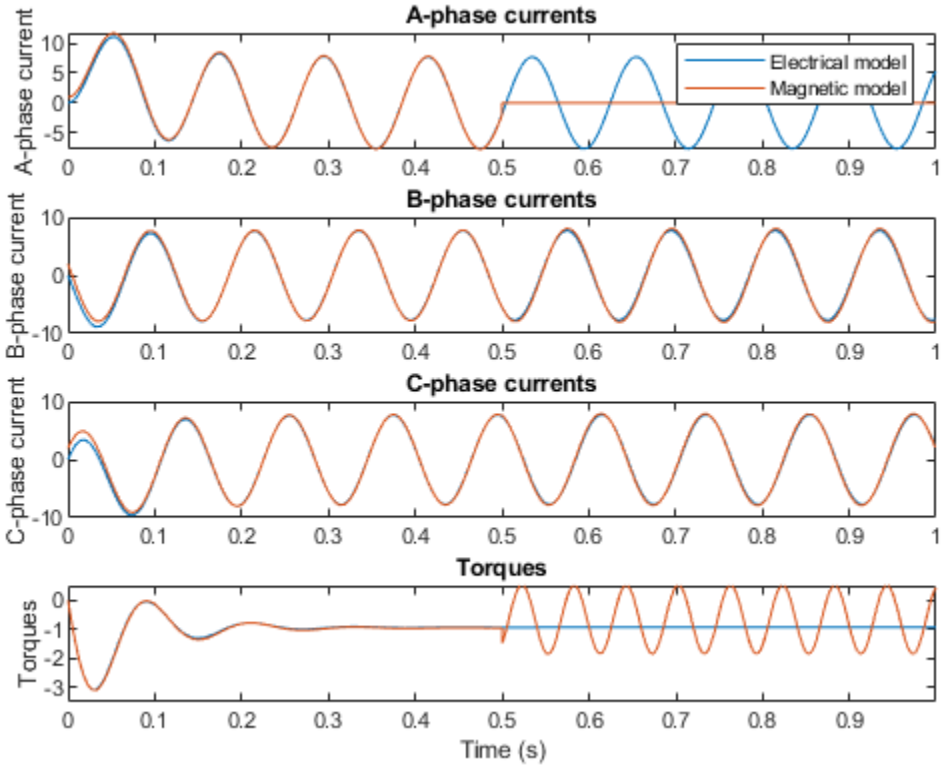

### Validation Against Reference Model

The plot below shows simulation results from the custom magnetic domain model superimposed on simulation results from the Simscape Electrical standard PMSM library block.



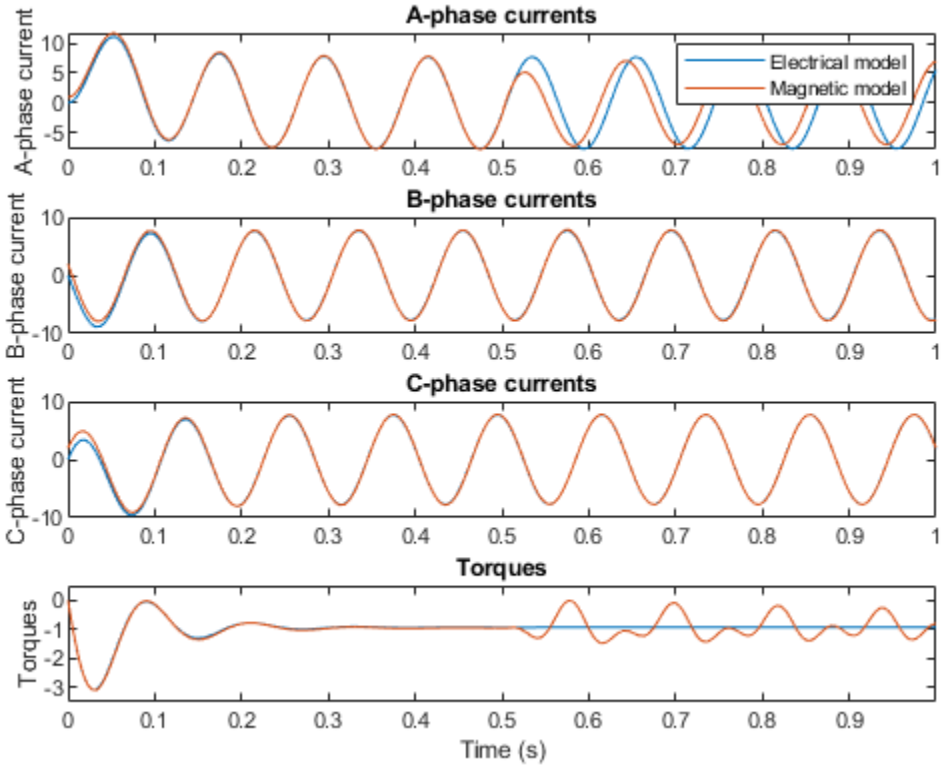

### Open Circuit Fault

The Slot 1 turns are open-circuited half-way through the simulation.



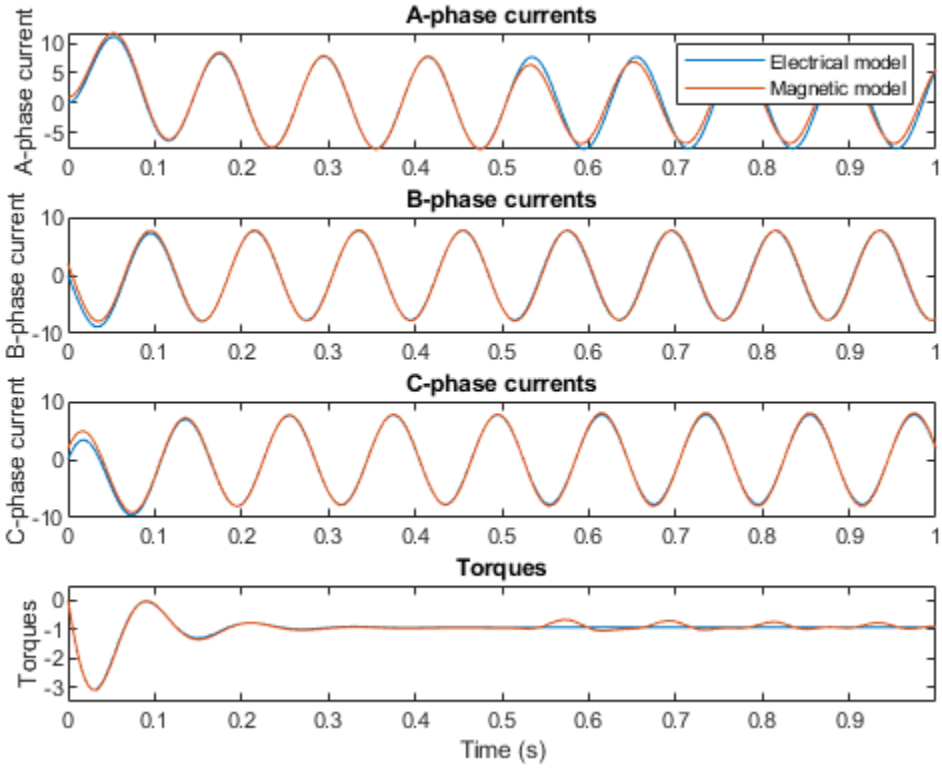

### Ground Fault

The midpoint of the Slot 1 turns is shorted to ground half-way through the simulation.




### Isolated Turns Fault

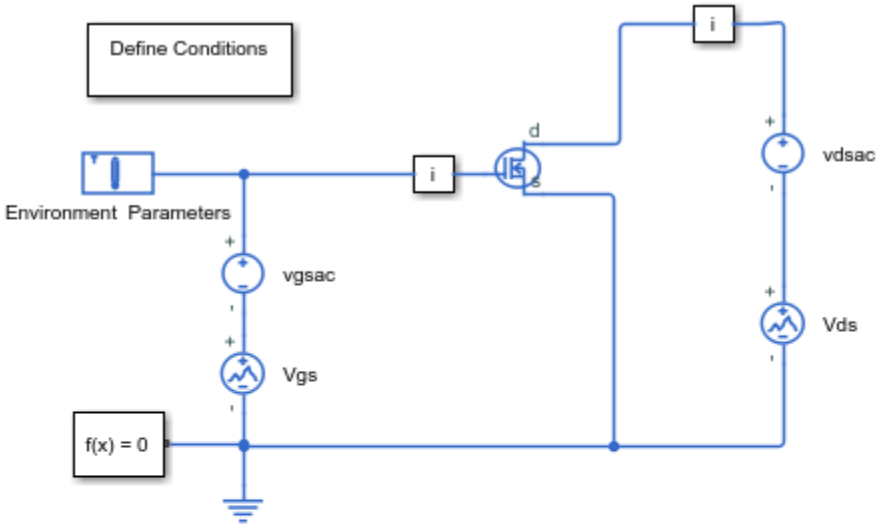
Half of the Slot 1 turns become isolated half-way through the simulation.


A-phase currents
A-phase current
Electrical model
Magnetic model
B-phase currents
B-phase current
C-phase currents
C-phase current
Torques
Torques
Time (s)

# Parameterize a Lookup Table-Based MOSFET from SPICE

This example shows how to use the SPICE simulation results of a metal–oxide–semiconductor field-effect transistor (MOSFET) to set the parameter values of an N-Channel MOSFET (Lookup table-based) in Simscape™. Then, it compares the N-Channel MOSFET characteristics in Simscape with the SPICE netlist simulation results.

### Open the N-Channel MOSFET (Lookup Table-Based) Model




**Parameterize a Lookup Table-Based MOSFET from SPICE**

1. MOSFET curves: Define Conditions
2. SPICE subcircuit (see SPICE netlist)
3. Set lookup table parameters (see code)
4. Check results (see code)
5. Explore simulation results using Simscape Results Explorer
6. Learn more about this example

Copyright 2020-2022 The MathWorks, Inc.

### Open the MOSFET Subcircuit

The subcircuit contains a SPICE model of an N-Channel power transistor. To open it, in the MATLAB command window, enter edit IAUC100N04S6L014.cir.

### Create a Transfer Characteristics Netlist

You can create a SPICE netlist that specifies the target operating range for the subcircuit. The IAUC100N04S6L014_idvgs.net netlist simulates the transfer characteristics with drain-source voltage list.

```
* Transfer characteristic of MOSFET IAUC100N04S6L014
.opt DampInductors=0 Thev_Induc=1 Gfarad=0 Gfloat=0 reltol=1e-05 abstol=1e-05 vntol= 1e-05 Gmin=(
```

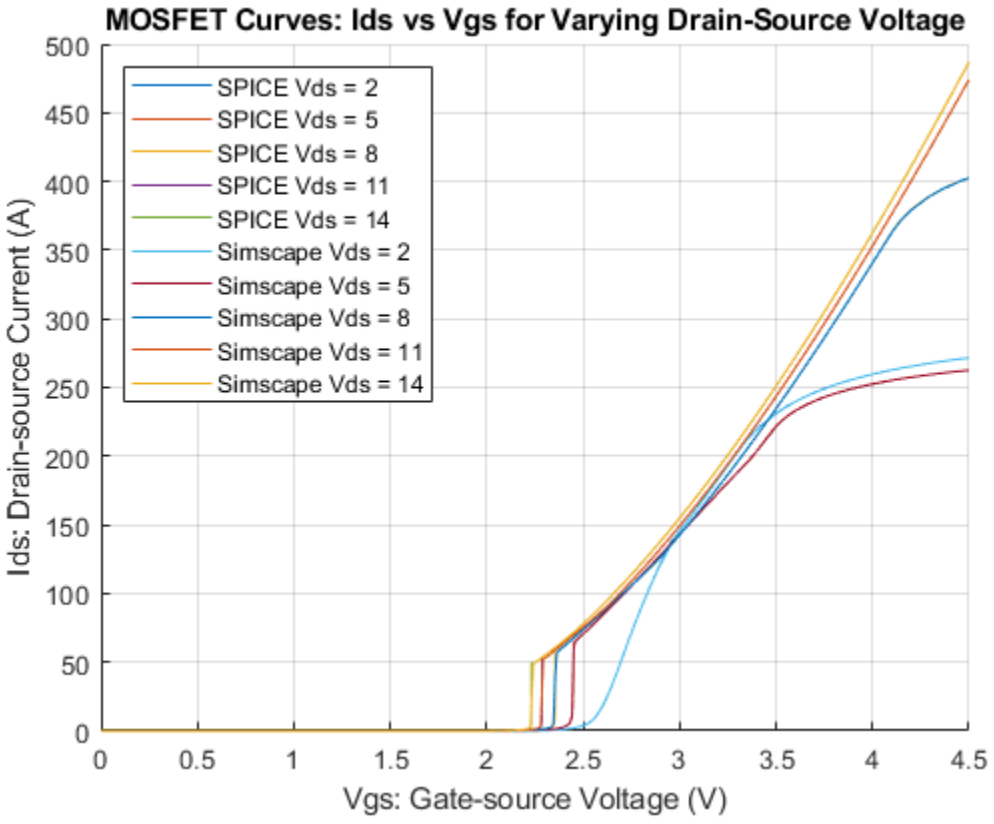
```
X1 dut1 dut2 dut3 dut4 dut5 IAUC100N04S6L014
V1 dut1 0 2
V2 dut2 0 pwl(0 0 20 4.5)
V3 dut3 0 0
V5 dut5 0 27
.step V1 list 2 5 8 11 14
.tran 1m 20
.lib IAUC100N04S6L014.cir
.end
```

### Set Lookup Table Parameters for Transfer Characteristics

To obtain the required MOSFET transfer characteristics, run the simulation in SPICE. The `ee_mosfet_tabulated_setparam` script uses the simulation results stored in the `IAUC100N04S6L014_idvgs1.raw` raw-files to set the parameter values for the N-Channel MOSFET (Lookup Table-Based). To set the MOSFET channel parameters, in the Define Condition tab, check the `Set MOSFET channel parameters` box, or call the `set_param` function.

### Compare Transfer Characteristics

The plot below compares the transfer characteristics of the N-Channel MOSFET (Lookup table-based) with the SPICE subcircuit simulation results.



### Create an Output Characteristics Netlist

To parameterize an N-Channel MOSFET (Lookup table-based), you can also use an output characteristics SPICE data. The IAUC100N04S6L014_idvds.net netlist simulates the output characteristics with gate-source voltage list.

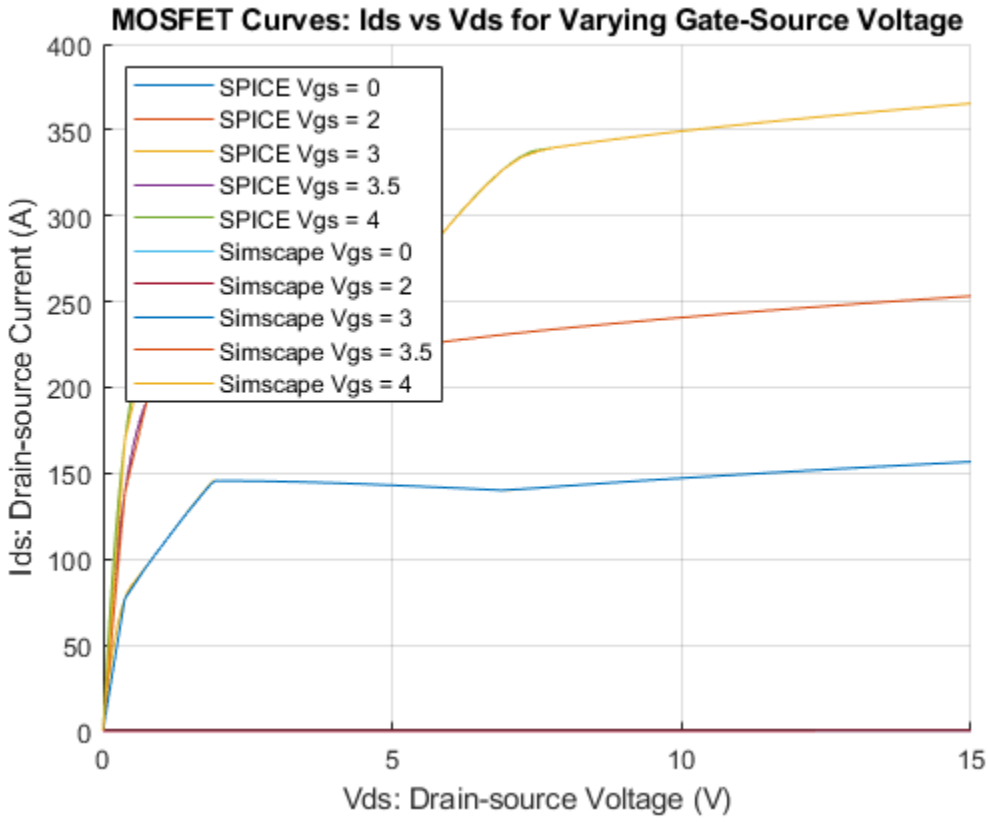
```
* Output characteristic of MOSFET IAUC100N04S6L014
.opt DampInductors=0 Thev_Induc=1 Gfarad=0 Gfloat=0 reltol=1e-05 abstol=1e-05 vntol= 1e-05 Gmin=0

X1 dut1 dut2 dut3 dut4 dut5 IAUC100N04S6L014
V1 dut1 0 pwl(0 0 20 15)
V2 dut2 0 2
V3 dut3 0 0
V5 dut5 0 27
.step V2 1 4 0.5
.tran 2m 20
.lib IAUC100N04S6L014.cir
.end
```

### Set Lookup Table Parameters for Output Characteristics

The `IAUC100N04S6L014_idvds1.raw` raw file stores the output characteristics simulated in SPICE. To change the raw file path and plot type, specify the relevant parameters in the Define Condition tab, or call the `set_param` function.

### Compare Output Characteristics

The plot below compares the output characteristics of the N-Channel MOSFET (Lookup table-based) with the SPICE subcircuit simulation results.




### Create a Capacitance vs Drain-source Voltage Netlist

To parameterize the tabulated capacitance of an N-Channel MOSFET, create a SPICE netlist that simulates the input capacitance (Ciss), the reverse transfer capacitance (Crss), and the output capacitance versus drain-source voltage with specific gate-source voltage. The IAUC100N04S6L014_cvds.net netlist simulates the I-V results of Ciss and Crss by a small AC signal on gate-source voltage and the I-V results of Coss by a small AC signal on drain-source voltage.

```
* Ciss Crss Coss measurement of MOSFET IAUC100N04S6L014
.opt DampInductors=0 Thev_Induc=1 Gfarad=0 Gfloat=0 reltol= 1e-05 abstol= 1e-05 vntol= 1e-05 Gmi

* Ciss Crss I-V with Vgs small AC
X1 dut11 dut12 dut13 dut14 dut15 iauc100n04s6l014
V11 dut11 0 sine({VDC} 0 1e6)
V12 dut12 0 sine(0 0.05 1e6)
V13 dut13 0 0
V15 dut15 0 27

* Coss I-V with Vds small AC
X2 dut21 dut22 dut23 dut24 dut25 iauc100n04s6l014
V21 dut21 0 sine({VDC} 0.05 1e6)
V22 dut22 0 sine(0 0 1e6)
V23 dut23 0 0
V25 dut25 0 27
*.param VDC = 0
.step param VDC 0 30 1
```

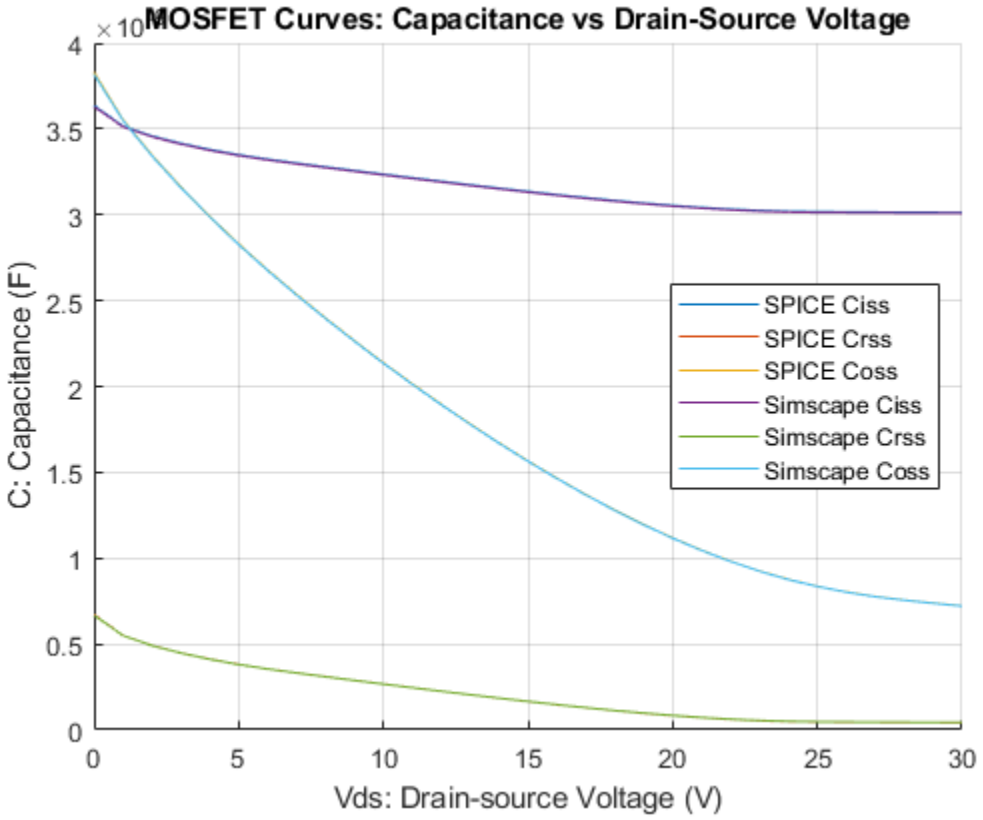
```
.tran 1e-7 5e-6 0 2e-8
.inc IAUC100N04S6L014.cir
.end
```

### Set Lookup Table Parameters for Capacitance

The `IAUC100N04S6L014_cvds1.raw` raw file stores the capacitance I-V results simulated in SPICE. To set MOSFET capacitance parameters, in the `Define Conditions` tab, check the `Set MOSFET capacitance parameters` box, or call the `set_param` function.

### Compare Ciss, Crss and Coss Capacitance

The plot below compares the Ciss, Crss, and Coss capacitance of the N-Channel MOSFET with the SPICE subcircuit simulation results.




### Create a Source-drain Forward Diode Characteristics Netlist

To parameterize the tabulated forward diode of a N-Channel MOSFET, create a SPICE netlist with the MOSFET operating in the off region and with reverse biasing drain-source voltage. The IAUC100N04S6L014_diode.net netlist simulates the current voltage characteristics of a forward diode.

```
* Source-Drain Forward Diode of MOSFET IAUC100N04S6L014
.opt DampInductors=0 Thev_Induc=1 Gfarad=0 Gfloat=0 reltol=1e-05 abstol=1e-05 vntol= 1e-05 Gmin=0

X1 dut1 dut2 dut3 dut4 dut5 IAUC100N04S6L014
```

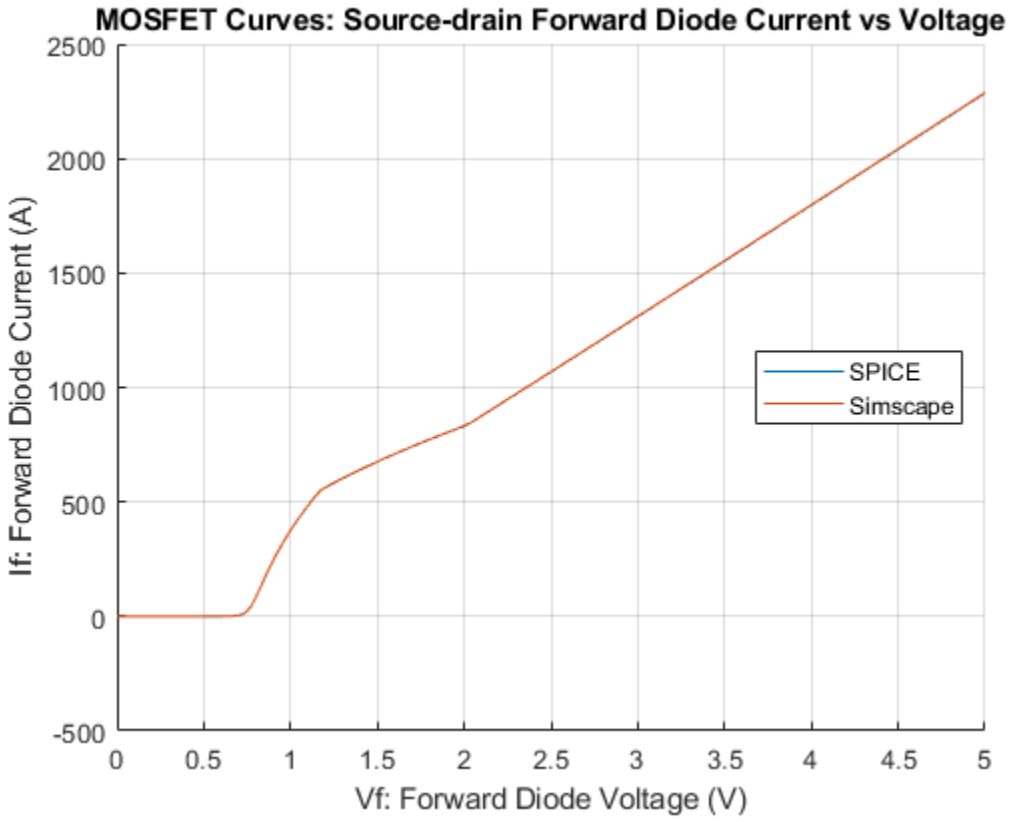
```
V1 dut1 0 pwl(0 0 20 -5)
V2 dut2 0 -10
V3 dut3 0 0
V5 dut5 0 27
.tran 0.1 20
.lib IAUC100N04S6L014.cir
.end
```

### Set Lookup Table Parameters for Source-drain Forward Diode

The `IAUC100N04S6L014_diode1.raw` raw file stores the foward diode characteristics simulated in SPICE. To set MOSFET diode parameters, in the `Define Conditions` tab, check the `Set MOSFET diode parameters` box, or call the `set_param` function

### Compare Forward Diode Characteristics

The plot below compares the forward diode characteristics of the N-Channel MOSFET with the SPICE subcircuit simulation results.

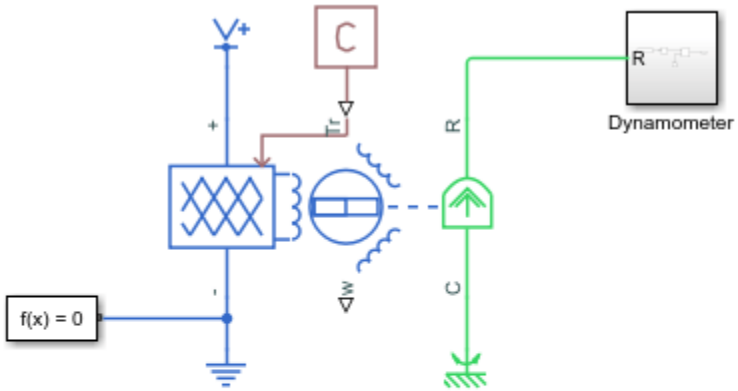


# Import Efficiency Map Data from Motor-CAD

This example shows how you can import efficiency map data from Motor-CAD to parameterize the Simscape™ Electrical™ Motor & Drive (System Level) block. This provides fast system-level simulation capability of a motor drive whilst still making accurate predictions about losses.

Copyright 2020 The MathWorks, Inc. Portions Copyright Motor Design Ltd 2020.

### Model Overview



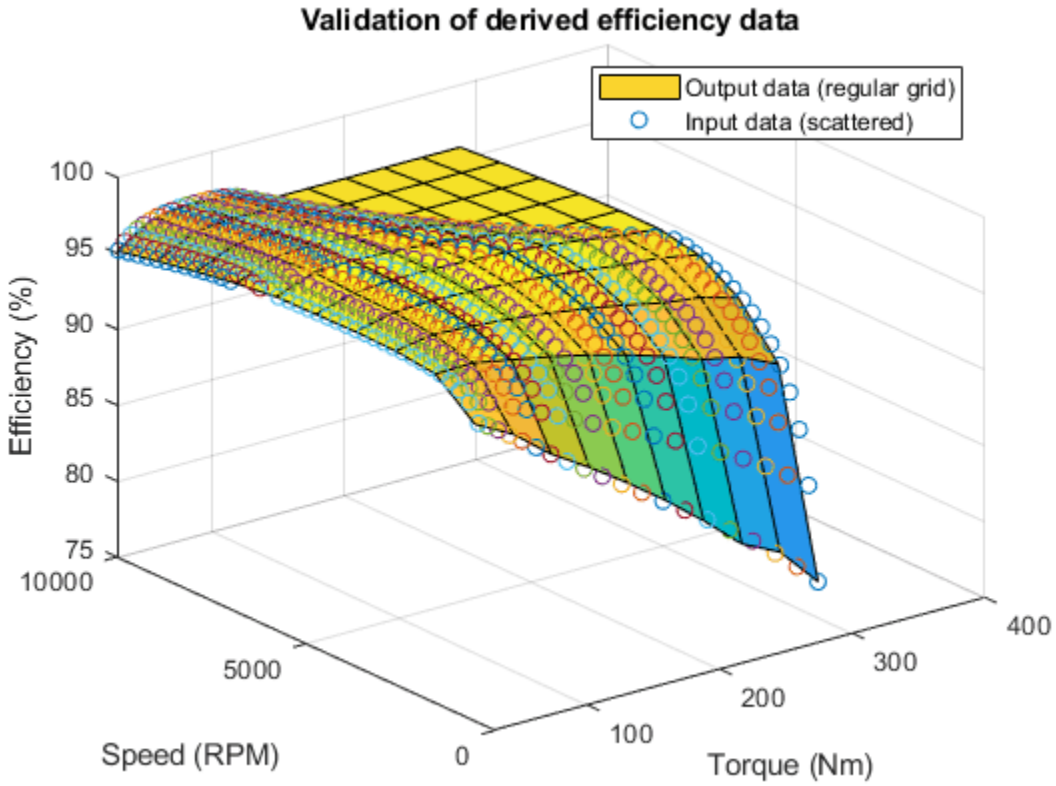

**Import Efficiency Map Data from Motor-CAD**

1. Efficiency data: Open MATLAB script, run script
2. Plot efficiency (see code)
3. Simulate and validate the model (see code)
4. Explore simulation results using Simscape Results Explorer
5. Learn more about this example

Copyright 2017-2022 The MathWorks, Inc.

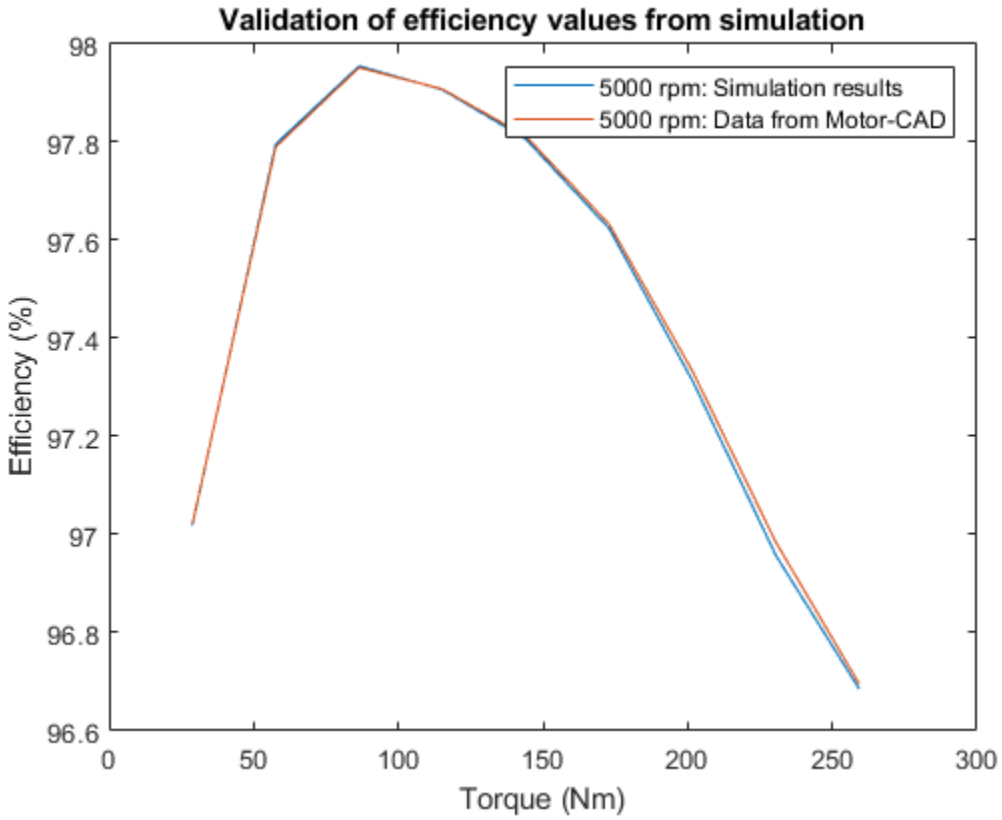
### Efficiency Data

The plot below shows the original scattered efficiency data from simulating a drive cycle in Motor-CAD plus the derived regular grid efficiency data for use by the Motor & Drive (System Level) block. The MATLAB® `meshgrid` and `griddata` functions determine the regular grid data.




### Validation

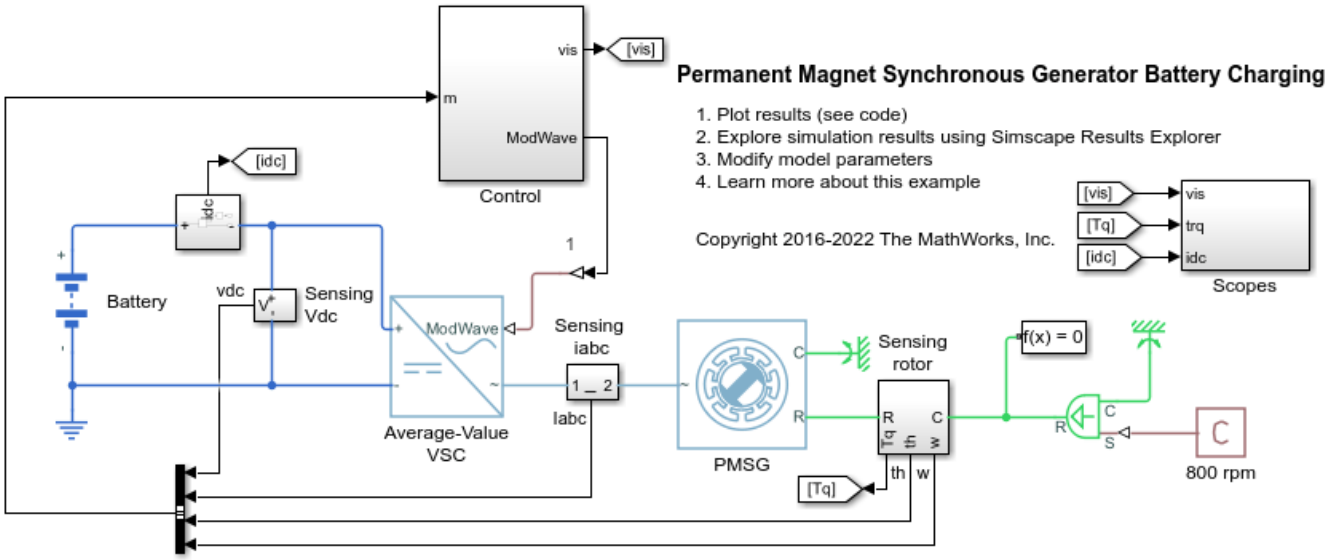
The plot below shows efficiency values extracted from simulation results using the parameterized Simscape Electrical block.


Validation of efficiency values from simulation
5000 rpm: Simulation results
5000 rpm: Data from Motor-CAD
Efficiency (%)
Torque (Nm)
98
97.8
97.6
97.4
97.2
97
96.8
96.6
0
50
100
150
200
250
300

# Permanent Magnet Synchronous Generator Battery Charging

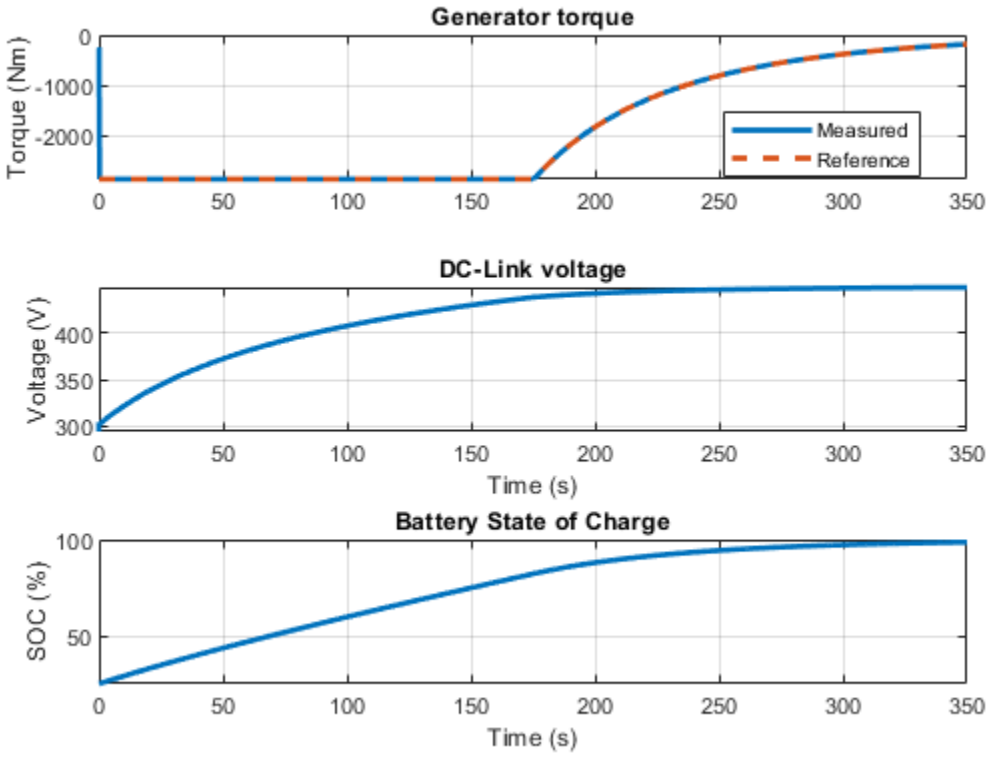
This example shows how to use a permanent magnet synchronous generator (PMSG) to charge a battery. An ideal angular velocity source is used to maintain the rotor speed constant. The Control subsystem uses field oriented control to regulate the torque of the PMSG. The torque reference is obtained as a function of dc-link voltage. The initial battery state of charge is 25%. The Scopes subsystem contains scopes that allow you to see the simulation results.

### Model




### Simulation Results from Simscape Logging

The plot below shows the generator torque and the battery voltage and state of charge.



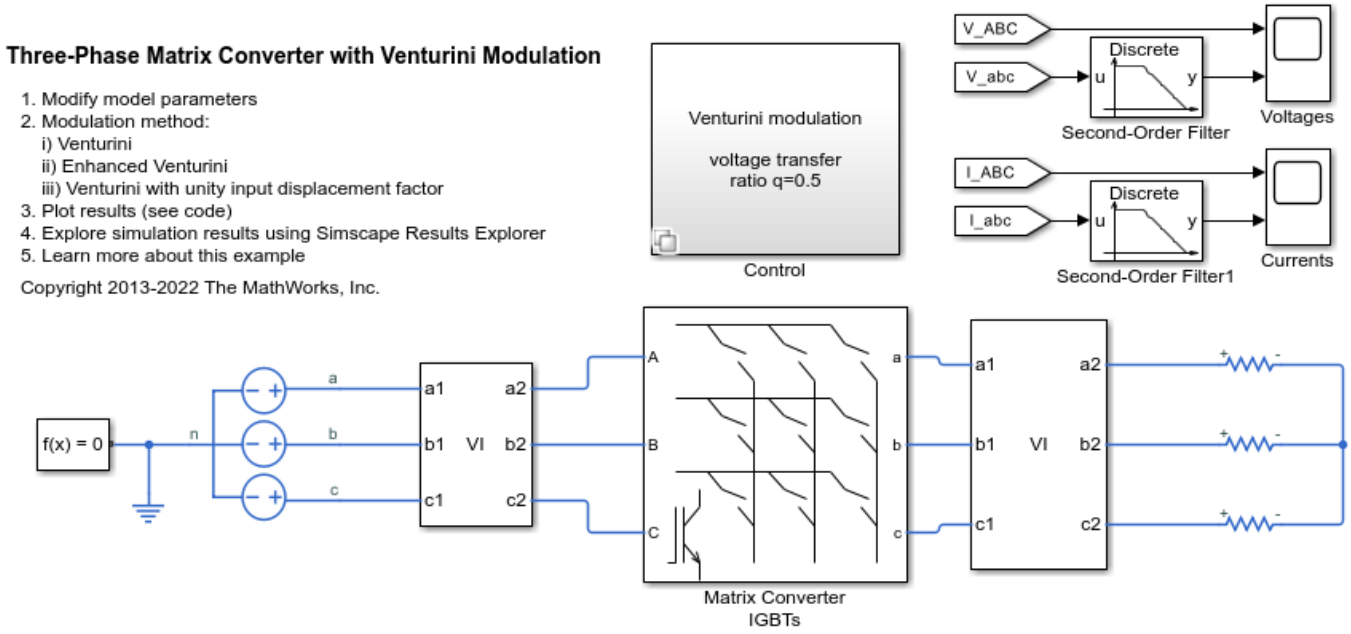

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 50 microseconds.

# Three-Phase Matrix Converter with Venturini Modulation

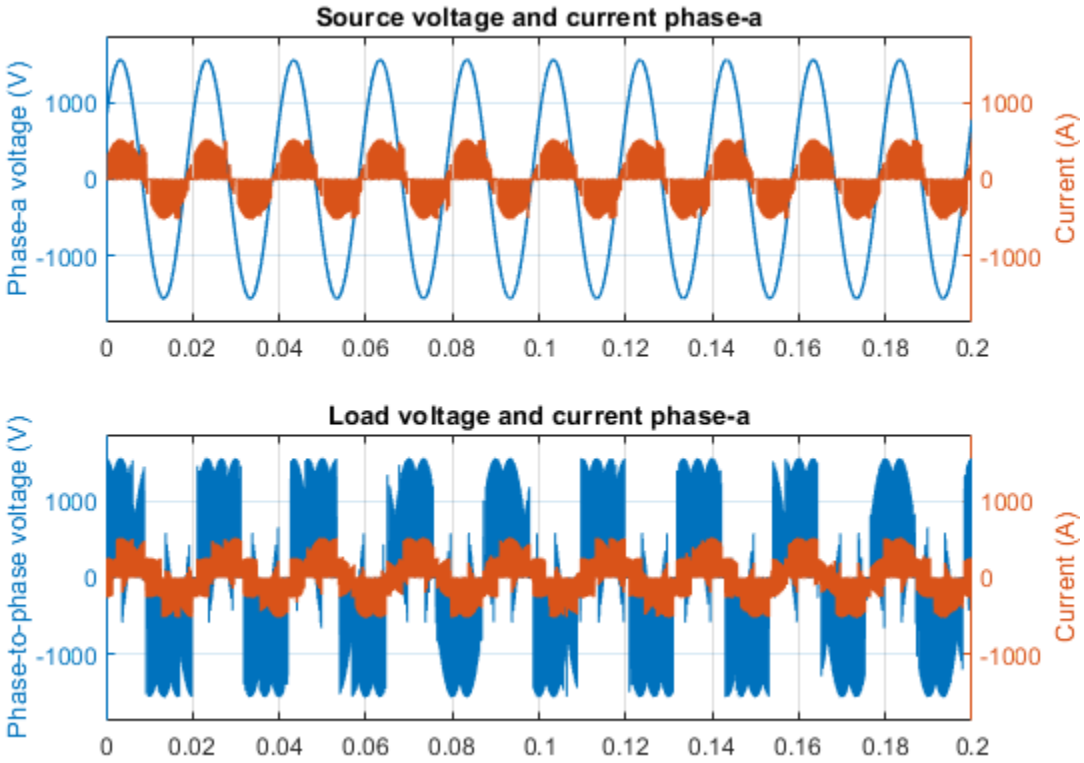
This example shows how to use Venturini modulation techniques to compute the duty cycles and logic statements of a three-phase matrix converter that drives a static load. The control subsystem implements three different modulation algorithms: Venturini modulation, third harmonic enhanced Venturini modulation, and third harmonic injection Venturini modulation with unity input displacement factor. The maximum voltage transfer ratio between input and output depends on the modulation technique and it is equal to either q=0.5 or q=0.866. The Scope blocks show the voltages and currents V_ABC, V_abc, I_ABC, and I_abc, where _UPPERCASE is used for inputs and _LOWERCASE for outputs.

### Model

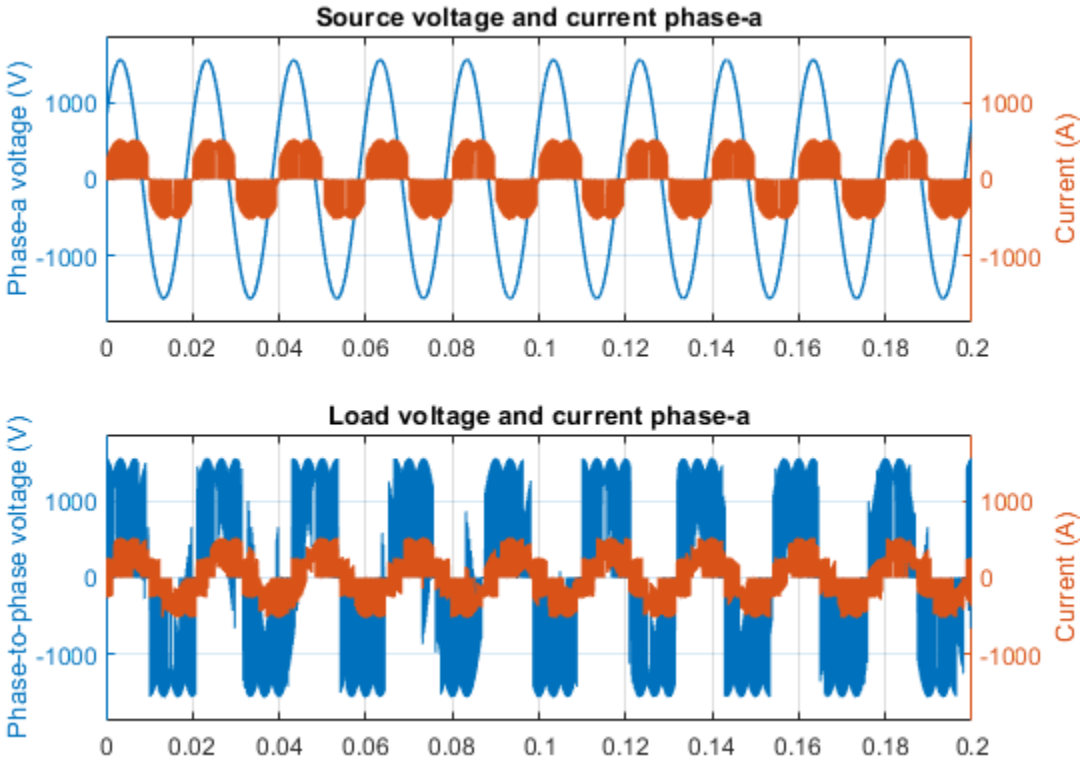



### Simulation Results from Simscape Logging

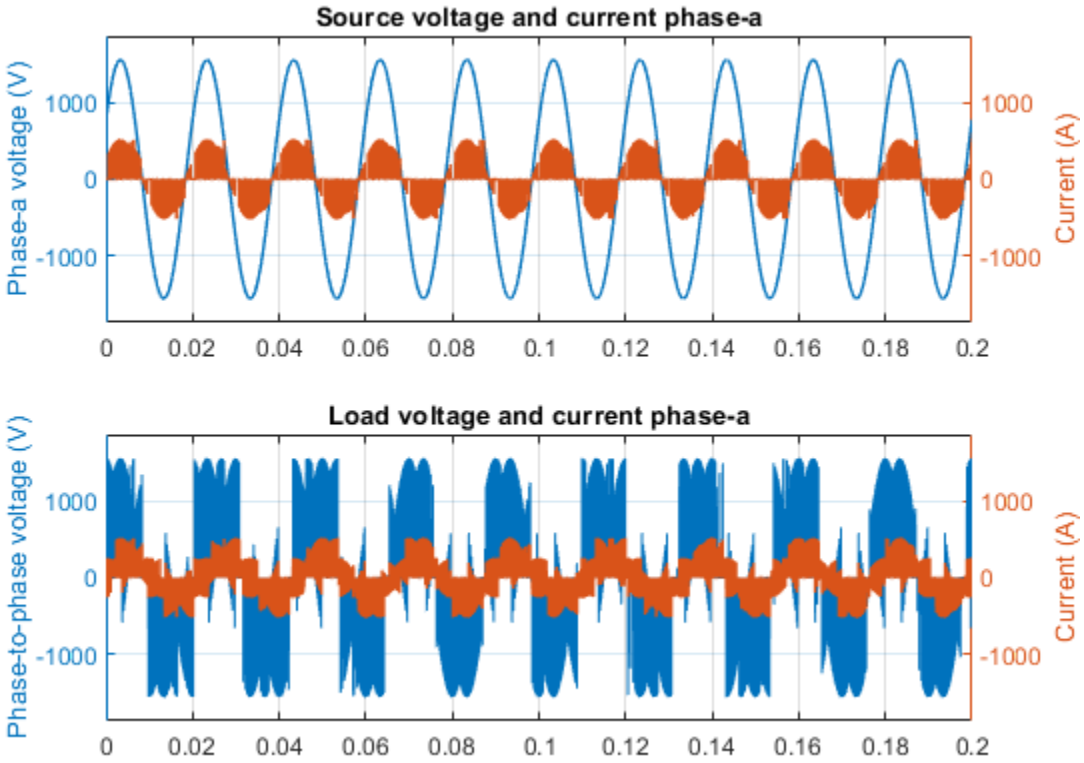
The plot below shows the voltages and currents for the source and load using Venturini method.




The plot below shows the voltages and currents for the source and load using enhanced Venturini method.



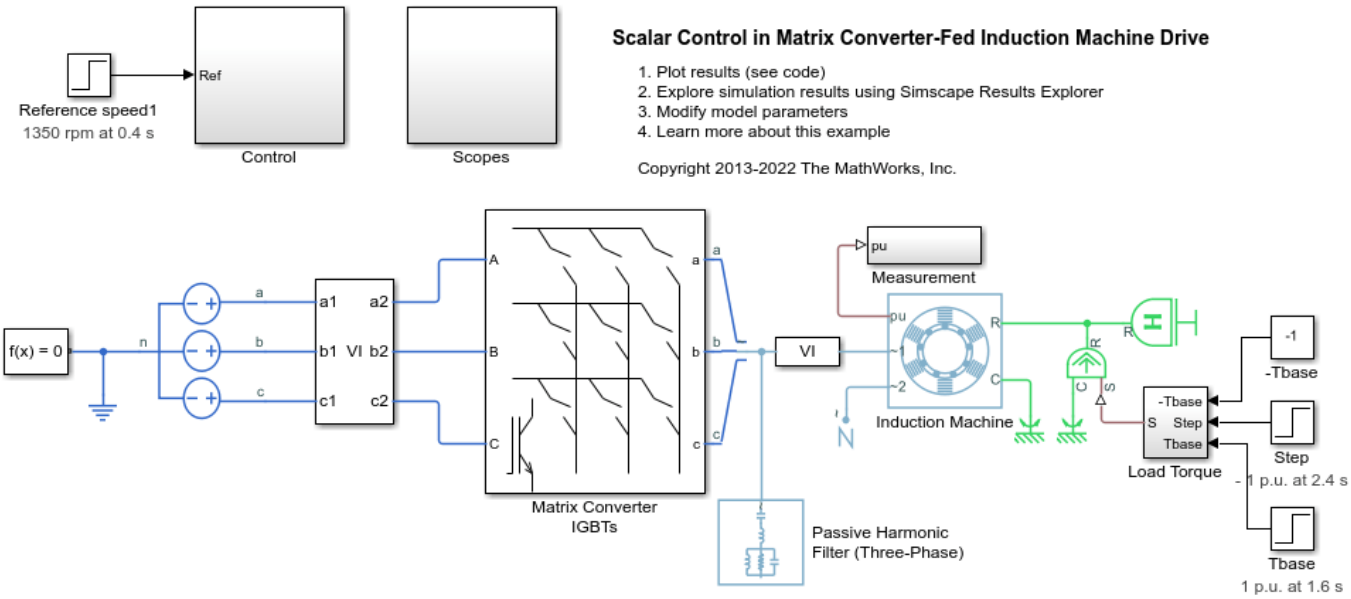

The plot below shows the voltages and currents for the source and load using Venturini method with unity input displacement factor.


Source voltage and current phase-a
Phase-a voltage (V)
Current (A)
1000
0
-1000
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
0.18
0.2
Load voltage and current phase-a
Phase-to-phase voltage (V)
Current (A)

# Scalar Control in Matrix Converter-Fed Induction Machine Drive

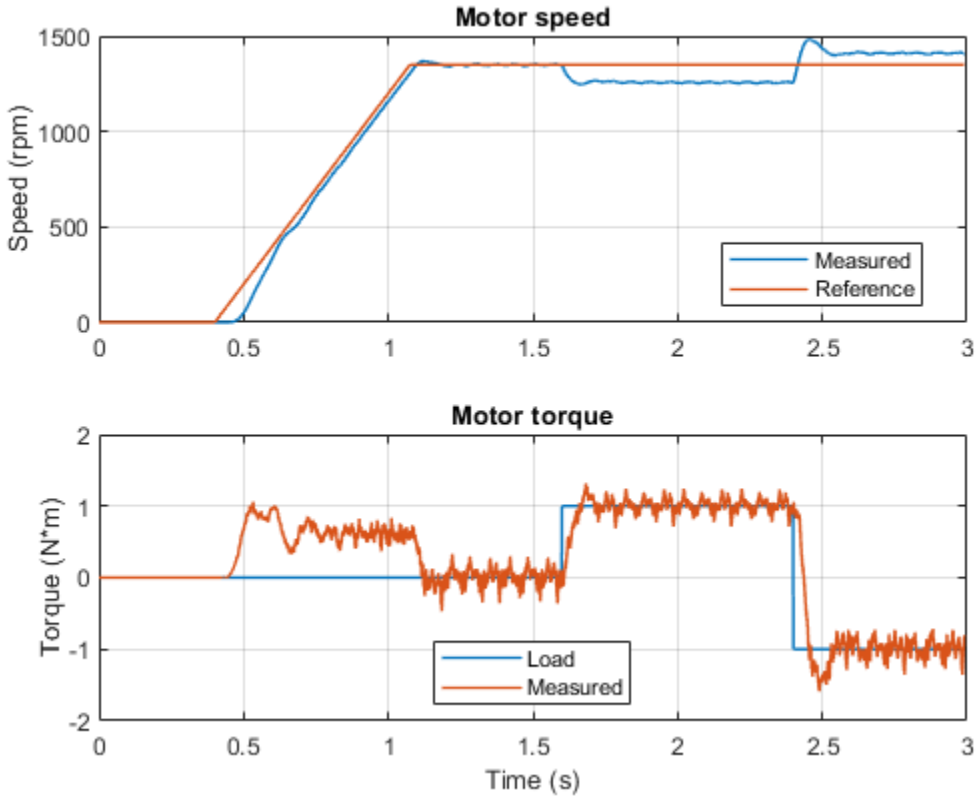
This example shows how to control the rotor speed in a matrix converter-fed induction machine drive by using the scalar V/f control method. To generate three-phase voltage with reference frequency, the controller maintains a constant voltage-to frequency ratio though scalar V/f control. A three-phase voltage source with fixed amplitude and frequency feeds the induction machine through a three-phase matrix converter. The matric converter is controlled using third harmonic injection Venturini modulation with unity input displacement factor. The induction machine operates in both motoring and generating modes. The Scopes subsystem contains scopes that allow you to see the simulation results.

### Model




### Simulation Results from Simscape Logging

The plot below shows the requested and measured speed for the test, as well as the torque in the electric drive.


Motor speed
Speed (rpm)
1500
1000
500
0
0
0.5
1
1.5
2
2.5
3
Measured
Reference
Motor torque
Torque (N*m)
2
1
0
-1
-2
Load
Measured
Time (s)

# Piezo Bender Energy Harvester

This example shows how to model a device that harvests energy from a vibrating object by using a piezo bender. The device uses this energy to charge a battery and power a load.

These devices are common in low-power applications that require energy autonomy, such as wearable devices or sensors in vehicles.

In this example, the vibration sources from which the energy is harvested are: * **Sinusoidal** — The vibration source is an engine of a vehicle rotating at constant speed. * **Chirp signal** with increasing or decreasing frequency — The vibration source is an engine that is ramping up or down in speed.

### Model Overview

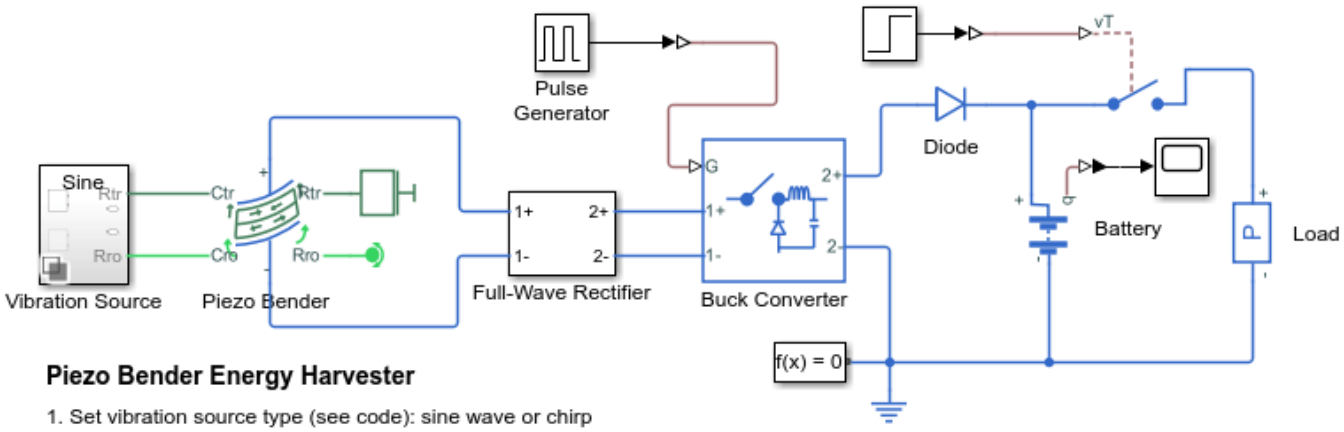
This energy harvester consists of a piezo bender, a rectifier, and a DC-DC converter.

The left end of the piezo bender is clamped to a vibrating object, forcing the motion. The right end of the piezo bender is connected to an extra mass. Due to the elasticity, mass, and inertia of the piezo bender, the motion of the right end is not synchronous to the left end. The deformations produce then a charge and voltage across the electrical terminals of the piezo bender, that are harvested into power.

The full-wave rectifier transforms the AC power generated by the piezo bender into DC power. It comprises four diodes and a capacitor that acts as a filter to smooth the DC voltage.

The buck converter regulates the voltage to transfer the maximum possible power to the load and ensures that the transfer of power is unidirectional. In this example a pulse generator controls the converter in open-loop with a fixed switching frequency and duty cycle. If the vibration source does not have a constant frequency or it contains harmonics, you can design a more sophisticated closed-loop controller to optimize the transfer of power and improve the efficiency of the energy harvester in different conditions.

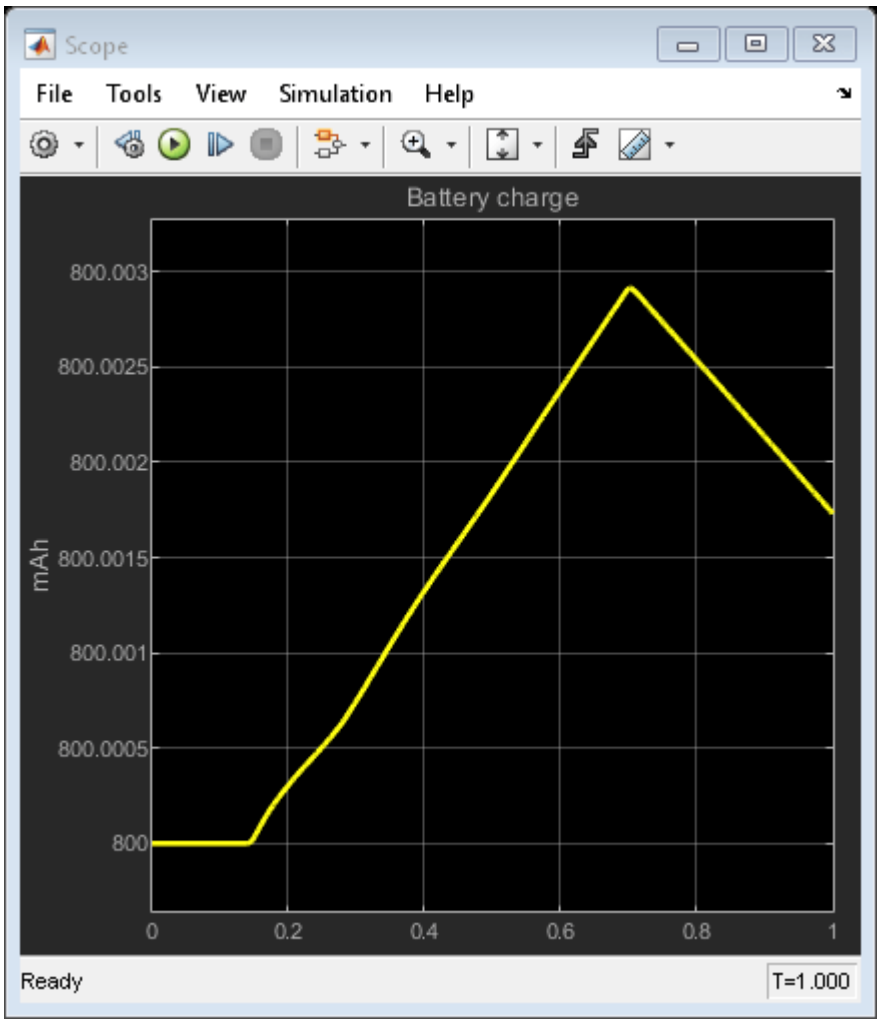
Initially, the energy harvester charges a battery. Then both the energy harvester and the battery power up a constant power load.

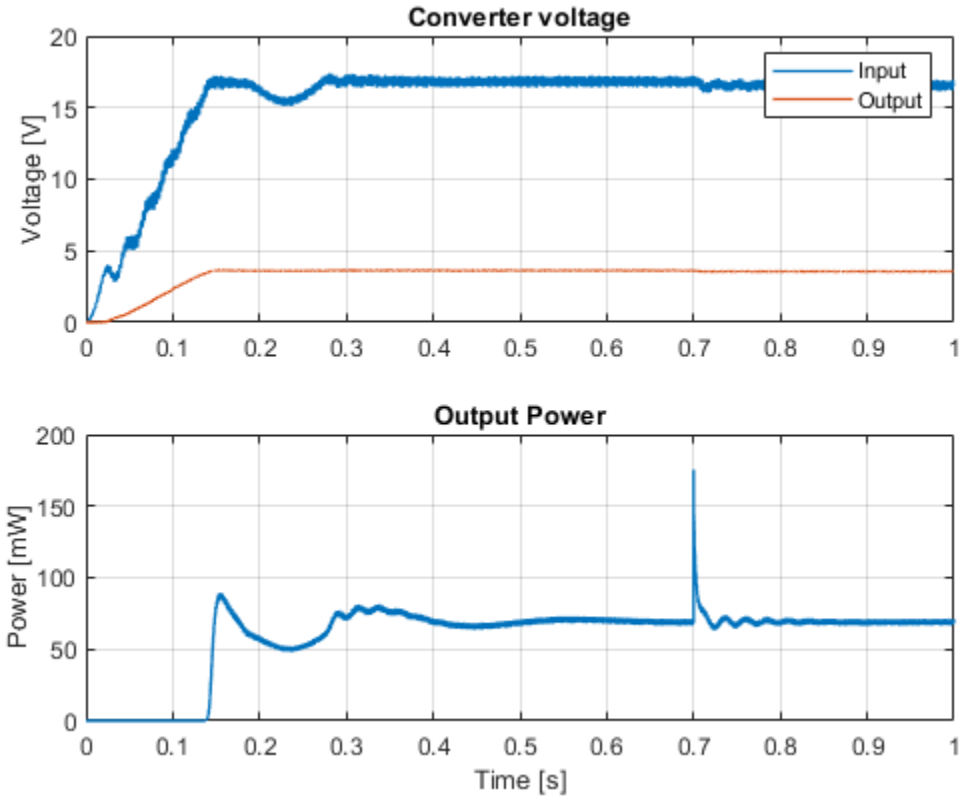


### Plot Results for Sinusoidal Source

The plot below shows the results of a vibration source oscillating at a constant frequency

```
Output power at final time = 68.7332 mW
```





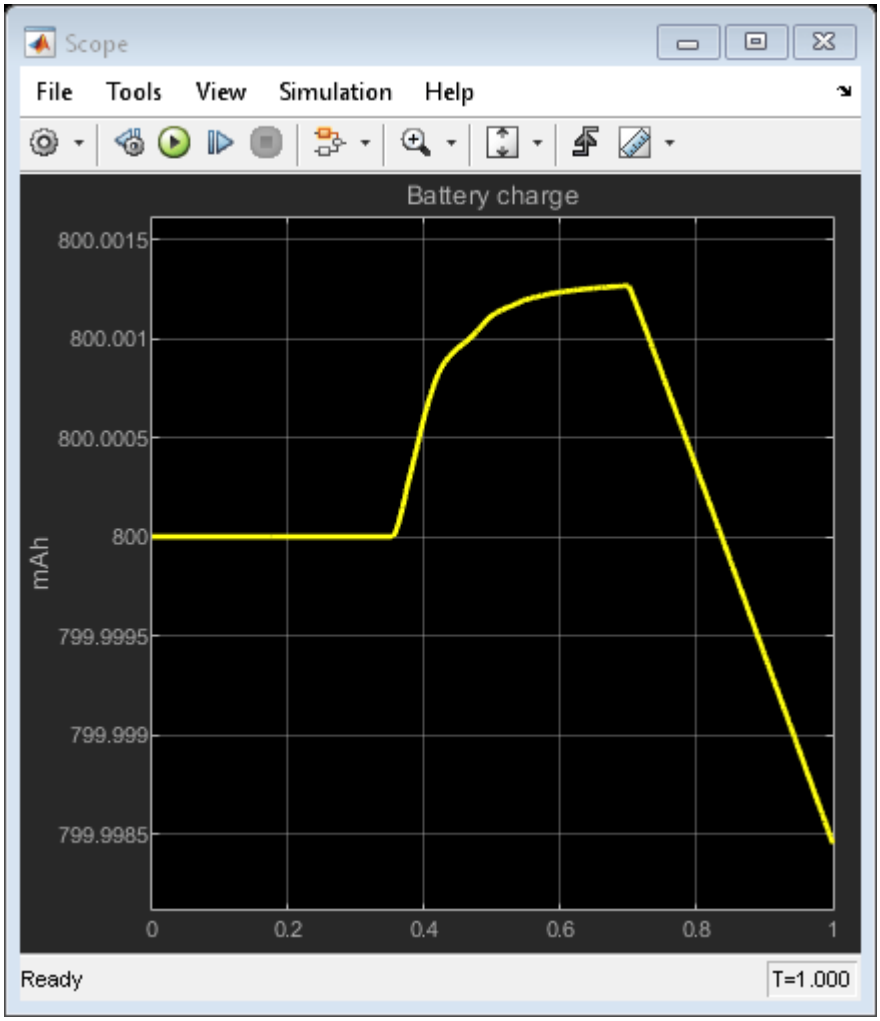

From a mechanical perspective, piezoelectric benders are flexible beams with a natural fundamental frequency of oscillation. When a piezo bender oscillates at this resonant frequency, it generates the maximum power.

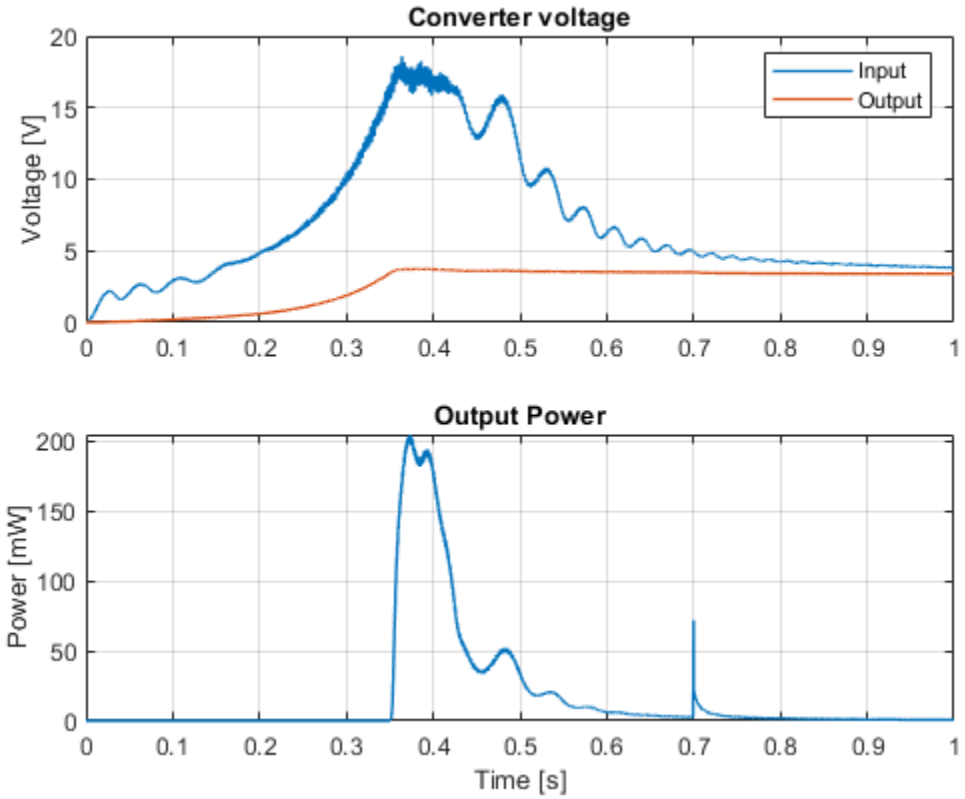
The frequency of the source is close to the resonant frequency, so it's generating almost the maximum possible power for this energy harvester.

### Plot Results for Chirp Source

The plot below shows the results of a vibration source with a linearly-increasing frequency

```
Output power at final time = 0.71343 mW
```


Scope
File
Tools
View
Simulation
Help
Battery charge
800.0015
800.001
800.0005
800
799.9995
799.999
799.9985
mAh
0
0.2
0.4
0.6
0.8
1
Ready
T=1.000



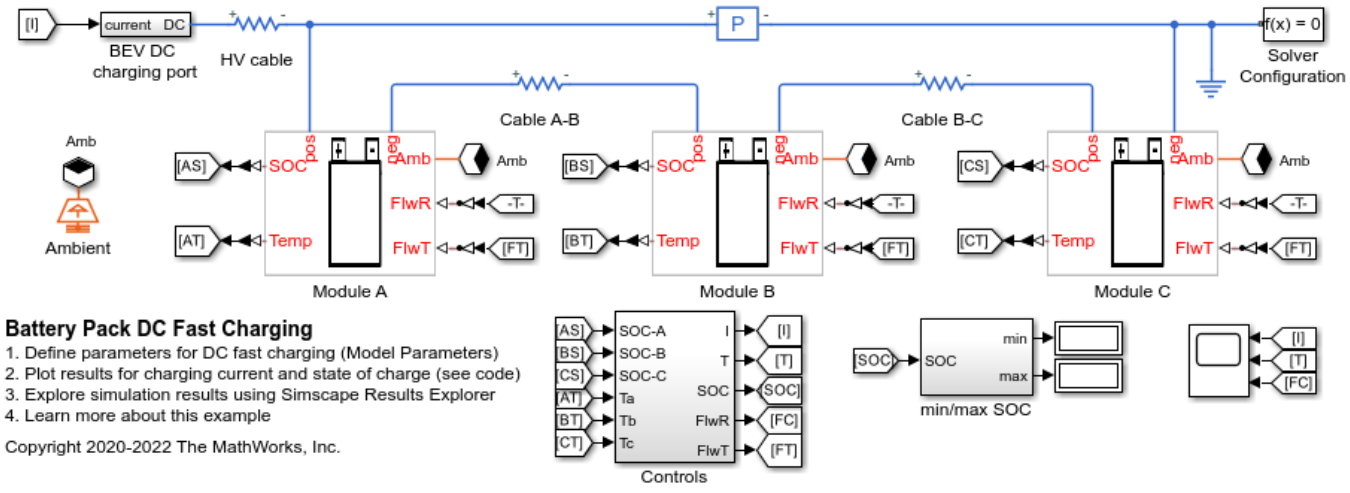

The output power increases as the source is approaching the resonant frequency. Then, it decreases as the source exceeds the resonant frequency.

# Battery Pack DC Fast Charging

This example shows how to model an automotive battery pack for DC fast charging tasks. The battery pack consists of several battery modules, which are combinations of cells in series and parallel. Each battery cell is modeled using the **Battery (Table-Based)** Simscape Electrical block. In this example, the initial temperature and the state of charge are the same for all cells. The cell capacity varies according to the manufacturing tolerances or uncertainities. Three battery modules, two similar and one differing from the other two, are connected in series to simulate a battery pack. The results in this example assume an initial ambient temperature equal to zero degree Celsius. The Controls subsystem defines the logic to determine the battery pack charging time and current.

### Model Overview

The example models a battery pack connected to an auxiliary power load from a chiller, a cooler, or other EV accessories. The Controls subsystem defines how much current the charger can feed into the battery pack based on the measurements of the cell state of charge, temperatures, and the maximum cell C-rate at a given temperature. The Controlled_Current_Source block in the BEV DC charging port subsystem models the battery charger. The logic defined in the Controls subsystem determines the value of the current. A resistor models the HV cable and it is used to connect the charging port to the battery pack. The battery pack comprises three series-connected battery modules, with a total of 130 battery cells.




### Battery Cell Overview

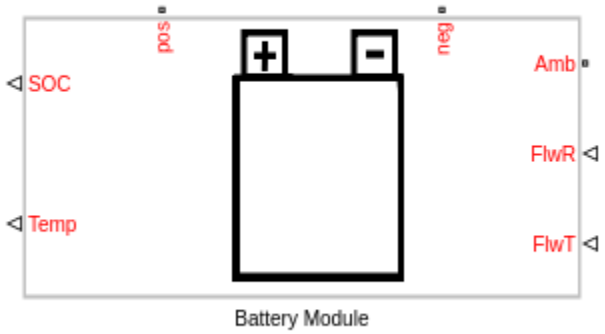
The battery cell is modeled using the equivalent circuit method. The equivalent circuit parameters used for each cell can be found in the reference [1]. To characterize a lithium-ion cell, this example uses a 2-RC model with default parameter values. You can use the **Cell Ahr rating variation** parameter, defined in the **Battery Module** custom component, to introduce cell-to-cell variations in the cell the capacity. No cell capacity fade or charge leakage is considered in this example.

### Battery Module Overview

To use this module to create a unique battery module, first specify the number of series and parallel-connected cells. Then specify the cell type for all individual cells by choosing one of these options for **Choose cell type** parameter of the **Battery Module** block:

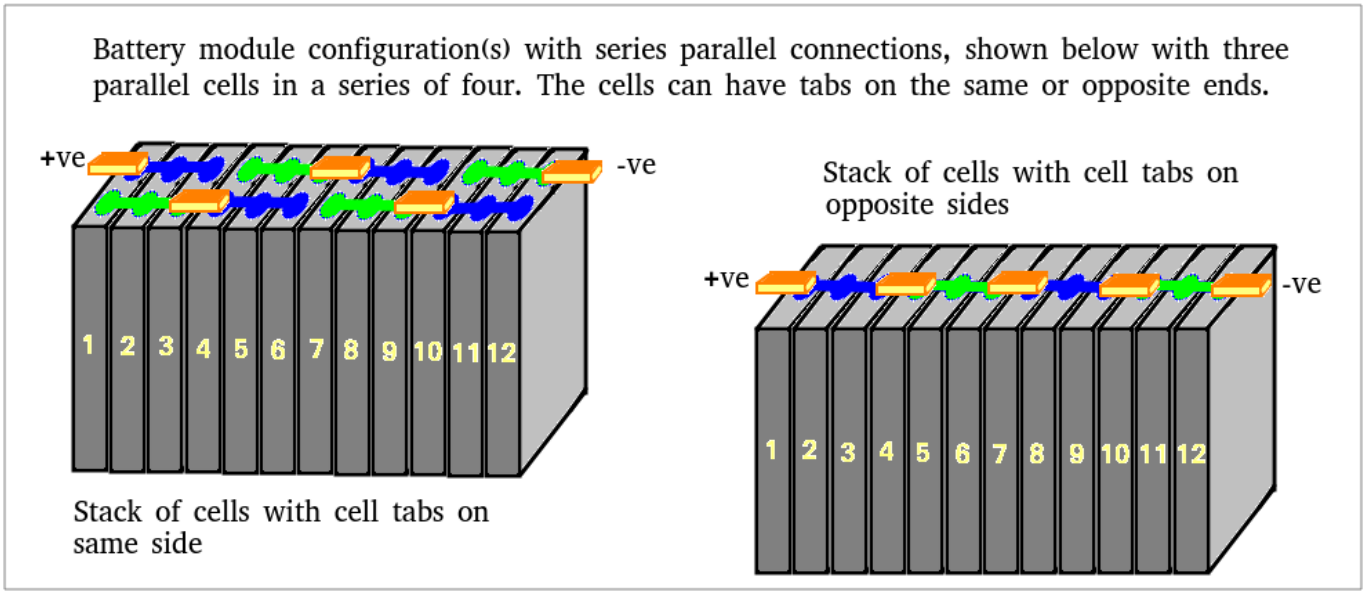
- `Pouch`
- `Can`
- `Compact cylindrical`
- `Regular cylindrical`

This example uses pouch-type cells. Module A and Module B, each consist of 20 series-connected and two parallel-connected cells. Module C consists of 25 series-connected and two parallel-connected cells.

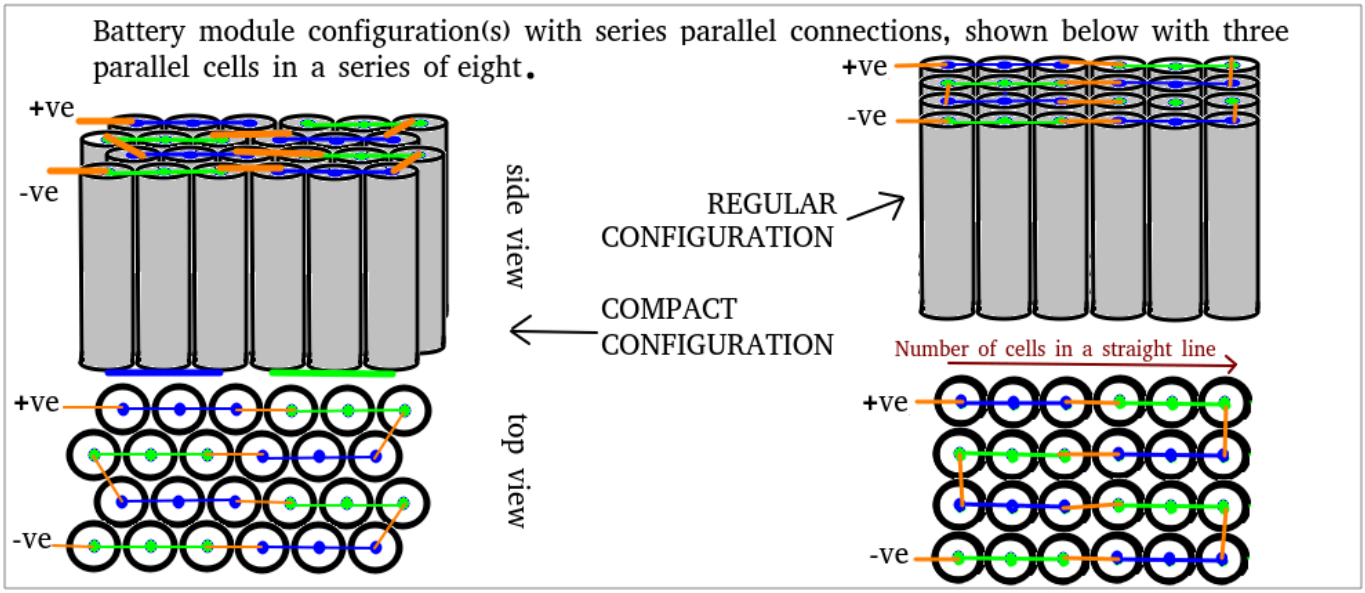



The two output ports, **SOC** and **Temp**, provide information regarding the state of charge and the temperature of each cell in the module. The thermal port, **Amb** is used to define the ambient temperature in the simulation. The electrical ports, **pos** and **neg**, define the electrical positive and negative terminals, respectively. The two input ports, **FlwR** and **FlwT**, define the battery coolant flow rate control and inlet temperature into the module.

The figure below shows examples of battery cells in `Pouch` and `Can` configurations.



The figure below shows examples of battery cells in `Compact cylindrical` and `Regular cylindrical` configurations.




These are the parameters in the battery module:

- **Vector of temperatures, T** — Temperatures at which the cell or module data for temperature-varying properties are tabulated, specified as a vector.
- **Single cell Ahr rating, baseline** — Cell capacity at the temperatures defined in the **Vector of temperatures, T** parameter, specified as a vector.
- **Vector of state of charge values, SOC** — Range of values between 0 and 1 at which the cell electrical parameters are defined, specified as a vector.
- **Vector of coolant flowrates, L** — Coolant mass flow rate values at which a lookup table for cell cooling is defined. This parameter needs to cover multiple points in the flow range of interest. This parameter defines the size of the **Effective rate of coolant heat transfer** parameter and is specified as a vector.
- **No load voltage, V0** — Cell open-circuit potential values at different **Vector of state of charge values, SOC** and **Vector of temperatures, T** points, specified as a matrix.
- **Terminal resistance, R0** — Cell ohmic resistance values at different **Vector of state of charge values, SOC** and **Vector of temperatures, T** points, specified as a matrix.
- **Polarization resistance** — Polarization resistance values at different **Vector of state of charge values, SOC** and **Vector of temperatures, T** points, specified as a matrix.
- **Time constant** — Time constant at different **Vector of state of charge values, SOC** and **Vector of temperatures, T** points, specified as a matrix.
- **Cell thermal mass** — Thermal mass of a single cell, specified as a scalar.
- **Cell thermal conductivity** — Cell through-plane conductivity for pouch and can cells, or the radial conductivity for cylindrical cells, specified as a scalar.
- **Heat transfer coefficient to ambient** — Heat transfer coefficient value, specified as a scalar.

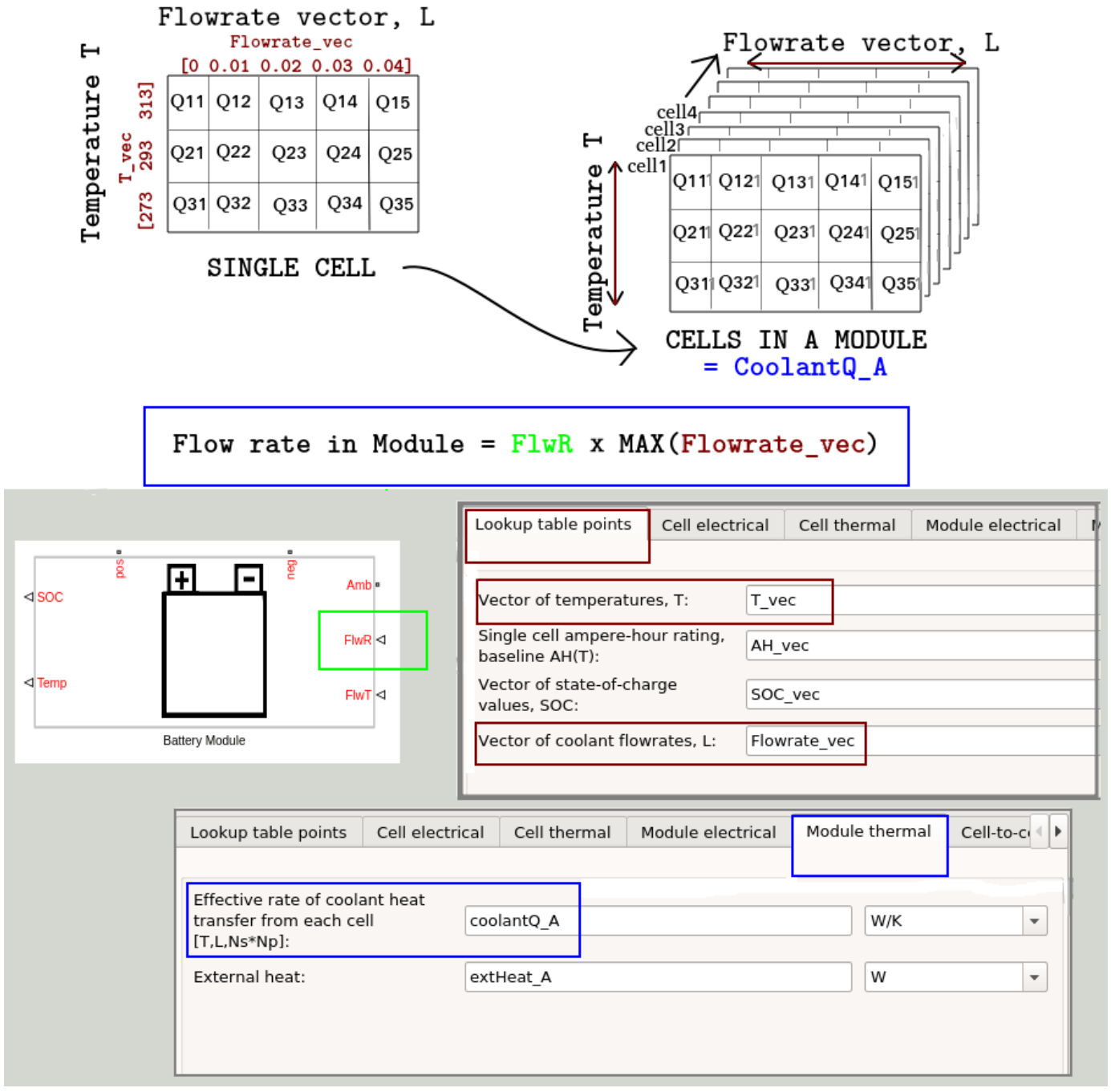
- **Number of series connected cells Ns** — Number of strings in series, specified as an integer.
- **Number of parallel connected cells Np** — Number of parallel-cells in a string, specified as an integer.
- **Choose cell type** — Type of cell, specified as either `Pouch`, `Can`, `Compact cylindrical`, or `Regular cylindrical`.
- **Cell height** — Cell height, specified as a scalar.
- **Cell width** — Cell width for `Pouch` and `Can` cells, specified as a scalar.
- **Cell thickness** — Cell thickness for `Pouch` or `Can` cells, specified as a scalar.
- **Cell diameter** — Cell diameter for `Compact cylindrical` or `Regular cylindrical`, specified as a scalar.
- **Number of cylindrical cells in a straight line** — Number of cylindrical cells arranged in a straight line for packaging, specified as an integer.
- **Accessory total resistance** — Resistance that combines all inline resistance in a module, specified as a scalar. This resistance is the sum of cell tab, busbar, cable and/or weld resistances, specified as a scalar.
- **Cell balancing** — Cell balancing method, specified as either `none` or `passive`. In this example, this parameter is set to `none`.
- **Effective rate of coolant heat transfer from each cell** — Estimate of the thermal resistance (W/K) of heat transfer from battery cells to coolant, specified as a 3-D matrix of scalar values. The 3-D matrix size depends on the **Vector of temperatures, T**, **Vector of coolant flowrates, L** and **NsxNp** parameters. The **NsxNp** parameter is the total number of cells in the module. The battery cooling is represented as a lookup table or 3-D matrix of size [T,L,Ns*Np] and the values are calculated using detailed 3-D methods such as computational fluid dynamics. The values of the matrix depend on the actual hardware design of the cooling system or cold plates in the module. The performance of the cold plate is controlled using input values **FlwR** and **FlwT**.
- **External heat** — External heat input to each cell in a module due to a hot component placed near the module, specified as a vector.
- **Vector of initial cell temperature** — Cell initial temperature, specified as a vector.
- **Vector of initial cell state of charge** — Cell initial state of charge, specified as a vector.
- **Cell Ahr rating variation** — Cell-to-cell variations in cell capacity at all **Vector of temperatures, T** points for each cell, specified as a vector of scalar values. If you set this array to 1, all cell capacity is the same. The array values for a cell are multiplied with the value specified in the **Single cell Ahr rating, baseline** parameter to calculate the actual capacity or the Ahr rating of the cell.

To define the battery coolant flow rate and temperature, specify these inputs:

- **FlwR** — Value between 0 and 1, specified as a scalar. The **FlwR** input value is used to dynamically choose the right value of the flow rate during the simulations. The value of the **FlwR** input defines the actual flow rate in the module. In the **Vector of coolant flowrates L** parameter, **FlwR** equal to 0 means no flow, while **FlwR** equal to 1 means highest flow rate value.
- **FlwT** — Positive or negative value that, when summed to the ambient temperature, equals the coolant inlet temperature. A value of +15 for the **FlwT** input and 273.15 K at the **Amb** port makes the coolant inlet temperature equal to 273.15 + 15 = 288.15K. A value of -15 for the **FlwT** input and 273.15 K at the **Amb** makes the coolant inlet temperature equal to 273.15 - 15 = 258.15 K

### Build the Battery Pack

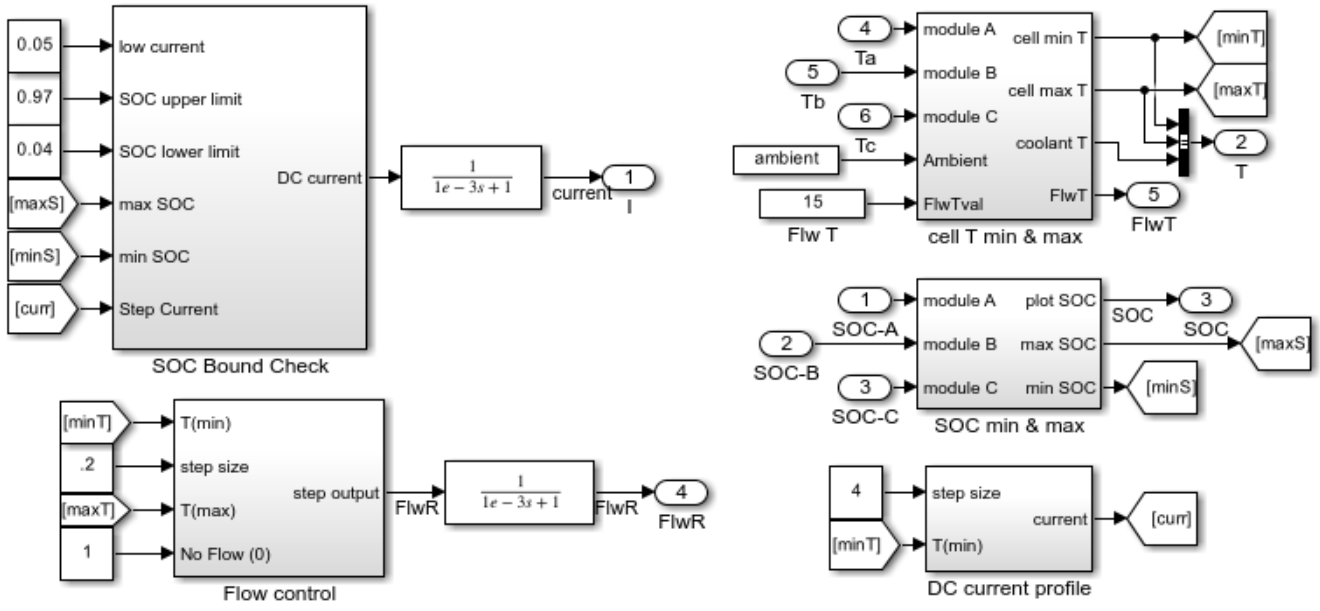
In this example, a battery pack is created by connecting three battery modules in series. A resistance models the cable connection between individual modules. A DC current source models the charger current and it is connected to the battery pack using a cable modeled as a resistance. A power load across the battery terminals models the power consumption due to the chiller or the heater for coolant circuit. The **Effective rate of coolant heat transfer from each cell** parameter models the battery coolant heat transfer rate. In this example, all the cells of a module have the same heat removal rate (W/K), but this value differs from module to module. In absence of detailed data, you can define the heat removal rate value to be the same for all cells at all temperatures and flow rates. The figure below shows the 3-D array lookup table, Qij, where i = 1:size(T) and j = 1:size(L), defines the heat removal rate in W/K for each cell where the temperature value equals *i* and coolant flow rate equals *j*. The values are linearly interpolated between the numbers you specify in the **Vector of coolant flowrates, L** and **Vector of temperatures T** parameters. The cooling system implementation and parameters, like Qij, are defined in the figure below.




### Define the Battery Control

To enable fast charging, a cold battery pack is heated up to allow the passage of larger currents. The DC current profile subsystem estimates the DC current as a function of the minimum cell temperature in the battery pack. The coolant inlet temperature is constant at 288.15 K and defined by setting **FlwT** to a constant input value of 15 and the **Amb** port to 273.15 K. If the temperature

gradient between the cells is greater than five degrees Celsius, the Flow Control subsystem reduces the coolant flow rate.

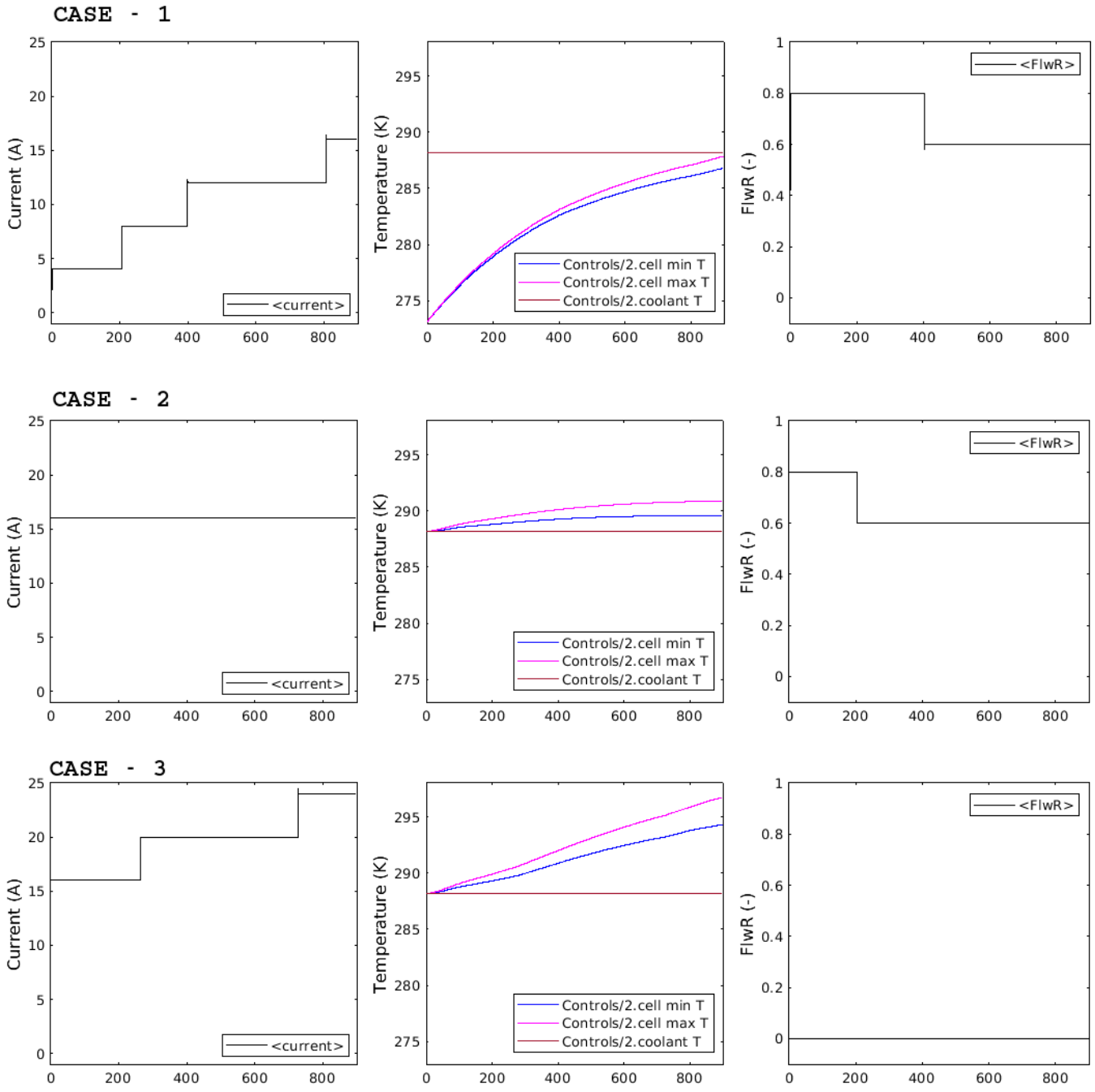



### Simulation Results

This example uses the parameters defined in the `ee_lithium_pack_DCFC_ini.m` file. The cell parameters are the same in all the modules. An external heat source is added to the module C to model the hot components around it. The ambient temperature is set to zero degrees Celsius, the model determines a suitable DC current profile, and the pack charge percentage changes. The initial condition of the pack is equal to 20% of the state of charge. The charge time available is equal to 15 minutes. Three cases are considered:

- Case 1 — The vehicle is parked in the parking area for a long time. The initial cell temperature is the same as the ambient temperature. The battery is heated during charging, with the initial battery state of charge equal to 20%.
- Case 2 — The vehicle is driven and immediately charged. The initial battery cell temperature is equal to 285 K. The battery is heated during charging, with the initial battery state of charge equal to 20%. The `cellInitialTemp` workspace variable, defined in the `ee_lithium_pack_DCFC_ini.m` file, is changed to a value equal to the value of the **Amb** port plus 15.
- Case 3 — The vehicle is driven and immediately charged. The initial battery cell temperature is 285 K. The battery is not heated during charging (no auxillary power consumption), with the initial battery state of charge equal to 20%. The `cellInitialTemp` workspace variable, defined in the `ee_lithium_pack_DCFC_ini.m file`, is changed to a value equal to the value of the **Amb** port plus 15 and `auxLoad` is set to a low value equal to 1e-4. The coolant flow rate **FlwR** is set to zero by turning off the coolant flow inside the Controls/Flow_Control subsystem, setting **NoFlow** to 0.

This figure shows the results from the simulation of the three cases:




In the first case, the battery takes time to heat up. Because the battery temperature is low, the amount of current that it can safely accept from the charger is also low. The battery pack net state of charge rises from 20% to about 42% during the 15 minutes of the charging process.

In the second case, the battery initial temperature is higher, so the control module can put more current into the battery pack. The temperature of the battery further rises due to the heat. This enables the control module to put more charging current into the battery pack. As a result, the battery pack net state of charge rises from 20% to about 66% in 15 minutes of charging time.

The third case represents the best case scenario. The cooling or heating system is not used, so there is no auxillary power load, which leads to the battery getting charged up to 80%. About 60% (80% – 20%) of the net charge is recovered after 15 minutes of charging time.

### References

1 T. Huria, M. Ceraolo, J. Gazzarri,R. Jackey. "High Fidelity Electrical Model with Thermal Dependence for Characterization and Simulation of High Power Lithium Battery Cells", IEEE International Electric Vehicle Conference, March 2012

# SPICE Conversion of a MOSFET Subcircuit and Validation

This example shows how to convert a metal–oxide–semiconductor field-effect transistor (MOSFET) subcircuit into an equivalent Simscape™ component and compare the Spice and Simscape plots for some standard MOSFET characteristics, namely Id versus Vgs, Id versus Vds, Qiss/ Gate charge, Qoss/ Output charge, and Breakdown voltage. The `subcircuit2ssc` function converts all the subcircuit components inside a SPICE netlist file into one or more equivalent Simscape files.

### Open the MOSFET Subcircuit

In this example, you will convert this MOSFET subcircuit to a Simscape component. To open the SPICE netlist, in the MATLAB command window, enter edit IAUC100N04S6L014.cir.

```
* ---------------------------------------------------------------------------
*
* DISCLAIMER
*
* INFINEON'S MODEL TERMS OF USE
*
* BY DOWNLOADING AND/OR USING THIS INFINEON MODEL ("MODEL"), THE USER
* (INCLUDING YOU) AGREES TO BE BOUND BY THE TERMS OF USE HERE STATED. IF USER
* DOES NOT AGREE TO ALL TERMS OF USE HERE STATED, USER SHALL NOT DOWNLOAD,
* USE OR COPY THE MODEL BUT IMMEDIATELY DELETE IT (TO THE EXTENT THAT IT
* WAS DOWNLOADED ALREADY).
*
* 1. SCOPE OF USE
* 1.1 Any use of this Model provided by Infineon Technologies AG is subject
*     to these Terms of Use.
* 1.2 This Model, provided by Infineon, does not fully represent all of the
*     specifications and operating characteristics of the product to which
*     this Model relates.
* 1.3 This Model only describes the characteristics of a typical product.
*     In all cases, the current data sheet information for a given product
*     is the final design guideline and the only actual performance
*     specification. Although this Model can be a useful tool to evaluate
*     the product performance, it cannot simulate the exact product performance
*     under all conditions and it is also not intended to replace
*     bread-boarding for final verification.
*
* 2. IMPORTANT NOTICE
* 2.1 Infineon Technologies AG ("Infineon") is not and cannot be aware of the
*     specific application of the Infineon's Model by User. However, Model may
*     from time to time be used by User in potentially harmful and/or life-
*     endangering applications such as traffic, logistic, medical, nuclear
*     or military applications or in other applications where failure of the
*     Model may predictably cause damage to persons' life or health or to
*     property (hereinafter "Critical Applications").
* 2.2 User acknowledges that Infineon has not specifically designed or
*     qualified the Model for Critical Applications that the Model may contain
*     errors and bugs and that User is required to qualify the Model for
*     Critical Applications pursuant to the applicable local quality, safety
*     and legal requirements before permitting or giving access to any such use.
*
* 3. CONFIDENTIAL INFORMATION
*     User shall treat ideas, concepts and information incorporated in the
```

```
*     Model, the documentation and the content of this Terms of Use (together
*     hereinafter "Confidential Information") confidential, not disclose it to
*     any third party unless otherwise agreed in writing between User and
*     Infineon, not use it for any other purposes than using the Model for
*     simulation and testing purposes only.
*
* 4. WARRANTY
* 4.1 User acknowledges that the Model is provided by Infineon under this Terms
*     of Use is provided free of charge and "AS IS" without any warranty or
*     liability of any kind and Infineon hereby expressly disclaims any
*     warranties or representations, whether express, implied, statutory or
*     otherwise, including but not limited to warranties of workmanship,
*     merchantability, fitness for a particular purpose, defects in the
*     Model, or non-infringement of third parties intellectual property rights.
* 4.2 Infineon reserves the right to make corrections, deletions, modifications,
*     enhancements, improvements and other changes to the Model at any time
*     or to move or discontinue any Model without notice.
*
* 5. LIABILITY
* 5.1 Nothing in this Terms of Use shall limit or exclude Infineon's liability
*     under mandatory liability laws, for injuries to life, body or health,
*     for fraudulent concealment of defects in the software, or in cases
*     of Infineon's intentional misconduct or gross negligence.
* 5.2 Without prejudice to Sections 5.1, in cases of Infineon's slight
*     negligent breach of obligations that restrict essential rights or duties
*     arising from the nature of this Terms of Use in a way that there is a
*     risk of non-achievement of the purpose of this Terms of Use or of an
*     obligation whose observance User regularly may trust in and whereas
*     compliance with only makes proper execution of this Terms of Use
*     possible, Infineon's liability shall be limited to the typically,
*     foreseeable damage.
* 5.3 Without prejudice to Sections 8.1 and 8.2, Infineon's liability under
*     this Agreement shall be excluded in all other cases.
*
* 6. EXPORT REGULATIONS
*     The User shall comply with all applicable national and international
*     laws and regulations, in particular the applicable export control
*     regulations and sanction programs. The User also agrees not to
*     export, re-export or transfer any software or technology developed
*     with or using information, software or technology offered by Infineon,
*     in violation of any applicable laws or regulations of the competent
*     authorities. Further, the User shall neither use any products,
*     information, software and technology offered by Infineon in or in
*     connection with nuclear technology or weapons of mass destruction
*     (nuclear, biological or chemical) and carriers thereof nor supply
*     military consignees.
*
* 7. TERMINATION OF USE PERMIT
*     If the User violates these Terms of Use, such User's permit to use
*     this Model terminates automatically. In addition, Infineon may
*     terminate the User's permit to use this Model at its discretion and
*     at any time regardless of any violation of these Terms of Use. In
*     any of the foregoing events, the User is obliged to immediately destroy
*     any content that has been downloaded or printed from Infineon's website.
*
* 8. MISCELLANEOUS
* 8.1 These Terms of Use are subject to the laws of the Federal Republic
*     of Germany with the exception of the United Nations on Purchase
```

```
*     Contracts on the International Sale of Goods dated April 11, 1980 (CISG).
*     The exclusive place of jurisdiction is Munich, Germany.
* 8.2 Should any provision in these Terms of Use be or become invalid, the
*     validity of all other provisions or agreements shall remain unaffected
*     thereby.
*
* --------------------------------------------------------------------------
*
* Title:    INFINEON Power Transistors Simulation Models for PSpice
* Description: n-channel Transistors
*                 OptiMOS5 40V
* Authors:  Dr. Elmar Gondro, Tel: +49/89/234-29391
*           Günther Moyses,   Tel: +49/89/234-24811
*           Sebastian Koch,   Tel: +49/89/234-89858
* Email:    Elmar.Gondro@Infineon.com
*           Guenther.Moyses@Infineon.com
*           Sebastian.Koch@Infineon.com
* Support:  Simulate@Infineon.com
*
.SUBCKT IAUC100N04S6L014 drain gate source Tj Tcase PARAMS: dVth=0 dRdson=0 dgfs=0 dC=0 dZth=0 Ls

.PARAM Rs=222u      Rg=2.2       Rd=5u        Rmet=45u
.PARAM Inn=50       Unn=10       Rmax=1.40m   gmin=69
.PARAM act=4.672    Rsp=2.4

X_1  d1 g s sp Tj A6_40_d_var PARAMS: a={act} Rsp={Rsp} dVth={dVth} dRdson={dRdson} dgfs={dgfs} I
+                                     Rmax={Rmax} gmin={gmin} Rs={Rs} Rp={Rd} dC={dC} Rmet={Rmet}
R_g    g1     g    {Rg}
L_g    gate   g1   {Lg*if(dgfs==99,0,1)}
G_s    s1     s    VALUE={V(s1,s)/(Rs*(1+(limit(V(Tj),-200,999)-25)*4m)-Rmet)}
R_sa   s1     s    1Meg
L_s    source s1   {Ls*if(dgfs==99,0,1)}
R_da   d1     d2   {Rd}
L_d    drain  d2   {Ld*if(dgfs==99,0,1)}
R_daux drain  d2    10
R_gaux gate   g1    10
R_saux source s1    10

R_th1  Tj      t1              {5.98m+limit(dZth,0,1)*2.22m}
R_th2  t1      t2              {73.91m+limit(dZth,0,1)*27.36m}
R_th3  t2      t3              {258.69m+limit(dZth,0,1)*10.72m}
R_th4  t3      t4              {305.94m+limit(dZth,0,1)*195.71m}
R_th5  t4      Tcase           {377.8m+limit(dZth,0,1)*241.67m}
C_th1  Tj      0               40.653u
C_th2  t1      0               180.521u
C_th3  t2      0               1.089m
C_th4  t3      0               823.295u
C_th5  t4      0               25.88m
C_th6  Tcase   0               30m

.ENDS


.SUBCKT A6_40_d_var dd g s0 sp Tj PARAMS: a=1 Rsp=1 dVth=0 dRdson=0 dgfs=0 Inn=1
+ Unn=1 Rmax=0 gmin=1 Rs=1 Rp=1 dC=0 Rmet=1u

.PARAM  Fm=0.05    Fn=1        al=0.5
.PARAM  c=0.95     Vth0=2.479  auth=3.056m
```

```
.PARAM  UT=15.11m  ab=18.74m   lB=-23       UB=41.93

.PARAM  b0=244.4   p0=8.687    p1=-30.58m   p2=45.79u

.PARAM  Rd=3.508m  nmu=2.72    Tref=298     T0=273      lnIsj=-26.42
.PARAM  ndi=1.039  nisj=1.028  Rdi=4.29m    nmu2=0.0
.PARAM  td=20n     ta=2n
.PARAM  Rf=0.5949  nmu3=1.774

.PARAM  kbq=86.2u

* Cgs
.PARAM  f3=606.2p

* Cgfp
.PARAM  f3a=31.78p

* Cds
.PARAM q81=145.0p
.PARAM qs1=65.62p  qs2=345.4p  qs3=-87.45m
.PARAM qs4=85.58p  qs5=-16.63m
.PARAM x0=308.2m   x1=27.68    dx={x1-x0}
.PARAM f2r=226.4m

* Cgd
.PARAM  ps0=14.83p ps1=9.092p ps2=-2.097 ps3=84.63p ps4=-160.7m ps5=6.584p ps6=8.636p
.PARAM  f5=35.38p
.PARAM  x2=6.361 x3=24.84      dx2={x3-x2}

* corner parameters
.PARAM  dVthmax=0.3   dCmax=0.3

.PARAM  Vth={Vth0+dVthmax*dVth}
.PARAM  q0={b0*((T0/Tref)**nmu3)*a}
.PARAM  q1={(Unn-Inn*Rs-Vth0)*q0}
.PARAM  q2={(Fm*sqrt(0.4)-c)*Inn*q0}
.PARAM  Rlim={(q1+2*q2*Rmax-sqrt(max(q1**2+4*q2,0)))/(2*q2)}
.PARAM  dRd={Rd/a+if(dVth==0,limit(dRdson,0,1)*max(Rlim-Rd/a-Rs-Rp,0),0)}
.PARAM  bm={c/((1/gmin-Rs)**2*Inn*a*(T0/Tref)**nmu3)}
.PARAM  bet={b0+(b0-bm)*if(dRdson==0,if(dVth==0,limit(dgfs,-1,0),0),0)}
.PARAM  dC1={1+dCmax*limit(dC,0,1)}
* .PARAM  dC2={1+1.5*dCmax*limit(dC,0,1)}

.PARAM  Cgs0={f3*a*dC1}
.PARAM  Cgs1={f3a*a*dC1}
.PARAM  dRdi={Rdi/a}

.PARAM  Cox1={(ps1*a+ps0*sqrt(a))*dC1}
.PARAM  Cox2={ps3*a*dC1}
.PARAM  Cox3={(ps5*a+ps6)*dC1}
.PARAM  Cox4={(f5*a+(ps5*a+ps6))*dC1}

.PARAM  Cds0={qs1*a*dC1}
.PARAM  Cds1={qs2*(1+f2r/sqrt(a))*a*dC1}
.PARAM  Cds2={qs4*a*dC1}
.PARAM  Cds3={(q81+qs1)*a*dC1}

* .FUNC  VBR(Usps)    {max(UB+min(Usps,dUmax)*s1+max(Usps-dUmax,0)*s2,Umin)}
```

```
.FUNC  VBR(Usps)    {UB}

.FUNC  I0(Uee,p,pp,z1)  {if(Uee>pp,(Uee-c*z1)*z1,p*(pp-p)/c*exp((Uee-pp)/p))}
.FUNC  Ig(Uds,T,p,Uee)  {bet*(T0/T)**nmu3*I0(Uee,p,min(2*p,p+c*Uds),min(Uds,Uee/(2*c)))}
.FUNC J(d,g,T,da,s,Usps)
+ {a*(s*(Ig(da,T,(p0+(p1+p2*T)*T)*kbq*T,g-Vth+auth*(T-Tref)+Fm*da**Fn+1*limit(-d,0,1))+
+ exp(min(lB+(d-VBR(Usps)-ab*(T-Tref))/UT,25))))}

.FUNC  Idiode(Usd,Tj,Iss) {max(exp(min(log(Iss)+Usd/(ndi*kbq*Tj),7))-Iss,0)}
.FUNC  Idiod(Usd,Tj)      {a*Idiode(Usd,Tj,exp(min(lnIsj+(Tj/Tref-1)*1.12/(ndi*kbq*Tj),7))*(Tj/T

.FUNC  Pr(Vss0,Vssp)  {Vss0*Vss0/max(Rmet,1u)+Vssp*Vssp/Rsp}

* .FUNC J1(d,g,T,da,s,Usps) {a*(s*(exp(min(lB+(d-VBR(Usps)-ab*(T-Tref))/UT,25))))}

.FUNC QCds(x) {Cds3*min(x,x1)+Cds0*max(x-x1,0)+(Cds3-Cds0)*((limit(x,x0,x1)-x0)**3/(dx*dx)*((limi
.FUNC QCdg(x) {Cox4*min(x,x3)+Cox3*max(x-x3,0)+(Cox4-Cox3)*((limit(x,x2,x3)-x2)**3/(dx2*dx2)*((li


E_Edg1     d   ox    VALUE {if(V(d,g)>0,V(d,g)-(exp(ps2*max(V(d,g),0))-1)/ps2,0)}
C_Cdg1    ox   g     {Cox1}
E_Edg2     d   ox1   VALUE {if(V(d,g)>0,V(d,g)-(exp(ps4*max(V(d,g),0))-1)/ps4,0)}
C_Cdg2   ox1   g     {Cox2}
E_Edg3     d   ox2   VALUE {V(d,g)-QCdg(V(d,g))/Cox4}
C_Cdg3   ox2   g     {Cox4}

E_Eds      d   edep  VALUE {V(d,s)-QCds(V(d,s))/Cds3}
C_Cds  edep    s    {Cds3/2}

E_Eds1     d   edep1 VALUE {V(d,sp)-QCds(V(d,sp))/Cds3}
C_Cds1 edep1   sp    {Cds3/2}
E_Eds2     d   edep2 VALUE {if(V(d,sp)>0,V(d,sp)-(exp(qs5*max(V(d,sp),0))-1)/qs5,0)}
C_Cds2 edep2   sp    {Cds2}
E_Eds3     d   edep3 VALUE {if(V(d,sp)>0,V(d,sp)-(exp(qs3*max(V(d,sp),0))-1)/qs3,0)}
C_Cds3 edep3   sp    {Cds1}

C_Cgs      g    s      {Cgs0}
C_Cgs1     g    sp     {Cgs1}

R_fp       s    sp     {Rsp}

G_chan   d5a    s  VALUE={J(V(d5a,s),V(g,s),T0+limit(V(Tj),-200,300),(sqrt(1+4*al*abs(V(d5a,s)))-1
R_d06    d5a   d5  1u
V_sm     d     d5  0
G_RMos   d1    d  VALUE={V(d1,d)/(Rf*dRd+(1-Rf)*dRd*((limit(V(Tj),-200,999)+T0)/Tref)**nmu)}
V_sense  dd    d1  0
G_diode   s    d3  VALUE={Idiod(V(s,d3),T0+limit(V(Tj),-200,499))}
G_Rdio   d2    d1  VALUE={V(d2,d1)/(dRdi*((limit(V(Tj),-200,999)+T0)/Tref)**nmu2)}
V_sense2 d2    d3  0

* includes RevRec, but has no Tj dependence...
* D_body s d3 dbody
* .model dbody D ( BV={UB*10} CJO={Cds0/100} TT={ta} IS={a*exp(lnIsj)} m=0.3 RS={dRdi*1m} n={ndi

R_1        g    s  1G
R_d01      d    s  500Meg
R_d02     d2    s  500Meg
R_d03     d1    d  1k
```

```
R_ssp     g    sp  100Meg

R_met      s    s0 {Rmet}

G_th      0   Tj  VALUE={(I(V_sense)-I(V_sense2))*V(d1,d)+I(V_sm)*V(d,s)+I(V_sense2)*V(d1,s)+Pr(V

.ENDS
```

Run this command to convert the SPICE subcircuit of the `IAUC100N04S6L014.cir` model to a Simscape component and place the generated files in the newly created directory called `+myMOSFET`:

```
Netlist converted. Review Simscape component files and make
manual edits for any unsupported items before building the
Simscape library located at:
+myMOSFET.
```

### Open the Converted MOSFET Model

The `iauc100n04s6l014.ssc` file stored in the `+myMOSFET` directory is the converted Simscape component obtained by running the `subcircuit2ssc` function on the modified SPICE netlist. The `subcircuit2ssc` function also converted all the functions implemented in the SPICE subcircuit. To edit the generated simscape component, in the MATLAB command window, enter edit +myMOSFET/iauc100n04s6l014.ssc.

```
component iauc100n04s6l014
% iauc100n04s6l014
% Component automatically generated from a SPICE netlist for subcircuit IAUC100N04S6L014.
%   MATLAB version: 9.13.
%   spice2ssc version: unknown.
%   Simscape code generated on: 31-Aug-2022 13:52:35


    nodes
        drain = foundation.electrical.electrical; % drain
        gate = foundation.electrical.electrical; % gate
        source = foundation.electrical.electrical; % source
        tj = foundation.electrical.electrical; % tj
        tcase = foundation.electrical.electrical; % tcase
    end

    nodes(Access=protected, ExternalAccess=none)
        d1 = foundation.electrical.electrical;
        g = foundation.electrical.electrical;
        s = foundation.electrical.electrical;
        sp = foundation.electrical.electrical;
        g1 = foundation.electrical.electrical;
        s1 = foundation.electrical.electrical;
        d2 = foundation.electrical.electrical;
        t1 = foundation.electrical.electrical;
        t2 = foundation.electrical.electrical;
        t3 = foundation.electrical.electrical;
        t4 = foundation.electrical.electrical;
    end

    annotations
```

```
    UILayout = [
        UIGroup('Main', ...
                dvth, drdson, dgfs, dc, dzth, ls, ld, lg)
        UIGroup('Parasitic elements', ...
                specifyParasiticValues, capacitorSeriesResistance, inductorParallelConductanc
        UIGroup('Smoothing functions', ...
                specifySmoothValues, aWarning, crossZero, dropPowerFlag, dropPowerHypEpsilon
               ]
end

parameters
    dvth = {0, '1'};
    drdson = {0, '1'};
    dgfs = {0, '1'};
    dc = {0, '1'};
    dzth = {0, '1'};
    ls = {5e-11, '1'};
    ld = {1e-09, '1'};
    lg = {3e-09, '1'};
end

parameters
    specifyParasiticValues = ee.enum.include.no;    % Specify parasitic values
end

parameters(ExternalAccess=none)
    capacitorSeriesResistance = {0, 'Ohm'};    % Capacitor parasitic series resistance
    inductorParallelConductance = {0, '1/Ohm'};    % Inductor parasitic parallel conductance
end

if specifyParasiticValues == ee.enum.include.yes
    annotations
        [capacitorSeriesResistance, inductorParallelConductance] : ExternalAccess=modify;
    end
end

parameters
    specifySmoothValues = ee.enum.include.no;    % Specify function smooth parameters
end

parameters(ExternalAccess=none)
    aWarning = ee.enum.include.no;    % Include warning
    crossZero = ee.enum.include.no;    % Zero crossing for abs(x) and sign(x)
    dropPowerFlag = ee.enum.function.powerFlag.hyp;    % sqrt(x) and x**y protection method
    dropPowerHypEpsilon = {0, '1'};    % Epsilon for protection hyp function in modified sqr
    expXh = {Inf, '1'};    % Upper boundary of linear extrapolation for exp(x), typical x=80
    expXl = {-Inf, '1'};    % Lower boundary of linear extrapolation for exp(x), typical x=-8
    logX0 = {0, '1'};    % Boundary value of linear extrapolation for log(x) and log10(x), ty
    smoothEpsilon = {0, '1'};    % Epsilon determines the start of the transition and the abs
    smoothN = {1, '1'};    % The order of the smoothing function in limit(x, a, b), max(x, y
end

if specifySmoothValues == ee.enum.include.yes
    annotations
        [aWarning, crossZero, dropPowerFlag, expXh, expXl, logX0, smoothEpsilon, smoothN] : E
    end
    if dropPowerFlag == ee.enum.function.powerFlag.hyp
        annotations
```

```
            dropPowerHypEpsilon : ExternalAccess=modify;
        end
    end
end

parameters(Access=private,ExternalAccess=none)
    rs = {0.000222, '1'};
    rg = {2.2, '1'};
    rd = {5e-06, '1'};
    rmet = {4.5e-05, '1'};
    inn = {50, '1'};
    unn = {10, '1'};
    rmax = {0.0014, '1'};
    gmin = {69, '1'};
    act = {4.672, '1'};
    rsp = {2.4, '1'};
end

variables
    G_S = {value={0,'A'},priority=priority.none};
end

components(ExternalAccess=observe)
    X_1 = myMOSFET.a6_40_d_var(a=act,rsp=rsp,dvth=dvth,drdson=drdson,dgfs=dgfs,inn=inn,unn=un
        gmin=gmin,rs=rs,rp=rd,dc=dc,rmet=rmet,capacitorSeriesResistance=capacitorSeriesResist
    R_G = foundation.electrical.elements.resistor(R={rg,'Ohm'});
    L_G = foundation.electrical.elements.inductor(l={lg*(if dgfs==99, 0 else 1 end),'H'},r={0
        i_L.priority=priority.none);
    R_SA = foundation.electrical.elements.resistor(R={(1*1000000),'Ohm'});
    L_S = foundation.electrical.elements.inductor(l={ls*(if dgfs==99, 0 else 1 end),'H'},r={0
        i_L.priority=priority.none);
    R_DA = foundation.electrical.elements.resistor(R={rd,'Ohm'});
    L_D = foundation.electrical.elements.inductor(l={ld*(if dgfs==99, 0 else 1 end),'H'},r={0
        i_L.priority=priority.none);
    R_DAUX = foundation.electrical.elements.resistor(R={10,'Ohm'});
    R_GAUX = foundation.electrical.elements.resistor(R={10,'Ohm'});
    R_SAUX = foundation.electrical.elements.resistor(R={10,'Ohm'});
    R_TH1 = foundation.electrical.elements.resistor(R={(5.98*0.001)+simscape.function.limitm
        smoothN, smoothEpsilon)*(2.22*0.001),'Ohm'});
    R_TH2 = foundation.electrical.elements.resistor(R={(73.91*0.001)+simscape.function.limitm
        smoothN, smoothEpsilon)*(27.36*0.001),'Ohm'});
    R_TH3 = foundation.electrical.elements.resistor(R={(258.69*0.001)+simscape.function.limit
        smoothN, smoothEpsilon)*(10.72*0.001),'Ohm'});
    R_TH4 = foundation.electrical.elements.resistor(R={(305.94*0.001)+simscape.function.limit
        smoothN, smoothEpsilon)*(195.71*0.001),'Ohm'});
    R_TH5 = foundation.electrical.elements.resistor(R={(377.8*0.001)+simscape.function.limitm
        smoothN, smoothEpsilon)*(241.67*0.001),'Ohm'});
    C_TH1 = foundation.electrical.elements.capacitor(c={(40.653*1e-06),'F'},r=capacitorSeries
        vc.priority=priority.none);
    C_TH2 = foundation.electrical.elements.capacitor(c={(180.521*1e-06),'F'},r=capacitorSerie
        g={0,'1/Ohm'},vc.priority=priority.none);
    C_TH3 = foundation.electrical.elements.capacitor(c={(1.089*0.001),'F'},r=capacitorSeriesR
        vc.priority=priority.none);
    C_TH4 = foundation.electrical.elements.capacitor(c={(823.295*1e-06),'F'},r=capacitorSerie
        g={0,'1/Ohm'},vc.priority=priority.none);
    C_TH5 = foundation.electrical.elements.capacitor(c={(25.88*0.001),'F'},r=capacitorSeriesR
        vc.priority=priority.none);
    C_TH6 = foundation.electrical.elements.capacitor(c={(30*0.001),'F'},r=capacitorSeriesResi
```

```
        vc.priority=priority.none);
end

connections
    connect(X_1.dd,d1);
    connect(X_1.g,g);
    connect(X_1.s0,s);
    connect(X_1.sp,sp);
    connect(X_1.tj,tj);
    connect(R_G.p,g1);
    connect(R_G.n,g);
    connect(L_G.p,gate);
    connect(L_G.n,g1);
    connect(R_SA.p,s1);
    connect(R_SA.n,s);
    connect(L_S.p,source);
    connect(L_S.n,s1);
    connect(R_DA.p,d1);
    connect(R_DA.n,d2);
    connect(L_D.p,drain);
    connect(L_D.n,d2);
    connect(R_DAUX.p,drain);
    connect(R_DAUX.n,d2);
    connect(R_GAUX.p,gate);
    connect(R_GAUX.n,g1);
    connect(R_SAUX.p,source);
    connect(R_SAUX.n,s1);
    connect(R_TH1.p,tj);
    connect(R_TH1.n,t1);
    connect(R_TH2.p,t1);
    connect(R_TH2.n,t2);
    connect(R_TH3.p,t2);
    connect(R_TH3.n,t3);
    connect(R_TH4.p,t3);
    connect(R_TH4.n,t4);
    connect(R_TH5.p,t4);
    connect(R_TH5.n,tcase);
    connect(C_TH1.p,tj);
    connect(C_TH1.n,*);
    connect(C_TH2.p,t1);
    connect(C_TH2.n,*);
    connect(C_TH3.p,t2);
    connect(C_TH3.n,*);
    connect(C_TH4.p,t3);
    connect(C_TH4.n,*);
    connect(C_TH5.p,t4);
    connect(C_TH5.n,*);
    connect(C_TH6.p,tcase);
    connect(C_TH6.n,*);
end

branches
    G_S: s1.i -> s.i;
end

equations
    value(G_S,'A') == value(s1.v-s.v,'V')/(rs*(1+(simscape.function.limitm(value(tj.v,'V'),
        smoothEpsilon)-25)*(4*0.001))-rmet);
```

```
    end

end
```

The `subcircuit2ssc` function fully imports this particular subcircuit, with no need for any further manual steps. However, in this example, to review the simulation results more easily, you can access the gate, drain, and source currents. To access the drain, gate, and source currents in simlog, add to the % generated simscape component the drainInternal, gateInternal, and % sourceInternal internal nodes, and the idrain, igate, and isource % corresponding through variables. Then, save the changes into a new file named `iauc100n04s6l014_updated.ssc` and pass the file as an argument to the ee_convertedMosfetValidation MATLAB function.

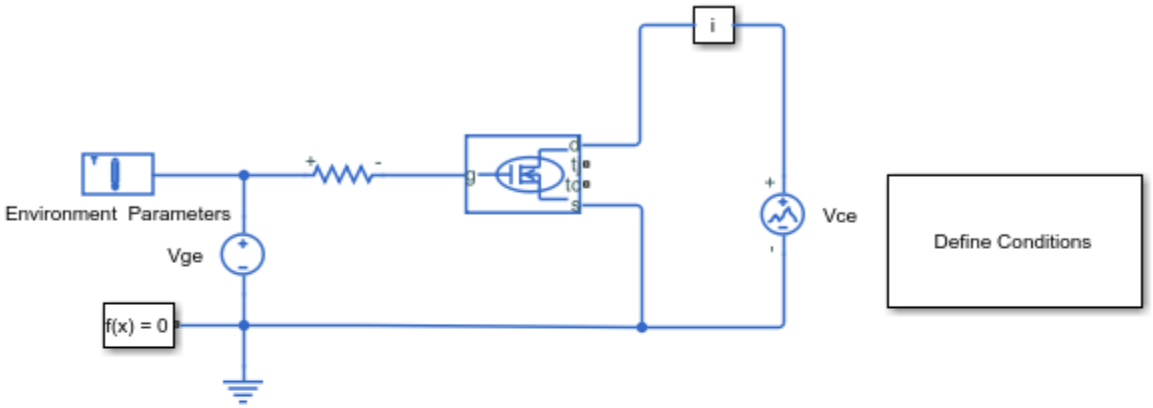
```
component iauc100n04s6l014_updated
% iauc100n04s6l014_updated
% Component automatically generated from a SPICE netlist for subcircuit iauc100n04s6l014 and adde
%   MATLAB version: 9.9.
%   Simscape Electrical version: 7.4.
%   Simscape code generated on: 11-May-2020 05:10:47


    nodes
        drain = foundation.electrical.electrical; % drain
        gate = foundation.electrical.electrical; % gate
        source = foundation.electrical.electrical; % source
        tj = foundation.electrical.electrical; % tj
        tcase = foundation.electrical.electrical; % tcase
    end

    nodes(Access=protected, ExternalAccess=none)
        drainInternal = foundation.electrical.electrical; % internal drain
        gateInternal = foundation.electrical.electrical; % internal gate
        sourceInternal = foundation.electrical.electrical; % internal source
        d1 = foundation.electrical.electrical;
        g = foundation.electrical.electrical;
        s = foundation.electrical.electrical;
        sp = foundation.electrical.electrical;
        g1 = foundation.electrical.electrical;
        s1 = foundation.electrical.electrical;
        d2 = foundation.electrical.electrical;
        t1 = foundation.electrical.electrical;
        t2 = foundation.electrical.electrical;
        t3 = foundation.electrical.electrical;
        t4 = foundation.electrical.electrical;
    end

    parameters
        dvth = {0, '1'};
        drdson = {0, '1'};
        dgfs = {0, '1'};
        dc = {0, '1'};
        dzth = {0, '1'};
        ls = {5e-11, '1'};
        ld = {1e-09, '1'};
        lg = {3e-09, '1'};
    end
```

```
parameters(Access=private,ExternalAccess=none)
    rs = {0.000222, '1'};
    rg = {2.2, '1'};
    rd = {5e-06, '1'};
    rmet = {4.5e-05, '1'};
    inn = {50, '1'};
    unn = {10, '1'};
    rmax = {0.0014, '1'};
    gmin = {69, '1'};
    act = {4.672, '1'};
    rsp = {2.4, '1'};
end

variables
    idrain = {value={0,'A'},priority=priority.none};
    igate = {value={0,'A'},priority=priority.none};
    isource = {value={0,'A'},priority=priority.none};
    G_S = {value={0,'A'},priority=priority.none};
end

components(ExternalAccess=observe)
    X_1 = myMOSFET.a6_40_d_var(a=act,rsp=rsp,dvth=dvth,drdson=drdson,dgfs=dgfs,inn=inn,unn=un
        gmin=gmin,rs=rs,rp=rd,dc=dc,rmet=rmet);
    R_G = foundation.electrical.elements.resistor(R={rg,'Ohm'});
    L_G = foundation.electrical.elements.inductor(l={lg*(if dgfs==99, 0 else 1 end),'H'},r={0
        i_L.priority=priority.none);
    R_SA = foundation.electrical.elements.resistor(R={(1*1000000),'Ohm'});
    L_S = foundation.electrical.elements.inductor(l={ls*(if dgfs==99, 0 else 1 end),'H'},r={0
        i_L.priority=priority.none);
    R_DA = foundation.electrical.elements.resistor(R={rd,'Ohm'});
    L_D = foundation.electrical.elements.inductor(l={ld*(if dgfs==99, 0 else 1 end),'H'},r={0
        i_L.priority=priority.none);
    R_DAUX = foundation.electrical.elements.resistor(R={10,'Ohm'});
    R_GAUX = foundation.electrical.elements.resistor(R={10,'Ohm'});
    R_SAUX = foundation.electrical.elements.resistor(R={10,'Ohm'});
    R_TH1 = foundation.electrical.elements.resistor(R={(5.98*0.001)+simscape.function.limit(c
    R_TH2 = foundation.electrical.elements.resistor(R={(73.91*0.001)+simscape.function.limit
    R_TH3 = foundation.electrical.elements.resistor(R={(258.69*0.001)+simscape.function.limi
    R_TH4 = foundation.electrical.elements.resistor(R={(305.94*0.001)+simscape.function.limi
    R_TH5 = foundation.electrical.elements.resistor(R={(377.8*0.001)+simscape.function.limit
    C_TH1 = foundation.electrical.elements.capacitor(c={(40.653*1e-06),'F'},r={0,'Ohm'},g={0
    C_TH2 = foundation.electrical.elements.capacitor(c={(180.521*1e-06),'F'},r={0,'Ohm'},g={0
    C_TH3 = foundation.electrical.elements.capacitor(c={(1.089*0.001),'F'},r={0,'Ohm'},g={0,
    C_TH4 = foundation.electrical.elements.capacitor(c={(823.295*1e-06),'F'},r={0,'Ohm'},g={0
    C_TH5 = foundation.electrical.elements.capacitor(c={(25.88*0.001),'F'},r={0,'Ohm'},g={0,
    C_TH6 = foundation.electrical.elements.capacitor(c={(30*0.001),'F'},r={0,'Ohm'},g={0,'1/0
end

connections
    connect(X_1.dd,d1);
    connect(X_1.g,g);
    connect(X_1.s0,s);
    connect(X_1.sp,sp);
    connect(X_1.tj,tj);
    connect(R_G.p,g1);
    connect(R_G.n,g);
    connect(L_G.p,gateInternal);
    connect(L_G.n,g1);
```

```
        connect(R_SA.p,s1);
        connect(R_SA.n,s);
        connect(L_S.p,sourceInternal);
        connect(L_S.n,s1);
        connect(R_DA.p,d1);
        connect(R_DA.n,d2);
        connect(L_D.p,drainInternal);
        connect(L_D.n,d2);
        connect(R_DAUX.p,drainInternal);
        connect(R_DAUX.n,d2);
        connect(R_GAUX.p,gateInternal);
        connect(R_GAUX.n,g1);
        connect(R_SAUX.p,sourceInternal);
        connect(R_SAUX.n,s1);
        connect(R_TH1.p,tj);
        connect(R_TH1.n,t1);
        connect(R_TH2.p,t1);
        connect(R_TH2.n,t2);
        connect(R_TH3.p,t2);
        connect(R_TH3.n,t3);
        connect(R_TH4.p,t3);
        connect(R_TH4.n,t4);
        connect(R_TH5.p,t4);
        connect(R_TH5.n,tcase);
        connect(C_TH1.p,tj);
        connect(C_TH1.n,*);
        connect(C_TH2.p,t1);
        connect(C_TH2.n,*);
        connect(C_TH3.p,t2);
        connect(C_TH3.n,*);
        connect(C_TH4.p,t3);
        connect(C_TH4.n,*);
        connect(C_TH5.p,t4);
        connect(C_TH5.n,*);
        connect(C_TH6.p,tcase);
        connect(C_TH6.n,*);
    end

    branches
        idrain : drain.i -> drainInternal.i;
        igate : gate.i -> gateInternal.i;
        isource : source.i -> sourceInternal.i;
        G_S: s1.i -> s.i;
    end

    equations
        drain.v == drainInternal.v;
        gate.v == gateInternal.v;
        source.v == sourceInternal.v;
        value(G_S,'A') == value(s1.v-s.v,'V')/(rs*(1+(simscape.function.limit(value(tj.v,'V'), -2
            4*0.001))-rmet);
    end

end
```

Create the file named `iauc100n04s6l014_wrapped.ssc`. This file is the component file for the Simscape component block in the model `ee_MOSFET_subckt`.

```
component iauc100n04s6l014_wrapped
% iauc100n04s6l014_wrapped :2.0
% MOSFET model converted from SPICE.<br/>
% <br/>
% To view information pertaining to this device model, see the <a href="matlab: open ee_MOSFET_su
    nodes
        drain = foundation.electrical.electrical; % d:right
        gate = foundation.electrical.electrical; % g:left
        tj = foundation.electrical.electrical; % tj:right
        tcase = foundation.electrical.electrical; % tc:right
        source = foundation.electrical.electrical; % s:right
    end

    components(ExternalAccess=observe)
        core_mosfet = myMOSFET.iauc100n04s6l014_updated();
    end

    annotations
        Icon = 'nmos.svg'
    end

    connections
        connect (drain, core_mosfet.drain)
        connect (gate, core_mosfet.gate)
        connect (tj, core_mosfet.tj)
        connect (tcase, core_mosfet.tcase)
        connect (source, core_mosfet.source)
    end
end
```



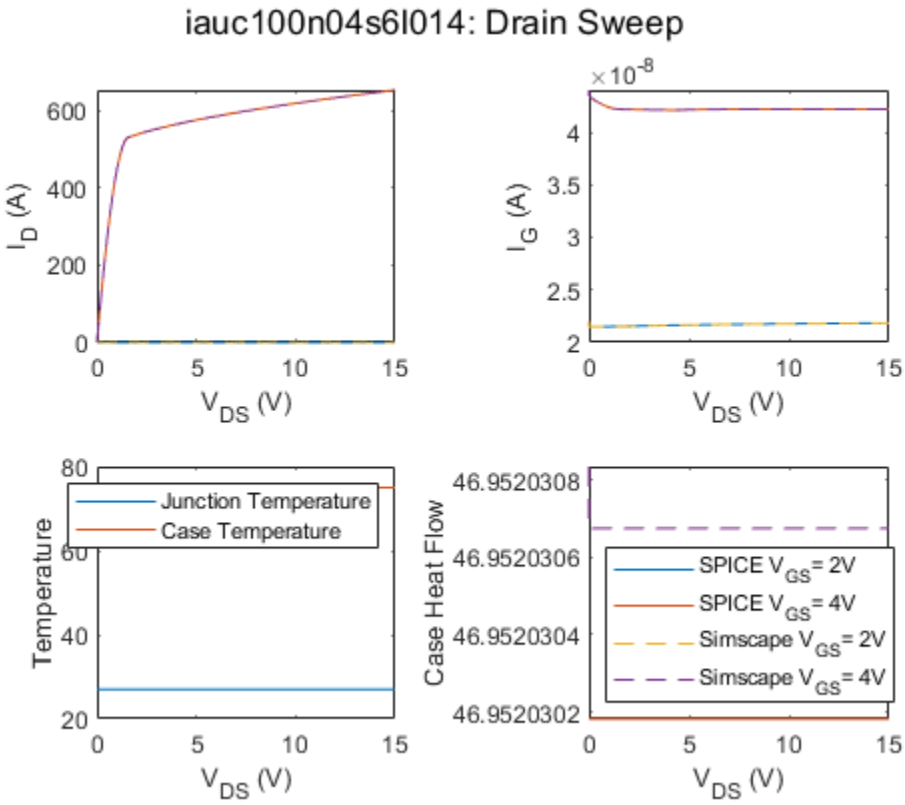

**SPICE Conversion of a MOSFET Subcircuit and Validation**

1. MOSFET curves: Define Conditions
2. SPICE subcircuit to Simscape component (see SPICE netlist), (see ssc), (see wrapped ssc)
3. Check results (see code)
4. Explore simulation results using Simscape Results Explorer
5. Learn more about this example

Copyright 2019-2022 The MathWorks, Inc.

### Verify Simulation Results

The `ee_MOSFET_subckt_results` script calls the `ee_convertedMosfetValidation` function to generate some standard characteristics such as Id versus Vgs, Id versus Vds, Qiss/ Gate charge, Qoss/ Output charge, and Breakdown voltage.




To plot these characteristics with different settings, double-click the block labeled Define Conditions and define the parameters. In the model, to run the simulations and plot the results, click **check results**.

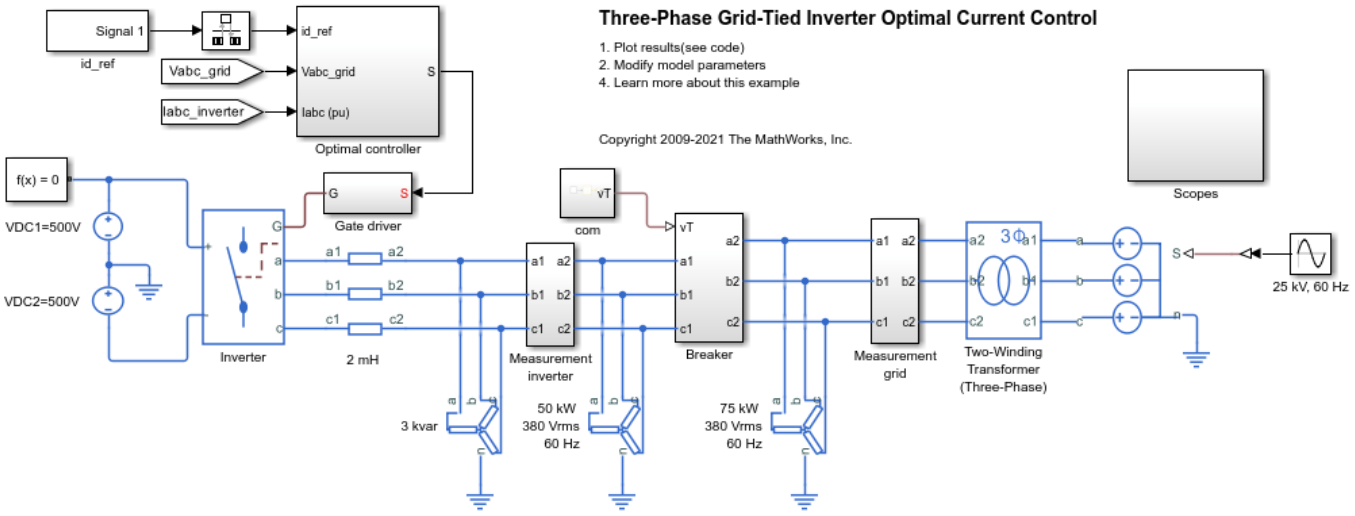
### Clean Up

Finally, delete the temporary directory and all its subdirectories.

# Three-Phase Grid-Tied Inverter Optimal Current Control

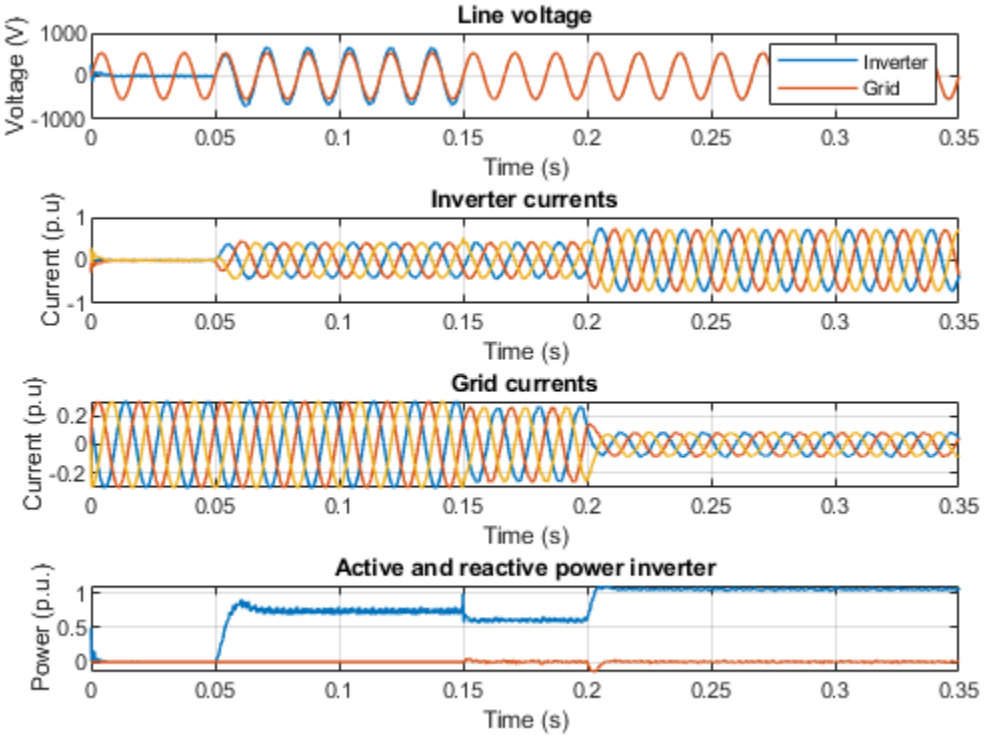
This example shows how to control the currents in a grid-tied inverter system. The Optimal controller subsystem implements an observer-based linear quadratic regulator strategy. To ensure zero steady state error, this example uses the observer as an alternative to the integral action. SPST switches connect the three-phase inverter to the grid. The switches are open at the beginning of the simulation to allow synchronization. At 0.15 seconds, the inverter is connected to the grid. Then, at 0.2 seconds the inverter increases the active power supplied to the grid. The Scopes subsystem contains scopes that allow you to see the simulation results. The inverter is implemented using Ideal Semiconductor Switch blocks. If you have a license for HDL Coder™, you can generate VHDL code for an FPGA using the Simscape™ HDL Workflow Advisor.

### Model




### Simulation Results

The plot below shows the synchronization voltage, inverter and grid currents, and inverter active and reactive power.


Line voltage
Voltage (V)
1000
0
-1000
Inverter
Grid
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
Time (s)
Inverter currents
Current (p.u)
1
0
-1
Time (s)
Grid currents
Current (p.u)
0.2
0
-0.2
Time (s)
Active and reactive power inverter
Power (p.u.)
1
0.5
0
Time (s)

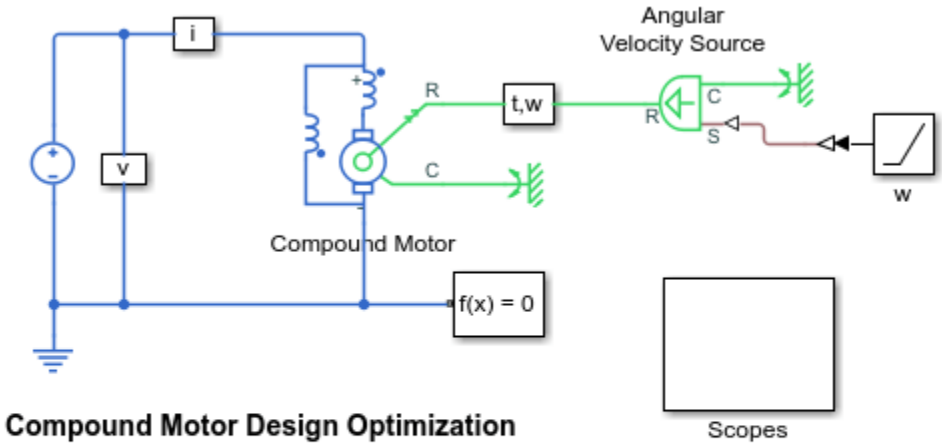
# Compound Motor Design Optimization

This example shows how to find design parameters that optimize a compound motor torque-speed curve to match the desired curve.

This example uses the **fminsearch** optimization algorithm. You can also use other optimization tools available in the Optimization Toolbox (TM), or develop your own.

### Model Overview

The compound motor has a DC supply voltage and the shaft motion is a ramp angular velocity. Ideal sensors measure torque and current to compute torque-speed, power-speed, and efficiency-speed curves.




### Modify Parameters

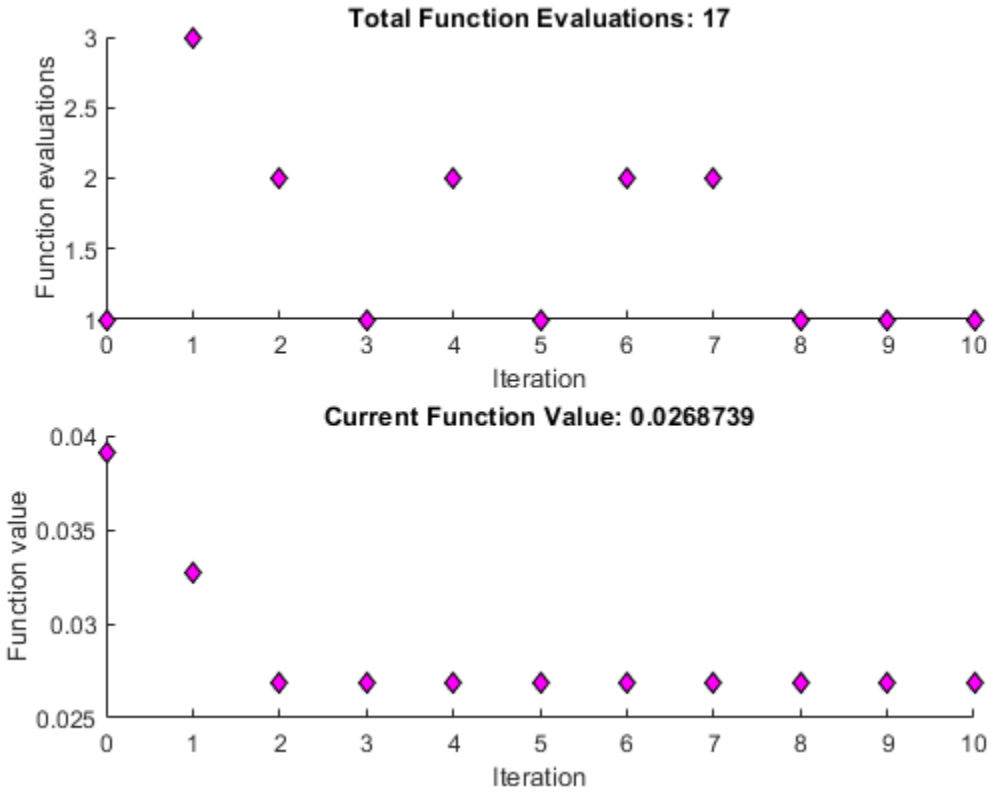
Open the data script to modify parameters that remain constant during optimization such as supply voltage and rotor damping. You can also choose the desired torque-speed curve used in the objective function by modifying nominal speed, nominal torque, and torque-speed slope at nominal speed.

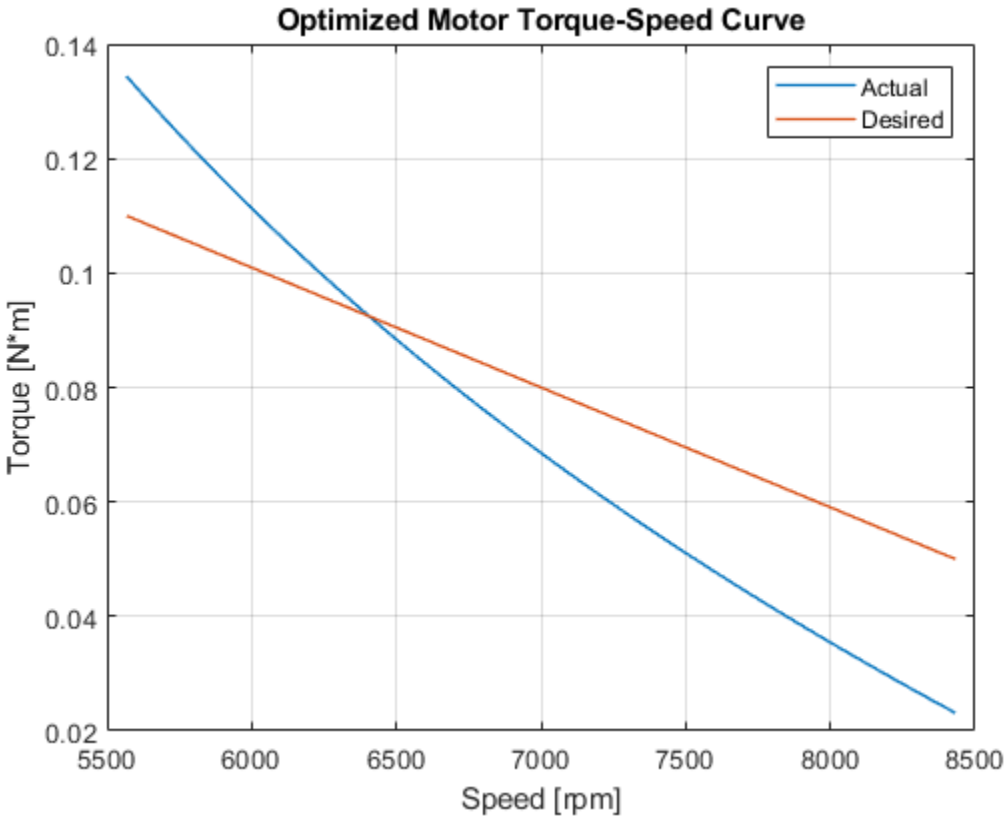
### Optimization

To find the minimum of a provided objective function, given an initial guess x0, this example uses the **fminsearch** optimization function. The objective function used is the maximum distance between the actual speed curve and the desired speed curve. The parameters to optimize are the parallel field winding resistance, the total back EMF constant and the ratio of parallel field-winding back EMF to total back EMF

```
 Iteration   Func-count     min f(x)         Procedure
```

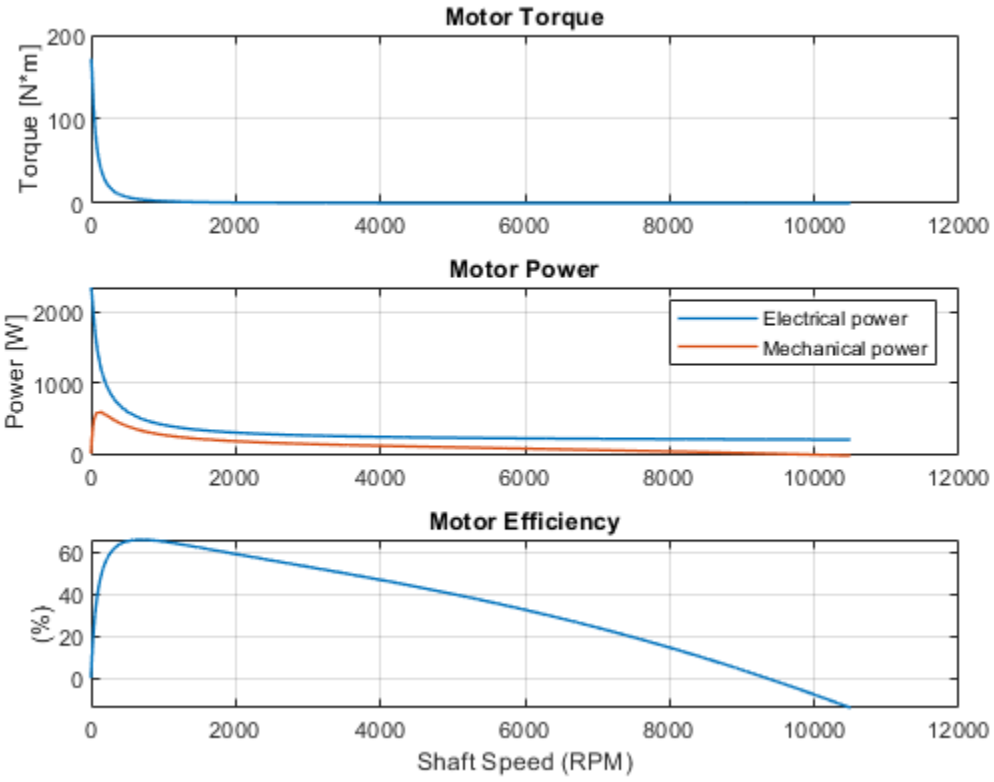
```
     0            1        0.0390633
     1            4        0.0326723         initial simplex
     2            6        0.0268739         expand
     3            7        0.0268739         reflect
     4            9        0.0268739         contract inside
     5           10        0.0268739         reflect
     6           12        0.0268739         contract inside
     7           14        0.0268739         contract outside
     8           15        0.0268739         reflect
     9           16        0.0268739         reflect
    10           17        0.0268739         reflect

Exiting: Maximum number of iterations has been exceeded
         - increase MaxIter option.
         Current function value: 0.026874

****************************
Optimal minimized cost: 0.026874 N*m
****************************
Motor parameters that produce the minimum distance to the desired torque-speed curve:
Optimal parallel field winding resistance = 225 Ohm
Optimal total back EMF = 1.365 H
Optimal ratio of parallel field winding back EMF divided by total back EMF = 0.42
****************************
```






**Simulation Results**

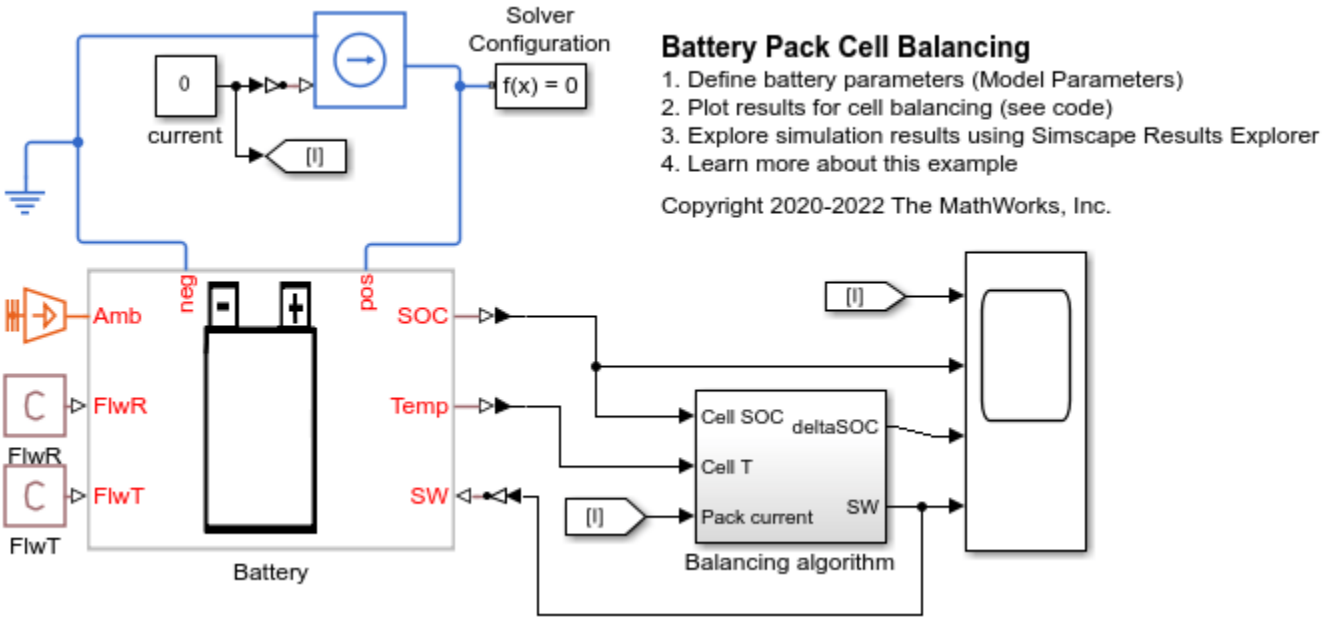
Plot torque-speed, power-speed, and efficiency-speed curves over an extended range of shaft speeds.


Motor Torque
Torque [N*m]
200
100
0
0
2000
4000
6000
8000
10000
12000
Motor Power
Power [W]
2000
1000
0
Electrical power
Mechanical power
Motor Efficiency
(%)
60
40
20
0
Shaft Speed (RPM)

# Battery Pack Cell Balancing

This example shows how to implement a passive cell balancing for a Lithium-ion battery pack. Cell-to-cell differences in the module create imbalance in cell state of charge and hence voltages. In this example, the balancing algorithm starts when the battery pack is idle and the difference in the cell state of charge is above a certain predefined value.

### Model Overview

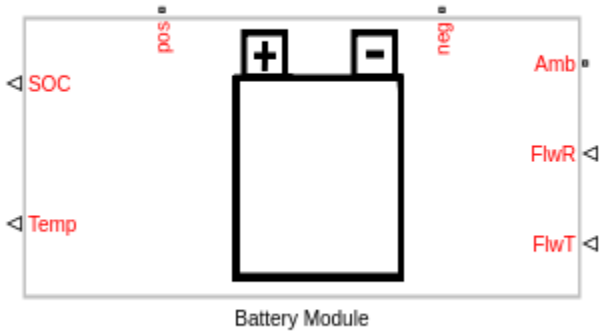



### Parameters and Inputs Overview

To use this model to create a unique battery module, first specify the number of series- and parallel-connected cells. Then specify the cell type for all individual cells by choosing one of these options for the **Choose cell type** parameter of the **Battery Module** block:

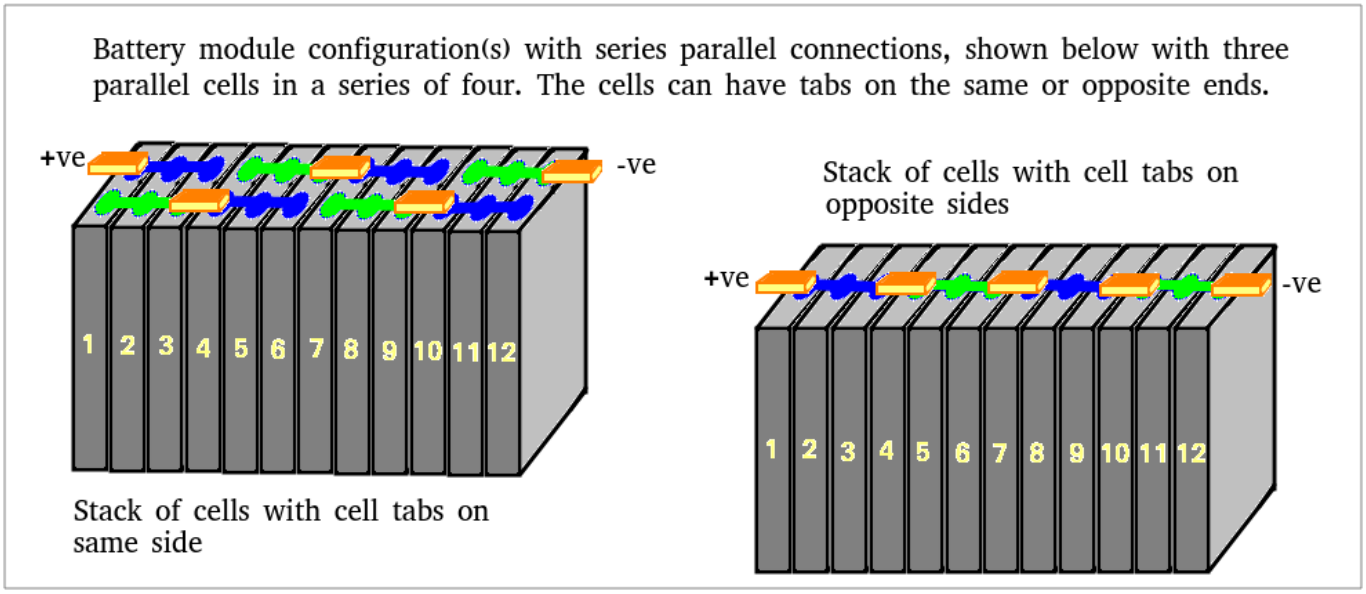
- `Pouch`
- `Can`
- `Compact cylindrical`
- `Regular cylindrical`

The battery module in this example comprises ten series-connected pouch cells.

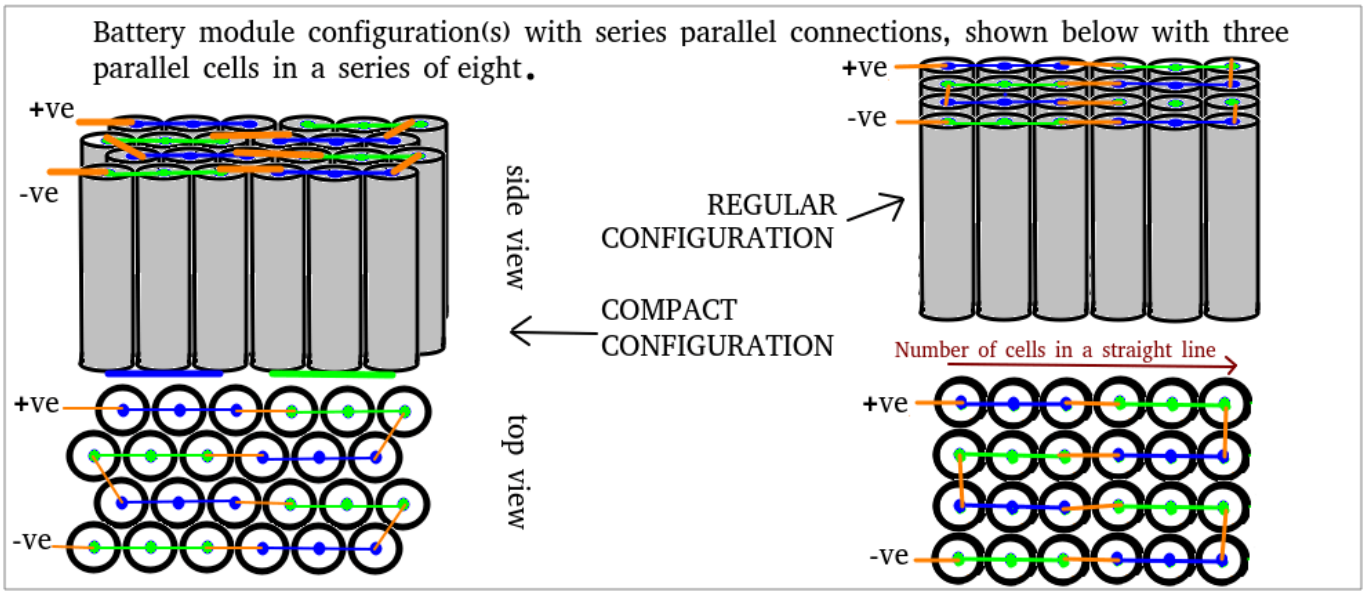



The two output ports, **SOC** and **Temp**, provide information regarding the state of charge and the temperature of each cell in the module. The thermal port, **Amb**, is used to define the ambient temperature in the simulation. The electrical ports, **pos** and **neg**, define the electrical positive and negative terminals, respectively. The two input ports, **FlwR** and **FlwT**, define the battery coolant flow rate control and inlet temperature into the module. The third input port, **SW**, defines the switch state, either on or off, for the passive cell balancing through a resistor.

The figure below shows examples of battery cells in `Pouch` and `Can` configurations.



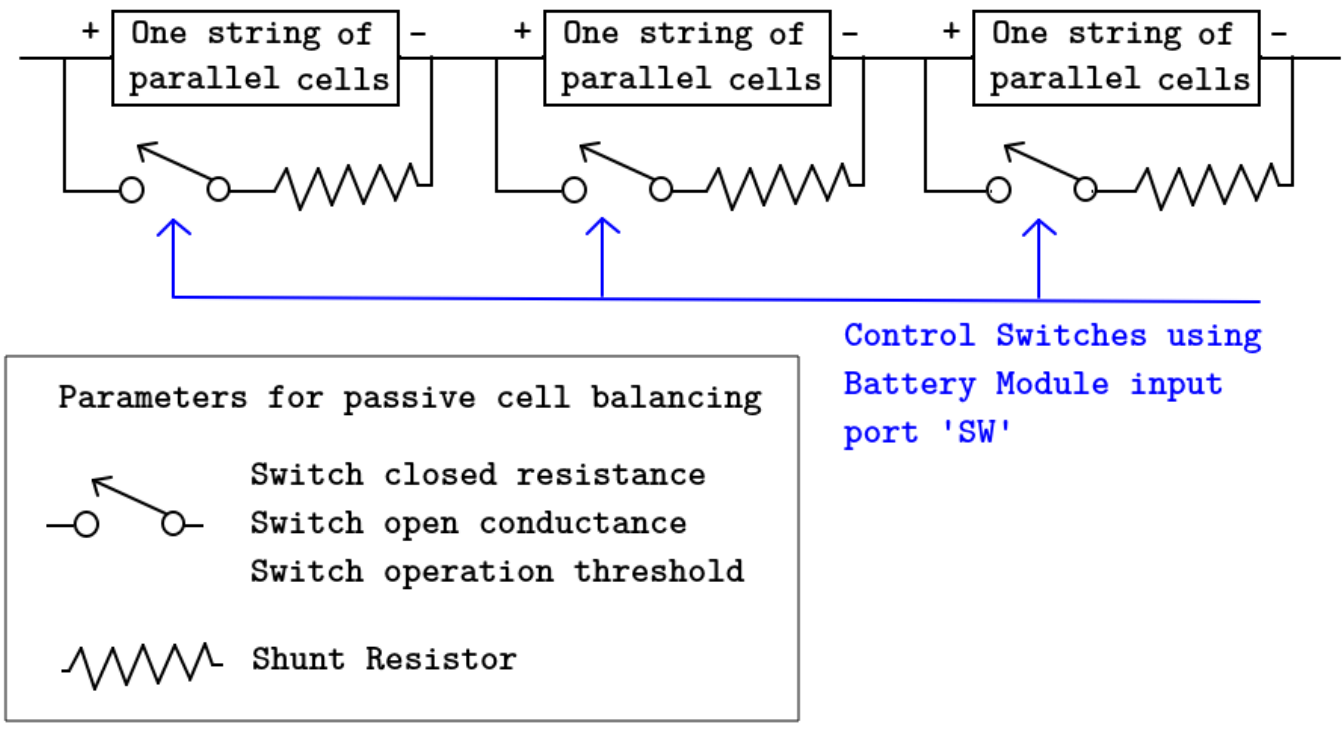

The figure below shows examples of battery cells in `Compact cylindrical` and `Regular cylindrical` configurations.




These are the parameters in the battery module:

- **Vector of temperatures, T** — Temperatures at which the cell or module data for temperature-varying properties are tabulated, specified as a vector.
- **Single cell Ahr rating, baseline** — Cell capacity at the temperatures defined in the **Vector of temperatures, T** parameter, specified as a vector.
- **Vector of state of charge values, SOC** — Range of values between 0 and 1 at which the cell electrical parameters are defined, specified as a vector.
- **Vector of coolant flowrates, L** — Coolant mass flow rate values at which a lookup table for cell cooling is defined. This parameter needs to cover multiple points in the flow range of interest. This parameter defines the size of the **Effective rate of coolant heat transfer** parameter and is specified as a vector.
- **No load voltage, V0** — Cell open-circuit potential values at different **Vector of state of charge values, SOC** and **Vector of temperatures, T** points, specified as a matrix.
- **Terminal resistance, R0** — Cell ohmic resistance values at different **Vector of state of charge values, SOC** and **Vector of temperatures, T** points, specified as a matrix.
- **Self-discharge** — Cell self discharge option; `Disabled` is selected for this example.
- **Polarization resistance** — Polarization resistance values at different **Vector of state of charge values, SOC** and **Vector of temperatures, T** points, specified as a matrix.
- **Time constant** — Time constant at different **Vector of state of charge values, SOC** and **Vector of temperatures, T** points, specified as a matrix.
- **Cell thermal mass** — Thermal mass of a single cell, specified as a scalar.
- **Cell thermal conductivity** — Cell through-plane conductivity for pouch and can cells, or the radial conductivity for cylindrical cells, specified as a scalar.
- **Heat transfer coefficient to ambient** — Heat transfer coefficient value, specified as a scalar.
- **Number of series connected cells Ns** — Number of strings in series, specified as an integer.

- **Number of parallel connected cells Np** — Number of parallel-cells in a string, specified as an integer.
- **Choose cell type** — Type of cell, specified as either `Pouch`, `Can`, `Compact cylindrical`, or `Regular cylindrical`.
- **Cell height** — Cell height, specified as a scalar.
- **Cell width** — Cell width for `Pouch` and `Can` cells, specified as a scalar.
- **Cell thickness** — Cell thickness for `Pouch` or `Can` cells, specified as a scalar.
- **Cell diameter** — Cell diameter for `Compact cylindrical` or `Regular cylindrical`, specified as a scalar.
- **Number of cylindrical cells in a straight line** — Number of cylindrical cells arranged in a straight line for packaging, specified as an integer.
- **Accessory total resistance** — Resistance that combines all inline resistance in a module, specified as a scalar. This resistance is the sum of cell tab, busbar, cable and/or weld resistances, specified as a scalar.
- **Cell balancing** — Cell balancing method, specified as either `none` or `passive`. In this example, this parameter is set to `passive`. Upon selection of `passive` cell balancing, four parameters become visible. These parameters are: **Shunt resistor**, **Switch closed resistance**, **Switch open conductance** and **Switch operation threshold**. Passive cell balancing is shown schematically in the figure below:




- **Effective rate of coolant heat transfer from each cell** — Estimate of the thermal resistance (W/K) of heat transfer from battery cells to coolant, specified as a 3-D matrix of scalar values. The 3-D matrix size depends on the **Vector of temperatures, T**, **Vector of coolant flowrates, L** and

**NsxNp** parameters. The **NsxNp** parameter is the total number of cells in the module. The battery cooling is represented as a lookup table or 3-D matrix of size [T,L,Ns*Np] and the values are calculated using detailed 3-D methods such as computational fluid dynamics. The values of the matrix depend on the actual hardware design of the cooling system or cold plates in the module. This matrix is set to zero in the current example, in the file `ee_lithium_pack_cellBalancing_ini.m`.

- **External heat** — External heat input to each cell in a module due to a hot component placed near the module, specified as a vector. It is set to zero in the current example.
- **Vector of initial cell temperature** — Cell initial temperature, specified as a vector.
- **Vector of initial cell state of charge** — Cell initial state of charge, specified as a vector. First cell is imbalanced and has the lowest state of charge compared to other cells, as specified in the file `ee_lithium_pack_cellBalancing_ini.m`.
- **Cell Ahr rating variation** — Cell-to-cell variations in cell capacity at all **Vector of temperatures, T** points for each cell, specified as a vector of scalar values. If you set this array to 1, all cell capacity is the same. The array values for a cell are multiplied with the value specified in the **Single cell Ahr rating, baseline** parameter to calculate the actual capacity or the Ahr rating of the cell.

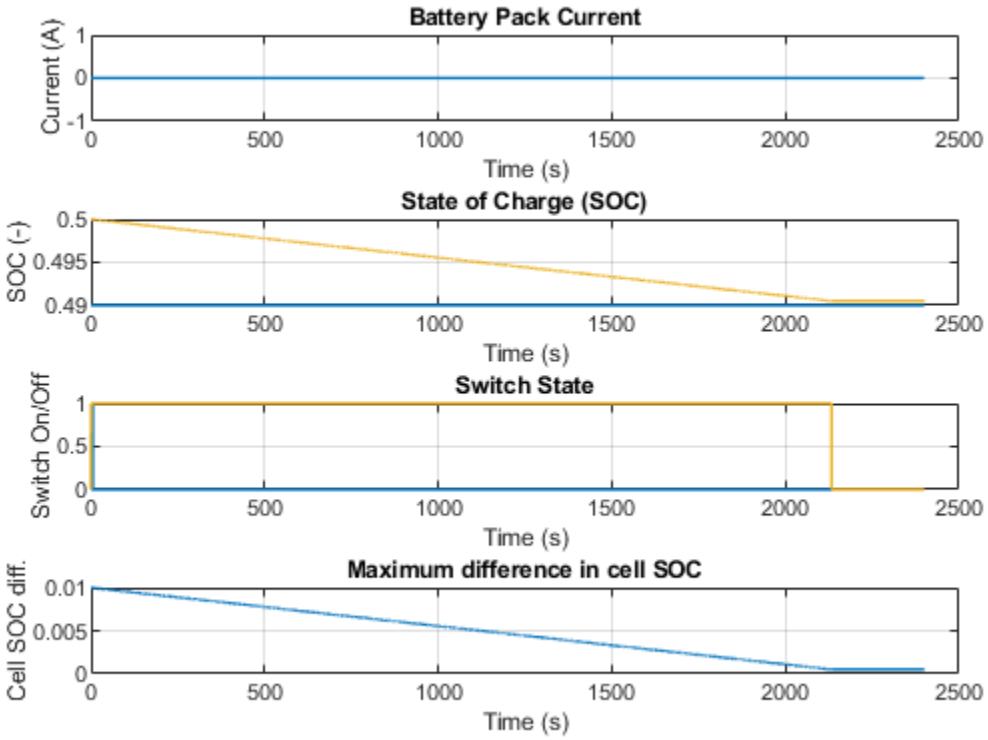
To define the battery coolant flow rate and temperature, specify these inputs:

- **FlwR** — Value between 0 and 1, specified as a scalar. The **FlwR** input value is used to dynamically choose the right value of the flow rate during the simulations. The value of the **FlwR** input defines the actual flow rate in the module. In the **Vector of coolant flowrates L** parameter, **FlwR** equal to 0 means no flow, while **FlwR** equal to 1 means highest flow rate value. **FlwR** is set to a low value of 0.001 in this example.
- **FlwT** — Positive or negative value that, when summed to the ambient temperature, equals the coolant inlet temperature. A value of +15 for the **FlwT** input and 273.15 K at the **Amb** port makes the coolant inlet temperature equal to 273.15 + 15 = 288.15K. A value of -15 for the **FlwT** input and 273.15 K at the **Amb** makes the coolant inlet temperature equal to 273.15 - 15 = 258.15 K. **FlwT** is set to 0 in this example.

In this example, the battery pack starts at an ambient temperature of 25 degrees Celsius. The battery pack is idle and there is no current flowing through it. The cell balancing algorithm activates when the minimum difference in the cell state of charge is greater than 0.05% and the battery pack is idle. The algorithm charges closes switches for all cells other than the one with lowest state of charge. Small current flows through the balancing resistor and all the cell state of charge come to almost similar values in about 2400s.

### Simulation Results from Simscape Logging

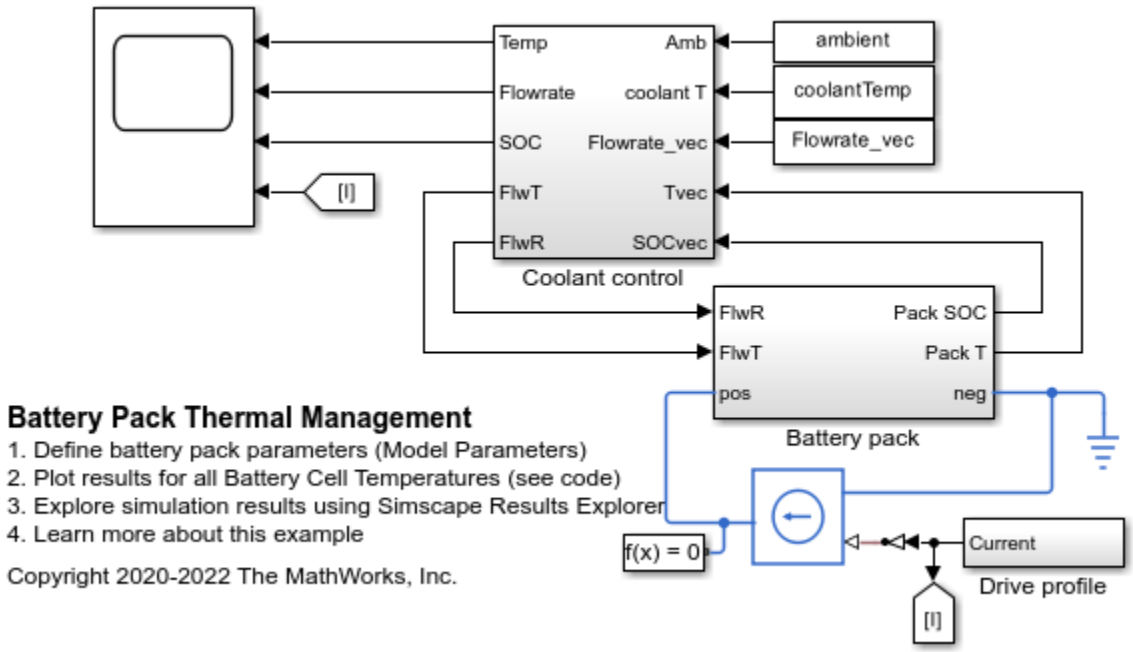
The plot below shows the battery current, the state of charge, the maximum difference in the cell state of charge, and the switching operation based on the logic defined in the Balancing algorithm subsystem. All cells, other than the first cell with lowest state of charge, have their switch closed and lose energy through the balancing resistor. All cells of the module reach almost identical state of charge values in less than 45 minutes.


Battery Pack Current
Current (A)
1
0
-1
0
500
1000
1500
2000
2500
Time (s)
State of Charge (SOC)
SOC (-)
0.5
0.495
0.49
0
500
1000
1500
2000
2500
Time (s)
Switch State
Switch On/Off
1
0.5
0
0
500
1000
1500
2000
2500
Time (s)
Maximum difference in cell SOC
Cell SOC diff.
0.01
0.005
0
0
500
1000
1500
2000
2500
Time (s)

# Battery Pack Thermal Management

This example shows how to model an automotive battery pack for thermal management tasks. The battery pack consists of several battery modules, which are combinations of cells in series and parallel. Each battery cell is modeled using the **Battery (Table-Based)** Simscape Electrical block. In this example, the initial temperature and the state of charge are the same for all cells. Four battery modules, three similar and one differing from the other three, are connected in series to simulate a battery pack. The results in this example assume an initial ambient temperature equal to 25 degree Celsius. The Coolant Controls subsystem defines the logic used to determine the battery pack coolant flow rate.

### Model Overview

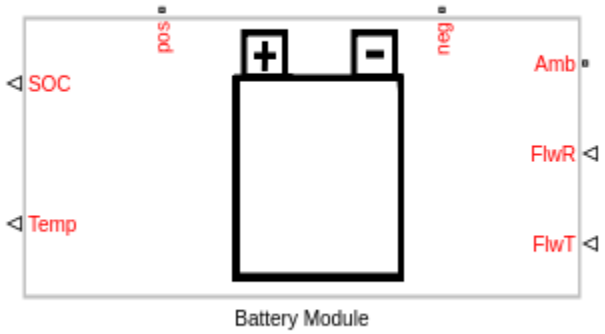



### Parameters and Input Overview

To use this model to create a unique battery module, first specify the number of series- and parallel-connected cells. Then specify the cell type for all individual cells by choosing one of these options for the **Choose cell type** parameter of the **Battery Module** block:

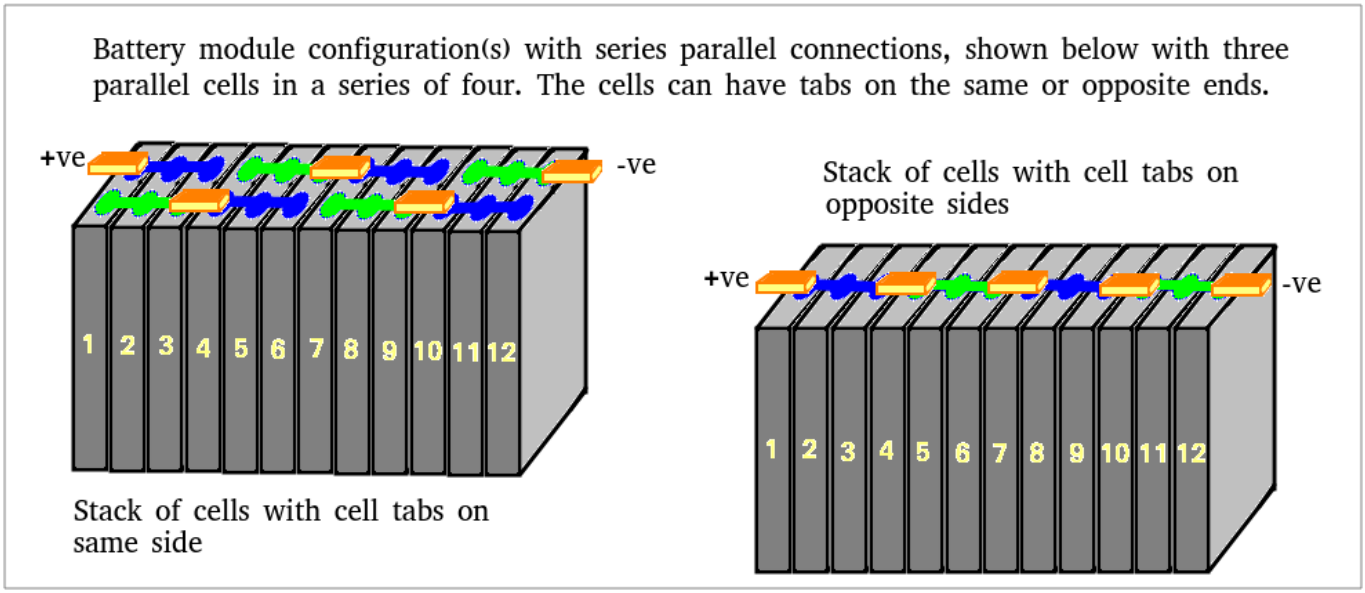
- `Pouch`
- `Can`
- `Compact cylindrical`
- `Regular cylindrical`

This example uses pouch-type cells. Module A,B and C consist of five series-connected and two parallel-connected cells. Module D consists of six series-connected and two parallel-connected cells.

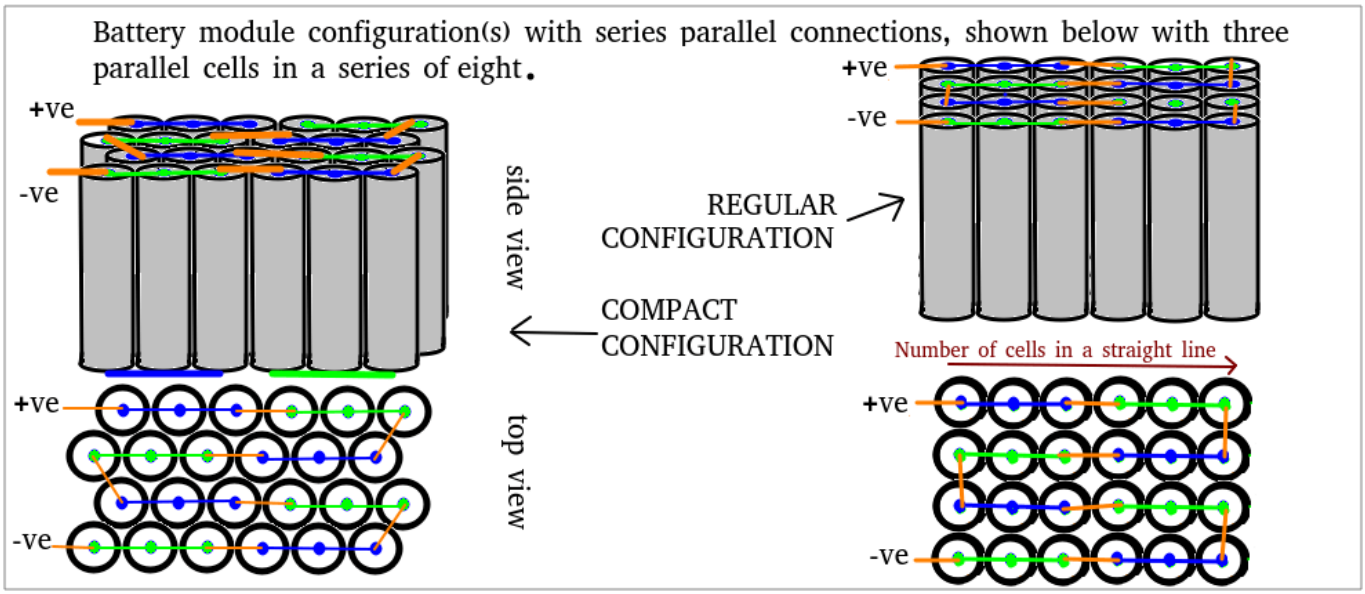



The two output ports, **SOC** and **Temp**, provide information regarding the state of charge and the temperature of each cell in the module. The thermal port, **Amb**, is used to define the ambient temperature in the simulation. The electrical ports, **pos** and **neg**, define the electrical positive and negative terminals, respectively. The two input ports, **FlwR** and **FlwT**, define the battery coolant flow rate control and inlet temperature into the module.

The figure below shows examples of battery cells in `Pouch` and `Can` configurations.




The figure below shows examples of battery cells in `Compact cylindrical` and `Regular cylindrical` configurations.




These are the parameters in the battery module:

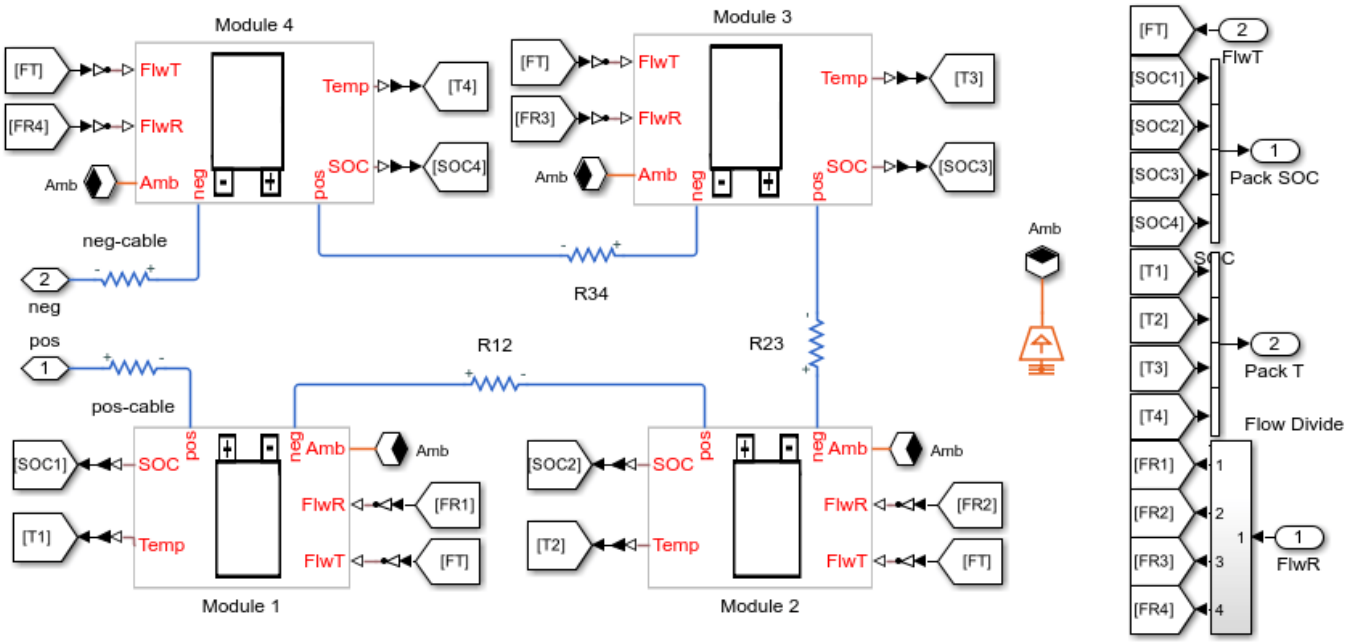
- **Vector of temperatures, T** — Temperatures at which the cell or module data for temperature-varying properties are tabulated, specified as a vector.
- **Single cell Ahr rating, baseline** — Cell capacity at the temperatures defined in the **Vector of temperatures, T** parameter, specified as a vector.
- **Vector of state of charge values, SOC** — Range of values between 0 and 1 at which the cell electrical parameters are defined, specified as a vector.
- **Vector of coolant flowrates, L** — Coolant mass flow rate values at which a lookup table for cell cooling is defined. This parameter needs to cover multiple points in the flow range of interest. This parameter defines the size of the **Effective rate of coolant heat transfer** parameter and is specified as a vector.
- **No load voltage, V0** — Cell open-circuit potential values at different **Vector of state of charge values, SOC** and **Vector of temperatures, T** points, specified as a matrix.
- **Terminal resistance, R0** — Cell ohmic resistance values at different **Vector of state of charge values, SOC** and **Vector of temperatures, T** points, specified as a matrix.
- **Polarization resistance** — Polarization resistance values at different **Vector of state of charge values, SOC** and **Vector of temperatures, T** points, specified as a matrix.
- **Time constant** — Time constant at different **Vector of state of charge values, SOC** and **Vector of temperatures, T** points, specified as a matrix.
- **Cell thermal mass** — Thermal mass of a single cell, specified as a scalar.
- **Cell thermal conductivity** — Cell through-plane conductivity for pouch and can cells, or the radial conductivity for cylindrical cells, specified as a scalar.
- **Heat transfer coefficient to ambient** — Heat transfer coefficient value, specified as a scalar.
- **Number of series connected cells Ns** — Number of strings in series, specified as an integer.
- **Number of parallel connected cells Np** — Number of parallel-cells in a string, specified as an integer.

- **Choose cell type** — Type of cell, specified as either `Pouch`, `Can`, `Compact cylindrical`, or `Regular cylindrical`.
- **Cell height** — Cell height, specified as a scalar.
- **Cell width** — Cell width for `Pouch` and `Can` cells, specified as a scalar.
- **Cell thickness** — Cell thickness for `Pouch` or `Can` cells, specified as a scalar.
- **Cell diameter** — Cell diameter for `Compact cylindrical` or `Regular cylindrical`, specified as a scalar.
- **Number of cylindrical cells in a straight line** — Number of cylindrical cells arranged in a straight line for packaging, specified as an integer.
- **Accessory total resistance** — Resistance that combines all inline resistance in a module, specified as a scalar. This resistance is the sum of cell tab, busbar, cable and/or weld resistances, specified as a scalar.
- **Cell balancing** — Cell balancing method, specified as either `none` or `passive`. In this example, this parameter is set to `none`.
- **Effective rate of coolant heat transfer from each cell** — Estimate of the thermal resistance (W/K) of heat transfer from battery cells to coolant, specified as a 3-D matrix of scalar values. The 3-D matrix size depends on the **Vector of temperatures, T**, **Vector of coolant flowrates, L** and **NsxNp** parameters. The **NsxNp** parameter is the total number of cells in the module. The battery cooling is represented as a lookup table or 3-D matrix of size [T,L,Ns*Np] and the values are calculated using detailed 3-D methods such as computational fluid dynamics. The values of the matrix depend on the actual hardware design of the cooling system or cold plates in the module. The performance of the cold plate is controlled using input values **FlwR** and **FlwT**.
- **External heat** — External heat input to each cell in a module due to a hot component placed near the module, specified as a vector.
- **Vector of initial cell temperature** — Cell initial temperature, specified as a vector.
- **Vector of initial cell state of charge** — Cell initial state of charge, specified as a vector.
- **Cell Ahr rating variation** — Cell-to-cell variations in cell capacity at all **Vector of temperatures, T** points for each cell, specified as a vector of scalar values. If you set this array to 1, all cell capacity is the same. The array values for a cell are multiplied with the value specified in the **Single cell Ahr rating, baseline** parameter to calculate the actual capacity or the Ahr rating of the cell.

To define the battery coolant flow rate and temperature, specify these inputs:

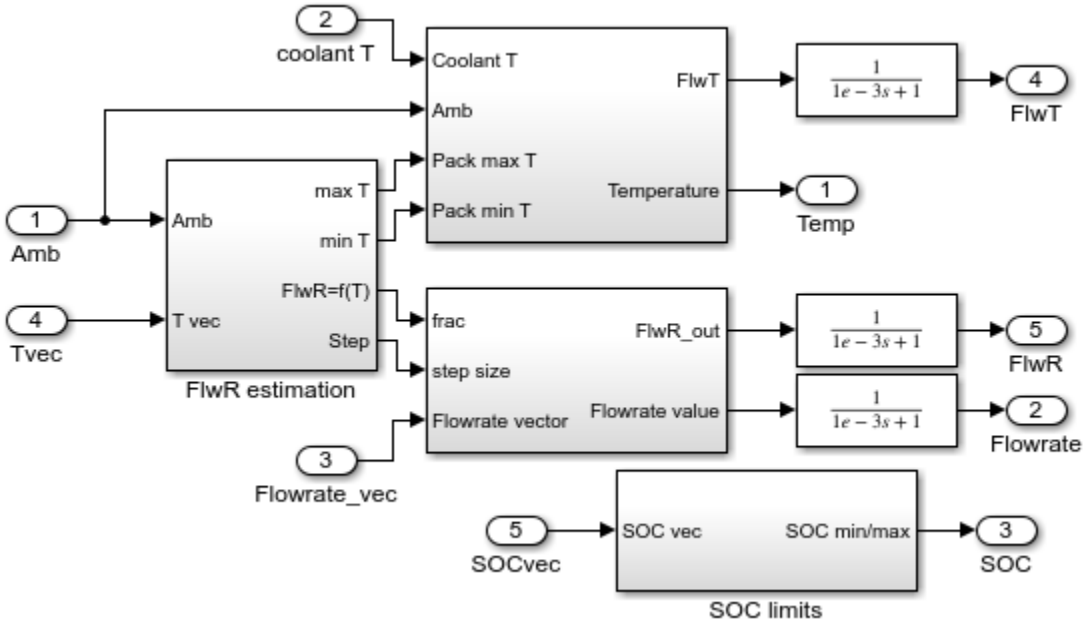
- **FlwR** — Value between 0 and 1, specified as a scalar. The **FlwR** input value is used to dynamically choose the right value of the flow rate during the simulations. The value of the **FlwR** input defines the actual flow rate in the module. In the **Vector of coolant flowrates L** parameter, **FlwR** equal to 0 means no flow, while **FlwR** equal to 1 means highest flow rate value.
- **FlwT** — Positive or negative value that, when summed to the ambient temperature, equals the coolant inlet temperature. A value of +15 for the **FlwT** input and 273.15 K at the **Amb** port makes the coolant inlet temperature equal to 273.15 + 15 = 288.15K. A value of -15 for the **FlwT** input and 273.15 K at the **Amb** makes the coolant inlet temperature equal to 273.15 - 15 = 258.15 K

In this example, the battery pack consists of four modules connected in series. The first three modules are identical. The fourth module has a different number of cells, `Ns4`, and cooling efficiency, `coolantQ4`, defined in the `ee_lithium_pack_cooling_ini.m` file. Cables, modeled as resistors, connect the modules to each other. All modules have a different flow rate of coolant. The Flow Divide block in the Battery Pack subsystem determines the flow rate that reaches each battery module.



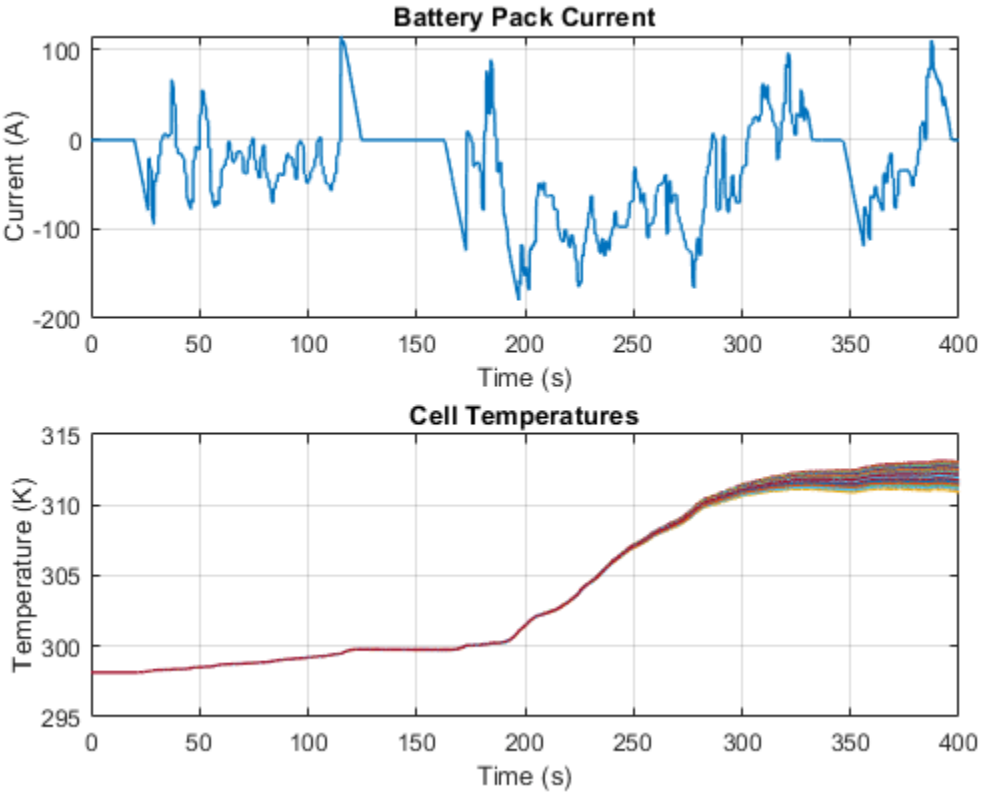

### Coolant Control Subsystem Overview

The Coolant Control subsystem tracks the minimum and maximum temperatures in the battery pack. This subsystem calculates the flow rate based on the largest value between the difference in the maximum and the minimum cell temperatures in the pack, and the difference in the maximum temperature in the pack and the value at the **Amb** port. For a difference of 10 degrees Celsius or more, *FlwR* is set to 1, else it is linearly scaled till zero, when there is no temperature difference between the different cells and the battery pack temperature is very close the the value set at the **Amb** port. In this example, the coolant inlet temperature, defined in the `coolantTemp` workspace variable in the `ee_lithium_pack_cooling_ini.m` file, is constant.




**Simulation Results from Simscape Logging**

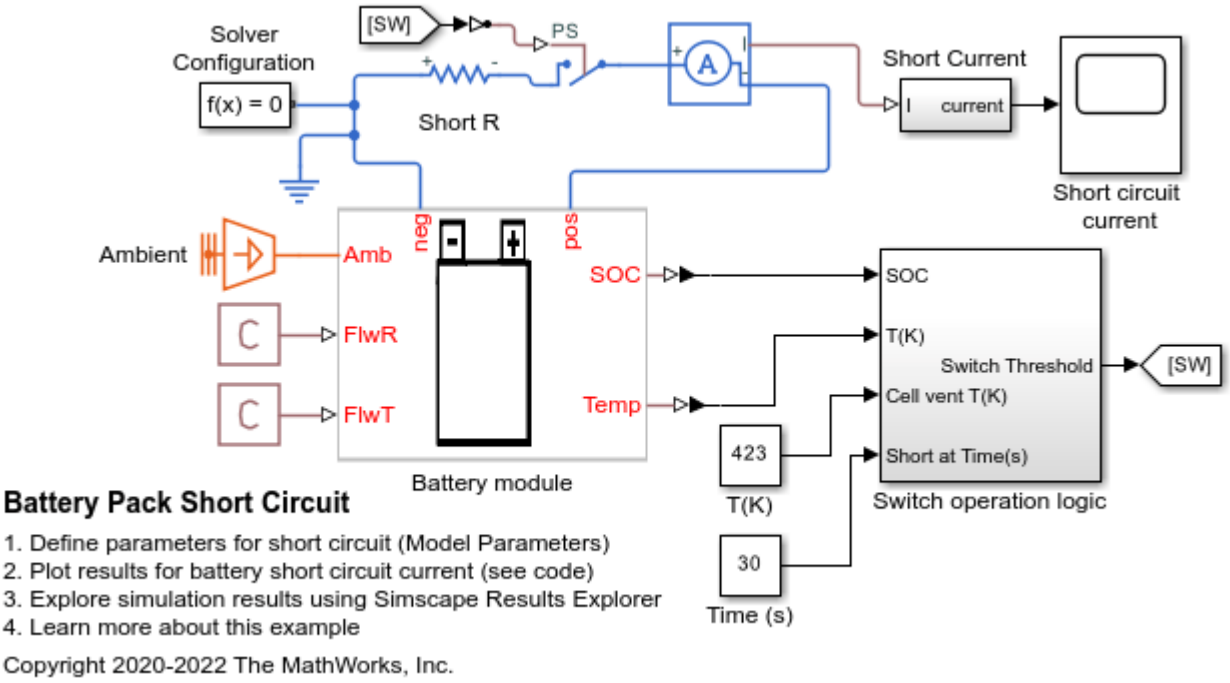
This example simulates a 600 seconds drive profile. The flow rate increases along with the battery pack temperature, leading to a better cooling of the battery pack.


Battery Pack Current
Current (A)
100
0
-100
-200
0
50
100
150
200
250
300
350
400
Time (s)
Cell Temperatures
Temperature (K)
315
310
305
300
295
0
50
100
150
200
250
300
350
400
Time (s)

# Battery Pack Short Circuit

This example shows how to model a short-circuit in a lithium-ion battery module. The battery module consists of 30 cells with a string of three parallel cells connected in a series of ten strings. Each battery cell is modeled using the **Battery (Table-Based)** Simscape Electrical block. In this example, the initial temperature and the state of charge are the same for all cells. There is no coolant flow modeled in this example. The battery module is shorted with a 0.1mOhm resistor. There is an inrush current followed by cell quick discharge and heating up. Once the cell reaches the trigger temperature for thermal runaway and cell venting, the electrical circuit is disconnected to stop the electrical simulation.

### Model

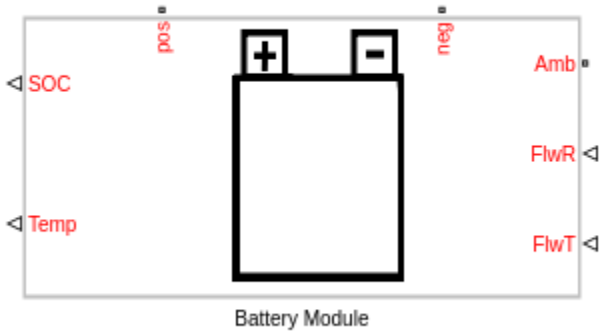



### Define Parameters and Inputs

To use this module to create a unique battery module, first specify the number of series and parallel-connected cells. Then specify the cell type for all individual cells by choosing one of these options for **Choose cell type** parameter of the **Battery Module** block:

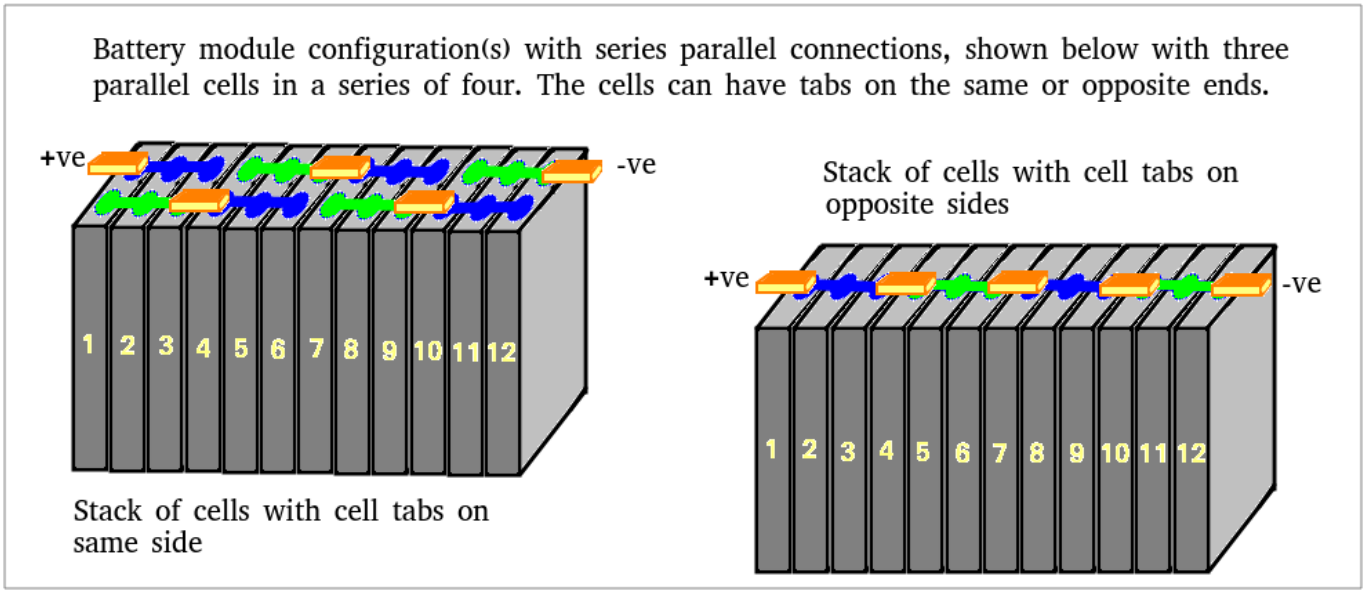
- `Pouch`
- `Can`
- `Compact cylindrical`
- `Regular cylindrical`

This example uses pouch-type cells. Module A,B and C consist of 8 series- connected and two parallel-connected cells. Module D consists of 12 series-connected and two parallel-connected cells.

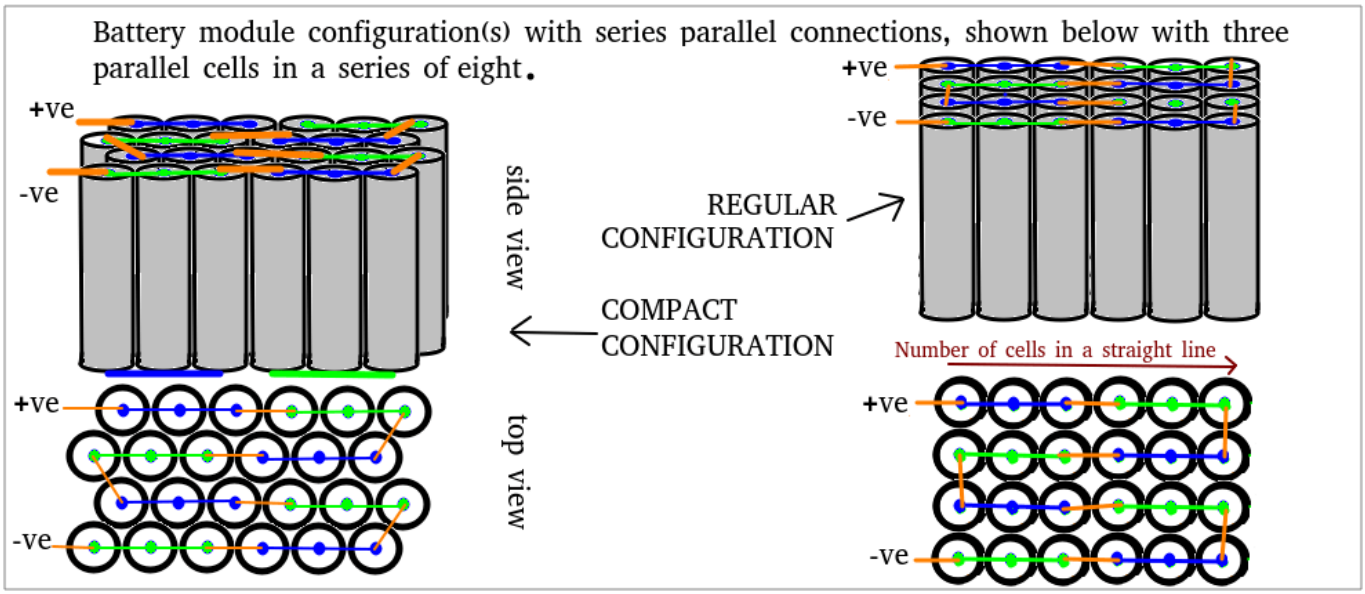



The two output ports, **SOC** and **Temp**, provide information regarding the state of charge and the temperature of each cell in the module. The thermal port, **Amb** is used to define the ambient temperature in the simulation. The electrical ports, **pos** and **neg**, define the electrical positive and negative terminals, respectively. The two input ports, **FlwR** and **FlwT**, define the battery coolant flow rate control and inlet temperature into the module.

The figure below shows examples of battery cells in `Pouch` and `Can` configurations.




The figure below shows examples of battery cells in `Compact cylindrical` and `Regular cylindrical` configurations.




These are the parameters in the battery module:

- **Vector of temperatures, T** — Temperatures at which the cell or module data for temperature-varying properties are tabulated, specified as a vector.
- **Single cell Ahr rating, baseline** — Cell capacity at the temperatures defined in the **Vector of temperatures, T** parameter, specified as a vector.
- **Vector of state of charge values, SOC** — Range of values between 0 and 1 at which the cell electrical parameters are defined, specified as a vector.
- **Vector of coolant flowrates, L** — Coolant mass flow rate values at which a lookup table for cell cooling is defined. This parameter needs to cover multiple points in the flow range of interest. This parameter defines the size of the **Effective rate of coolant heat transfer** parameter and is specified as a vector.
- **No load voltage, V0** — Cell open-circuit potential values at different **Vector of state of charge values, SOC** and **Vector of temperatures, T** points, specified as a matrix.
- **Terminal resistance, R0** — Cell ohmic resistance values at different **Vector of state of charge values, SOC** and **Vector of temperatures, T** points, specified as a matrix.
- **Polarization resistance** — Polarization resistance values at different **Vector of state of charge values, SOC** and **Vector of temperatures, T** points, specified as a matrix.
- **Time constant** — Time constant at different **Vector of state of charge values, SOC** and **Vector of temperatures, T** points, specified as a matrix.
- **Cell thermal mass** — Thermal mass of a single cell, specified as a scalar.
- **Cell thermal conductivity** — Cell through-plane conductivity for pouch and can cells, or the radial conductivity for cylindrical cells, specified as a scalar.
- **Heat transfer coefficient to ambient** — Heat transfer coefficient value, specified as a scalar.
- **Number of series connected cells Ns** — Number of strings in series, specified as an integer.
- **Number of parallel connected cells Np** — Number of parallel-cells in a string, specified as an integer.

- **Choose cell type** — Type of cell, specified as either `Pouch`, `Can`, `Compact cylindrical`, or `Regular cylindrical`.
- **Cell height** — Cell height, specified as a scalar.
- **Cell width** — Cell width for `Pouch` and `Can` cells, specified as a scalar.
- **Cell thickness** — Cell thickness for `Pouch` or `Can` cells, specified as a scalar.
- **Cell diameter** — Cell diameter for `Compact cylindrical` or `Regular cylindrical`, specified as a scalar.
- **Number of cylindrical cells in a straight line** — Number of cylindrical cells arranged in a straight line for packaging, specified as an integer.
- **Accessory total resistance** — Resistance that combines all inline resistance in a module, specified as a scalar. This resistance is the sum of cell tab, busbar, cable and/or weld resistances, specified as a scalar.
- **Cell balancing** — Cell balancing method, specified as either `none` or `passive`. In this example, this parameter is set to `none`.
- **Effective rate of coolant heat transfer from each cell** — Estimate of the thermal resistance (W/K) of heat transfer from battery cells to coolant, specified as a 3-D matrix of scalar values. The 3-D matrix size depends on the **Vector of temperatures, T**, **Vector of coolant flowrates, L** and **NsxNp** parameters. The **NsxNp** parameter is the total number of cells in the module. The battery cooling is represented as a lookup table or 3-D matrix of size [T,L,Ns*Np] and the values are calculated using detailed 3-D methods such as computational fluid dynamics. The values of the matrix depend on the actual hardware design of the cooling system or cold plates in the module. The performance of the cold plate is controlled using input values **FlwR** and **FlwT**.
- **External heat** — External heat input to each cell in a module due to a hot component placed near the module, specified as a vector.
- **Vector of initial cell temperature** — Cell initial temperature, specified as a vector.
- **Vector of initial cell state of charge** — Cell initial state of charge, specified as a vector.
- **Cell Ahr rating variation** — Cell-to-cell variations in cell capacity at all **Vector of temperatures, T** points for each cell, specified as a vector of scalar values. If you set this array to 1, all cell capacity is the same. The array values for a cell are multiplied with the value specified in the **Single cell Ahr rating, baseline** parameter to calculate the actual capacity or the Ahr rating of the cell.

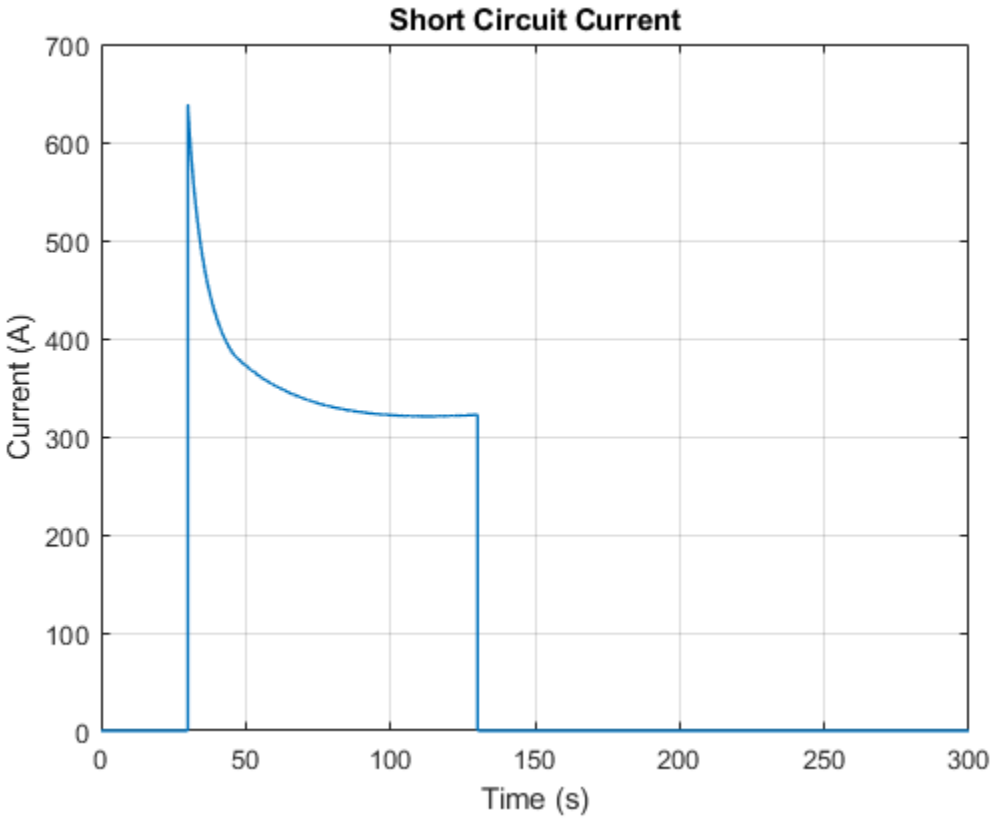
To define the battery coolant flow rate and temperature, specify these inputs:

- **FlwR** — Value between 0 and 1, specified as a scalar. The **FlwR** input value is used to dynamically choose the right value of the flow rate during the simulations. The value of the **FlwR** input defines the actual flow rate in the module. In the **Vector of coolant flowrates L** parameter, **FlwR** equal to 0 means no flow, while **FlwR** equal to 1 means highest flow rate value. In the current example, **FlwR** is set to a low value of 1e-3.
- **FlwT** — Positive or negative value that, when summed to the ambient temperature, equals the coolant inlet temperature. A value of +15 for the **FlwT** input and 273.15 K at the **Amb** port makes the coolant inlet temperature equal to 273.15 + 15 = 288.15K. A value of -15 for the **FlwT** input and 273.15 K at the **Amb** makes the coolant inlet temperature equal to 273.15 - 15 = 258.15 K. In this example, **FlwT** is set to 0.

#### Simulation Results from Simscape Logging

The switch in the circuit is closed at 30s time in the Switch operation logic subsystem. The circuit is completed and short circuits the system through a resistance of 0.1m-Ohm. As a high current passes

through all the cells in the module, the cell temperature rises and quickly attains the trigger temperature for thermal runaway and gas venting. The switch is opened to disconnect the electrical circuit as the cell is now dead.



# Input Admittance Response of RLC Ladder Network with Mutual Coupling Between Multiple Coils

This example shows how to model a four-section ladder network that comprises RLC components with mutual coupling between multiple coils. You can use this ladder network representation to model the disc winding of a transformer. The number of sections of a ladder network depends on the number of discs in the winding. Each section can model two or three discs in the winding.

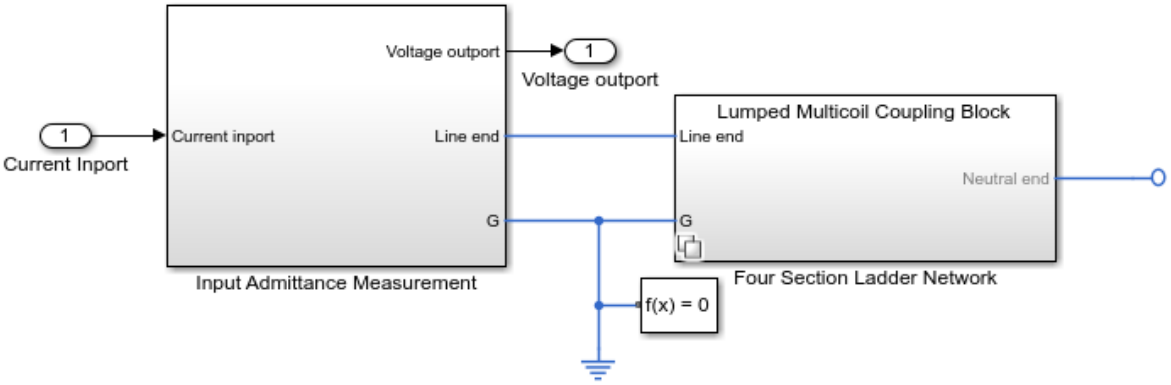
The winding resistance and self-inductance are represented as Rs and Ls for each section. The series capacitance, Cs, and the ground capacitance, Cg, model the inter-disc capacitance and the capacitance between the winding and ground, respectively.

In this example, each section is mutually coupled with three other sections. You can model the inductance of the four sections and incorporate the mutual inductance by using either a Lumped Inductance block or Distributed Inductance blocks.

To calculate the mutual inductance, each inductor must connect to the input port of the other three inductors. Because the Lumped Inductance block models the entire inductance of all sections, use this block to minimize the clustering of the branches.

The Distributed Inductance blocks model the four inductors. These blocks receive the derivative of the four branch currents as physical signals, and output the derivative of the branch current of the section.

### Open the Model



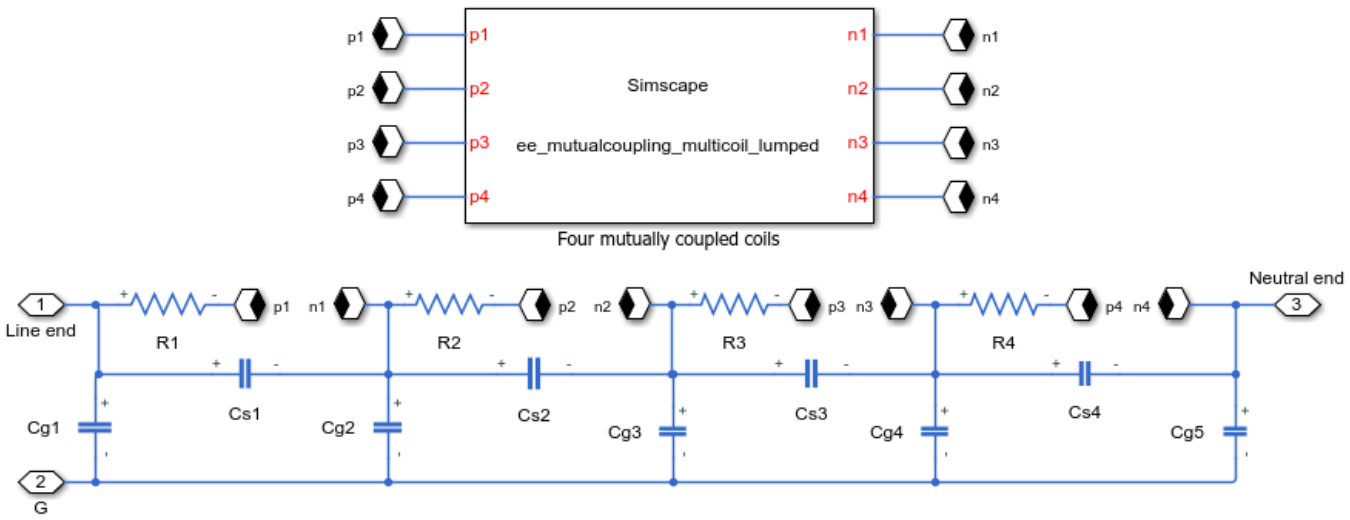

**Input Admittance Response of RLC Ladder Network with Mutual Coupling Between Multiple Coils**

1. Input model parameters (see code)
2. Plot Input admittance response (see code)
3. Configure implementation of mutual inductance for ladder network: Lumped Multicoil Coupling Block, Distributed Multicoil Coupling Block
4. Explore simulation results using Simscape Results Explorer
5. Learn more about this example

Copyright 2008-2022 The MathWorks, Inc.

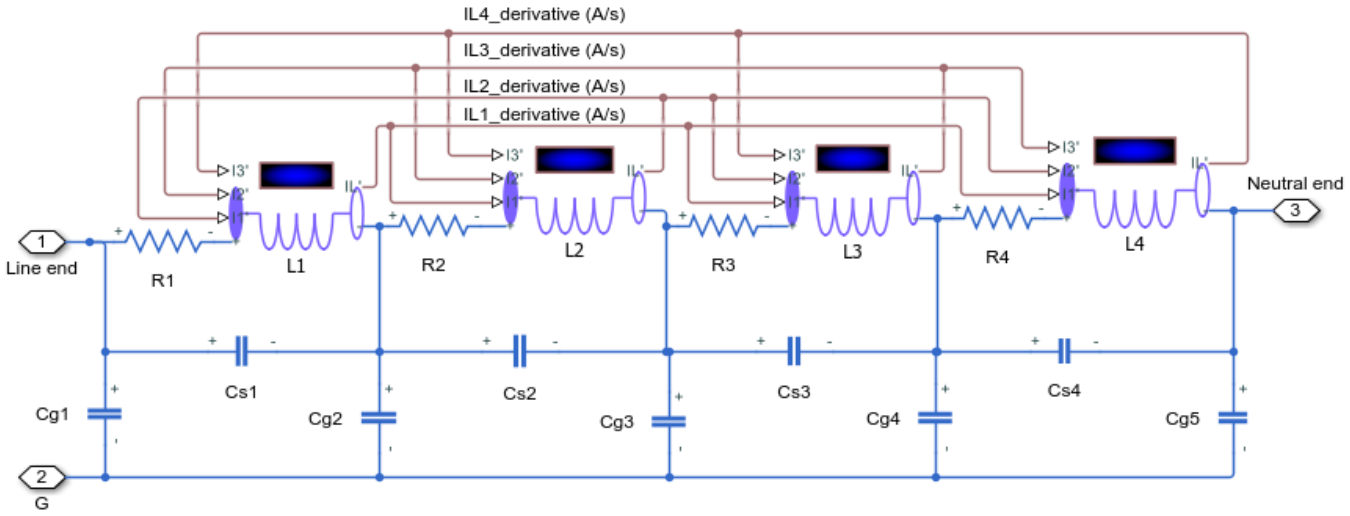
### Model the Ladder Network with a Lumped Inductance Block

The Lumped Inductance block models the self inductance, Ls, of the four sections lumped with the mutual inductances between each section. The Lumped Inductance block comprises the phase and neutral connections, which are labeled p and n in the diagram.



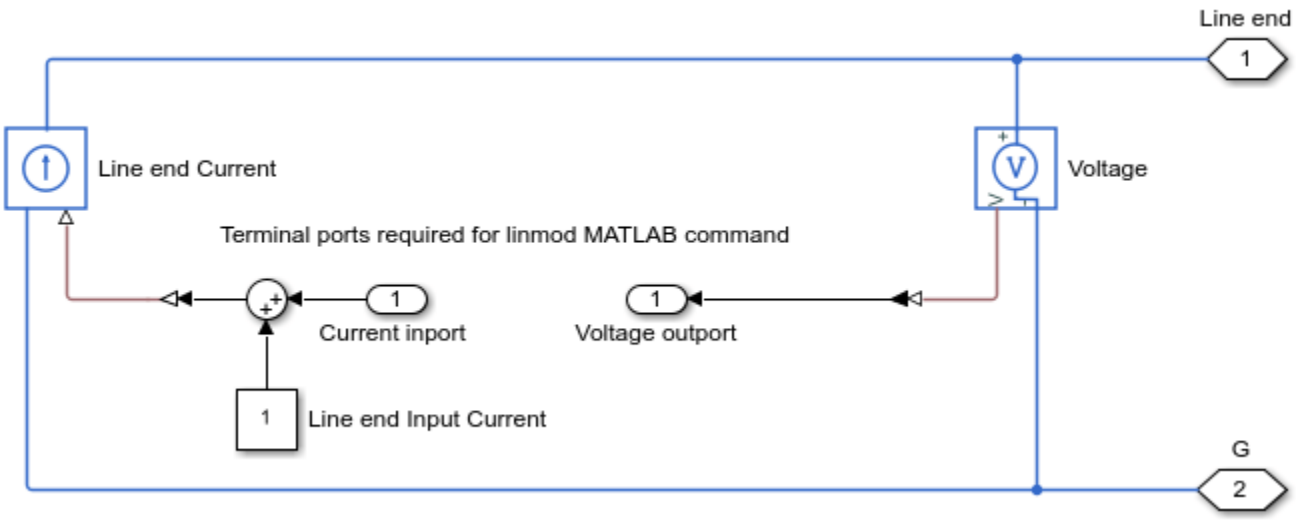

### Model the Ladder Network with Multiple Distributed Inductance Blocks

Each section comprises an Inductor block that inputs the derivative of branch currents from the three other sections as physical signals and outputs the derivative of its branch current as a physical signal.



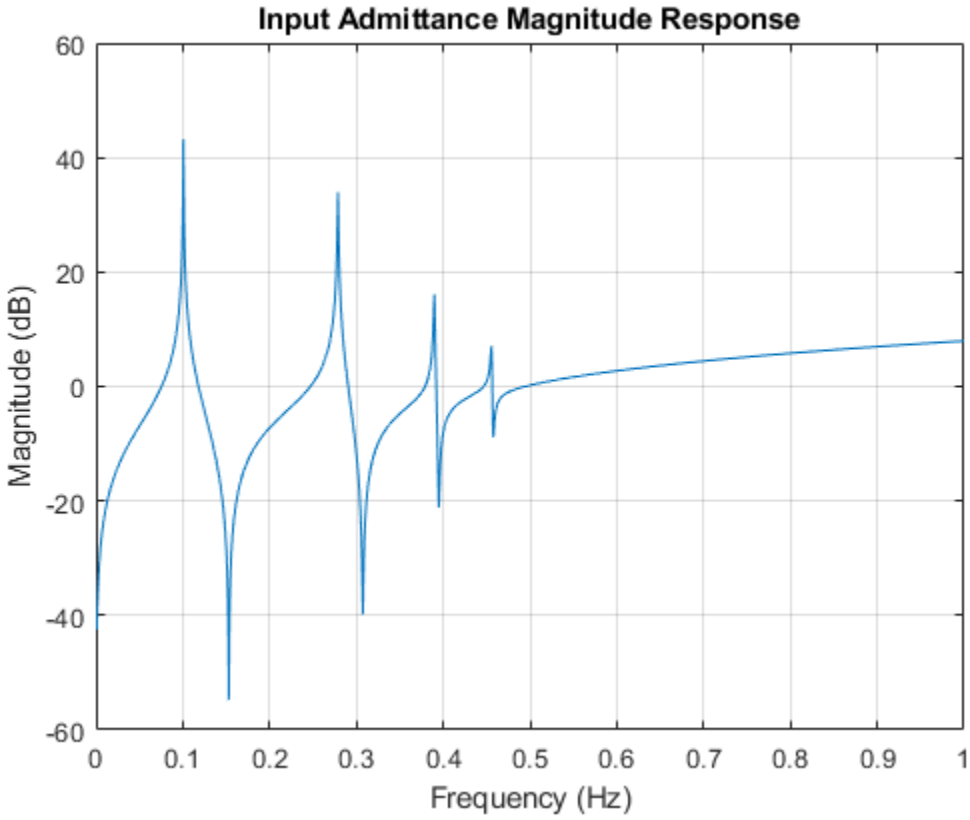

### Set Up the Input Admittance Measurement

This example estimates the input admittance response of the ladder network through a voltage sensor and a controlled current source. The Input Admittance Measurement Setup subsystem calculates the system transfer function. The admittance response is then computed as the reciprocal of the estimated transfer function.




### Simulation Results

The plot below shows the input admittance magnitude response for a four section ladder network.



# Tolerance Study Using Monte Carlo Simulations in Resonant LLC DC-DC Converter

This example shows how to use Simscape™ Electrical™ to perform a Monte Carlo analysis to optimize the design of an LLC resonant DC-DC converter when some of its components have tolerances.

### Motivation

When designing an electrical system, some components might possess certain tolerance on some key parameters (for example the inductance of an inductor). Looser tolerance components usually have a lower cost, so it is cost-effective to choose them.

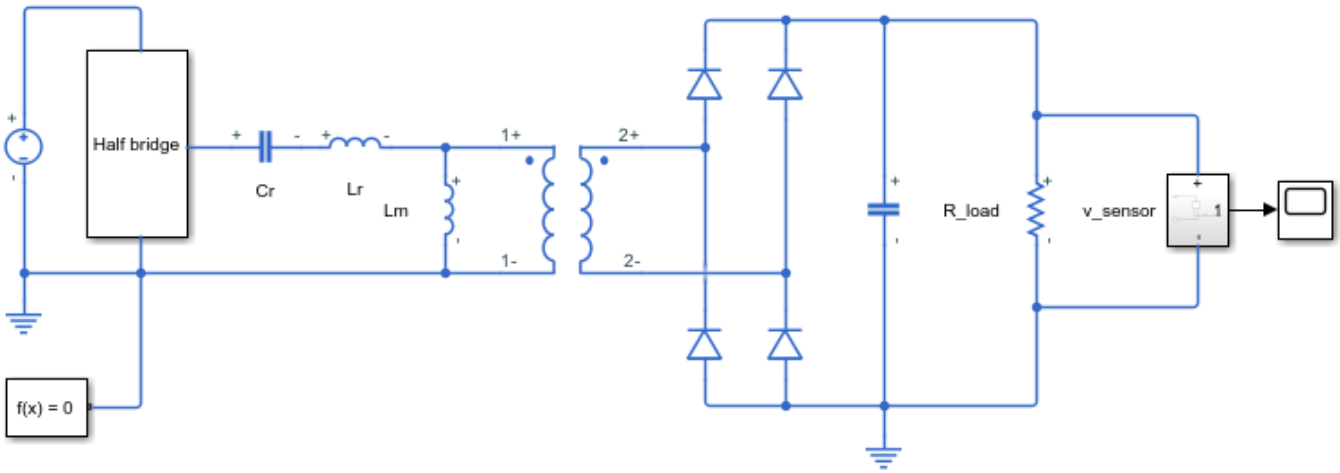
However, component tolerance impacts system performance, introducing a variability on some key specifications. This example shows you how to choose design options that minimize the variability of the entire system.

In this example, an LLC resonant DC-DC converter comprises capacitors and inductors with a given tolerance, and there are several choices for design parameters that provide the desired gain output. The goal is to find which of these design choices have the minimum output gain tolerance, or global system variability.

### Monte Carlo Method

The components' statistical variability influences the global system statistical variability. Monte Carlo method consists of inferring an estimation of this influence by statistical repetition. A given known distribution generates the component parameter values and then a simulation is performed to obtain system level results. Simulations are repeated many times. If the number of simulations is large enough, the system output distribution is reasonably well inferred.

### Model




**Tolerance Study Using Monte Carlo Simulations in Resonant LLC DC-DC Converter**

1. Run Monte Carlo study over inductance ratio (see code)
2. Run Monte Carlo study over inductance ratio and quality factor (computationally intensive) (see code)
3. Use Parallel Computing Toolbox(TM) to scale this analysis to a large number of simulations
4. Explore simulation results using Simscape Results Explorer
5. Learn more about this example

Copyright 2019-2022 The MathWorks, Inc.

In this particular design, the control system has been chosen to be very simple: a fixed switching frequency in open-loop.

### Theoretical Approximation for Output Voltage Gain

For this converter, a first harmonic approximation to compute output voltage gain from design parameters is:

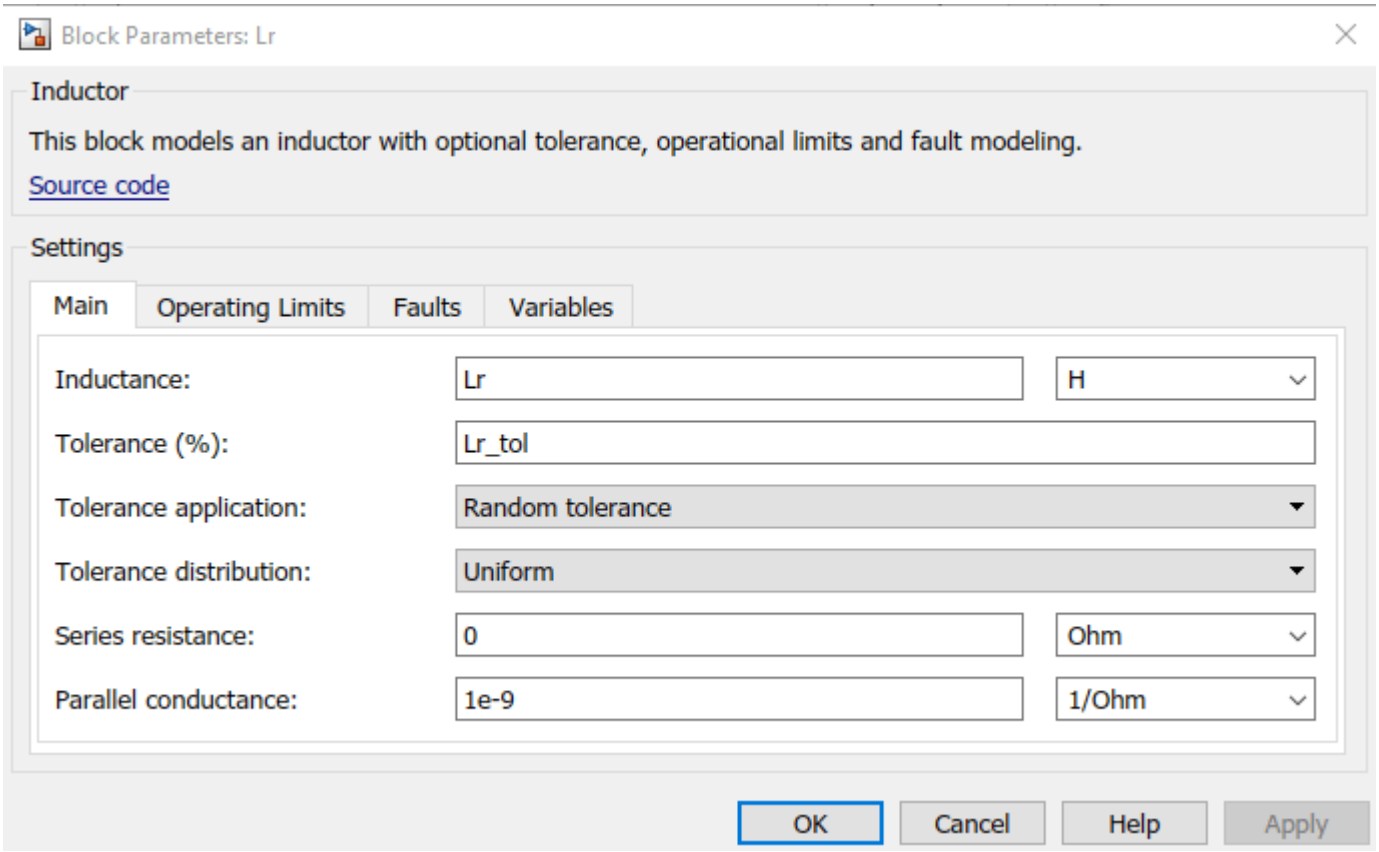
$$M = \frac{1}{2\sqrt{\left(1 + L_N - \frac{L_N}{f_N^2}\right)^2 + Q^2\left(f_N - \frac{1}{f_N}\right)^2}}$$

in which $L_N$ and $Q$ are the inductance ratio and quality factor, which directly depend on the inductances and capacitance of the components of interest in this study.

We will use this formula to compute an approximation to the expected output gain when components are perfect (do not have any tolerance).

### Specifying Component Tolerance

You can specify the component tolerances in the block mask. For example, in the inductor, you can apply a tolerance value with a uniform distribution on a certain range of inductances.

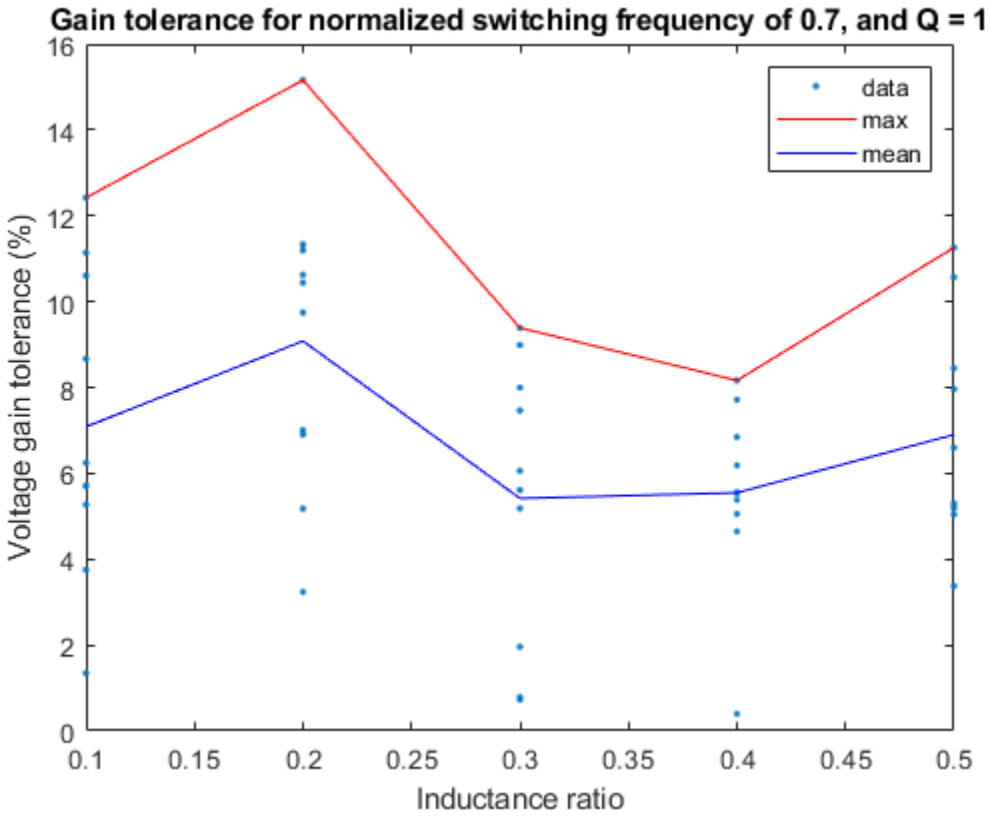



**System Tolerance Study (Over Ln Only)**

After you apply the tolerances (20% for the inductors and 7% for the capacitor in this example), you run multiple simulations during which the indunctances and capacitance are randomly generated at t=0 with a uniform distribution. The output gain measured on the load varies each time, as the system is different during each simulation due to the tolerances. For a fixed design choice (fixed $L_N$ and fixed $Q$), multiple repeated Monte Carlo simulations infer a statistical distribution of the output voltage.

In this section, to find out which configuration gives a lower output voltage gain variability, you fix a value of $Q$ and change the value of $L_N$ over a range you want to study.

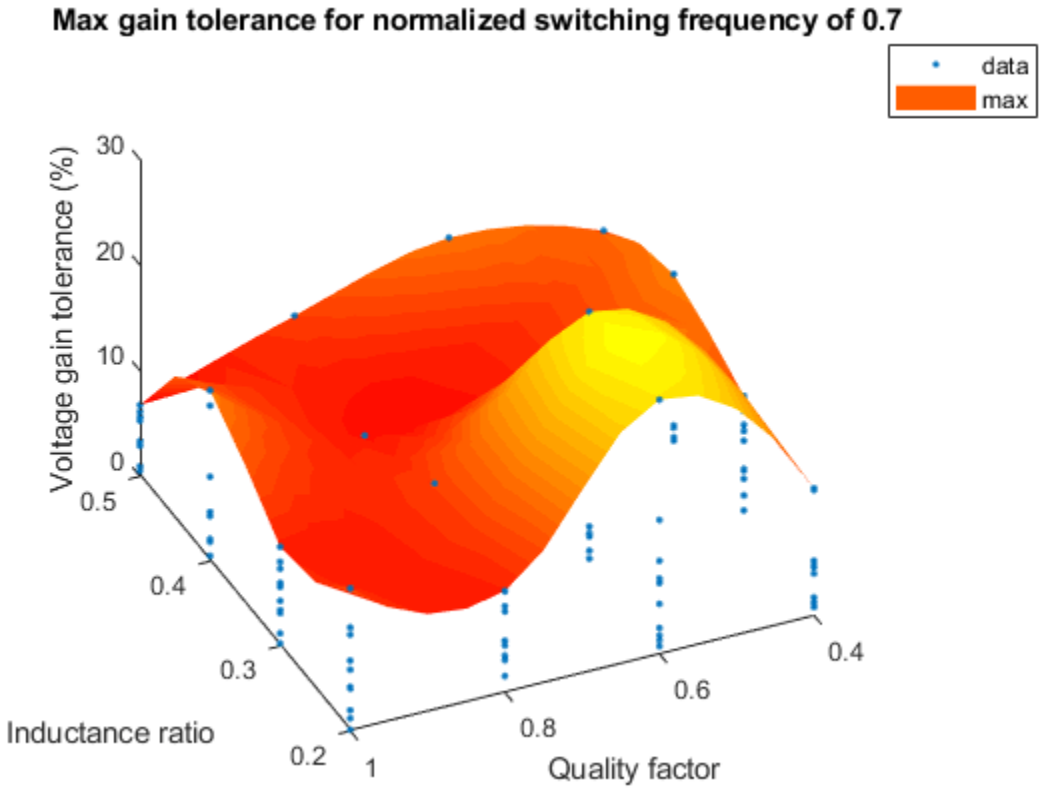
```
This script will run 50 simulations.
```

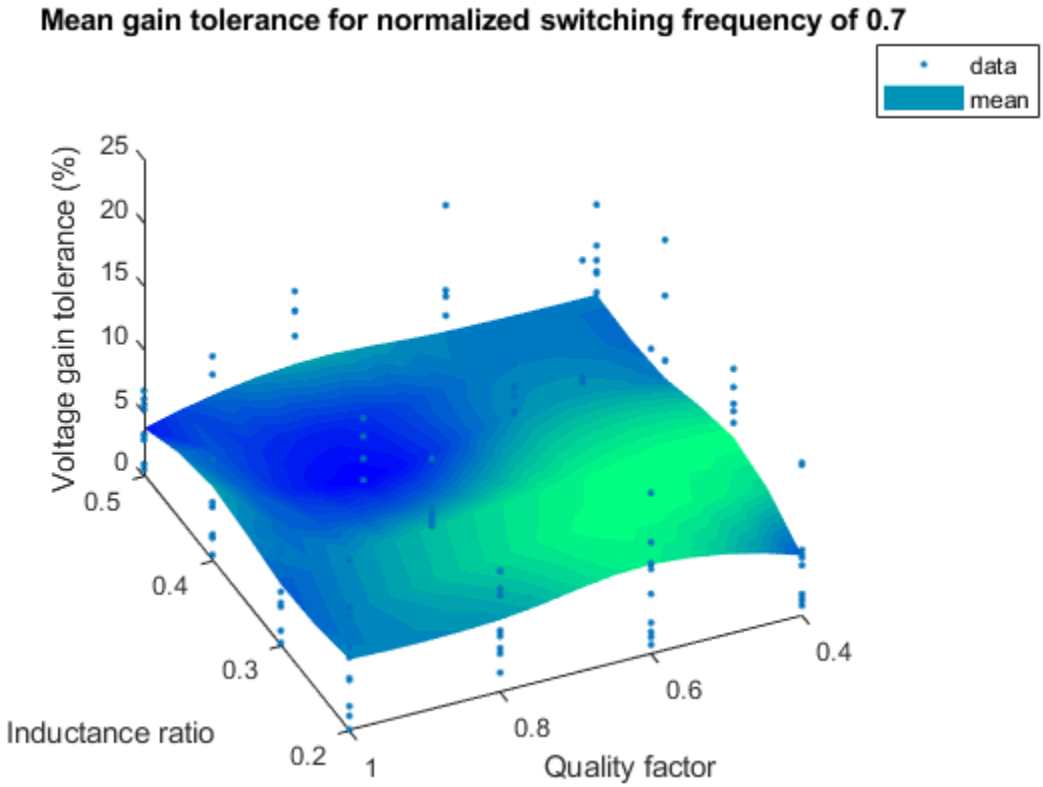



**System Tolerance Study (Over Ln and Quality Factor)**

To study the system tolerance over $L_N$ and the quality factor, change the value of $Q$ as well. A grid of values for L_N and Q is then selected and, for every configuration, the model estimates the output gain distribution through repetition.

```
This script will run 160 simulations.
```


Max gain tolerance for normalized switching frequency of 0.7
data
max
Voltage gain tolerance (%)
30
20
10
0
0.5
0.4
0.3
0.2
Inductance ratio
1
0.8
0.6
0.4
Quality factor

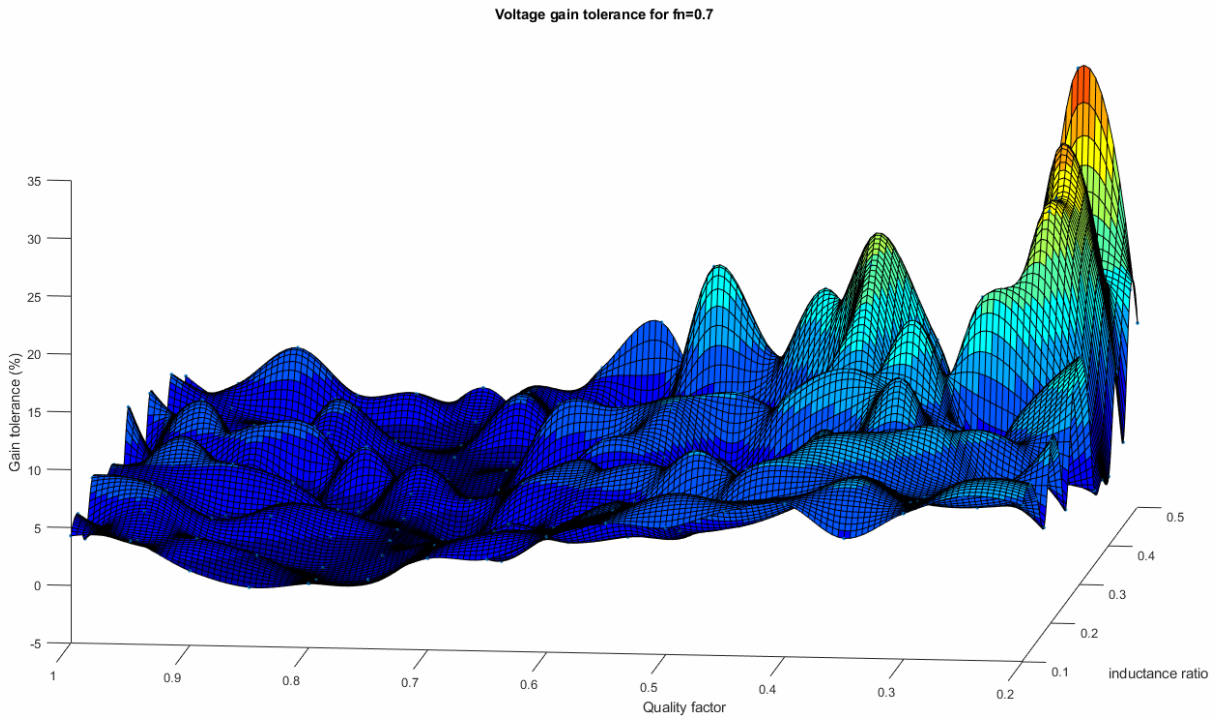



### Advantages of Parallel Computing

Simulating hundreds or thousands of times is computationally expensive and time consuming. To reduce computation time, you can run multiple simulations in parallel. This is especially advantageous for independent Monte Carlo simulations.

For this workflow, Parallel Computing Toolbox™ is used to speed up the process and scale it up to a higher number of simulations.

Up until this point, this example worked on a grid of 4x4 configurations with 10 simulations each. This figure shows an animation of the same workflow scaled up to a grid of 17x17 configurations with 100 simulations each.



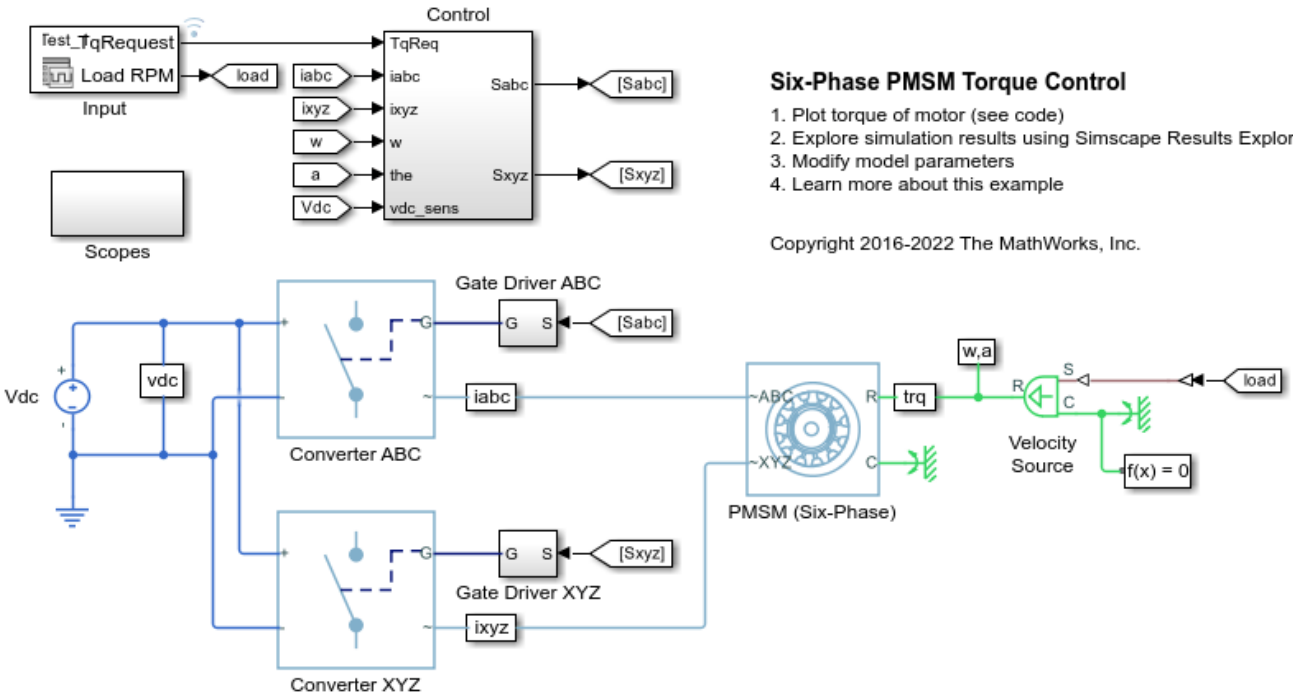

The surface plot shows the worst case scenario with the maximum tolerance found for each configuration. The surface plot shapes to a more reliable representation of the distribution of the output gain tolerances as the number of simulation increases. The configuration with $Q = 0.8$ and $L_N = 0.3$ has the lowest possible output gain tolerance equal to around 9%.

To reproduce these results, use the `ee_converter_dcdc_llc_tolerance_montecarlo_parallel` script. This script uses functions and methods from the Parallel Computing Toolbox™ such as `parsim`, `batchsim`, `parcluster`, `Simulink.SimulationInput.setModelParameter`, `Simulink.SimulationInput.setVariable`, and other useful tools.

# Six-Phase PMSM Torque Control

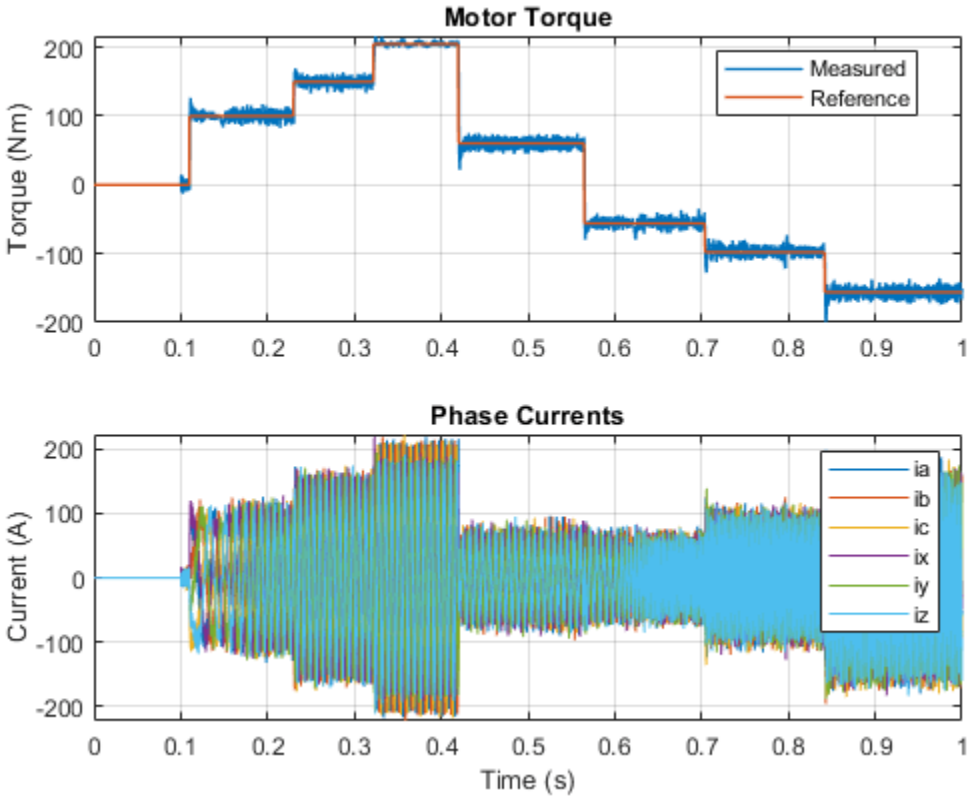
This example shows how to control the torque in an electrical-traction drive based on a six-phase permanent magnet synchronous machine (PMSM). A DC voltage source feeds the PMSM through two controlled three-phase converters. The PMSM operates in both motoring and generating modes according to the load. An ideal angular velocity source provides the load. The Control subsystem uses an open-loop approach to control the torque and a closed-loop approach to control the current. At each sample instant, the torque request is converted to a relevant q-axis current reference. The current control is PI-based. The simulation uses several torque steps in both motor and generator modes. The Scopes subsystem contains scopes that allow you to see the simulation results.

### Model




### Simulation Results from Simscape Logging

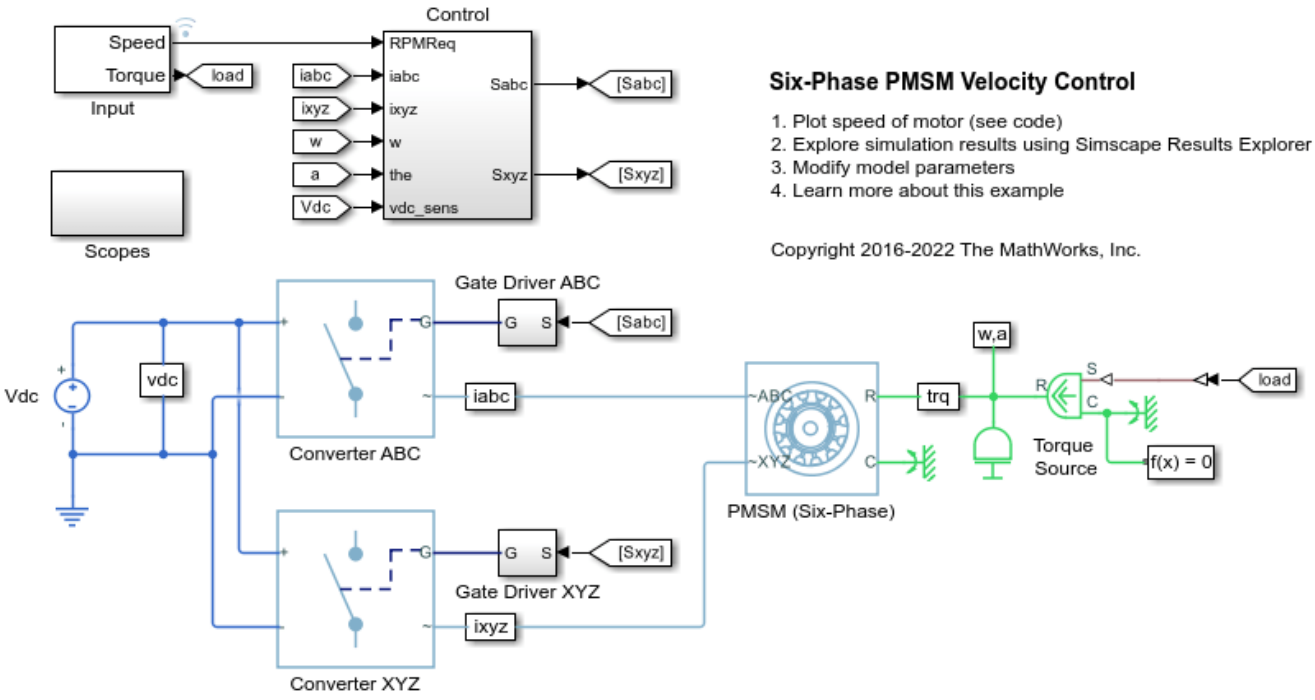
The plot below shows the requested and measured torque for the test, as well as the phase currents in the electric drive.


Motor Torque
Torque (Nm)
200
100
0
-100
-200
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Measured
Reference
Phase Currents
Current (A)
200
100
0
-100
-200
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Time (s)
ia
ib
ic
ix
iy
iz

# Six-Phase PMSM Velocity Control

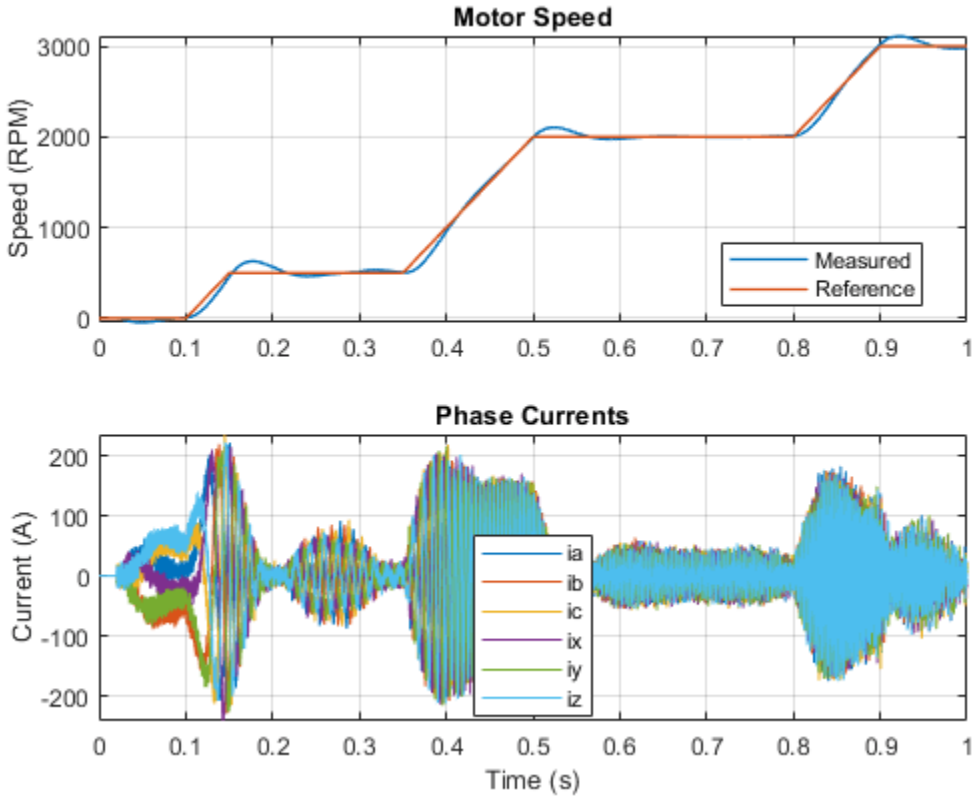
This example shows how to control the rotor angular velocity in an electrical-traction drive based on a six-phase permanent magnet synchronous machine (PMSM). A DC voltage source feeds the PMSM through two controlled three-phase converters. The PMSM operates in both motoring and generating modes according to the load. An ideal torque source provides the load. The Scopes subsystem contains scopes that allow you to see the simulation results. The Control subsystem includes a PI-based cascade control structure that has an outer angular-velocity-control loop and four inner current-control loops. During the one second simulation, the angular velocity demand is 0 rpm, 500 rpm, 2000 rpm, and then 3000 rpm.

### Model




### Simulation Results from Simscape Logging

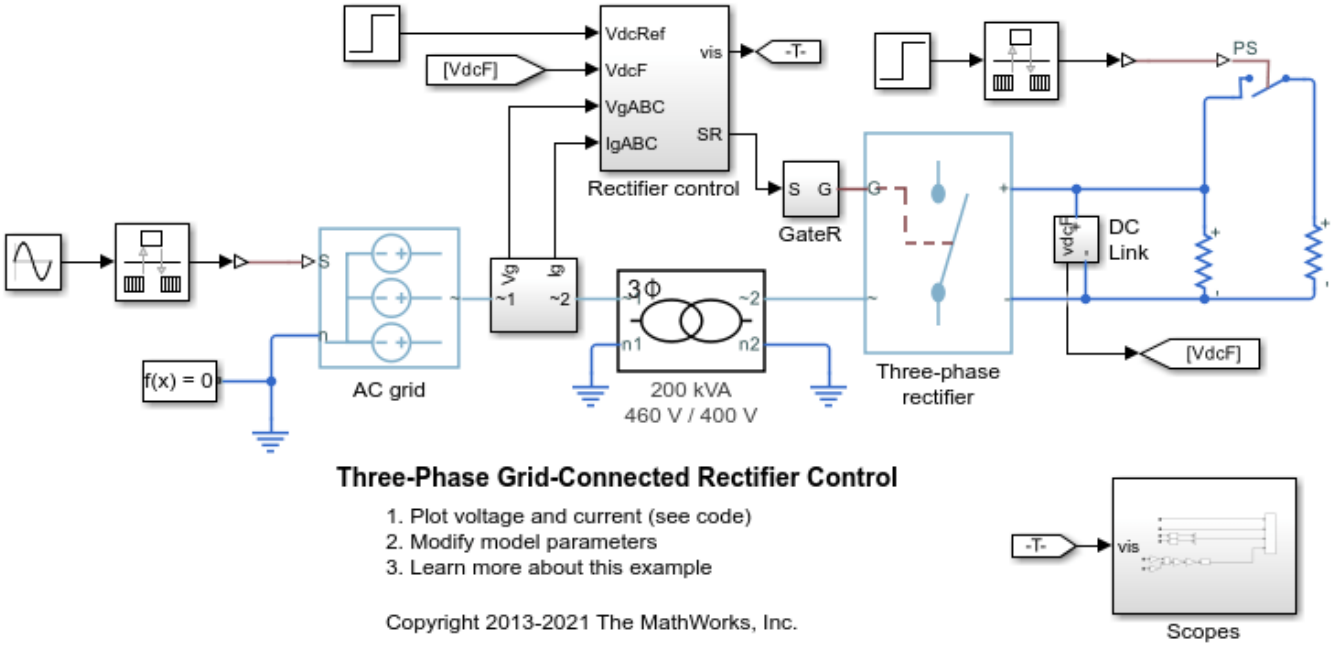
The plot below shows the requested and measured speed for the test, as well as the phase currents in the electric drive.


Motor Speed
Speed (RPM)
3000
2000
1000
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Measured
Reference
Phase Currents
Current (A)
200
100
0
-100
-200
ia
ib
ic
ix
iy
iz
Time (s)

# Three-Phase Grid-Connected Rectifier Control

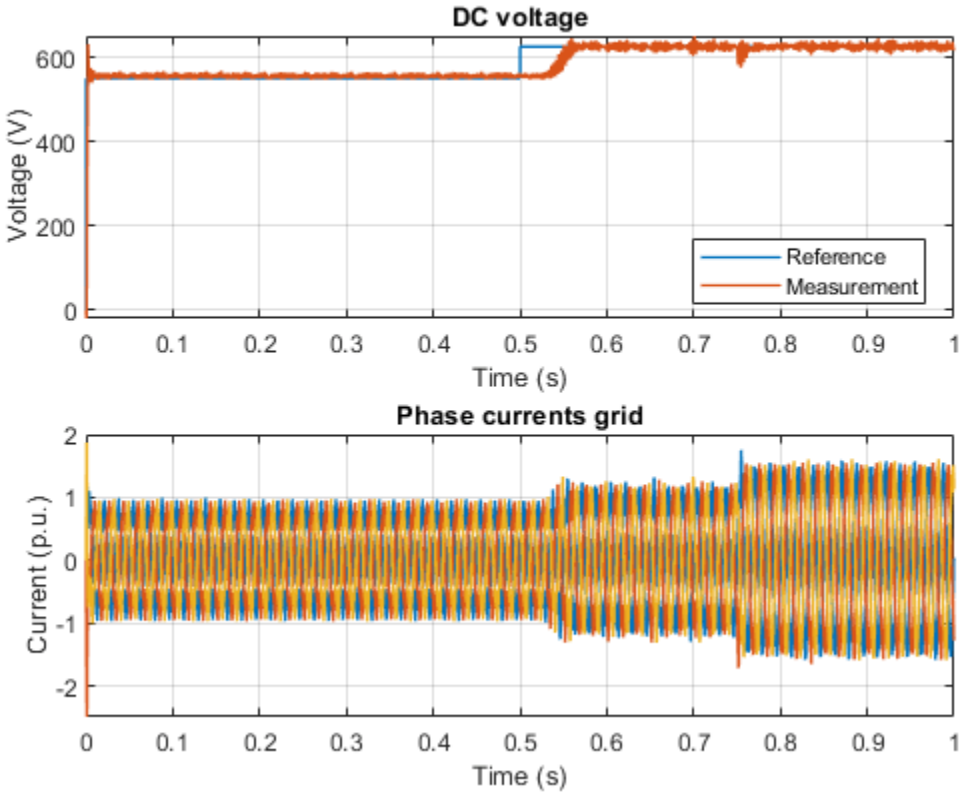
This example shows how to control the DC-link voltage using a grid-connected rectifier. The Rectifier control subsystem uses a PI-based cascade control structure. The Scopes subsystem contains scopes that allow you to see the simulation results. If you have a license for HDL Coder™, you can generate VHDL code for an FPGA using the Simscape™ HDL Workflow Advisor.

### Model




### Simulation Results from Simscape Logging

The plot below shows the DC voltage and AC phase currents.



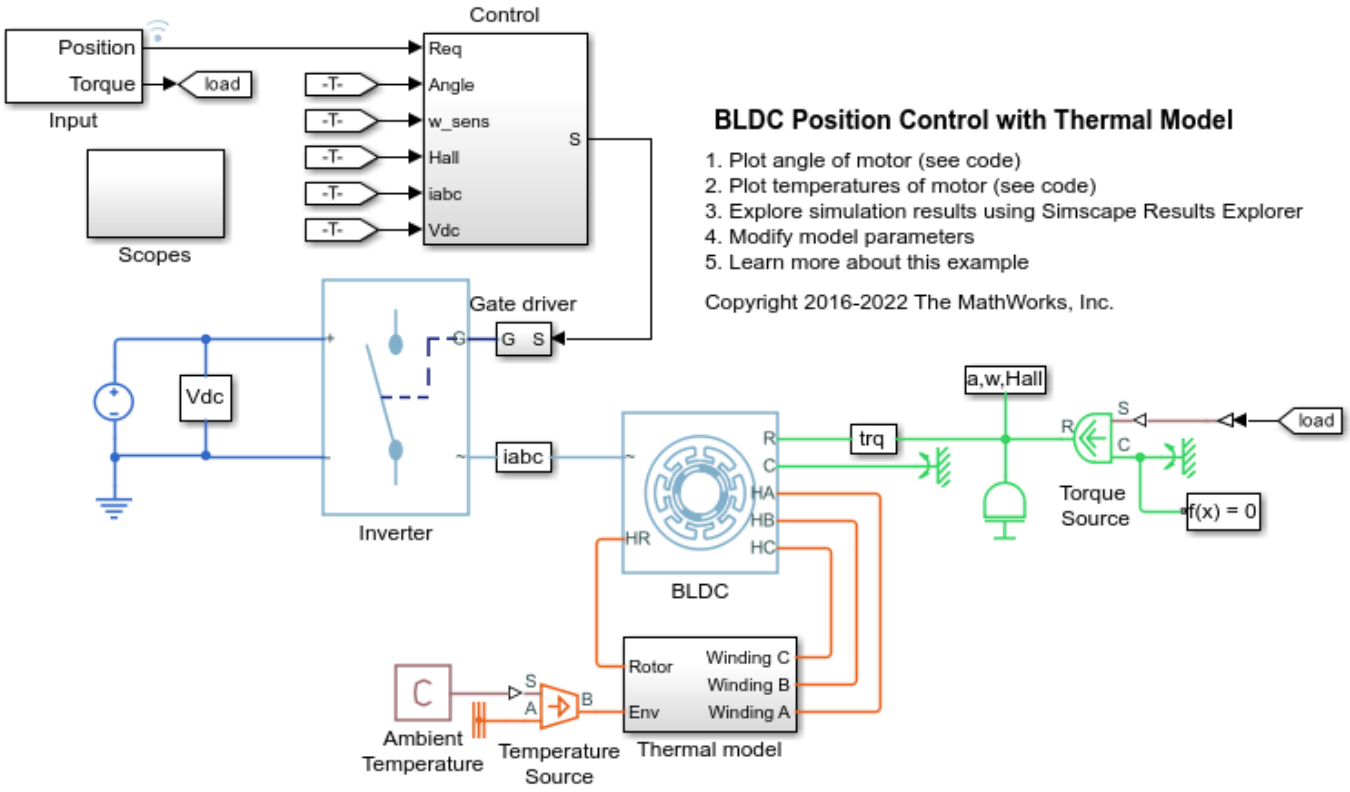

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 10 microseconds.

# BLDC Position Control with Thermal Model

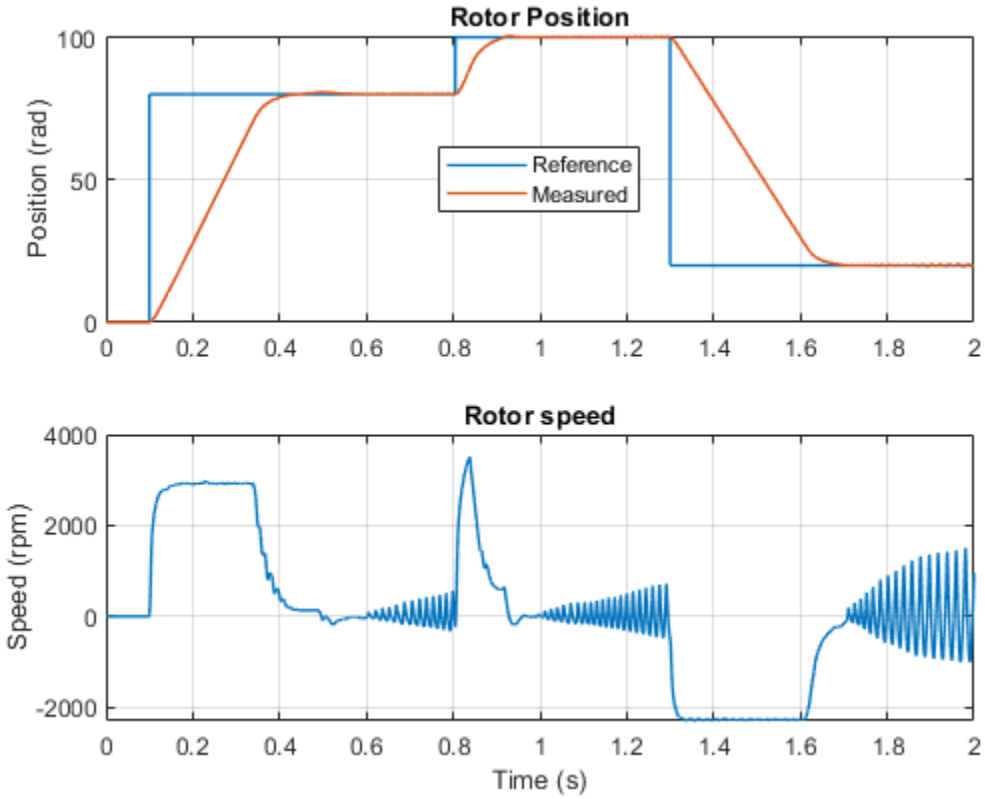
This example shows how to control the rotor angle in a BLDC based electrical drive. The BLDC includes a thermal model and empirical iron losses. An ideal torque source provides the load. The Control subsystem uses a PI-based cascade control structure with three control loops: an outer position control loop, a speed control loop, and an inner current control loop. The BLDC is fed by a controlled three-phase inverter. The gate signals for the inverter are obtained from hall signals. The simulation uses step references. The initial temperature of the stator windings and rotor is set to 25 degrees Celsius. Ambient temperature is 27 degrees Celsius. The Scopes subsystem contains scopes that allow you to see the simulation results.

### Model

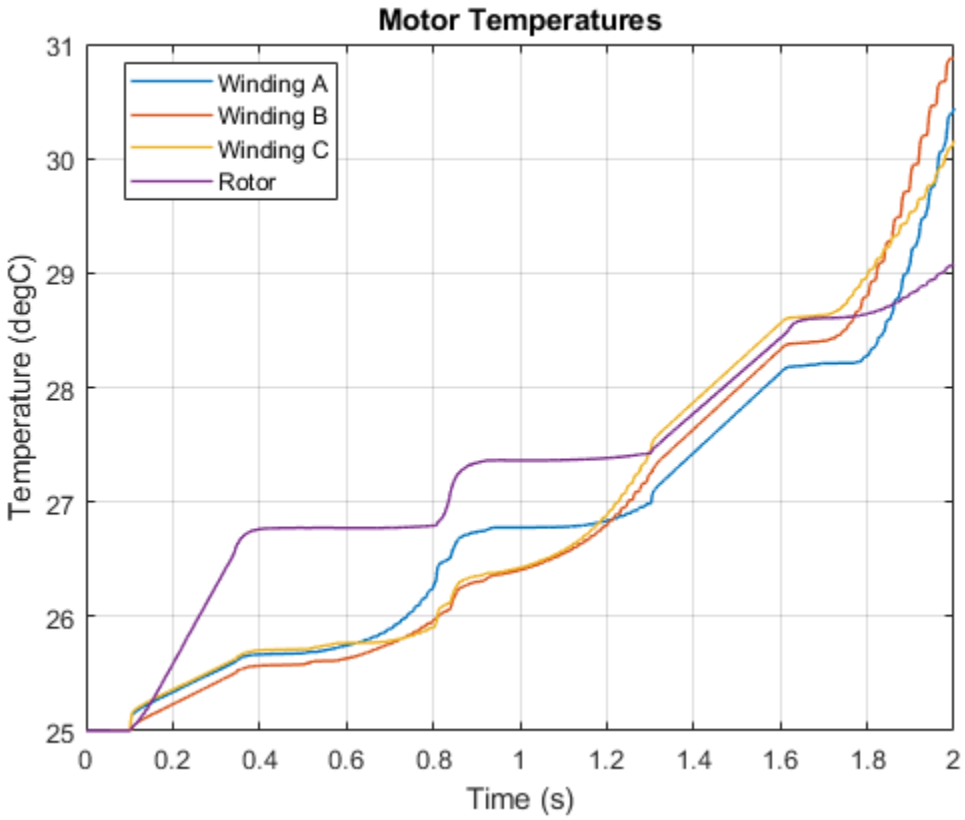



### Simulation Results from Simscape Logging

The plot below shows the requested and measured angle for the test and the rotor speed in the electric drive.



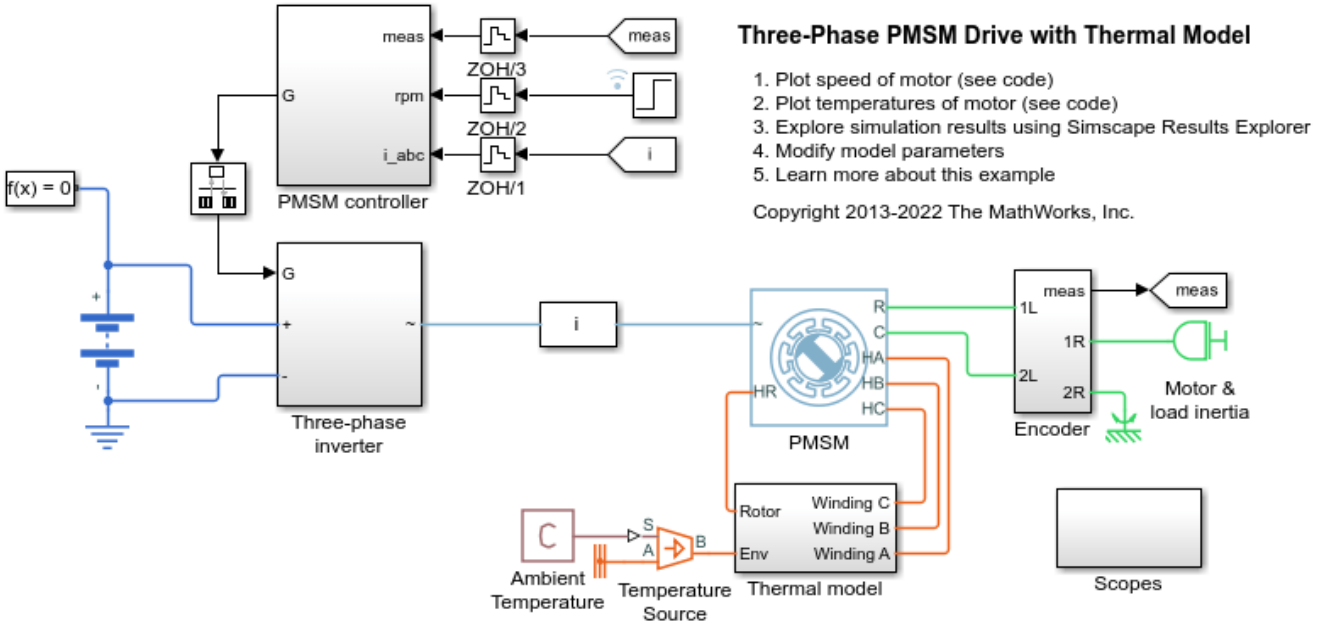

The plot below shows the winding and rotor temperatures of the motor.


Motor Temperatures
Winding A
Winding B
Winding C
Rotor
Temperature (degC)
Time (s)
31
30
29
28
27
26
25
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2

# Three-Phase PMSM Drive with Thermal Model

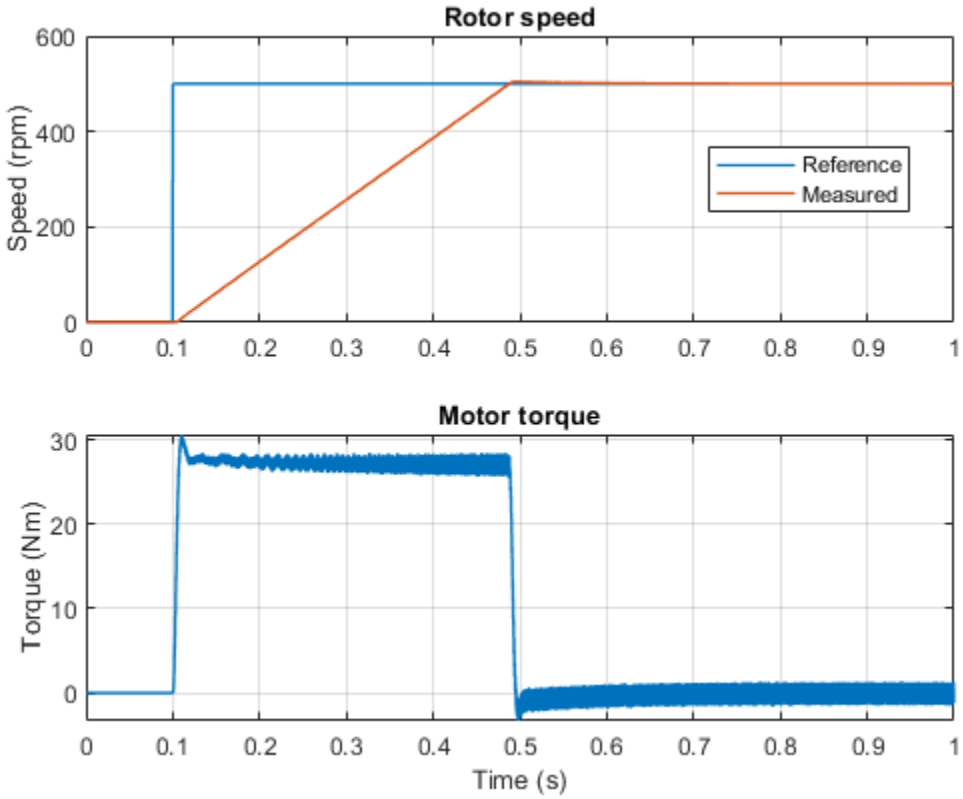
This example shows a Permanent Magnet Synchronous Machine (PMSM) and an inverter sized for use in a typical hybrid vehicle. The PMSM includes a thermal model and empirical iron losses. The inverter is connected directly to the vehicle battery, but you can also implement a DC-DC converter stage in between. You can use this model to design the PMSM controller, by selecting the architecture and the gains to achieve the desired performance. The initial temperature of the stator windings and rotor is set to 25 degrees Celsius. Ambient temperature is 27 degrees Celsius. The Scopes subsystem contains scopes that allow you to see the simulation results.

### Model

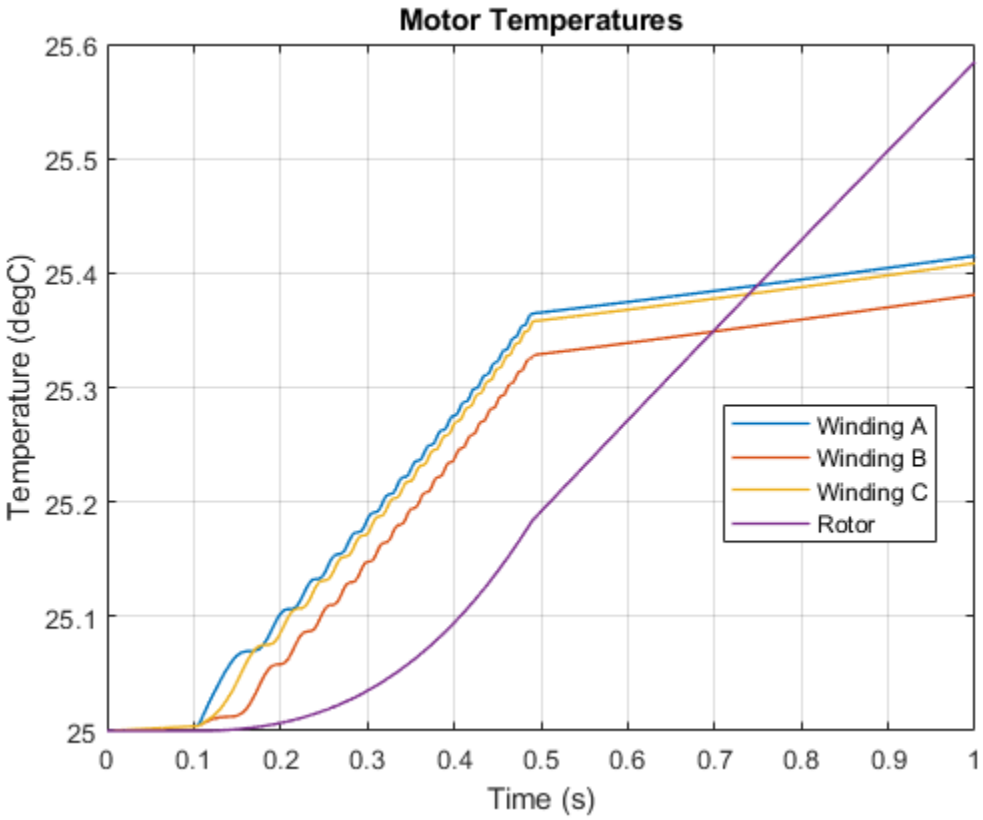



### Simulation Results from Simscape Logging

The plot below shows the requested and measured rotor speed for the test and the torque in the electric drive.



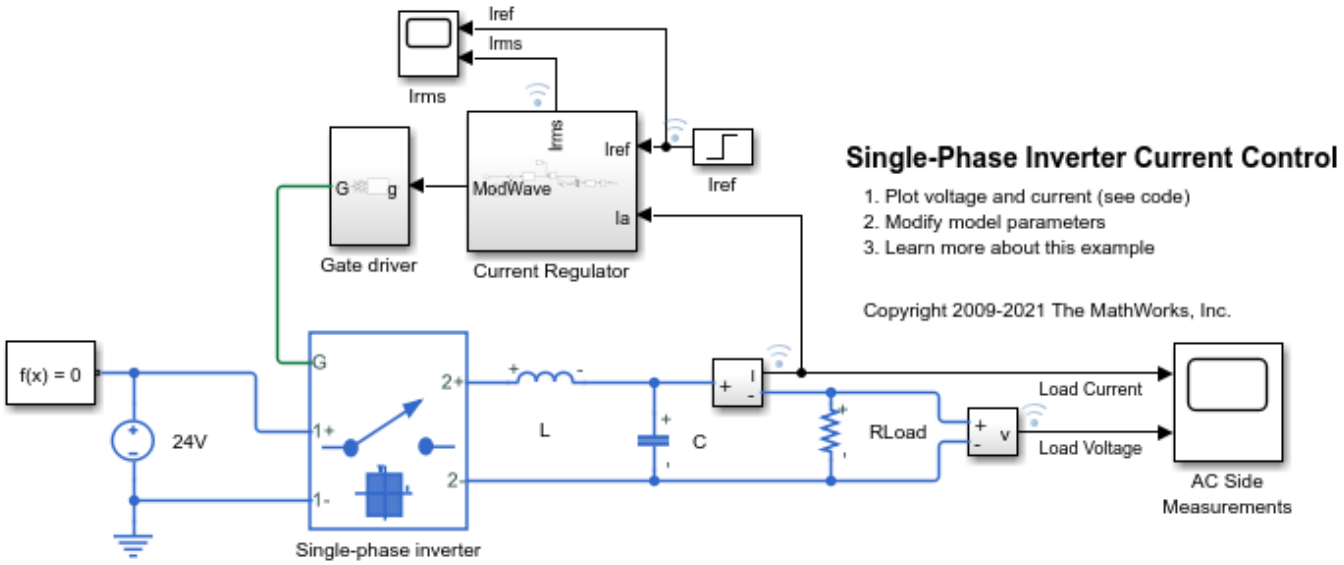

The plot below shows the winding and rotor temperatures of the motor.


Motor Temperatures
Temperature (degC)
Time (s)
25.6
25.5
25.4
25.3
25.2
25.1
25
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Winding A
Winding B
Winding C
Rotor

# Single-Phase Inverter Current Control

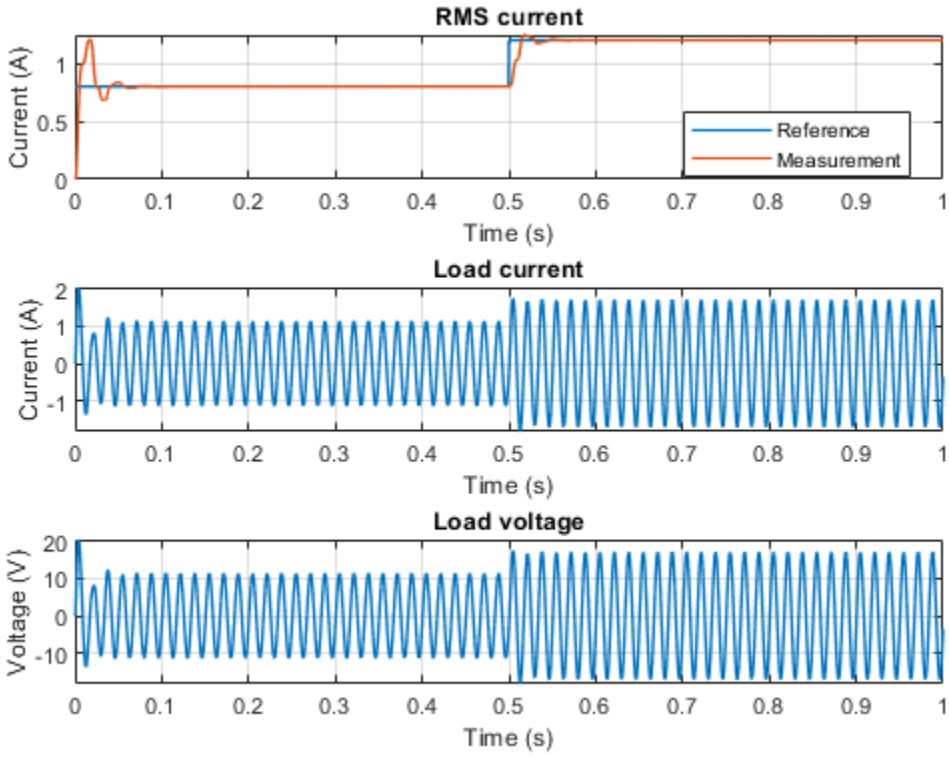
This example shows how to control the current in a single-phase inverter system. The single-phase inverter uses averaged switches fed by modulation waveforms. This example is suitable for real-time evaluation on a dedicated real-time emulator.

### Model




### Simulation Results from Simscape Logging

The plot below shows the load current and voltage.



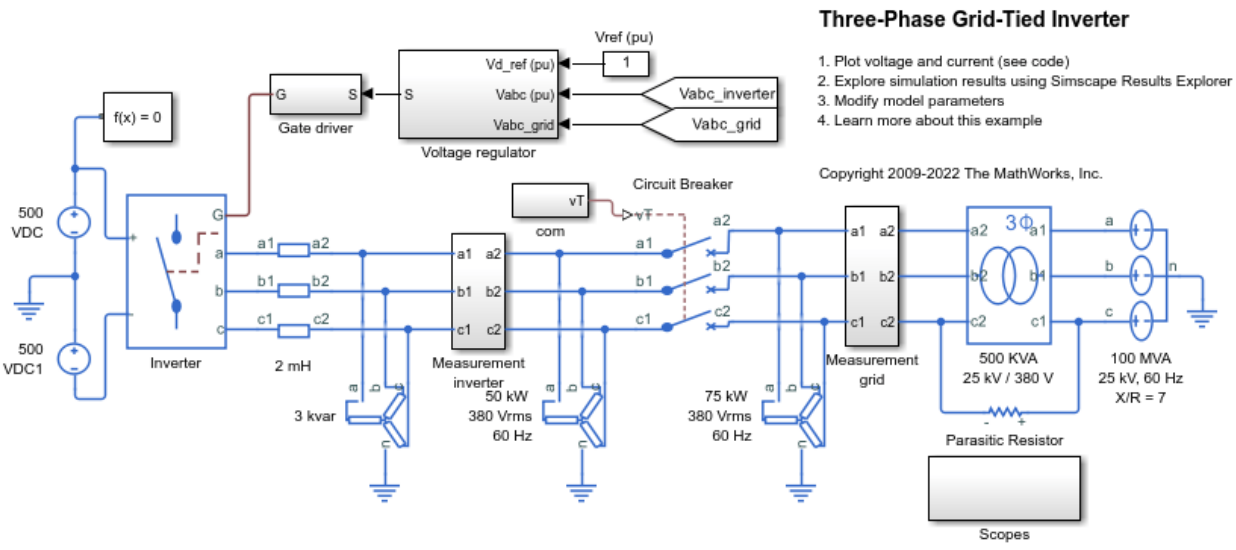

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 20 microseconds.

# Three-Phase Grid-Tied Inverter

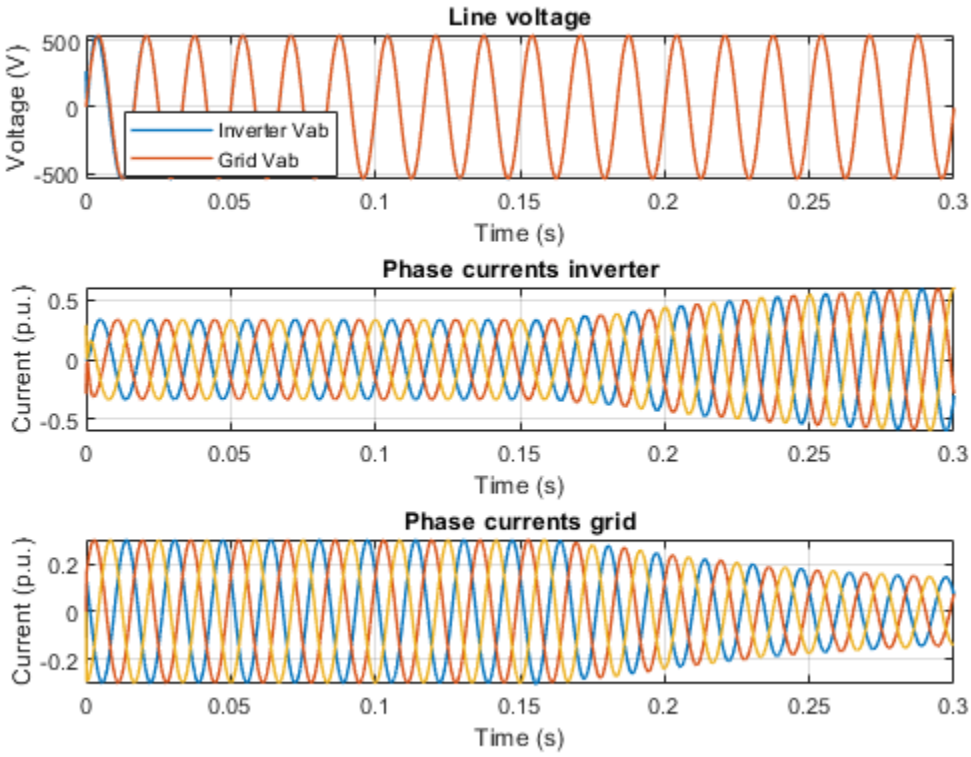
This example shows how to control the voltage in a grid-tied inverter system. The Voltage regulator subsystem implements the PI-based control strategy. The three-phase inverter is connected to the grid via a Circuit Breaker. The Circuit Breaker is open at the beginning of the simulation to allow synchronization. At time 0.15 seconds, the Circuit breaker closes, and the inverter is connected to the grid. The Scopes subsystem contains scopes that allow you to see the simulation results. The inverter is implemented using IGBTs. To speed up simulation, or for real-time deployment, the IGBTs can be replaced with Averaged Switches. In this way the gate signals can be averaged over a specified period or replaced with modulation waveforms.

### Model




### Simulation Results from Simscape Logging

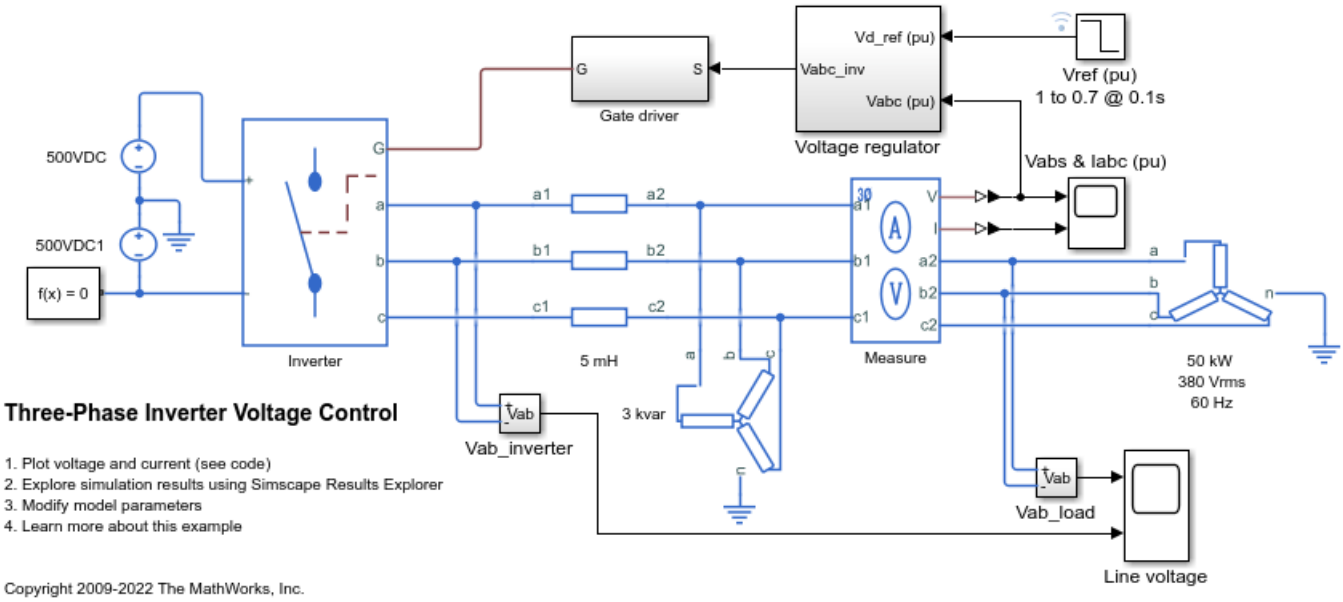
The plot below shows the line voltage and phase currents in the inverter and grid.


Line voltage
Voltage (V)
500
0
-500
Inverter Vab
Grid Vab
0
0.05
0.1
0.15
0.2
0.25
0.3
Time (s)
Phase currents inverter
Current (p.u.)
0.5
0
-0.5
0
0.05
0.1
0.15
0.2
0.25
0.3
Time (s)
Phase currents grid
Current (p.u.)
0.2
0
-0.2
0
0.05
0.1
0.15
0.2
0.25
0.3
Time (s)

# Three-Phase Inverter Voltage Control

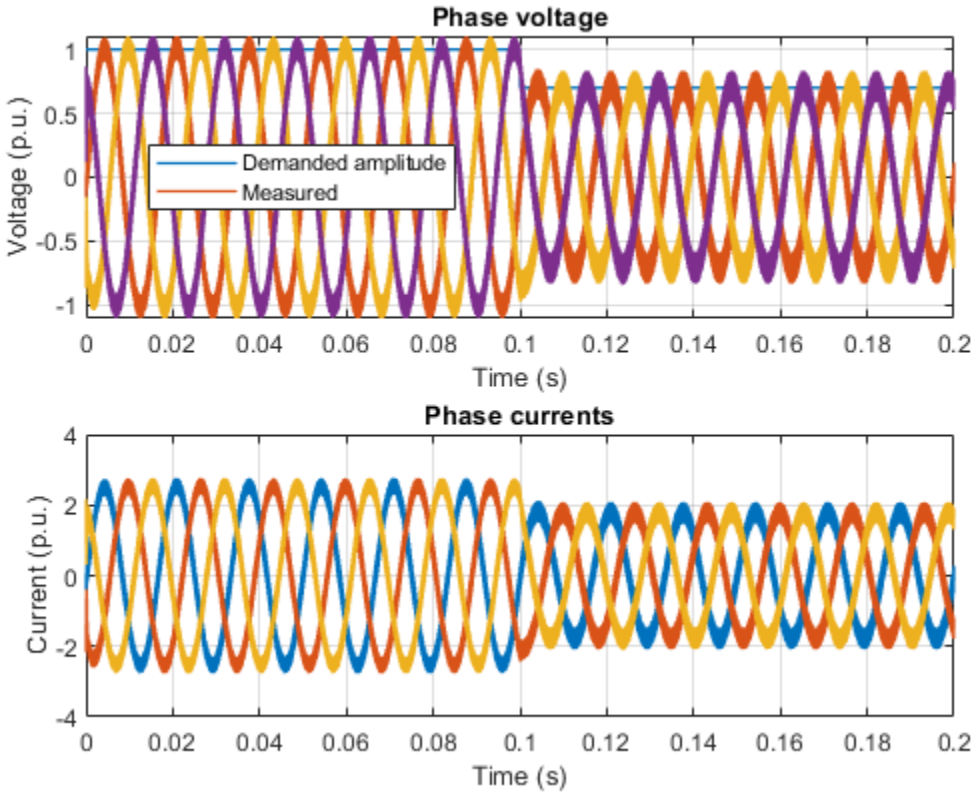
This example shows how to control the voltage in a three-phase inverter system. The inverter is implemented using IGBTs. To speed up simulation, or for real-time deployment, the IGBTs can be replaced with Averaged Switches. In this way the gate signals can be averaged over a specified period or replaced with modulation waveforms.

### Model




### Simulation Results from Simscape Logging

The plot below shows the phase voltages and currents.


Phase voltage
Voltage (p.u.)
Demanded amplitude
Measured
Time (s)
Phase currents
Current (p.u.)
Time (s)

# SPICE Conversion of a CMOS Voltage Comparator

This example shows a typical implementation of a CMOS voltage comparator and how you can convert a SPICE subcircuit to a Simscape™ component using the `subcircuit2ssc` function. You can use CMOS voltage comparators in analog to digital converter (ADC) or relaxation oscillator circuits.

In order to obtain the small-signal frequency response of the system, this example includes input *u*, output *y*, and includes a Solver Configuration block with the parameter **Start simulation from steady state** selected. You can use the `linmod` function to linearize this model.

### Open the CMOS Voltage Comparator Subcircuit

In this example, you will convert this voltage comparator subcircuit to a Simscape component and analyze it in an existing model. The SPICE netlist named ee_CMOS_comparator.cir is a netlist that describes the model of a CMOS voltage comparator.

```
% Display ee_CMOS_comparator.cir
type('ee_CMOS_comparator.cir')
```

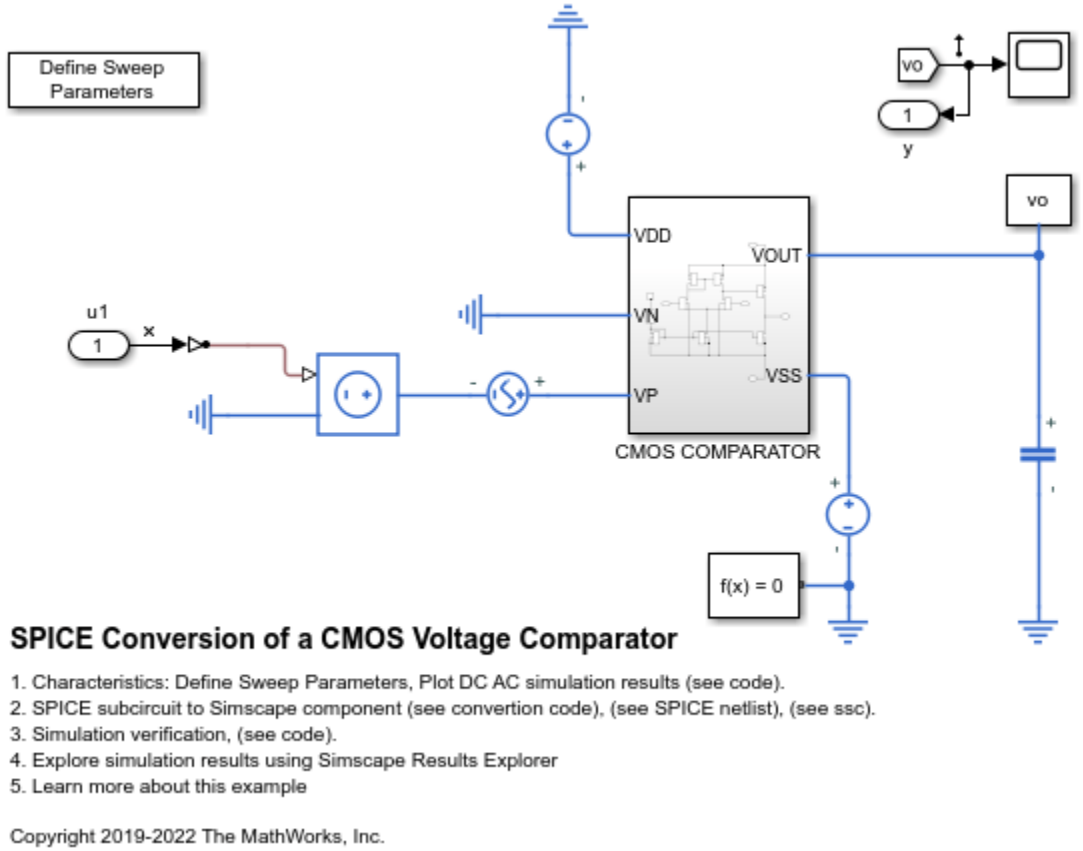
```
* CMOS Voltage Comparator
* Simulation of Two-stage comparator

* Input Signals
VIN VP 0 AC 1V
VOS VN 0 DC 0V

* Power Supplies
VDD VDD 0 DC 5V
VSS VSS 0 DC -5V

* External Components
CL  VOUT  0        2pF

X1 VDD VSS VP VN VOUT COMPARATOR1
* Subcircuit for CMOS Voltage Comparator
.SUBCKT COMPARATOR1 VDD VSS VP VN VOUT
M1  N1  VN N2 VSS    NMOS1  W=680u   L=5u
M2  N3  VP N2 VSS    NMOS1  W=680u   L=5u
M3  N1  N1 VDD VDD    PMOS1  W=5u     L=5u
M4  N3  N1 VDD VDD    PMOS1  W=5u     L=5u
M5  N2  N4 VSS VSS    NMOS1  W=5u     L=5u
M6  VOUT  N3 VDD VDD    PMOS1  W=60u    L=5u
M7  VOUT  N4 VSS VSS    NMOS1  W=30u    L=5u
M8  N4  N4 VSS VSS    NMOS1  W=30u    L=5u
IS  0  N4      20u

.MODEL NMOS1 NMOS VTO=1 KP=17U
+ LEVEL=1
+ GAMMA=0.8 LAMBDA=0.015 PHI=0.6
+ LD=0.5U CJ=5E-4 CJSW=10E-10
+ U0=425 MJ=0.5 MJSW=0.5 CGSO=0.4E-9 CGDO=0.4E-9
.MODEL PMOS1 PMOS VTO=-1 KP=8U
+ LEVEL=1
+ GAMMA=0.4 LAMBDA=0.02 PHI=0.6
+ LD=0.8U CJ=5E-4 CJSW=10E-10
```

```
+ U0=200 MJ=0.5 MJSW=0.5 CGSO=0.4E-9 CGDO=0.4E-9
.ENDS

* Analysis
*.DC VIN  -1e-3 1e-3 1e-5
.AC DEC 10 1e-1 1e6
.PROBE
.END
```

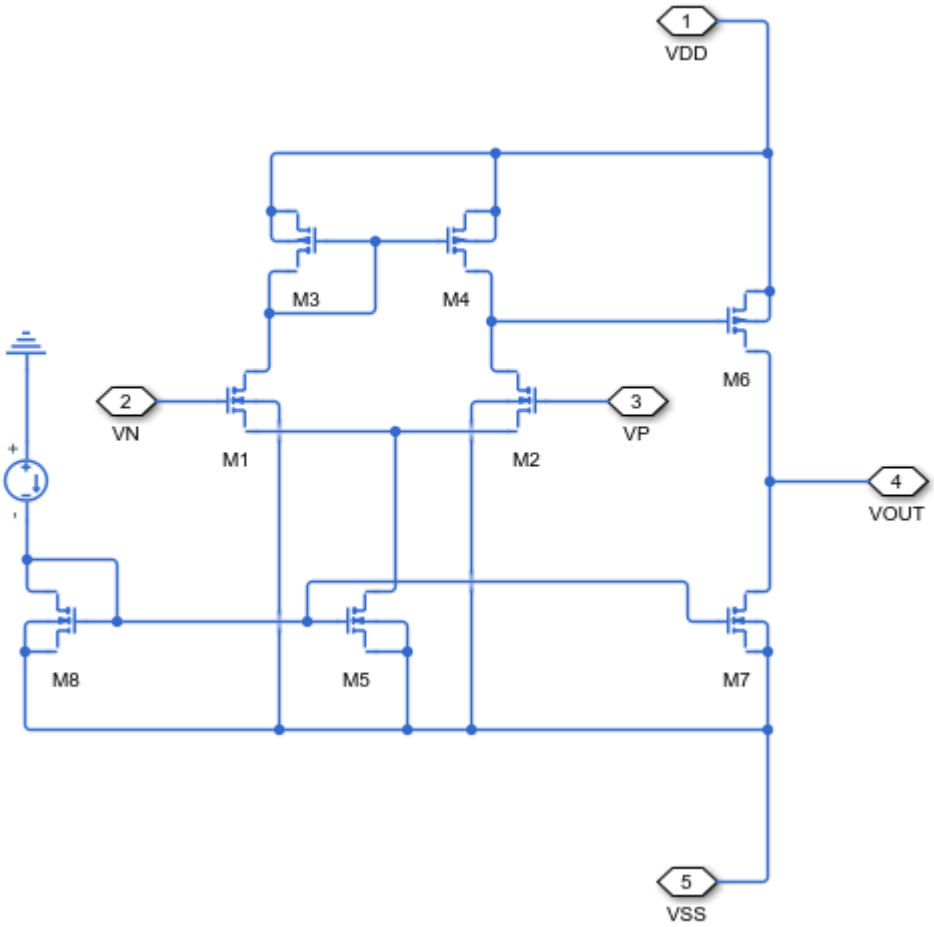
### Open the CMOS Comparator Subsystem

Open the `ee_CMOS_comparator` model, which models the SPICE netlist by using SPICE blocks from the Simscape Electrical™ Additional Components library. This model includes a subsystem called CMOS Comparator. This subsystem is a manual implementation of the voltage comparator subcircuit, which you will replace with a Simscape component converted from the SPICE subcircuit.

```
% Open model
open_system('ee_CMOS_comparator')
set_param(find_system('ee_CMOS_comparator','FindAll', 'on','type','annotation','Tag','ModelFeatu
```

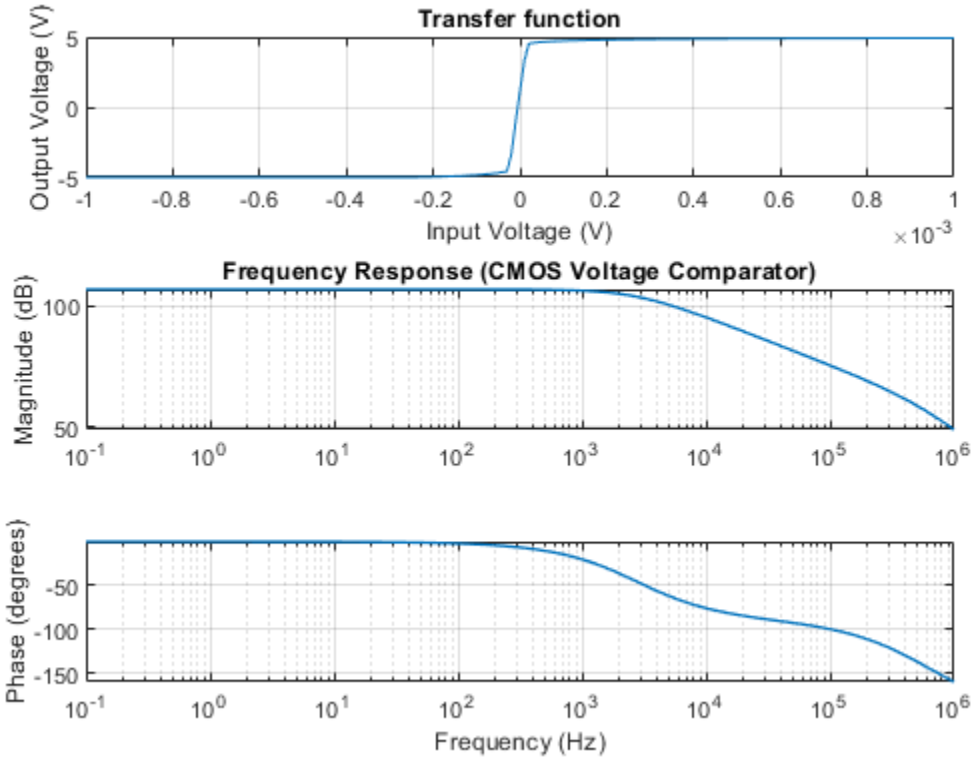



```
% Display CMOS Voltage Comparator Subsystem
open_system('ee_CMOS_comparator/CMOS COMPARATOR','force');
```




The subsystem describes the comparator in a schematic way. The plot below shows the outputs of the CMOS comparator circuit.

```
ee_CMOS_comparator_plot;
```




The gain is around 100 dB and the bandwidth is around 3 kHz. To define the minimum and maximum DC sweep input voltages and AC sweep frequencies, return to the top level of the model and double-click the block labeled Define Sweep Parameters. Click Plot DC AC simulation results in the model to run the simulation and plot the results.

However, for large SPICE netlists with multiple subcircuits, a manual conversion can be inefficient, time consuming, and error-prone. Simscape Electrical provides you with a way to automatically convert a SPICE netlist by using the `subcircuit2ssc` function.

### Convert the SPICE Subcircuit to Simscape Component

You can convert SPICE components with either `.cir` or `.lib` extensions into Simscape components using the `subcircuit2ssc` function. The `subcircuit2ssc` function converts each `.subckt` section inside the SPICE netlist into a single component.

Create a temporary directory that contains validation files and change the present working directory.

```
ee_CMOS_comparator_temporary_directory;
```

The validation files `ee_comparator_traf.mat` and `ee_comparator_freq.mat` are MATLAB files that store data for the SPICE transfer characteristics and the small-signal frequency response.

Use the `subcircuit2ssc` function to convert the voltage comparator subcircuit, `ee_CMOS_comparator.cir`, to a Simscape file and place it in a new directory called `+myComparator`.

```
subcircuit2ssc('ee_CMOS_comparator.cir','+myComparator');
```

```
Netlist converted. Review Simscape component files and make
manual edits for any unsupported items before building the
Simscape library located at:
+myComparator.
```

Generate the Simscape library using the `ssc_build` function.

```
ssc_build myComparator;
```

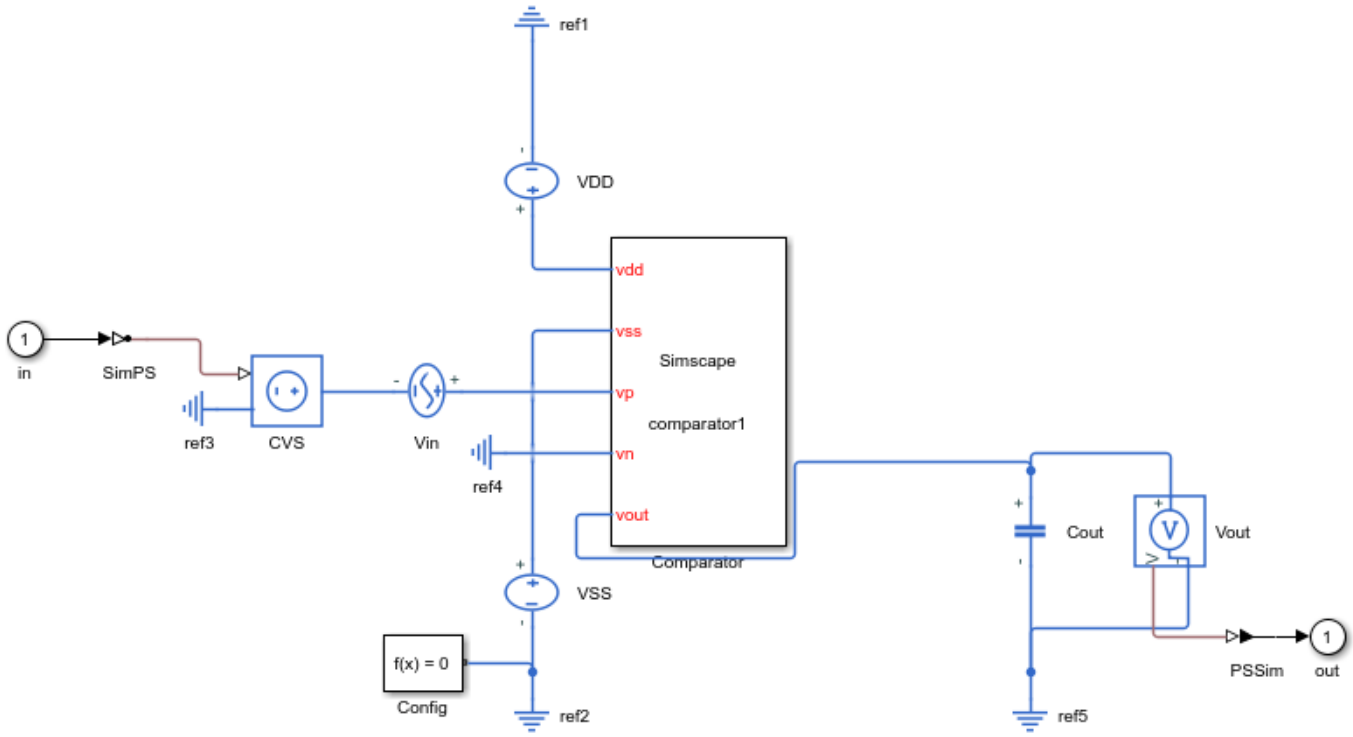
```
Generating Simulink library 'myComparator_lib' in the current directory 'C:\TEMP\Bdoc22b_2054784_
```

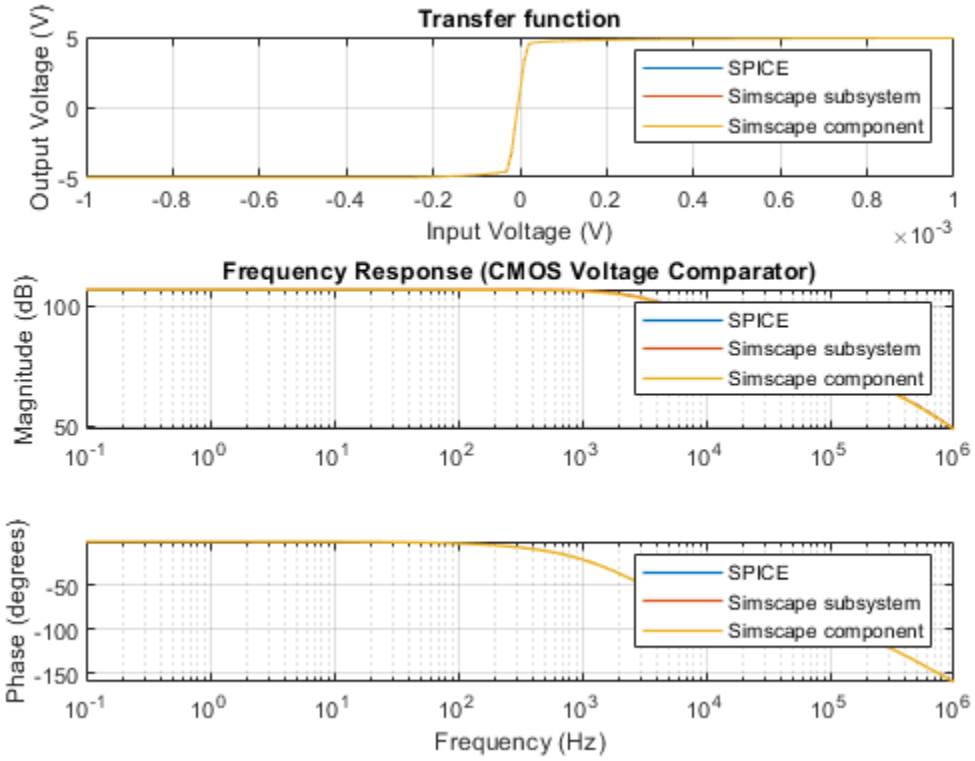
### Verify Simulation Results

You must replace the CMOS Comparator subsystem with the converted component and rewire the terminals. To see the verification code, in the MATLAB Command Window, enter edit ee_CMOS_comparator_verification. The script automatically connects the Simscape component generated from the conversion into the model. Then it obtains the transfer characteristics and the small-signal frequency response of the Simscape model by using a linearization method. The input voltage of the transfer characteristics is in the range of -1 V to 1 V, while the frequency range of the Bode plot is in the range of 0.1 Hz to 1 GHz. Finally, the script plots and compares the simulation results between a pure SPICE netlist model, the original model with the Simscape subsystem, and the model with the converted Simscape component obtained using `subcircuit2ssc`.

The plot below shows the simulation results from the three different models. The transfer characteristics and frequency response of the model with the converted Simscape component match those of the original SPICE netlist.

```
ee_CMOS_comparator_verification;
```






### Clean Up

Finally, delete the temporary directory and all its subdirectories.

```
% Delete temporary directory
if exist('originalDirectory','var')
    cd (originalDirectory);
    rmdir(temporaryDirectory,'s')
    clear temporaryDirectory originalDirectory;
end
```

# Troubleshoot a SPICE Conversion of an IGBT Subcircuit

This example shows how to troubleshoot the conversion of a SPICE insulated gate bipolar transistor (IGBT) subcircuit and convert it into an equivalent Simscape™ component. The `subcircuit2ssc` function converts all the subcircuit components inside a SPICE netlist file into one or more equivalent Simscape files. However, in some cases, you must manually edit the original SPICE netlist.

### Open the IGBT Subcircuit

In this example, you will convert this IGBT to a Simscape component, add it to a model, and simulate the model. The IGBT subcircuit combines the bipolar junction transistor (BJT) and SPICE Level 1 metal-oxide-semiconductor field-effect transistor (MOSFET) models and equations with a drain-gate capacitance model. To open the SPICE netlist, in the MATLAB command window, enter edit ee_igbt_subckt_original.cir.

```
% Display ee_igbt_subckt_original.cir
type('ee_igbt_subckt_original.cir')

% Create a temporary directory and change present working directory
ee_igbt_subckt_temporary_directory;

% Create a subdirectory called +myIGBT
if ~exist('+myIGBT', 'dir')
    mkdir('+myIGBT')
end
```

```
* Fast and accurate SPICE IGBT model

.SUBCKT IGBTMODEL C G 0
.FUNC WDSJ(X1,X2) {SQRT(12.422*{ESI}*VDS(X1)/(X2))}
.FUNC VDS(X) {SQRT(X^2+0.01^2)/2}
M1 N1 N2 0 0 MOSN
RG G N2 {5.7-7M*(125-TEMP)}
Q1 0 N1 C BJTP
CGDOX N3 N2 0.1N
VICGDOX N1 N3 0
GCGD N1 N2
+VALUE = {(COXD*AGD*ESI/(WDSJ(V(N1,N2),{NB})*COXD+AGD*ESI))*I(VICGDOX)}

.PARAM TEMP=25
.PARAM AGD=50 ESI=1.0359 NB=2 COXD=175 VTO={4+7M*(125-TEMP)}
.MODEL MOSN NMOS (IS=20N KP=20 VTO={VTO} RD=0.5M CGSO=160U)
.MODEL BJTP PNP (BF=0.26 TF=1.5U CJC=0.6N RE=1M BR=0.44 IS=2E-15 NF=2 XTI=6 XTB=5)

.ENDS IGBTMODEL
```

### Modify the IGBT SPICE Subcircuit

Not all SPICE subcircuits can be directly converted with the `subcircuit2ssc` function. If you run the `subcircuit2ssc` conversion directly on this file you will receive the following error:

```
"Error using spiceSubckt/extractSubcktDefinitionStrings (line 1182)"
"Connecting public nodes to ground is illegal."
```

```
"Error in spiceSubckt/loadSubckt (line 379)"
                "this.extractSubcktDefinitionStrings; % parse the subckt line"

"Error in spiceSubckt (line 69)"
                "this.loadSubckt(fullNetlist); % populate the object properties"

"Error in subcircuit2ssc (line 55)"
    "subcircuitArray(ii) = spiceSubckt(file,subcctName(ii));"

"Error in subcircuit2ssc (line 31)"
        "subcircuit2ssc(netlist,target);"
```

The conversion generates an error because the SPICE subcircuit connected a public node directly to node 0. In SPICE, node 0 is ground. To resolve the error, update the SPICE netlist file by renaming node 0 to node E.

If you now run `subcircuit2ssc` and then `ssc_build` you will receive the following error:

```
"Failed to generate 'myIGBT_lib'"

"Caused by:"
    "Error using ne_parselibrarypackage (line 66)"
    "Error: Class member redefinition has been found:"
        "Error using myIGBT.igbtmodel> (line 22)"
        "Class member 'temp' is defined."
        "Error using myIGBT.igbtmodel> (line 31)"
        "Class member 'temp' is defined."
```

The build generates an error because the SPICE subcircuit includes the definition of a local parameter named TEMP which is a reserved SPICE variable name for global temperature. To resolve the error and use the global temperature, remove the local parameter definition of TEMP from the SPICE library file. Alternatively, if you want to specify a local temperature, choose a name other than TEMP for the local parameter.

Save the changes into a new file named `ee_igbt_subckt_modified.cir`.

```
% Display ee_igbt_subckt_modified.cir
type('ee_igbt_subckt_modified.cir')


* Fast and accurate SPICE IGBT model

.SUBCKT IGBTMODEL C G E
.FUNC WDSJ(X1,X2) {SQRT(12.422*{ESI}*VDS(X1)/(X2))}
.FUNC VDS(X) {SQRT(X^2+0.01^2)/2}
M1 N1 N2 E E MOSN
RG G N2 {5.7-7M*(125-TEMP)}
Q1 E N1 C BJTP
CGDOX N3 N2 0.1N
VICGDOX N1 N3 0
GCGD N1 N2
+VALUE = {(COXD*AGD*ESI/(WDSJ(V(N1,N2),{NB})*COXD+AGD*ESI))*I(VICGDOX)}

.PARAM AGD=50 ESI=1.0359 NB=2 COXD=175 VTO={4+7M*(125-TEMP)}
.MODEL MOSN NMOS (IS=20N KP=20 VTO={VTO} RD=0.5M CGSO=160U)
.MODEL BJTP PNP (BF=0.26 TF=1.5U CJC=0.6N RE=1M BR=0.44 IS=2E-15 NF=2 XTI=6 XTB=5)

.ENDS IGBTMODEL
```

To see the differences between the two files, in the MATLAB command window, enter visdiff('ee_igbt_subckt_original.cir','ee_igbt_subckt_modified.cir').

Run this command to convert the SPICE subcircuit of the `ee_igbt_subckt_modified.cir` model to a Simscape component and place the generated files in the newly created directory called `+myIGBT`:

```
subcircuit2ssc('ee_igbt_subckt_modified.cir','+myIGBT')
```

```
Netlist converted. Review Simscape component files and make
manual edits for any unsupported items before building the
Simscape library located at:
+myIGBT.
```

Generate the Simscape library using the `ssc_build` function.

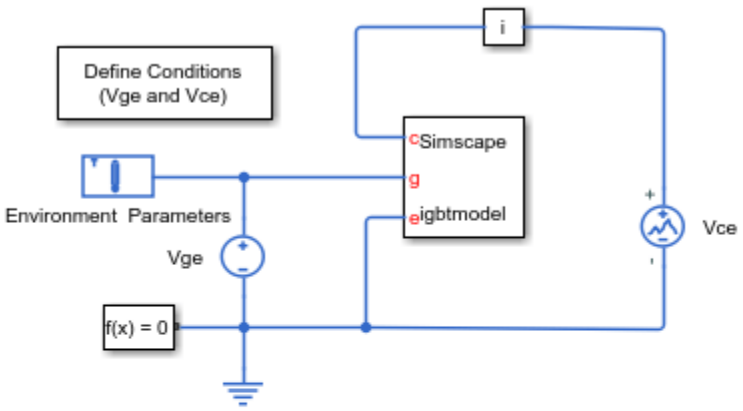
```
ssc_build myIGBT
```

```
Generating Simulink library 'myIGBT_lib' in the current directory 'C:\TEMP\Bdoc22b_2054784_6060\
```

### Open the Converted IGBT Model

The `igbtmodel.ssc` file stored in the `+myIGBT` directory is the converted Simscape component obtained by running the `subcircuit2ssc` function on the modified SPICE netlist. The `subcircuit2ssc` function also converted all the functions implemented in the SPICE subcircuit and stored them in a subdirectory named `+igbtmodel_simscape_functions`. To edit the generated simscape component, in the MATLAB command window, enter edit +myIGBT/igbtmodel.ssc.

```
open_system('ee_igbt_subckt')
set_param(find_system('ee_igbt_subckt','FindAll', 'on','type','annotation','Tag','ModelFeatures'
```




**Troubleshoot a SPICE Conversion of an IGBT Subcircuit**

1. IGBT curves: Define Vge and Vce, plot curves (see code)
2. SPICE subcircuit to Simscape component (see SPICE netlist), (see ssc)
3. Simulation verification, (see code)
4. Explore simulation results using Simscape Results Explorer
5. Learn more about this example

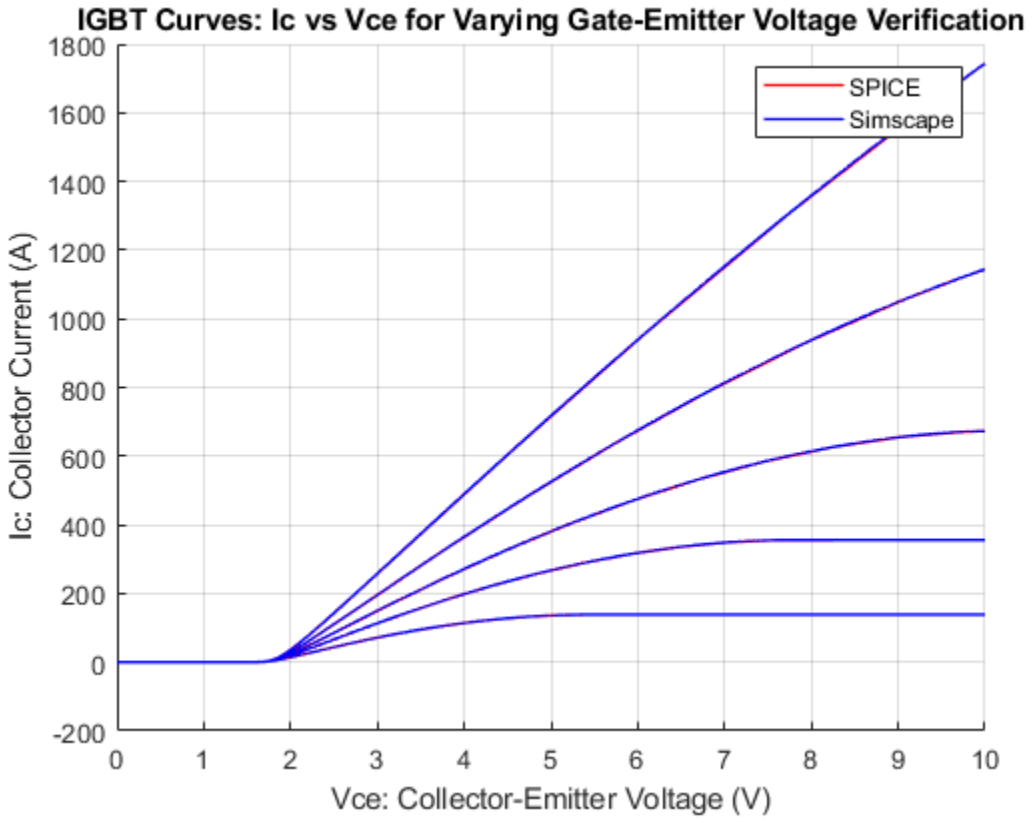
Copyright 2019-2022 The MathWorks, Inc.

To adjust the Simscape global temperature, in the Environment Parameters block, modify the **Temperature** parameter.

### Verify Simulation Results

The `ee_igbt_subckt_verification` script compares collector current versus collector-emitter voltage curves for a range of gate-emitter voltages with the SPICE netlist simulation results, which are stored in the `ee_igbt_subckt_data.mat` MAT-file. The gate-emitter voltages, Vge, are 8 V, 10 V, 12 V, 15 V, and 20 V. The Vce voltage sweeps from 0 V to 10 V.

```
ee_igbt_subckt_verification;
```




To plot these characteristics with different settings, double-click the block labeled Define Conditions (Vge and Vce) and define the vector of Vge and the minimum and maximum Vce voltages. In the model, to run the simulations and plot the results, click **plot curves**.

### Clean Up

Finally, delete the temporary directory and all its subdirectories.

```
% Delete temporary directory
if exist('originalDirectory','var')
    cd (originalDirectory);
    rmdir(temporaryDirectory,'s')
    clear temporaryDirectory originalDirectory;
end
```

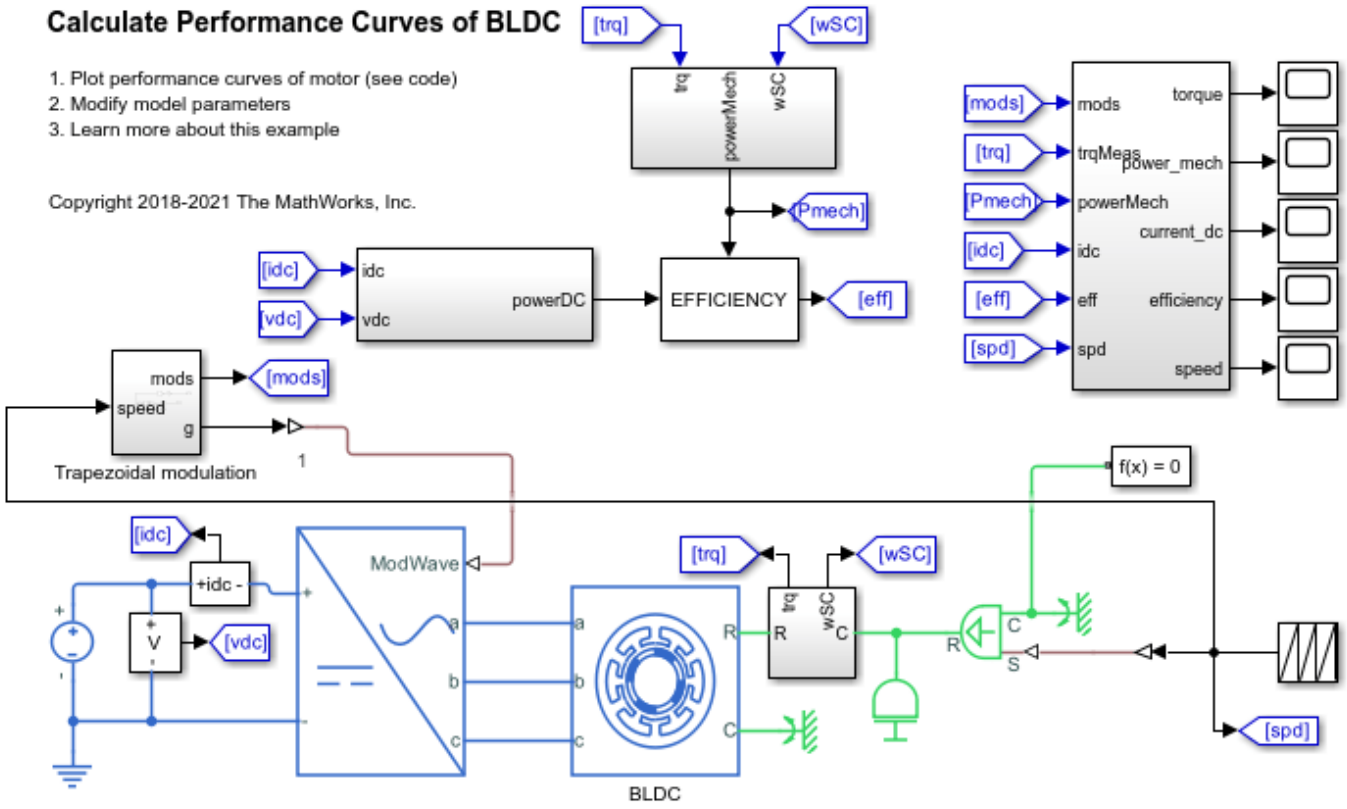
### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 50 microseconds.

# Calculate Performance Curves of BLDC

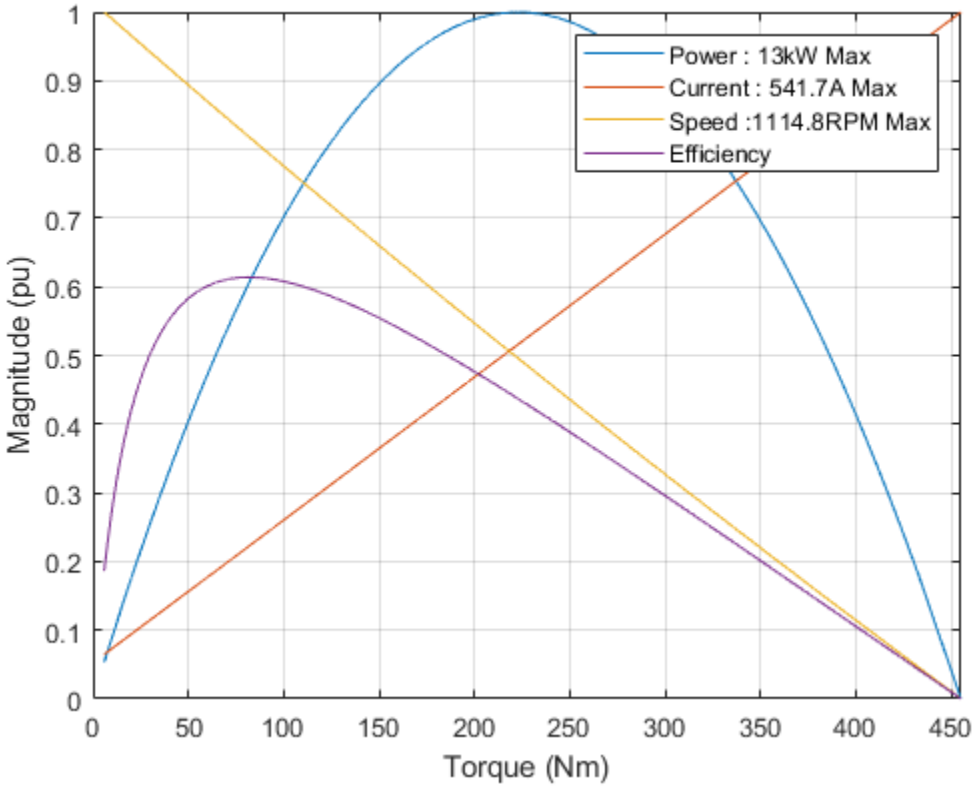
This example shows how to calculate performance curves for a brushless DC (BLDC) motor. The simulation includes a speed ramp. Ideal trapezoidal modulation waves are used to drive an average-value converter. A triggered subsystem is used to determine the peak torque, power, current, and efficiency values for a given speed.

### Model




### Simulation Results

The plot below shows the performance curves of BLDC using ideal input.



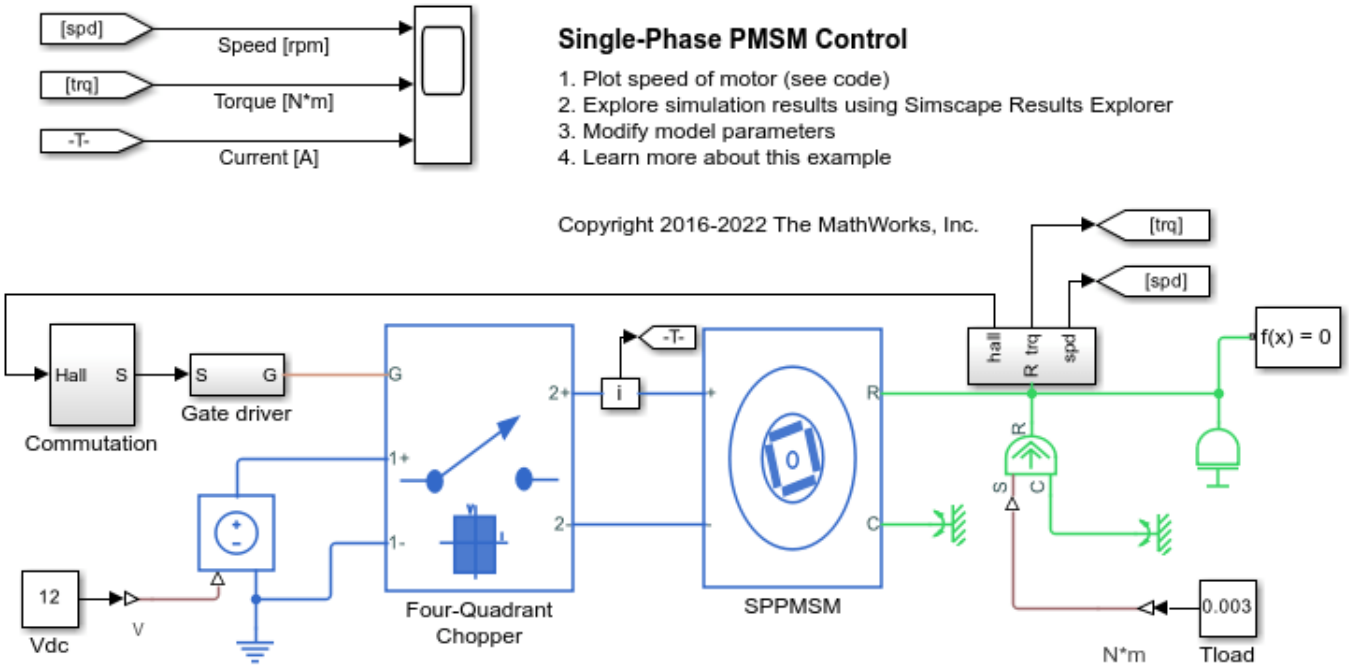

**Results from Real-Time Simulation**

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 40 microseconds.

# Single-Phase PMSM Control

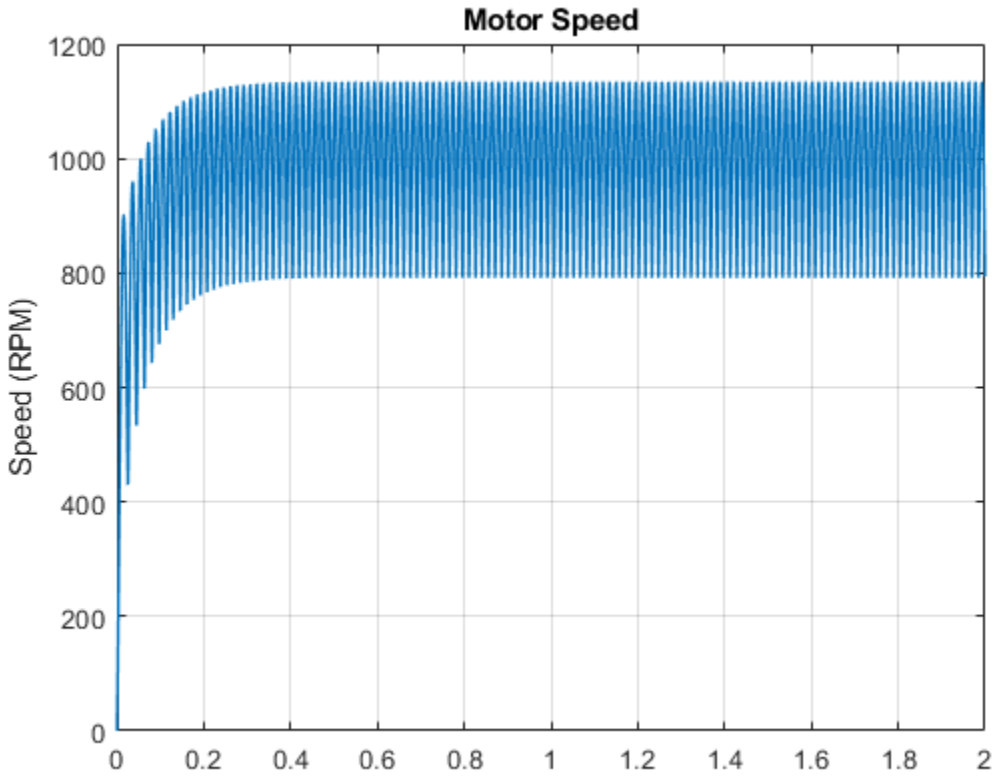
This example shows how to control the rotor speed in an electrical drive based on a single-phase permanent magnet synchronous motor by using open loop control. An ideal torque source provides the load. The Commutation subsystem uses a Hall signal to compute the gate signals. An H-Bridge feeds the SPPMSM.

### Model




### Simulation Results from Simscape Logging

The plot below shows the measured speed for the test.



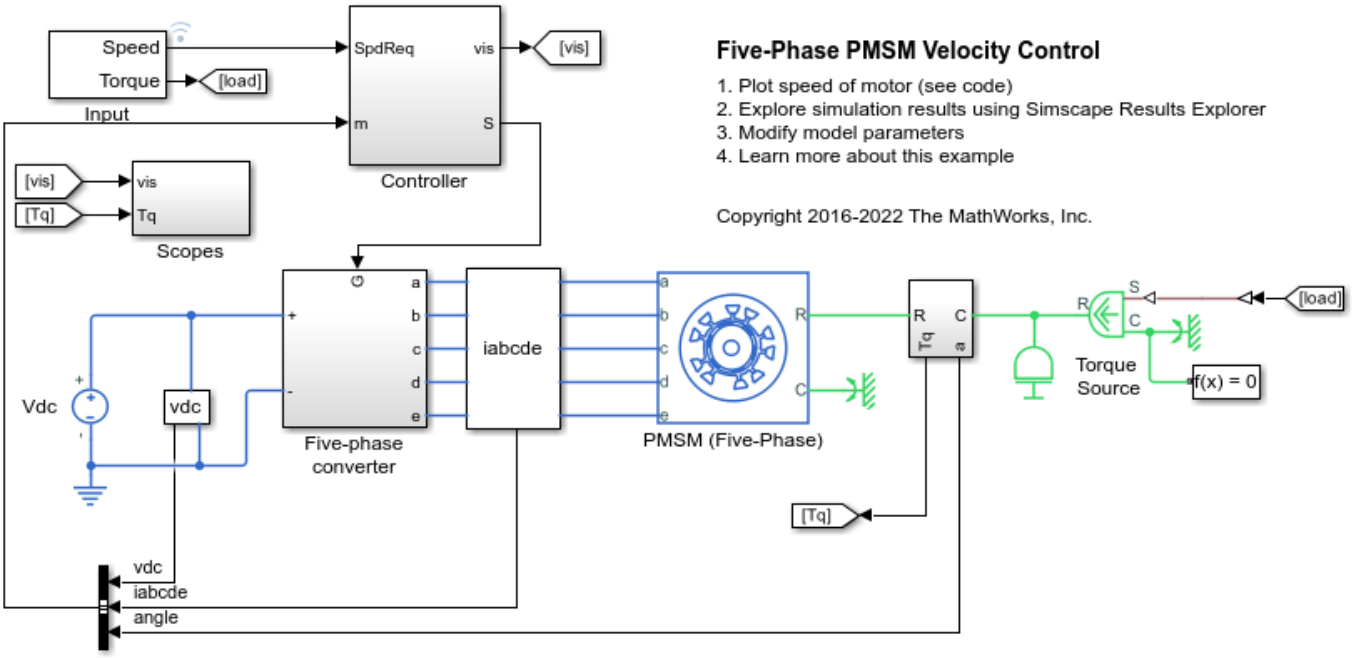

**Results from Real-Time Simulation**

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 10 microseconds.

# Five-Phase PMSM Velocity Control

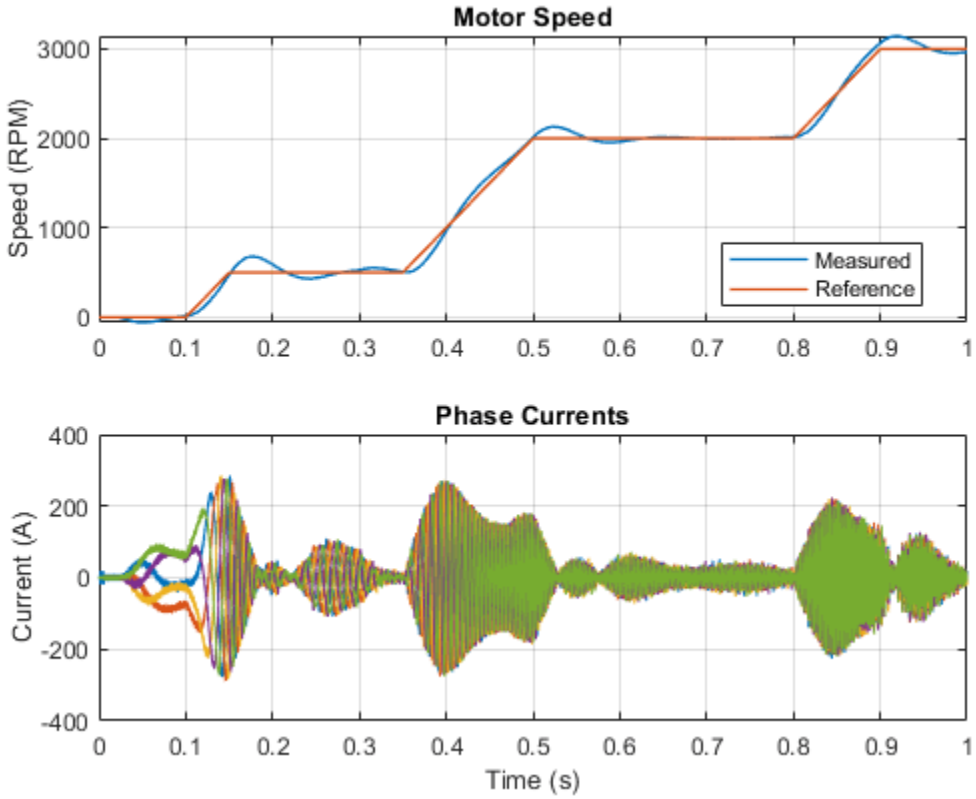
This example shows how to control the rotor angular velocity in an electrical-traction drive based on a five-phase permanent magnet synchronous machine (PMSM). A DC voltage source feeds the PMSM through a controlled five-phase converter. The PMSM operates in both motoring and generating modes according to the load. An ideal torque source provides the load. The Scopes subsystem contains scopes that allow you to see the simulation results. The Control subsystem includes a PI-based cascade control structure that has an outer angular-velocity-control loop and four inner current-control loops. During the one second simulation, the angular velocity demand is 0 rpm, 500 rpm, 2000 rpm, and then 3000 rpm.

### Model




### Simulation Results from Simscape Logging

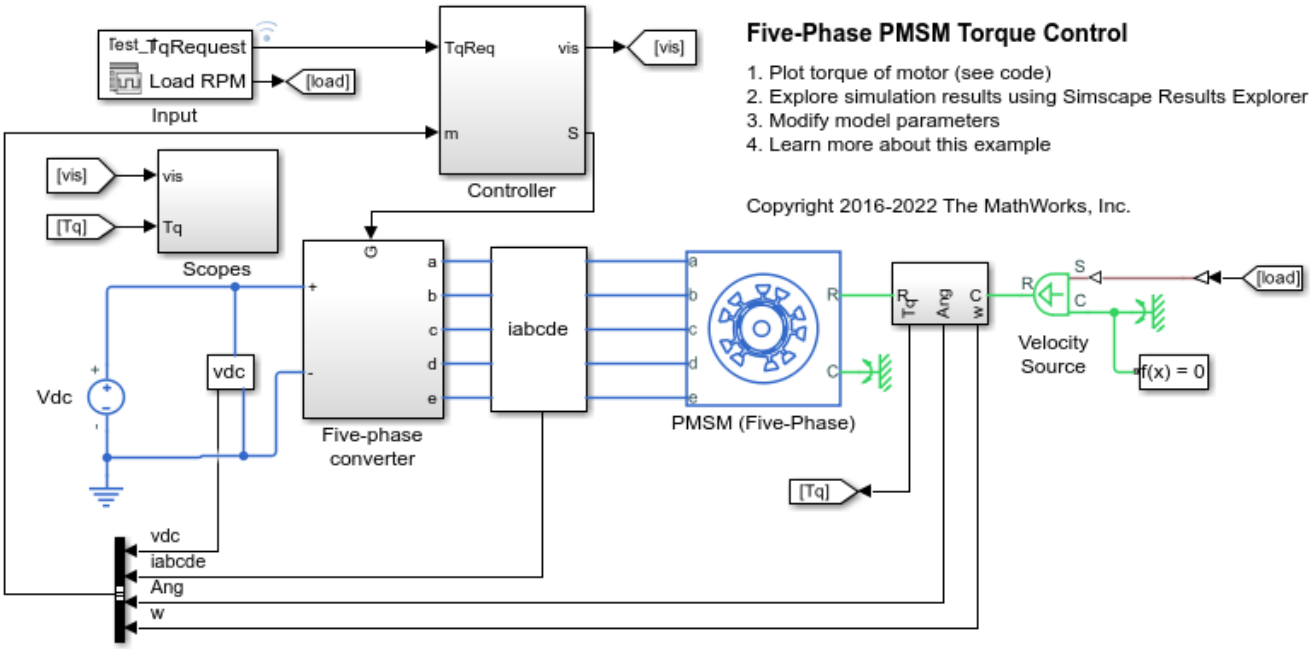
The plot below shows the requested and measured speed for the test, as well as the phase currents in the electric drive.


Motor Speed
Speed (RPM)
3000
2000
1000
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Measured
Reference
Phase Currents
Current (A)
400
200
0
-200
-400
Time (s)

# Five-Phase PMSM Torque Control

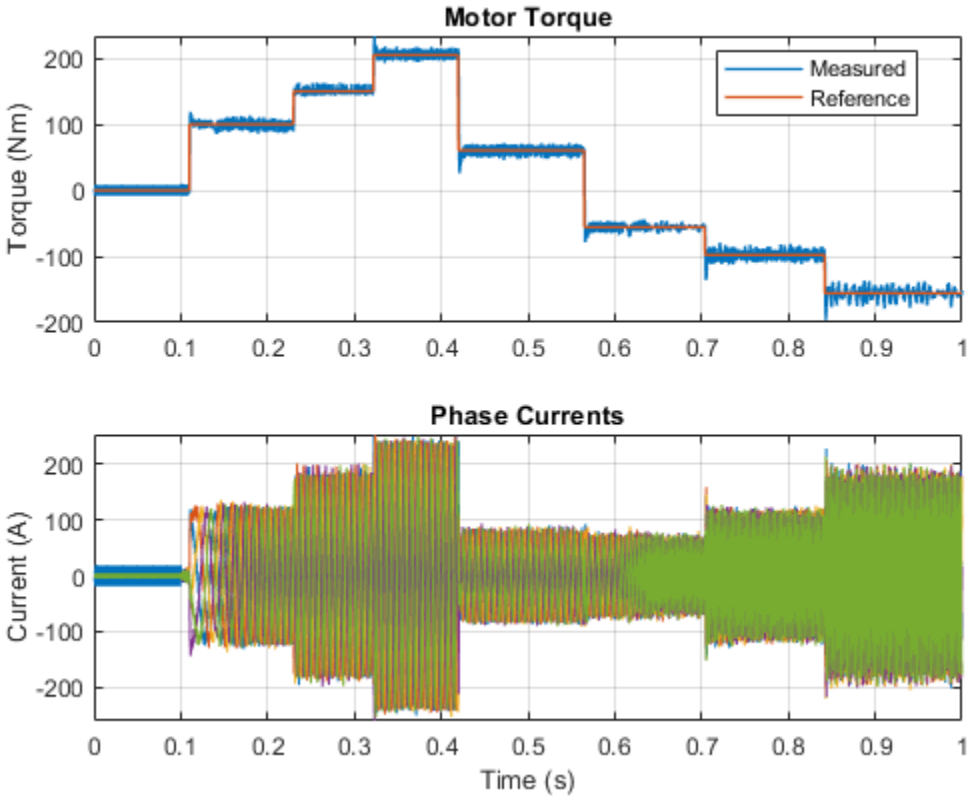
This example shows how to control the torque in an electrical-traction drive based on a five-phase permanent magnet synchronous machine (PMSM). A DC voltage source feeds the PMSM through a controlled five-phase converter. The PMSM operates in both motoring and generating modes according to the load. An ideal angular velocity source provides the load. The Control subsystem uses an open-loop approach to control the PMSM torque and a closed-loop approach to control the current. At each sample instant, the torque request is converted to a relevant q-axis current reference. The current control is PI-based. The simulation uses several torque steps in both motor and generator modes. The Scopes subsystem contains scopes that allow you to see the simulation results.

### Model




### Simulation Results from Simscape Logging

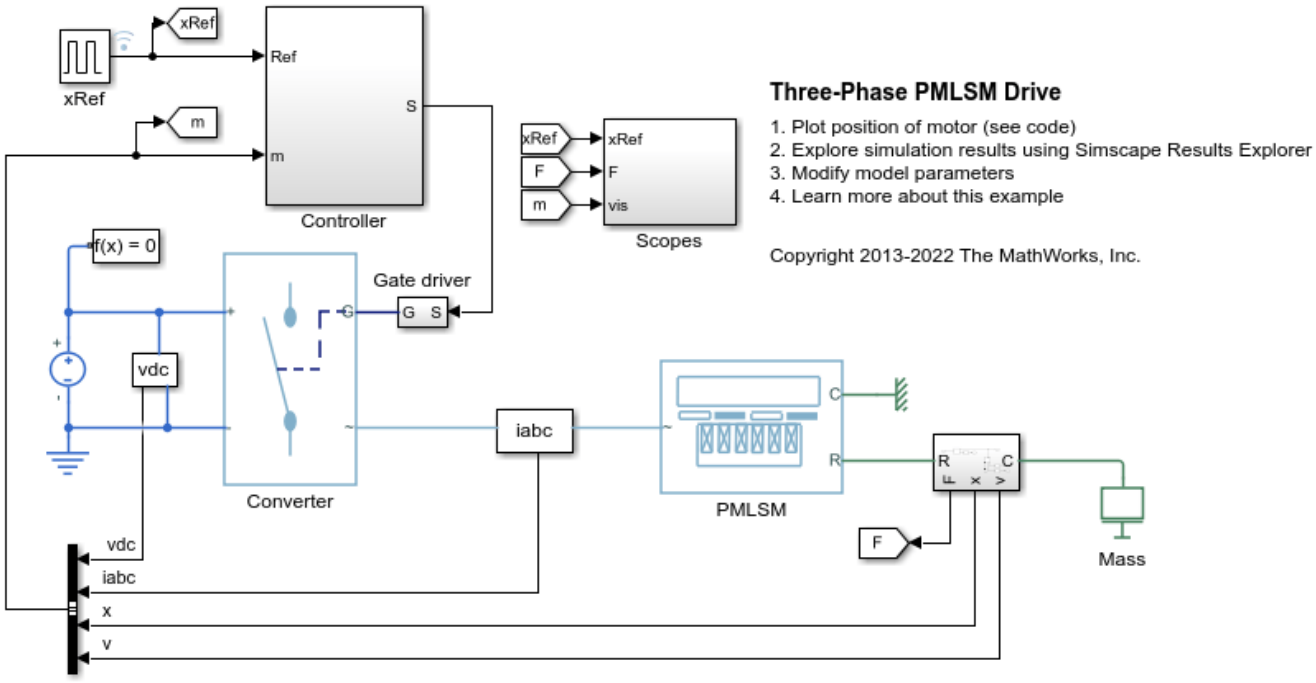
The plot below shows the requested and measured torque for the test, as well as the phase currents in the electric drive.


Motor Torque
Measured
Reference
Torque (Nm)
200
100
0
-100
-200
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Phase Currents
Current (A)
200
100
0
-100
-200
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Time (s)

# Three-Phase PMLSM Drive

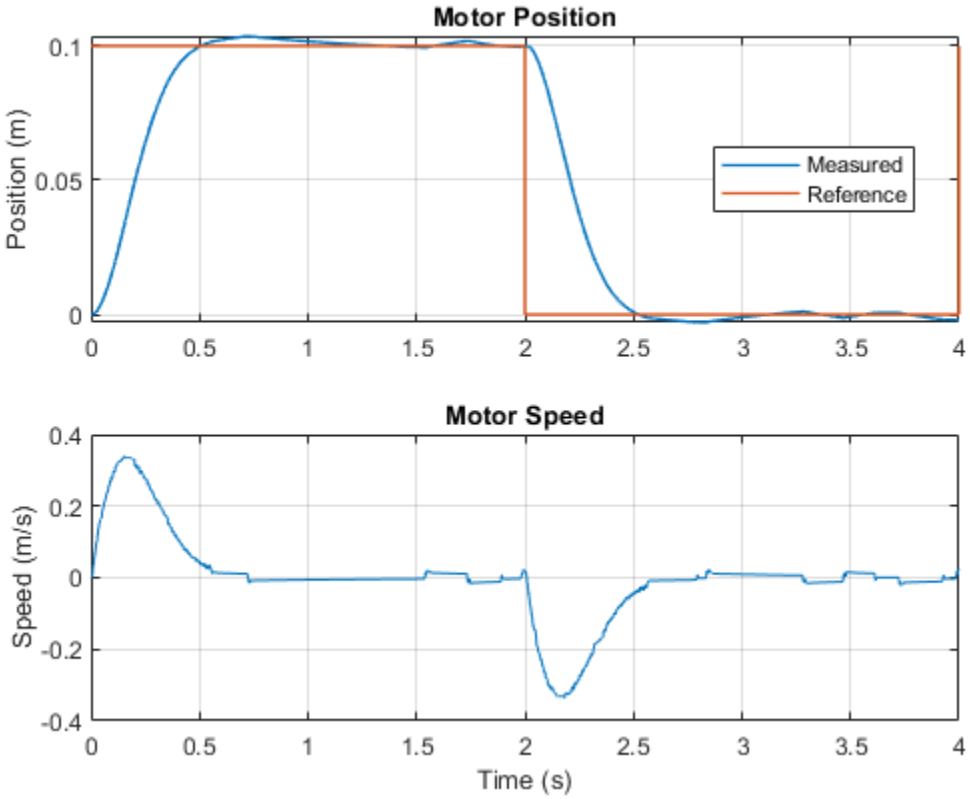
This example shows how to control the position in a three-phase permanent magnet linear synchronous machine (PMLSM) drive. The Control subsystem uses a PI-based cascade control structure with three control loops, an outer position control loop, a speed control loop and two inner current control loops. A controlled three-phase converter feeds the PMLSM. The simulation uses step references. The Scopes subsystem contains scopes that allow you to see the simulation results.

### Model




### Simulation Results from Simscape Logging

The plot below shows the requested and measured position for the test, as well as the linear speed in the electric drive.



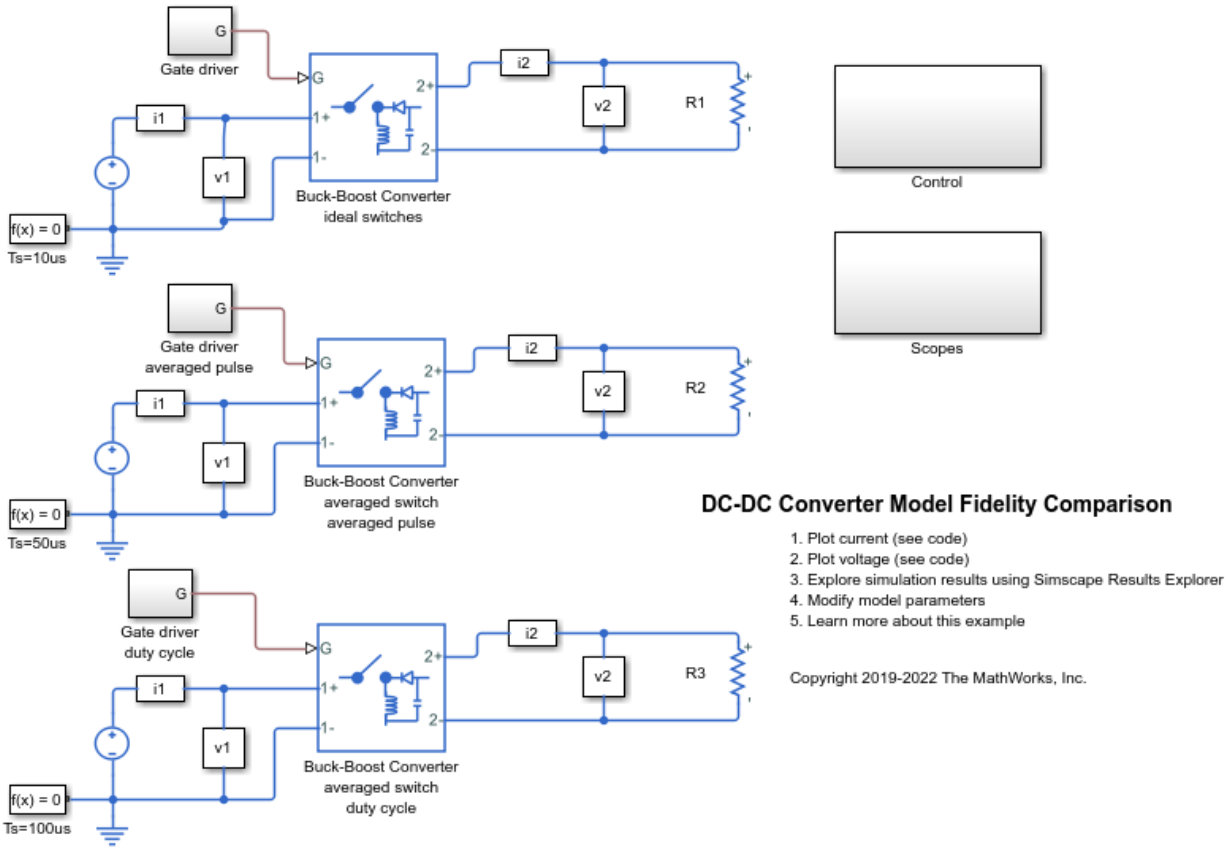

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 50 microseconds.

# DC-DC Converter Model Fidelity Comparison

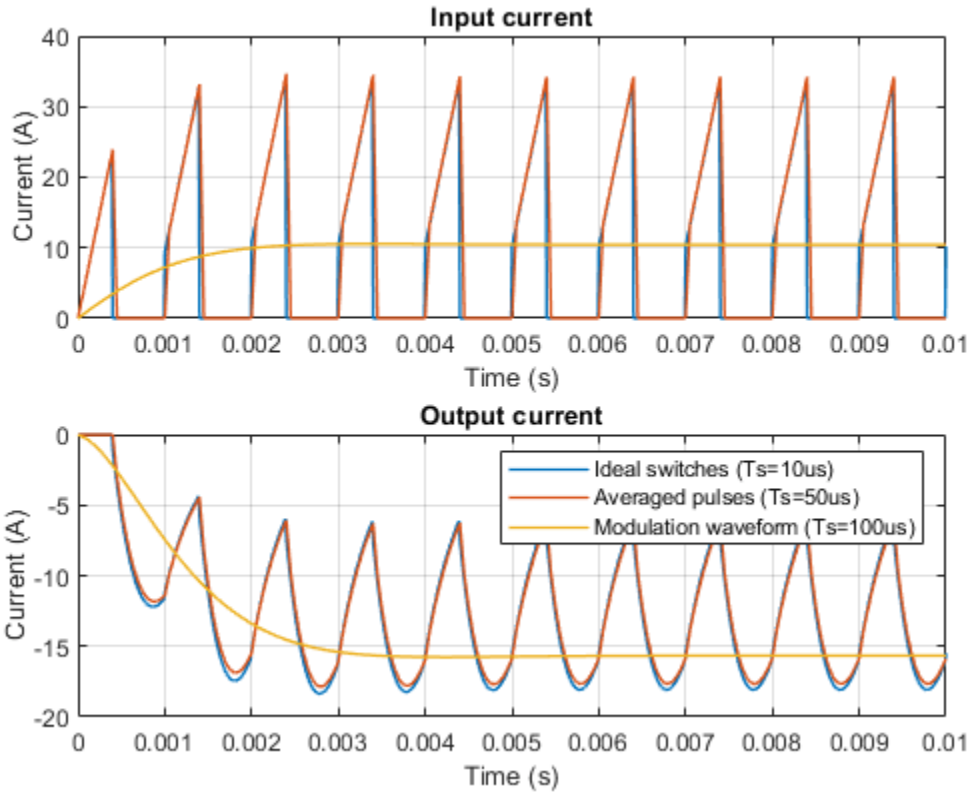
This example shows how to use different levels of fidelity in DC-DC converters. The system contains three buck-boost converters. The top converter uses an ideal switch at a sample time of 10 us. To yield accurate results even though the model is under sampled at a sample time of 50 us, the middle converter uses an averaged switch with averaged pulse. To further increase the sample rate and to operate as an ideal averaged converter, the bottom converter uses an averaged switch and duty cycle instead of gate pulse. The Control subsystem contains a PWM generator. The Scopes subsystem contains Scope blocks that allow you to see the simulation results.

### Model

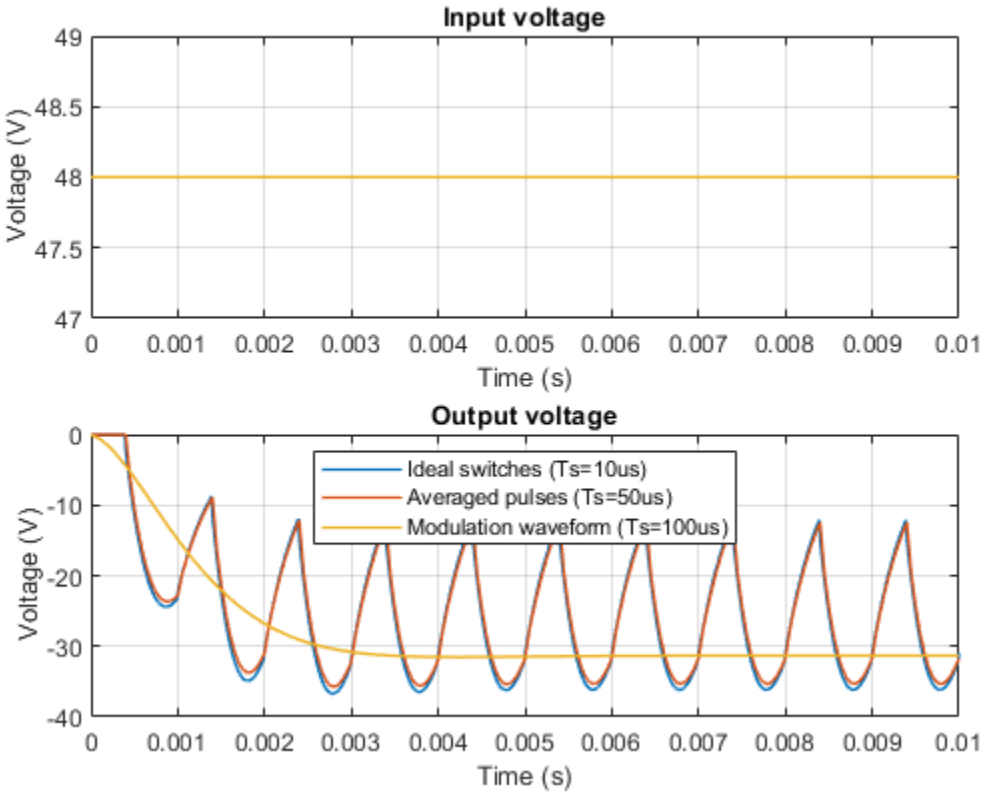



### Simulation Results from Simscape Logging

The plot below shows the comparison of the currents on both sides of the converter.



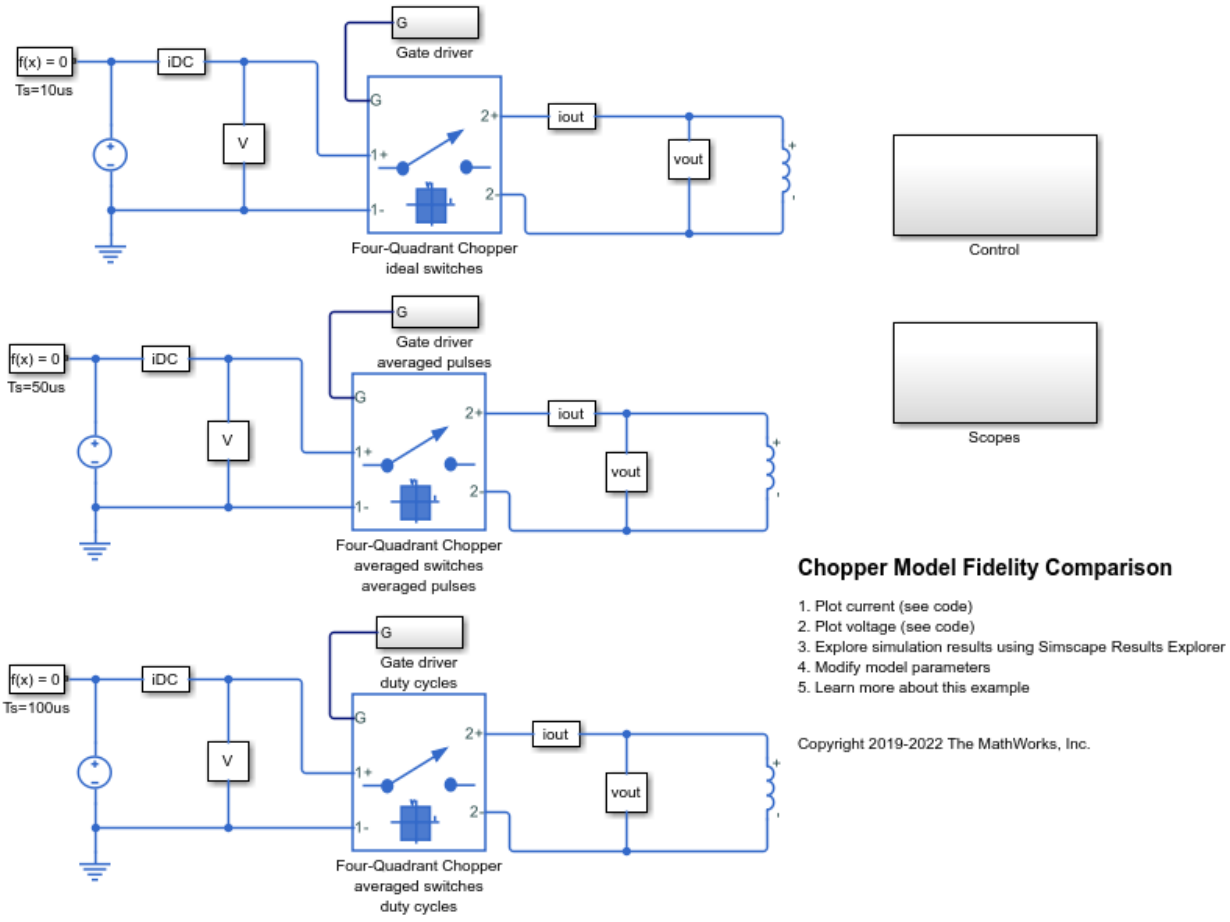

The plot below shows the comparison of the voltages on both sides of the converter.


Input voltage
Voltage (V)
49
48.5
48
47.5
47
0
0.001
0.002
0.003
0.004
0.005
0.006
0.007
0.008
0.009
0.01
Time (s)
Output voltage
Ideal switches (Ts=10us)
Averaged pulses (Ts=50us)
Modulation waveform (Ts=100us)
Voltage (V)
0
-10
-20
-30
-40
Time (s)

# Chopper Model Fidelity Comparison

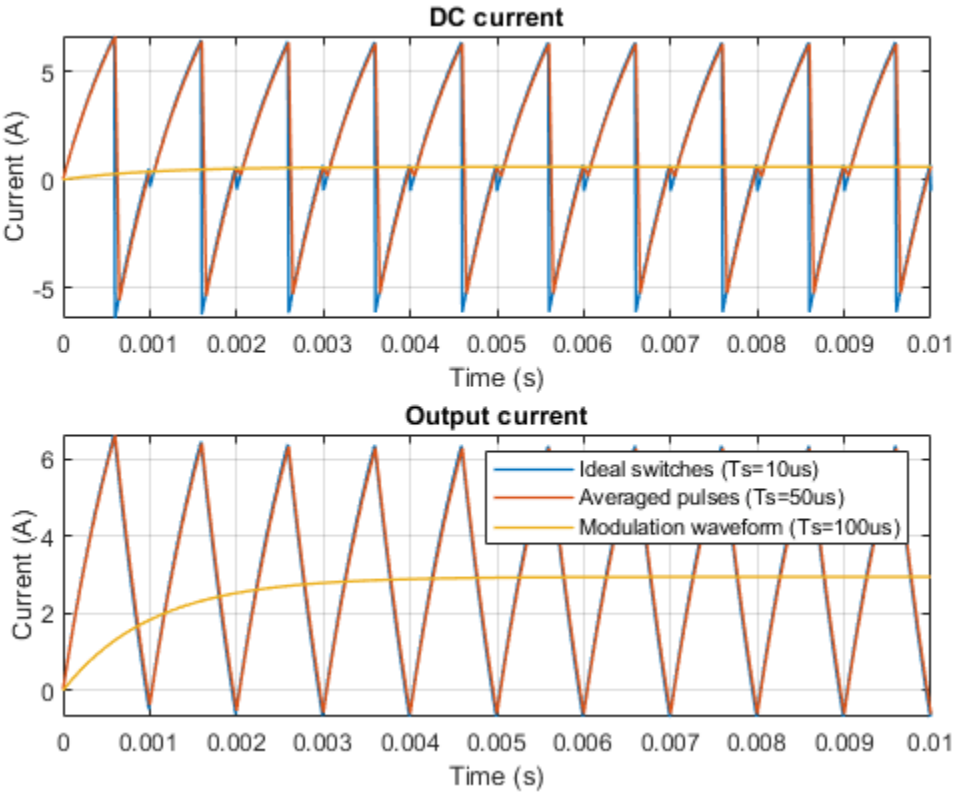
This example shows how to use different levels of fidelity in chopper converters. The system contains three four-quadrant choppers. The top converter uses ideal switches at a sample time of 10 us. To yield accurate results even though the model is under sampled at a sample time of 50 us, the middle chopper uses averaged switches with averaged pulses. To further increase the sample rate and to operate as an ideal averaged chopper, the bottom converter uses averaged switches and duty cycles instead of gate pulses. The Control subsystem contains a PWM generator. The Scopes subsystem contains Scope blocks that allow you to see the simulation results.

### Model

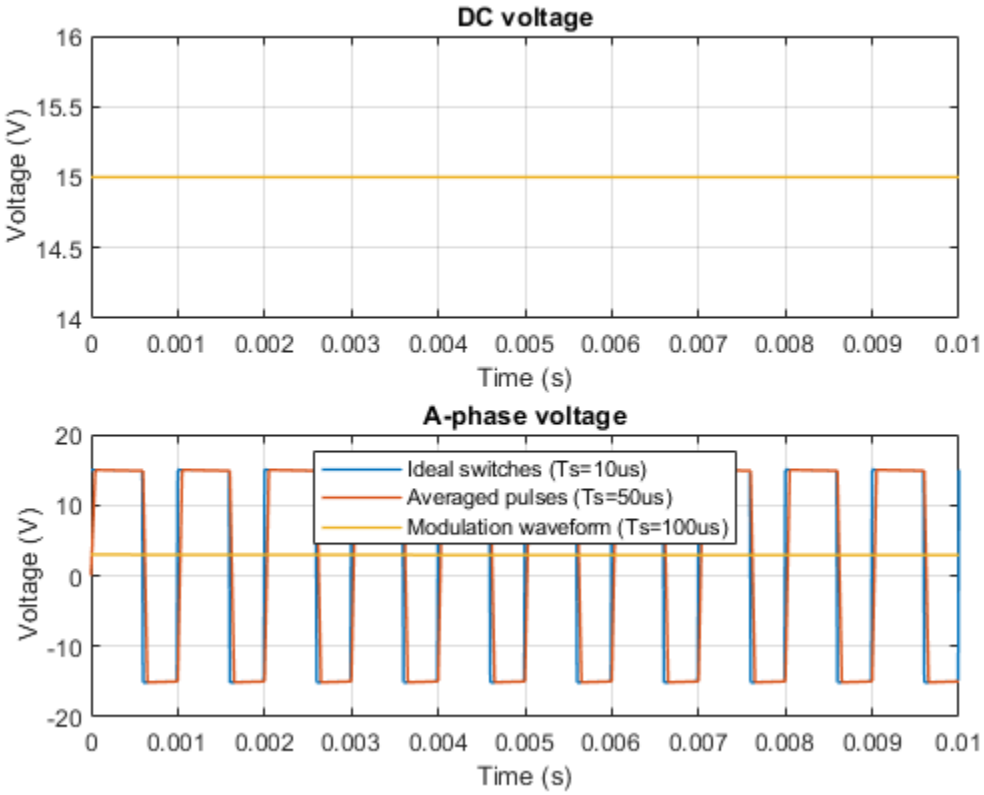



### Simulation Results from Simscape Logging

The plot below shows the comparison of the currents on both sides of the converter..




The plot below shows the comparison of the voltages on both sides of the converter.


DC voltage
Voltage (V)
16
15.5
15
14.5
14
0
0.001
0.002
0.003
0.004
0.005
0.006
0.007
0.008
0.009
0.01
Time (s)
A-phase voltage
Ideal switches (Ts=10us)
Averaged pulses (Ts=50us)
Modulation waveform (Ts=100us)
Voltage (V)
20
10
0
-10
-20
0
0.001
0.002
0.003
0.004
0.005
0.006
0.007
0.008
0.009
0.01
Time (s)

# Single-Phase Grid-Connected Solar Photovoltaic System

This example shows how to model a rooftop single-phase grid-connected solar photovoltaic (PV) system. This example supports design decisions about the number of panels and the connection topology required to deliver the target power. The model represents a grid-connected rooftop solar PV that is implemented without an intermediate DC-DC converter. To parameterize the model, the example uses information from a solar panel manufacturer datasheet. Solar power is injected into the grid with unity power factor (UPF).

To track the maximum power point (MPP), the example uses these maximum power point tracking (MPPT) techniques:

- Incremental conductance
- Perturbation and observation

Three inverter options are available:

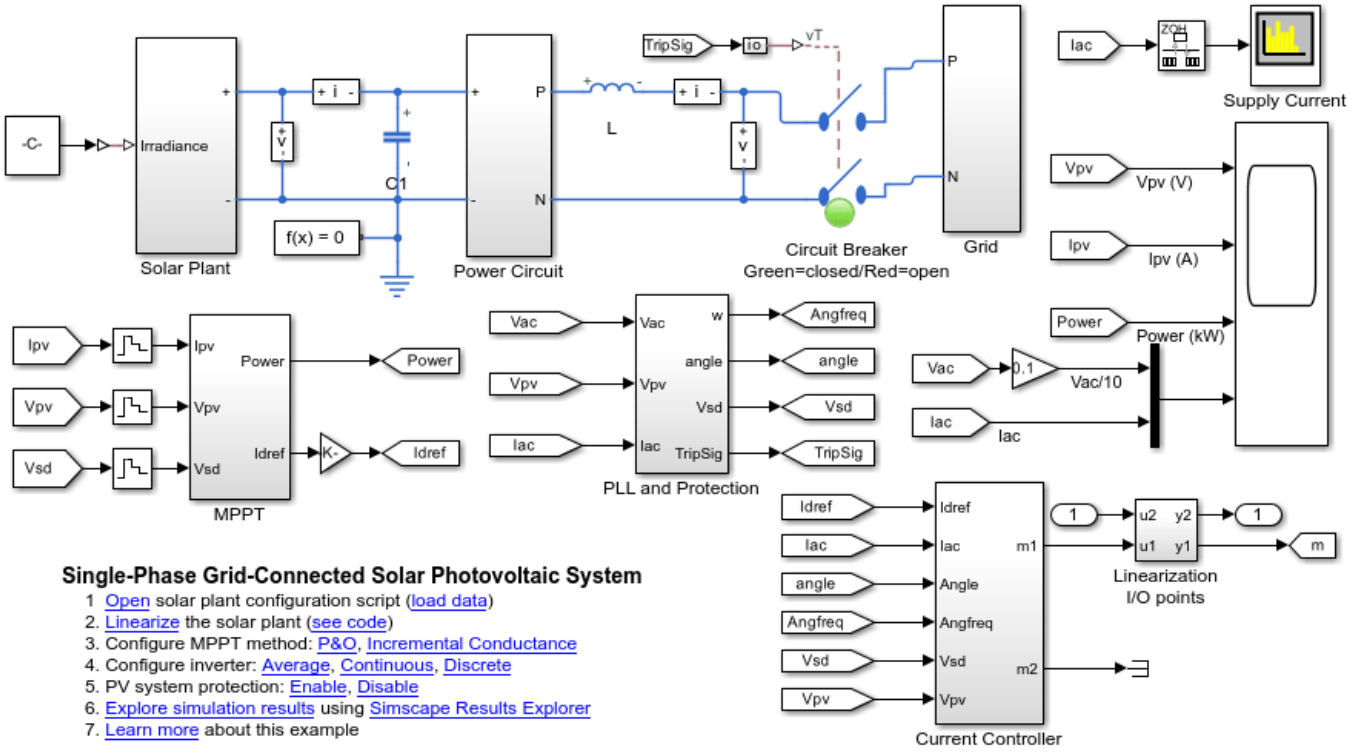
- Average
- Continuous
- Discrete

This example linearizes the system to generate an open-loop Bode plot from which phase and gain margin can be determined.

To open a script that provides information on the parameterization, features, and options in this example, at the MATLAB® command line, enter: edit 'ee_solar_gridconnected_singlephase_data'

### PV System Model

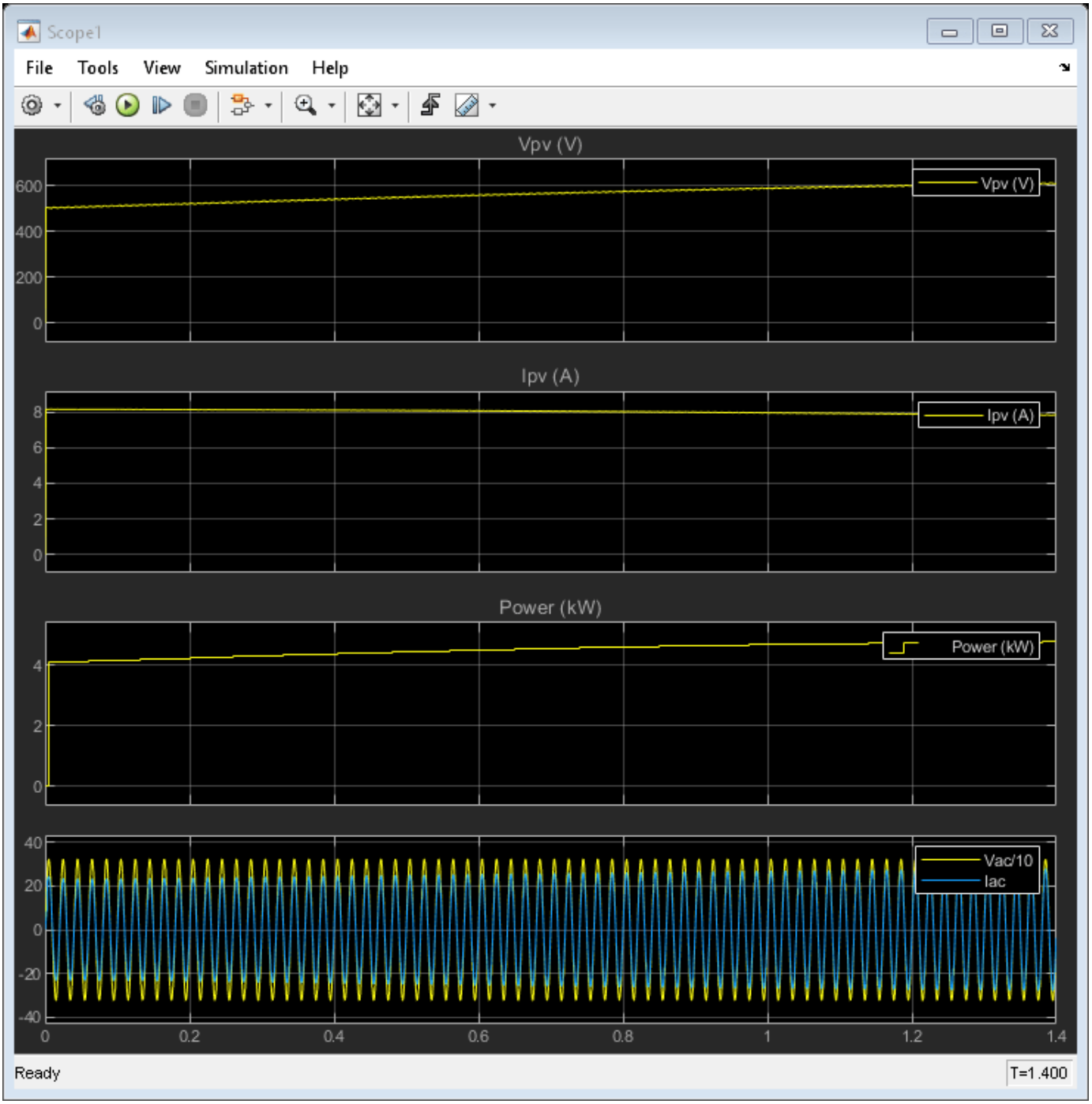
```
*********************************************************************************************
****          For the given solar panel, estimated boostless PV plant parameters          ****
*********************************************************************************************
*** Power rating input from the user = 4.70 kW
*** Minimum number of panel required per string = 17
*** Maximum number of panel connected per string without reaching maximum system voltage = 27
*** Minimum power rating of the boost-less solar PV plant = 3.83 kW
*** Maximum power possible per string without reaching maximum DC voltage = 6.08 kW
*** Actual number of panel per string = 21
*** Number of strings connected in parallel = 1
*** Actual solar PV plant power = 4.73 kW
*********************************************************************************************
```




**Single-Phase Grid-Connected Solar Photovoltaic System**

1. Open solar plant configuration script (load data)
2. Linearize the solar plant (see code)
3. Configure MPPT method: P&O, Incremental Conductance
4. Configure inverter: Average, Continuous, Discrete
5. PV system protection: Enable, Disable
6. Explore simulation results using Simscape Results Explorer
7. Learn more about this example

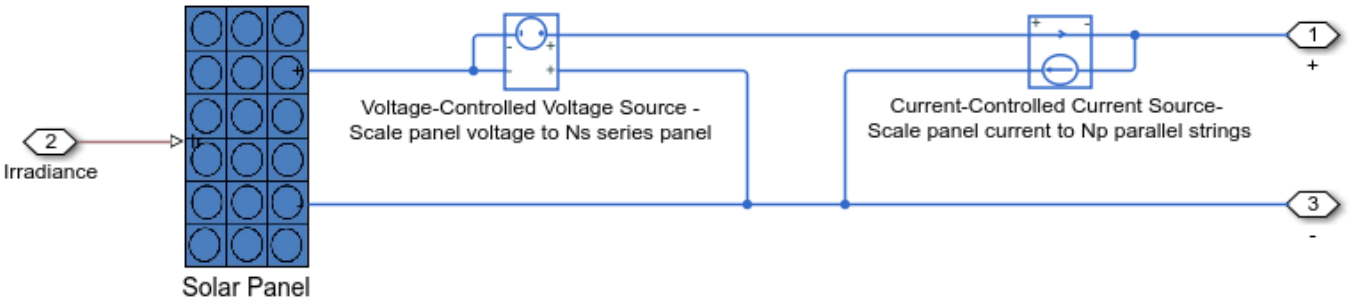
Copyright 2008-2022 The MathWorks, Inc.




### Solar Plant Subsystem

The solar plant subsystem models a solar plant that contains parallel-connected strings of solar panels. The solar panel is modeled using the Solar Cell block from the Simscape™ Electrical™ library. The number of series-connected solar panels in a string is estimated based on supply voltage, voltage drop across the line inductor, supply voltage fluctuation, open circuit voltage dependence on temperature and irradiance. The number of solar panel strings connected in parallel is estimated based on the plant power rating. Connecting multiple panels can slow the simulation because it

increases the number of elements in a model. By assuming uniform irradiance and temperature across all the solar panels, the number of solar elements can be reduced by using the controlled current and voltage sources as shown in the solar panel subsystem.



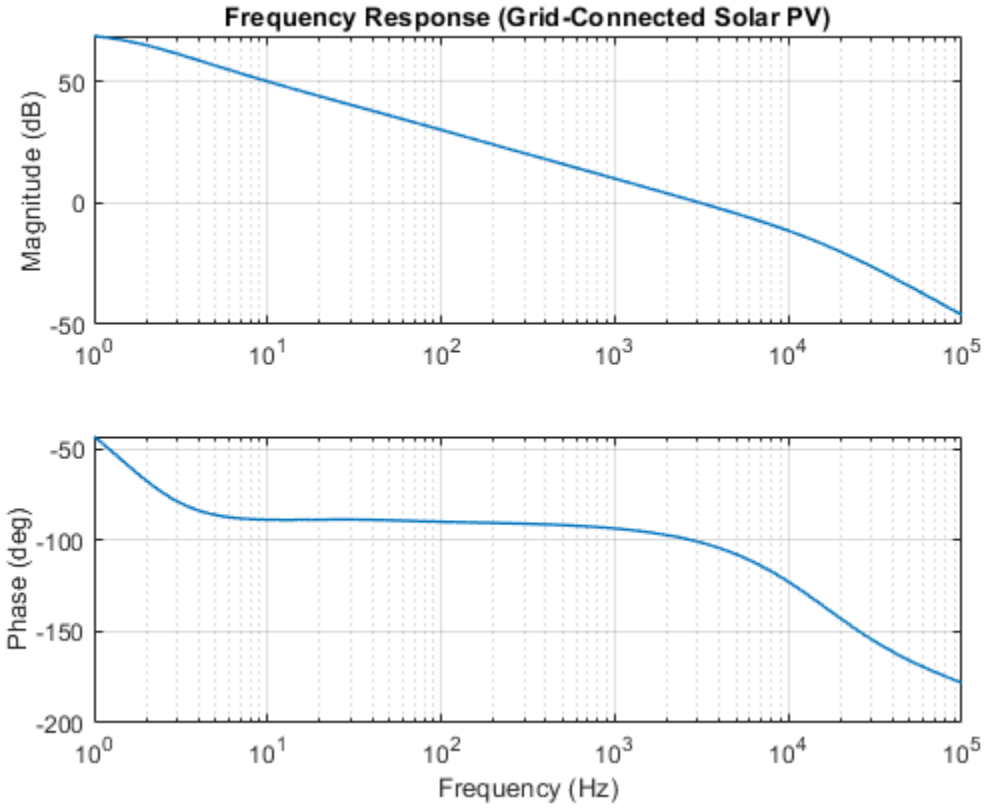

### Maximum Power Point Tracking (MPPT)

Two MPPT techniques are implemented. By using the variant variable 'MPPT', you can choose incremental conductance MPPT or perturbation and observation MPPT. For perturbation and observation, set the variable MPPT to zero and MPPT to one for incremental conductance. The maximum power point tracking block influences the current that is supplied to the grid. To increase the voltage across the PV panel string, current supplied to the grid is reduced to inject more current into the DC bus capacitor.

### Open Loop Bode Plot

Before linearizing the system, to disconnect the MPPT outer loop and break the current inner current loop, set the workspace variable 'closeLoop' to zero and use the average inverter model.

To use an average mode inverter, set the variant workspace variable 'powerCircuit' to zero.


Frequency Response (Grid-Connected Solar PV)
Magnitude (dB)
50
0
-50
Phase (deg)
-50
-100
-150
-200
$10^0$
$10^1$
$10^2$
$10^3$
$10^4$
$10^5$
Frequency (Hz)

# Three-Phase Grid-Connected Solar Photovoltaic System

This example shows how to model a three-phase grid-connected solar photovoltaic (PV) system. This example supports design decisions about the number of panels and the connection topology required to deliver the target power. The model represents a grid-connected rooftop solar PV that is implemented without an intermediate DC-DC converter. To parameterize the model, the example uses information from a solar panel manufacturer datasheet. Solar power is injected into the grid with unity power factor (UPF).

To track the maximum power point (MPP), the example uses these maximum power point tracking (MPPT) techniques:

- Incremental conductance
- Perturbation and observation

Three inverter options are available:

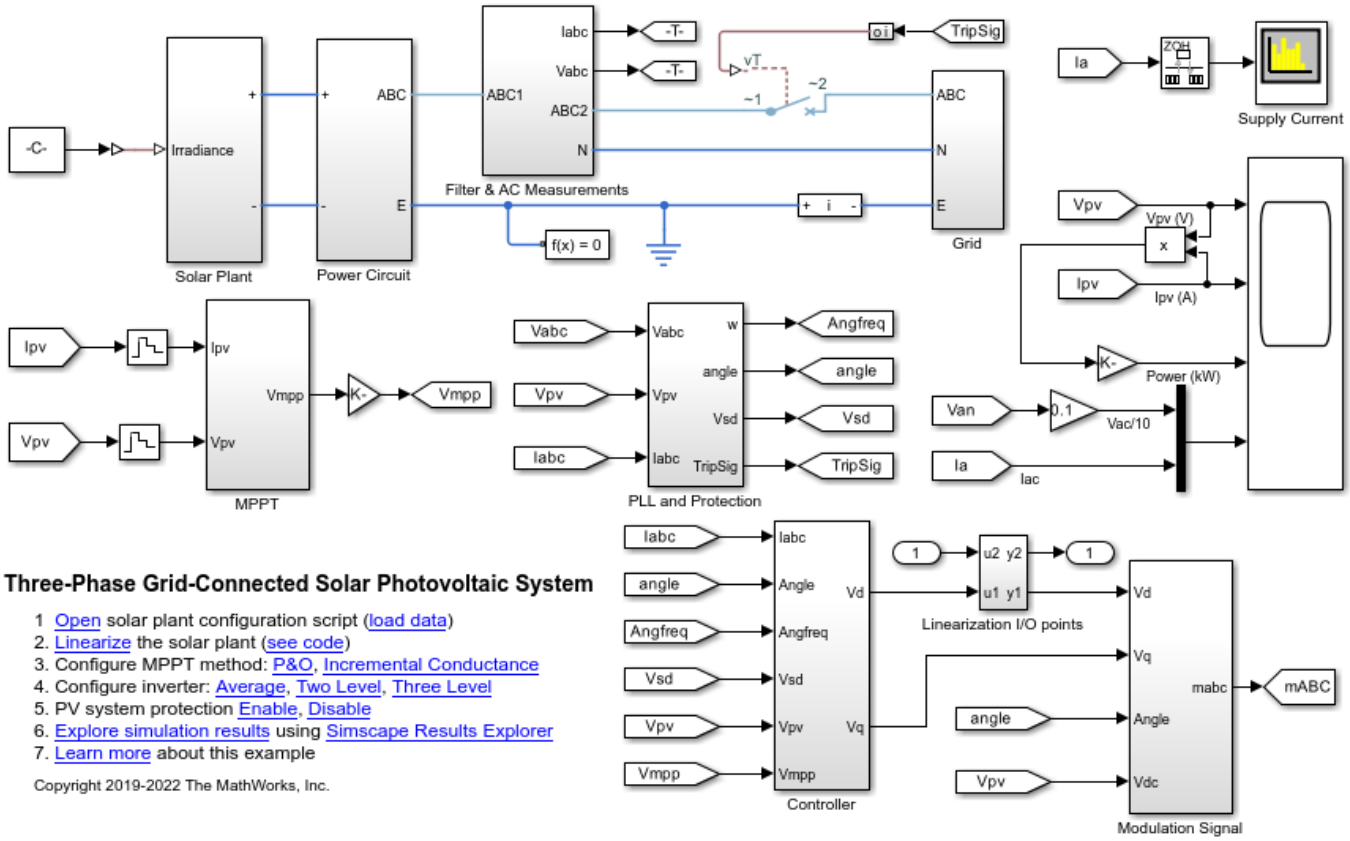
- Average
- Two-level
- Three-level

This example linearizes the system to generate an open-loop Bode plot from which phase and gain margin can be determined.

To open a script that provides information on the parameterization, features, and options in this example, at the MATLAB® command line, enter: edit 'ee_solar_gridconnected_threephase_data'

### PV System Model

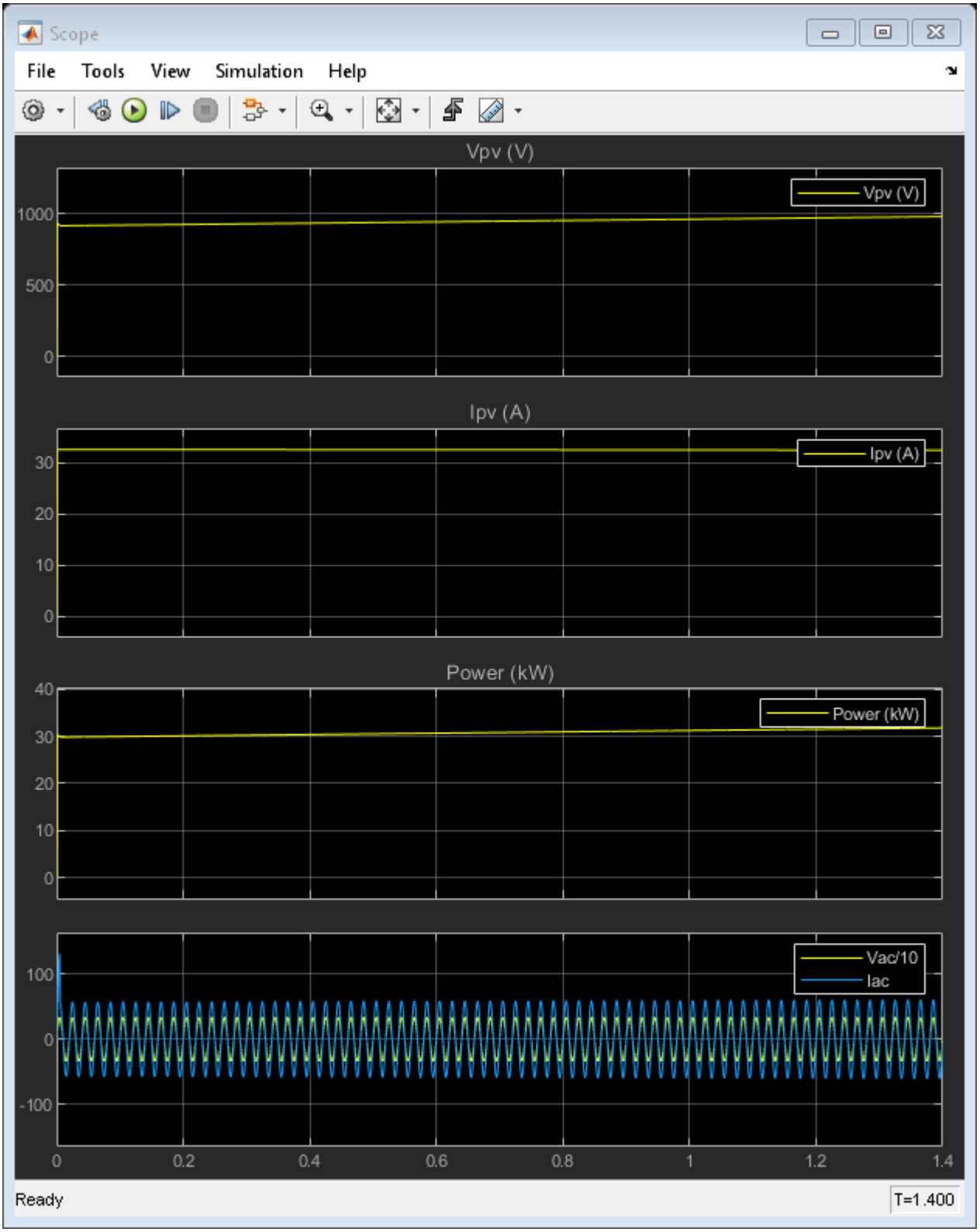
```
*********************************************************************************************
****          For the given solar panel, estimated boostless PV plant parameters          ****
*********************************************************************************************
*** Power rating input from the user = 35.00 kW
*** Minimum number of panel required per string = 33
*** Maximum number of panel connected per string without reaching maximum system voltage = 41
*** Minimum power rating of the boost-less solar PV plant = 7.43 kW
*** Maximum power possible per string without reaching maximum DC voltage = 9.23 kW
*** Actual number of panel per string = 39
*** Number of strings connected in parallel = 4
*** Actual solar PV plant power = 35.12 kW
*********************************************************************************************
```




**Three-Phase Grid-Connected Solar Photovoltaic System**

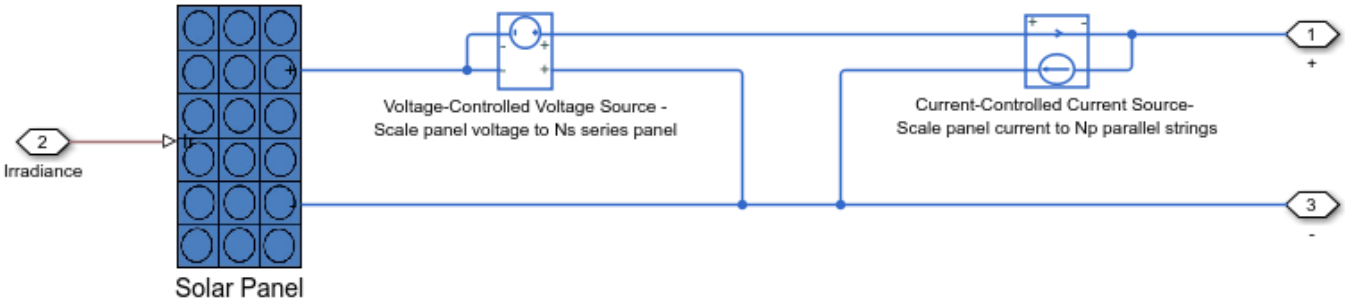
1. Open solar plant configuration script (load data)
2. Linearize the solar plant (see code)
3. Configure MPPT method: P&O, Incremental Conductance
4. Configure inverter: Average, Two Level, Three Level
5. PV system protection Enable, Disable
6. Explore simulation results using Simscape Results Explorer
7. Learn more about this example

Copyright 2019-2022 The MathWorks, Inc.


Scope
File Tools View Simulation Help
Vpv (V)
Vpv (V)
1000
500
0
Ipv (A)
Ipv (A)
30
20
10
0
Power (kW)
Power (kW)
40
30
20
10
0
Vac/10
Iac
100
0
-100
0 0.2 0.4 0.6 0.8 1 1.2 1.4
Ready
T=1.400

### Solar Plant Subsystem

The solar plant subsystem models a solar plant that contains parallel-connected strings of solar panels. The solar panel is modeled using the Solar Cell block from the Simscape™ Electrical™ library. The number of series-connected solar panels in a string is estimated based on supply voltage, voltage drop across the line inductor, supply voltage fluctuation, open circuit voltage dependence on temperature and irradiance. The number of solar panel strings connected in parallel is estimated based on the plant power rating. Connecting multiple panels can slow the simulation because it increases the number of elements in a model. By assuming uniform irradiance and temperature across all the solar panels, the number of solar elements can be reduced by using the controlled current and voltage sources as shown in the solar panel subsystem. Parasitic capacitance of solar panel is modelled using two lumped capacitors connected in both the terminal of the solar plant.



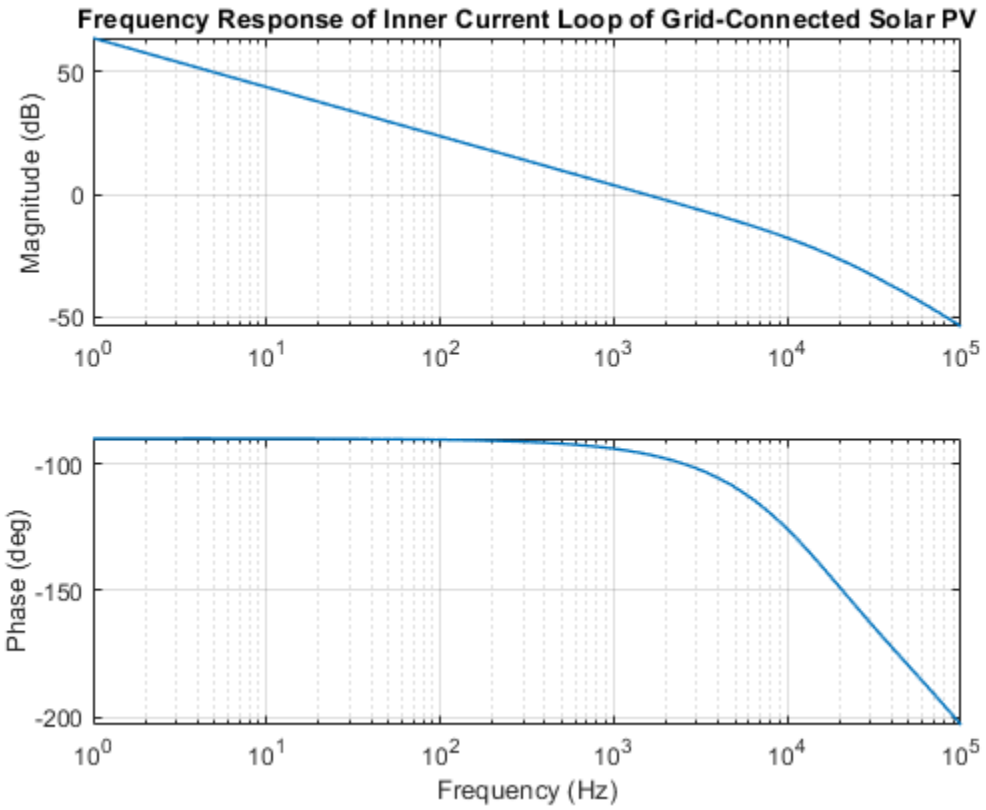

### Maximum Power Point Tracking (MPPT)

Two MPPT techniques are implemented. By using the variant variable 'MPPT', you can choose incremental conductance MPPT or perturbation and observation MPPT. For perturbation and observation, set the variable MPPT to zero and MPPT to one for incremental conductance. The voltage controller that tracks the maximum power point influences the current that is supplied to the grid.

### Open Loop Bode Plot

Before linearizing the system, to disconnect the MPPT outer loop and break the current inner current loop, set the workspace variable 'closeLoop' to zero and use the average inverter model.

To use an average mode inverter, set the variant workspace variable 'powerCircuit' to zero.


Frequency Response of Inner Current Loop of Grid-Connected Solar PV
Magnitude (dB)
50
0
-50
Phase (deg)
-100
-150
-200
$10^0$
$10^1$
$10^2$
$10^3$
$10^4$
$10^5$
Frequency (Hz)

# Solar PV System with MPPT Using Boost Converter

This example shows the design of a boost converter for controlling the power output of a solar PV system and helps you to:

- Determine how the panels should be arranged in terms of the number of series-connected strings and the number of panels per string to achieve the required power rating.
- Implement the MPPT algorithm using boost converter.
- Operate the solar PVsystem in the voltage control mode.
- Select a suitable `proportional` gain $(K_v)$ and `phase-lead time constant` $(T_v)$ for the PI controller, $\frac{k_v(sT_v+1)}{sT_v}$.

The DC load is connected across the boost converter output. The solar PV system operates in both maximum power point tracking and de-rated voltage control modes. To track the maximum power point (MPP) of the solar PV, you can choose between two maximum power point tracking (MPPT) techniques:

- Incremental conductance
- Perturbation and observation

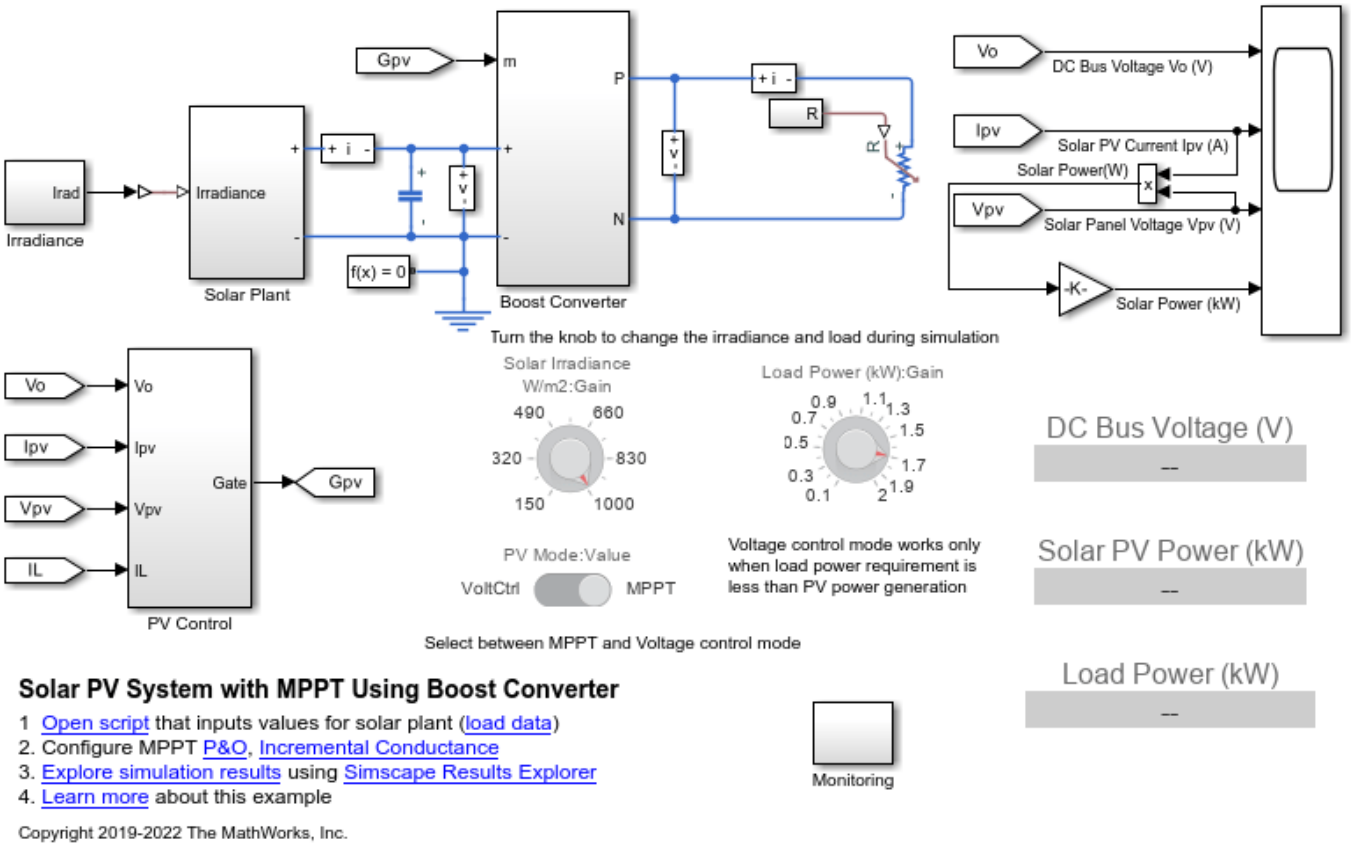
You can specify the output DC bus voltage, solar PV system operating temperature and solar panel specification. Solar panel manufacturer data is used to determine number of PV panels required to deliver the specified generation capability.

### Solar PV System with MPPT Using Boost Converter

To open a script that designs the Solar PV System with MPPT Using Boost Converter, at the MATLAB® command line, enter: edit 'ee_solar_boostconverter_maxpowerpoint_data'

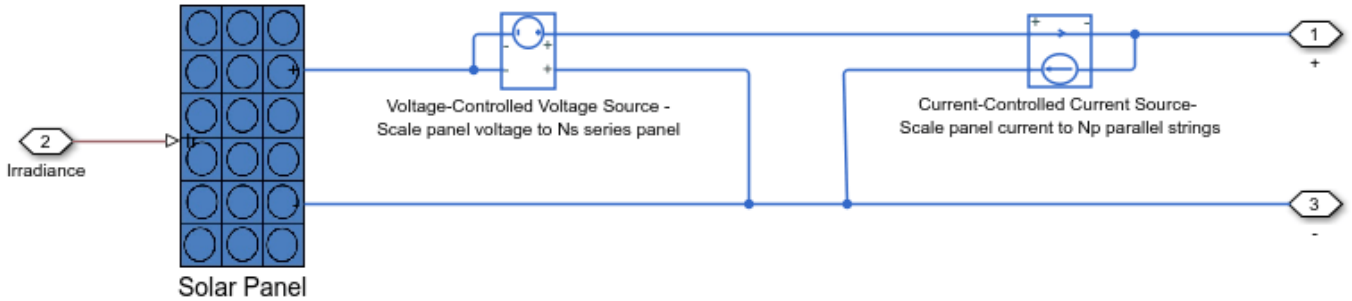
The chosen solar PV plant parameters are:

```
*********************************************************************************************
****                PV Plant Parameters for the Specified Solar Panel                  ****
*********************************************************************************************
*** Power rating input from the user  =  2.00 kW
*** Minimum number of panel required per string  =  8
*** Maximum number of panel connected per string without reaching maximum voltage  =  10
*** Minimum power rating of the solar PV plant  =  1.80 kW
*** Maximum power possible per string without reaching maximum DC voltage  =  2.25 kW
*** Actual number of panel per string  =  9
*** Number of strings connected in parallel  =  1
*** Actual solar PV plant power  =  2.03 kW
*********************************************************************************************
```




### Solar Plant Subsystem

The solar plant subsystem models a solar plant that contains parallel-connected strings of solar panels. The solar panel is modeled using the Solar Cell block from the Simscape™ Electrical™ library. Given the specified DC bus voltage, solar cell characteristics and specified power rating, a calculation is made of the solar panel string length and number of parallel-connected strings. Connecting multiple panels can slow the simulation because it increases the number of elements in a model. By assuming uniform irradiance and temperature across all the solar panels, it is possible to reduce the number of solar elements by using the controlled current and voltage sources as shown in the solar panel subsystem.

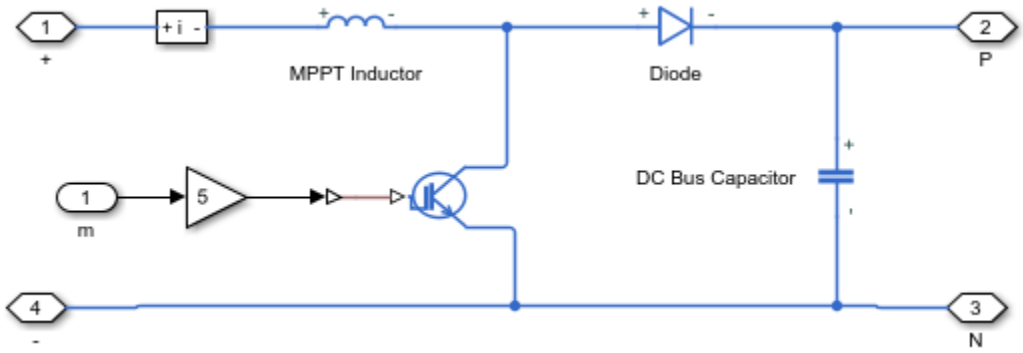


### Maximum Power Point Tracking (MPPT)

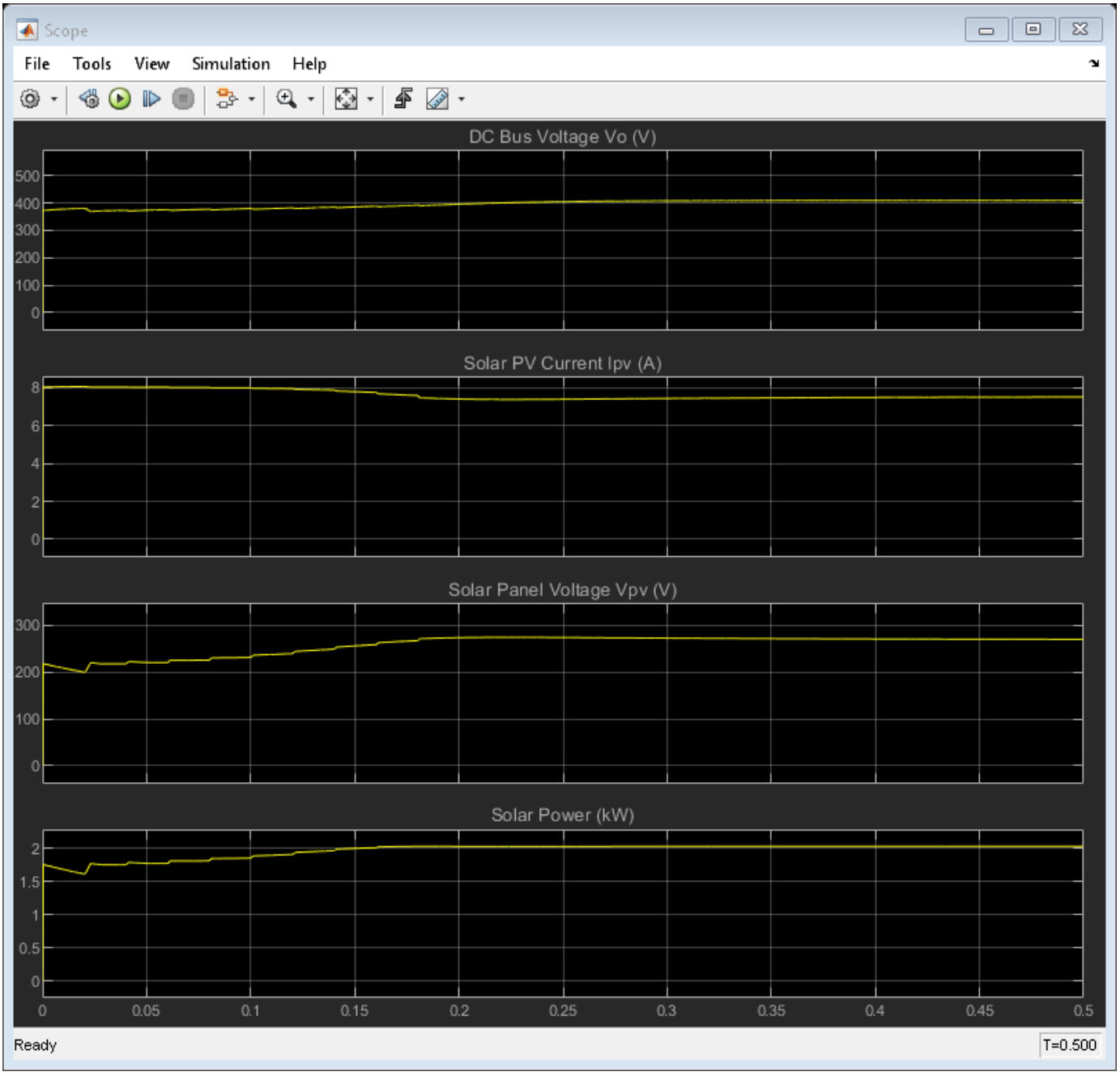
Two MPPT techniques are implemented using the variant subsystem. Set the variant variable MPPT to 0 to choose the perturbation and observation MPPT method. Set the variable MPPT to 1 to choose the incremental conductance method.

### Intermediate Boost DC-DC Converter

A boost DC-DC converter is used to control the solar PV power. The boost converter operates in both MPPT mode and voltage control mode. The voltage control mode is used only when load power is less than the maximum power generated by solar PV plant given the incident irradiance and panel temperature.



### Simulation Output (MPPT Mode)



# Stand-Alone Solar PV DC Power System with Battery Backup

This example shows the design of a stand-alone PV DC power system with battery backup and helps you to:

- Choose the necessary battery rating based on the connected load profile and available solar power.
- Determine how the panels should be arranged in terms of the number of series-connected strings and the number of panels per string.
- Select a proper PI controller proportional gain, $(K_v)$, and phase-lead constant, $(T_v)$.

Both solar PV and battery storage support stand-alone loads. The load is connected across the constant DC output. A solar PV system operates in both maximum power point tracking and de-rated voltage control modes. The battery management system uses bi-directional DC-DC converters.

A stand-alone PV system requires six normal operating modes based on the solar irradiance, generated solar power, connected load, state of charge of the battery, maximum battery charging and discharging current limits.

To track the maximum power point (MPP) of solar PV, you can choose between two maximum power point tracking (MPPT) techniques:

- Incremental conductance
- Perturbation and observation

You can specify the average daily connected load profile, region's daily available average solar energy (kWhr), solar PV system operating temperature, day of autonomy, battery recharge time, output DC voltage and solar panel specification. solar panel manufacturer data is used to determine number of PV panels required to deliver the specified generation capability.

PI controller of the form $\frac{K_v(sT_v+1)}{sT_v}$ is chosen to control the solar PV and battery management system (BMS).

This example uses:

- A MATLAB® live script to design the overall standalone PV system.
- Simulink® to design/simulate the control logic for the system.
- Simscape™ to simulate the power circuit.
- Stateflow™ to implement the supervisory control logic.

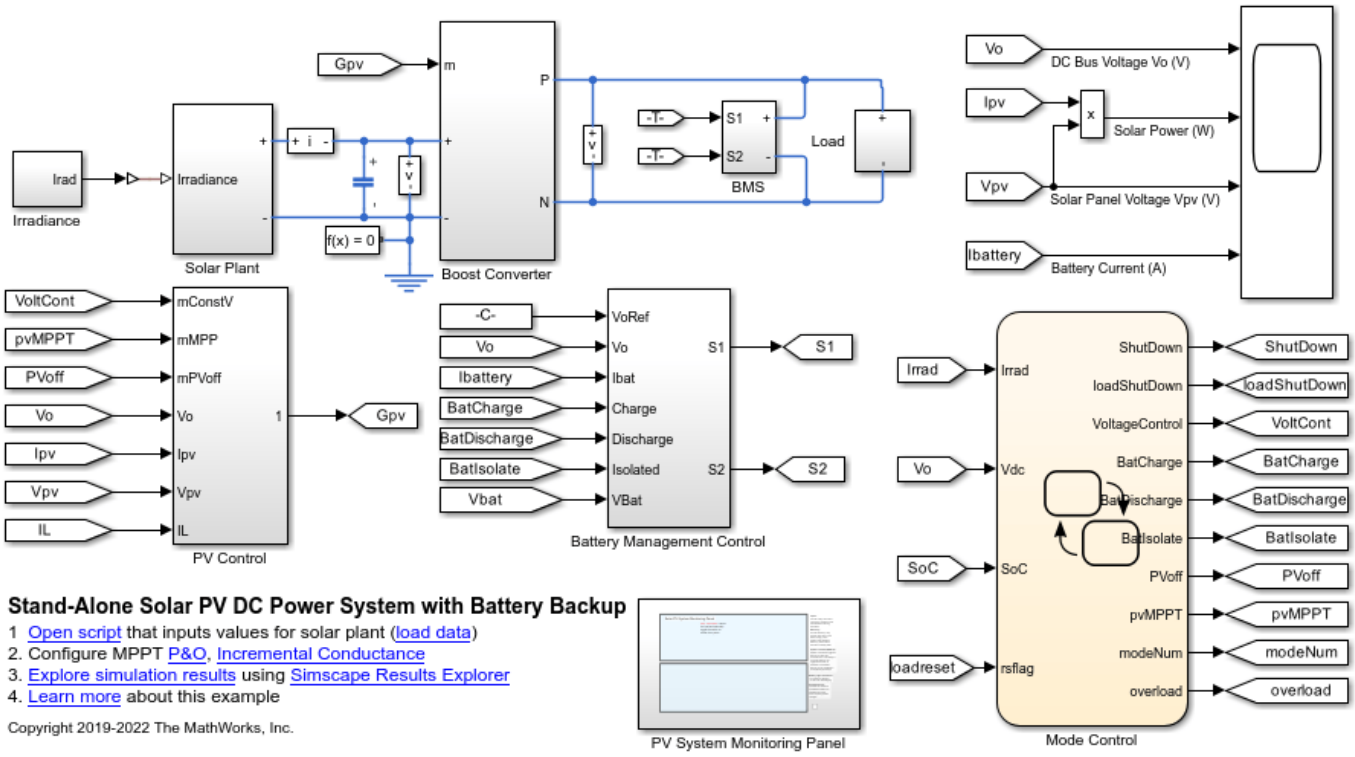
### Stand-Alone PV DC Power System Model

To open a script that designs the standalone PV DC power system, at the MATLAB command line, enter: edit 'ee_solar_standalone_dcsystem_withbatterybackup_data'

The chosen battery and solar PV plant parameters are:

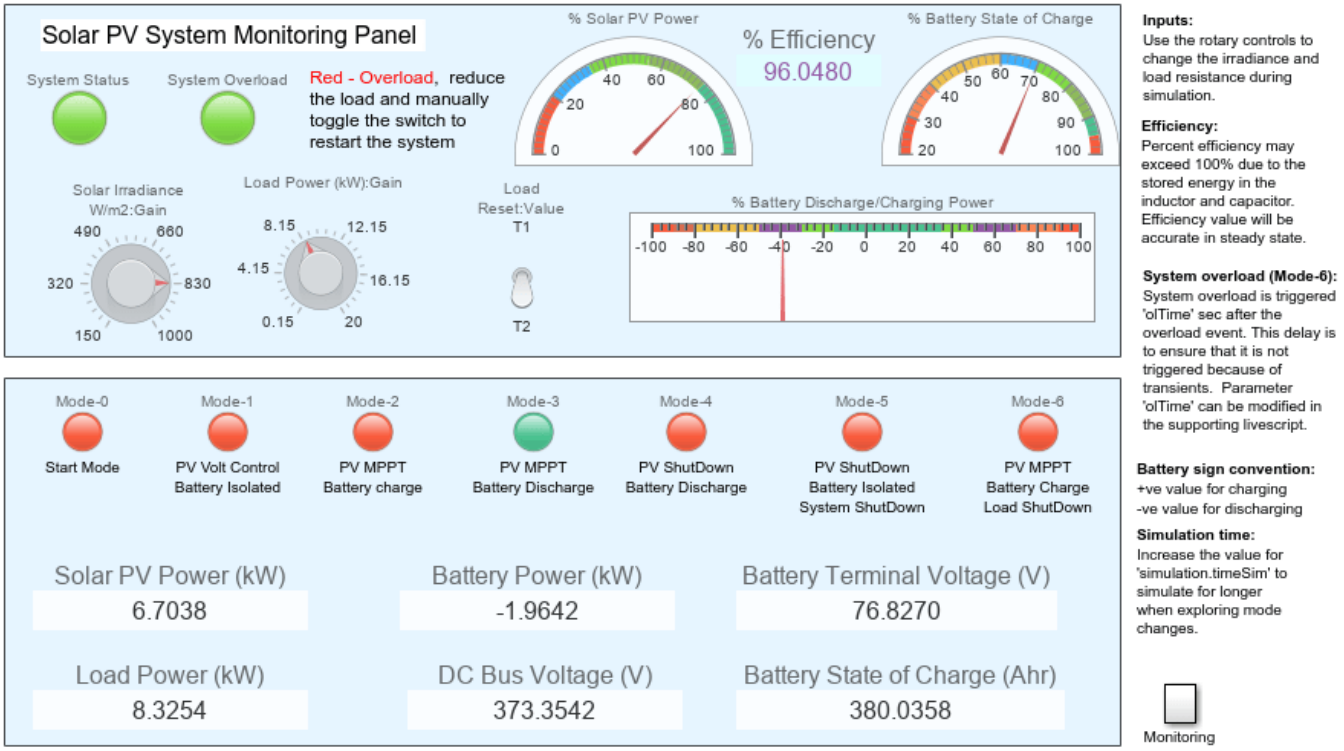
```
*********************************************************************************************
****          For the Given Stand-Alone PV System, Battery Sizing Parameters           ****
*********************************************************************************************
```

```
*** Calculated amphr of the battery  =  542.91 Ahr
*** Battery nominal voltage  =  78 V
*** Battery voltage at 80% discharge  =  70.20 V
*** Number of required battery cell  =  39.00
*** Average discharge current  =  4.28 A
*********************************************************************************************
*********************************************************************************************
****          For the Given Solar Panel, PV Plant Parameters           ****
*********************************************************************************************
*** Required PV Power rating  =  9.36 kW
*** Minimum number of panels required per string  =  8
*** Maximum number of panels connected per string without reaching maximum voltage  =  10
*** Minimum power rating of the solar PV plant  =  1.80 kW
*** Maximum power possible per string without reaching maximum DC voltage  =  2.25 kW
*** Actual number of panels per string  =  8
*** Number of strings connected in parallel  =  5
*** Actual solar PV plant power  =  9.01 kW
*********************************************************************************************
*********************************************************************************************
****                    Battery Charging/Discharging Parameters                          ****
*********************************************************************************************
Reference battery charging current = 45.24 A
Maximum battery charging current = 128.29 A
Maximum battery discharging current = 64.14 A
Maximum battery charging Power = 10.01 kW
Maximum battery discharging Power = 5.00 kW
*********************************************************************************************
```



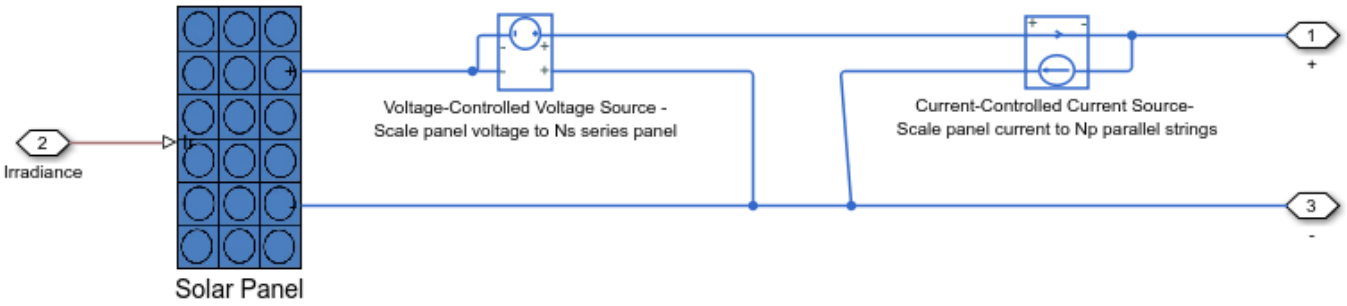
### Stand-Alone Solar PV DC Power System Monitoring Panel

This example uses the Simulink Dashboard feature to display all the real time system parameters. Turn the dashboard knob in the monitoring panel to modify the solar irradiance and the load during the simulation. By changing these parameters, you can observe how the PV system switches between the operating modes.




### Solar Plant Subsystem

The solar plant subsystem models a solar plant that contains parallel-connected strings of solar panels. The solar panel is modeled using the Solar Cell block from the Simscape Electrical library. This example uses the output voltage from the DC bus, open circuit voltage depending on temperature and irradiance to estimate the number of solar panel strings connected in series, and the plant power rating to estimate the number of solar panel strings connected in parallel. Connecting multiple panels can slow the simulation because it increases the number of elements in a model. By assuming uniform irradiance and temperature across all the solar panels, it is possible to reduce the number of solar elements by using the controlled current and voltage sources as shown in the solar panel subsystem.

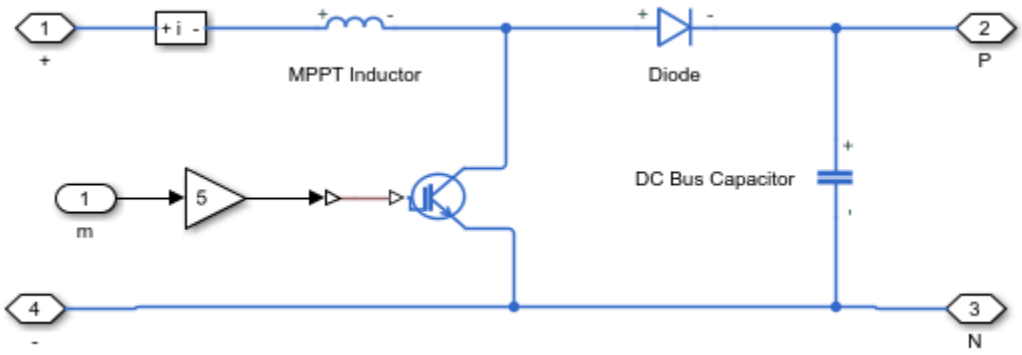



### Maximum Power Point Tracking (MPPT)

Two MPPT techniques are implemented using the variant subsystem. Set the variant variable MPPT to 0 to choose the perturbation and observation MPPT. Set the variable MPPT to 1 to choose incremental conductance.

### Intermediate Boost DC-DC Converter

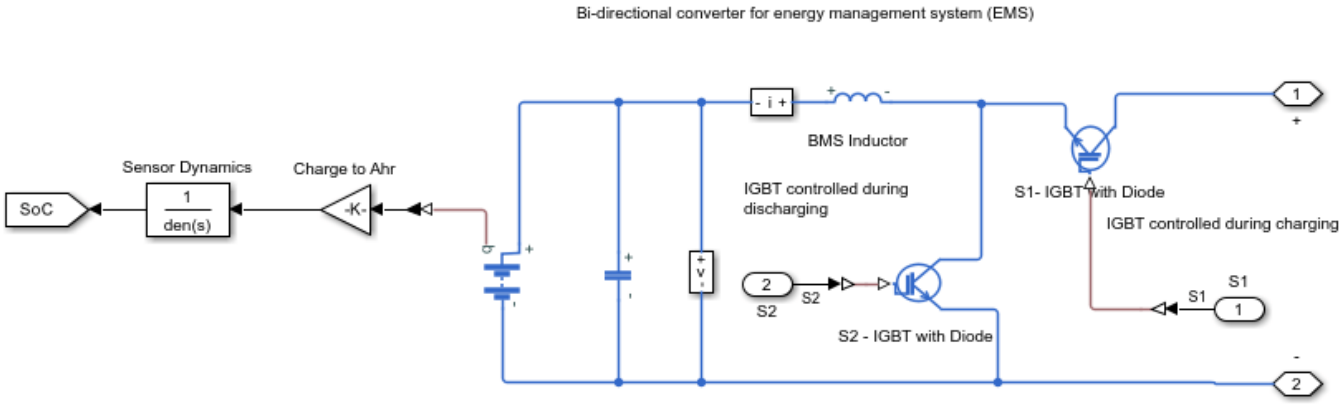
Boost DC-DC converter is used to control the solar PV power. When battery is not fully charged solar PV plant operated in maximum power point. When battery is fully charged and load is smaller than the PV power, solar PV is operated in constant output DC bus voltage control mode.




### Battery Management System (BMS)

The battery management system uses a bi-directional DC-DC converter. The battery is charged by the buck converter configuration and it is discharged using the boost converter configuration. To improve battery performance and life cycle, systems with battery backup have limited maximum battery charging and discharging current. This example sets a limit on the maximum amount of power that a battery can supply to the load and absorb from the solar PV source. Here, the maximum charging power is equal to the solar plant capacity at the standard test condition. The chosen maximum charging power should be able to recharge the battery sooner than the battery recharge time specified by the user

Here, separate controller is used for charging and discharging operation. BMS controller have two loops, an outer voltage loop and inner current loop.




### Supervisory Control(Mode Control) Parameters

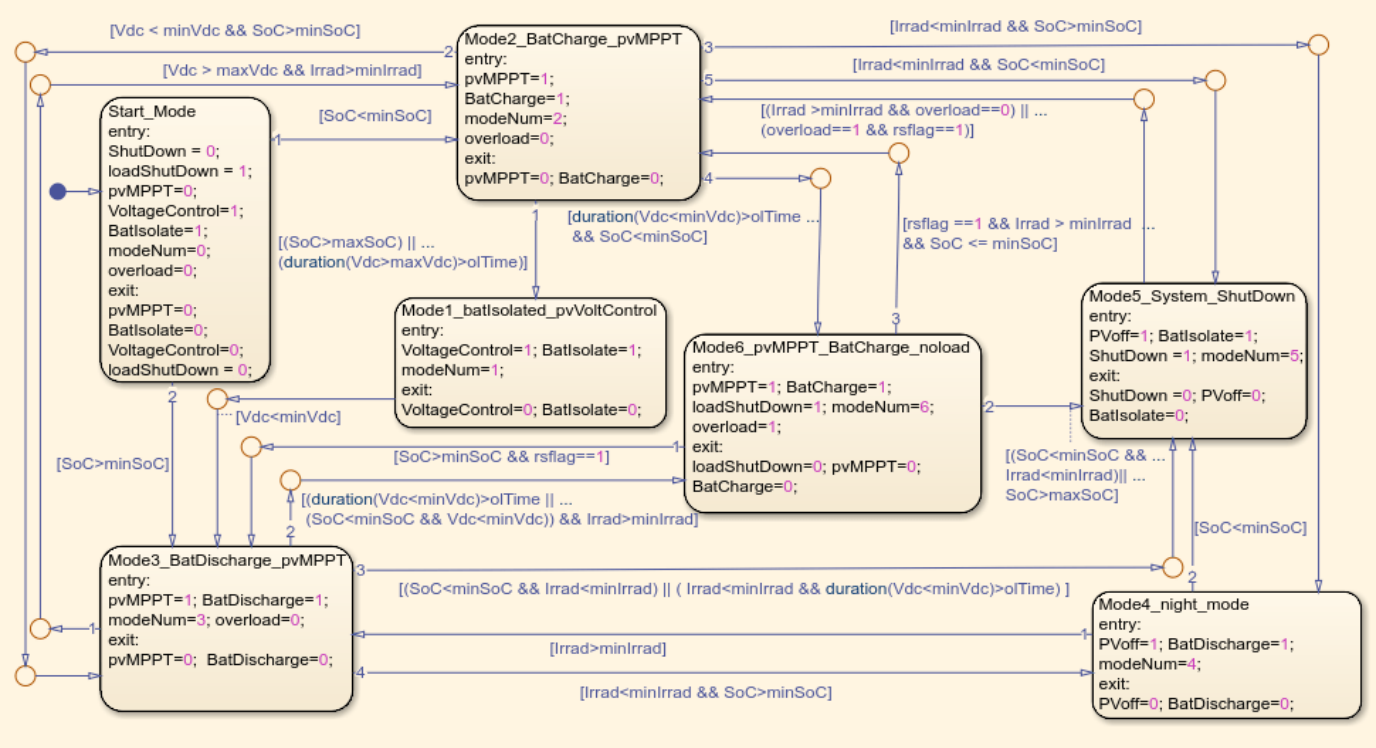
Stand-alone PV system in this example comprises seven operating modes. These modes are selected based on DC bus voltage, solar irradiance and state of charge of the battery. The DC bus voltage level is used as a measure to detect a load imbalance. If the DC bus voltage is greater than $V_{dc-max}$, the system is generating more power than what the load is requiring. If the DC bus voltage is less than $V_{dc-min}$, then the load is requiring more power than what the system is generating.

DC bus voltage level $(V_{dc})$, solar irradiance $(Irrad)$ and the battery state of charge $(SoC)$ are used to decide the suitable operating mode.

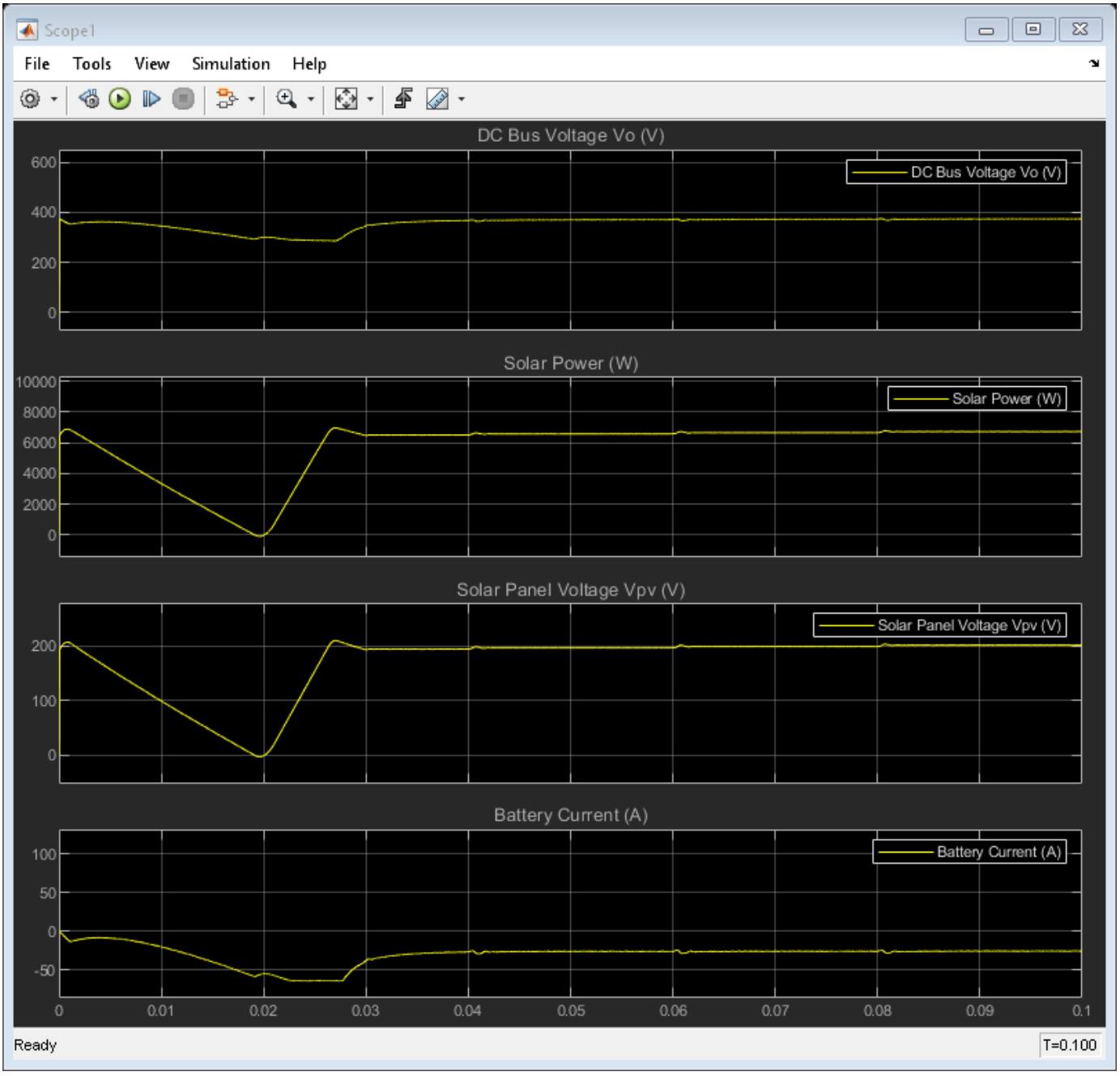
Operating modes of the stand-alone PV DC System are:

- Mode-0 - Start mode (Default simulation starting mode)
- Mode-1 - PV in output voltage control, battery fully charged and isolated
- Mode-2 - PV in maximum power point, battery is charging
- Mode-3 - PV in maximum power point, battery is discharging
- Mode-4 - Night mode, PV shutdown, battery is discharging
- Mode-5 - Total system shutdown
- Mode-6 - PV in maximum power point, battery is charging, load is disconnected

Stateflow mode control diagram


[Vdc < minVdc && SoC>minSoC]
[Vdc > maxVdc && Irrad>minIrrad]
Start_Mode
entry:
ShutDown = 0;
loadShutDown = 1;
pvMPPT=0;
VoltageControl=1;
BatIsolate=1;
modeNum=0;
overload=0;
exit:
pvMPPT=0;
BatIsolate=0;
VoltageControl=0;
loadShutDown = 0;
[SoC<minSoC]
Mode2_BatCharge_pvMPPT
entry:
pvMPPT=1;
BatCharge=1;
modeNum=2;
overload=0;
exit:
pvMPPT=0; BatCharge=0;
[Irrad<minIrrad && SoC>minSoC]
[Irrad<minIrrad && SoC<minSoC]
[(Irrad >minIrrad && overload==0) || ...
(overload==1 && rsflag==1)]
[(SoC>maxSoC) || ...
(duration(Vdc>maxVdc)>olTime)]
[duration(Vdc<minVdc)>olTime ...
&& SoC<minSoC]
[rsflag ==1 && Irrad > minIrrad ...
&& SoC <= minSoC]
Mode1_batIsolated_pvVoltControl
entry:
VoltageControl=1; BatIsolate=1;
modeNum=1;
exit:
VoltageControl=0; BatIsolate=0;
Mode6_pvMPPT_BatCharge_noload
entry:
pvMPPT=1; BatCharge=1;
loadShutDown=1; modeNum=6;
overload=1;
exit:
loadShutDown=0; pvMPPT=0;
BatCharge=0;
Mode5_System_ShutDown
entry:
PVoff=1; BatIsolate=1;
ShutDown =1; modeNum=5;
exit:
ShutDown =0; PVoff=0;
BatIsolate=0;
[Vdc<minVdc]
[SoC>minSoC]
[SoC>minSoC && rsflag==1]
[(duration(Vdc<minVdc)>olTime || ...
(SoC<minSoC && Vdc<minVdc)) && Irrad>minIrrad]
[(SoC<minSoC && ...
Irrad<minIrrad)|| ...
SoC>maxSoC]
[SoC<minSoC]
Mode3_BatDischarge_pvMPPT
entry:
pvMPPT=1; BatDischarge=1;
modeNum=3; overload=0;
exit:
pvMPPT=0; BatDischarge=0;
[(SoC<minSoC && Irrad<minIrrad) || ( Irrad<minIrrad && duration(Vdc<minVdc)>olTime) ]
[Irrad>minIrrad]
[Irrad<minIrrad && SoC>minSoC]
Mode4_night_mode
entry:
PVoff=1; BatDischarge=1;
modeNum=4;
exit:
PVoff=0; BatDischarge=0;

### Simulation Output



# Stand-Alone Solar PV AC Power System with Battery Backup

This example shows the design of a stand-alone PV AC power system with battery backup and helps you to:

- Choose the necessary battery rating based on the connected load profile and available solar power.
- Determine how the panels should be arranged in terms of the number of series-connected strings and the number of panels per string.
- Helps to design constant voltage single-phase AC supply.
- Select a suitable value for PI controller proportional gain, $(K_v)$, and phase-lead constant, $(T_v)$.

Both solar PV and battery storage support stand-alone loads. The load is connected across the constant voltage single-phase AC supply. A solar PV system operates in both maximum power point tracking and de-rated voltage control modes. The battery management system uses bi-directional DC-DC converters.

A stand-alone PV system requires six normal operating modes based on the solar irradiance, generated solar power, connected load, state of charge of the battery, maximum battery charging and discharging current limits.

To track the maximum power point (MPP) of solar PV, you can choose between two maximum power point tracking (MPPT) techniques:

- Incremental conductance
- Perturbation and observation

You can specify the average daily connected load profile, region's daily available average solar energy (kWhr), solar PV system operating temperature, day of autonomy, battery recharge time, ac supply and solar panel specification. solar panel manufacturer data is used to determine number of PV panels required to deliver the specified generation capability.

A $\frac{K_v(sT_v+1)}{sT_v}$ PI controller is chosen to control the solar PV and battery management system (BMS).

This example uses:

- A MATLAB® live script to design the overall standalone PV system.
- Simulink® to design/simulate the control logic for the system.
- Simscape™ to simulate the power circuit.
- Stateflow™ to implement the supervisory control logic.

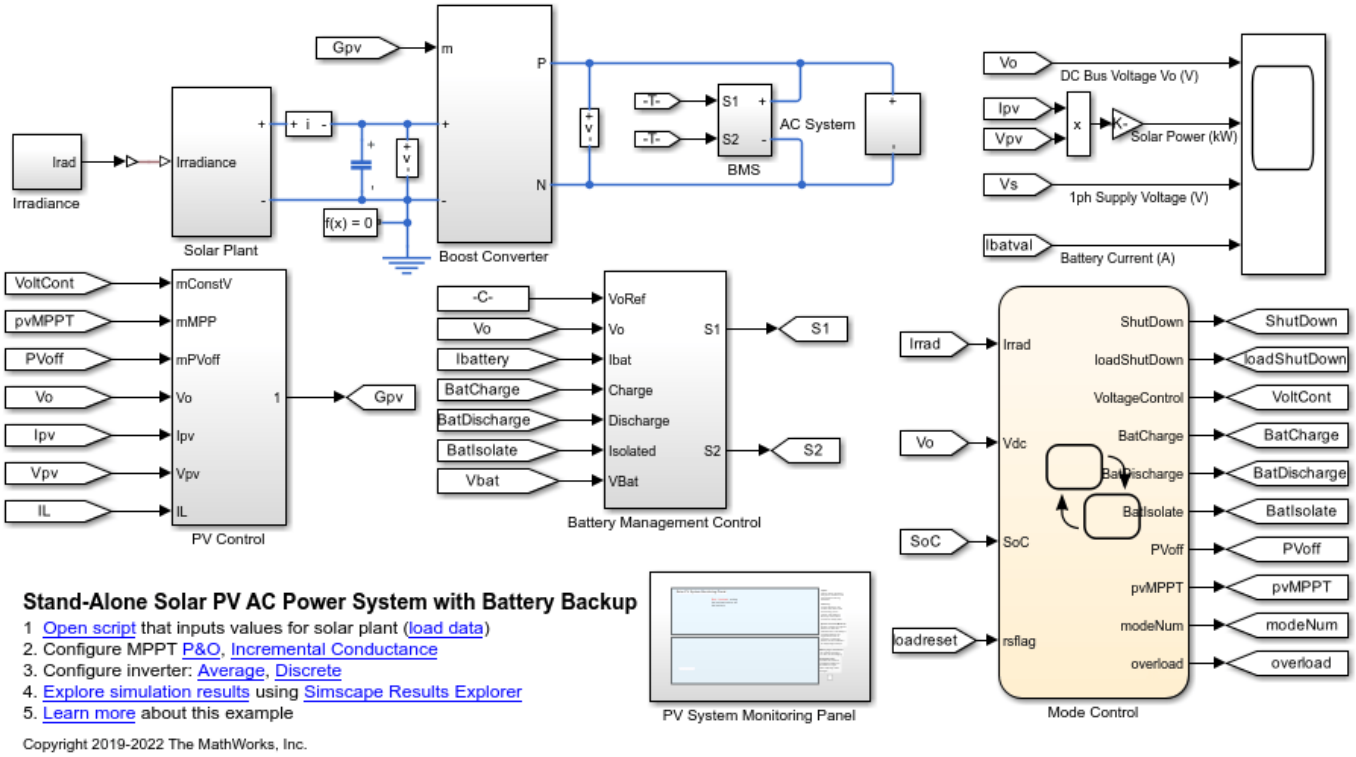
### Stand-Alone PV AC Power System Model

To open a script that designs the standalone PV AC power system, at the MATLAB command line, enter: edit 'ee_solar_standalone_acsystem_withbatterybackup_data'

The chosen battery and solar PV plant parameters are:

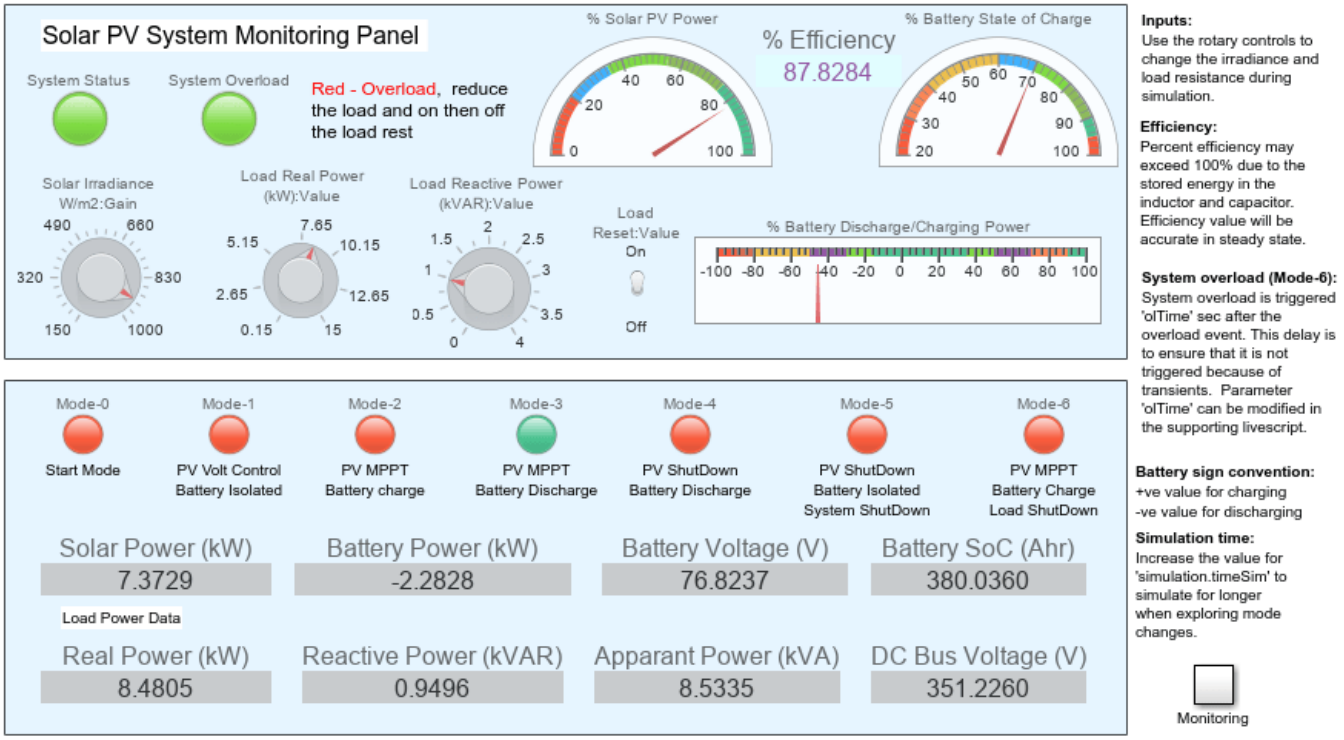
```
********************************************************************************************
****          For the Given Stand-Alone PV System, Battery Sizing Parameters          ****
```

```
*********************************************************************************************
*** Calculated amphr of the battery  =  542.91 Ahr
*** Battery nominal voltage  =  78 V
*** Battery voltage at 80% discharge  =  70.20 V
*** Number of required battery cell  =  39.00
*** Average discharge current  =  4.28 A
*********************************************************************************************
*********************************************************************************************
****          For the Given Solar Panel, PV Plant Parameters           ****
*********************************************************************************************
*** Required PV Power rating  =  9.36 kW
*** Minimum number of panels required per string  =  8
*** Maximum number of panels connected per string without reaching maximum voltage  =  10
*** Minimum power rating of the solar PV plant  =  1.80 kW
*** Maximum power possible per string without reaching maximum DC voltage  =  2.25 kW
*** Actual number of panels per string  =  8
*** Number of strings connected in parallel  =  5
*** Actual solar PV plant power  =  9.01 kW
*********************************************************************************************
*********************************************************************************************
****                    Battery Charging/Discharging Parameters                          ****
*********************************************************************************************
Reference battery charging current = 45.24 A
Maximum battery charging current = 128.29 A
Maximum battery discharging current = 64.14 A
Maximum battery charging Power = 10.01 kW
Maximum battery discharging Power = 5.00 kW
*********************************************************************************************
```



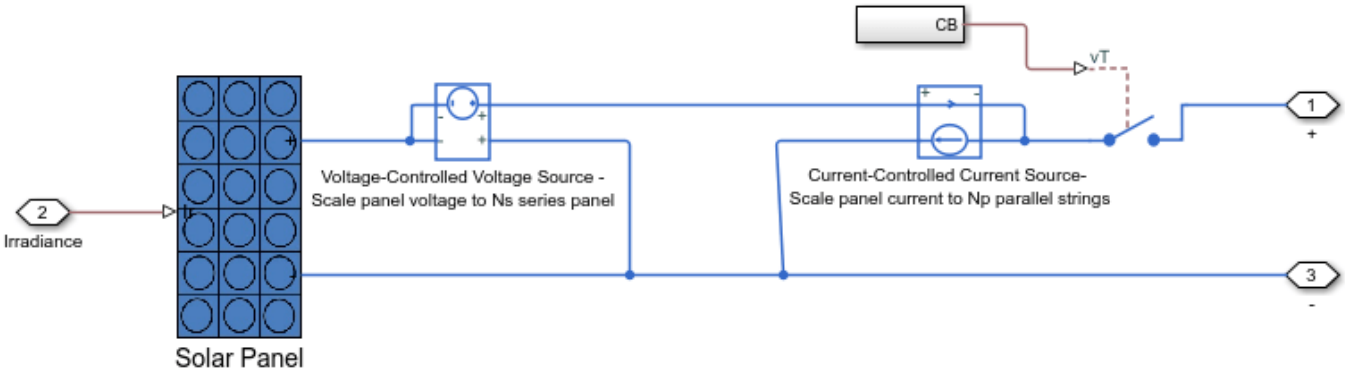
### Stand-Alone Solar PV AC Power System Monitoring Panel

This example uses the Simulink Dashboard feature to display all the real time system parameters. Turn the dashboard knob in the monitoring panel to modify the solar irradiance and the real and reactive power of the connected load during the simulation. By changing these parameters, you can observe how the PV system switches between the operating modes.




### Solar Plant Subsystem

The solar plant subsystem models a solar plant that contains parallel-connected strings of solar panels. The solar panel is modeled using the Solar Cell block from the Simscape Electrical library. This example uses the output voltage from the DC bus, open circuit voltage depending on temperature and irradiance to estimate the number of solar panel strings connected in series, and the plant power rating to estimate the number of solar panel strings connected in parallel. Connecting multiple panels can slow the simulation because it increases the number of elements in a model. By assuming uniform irradiance and temperature across all the solar panels, it is possible to reduce the number of solar elements by using the controlled current and voltage sources as shown in the solar panel subsystem.

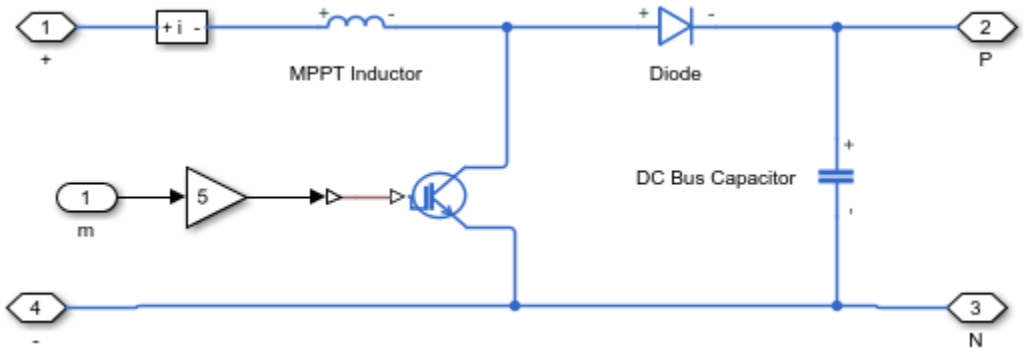



### Maximum Power Point Tracking (MPPT)

Two MPPT techniques are implemented using the variant subsystem. Set the variant variable MPPT to 0 to choose the perturbation and observation MPPT. Set the variable MPPT to 1 to choose incremental conductance.

### Intermediate Boost DC-DC Converter

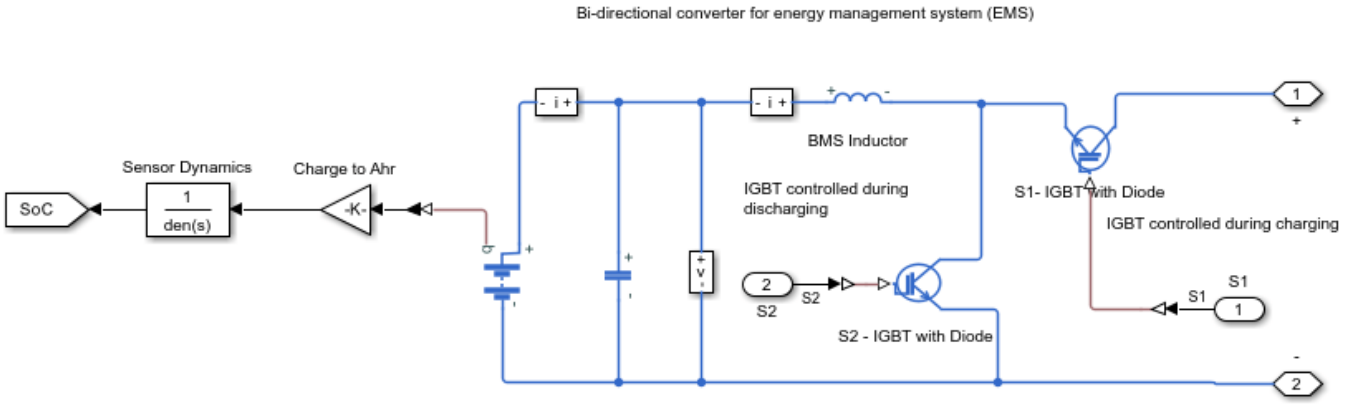
Boost DC-DC converter is used to control the solar PV power. When battery is not fully charged solar PV plant operated in maximum power point. When battery is fully charged and load is smaller than the PV power, solar PV is operated in constant output DC bus voltage control mode.




### Battery Management System (BMS)

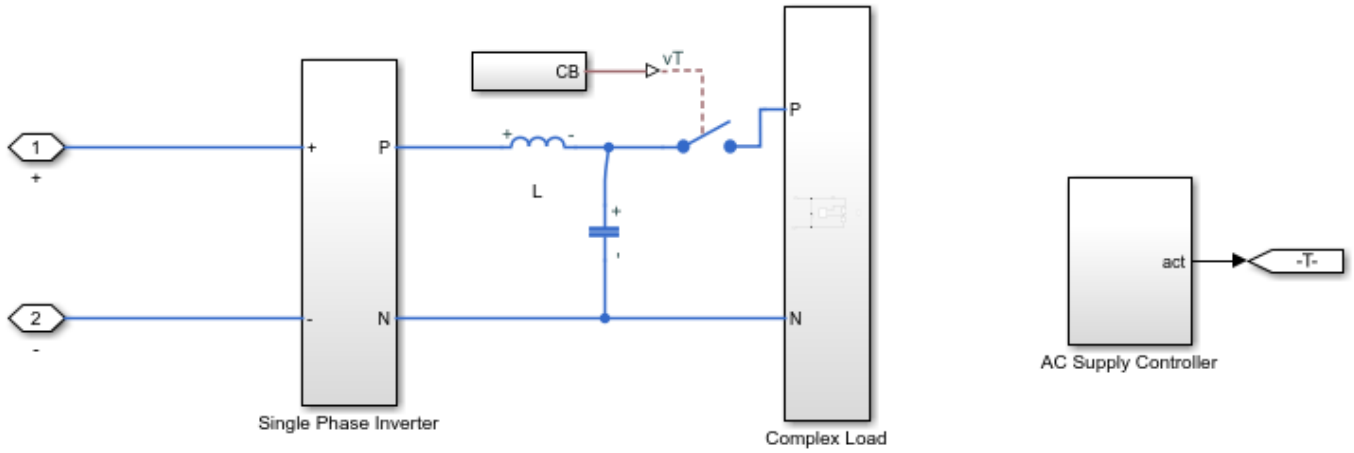
The battery management system uses a bi-directional DC-DC converter. The battery is charged by the buck converter configuration and it is discharged using the boost converter configuration. To improve battery performance and life cycle, systems with battery backup have limited maximum battery charging and discharging current. This example sets a limit on the maximum amount of power that a battery can supply to the load and absorb from the solar PV source. Here, the maximum charging power is equal to the solar plant capacity at the standard test condition. The chosen maximum charging power should be able to recharge the battery sooner than the battery recharge time specified by the user.

Here, separate controller is used for charging and discharging operation. BMS controller have two loops, an outer voltage loop and inner current loop.




### Single-Phase Constant Voltage AC Power Supply

The single-phase constant voltage AC power supply provides a constant AC voltage to the connected complex loads. A single-phase inverter converts the output DC voltage from the boost converter to a constant single AC voltage supply. Choose a suitable PI controller to control the output voltage of the single-pahse inverter. To provide a smooth AC supply to the load, this model uses a LC filter.




### Supervisory Control(Mode Control) Parameters

Stand-alone PV system in this example comprises seven operating modes. These modes are selected based on DC bus voltage, solar irradiance and state of charge of the battery. The DC bus voltage level is used as a measure to detect a load imbalance. If the DC bus voltage is greater than $V_{dc-max}$, the system is generating more power than what the load is requiring. If the DC bus voltage is less than $V_{dc-min}$, then the load is requiring more power than what the system is generating.

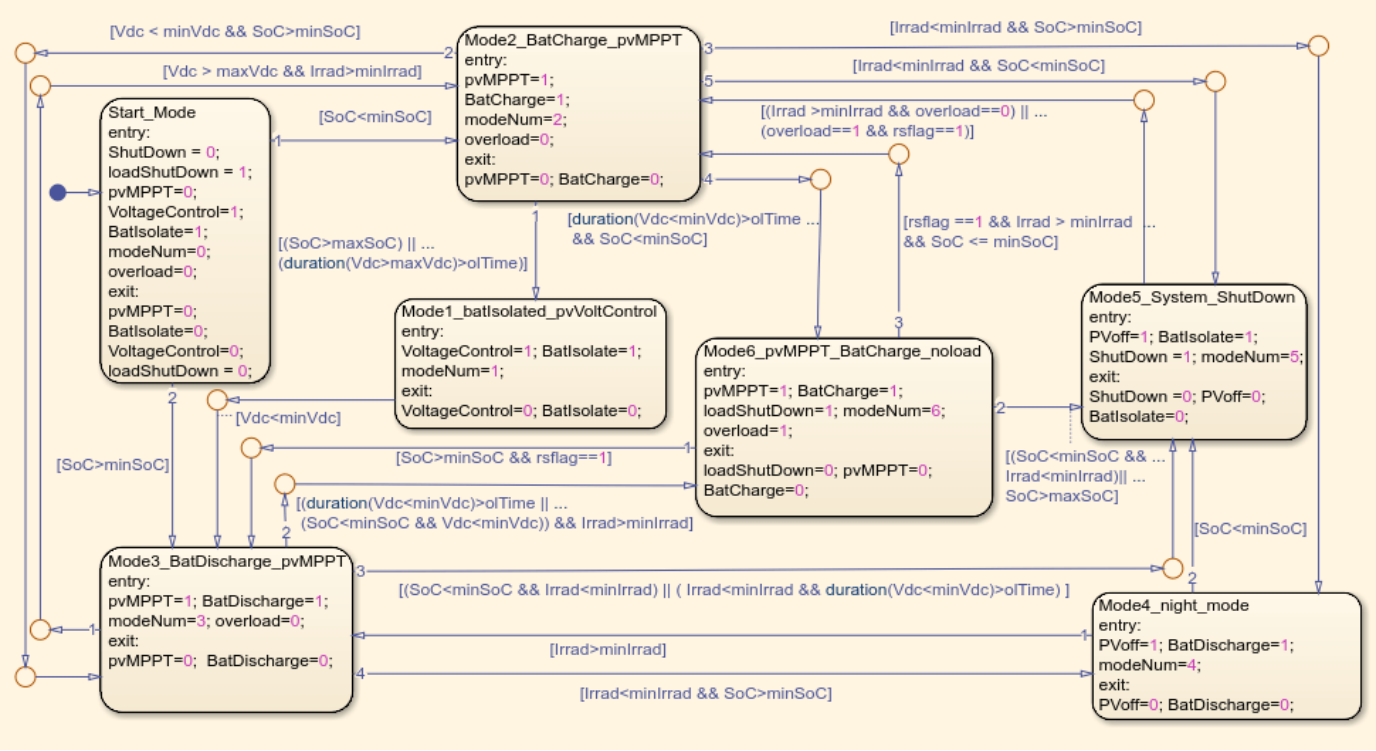
DC bus voltage level $(V_{dc})$, solar irradiance $(Irrad)$ and the battery state of charge $(SoC)$ are used to decide the suitable operating mode.

Operating modes of the stand-alone PV AC System are:

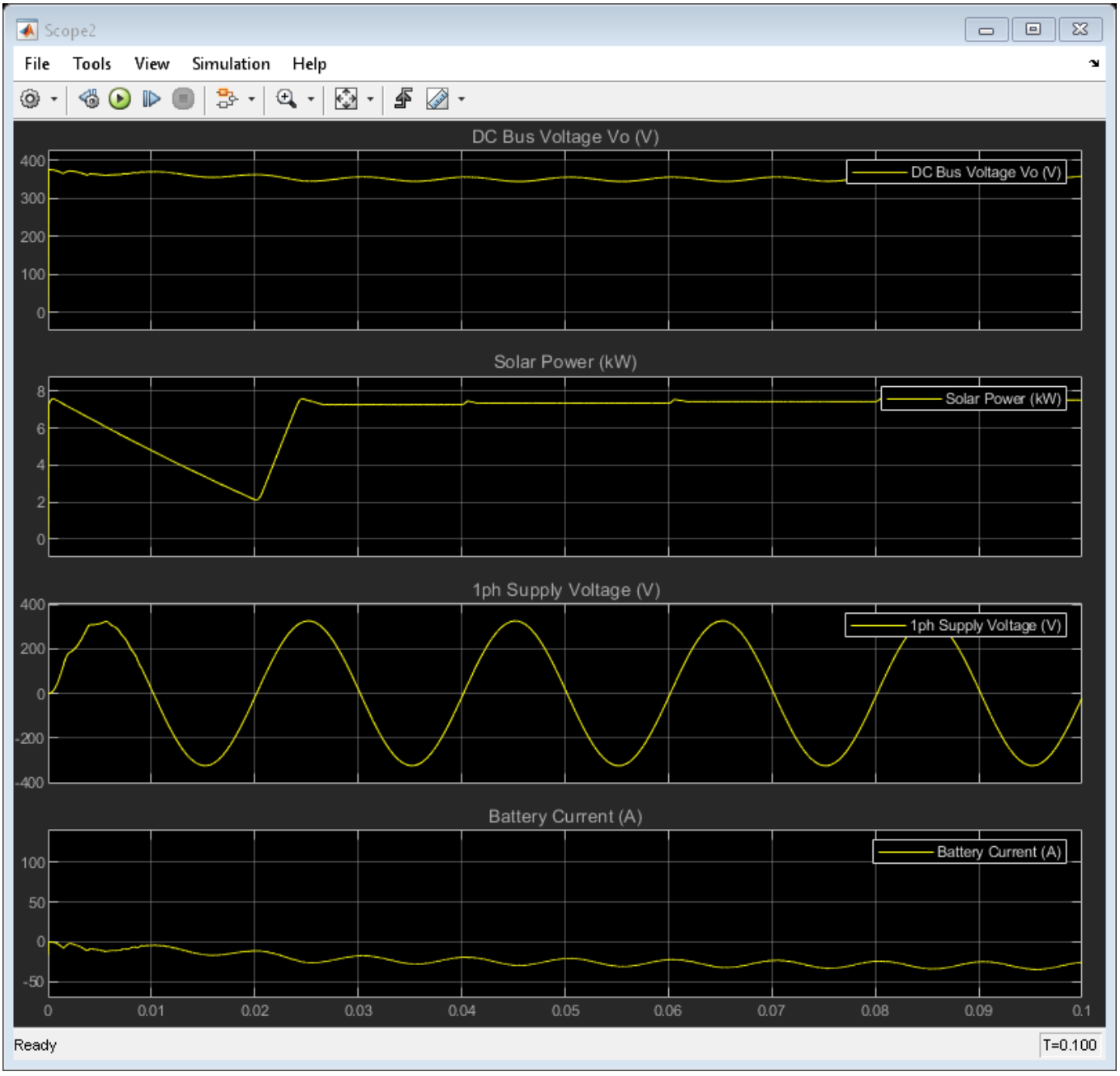
- Mode-0 - Start mode (Default simulation starting mode)

- Mode-1 - PV in output voltage control, battery fully charged and isolated
- Mode-2 - PV in maximum power point, battery is charging
- Mode-3 - PV in maximum power point, battery is discharging
- Mode-4 - Night mode, PV shutdown, battery is discharging
- Mode-5 - Total system shutdown
- Mode-6 - PV in maximum power point, battery is charging, load is disconnected

Stateflow mode control diagram



### Simulation Output

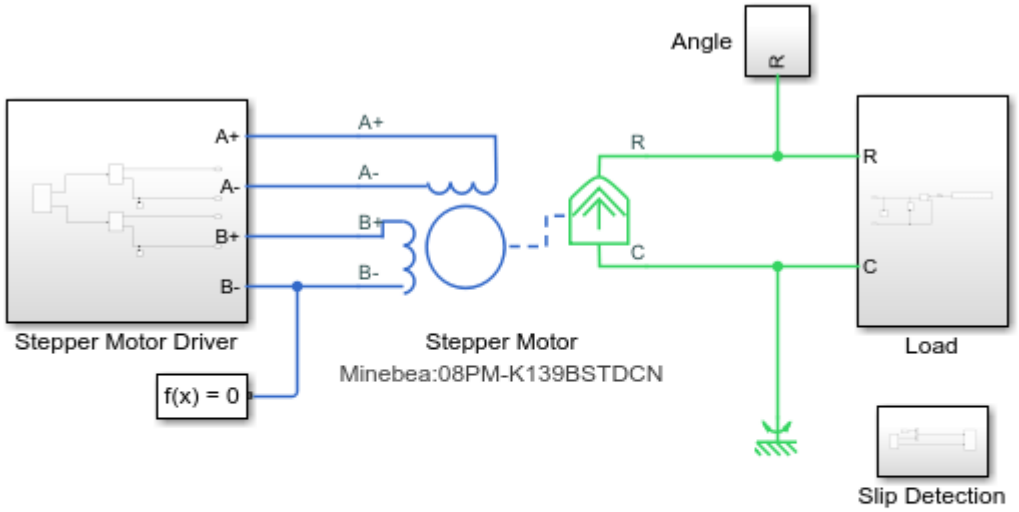


# Pre-Parameterized Stepper Motor Block Validation

This example demonstrates and validates a set of predefined parameterizations that are supplied with the Simscape™ Electrical™ Stepper Motor block. These predefined parameterizations are determined from manufacturer datasheets.

### Test Harness

First the particular stepper motor to be modeled must be selected. To do this, open the stepper motor block dialog and click on Select a predefined parameterization. This opens the Block Parameterization Manager from which you select the stepper part and apply it to the block. Based on the part you select, this script then loads the corresponding data that was used to parameterize the part, including pull-in curves from the original datasheet. The test harness works by gradually increasing the load until slip occurs for each step rate demand tested. Slip detection is implemented in the Slip Detection subsystem and uses a Simulink® assertion block that checks the difference between expected and actual rotor angle.




**Pre-Parameterized Stepper Motor Block Validation**

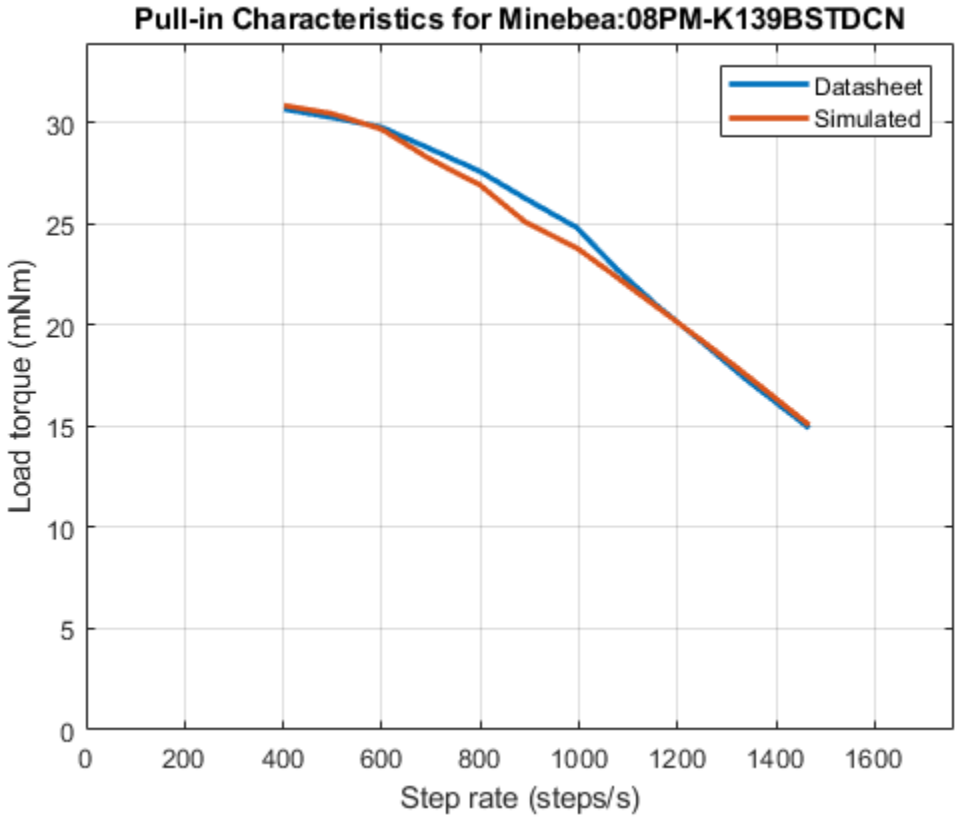
1. Open script to view the manufacturer data and define load torque
2. Plot pull-in characteristics of the selected stepper motor (see code)
3. Regenerate the pull-in characteristics for given step rate (computationally intensive, see code)
4. Explore simulation results using Simscape Results Explorer
5. Learn more about this example

Copyright 2008-2022 The MathWorks, Inc.

### Simulated Results and Datasheet Pull-in Torque Characteristics

The plot below shows the simulated pull-in characteristics of one of the parameterized stepper motor options, along with the curve given in the manufacturer datasheet. Achieving an exact match for the pull-in characteristics can be challenging, a key factor being that most datasheets do not specify the test conditions used. Engineering judgement and simplifying assumptions are used to fill in for missing data. As a result deviations between simulated and actual physical behavior are to be expected. To ensure requisite accuracy, following initial parameterization from a test harness like

this, the next step is to validate simulated behavior against experimental data and to refine component models as necessary.



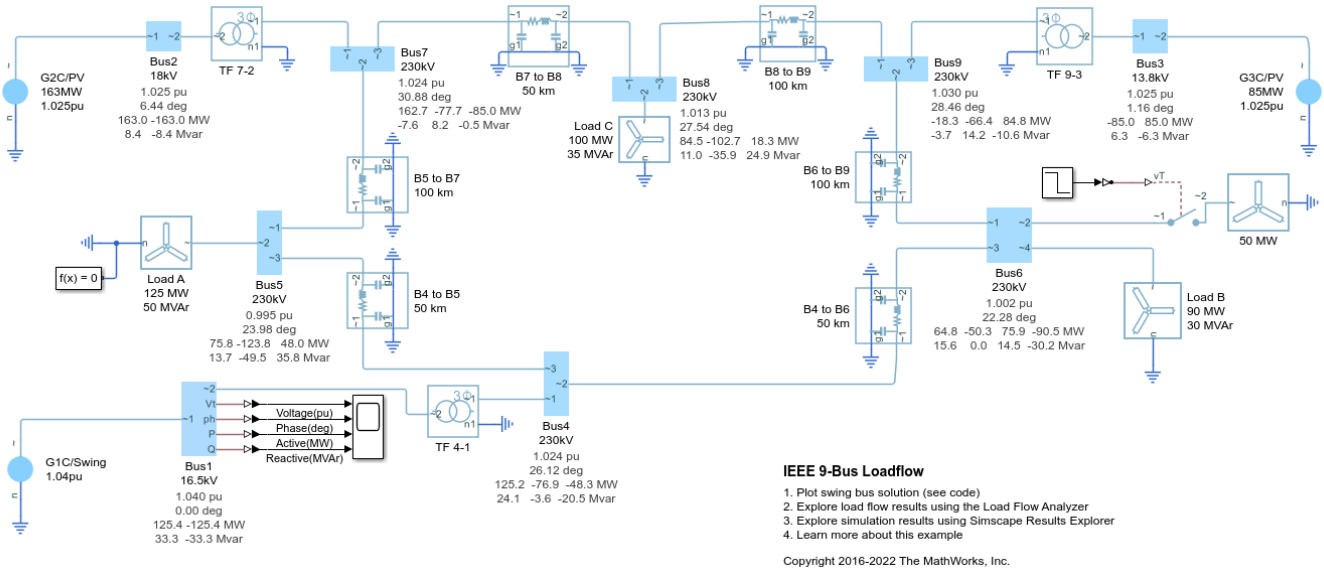

For any given stepper motor, the set of model parameter values that matches the pull-in torque characteristics is often not unique. To ensure the parameterization is representative, it is good practice to also generate the pullout curve and compare it with the curve given in the datasheet.

To learn more about parameterizing a stepper motor refer to example ee_motor_stepper_pullin_characteristics.

# IEEE 9-Bus Loadflow

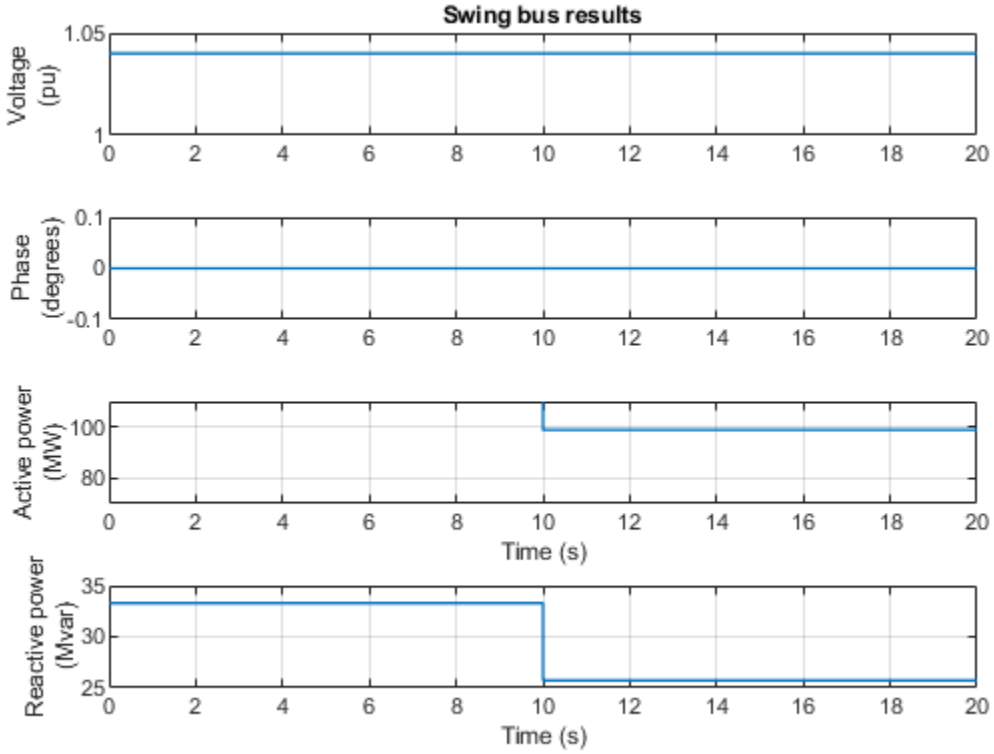
This example shows a model of a 9-bus three-phase power system network. This example is based on an IEEE® benchmark test case, further details of which can be found in "Power System Control and Stability" by P. M. Anderson and A. A. Fouad (IEEE Press, 2003). Simscape™ initializes two of the generators to the specified powers and terminal voltages, and initializes the remaining swing bus generator to meet just the specified voltage. The resulting load flow solution is appended to each of the busbars post-simulation. The four rows correspond to per-unit voltage, phase, active power, and reactive power respectively. Looking at Bus 1, it can be seen from the annotation that the swing generator delivers 76.4MW of active power and 27.5MVAr or reactive power to the network. Differences to the original benchmark are due to the transmission line models and transformer configurations used.

### Model




### Simulation Results from Simscape Logging

The plot below shows the dynamic response due to adding a 50MW load to Bus 6 half-way through the simulation. The additional active and reactive powers that the swing bus must deliver can be seen.


Swing bus results
Voltage (pu)
1.05
1
Phase (degrees)
0.1
0
-0.1
Active power (MW)
100
80
Time (s)
Reactive power (Mvar)
35
30
25
Time (s)
0
2
4
6
8
10
12
14
16
18
20

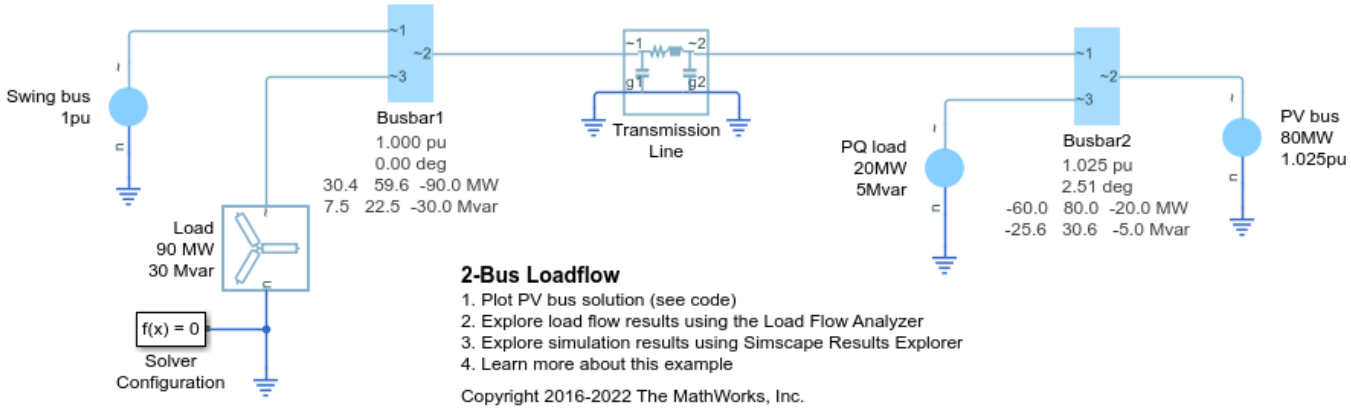
# 2-Bus Loadflow

This example shows a model of a two-bus three-phase power system network. The model uses three instances of the Load Flow Source block from Simscape™ Electrical™, one configured to be the swing bus, one configured to be the PV bus, and one configured to be the PQ load. The PV bus regulates its output to be at a voltage of 1.025 times rated voltage and to deliver 80MW active power to the network. The Swing bus regulates voltage at the other end of the transmission line to be one times rated voltage, and it delivers the requisite power to the network so that overall active and reactive powers balance. The Simscape initialization solver determines the required internal initial voltage amplitudes and phases in both the PV bus and the Swing bus so as to start in steady state.

The two busbar blocks are tagged with a summary of the load flow initialization results. These tags are updated immediately following simulation of the model. From the first row of the tags it can be seen that the requested voltages at the terminals of the two sources have been honored. The second rows of the tags show that Busbar2 leads Busbar1 by 2.51 degrees, this reflecting that there is a net power flow from Busbar2 to Busbar1. The third and fourth rows show the active and reactive powers respectively. For example, the third row of the Busbar2 tag indicates that 60MW of active power flows out of the busbar to the transmission line, 80MW of active power is supplied by the PV bus, and that 20MW of active power is delivered to the PQ load.

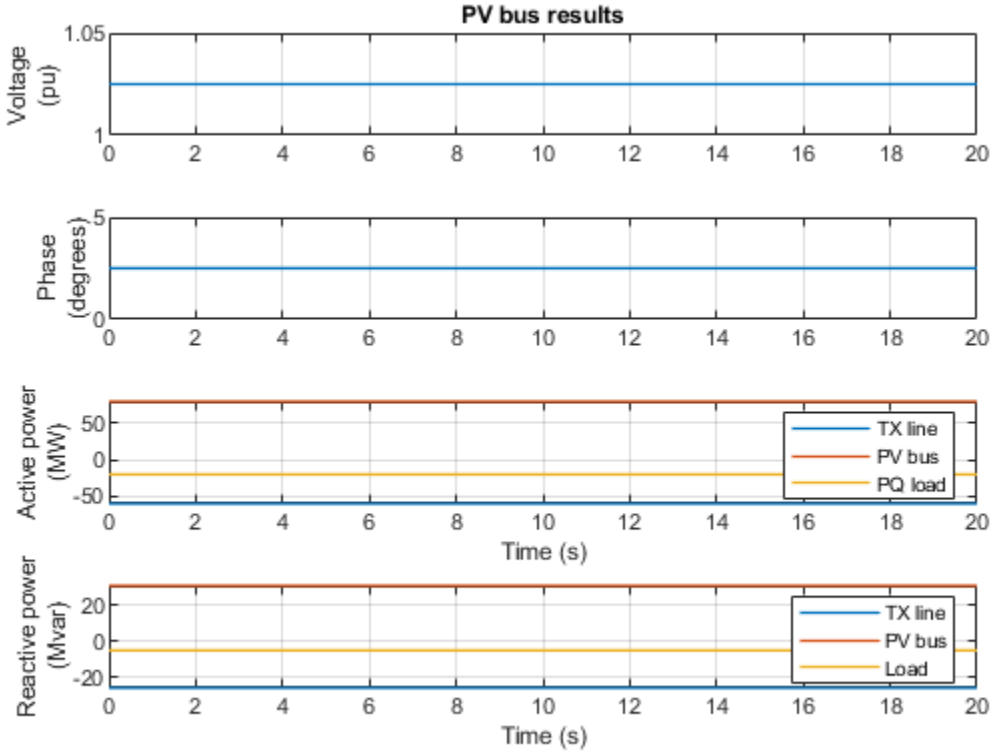
Copyright 2008-2019

### Model




### Simulation Results from Simscape Logging

The plot below shows the simulation results following the steady state initialization to meet load flow targets. The Equation formulation option on the Solver Configuration block is set to Frequency and time for fast simulation. This model can also be run using the Time option provided that the Start simulation from steady state option is checked on the Solver Configuration block. However, simulating using the Time option does not update the Busbar blocks with loadflow results.


PV bus results
Voltage (pu)
1.05
1
Phase (degrees)
5
0
Active power (MW)
50
0
-50
TX line
PV bus
PQ load
Time (s)
Reactive power (Mvar)
20
0
-20
TX line
PV bus
Load
Time (s)
0
2
4
6
8
10
12
14
16
18
20

# Induction Motor Initialization with Loadflow

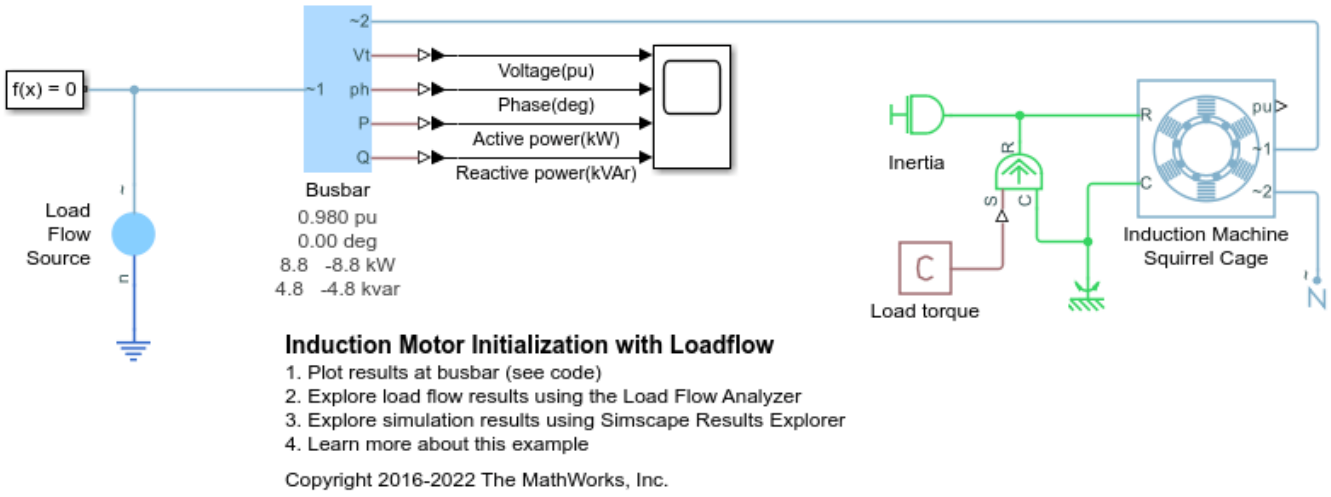
This example shows how to initialize a three-phase induction motor as part of a load flow analysis. When initializing an induction machine that is directly connected to an AC network, in steady state there is one degree of freedom which can be set by any one of shaft torque, shaft power, motor speed or electrical power.

Here the model is configured to initialize delivering 8kW of mechanical power to a load. Hence the mechanical power consumed variable on the induction motor variables tab is set to -8kW with high priority. The slip and real power generated variables are left at priority none. Similarly, because the inertia initial speed is equivalent to slip, the rotational velocity is set to priority none. To help the intialisation converge, the beginning value is set to synchronous speed rather than left at zero.

The shaft torque is set to -8kW divided by the synchronous mechanical speed. This is an approximation as some negative slip is expected when generating. The error will result in an initial acceleration of the inertia. If increased accuracy is required, the measured slip from the first simulation can be used to correct the shaft torque applied, and the simulation run a second time. However, it is important to note that this error in the shaft torque will not change load flow results for the rest of the electrical network.

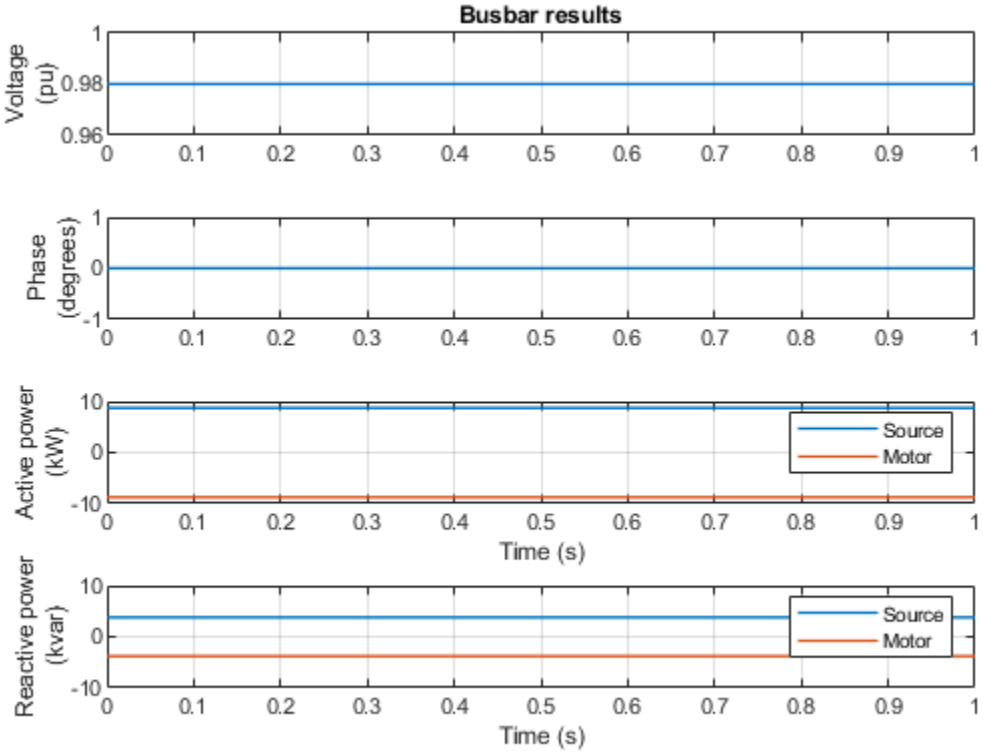
Copyright 2008-2019

### Model




### Simulation Results from Simscape Logging

The plot below shows the simulation results following the steady state initialization to meet load flow targets. It can be seen that there is no transient, indicating that the induction motor has been correctly initialized.


Busbar results
Voltage (pu)
1
0.98
0.96
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Phase (degrees)
1
0
-1
Active power (kW)
10
0
-10
Source
Motor
Time (s)
Reactive power (kvar)
10
0
-10
Source
Motor
Time (s)

# Synchronous Machine Initialization with Loadflow

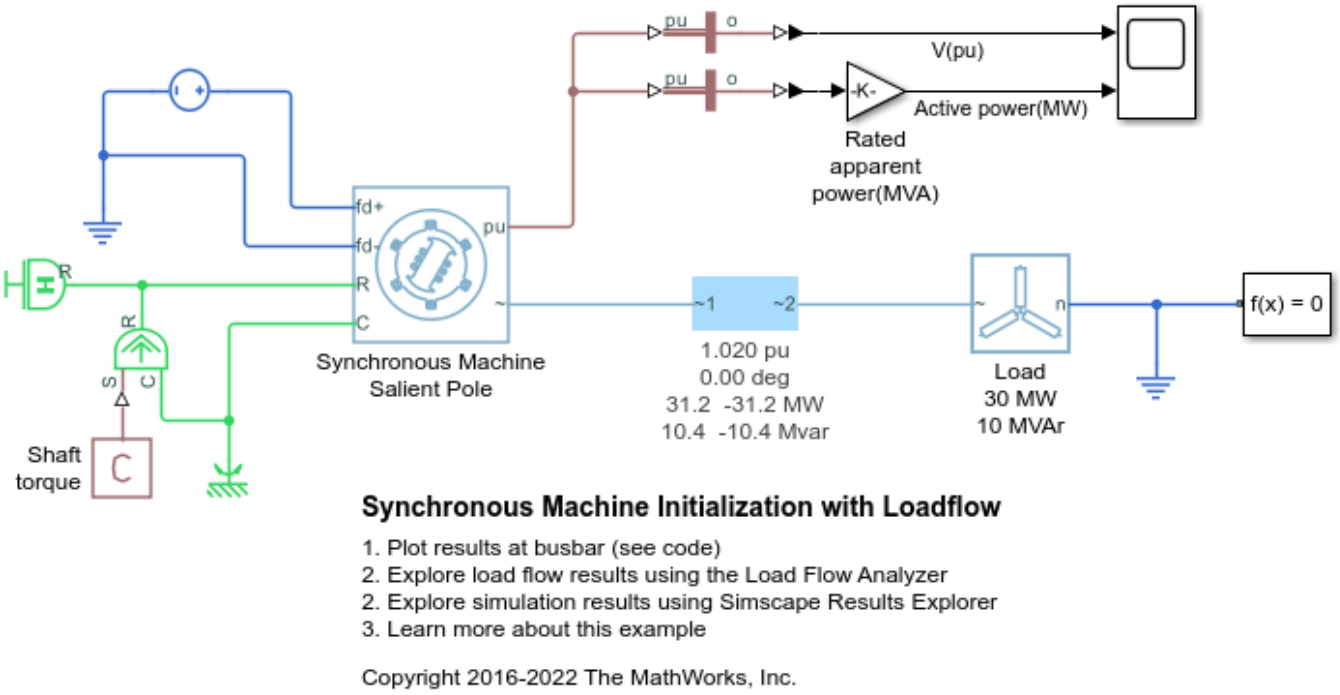
This example shows how to initialize synchronous machine as part of a load flow analysis. When initializing a synchronous machine there are two degrees of freedom which can be set by any two of rotor angle, active power, reactive power and terminal voltage. The pair of variables that are constrained is set by the source type drop-down menu, this having options of Swing bus, PV bus and PQ bus. Here the machine is configured for a swing bus with a 1.02 per-unit voltage and zero degrees phase.

The shaft torque is set to 30MW divided by the synchronous mechanical speed. This is only an approximate value, but any difference to the value required for steady state will just result in a non-zero initial acceleration for the machine inertia. This will not change load flow results for the rest of the electrical network. Similarly the AVR is initialized close to the expected steady state field voltage, but any difference to the required value only creates an initial transient for field winding flux linkage. Again, this does not impact load flow results for the rest of the electrical network.

Exact values for the AVR and shaft torque for steady state can be determined after the initial load flow solution is found. This is done by setting the initialization option to Set real power, reactive power, terminal voltage, and terminal phase. The load flow solution can then be copied to the four corresponding parameters. Right-click the block and select Electrical->Display Associated Initial Conditions. This then prints the initial mechanical torque and field circuit voltage to the command prompt.

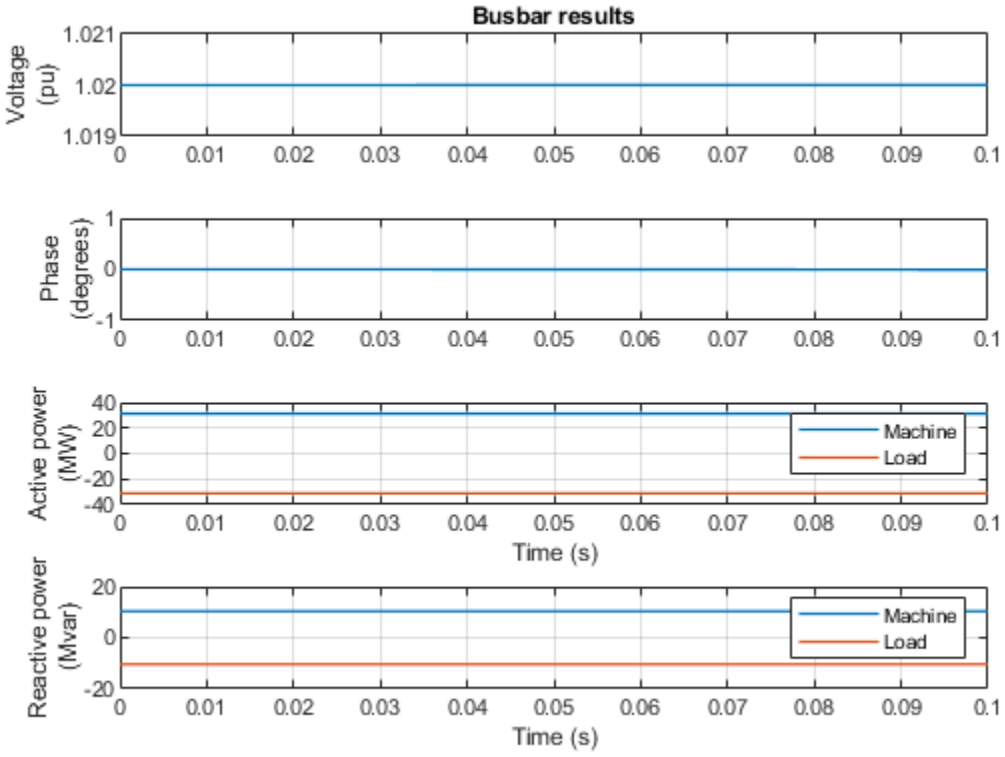
Copyright 2008-2019

**Model**



### Simulation Results from Simscape Logging

The plot below shows the simulation results following the steady state initialization to meet load flow targets. The ramp away from the steady state solution reflects that the governor and AVR are not modeled here. However, the initial values show that the loadflow targets are met.

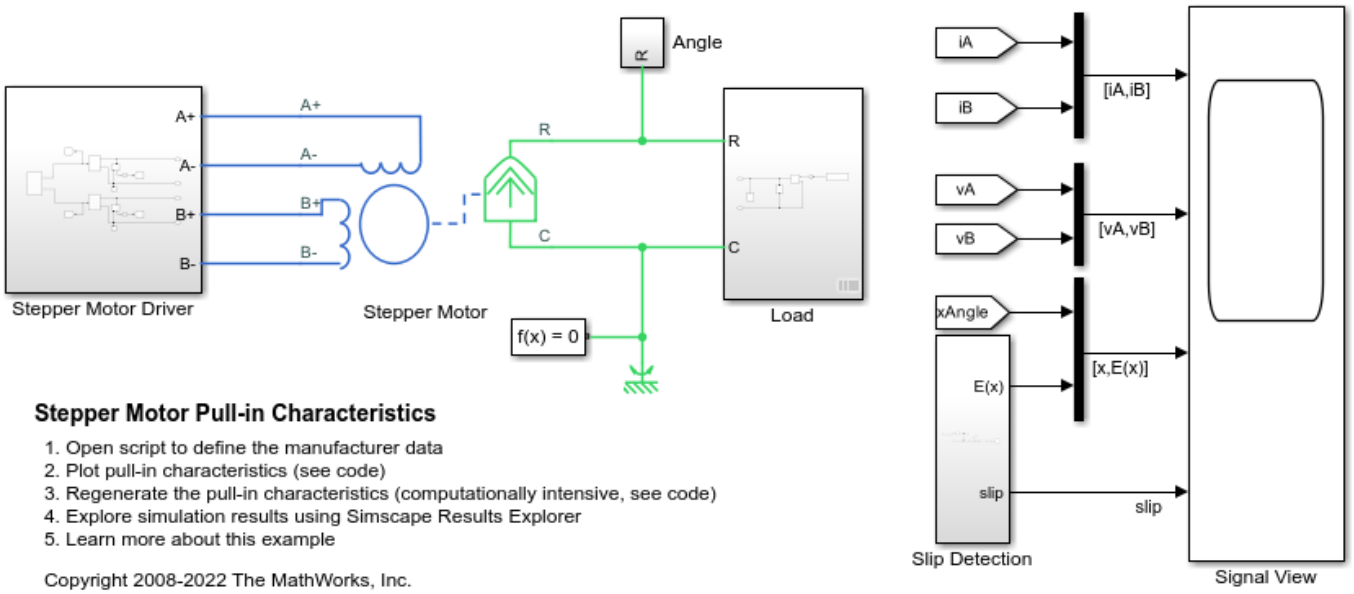


# Stepper Motor Pull-in Characteristics

This example shows how to parameterize and tune a stepper motor using manufacturer datasheet information and a test harness. The model is parameterized using numerical data extracted from a datasheet. The simulation generates pull-in torque characteristics that you can compare to a manufacturer-provided pull-in curve. To tune the stepper motor model, the example uses a test harness that varies the drive type and load parameters.

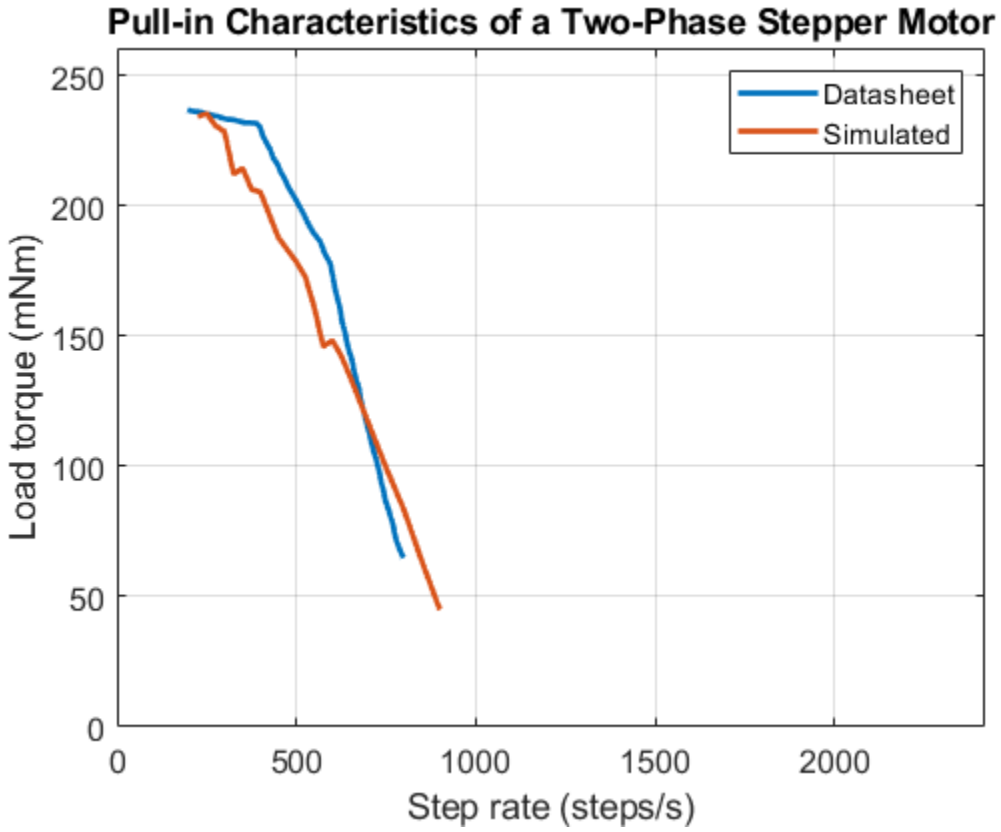
### Test Harness

The stepper is modeled using the Stepper Motor block from the Simscape™ Electrical™ library. The stepper motor driver is modelled by a current source and a first order filter. The motor is placed in a test harness. The test harness gradually increases the load until slip occurs for each step rate demand tested. Slip detection is implemented in the Slip Detection subsystem. The subsystem contains a Simulink® assertion block that determines the difference between expected and actual rotor angle.




### Pull-in Torque Characteristics

The plot shows the pull-in characteristics generated by the test harness simulation. Results are superimposed on the pull-in curve from the manufacturer datasheet. Achieving an exact match for the pull-in characteristics can be challenging because most datasheets do not specify the test conditions. Furthermore, some datasheets do not provide all the numerical values required to model the stepper motor. In this case, the pull-in curve from the simulation is used to determine for representative values that yield an acceptable pull-in curve match.



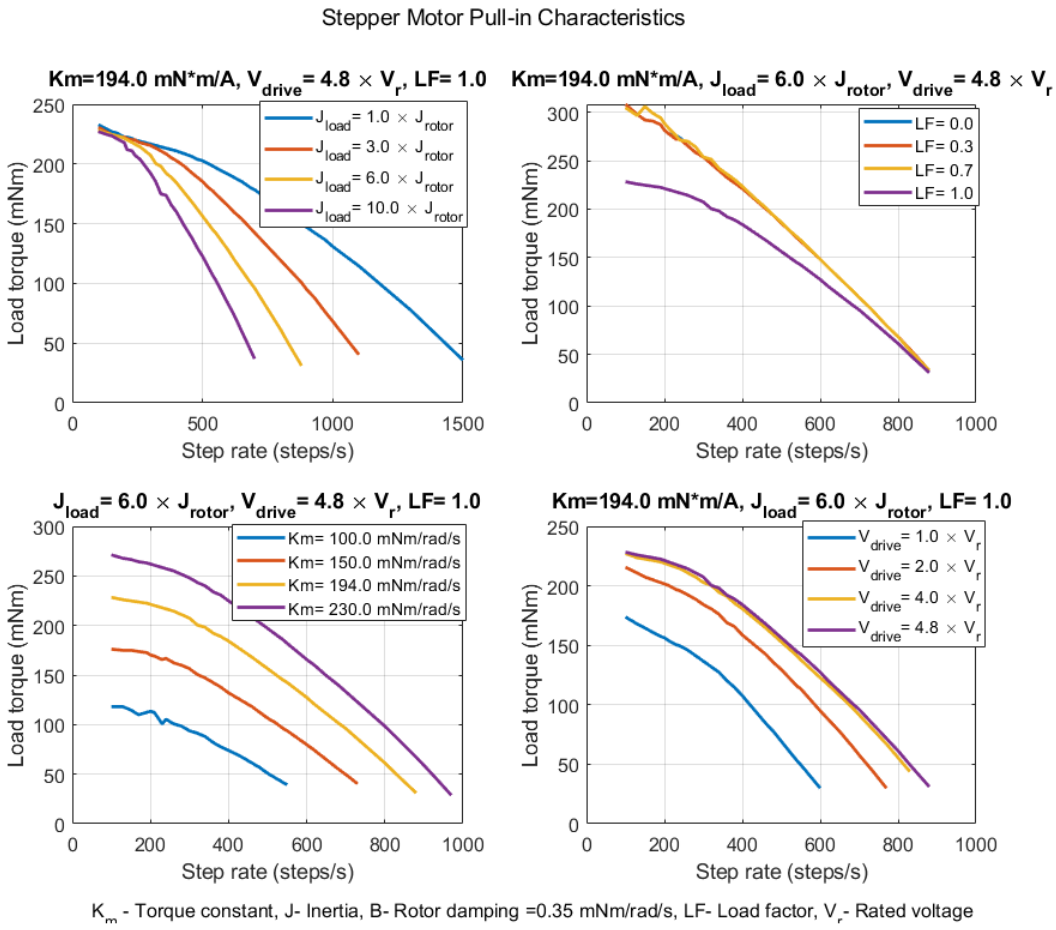

**Pull-in Characteristics Sensitivity to Drive and Load Parameters**

In pull-in mode, a stepper motor must start and stop without losing synchronization. Due to the dynamic nature of stepper motors, pull-in torque-speed performance is highly sensitive to stepper motor drive configuration and load parameters.

When modeling or tuning a stepper motor, consider that:

- Increasing the load inertia reduces the stepper motor pull-in torque at higher step rates. Generally, for high step-rate operation, load inertia is less than three times that of rotor inertia. It is a generally-accepted practice to limit the load inertia to less than ten times that of rotor inertia.
- Loads with higher damping component results in better performance at lower step rates because damping helps to overcome the effects of stepper motor resonance. Similarly, internal motor rotor damping can also help to improve the performance at low step rates.
- Stepper motors with low winding resistance are often driven using a constant current drive. To reduce the current rise time, higher than the rated voltage is applied to the motor. The higher supply voltage yields higher pull-in torque capability at higher step rates.
- Pull-in torque curve provided on a manufacturer datasheet is typically given for a particular driver and load configuration (load type, load inertia and load damping). Manufacturers tend to test stepper motors either with a dyno setup or by applying friction to the rotor wheel. The test method is rarely included on the data sheet. Therefore, it is always important to simulate the stepper motor to generate the curve for parameterization validation purposes.

The plot shows how the pull-in torque is sensitive to rotor damping, drive voltage, load inertia, and load type.

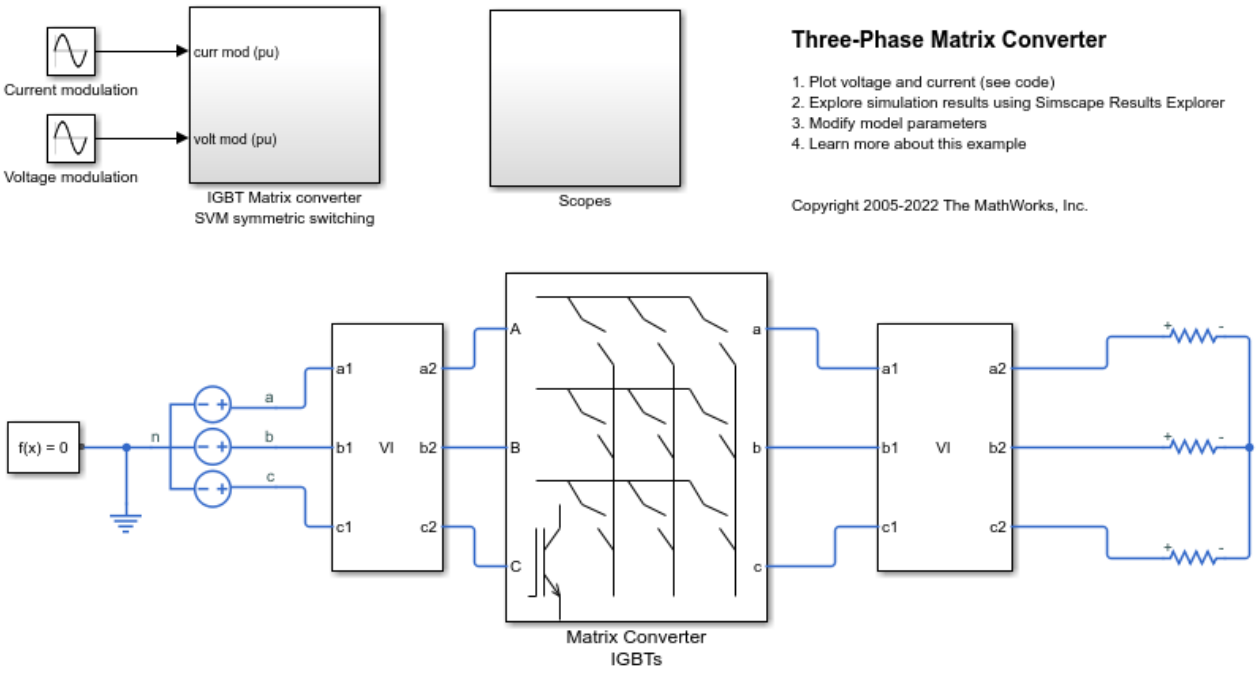



For any given stepper motor, the set of model parameter values that matches the pull-in torque characteristics is not typically unique. To ensure the parameterization is representative, it is good practice to also generate and compare the pull-out curve with the datasheet.

# Three-Phase Matrix Converter

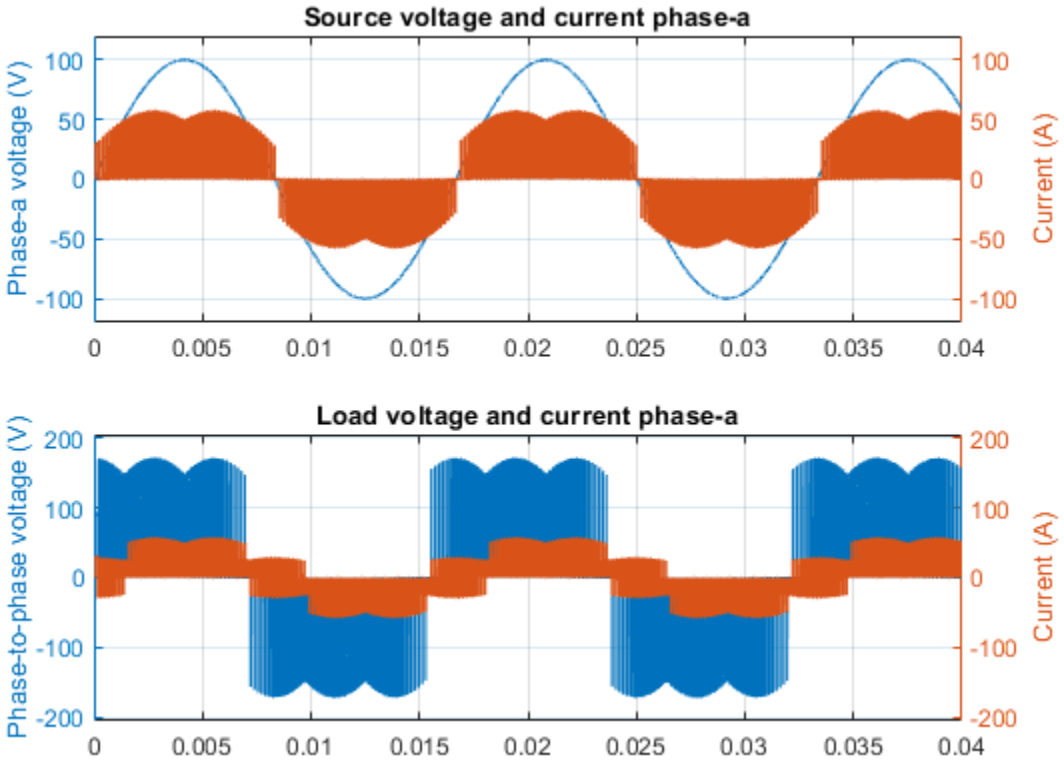
This example shows a three-phase matrix converter that drives a static load and draws unity power factor at the source. The Scopes subsystem contains scopes that allow you to see the simulation results.

### Model




### Simulation Results from Simscape Logging

The plot below shows the voltages and currents for the source and load.


Source voltage and current phase-a
Phase-a voltage (V)
Current (A)
100
50
0
-50
-100
0
0.005
0.01
0.015
0.02
0.025
0.03
0.035
0.04
Load voltage and current phase-a
Phase-to-phase voltage (V)
Current (A)
200
100
0
-100
-200

# Battery Parameter Extraction from Data

This example shows optimization of the Battery block's parameters to fit data defined over different temperatures. It uses the MATLAB® optimization function `fminsearch`. Other products available for performing this type of parameter fitting with Simscape™ Electrical™ models are the Optimization Toolbox™ and Simulink® Design Optimization™. These products provide predefined functions to manipulate and analyze blocks using GUIs or a command line approach.

### Strategy

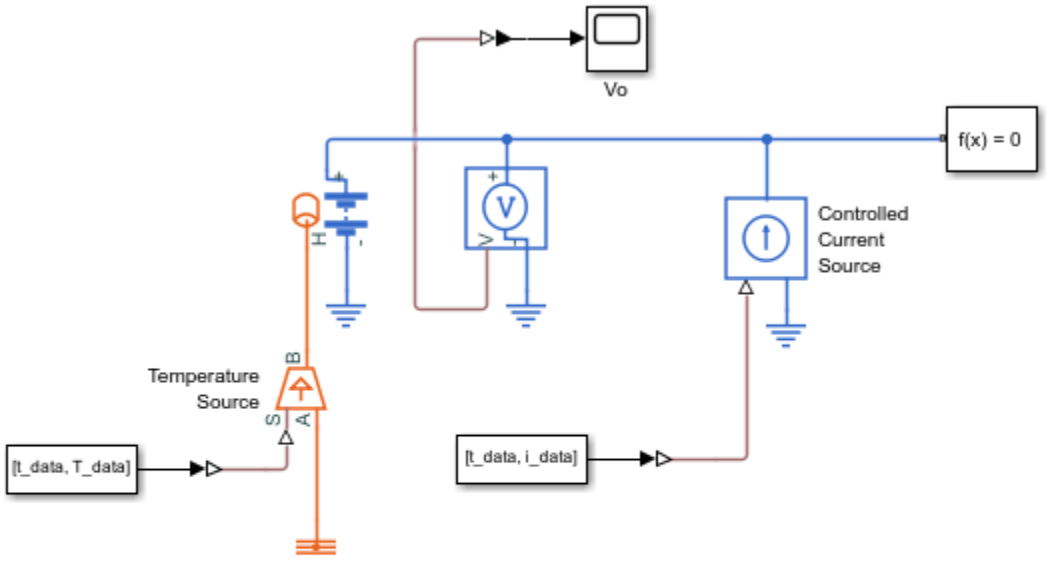
Fit output voltage curves for a Battery to data using a 4 step procedure:

1. Optimize parameters in the Battery Main dialog tab.
2. Optimize parameters in the Battery Dynamics dialog tab.
3. Optimize nominal voltage and internal resistance in the Battery Temperature Dependence dialog tab.
4. Optimize temperature dependent charge dynamics parameters in the Battery Temperature Dependence dialog tab.

### Data and Block Setup

The MATLAB data file, *ee_battery_data.mat*, stores Battery data as an array of structures. Each structure contains 5 fields: *v* (voltage) , *i* (current), *t* (time), *SOC0* (initial state of charge) and *T* (temperature). Scope save the output voltage as structure data, out.Vo.signals.values.

```
% Load Battery data
load ee_battery_data.mat
assignin('base','T1',battery_data(find([battery_data(1:2).T]==25)).T);
assignin('base','T2',battery_data(find([battery_data(1:2).T]~=25)).T);
% Display the Battery model
Model = 'ee_battery';
open_system(Model)
```




```
close_system(Model, 0);
```

### Initial Parameter Specification

Starting values for `fminsearch` can be estimated using a combination of Battery block defaults and data sheet values

List of parameters and initial values prior to optimization

```
ParsListMain = {'Vnom', 'R1', 'AH', 'V1', 'AH1'};
InitGuessMain = [3.6, 0.045, 2.7, 3.4, 1.4];
ParsListDyn = {'Rp1', 'tau1'};
InitGuessDyn = [0.006, 200];
ParsListTemp = {'Vnom_T2', 'R1_T2', 'V1_T2','Rp1_T2','tau1_T2'};
InitGuessTemp = [3.8, 0.055, 3.6, 0.006, 200 ];

Pars0 = reshape([[ParsListMain ParsListDyn ParsListTemp]; cellstr(num2str([InitGuessMain InitGu
fprintf('\t%5s = %s\n', Pars0{:});
clear Pars0
```

```
   Vnom =   3.6
     R1 = 0.045
     AH =   2.7
     V1 =   3.4
    AH1 =   1.4
    Rp1 = 0.006
   tau1 =   200
  Vnom_T2 =   3.8
  R1_T2 = 0.055
```

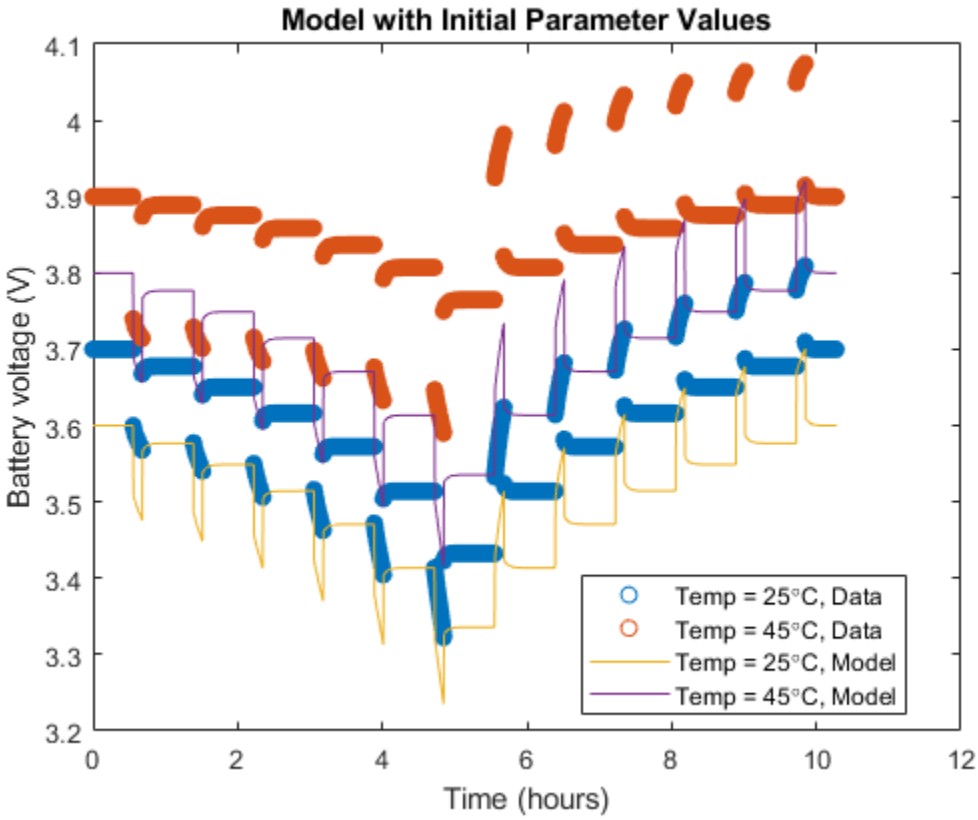
```
V1_T2 =   3.6
Rp1_T2 = 0.006
tau1_T2 =   200
```

Since `fminsearch` is an unconstrained nonlinear optimizer that locates a local minimum of a function, varying the initial estimate will result in a different solution set.

### Plot Data Versus Battery Output Using Initial Parameters

Load single cell Battery model and set parameters

```
load_system(Model);
% Enable Fast Restart to speedup the simulation
set_param(Model,'FastRestart','on')

Pars = reshape([ParsListMain; cellstr(num2str(InitGuessMain'))'],1,[]);
for k=1:2:length(Pars)
    evalin('base',[Pars{k} '=' Pars{k+1} ';'])
end

Pars = reshape([ParsListDyn; cellstr(num2str(InitGuessDyn'))'],1,[]);
for k=1:2:length(Pars)
    evalin('base',[Pars{k} '=' Pars{k+1} ';'])
end

Pars = reshape([ParsListTemp; cellstr(num2str(InitGuessTemp'))'],1,[]);
for k=1:2:length(Pars)
    evalin('base',[Pars{k} '=' Pars{k+1} ';'])
end

% Generate preliminary model curves and plot against data
num_lines = length(battery_data)-1;
v_model = cell(1, num_lines);
t_model = cell(1, num_lines);
legend_info_data  = cell(1, num_lines);
legend_info_model = cell(1, num_lines);

for idx_data = 1:num_lines
    assignin('base','t_data',battery_data(idx_data).t);
    assignin('base','i_data',battery_data(idx_data).i);
    assignin('base','T_data',battery_data(idx_data).T*ones(length(t_data),1));
    assignin('base','T0',battery_data(idx_data).T);
    assignin('base','Ts',t_data(2)-t_data(1));
    assignin('base','AH0',AH*battery_data(idx_data).SOC0);

    out = sim(Model);
    v_model{idx_data} = out.Vo.signals.values;
    t_model{idx_data} = out.Vo.time;
    legend_info_data{idx_data}  = [ 'Temp = '                          ...
        num2str(battery_data(idx_data).T) '\circC, Data'];
    legend_info_model{idx_data} = [ 'Temp = '                          ...
        num2str(battery_data(idx_data).T) '\circC, Model'];
end
plot([battery_data(1:num_lines).t]/3600, [battery_data(1:num_lines).v], 'o', [t_model{:}]/3600
xlabel('Time (hours)');
ylabel('Battery voltage (V)');
legend([legend_info_data legend_info_model], 'Location', 'Best');
title('Model with Initial Parameter Values');
```




### Sum of Squares of Error Calculation

`ee_battery_lse` is the function to be minimized by `fminsearch`. This function returns a sum of squares of error for the difference between the Battery output voltage and the data. If an invalid parameter value is supplied by `fminsearch`, the `catch` statement returns a large value for the error.

### Optimize Main Tab Dialog Parameters Without Charge Dynamics (Step 1)

```
% Find ambient temperature data index
idx_data = find([battery_data(1:num_lines).T]==25);
assignin('base','t_data',battery_data(idx_data).t);
assignin('base','i_data',battery_data(idx_data).i);
assignin('base','T_data',battery_data(idx_data).T*ones(length(t_data),1));
assignin('base','T0',battery_data(idx_data).T);
assignin('base','Ts',t_data(2)-t_data(1));
assignin('base','v_data',battery_data(idx_data).v);

% Optimize parameters in main dialog tab of Battery
assignin('base','ParsList',ParsListMain(1:4));
InitGuess = InitGuessMain(1:4);
OptPars = fminsearch(@ee_battery_lse, InitGuess,              ...
    optimset('TolX', 1e-3));

OptParsMain = [OptPars(1:4) InitGuessMain(5)];

% Update Battery block with optimized parameters
Pars = reshape([ParsListMain; cellstr(num2str(OptParsMain'))'],1,[]);
```

```
for k=1:2:length(Pars)
    evalin('base',[Pars{k} '=' Pars{k+1} ';'])
end

% Display optimized parameters
fprintf(['Optimized parameters for the battery main '      ...
  'dialog tab are:\n']);
fprintf('\t%5s = %s\n', Pars{:});

clear i_data v_data t_data T_data Ts
clear k InitGuess
```

```
Optimized parameters for the battery main dialog tab are:
     Vnom =   3.6999
       R1 = 0.050299
       AH =   2.6033
       V1 =   3.5265
      AH1 =      1.4
```

### Optimize Charge Dynamics Parameters (Step 2)

```
% Use only one current pulse for optimizing the charge dynamics
i_pos=battery_data(1).i.*(battery_data(1).i>=0);
a=find(diff(i_pos)>0,2);
b = find(diff(battery_data(1).i));
c = fix((b(find(b<a(1),1,'last'))+a(1))/2);
assignin('base','i_data',battery_data(idx_data).i(c+1:a(2)));
assignin('base','v_data',battery_data(1).v(c+1:a(2)));
assignin('base','t_data',battery_data(idx_data).t(1:length(i_data)));
assignin('base','T_data',battery_data(idx_data).T*ones(length(t_data),1));
assignin('base','T0',battery_data(idx_data).T);
assignin('base','Ts',t_data(2)-t_data(1));

% Find Battery initial charge before optimizing charge dynamics parameters
assignin('base','ParsList',{'charge'});
InitGuessCharge = OptParsMain(3);
OptCharge = fminsearch(@ee_battery_lse, InitGuessCharge,              ...
    optimset('TolX', 1e-3));
assignin('base','AH0',OptCharge);
% Optimize Battery charge dynamics parameters
assignin('base','ParsList',[ParsListMain(2) ParsListDyn]);
InitGuessDyn = [OptPars(2) InitGuessDyn];
OptParsDyn = fminsearch(@ee_battery_lse, InitGuessDyn,             ...
    optimset('TolX', 1e-3));

% Update Battery block with optimized charge dynamics parameters
ParsListMainDyn = [ParsListMain ParsListDyn];
OptParsMainDyn = [OptPars(1) OptParsDyn(1) OptPars(3:4) InitGuessMain(5) OptParsDyn(2:3)];
Pars = reshape([ParsListMainDyn; cellstr(num2str(OptParsMainDyn'))'],1,[]);
for k=1:2:length(Pars)
    evalin('base',[Pars{k} '=' Pars{k+1} ';'])
end
assignin('base','AH0',AH*battery_data(idx_data).SOC0);

% Display optimized parameters
fprintf(['Optimized parameters for the Battery, '          ...
   'including charge dynamics, are:\n']);
fprintf('\t%5s = %s\n', Pars{:});
```

```
clear i_data v_data t_data T_data Ts
clear i_pos a b c
clear k
clear OptPars OptParsDyn ParsListMainDyn InitGuessMain InitGuessDyn ParsListDyn ParsListMain
```

```
Optimized parameters for the Battery, including charge dynamics, are:
     Vnom =    3.699931
       R1 =  0.05019736
       AH =    2.603326
       V1 =    3.526493
      AH1 =         1.4
      Rp1 = 0.005029392
     tau1 =     109.691
```

**Optimize Temperature Dependent Parameters (Step 3)**

```
idx_data = find([battery_data(1:num_lines).T]~=25);
assignin('base','t_data',battery_data(idx_data).t);
assignin('base','i_data',battery_data(idx_data).i);
assignin('base','T_data',battery_data(idx_data).T*ones(length(t_data),1));
assignin('base','T0',battery_data(idx_data).T);
assignin('base','Ts',t_data(2)-t_data(1))
assignin('base','v_data', battery_data(2).v);

% Use parameters for T1 as initial guess for T2 parameters
InitGuessTemp = [OptParsMainDyn(1:2) OptParsMainDyn(4) OptParsMainDyn(6:7)];
Pars = reshape([ParsListTemp; cellstr(num2str(InitGuessTemp'))'],1,[]);
for k=1:2:length(Pars)
    evalin('base',[Pars{k} '=' Pars{k+1} ';'])
end

% Optimize Battery temperature dependent parameters
assignin('base','ParsList',ParsListTemp(1:3));
OptParsTemp = fminsearch(@ee_battery_lse, InitGuessTemp(1:3),              ...
    optimset('TolX', 1e-3));

% Update Battery block with optimized parameters
Pars = reshape([ParsListTemp(1:3); cellstr(num2str(OptParsTemp'))'],1,[]);
for k=1:2:length(Pars)
    evalin('base',[Pars{k} '=' Pars{k+1} ';'])
end
assignin('base','AH0',AH*battery_data(idx_data).SOC0);

% Display optimized parameters
fprintf(['Optimized temperature dependent parameters for the Battery '           ...
   'are:\n']);
fprintf('\t%5s = %s\n', Pars{:});

clear i_data v_data t_data T_data Ts
clear k
clear OptParsMainDyn
```

```
Optimized temperature dependent parameters for the Battery are:
    Vnom_T2 =   3.9003
    R1_T2 = 0.081404
    V1_T2 =   3.8133
```

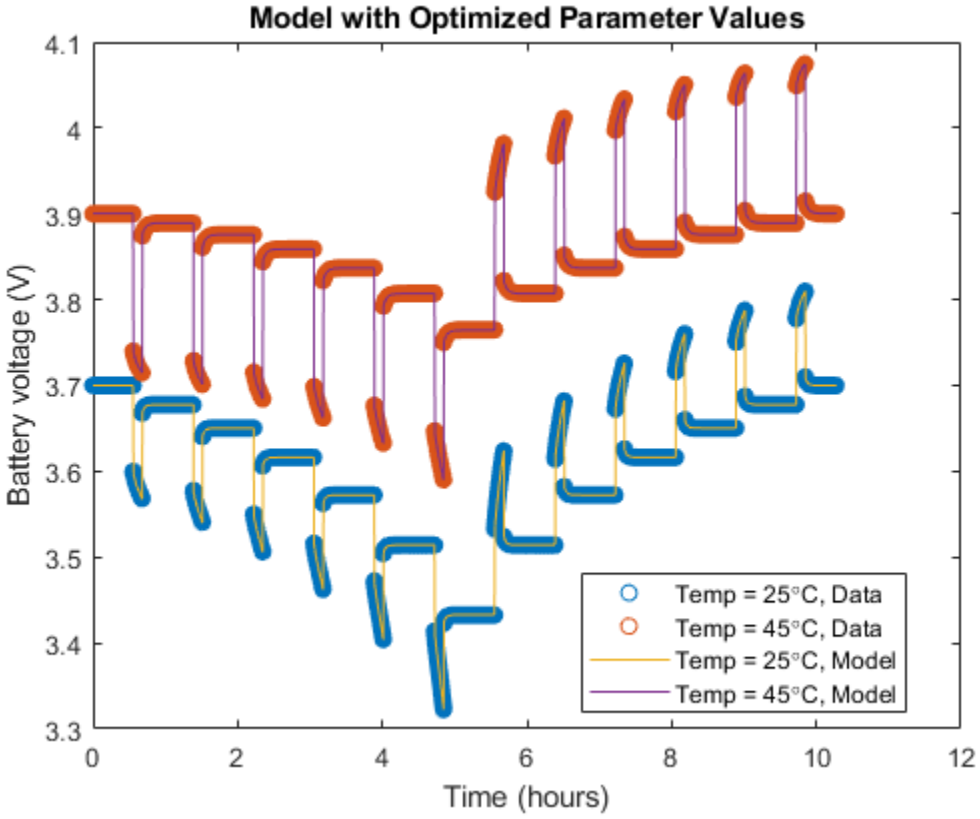
### Optimize Charge Dynamics Parameters for Second Temperature (Step 4)

```
% Find index into data for non-room temperatures

% Use only one current pulse for optimizing the charge dynamics
i_pos=battery_data(idx_data).i.*(battery_data(idx_data).i>=0);
a=find(diff(i_pos)>0,2);
b = find(diff(battery_data(idx_data).i));
c = fix((b(find(b<a(1),1,'last'))+a(1))/2);
assignin('base','i_data',battery_data(idx_data).i(c+1:a(2)));
assignin('base','v_data',battery_data(idx_data).v(c+1:a(2)));
assignin('base','t_data',battery_data(idx_data).t(1:length(i_data)));
assignin('base','T_data',battery_data(idx_data).T*ones(length(t_data),1));
assignin('base','T0',battery_data(idx_data).T);
assignin('base','Ts',t_data(2)-t_data(1))

% Find Battery initial charge before optimizing charge dynamics parameters
assignin('base','ParsList',{'charge'});
InitGuessCharge = OptParsMain(3);
OptCharge = fminsearch(@ee_battery_lse, InitGuessCharge,              ...
    optimset('TolX', 1e-3));
assignin('base','AH0',OptCharge);

% Optimize Battery charge dynamics parameters
assignin('base','ParsList', [ParsListTemp(2) ParsListTemp(4:5)]);
InitGuessTempDyn = [OptParsTemp(2) InitGuessTemp(4:5)];
OptParsTempDyn = fminsearch(@ee_battery_lse, InitGuessTempDyn,              ...
    optimset('TolX', 1e-3));

% Update Battery block with optimized parameters
OptParsTempDyn = [OptParsTemp(1) OptParsTempDyn(1) OptParsTemp(3) OptParsTempDyn(2:3)];
Pars = reshape([ParsListTemp; cellstr(num2str(OptParsTempDyn'))'],1,[]);
for k=1:2:length(Pars)
    evalin('base',[Pars{k} '=' Pars{k+1} ';'])
end
assignin('base','AH0',AH*battery_data(idx_data).SOC0);

% Display optimized parameters
fprintf(['Optimized temperature dependent parameters for the Battery, '           ...
   'including charge dynamics, are:\n']);
fprintf('\t%5s = %s\n', Pars{:});

clear i_data v_data t_data T_data Ts
clear i_pos a b c
clear k
clear OptCharge OptParsMain OptParsTemp OptParsTempDyn
clear Pars ParsList ParsListTemp InitGuessCharge InitGuessTemp InitGuessTempDyn
```

```
Optimized temperature dependent parameters for the Battery, including charge dynamics, are:
    Vnom_T2 =    3.900266
    R1_T2 =  0.07979468
    V1_T2 =    3.813256
    Rp1_T2 = 0.007920818
    tau1_T2 =    160.2999
```

### Display Optimized Curves

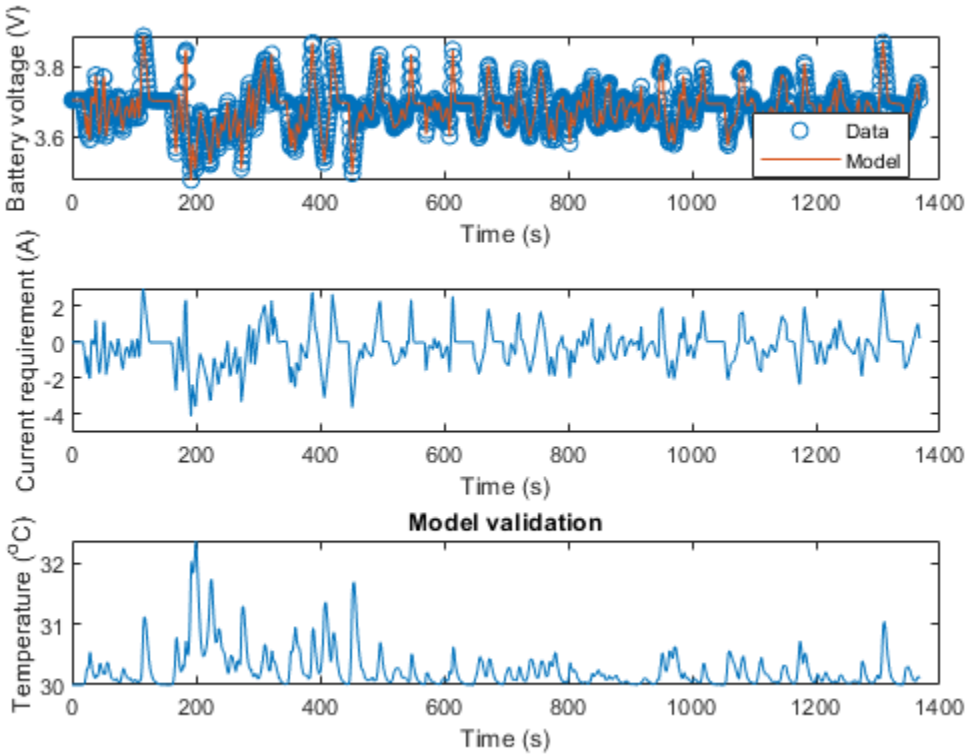
```
for idx_data = 1:num_lines
   assignin('base','t_data',battery_data(idx_data).t);
   assignin('base','i_data',battery_data(idx_data).i);
   assignin('base','T_data',battery_data(idx_data).T*ones(length(t_data),1));
   assignin('base','T0',battery_data(idx_data).T);
   assignin('base','Ts',t_data(2)-t_data(1));

   out = sim(Model);
   v_model{idx_data} = out.Vo.signals.values;
   t_model{idx_data} = out.Vo.time;
end
plot([battery_data(1:num_lines).t]/3600, [battery_data(1:num_lines).v], 'o', [t_model{:}]/3600,
xlabel('Time (hours)');
ylabel('Battery voltage (V)');
legend([legend_info_data legend_info_model], 'Location', 'Best');
title('Model with Optimized Parameter Values');
```




### Validation using a dynamic current cycle

```
idx_data = 3;
assignin('base','t_data',battery_data(idx_data).t);
assignin('base','i_data',battery_data(idx_data).i);
assignin('base','T_data',battery_data(idx_data).T);
assignin('base','T0',battery_data(idx_data).T(1));
assignin('base','Ts',t_data(2)-t_data(1));
assignin('base','AH0',AH*battery_data(idx_data).SOC0)
```

```
out = sim(Model);
subplot(3,1,1)
plot(t_data, battery_data(3).v, 'o', out.Vo.time, out.Vo.signals.values)
xlabel('Time (s)');
ylabel('Battery voltage (V)');
legend('Data','Model', 'Location', 'Best');
subplot(3,1,2)
plot(t_data,i_data)
xlabel('Time (s)');
ylabel('Current requirement (A)');
subplot(3,1,3)
plot(t_data,T_data)
xlabel('Time (s)');
ylabel('Temperature (^oC)');
title('Model validation');
```

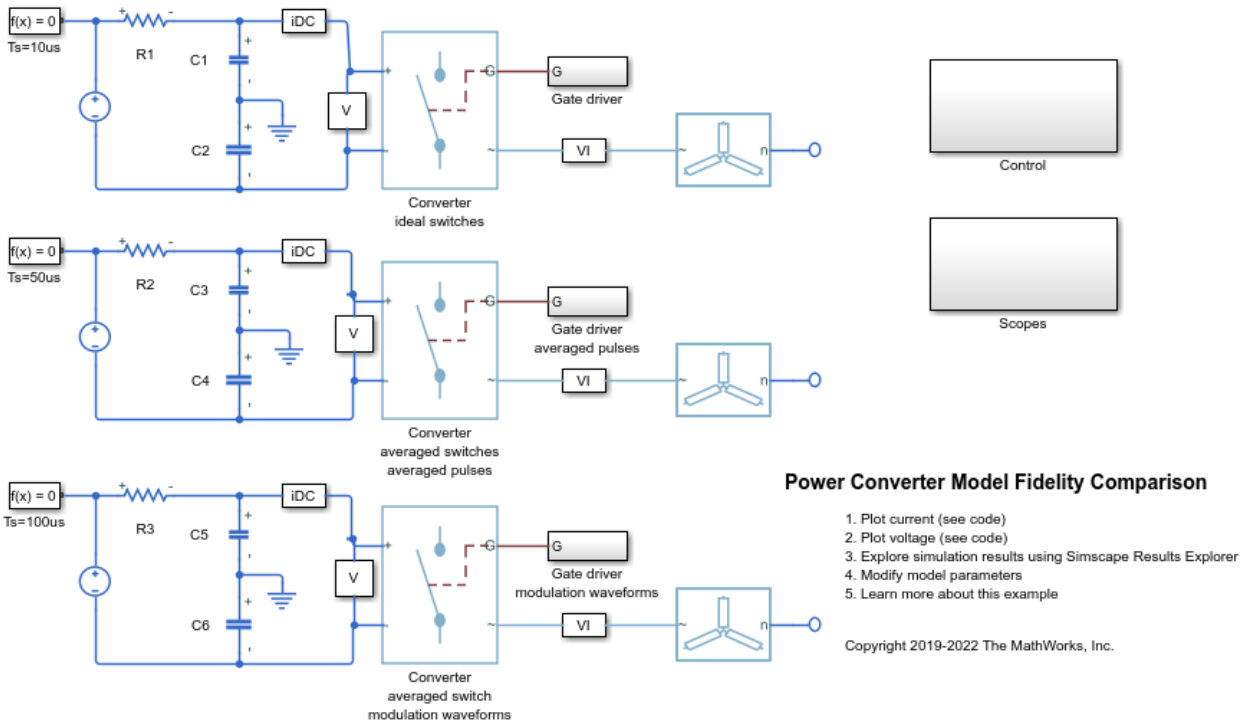



```
bdclose(Model)
clear num_lines legend_info_data legend_info_model out v_model t_model
clear battery_data idx_data Ts Model
```

# Power Converter Model Fidelity Comparison

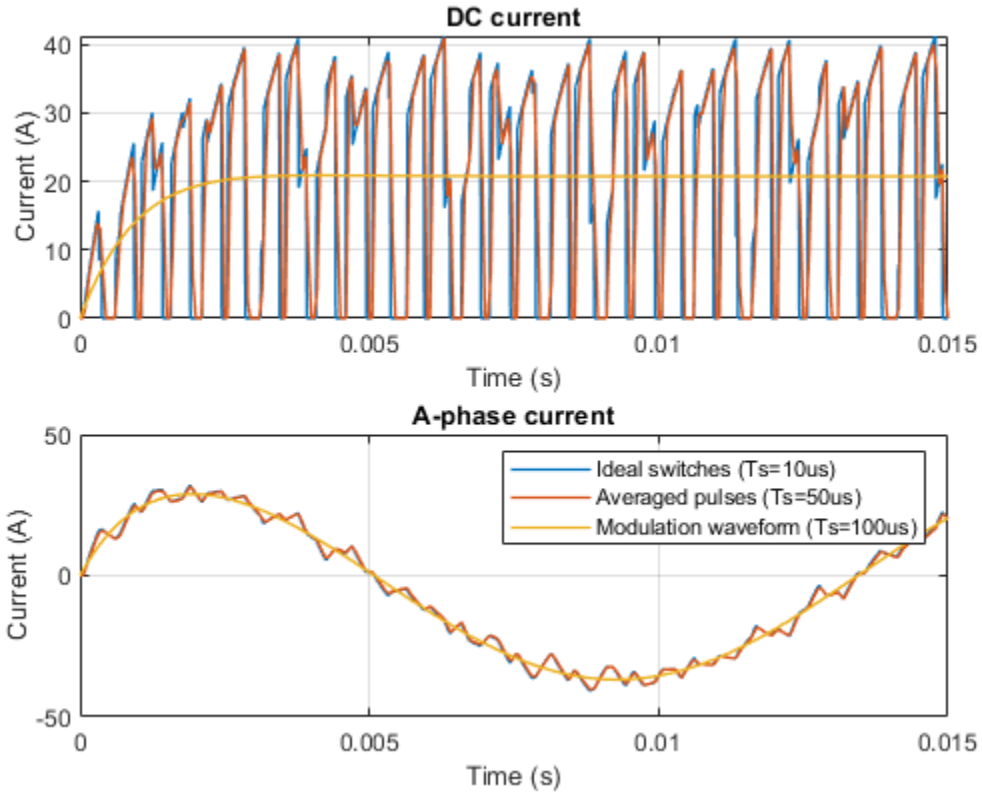
This example shows how to use different levels of fidelity in power converters. The system contains three converters. The top converter uses ideal switches and protection diodes at a 10 us sample time. To yield accurate results even though the model is under sampled at 50 us sample time, the middle converter uses averaged switches with averaged pulses. To further increase the sample rate and to operate as an ideal averaged converter, the bottom converter uses averaged switches and modulation waveforms instead of gate pulses. The Control subsystem contains a three-phase, two-level PWM waveform generator. The Scopes subsystem contains Scope blocks that allow you to see the simulation results.

### Model

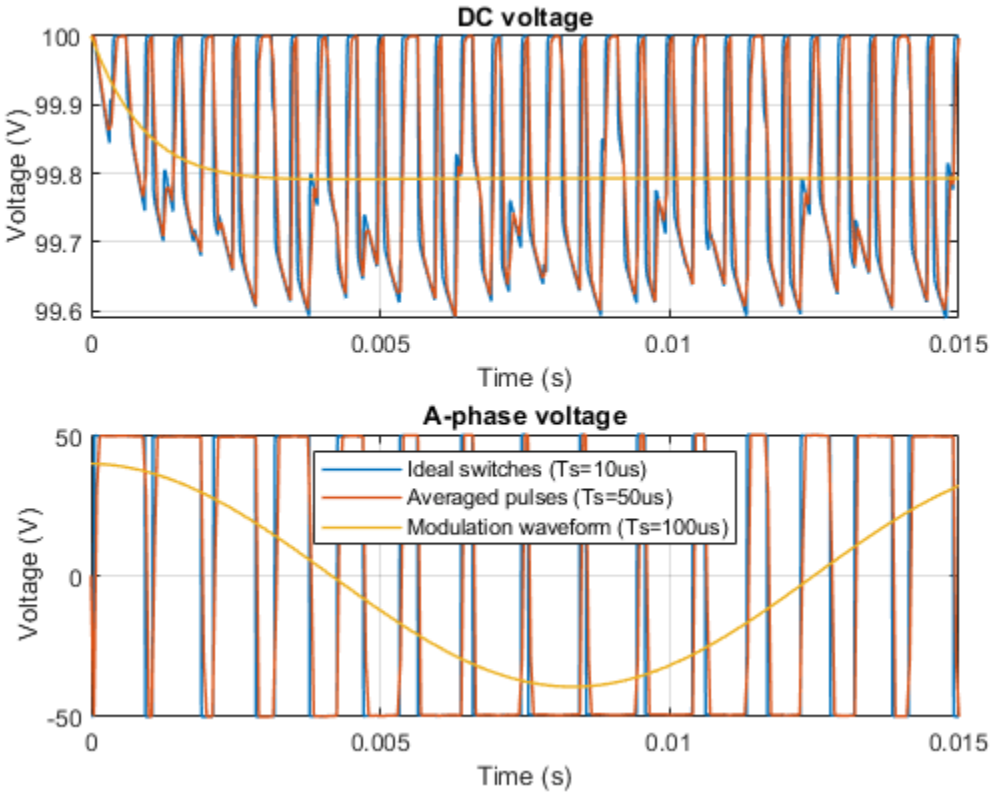



### Simulation Results from Simscape Logging

The plot below shows the comparison of the DC currents, as well as the AC a-phase currents.




The plot below shows the comparison of the DC voltage, as well as the AC a-phase voltage.


DC voltage
100
99.9
99.8
99.7
99.6
Voltage (V)
0
0.005
0.01
0.015
Time (s)
A-phase voltage
50
0
-50
Voltage (V)
Ideal switches (Ts=10us)
Averaged pulses (Ts=50us)
Modulation waveform (Ts=100us)
0
0.005
0.01
0.015
Time (s)

# Earthing Effects with Unbalanced Load

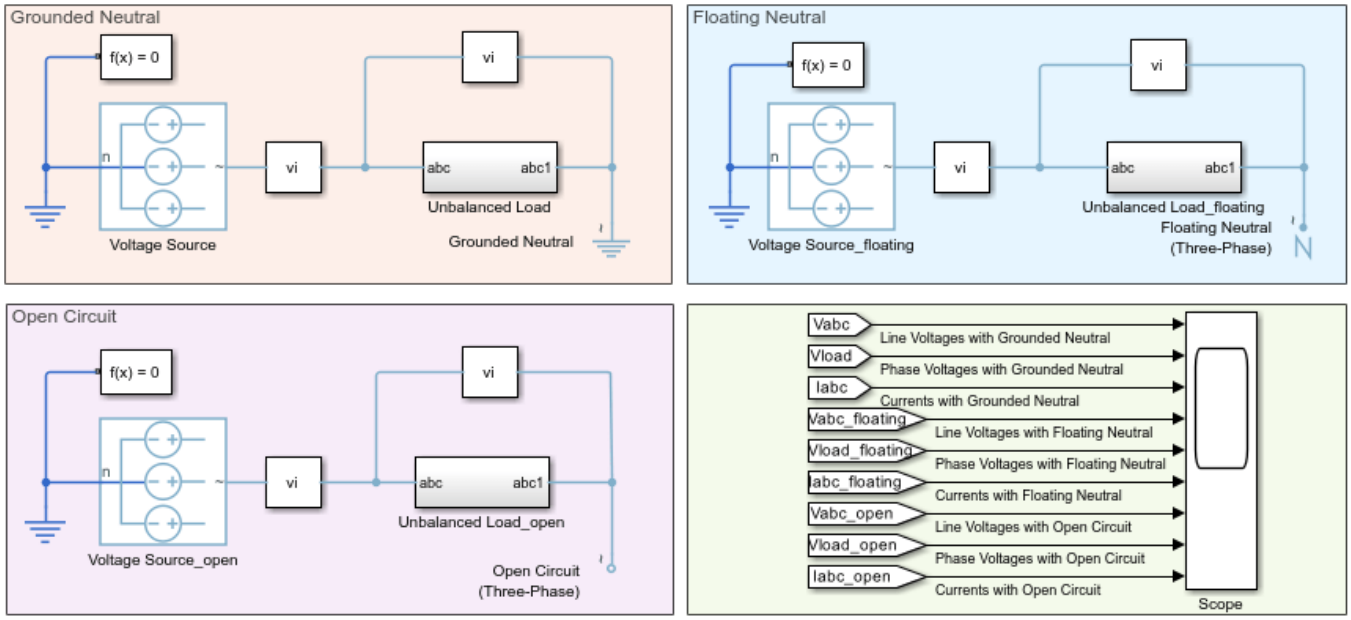
This example shows the effects of three different types of earthing connections on network voltages and currents.

In each three-phase network, an AC voltage source drives a resistive load. The load is modeled as subsystem that contains two Variable Resistor blocks from the Simscape™ Electrical™ Passive library.

The earthing connection in each network is modeled using one of these blocks from the Simscape Electrical Connections library: * Grounded Neutral * Floating Neutral * Open Circuit

Initially the load is balanced. At 0.05 s, the resistance on phase-a increases from 1 to 3 Ohms. At 0.07 s, the resistance on phase-c increases from 1 to 1.5 Ohms. The resistance on phase-b does not change during the simulation.

### Model

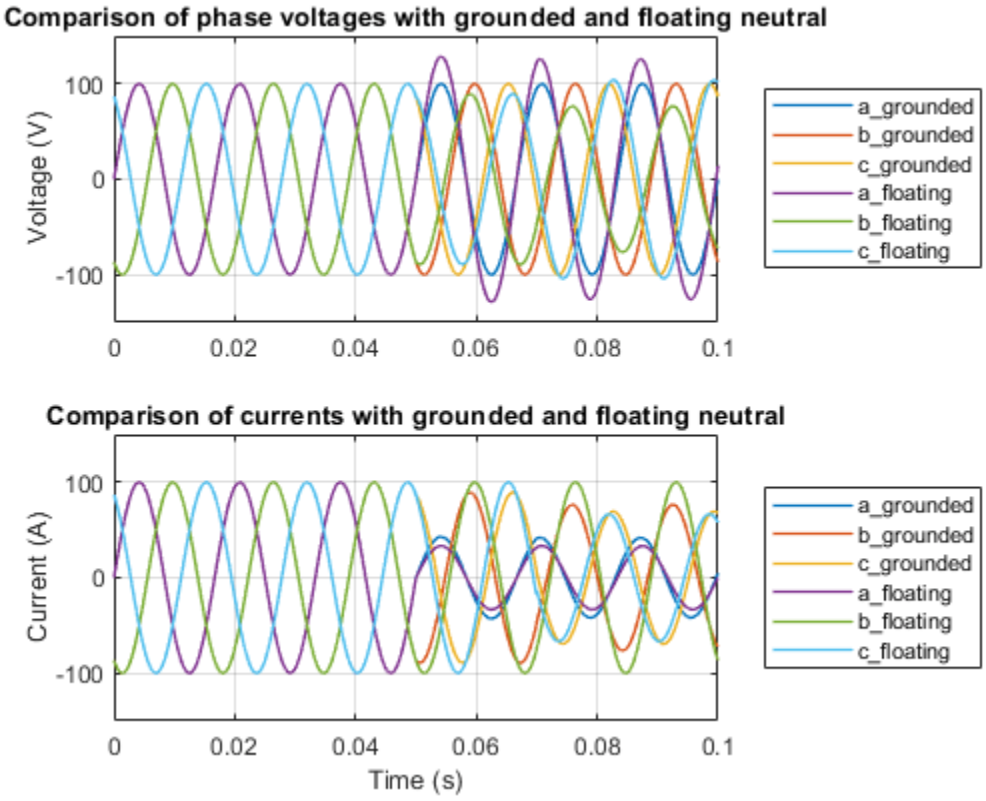



### Simulation Results from Simscape Logging

The plots below show the line voltage, phase voltages, and phase currents flowing through an unbalanced three-phase resistive load when it is connected to different earthing options.

Notice that until 0.05 seconds there is no difference between grounded neutral and floating neutral. At 0.05 seconds, the load becomes unbalanced and the currents flowing through it change in both the scenarios. Due to the load being effectively connected to ground, the phase voltage with a grounded neutral remains unaffected throughout the whole simulation.

In the floating neutral scenario, the phase voltage changes significantly as the load becomes unbalanced, accordingly to the different resistance values in each phase. In this case, if a system operator only monitors line-voltages, undesirable overvoltages could occur on individual phases without being observed. This could lead to overheating, accelerated breakdown of insulation, or other system problems.



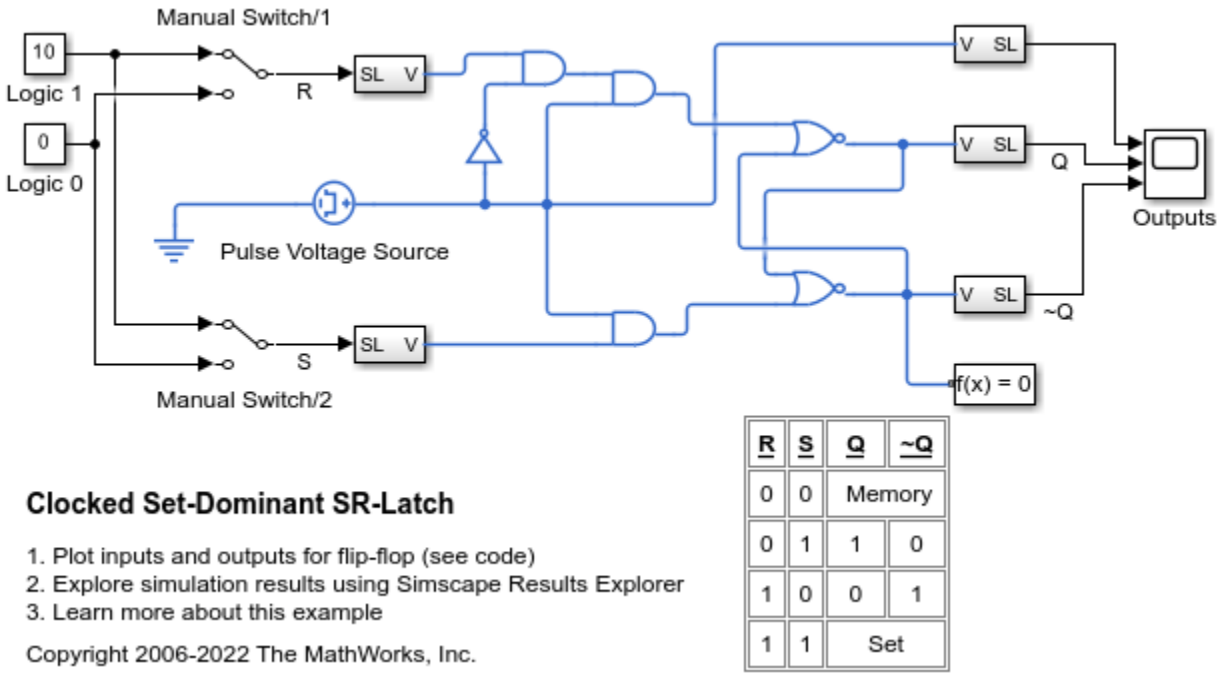

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 30 microseconds.

# Clocked Set-Dominant SR-Latch

This example shows how to model a Set-dominant SR-Latch from Simscape™ Electrical™ logic components. Initial conditions are passed to the relevant AND gates via the initialization commands of the switches.

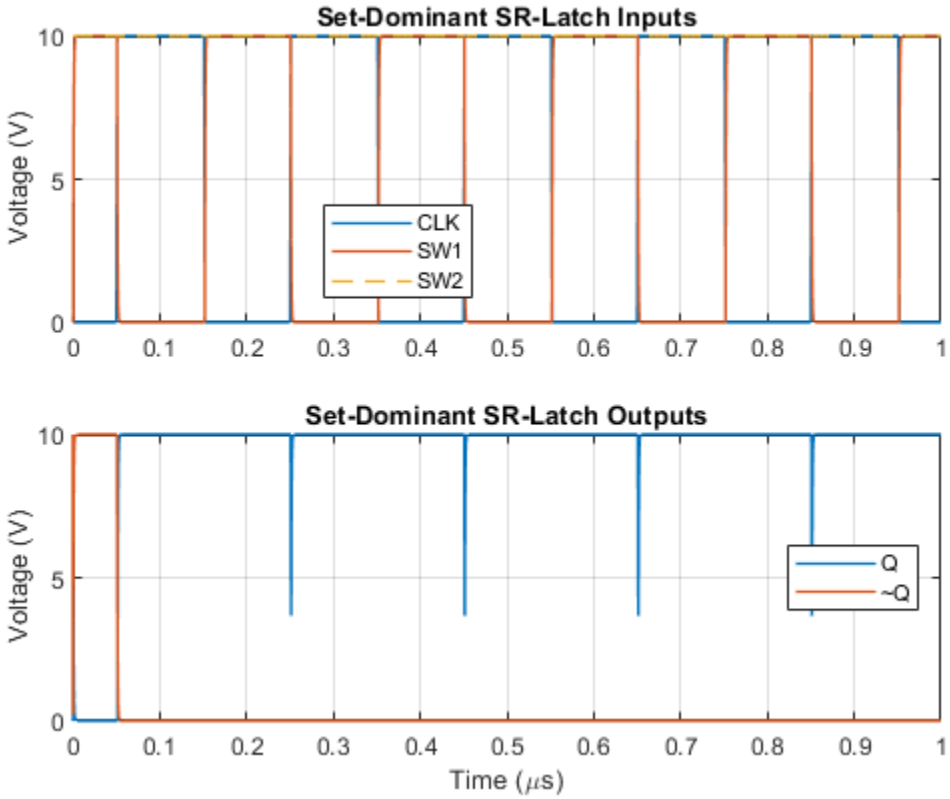
### Model




| R | S | Q | ~Q |
|---|---|---|---|
| 0 | 0 | Memory | |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | Set | |

### Simulation Results from Simscape Logging

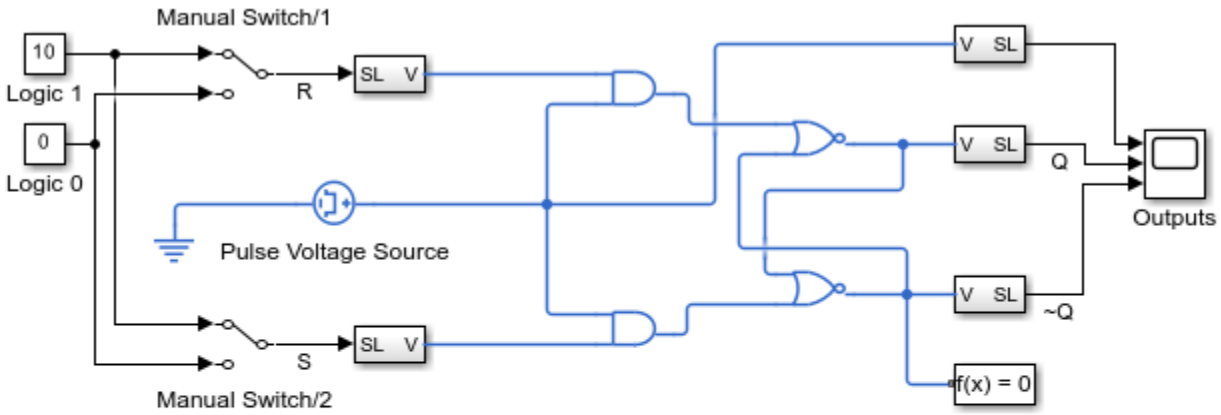
The plots below show the inputs and outputs for the Set-Dominant SR-Latch. Both inputs to the SR-Latch are set high, so its output state is set high for Q, and low for ~Q.


Set-Dominant SR-Latch Inputs
Voltage (V)
10
5
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
CLK
SW1
SW2
Set-Dominant SR-Latch Outputs
Voltage (V)
10
5
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Time (μs)
Q
~Q

# Clocked Reset-Dominant SR-Latch

This example shows how to model a Reset-dominant SR-Latch from Simscape™ Electrical™ logic components. Initial conditions are passed to the relevant AND gates via the initialization commands of the switches.

### Model




**Clocked Reset-Dominant SR-Latch**

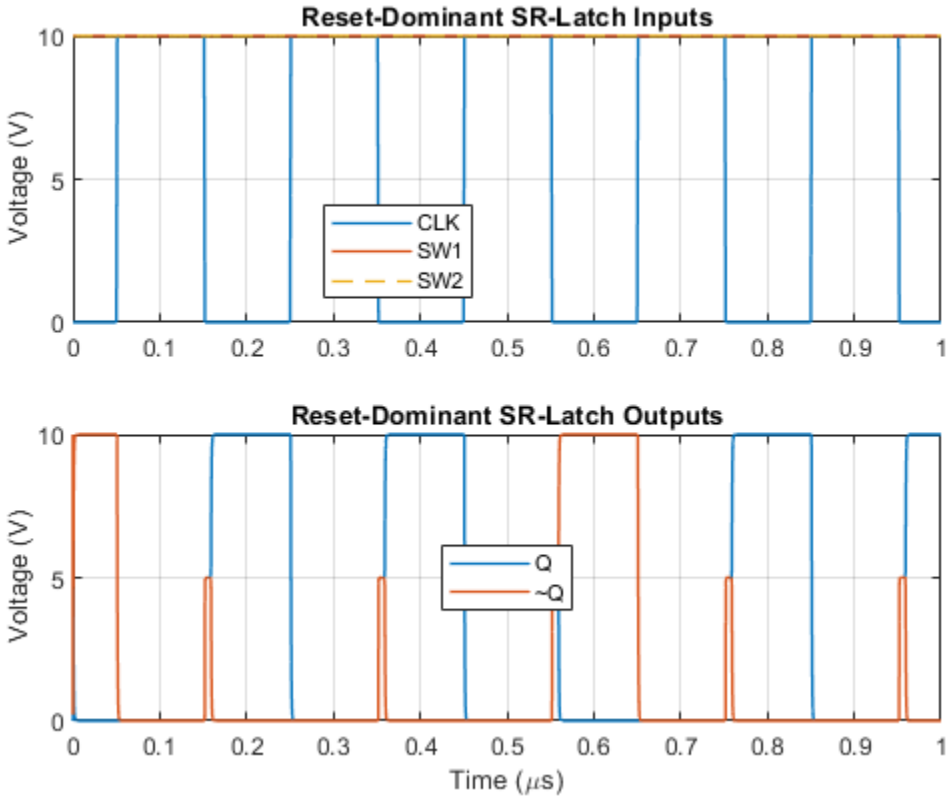
1. Plot inputs and outputs for flip-flop (see code)
2. Explore simulation results using Simscape Results Explorer
3. Learn more about this example

Copyright 2006-2022 The MathWorks, Inc.

| R | S | Q | ~Q |
|---|---|---|---|
| 0 | 0 | Memory | |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | Reset | |

### Simulation Results from Simscape Logging

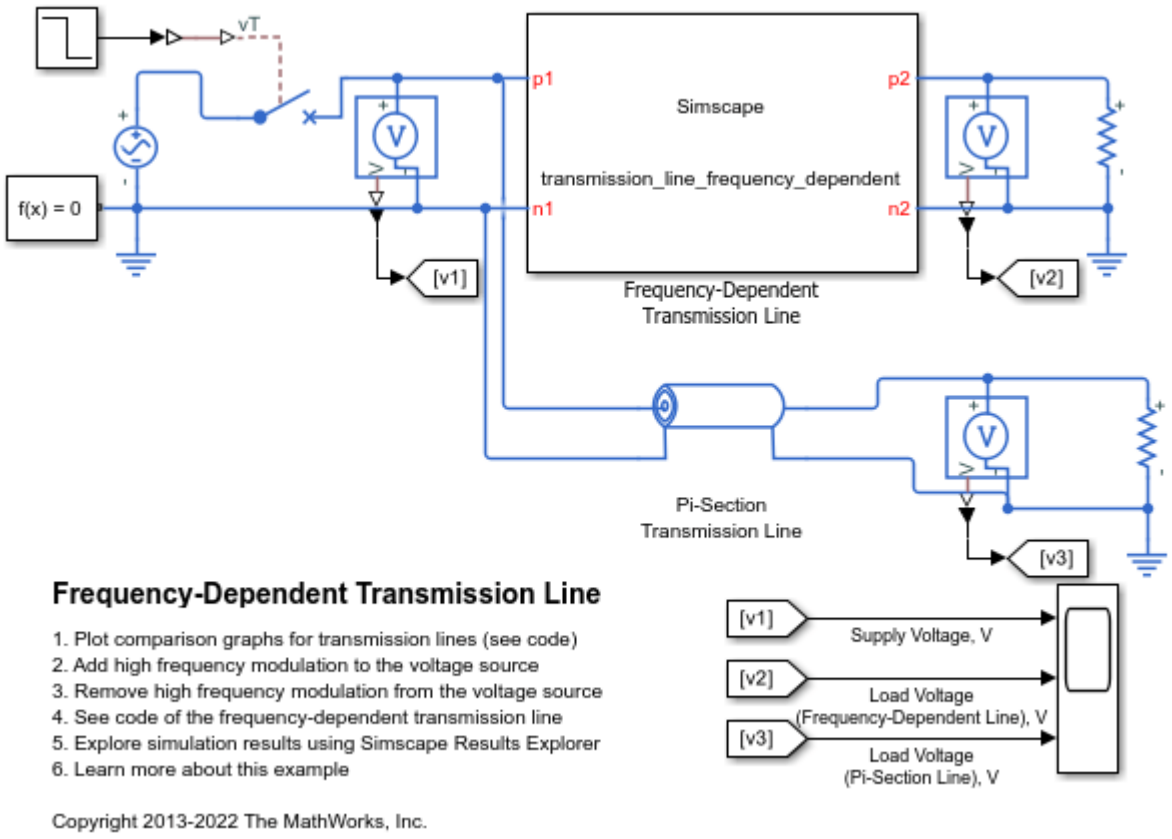
The plots below show the inputs and outputs for the Reset-Dominant SR-Latch. Both inputs to the SR-Latch are set high, and Q follows the opposite to the pulse signal. Output Q changes everytime the pulse is modified. The output ~Q stays low.


Reset-Dominant SR-Latch Inputs
Voltage (V)
10
5
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
CLK
SW1
SW2
Reset-Dominant SR-Latch Outputs
Voltage (V)
Q
~Q
Time (μs)

# Frequency-Dependent Transmission Line

This example shows a custom frequency-dependent transmission line model. The characteristic admittance and propagation function are first derived from the frequency-dependent resistance, reactance, and susceptance. The derived values are fitted using “RF Toolbox”. The Universal Line Model (ULM) [1] is then implemented in “Simscape” based on the fitted parameters. The results from the frequency-dependent transmission line model and the classic pi-section transmission line model are compared.

### Model




### Specification of Parameters

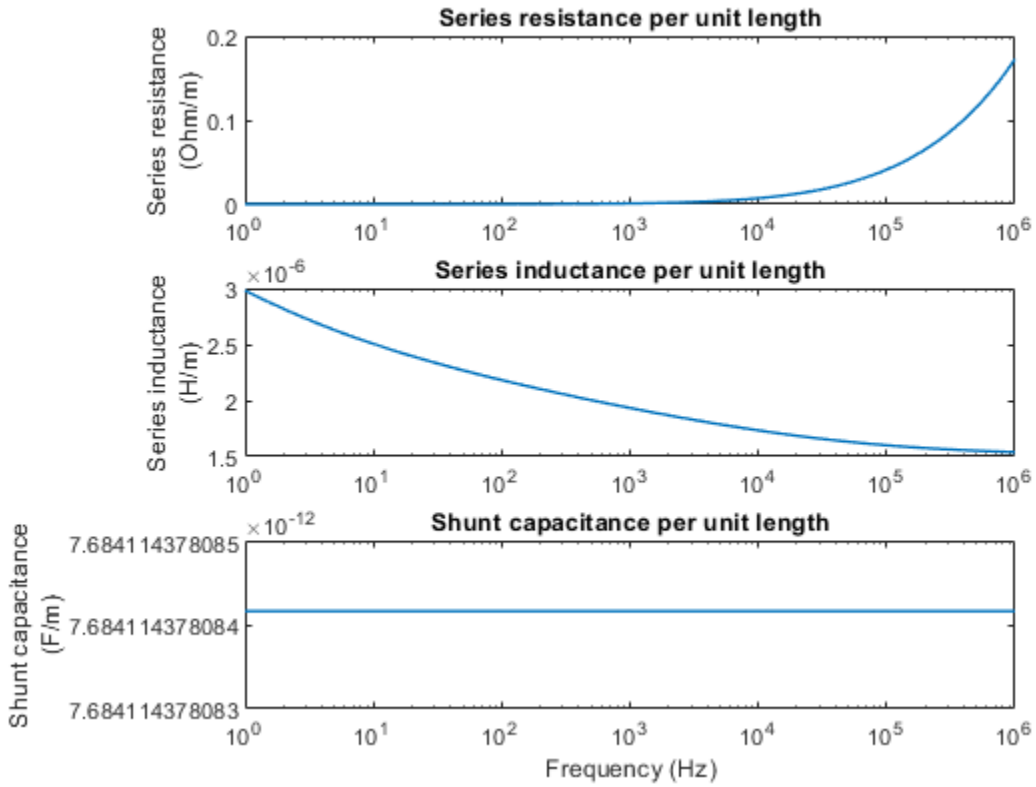
Import the frequency-dependent parameters of the transmission line. These parameters are computed for an overhead line that is 20 m above the ground [2]. The ground resistivity and the skin effect of the conductor are considered negligible. The following parameters are pre-computed for the simulation:

- Frequency-dependent series resistance per unit length, $R$
- Frequency-dependent series reactance per unit length, $X$
- Frequency-dependent shunt susceptance per unit length, $B$

- Corresponding frequency, $freq$
- Length of the transmission line, $len$

```
  Name         Size            Bytes  Class     Attributes

  B         1000x1              8000  double
  R         1000x1              8000  double
  X         1000x1              8000  double
  freq      1000x1              8000  double
  len          1x1                 8  double
```

The frequency-dependent R, L and C are shown in these figures:




### Characteristic Admittance and Propagation Function

The characteristic admittance is expressed as $Y_c = \sqrt{Y/Z}$, where $Z$ and $Y$ are the frequency-dependent series impedance and shunt admittance per unit length.

The propagation velocity is expressed as: $V = \omega / Imag(\Gamma)$, where $\Gamma = \sqrt{YZ}$ is the propagation constant, and $\omega$ is the corresponding angular velocity.

The propagation function, $H$, is then expressed as $H = e^{-\Gamma len}$.

### Rational Fitting for Characteristic Admittance

To convert the characteristic admittance to the rational form, use the rational fit function `rationalfit` (RF Toolbox) from “RF Toolbox”.

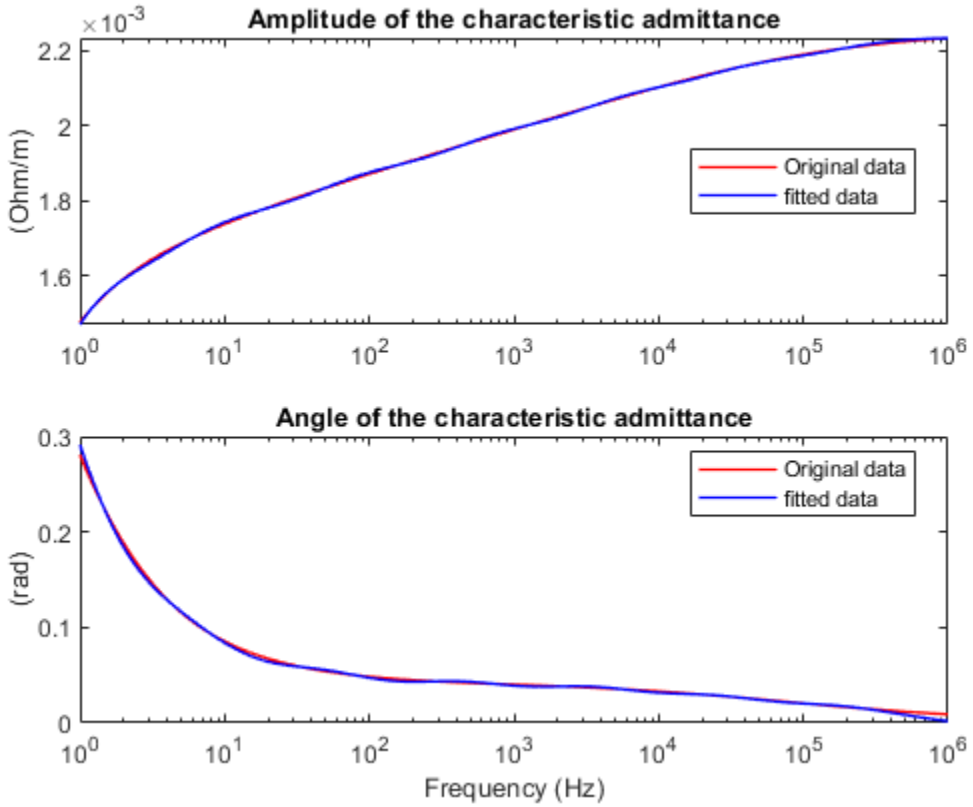
$$Y_c = \sum_{i=1}^{n} \frac{YcResidues_i}{s - YcPoles_i}$$

where:

- $n$ is the number of poles (order of the fit).
- $YcPoles_i$ is the $i_{th}$ pole of $Y_c$.
- $YcResidues_i$ is the $i_{th}$ residue of $Y_c$.

In this case, an eighth order fit is performed.

These figures show the comparison between the characteristic admittance before and after rational fitting.




### Rational Fitting for Propagation Function

The time delay of the propagation function is first removed to help reduce the order of rational fitting, where $\tau$ is the propagation time delay and $H_k$ is the propagation function without time delay. The time delay is represented by a delay unit in the model.

To convert the propagation function without time delay to the rational form, use the `rationalfit` (RF Toolbox) function from “RF Toolbox”.

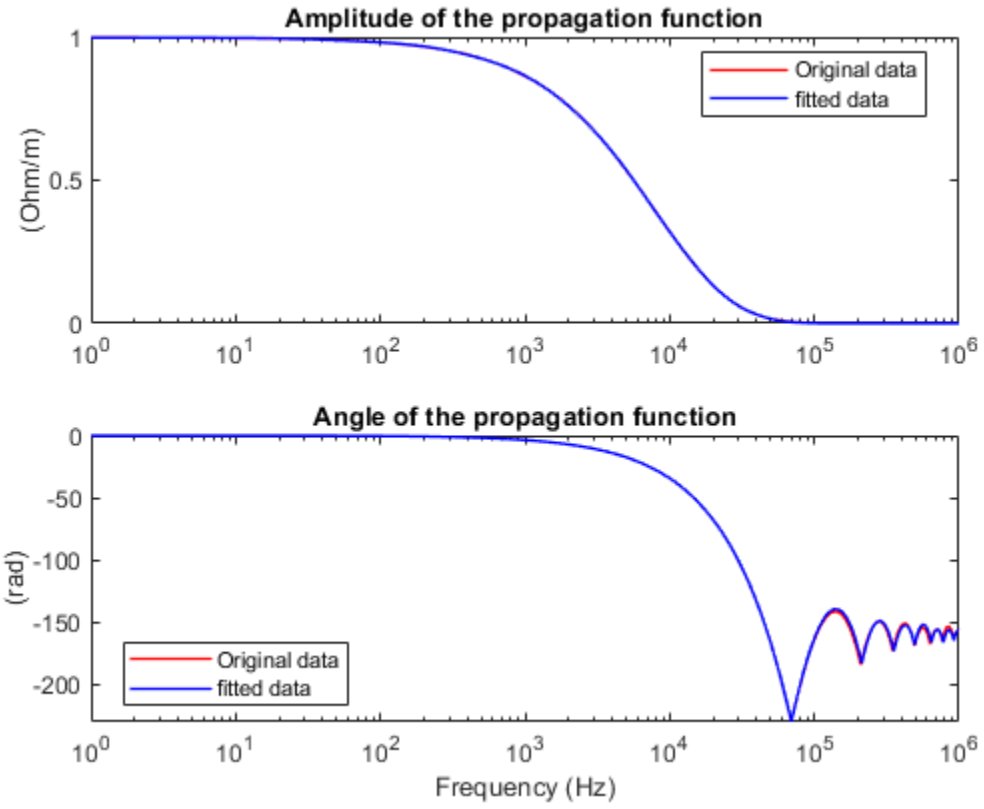
$$H_k = \sum_{i=1}^{n} \frac{HkResidues_i}{s - HkPoles_i}$$

where:

- $n$ is the number of poles (order of the fit).
- $HkPoles_i$ is the $i_{th}$ pole of $H_k$.
- $HkResidues_i$ is the $i_{th}$ residue of $H_k$.

In this case, an eighth order fit is performed.

These figures show that the propagation function H (with time delay) before and after rational fitting agree.

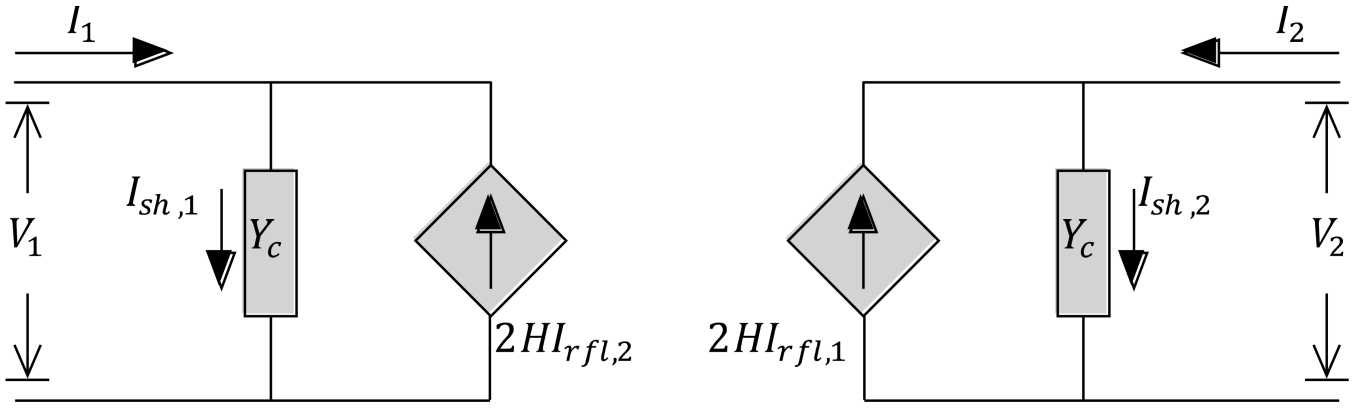



### Universal Line Model Implemented in Simscape

In this example, only a single conductor and the ground return is considered. The equivalent circuit of the line in Laplace domain can be deduced from the Universal Line Model (ULM) [1]. Key variables are:

- $V_j$ is the voltage at terminal $j$.

- $I_j$ is the current at terminal $j$.
- $I_{sh,j}$ is the shunt current at terminal $j$.
- $I_{rfl,j}$ is the reflected current from terminal $j$.
- $I_{aux,j}$ is the auxiliary current from terminal $j$.
- $H$ is the propagation function.




From this equivalent circuit, the system of equations can be written as:

- $I_1 = I_{sh,1} - I_{aux,2}$
- $I_2 = I_{sh,2} - I_{aux,1}$

where:

- $I_{aux,1} = 2HI_{rfl,1}$
- $I_{aux,2} = 2HI_{rfl,2}$
- $I_{rfl,1} = (I_1 + I_{sh,1})/2$
- $I_{rfl,2} = (I_2 + I_{sh,2})/2$
- $I_{sh,1} = Y_c V_1$
- $I_{sh,2} = Y_c V_2$

Considering the rational form of the characteristic admittance, the shunt current on terminal one is:

- $I_{sh,1} = Y_c V_1 = \sum_{i=1}^{n} W_i$
- $W_i = \frac{YcResidues_i V_i}{s - YcPoles_i}$

To transform these equations from Laplace domain to time domain, an inverse Laplace transform is performed. This transformation leads to:

- $i_{sh,1} = Y_c v_1 = \sum_{i=1}^{n} w_i$

- $\frac{dw_i}{dt} = YcPoles_i w_i + YcResidues_i v_1$

where $w_i$, $v_1$ and $i_{sh,1}$ are the time domain representation of $W_i$, $V_1$ and $I_{sh,1}$.

Similarly, considering the rational form of the propagation function, the auxiliary current on terminal one is:

- $I_{aux,1} = 2HI_{rfl,1} = 2\sum_{i=1}^{n} X_i$
- $X_i = \frac{HkResidues_i}{s - HkPoles_i} e^{-s\tau} I_{rfl,1}$

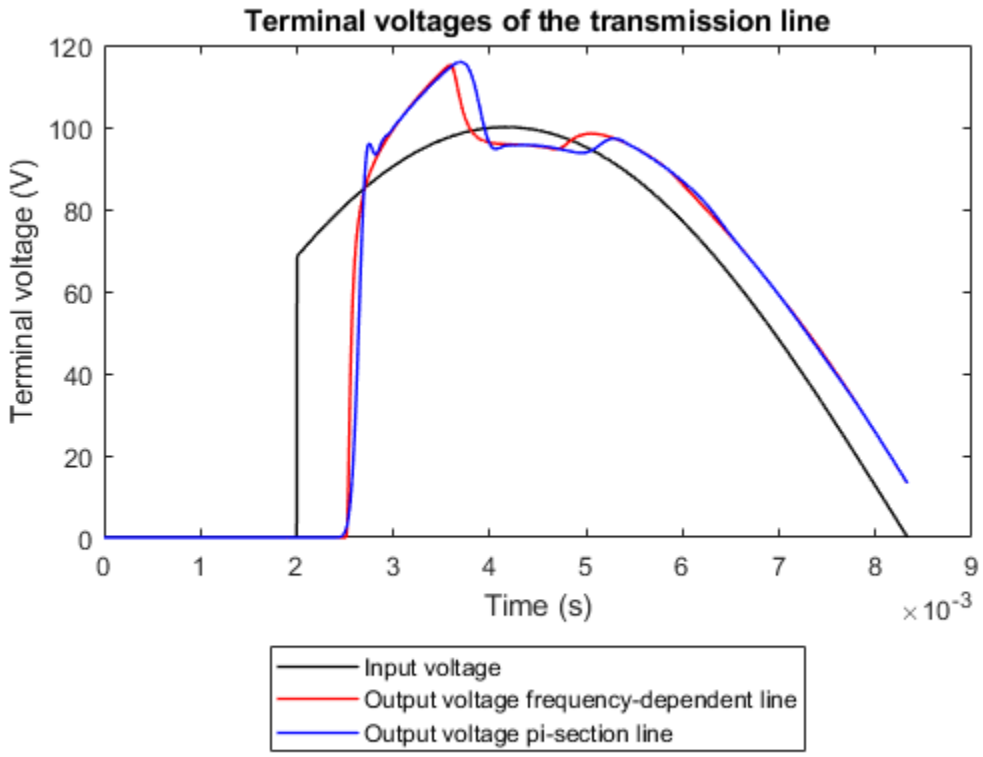
To transform these equations from Laplace domain to time domain, an inverse Laplace transform is performed. This transformation leads to:

- $i_{aux,1} = 2\sum_{i=1}^{n} x_i$
- $\frac{dx_i}{dt} = HkPoles_i x_i + HkResidues_i i_{rfl,1}(t - \tau)$

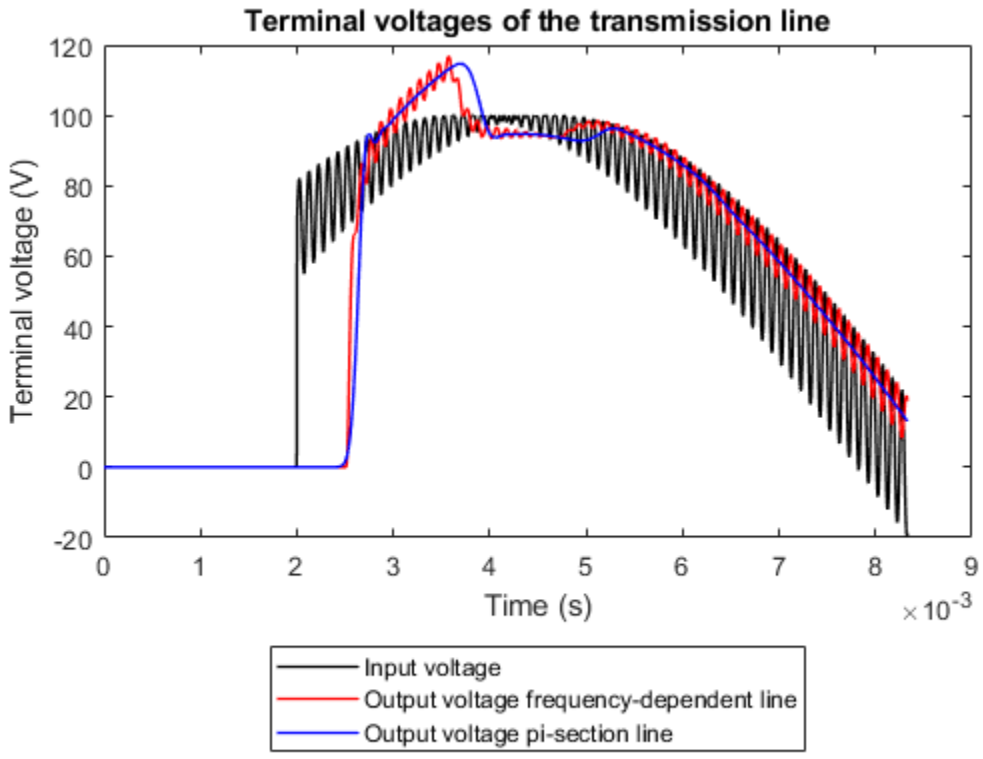
Currents on terminal two can be deduced using the same procedure. Time domain equations are then implemented in “Simscape” using Simscape language.

### Simulation Results from Simscape Logging

For the first simulation case, the voltage source is generating a 60 Hz sine wave. The Pi-Section Transmission Line uses an RLC parameterized assuming a 60 Hz input, which matches the frequency of the voltage source. The plot shows the input and output terminal voltages of the transmission line. The two models show good agreement at steady state.




For the second simulation case, the voltage source is generating a 60 Hz sine wave with a 10 kHz modulation. The Pi-Section Transmission Line still uses an RLC parameterized assuming a 60 Hz input. It is clear that the custom frequency-dependent transmission line model is suitable for broader band signals whereas the pi-section model is only applicable for extremely narrow band signals.




## References

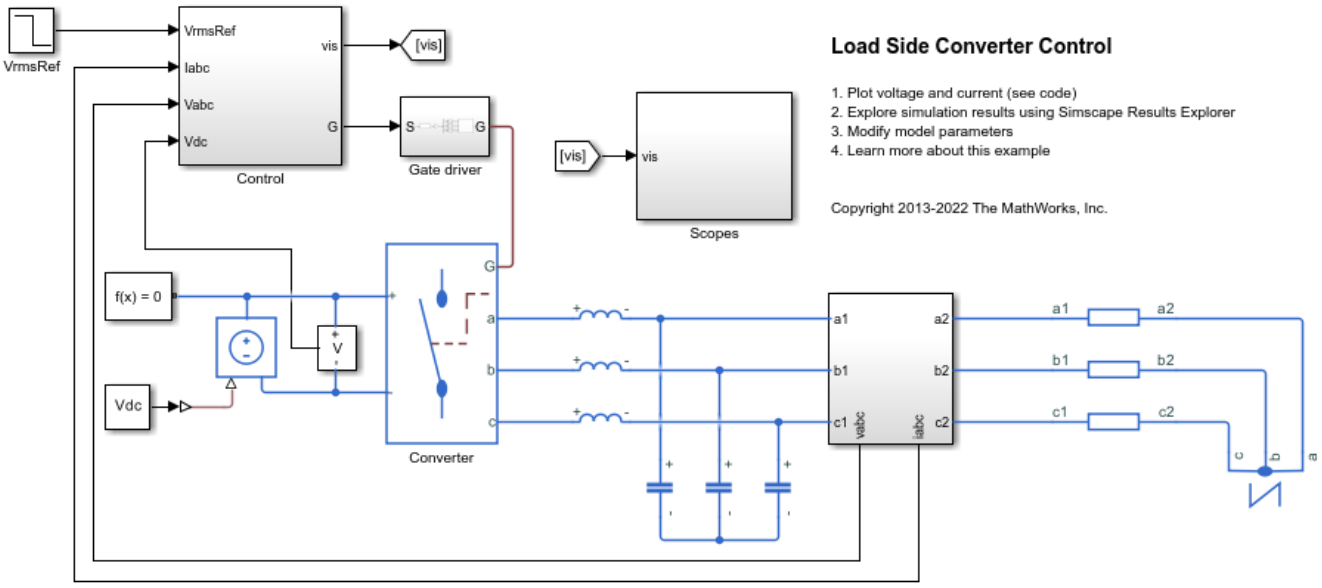
[1] Morched, Atef, Bjorn Gustavsen, and Manoocher Tartibi. "A universal model for accurate calculation of electromagnetic transients on overhead lines and underground cables." IEEE Transactions on Power Delivery 14.3 (1999): 1032-1038.

[2] Dommel, Herman W. "Overhead line parameters from handbook formulas and computer programs." IEEE Transactions on Power Apparatus and Systems 2 (1985): 366-372.

# Load Side Converter Control

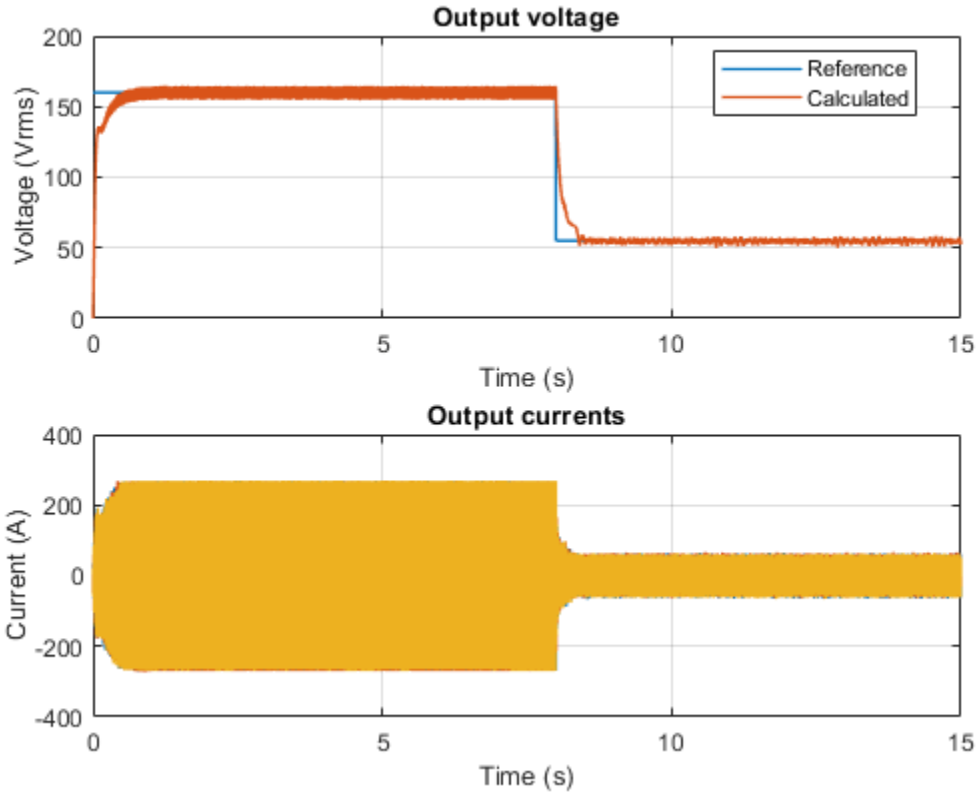
This example shows how to control the RMS voltage in a load-side converter. The load is provided by a three-phase series RL element. The Control subsystem uses a PI-based cascade control structure with two control loops, an outer voltage control loop and an inner current control loop. The simulation uses step references. The Scopes subsystem contains scopes that allow you to see the simulation results.

### Model




### Simulation Results from Simscape Logging

The plot below shows the output RMS voltage and the AC phase currents.



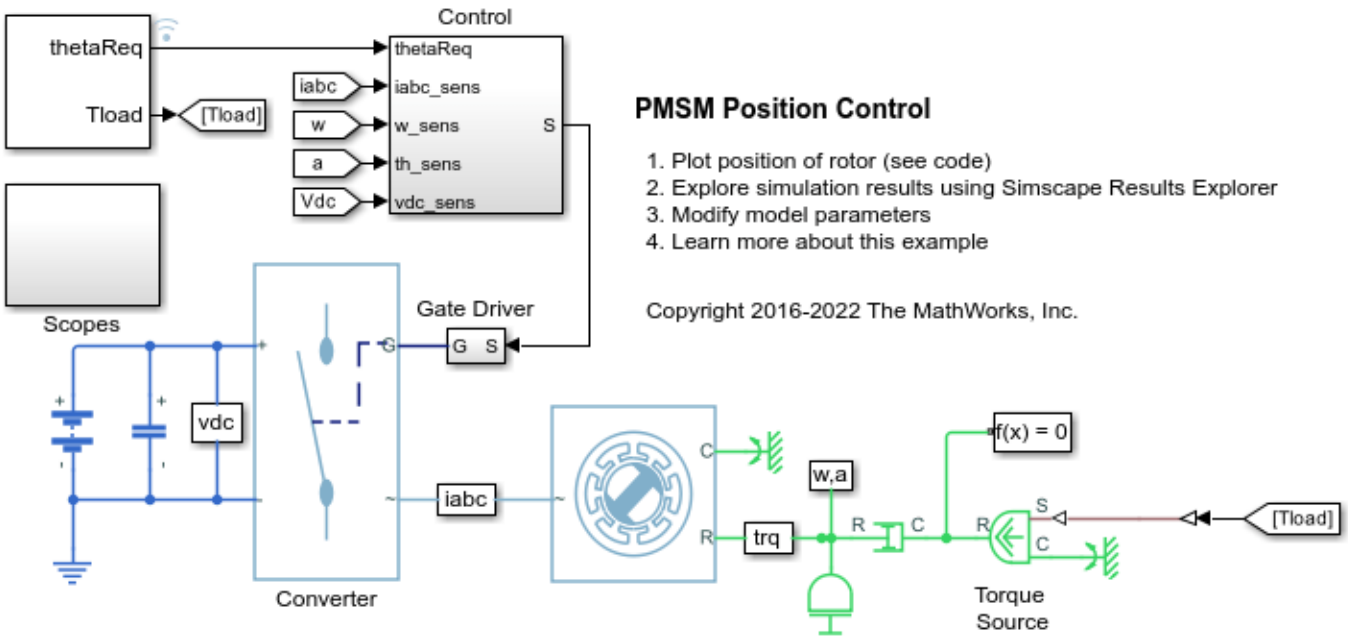

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 50 microseconds.

# PMSM Position Control

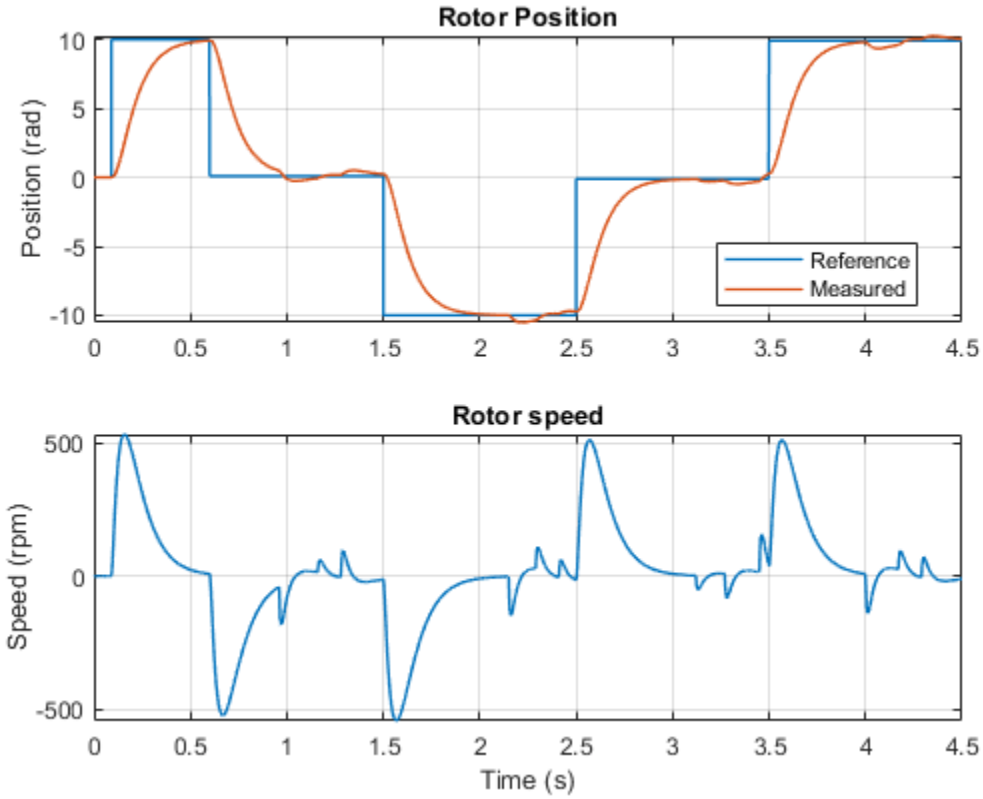
This example shows how to control the rotor position in a PMSM based electrical drive. An ideal torque source provides the load. The Control subsystem uses a cascade control structure with two control loops, an outer loop for position and speed control and an inner loop for current control. An optimal state-feedback linear quadratic regulator controls the position and speed. A Luenberger observer estimates the load. The inner current-control loop is implemented using PI controllers. The PMSM is fed by a controlled three-phase inverter. The simulation uses step references. The Scopes subsystem contains scopes that allow you to see the simulation results.

### Model




### Simulation Results from Simscape Logging

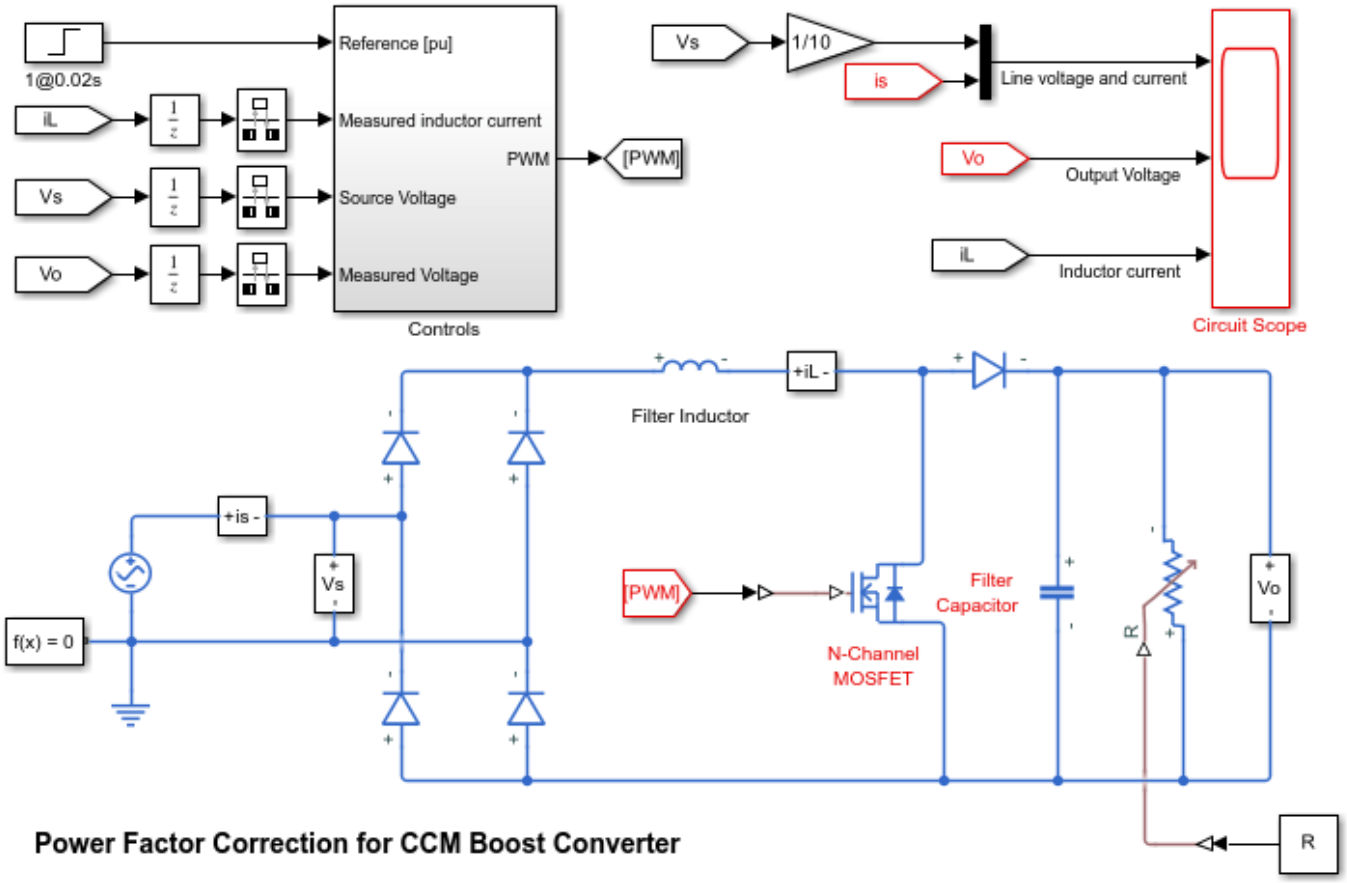
The plot below shows the requested and measured angle for the test and the rotor speed in the electric drive.


Rotor Position
Position (rad)
10
5
0
-5
-10
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
Reference
Measured
Rotor speed
Speed (rpm)
500
0
-500
Time (s)

# Power Factor Correction for CCM Boost Converter

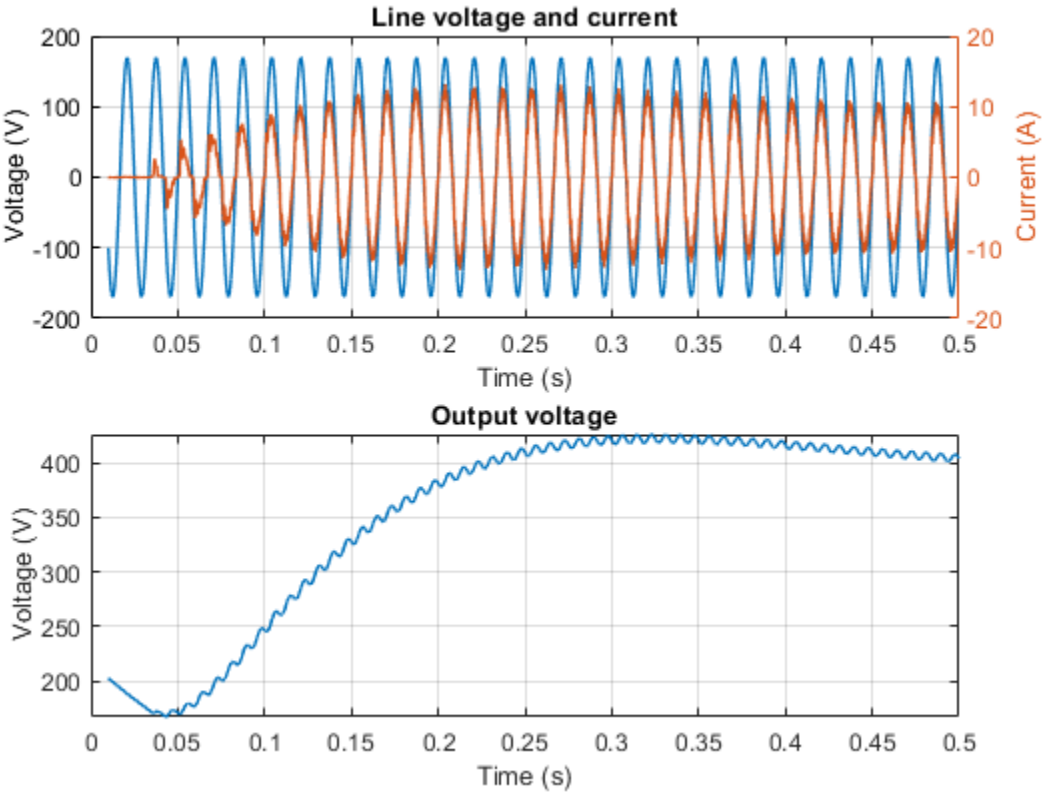
This example shows how to correct the power factor using a PFC pre-converter. This technique is useful when non-linear impedances, such as Switch Mode Power Supplies, are connected to an AC grid. As the current flowing through the inductor is never zero during the switching cycle, the boost converter operates in Continuous Conduction Mode (CCM). The inductor current and the output voltage profiles are controlled using simple integral control. During start up, the reference output voltage is ramped up to the desired voltage.

### Model




### Simulation Results from Simscape Logging

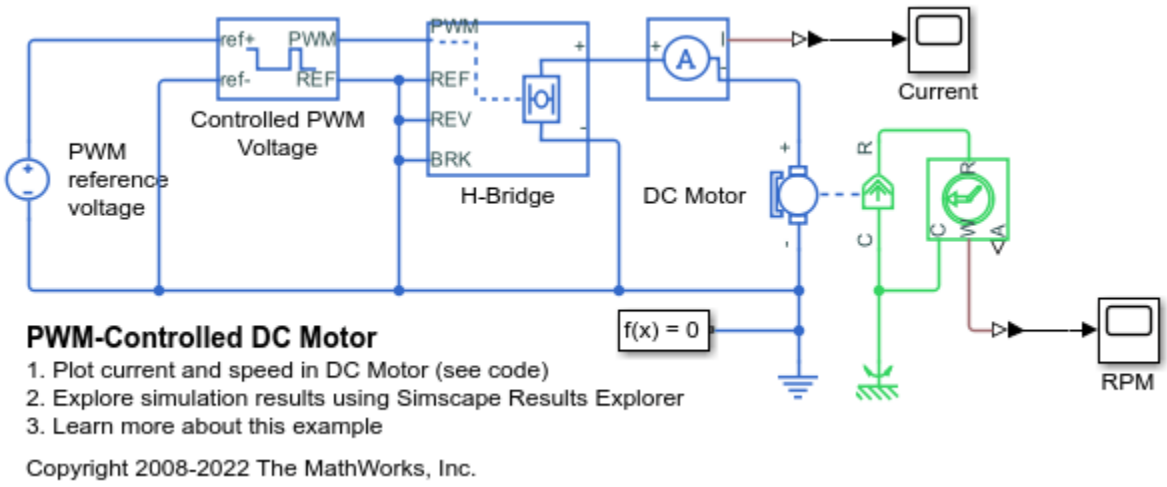
The plot below shows the line voltage and current and the output voltage.


Line voltage and current
Voltage (V)
Current (A)
Time (s)
Output voltage
Voltage (V)
Time (s)

# PWM-Controlled DC Motor

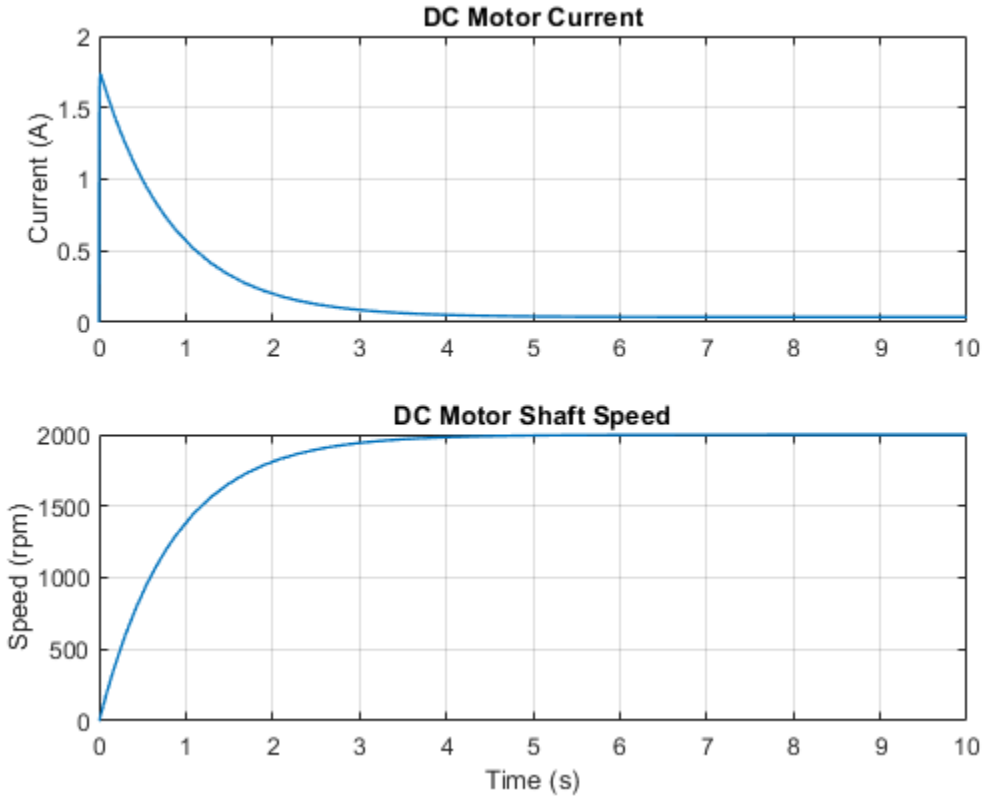
This model shows how to use the Controlled PWM Voltage and H-Bridge blocks to control a motor. The DC Motor block uses manufacturer datasheet parameters, which specify the motor as delivering 10W mechanical power at 2500 rpm and no-load speed as 4000 rpm when run from a 12V DC supply. Hence if the PWM reference voltage is set to its maximum value of +5V, then the motor should run at 4000 rpm. If it is set to +2.5V, then it should run at approximately 2000 rpm. The Simulation model parameter is set to Averaged for both the Controlled PWM Voltage and H-Bridge blocks, resulting in fast simulation. To validate the averaged behavior, change the Simulation mode parameter to PWM in both blocks.

### Model




### Simulation Results from Simscape Logging

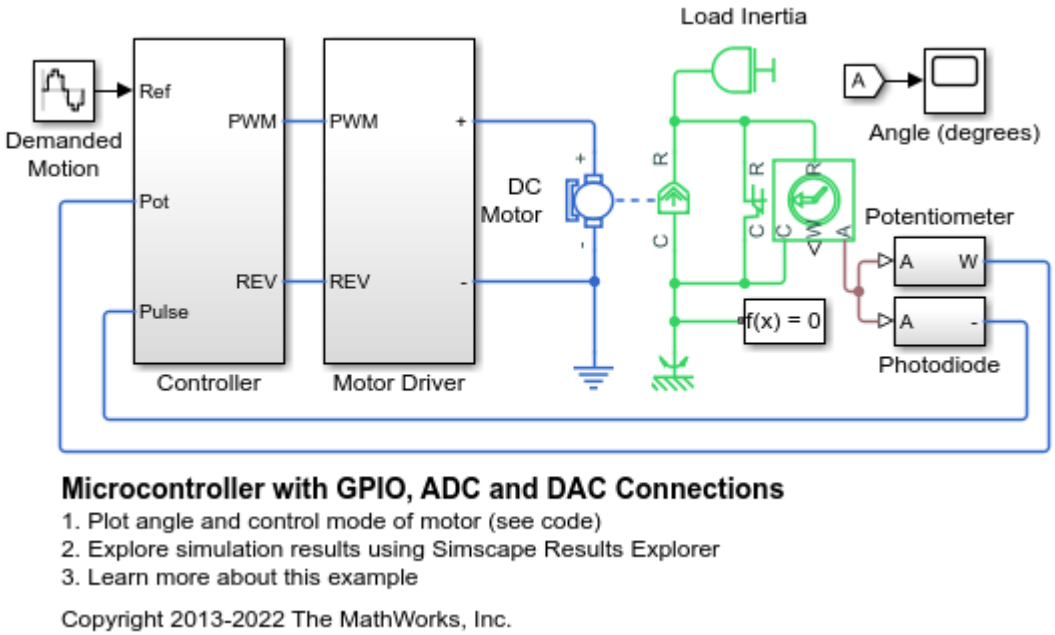
The plot below shows the current passing through the motor and the speed of the motor shaft.


DC Motor Current
Current (A)
0
0.5
1
1.5
2
0
1
2
3
4
5
6
7
8
9
10
DC Motor Shaft Speed
Speed (rpm)
0
500
1000
1500
2000
Time (s)

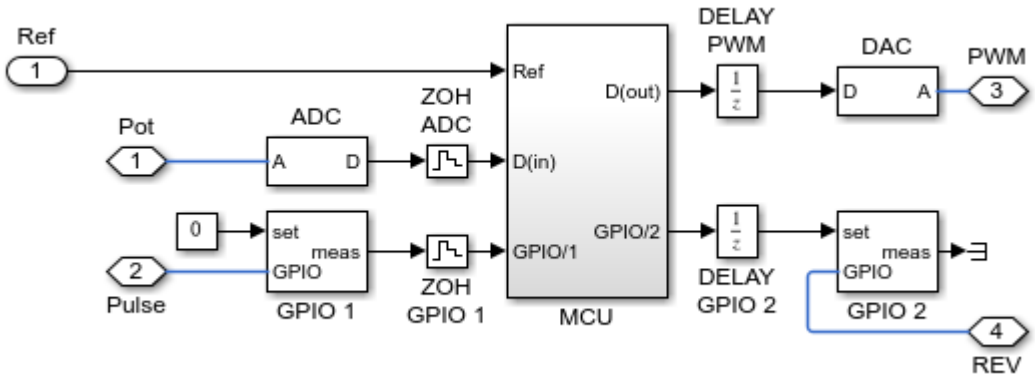
# Microcontroller with GPIO, ADC and DAC Connections

This example shows how to model the interface between a microcontroller unit (MCU) and a physical system. Here the microcontroller's GPIO, ADC and DAC connections are used to control a DC motor and connected load with limited angle travel. Load angle measurement is via a potentiometer sensor. This measurement is calibrated by initially ramping the rotor position until the photodiode detects the zero-angle light pulse from the LED. Once calibrated the MCU commands a 0.1Hz 45 degree amplitude sinusoid.

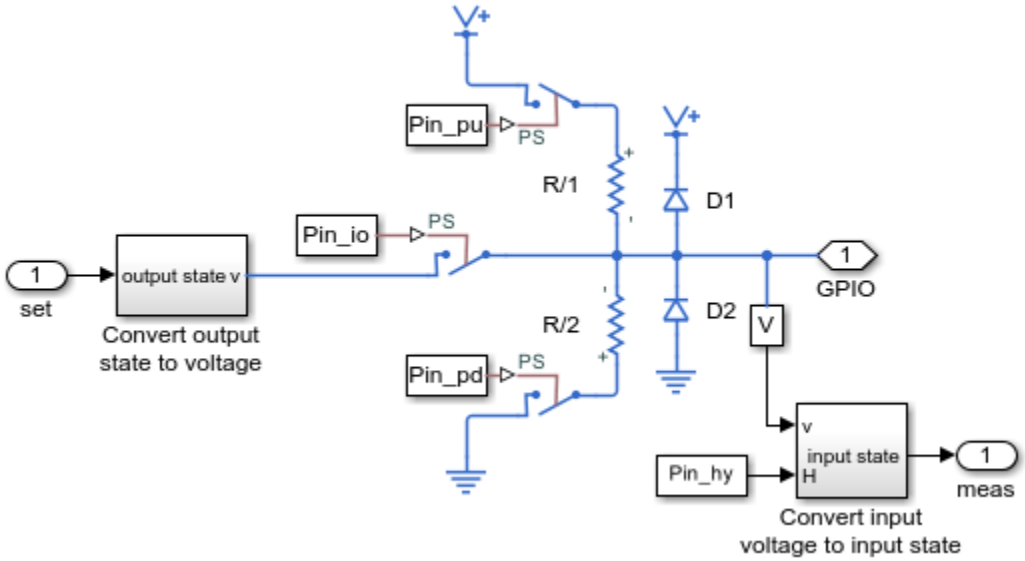
### Model



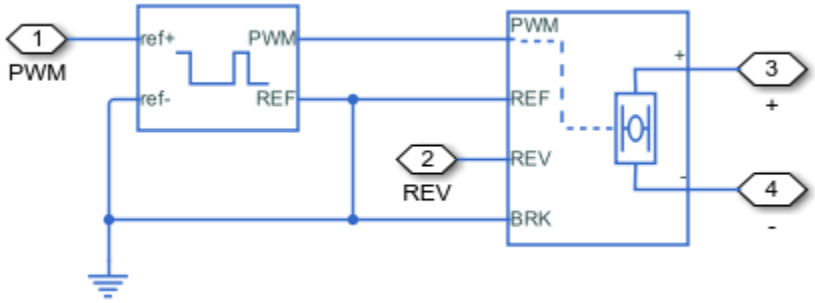

### Controller Subsystem



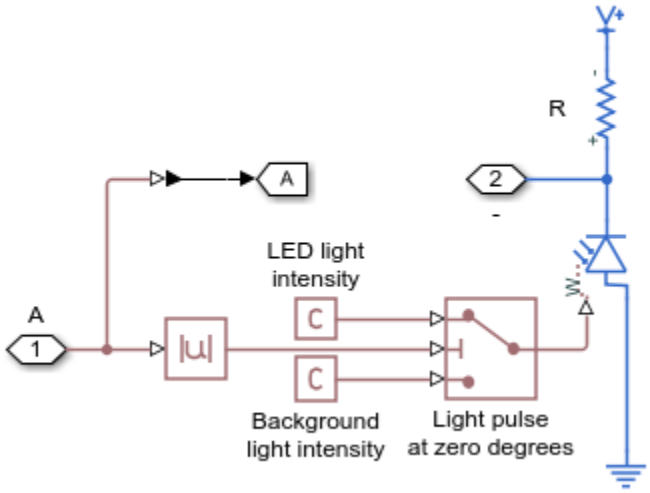
### GPIO 1 Subsystem



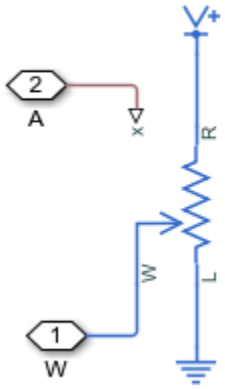

### Motor Driver Subsystem



### Photodiode Subsystem

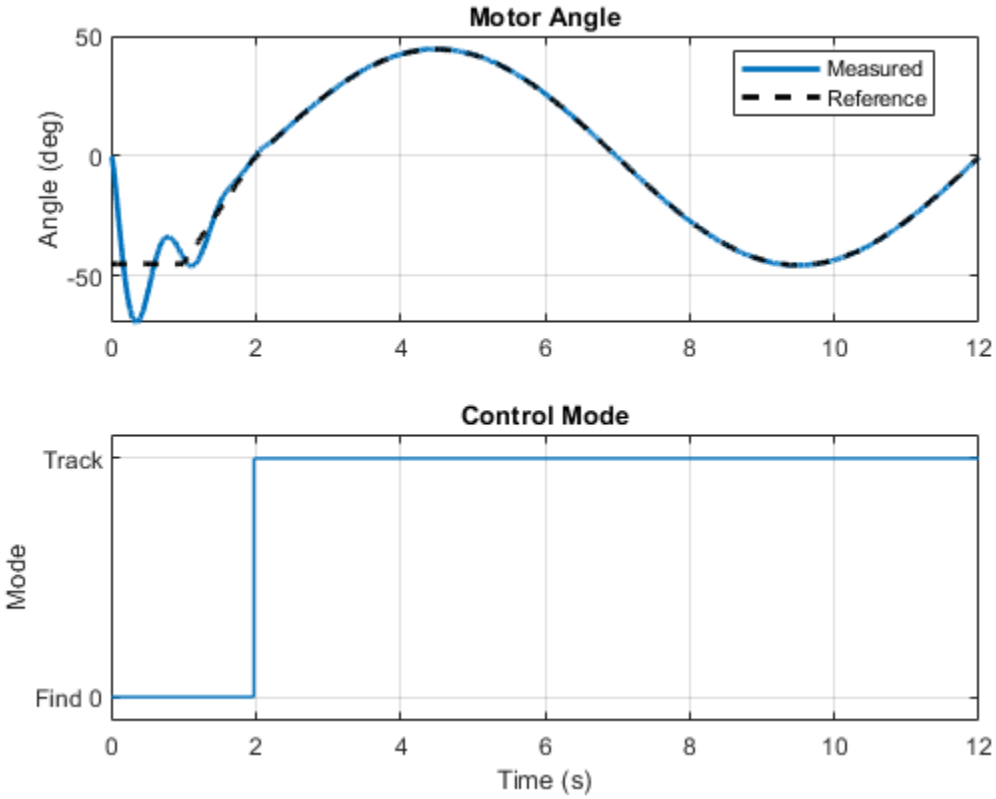



### Potentiometer Subsystem




### Simulation Results from Simscape Logging

The plot below shows the performance of the motor control system. At the start of simulation, the controller spins the motor until it finds a known position of the motor shaft (0 degrees). After that, the controller tracks the reference signal.




### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 300 microseconds.

# Motor Torque-Speed Curves

This example shows a comparison of the torque-speed characteristics for five different motor types. To select the motor type, right-click on the Electric Motor block, select Variant->Override using and then the desired motor. All motors have been sized for roughly the same mechanical power rating.

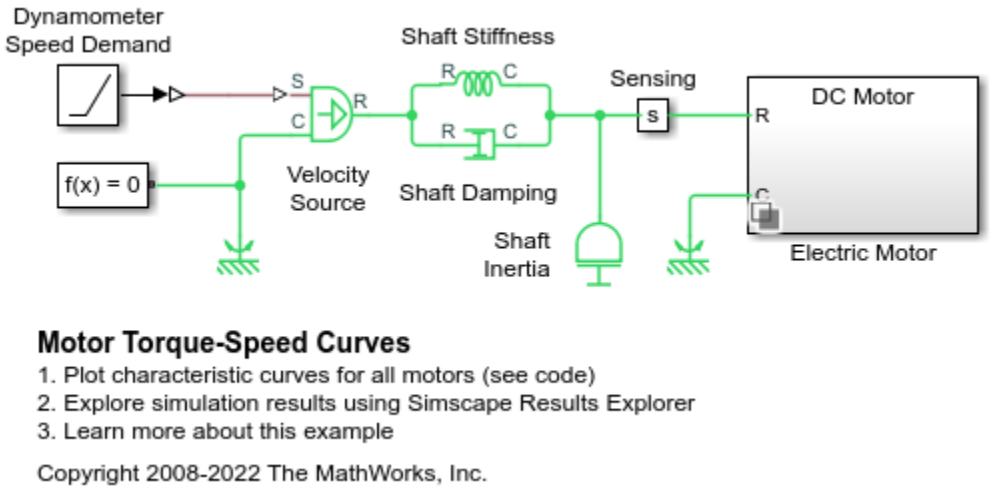
To run and compare all five motors, click on the hyperlink to plot characteristic curves. Understanding the torque-speed and power-speed behavior of a motor is critical when choosing a motor for a given application. Modify this model and its supporting library to compare specific motors based on their datasheet values. Critical parameters are typically starting torque (torque at zero RPM), and peak/maximum power and the corresponding RPM.

The DC Motor has a linear torque-speed characteristic, and power peaks at about half the maximum RPM. The Shunt Motor also has a linear torque-speed characteristic and similarly-shaped power curve. In practice a shunt motor cannot run at its peak power or torque as the currents involved are too large. This particular motor is designed to deliver a maximum 50 Watts of mechanical power at 4000 rpm.

The Universal Motor torque-speed characteristic follows 1/x^2 relationship. In this case, the relatively large starting torque is achieved with realistic current values. The power is just peaking at around 6000rpm. The Induction Motor torque-speed curve is typical of this type of motor having a low starting torque, and a peak torque just below synchronous speed. Note that both motoring and generating regions are plotted.

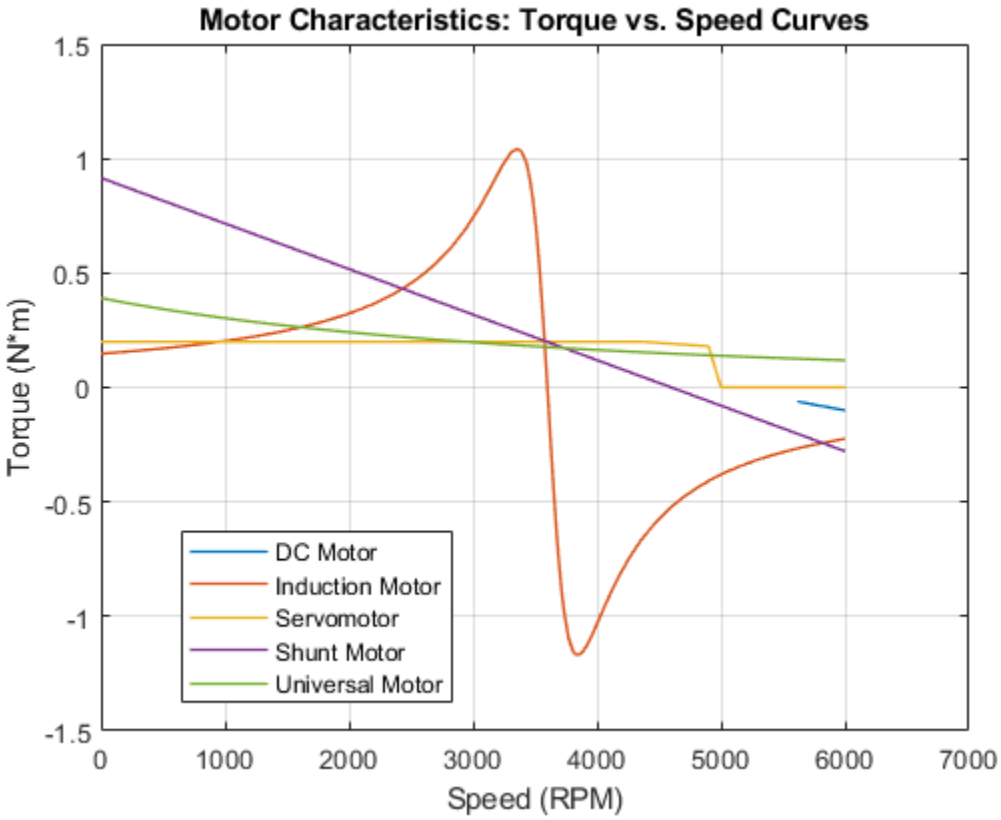
The Servomotor is a controlled servomotor with a maximum torque of 0.25 Nm. The torque demand is set to 0.2 Nm, which is achievable up to 4350rpm above which envelope limiting occurs. To see the permissible torque-envelope, set the Electric Motor variant choice to 'Servomotor', open the Electric Motor block and double click on the Servomotor block. The envelope is defined by the first two block parameters.

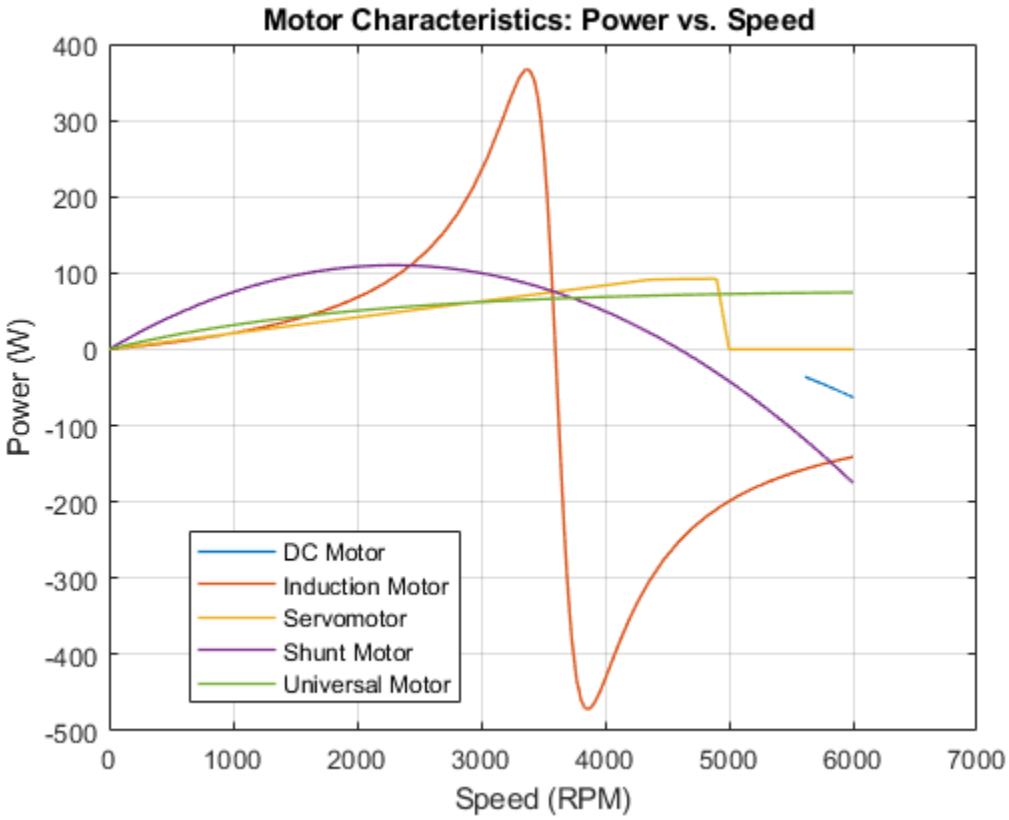
### Model




### Simulation Results from Simscape Logging

The plots below show characteristic curves for different types of motors.


Motor Characteristics: Torque vs. Speed Curves
Torque (N*m)
Speed (RPM)
1.5
1
0.5
0
-0.5
-1
-1.5
0
1000
2000
3000
4000
5000
6000
7000
DC Motor
Induction Motor
Servomotor
Shunt Motor
Universal Motor


Motor Characteristics: Power vs. Speed
Power (W)
Speed (RPM)
400
300
200
100
0
-100
-200
-300
-400
-500
0
1000
2000
3000
4000
5000
6000
7000
DC Motor
Induction Motor
Servomotor
Shunt Motor
Universal Motor

# Linear Electric Actuator (Motor Model)

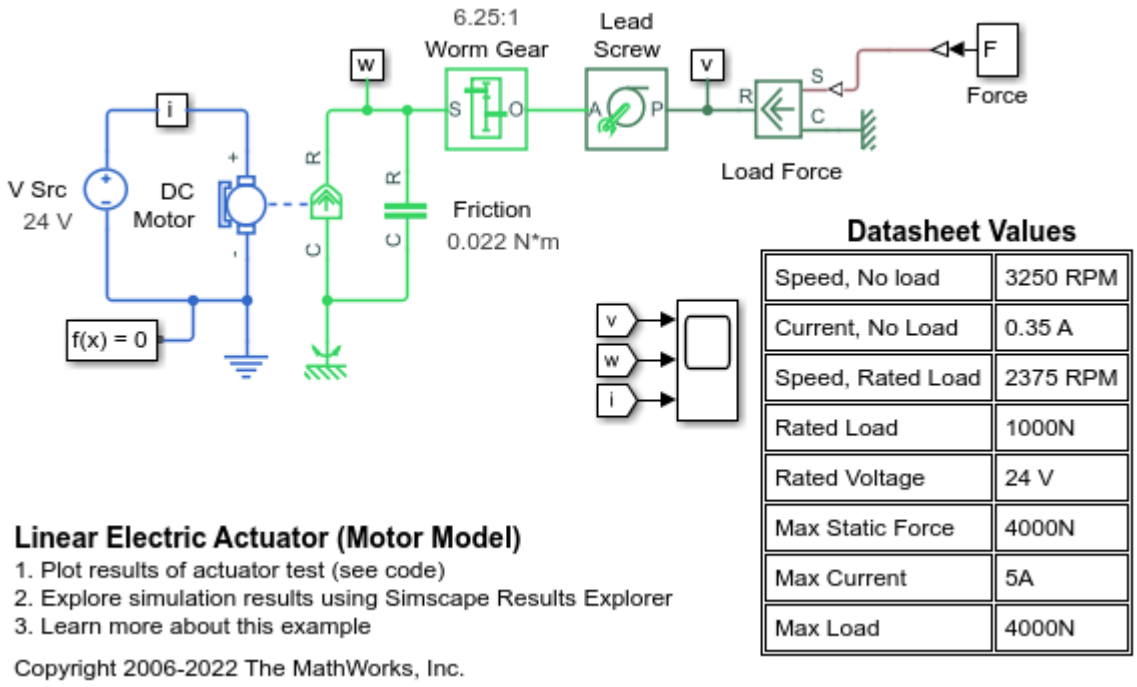
This model shows how to develop a model of an uncontrolled linear actuator using datasheet parameter values. The actuator consists of a DC motor driving a 6.25:1 worm gear which in turn drives a 3mm lead screw to produce linear motion. Manufacturer data for the actuator defines the no-load linear speed (26mm/s), rated load (1000N), rated-load linear speed (19mm/s), and maximum current (5A). The maximum static force is 4000N and the rated voltage is 24V DC.

If friction and rotor damping are neglected, the DC Motor mask parameters can be calculated as follows. A no-load speed of 26mm/s is equivalent to (26/3)*6.25*60=3250rpm. The rated motor speed is (19/3)*6.25*60=2375rpm, and the rated power is 1000N*19e-3m/s=19W. To run the model based on these parameters, set the DC Motor Model parameterization parameter to By rated power, rated speed & no-load speed and comment out the Friction block. The results validate the speeds for zero and 1000N load, but under-predict the maximum current and maximum static force.

For a more accurate approximation of the motor, friction effects must be included. The no-load and rated-load speed information depends on the unknown friction levels, so instead we parameterize the motor in terms of the maximum current which occurs when the rotor is locked (i.e. no back emf). Then the winding resistance is given by rated voltage divided by maximum current i.e. 24V/5A = 4.8 ohms. The no-load current, measured to be 0.35A, indirectly provides information about the friction. Given this value, the torque constant is given by maximum static torque divided by the net useful current i.e. (4000N*3e-3m/s/(2*pi*6.25))/(5A-0.35A) = 0.066Nm/A. To run the model based on these parameters, set the DC Motor Model Parameterization parameter to By equivalent circuit parameters and reinstate the Friction block. The torsional friction must now be determined such that the no-load speed is 26mm/s. A value of 0.022Nm is required to achieve this. The model also confirms maximum current is 5A and maximum linear force is 4000N.

Note that a limitation of this model is that the load can back drive the motor through the worm gear. A more detailed friction model would be required to prevent this.

### Model

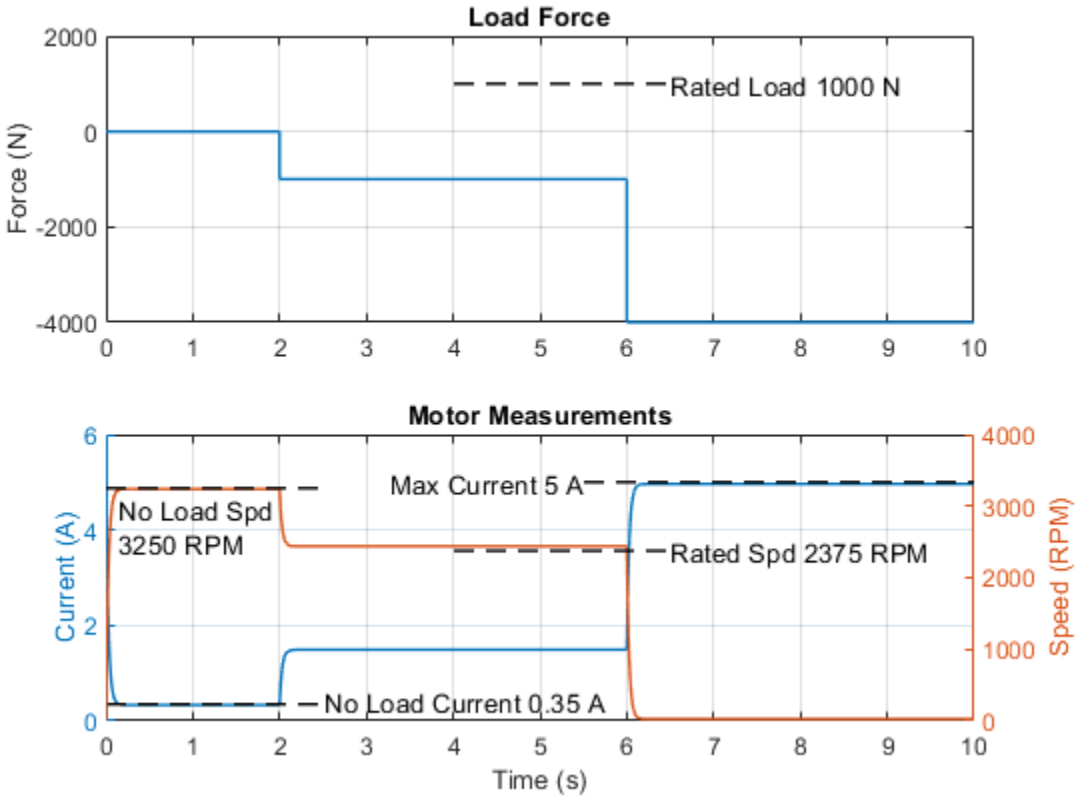



**Datasheet Values**

| | |
|---|---|
| Speed, No load | 3250 RPM |
| Current, No Load | 0.35 A |
| Speed, Rated Load | 2375 RPM |
| Rated Load | 1000N |
| Rated Voltage | 24 V |
| Max Static Force | 4000N |
| Max Current | 5A |
| Max Load | 4000N |

### Simulation Results from Simscape Logging

The plot below shows the performance of the motor compared to values found on the datasheet for a linear actuator. The test includes no load, rated load, and max load. The motor speed and currents match the specification closely, indicating that we have assigned parameter values that match the datasheet.




### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 30 microseconds.

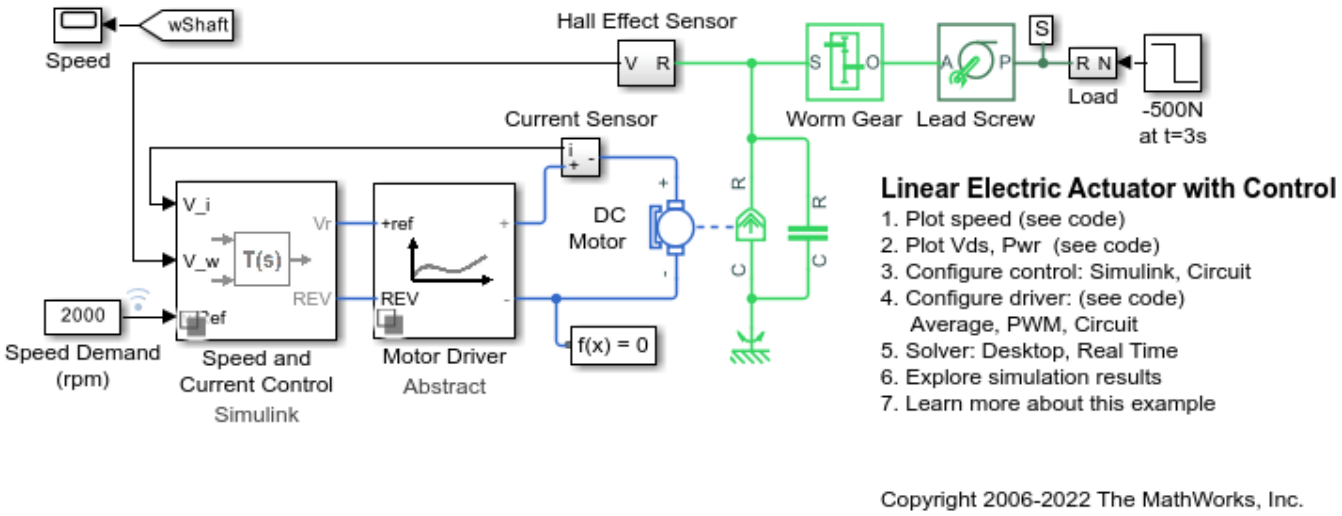
# Linear Electric Actuator with Control

This example shows a detailed implementation model of a controlled linear actuator. The actuator consists of a DC motor driving a worm gear which in turn drives a lead screw to produce linear motion. The model includes quantization effects of the Hall-effect sensor and the implementation of the control in analog electronics. There are multiple variant subsystems in this model that have models at varying levels of fidelity.

The speed control and current control models are implemented using pure Simulink® blocks. This permits us to easily specify the control algorithm and identify the requirements for an analog circuit implementation, which is shown in another variant.

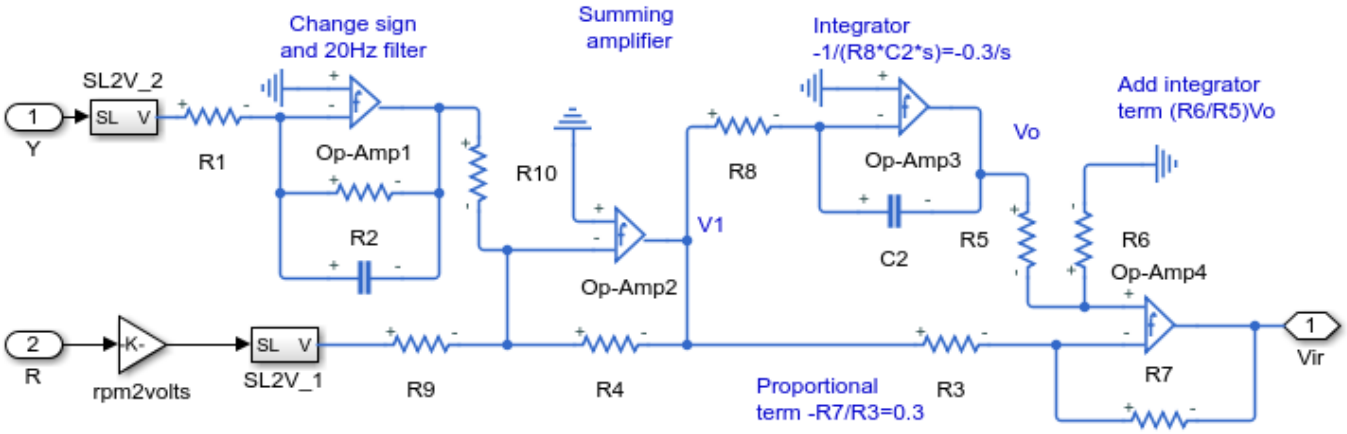
The motor drive circuit model is implemented in an abstract circuit which can simulate in two different simulation modes. In average mode, the voltage applied to the motor varies continuously between the maximum and minimum values depending upon the voltages at its input pins. In PWM mode, a PWM voltage is applied to the motor and its duty cycle depends on the voltage applied at its input pins. The motor circuit is also implemented using semiconductor devices. This enables exploration of individual switching events.

### Model



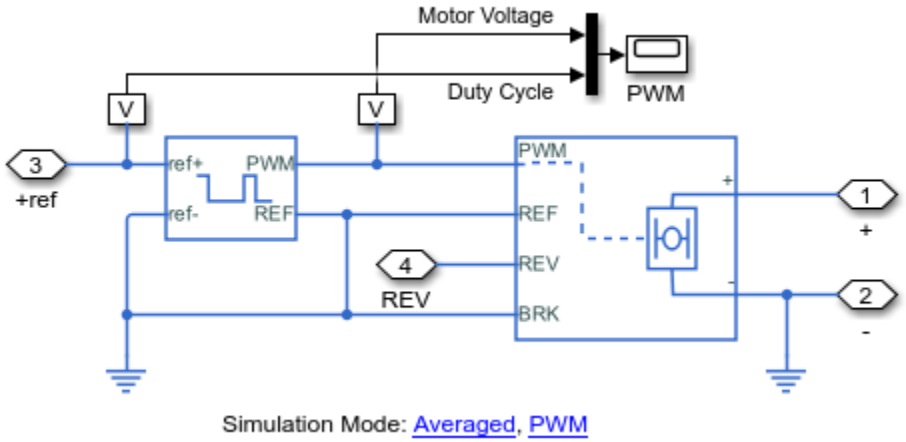

### Speed Controller Subsystem

The circuit shown below implements a PI controller with a filter using Simscape™ Electrical™ components. This is enabled when the Circuit variant is selected within the Speed and Current Control subsystem.



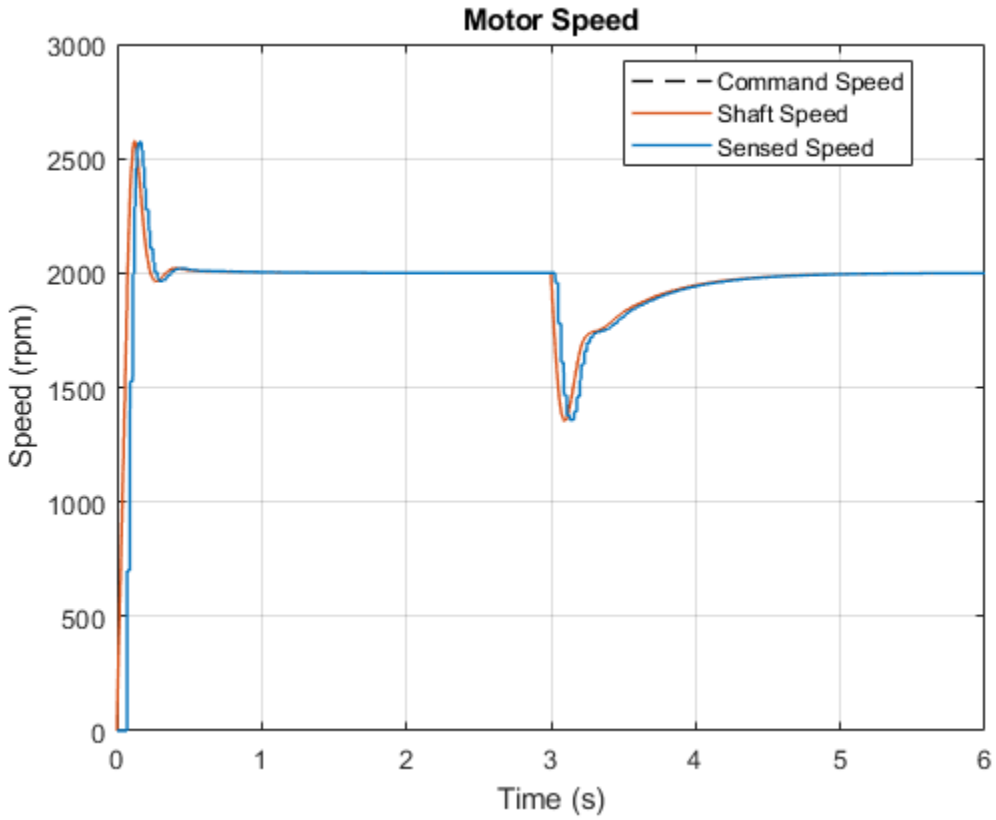

### Abstract Motor Driver Circuit Subsystem

The blocks below are an abstract model of a motor driver circuit. The Controlled PWM Voltage block generates a PWM signal based on the voltages at its +ref and -ref ports. The H-Bridge block represents an H-bridge circuit commonly used to drive electric motors. These blocks can be configured to run in average mode or PWM mode.



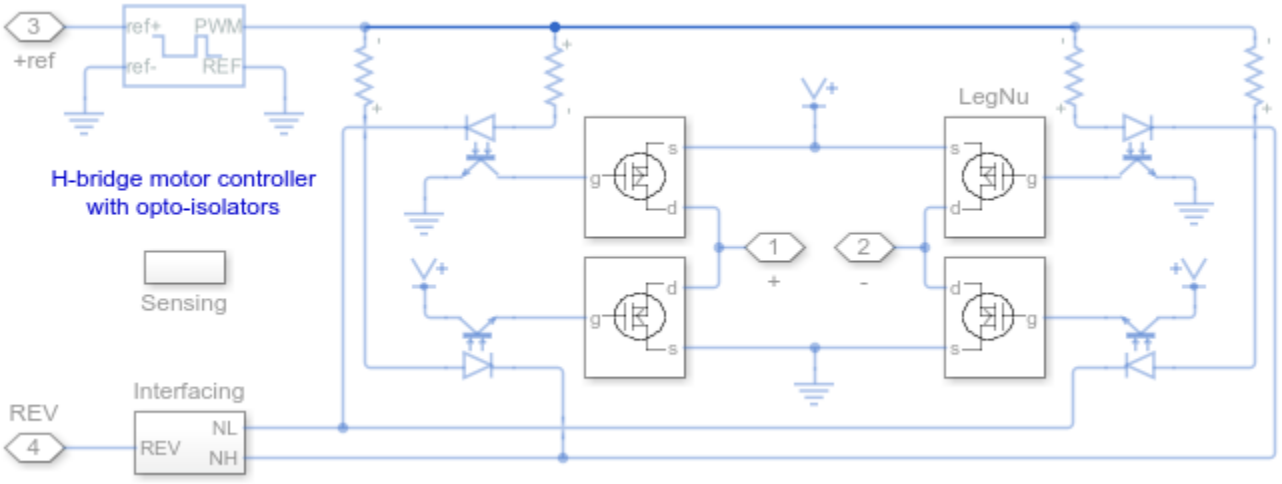

### Simulation Results from Simscape Logging - Abstract Model

The plot below shows how the motor speed tracks a reference input. The actual speed of the shaft and the speed determined by the shaft encoder are shown, indicating that the signal that the control system sees is not a perfect measurement of the shaft speed. The system response to a disturbance torque at 3 seconds is shown.



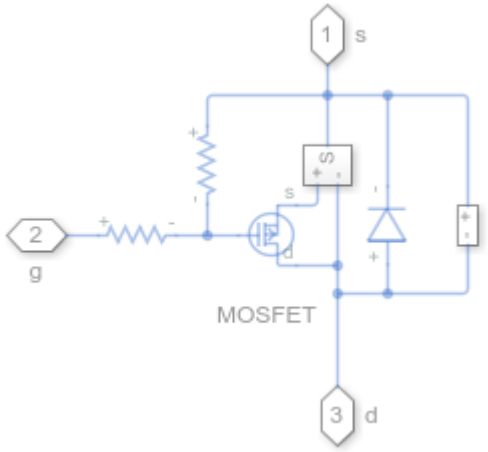

### Circuit Subsystem

The circuit shown below is an H-bridge circuit implemented using Simscape Electrical components. It is enabled when the Circuit variant of the Motor Driver variant subsystem is selected.



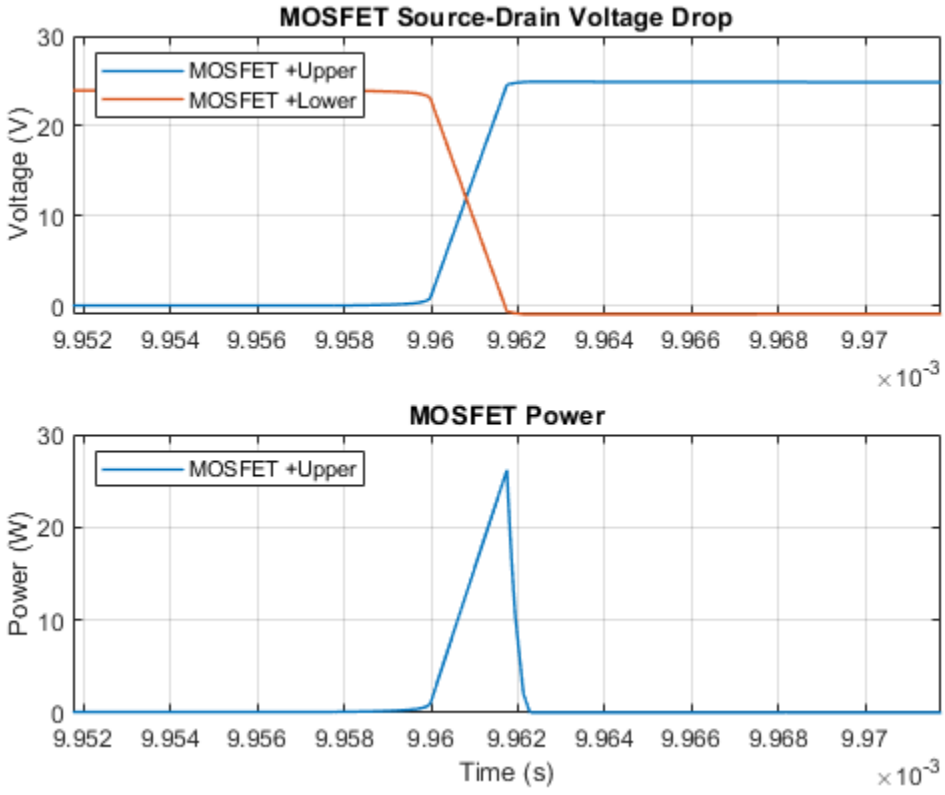
### MOSFET Subsystem

The circuit below is one of the four legs of the H-bridge. The MOSFET semiconductor and other components control the flow of current through the leg of the H-bridge.




### Simulation Results from Simscape Logging - Detailed Model

The plot below shows a single switching event during a simulation using the detailed model of the H-bridge. The spike in MOSFET power is due to the fact that for a brief period of time, both MOSFETs on the same side of the motor are conducting, creating a short circuit.


MOSFET Source-Drain Voltage Drop
MOSFET +Upper
MOSFET +Lower
Voltage (V)
30
20
10
0
9.952
9.954
9.956
9.958
9.96
9.962
9.964
9.966
9.968
9.97
×10-3
MOSFET Power
MOSFET +Upper
Power (W)
30
20
10
0
9.952
9.954
9.956
9.958
9.96
9.962
9.964
9.966
9.968
9.97
Time (s)
×10-3

# Brushless DC Motor

This example shows how a system-level model of a brushless DC motor (i.e. a servomotor) can be constructed and parameterized based on datasheet information. The motor and driver are modeled as a single masked subsystem. If viewing the model in Simulink®, select the Motor and driver block, and type Ctrl+U to look under the mask and see the model structure.

This model of a brushless DC motor uses a standard configuration. An inner feedback loop controls current and an outer feedback loop controls motor speed. Speed demand is set by the voltage presented at the Vref pin, and motor direction by the voltage presented at the Vdir pin. If the voltage at the Vbrk pin goes high, then Vref is overridden and speed demand set to zero to implement a braking action.

In this model, the speed demand is set to 2V which corresponds to 40,000rpm. After one second, the Vdir pin is set high, and the motor reverses to -40,000rpm. At 2 seconds, the braking pin Vbrk is set high, and the motor slows to zero rpm. The efficiency of the brushless DC motor is calculated as the ratio of mechanical power out to electrical power in. Hence it can be transiently outside of the 0 to 100 percent range due to rotor inertia.

The manufacturer datasheet for the brushless dc motor gives the stall torque as 0.44mNm, the maximum permissible speed as 100,000rpm, mechanical time constant as 5ms, rotor inertia as 0.005gcm^2, efficiency as 41 percent at 0.23mNm and 40,000rpm, no-load current as 22mA and nominal voltage as 12V. The time constant is based on using the manufacturer motor driver which uses block commutation. The motor driver can be configured so that when the speed reference voltage is as at its maximum of +5V, the commanded speed is 100,000rpm.

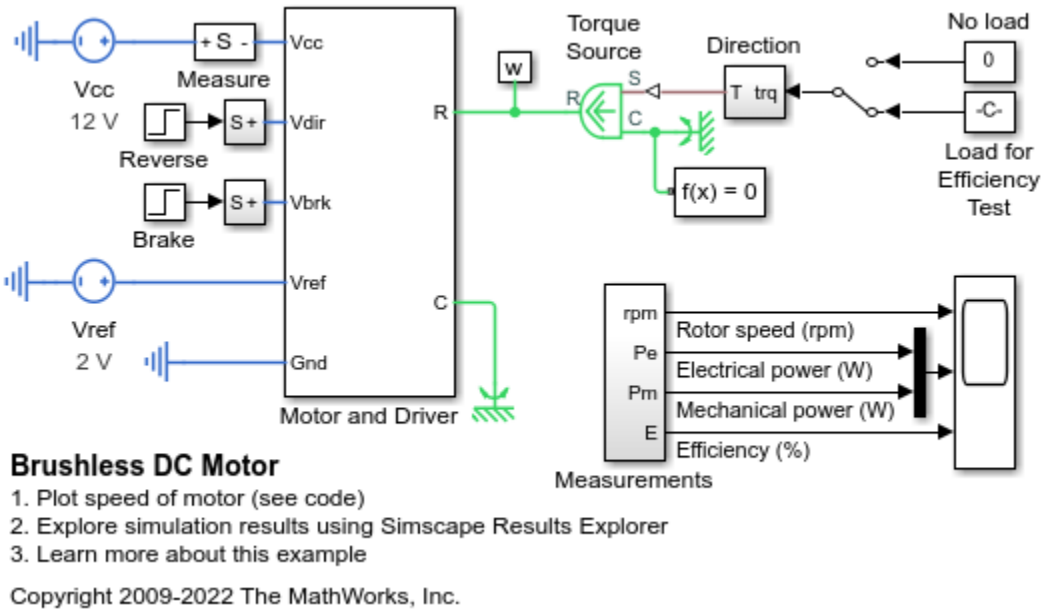
The Servomotor block in the Motor and driver subsystem is used to model the inner current feedback loop, plus balance mechanical and electrical powers. For system design, it is not usually necessary to model the current switching controlled by the motor driver, whereas ensuring the correct torque-speed characteristics and current drawn from the DC supply is. The vector of maximum torque values is in practice determined by the maximum driver current. Motor drivers normally have a maximum current setting which should be matched to the maximum rated motor torque, or the maximum torque that is to be applied to the load if the motor is over-specified. Here the vector of maximum torque values is set to be the motor stall torque up to and beyond the maximum speed of 100,000rpm. The assumption is that system in which the motor and driver are used will take care of ensuring that the motor does not overheat by operating too long at high torque and speed combinations.

Motor electrical losses are assumed to consist of two terms. The first is a fixed loss that is independent of load, and this is calculated as Vcc*I0 where Vcc is the nominal supply voltage and I0 is the no-load DC current drawn from the driver power supply. Note that if block commutation is used, as for the driver this example is based on, I0 will be twice the current in an energized phase winding. The second loss term is proportional to the square of instantaneous motor winding current. This can be approximated as a term that is proportional to the square of the average torque. The two loss terms are implemented by the Servomotor block.

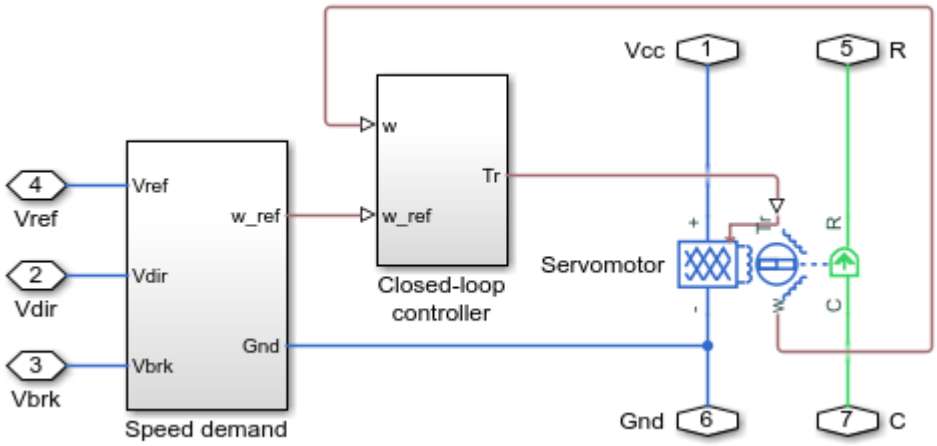
There are three Motor and driver mask parameters which have to be tuned to match datasheet values. These are the proportional and integral gains for the speed feedback controller, and the time constant for the inner-loop current controller. Here, the datasheet gives the no-load time constant as 5ms. A typical rule of thumb is that an inner control loop should be at least ten times faster than the outer loop. This means a time constant of 0.5ms for the current controller. With this value set, the proportional term is then increased until the speed time constant is approximately 5ms. The integral gain should then be set when performing a speed step under load, and increased until steady-state

error is removed in the order of 5ms. Some fine tuning of the two gains is then needed to recover the 5ms rise time under no load.

### Model

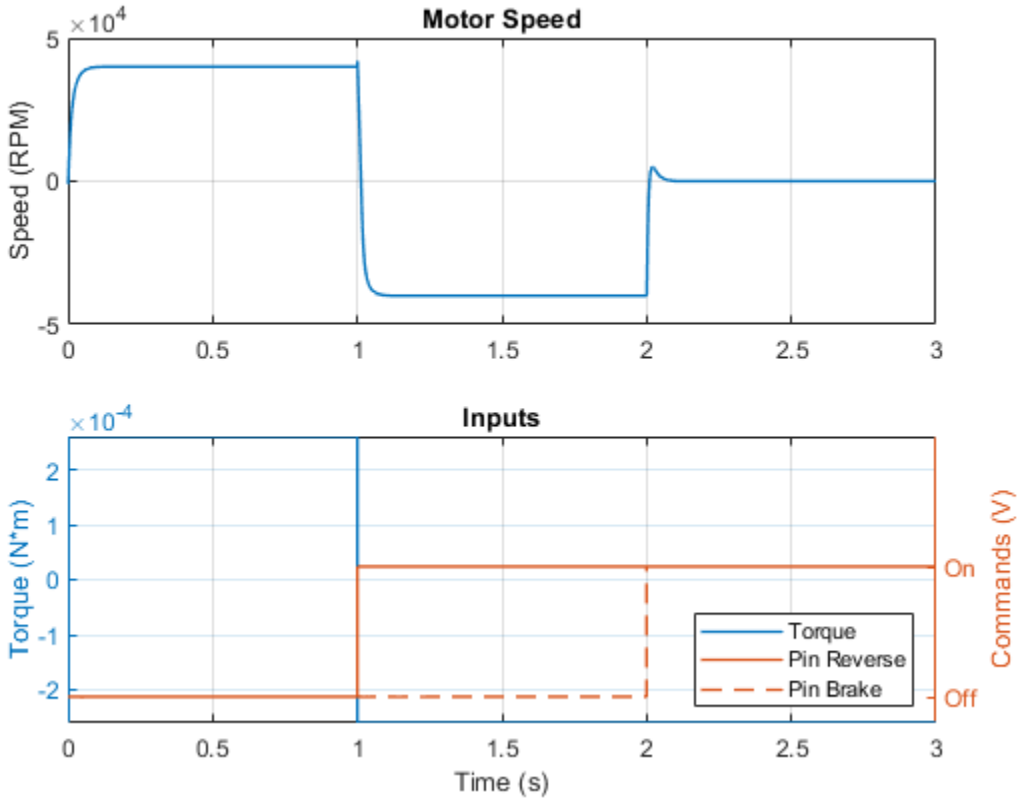



### Motor and Driver Subsystem




### Simulation Results from Simscape Logging

The plot below shows the speed of the brushless DC motor under varying conditions. The load torque is a constant value always opposed to the rotation of the shaft. Commands to reverse direction and brake are applied.


Motor Speed
×10^4
5
0
-5
Speed (RPM)
0
0.5
1
1.5
2
2.5
3
Inputs
×10^-4
2
1
0
-1
-2
Torque (N*m)
On
Off
Commands (V)
Torque
Pin Reverse
Pin Brake
Time (s)

# Import IPMSM Flux Linkage Data from ANSYS Maxwell

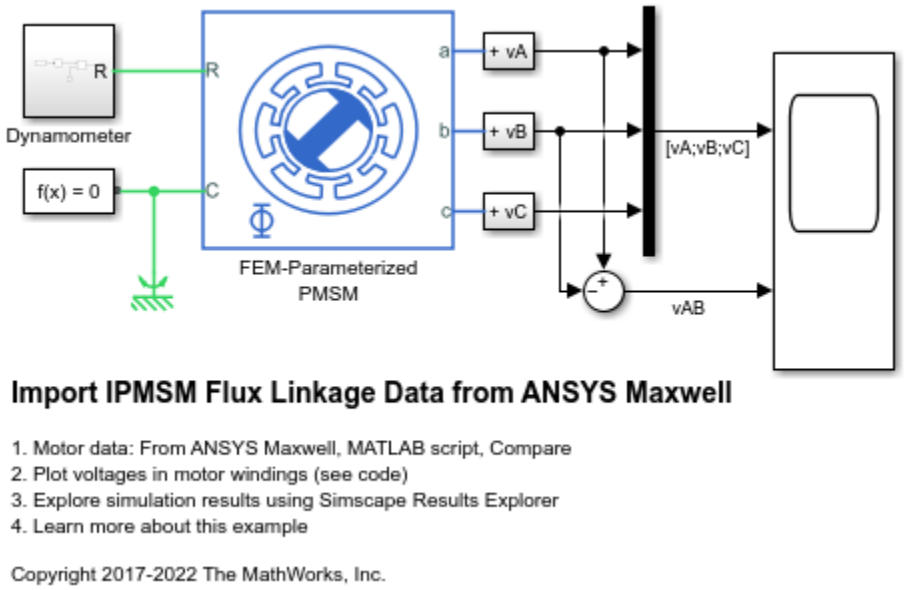
This example shows how you can import a motor design from ANSYS® Maxwell® into a Simscape™ simulation.

### Flux Linkage Data

The data file defining the motor flux linkage is ee_ece_data.txt, and is reproduced with permission of ANSYS, Inc. To view the data, open the model from MATLAB® and click on the "From ANSYS Maxwell" link.

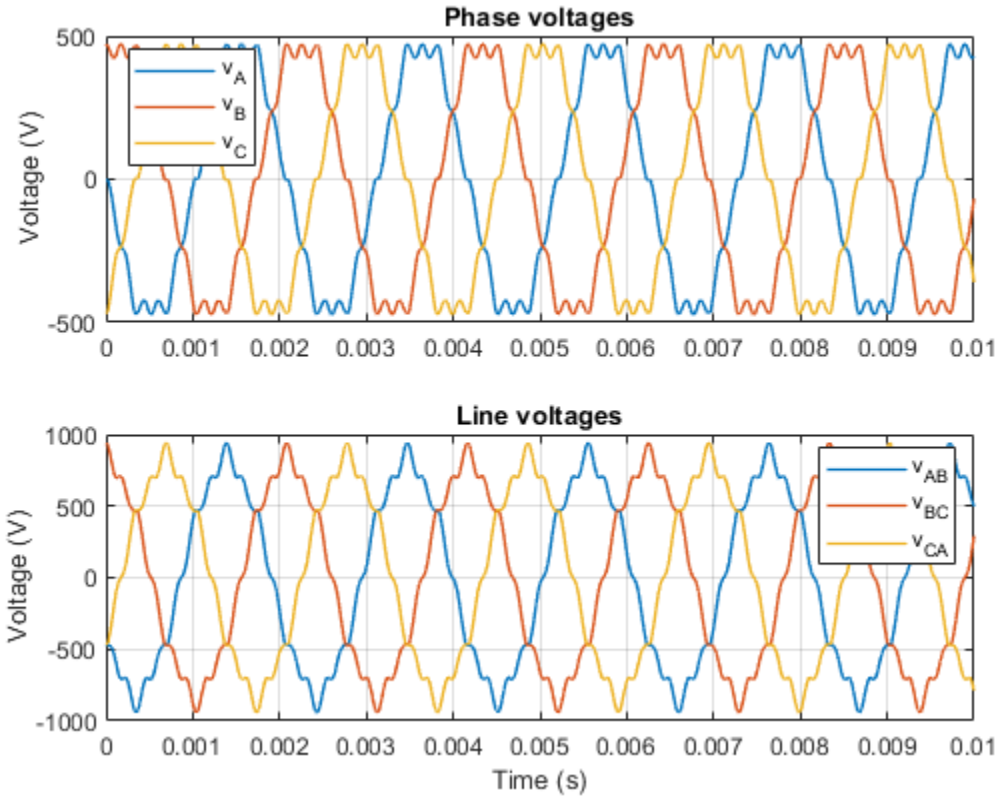
To read this data into MATLAB, the text file requires some minor editing to make it executable from the MATLAB command line. The resultant script is ee_ece_data.m and can be viewed by clicking on the "MATLAB script" link from the model. To see a summary of the edits, click on the "Compare" link.

### Model




### Simulation Results from Simscape Logging

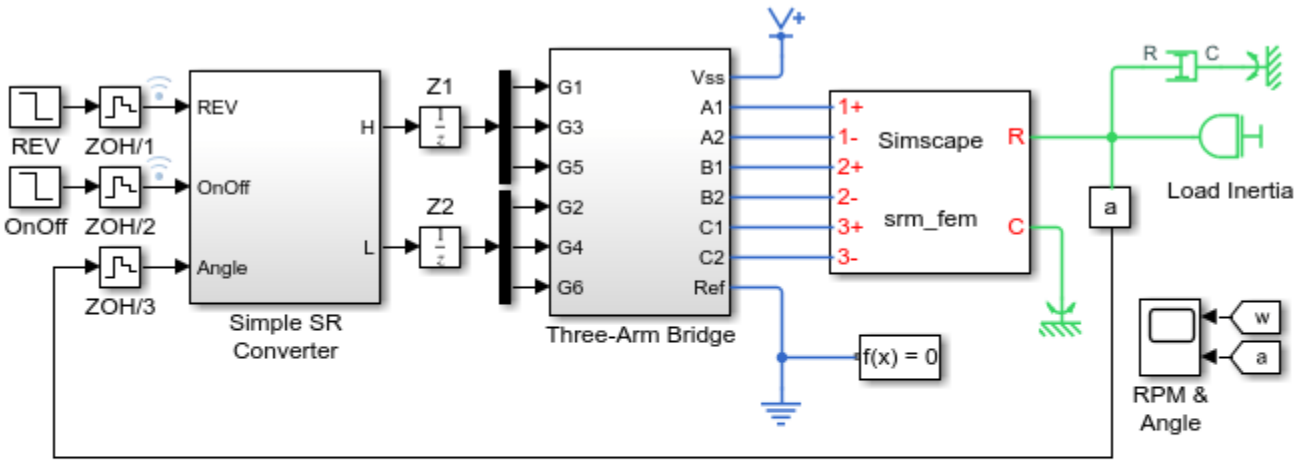
The plot below shows the simulated open-circuit voltages. Notice the nonlinear back EMF profile that would not be captured if using a simplified PMSM model with fixed back EMF constant.


Phase voltages
Voltage (V)
500
0
-500
0
0.001
0.002
0.003
0.004
0.005
0.006
0.007
0.008
0.009
0.01
$v_A$
$v_B$
$v_C$
Line voltages
Voltage (V)
1000
500
0
-500
-1000
Time (s)
$v_{AB}$
$v_{BC}$
$v_{CA}$

# Switched Reluctance Motor Parameterized with FEM Data

This example shows how to build a model of a Switched Reluctance Motor (SRM). A custom Simscape™ composite component is used to construct the three stator pole pairs using the Simscape Electrical™ FEM-Parameterized Rotary Actuator as a base component. The states of the input pins (reverse, on/off) control the torque applied to the motor shaft.

**Model**

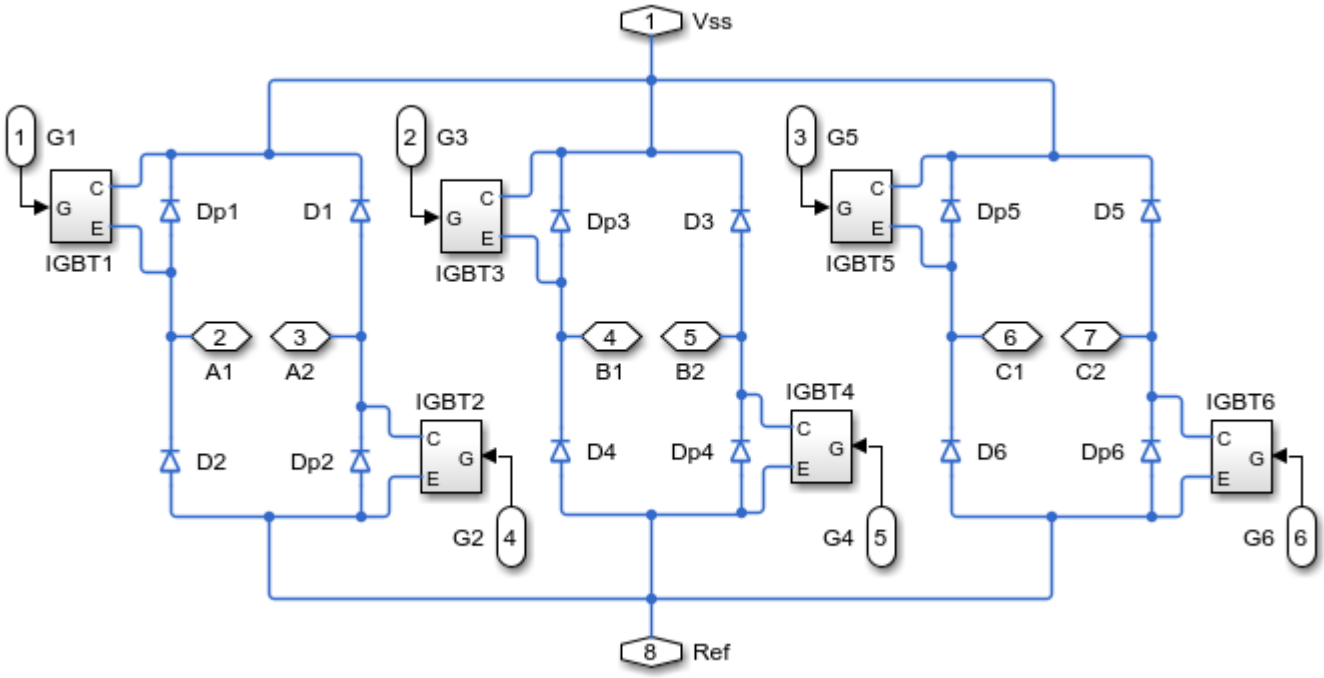



**Switched Reluctance Motor Parameterized with FEM Data**

1. Plot speed of motor (see code)
2. Plot torque applied by each winding (see code)
3. Plot motor data used in parameterization (see code)
4. Open custom motor library created using Simscape language
5. Explore simulation results using Simscape Results Explorer
6. Learn more about this example

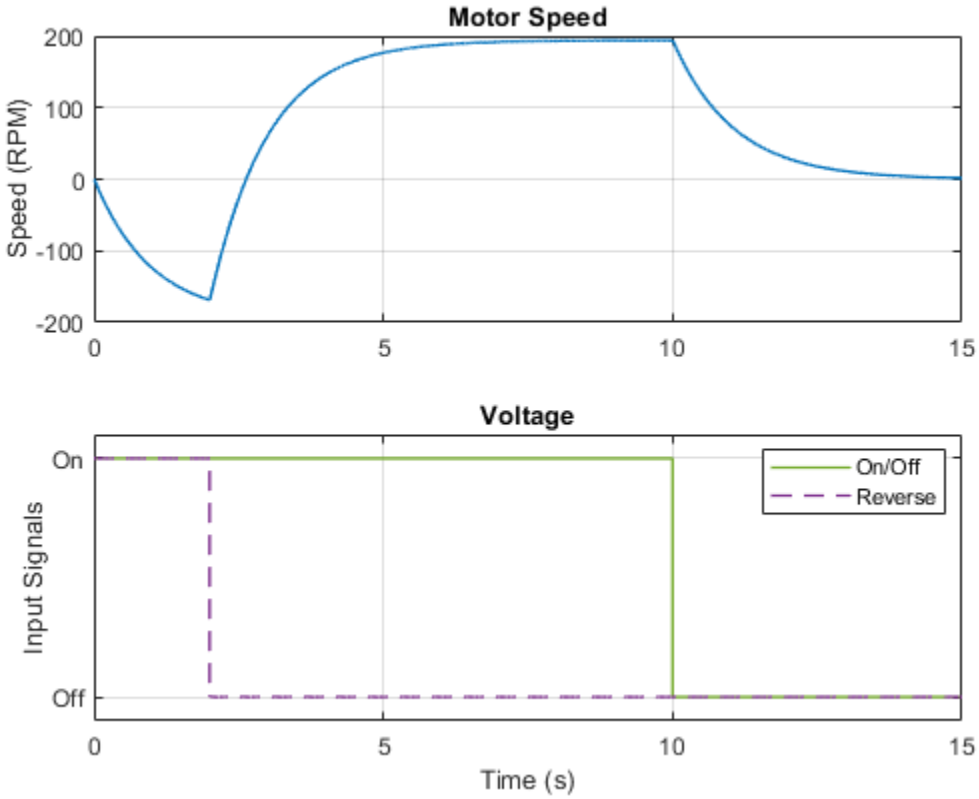
Copyright 2013-2022 The MathWorks, Inc.

### Three-Arm Bridge Subsystem

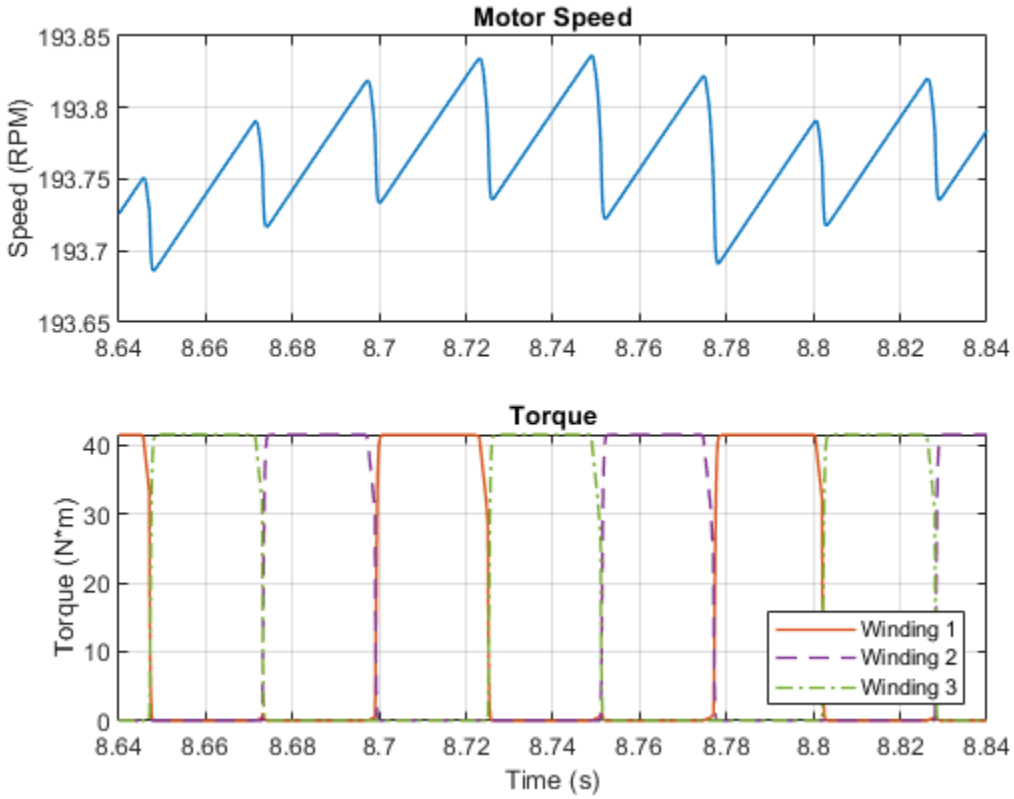



### Simulation Results from Simscape Logging

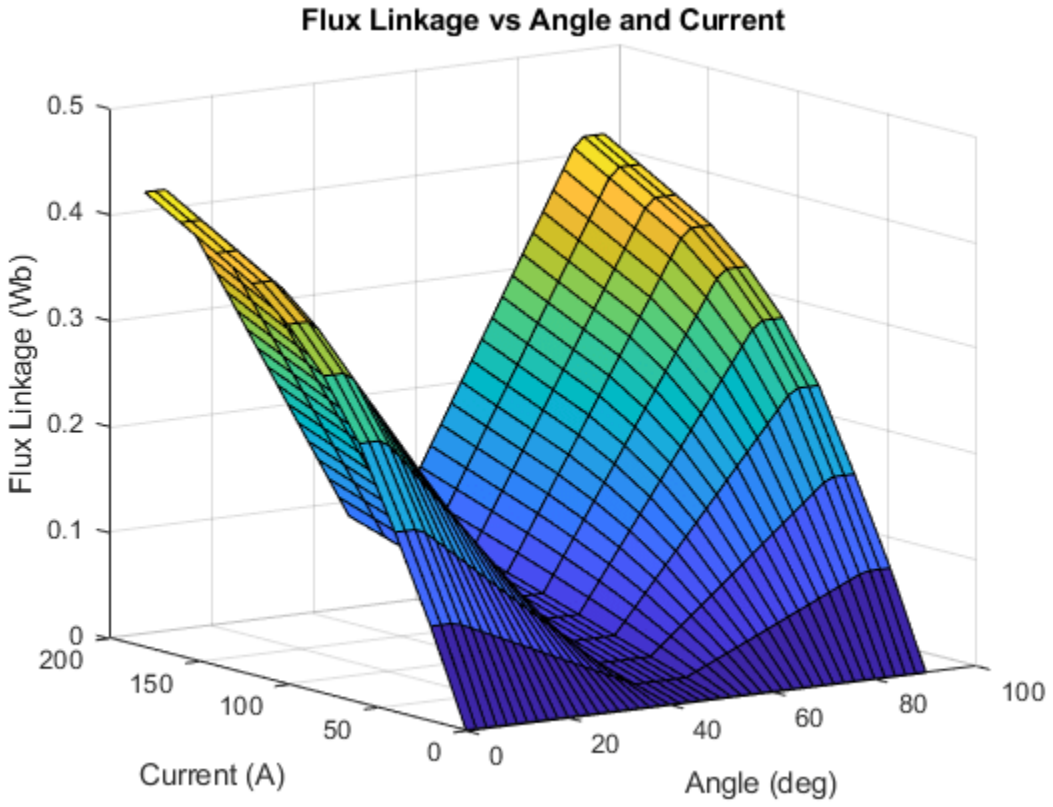
The plot below shows the behavior of the motor as control signals at the input pins of the converter are varied. The sign of the torque the motor applies to its shaft is controlled by the reverse signal. If the On/Off signal is set to 0, the motor applies no torque to the motor shaft. In this test, after the On/Off signal is set to zero, the motor shaft slows down due to viscous friction applied to the shaft.

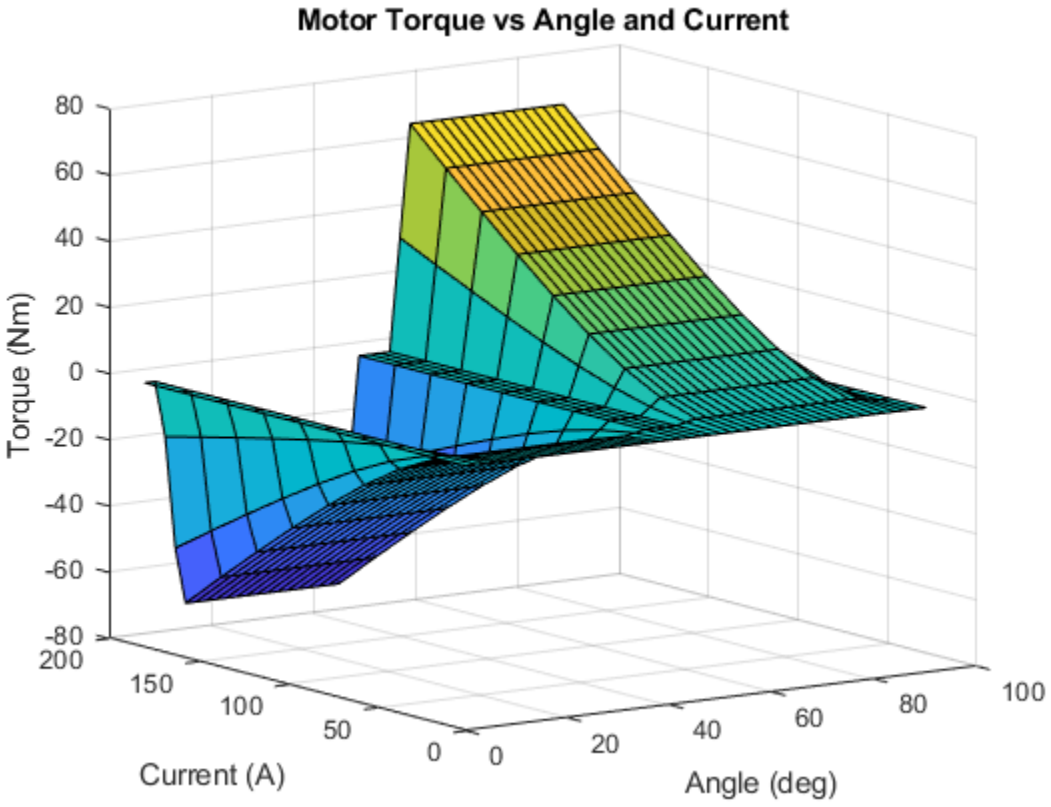



The plot below shows the behavior of the motor over a very short period of time so that the effect of the torque applied by each winding can be observed. A very small ripple in motor speed can be observed as the individual windings are energized and de-energized.




The plots below show the data used to parameterize the motor. Lookup tables for both flux linkage and motor torque dependent upon angle and current are plotted.

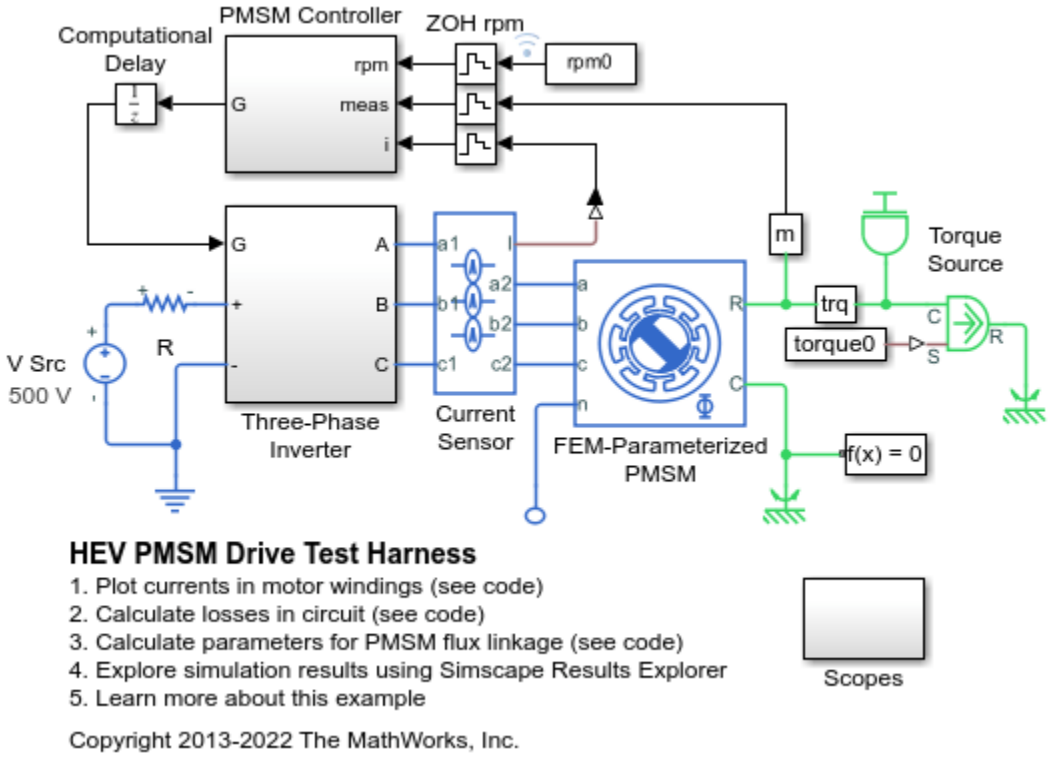

Flux Linkage vs Angle and Current
Flux Linkage (Wb)
0.5
0.4
0.3
0.2
0.1
0
200
150
100
50
0
Current (A)
0
20
40
60
80
100
Angle (deg)


Motor Torque vs Angle and Current
Torque (Nm)
80
60
40
20
0
-20
-40
-60
-80
200
150
100
50
0
Current (A)
0
20
40
60
80
100
Angle (deg)

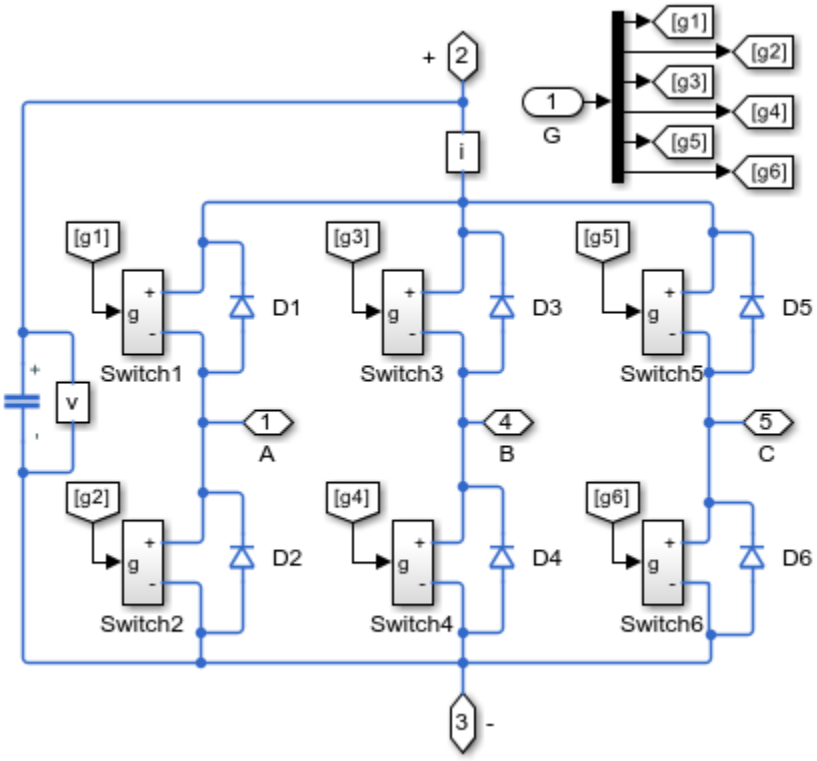
# HEV PMSM Drive Test Harness

This example shows a test harness for a Permanent Magnet Synchronous Motor (PMSM) drive sized for use in a typical hybrid vehicle. The test harness can be used to determine overall drive losses when operating at a given speed and torque. Tabulated losses information from this test harness can then be used by the Simscape™ Electrical™ Motor & Drive (System Level) block for rapid simulation of complete drive cycles whilst still accurately predicting overall system efficiency.

**Model**

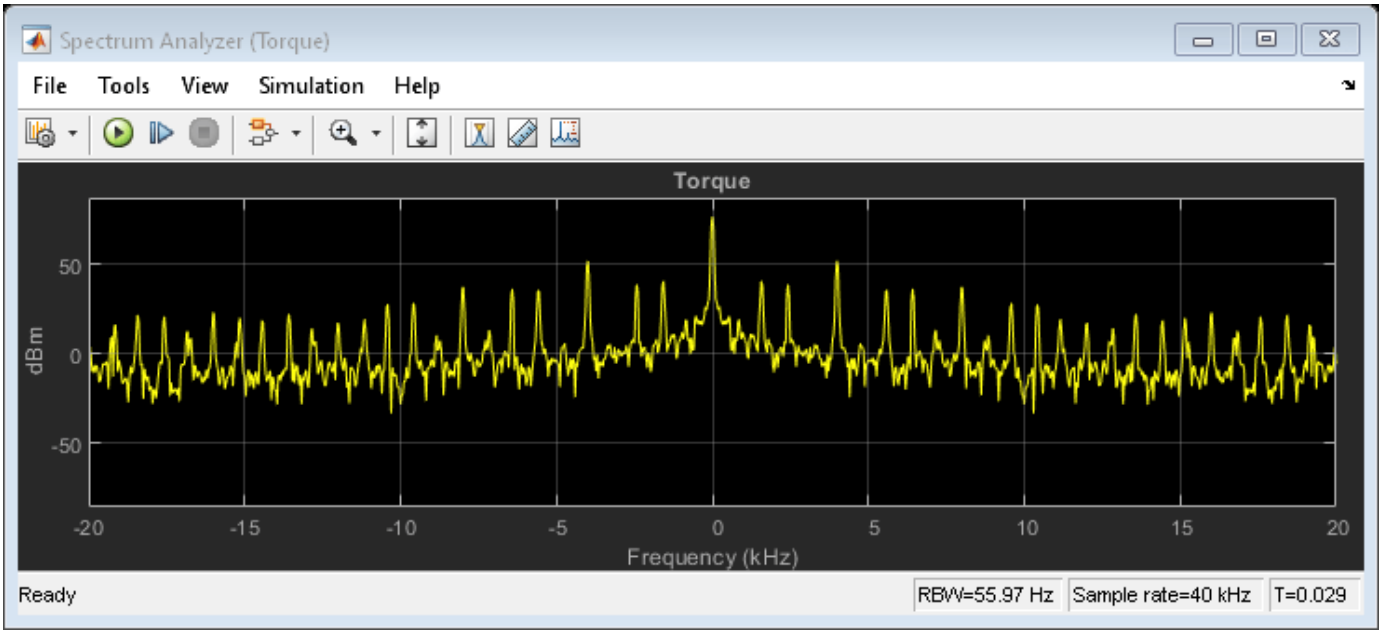


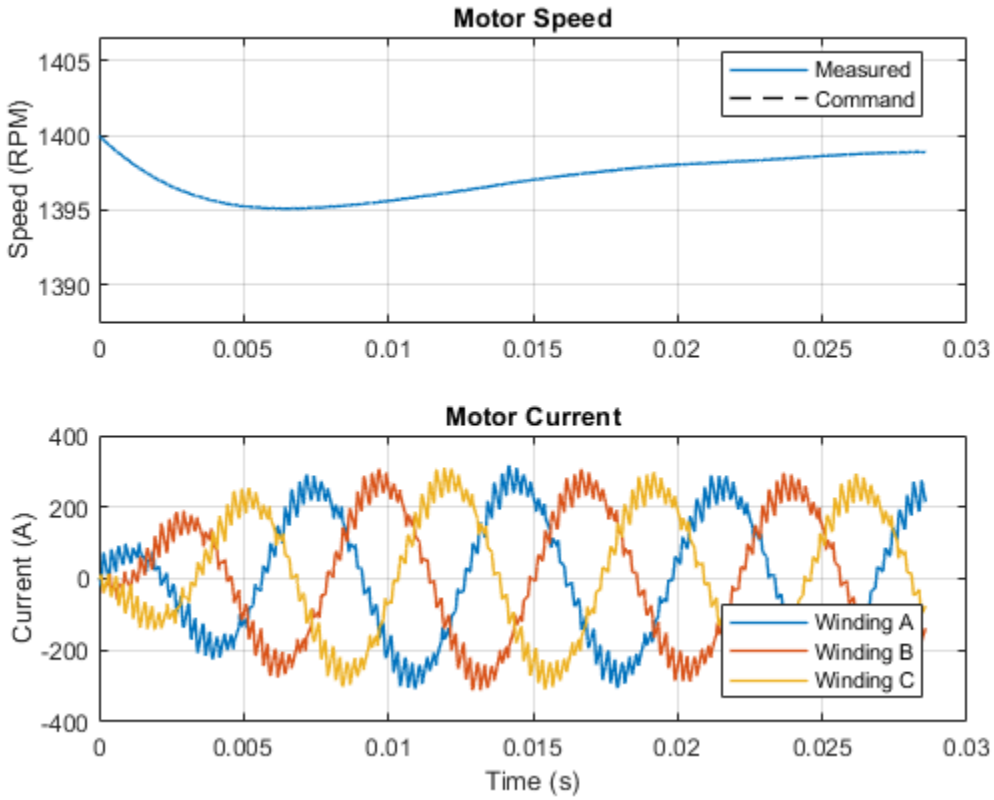
### Three-Phase Inverter Subsystem




### Simulation Results from Scopes and Simscape Logging

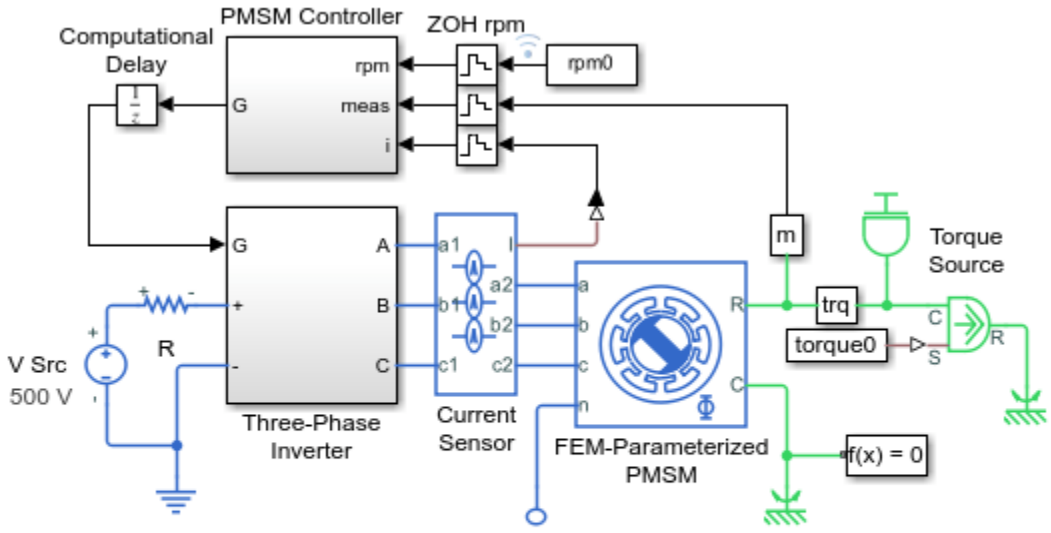
The first plot below shows the result from Spectrum Analyzer. Second plot shows measured and commanded rotor speed. The winding currents are also plotted, which explains the high frequency ripple in the motor speed.




The table below shows the power dissipated by individual components in the ee_pmsm_drive model. These totals were calculated from simulation results using logged Simscape variables and the losses calculation utility ee_getPowerLossSummary. The total output power in the load, as well as the losses are shown.

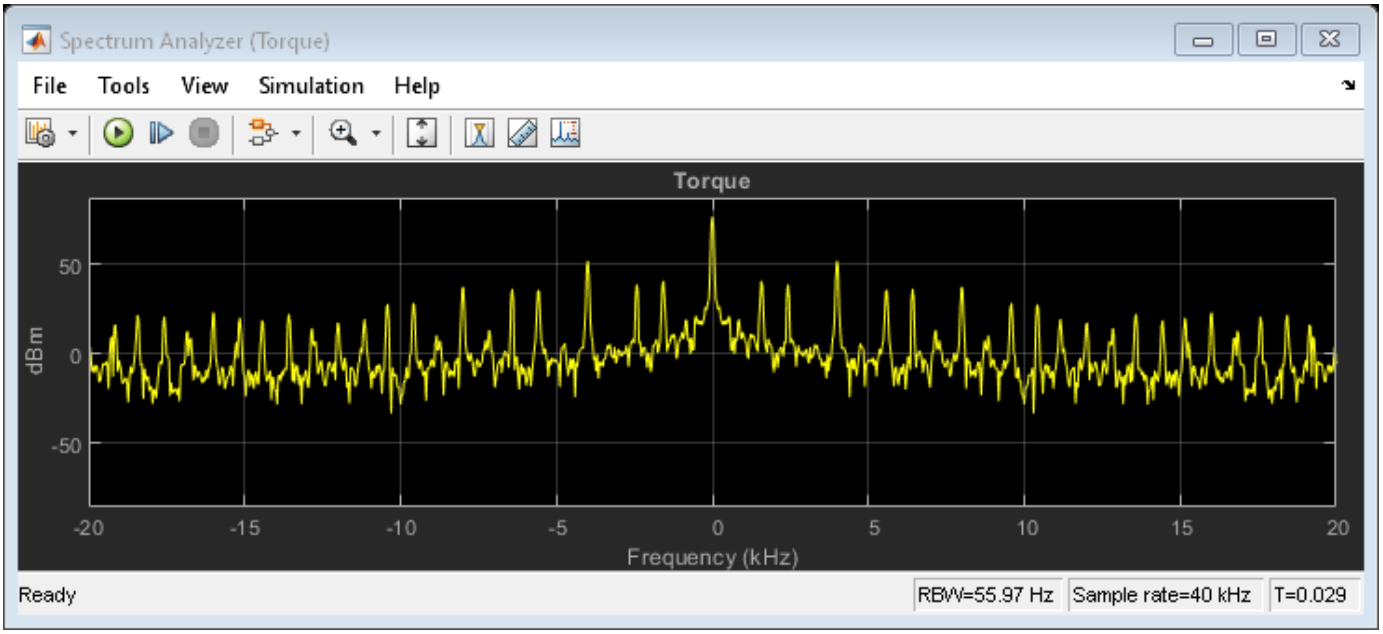
```
Running the PMSM Drive Test Harness model to generate simulation data
Efficiency = 85.9702% when speed = 1400rpm and torque = 200Nm
Losses in watts by component are as follows:
                                LoggingNode                                  Power
    ____________________________________________________________________    _____

    {'ee_motor_pmsm_drive.FEM_Parameterized_PMSM'                       }    3610.3
    {'ee_motor_pmsm_drive.Three_Phase_Inverter.Switch2.IGBT'            }     158.6
    {'ee_motor_pmsm_drive.Three_Phase_Inverter.Switch3.IGBT'            }     157.2
    {'ee_motor_pmsm_drive.Three_Phase_Inverter.Switch5.IGBT'            }     155.3
    {'ee_motor_pmsm_drive.Three_Phase_Inverter.Switch6.IGBT'            }     153.4
    {'ee_motor_pmsm_drive.Three_Phase_Inverter.Switch4.IGBT'            }     151.4
    {'ee_motor_pmsm_drive.Three_Phase_Inverter.Switch1.IGBT'            }     143.7
    {'ee_motor_pmsm_drive.Three_Phase_Inverter.D2'                      }      23.5
    {'ee_motor_pmsm_drive.Three_Phase_Inverter.D1'                      }      21.6
    {'ee_motor_pmsm_drive.Three_Phase_Inverter.D4'                      }      21.4
    {'ee_motor_pmsm_drive.Three_Phase_Inverter.D6'                      }      20.7
    {'ee_motor_pmsm_drive.Three_Phase_Inverter.D5'                      }      20.4
    {'ee_motor_pmsm_drive.Three_Phase_Inverter.D3'                      }      19.5
    {'ee_motor_pmsm_drive.Three_Phase_Inverter.C0'                      }      11.2
    {'ee_motor_pmsm_drive.R'                                            }       4.4
    {'ee_motor_pmsm_drive.Three_Phase_Inverter.Switch3.Gate_Driver'}            0.8
    {'ee_motor_pmsm_drive.Three_Phase_Inverter.Switch4.Gate_Driver'}            0.8
    {'ee_motor_pmsm_drive.Three_Phase_Inverter.Switch1.Gate_Driver'}            0.8
    {'ee_motor_pmsm_drive.Three_Phase_Inverter.Switch6.Gate_Driver'}            0.7
    {'ee_motor_pmsm_drive.Three_Phase_Inverter.Switch2.Gate_Driver'}            0.6
    {'ee_motor_pmsm_drive.Three_Phase_Inverter.Switch5.Gate_Driver'}            0.6
```




**HEV PMSM Drive Test Harness**

1. Plot currents in motor windings (see code)
2. Calculate losses in circuit (see code)
3. Calculate parameters for PMSM flux linkage (see code)
4. Explore simulation results using Simscape Results Explorer
5. Learn more about this example

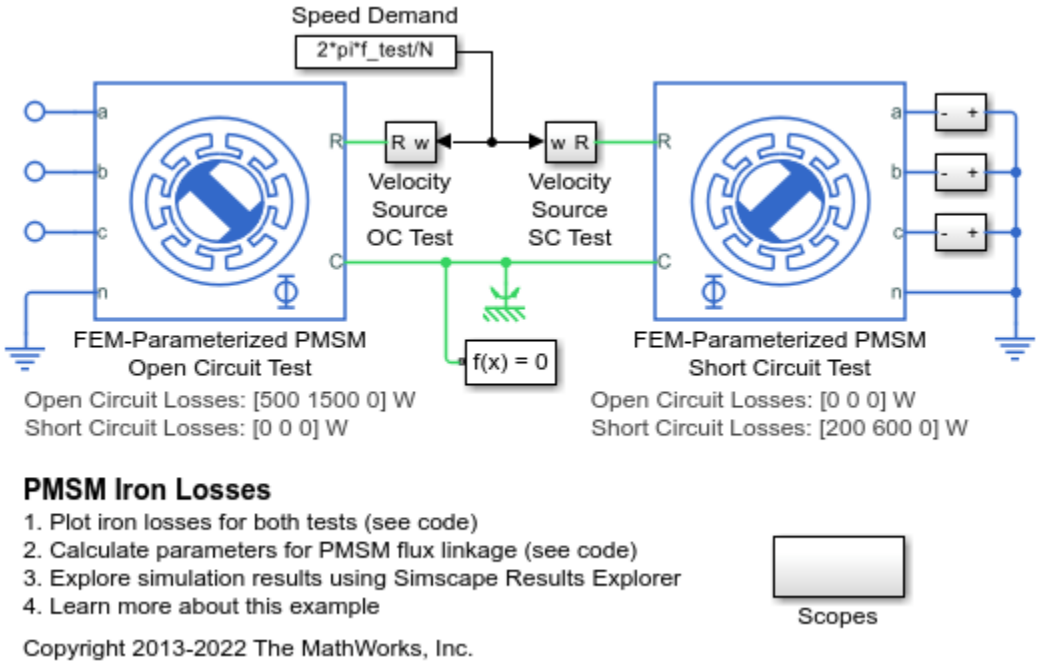
Copyright 2013-2022 The MathWorks, Inc.


Spectrum Analyzer (Torque)
File
Tools
View
Simulation
Help
Torque
50
0
-50
dBm
-20
-15
-10
-5
0
5
10
15
20
Frequency (kHz)
Ready
RBW=55.97 Hz
Sample rate=40 kHz
T=0.029

# PMSM Iron Losses

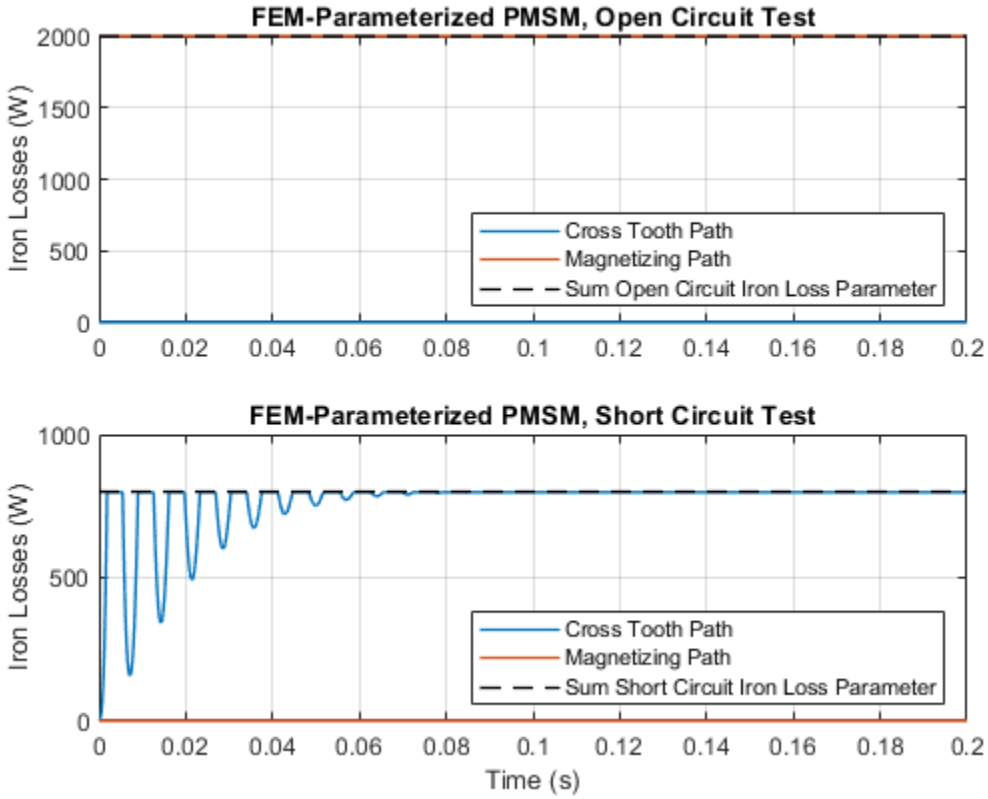
This example shows a test harness for a Permanent Magnet Synchronous Motor (PMSM) that validates that the iron losses are as expected. The open-circuit test validates the main flux path losses which are defined to be 500W due to hysteresis and 1500W due to eddy current losses. The short-circuit test validates the cross-tooth flux path losses which are defined to be 200W due to hysteresis and 600W due to eddy current losses.

### Model




### Simulation Results from Simscape Logging

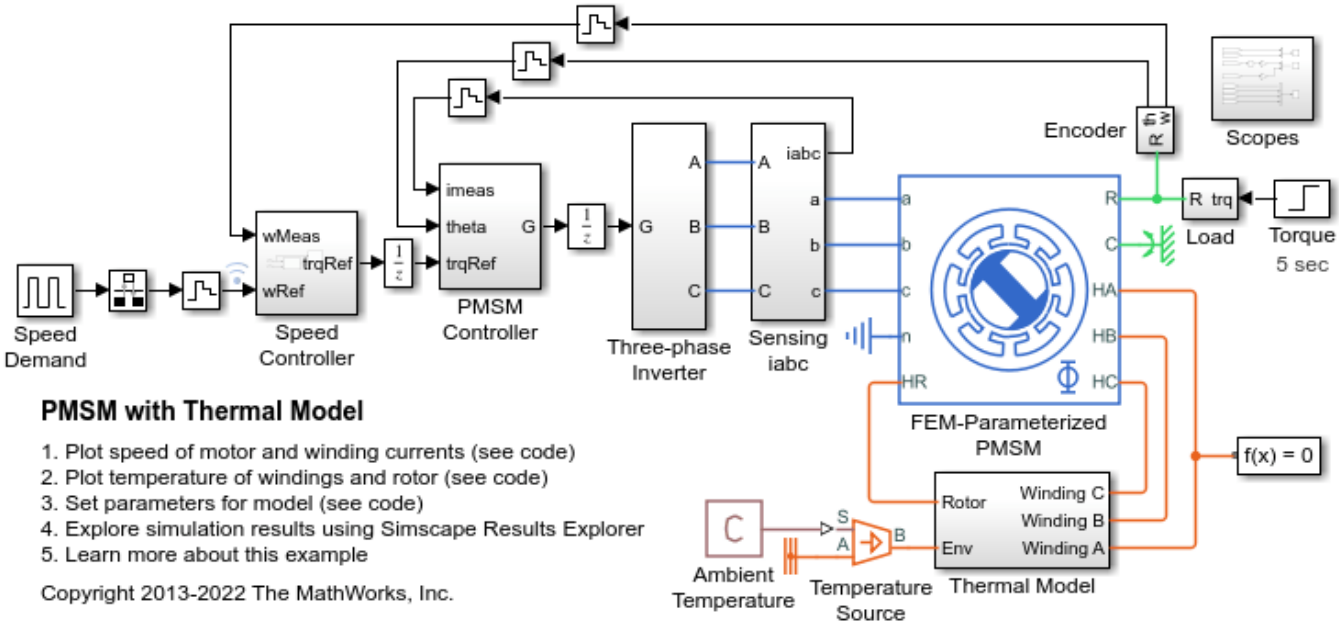
The plot below validates that the iron loss parameters entered into the FEM-Parameterized PMSM block provided the expected behavior in simulation. The sum of the parameter values relevant for the respective test match the transient simulation results.


FEM-Parameterized PMSM, Open Circuit Test
Iron Losses (W)
2000
1500
1000
500
0
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
0.18
0.2
Cross Tooth Path
Magnetizing Path
Sum Open Circuit Iron Loss Parameter
FEM-Parameterized PMSM, Short Circuit Test
Iron Losses (W)
1000
500
0
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
0.18
0.2
Time (s)
Cross Tooth Path
Magnetizing Path
Sum Short Circuit Iron Loss Parameter

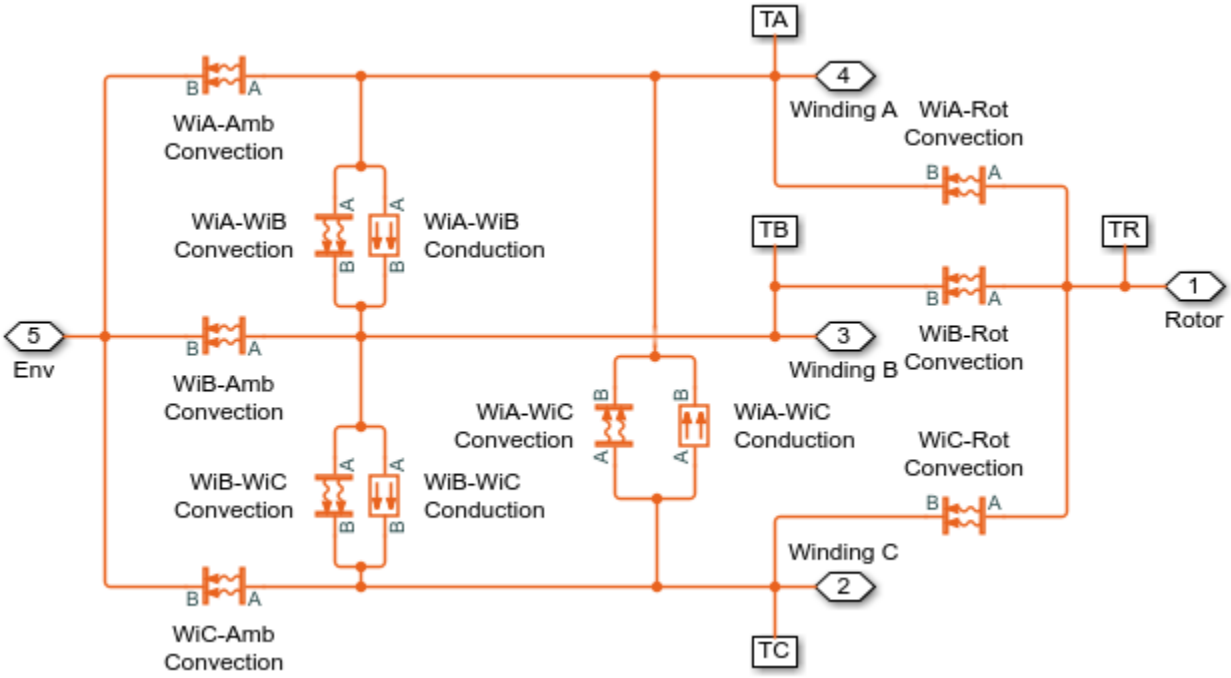
# PMSM with Thermal Model

This example shows a nonlinear model of a PMSM with thermal dependency. The PMSM behavior is defined by tabulated nonlinear flux linkage data. Motor losses are turned into heat in the stator winding and rotor thermal ports.

### Model

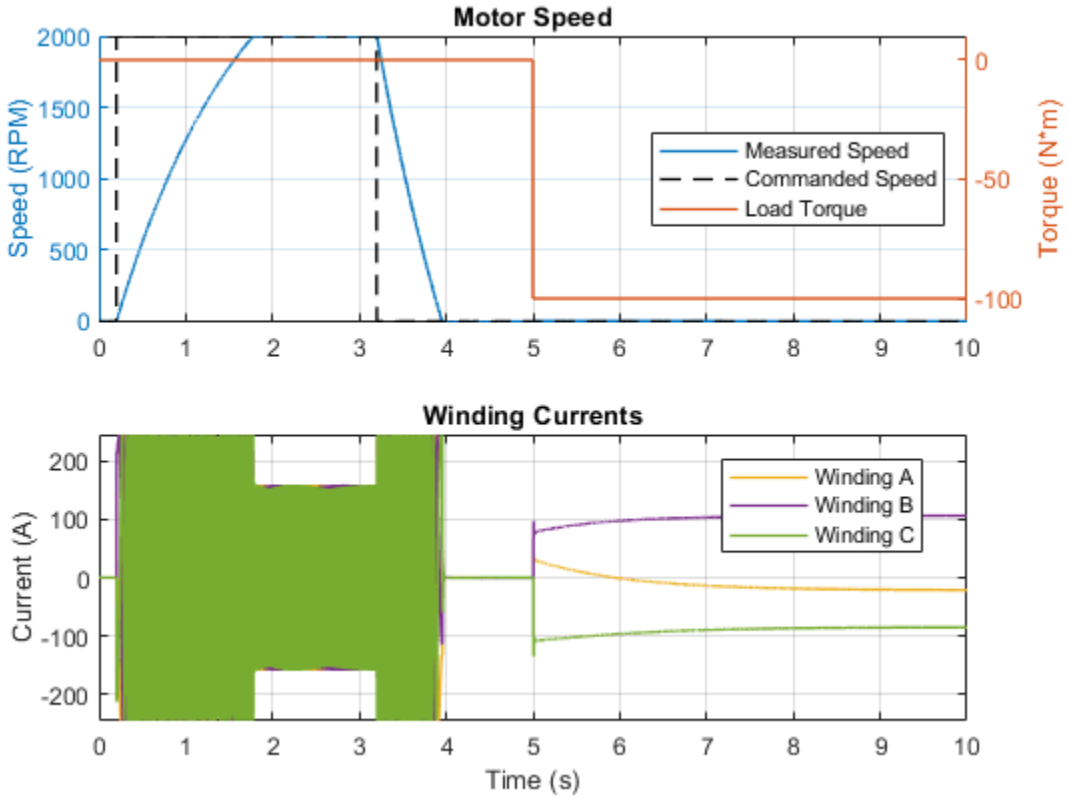


### Thermal Model Subsystem

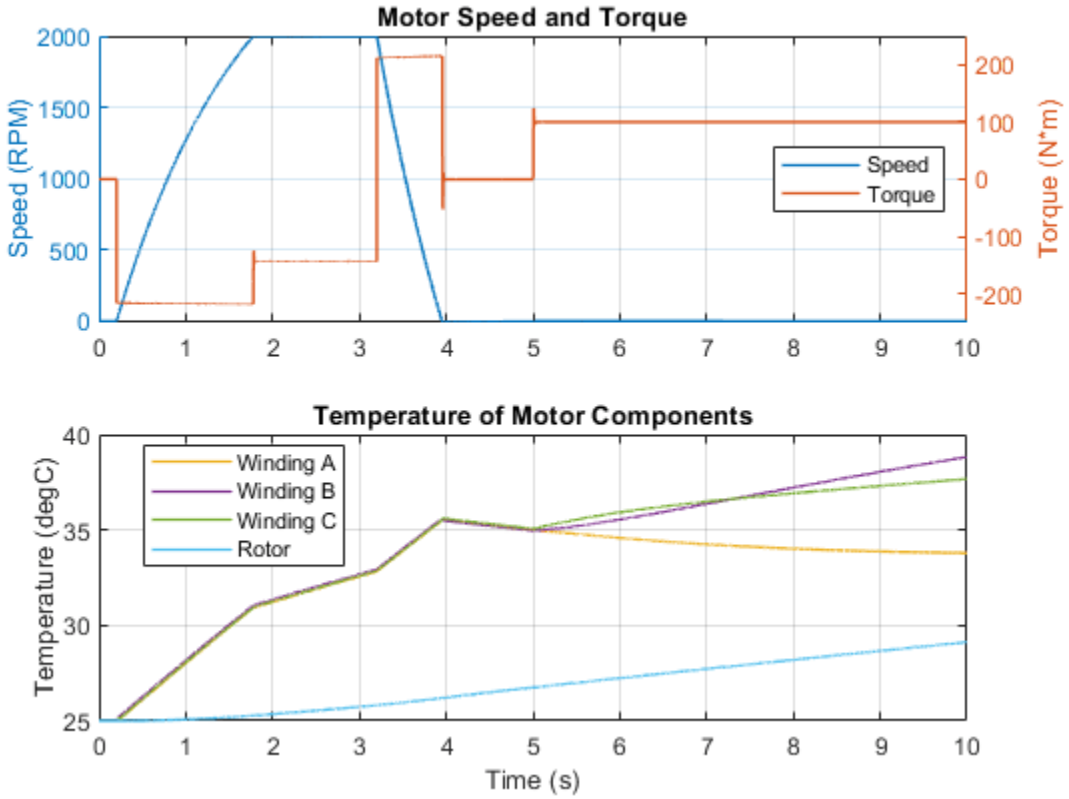



### Simulation Results from Simscape Logging

The plot below shows motor speed and winding currents as the control system attempts to track a reference signal. A load torque is applied to the shaft, and the effect on the system is shown in the winding currents.




The plot below shows motor speed, torque, and temperature of motor components. The thermal model of the motor models heat transfer between the three windings and the rotor.




### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 50 microseconds.

# Stepper Motor with Control

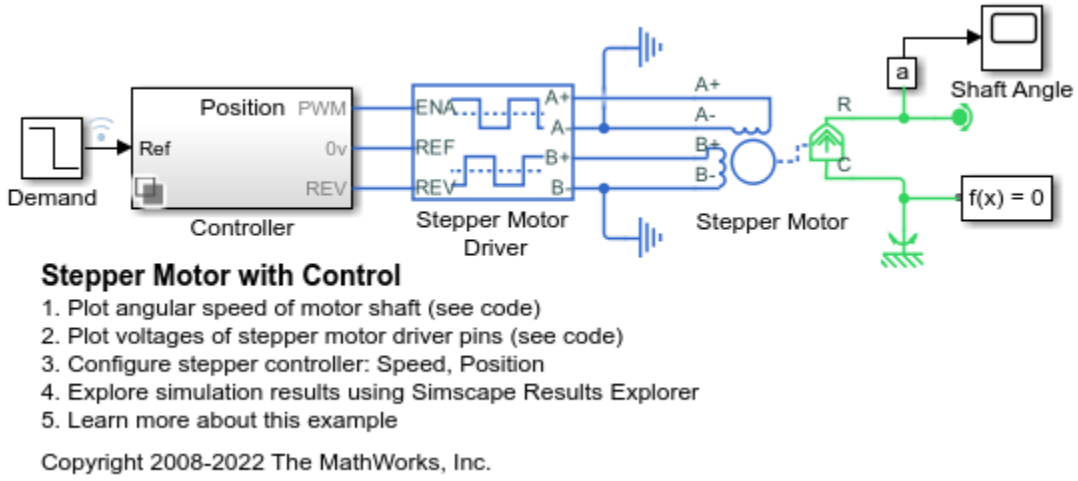
This model shows how to use the Stepper Motor Driver and Stepper Motor blocks together to implement a controlled permanent magnet stepper motor. The model provides two controller options: one to control position and one to control speed. To change the controller type, right-click on the Controller block, select Variant->Override using-> and select Position or Speed.

The stepper has a full step size of 1.8 degrees. In position control model, the input Ref is the desired number of steps. In speed control mode, the input Ref is the desired number of steps per second.

This model is a system-level model suitable for studying the dynamics of the stepper and whether step angle will slip when driving a given load. It can also be used to tune the stepper controller to improve stepping performance. Often the controller is either partly or fully implemented on an off-the-shelf stepper controller module.

The alternative of implementing the algorithm on a microprocessor (such as a PIC) gives greater flexibility, and the microprocessor may also be used to control other parts of the overall system. In this case parts of the Stepper Motor Driver block may also be implemented in on the microprocessor, leaving only the power amplifier stage in analog electronics.

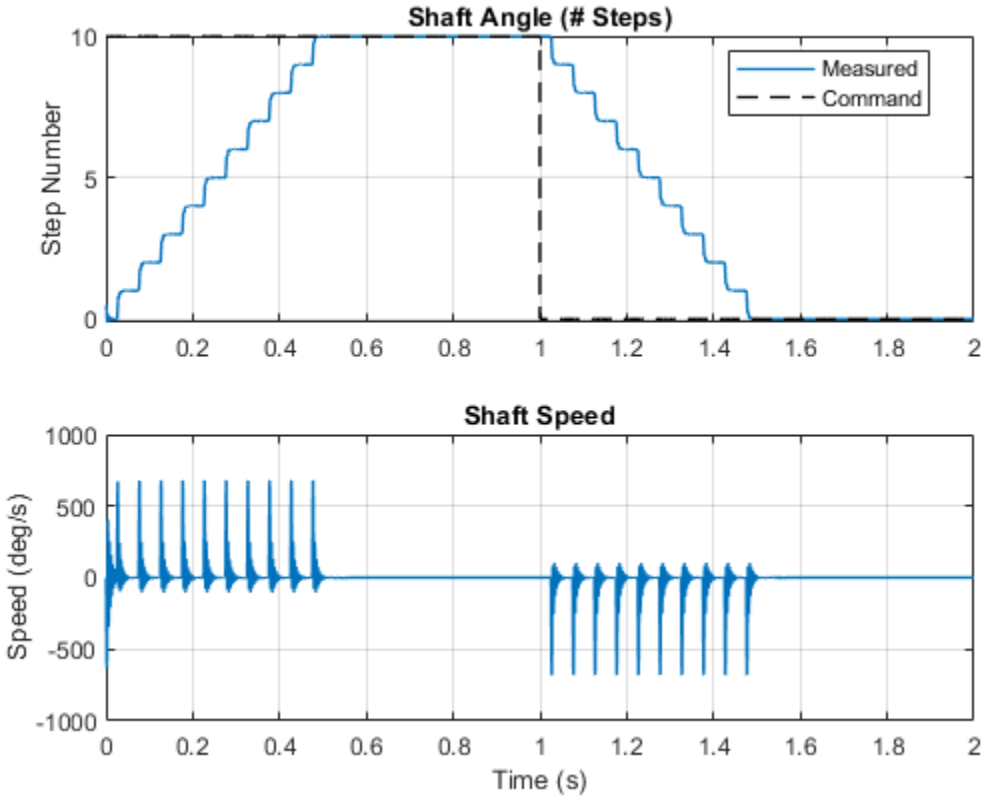
### Model




### Simulation Results from Simscape Logging

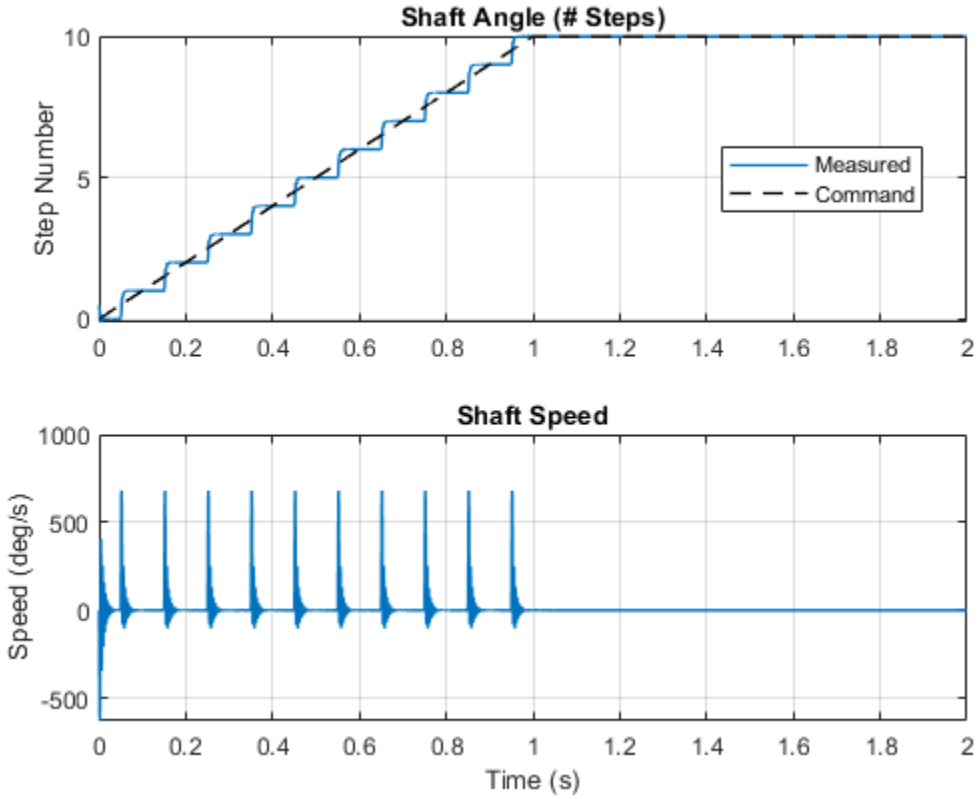
Position Control Test

The motor shaft angle as compared to the demand signal. A position control algorithm accepts a position command as a number of steps and converts it to a pulse train that controls the stepper motor driver. The spikes on the angular velocity plot occur as the shaft settles into its commanded position.

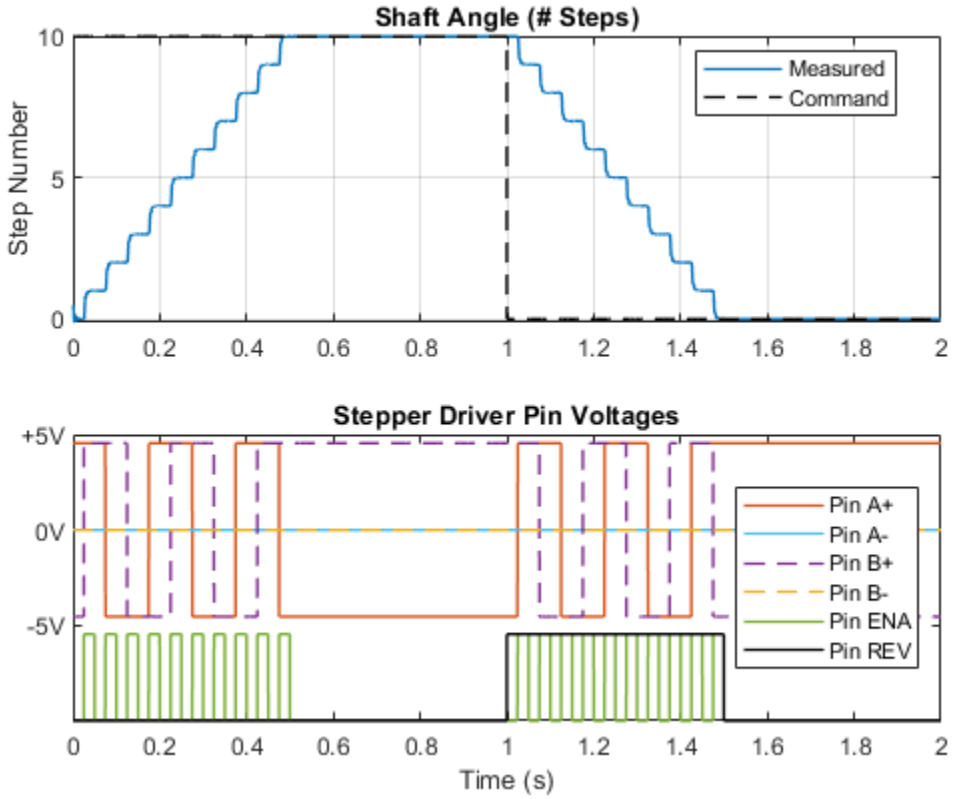



Speed Control Test

The plots below show the motor shaft angle as compared to the demand signal. A speed control algorithm accepts a speed command as a number of steps per second and converts it to a pulse train that controls the stepper motor driver. The spikes on the angular velocity plot occur as the shaft settles into the current step.




The plot below shows how the states of the pins on the stepper driver affect the motion of the stepper motor. The driver initiates a step each time the ENA signal rises above the Enable threshold voltage.

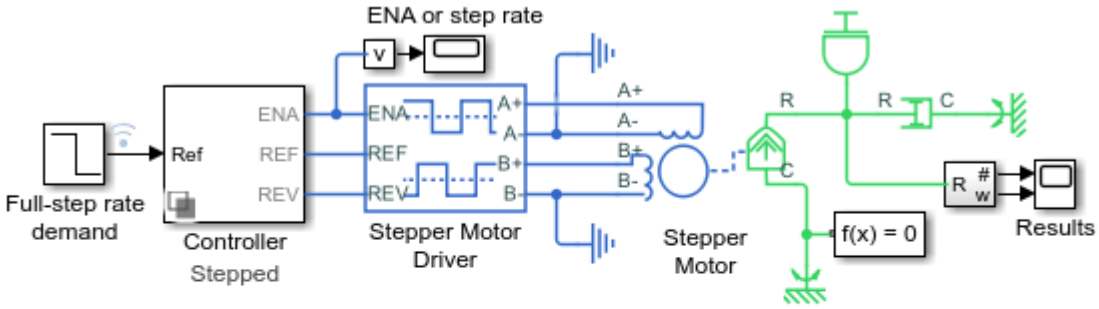

Shaft Angle (# Steps)
Step Number
Measured
Command
Stepper Driver Pin Voltages
+5V
0V
-5V
Pin A+
Pin A-
Pin B+
Pin B-
Pin ENA
Pin REV
Time (s)

# Stepper Motor Averaged Mode

This model shows the Stepper Motor simulating in Stepping and Averaged simulation modes. The purpose of Averaged mode is faster simulation for any loads that do not cause slip. To avoid incorrect interpretation of results, the stepper motor has an approximate detection of slip which can be set to generate a warning or an error.

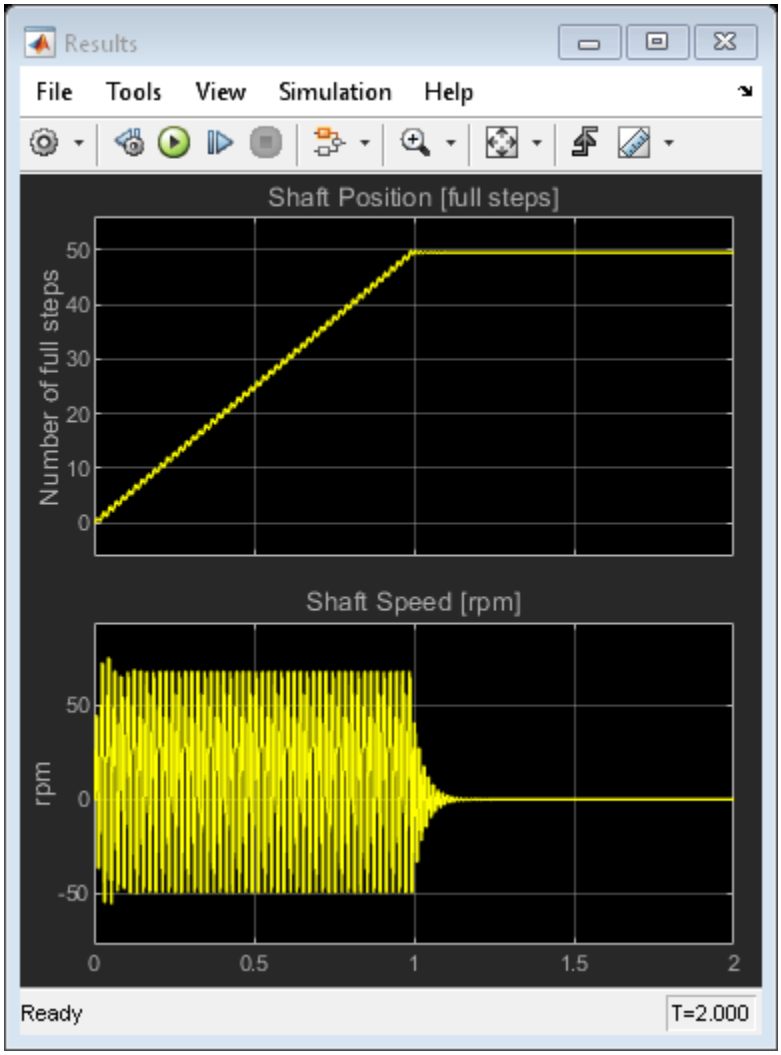
The nominal inertial load for the stepper motor is 1e-3 kg*m^2. If you increase load inertia (workspace parameter J) to 0.2 kg*m^2 or load damping (workspace parameter L) to 2 N*m/s, both simulation modes will indicate slip for a step rate demand of 50 full steps per second. Once slip occurs, results will diverge as the Averaged mode assumes the stepper speed controller detects slip and adjusts the demand to maintain synchronous operation.

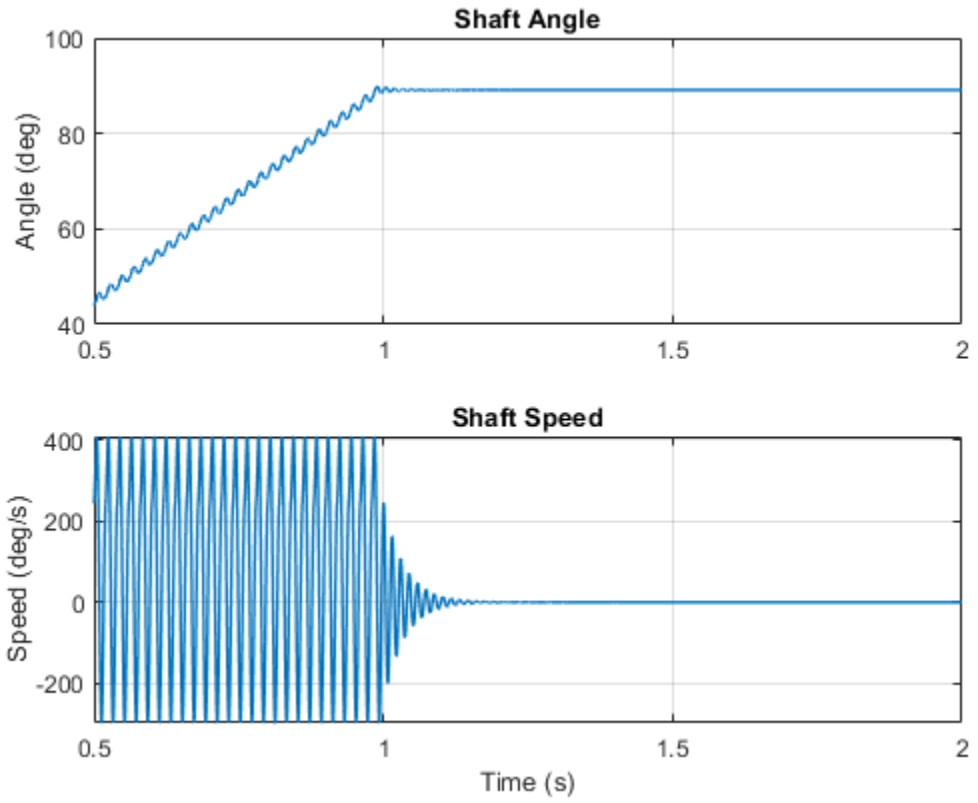
### Model




### Simulation Results from Simscape Logging

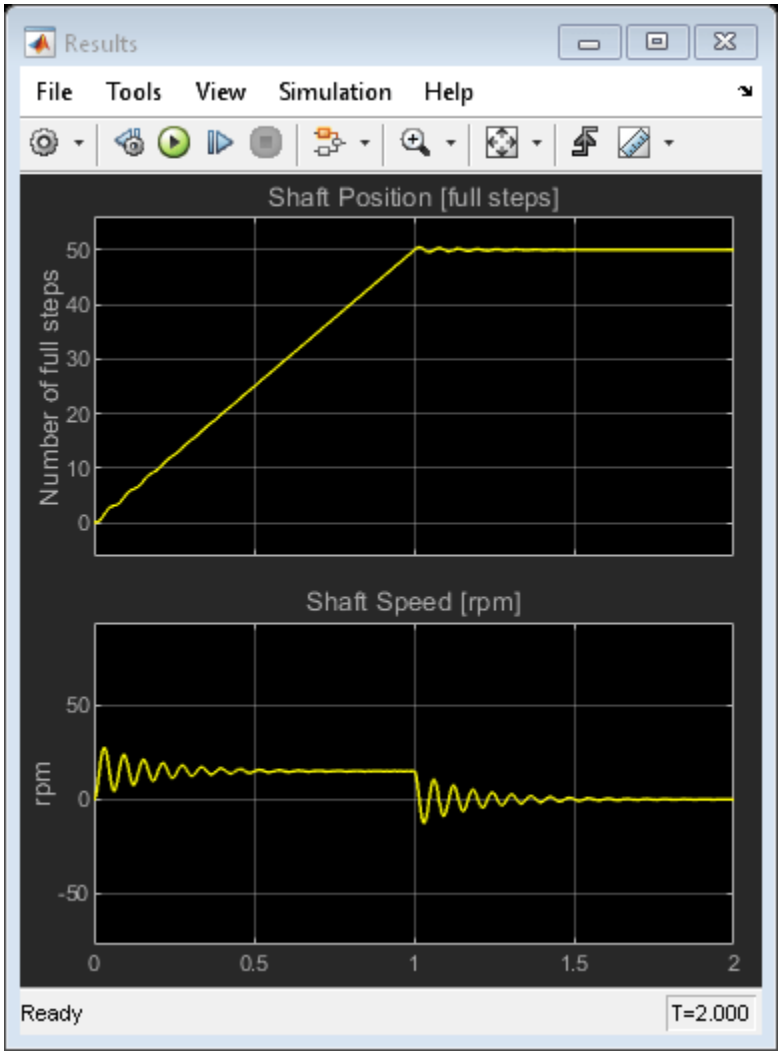
The plot below shows the shaft angle and shaft speed of the stepper motor in full stepping mode. Oscillations in the shaft speed occur as the shaft settles into the current step. In averaged mode, the angle and speed are smooth as individual steps are not simulated.

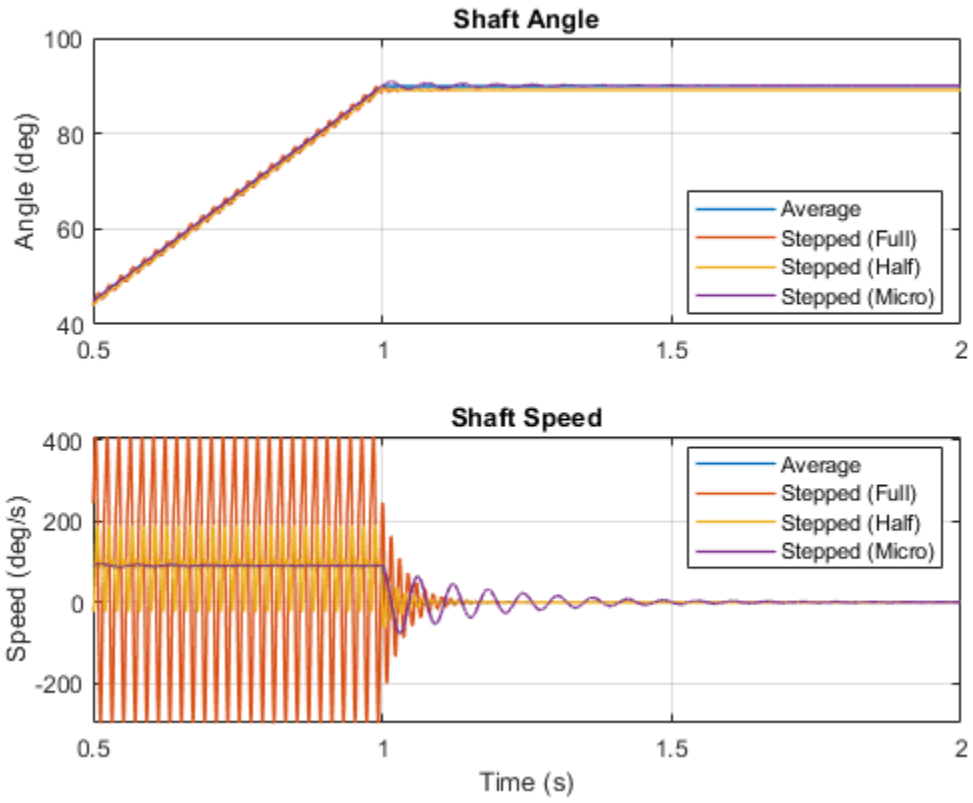

Results
File
Tools
View
Simulation
Help
Shaft Position [full steps]
Number of full steps
50
40
30
20
10
0
Shaft Speed [rpm]
rpm
50
0
-50
0
0.5
1
1.5
2
Ready
T=2.000




The plot below compares the shaft angle and shaft speed of the stepper motor in averaged and stepped mode. For the stepped mode, the plot compares three stepping methods: Full stepping, Half stepping, and Microstepping.

```
Full stepping mode selected
Half stepping mode selected
Microstepping mode selected
```


Results
File
Tools
View
Simulation
Help
Shaft Position [full steps]
Number of full steps
50
40
30
20
10
0
Shaft Speed [rpm]
rpm
50
0
-50
0
0.5
1
1.5
2
Ready
T=2.000




You can observe oscillations in the shaft speed in stepped mode, as the individual steps are simulated.

You can observe reduced oscillations with microstepping.

# Unipolar Stepper Motor with Control

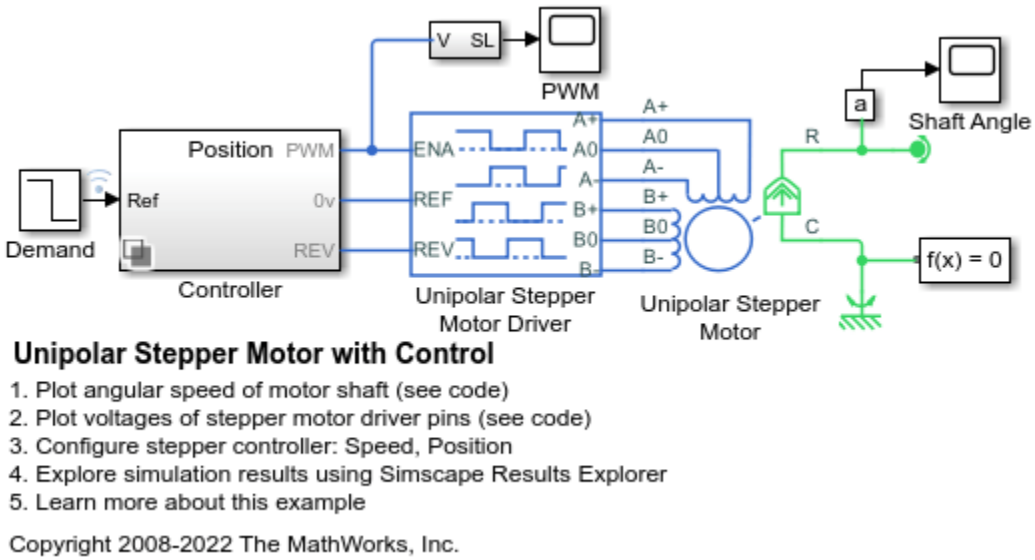
This model shows how to use the Unipolar Stepper Motor Driver and Unipolar Stepper Motor blocks together to implement a controlled permanent magnet stepper motor. The model provides two controller options: one to control position and one to control speed. To change the controller type, right-click on the Controller block, select Variant->Override using-> and select Position or Speed.

The stepper has a full step size of 1.8 degrees. In position control mode, the input Ref is the desired number of steps. In speed control mode, the input Ref is the desired number of steps per second.

This model is a system-level model suitable for studying the dynamics of the stepper and whether step angle will slip when driving a given load. It can also be used to tune the stepper controller to improve stepping performance. Often the controller is either partly or fully implemented on an off-the-shelf stepper controller module.

The alternative of implementing the algorithm on a microprocessor (such as a PIC) gives greater flexibility, and the microprocessor may also be used to control other parts of the overall system. In this case parts of the Stepper Motor Driver block may also be implemented in on the microprocessor, leaving only the power amplifier stage in analog electronics.

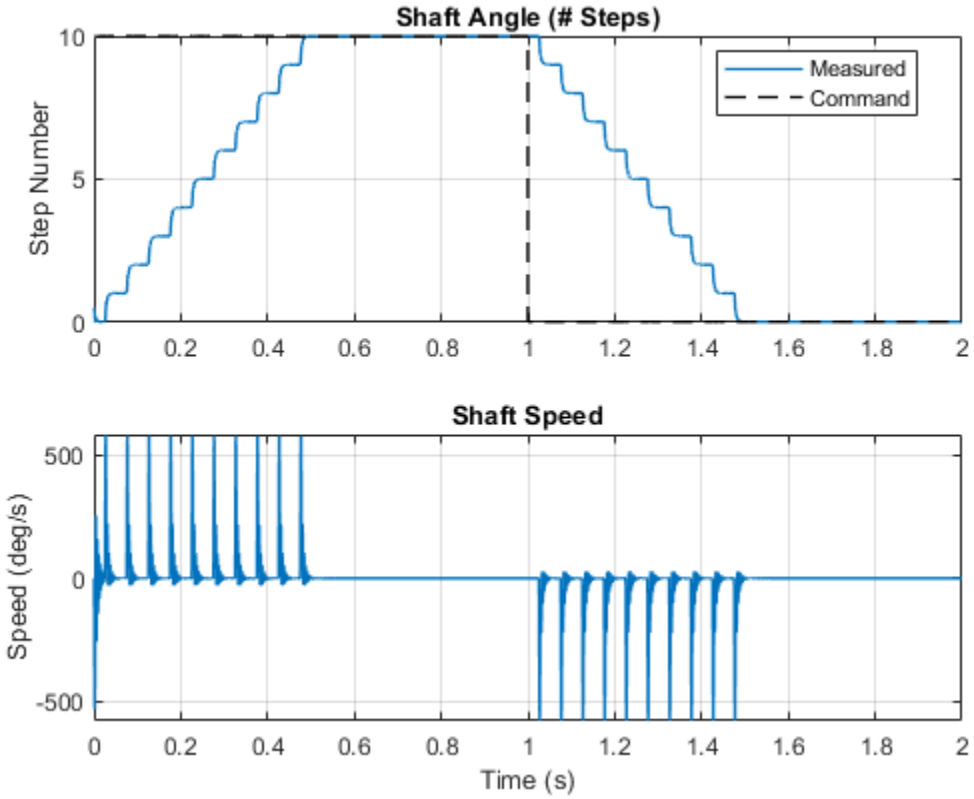
### Model




### Simulation Results from Simscape Logging

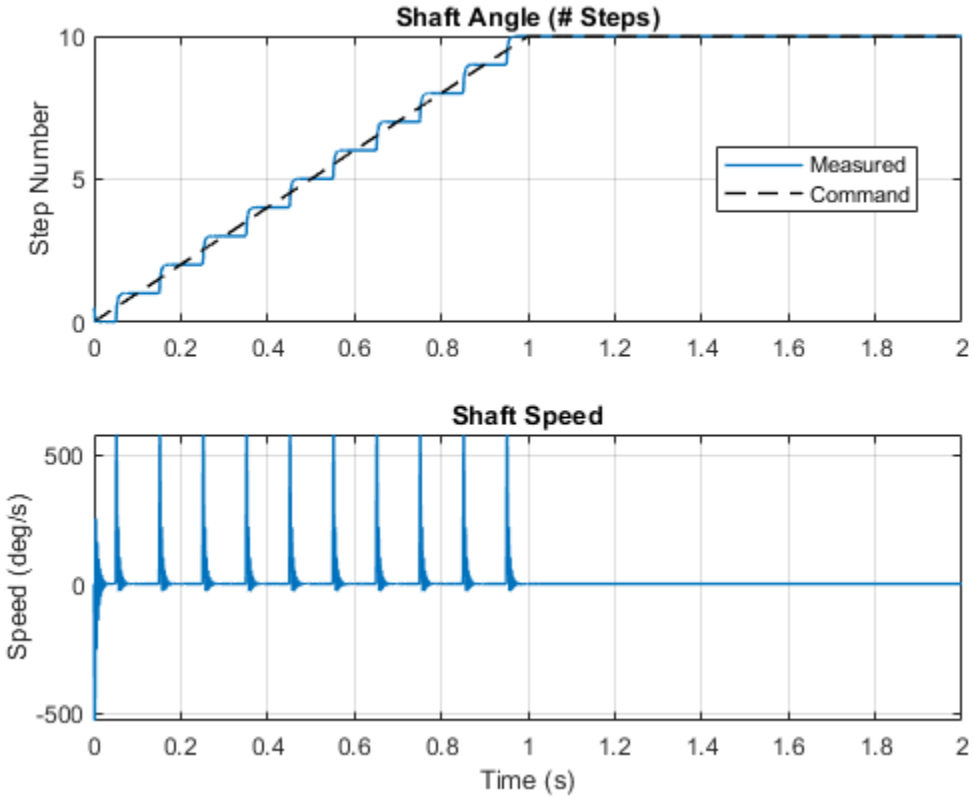
Position Control Test

The motor shaft angle as compared to the demand signal. A position control algorithm accepts a position command as a number of steps and converts it to a pulse train that controls the stepper motor driver. The spikes on the angular velocity plot occur as the shaft settles into its commanded position.

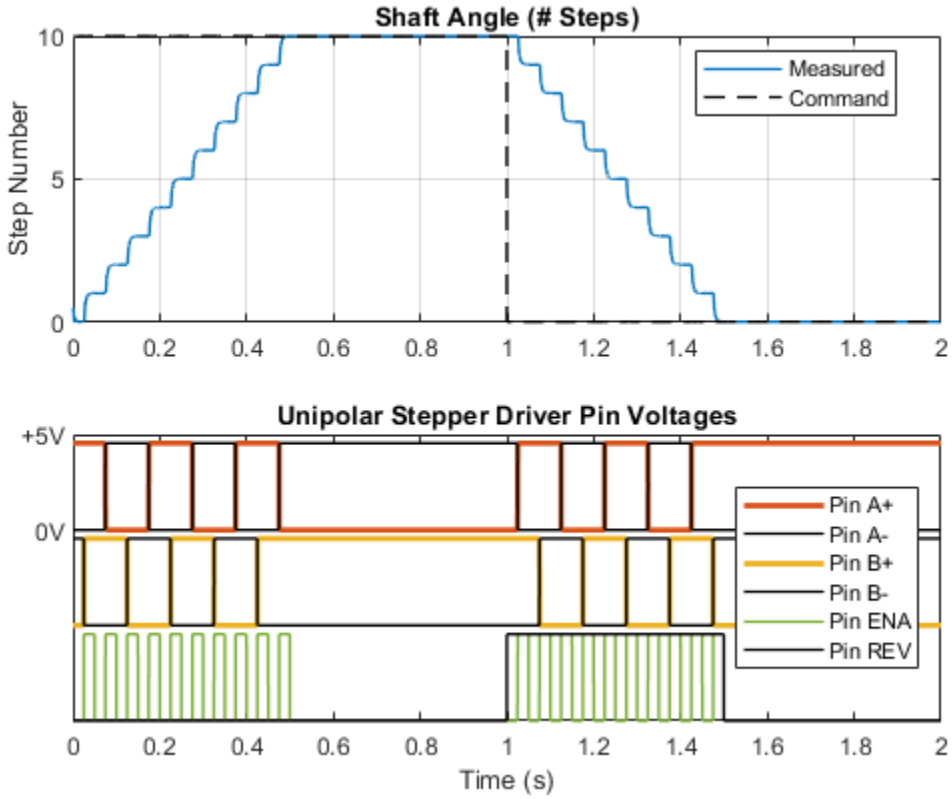



Speed Control Test

The plots below show the motor shaft angle as compared to the demand signal. A speed control algorithm accepts a speed command as a number of steps per second and converts it to a pulse train that controls the stepper motor driver. The spikes on the angular velocity plot occur as the shaft settles into the current step.




The plot below shows how the states of the pins on the unipolar stepper driver affect the motion of the unipolar stepper motor. The driver initiates a step each time the ENA signal rises above the Enable threshold voltage. The driver sets the common connections A0 and B0 to always be high, and grounds the A+, A-, B+ and B- as required.

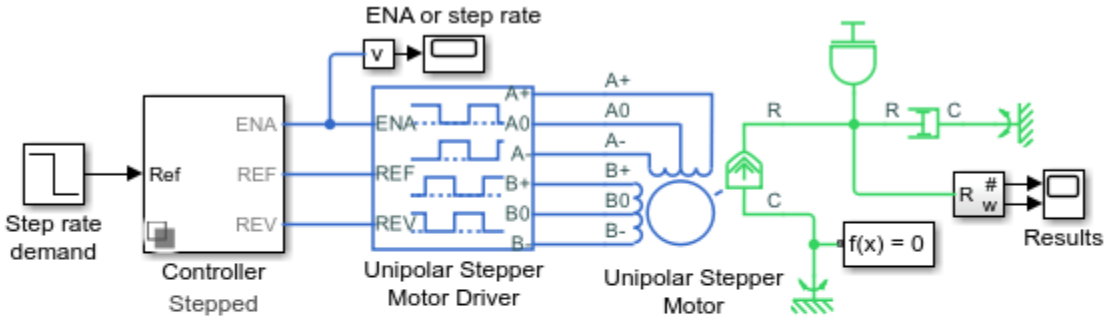

Shaft Angle (# Steps)
Step Number
10
5
0
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
Measured
Command
Unipolar Stepper Driver Pin Voltages
+5V
0V
Pin A+
Pin A-
Pin B+
Pin B-
Pin ENA
Pin REV
Time (s)

# Unipolar Stepper Motor Averaged Mode

This model shows the Unipolar Stepper Motor simulating in Stepping and Averaged simulation modes. The purpose of Averaged mode is faster simulation for any loads that do not cause slip. To avoid incorrect interpretation of results, the stepper motor has an approximate detection of slip which can be set to generate a warning or an error.

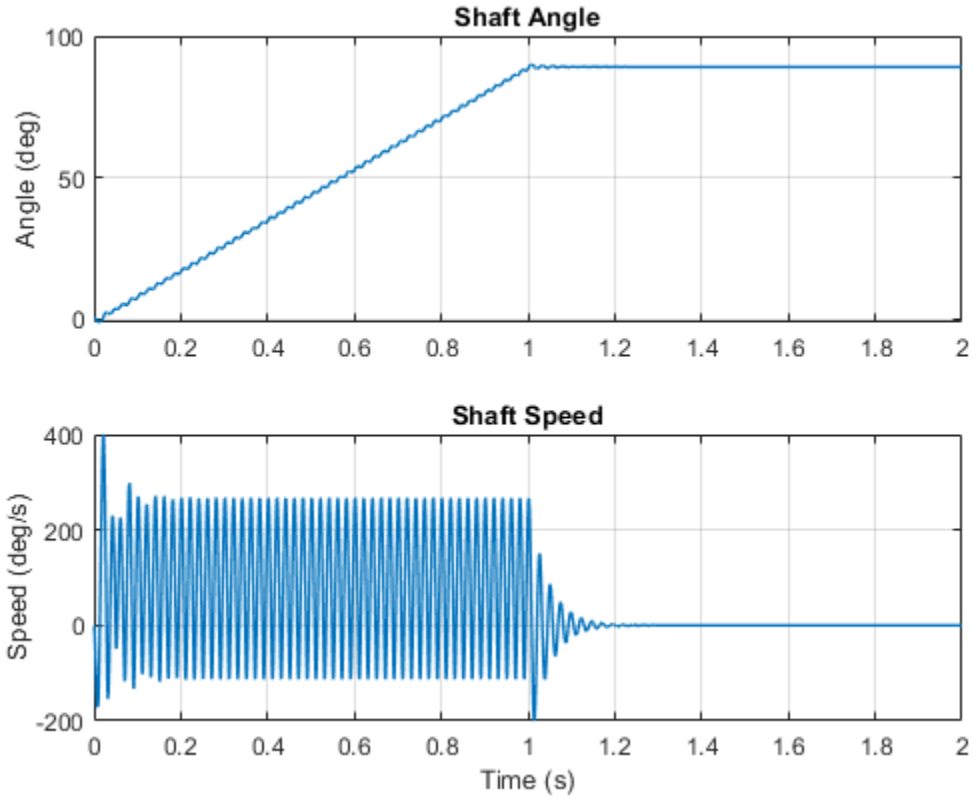
Use the two callback blocks to switch between simulation modes to verify results. If you increase load inertia (workspace parameter J) to 0.05 kg*m^2 or load damping (workspace parameter L) to 2 N*m/s, both simulation modes will indicate slip. Once slip occurs, results will diverge as the Averaged mode assumes the stepper speed controller detects slip and adjusts the demand to maintain synchronous operation.

### Model

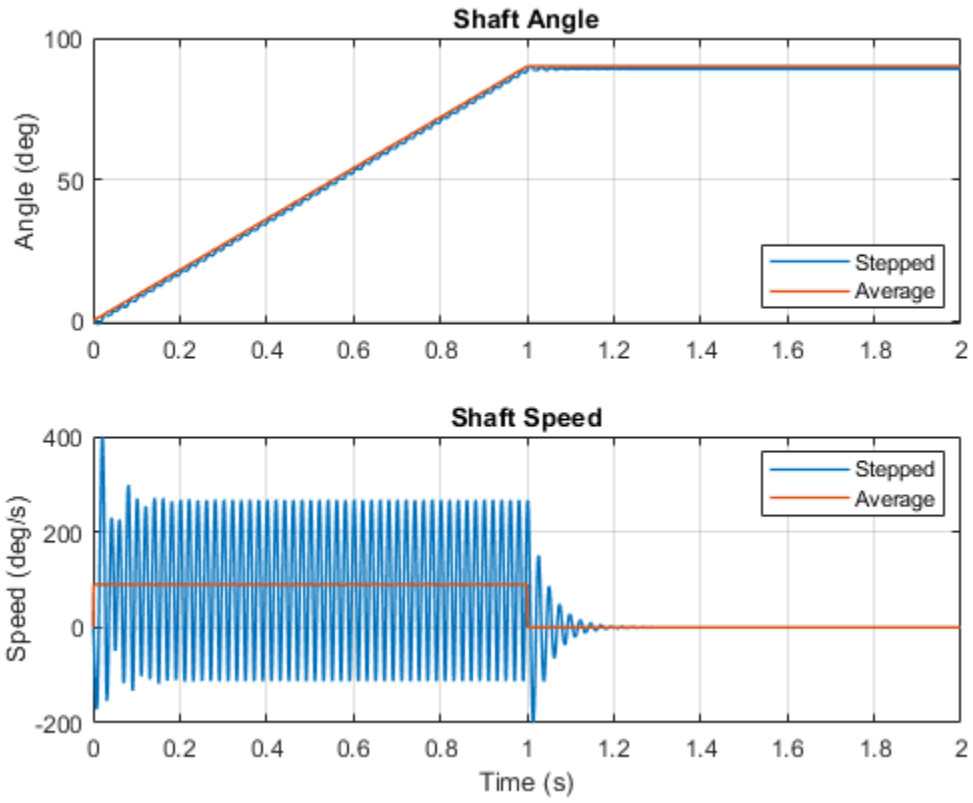



### Simulation Results from Simscape Logging

The plot below shows the shaft angle and shaft speed of the stepper motor in full stepping mode. Oscillations in the shaft speed occur as the shaft settles into the current step. In averaged mode, the angle and speed are smooth as individual steps are not simulated.




The plot below compares the shaft angle and shaft speed of the stepper motor in stepped and averaged mode. Oscillations in the shaft speed are seen in stepped mode as the individual steps are simulated.


Shaft Angle
Angle (deg)
100
50
0
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
Stepped
Average
Shaft Speed
Speed (deg/s)
400
200
0
-200
Time (s)
Stepped
Average

# Hybrid Linear Actuator

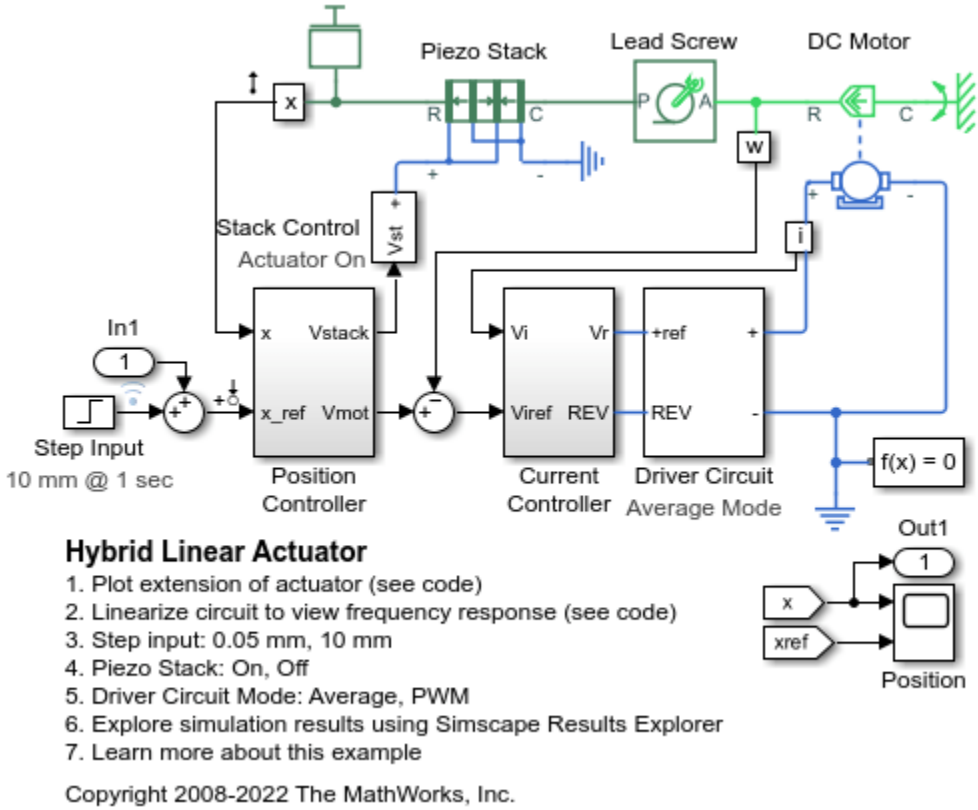
This example shows a hybrid actuator consisting of a DC motor plus lead screw in series with a piezoelectric stack. The DC motor and lead screw combination supports large displacements (tens of millimeters), but is dynamically slow when tracking the reference demand x_ref. Conversely the piezoelectric stack only supports a maximum displacement of +-0.1mm, but has a very fast dynamic response. Combining the two actuator technologies creates a large stroke actuator with highly precise positioning.

An example application is antenna pointing control when tracking a satellite - large amplitude motion with time constants measured in a few seconds is required to move between satellites, whereas fine control is required to track a given non-geostationary satellite.

Setting the step input to 0.05mm shows system performance for fine control. The initial motion is provided by the piezoelectric stack, and this is then washed out as the DC Motor rotor angle moves to the new operating point. Setting the input to 10mm shows system performance for a larger input.

This model can be used to obtain the frequency response of the system. MATLAB® command linmod can be used to linearize the model. If you have Simulink® Control Design™, then a good way to understand the overall actuator performance is to plot a Bode plot. Open the model ee_actuator_dc_piezo. On the Apps tab, under Control Systems, click Model Linearizer. In the Model Linearizer, on the Linear Analysis tab, in the Linearize section, click Bode. Note that the initial state on the current controller integrator has been set to a small non-zero number to avoid linearizing the Abs block around zero. You can disable the piezoelectric stack to generate a Bode plot for the DC Motor by itself.

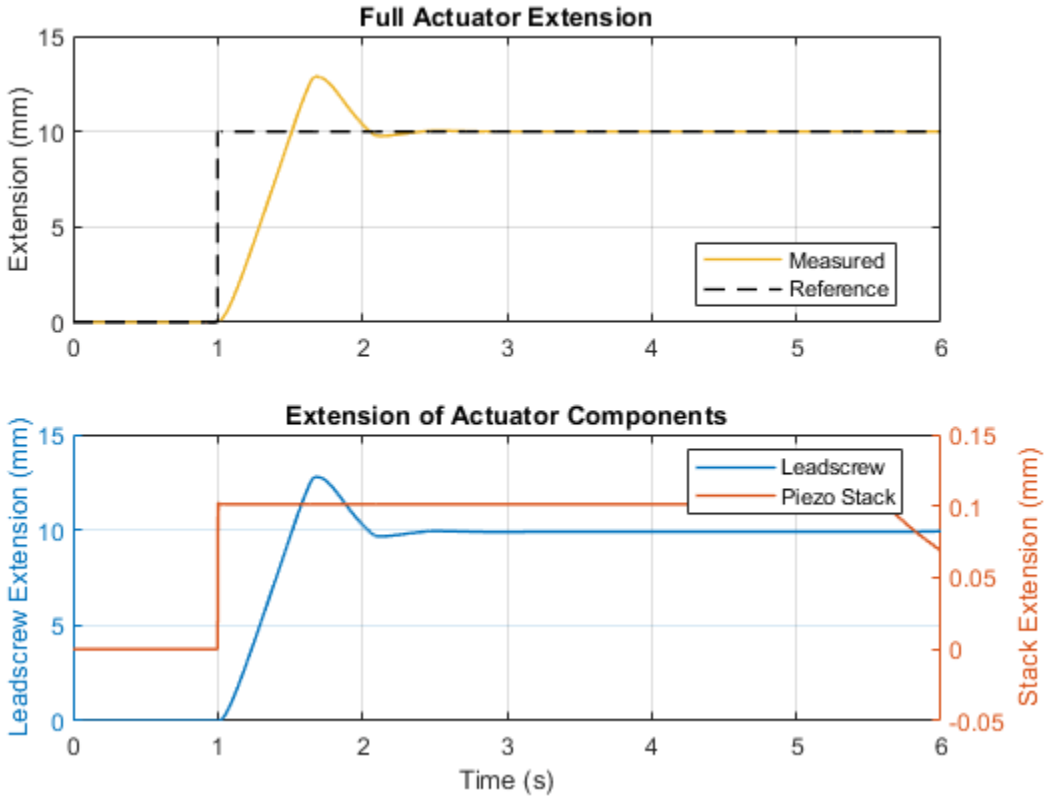
### Model



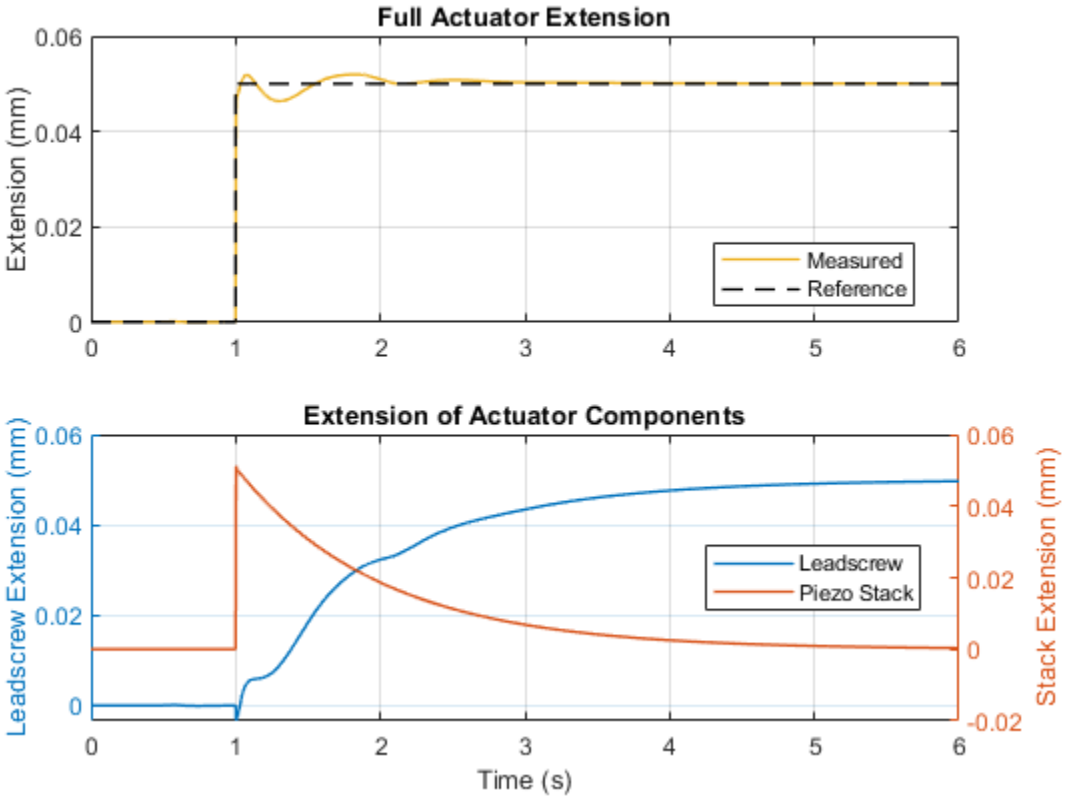

### Simulation Results from Simscape Logging

The plot below shows the extension of the actuator as the control system attempts to track a reference input. The Piezo Stack can react much faster than the DC motor but has a limited range. The control system uses them together to track the reference signal.

Test with step input of 10 mm

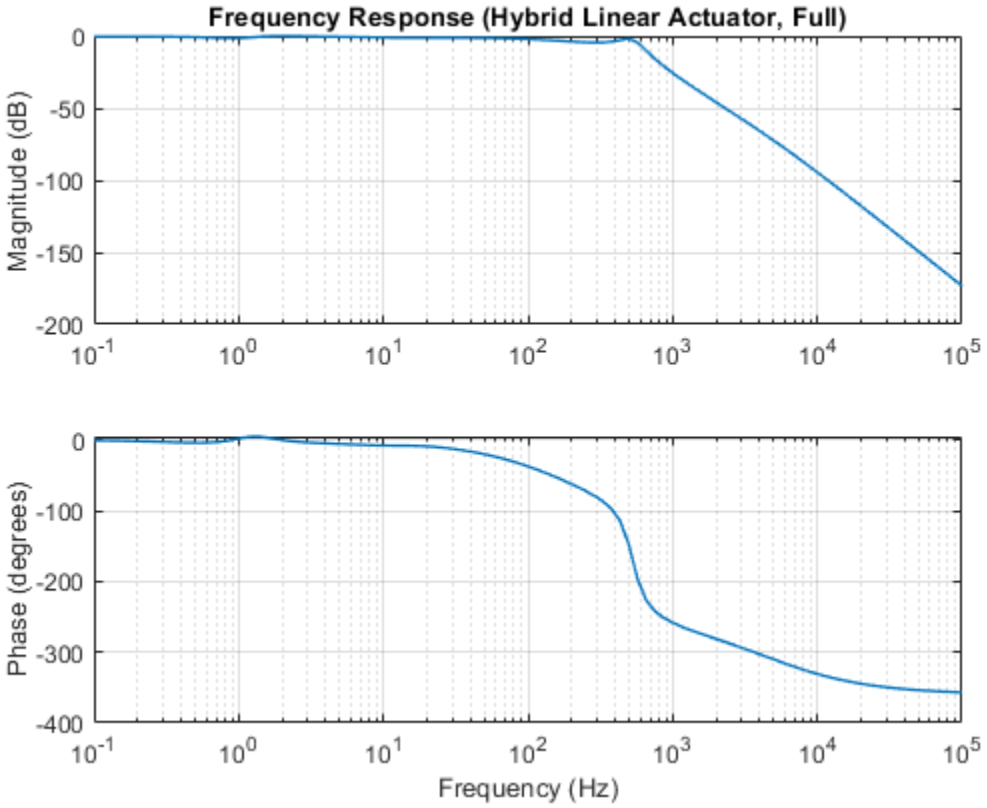



Test with step input of 0.05 mm

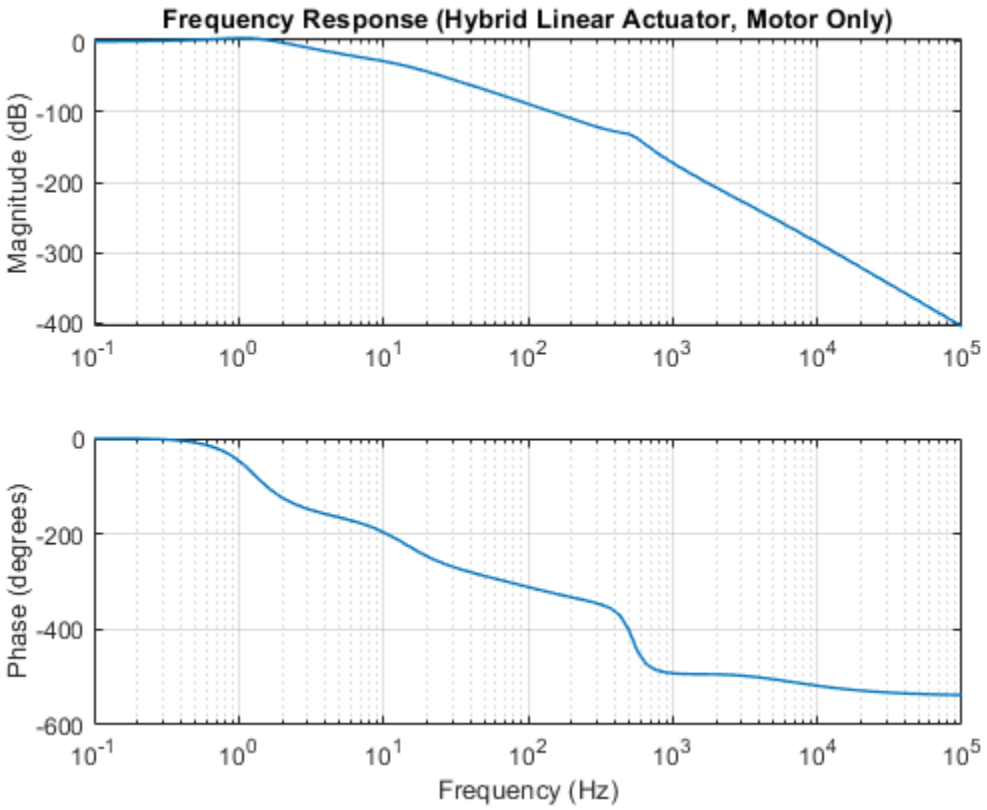



### Frequency Response

Frequency response of full hybrid actuator




Frequency response of DC motor only

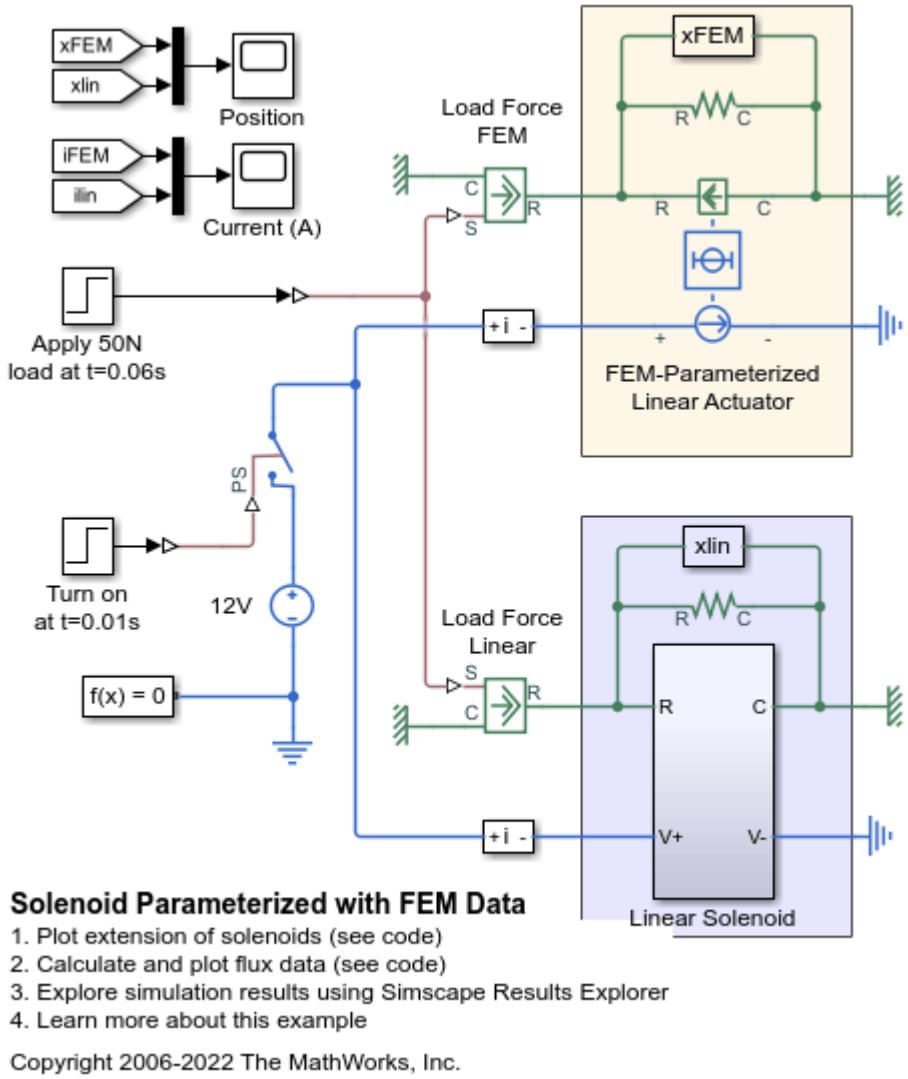

Frequency Response (Hybrid Linear Actuator, Motor Only)
Magnitude (dB)
0
-100
-200
-300
-400
Phase (degrees)
0
-200
-400
-600
$10^{-1}$
$10^{0}$
$10^{1}$
$10^{2}$
$10^{3}$
$10^{4}$
$10^{5}$
Frequency (Hz)

# Solenoid Parameterized with FEM Data

This example shows a limited travel solenoid with return spring. When unpowered, the spring holds the plunger at a distance of 0.1mm from the fully energized position. At 0.1 seconds, the solenoid is powered on and the displacement goes to zero. At 0.06s a force higher than the holding force is applied, and the plunger moves to its maximum travel of 0.2mm. The solenoid force and back emf characteristics are defined by the FEM-Parameterized Linear Actuator block. This block takes data in the format typically provided by a finite element magnetic field modeling tool. There are two parameterization options, one which works directly with flux data, and one which uses partial derivatives of flux with respect to current and displacement. The latter option is usually the better choice, it giving more accurate results for a given density of current and position data points. However, it requires more data pre-processing.

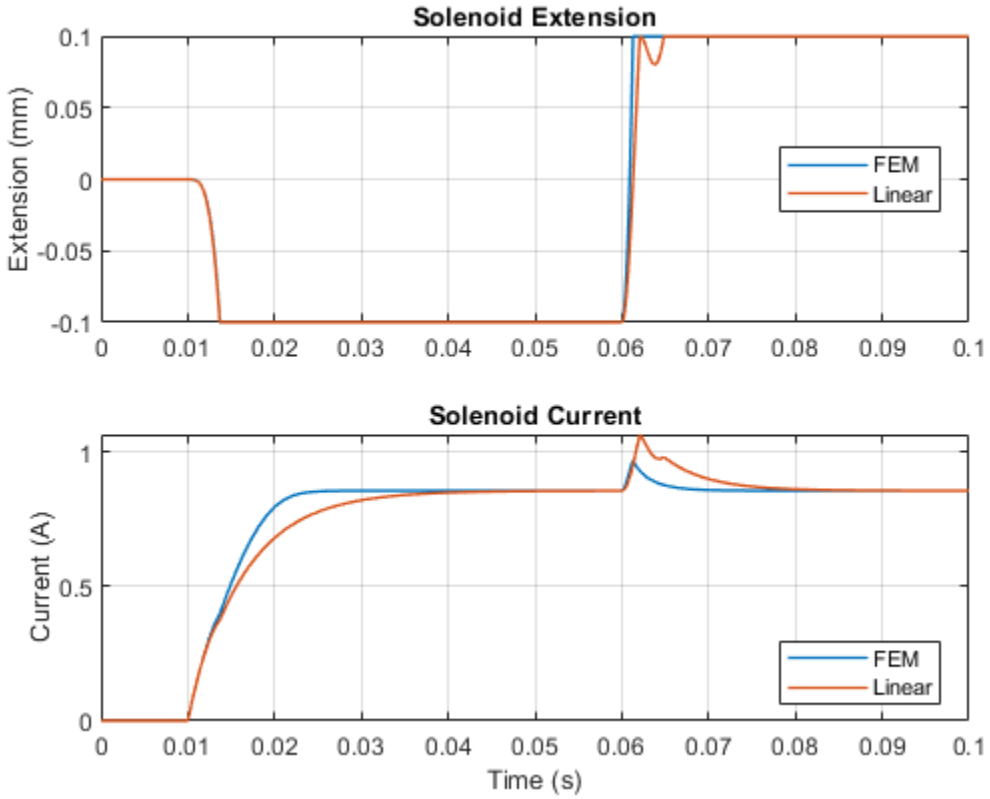
A key difference between this approach to modeling a solenoid and the approach used by the Simscape™ solenoid example 'ssc_solenoid' is that it can capture magnetic saturation effects. Conversely the Simscape example assumes a linear relationship between current and magnetic force. This linearity assumption is also made by the Simscape Electrical™ Solenoid component. In this model, the subsystem 'Simplified solenoid with no saturation effects' is parameterized with solenoid inductance data for a current of 0.1A. The inductance is the partial derivative of the flux with respect to current, and is also calculated by the initialization file. The difference in results compared to the linear model of the solenoid illustrates the effects of flux saturation. If you reduce the supply voltage from 12V to 1.4V, then the steady-state current is approximately 0.1A and the results then match once the current has reached this value.

### Model



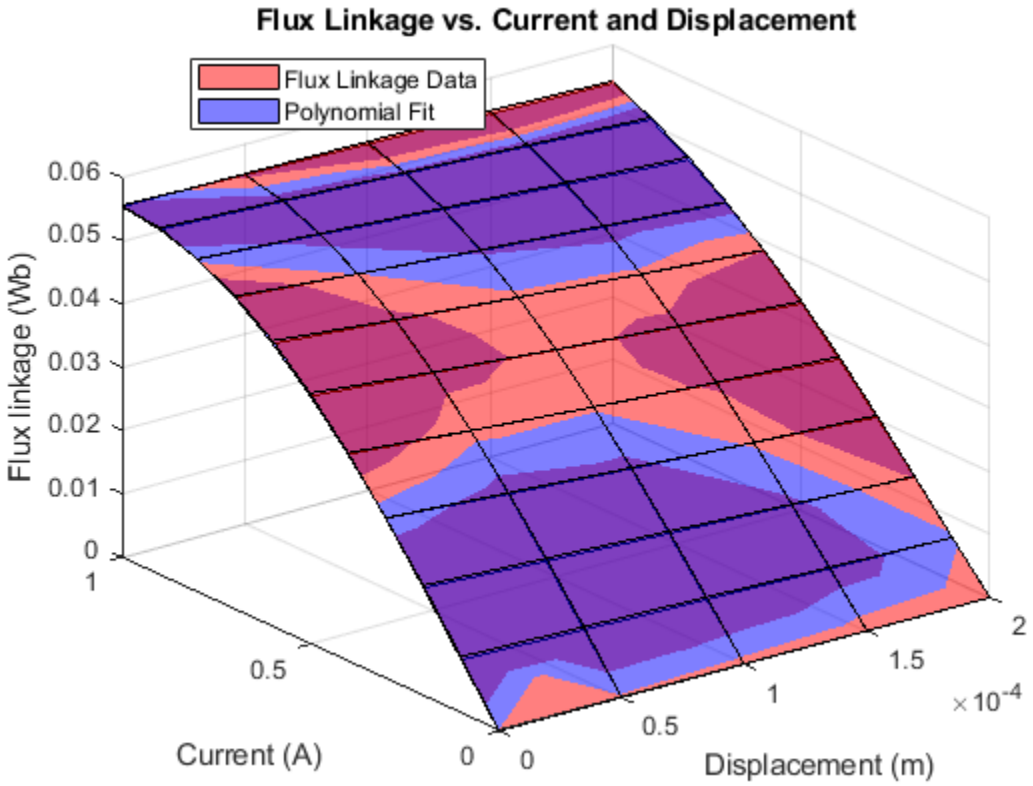

### Simulation Results from Simscape Logging

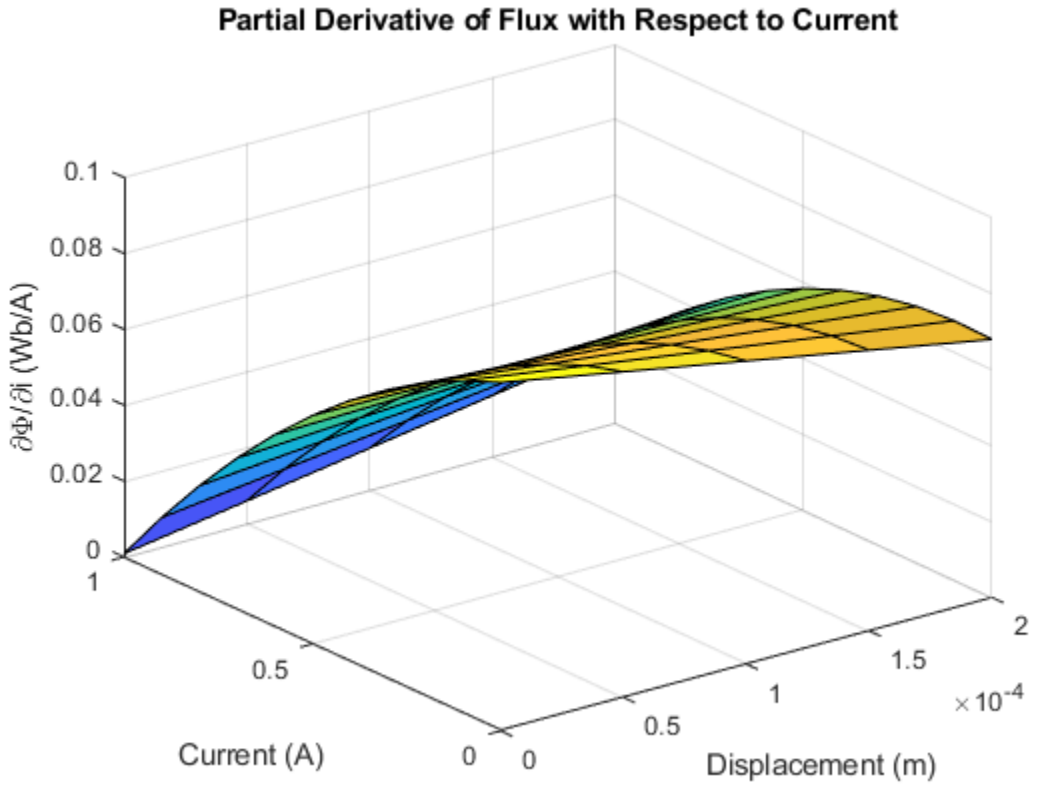
The plot below shows the difference in behavior between a linear solenoid model and a solenoid model parameterized with data provided from a finite element magnetic field modeling tool.

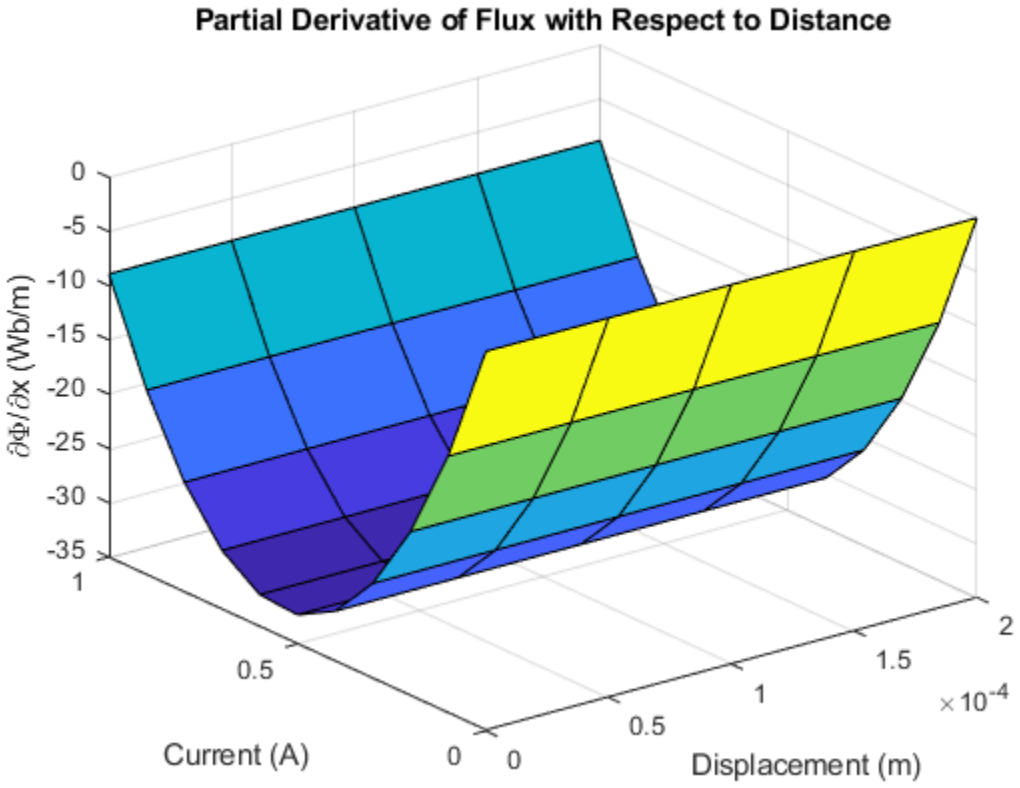


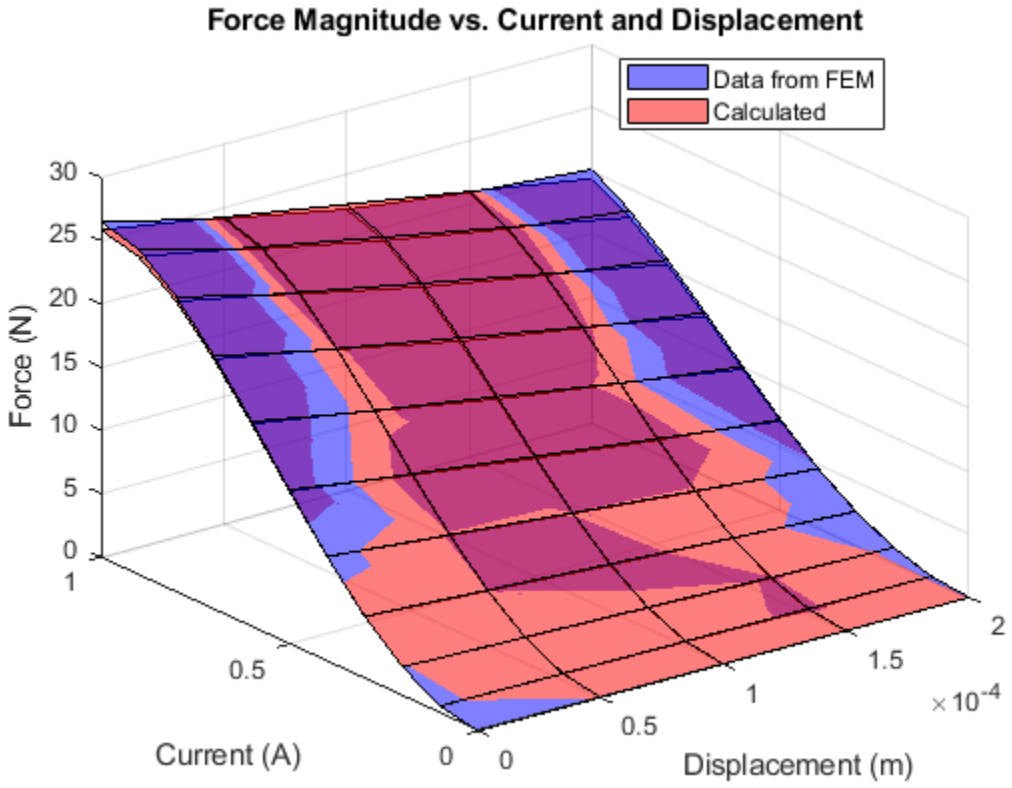

**Simulation Results from Simscape Logging**

The plots below show the flux data, including comparisons between methods of calculating the necessary parameters for the FEM-Parameterized Solenoid block.


Flux Linkage vs. Current and Displacement
Flux Linkage Data
Polynomial Fit
Flux linkage (Wb)
0.06
0.05
0.04
0.03
0.02
0.01
0
1
0.5
0
Current (A)
0
0.5
1
1.5
2
×10^-4
Displacement (m)


Partial Derivative of Flux with Respect to Current
∂Φ/∂i (Wb/A)
0.1
0.08
0.06
0.04
0.02
0
1
0.5
0
Current (A)
0
0.5
1
1.5
2
×10-4
Displacement (m)

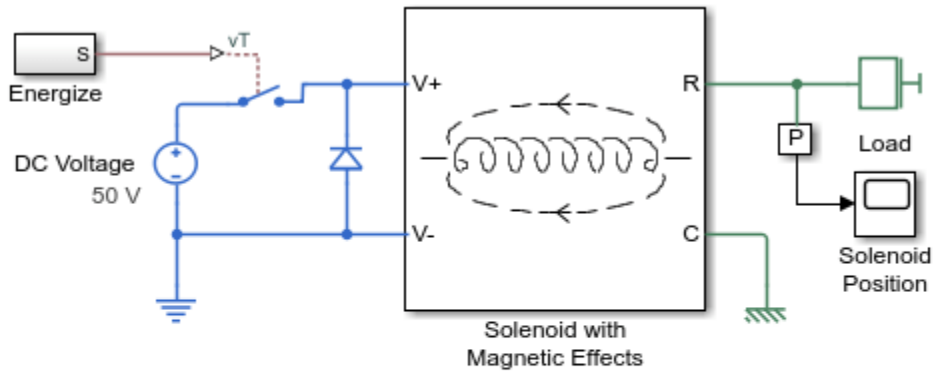

Partial Derivative of Flux with Respect to Distance
0
-5
-10
-15
-20
-25
-30
-35
∂Φ/∂x (Wb/m)
1
0.5
0
Current (A)
0
0.5
1
1.5
2
×10^-4
Displacement (m)


Force Magnitude vs. Current and Displacement
Data from FEM
Calculated
Force (N)
30
25
20
15
10
5
0
Current (A)
1
0.5
0
Displacement (m)
0
0.5
1
1.5
2
×10^-4

# Custom Solenoid with Magnetic Hysteresis

This example shows a limited travel solenoid with return spring. Magnetic hysteresis is modeled using the Reluctance with Hysteresis library block.

### Model

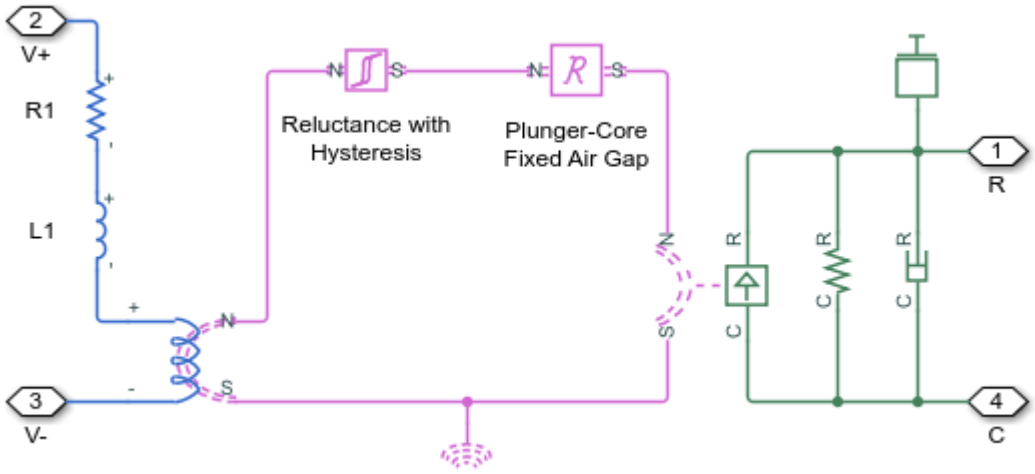



**Custom Solenoid with Magnetic Hysteresis**

1. Plot position and B-H curve of solenoid (see code)
2. Explore simulation results using Simscape Results Explorer
3. Learn more about this example

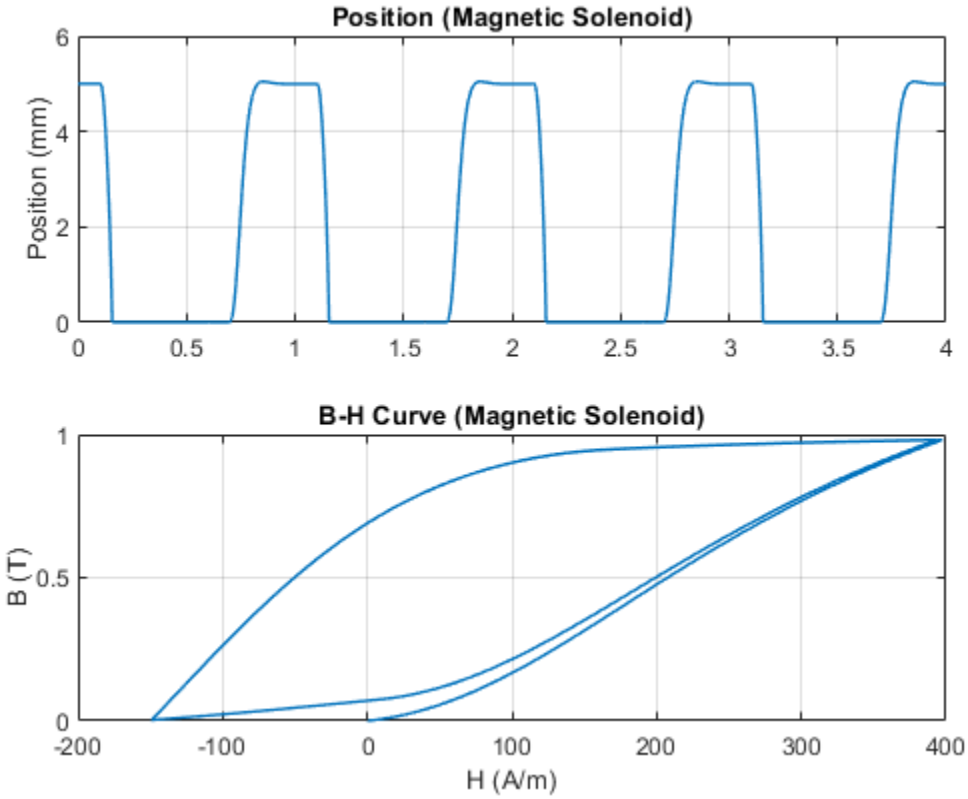
Copyright 2015-2022 The MathWorks, Inc.

### Solenoid with Magnetic Effects Subsystem




### Simulation Results from Simscape Logging

The plot below shows the solenoid position when pulsed on and off, plus the corresponding B-H trajectory.

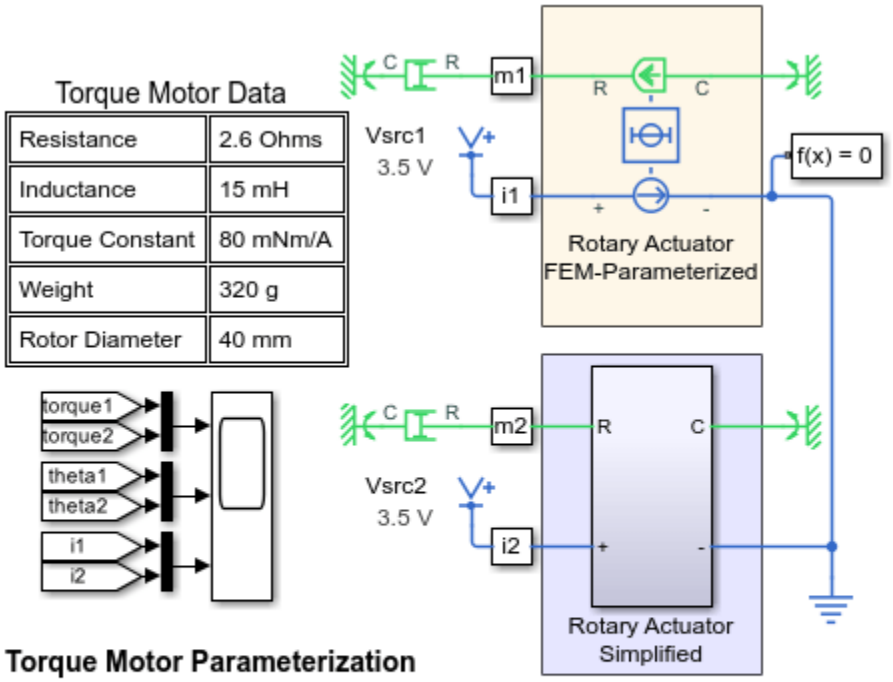

Position (Magnetic Solenoid)
Position (mm)
0
2
4
6
0
0.5
1
1.5
2
2.5
3
3.5
4
B-H Curve (Magnetic Solenoid)
B (T)
0
0.5
1
-200
-100
0
100
200
300
400
H (A/m)

# Torque Motor Parameterization

This example shows how manufacturer data for torque as a function of current and angle can be used to model a torque motor. The datasheet shows linear characteristics for rotor angles between 20 and 70 degrees and for currents where saturation does not occur. Data in this range is used to parameterize the simplified model of the torque motor. Using MATLAB® to process the data points extracted from the datasheet, we can convert manufacturer data into motor parameters that are often obtained from finite element software.

The motor models show similar results when tested under conditions where the datasheet shows linear behavior. When tested over the full range, the behaviors deviate as specified in the datasheet.

**Model**

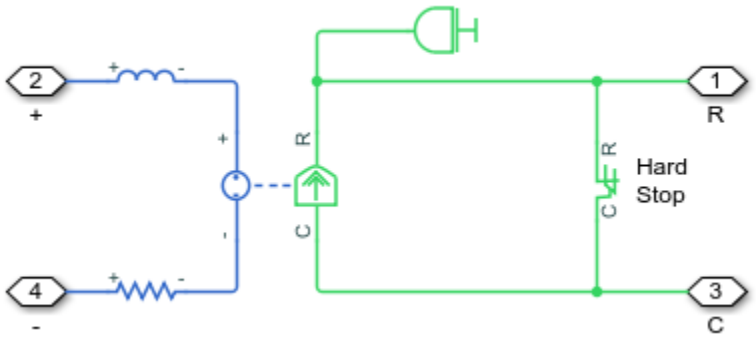



**Torque Motor Parameterization**

1. Plot shaft angle of motors (see code)
2. Set range for test: Full, Linear region
3. Calculate and plot flux data (see code)
4. Explore simulation results using Simscape Results Explorer
5. Learn more about this example

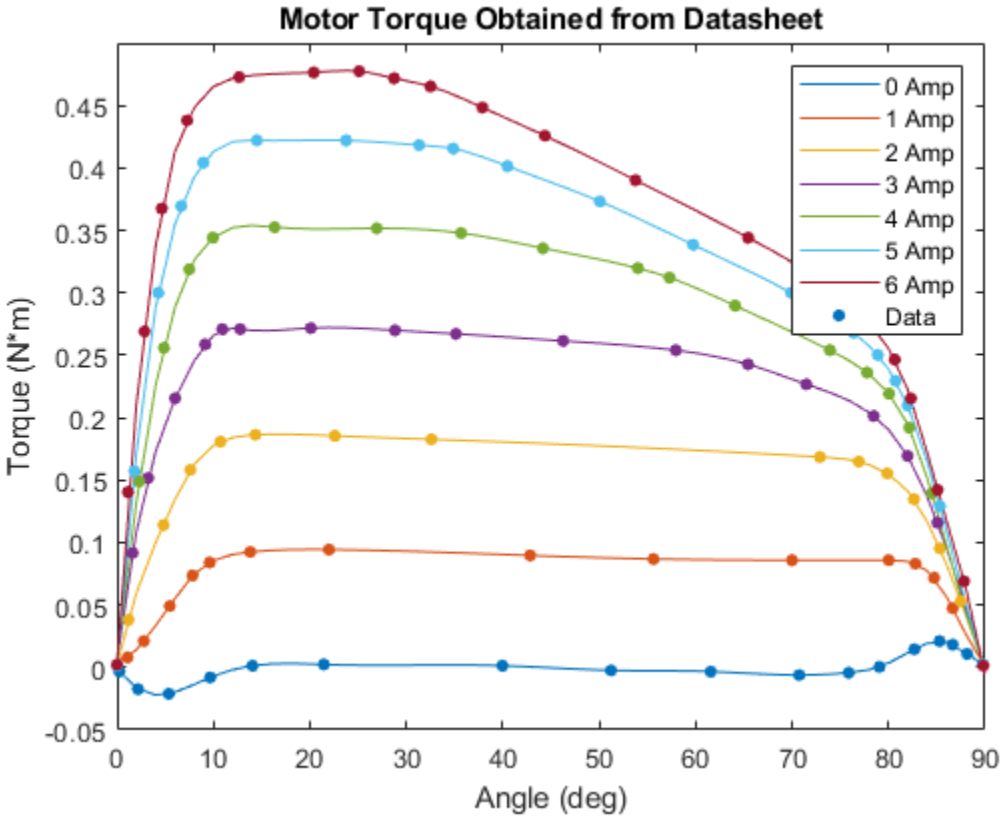
Copyright 2008-2022 The MathWorks, Inc.

### Rotary Actuator Simplified Subsystem

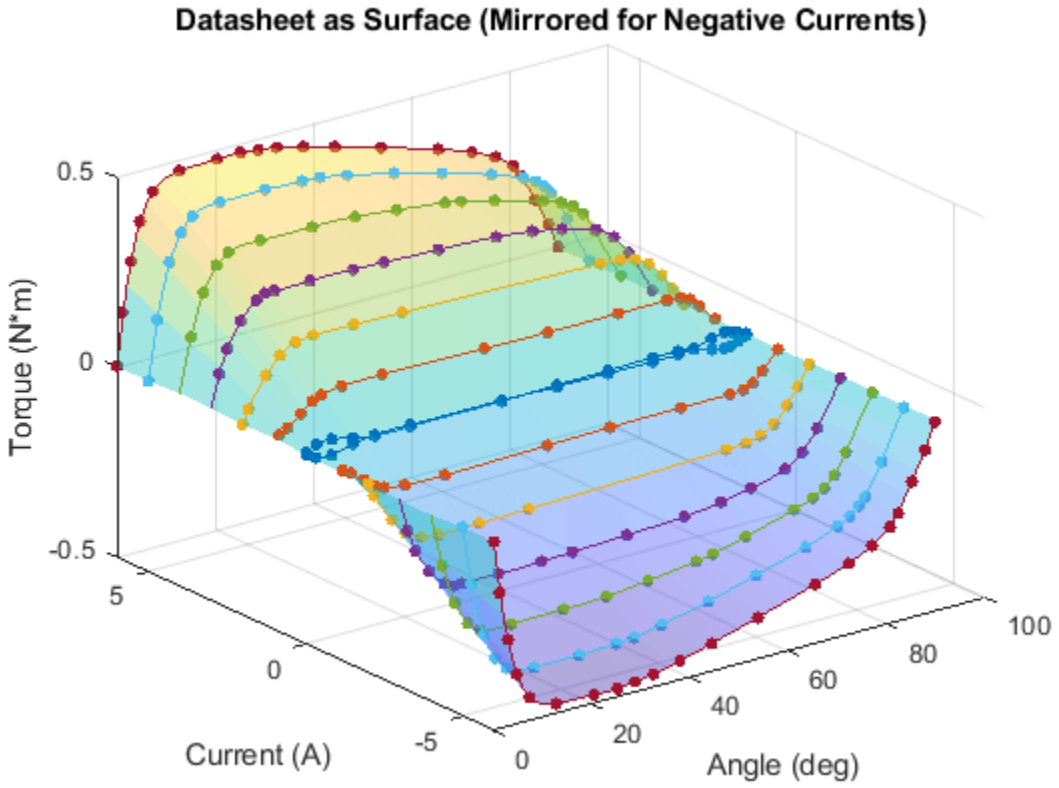



### Obtaining Motor Data from Data Sheet

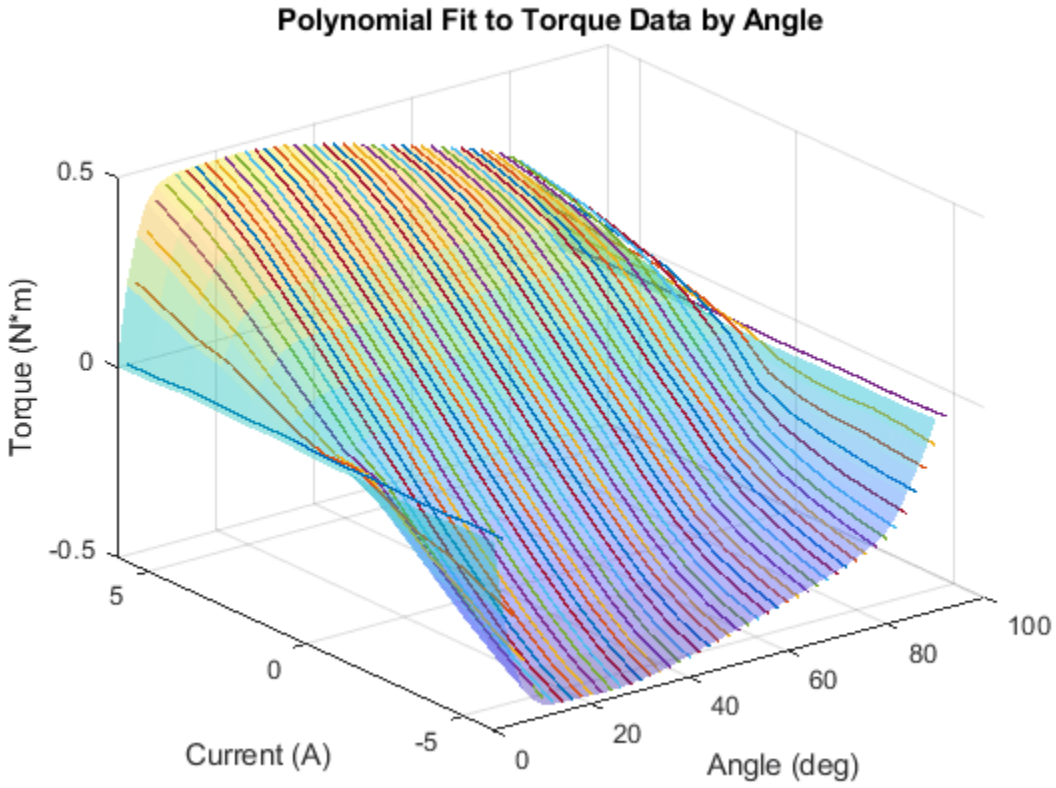
The plot below shows resampled data obtained from a motor data sheet. It shows torque produced at different rotor angles at different current levels. For some conditions (such as 2 amps, 20 deg to 70 deg), the torque is constant, but at other levels it is highly nonlinear.



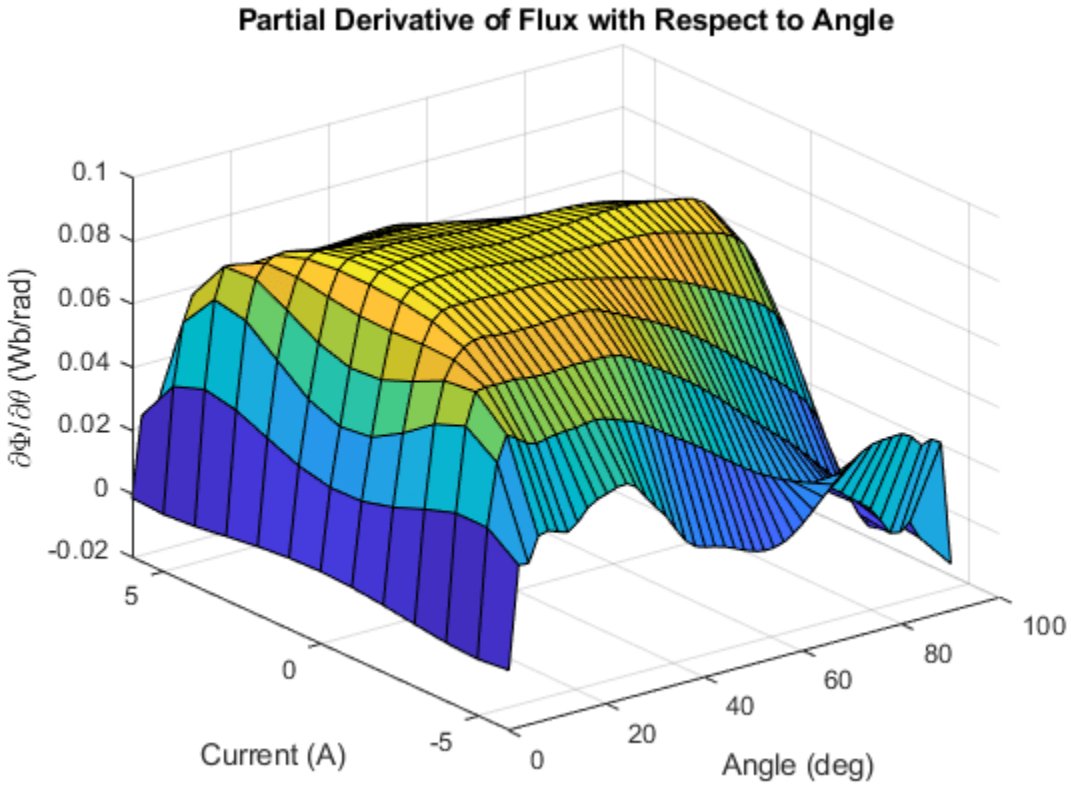

To parameterize our motor model, we need to obtain flux partial derivative with respect to angle. This script estimates dPhi/dx from torque. First, we mirror the datasheet to obtain data for negative currents and plot it as a surface.




Next, we use MATLAB to fit polynomial curves to the surface along lines of constant angle.

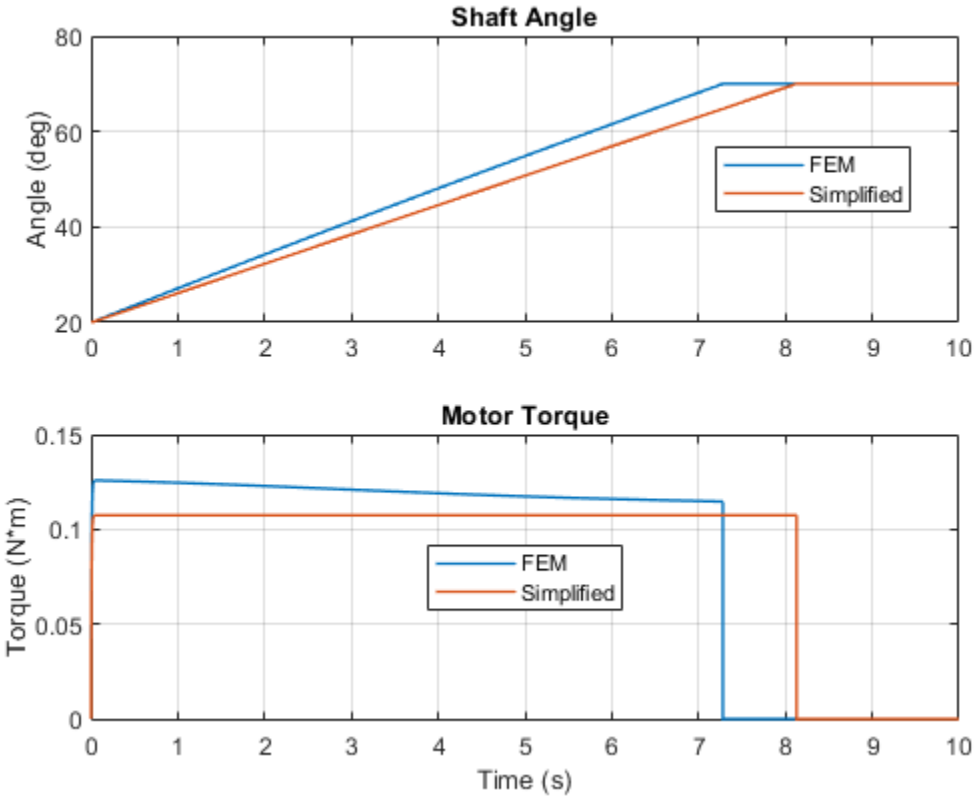



Finally, we use MATLAB to obtain the derivative of the polynomial along those curves. Extracting a lookup table from this surface yields the parameters we need for our motor model.

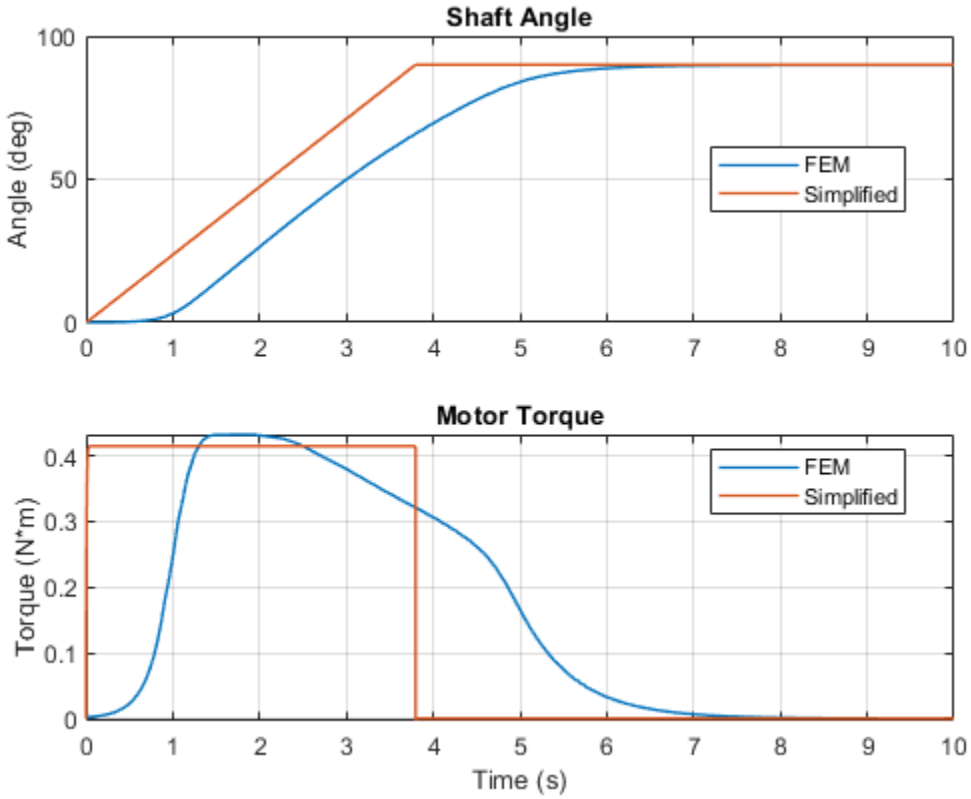



### Simulation Results from Simscape Logging

The plot below shows the behavior of the FEM-Parameterized Rotary Actuator and simplified model built from Simscape™ Foundation Library elements. This test was performed over the range of travel where the finite-element data is linear, so the results are similar.




Performing the test over wider range over the range of travel where the finite-element data is nonlinear shows the effect of our parameterization as two motors behave very differently.



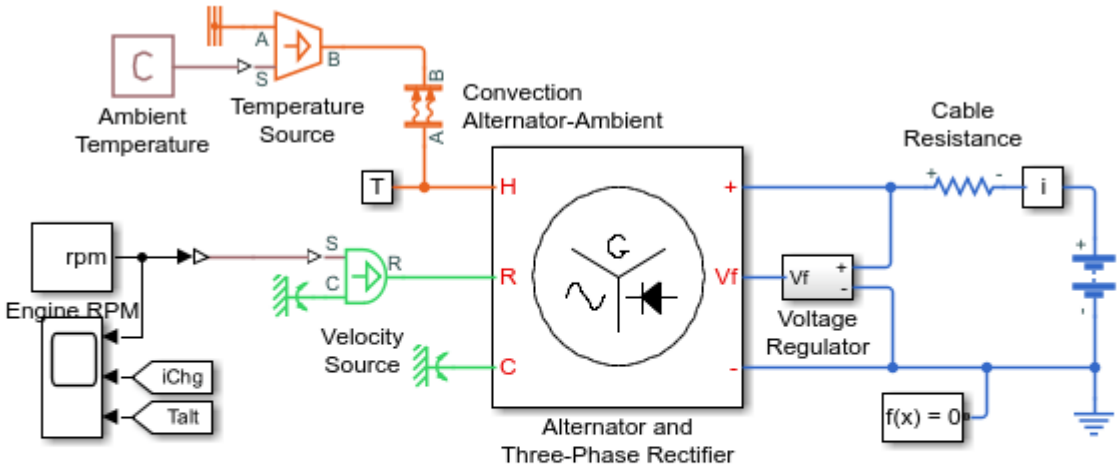

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 50 microseconds.

# Automotive Alternator Charging a Battery

This example shows how alternator behavior can be abstracted to a DC model that simulates efficiently. This test harness first ramps the alternator speed linearly from zero to a typical idle speed of 900 RPM. When the generated voltage is sufficient to overcome the forward voltage drop associated with the rectifier diodes, the battery charging current starts to ramp up. The test harness then ramps up the speed to 5000 RPM, and the alternator has to back off the field voltage to maintain the regulated voltage. The model captures the increase in stator resistance as the alternator heats up, this reducing device performance

### Model



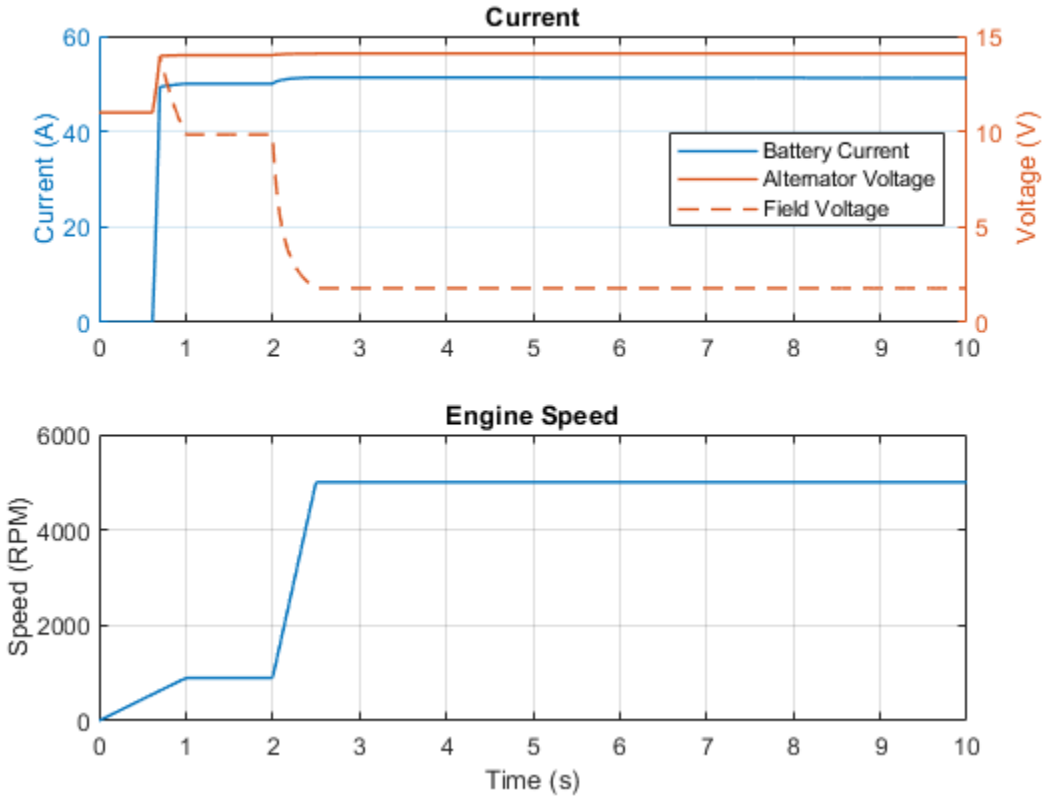

**Automotive Alternator Charging a Battery**

1. Plot charge current applied to battery (see code)
2. Explore simulation results using Simscape Results Explorer
3. Learn more about this example

Copyright 2012-2022 The MathWorks, Inc.

### Simulation Results from Simscape Logging

The plot below shows the behavior of the alternator as the engine speed varies.




### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 50 microseconds.

# Digital Potentiometer Parameterized from Datasheet

This model shows how to model a digital potentiometer such as is used to control audio amplifiers from a digital circuit or microprocessor-controlled system. The model also shows how you can create your own custom blocks in order to extend the Simscape™ Electrical™ library.

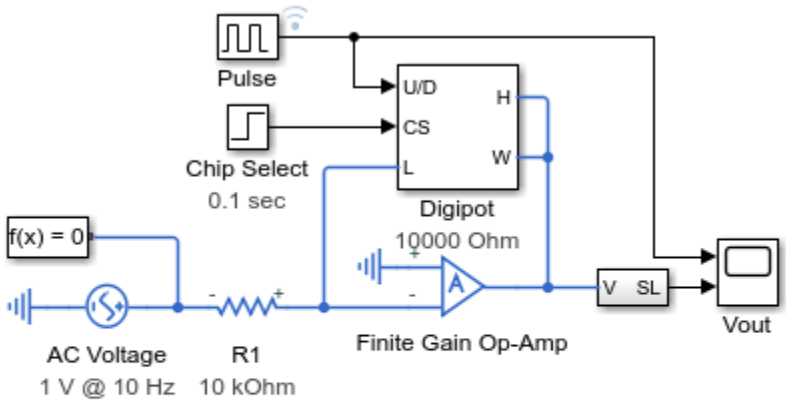
The datasheet specifies an end-to-end resistance R of 10K ohms and a quantization of 16 steps. To look under the Digipot masked subsystem, select the block and type Ctrl-U. The low (L) to wiper (W) resistance is R*N/16, and high (H) to wiper resistance is R*(16-N)/16 where N is the value of an internal counter. The internal counter is enabled by the chip-select (CS) pin, and counts each time the up/down (U/D) pin goes high. The direction of the stepping (up or down) is set by the state of input U/D when digipot is first enabled by the chip select input CS.

In this circuit, the op-amp gain is set by R2/R1 where R2 is the Digipot block resistance. R1 is set to 10K and R2 starts at 10K*8/16 = 5K. Hence the initial gain is 0.5, and the output voltage amplitude is 0.5 volts for a 1 volt peak input sinewave. The Pulse Generator increases the counter N by 1 every second, and the op-amp gain tends to 1 as N approaches 16.

The Digipot block has been implemented as a masked subsystem, and as such is suitable for re-use in other models. Use of triggered and enabled Simulink® subsystems is used to implement the asynchronous counter. To support asynchronous operation, the discrete-time integrator's sample time is set to -1.

Sometimes digipots will only change their resistance when the wiper current is zero. For example, in audio applications this helps prevent audible clicks when the state changes. To implement this, another edge-triggered block can be added after the Pulse Counter block, its trigger signal being driven by a wiper current measurement. Another possible enhancement is to model the wiper resistance by adding a resistor between the wiper port W and the junction between the two variable resistors.

**Model**

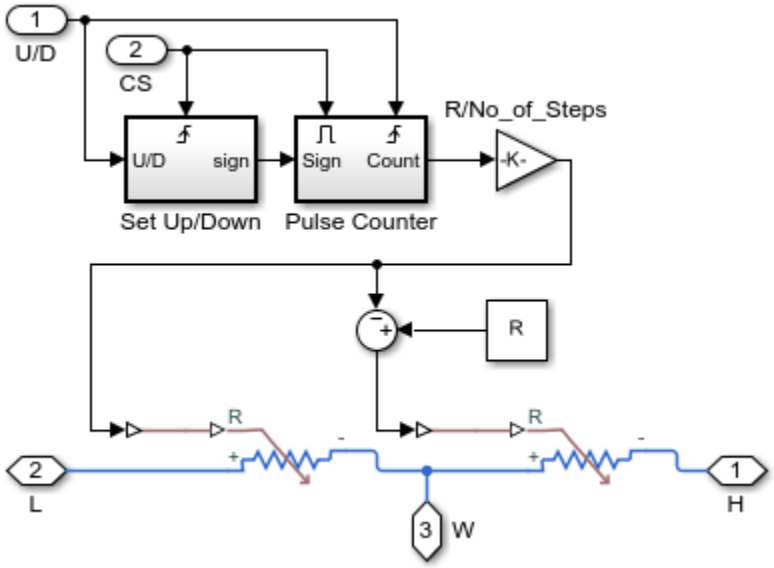



**Digital Potentiometer Parameterized from Datasheet**

1. Plot voltages in circuit (see code)
2. Explore simulation results using Simscape Results Explorer
3. Learn more about this example

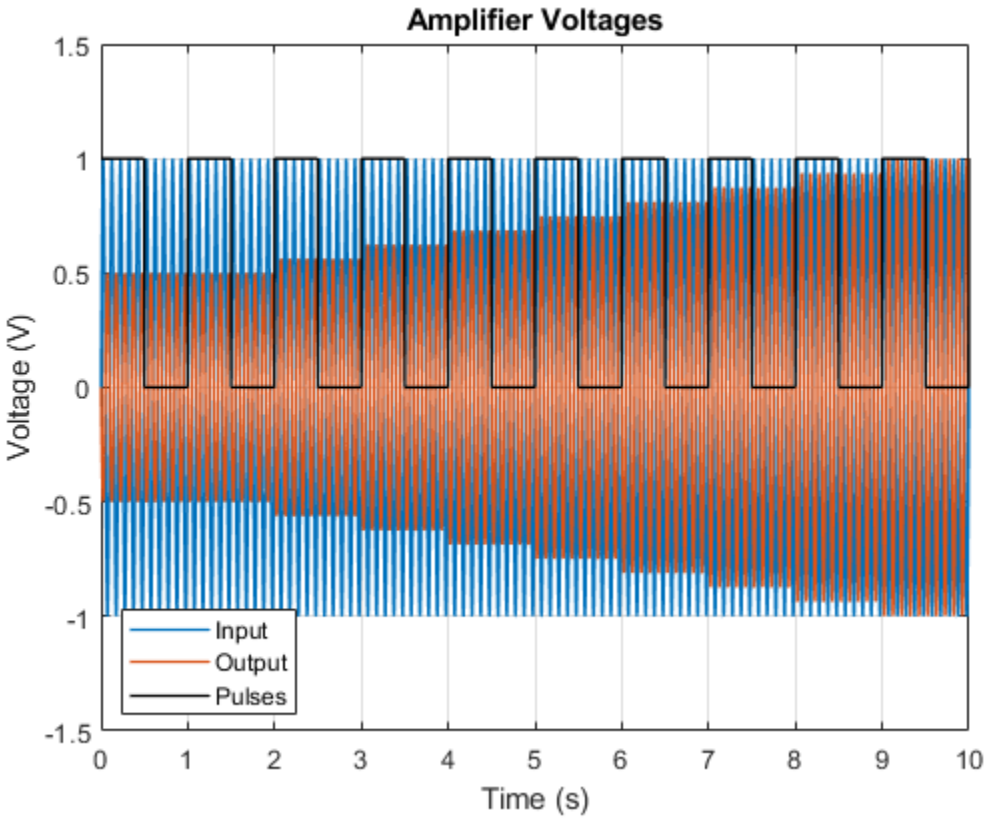
Copyright 2006-2022 The MathWorks, Inc.

### Digipot Subsystem




### Simulation Results from Simscape Logging

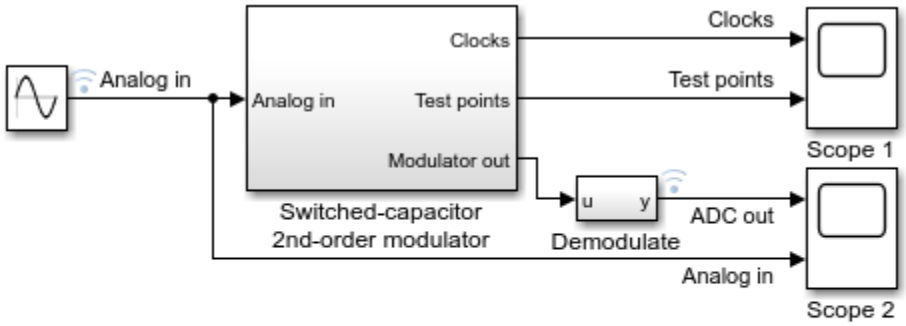
The plot below shows the input and output voltages of an amplifier whose gain is controlled by a digital potentiometer. The gain is adjusted on the rising edges of the pulses applied to the digital potentiometer.


Amplifier Voltages
Voltage (V)
Time (s)
Input
Output
Pulses
1.5
1
0.5
0
-0.5
-1
-1.5
0
1
2
3
4
5
6
7
8
9
10

# Switched Capacitor Analog to Digital Converter

This example shows how a sigma-delta ADC (analog to digital converter) uses sigma-delta modulation to convert an analog input signal into a digital output signal. The analog input to the sigma-delta ADC controls an oscillator that produces pulses of fixed voltage and duration, but with period between pulses being inversely proportional to the analog input. The oscillator pulses are integrated over a fixed time interval to give a digital representation of the analog input signal.

### Model

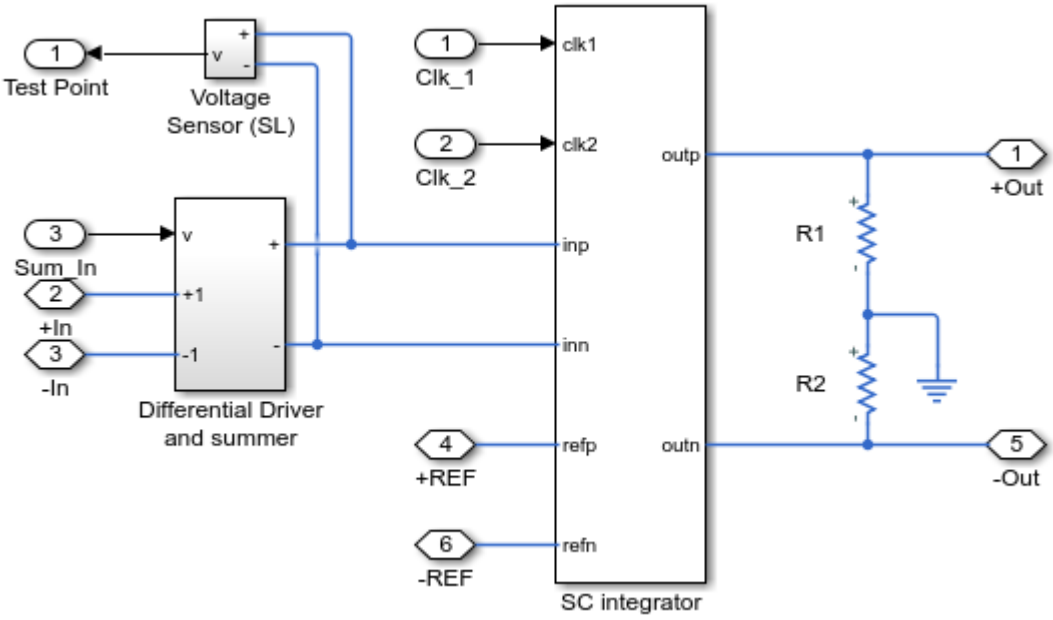



**Switched Capacitor Analog to Digital Converter**

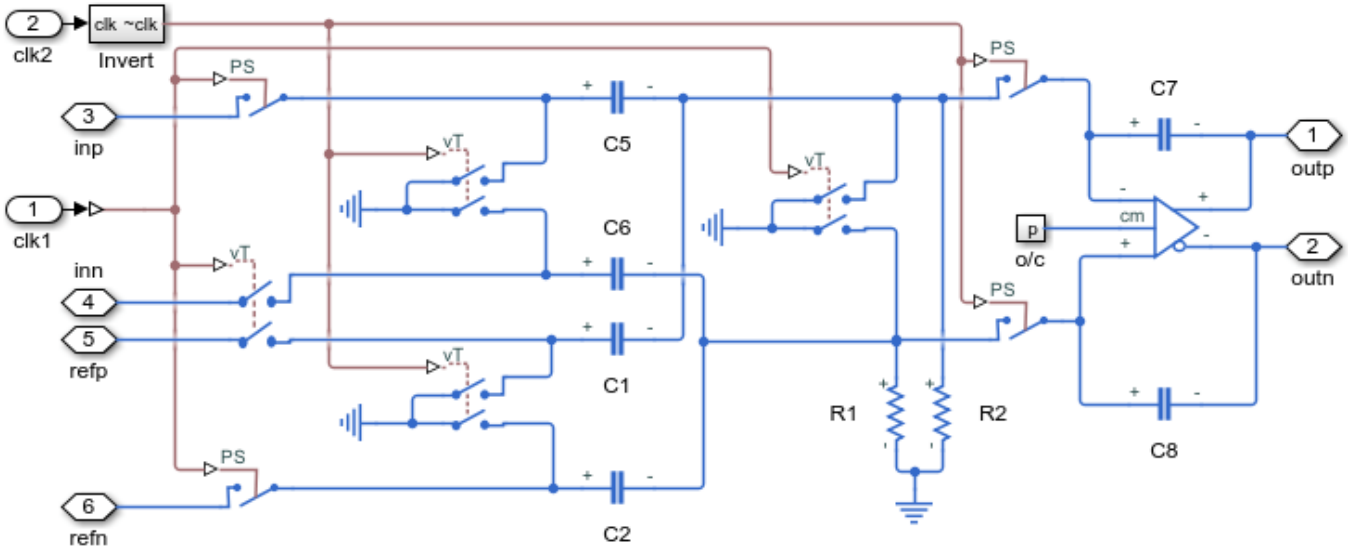
1. Plot voltages at input and output of converter circuit (see code)
2. Explore simulation results using Simscape Results Explorer
3. Learn more about this example

Copyright 2004-2022 The MathWorks, Inc.

### Summer and Integrator 1 Subsystem

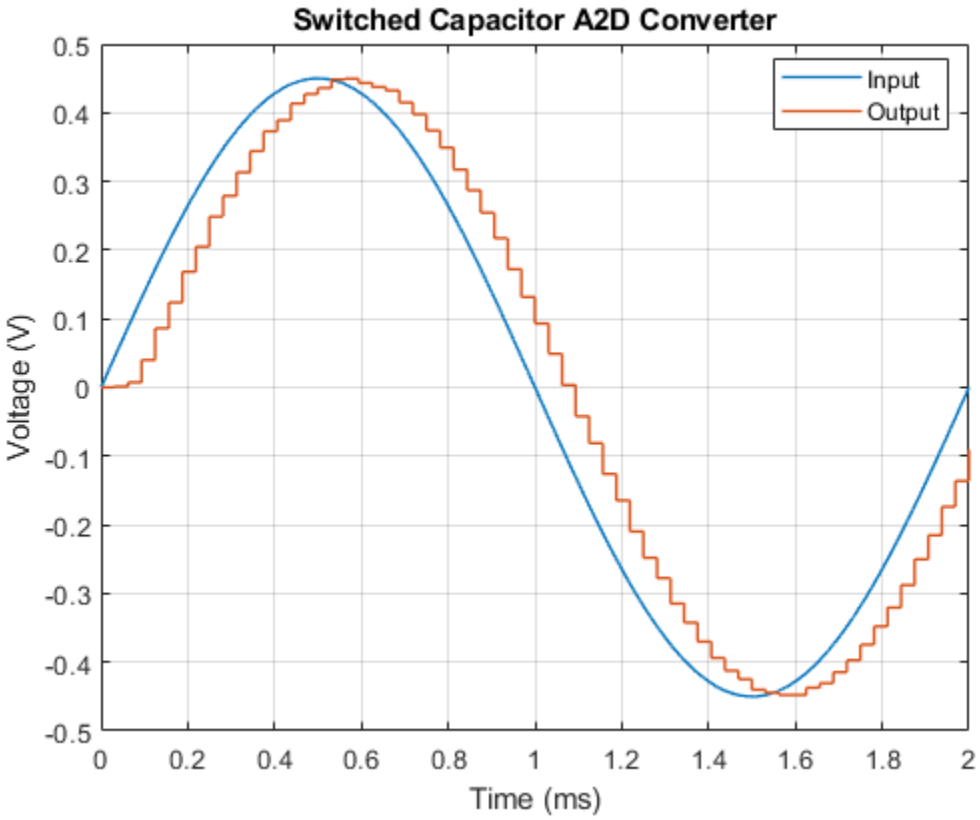


### SC Integrator Subsystem




### Simulation Results from Simscape Logging

The plot below shows the input and output of a switched capacitor analog-to-digital converter circuit model.

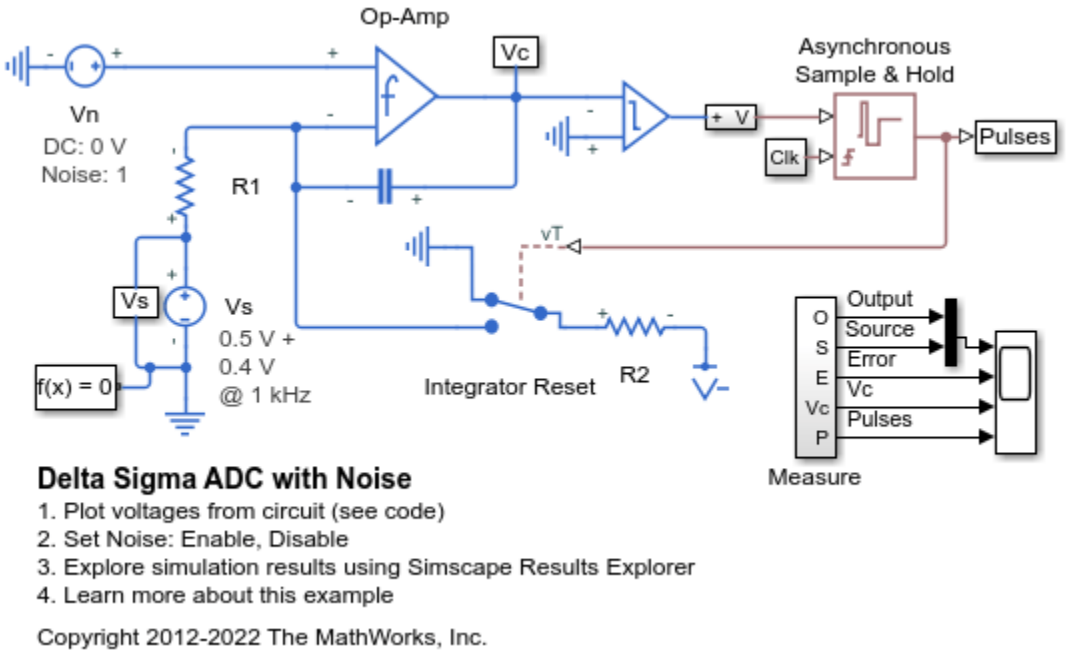

Switched Capacitor A2D Converter
Input
Output
Voltage (V)
Time (ms)
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2

# Delta Sigma ADC with Noise

This example shows a simple implementation of a sigma delta analog-to-digital converter. An input in the range 0 to Vref (=1V) is integrated until it causes the integrator to reset. The time to reset is proportional to the input value. Demodulation of the pulses is performed by a low-pass filter. The Asynchronous Sample & Hold block behaves like an edge-triggered D-type flip-flop, passing input U to output Y only on a rising edge of the clock. This model can be used to explore and understand the effect of op-amp impairments such as equivalent input noise on converter accuracy. To turn off the noise, open block Vn and select 'Disabled' for the noise mode.

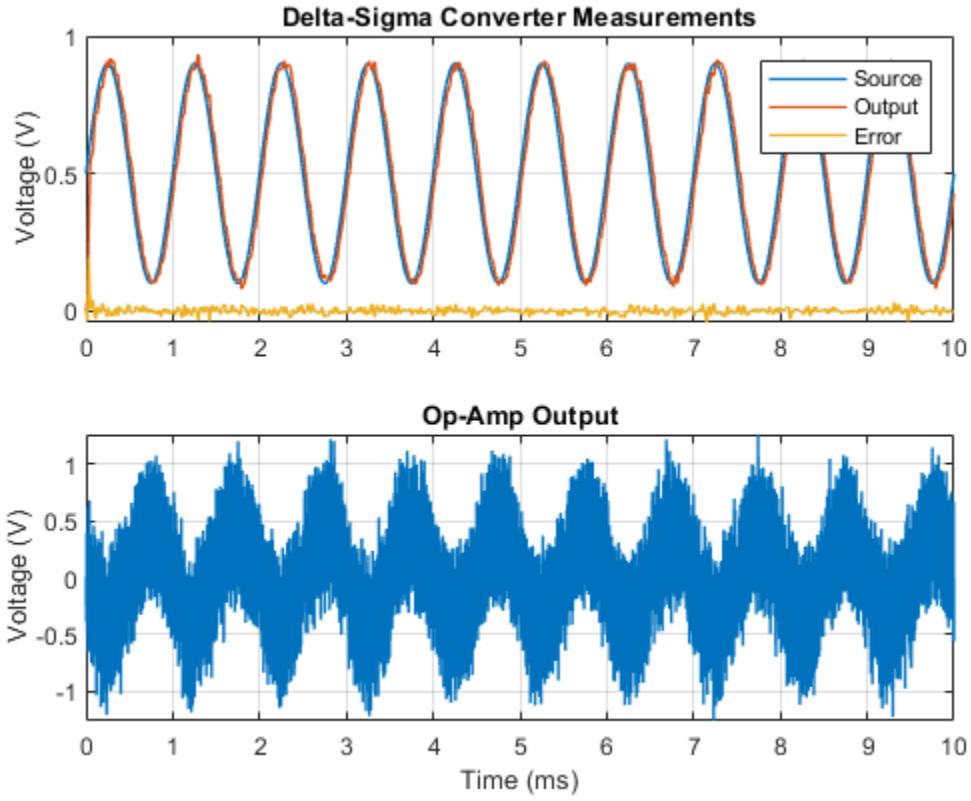
The Solver Configuration block is set to use the fixed-step trapezoidal Simscape™ local solver. This is selected for simulation speed. The simulation step size is parameterized by workspace parameter Tsim, the value of which should be increased from some small value until the point where results are affected. The parameter Tsim also sets the noise sample time in block Vn.

### Model




### Simulation Results from Simscape Logging

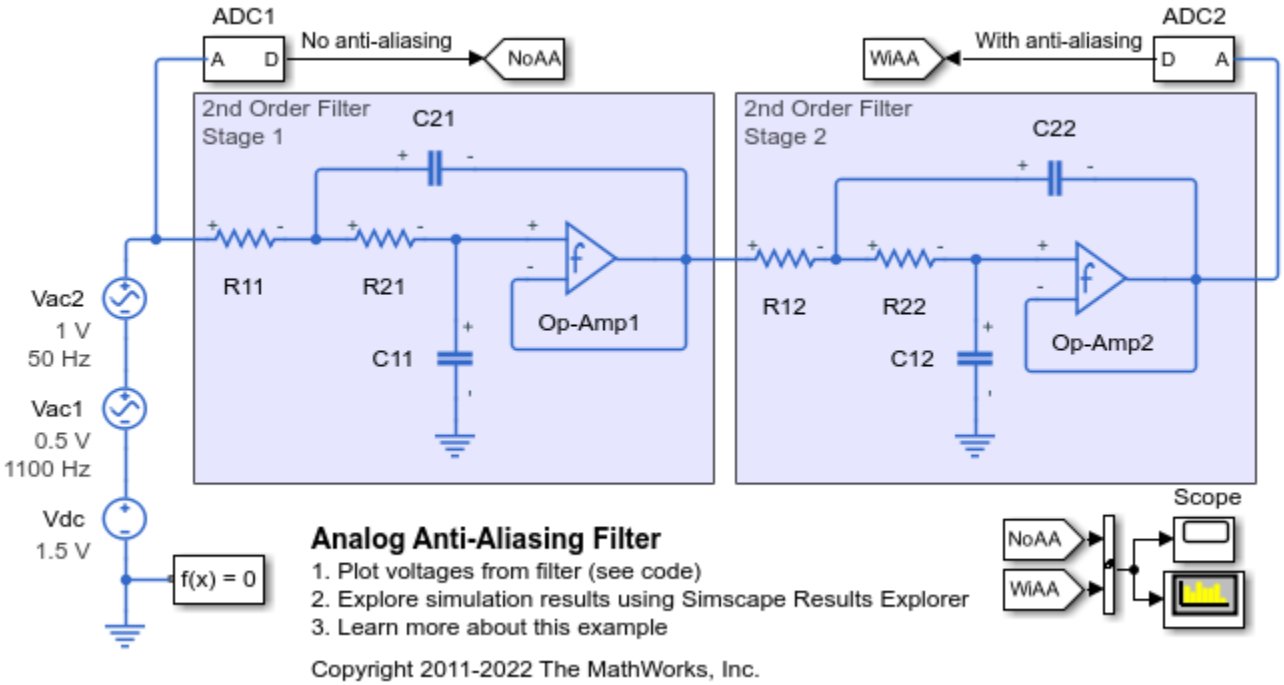
The plot below shows the outputs of the Delta Sigma converter circuit.


Delta-Sigma Converter Measurements
Source
Output
Error
Voltage (V)
1
0.5
0
0
1
2
3
4
5
6
7
8
9
10
Op-Amp Output
Voltage (V)
1
0.5
0
-0.5
-1
0
1
2
3
4
5
6
7
8
9
10
Time (ms)

# Analog Anti-Aliasing Filter

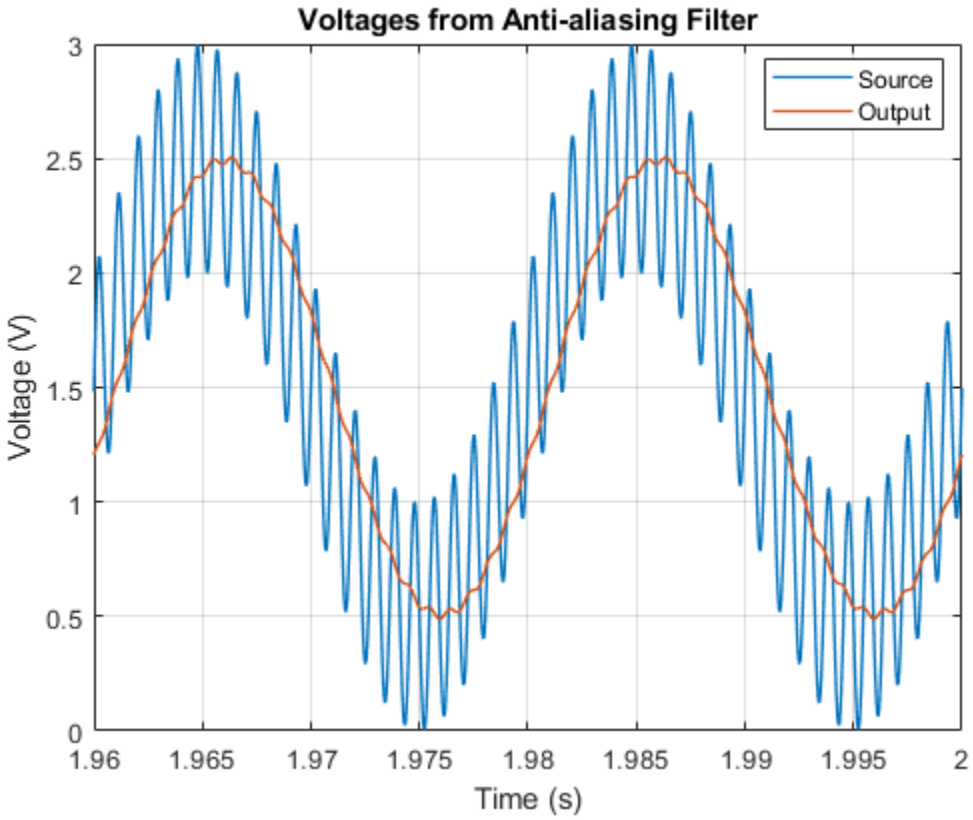
This example shows an analog implementation of an anti-aliasing filter for use with an A-to-D converter. The filter cut-off frequency is set to 500Hz in order to match the A-to-D converter sampling frequency of 1kHz. The test signal incorporates a desired 50Hz sinusoid plus a higher frequency component at 1100Hz that cannot be captured with a 1kHz A-to-D sampling frequency. The scope shows the captured signal without and with anti-aliasing. With the anti-alias filter the 50Hz sine wave amplitude is correctly measured with an amplitude of 1 and corresponding power of 0.5W, i.e., 27dBm for a 1ohm reference load.

### Model



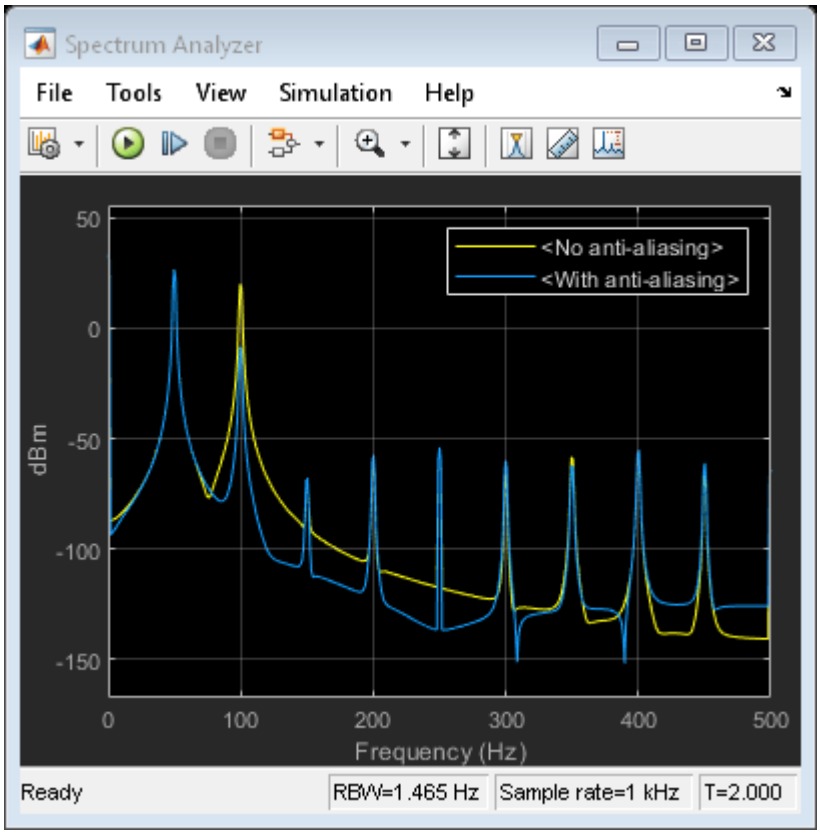

### Simulation Results from Simscape Logging

The plot below shows the input and output voltages of the anti-aliasing filter.




**Simulation Results from Spectrum Analyzer**

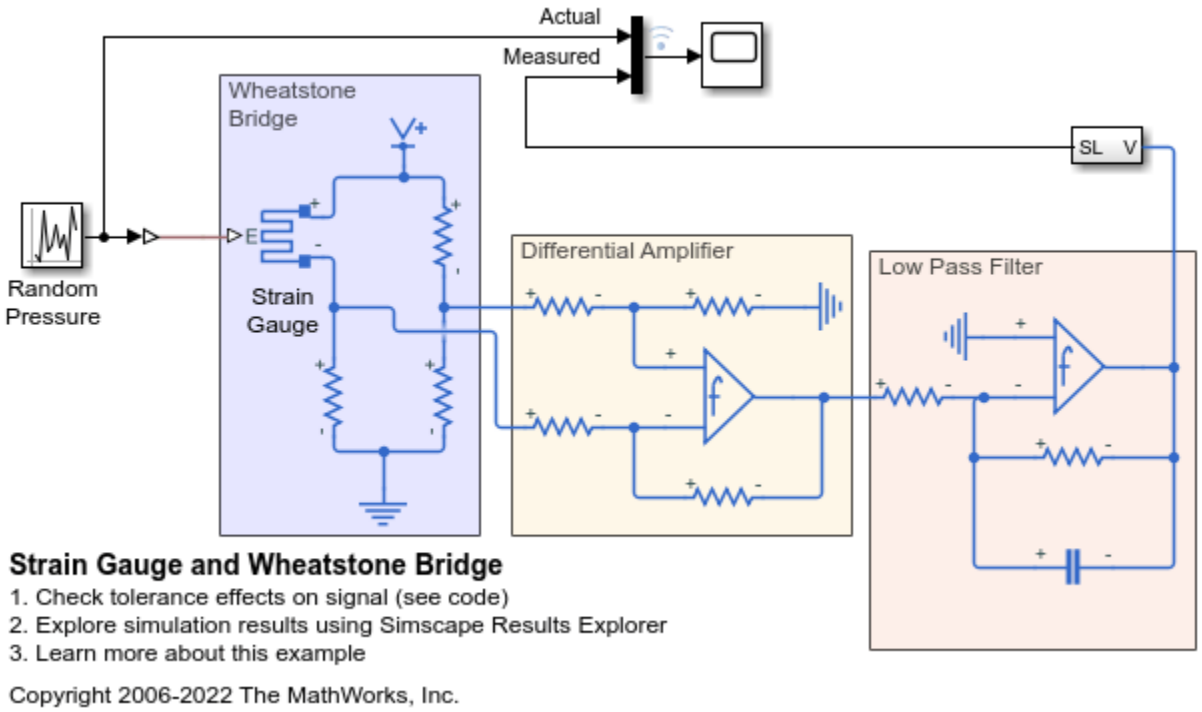
With the anti-alias filter the 50Hz sine wave amplitude is correctly measured with an amplitude of 1 and corresponding power of 0.5W, i.e., 27dBm for a 1ohm reference load.


Spectrum Analyzer
File
Tools
View
Simulation
Help
<No anti-aliasing>
<With anti-aliasing>
50
0
-50
-100
-150
dBm
0
100
200
300
400
500
Frequency (Hz)
Ready
RBW=1.465 Hz
Sample rate=1 kHz
T=2.000

# Strain Gauge and Wheatstone Bridge

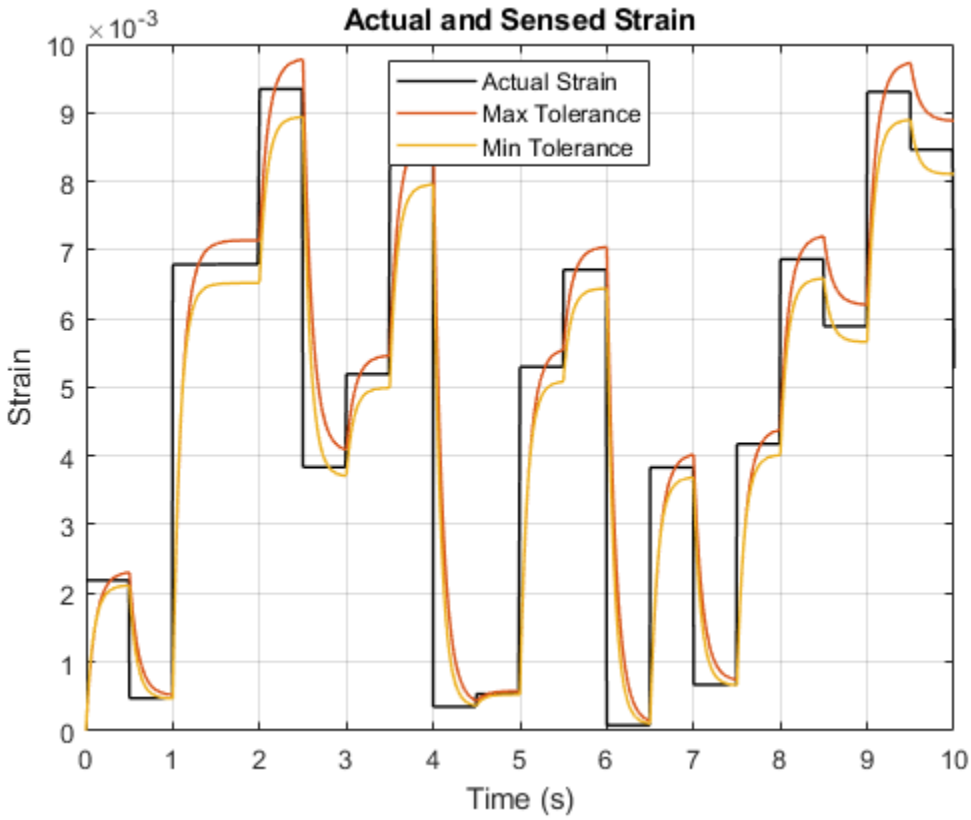
This model shows how to model a strain gauge and measurement amplifier. The strain gauge forms one leg of a Wheatstone bridge, which is connected to a differential amplifier. A second op-amp is then used to both amplify and apply a low-pass filter to the measurement signal. The op-amps are modeled at a system level, with the user specifying parameters such as open-loop bandwidth, gain and maximum slew rate. In this circuit, the dynamics are primarily set by the low-pass filter. The op-amp bandwidth and maximum slew rate have little impact on the step response.

### Model




### Simulation Results from Simscape Logging

This code plots the actual and measured strain from the model of ee_strain_gauge resulting from two tests. The first test sets the resistor and capacitor in the low pass filter to the maximum value of their tolerance ranges. The second test sets the values for those components to the minimum of their tolerance ranges. The plot shows the effects this has on the strain measurement.


Actual and Sensed Strain
×10^-3
Actual Strain
Max Tolerance
Min Tolerance
Strain
Time (s)
0
1
2
3
4
5
6
7
8
9
10

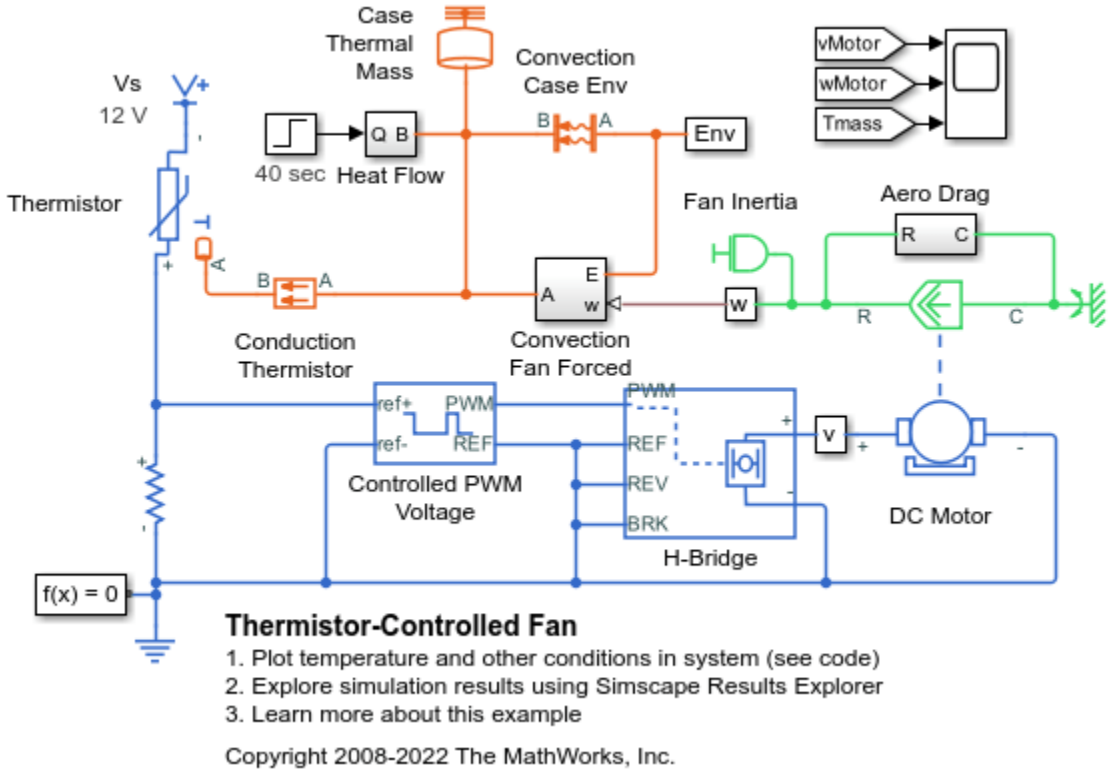
# Thermistor-Controlled Fan

This model shows how fundamental thermal, mechanical and electrical components can be used to model a thermistor-controlled fan. The heat-generating device starts producing 2 watts at time zero, and then at 40 seconds this increases to 20 watts. The thermistor therefore heats up, and its resistance decreases thereby increasing the voltage across the PWM reference pins. This increases the PWM frequency which in turn increases average motor current, and the fan speeds up. The additional fan speed increases the convective cooling of the device, moderating the temperature increase of the device.

This is a system-level model such as might be used for selecting an appropriate thermistor characteristic. The convective heat transfer coefficient used to model nominal cooling (i.e. when the fan isn't running) would typically be determined by experiment. Knowing the temperature difference and having an estimate of the device area, the heat transfer coefficient can be calculated. The coefficient for the fan-assisted cooling could then be estimated by running the motor at maximum RPM, and again measuring the temperature difference. The nominal cooling term just also be taken into account when calculating the fan-cooling coefficient.

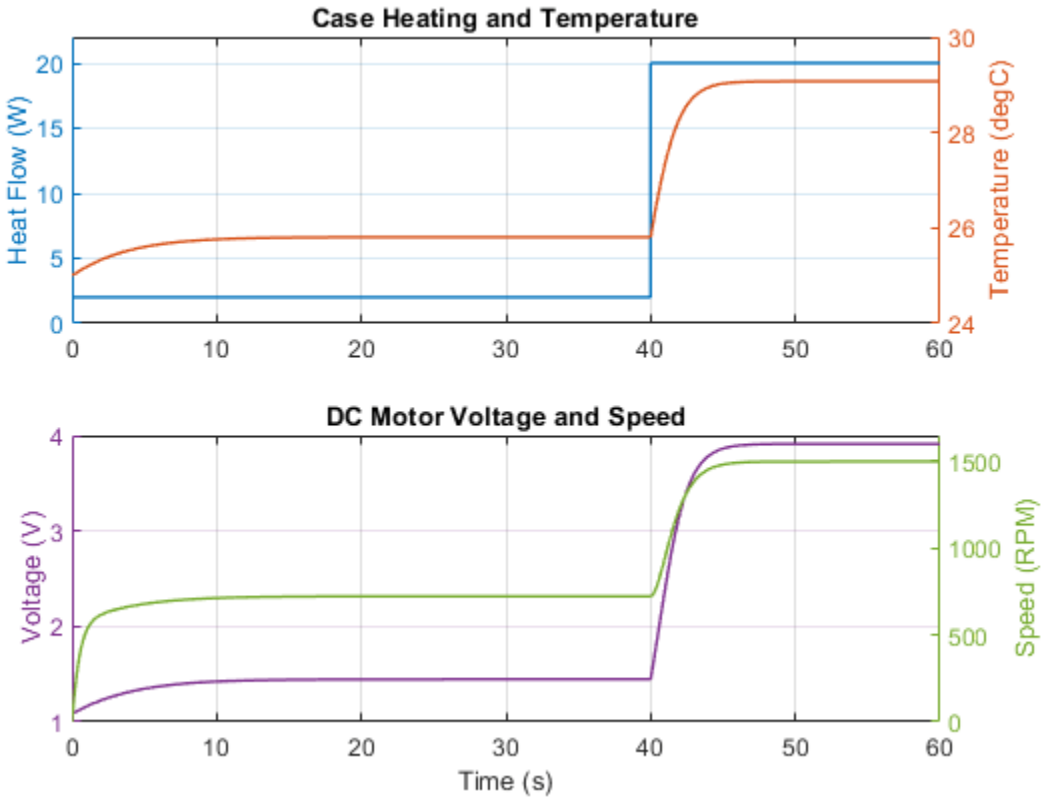
The Controlled PWM Voltage and H-Bridge blocks have two modes of operation, namely Averaged and PWM. As this is a system-level model, and the thermal time constants are measured in seconds, the Averaged mode of operation is used. The PWM mode replicates the PWM control signal which would typically operate at a few kilohertz.

**Model**



### Simulation Results from Simscape Logging

The plot below shows the electrical, mechanical, and thermal behavior of the thermistor controlled motor. As the temperature of the thermistor changes, the voltage applied to the motor changes, which alters the speed of the motor and the convective heat transfer from the case. The system reaches steady state after a short period of time.



# JFET Amplifier and Frequency Response Analysis

This example shows an audio amplifier circuit based on an N-channel JFET. The desired operating point is taken to be Vds=5V, Id=2mA and Vgs=-2V. The manufacturer datasheet gives the JFET forward transfer conductance and output conductance values as 3mS and 50uS. These values are used to populate the mask of the N-Channel JFET block.

The bias resistor values are calculated as follows:

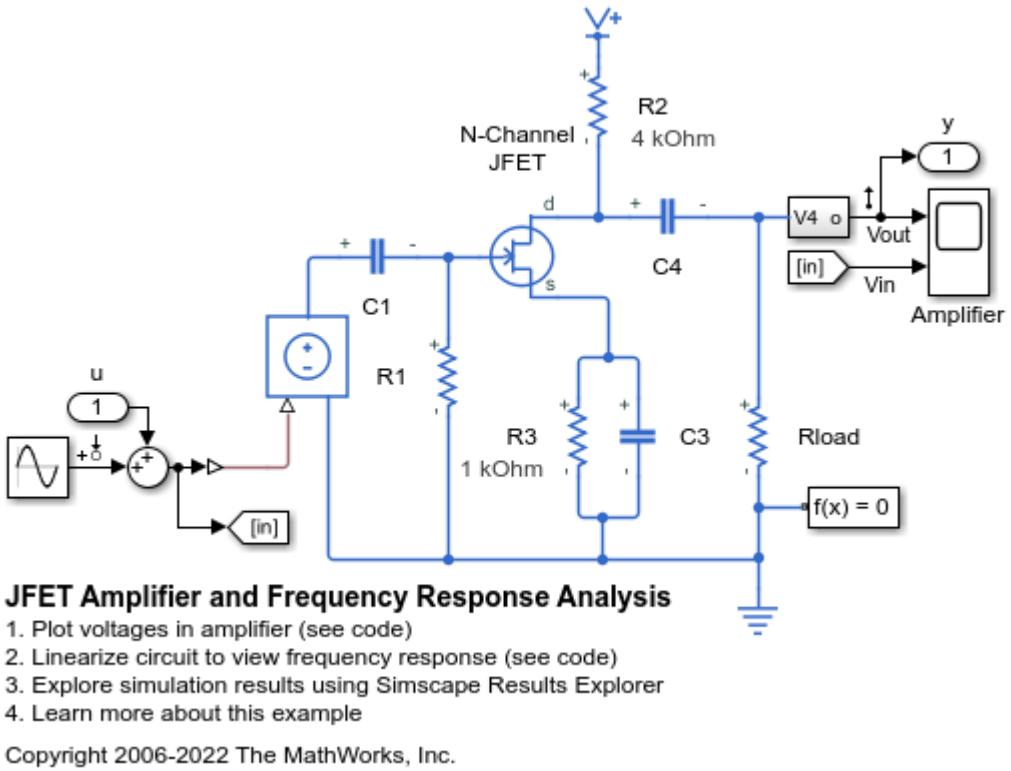
Resistor R1 effectively ties the gate to ground. Hence the voltage across resistor R3 is -Vgs, and must be 2V. Hence R3=-Vgs/Id=2/2e-3 = 1K ohm.

The total voltage across R3, the JFET drain-source connections and R2 must be 15V. Hence the voltage across R2 is 8V if Vds is 5V, and R2=8/2e-3 = 4K ohm.

C3 has to be large enough such that at the lowest frequency of interest (20Hz), it is effectively a short circuit. C4 is chosen so that the loss in gain compared to the mid-band gain is about 6dB.

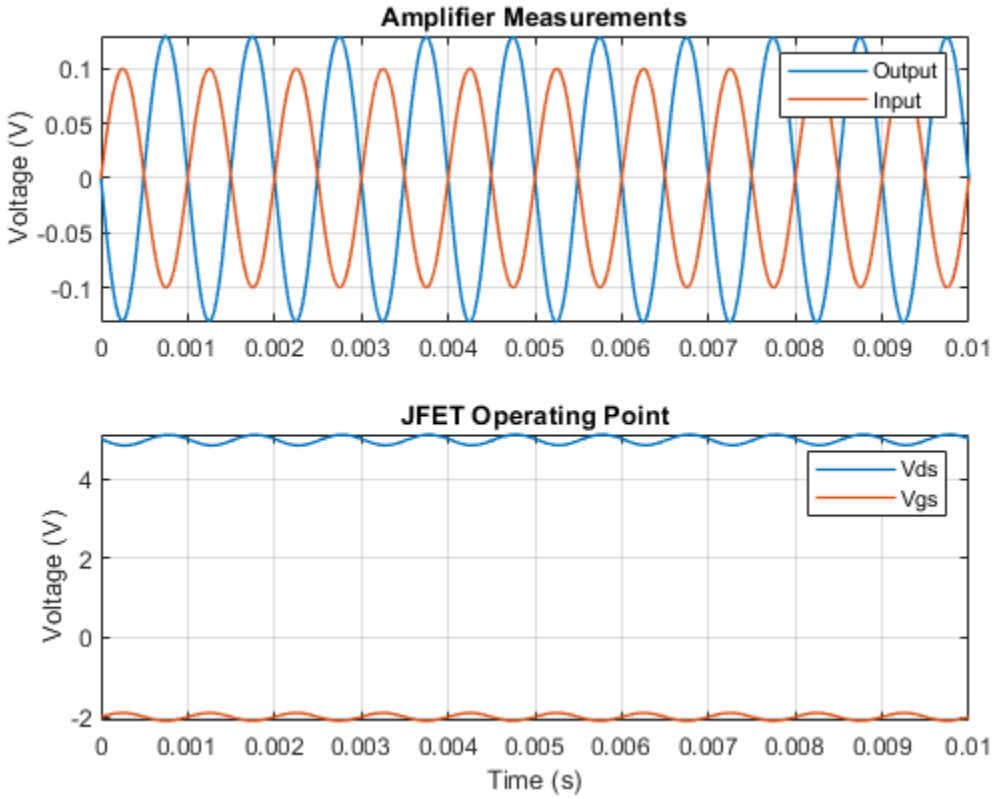
If you have Simulink® Control Design™, then to plot the frequency response, open the model ee_amp_jfet. On the Apps tab, under Control Systems, click Model Linearizer. In the Model Linearizer, on the Linear Analysis tab, in the Linearize section, click Bode. The linearization points were defined in this model by right-clicking on a Simulink line, and selecting Linearization Points.

**Model**

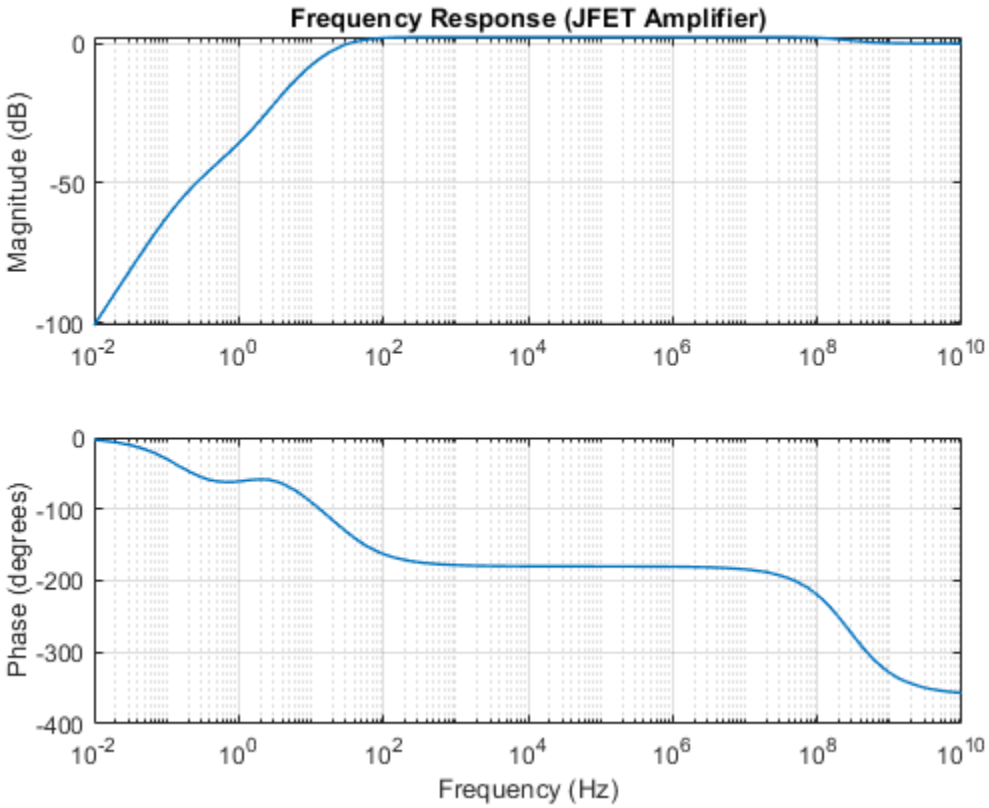


### Simulation Results from Simscape Logging

The plots below shows the voltage from the amplifier circuit. The upper plot shows the gain of the amplifier, and the lower plot shows that the JFET is being tested near its desired operating point.



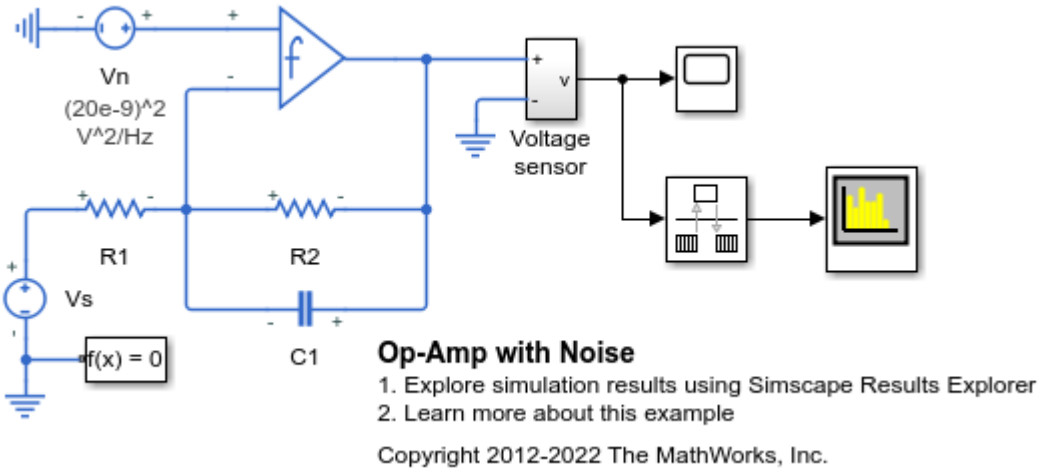
### Frequency Response



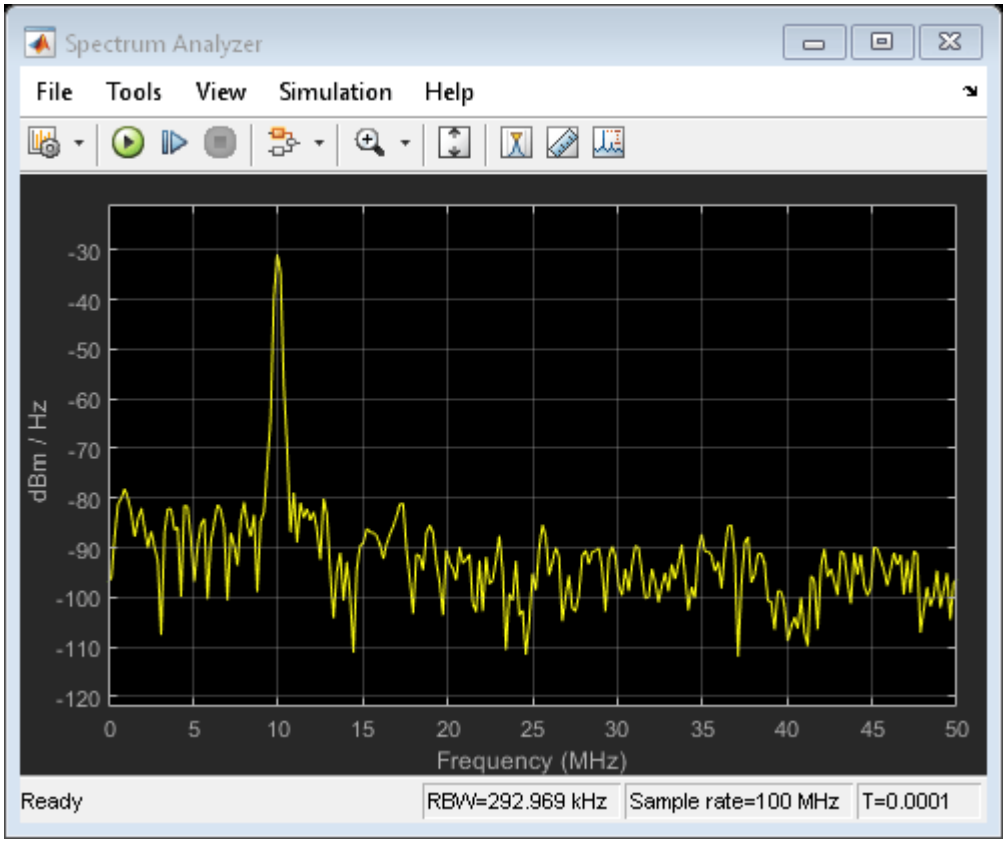
# Op-Amp with Noise

This example shows how noise can be incorporated into an electrical simulation. The circuit models an amplifier with gain 100 and a high-frequency roll off frequency of 10MHz. The op-amp adds noise, and it is assumed that the datasheet specifies an equivalent voltage noise density of 20nV/Hz^0.5. This is implemented using the noise voltage source Vn. Optionally, the thermal noise generated by resistors R1 and R2 can also be included by selecting 'Enabled' for the blocks' noise mode. However, running this model with different combinations of noise sources shows that the main source of noise is the equivalent noise voltage.

**Model**



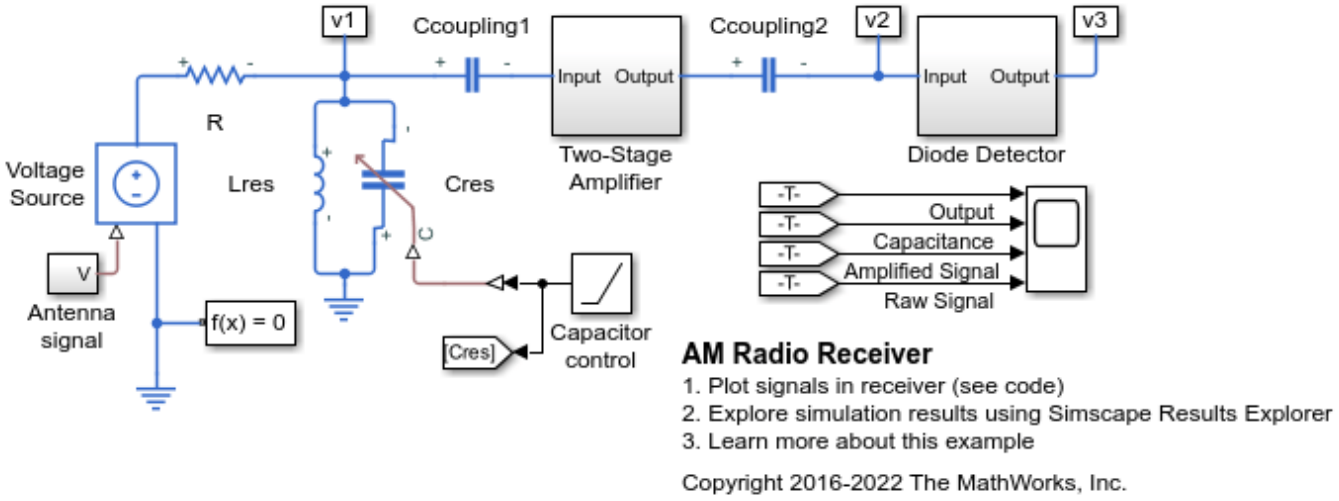
### Simulation Results from Spectrum Analyzers



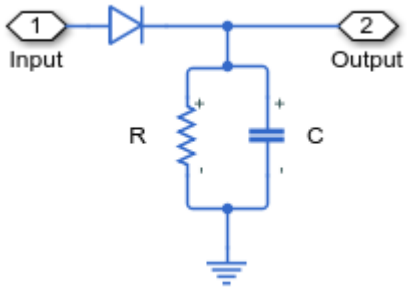
# AM Radio Receiver

This example shows a simplified AM radio receiver. A single tone signal at 2kHz is transmitted with a carrier frequency of 600kHz. The variable capacitor, Cres, in the resonant circuit is used in order to sweep through a certain frequency span. When the resonance passes through 600kHz, the signal is picked up and amplified by a two-stage Class A RF power amplifier. The signal is finally extracted by a diode detector, where it would normally be passed on to an audio amplifier (not included here). The Scope displays the final output, the value of the resonant capacitance, and the received and amplified signals.

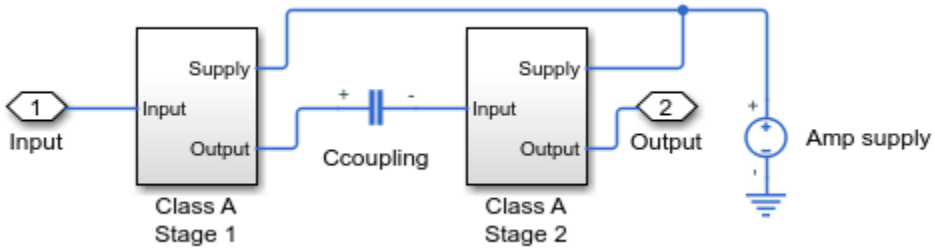
### Model



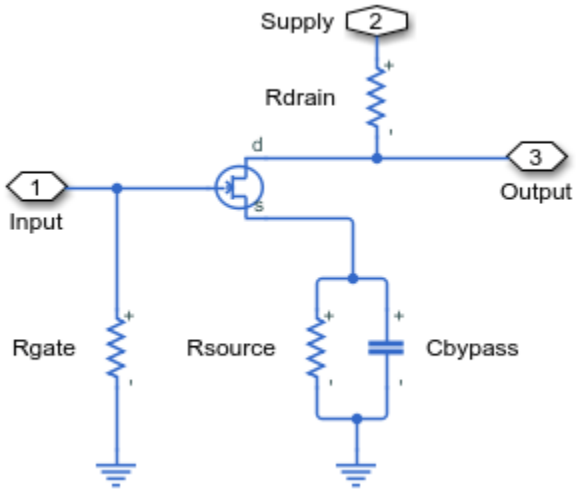

### Diode Detector Subsystem



### Two-Stage Amplifier Subsystem

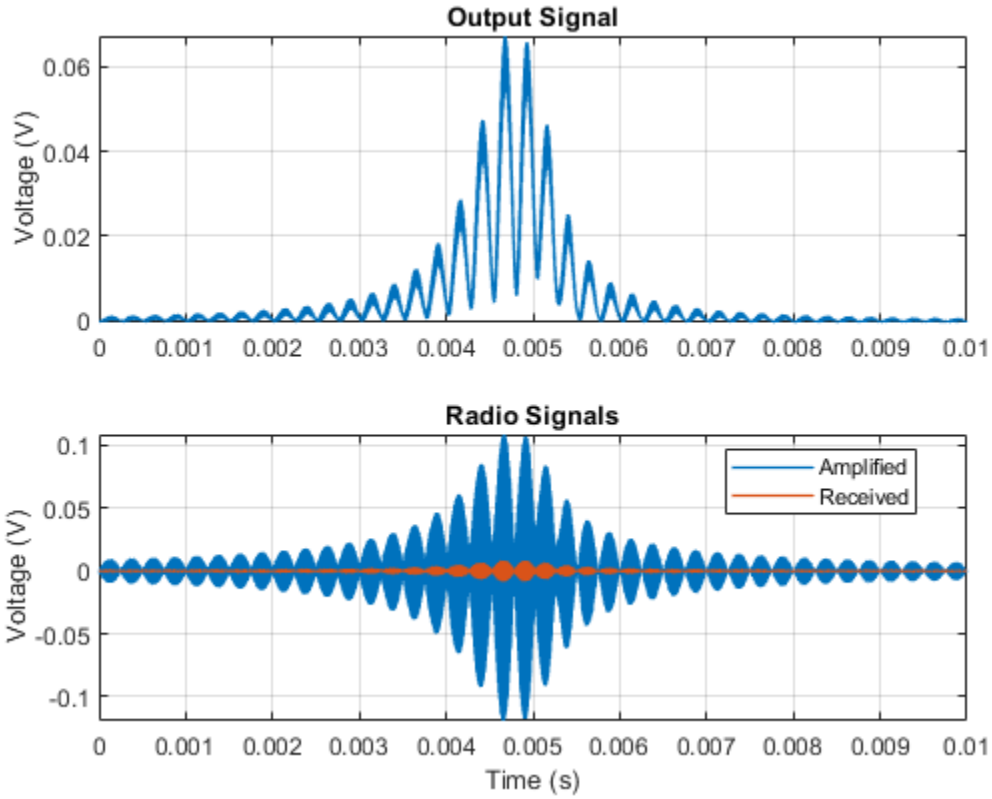



### Class A Stage 1 Subsystem




### Simulation Results from Simscape Logging

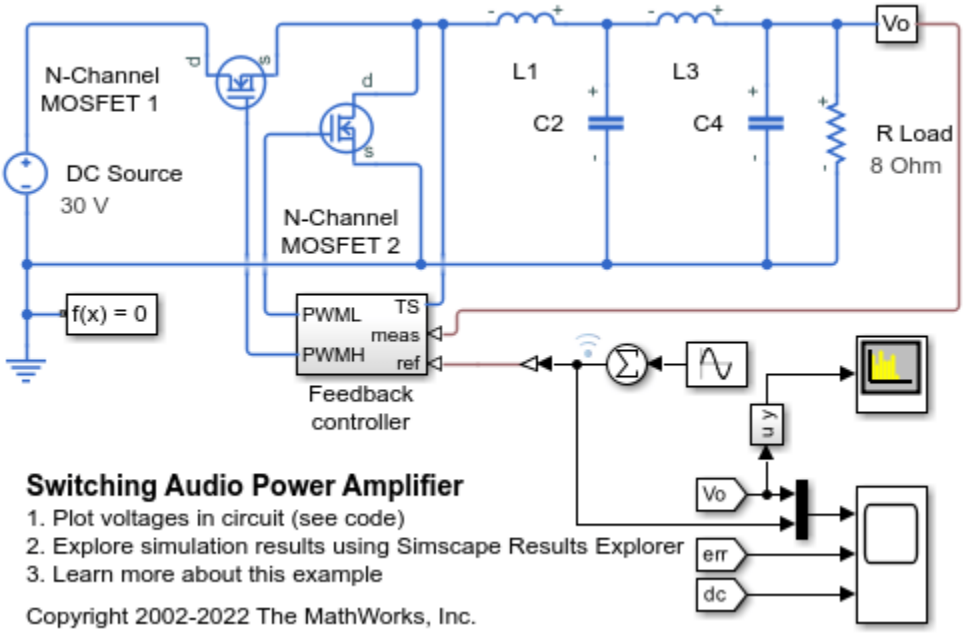
The plots below shows received, amplified, and output signals in the radio receiver. As the resonance in the resonant circuit passes through 600kHz, the signal is picked up and amplified by a two-stage Class A RF power amplifier.


Output Signal
Voltage (V)
0.06
0.04
0.02
0
0
0.001
0.002
0.003
0.004
0.005
0.006
0.007
0.008
0.009
0.01
Radio Signals
Amplified
Received
Voltage (V)
0.1
0.05
0
-0.05
-0.1
Time (s)

# Switching Audio Power Amplifier

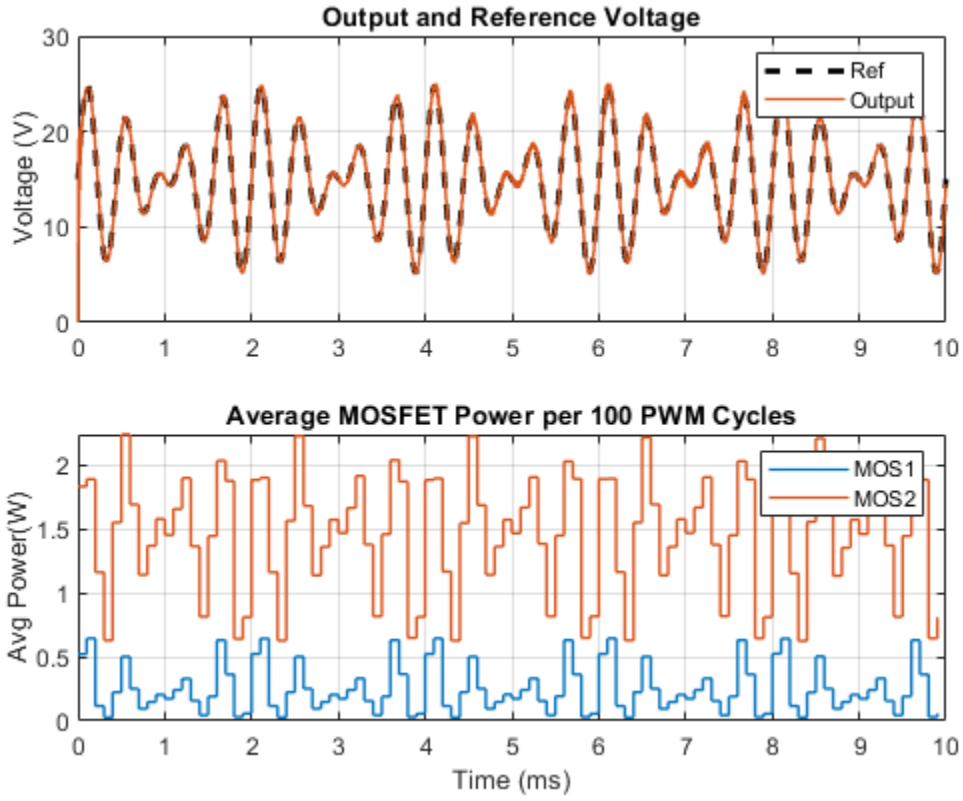
This example shows a switching audio power amplifier circuit. Switching audio power amplifiers are more efficient than conventional power amplifiers as switching devices are operating only in the fully-on and fully-off states. This audio power amplifier uses a 1MHz switching frequency and has a PI feedback controller to ensure that output voltage tracks the 2kHz and 2.5kHz sine wave inputs. The power spectrum is plotted in the Spectrum Analyzer, and can be used to inform selection of controller and filter parameters.

### Model



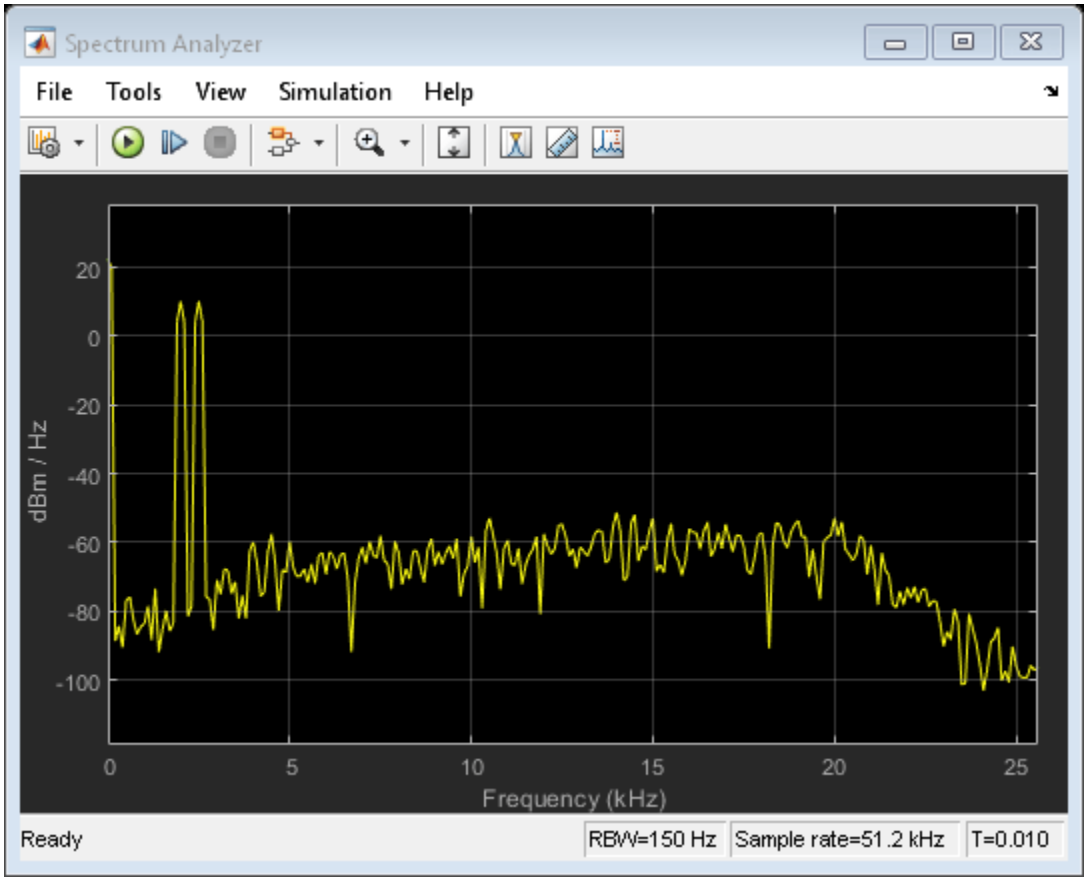

### Simulation Results from Simscape Logging

The plot below shows the output voltage as compared to the reference voltage. It also shows the dissipated power of the two MOSFETS averaged over 100 PWM cycles.




### Simulation Results from Scopes

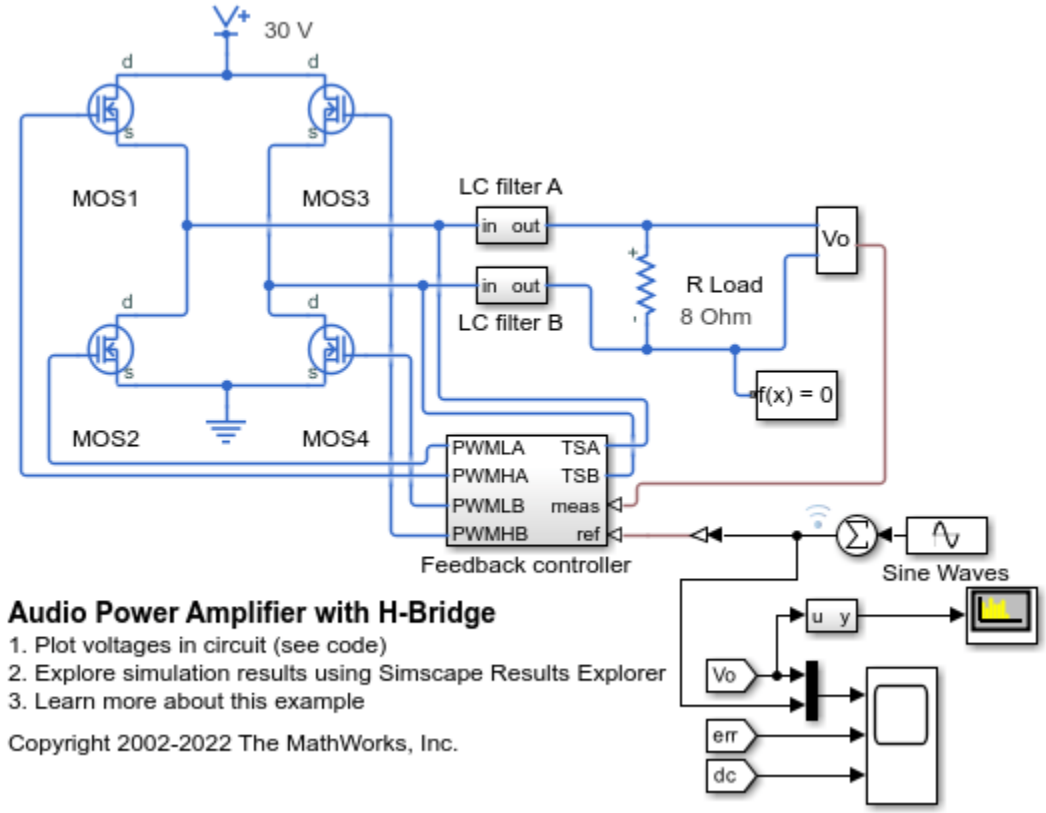
The power spectrum plotted in the Spectrum Analyzer shows peaks at 2kHz and 2.5kHz, the two components of the reference voltage.


Spectrum Analyzer
File
Tools
View
Simulation
Help
20
0
-20
-40
-60
-80
-100
dBm / Hz
0
5
10
15
20
25
Frequency (kHz)
Ready
RBW=150 Hz
Sample rate=51.2 kHz
T=0.010

# Audio Power Amplifier with H-Bridge

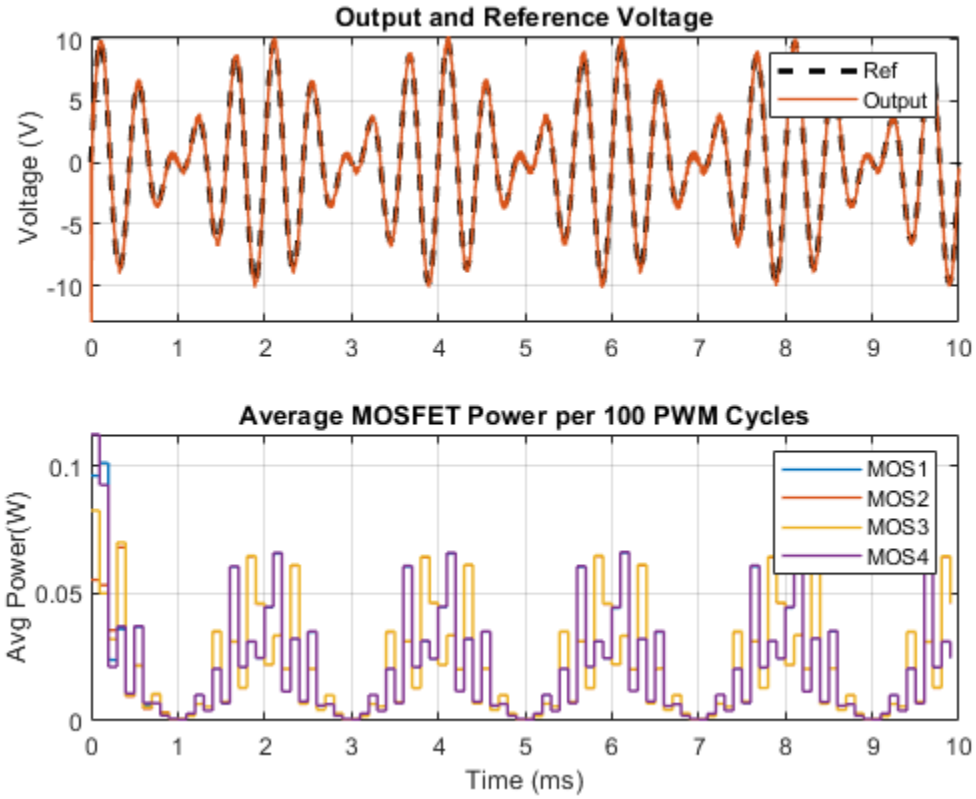
This example shows an H-bridge configuration switching audio power amplifier circuit. Switching audio power amplifiers are more efficient than conventional power amplifiers as switching devices are operating only in the fully-on and fully-off states. This audio power amplifier uses a 1MHz switching frequency and has a PI feedback controller to ensure that output voltage tracks the 2kHz and 2.5kHz sine wave inputs. The power spectrum is plotted in the Spectrum Analyzer, and can be used to inform selection of controller and filter parameters.

### Model



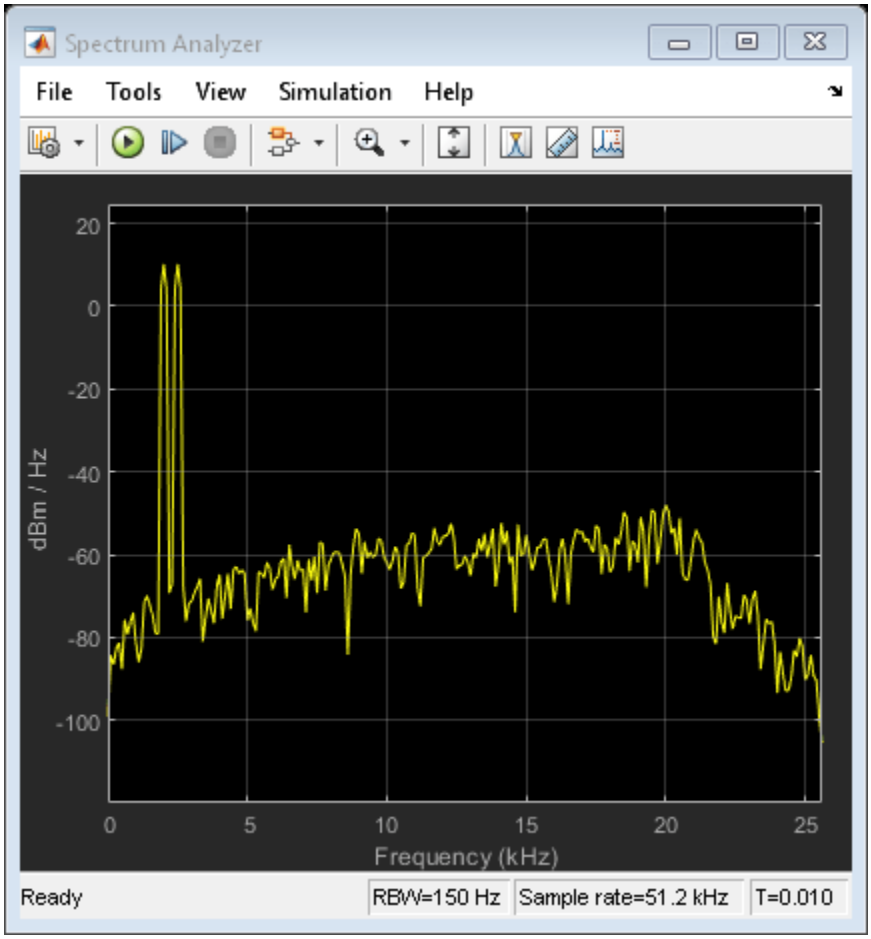

### Simulation Results from Simscape Logging

The plot below shows the output voltage as compared to the reference voltage. It also shows the dissipated power for the four MOSFETS averaged over 100 PWM cycles.




### Simulation Results from Scopes

The power spectrum plotted in the Spectrum Analyzer shows peaks at 2kHz and 2.5kHz, the two components of the reference voltage.


Spectrum Analyzer
File
Tools
View
Simulation
Help
20
0
-20
-40
-60
-80
-100
dBm / Hz
0
5
10
15
20
25
Frequency (kHz)
Ready
RBW=150 Hz
Sample rate=51.2 kHz
T=0.010

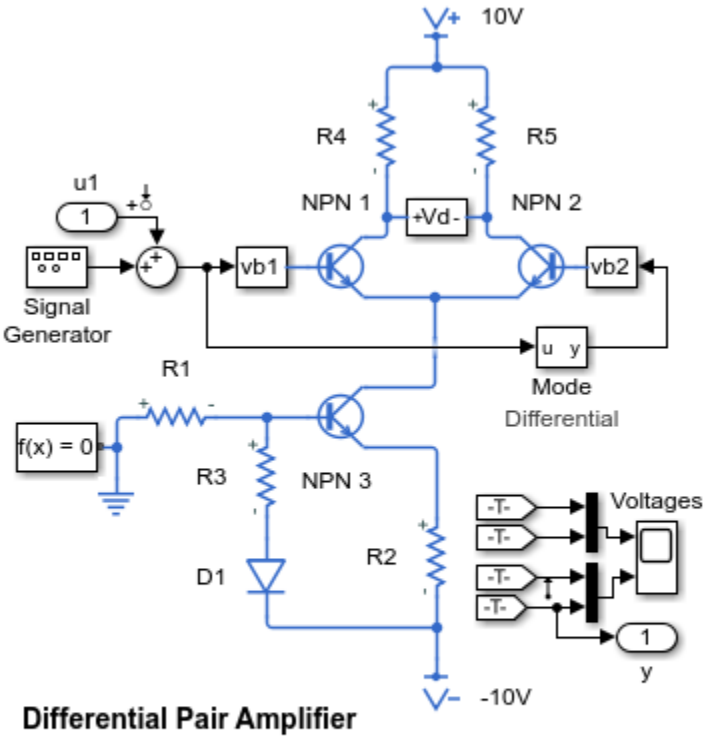
# Differential Pair Amplifier

This example shows a differential pair amplifier circuit. The circuit can be used to explore the properties of a differential pair amplifier. The model can be tested using differential and common-mode inputs. The balanced output has zero gain in common-mode provided that the two transistors have identical properties.

Transistor NPN 3 acts as a constant current source that helps stabilize differential-mode gain. The combination of R1, R3 and D1 sets the base voltage to about 1V above the negative power rail. The base-emitter saturation voltage is 0.4V, and hence the voltage across R2 is 0.6V giving an emitter current of 0.6/220=2.7mA. D1 provides temperature compensation, the forward voltage variation with temperature being similar to that of the base-emitter junction. Transistor component values are typical for a BC107, and the diode component values are typical for an IN4148.

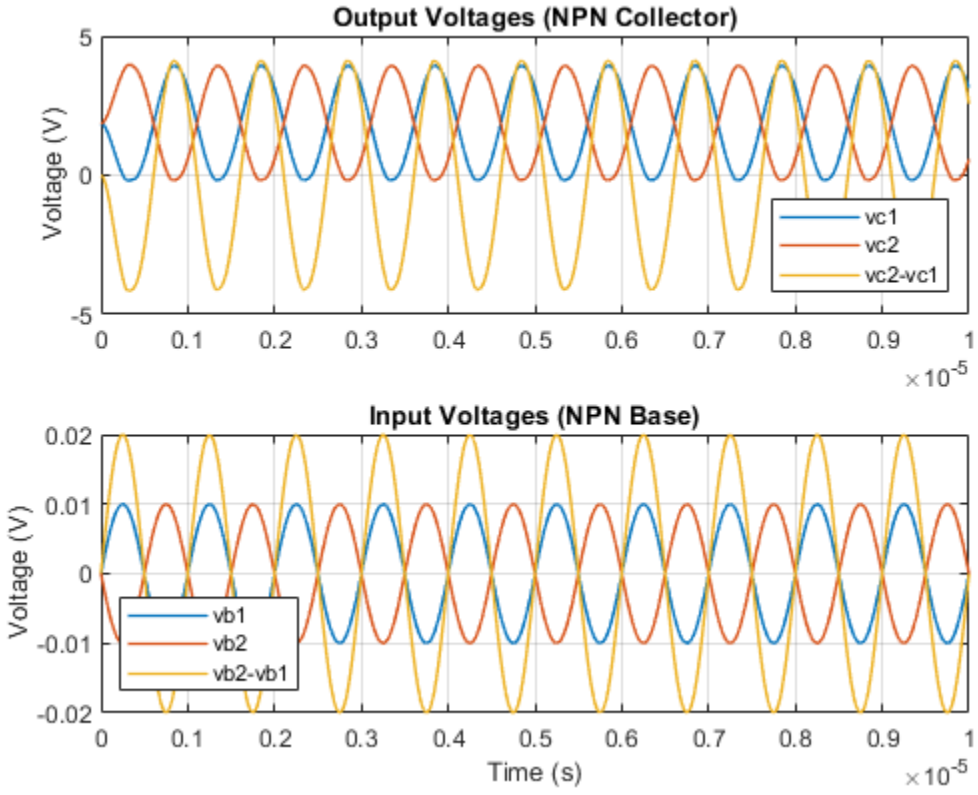
This model can be used to obtain the frequency response of the system. The Solver Configuration block option "Start simulation from steady state" is enabled to ensure that the model is linearized about its nominal operating point. MATLAB® command linmod can be used to linearize the model. If you have Simulink® Control Design™, open the model ee_amp_diff_pair. On the Apps tab, under Control Systems, click Model Linearizer. In the Model Linearizer, on the Linear Analysis tab, in the Linearize section, click Bode. The linearization points are defined by right-clicking on a Simulink line, and selecting Linearization Points. If you look at the Sensor block, you will see the linearization output symbol at the output of the PS-Simulink block. Similarly under the Source block you will see a linearization input point symbol on the output of the Sine Wave block.

### Model

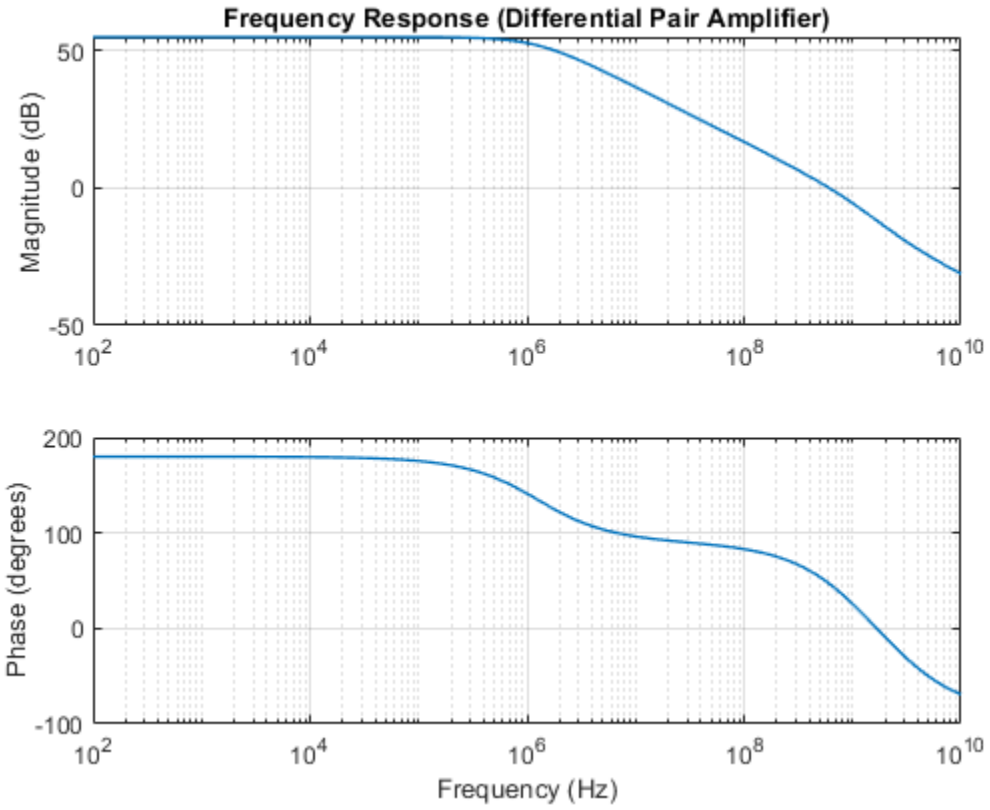



### Simulation Results from Simscape Logging

The plot below shows the output and input of the differential pair amplifier model. The difference in the base voltages (vb2-vb1) is amplified by more than a factor of 40. The output is measured as the difference between the two collector outputs (vc2-vc1).


Output Voltages (NPN Collector)
Voltage (V)
5
0
-5
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
×10^-5
vc1
vc2
vc2-vc1
Input Voltages (NPN Base)
Voltage (V)
0.02
0.01
0
-0.01
-0.02
vb1
vb2
vb2-vb1
Time (s)
×10^-5

**Frequency Response**



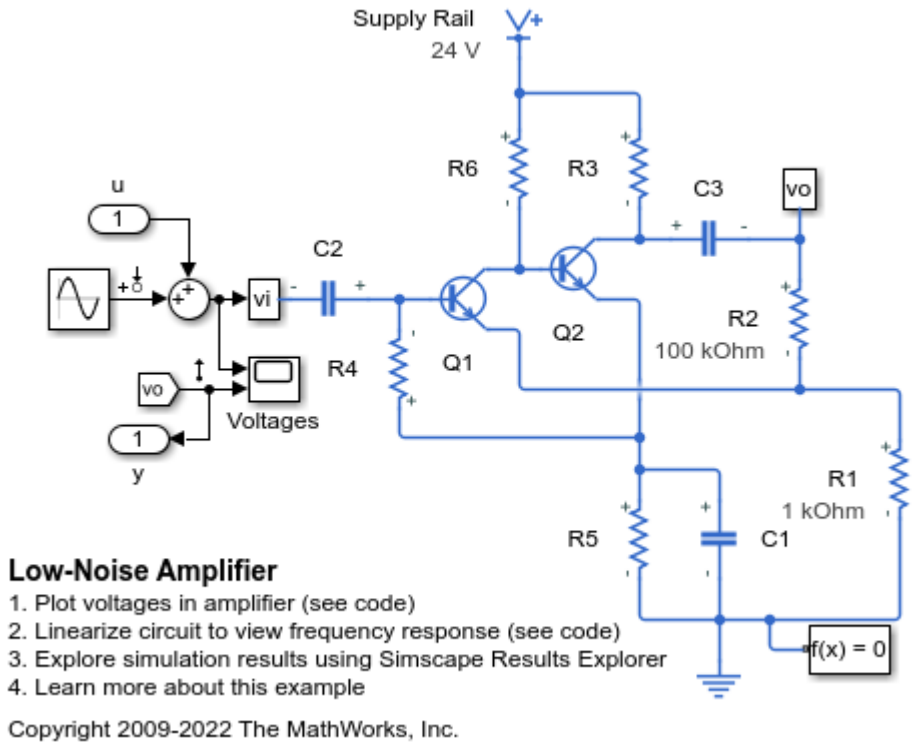
# Low-Noise Amplifier

This example shows a typical low-noise audio amplifier circuit. Resistor R2 provides negative feedback to stabilize the overall amplifier gain, making it independent of transistor open-loop forward transfer gain. Circuit gain is approximately defined by (R1+R2)/R1 = 101. The simulation shows that it takes the circuit a few seconds to settle to its steady operating point, and that the output is initially clipped. Input u and output y are included to support linearization.

This model can be used to obtain the frequency response of the system. The Solver Configuration block option "Start simulation from steady state" should be set to ensure that the model is linearized about its nominal operating point. MATLAB® command linmod can be used to linearize the model.

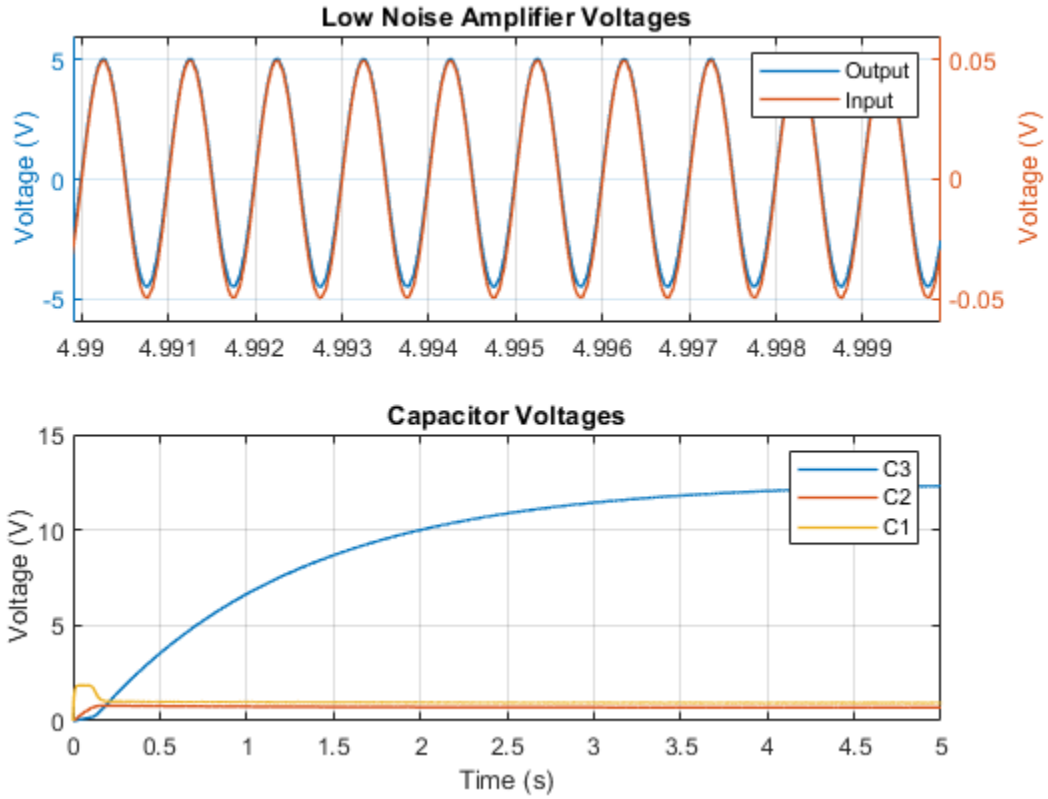
If you have Simulink Control Design, open the model ee_amp_low_noise. On the Apps tab, under Control Systems, click Model Linearizer. In the Model Linearizer, on the Linear Analysis tab, in the Linearize section, click Bode. The linearization points are defined by right-clicking on a Simulink line, and selecting Linearization Points.

### Model

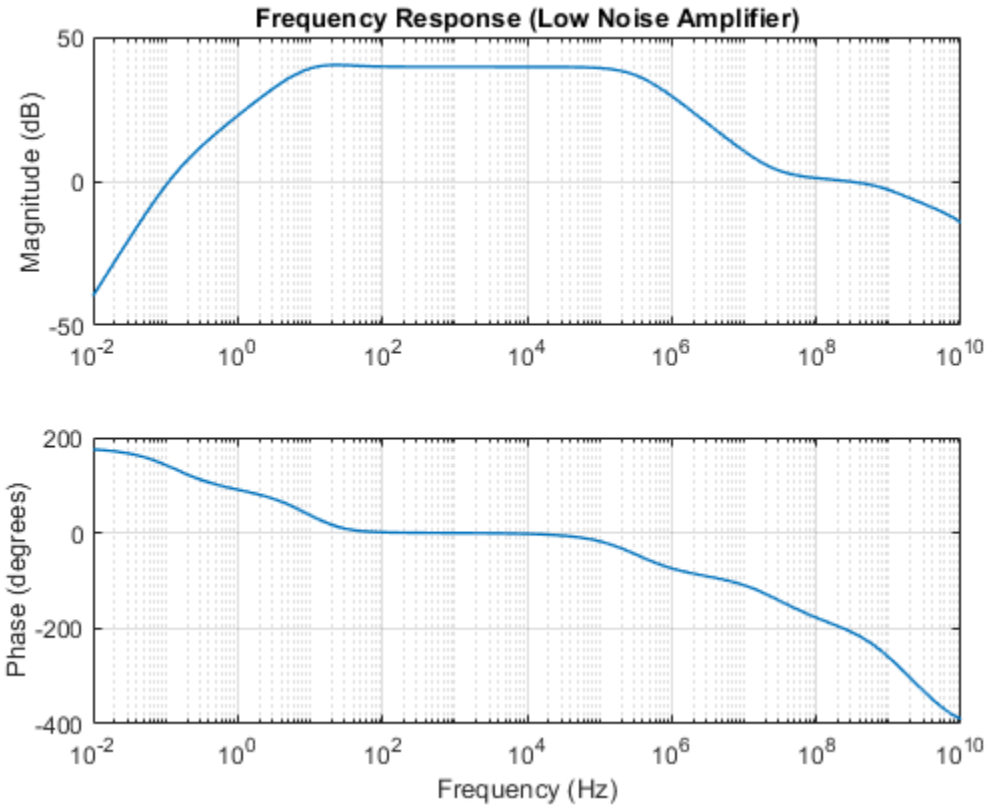



### Simulation Results from Simscape Logging

This plot shows the amplifier and capacitor voltages. The gain is about 100. For visual comparison, the amplifier subplot shows only the last 10 periods of the simulation. It takes a few seconds for the circuit to settle to its steady operating point due to the charging of the capacitors, shown in the capacitor subplot.


Low Noise Amplifier Voltages
Voltage (V)
Voltage (V)
Output
Input
5
0
-5
0.05
0
-0.05
4.99
4.991
4.992
4.993
4.994
4.995
4.996
4.997
4.998
4.999
Capacitor Voltages
Voltage (V)
Time (s)
C3
C2
C1
15
10
5
0
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5

### Frequency Response

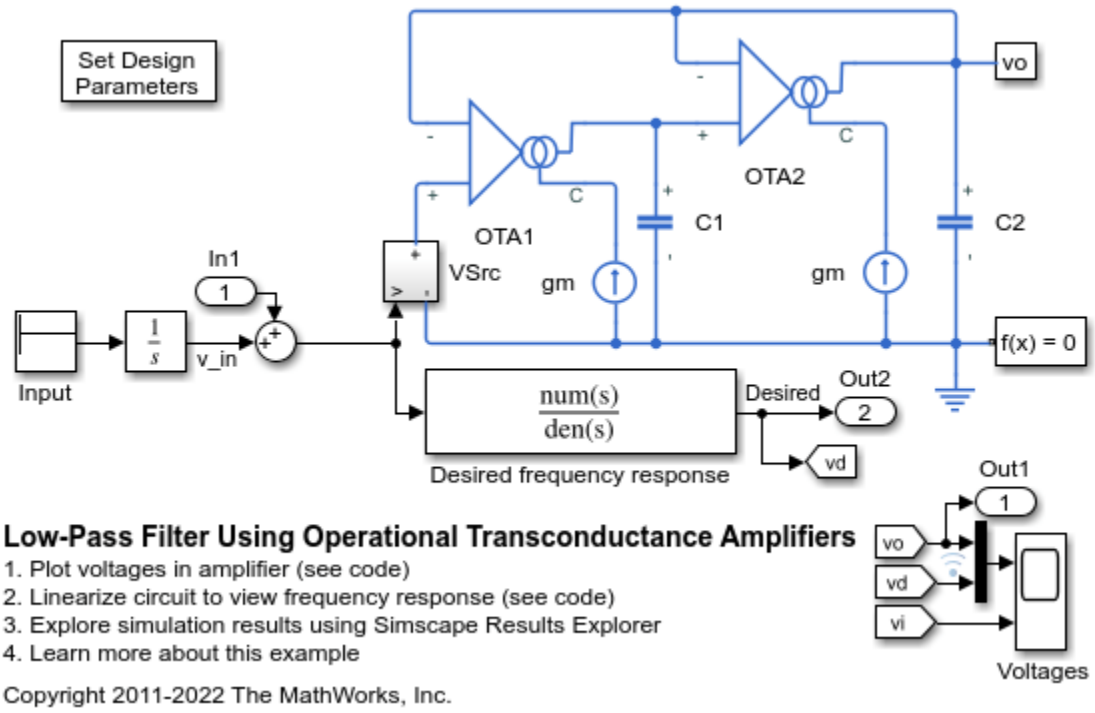


# Low-Pass Filter Using Operational Transconductance Amplifiers

This example shows how to model a second-order active low-pass filter. The filter is characterized by the transfer function H(s) = 1 / ( (s/w1)^2 + (1/Q)*(s/w1) + 1 ) where w1 = 2*pi*f1, f1 is the cut-off frequency and Q is the quality factor. Double-click on the Set Design Parameters block to set parameters f1 and Q. The block mask calls a function which sets the parameter values in the model workspace.

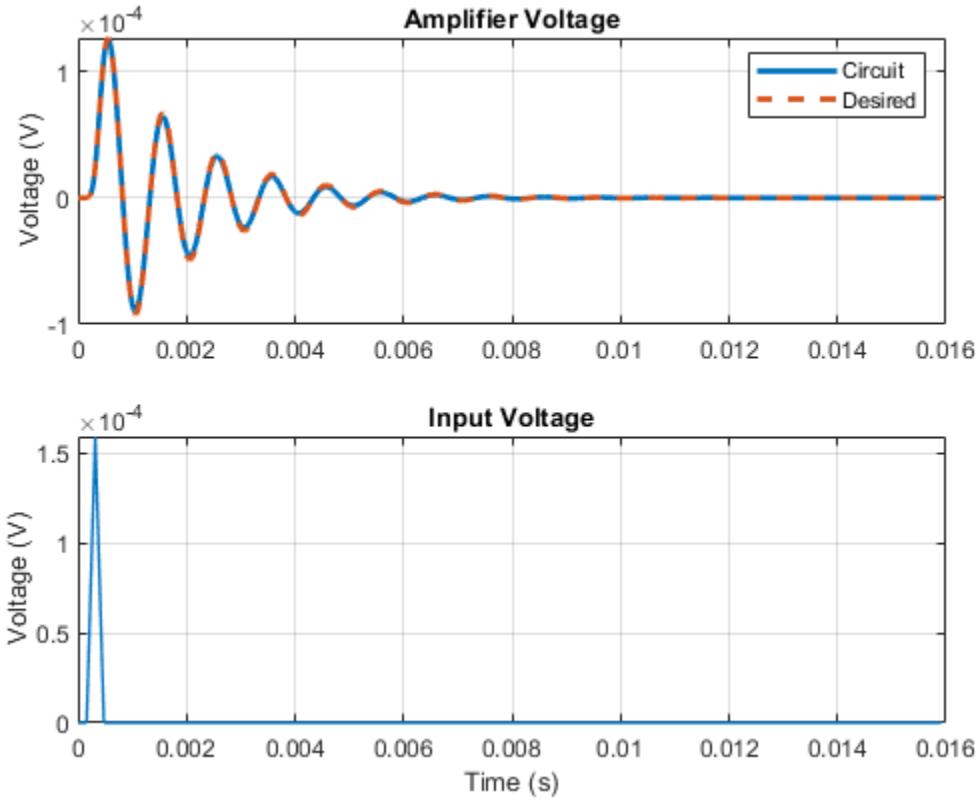
This model can be used to generate the filter frequency response. In the ideal case, the gain is zero dB at DC, -20*log10(1/Q) dB at frequency f1, and should attenuate at -12dB/octave at high frequency. The model can be used to determine the impact of impairments, such as finite transconductance gain, on the filter frequency response. Using an operational transconductance amplifier permits digital control of the filter by varying the values of the two current sources.

### Model



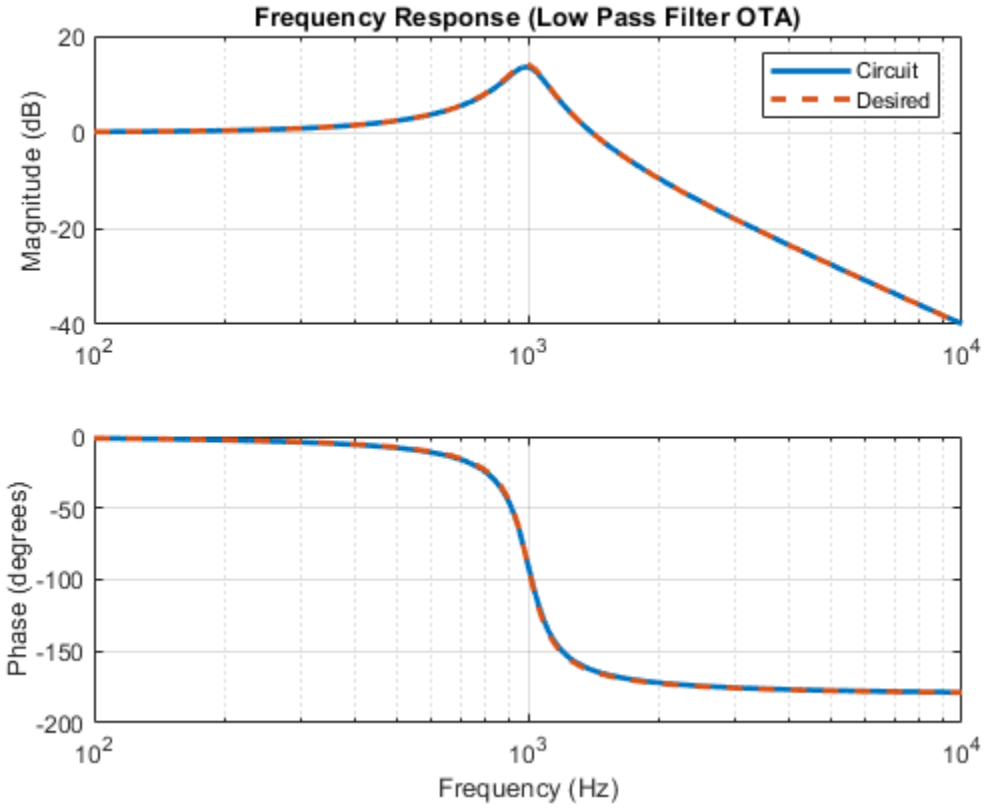

### Simulation Results from Simscape Logging

The plot below shows the response of the filter to a brief voltage pulse. This response can be analyzed numerically to determine the frequency response of the filter. The result from the circuit is compared with the result from a transfer function which was specified using the desired frequency response behavior, and we see that the results match nearly perfectly.




### Frequency Response

The plot below shows the frequency response of the filter. The gain is zero dB at DC, -20*log10(1/Q) dB at the cutoff frequency, and attenuates at -12dB/octave at high frequency. The frequency response obtained from the circuit nearly perfectly matches the results from a transfer function specified using the desired frequency response behavior.

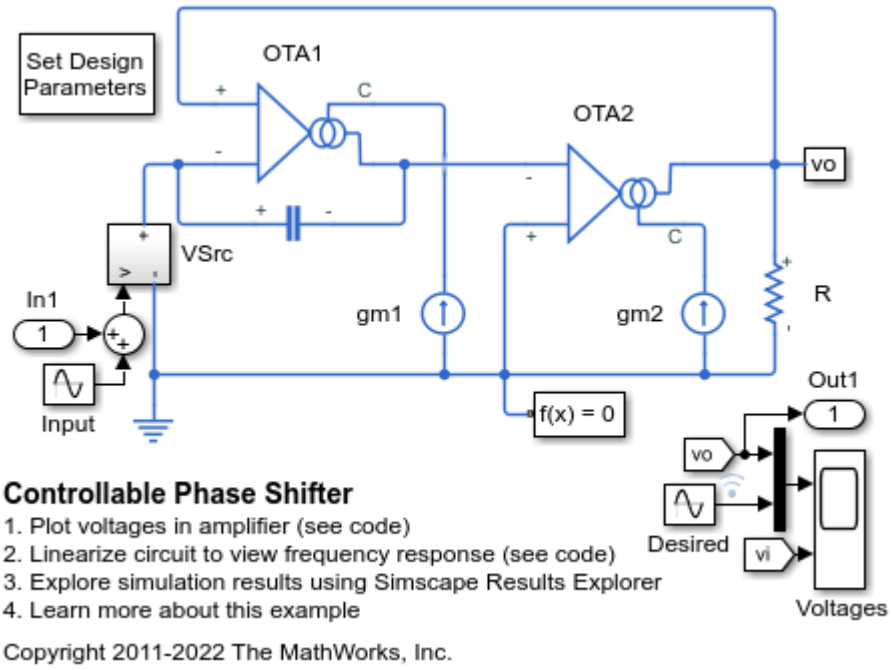

Frequency Response (Low Pass Filter OTA)
Circuit
Desired
Magnitude (dB)
20
0
-20
-40
$10^2$
$10^3$
$10^4$
Phase (degrees)
0
-50
-100
-150
-200
$10^2$
$10^3$
$10^4$
Frequency (Hz)

# Controllable Phase Shifter

This example shows an implementation of a first order phase shifting filter. The filter is characterized by the transfer function H(s) = (sC - gm1)/(sC + gm1). Double-click on the Set Design Parameters block to set the desired phase shift, amplitude of the input signal, and the frequency of the input signal. The block mask calls a function which sets the parameter values in the model workspace.

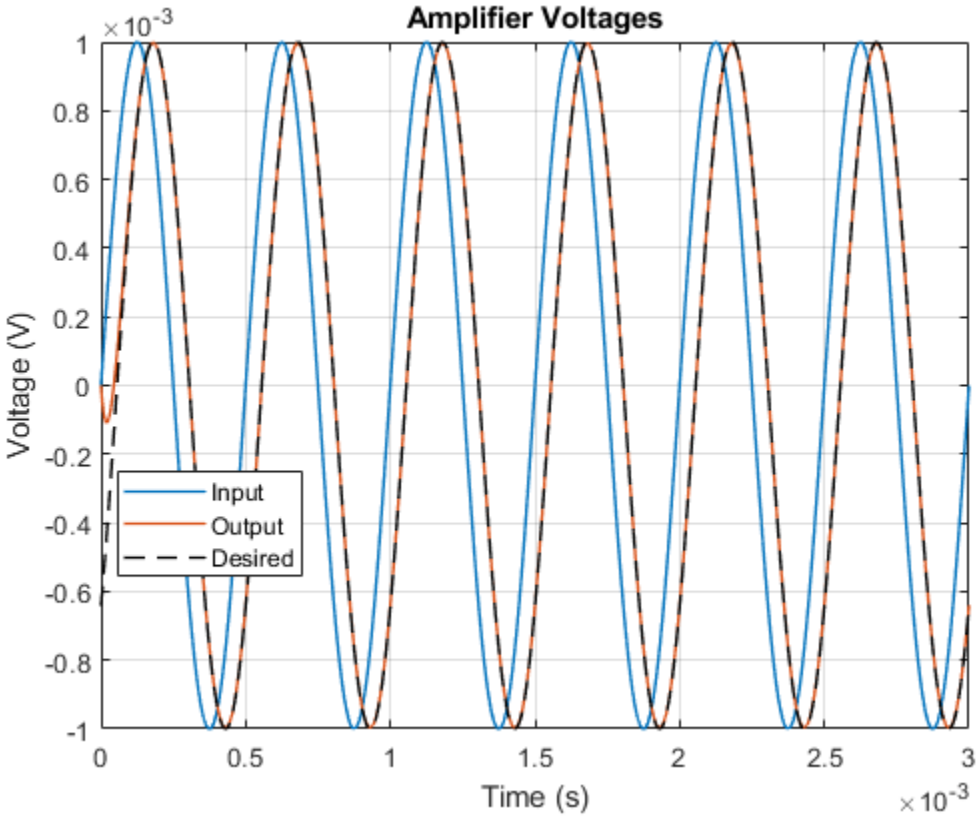
This model can be used to generate the filter frequency response. Using operational transconductance amplifiers permits digital control of the phase delay by varying the values of the current source gm1.

### Model



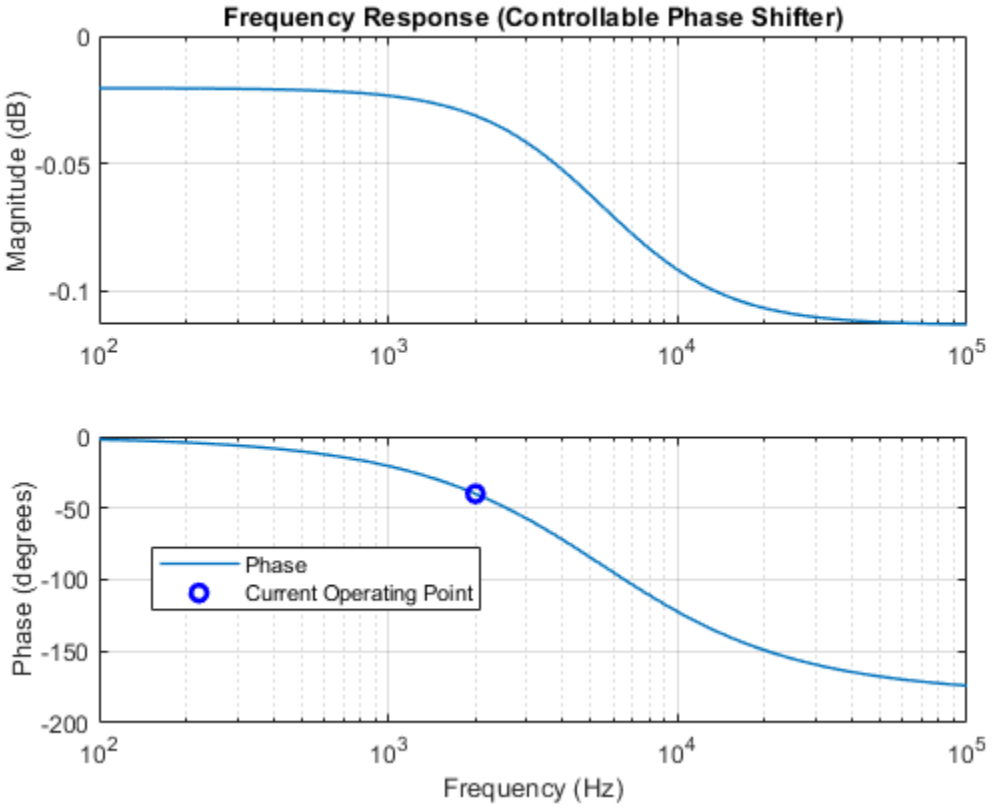

### Simulation Results from Simscape Logging

The plot below shows the input and output voltages of the phase shifting filter. The gain is 1, and the output signal is offset by the specified phase.




### Frequency Response

The plot below shows frequency response of the phase shifting filter. The specified operating condition is shown on the frequency response plot.

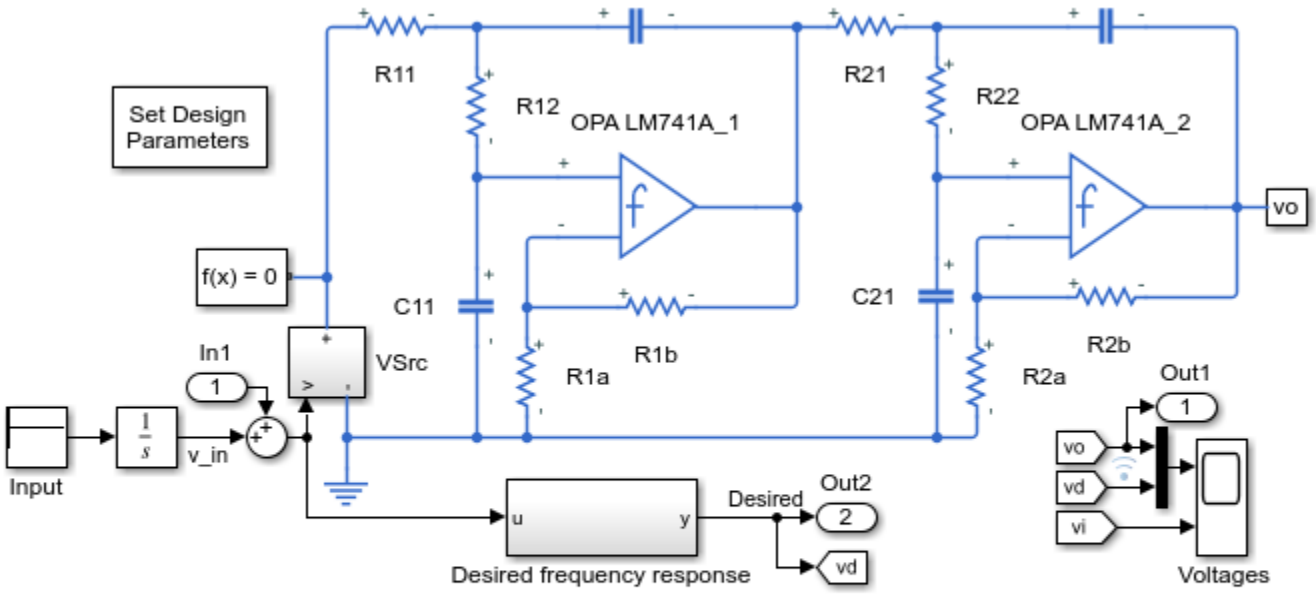

Frequency Response (Controllable Phase Shifter)
Magnitude (dB)
0
-0.05
-0.1
$10^2$
$10^3$
$10^4$
$10^5$
Phase (degrees)
0
-50
-100
-150
-200
Phase
Current Operating Point
Frequency (Hz)

# Fourth-Order Sallen-Key Lowpass Filter

This example shows an implementation of a fourth-order Sallen-Key low-pass filter using Operational Amplifiers (OPAs). The filter design parameters, cut-off frequency (f1) and DC gain (K), are specified by double-clicking on the Set Design Parameters block. Pass-band ripple is predefined to be 1dB using a Chebyshev response. The block mask calls a function which sets the parameter values in the model workspace.

This model can be used to generate the filter frequency response. In the ideal case, the gain is equal to 20*log10(K) dB at DC, 20*log(K) at the cut-off frequency, maximum value 20*log10(K) + 1, and -24dB/octave attenuation at high frequency.

### Model



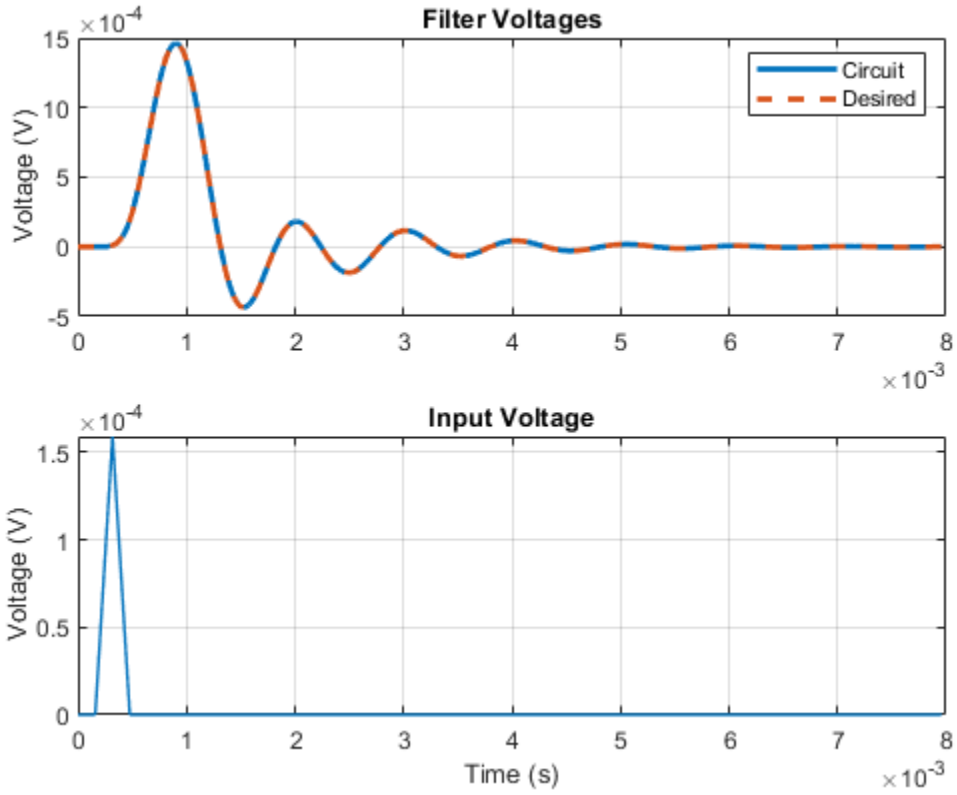

**Fourth-Order Sallen-Key Lowpass Filter**

1. Plot voltages in circuit (see code)
2. Linearize circuit to view frequency response (see code)
3. Explore simulation results using Simscape Results Explorer
4. Learn more about this example

Copyright 2011-2022 The MathWorks, Inc.

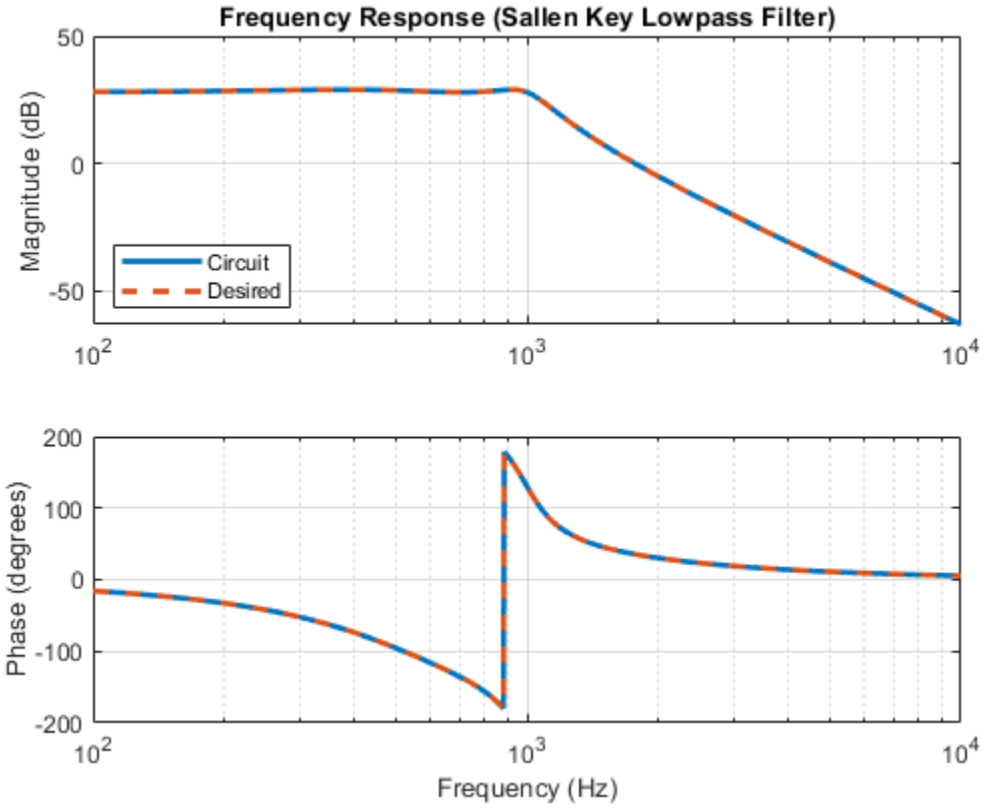
### Simulation Results from Simscape Logging

The plot below shows the response of the filter to a brief voltage pulse. This response can be analyzed numerically to determine the frequency response of the filter. The result from the circuit is compared with the result from a transfer function which was specified using the desired frequency response behavior, and we see that the results match nearly perfectly.



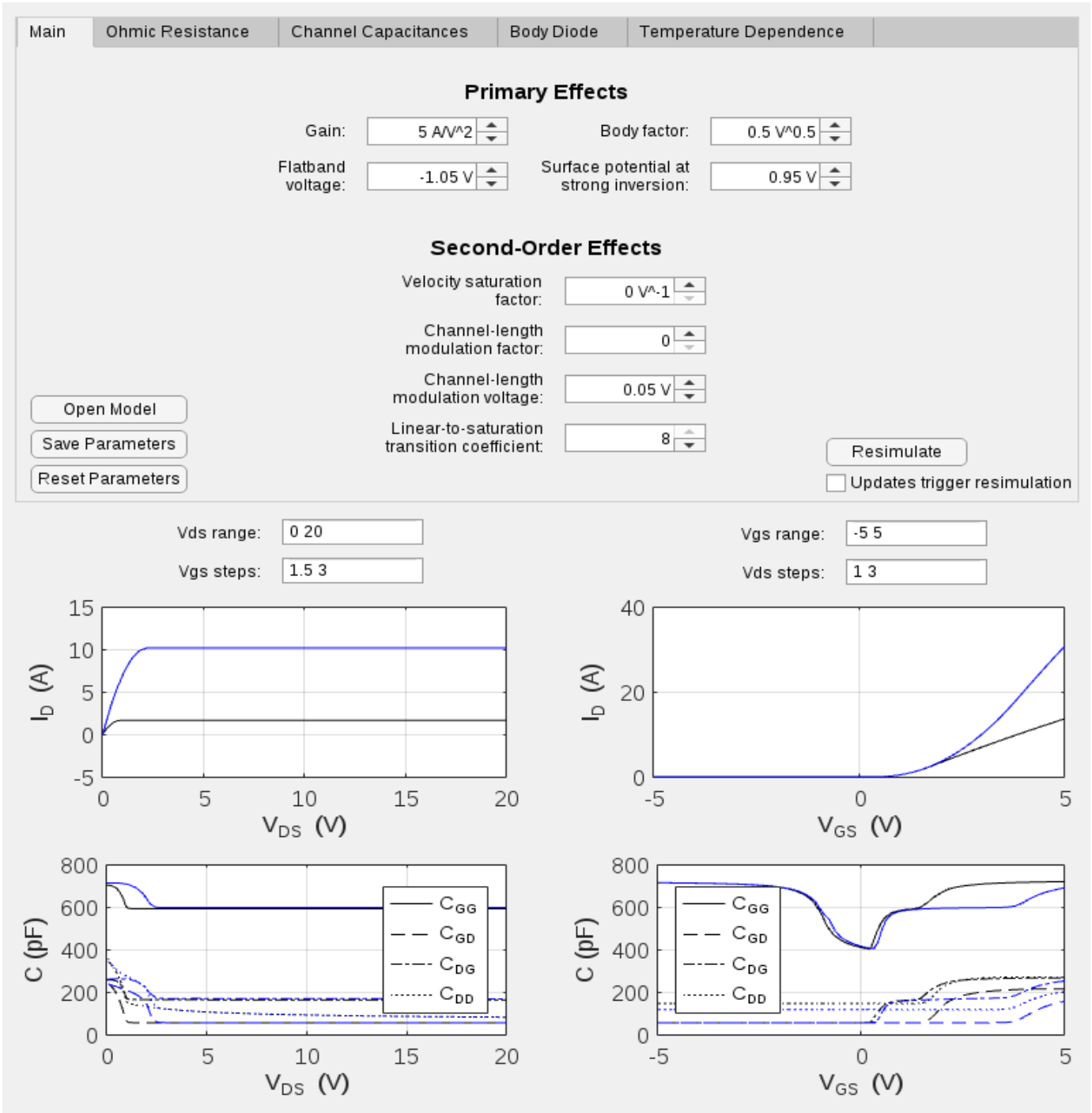

### Frequency Response

The plot below shows the frequency response of the filter. The gain is equal to 20*log10(K) dB at DC, 20*log(K) at the cut-off frequency, maximum value 20*log10(K) + 1, and -24dB/octave attenuation at high frequency. The frequency response obtained from the circuit nearly perfectly matches the results from a transfer function specified using the desired frequency response behavior.


Frequency Response (Sallen Key Lowpass Filter)
Magnitude (dB)
50
0
-50
Circuit
Desired
$10^2$
$10^3$
$10^4$
Phase (degrees)
200
100
0
-100
-200
$10^2$
$10^3$
$10^4$
Frequency (Hz)

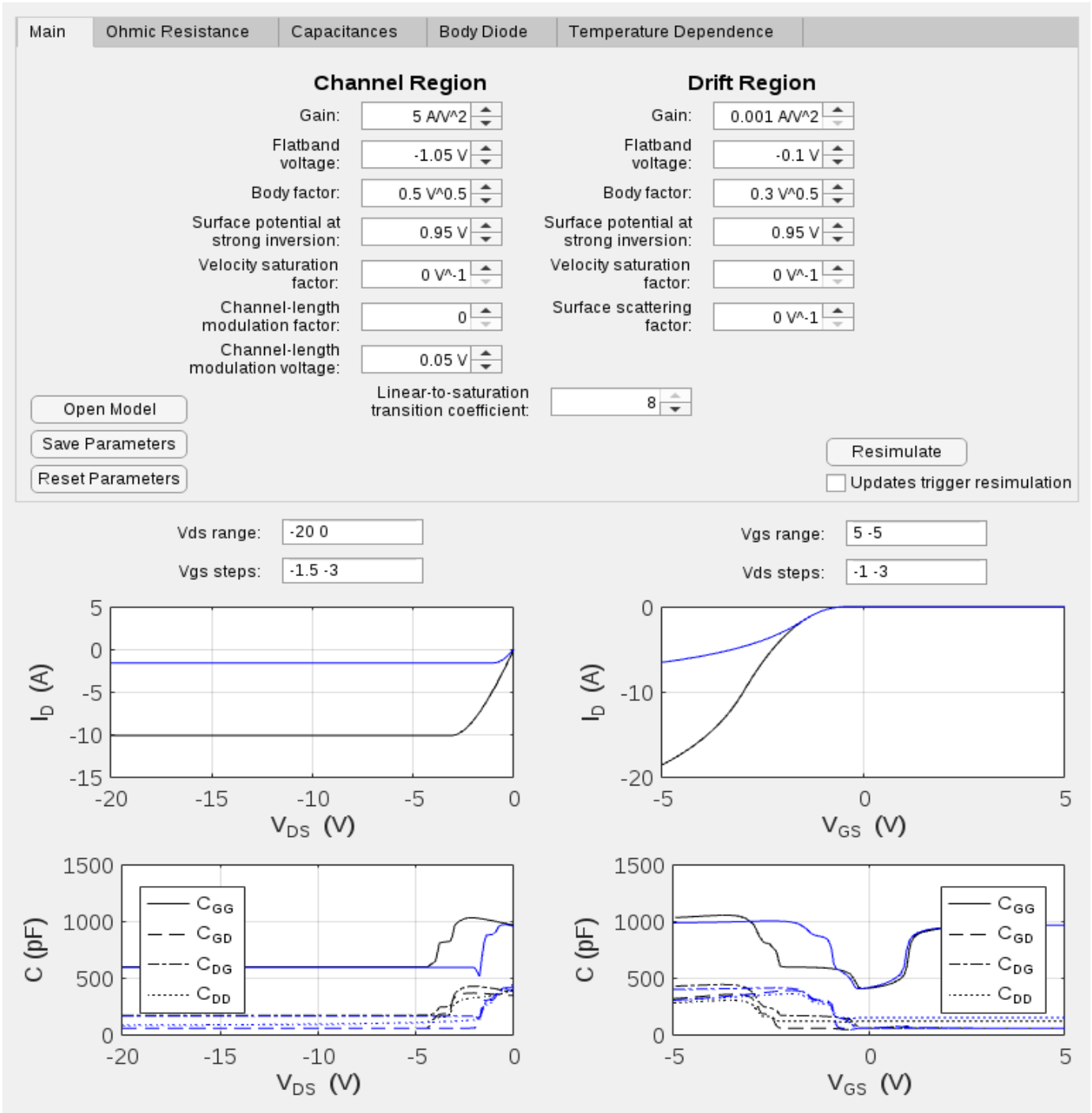
# Interactive Generation of MOSFET Characteristics

This example allows the user to explore the impact of parameter choices on the I-V and C-V characteristics for a surface-potential-based MOSFET model. To open this example, in the MATLAB® Command Window, type: ee_mosfet_characteristics.


Main
Ohmic Resistance
Channel Capacitances
Body Diode
Temperature Dependence
Primary Effects
Gain:
5 A/V^2
Body factor:
0.5 V^0.5
Flatband voltage:
-1.05 V
Surface potential at strong inversion:
0.95 V
Second-Order Effects
Velocity saturation factor:
0 V^-1
Channel-length modulation factor:
0
Channel-length modulation voltage:
0.05 V
Linear-to-saturation transition coefficient:
8
Open Model
Save Parameters
Reset Parameters
Resimulate
Updates trigger resimulation
Vds range:
0 20
Vgs steps:
1.5 3
Vgs range:
-5 5
Vds steps:
1 3
$I_D$ (A)
$V_{DS}$ (V)
$V_{GS}$ (V)
C (pF)
$C_{GG}$
$C_{GD}$
$C_{DG}$
$C_{DD}$

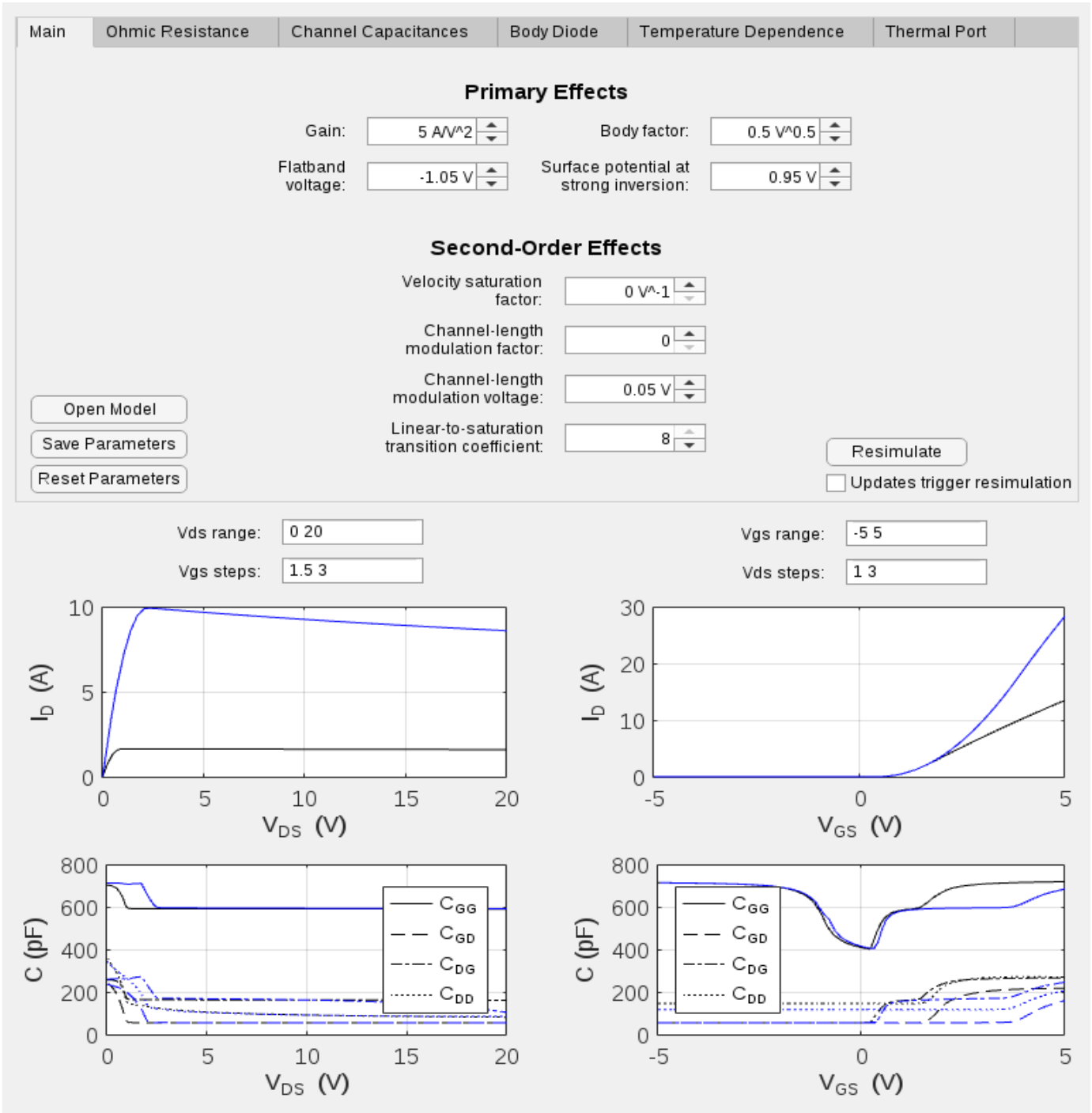
# Interactive Generation of LDMOS Characteristics

This example allows the user to explore the impact of parameter choices on the I-V and C-V characteristics for a surface-potential-based p-channel LDMOS field-effect transistor model. To open this example, in the MATLAB® Command Window, type: ee_p_ldmos_characteristics.


Main
Ohmic Resistance
Capacitances
Body Diode
Temperature Dependence
Channel Region
Gain: 5 A/V^2
Flatband voltage: -1.05 V
Body factor: 0.5 V^0.5
Surface potential at strong inversion: 0.95 V
Velocity saturation factor: 0 V^-1
Channel-length modulation factor: 0
Channel-length modulation voltage: 0.05 V
Drift Region
Gain: 0.001 A/V^2
Flatband voltage: -0.1 V
Body factor: 0.3 V^0.5
Surface potential at strong inversion: 0.95 V
Velocity saturation factor: 0 V^-1
Surface scattering factor: 0 V^-1
Linear-to-saturation transition coefficient: 8
Open Model
Save Parameters
Reset Parameters
Resimulate
Updates trigger resimulation
Vds range: -20 0
Vgs steps: -1.5 -3
Vgs range: 5 -5
Vds steps: -1 -3
$I_D$ (A)
$V_{DS}$ (V)
$V_{GS}$ (V)
C (pF)
$C_{GG}$
$C_{GD}$
$C_{DG}$
$C_{DD}$

# Interactive Generation of Thermal MOSFET Characteristics

This example allows the user to explore the impact of parameter choices on the I-V and C-V characteristics for a surface-potential-based thermal MOSFET model. To open this example, in the MATLAB® Command Window, type: ee_thermal_mosfet_characteristics.


Main
Ohmic Resistance
Channel Capacitances
Body Diode
Temperature Dependence
Thermal Port
Primary Effects
Gain:
5 A/V^2
Body factor:
0.5 V^0.5
Flatband voltage:
-1.05 V
Surface potential at strong inversion:
0.95 V
Second-Order Effects
Velocity saturation factor:
0 V^-1
Channel-length modulation factor:
0
Channel-length modulation voltage:
0.05 V
Linear-to-saturation transition coefficient:
8
Open Model
Save Parameters
Reset Parameters
Resimulate
Updates trigger resimulation
Vds range:
0 20
Vgs steps:
1.5 3
Vgs range:
-5 5
Vds steps:
1 3
$I_D$ (A)
$V_{DS}$ (V)
$V_{GS}$ (V)
C (pF)
$C_{GG}$
$C_{GD}$
$C_{DG}$
$C_{DD}$

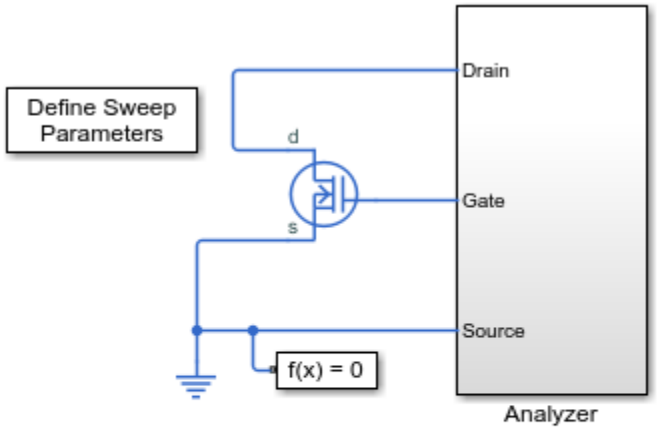
# MOSFET Characterization Using Y Parameters

This example shows the generation of I-V and C-V characteristics for an NMOS transistor. Define the bias conditions for the gate-source and drain- source voltage sweeps and the types of plots to be generated by double- clicking the Define Sweep Parameters block. Then click on the "Generate plots" hyperlink in the model . The output capacitance, C_oss, is only shown for sweeps of the drain-source voltage. Note that the C-V characteristics can take several minutes each to generate.

This type of analysis can be used in order to compare against a manufacturer datasheet to confirm a correct implementation of the transistor parameters. This applies to both the DC and AC transistor characteristics. You can also use this model to examine the nonlinear capacitance characteristics as functions of the bias conditions.

Double-click the Analyzer block to see the underlying circuit. It superimposes an AC small-signal voltage on top of a DC bias voltage. The DC characteristics are obtained by filtering out the AC part of the response. The small-signal Y-parameters are obtained by subtracting the DC current from the total (AC+DC) current, and then dividing by the small-signal excitation voltage.

### Model



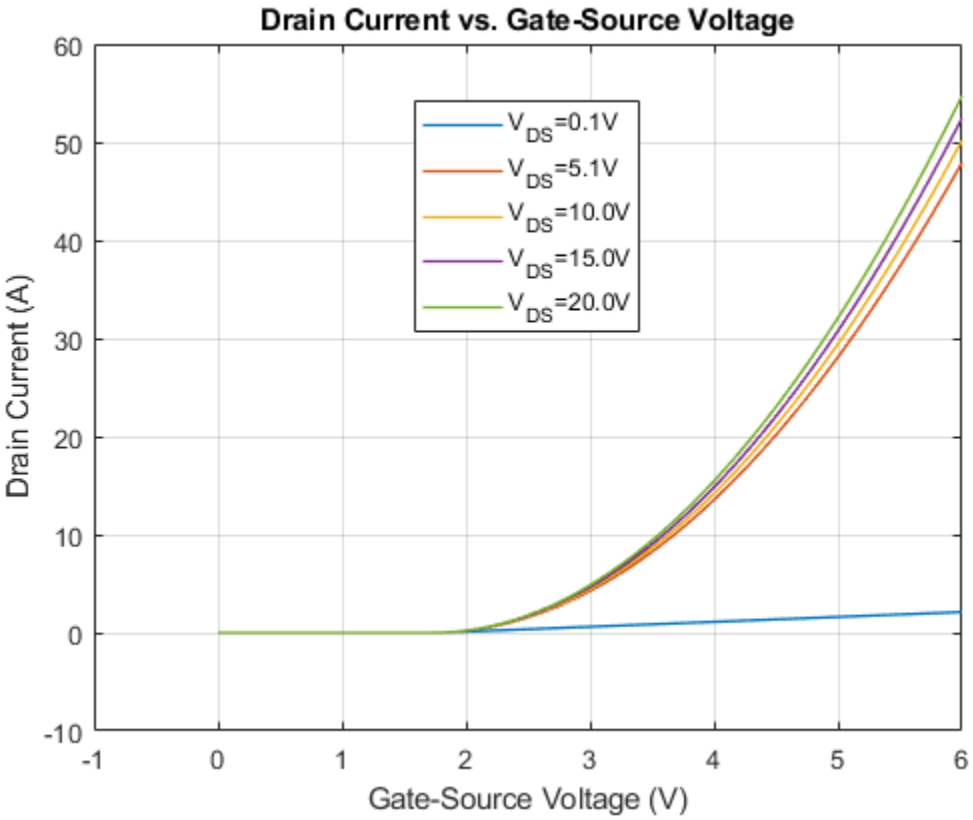

**MOSFET Characterization Using Y Parameters**

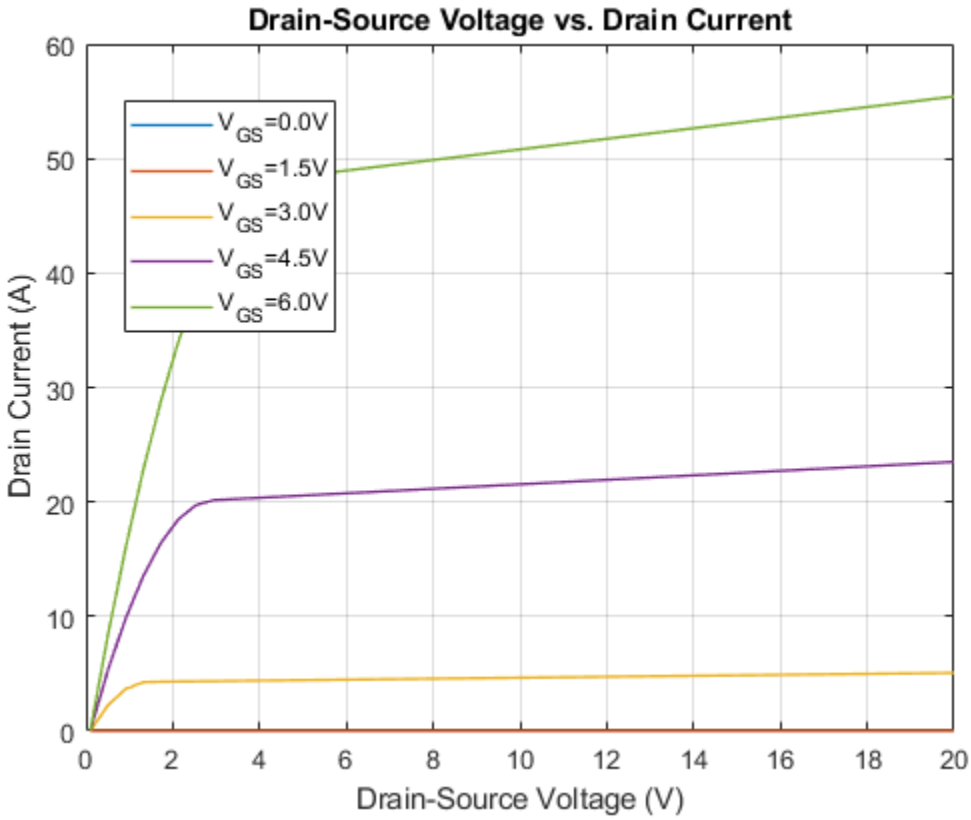
1. Characteristics: Define Sweep, Generate plots (see code)
2. Explore simulation results using Simscape Results Explorer
3. Learn more about this example

Copyright 2015-2022 The MathWorks, Inc.

### Simulation Results from Simscape Logging

The plots below show Id, Ciss, Crss and Coss as functions of Vgs and/or Vds as appropriate for the parameters chosen. Successive simulations are run for the range of bias conditions defined. In order to handle small-signal measurements, every bias point is left to settle for a certain number of AC cycles before the measurement is recorded for plotting.

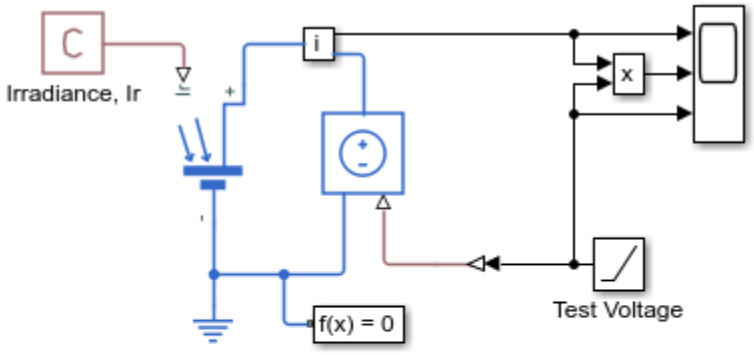

Drain Current vs. Gate-Source Voltage
Drain Current (A)
Gate-Source Voltage (V)
60
50
40
30
20
10
0
-10
-1
0
1
2
3
4
5
6
V_DS=0.1V
V_DS=5.1V
V_DS=10.0V
V_DS=15.0V
V_DS=20.0V


Drain-Source Voltage vs. Drain Current
Drain Current (A)
Drain-Source Voltage (V)
V_GS=0.0V
V_GS=1.5V
V_GS=3.0V
V_GS=4.5V
V_GS=6.0V
0
10
20
30
40
50
60
0
2
4
6
8
10
12
14
16
18
20

# Solar Cell Power Curve

This example shows how to generate the power-voltage curve for a solar array. Understanding the power-voltage curve is important for inverter design. Ideally the solar array would always be operating at peak power given the irradiance level and panel temperature.

### Model



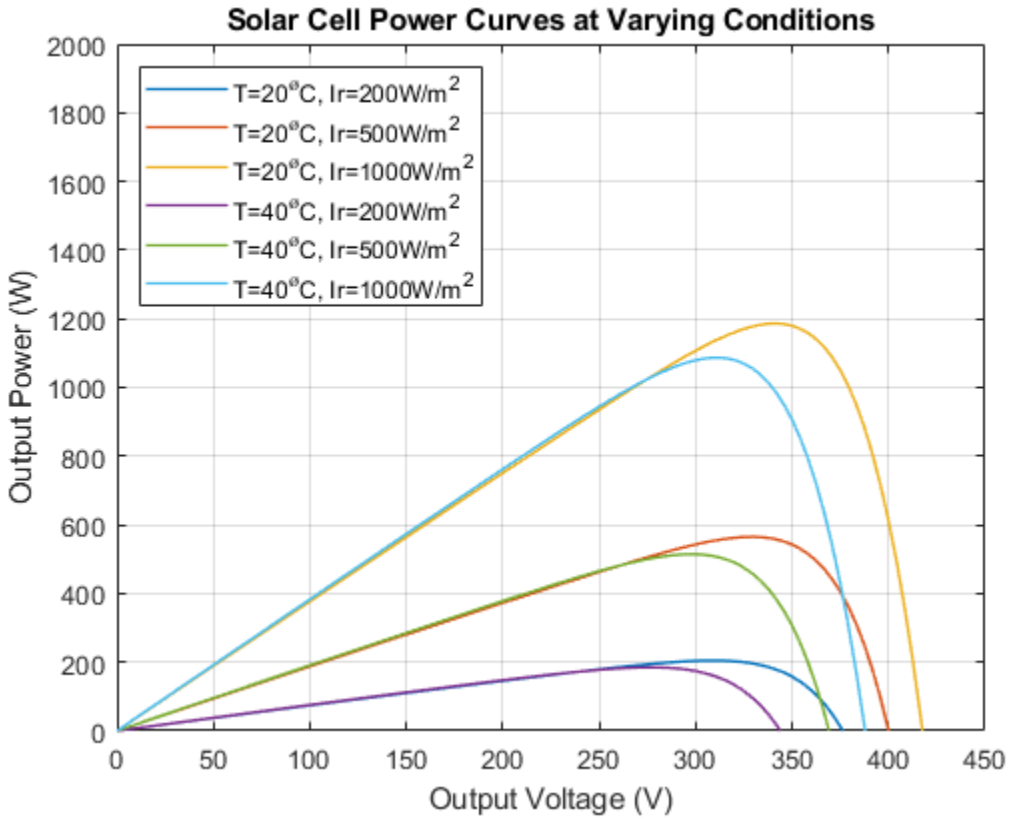

**Solar Cell Power Curve**

1. Plot power curves at varying conditions (see code)
2. Explore simulation results using Simscape Results Explorer
3. Learn more about this example

Copyright 2008-2022 The MathWorks, Inc.

### Simulation Results from Simscape Logging

This example shows how to generate the power-voltage curve for a solar array. Understanding the power-voltage curve is important for inverter design. Ideally the solar array would always be operating at peak power given the irradiance level and panel temperature.


Solar Cell Power Curves at Varying Conditions
T=20°C, Ir=200W/m2
T=20°C, Ir=500W/m2
T=20°C, Ir=1000W/m2
T=40°C, Ir=200W/m2
T=40°C, Ir=500W/m2
T=40°C, Ir=1000W/m2
Output Power (W)
Output Voltage (V)
0
200
400
600
800
1000
1200
1400
1600
1800
2000
0
50
100
150
200
250
300
350
400
450

# Use of Peltier Device as Thermoelectric Cooler

This example shows a Peltier device working in cooling mode with a hot side temperature of 50 degC. In cooling mode, the Coefficient of Performance (COP) of the Peltier cell is equal to the total heat transferred through the Thermoelectric cooler (TEC) divided by the electric input power, COP = Qc/Pin.

The first subplot shows the COP as a function of current for several temperature differences. It can be observed that, for the same current value, the COP value is lower for larger temperature differences. This happens because at higher temperature differences, the natural heat conduction from the hot side to the cold side tends to be greater.

The temperature source connected to the hot side of the Peltier module allows to represent an ideal heat sink. That is, a sink capable of dissipating an infinite amount of heat without increases in temperature at the hot side. This representation gives us a good approximation for the COP and cooling flow curves.

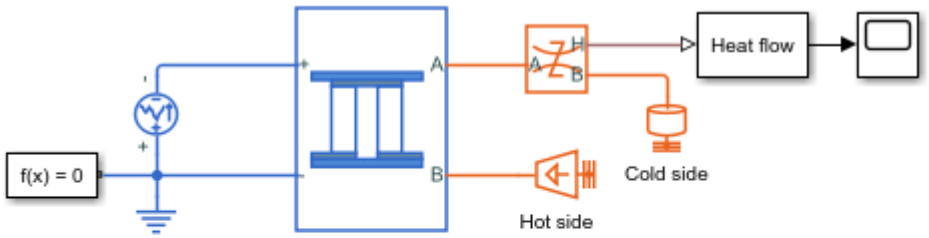
The second graph shows how the heat flow evolves with the input current. As the current increases, the heat extracted from the cold side increases until it reaches a maximum value. After this peak, due to joule heating, increases in current lead to a decrease in the cooling flow.

The last graph shows the operating current that maximizes the COP value for each temperature difference. This optimum operating point provides useful information when choosing the TEC controller output current.

The Peltier block used in this model is defined by:

- Seebeck coefficient = 220e-6 V/K;
- Internal resistance = 0.02 Ohm;
- Thermal conductance = 1.5e-3 W/K.

**Model**



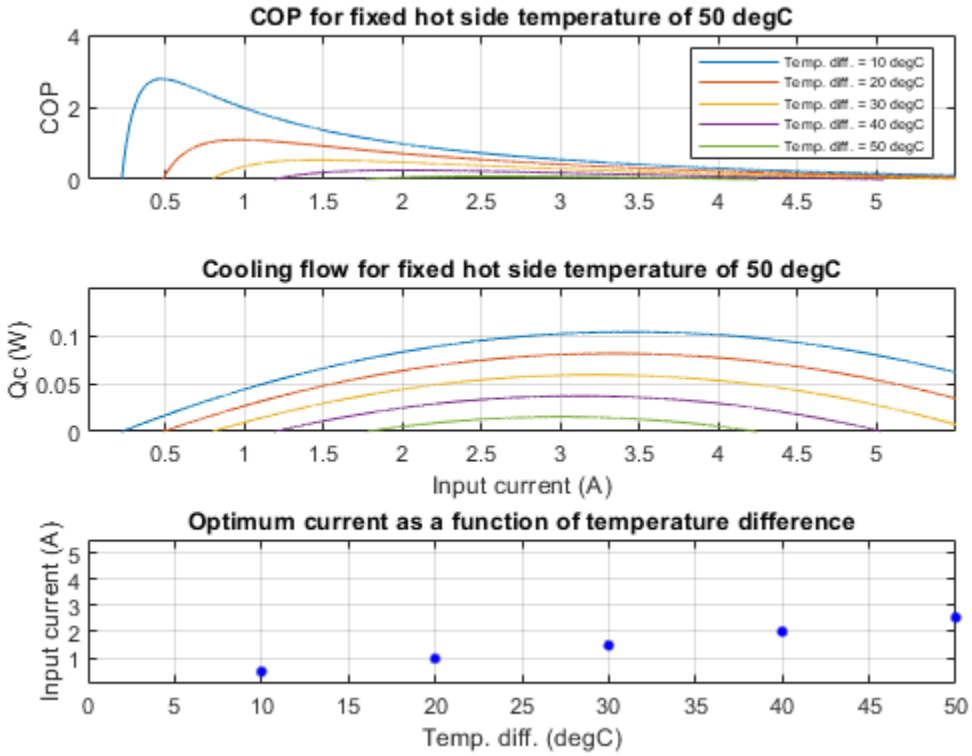

**Use of Peltier Device as Thermoelectric Cooler**

1. Plot characteristic curves for the Peltier cell (see code)
2. Explore simulation results using Simscape Results Explorer
3. Learn more about this example

Copyright 2018-2022 The MathWorks, Inc.

### Simulation Results from Simscape Logging

The plot below shows the results for the Peltier Device's coefficient of performance and heat pumping capacity against input current at various temperature differences.

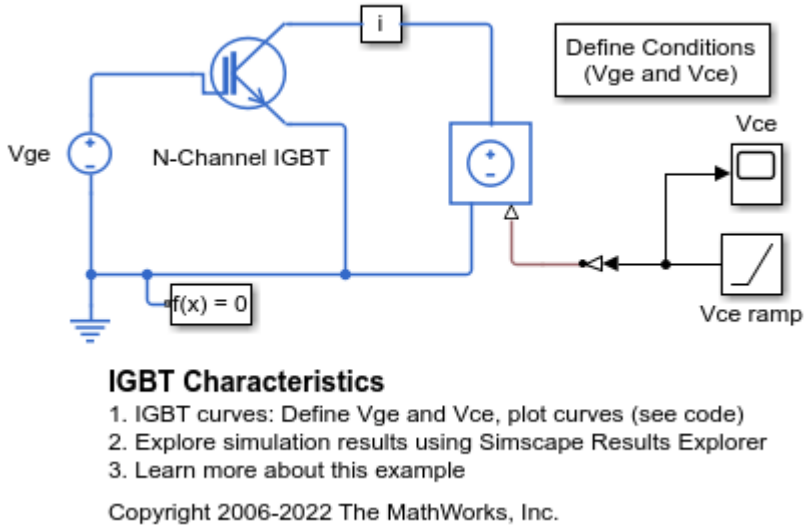


# IGBT Characteristics

This example shows generation of the Ic versus Vce curve for an insulated gate bipolar transistor. Define the vector of gate-emitter voltages and minimum and maximum collector-emitter voltages by double-clicking the block labeled 'Define Conditions (Vge and Vce)'. Click on the hyperlink 'plot curves' in the model to run the simulations plot the simulation results.

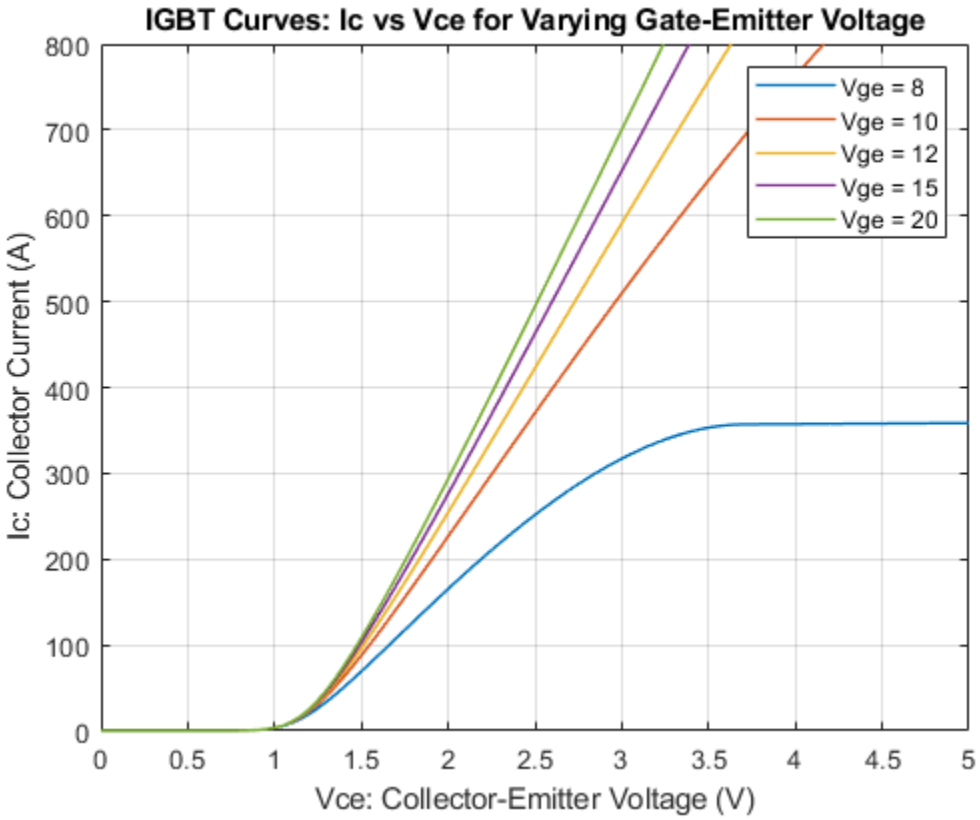
This type of plot can be compared against a manufacturer datasheet to confirm a correct implementation of the transistor parameters.

### Model




### Simulation Results from Simscape Logging

The plot below shows the collector current vs collector-emitter voltage characteristics for a range of gate-emitter voltages.

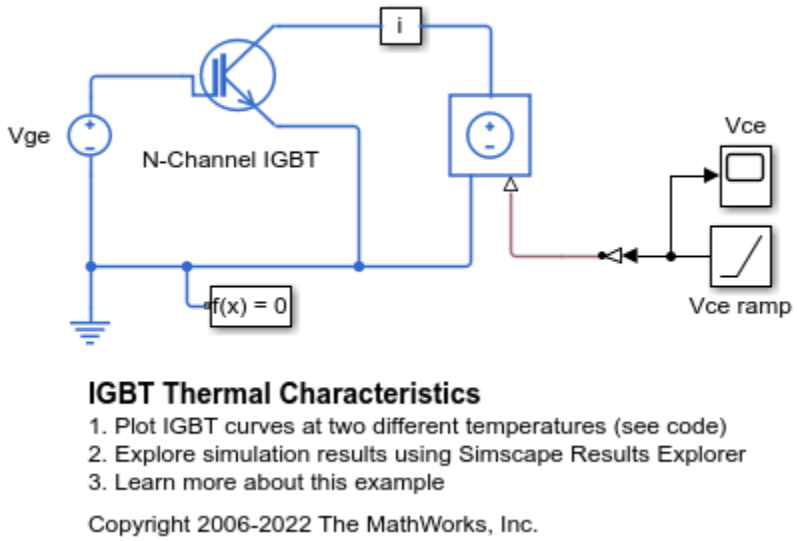

IGBT Curves: Ic vs Vce for Varying Gate-Emitter Voltage
Ic: Collector Current (A)
Vce: Collector-Emitter Voltage (V)
Vge = 8
Vge = 10
Vge = 12
Vge = 15
Vge = 20
0
100
200
300
400
500
600
700
800
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5

# IGBT Thermal Characteristics

This example shows generation of the Ic versus Vce curve for an IGBT at two different temperatures. To generate the plot, click on the hyperlink in the model labeled 'Plot IGBT curves'.

On the plot, the solid lines are from running the model, and the dotted lines are digitized datasheet data. The N-Channel IGBT component has been parameterized using Vce values for when Ic is 400A and Vge is 10V. These two points are intercepted by both model and datasheet plots for Vge=10V. The remaining block parameters must be tuned by simulation so that the datasheet and model plots match as best as is possible. See the N-Channel IGBT block reference page for further information.

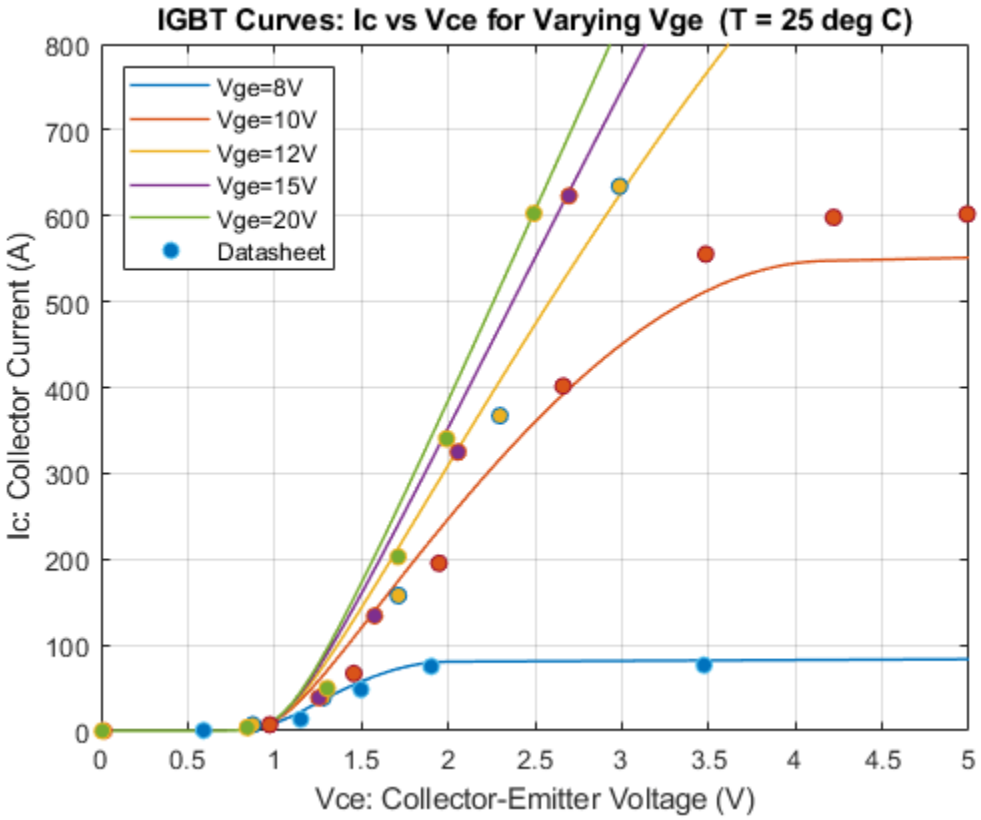
### Model

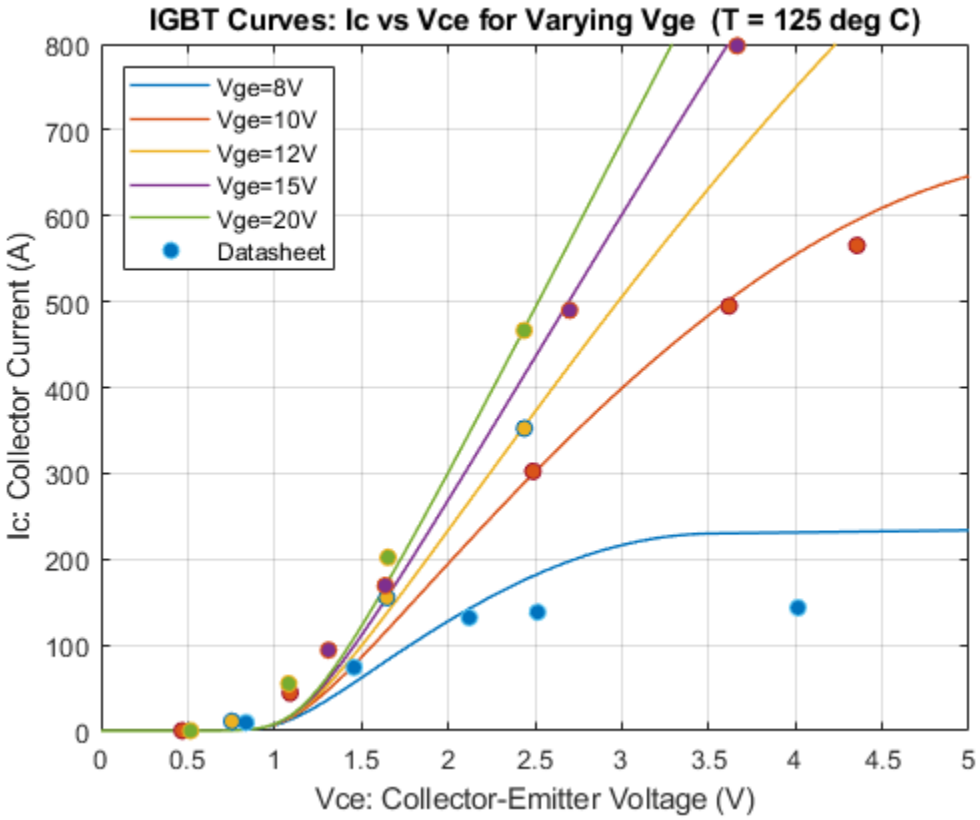



### Simulation Results from Simscape Logging

The plot below shows the collector current vs collector-emitter voltage characteristics for a range of gate-emitter voltages. The tests are run at two different temperatures. The results are compared to data points obtained from a manufacturer's data sheet.

You can digitize datasheet information using tools available on MATLAB® Central File Exchange: https://www.mathworks.com/matlabcentral/, such as GRABIT [1].


IGBT Curves: Ic vs Vce for Varying Vge (T = 25 deg C)
Vge=8V
Vge=10V
Vge=12V
Vge=15V
Vge=20V
Datasheet
Ic: Collector Current (A)
Vce: Collector-Emitter Voltage (V)
0
100
200
300
400
500
600
700
800
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5




## References

**1** jiro (2019). GRABIT (https://www.mathworks.com/matlabcentral/fileexchange/7173-grabit), MATLAB Central File Exchange. Retrieved December 16, 2019.

# IGBT Dynamic Characteristics

This example shows how the dynamic characteristics of an IGBT depend on its parameters. A prerequisite to matching dynamic characteristics to datasheet values or measured data is to set the parameters defining the static I-V curve. For this, see the 'IGBT Characteristics' example, ee_igbt. With static parameters correctly set, the dynamic parameters can then be set as follows:

1. Set input capacitance, Cies, to datasheet value, or iterate until Vge has the measured rise time up to the Miller length (flat region of Vge at Vge(th)).

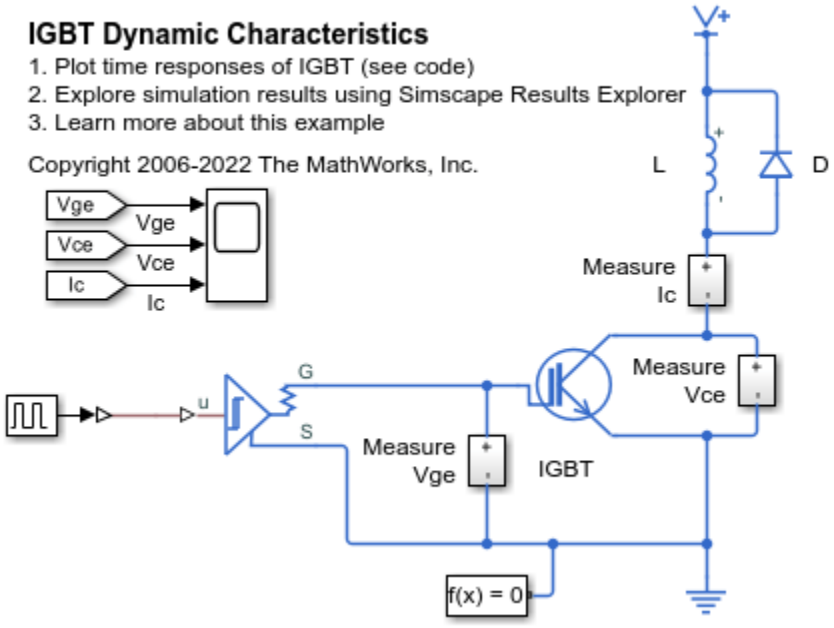
2. Set reverse transfer capacitance, Cres, to datasheet value, or iterate until the Miller length matches the measured value.

3. Iterate on the gate-collector oxide capacitance to achieve the measured Vge time constant following the Miller length.

4. Iterate on the total forward transit time, TF, until the off-going collector current, Ic, shows the measured time constant.

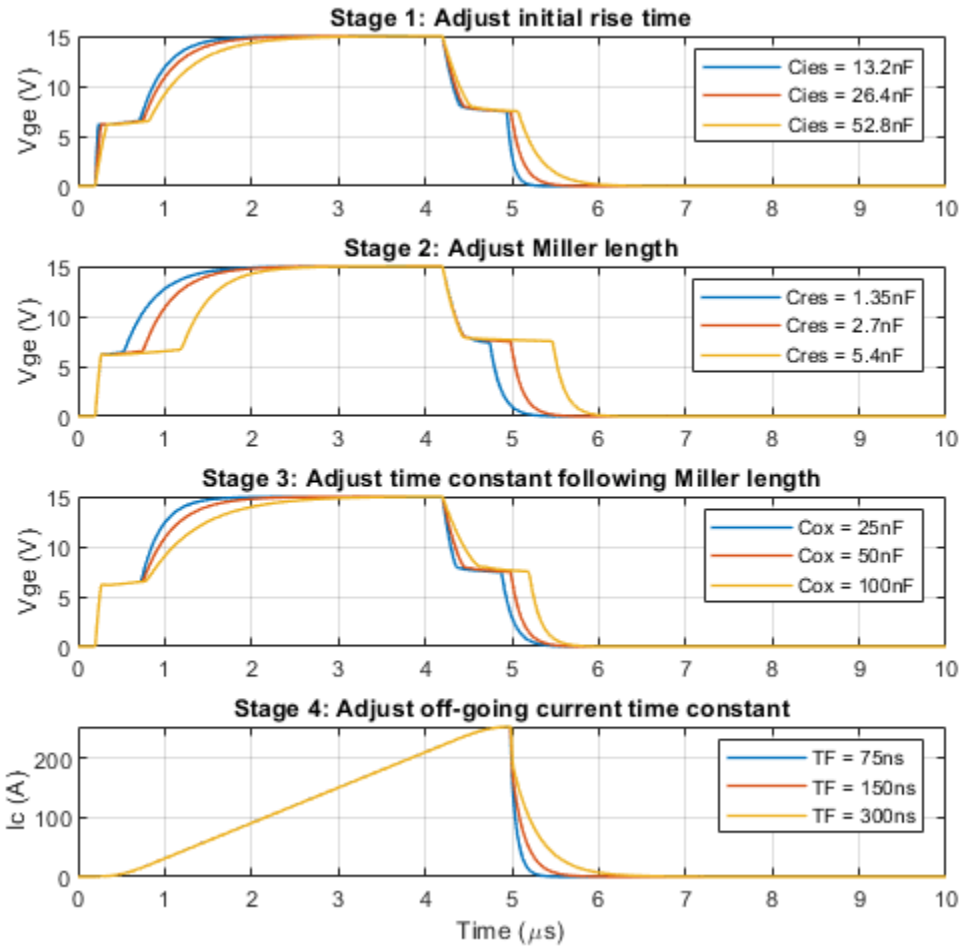
Click on the hyperlink 'Plot time responses' in the model to view time responses as a function of parameter values.

**Model**




**Simulation Results from Simscape Logging**

The plots below show the effect of changing IGBT mask parameters Cres, Cox and forward transit time.


Stage 1: Adjust initial rise time
Vge (V)
Cies = 13.2nF
Cies = 26.4nF
Cies = 52.8nF
Stage 2: Adjust Miller length
Vge (V)
Cres = 1.35nF
Cres = 2.7nF
Cres = 5.4nF
Stage 3: Adjust time constant following Miller length
Vge (V)
Cox = 25nF
Cox = 50nF
Cox = 100nF
Stage 4: Adjust off-going current time constant
Ic (A)
TF = 75ns
TF = 150ns
TF = 300ns
Time (μs)

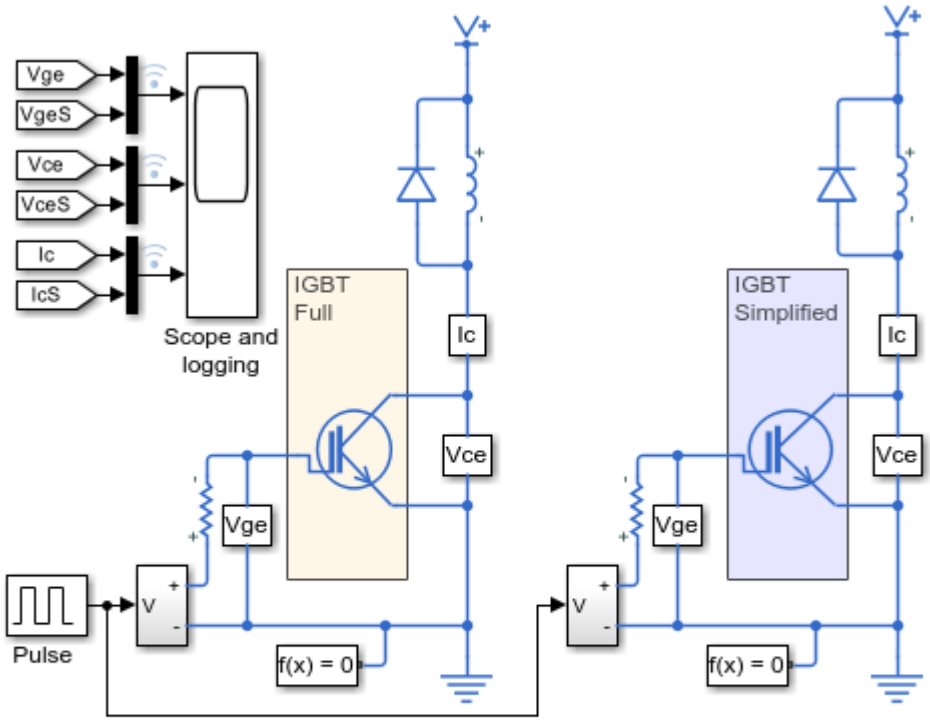
# IGBT Behavioral Model

This example test harness can be used to validate the 'Simplified I-V characteristics and event-based timing' Modeling option of the Simscape™ Electrical™ N-Channel IGBT. This modeling option only requires I-V data corresponding to the on-state gate voltage, and models turn-on rise time and turn-off fall time by making collector-emitter voltage a linear function of simulation time. Advantages of this approach are faster simulation and easier parameterization.

The test harness can be used to demonstrate that timing characteristics for this modeling option are very similar to those for the 'Full I-V and capacitance characteristics' modeling option.

Both modeling options have been parameterized from the same datasheet. The only event-based modeling option parameter that requires tuning is the Miller resistance. This is adjusted so that the current spike matches when turning on a load with pre-existing freewheeling current.

**Model**



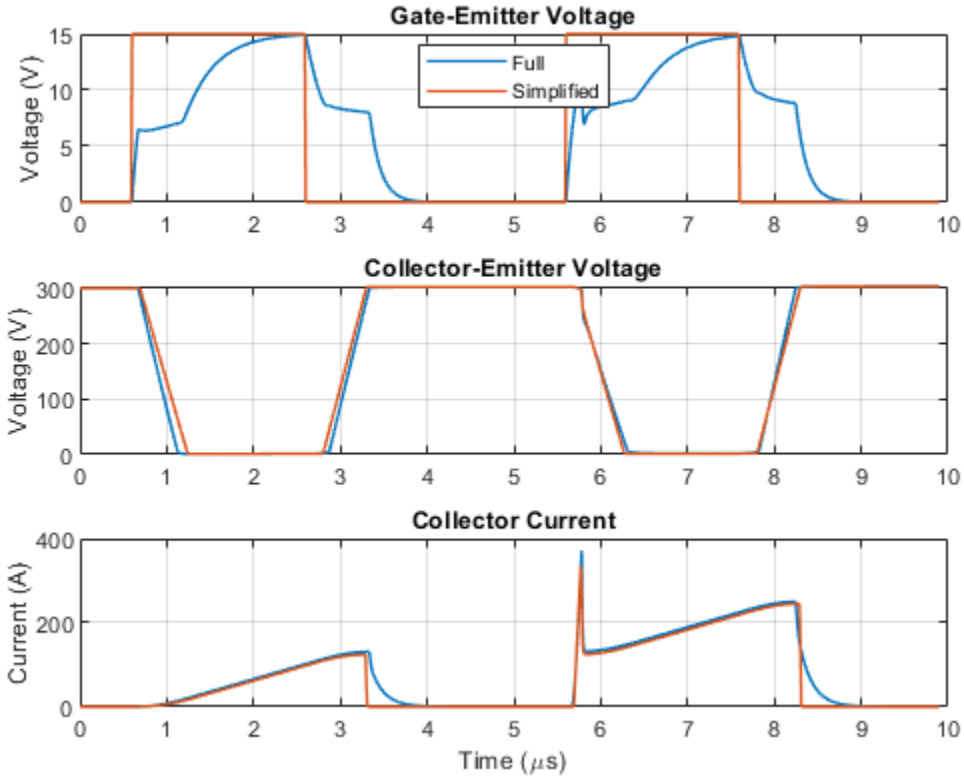

**IGBT Behavioral Model**

1. Plot voltages to compare the two IGBT models (see code)
2. Explore simulation results using Simscape Results Explorer
3. Learn more about this example

Copyright 2006-2022 The MathWorks, Inc.

### Simulation Results from Simscape Logging

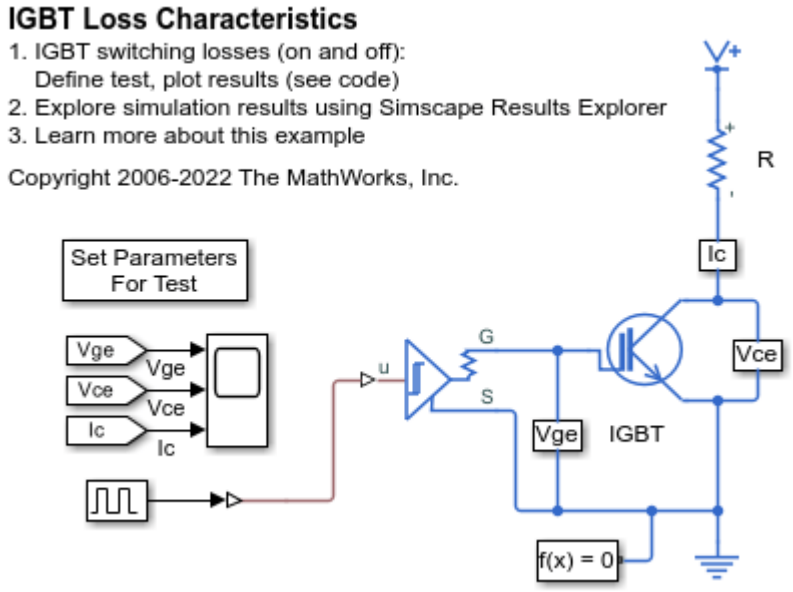
The plots below compare the behavior of two of the IGBT models available in Simscape Electrical. The simplified modeling option provides similar behavior as the full modeling option, and has been tuned so that the current spike matches when turning on a load with pre-existing freewheeling current.



# IGBT Loss Characteristics

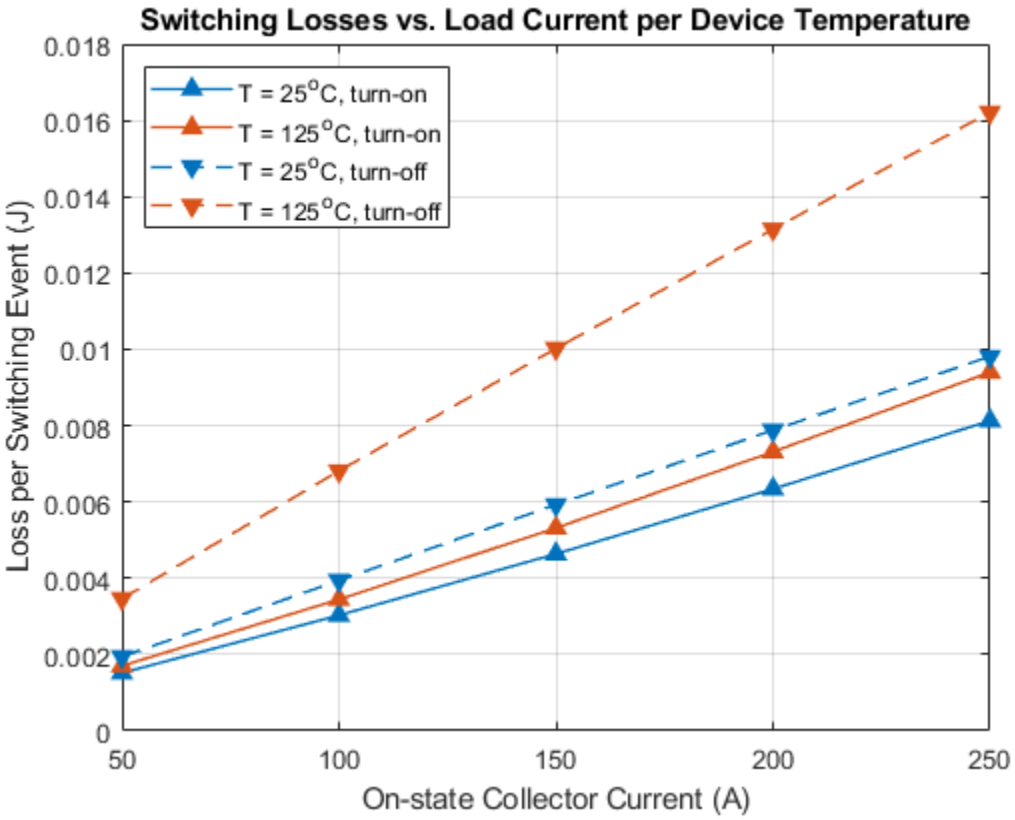
This example shows how to use Simscape™ Electrical™ detailed switching device models to create tabulated switching loss data. This tabulated data can then be used with the piecewise linear switching device component models to predict total losses in system models configured for fast and/or fixed-step simulation.

### Model




### Simulation Results from Simscape Logging

The plots below show the turn-on and turn-off switching losses as a function of load current for a range of device temperatures.


Switching Losses vs. Load Current per Device Temperature
Loss per Switching Event (J)
On-state Collector Current (A)
T = 25°C, turn-on
T = 125°C, turn-on
T = 25°C, turn-off
T = 125°C, turn-off
0.018
0.016
0.014
0.012
0.01
0.008
0.006
0.004
0.002
0
50
100
150
200
250

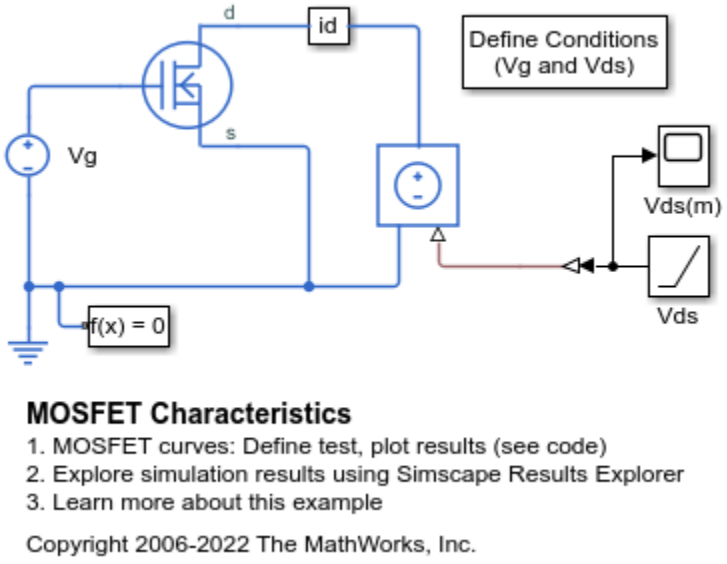
# MOSFET Characteristics

This example shows generation of the characteristic curves for an N-channel MOSFET. Define the vector of gate voltages and minimum and maximum drain-source voltages by double clicking on the block labeled 'Define Conditions (Vg and Vds)'. Then click on hyperlink 'plot results' in the model.

This type of plot can be compared against a manufacturer datasheet to confirm a correct implementation of the MOSFET parameters. You can also use this model to examine the MOSFET characteristics in the reverse region by specifying a range of negative Vds values.

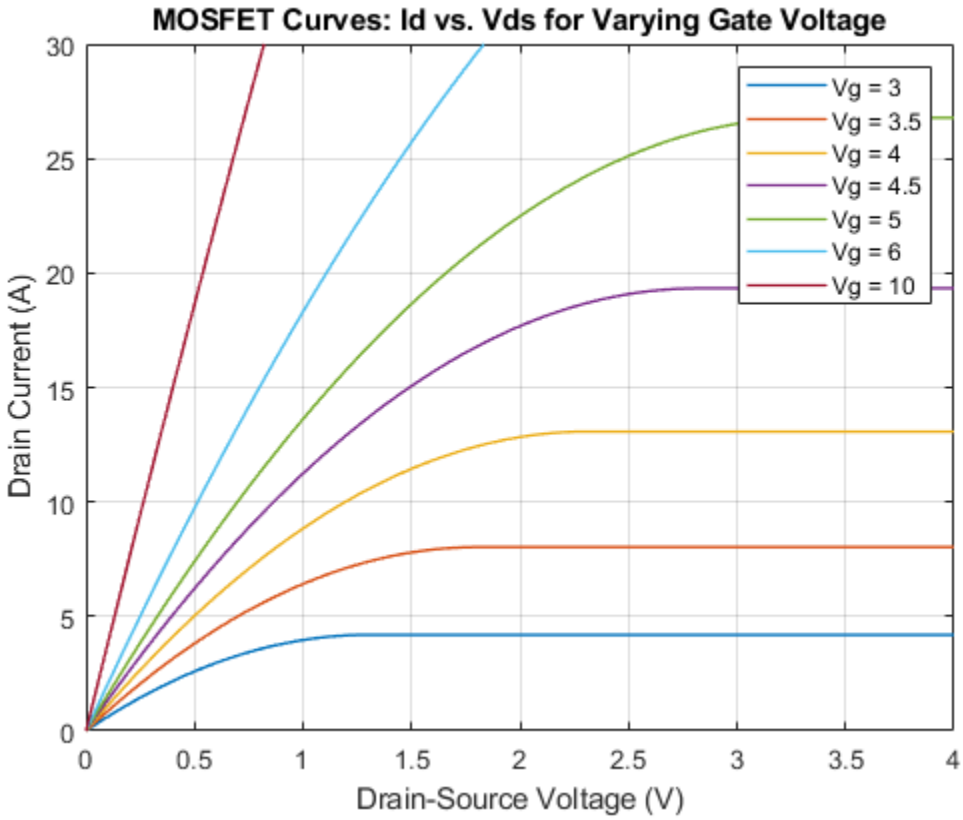
To explore the properties of a P-channel MOSFET, copy a P-Channel MOSFET from the library to replace the N-channel device, taking care to swap over the two output connections so that the source is still connected to ground. To specify normal operation, the vector of gate voltages should be negative, and the Vds range of values should be negative.

### Model




### Simulation Results from Simscape Logging

The plot below shows drain current vs. drain-source voltage for a range of gate voltages.


MOSFET Curves: Id vs. Vds for Varying Gate Voltage
Drain Current (A)
Drain-Source Voltage (V)
Vg = 3
Vg = 3.5
Vg = 4
Vg = 4.5
Vg = 5
Vg = 6
Vg = 10
0
5
10
15
20
25
30
0
0.5
1
1.5
2
2.5
3
3.5
4

# Nonlinear Inductor Characteristics

This example shows a comparison of nonlinear inductor behavior for different parameterizations. Starting with fundamental parameter values, the parameters for linear and nonlinear representations are derived. These parameters are then used in a Simscape™ model and the simulation outputs compared.

### Specification of Parameters

Fundamental parameter values used as the basis for subsequent calculations:

- Permeability of free space, $\mu_0, \mathrm{H/m}$
- Relative permeability of core, $\mu_r$
- Number of winding turns, $N_w$
- Effective magnetic core length, $l_e, \mathrm{m}$
- Effective magnetic core cross-sectional area, $A_e, \mathrm{m^2}$
- Core saturation begins, $B_{sat_{begin}}, \mathrm{T}$
- Core fully saturated, $B_{sat}, \mathrm{T}$

```
mu_0 = pi*4e-7;
mu_r = 3000;
Nw = 10;
le = 0.032;
Ae = 1.6e-5;
B_sat_begin = 0.75;
B_sat = 1.5;
```

### Calculate Magnetic Flux Density and Magnetic Field Strength Data

Where:

- Magnetic flux density, $B, \mathrm{T}$
- Magnetic field strength, $H, \mathrm{A/m}$

Linear representation:

- $B = \mu_0 \mu_r H$

Nonlinear representation (including coefficient, a):

- $B = B_{sat} \tanh(a.H)$

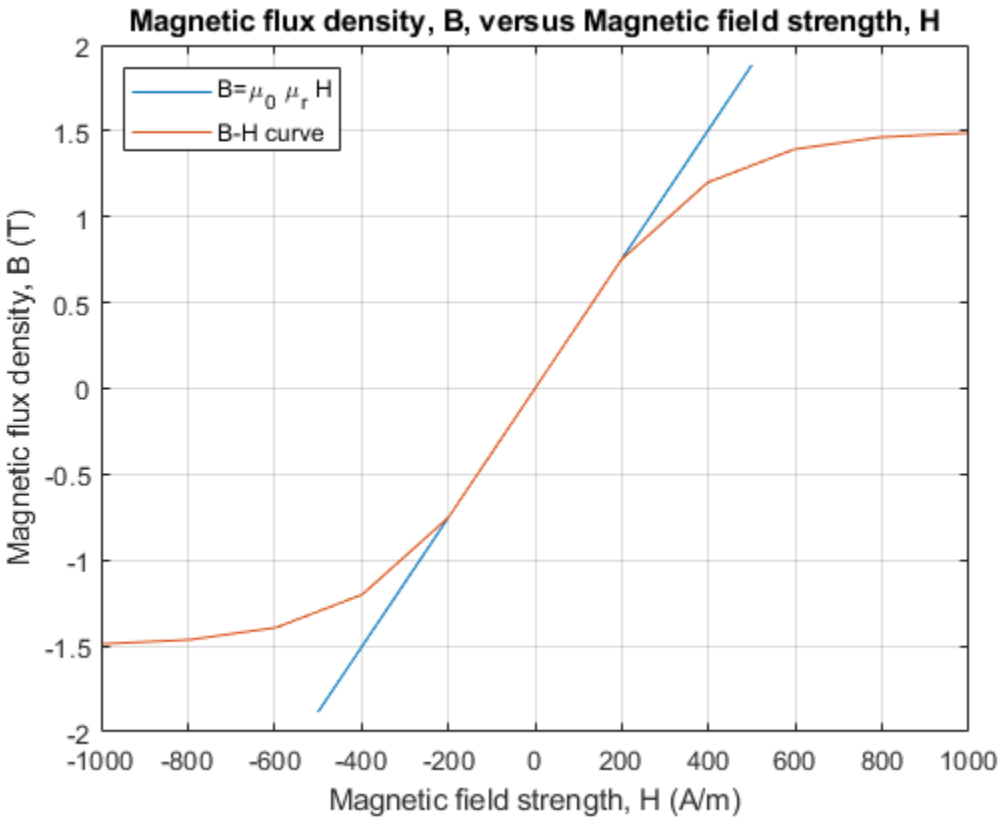
```
% Use linear representation to find value of H corresponding to B_sat_begin
H_sat_begin = B_sat_begin/(mu_0*mu_r);
% Rearrange nonlinear representation to calculate coefficient, a
a = atanh( B_sat_begin/B_sat )/H_sat_begin;

% Linear representation
H_linear = [-500 500];
B_linear = mu_0*mu_r*H_linear;
```

```
% Nonlinear representation
H_nonlinear = -5*H_sat_begin:H_sat_begin:5*H_sat_begin;
B_nonlinear = B_sat*tanh(a*H_nonlinear);
```

### Display Magnetic Flux Density Versus Magnetic Field Strength

The linear and nonlinear representations can be overlaid.

```
figure,plot( H_linear, B_linear, H_nonlinear, B_nonlinear );
grid( 'on' );
title( 'Magnetic flux density, B, versus Magnetic field strength, H' );
xlabel( 'Magnetic field strength, H (A/m)' );
ylabel( 'Magnetic flux density, B (T)' );
legend( 'B=\mu_0 \mu_r H', 'B-H curve', 'Location', 'NorthWest' )
```




### Calculate Magnetic Flux and Current Data

Where:

- Magnetic flux, $\phi, \mathrm{Wb}$
- Current, $I, \mathrm{A}$

Linear representation:

- $L = \mu_0 \mu_r A_e N_w^2 / l_e$

- $\phi = IL/N_w$

Nonlinear representation:

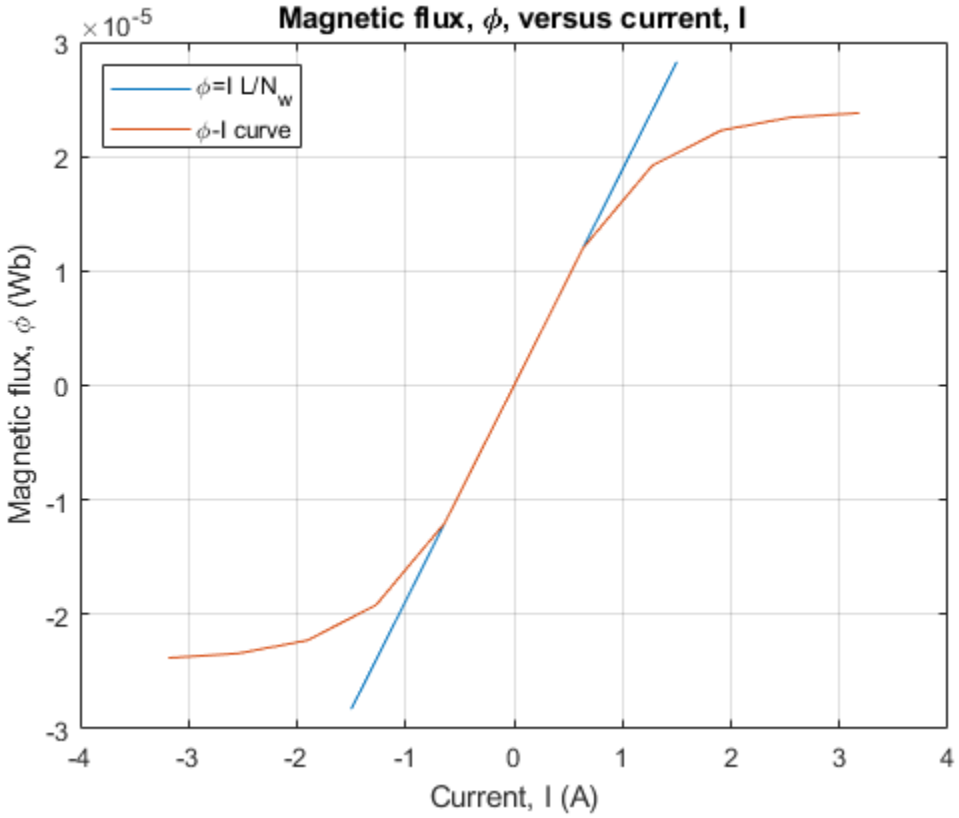
- $I = Hl_e/N_w$
- $\phi = BA_e$

```
% Linear inductance
L_linear = mu_0*mu_r*Ae*Nw^2/le;

% Linear representation
I_linear = [-1.5 1.5];
phi_linear = I_linear.*L_linear/Nw;

% Nonlinear representation
I_nonlinear = le.*H_nonlinear./Nw;
phi_nonlinear = B_nonlinear.*Ae;
```

**Display Magnetic Flux Versus Current**

The linear and nonlinear representations can be overlaid.

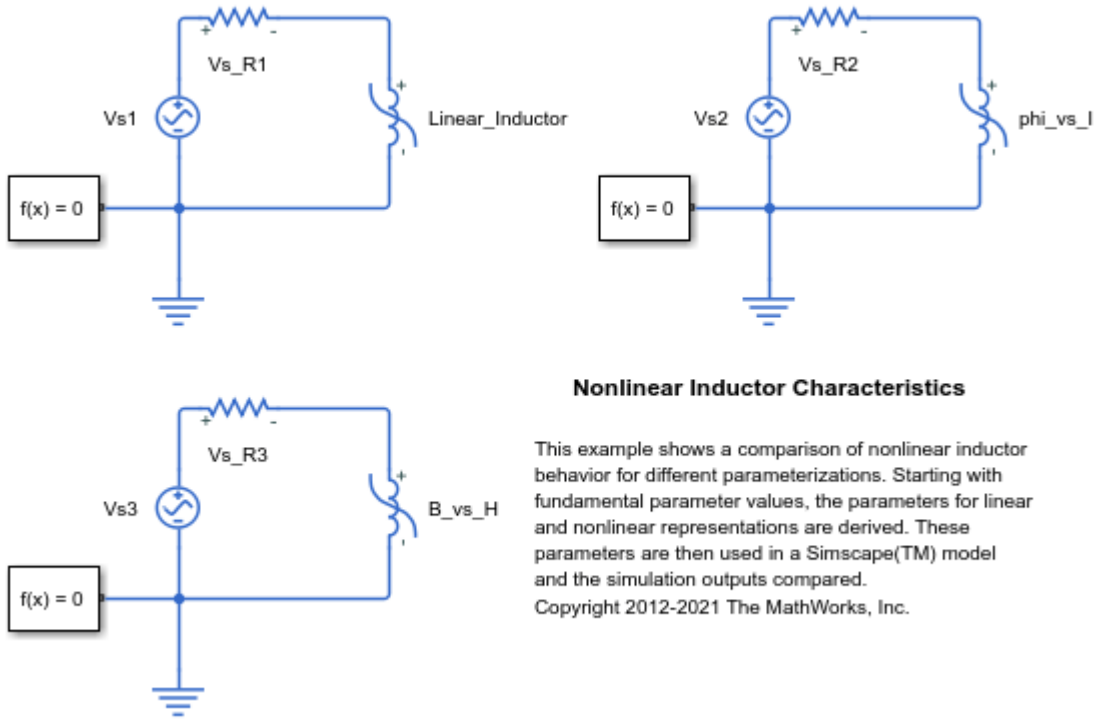
```
figure, plot( I_linear, phi_linear, I_nonlinear, phi_nonlinear );
grid( 'on' );
title( 'Magnetic flux, \phi, versus current, I' );
xlabel( 'Current, I (A)' );
ylabel( 'Magnetic flux, \phi (Wb)' );
legend( '\phi=I L/N_w', '\phi-I curve', 'Location', 'NorthWest' );
```




### Use Parameters in Simscape Model

The parameters calculated can now be used in a Simscape model. Once simulated, the model is set to create a Simscape logging variable, simlog.

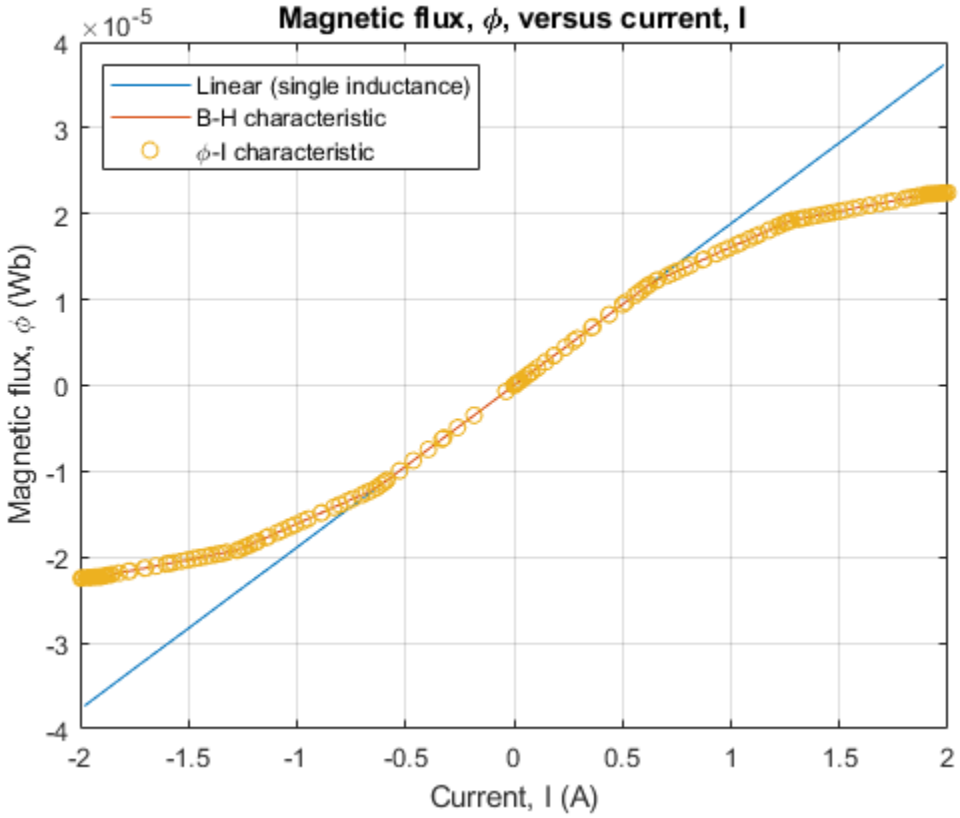
```
modelName = 'ee_nonlinear_inductor';
open_system( modelName );
sim( modelName );
```




### Conclusion

The state variable for all representations is magnetic flux, $\phi$. Current, I, and magnetic flux, $\phi$, can be obtained from the Simscape logging variable, simlog, for each representation. Overlaying the simulation results from the representations permits direct comparison.

```
figure, plot( ...
    simlog.Linear_Inductor.inductor.i.series.values,...
    simlog.Linear_Inductor.inductor.phi.series.values,...
    simlog.B_vs_H.inductor.i.series.values,...
    simlog.B_vs_H.inductor.phi.series.values,...
    simlog.phi_vs_I.inductor.i.series.values,...
    simlog.phi_vs_I.inductor.phi.series.values,...
    'o'...
    );
grid( 'on' );
title( 'Magnetic flux, \phi, versus current, I' );
xlabel( 'Current, I (A)' );
ylabel( 'Magnetic flux, \phi (Wb)' );
legend( 'Linear (single inductance)', 'B-H characteristic', '\phi-I characteristic', 'Location',
```


Magnetic flux, $\phi$, versus current, I
$\times 10^{-5}$
Linear (single inductance)
B-H characteristic
$\phi$-I characteristic
Magnetic flux, $\phi$ (Wb)
Current, I (A)
4
3
2
1
0
-1
-2
-3
-4
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2

# Inductor with Hysteresis

This example shows how modifying the equation coefficients of the Jiles-Atherton magnetic hysteresis equations affects the resulting B-H curve. The simulation parameters are configured to run four complete AC cycles with initial field strength (H) and magnetic flux density (B) both set to zero.

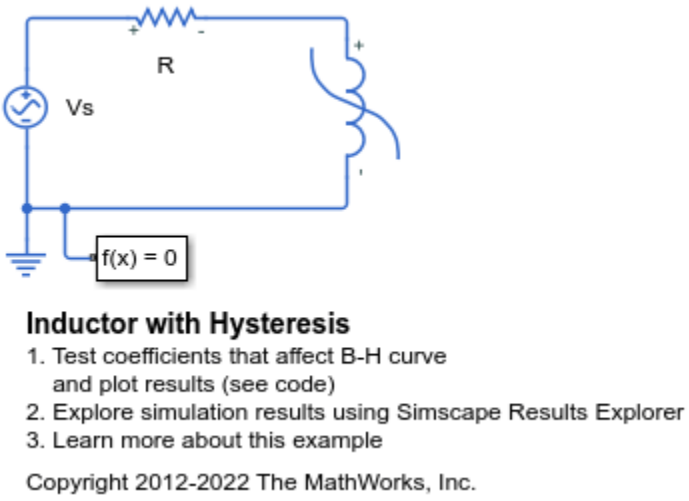
Parameters that set the shape of the anhysteretic curve are not perturbed as selecting values for these parameters is relatively easy. The remaining three parameters affect the B-H curve in multiple ways, and some iteration is typically needed to match a nominal B-H curve. The following steps are a good starting point:

1. Tune c to match the initial gradient when starting at B=H=0. As c approaches 1 the gradient will match that of the anhysteretic curve. Making it smaller reduces the initial gradient.

2. Tune K to get the desired H-axis intercepts. A good initial guess for K is the actual value of the desired intercept.

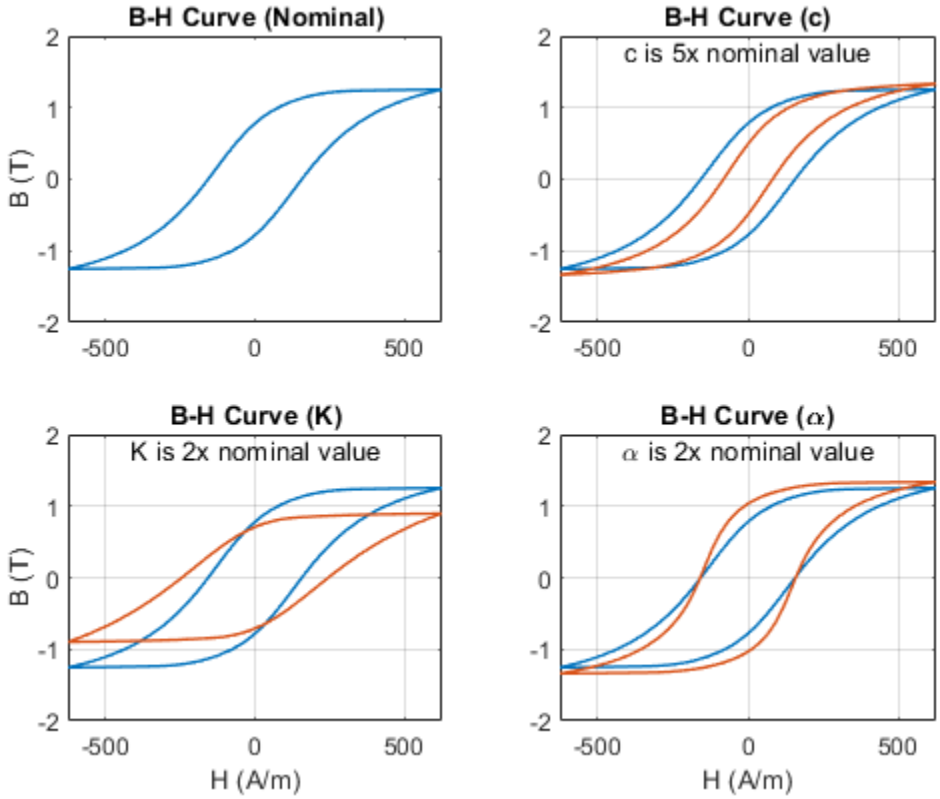
3. Gradually increase alpha (starting from a value like 1e-6) to fine tune the B-axis intercepts. Making alpha bigger increases the intercept values.

### Model




### Simulation Results from Simscape Logging

The plots below show how the individual Jiles-Atherton hysteresis coefficients affect the hysteresis curve for a nonlinear inductor. The model is simulated with a nominal set of Jiles-Atherton hysteresis equation coefficients, and then re-runs the model with perturbations applied to each coefficient individually.


B-H Curve (Nominal)
B (T)
2
1
0
-1
-2
-500
0
500
B-H Curve (c)
c is 5x nominal value
B-H Curve (K)
K is 2x nominal value
H (A/m)
B-H Curve (α)
α is 2x nominal value

# Nonlinear Transformer Characteristics

This example shows calculation and confirmation of a nonlinear transformer core magnetization characteristic. Starting with fundamental parameter values, the core characteristic is derived. This is then used in a Simscape™ model of an example test circuit which can be used to plot the core magnetization characteristic on an oscilloscope. Model outputs are then compared to the known values.

### Specification of Parameters

Fundamental parameter values used as the basis for subsequent calculations:

- Permeability of free space, $\mu_0, \mathrm{H/m}$
- Relative permeability of core, $\mu_r$
- Number of primary turns, $N_1$
- Number of secondary turns, $N_2$
- Effective magnetic core length, $l_e, \mathrm{m}$
- Effective magnetic core cross-sectional area, $A_e, \mathrm{m^2}$
- Core saturation begins, $B_{sat_{begin}}, \mathrm{T}$
- Core fully saturated, $B_{sat}, \mathrm{T}$

```
mu_0 = pi*4e-7;
mu_r = 3000;
N1 = 23;
N2 = 29;
le = 0.032;
Ae = 1.6e-5;
B_sat_begin = 0.6;
B_sat = 1.2;
```

### Calculate Magnetic Flux Density Versus Magnetic Field Strength Characteristic

Where:

- Magnetic flux density, $B, \mathrm{T}$
- Magnetic field strength, $H, \mathrm{A/m}$

Linear representation:

- $B = \mu_0 \mu_r H$

Nonlinear representation (including coefficient, a):

- $B = B_{sat} \tanh(a.H)$

```
% Use linear representation to find value of H corresponding to B_sat_begin
H_sat_begin = B_sat_begin/(mu_0*mu_r);
% Rearrange nonlinear representation to calculate coefficient, a
a = atanh( B_sat_begin/B_sat )/H_sat_begin;
```

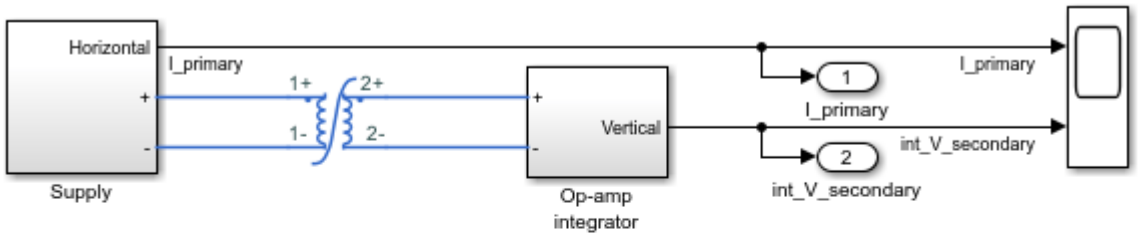
```
% Nonlinear representation
H_nonlinear = 0:25:750;
B_nonlinear = B_sat*tanh(a*H_nonlinear);
```

### Use Parameters in Simscape Model

The parameters calculated can now be used in a Simscape model. Once simulated, the model is set to output a Simscape logging variable, simlog, and some signals using output ports, yout. Circuit parameters are:

- Voltage source magnitude, $V_s = 10\text{V}$
- Voltage source frequency, $Freq_{Hz} = 60\text{Hz}$
- Voltage source resistance, $R_{Vs} = 10\Omega$
- Operational amplifier input resistance, $R_1 = 1\text{k}\Omega$
- Operational amplifier feedback resistance, $R_2 = 1\text{M}\Omega$
- Operational amplifier feedback capacitance, $C_2 = 1\mu\text{F}$

```
% Circuit parameters
Vs = 10;
Freq_Hz = 60;
R_Vs = 10;
R_1 = 1e3;
R_2 = 1e6;
C_2 = 1e-6;

% Open model and simulate
modelName = 'ee_nonlinear_transformer';
open_system( modelName );
simOut = sim( modelName );
yout = get(simOut, 'yout');
simlog = get(simOut, 'simlog');

% Collect internal Simscape logging data for comparison
I_simscape = simlog.Nonlinear_Transformer.Lm.i.series.values;
phi_simscape = simlog.Nonlinear_Transformer.Lm.phi.series.values;

% Collect model output data for comparison (as used for oscilloscope)
I_primary = yout(:,1);
int_V_secondary = yout(:,2);
```




**Nonlinear Transformer Characteristics**

This example shows calculation and confirmation of a nonlinear transformer core magnetization characteristic. Starting with fundamental parameter values, the core characteristic is derived. This is then used in a Simscape(TM) model of an example test circuit which can be used to plot the core magnetization characteristic on an oscilloscope. Model outputs are then compared to the known values.

Copyright 2012-2021 The MathWorks, Inc.

### Calculations on Logging and Output Data

The data needs to be processed to provide the Magnetic field strength and Magnetic flux density data for comparison. Where:

- Magnetomotive force, $F_m, \mathrm{A}$
- Magnetic flux, $\phi, \mathrm{Wb}$
- Operational amplifier input voltage, $V_{in}, \mathrm{V}$
- Operational amplifier output voltage, $V_{out}, \mathrm{V}$

Equations to be used are as follows:

- $F_m = I.N_1$
- $H = F_m / l_e$
- $B = \phi / A_e$
- $V_{out} = \frac{1}{R_1 C_2} \int V_{in}.dt + c$
- $\phi = \frac{1}{N_2} \int v.dt$

```
% Internal logging data
H_simscape = I_simscape.*N1./le;
B_simscape = phi_simscape./Ae;

% Oscilloscope scaling and model output data
H_measured = I_primary.*N1./le;
phi_measured = (int_V_secondary.*R_1.*C_2)./N2;
B_measured = phi_measured./Ae;
```

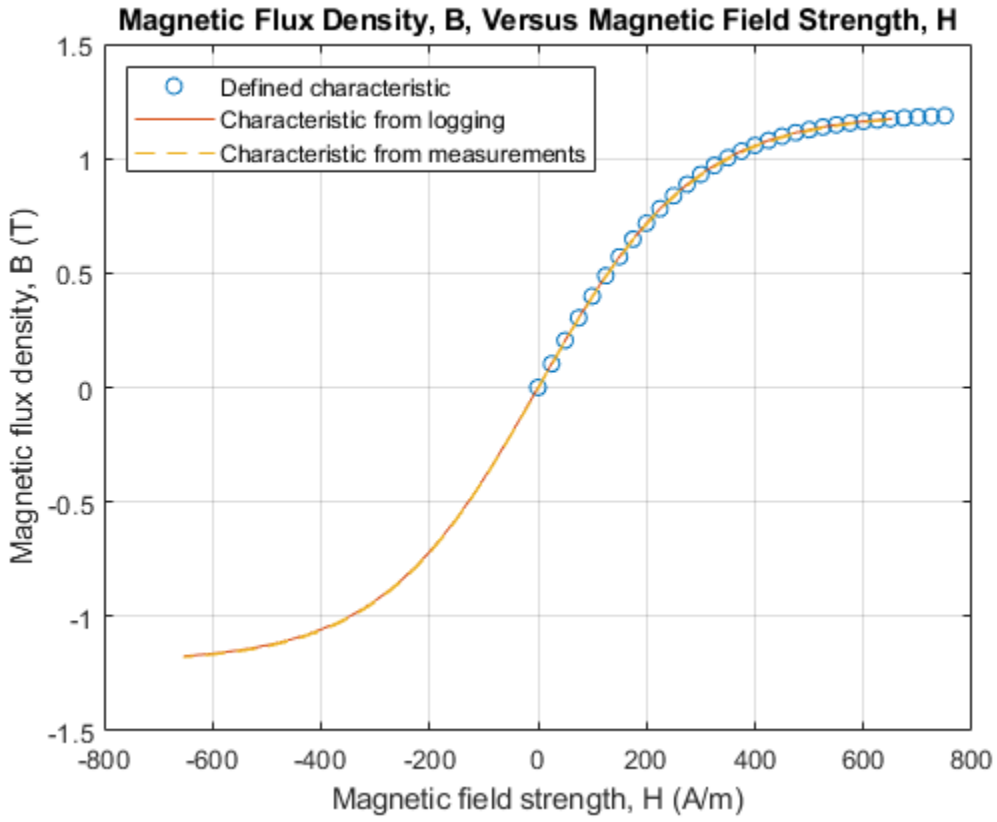
### Conclusion

The three characteristics can now be overlaid:

- Defined characteristic: calculated from fundamental parameters
- Characteristic from logging: calculated from internal Simscape logging data
- Characteristic from measurement: obtained by measurement and calculation using electronic test circuit

Due to leakage and parasitic parameters, the characteristic obtained from the electronic test circuit differs to the defined characteristic. However, the test circuit and its parameterization is shown to find the characteristic for the given transformer within suitable tolerances.

```
figure, plot( ...
    H_nonlinear,...
    B_nonlinear,...
    'o',...
    H_simscape,...
    B_simscape,...
    H_measured,...
    B_measured,...
    '--'...
    );
grid( 'on' );
title( 'Magnetic Flux Density, B, Versus Magnetic Field Strength, H' );
xlabel( 'Magnetic field strength, H (A/m)' );
ylabel( 'Magnetic flux density, B (T)' );
legend( 'Defined characteristic', 'Characteristic from logging',...
    'Characteristic from measurements', 'Location', 'NorthWest' );
```



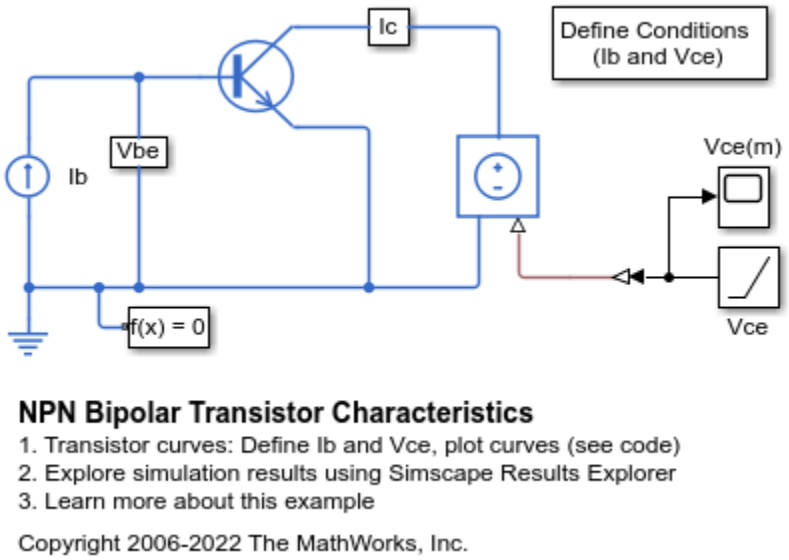
# NPN Bipolar Transistor Characteristics

This example shows generation of the Ic versus Vce curve for an NPN bipolar transistor. Define the vector of base currents and minimum and maximum collector-emitter voltages by double clicking on the block labeled 'Define Conditions (Ib and Vce)'. Run the tests and generate plots of the curves by clicking in the model on hyperlink 'plot curves'.

This type of plot can be compared against a manufacturer datasheet to confirm a correct implementation of the transistor parameters. You can also use this model to examine the transistor characteristics in the reverse region by specifying a range of negative Vce values. In this region, the gain is defined by the Reverse current transfer ratio BR parameter. Increase this parameter above one to produce a reverse current gain.

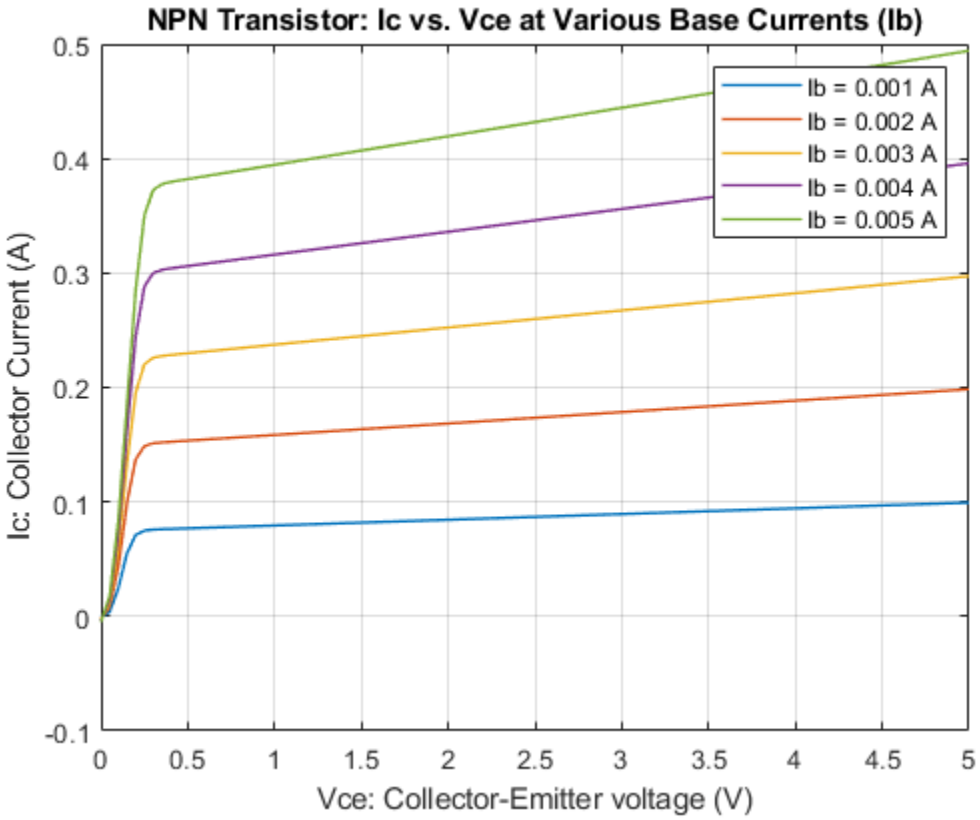
To explore the properties of a PNP bipolar transistor, open model ee_pnp.

### Model




### Simulation Results from Simscape Logging

The plot below shows collector current (Ic) versus collector-emitter voltage (Vce) characteristics for different levels of base current (Ib).


NPN Transistor: Ic vs. Vce at Various Base Currents (Ib)
Ic: Collector Current (A)
Vce: Collector-Emitter voltage (V)
Ib = 0.001 A
Ib = 0.002 A
Ib = 0.003 A
Ib = 0.004 A
Ib = 0.005 A
0.5
0.4
0.3
0.2
0.1
0
-0.1
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5

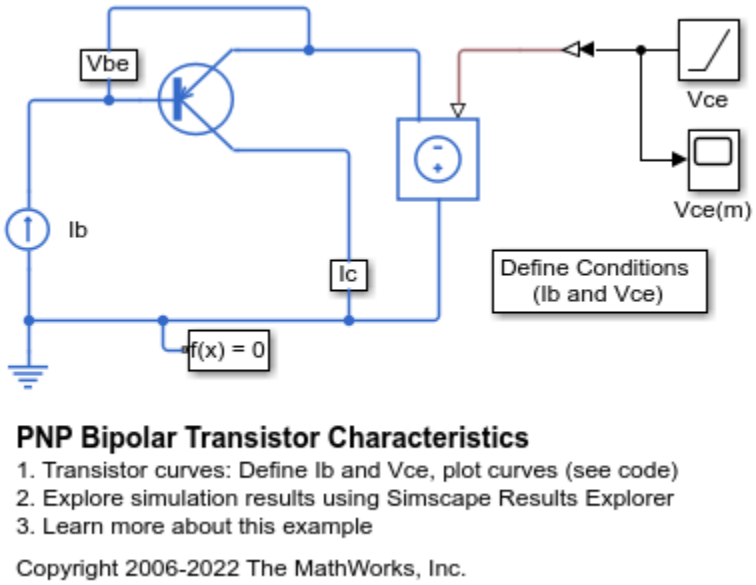
# PNP Bipolar Transistor Characteristics

This example shows generation of the Ic versus Vce curve for a PNP bipolar transistor. Define the vector of base currents and minimum and maximum collector-emitter voltages by double clicking on the block labeled 'Define Conditions (Ib and Vce)'. Run the tests and generate plots of the curves by clicking in the model on hyperlink 'plot curves'.

This type of plot can be compared against a manufacturer datasheet to confirm a correct implementation of the transistor parameters. You can also use this model to examine the transistor characteristics in the reverse region by specifying a range of positive Vce values. In this region, the gain is defined by the Reverse current transfer ratio BR parameter. Increase this parameter above one to produce a reverse current gain.

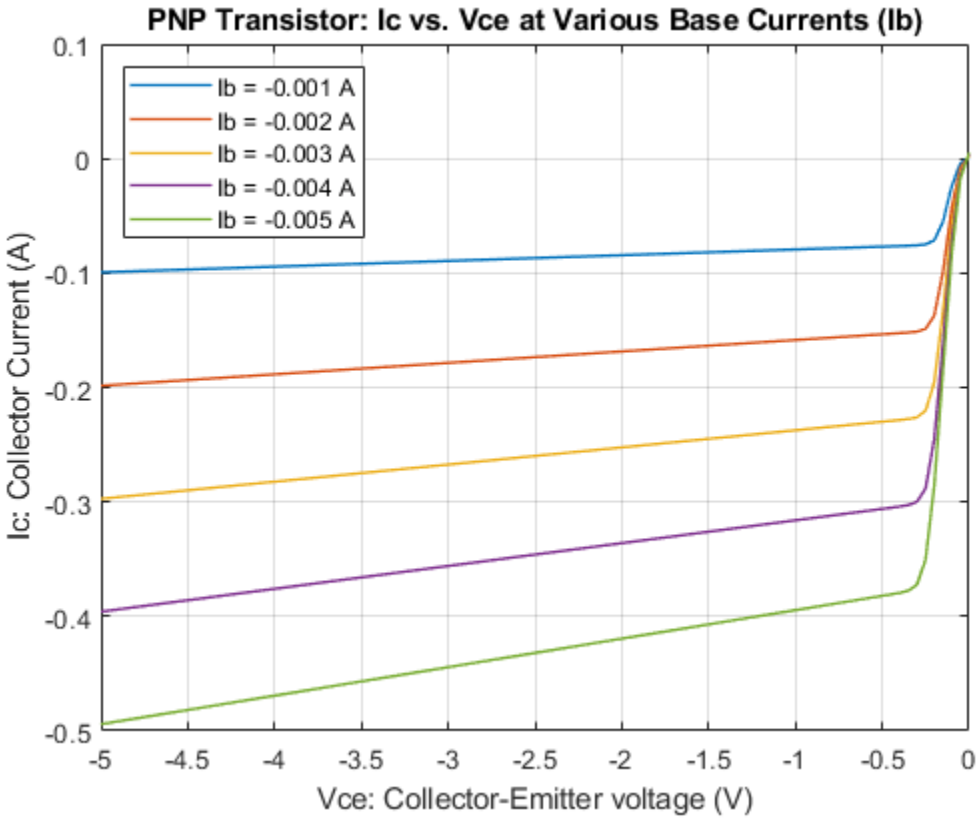
To explore the properties of an NPN bipolar transistor, open model ee_npn.

### Model




### Simulation Results from Simscape Logging

The plot below shows collector current (Ic) versus collector-emitter voltage (Vce) characteristics for different levels of base current (Ib).


PNP Transistor: Ic vs. Vce at Various Base Currents (Ib)
Ib = -0.001 A
Ib = -0.002 A
Ib = -0.003 A
Ib = -0.004 A
Ib = -0.005 A
Ic: Collector Current (A)
Vce: Collector-Emitter voltage (V)
0.1
0
-0.1
-0.2
-0.3
-0.4
-0.5
-5
-4.5
-4
-3.5
-3
-2.5
-2
-1.5
-1
-0.5
0

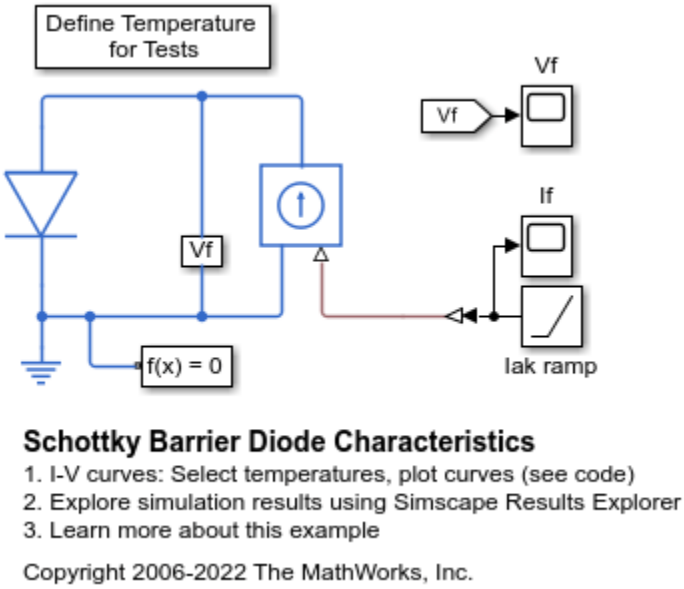
# Schottky Barrier Diode Characteristics

This example shows generation of the current versus voltage curve for a Schottky barrier diode. Define the vector of temperatures for which to plot the characteristics by double clicking on the block labeled 'Define Temperatures for Tests'. Run the tests and plot the I-V curves by clicking in the model on the hyperlink 'plot curves'.

The datasheet for this device gives Vf = 0.4V when If = 10mA, and Vf = 0.65V when If = 100mA. The ohmic resistance is set to one over the gradient of the datasheet I-V curve at higher voltages. The temperature dependence is then modeled by selecting the default energy gap and saturation current temperature exponent values for a Schottky barrier diode.

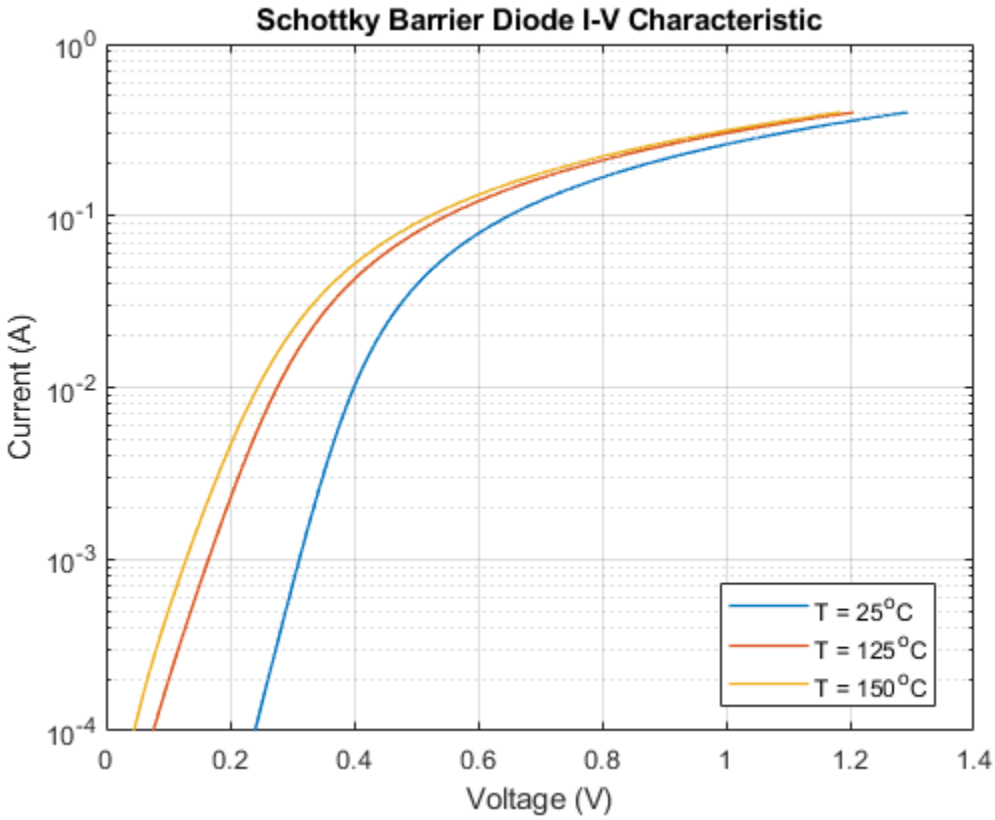
The plot produced by this test model can be used to validate the implementation against the datasheet I-V plots.

### Model




### Simulation Results from Simscape Logging

The plot below shows the I-V characteristic for a Schottkey barrier diode extracted from simulation results. The model is simulated once for each specified temperature.

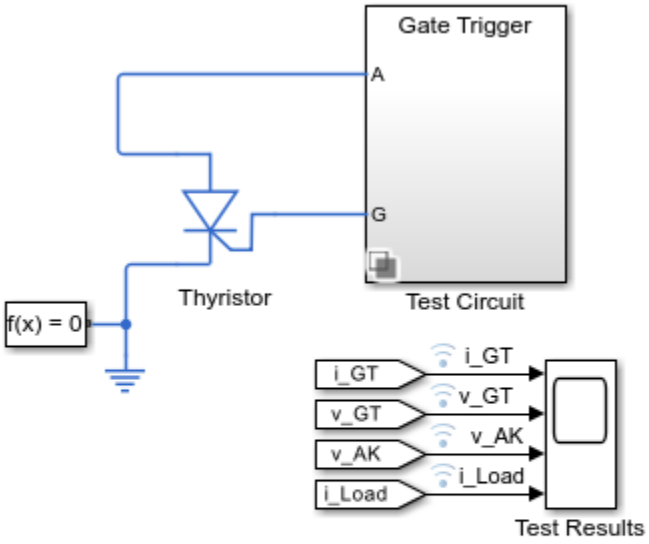

Schottky Barrier Diode I-V Characteristic
Current (A)
Voltage (V)
$10^{0}$
$10^{-1}$
$10^{-2}$
$10^{-3}$
$10^{-4}$
0
0.2
0.4
0.6
0.8
1
1.2
1.4
T = 25°C
T = 125°C
T = 150°C

# Thyristor Static Behavior Validation

This example shows validation of the static behavior of the Thyristor block against its mask values. Mask values are closely related to datasheet values, and the thyristor block uses these values to calculate the coefficients of the equations used to model it.

The model can be configured to run four separate tests: gate trigger, latch current, on state, and off state. Use the hyperlinks in the model to select the desired test. Each subsystem variant contains more information on the test that is run.

**Model**



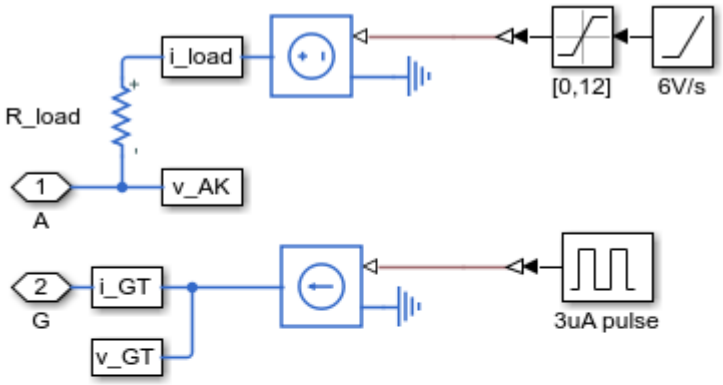

**Thyristor Static Behavior Validation**

1. Run test and plot results (see code)
2. Select Test: Gate Trigger, Latch Current, On State, Off State
3. Explore simulation results using Simscape Results Explorer
4. Learn more about this example

Copyright 2009-2022 The MathWorks, Inc.

### Simulation Results from Simscape Logging

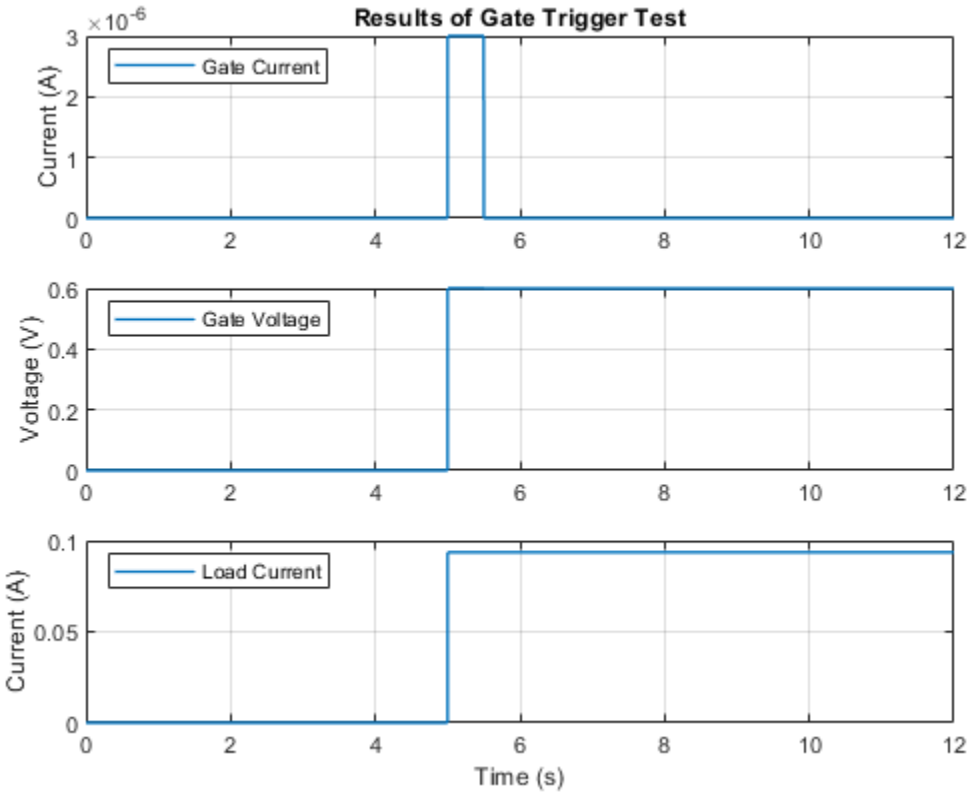
### Gate Trigger Test



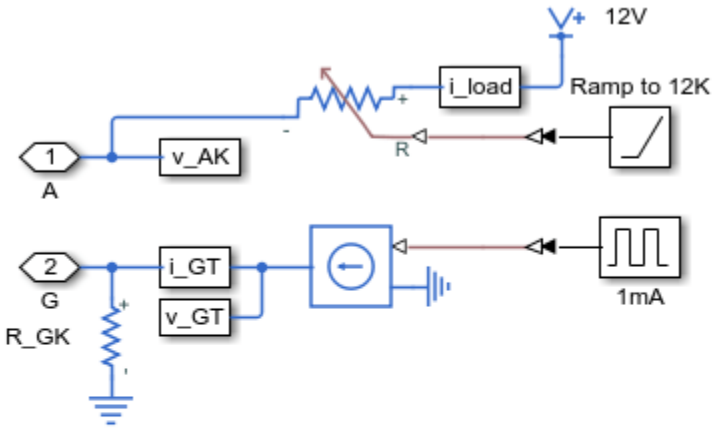

This test ensures that the thyristor turns on when a pulse of 3uA or more is applied to the gate. Specified conditions are 12V supply and 120 ohm load. The value of the internal shunt resistor Rs is critical to ensuring that the device only turns on when the pulse exceeds 3uA.

This circuit also verifies that when the gate current is 3uA, the gate-cathode voltage v_GK is 0.6V as specified in the Thyristor mask.

The supply voltage must be ramped up to 12V to prevent dV/dt gate triggering at turn-on.

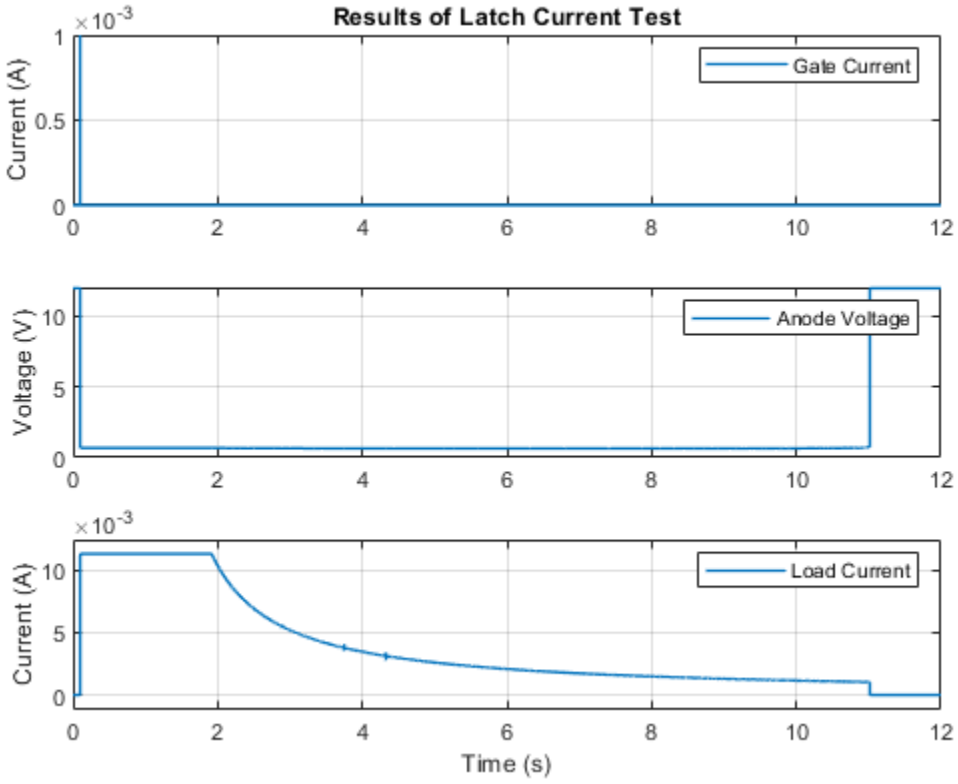

Results of Gate Trigger Test
×10^-6
Gate Current
Current (A)
Gate Voltage
Voltage (V)
Load Current
Current (A)
Time (s)

### Latch Current Test

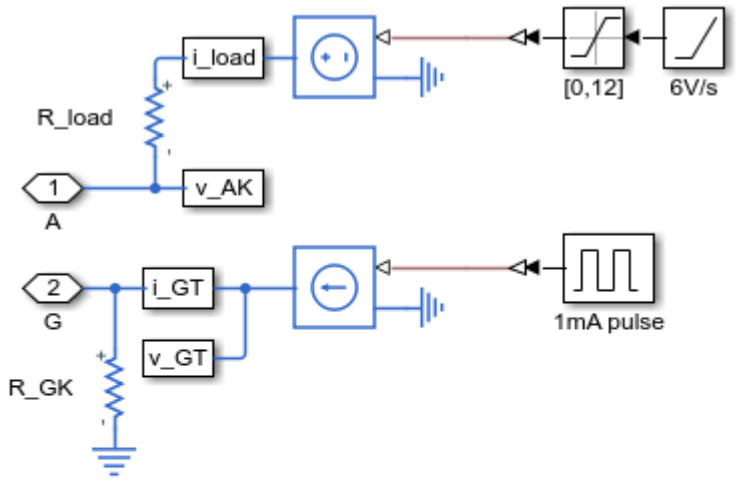



This test circuit determines the load current below which the thyristor will turn off. The Variable resistor increases between 1K and 12K during the simulation. The scope plot i_Load shows that the holding current is approximately 1mA.

The Thyristor block does not let you define a value for holding current. It is an emergent property of the other mask values. The Product of NPN and PNP forward current gains parameter on the Advanced tab has some influence over the holding current. Reducing if from 10 to 2 doubles the holding current to 2mA.

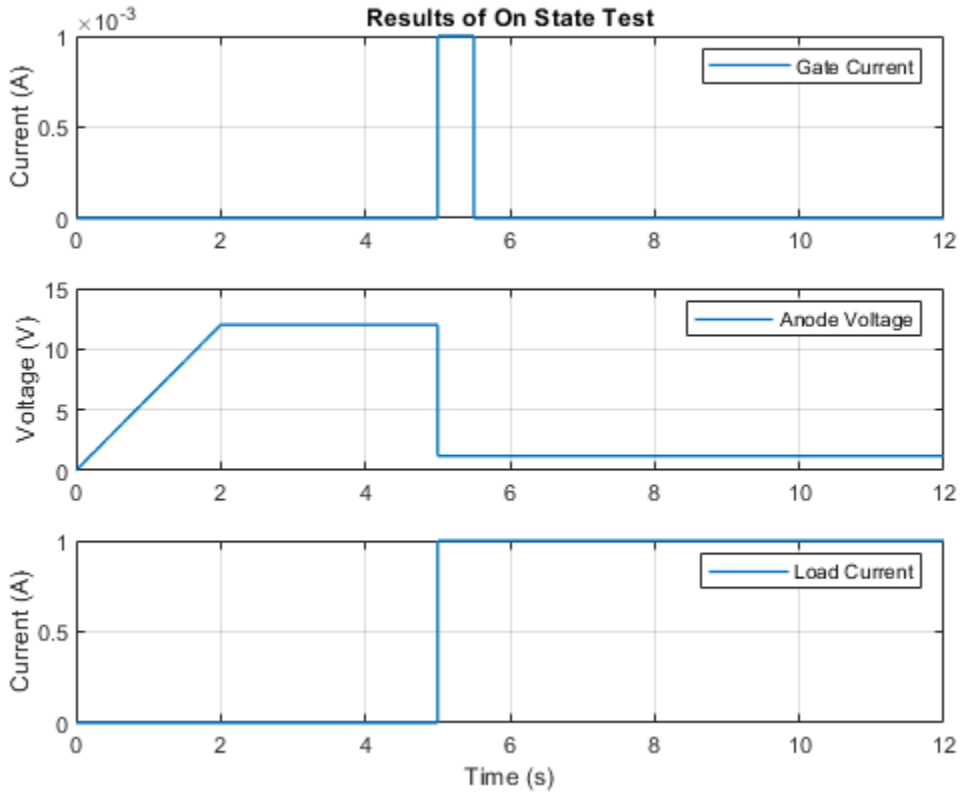

Results of Latch Current Test
×10^-3
Current (A)
Gate Current
Voltage (V)
Anode Voltage
×10^-3
Current (A)
Load Current
Time (s)

### On-State Test

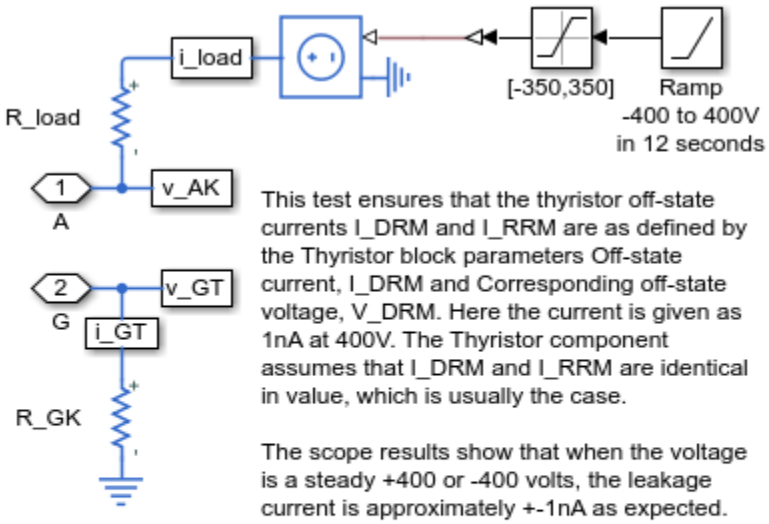



This test ensures that the thyristor on-state voltage drop is correct. The Thyristor mask parameters define the voltage drop V_T to be 1.2V when the current I_T is 1A. To test this, the load resistor must be set to (12-1.2)/1 = 10.8ohms.

Note the use of the external R_GK resistor. This reduces the triggering sensitivity, the trigger current being increased to approximately 0.6V/1000ohms = 0.6mA. The critical current will be slightly less than 0.6mA due to increased anode-gate current which contributes to the gate current.


Results of On State Test
×10^-3
Current (A)
Gate Current
Voltage (V)
Anode Voltage
Current (A)
Load Current
Time (s)

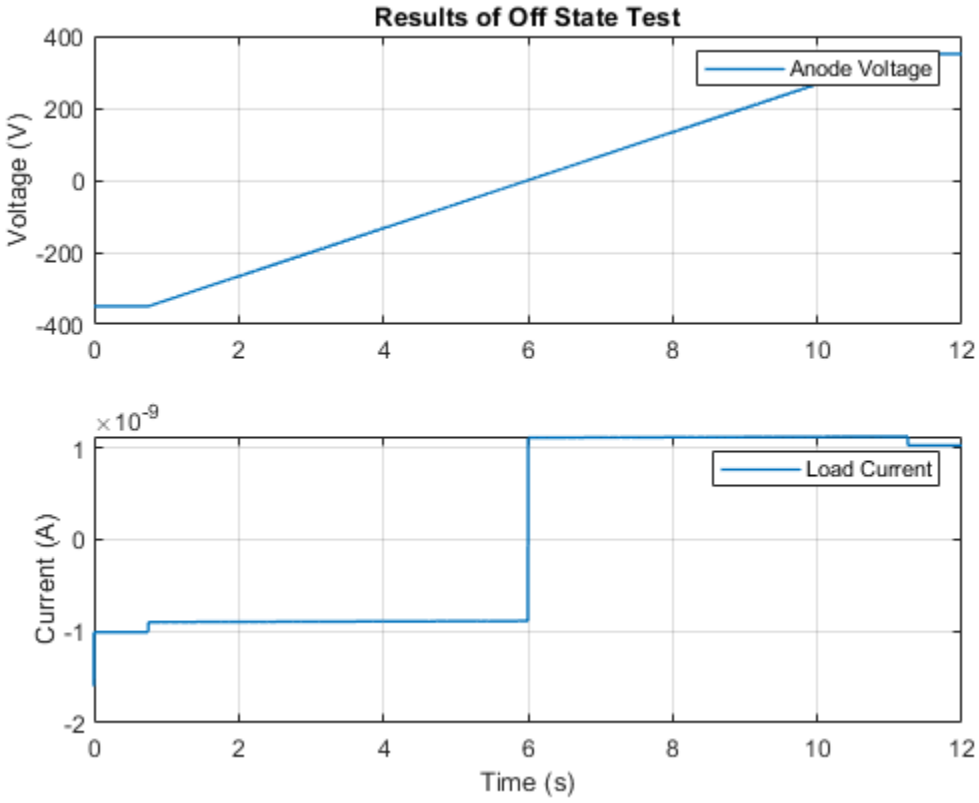
### Off-State Test




This test ensures that the thyristor off-state currents I_DRM and I_RRM are as defined by the Thyristor block parameters Off-state current, I_DRM and Corresponding off-state voltage, V_DRM. Here the current is given as 1nA at 400V. The Thyristor component assumes that I_DRM and I_RRM are identical in value, which is usually the case.

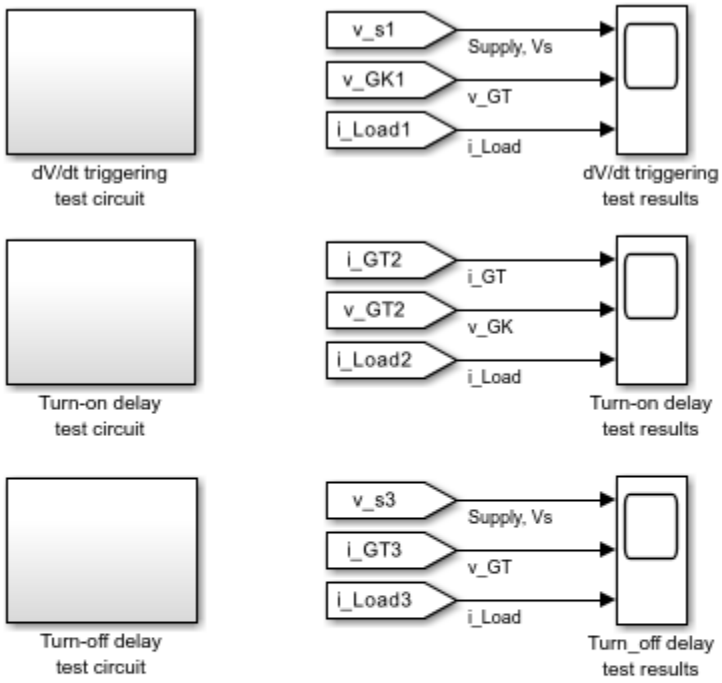
The scope results show that when the voltage is a steady +400 or -400 volts, the leakage current is approximately +-1nA as expected. The values are not exact due to approximations made when solving for the thyristor component internal parameter values.

The abrupt change in current when the voltage is ramped is due to the current required to charge the thyristor junction capacitances.


Results of Off State Test
Anode Voltage
Voltage (V)
400
200
0
-200
-400
0
2
4
6
8
10
12
×10^-9
Load Current
Current (A)
1
0
-1
-2
Time (s)

# Thyristor Dynamic Behavior Validation

This example shows validation of the dynamic behavior of the Thyristor block against its mask values. Mask values are closely related to datasheet values, and the Thyristor block uses these values to calculate the coefficients of the equations used to model it. Double-click on each of the test subsystems for further information on the tests.




**Thyristor Dynamic Behavior Validation**

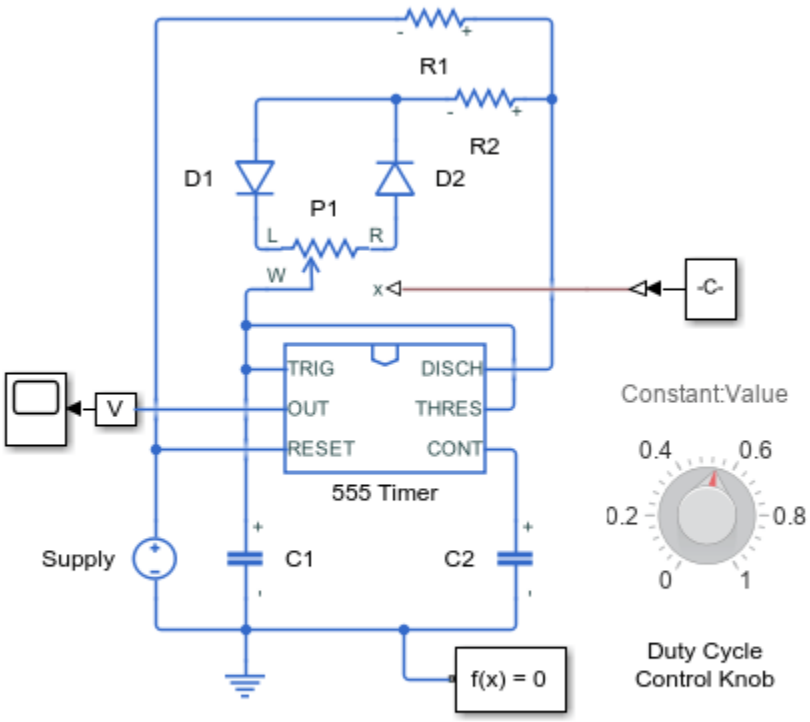
This example shows validation of the dynamic behavior of the Thyristor block against its mask values. Mask values are closely related to datasheet values, and the Thyristor block uses these values to calculate the coefficients of the equations used to model it. Double-click on each of the test subsystems for further information on the tests.

Copyright 2009-2021 The MathWorks, Inc.

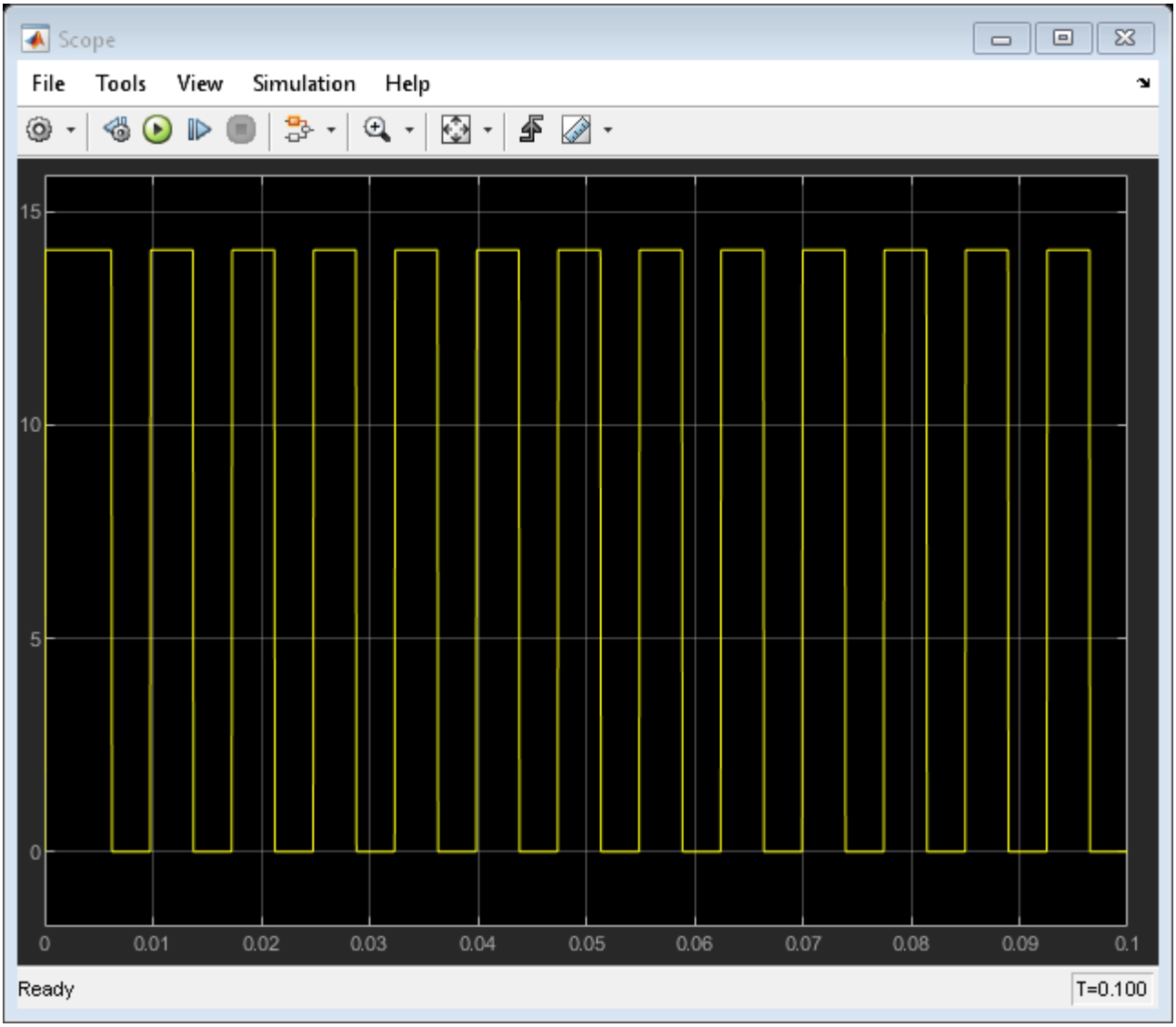
# PWM Circuit Using 555 Timer

This example shows a pulse-width-modulated (PWM) output implemented using a 555 Timer in astable mode. The duty cycle is set by a potentiometer, P1. The potentiometer is controlled during run-time via Duty Cycle Control Knob. The scope shows the resultant output from the 555 Timer. To end the simulation, click on the Stop button.

**Model**



**Simulation Results from Scopes**



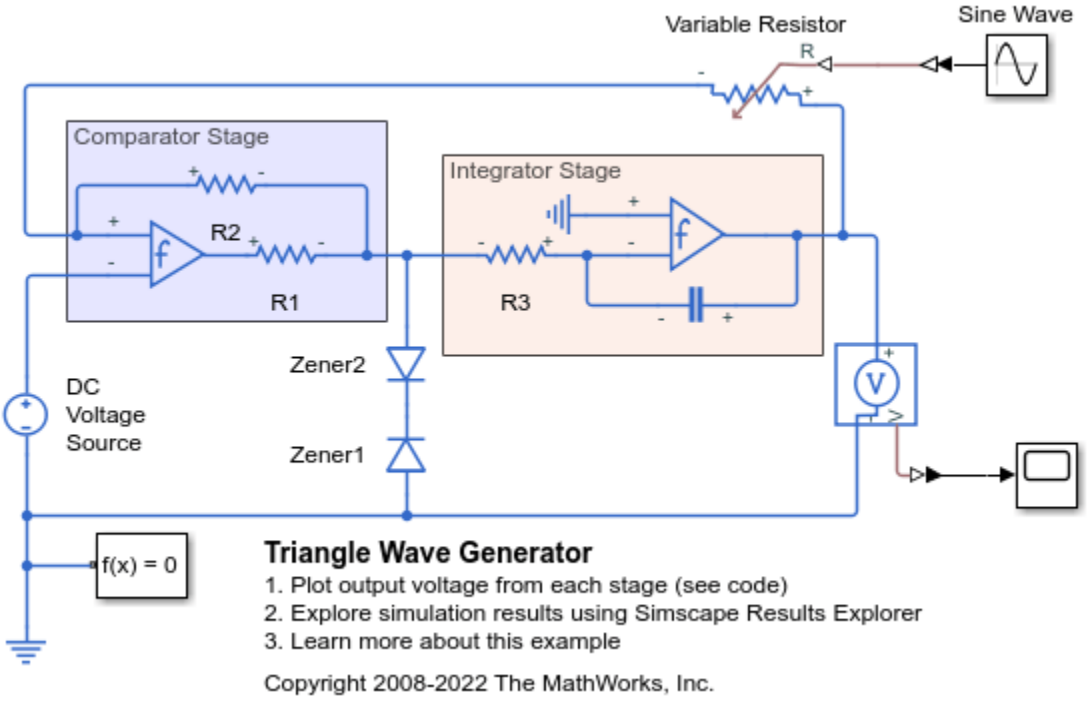
# Triangle Wave Generator

This example shows an implementation of a triangle wave generator circuit using two op-amps. The first stage of the circuit is a comparator constructed from an op-amp. The output of the comparator is limited to about plus or minus 5 volts by the two zener diodes. The limits imposed by the zener diodes result in a square wave.

The second stage of the circuit is an integrator. Integrating the square wave creates a triangle wave. The Sine Wave block modulates the waveform amplitude via the Variable Resistor block, and the DC Voltage Source can be used to add a DC offset. See the Example - Modeling a Triangle Wave Generator section of the Simscape™ Electrical™ User Guide for more information on how to construct this model using the Simscape and Simscape Electrical block libraries.

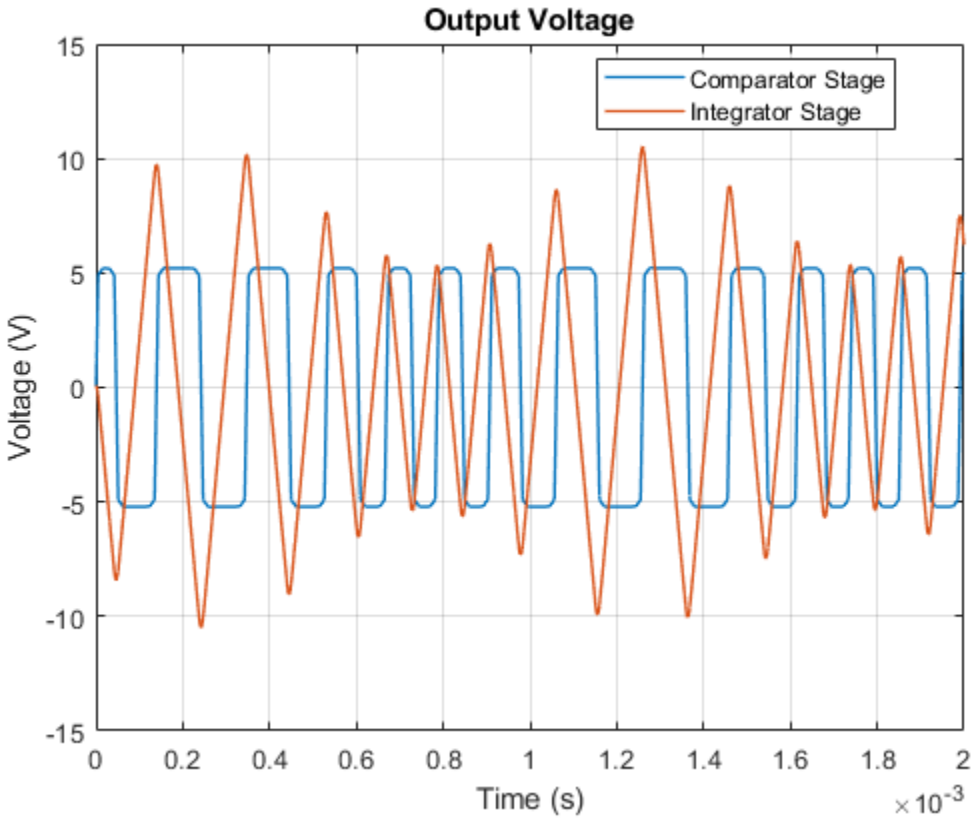
The two Band-Limited Op-Amp blocks are parameterized based on an LM7301 device. The datasheet gives the gain as 97dB which is equivalent to a gain of 10^(97/20)=7.1e4. Input resistance is given as 39e6 ohms, and slew rate as 1.25V per microsecond. Bandwidth is given as 4MHz. No value is given for the output resistance, so this is set to zero. In practice the output resistance will be small compared to output series 1000 ohm resistor in this circuit.

### Model




### Simulation Results from Simscape Logging

The plot below shows the output voltage of each stage for the triangle wave circuit. The comparator with limits imposed by the zener diodes creates a square wave. Integrating the square wave produces a triangle wave.


Output Voltage
Comparator Stage
Integrator Stage
Voltage (V)
Time (s)
15
10
5
0
-5
-10
-15
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
×10^-3

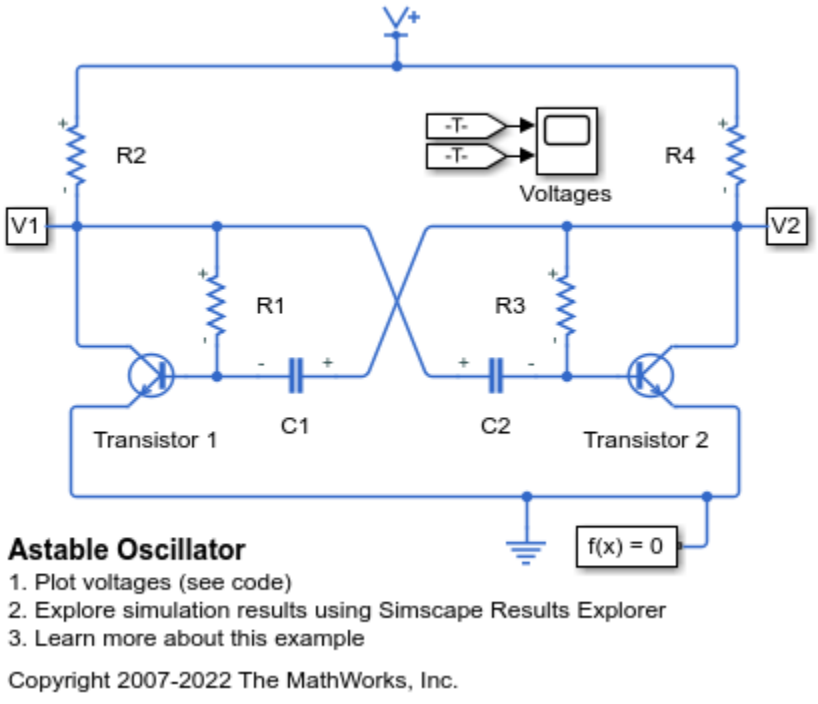
# Astable Oscillator

This example shows an implementation of an astable oscillator circuit. The circuit output voltage V2 oscillates in an unstable fashion between high and low states.

At time zero both transistors are off and the voltages across C1 and C2 are zero. C1 starts to charge via R4 and Transistor 1 base-emitter junction. Similarly C2 charges via R2 and Transistor 2's base-emitter junction. Hence voltages V1 and V2 both rise, and resistors R1 and R3 start to conduct, the current in R1 being slightly higher as R3 > R1. R1 acts to increase the potential on Transistor 1 base. Transistor 1 is therefore less saturated than Transistor 2, and V1-V2 increases. Eventually C2 becomes fully charged, and Transistor 2 base current stops, turning the transistor off. V2 then goes high, and C1 starts charging again, thereby turning on Transistor 1. When C1 becomes fully charged, Transistor 1 switches off, and the astable circuit continues to cycle between these two states.

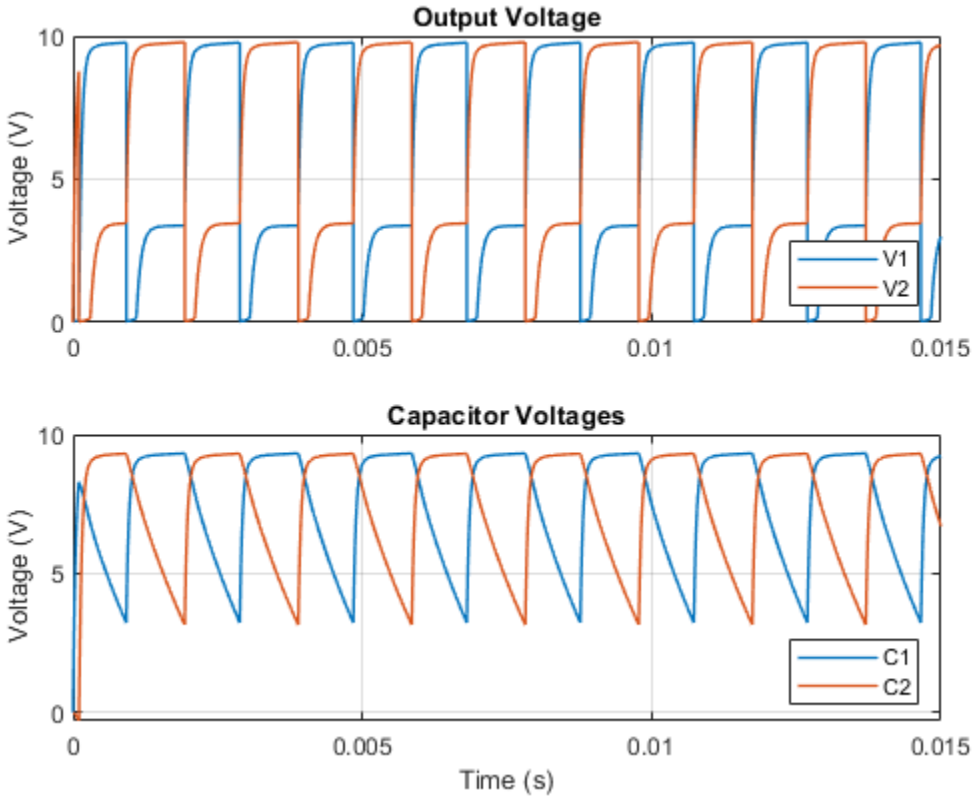
This circuit is an example of a numerically challenging circuit for network simulators. The numerical challenge is due to the exponentially increasing gradients occurring during switching. The default transistor ohmic resistance and junction capacitance terms are added to improve the numerical properties of the circuit.

### Model




### Simulation Results from Simscape Logging

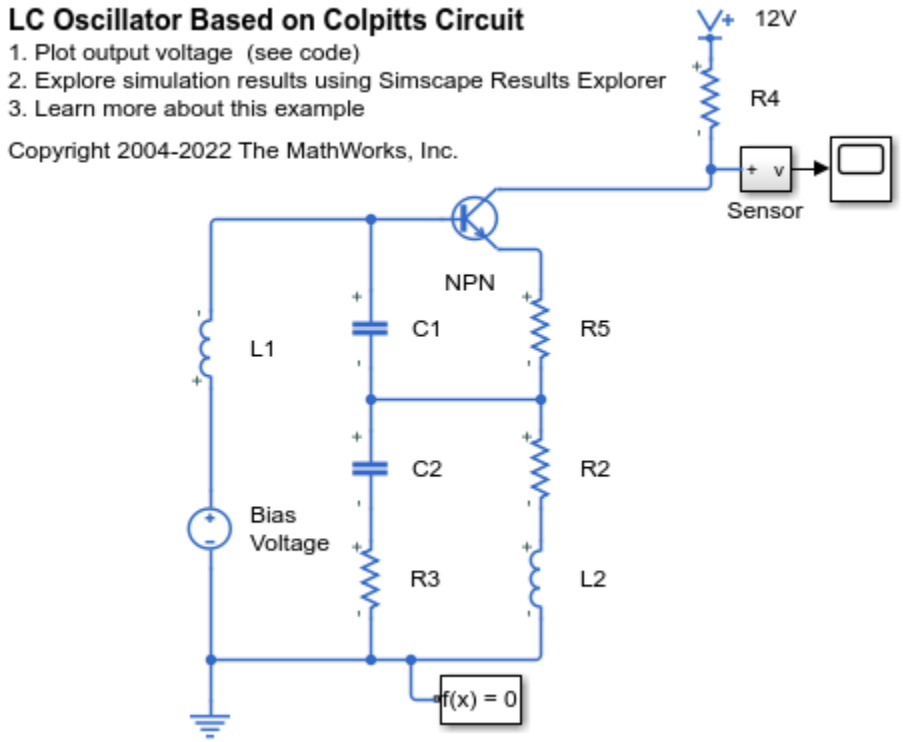
The plots below show voltages in the astable oscillator circuit. The charging and discharging of capacitors C1 and C2 turns the transistors on and off.


Output Voltage
Voltage (V)
10
5
0
0
0.005
0.01
0.015
V1
V2
Capacitor Voltages
Voltage (V)
10
5
0
0
0.005
0.01
0.015
Time (s)
C1
C2

# LC Oscillator Based on Colpitts Circuit

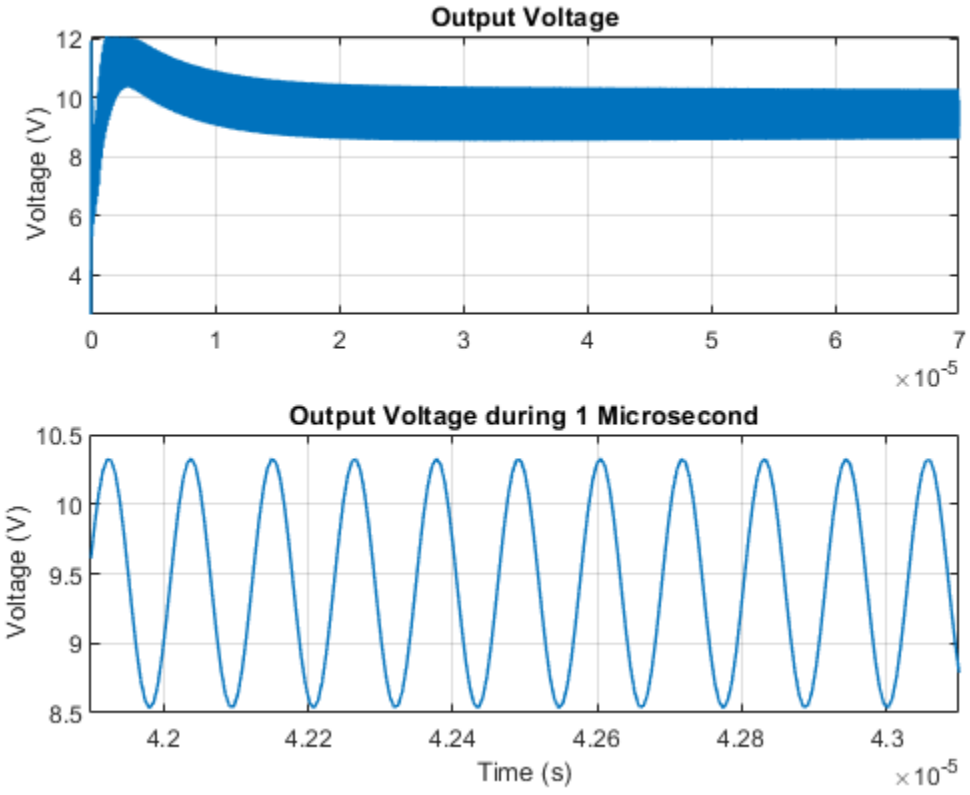
This example shows an implementation of a Colpitts oscillator circuit with a nominal frequency of 9MHz. The frequency of oscillation is given by 1/(2*pi*sqrt(L1*C1*C2/(C1+C2))). LC oscillators have good frequency selectivity due to higher Q levels than are achievable with RC oscillators.

### Model




### Simulation Results from Simscape Logging

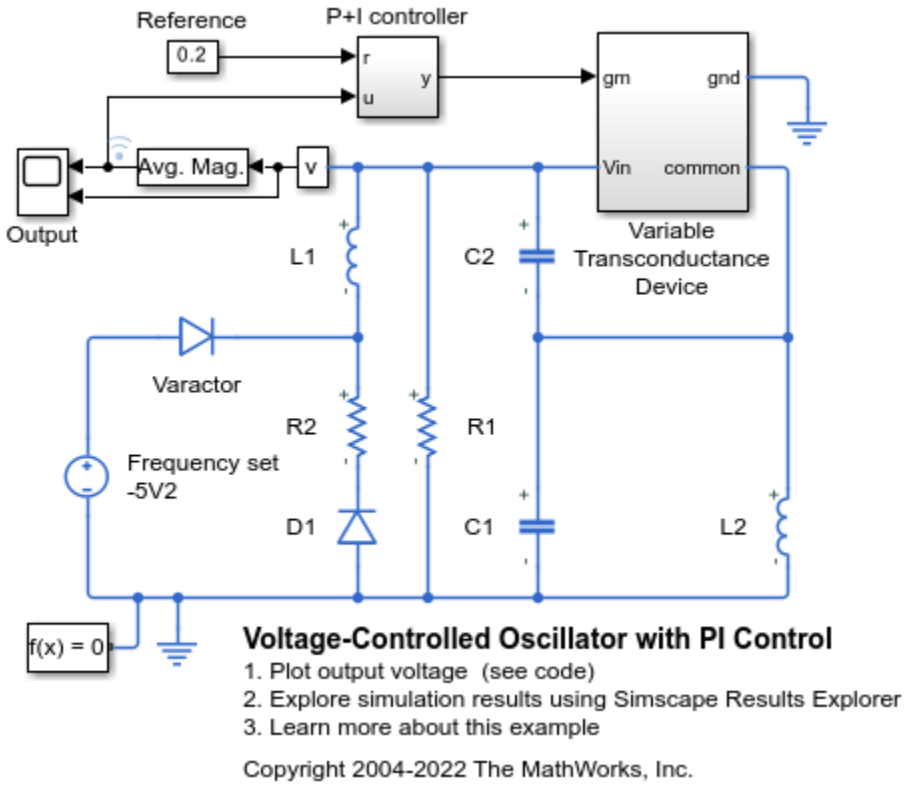
The plots below show the output voltage of the LC oscillator based on Colpitts circuit. The lower plot shows the voltage over 1 microsecond so that we can determine the frequency (around 9 MHz).


Output Voltage
Voltage (V)
12
10
8
6
4
0
1
2
3
4
5
6
7
×10^-5
Output Voltage during 1 Microsecond
Voltage (V)
10.5
10
9.5
9
8.5
4.2
4.22
4.24
4.26
4.28
4.3
Time (s)
×10^-5

# Voltage-Controlled Oscillator with PI Control

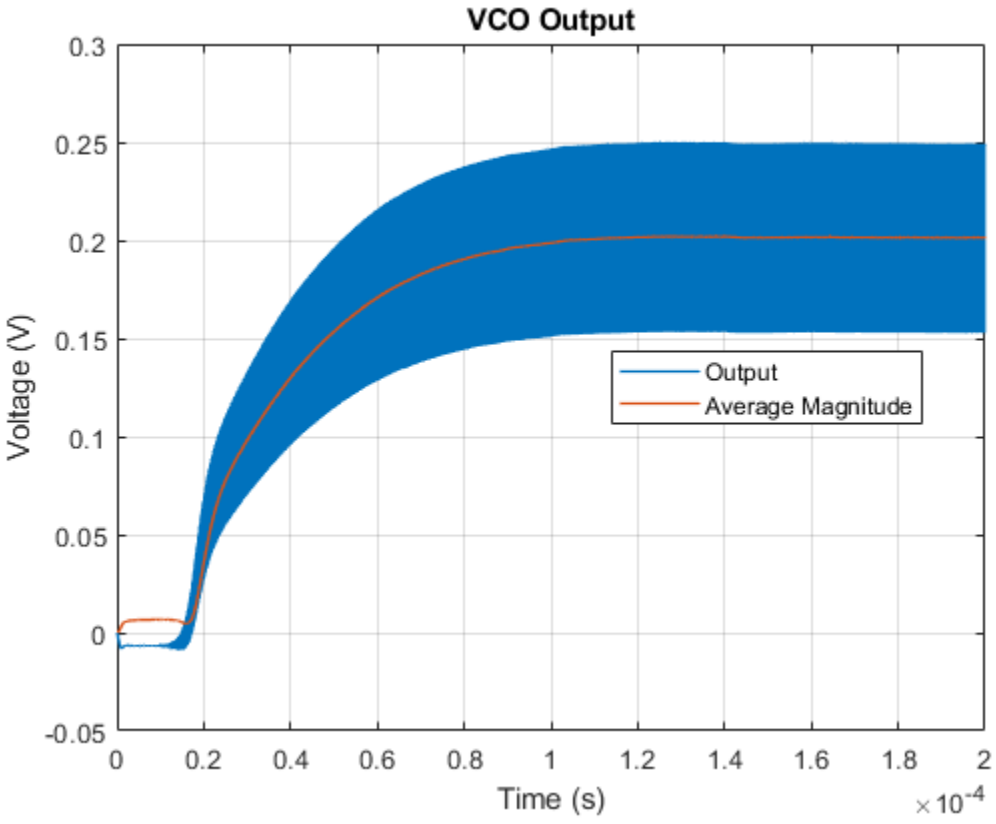
This model shows a voltage-controlled oscillator with feedback control to regulate the output voltage. The oscillation frequency is controlled by the reverse bias voltage applied to the to the varactor diode. The model shows how behavioral and component-level modeling can be mixed when designing a circuit. Provided that the variable trans-conductance device has a fast dynamic response relative to the oscillation frequency, the implementation details are not important. Similarly, the PI controller is in abstracted form, and could either be implemented with op-amps, or in software.

### Model




### Simulation Results from Simscape Logging

The plots below shows the output voltage of the voltage-controlled oscillator (vco) circuit. Both the true output and the averaged output are shown. The PI controller ensures that oscillations are centered about the reference value of 0.2 V.


VCO Output
Voltage (V)
Time (s)
Output
Average Magnitude
0.3
0.25
0.2
0.15
0.1
0.05
0
-0.05
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
×10^-4

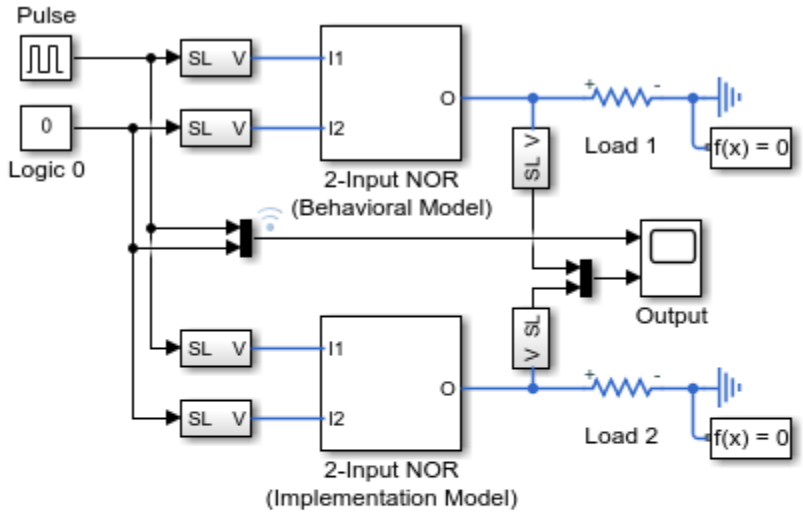
# Modeling an Integrated Circuit

This example shows two ways to create a model of an integrated circuit that can be used in conjunction with other Simscape™ Electrical™ blocks. The first approach is to build a behavioral model using Simscape Foundation Library PS blocks and/or Simulink® blocks. The second is to build an implementation model from basic Simscape Electrical blocks. To look under the masks and view the details, select the relevant block and type ctrl-U.

The behavioral model has the advantage of being simpler and potentially running faster. The approach can be extended to complex algorithms as might be implemented on a PIC or ASIC. Note that the model includes a first order lag to model the 1ns propagation delay. In many applications, this time constant will be much faster than the overall circuit dynamics, and may not be of interest. However, removing it potentially introduces an algebraic loop if the gate inputs depend in anyway on the gate output. An alternative is to implement the algorithm (the NOR gate in this case) using only PS blocks rather than Simulink blocks.

The implementation model implements a representation of the actual MOSFET gates used to implement the NOR function. Note that although it is a more representative model of the actual device, it still has limitations. A key challenge is selecting appropriate parameters for the MOSFET blocks based on the datasheet. Here all MOSFETS are assumed to have the same parameters which will not always be the case. The main disadvantage of using an implementation model is that simulation speed is likely to be much slower than the behavioral model.

**Model**

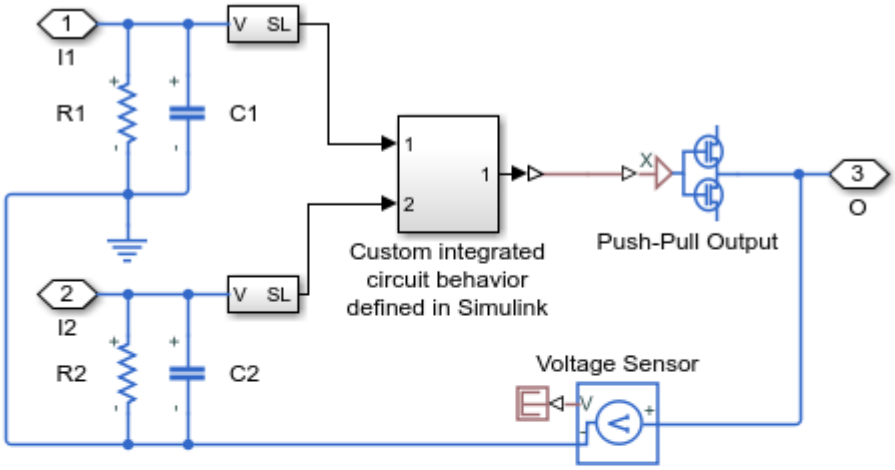



**Modeling an Integrated Circuit**

1. Plot inputs and outputs for NOR models (see code)
2. Explore simulation results using Simscape Results Explorer
3. Learn more about this example

Copyright 2006-2022 The MathWorks, Inc.

### 2-Input NOR (Behavioral Model) Subsystem

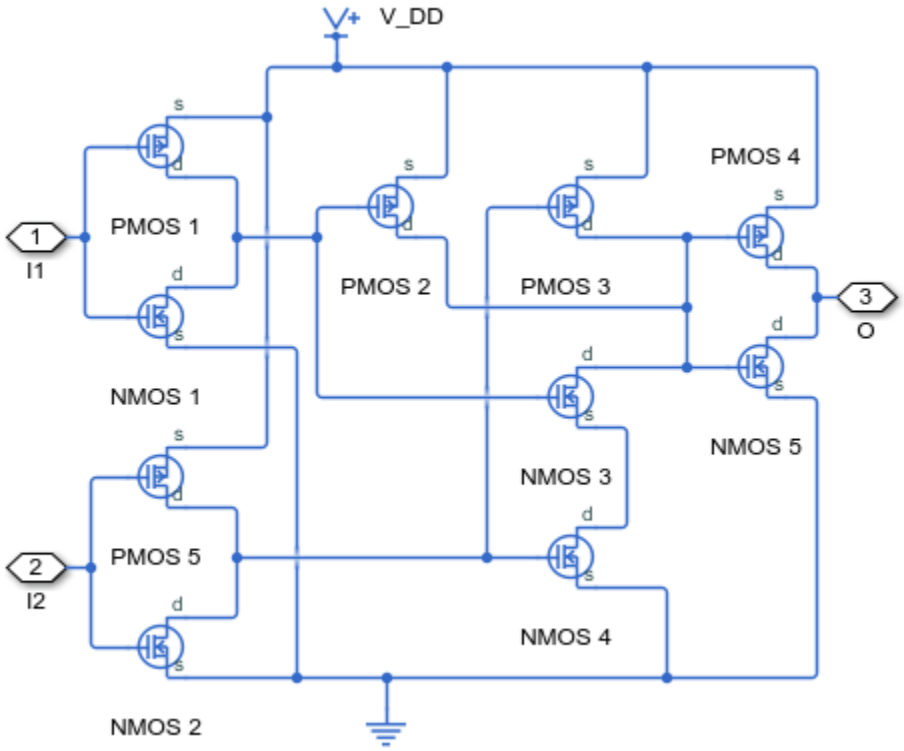



The integrated circuit behavior can be implemented in Simulink(R), or using the Simscape(TM) Foundation Library physical signal blocks. A Simulink implementation should a propagation delay to break any potential algebraic loops created by the external circuit.

The input resistances and capacitances should be set to match the input impedance of the device you are modeling. The Push-Pull Output block should be set to Quadratic. if the output pin must provide significant current and/or short-circuit output conditions are to be simulated.

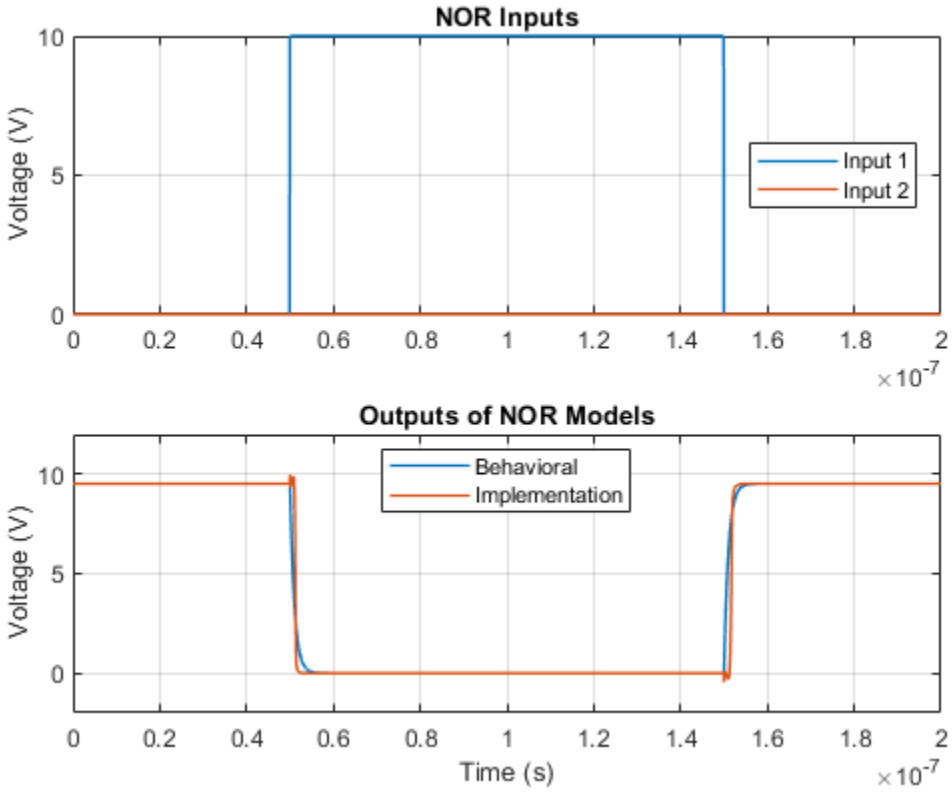
The Voltage Sensor links the networks attached to the inputs and output so that separate solver blocks are not required for the input and output networks

### 2-Input NOR (Implementation Model) Subsystem




### Simulation Results from Simscape Logging

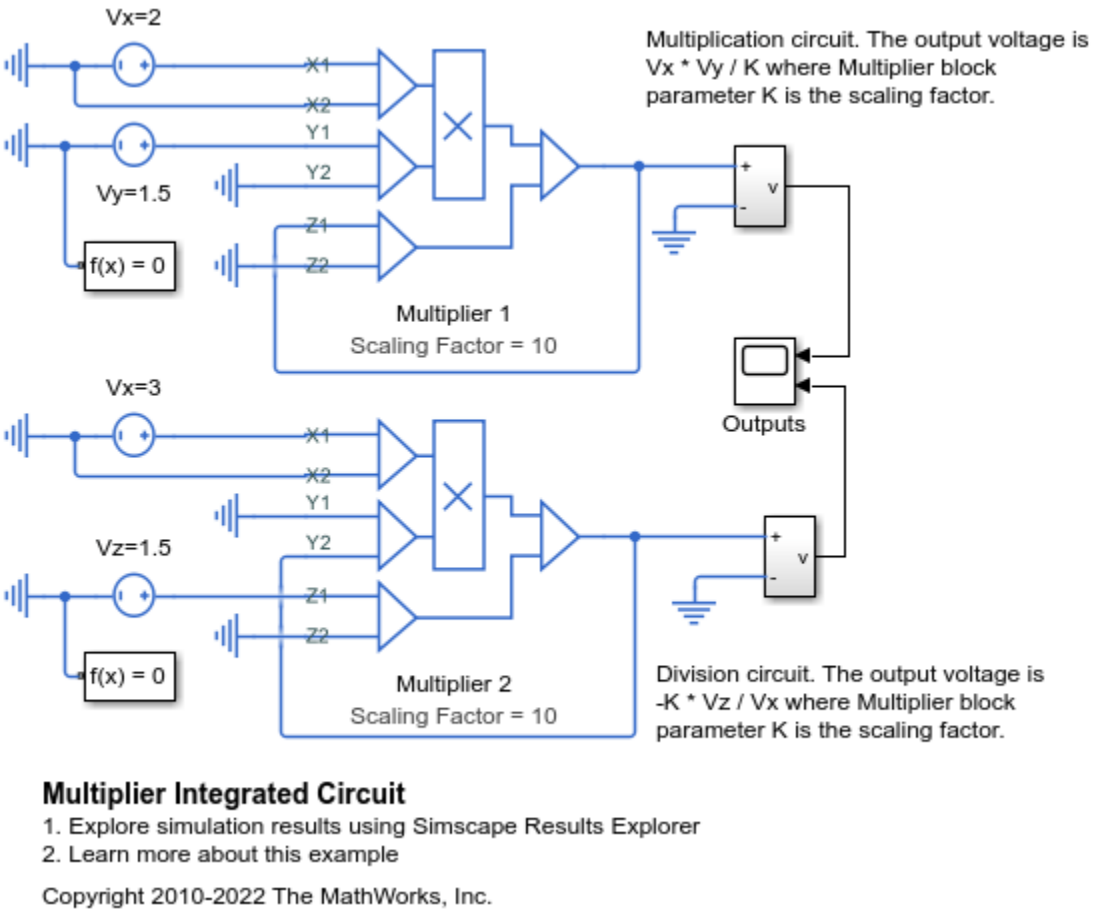
The plots below show the inputs and outputs for the two implementations of the NOR circuit. The behavioral model and implementation model have nearly identical outputs.


NOR Inputs
Voltage (V)
10
5
0
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
×10^-7
Input 1
Input 2
Outputs of NOR Models
Voltage (V)
10
5
0
Behavioral
Implementation
Time (s)
×10^-7

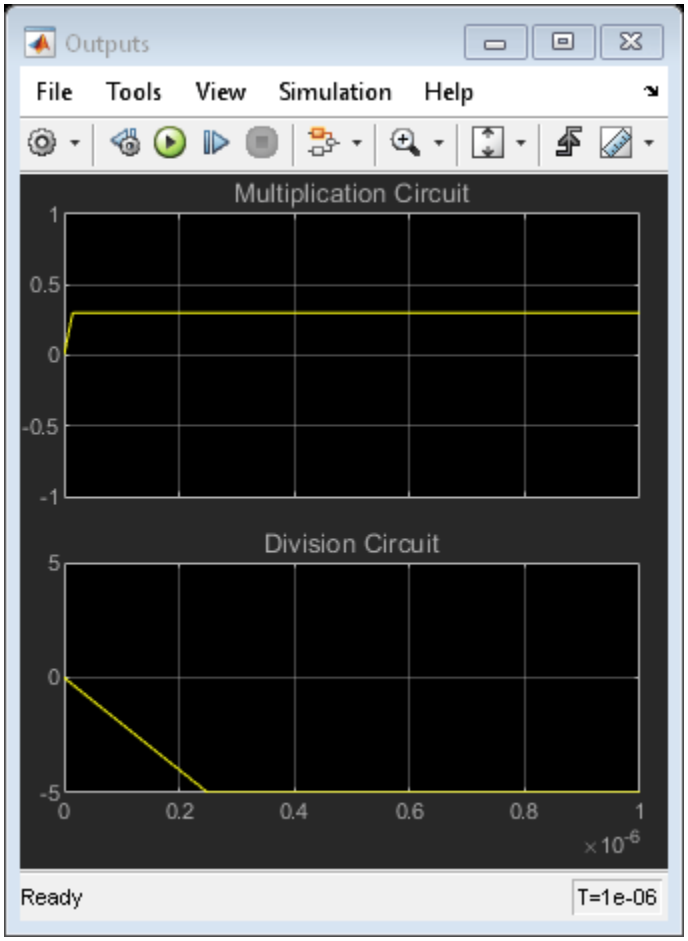
# Multiplier Integrated Circuit

This model shows how the Multiplier block can be used to multiply or divide two input voltages. In both cases, slew rate limiting occurs until the final voltage is reached. Note that the input-side circuits must have an Electrical Reference block so that the solver can determine a voltage for every node.

### Model



### Simulation Results from Scopes



# Solar Cell Parameter Extraction from Data

This example shows optimization of the Solar Cell block's parameters to fit data defined over a range of different temperatures. It uses the MATLAB® optimization function `fminsearch`. Other products available for performing this type of parameter fitting with Simscape™ Electrical™ models are the Optimization Toolbox™ and Simulink® Design Optimization™. These products provide predefined functions to manipulate and analyze blocks using GUIs or a command line approach.

### Strategy

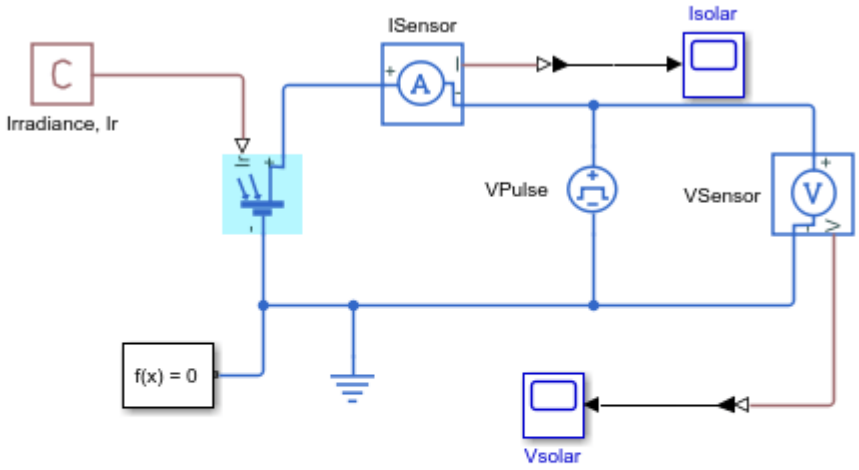
Fit I-V output curves for an 8 Parameter Solar Cell to data using a 2 step procedure:

1. Optimize parameters in the Solar Cell Main dialog tab to match output curves to data at room temperature.
2. Optimize parameters in the Solar Cell Temperature dialog tab to match output curves to data at non-room temperatures.

### Data and Block Setup

The MATLAB data file, *ee_solar_iv_data.mat*, stores Solar Cell data as an array of structures. Each structure contains 3 fields: *temperature* , *i* (current) and *v* (voltage). The Solar Cell block references a data structure to set the device operating temperature, the VPulse block's Pulse value, V2, and to generate simulation output currents at prescribed voltages. The prescribed voltages are set by outputting simulation results at prescribed output times. To set the output times, on the Modeling tab, click Model Settings; in the Configuration Parameters, in the Data Import/Export category, click Additional parameters and specify the Output times. Scopes save the output voltage and current responses as structure data, Io.signal.values and Vo.signal.values.

```
% Load Solar Cell data
load ee_solar_iv_data.mat
% Display the Solar Cell model
Model = 'ee_solar';
open_system(Model)
```




**Solar Cell Parameter Tuning**

This test model is used by the Solar Cell Parameter Extraction From Data example.
It is invoked by the parameter optimization when tuning parameters to fit the
measured I-V curve data.

Copyright 2008-2021 The MathWorks, Inc.

```
close_system(Model, 0);
```

### Initial Parameter Specification

Starting values for `fminsearch` can be estimated using a combination of Solar Cell block defaults, data sheet values and the following equations:

$$I_{ph} = I_{sc}$$

$$I_s = \frac{I_{ph}}{(exp(V_{oc}/(.025 * ec)) - 1)}$$

$$R_s = \frac{-dV}{dI} @ V_{oc}$$

$$R_p = \frac{-dV}{dI} @ I_{sc}$$

List of parameters and initial values prior to optimization

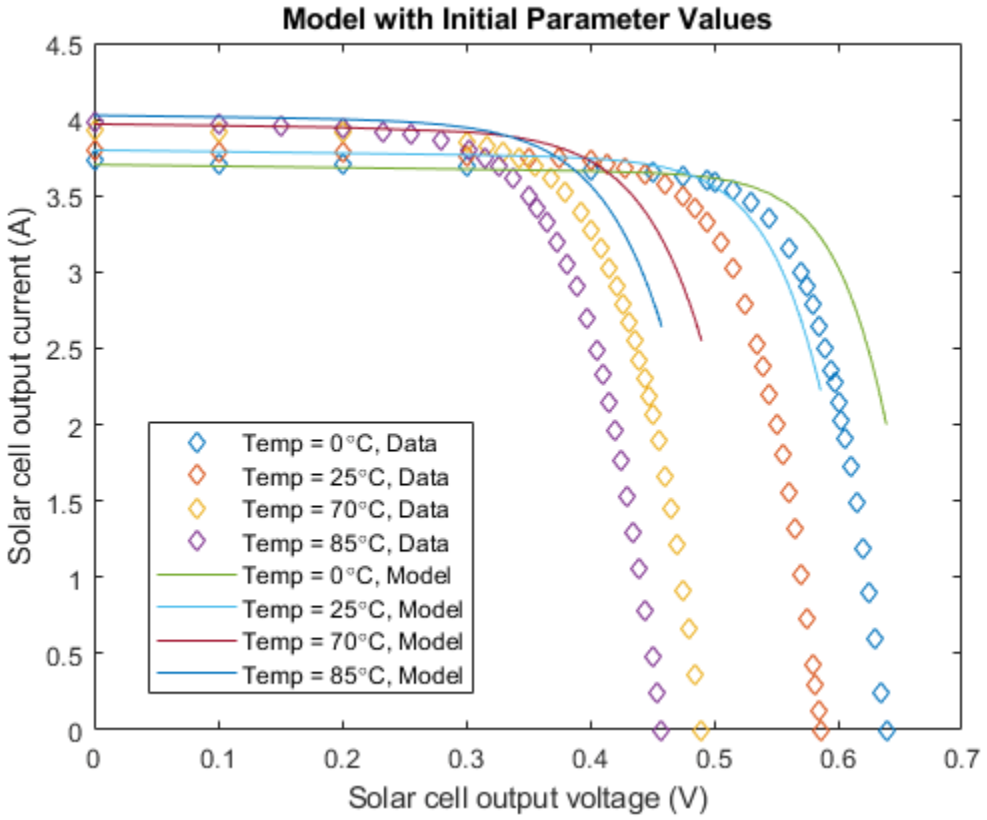
```
ParsListMain = {'Is', 'Iph', 'ec', 'Rs', 'Rp'};
InitGuessMain = [ 3e-7 3.8 1.5 .004 10 ];
ParsListTemp = {'TIPH1', 'EG', 'TXIS1'};
InitGuessTemp = [ .001 1.11 3 ];
```

Since `fminsearch` is an unconstrained nonlinear optimizer that locates a local minimum of a function, varying the initial estimate will result in a different solution set.

### Plot Data Versus Solar Cell Output Using Initial Parameters

Load 8 parameter Solar Cell model and set parameters

```
load_system(Model);
set_param([Model '/Solar Cell'], 'prm', '3')
Pars = reshape([ParsListMain; cellstr(num2str(InitGuessMain'))'],1,[]);
set_param([Model '/Solar Cell'], Pars{:})
Pars = reshape([ParsListTemp; cellstr(num2str(InitGuessTemp'))'],1,[]);
set_param([Model '/Solar Cell'], Pars{:})

% Generate preliminary model curves and plot against data
num_lines = length(iv_data);
v_model = cell(1, num_lines);
i_model = cell(1, num_lines);
legend_info_data  = cell(1, num_lines);
legend_info_model = cell(1, num_lines);
for idx_data = 1:num_lines
    sim(Model);
    v_model{idx_data} = Vo.signals.values;
    i_model{idx_data} = Io.signals.values;
    legend_info_data{idx_data}  = [ 'Temp = '                                ...
        num2str(iv_data(idx_data).temperature) '\circC, Data'];
    legend_info_model{idx_data} = [ 'Temp = '                                ...
        num2str(iv_data(idx_data).temperature) '\circC, Model'];
end
plot([iv_data.v], [iv_data.i], 'd', [v_model{:}], [i_model{:}])
xlabel('Solar cell output voltage (V)');
ylabel('Solar cell output current (A)');
legend([legend_info_data legend_info_model], 'Location', 'Best');
title('Model with Initial Parameter Values');
```



### Sum of Squares of Error Calculation

`ee_solar_lse` is the function to be minimized by `fminsearch`. This function returns a sum of squares of error for the difference between the solar cell output current and the data. If an invalid parameter value is supplied by `fminsearch`, the `catch` statement returns a large value for the error.

### Optimize Main Tab Dialog Parameters at Room Temperature (Step 1)

```
% Find room temperature data index
idx_data = find([iv_data.temperature]==25);%#ok

% Optimize parameters in main dialog tab of Solar Cell
ParsList = ParsListMain;
OptParsMain = fminsearch(@ee_solar_lse, InitGuessMain,              ...
    optimset('TolX', 1e-3));

% Update Solar Cell block with optimized parameters
Pars = reshape([ParsList; cellstr(num2str(OptParsMain'))'],1,[]);
set_param([Model '/Solar Cell'], Pars{:});
% Display optimized parameters
display(sprintf(['Optimized parameters for the solar cell main '      ...
    'dialog tab are:\n']));
display(sprintf('\t%5s = %s\n', Pars{:}));
```

```
Optimized parameters for the solar cell main dialog tab are:

      Is = 3.14978e-07
     Iph =     3.80137
      ec =     1.39989
      Rs =  0.00415132
      Rp =     10.1093
```

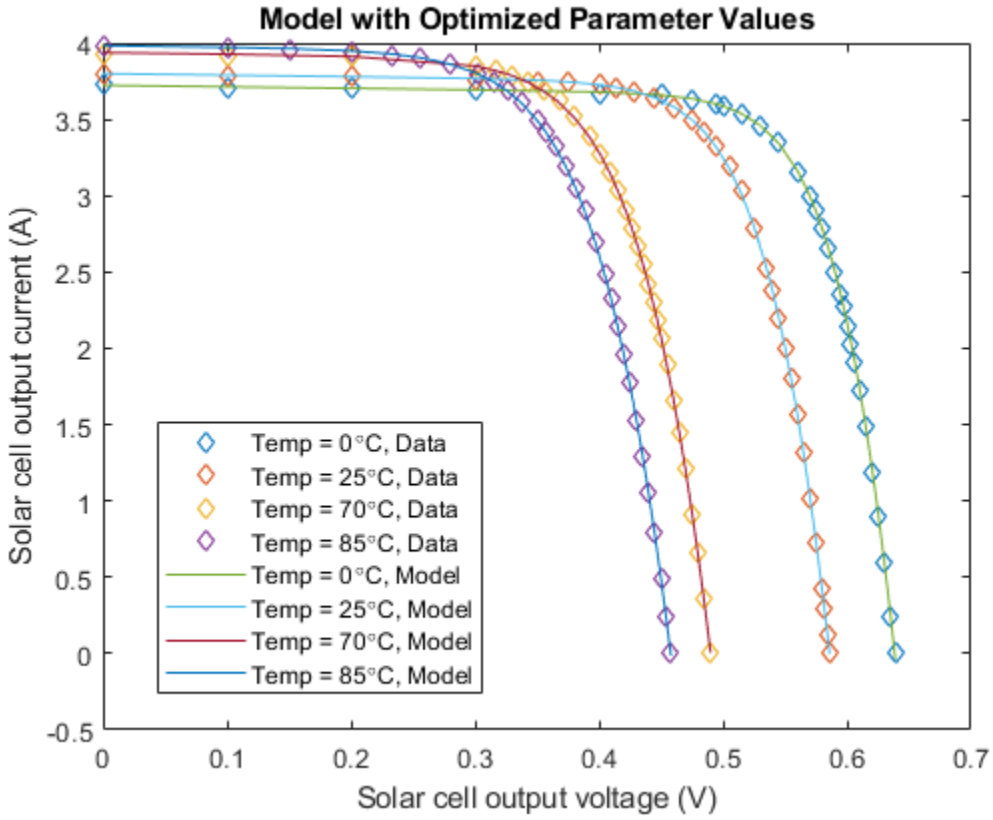
### Optimize Parameters Controlling Temperature Dependence (Step 2)

```
% Find index into data for non-room temperatures
idx_data = find([iv_data.temperature]~=25);

% Optimize parameters in temperature dialog tab of Solar Cell
ParsList = ParsListTemp;
OptParsTemp = fminsearch(@ee_solar_lse, InitGuessTemp,              ...
    optimset('TolX', 1e-3));

% Update Solar Cell block with optimized temperature parameters
Pars = reshape([ParsList; cellstr(num2str(OptParsTemp'))'],1,[]);
set_param([Model '/Solar Cell'], Pars{:});
% Display optimized parameters
display(sprintf(['Optimized parameters for the solar cell '           ...
    'temperature dialog tab are:\n']));
display(sprintf('\t%5s = %s\n', Pars{:}));
```

```
Optimized parameters for the solar cell temperature dialog tab are:

   TIPH1 = 0.00080492
      EG =     1.1384
   TXIS1 =     3.3842
```

### Display Optimized Curves

```
for idx_data = 1:num_lines
   sim(Model);
   v_model{idx_data} = Vo.signals.values;
   i_model{idx_data} = Io.signals.values;
end
plot([iv_data.v], [iv_data.i], 'd', [v_model{:}], [i_model{:}])
xlabel('Solar cell output voltage (V)');
ylabel('Solar cell output current (A)');
legend([legend_info_data legend_info_model], 'Location', 'Best');
title('Model with Optimized Parameter Values');
```

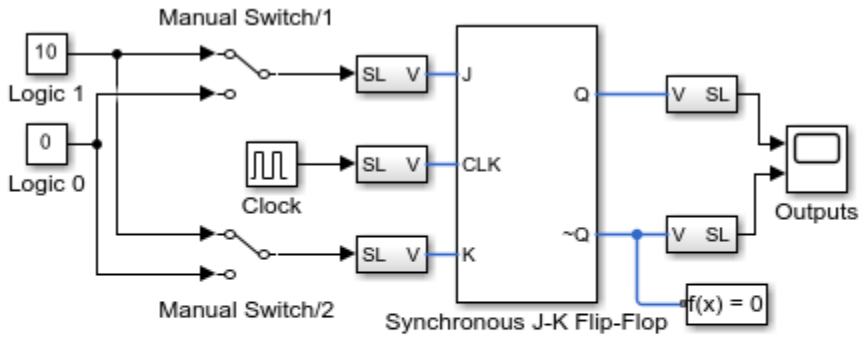



```
bdclose(Model)
```

# Synchronous J-K Flip-Flop

This example shows how to model a J-K flip-flop from Simscape™ Electrical™ logic components. With the two switches in their default positions, both inputs to the flip-flop are set high so its output state toggles each time the clock signal goes low. Initial conditions are passed to the relevant NAND gates via the initialization commands of the block mask.

### Model

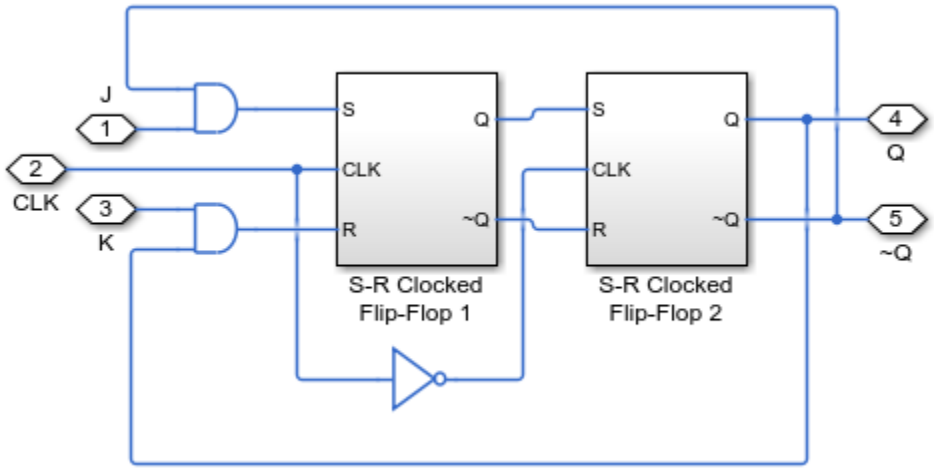



**Synchronous J-K Flip-Flop**

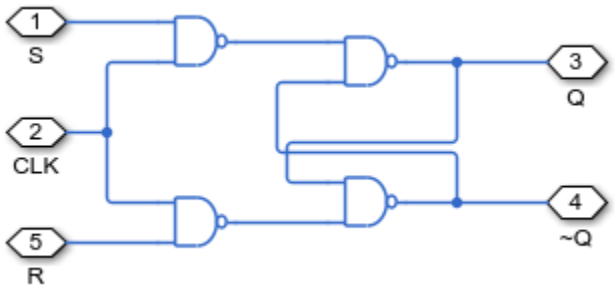
1. Plot inputs and outputs for flip-flop (see code)
2. Explore simulation results using Simscape Results Explorer
3. Learn more about this example

Copyright 2006-2022 The MathWorks, Inc.

### Synchronous J-K Flip-Flop Subsystem

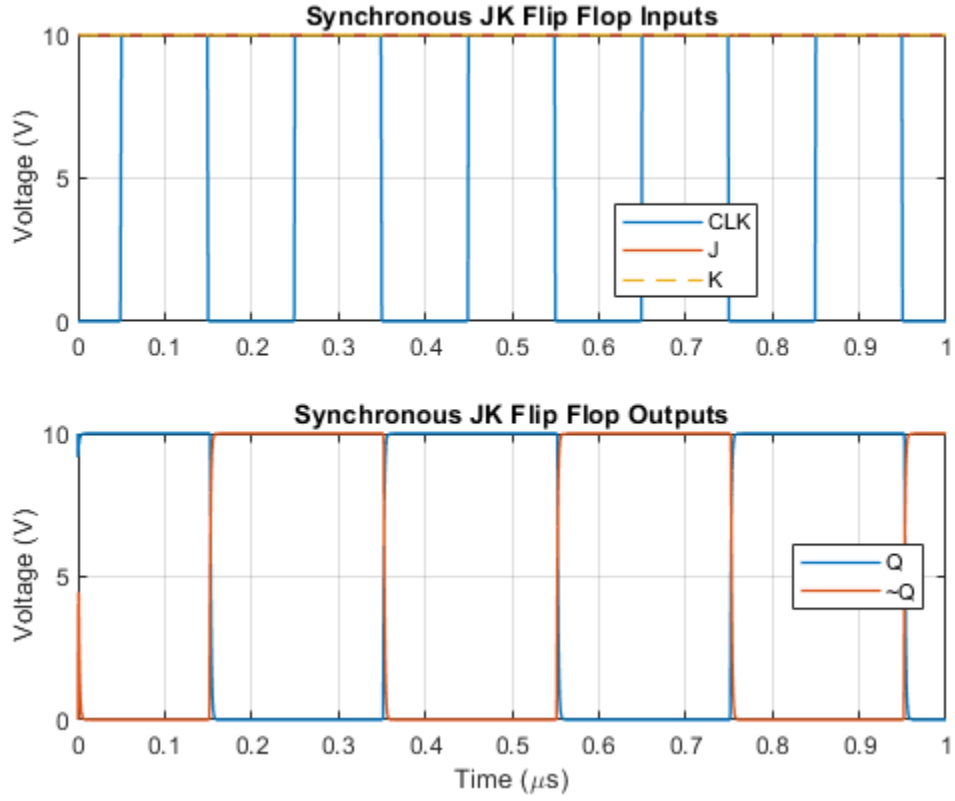


### S-R Clocked Flip-Flop 1 Subsystem




### Simulation Results from Simscape Logging

The plots below show the inputs and outputs for the synchronous J-K flip-flop. Both inputs to the flip-flop are set high so its output state toggles each time the clock signal goes low.



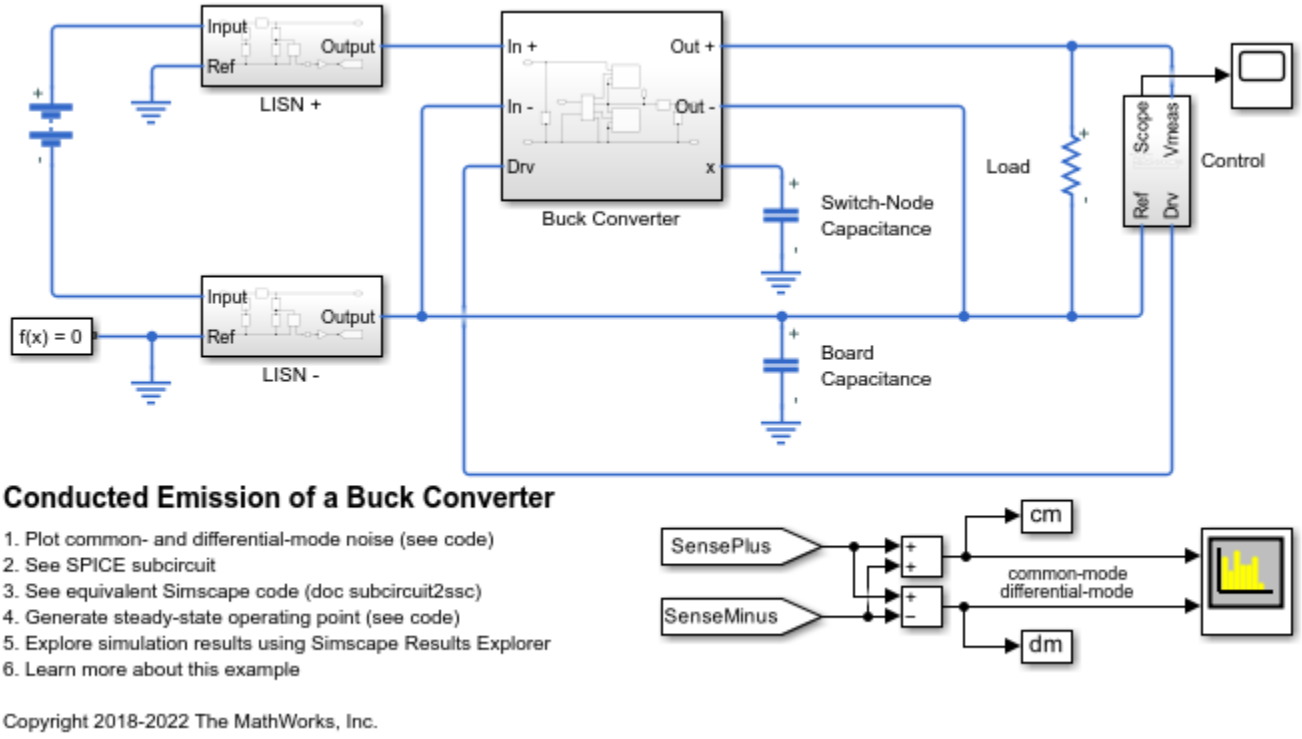
# Conducted Emission of a Buck Converter

This example shows a buck converter configured for a measurement of common- and differential-mode noise on the source. In order to simulate the common-mode noise, the capacitive coupling between the circuit and a reference plane must be included in the model. In this circuit, the capacitance between the switching node (between the high- and low-side transistors) and the reference plane is also included.

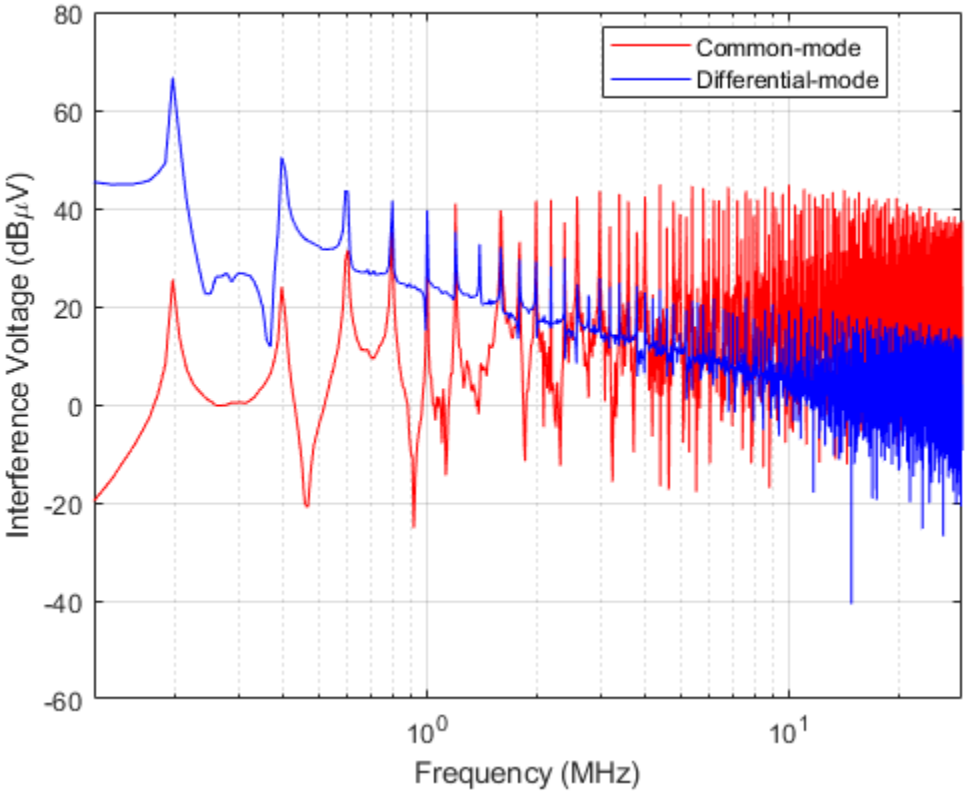
Ideal line impedance stabilization networks (LISNs) are placed between each terminal of the power supply and the inputs of the buck converter in order to provide a standard impedance and measurement port for the common- and differential-mode noise.

In order to obtain the noise measurement under steady-state conditions, a steady-state operating point has been saved and is used as the starting point for the noise simulation.

**Model**




The plot below shows the common- and differential-mode noise for a buck converter operating at a switching frequency of 200kHz.


Common-mode
Differential-mode
Interference Voltage (dBμV)
Frequency (MHz)
80
60
40
20
0
-20
-40
-60
10^0
10^1

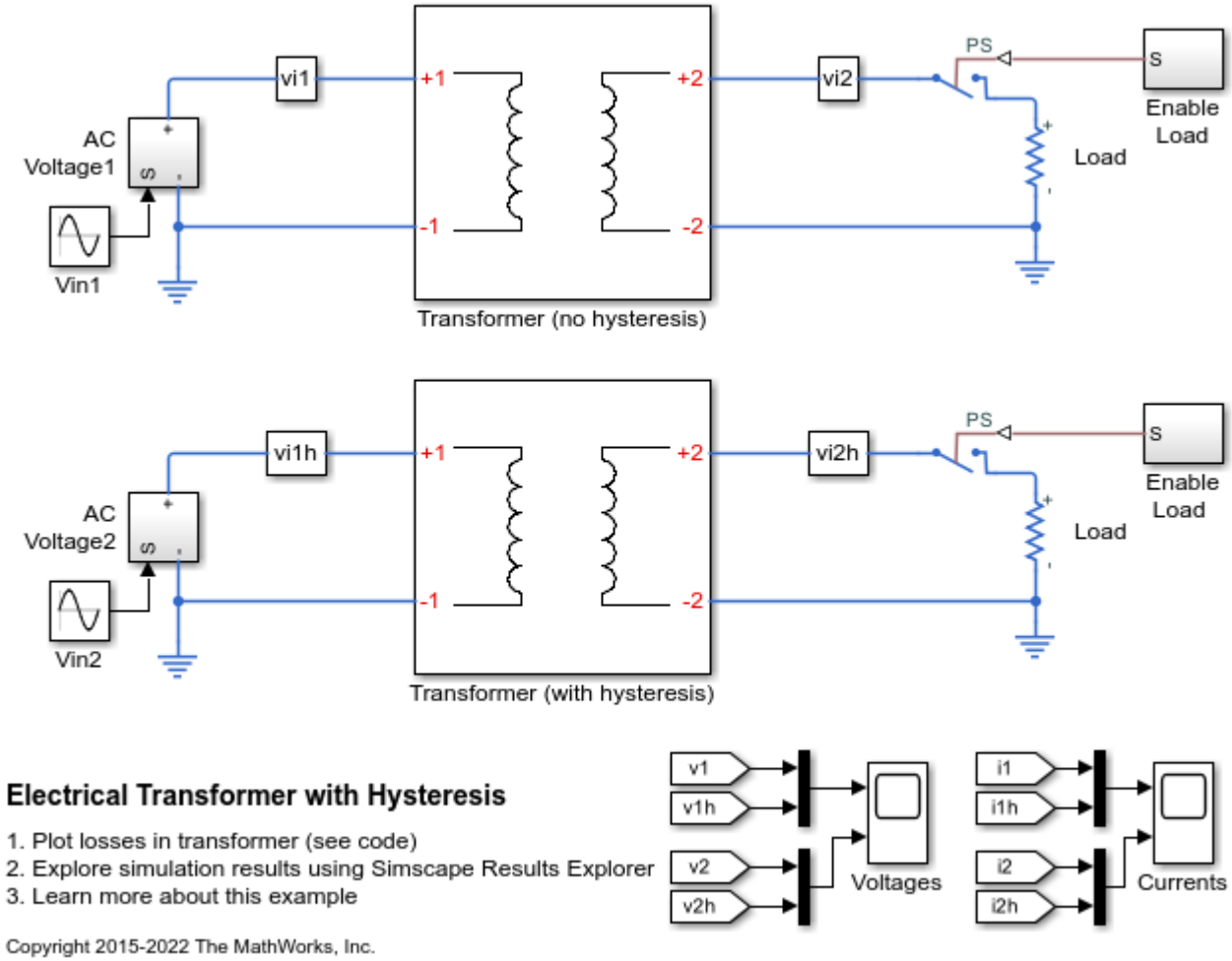
# Electrical Transformer with Hysteresis

This example shows how to model a custom transformer that exhibits hysteresis by using the Non Linear Reluctance block in a magnetic circuit. The transformer is rated for a 50W load and steps down from 120V to 12V rms. The magnetizing resistance Rm is modeled in the magnetic domain using an Eddy Loss block.

The transformer initially operates under no-load conditions. At time t=0.05s, the rated load is turned on. For comparison purposes, the transformer is modeled with and without hysteresis by two separate subsystems. The primary and secondary inductance values L1 and L2 plus associated winding numbers N1 and N2 are used to determine the two leakage reluctances.

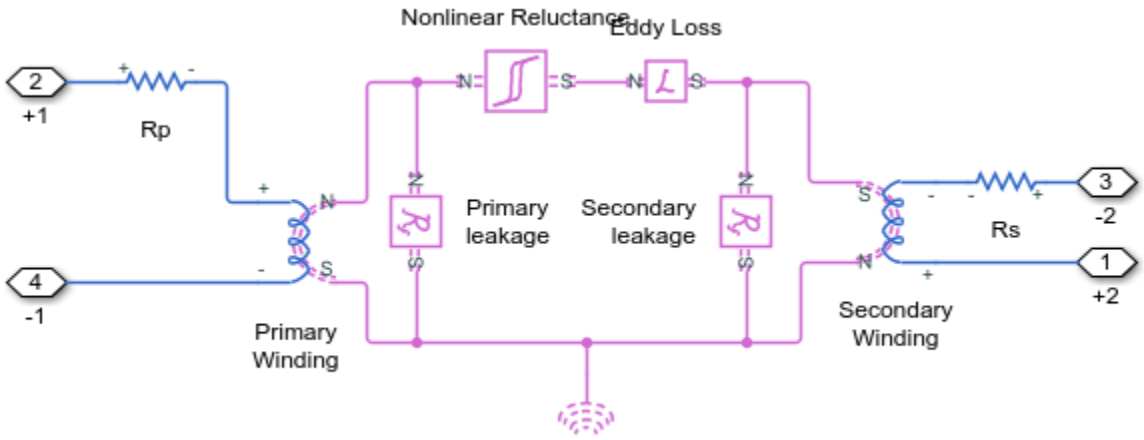
The magnetic hysteresis losses are determined by calculating the total electrical power flowing into the magnetic circuit from both windings. The difference between power in and power out over a complete AC cycle reflects the hysteresis and eddy losses. In order to obtain just the hysteresis loss, the eddy loss is subtracted from the total. The loss is calculated separately for the no-load and rated load conditions.

**Model**



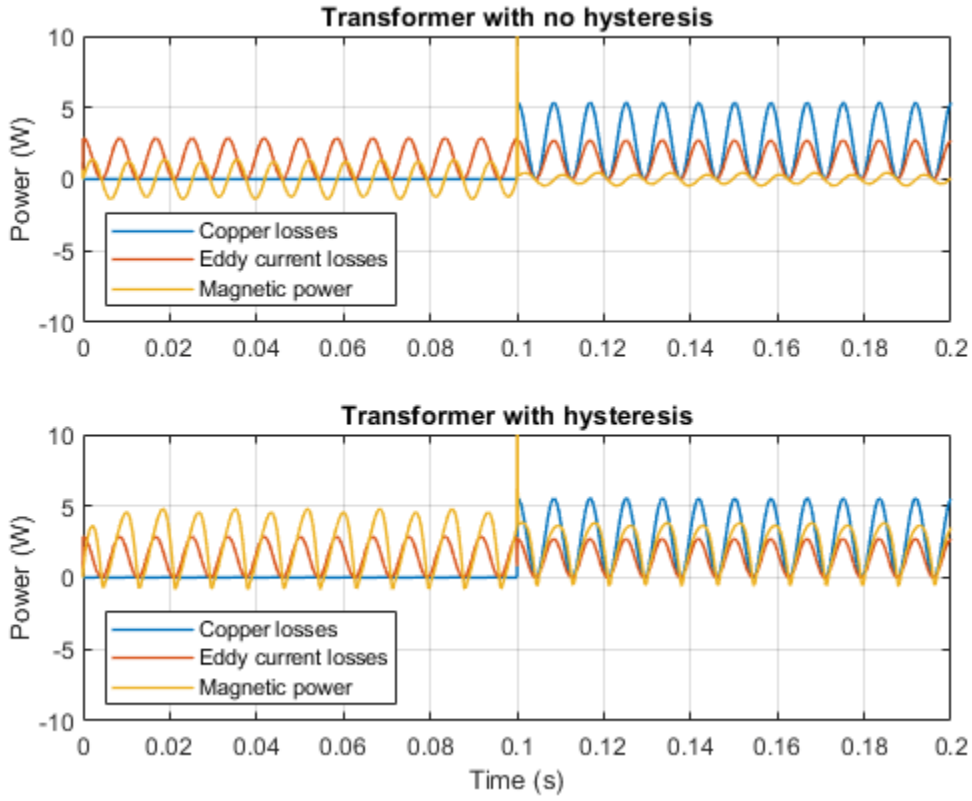

Copyright 2015-2022 The MathWorks, Inc.

### Transformer Subsystem




### Simulation Results from Simscape Logging

```
No-load hysteresis loss = 2.3112 watts
Rated-load hysteresis loss = 2.2047 watts
```



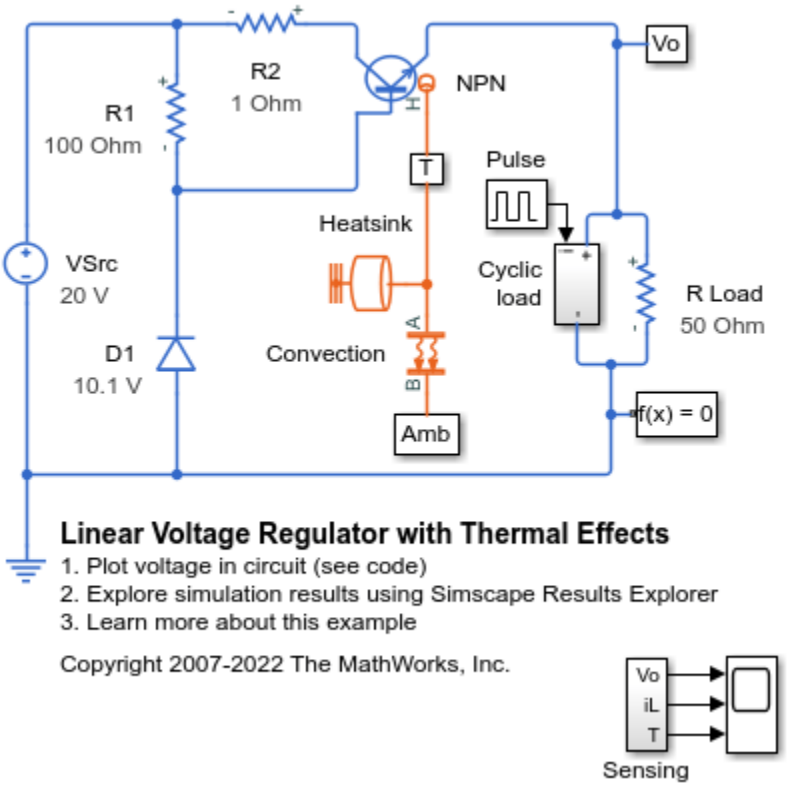
# Linear Voltage Regulator with Thermal Effects

This example shows a low-cost voltage regulator circuit whose performance depends on both load current and temperature. Bias resistor R1 ensures that the voltage at the transistor base is close to the rated zener voltage. The regulator output voltage is also approximately at this voltage, the base-emitter voltage being a few tenths of a volt. The precise base-emitter voltage depends on the transistor working point (which in turn depends on the load) and also the temperature. Resistor R2 only serves to provide some protection in the event of a transient output short circuit.

This type of model can be used to validate that selected circuit components result in an adequate level of voltage regulation. It can also be used to size the heatsink required to keep the transistor junction temperature within the permitted operating range. By modeling both electrical and thermal properties, understanding of the trade-offs between electrical and thermal parameter choices is gained.

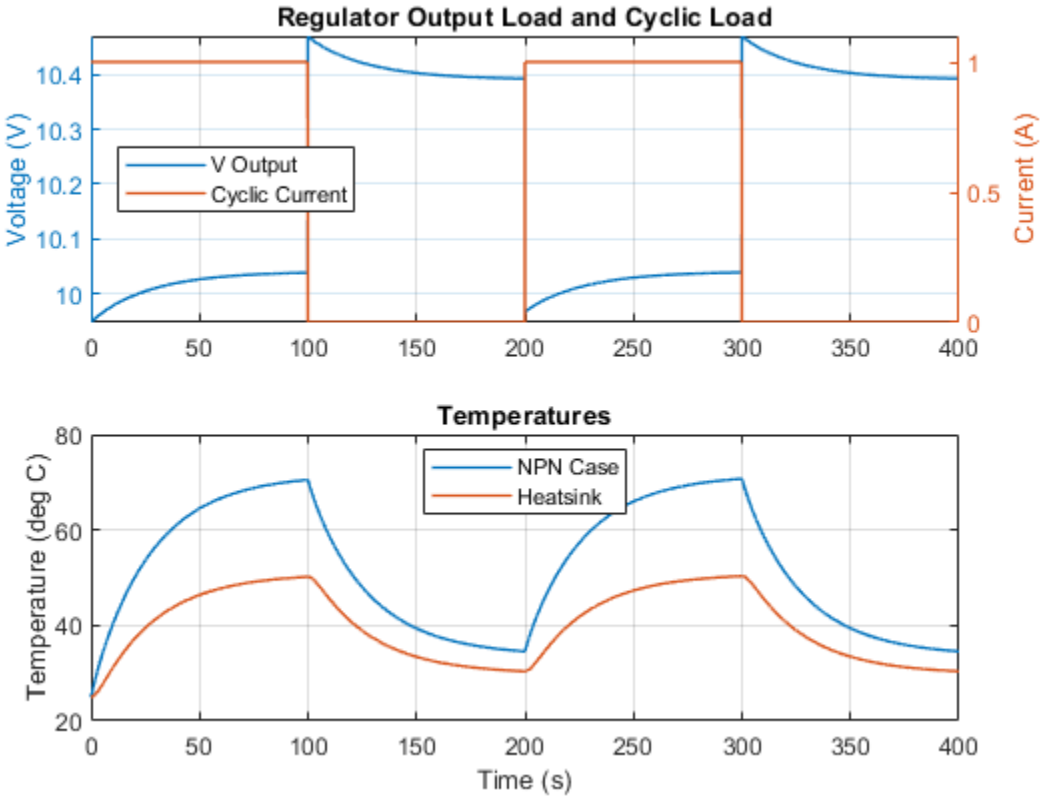
Better regulation can be achieved by using feedback. See example “Linear Voltage Regulator with Feedback” on page 8-657 (>> ee_voltreg_linear_feedback).

### Model




### Simulation Results from Simscape Logging

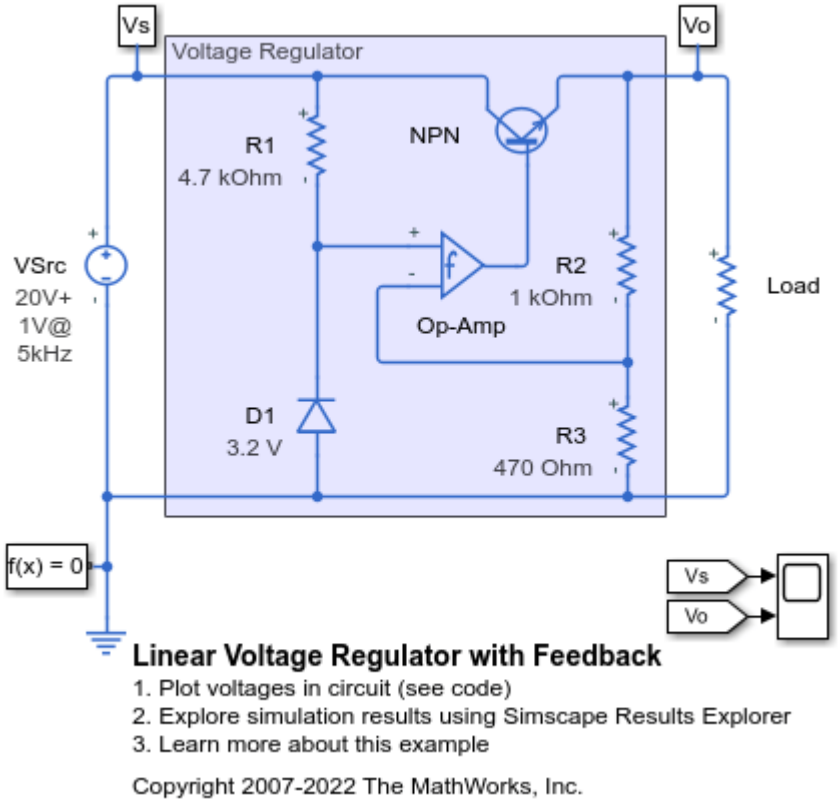
The plot below shows the output of the voltage regulator holding close to the desired output of 10 V. It also shows the temperature of the NPN case and its heatsink as the cyclic load turns on and off.


Regulator Output Load and Cyclic Load
Voltage (V)
Current (A)
V Output
Cyclic Current
10.4
10.3
10.2
10.1
10
1
0.5
0
0
50
100
150
200
250
300
350
400
Temperatures
Temperature (deg C)
NPN Case
Heatsink
80
60
40
20
Time (s)

# Linear Voltage Regulator with Feedback

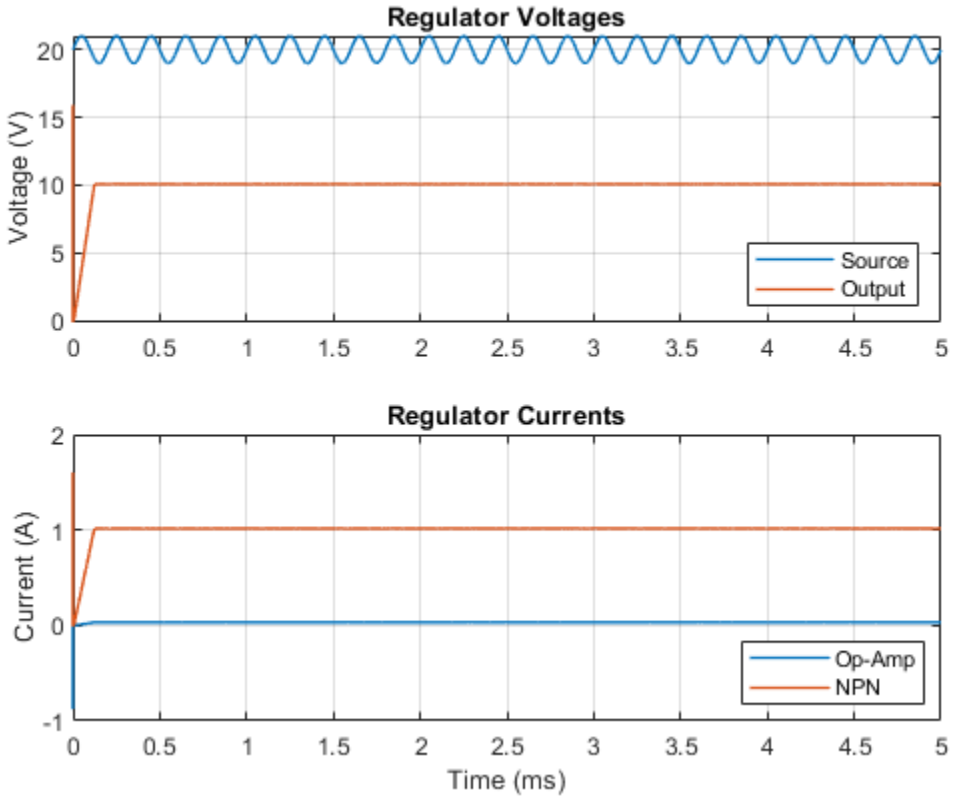
This example shows a simple voltage regulator circuit constructed from discrete components. A fluctuating supply is modeled as 20V DC plus a 1V sinusoidal variation. The zener diode D1 sets the non-inverting input of the op-amp to 3.2V, and hence as the op-amp has a large gain, the op-amp inverting input and output are also at 3.2V. Hence the regulator voltage output is regulated to be 3.2*(1000+470)/470=10V. The NPN bipolar transistor is required to provide higher currents than is possible from a typical op-amp. The model can be used to check circuit operation, and to support selection of components to achieve the desired voltage regulation.

### Model




### Simulation Results from Simscape Logging

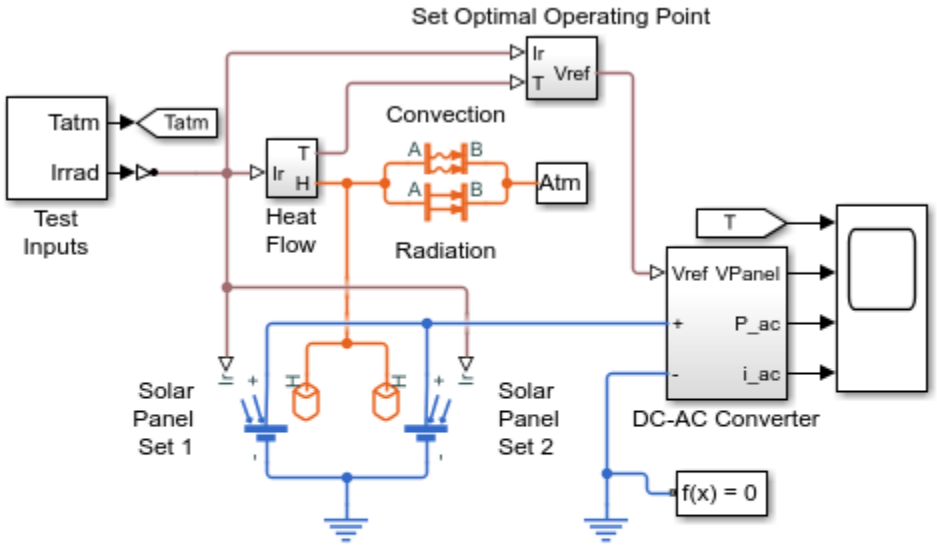
The plot below shows the output of the voltage regulator holding at a constant 10V even as the input voltage fluctuates.


Regulator Voltages
Voltage (V)
20
15
10
5
0
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
Source
Output
Regulator Currents
Current (A)
2
1
0
-1
Op-Amp
NPN
Time (ms)

# Photovoltaic Generator

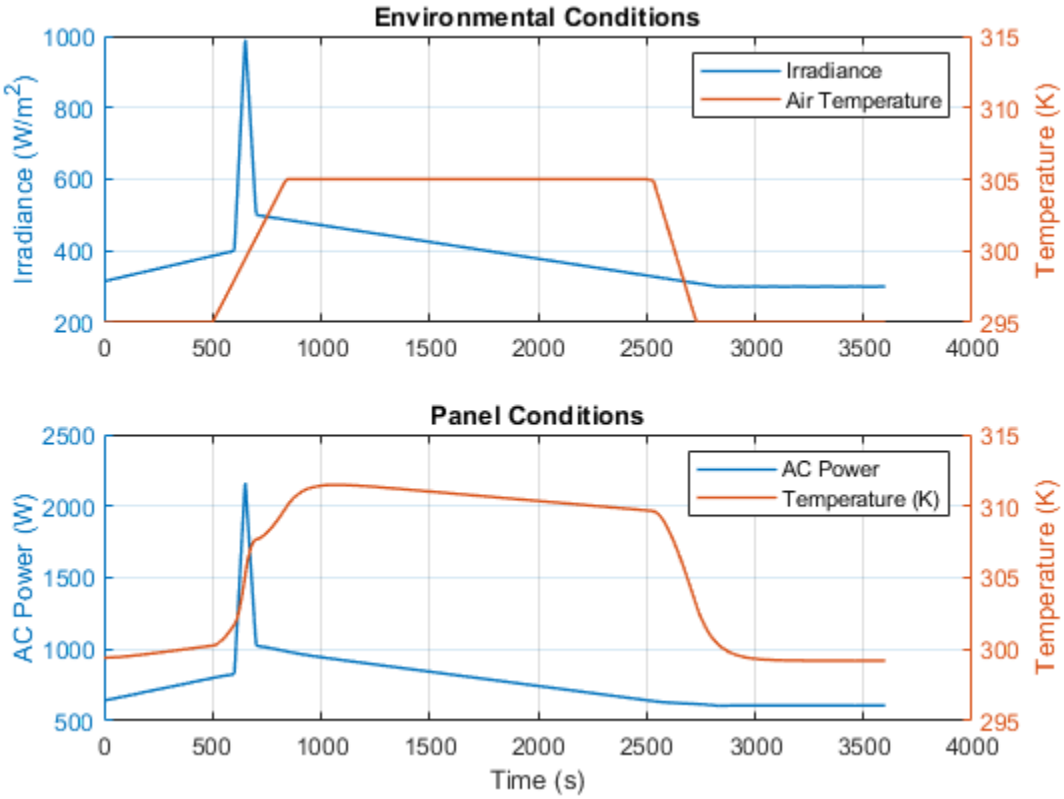
This example shows how to create system-level model of a photovoltaic generator that can be used to simulate performance using historical irradiance data. Here the model is tested by varying the irradiance which approximates the effect of varying cloud cover. Power generation steps immediately following the irradiance change. Environmental temperature also varies during the test. The DC-AC converter efficiency is assumed to be a fixed 97 percent, this value having been determined from the ee_solar_inverter example model.

### Model

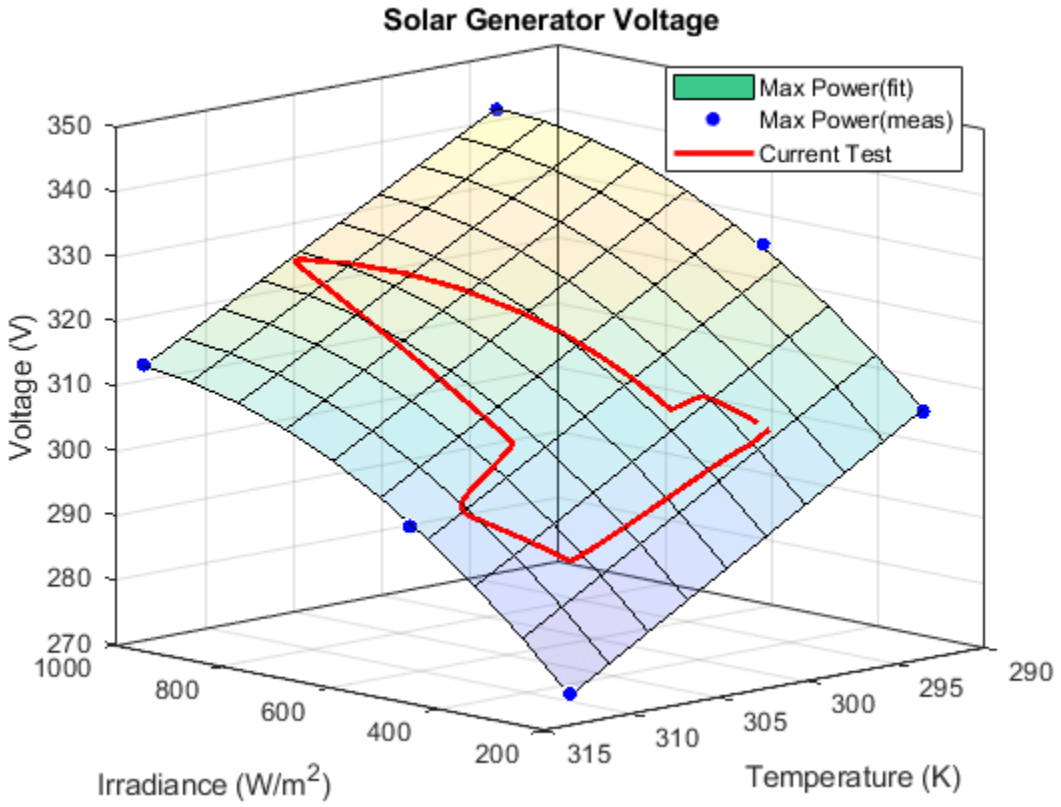



### Simulation Results from Simscape Logging

The plot below shows the power output and temperature of the panel over a one hour test. The environmental temperature varies over time, affecting the panel temperature and its conversion efficiency.




The plot below shows the voltage output of the panel with respect to panel temperature and irradiance. For a given temperature and irradiance, solar panels have a voltage draw that will result in maximum efficiency. The blue dots on the plot are measurements from specific tests, and the surface is a polynomial fit to those points. The control system attempts to keep the voltage drawn from the panel such that it is on this surface and maximizes the efficiency of the panel.



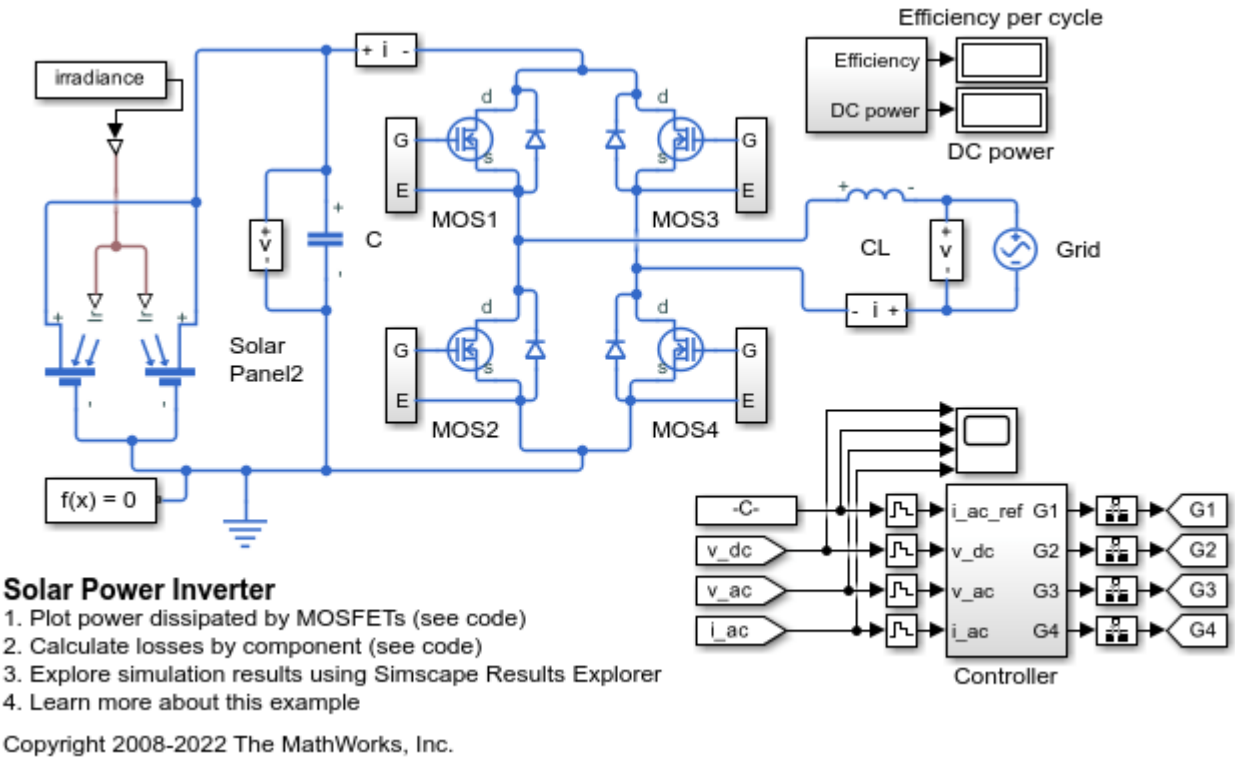

**Results from Real-Time Simulation**

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 250 microseconds.

# Solar Power Inverter

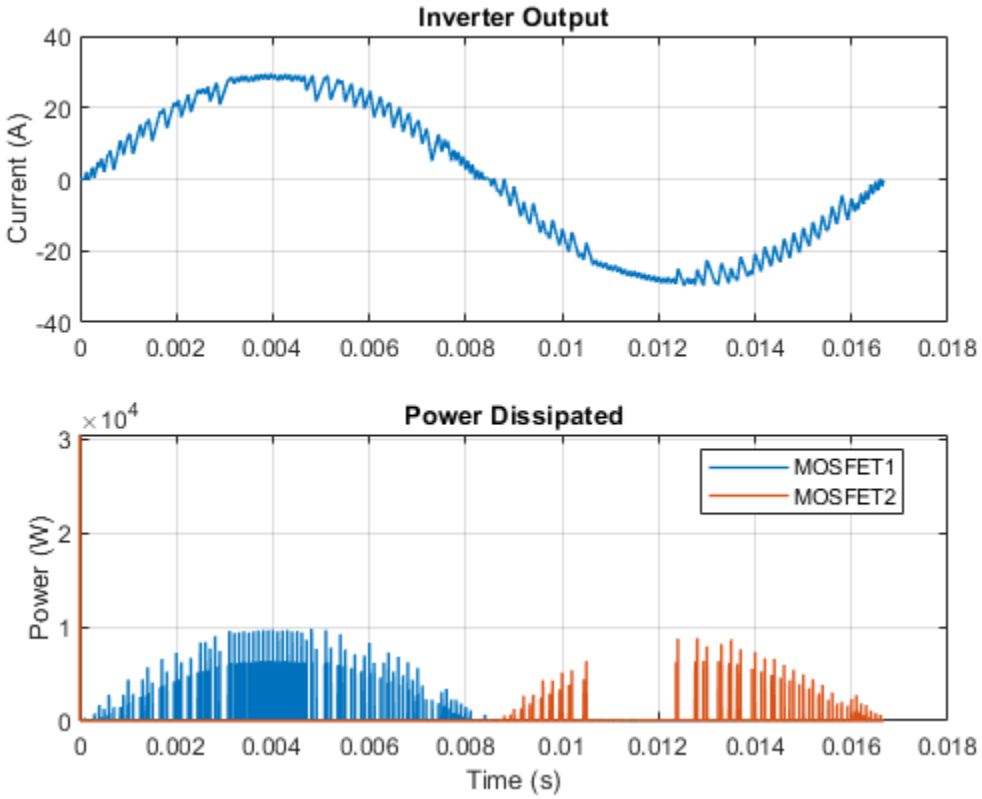
This example shows how to determine the efficiency of a single-stage solar inverter. The model simulates one complete AC cycle for a specified level of solar irradiance and corresponding optimal DC voltage and AC RMS current. Using the example model ee_solar_characteristics, the optimal values have been determined as 342V DC and 20.05A AC for an irradiance of 1000W/m^2 and panel temperature of 20 degrees Celsius. Inverter efficiency is determined in two independent ways. The first compares the ratio of AC power out to DC power in over one AC cycle. The second calculates losses by component by making use of Simscape™ logging. The small difference in calculated efficiency value is due to differences between trapezoidal integration used by the script and the greater accuracy achieved by the Simulink® variable-step solver.

### Model




### Simulation Results from Simscape Logging

The plots below show the current output from the inverter and the power dissipated by two of the MOSFETs.




The table below shows the power dissipated by individual components in the ee_solar_inverter model. These totals were calculated from simulation results using logged Simscape variables and the losses calculation utility ee_getPowerLossSummary.

```
Efficiency = 96.7384%
Losses in watts by component are as follows:
            LoggingNode              Power
    ________________________________    _____

    {'ee_solar_inverter.MOS4'  }     18.5
    {'ee_solar_inverter.MOS2'  }    18.43
    {'ee_solar_inverter.MOS1'  }    13.52
    {'ee_solar_inverter.MOS3'  }    13.48
    {'ee_solar_inverter.Diode4'}     3.47
    {'ee_solar_inverter.Diode2'}     3.46
    {'ee_solar_inverter.Diode3'}     1.49
    {'ee_solar_inverter.Diode1'}     1.46
    {'ee_solar_inverter.CL'    }      0.4
```

# Buck Converter

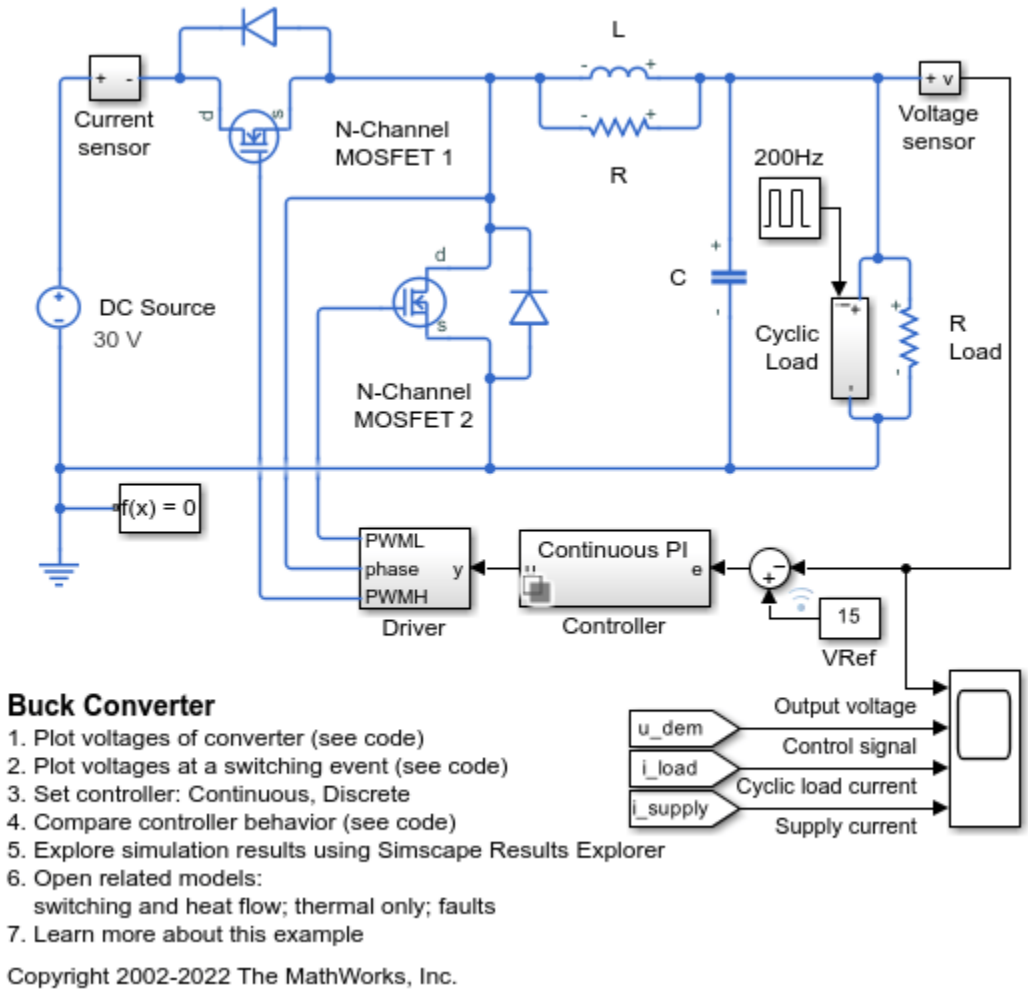
This example shows how to model a switching power supply that converts a 30V DC supply into a regulated 15V DC supply. The model can be used to both size the inductance L and smoothing capacitor C, as well as to design the feedback controller. By selecting between continuous and discrete controllers, the impact of discretization can be explored.

Modeling the switching devices as MOSFETs rather than ideal switches ensures that device on-resistances are correctly represented. The model also captures the switch-on/switch-off timing of the devices, this depending primarily on the gate capacitance values and the PWM driver output resistance.

See example model power_switching_power_supply for an abstracted version of this model that uses ideal switching to give faster simulation times. The model here can be used to determine the on-resistance values required for the ideal switches, plus timing offsets if necessary. Using the ideal switching approach of power_switching_power_supply can be used to simulate more complex power converters.

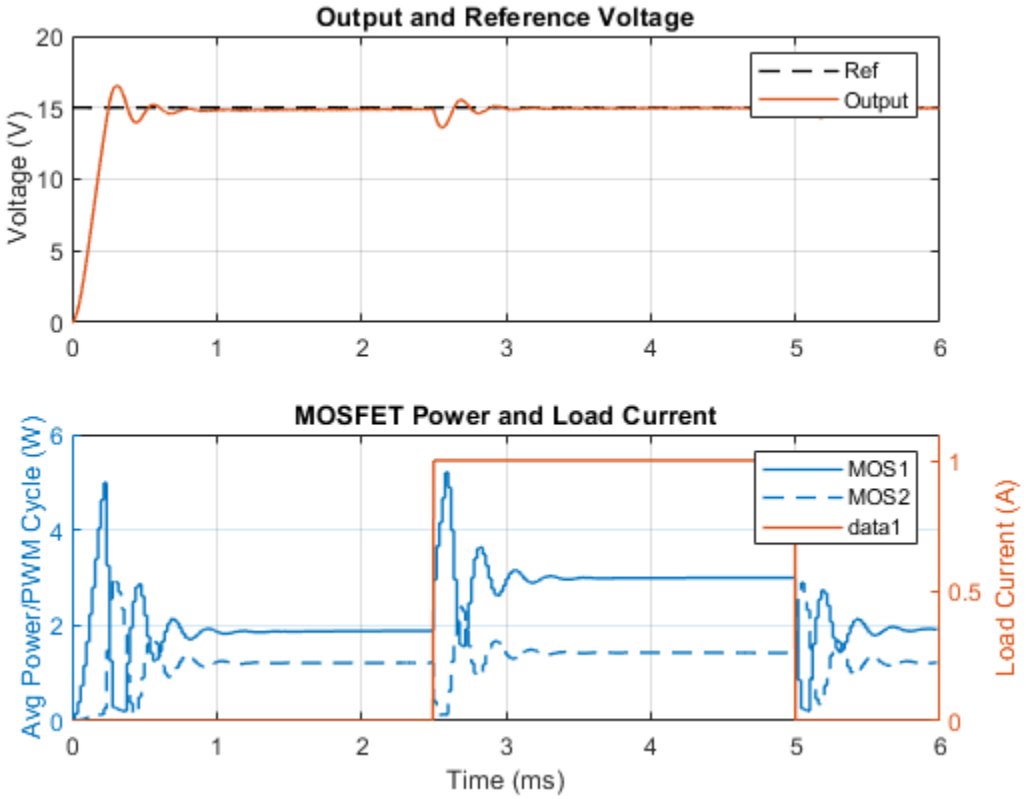
Workspace variables T_junction1 and T_junction2 are used to define the temperatures at which the two MOSFETs are simulated. Companion models ee_switching_power_supply_thermal and ee_switching_power_supply_thermal_only are used to determine these temperatures.

### Model

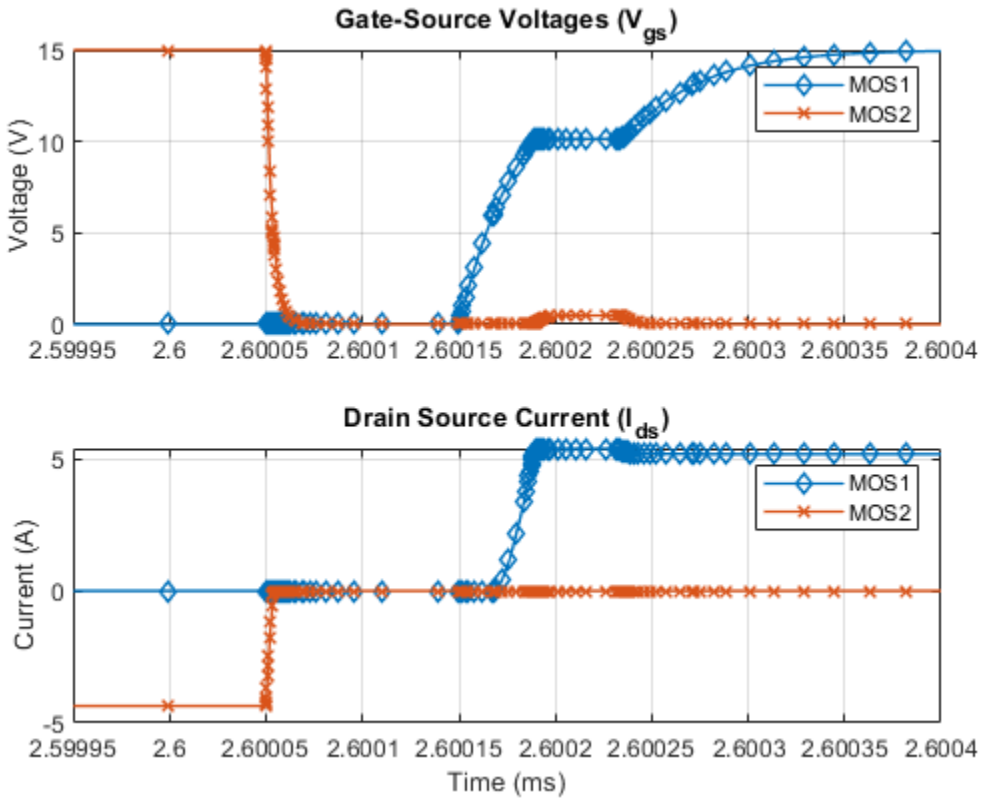



### Simulation Results from Simscape Logging

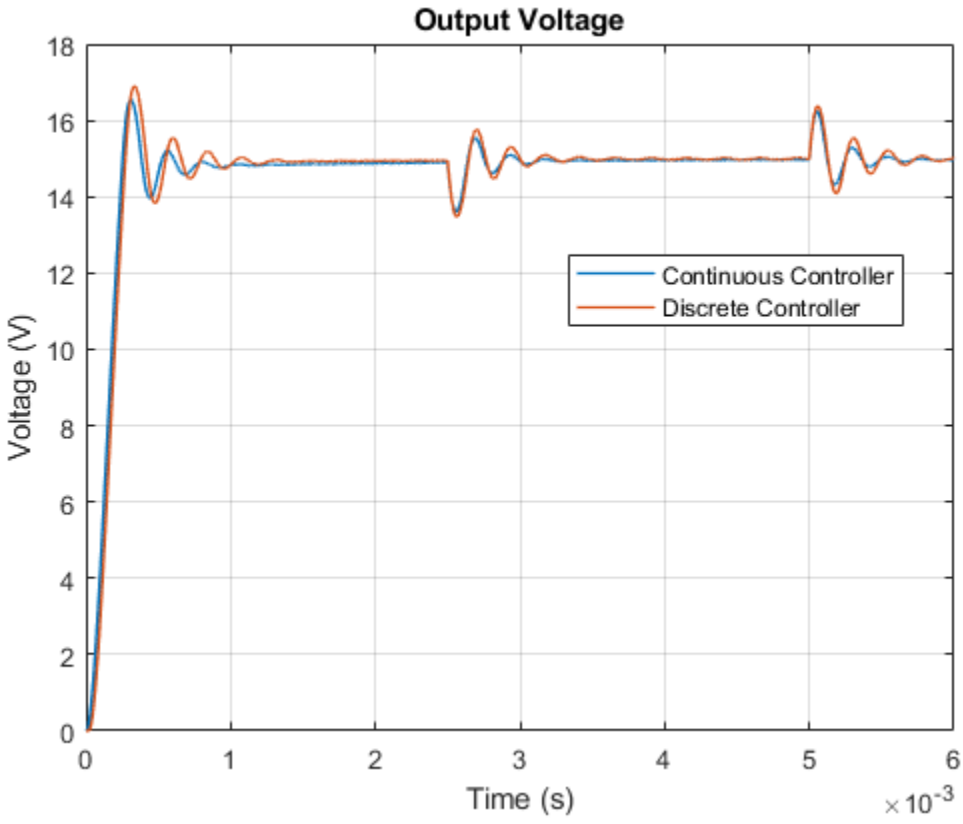
The plot below shows the output voltage as compared to the reference voltage. It also shows the changing load current and the dissipated power of the two MOSFETS averaged over the PWM cycle.




The plot below shows the switch-on/switch off timing of the two MOSFETS and the drain-source current.




The plots below show the behavior of the different implementations of the PI controller.


Output Voltage
Voltage (V)
Time (s)
×10^-3
Continuous Controller
Discrete Controller
0
2
4
6
8
10
12
14
16
18
0
1
2
3
4
5
6

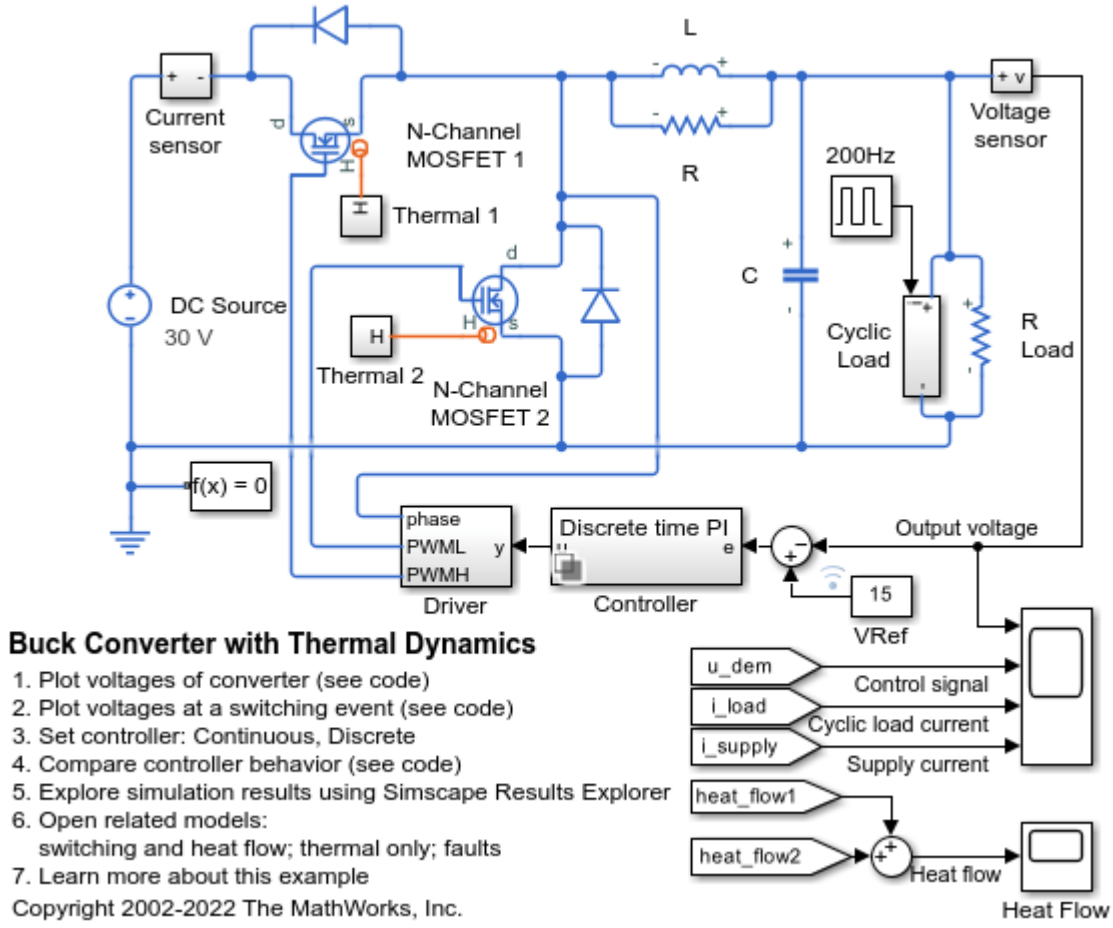
# Buck Converter with Thermal Dynamics

This example shows how to model a switching power supply that converts a 30V DC supply into a regulated 15V DC supply. The model can be used to both size the inductance L and smoothing capacitor C, as well as to design the feedback controller. By selecting between continuous and discrete controllers, the impact of discretization can be explored. Modeling the switching devices as MOSFETs rather than ideal switches ensures that device on-resistances are correctly represented. The model also captures the switch-on/switch-off timing of the devices, this depending primarily on the gate capacitance values and the PWM driver output resistance.

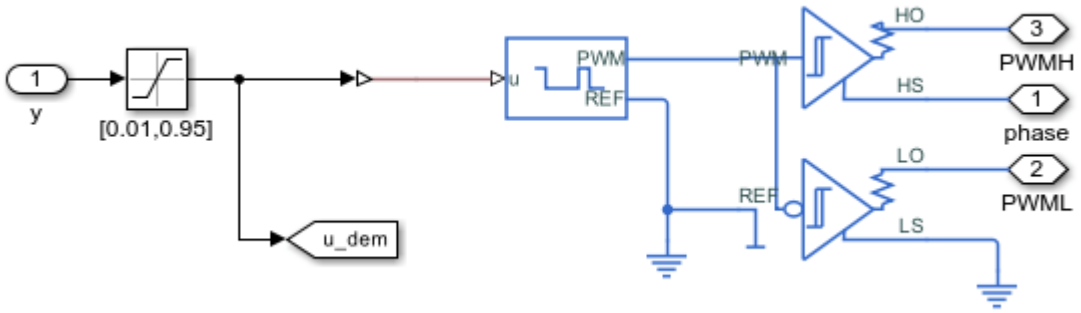
See example model power_switching_power_supply for an abstracted version of this model that uses ideal switching to give faster simulation times. The model here can be used to determine the on-resistance values required for the ideal switches, plus timing offsets if necessary. Using the ideal switching approach of power_switching_power_supply can be used to simulate more complex power converters.

The MOSFETs are configured to show thermal ports, these being connected to subsystems modeling the heatsinks and environment. The total heat flow is calculated and shown by Scope 3. For an abstracted model of the thermal dynamics only, see the 'Thermal Characteristics of a Synchronous Buck Converter' example, ee_switching_power_supply_thermal_only. The abstracted model is used to determine initial temperatures for this model. Diode thermal ports are not exposed as their thermal contribution is very small compared to that of the MOSFETs

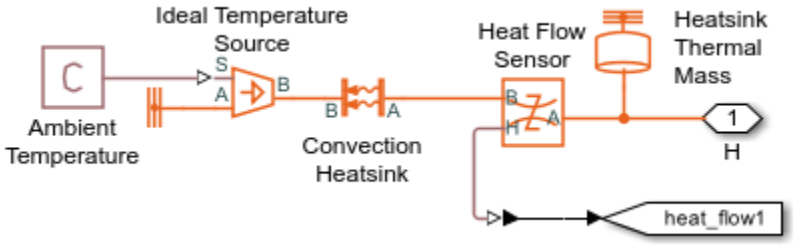
### Model




### Driver Subsystem

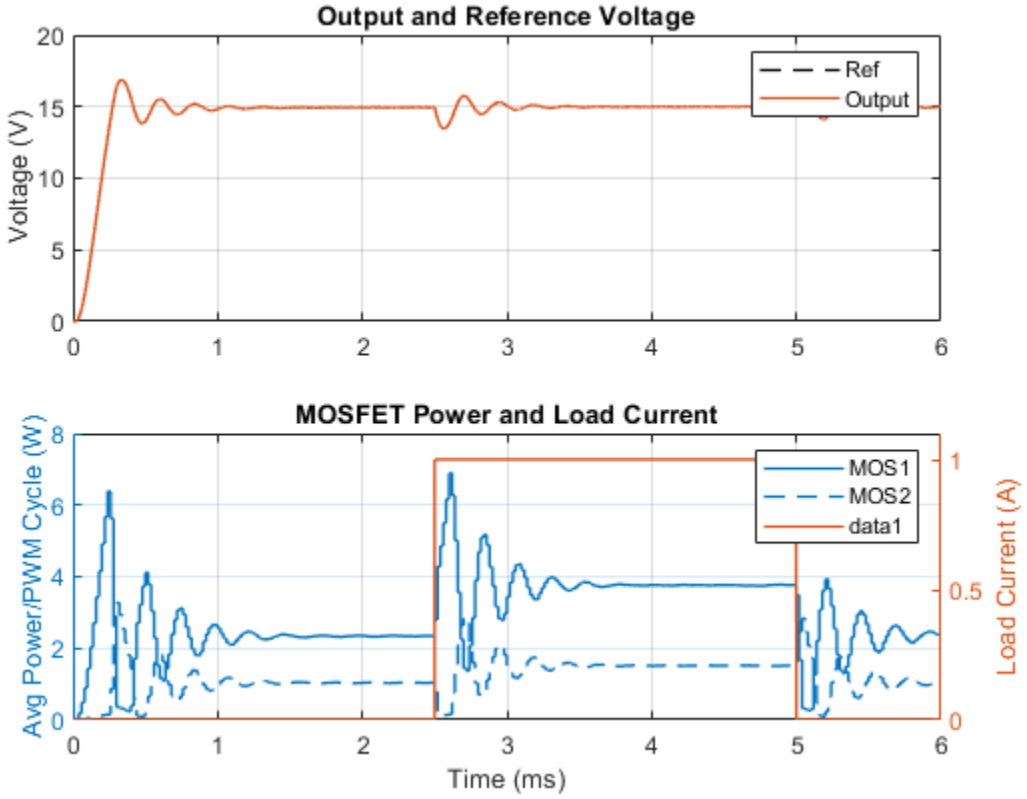


### Thermal 1 Subsystem

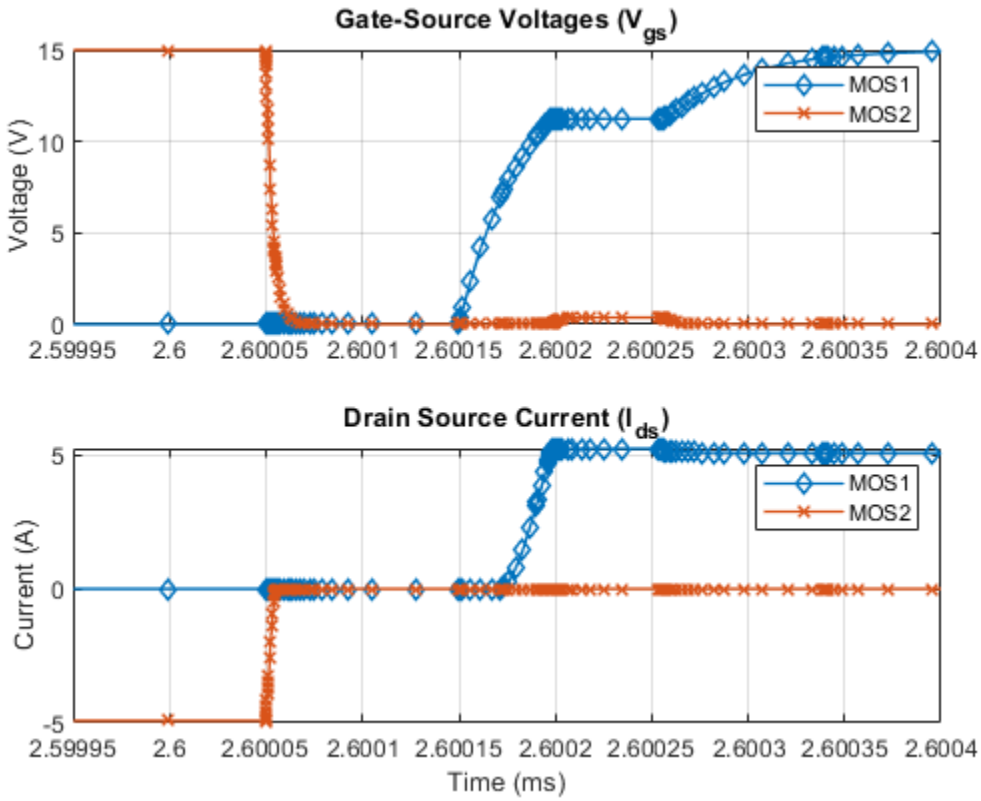



### Simulation Results from Simscape Logging

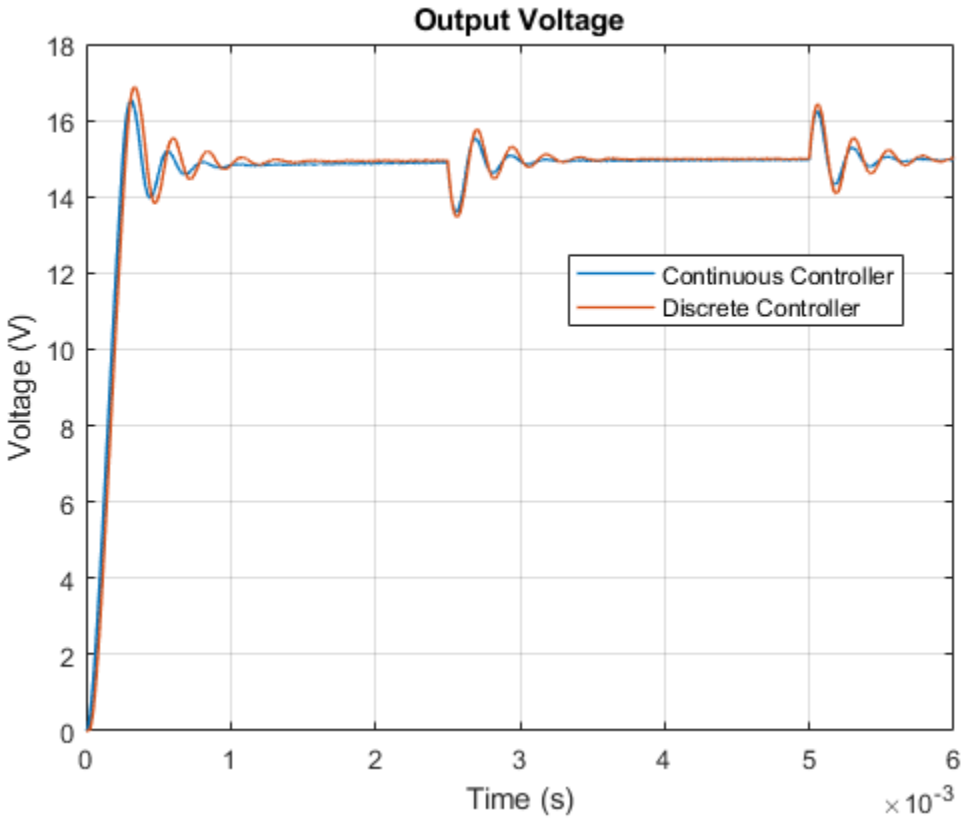
The plot below shows the output voltage as compared to the reference voltage. It also shows the changing load current and the dissipated power of the two MOSFETS averaged over the PWM cycle.




The plot below shows the switch-on/switch off timing of the two MOSFETS and the drain-source current.




The plots below show the behavior of the different implementations of the PI controller.

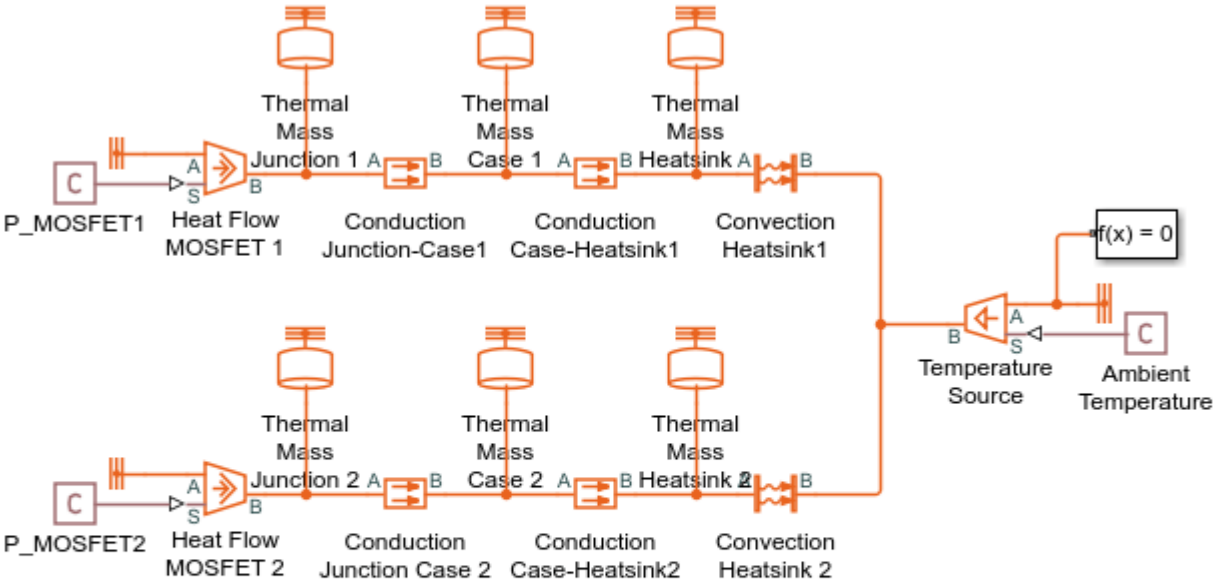

Output Voltage
Voltage (V)
Time (s)
×10^-3
Continuous Controller
Discrete Controller
0
2
4
6
8
10
12
14
16
18
0
1
2
3
4
5
6

# Buck Converter Thermal Model

This example models the thermal dynamics of MOSFETs in a synchronous buck converter. It matches the structure of the “Buck Converter with Thermal Dynamics” on page 8-669 model (>> ee_switching_power_supply_thermal). Omitting the electrical switching dynamics allows the simulation to take much larger time steps, dramatically reducing the amount of time it takes for the simulation to calculate steady-state temperatures for the MOSFETS.

The average losses during switching cycles are needed to determine the heat that the MOSFETs produce. This is obtained by running the detailed model (>> ee_switching_power_supply) which postprocesses the logged data and saves the losses to workspace variables P_MOSFET1 and P_MOSFET2. The simplified thermal model can then be run, and the final temperatures of the MOSFETs are saved to workspace variables T_junction1, T_case1, T_heatsink1, T_junction2, T_case2 and T_heatsink2. Re-running the detailed model using these temperatures as the starting conditions yields a more accurate result for the behavior of the system since the device characteristics depend on temperature.

### Model




**Buck Converter Thermal Model**

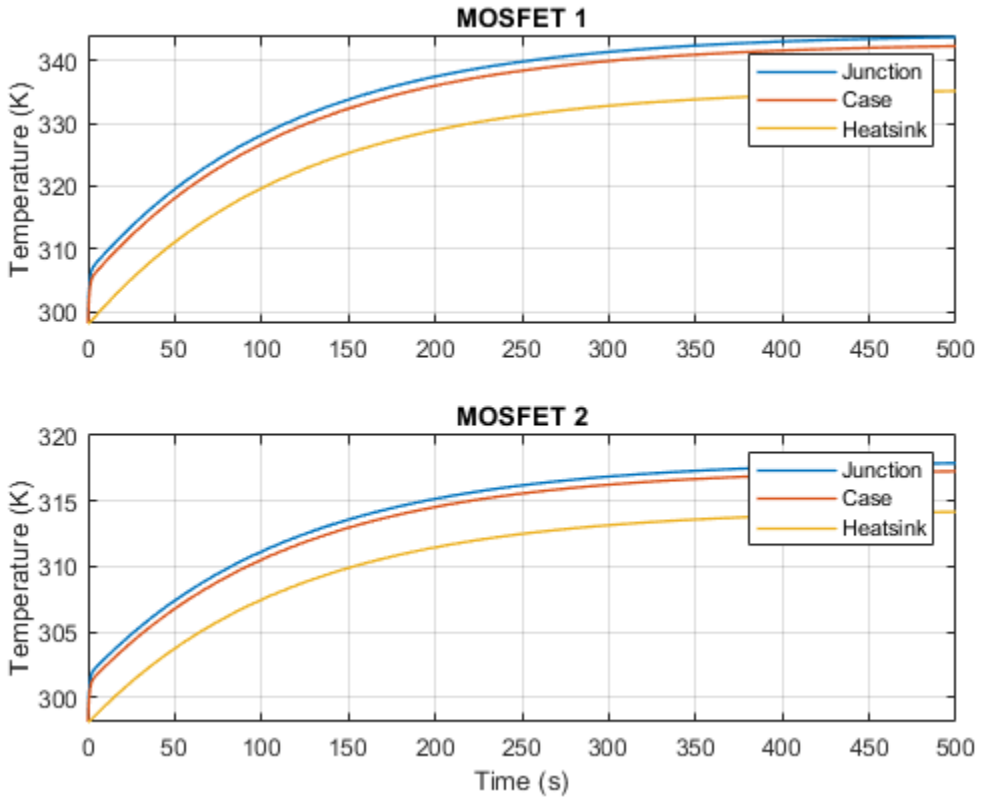
1. Plot temperatures of MOSFETS (see code)
2. Explore simulation results using Simscape Results Explorer
3. Open electrical model: switching; switching and heat flow; faults
4. Learn more about this example

Copyright 2002-2022 The MathWorks, Inc.

### Simulation Results from Simscape Logging

The plot below shows the temperature of the MOSFETS over time. Running the simulation for a long period of time enables the steady-state temperatures to be determined. The final temperatures of this

simulation can be used as the starting conditions for a detailed electrical simulation where the device characteristics depend on temperature.



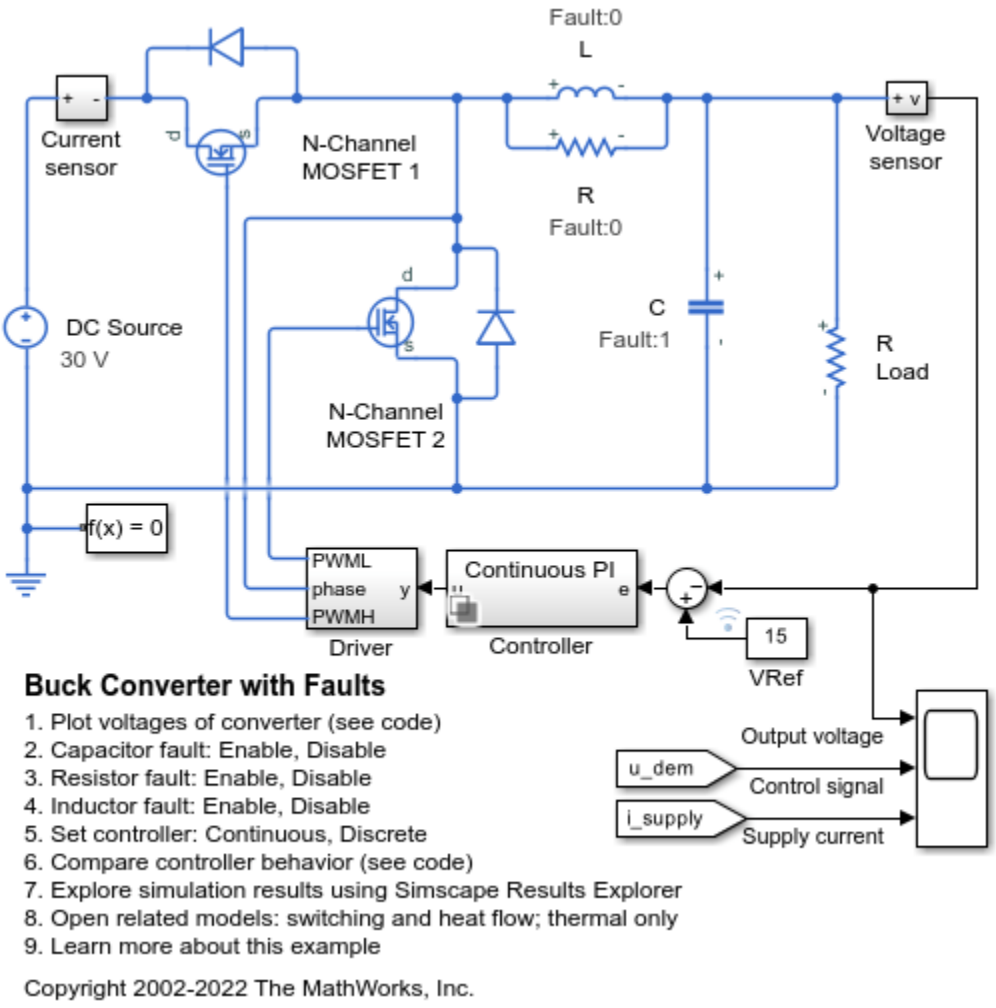

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 30 microseconds.

# Buck Converter with Faults

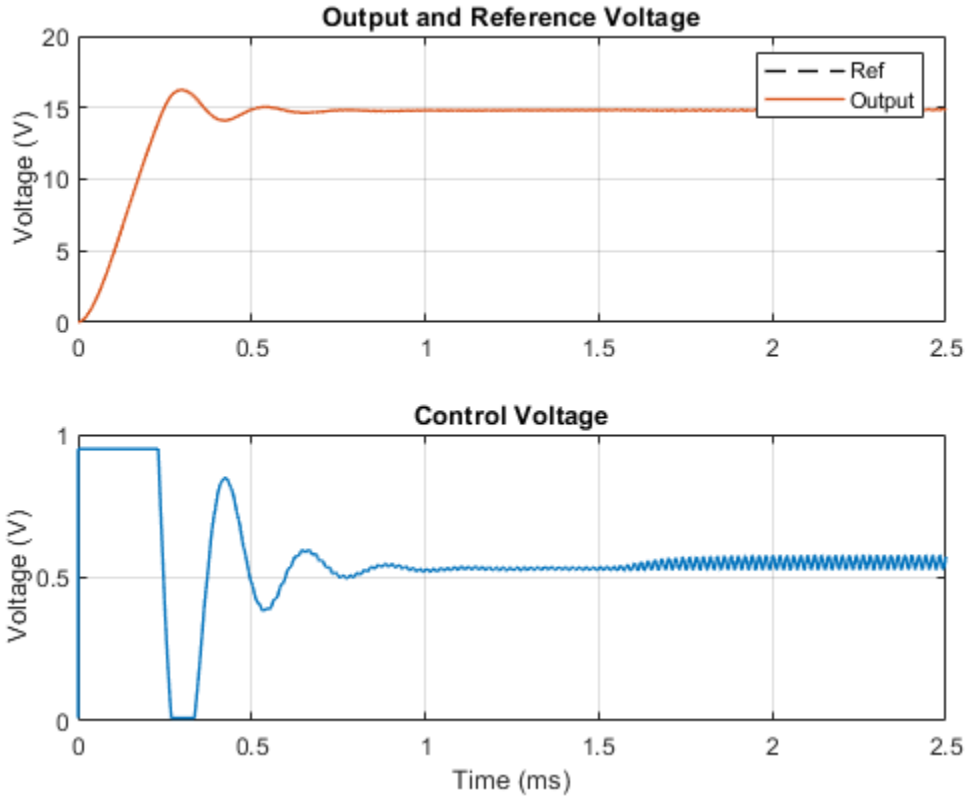
This example shows how to model and assess the impact of component tolerances and fault events on the operation of a switching power supply. The R, L, and C components all have tolerances, operational limits, and faults defined. The faults can be enabled within the block dialog or using MATLAB® Commands. The capacitor fault is already enabled to cut in at 1.5e-3 seconds.

### Model

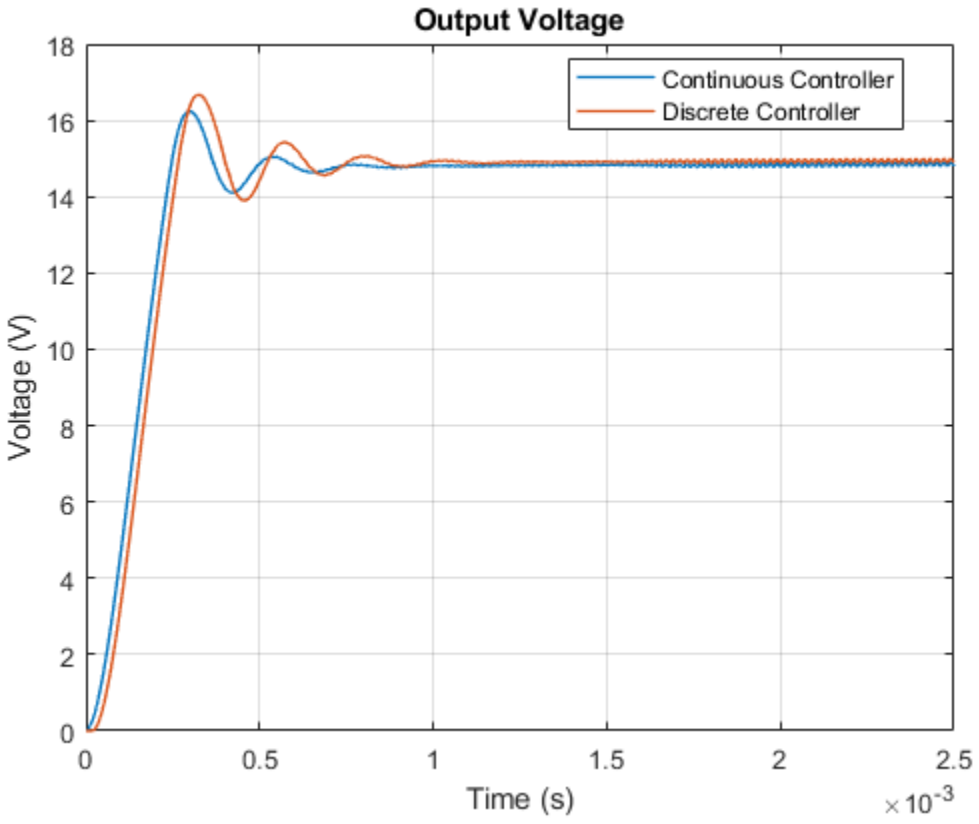



### Simulation Results from Simscape Logging

The plot below shows the output voltage as compared to the reference voltage. It also shows the control signal that adjusts the duty cycle of the PWM signals applied to the MOSFET gates. As the faults occur, the effect can be seen on the control signal and the voltage supplied by the converter.




The plots below show the behavior of the different implementations of the PI controller.

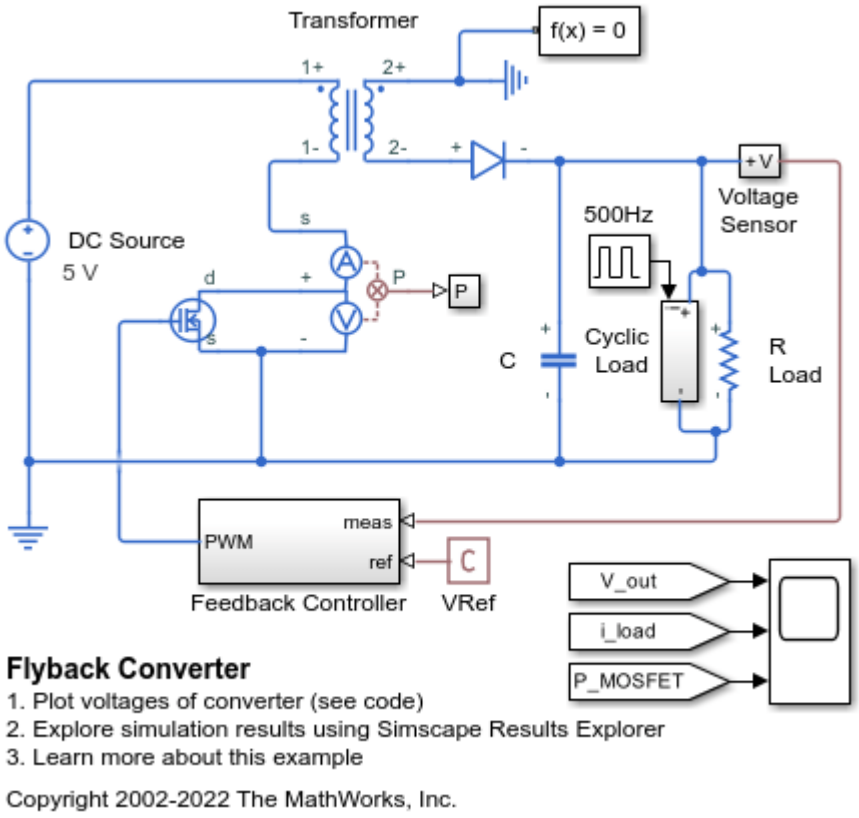

Output Voltage
Continuous Controller
Discrete Controller
Voltage (V)
Time (s)
0
2
4
6
8
10
12
14
16
18
0
0.5
1
1.5
2
2.5
×10-3

# Flyback Converter

This model shows how a flyback converter can step-up a 5V DC source into a 15V DC regulated supply. The voltage is increased by creating a time-varying voltage across a transformer primary. The transformer steps up the voltage which is then rectified back to DC by the diode. Closed-loop control over the output voltage is effected by controlling the switching frequency on the primary side.

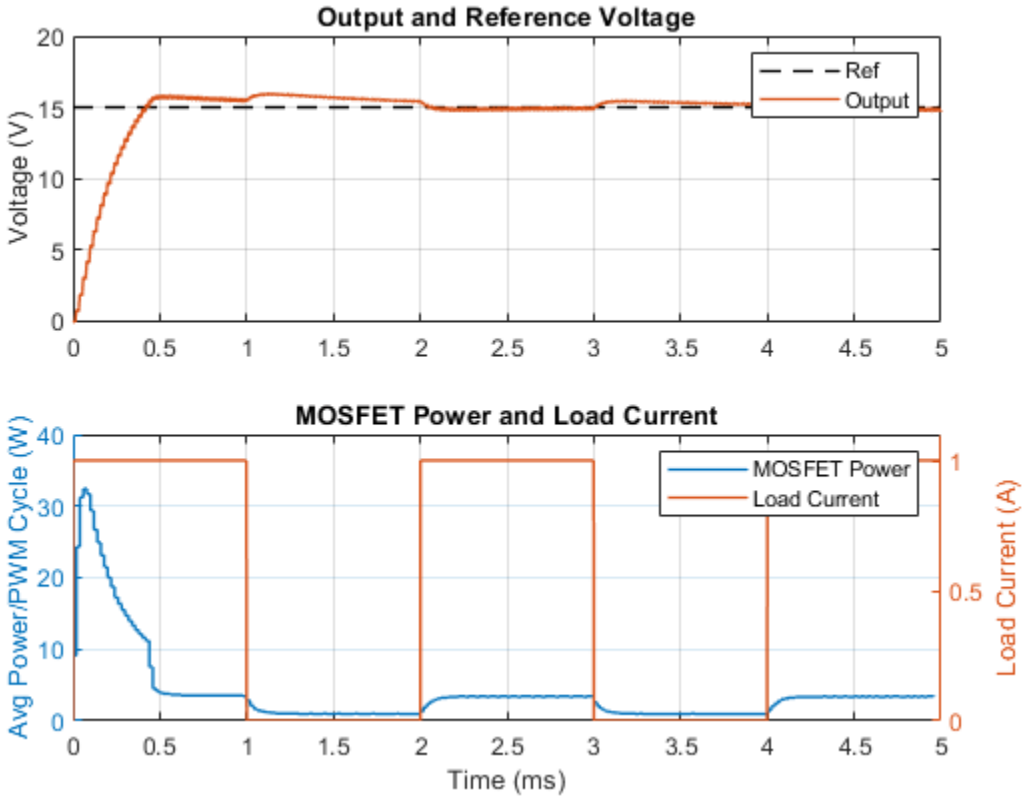
The model can be used to size the transformer inductance and smoothing capacitor values, as well as to design the feedback controller gains and switching frequency. The model includes a detailed representation of the switching device, in this case an enhancement-mode MOSFET, which lets you check that the switching device dissipated power is within acceptable limits. The Power Sensor block is used here in averaging mode, thereby calculating the average power over each PWM cycle. Postprocessing data available in Simscape™ logging can also show the power dissipated by the MOSFET.

### Model




### Simulation Results from Simscape Logging

The plot below shows the output voltage as compared to the reference voltage. It also shows the changing load current and the dissipated power of MOSFET averaged over the PWM cycle.

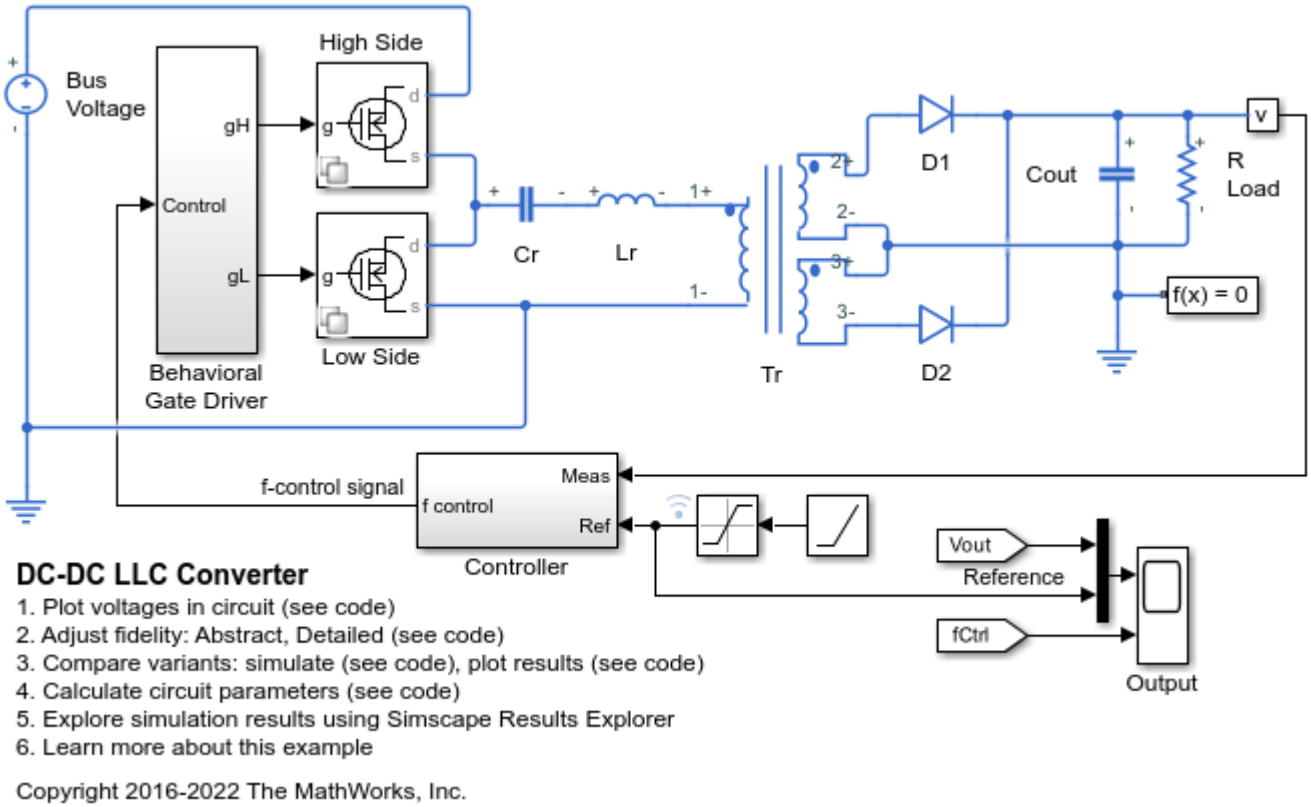

Output and Reference Voltage
Voltage (V)
Ref
Output
MOSFET Power and Load Current
Avg Power/PWM Cycle (W)
Load Current (A)
MOSFET Power
Load Current
Time (ms)

# DC-DC LLC Converter

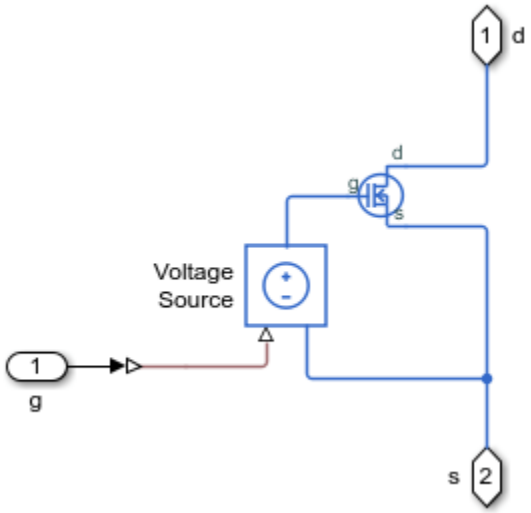
This example shows a DC-DC LLC power converter with frequency control. A simple integral control is implemented in Simulink® in the Controller block, and is designed to achieve a nominal output voltage defined by the variable Vout_nominal. The Output scope shows the frequency control signal, the output voltage, and the reference value for the output voltage. During startup, the reference value is ramped up to its desired setpoint. The design of the LLC powertrain is computed automatically using the first harmonic approximation.

This model contains two variants for the power electronic switches. The detailed version includes a MOSFET with nonlinear characteristics. The abstract version uses a piecewise linear model with an ideal switch, body diode, and output capacitance. The abstract version provides very similar behavior and quicker simulation.

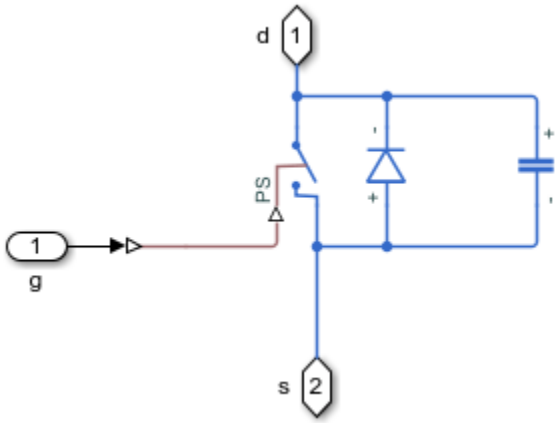
### Model



### Power Electronic Switch, Detailed Variant

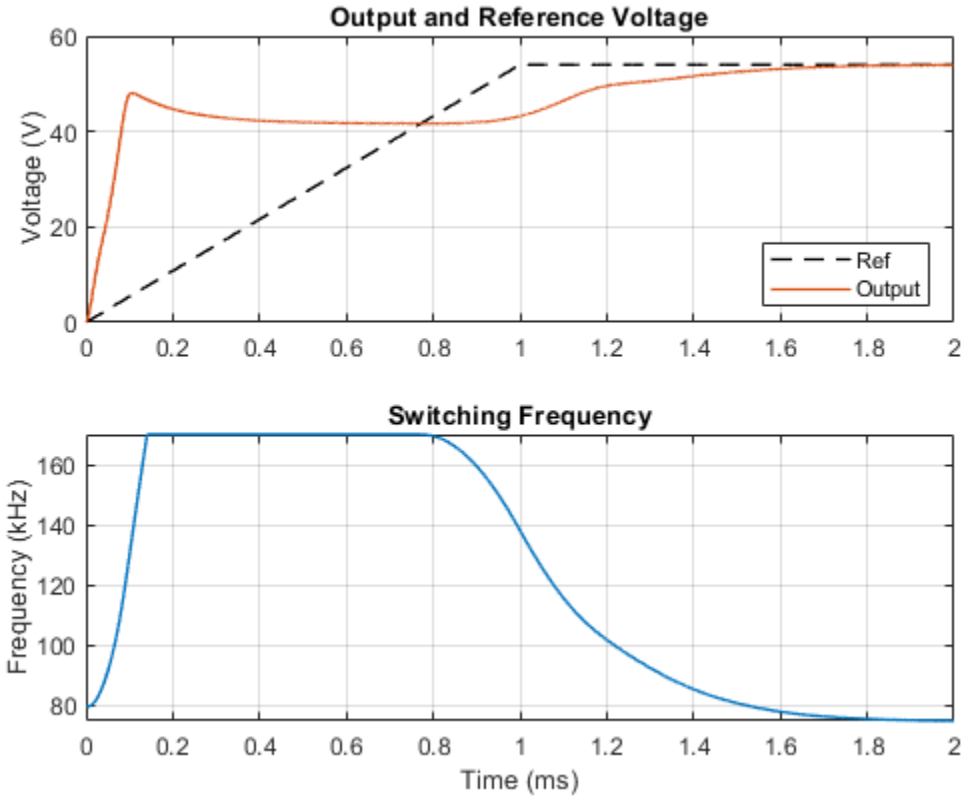



### Power Electronic Switch, Abstract Variant

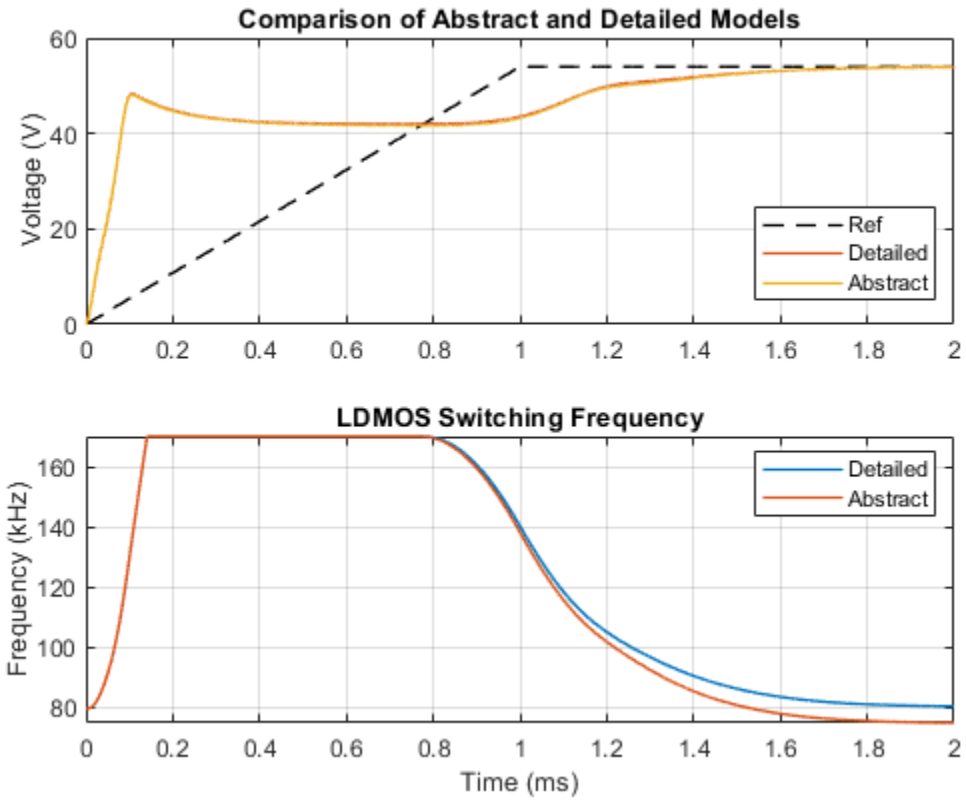


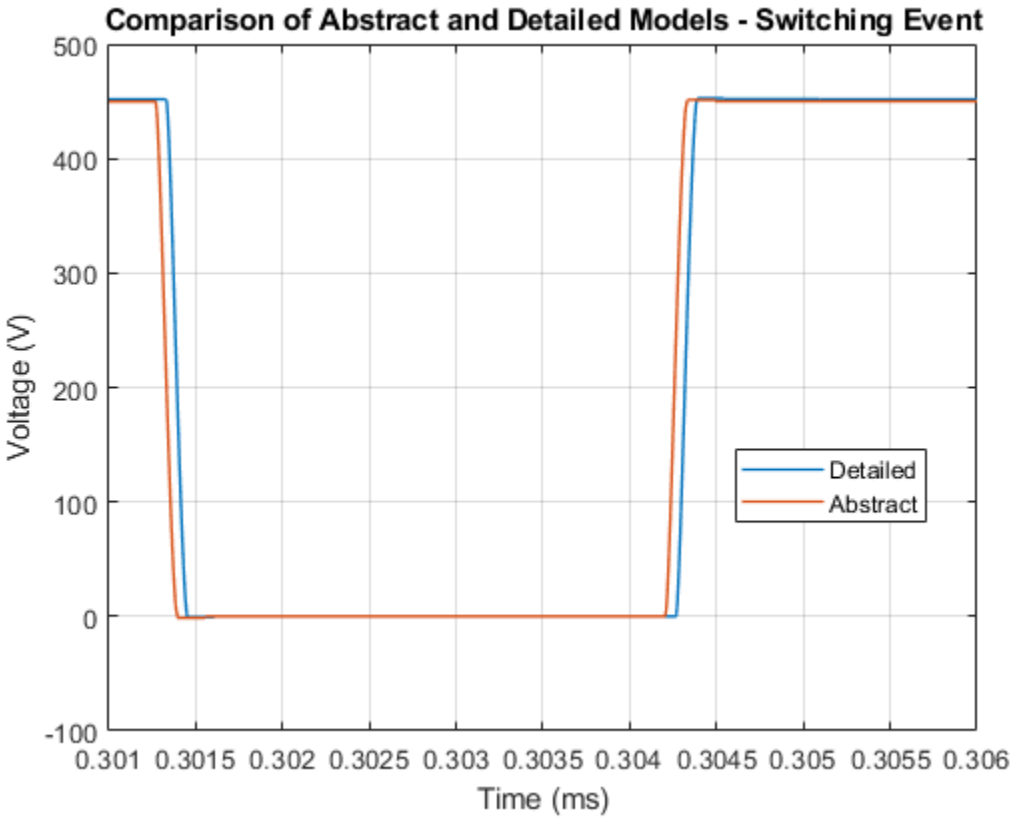

### Simulation Results from Simscape Logging

The plot below shows the output voltage as compared to the reference voltage. It also shows the variation in switching frequency as it is adjusted by the control system to track the reference voltage.




The plots below show the behavior of the abstract models and the detailed model. The results are very similar, indicating that the abstract models can be used without a significant loss of accuracy.

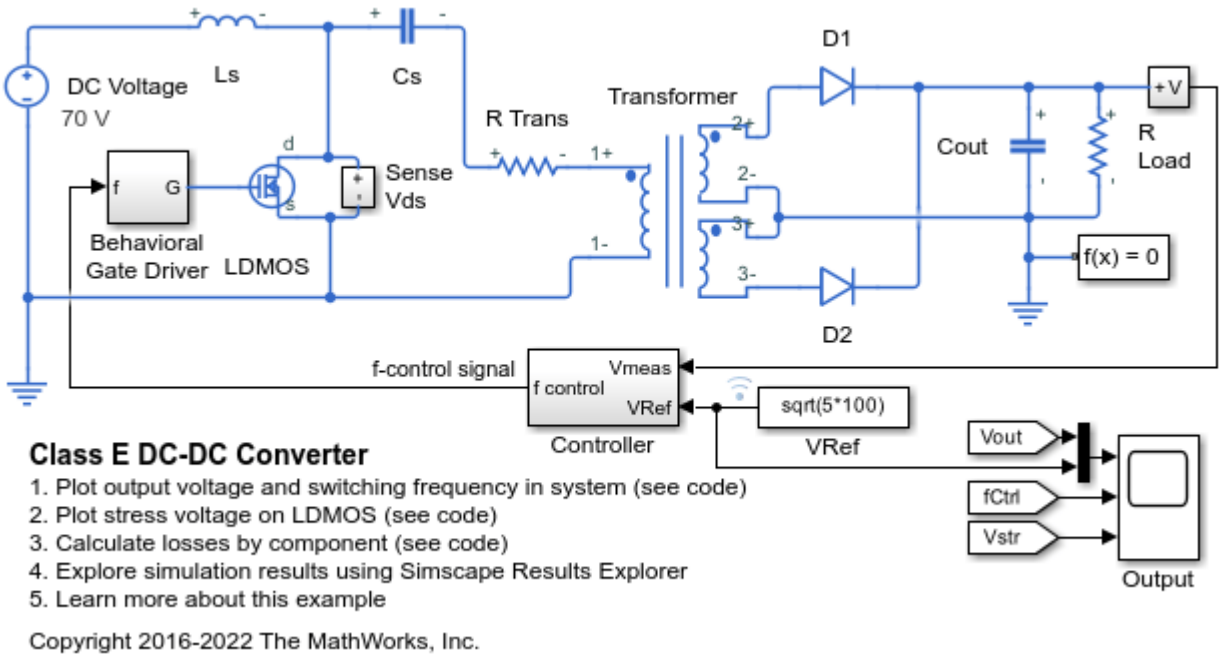

Comparison of Abstract and Detailed Models
Voltage (V)
60
40
20
0
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
Ref
Detailed
Abstract
LDMOS Switching Frequency
Frequency (kHz)
160
140
120
100
80
Detailed
Abstract
Time (ms)


Comparison of Abstract and Detailed Models - Switching Event
Voltage (V)
500
400
300
200
100
0
-100
0.301
0.3015
0.302
0.3025
0.303
0.3035
0.304
0.3045
0.305
0.3055
0.306
Time (ms)
Detailed
Abstract

# Class E DC-DC Converter

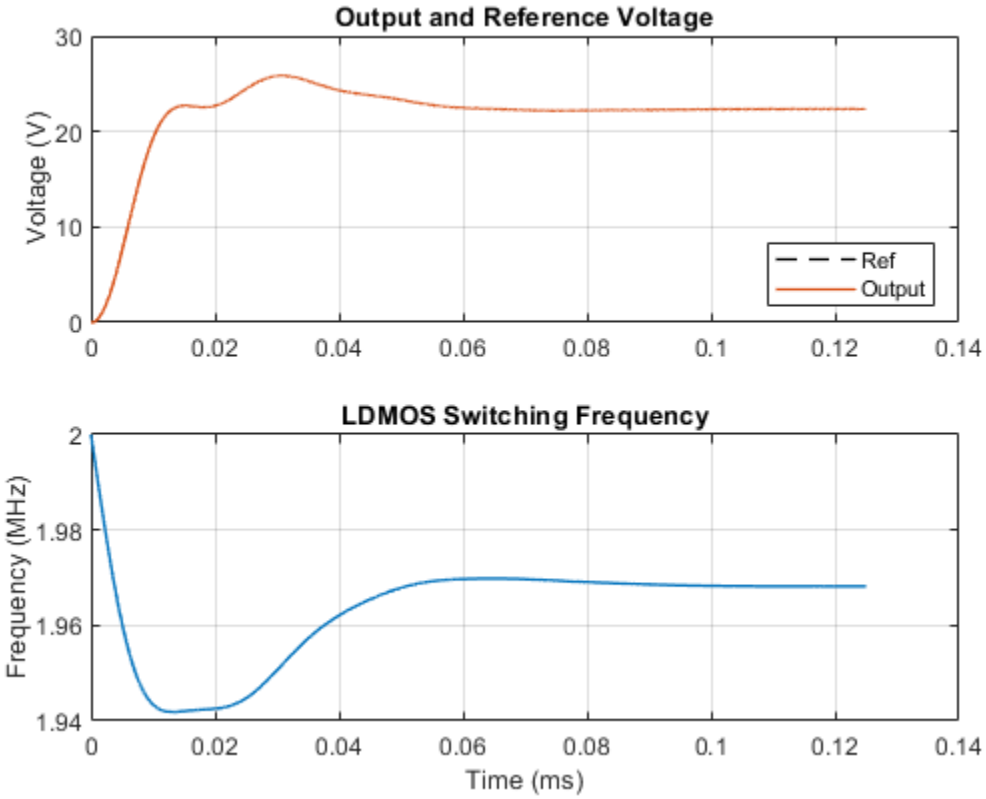
This example shows a Class E power converter with frequency control. A simple integral control is implemented in Simulink® in the Controller block, and is designed to deliver 100W into a 5ohm load. The switch is an LDMOS, high-voltage transistor with a nonlinear capacitance model, and R Trans is the equivalent series resistance of the transformer. The Output scope shows the drain-source voltage for evaluation of the voltage stress on the switch. Note that, due to the nonlinear output capacitance of the transistor, the peak voltage stress is higher than would be expected if the output capacitance were constant. In addition, the scope also shows the frequency control signal, the output voltage, and the reference value for the output voltage. This model can be used to calculate the output power information from components in the circuit.

### Model

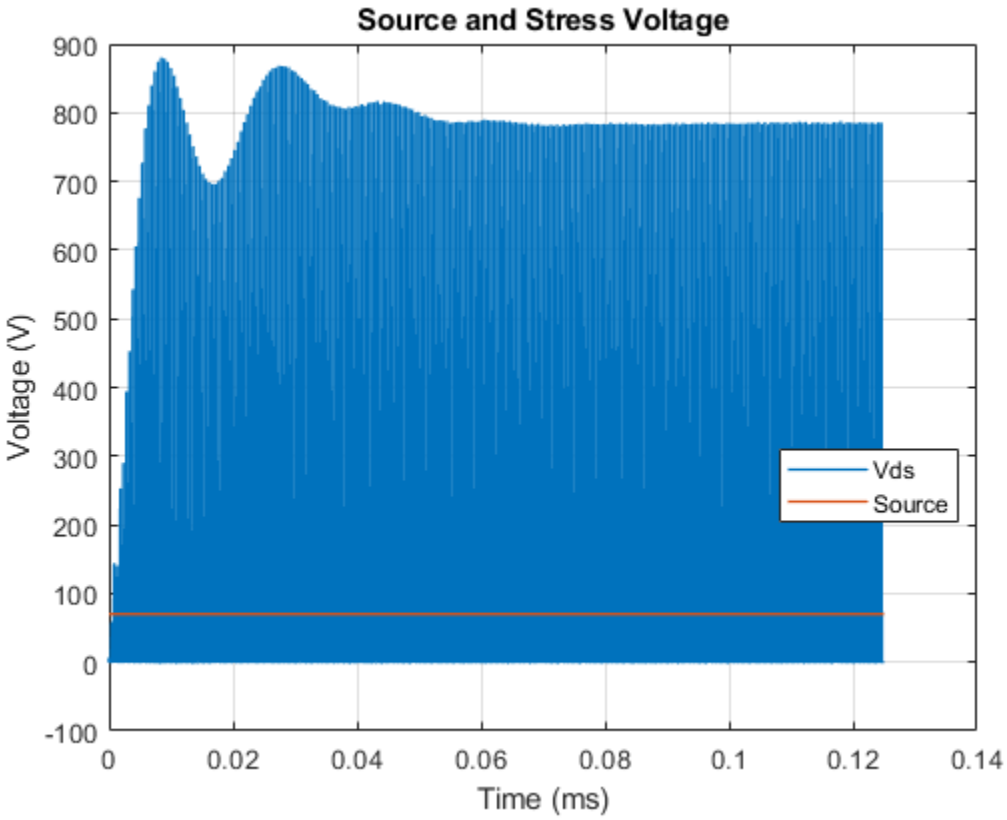



### Simulation Results from Simscape Logging

The plot below shows the output voltage as compared to the reference voltage. It also shows the change in switching frequency which is how the control system attempts to make the output track the reference voltage.




The plot below shows the voltage stress on the LDMOS as compared to the source voltage. This architecture for a power converter requires a significantly higher voltage stress than architectures with two semiconductor devices.



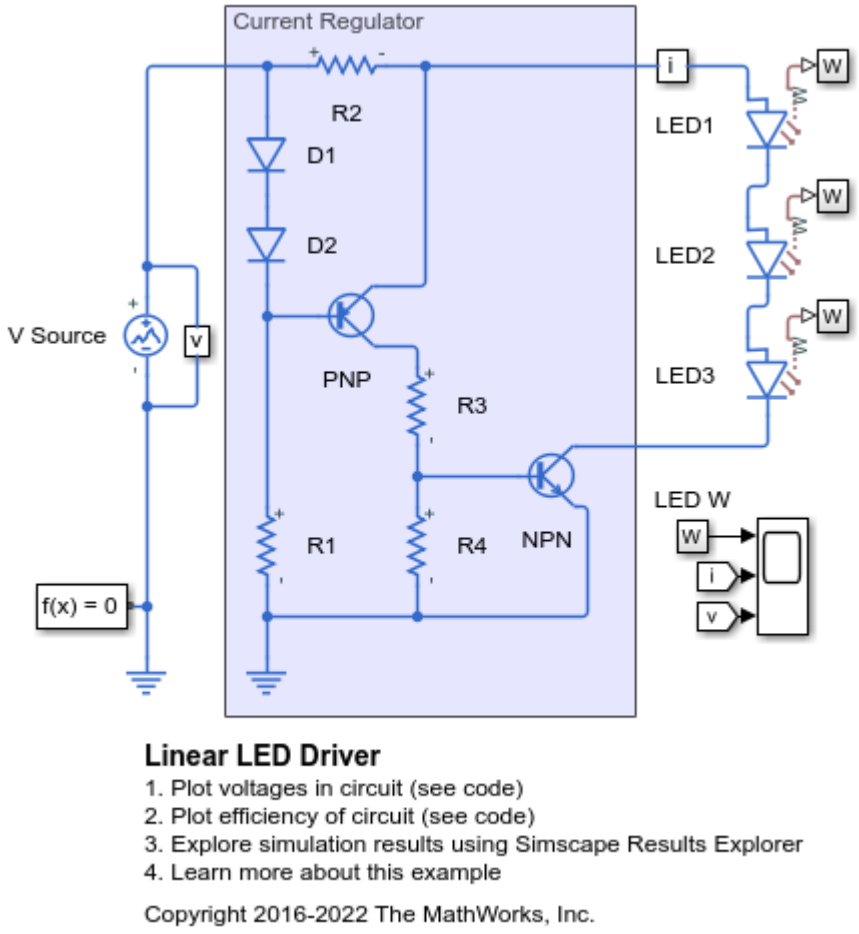

The table below shows the power dissipated by individual components in the ee_converter_dcdc_class_e model. These totals were calculated from simulation results using logged Simscape™ variables and the losses calculation utility ee_getPowerLossSummary. The total output power in the load, as well as the losses are shown.

```
Averaging power data from 0.0001s to 0.000125s.
Total efficiency = 90.4782% for an output power of 99.9017W.
Loss contributions are shown below.

                    LoggingNode                        Power
    ___________________________________________________    _____

    {'ee_converter_dcdc_class_e.LDMOS'           }    3.49
    {'ee_converter_dcdc_class_e.R_Trans.Resistor'}    2.81
    {'ee_converter_dcdc_class_e.D2'              }    1.91
    {'ee_converter_dcdc_class_e.D1'              }    1.79
    {'ee_converter_dcdc_class_e.Cs'              }    0.26
    {'ee_converter_dcdc_class_e.Ls'              }    0.25
    {'ee_converter_dcdc_class_e.Cout'            }       0
```

# Linear LED Driver

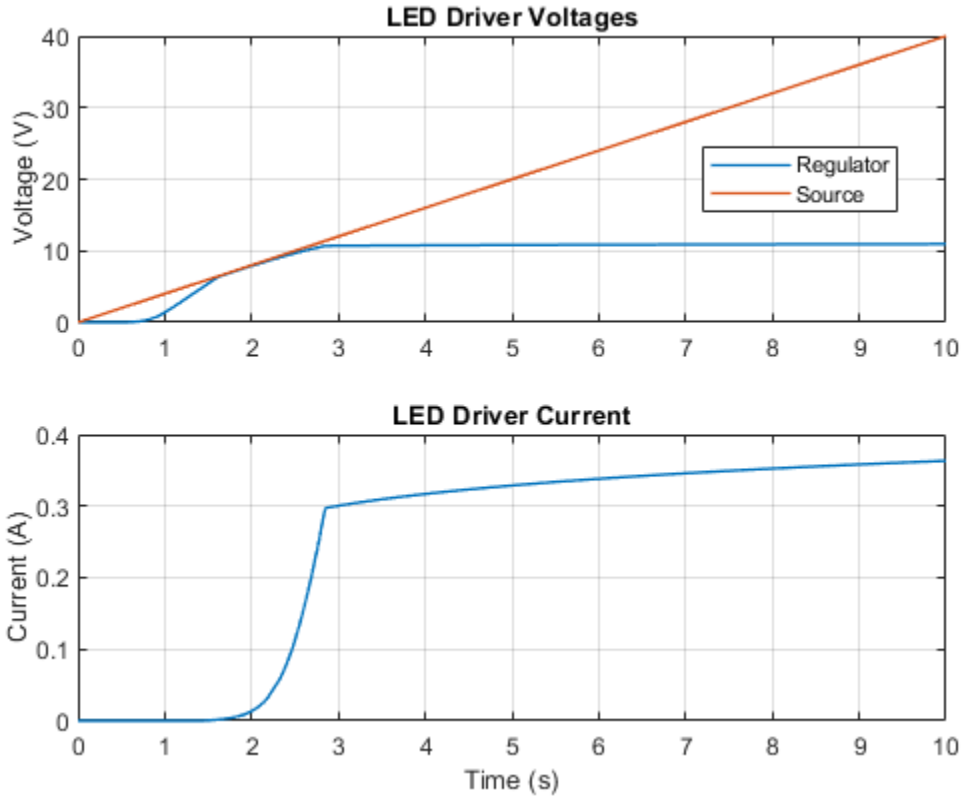
This example shows a LED driver based on a linear current regulator. The scope shows the light and current output and the supply voltage. The output comes into regulation for a supply voltage greater than about 12V.

### Model

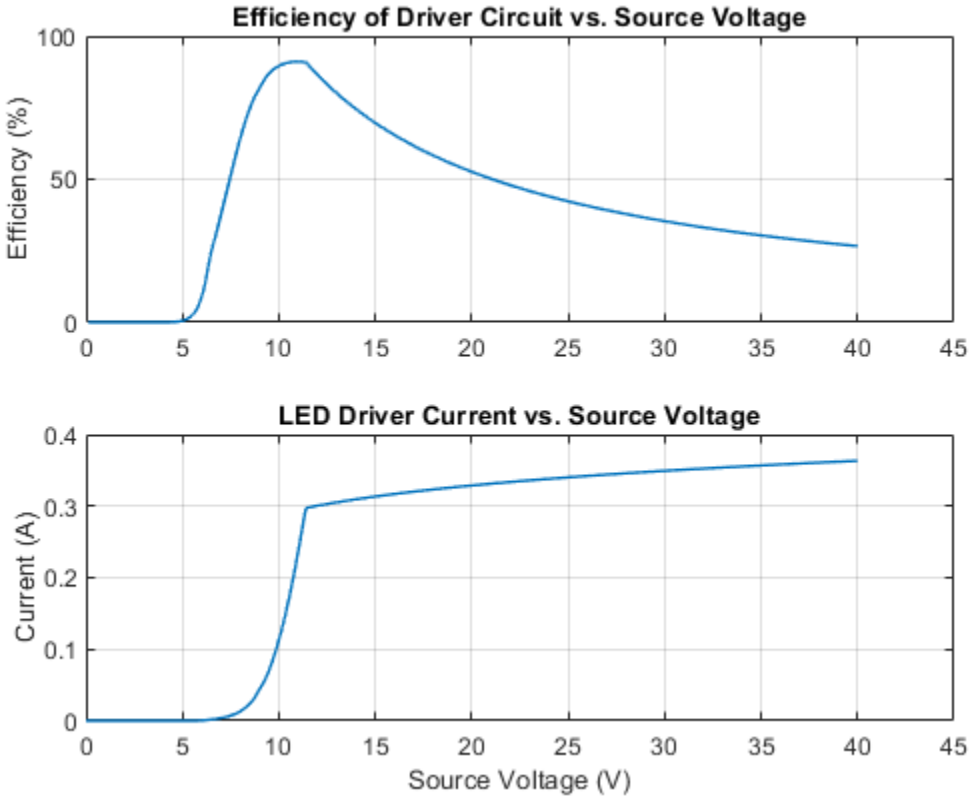



### Simulation Results from Simscape Logging

The plot below shows the output of the current regulator used in an LED driver circuit. Once the supply voltage reaches 12V, the driver circuit regulates its output.




The plot below shows the driver efficiency and LED current as functions of the supply voltage.


Efficiency of Driver Circuit vs. Source Voltage
Efficiency (%)
100
50
0
0
5
10
15
20
25
30
35
40
45
LED Driver Current vs. Source Voltage
Current (A)
0.4
0.3
0.2
0.1
0
0
5
10
15
20
25
30
35
40
45
Source Voltage (V)

# TVS Diode Parameterization

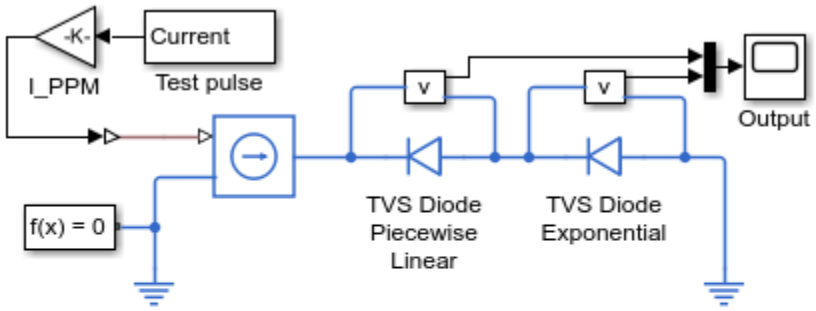
This model shows how to parameterize the Simscape™ Electrical™ diode to represent a Transient Voltage Suppression (TVS) diode. This example is for a TVS diode suited to protecting automotive electronics from voltage transients associated with turning off inductive loads. To view the data extracted from the datasheet, on the Modeling tab, in the Setup section, click Model Settings > Model Properties. On the Callbacks tab, click PreLoadFcn.

Two model options are shown here, the first using the Piecewise Linear Zener diode option, and the second using the Exponential diode option. The two diode blocks are parameterized directly in terms of the workspace datasheet values. Both models are tested here using a 10/1000us test pulse which raises the current to the peak pulse current I_PPM in 10us, and then lets it decay to I_PPM/2 in 1000us. The two scopes show that the max clamping voltage V_C of 18.8V is achieved to an acceptable tolerance.

The piecewise linear diode is the simplest to parameterize. The forward bias characteristics do not need to be accurate as a TVS diode is not operated in this region. A typical forward voltage drop of 0.6V is assumed. Given the [V_F I_F] forward bias data point, the on-resistance is given by (V_F-0.6)/I_F. The reverse bias data for the breakdown voltage V_BR and corresponding current I_T can then be used to determine the off conductance which is equal to I_T/V_BR. The diode reverse breakdown voltage, Vz, is then set equal to V_BR. Finally, zener resistance, Rz, is set so that when the current is I_PPM, the voltage is equal to the specified maximum clamping voltage V_C. A value of (V_C-V_BR)/I_PPM achieves this.

The exponential diode option models the reverse leakage current more accurately as the dependence on voltage is not assumed linear. By choosing to parameterize the exponential diode in terms of IS and N, the value of IS can be set equal to the reverse leakage current just before breakdown. The emission coefficient N is left as the default typical value of 1. Reverse breakdown voltage, BV, is set equal to V_BR. Finally, to ensure that the correct maximum clamping voltage V_C is achieved, the ohmic resistance, RS, must be appropriately set. This can either be by trial and error, or by approximate value of (V_C-V_BR-0.6)/I_PPM.

**Model**



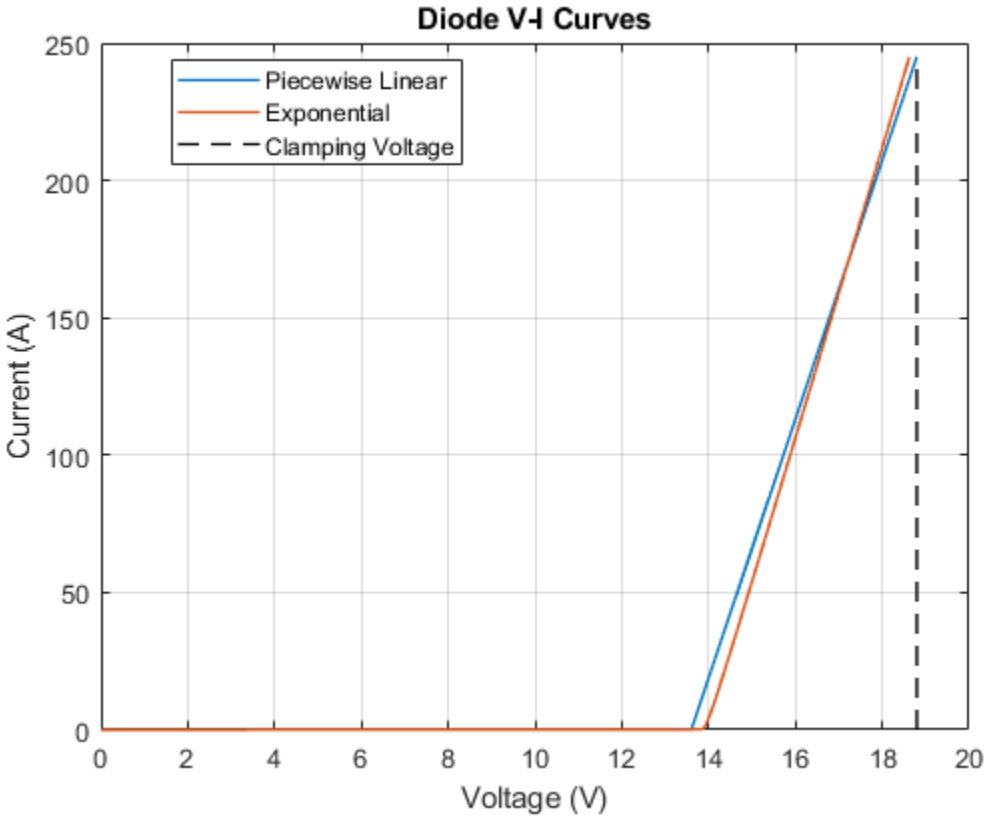

**TVS Diode Parameterization**

1. Plot v-i curves for diodes (see code)
2. Explore simulation results using Simscape Results Explorer
3. Learn more about this example

Copyright 2012-2022 The MathWorks, Inc.

### Simulation Results from Simscape Logging

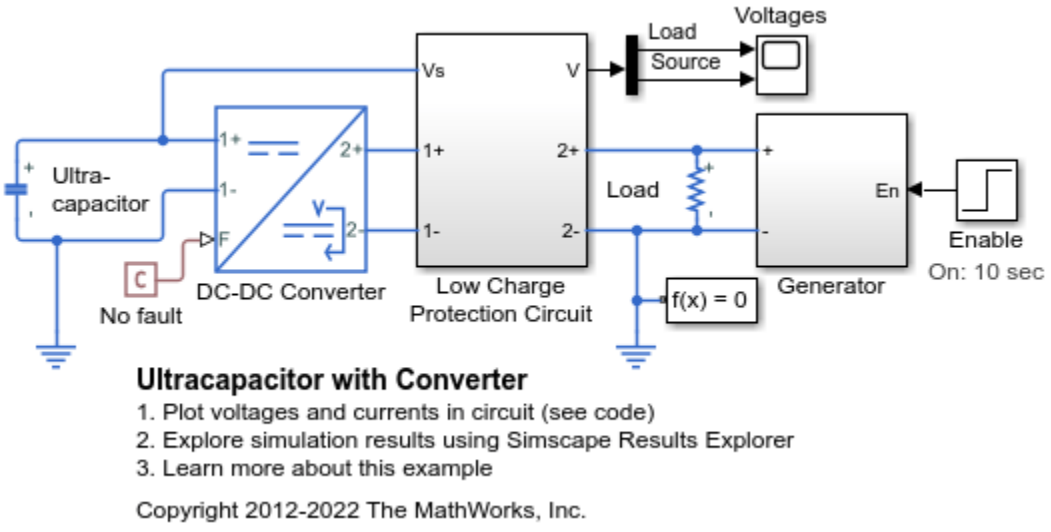
The plot below shows the voltage-current curves for two different diode models. The piecewise linear model and exponential model can provide similar behavior depending upon how they are parameterized.



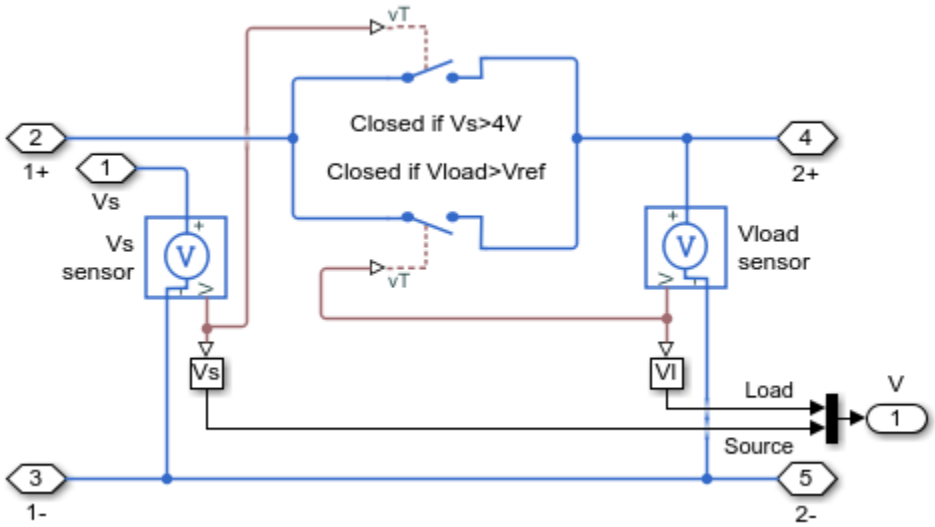
# Ultracapacitor with Converter

This example shows how a DC-DC converter can be used to maintain a constant load voltage when drawing power from an ultracapacitor. Initially the converter supplies power to the load, and as it does so, the capacitor voltage drops. The protection circuit disconnects the load when the capacitor voltage drops below a threshold of 4V. At 10 seconds the generator is turned on, and power is supplied to both the load and to the capacitor to recharge it.

### Model

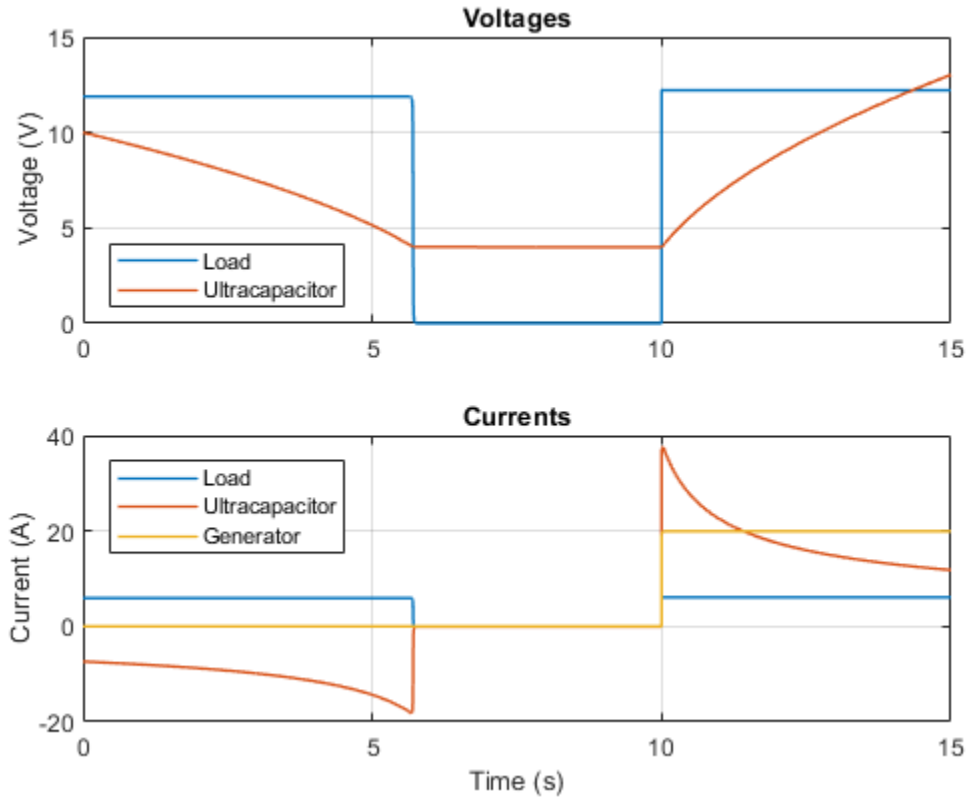



### Low Charge Protection Circuit Subsystem



### Simulation Results from Simscape Logging

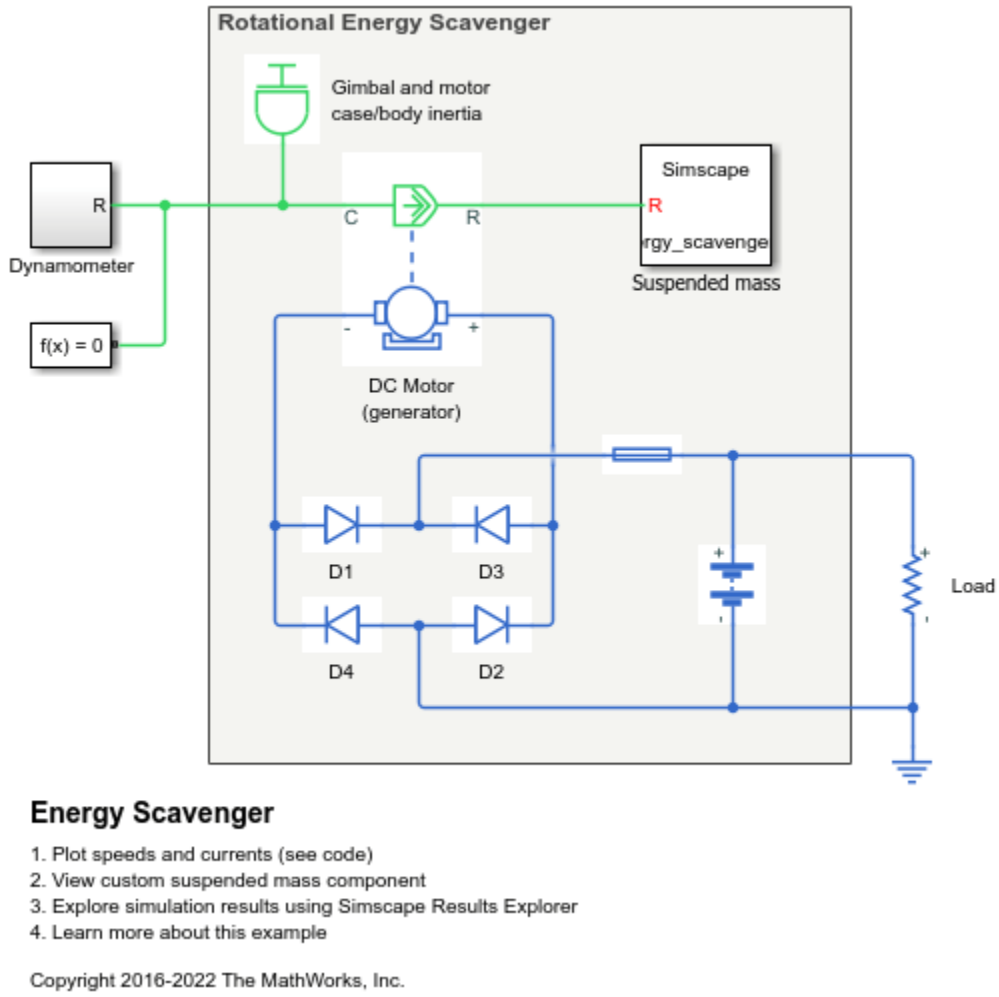
The plot below shows voltage supplied to a resistive load from an DC-DC converter and a generator. The DC-DC converter supplies power to the load. A low charge protection circuit disconnects the load when the ultracapacitor voltage drops below 4V. At 10 seconds the generator turns on, supplying power to the load and charging the ultracapacitor.



# Energy Scavenger

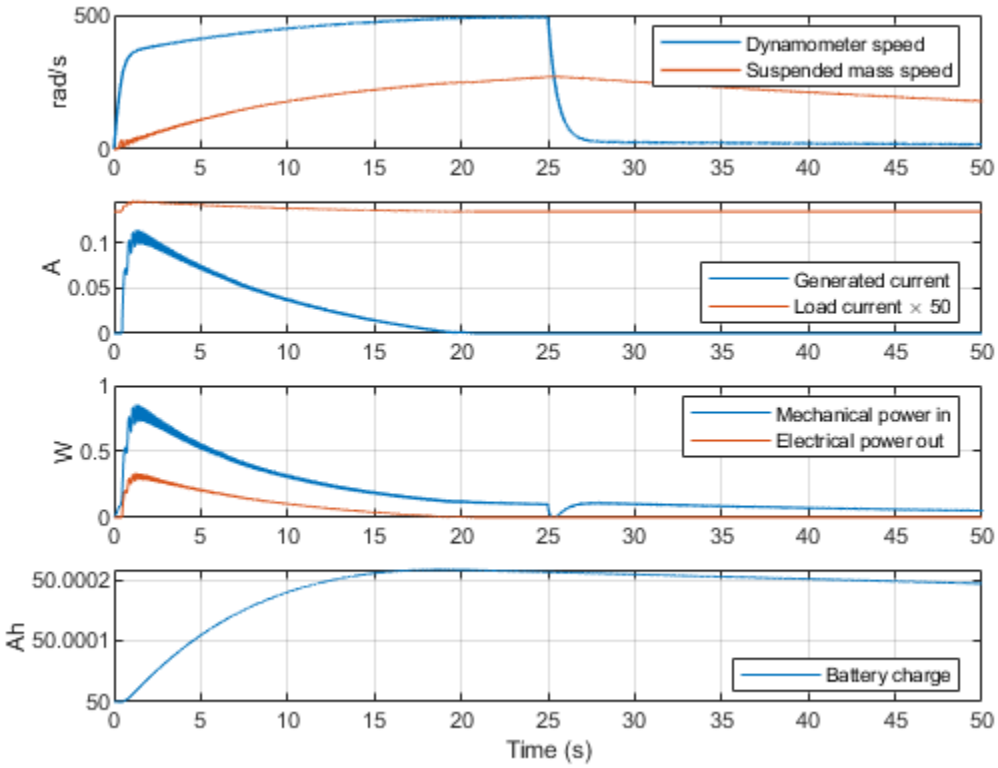
This model shows how the performance of a rotational energy scavenger can be explored using a simple representative model. Electrical energy is produced from an off-center mass attached to the shaft of a DC motor. The mass, geometry, motor and electrical parameters must be matched to the expected mechanical excitation. The generated electrical power is less than the extracted mechanical power primarily due to motor winding losses and viscous damping for the rotor. This example is based on Nunna, K. "Constructive interconnection and damping assignment passivity-based control with applications", Imperial College London (2014). The model here is simplified in that the DC-DC converter is omitted.

### Model




### Simulation Results from Simscape Logging

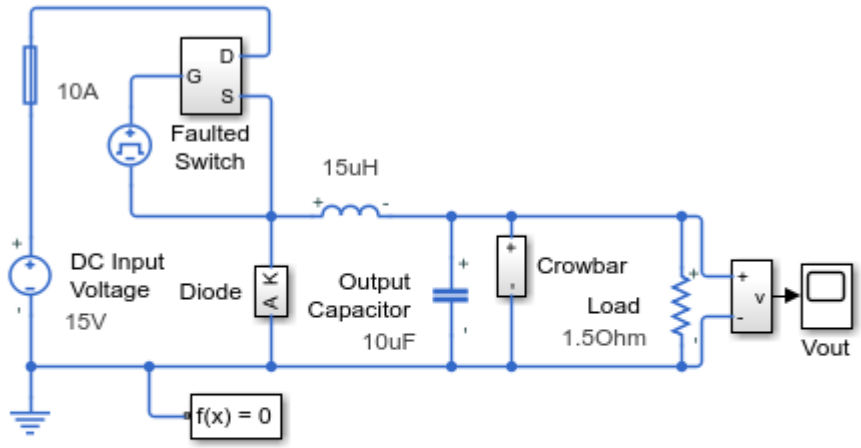
The plot below shows the dynamometer and suspended mass speeds along with generated and load currents plus battery charge.


rad/s
500
0
Dynamometer speed
Suspended mass speed
A
0.1
0.05
0
Generated current
Load current × 50
W
1
0.5
0
Mechanical power in
Electrical power out
Ah
50.0002
50.0001
50
Battery charge
0 5 10 15 20 25 30 35 40 45 50
Time (s)

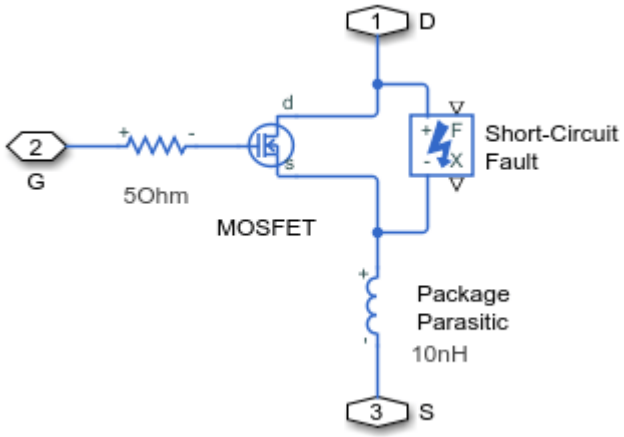
# MOSFET Fault in Buck Converter

This model shows how a fault may be applied to a MOSFET in a power converter in order to explore the operation of protection circuitry. After the MOSFET becomes faulted, the crowbar circuitry is activated in order to clamp the output voltage across the load and eventually to cause the fuse to blow.

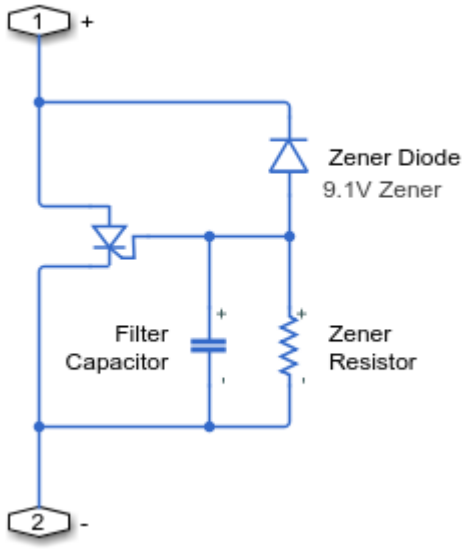
### Model




### Faulted Switch Subsystem

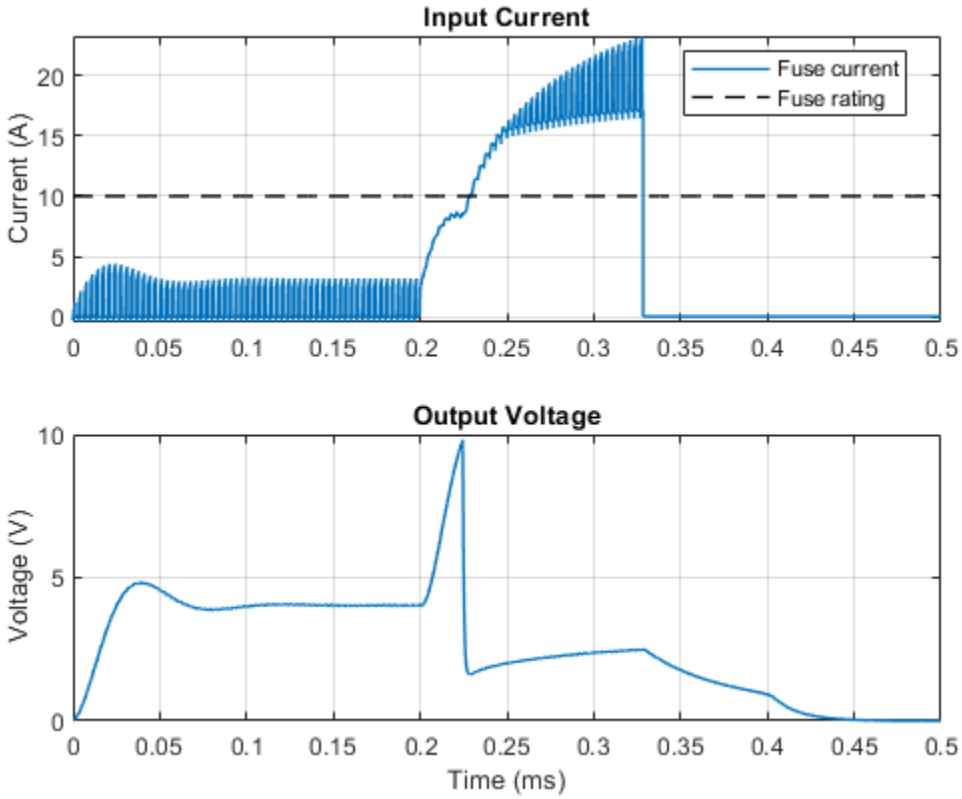


### Crowbar Subsystem




### Simulation Results from Simscape Logging

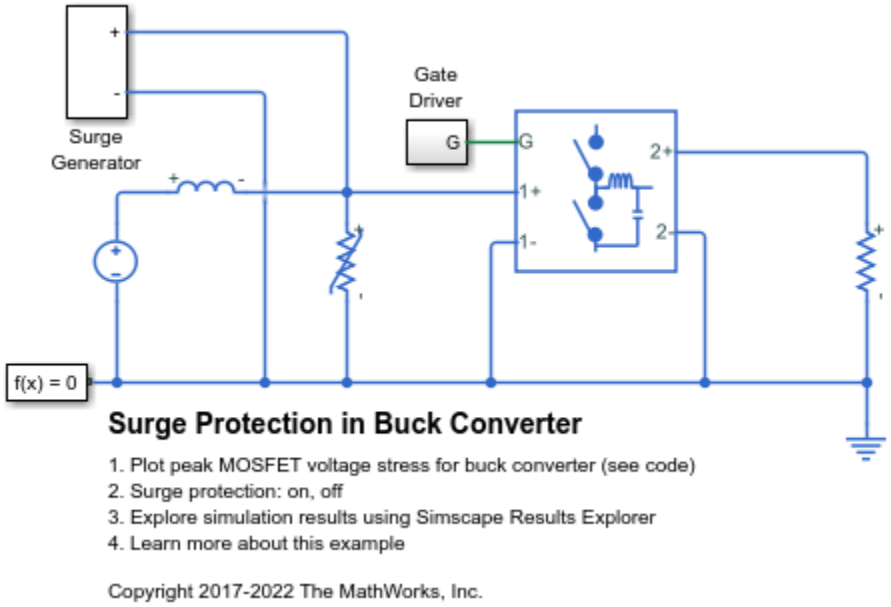
The plot below shows the current passing through the input Fuse and the voltage across the Load resistor.


Input Current
Fuse current
Fuse rating
Current (A)
20
15
10
5
0
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5
Output Voltage
Voltage (V)
10
5
0
Time (ms)

# Surge Protection in Buck Converter

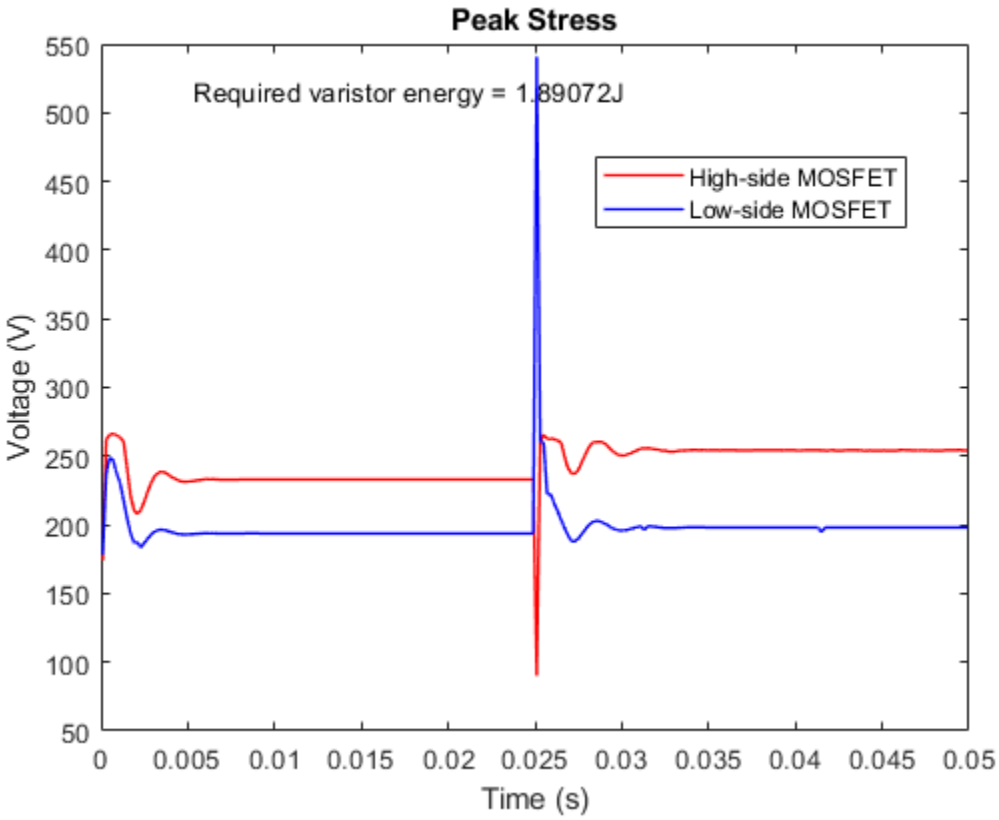
This model shows how a varistor may be applied to a buck converter in order to protect the switching MOSFETs from over-voltages due to a differential surge.

### Model




### Simulation Results from Simscape Logging

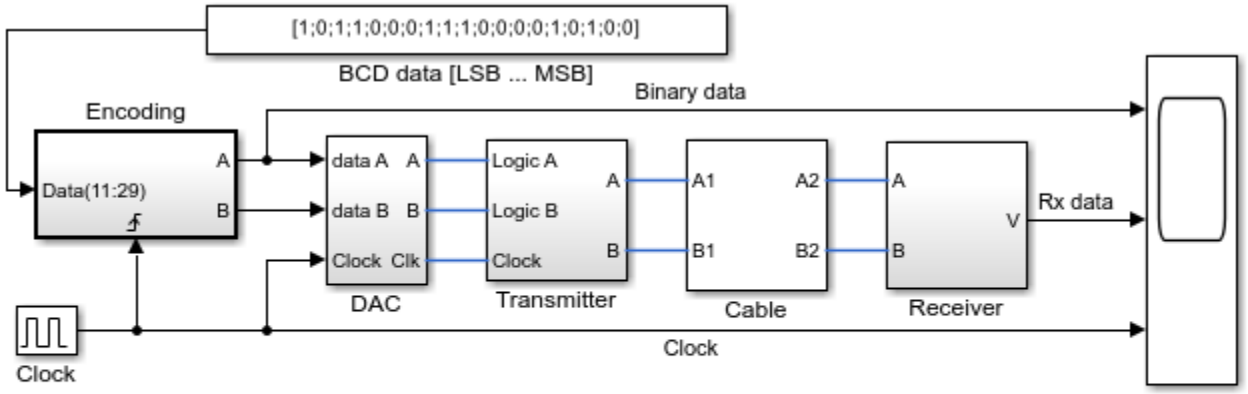
The plot below shows the peak voltage stress on each MOSFET of the buck converter on a cycle-by-cycle basis.


Peak Stress
Required varistor energy = 1.89072J
High-side MOSFET
Low-side MOSFET
Voltage (V)
Time (s)
550
500
450
400
350
300
250
200
150
100
50
0
0.005
0.01
0.015
0.02
0.025
0.03
0.035
0.04
0.045
0.05

# ARINC 429 Communication Link

This example shows a communication link that follows the ARINC 429 specification. Here it is configured for 100kBits per second operation. The ARINC 429 specification is for a simplex broadcast bus with odd parity checking. The cable is a balanced line comprising a twisted pair for data wires A and B, plus an outer shield. This model can be used to check compliance with the ARINC 429 specification, and to assess the impact of different cable lengths, cable parameters and abstracted transmitter and receiver characteristics. It can also be used to check operation with multiple receivers, the ARINC 429 specification allowing for up to 20 receivers to connect to a single transmitter.

**Model**

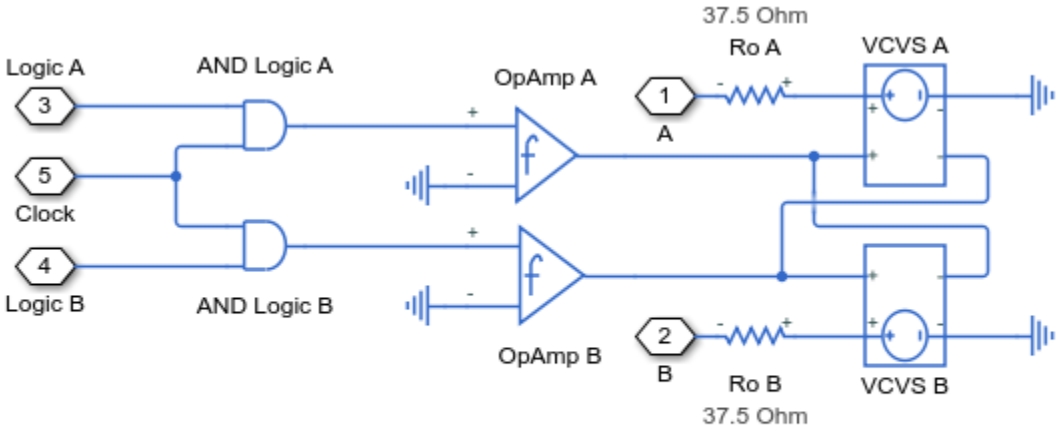



**ARINC 429 Communication Link**

1. Explore simulation results using Simscape Results Explorer
2. Learn more about this example

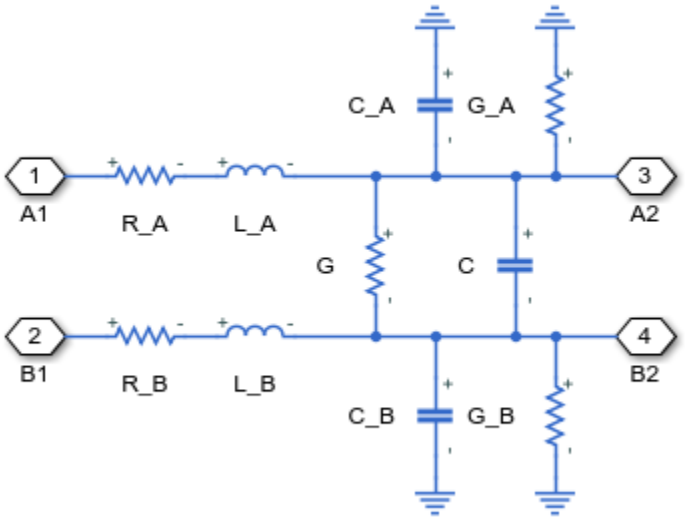
Copyright 2009-2022 The MathWorks, Inc.

### Transmitter Subsystem

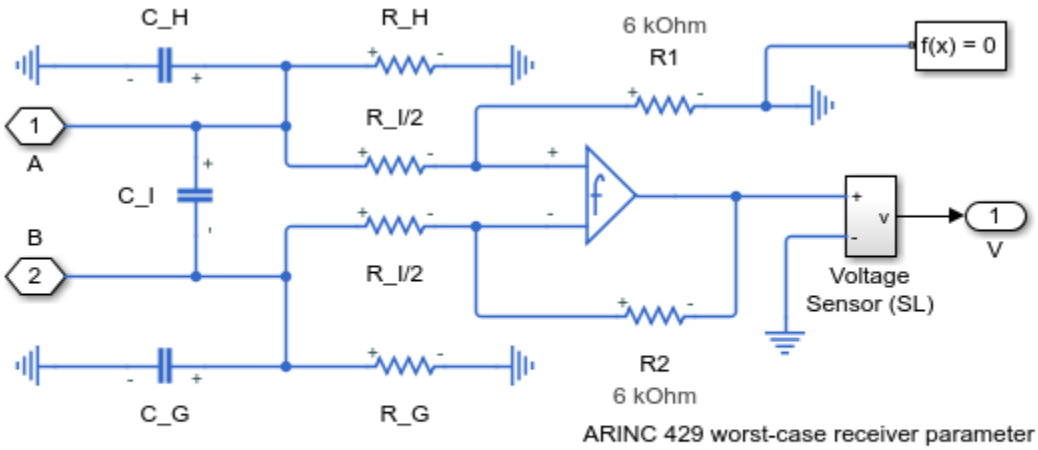



Level and slope control is implemented using a Band-Limited Op-Amp. The maximum slew rate parameter is used to shape the signal to comply with the ARINC 429 specification. As such, this is a simple behavioral model that replicates the input-output characteristics implemented by off-the-shelf chips such as the BD429.

### Cable Subsystem

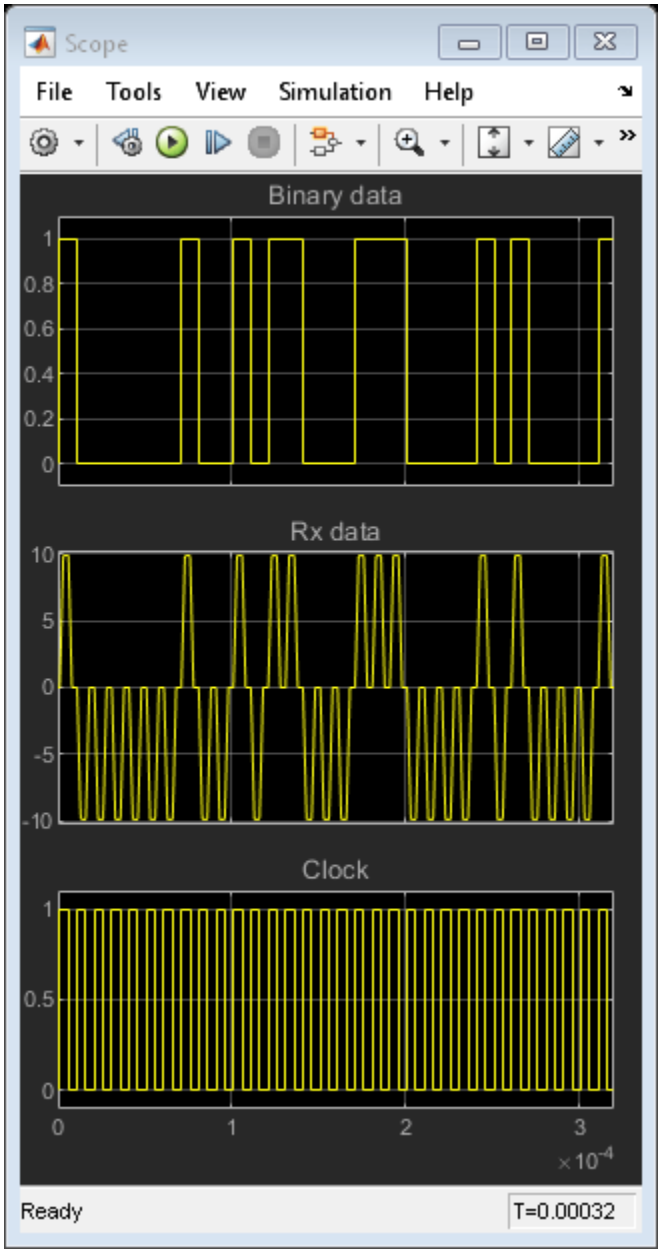


**Receiver Subsystem**




ARINC 429 worst-case receiver parameter values are:
> Differential input resistance R_I = 12K
> Differential input capacitance C_I = 50pF
> Resistance to ground R_H and R_G = 12K
> Capacitance to ground C_H and C_G = 50pF

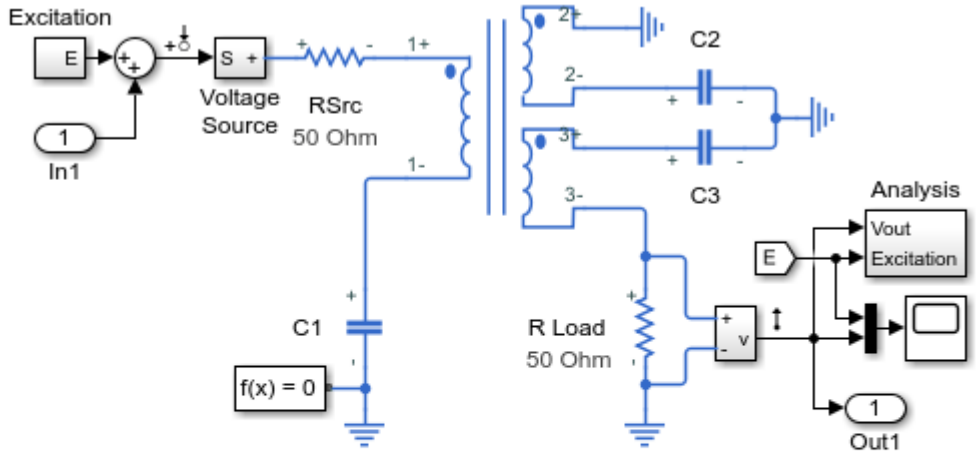
### Simulation Results from Scopes



# Band-Pass Filter Using Three Mutually-Coupled Inductors

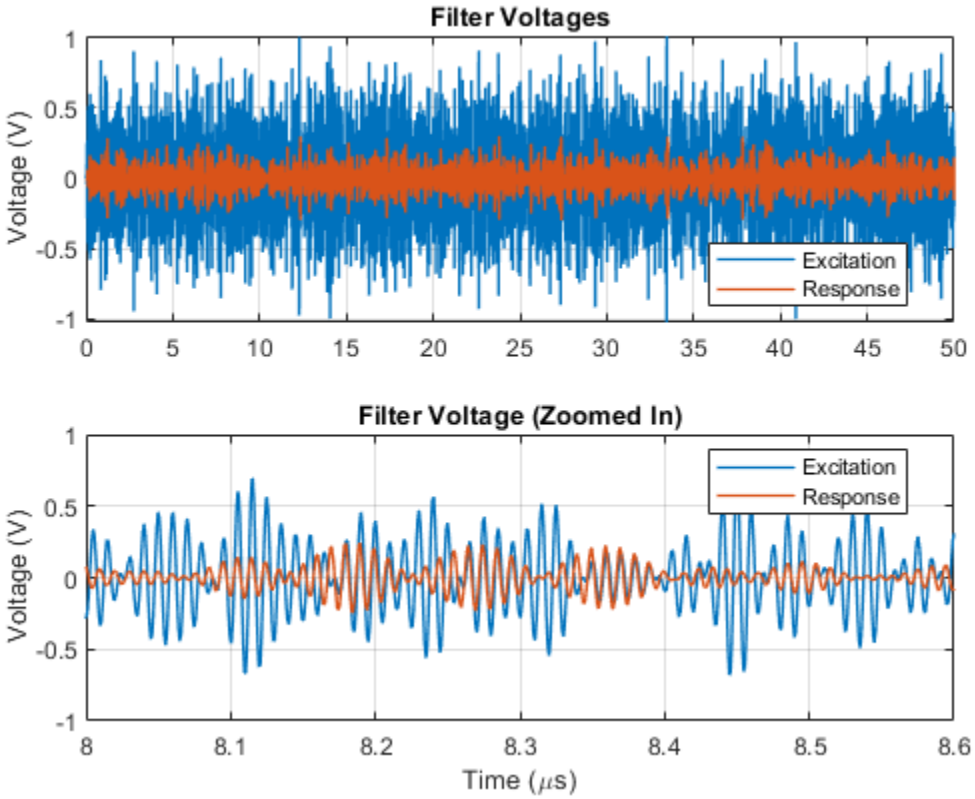
This example shows an implementation of a band-pass filter using three mutually-coupled inductors. The model can be used to validate filter parameters which are chosen to provide a band-pass centered on 100MHz. A band-limited noise source is up-shifted by a 100MHz oscillator and applied to the filter. The response is then down-shifted by the oscillator. The model StopFcn callback takes FFTs of the source and response and estimates the filter frequency response.

### Model



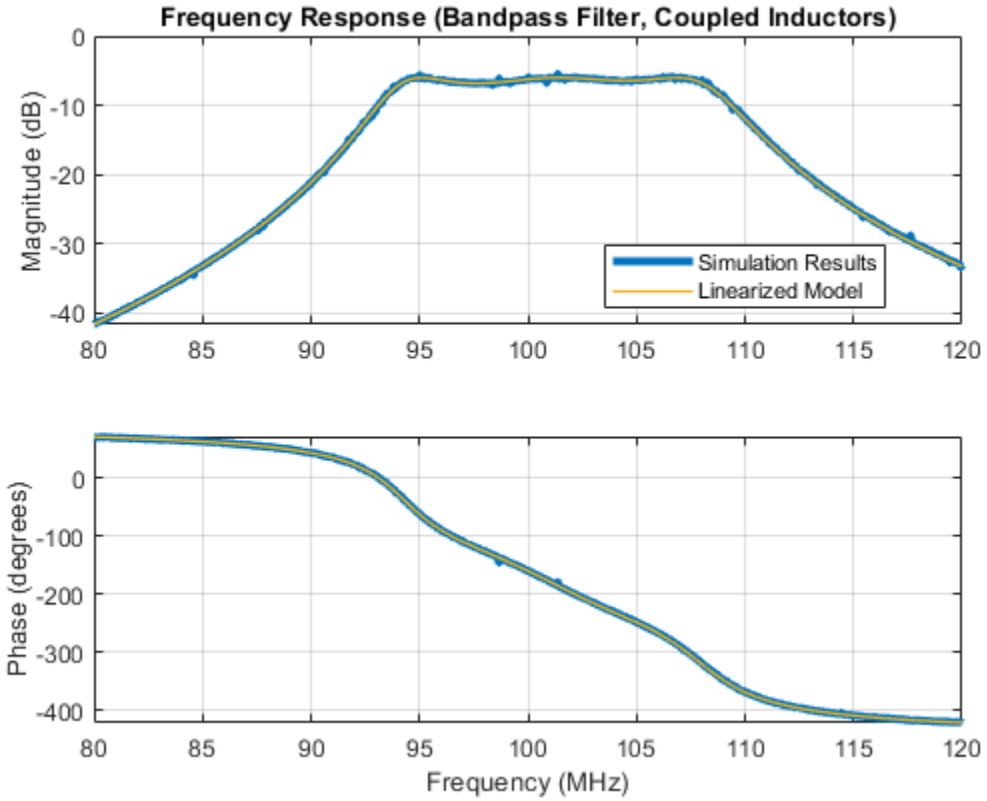

### Simulation Results from Simscape Logging

The plot below shows the excitation voltage and the response of the bandpass filter circuit. A band-limited noise source is up-shifted by a 100MHz oscillator and applied to the filter. The response is then down-shifted by the oscillator.




### Frequency Response

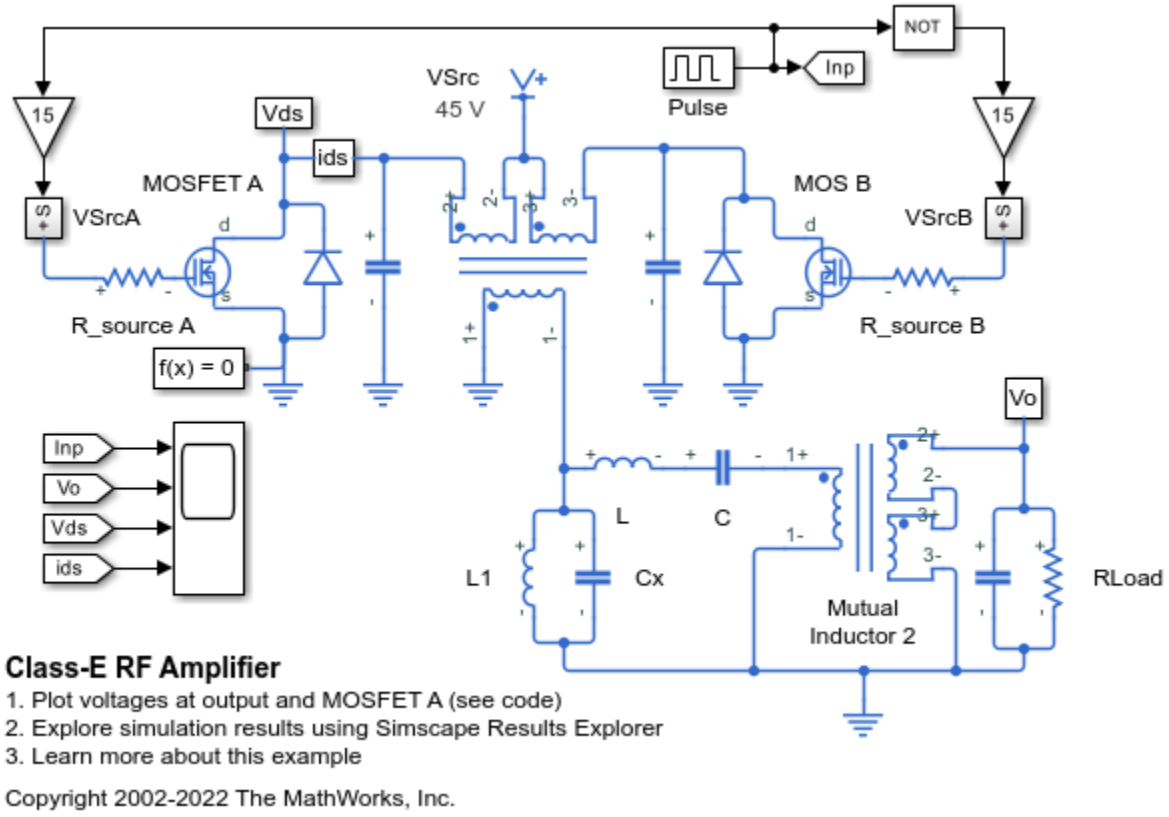
The plot below shows the frequency response of the circuit obtained using two different methods. MATLAB® is used to apply an FFT to the simulation results to estimate the frequency response. The other method uses the MATLAB command linearize to obtain the frequency response. The two responses match nearly perfectly. The frequency response validates the filter design, showing that it is centered at 100 MHz.


Frequency Response (Bandpass Filter, Coupled Inductors)
Magnitude (dB)
0
-10
-20
-30
-40
80
85
90
95
100
105
110
115
120
Simulation Results
Linearized Model
Phase (degrees)
0
-100
-200
-300
-400
Frequency (MHz)

# Class-E RF Amplifier

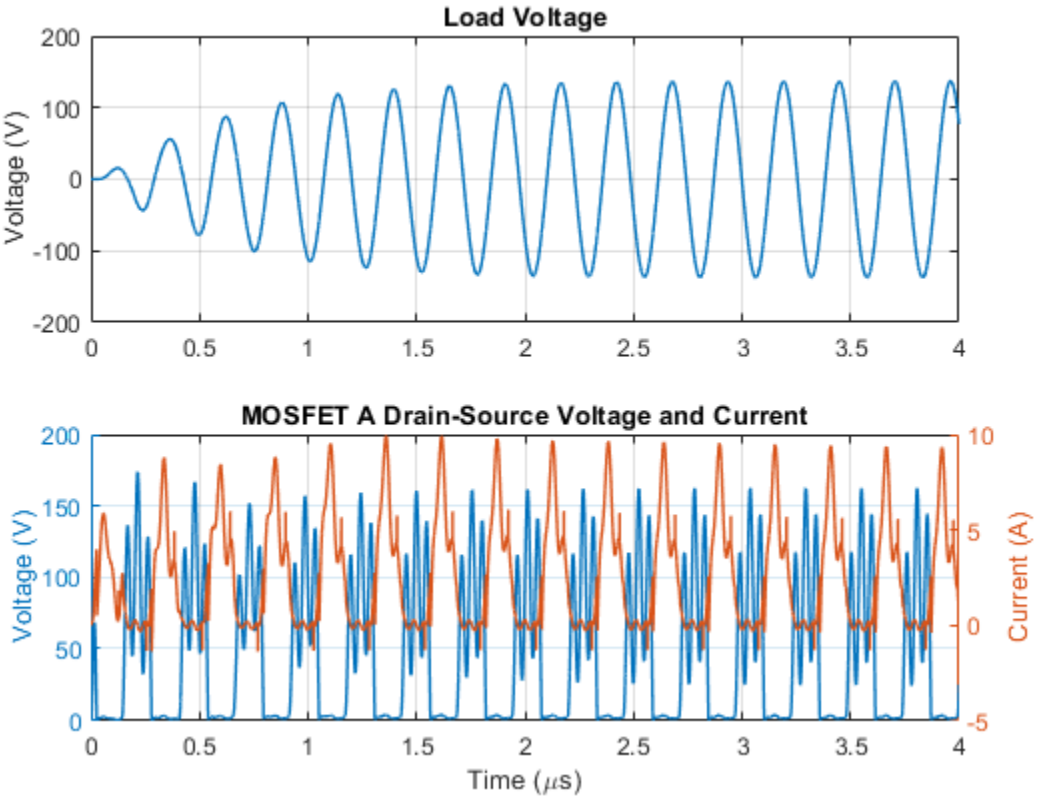
This model shows a class-E RF amplifier with circuit parameters chosen for an 80m wavelength. Class-E amplifiers achieve high efficiency levels as the MOSFETs never have simultaneously high Vds and Ids. The load network is used to shape the voltage and current waveforms. This model can be used to verify correct operation and to support component selection. Correct operation of the circuit is particularly sensitive to source resistance, R_source. The capacitance parameters for the two MOSFETs are representative for an FQA11N90 device.

### Model




### Simulation Results from Simscape Logging

The plot below shows the load voltage and the drain-source voltage and current for MOSFET A.

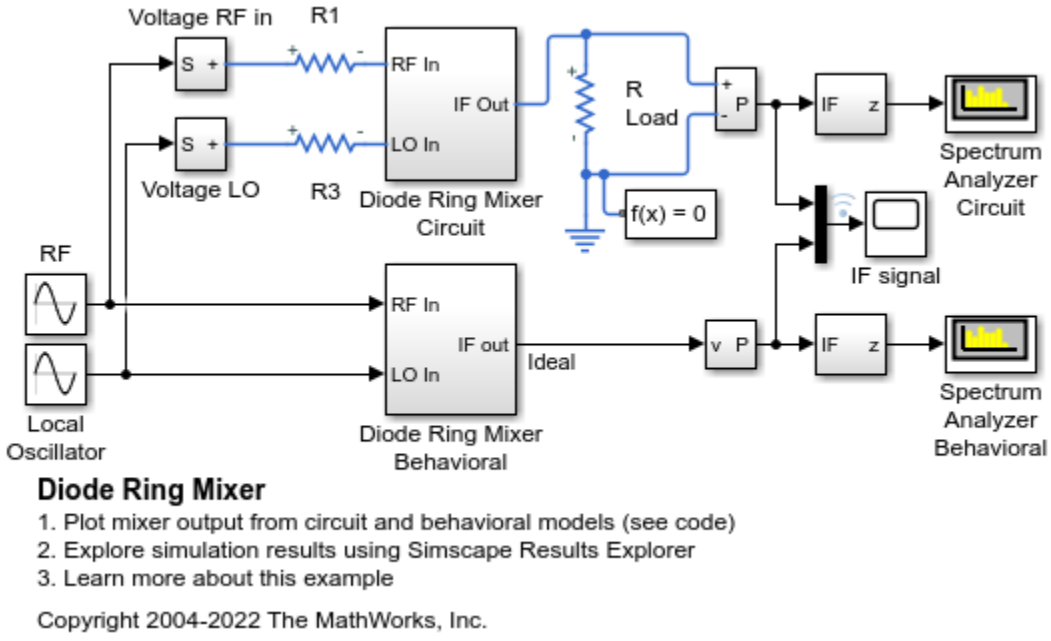

Load Voltage
Voltage (V)
200
100
0
-100
-200
0
0.5
1
1.5
2
2.5
3
3.5
4
MOSFET A Drain-Source Voltage and Current
Voltage (V)
Current (A)
200
150
100
50
0
10
5
0
-5
Time (μs)

# Diode Ring Mixer

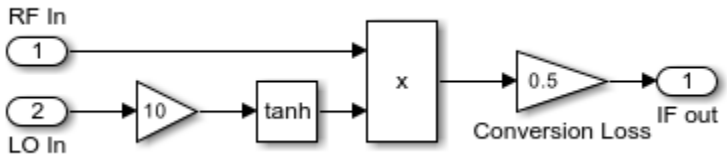
This model shows how a diode ring can be used to demodulate a frequency-modulated signal. The RF input has a fixed frequency of 9MHz, and the local oscillator has a fixed frequency of 11MHz. Hence the frequency-modulated signal is a sine wave of 2MHz. This 2MHz component is clearly visible in the IF response. The second component in the IF response is the sum of the RF and local oscillator frequencies i.e. 20MHz.

The spectrum shows that the diode ring mixer introduces some additional unwanted harmonics. The spectrum can be used to determine how to augment the behavioral ring mixer model with additional terms to produce an abstracted mixer model that can be used in a more complex circuit

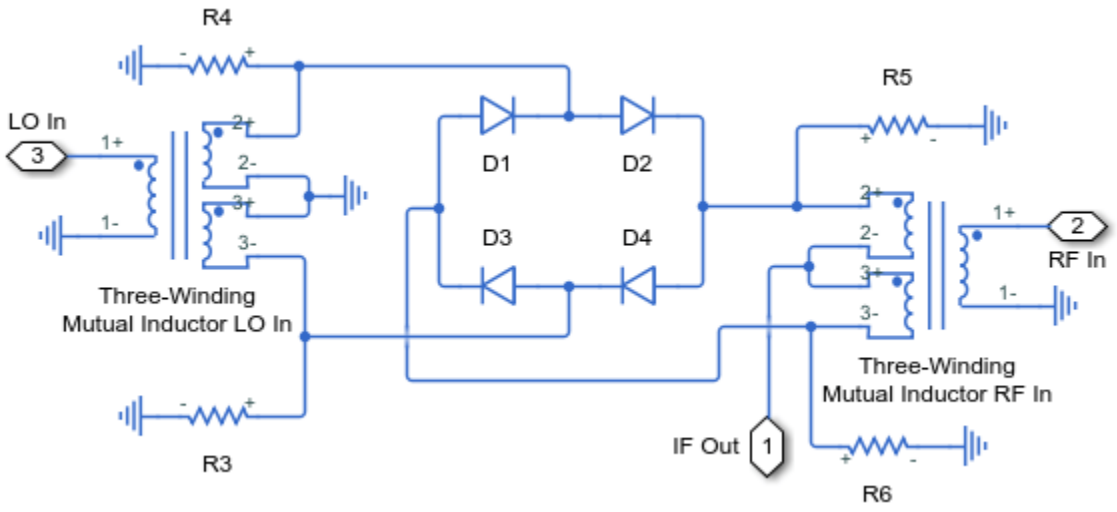
### Model




### Diode Ring Mixer Behavioral Subsystem

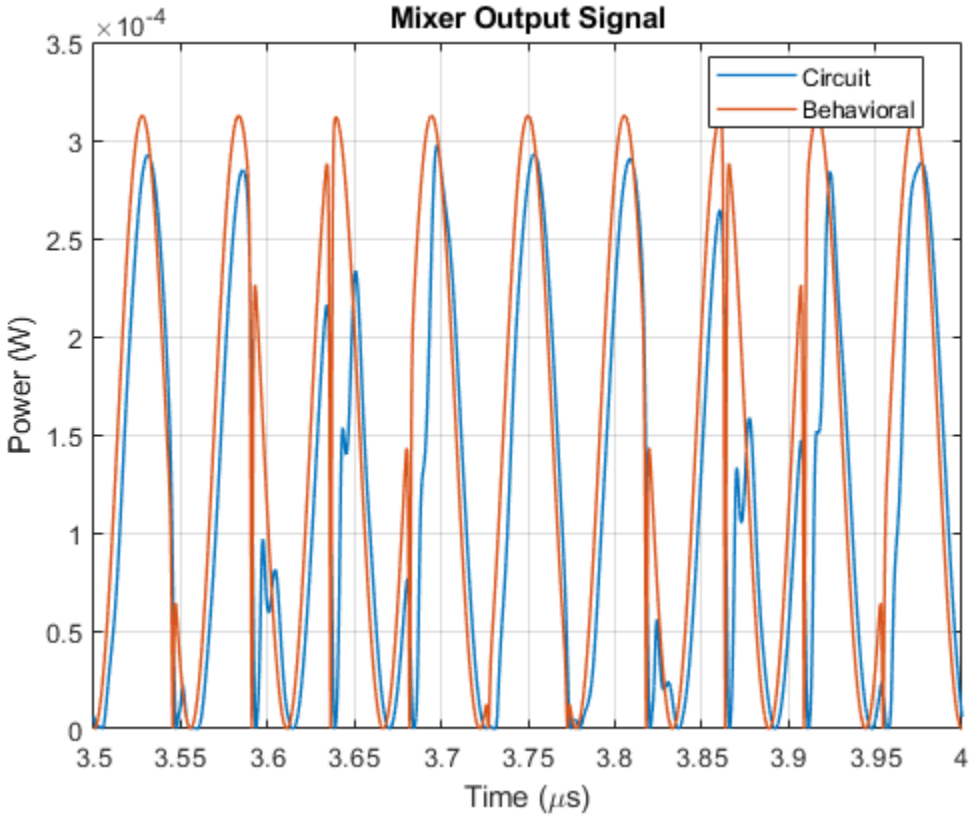


### Diode Ring Mixer Circuit Subsystem



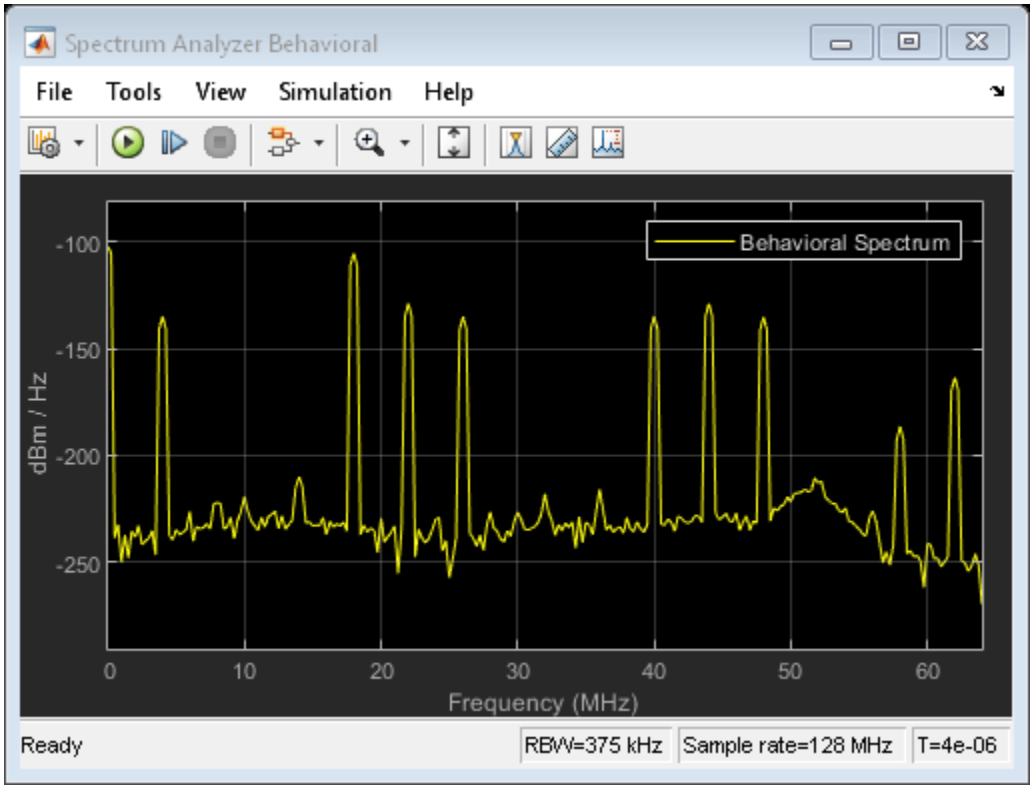

### Simulation Results from Simscape Logging

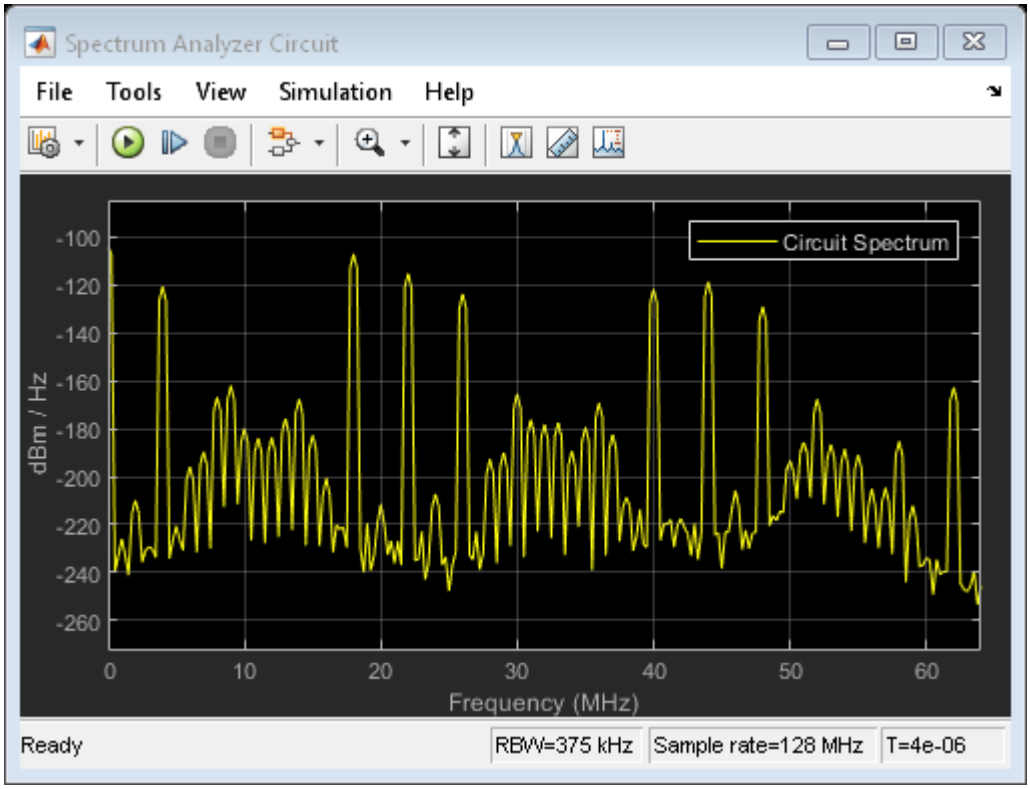
The plot below shows the output of the circuit and behavioral mixer models.



### Simulation Results from Spectrum Analyzer

The frequency spectrum plots below show that the behavioral and circuit models produce similar results. The spectrum from the circuit model shows additional unwanted harmonics which are not shown in the ideal behavioral model.

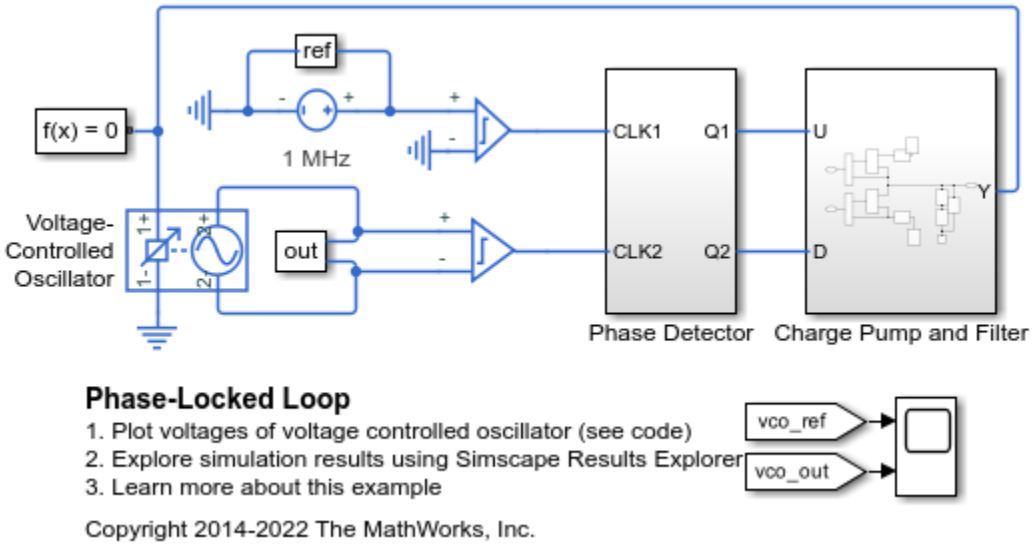



Spectrum Analyzer Circuit
File
Tools
View
Simulation
Help
Circuit Spectrum
-100
-120
-140
-160
-180
-200
-220
-240
-260
dBm / Hz
0
10
20
30
40
50
60
Frequency (MHz)
Ready
RBW=375 kHz
Sample rate=128 MHz
T=4e-06

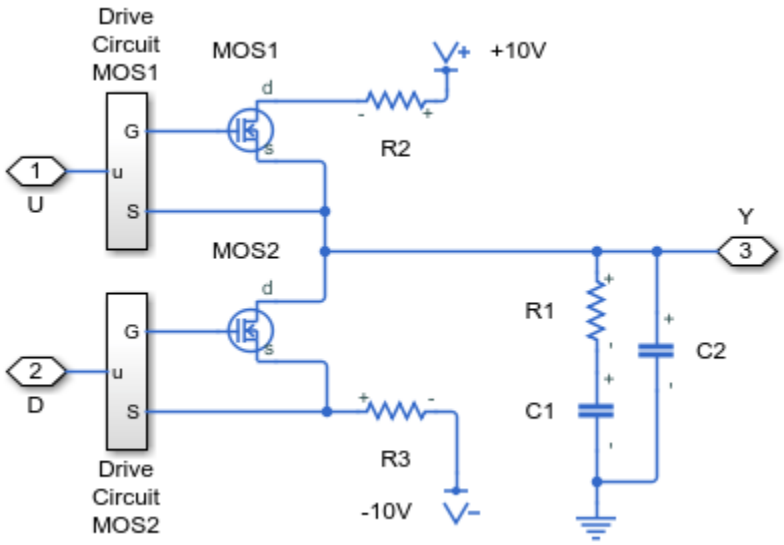
# Phase-Locked Loop

This model shows how to model a phase-locked loop. The charge pump and filter are modeled using discrete analog components whereas the oscillator is represented as behavioral component using the Simscape™ Electrical™ Voltage-Controlled Oscillator block. The D-type flip-flops in the phase detector are represented in a simplified form using Simulink® blocks to define the behavior, and electrical components are used just at the interface. Non-zero initial conditions are applied to C1 and C2 in order to start the VCO out of phase and test the tracking ability.

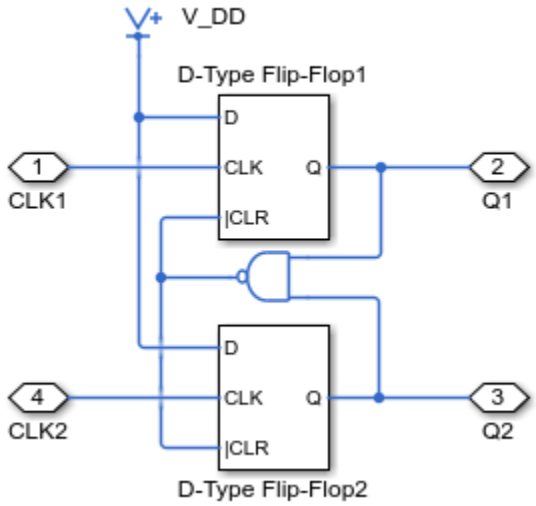
### Model




### Charge Pump and Filter Subsystem

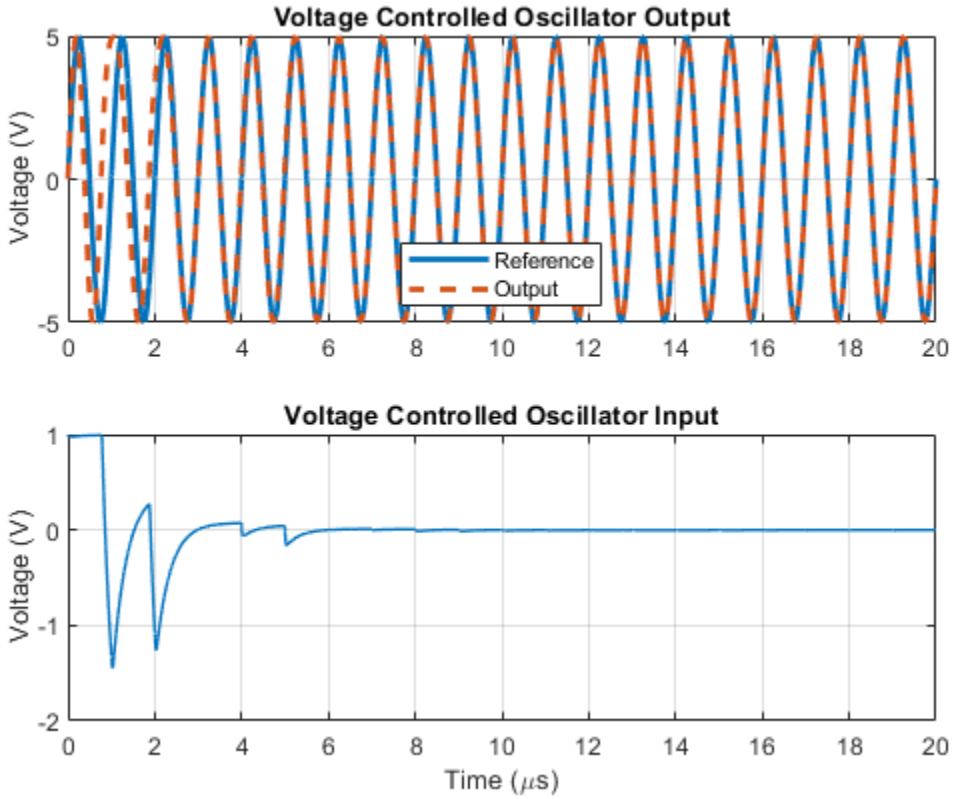


### Phase Detector Subsystem




### Simulation Results from Simscape Logging

The plot below shows the reference and output voltage for the voltage controlled oscillator within a phase locked loop.

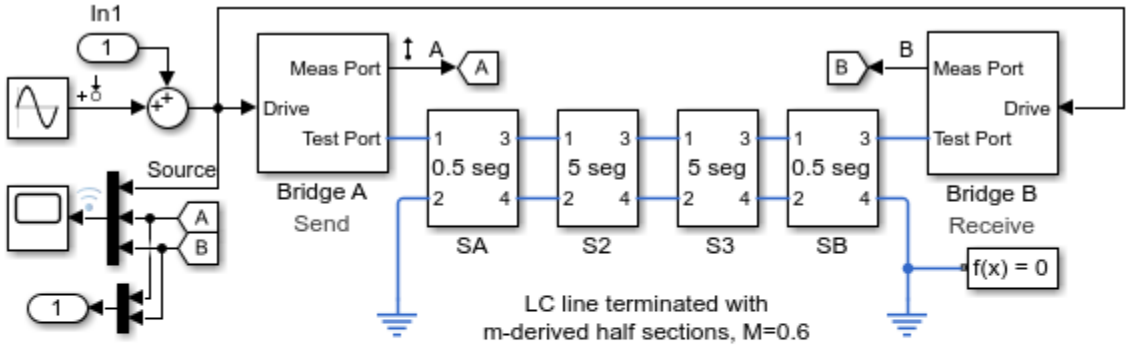

Voltage Controlled Oscillator Output
Voltage (V)
5
0
-5
0
2
4
6
8
10
12
14
16
18
20
Reference
Output
Voltage Controlled Oscillator Input
Voltage (V)
1
0
-1
-2
0
2
4
6
8
10
12
14
16
18
20
Time (μs)

# LC Transmission Line and Test Bridge

This model shows a transmission line model and bi-directional test bridge. Reflected and transmitted signals are slightly different if the test direction is changed. This is because the line model is not symmetric. This type of model can be used both to explore the impact of cable choice on transmission characteristics, and also to compare relative fidelity of different transmission line model structures and parameterizations.

This model can be used to obtain the frequency response of the system. MATLAB® command linmod can be used to linearize the model. If you have Simulink® Control Design™, open the model ee_transmission_line_lc. To open the Model Linearizer, on the Apps tab, under Control Systems, click Model Linearizer.

**Model**




**LC Transmission Line and Test Bridge**

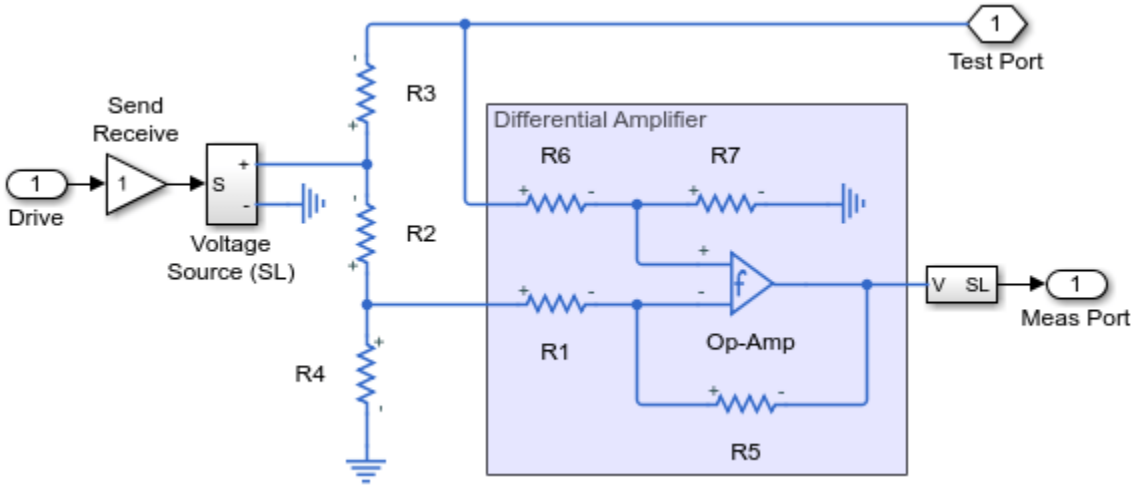
1. Configure transmission: A to B, B to A
2. Plot signals for A to B and B to A (see code)
3. Linearize circuit to view frequency response (see code)
4. Explore simulation results using Simscape Results Explorer
5. Learn more about this example

Transmission Line Specs

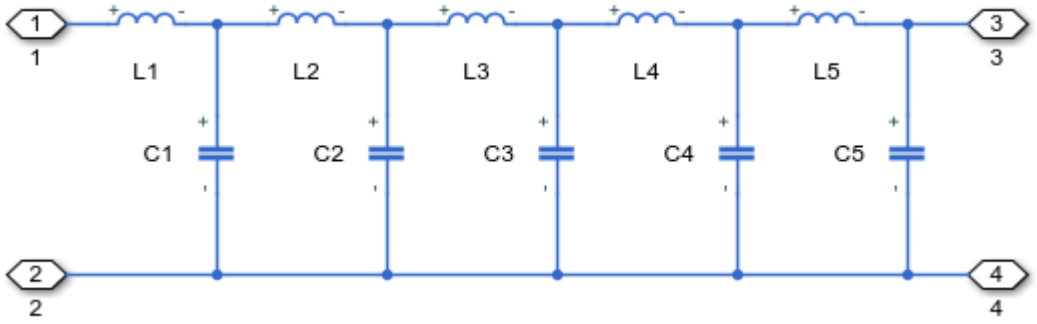
| | |
|---|---|
| Impedance Zo | = 50 |
| Trans. delay (ns) | = 3.1831 |
| Cutoff freq (GHz) | = 1 |

Copyright 2004-2022 The MathWorks, Inc.

### Bridge A Subsystem

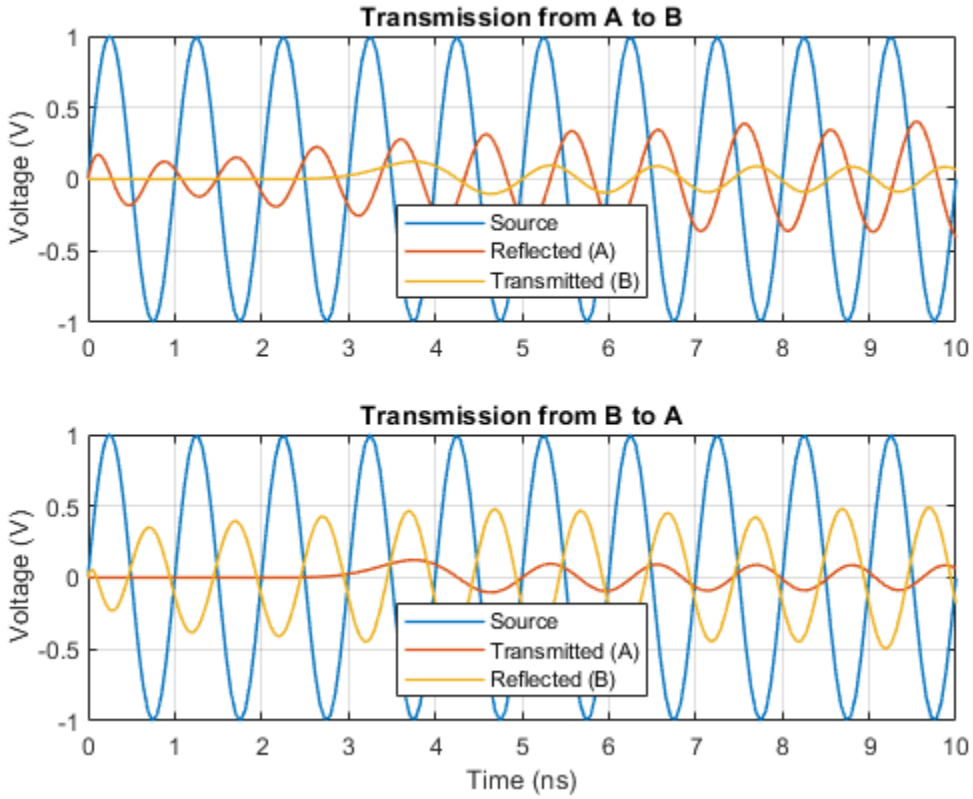



### 5 Segments (S2 Subsystem)

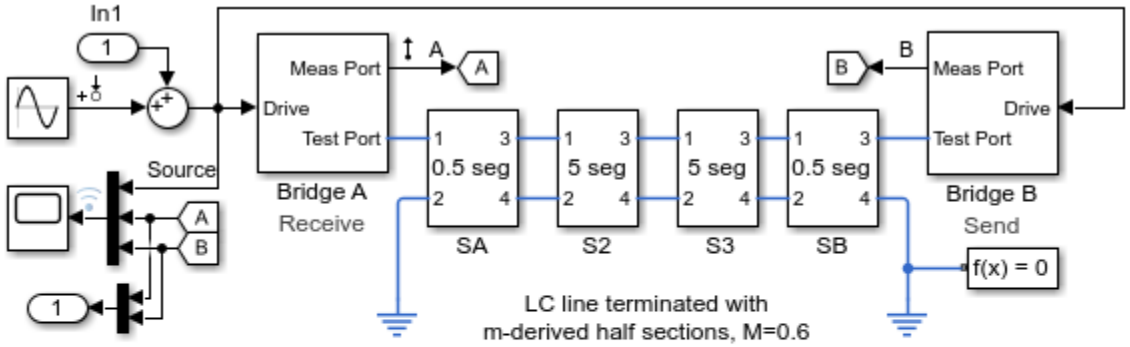



### Simulation Results from Simscape Logging

The plots below show the source, transmitted, and reflected signals for transmission tests from A to B and from B to A. Because the transmission line is asymmetric, differences can be seen in the reflected signals.




### Frequency Response



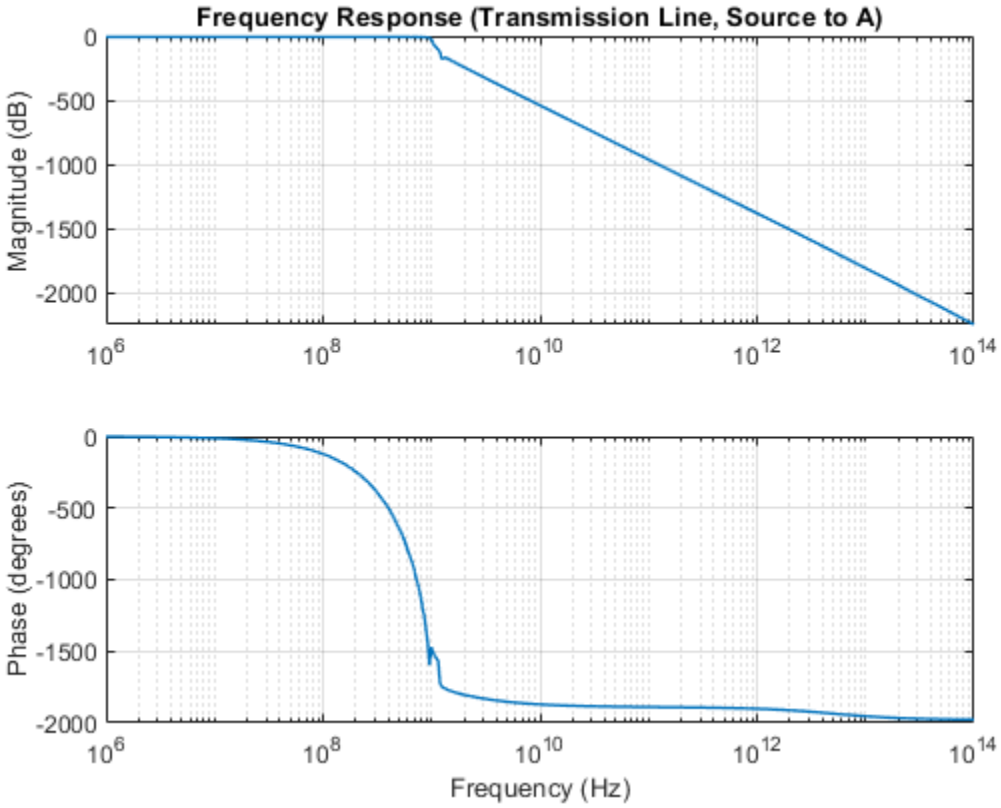

## LC Transmission Line and Test Bridge

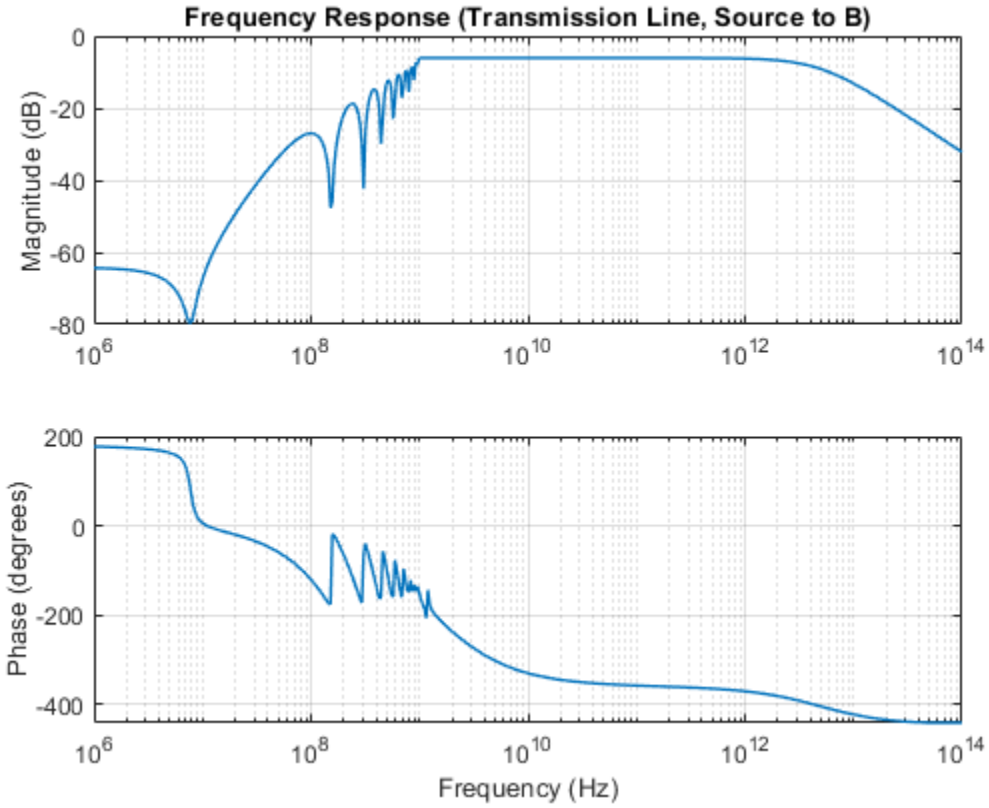
1. Configure transmission: A to B, B to A
2. Plot signals for A to B and B to A (see code)
3. Linearize circuit to view frequency response (see code)
4. Explore simulation results using Simscape Results Explorer
5. Learn more about this example

Transmission Line Specs

| | |
|---|---|
| Impedance Zo | = 50 |
| Trans. delay (ns) | = 3.1831 |
| Cutoff freq (GHz) | = 1 |

Copyright 2004-2022 The MathWorks, Inc.

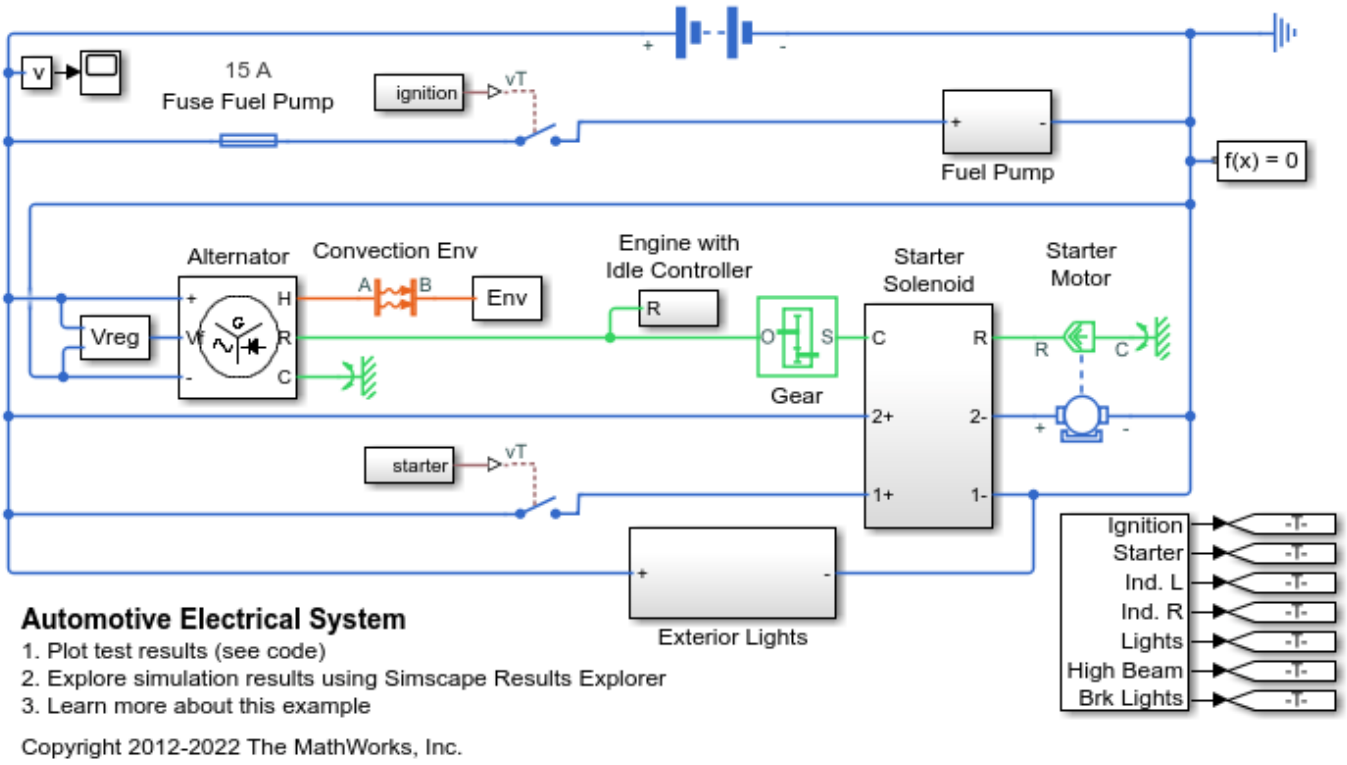

Frequency Response (Transmission Line, Source to A)
Magnitude (dB)
0
-500
-1000
-1500
-2000
$10^6$
$10^8$
$10^{10}$
$10^{12}$
$10^{14}$
Phase (degrees)
0
-500
-1000
-1500
-2000
$10^6$
$10^8$
$10^{10}$
$10^{12}$
$10^{14}$
Frequency (Hz)


Frequency Response (Transmission Line, Source to B)
Magnitude (dB)
0
-20
-40
-60
-80
Phase (degrees)
200
0
-200
-400
$10^6$
$10^8$
$10^{10}$
$10^{12}$
$10^{14}$
Frequency (Hz)

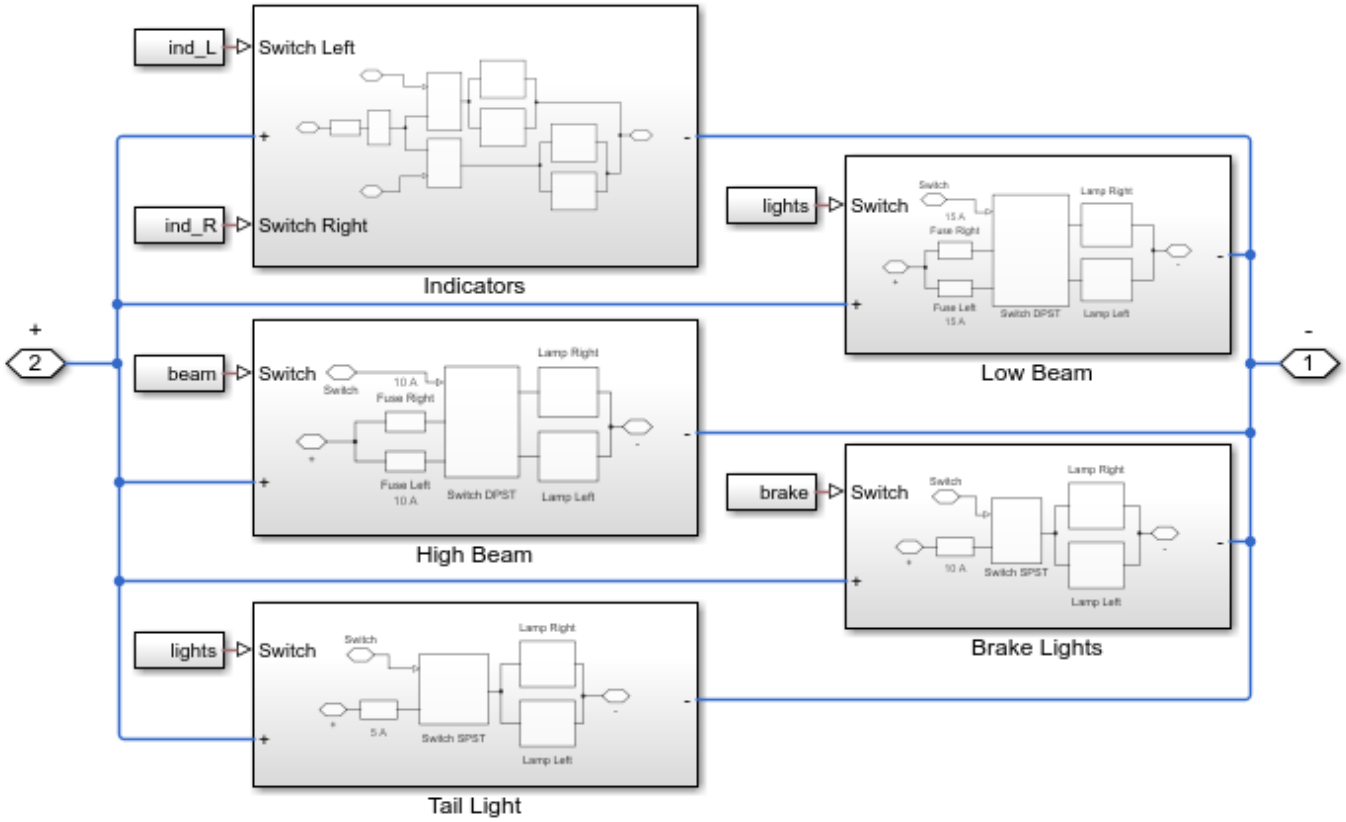
# Automotive Electrical System

This example shows a simplified dynamic model of an automotive electrical system. The model contains electrical, mechanical, and thermal systems, and is able to simulate the effect of engine starting on the electrical network.

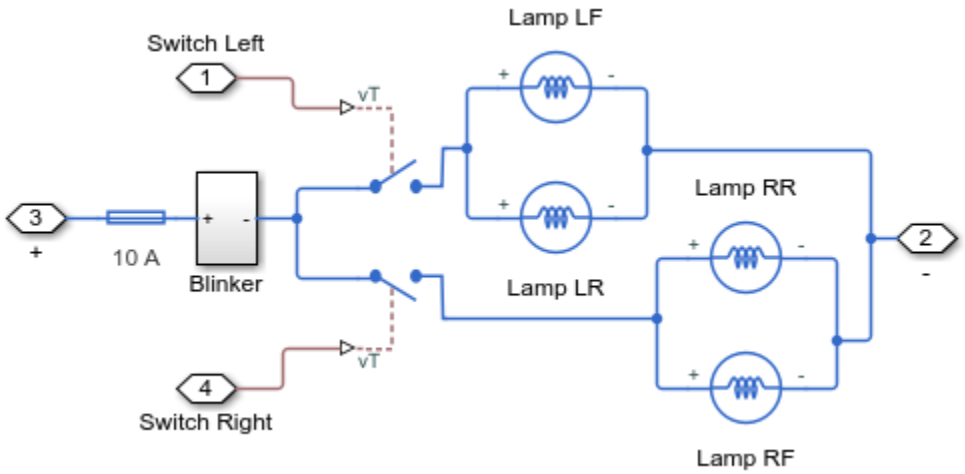
**Model**



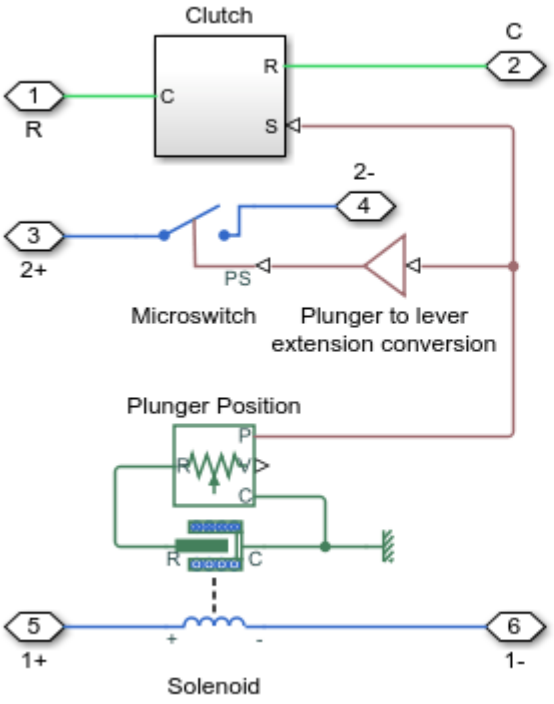
### Exterior Lights Subsystem




### Indicators Subsystem



### Starter Solenoid Subsystem

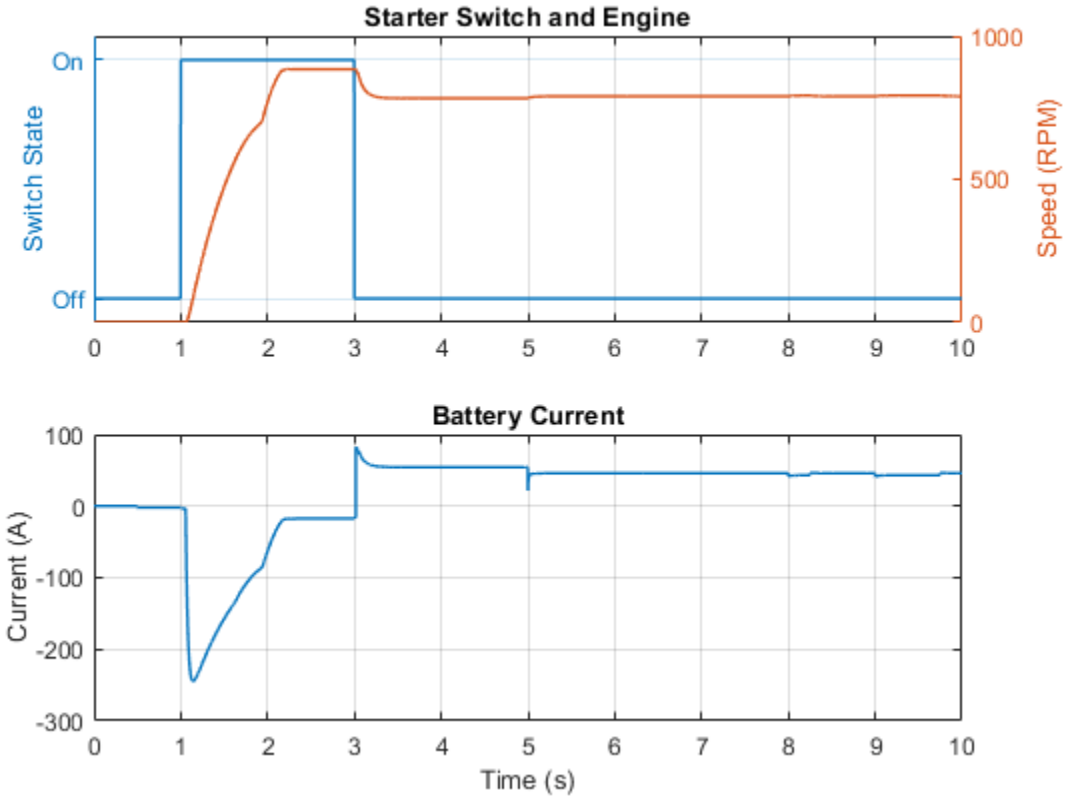



The clutch could be more accurately modeled using the Dog Clutch in Simscape Driveline(TM). Additionally, the Simscape Driveline Unidirectional Clutch could be used to ensure that once the engine catches it does not backdrive the starter motor.

Solenoid plunger has travel of 10mm. When plunger position is 10mm, the clutch is fully disengaged, and when the plunger is anywhere between 5mm and 0mm, the clutch is fully engaged. Switch closes and enables the starter motor only when the plunger is between 0 and 2mm.

### Simulation Results from Simscape Logging

The plot below shows a startup sequence for an automotive electrical network. The starter motor is turned on, which draws current from the battery in order to start the engine. As lights and other electrical loads are turned on and off, the current draw from the battery varies.




### Results from Real-Time Simulation

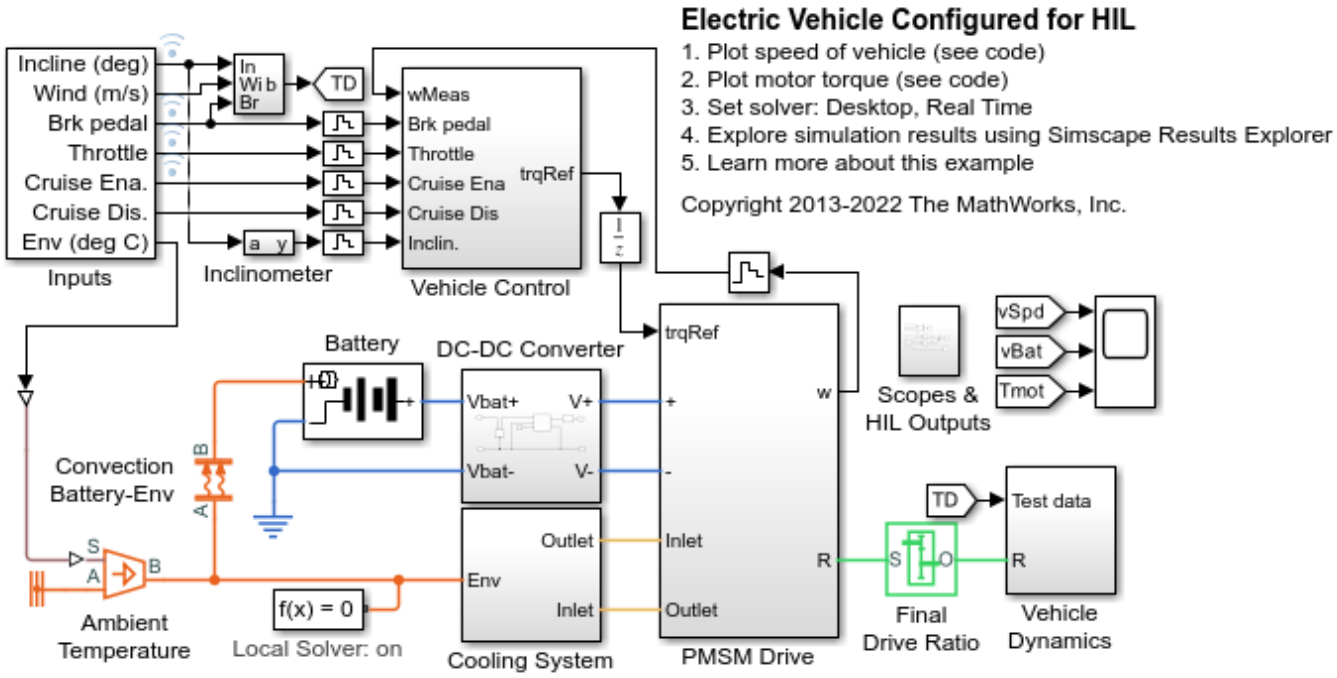
This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 500 microseconds.

# Electric Vehicle Configured for HIL

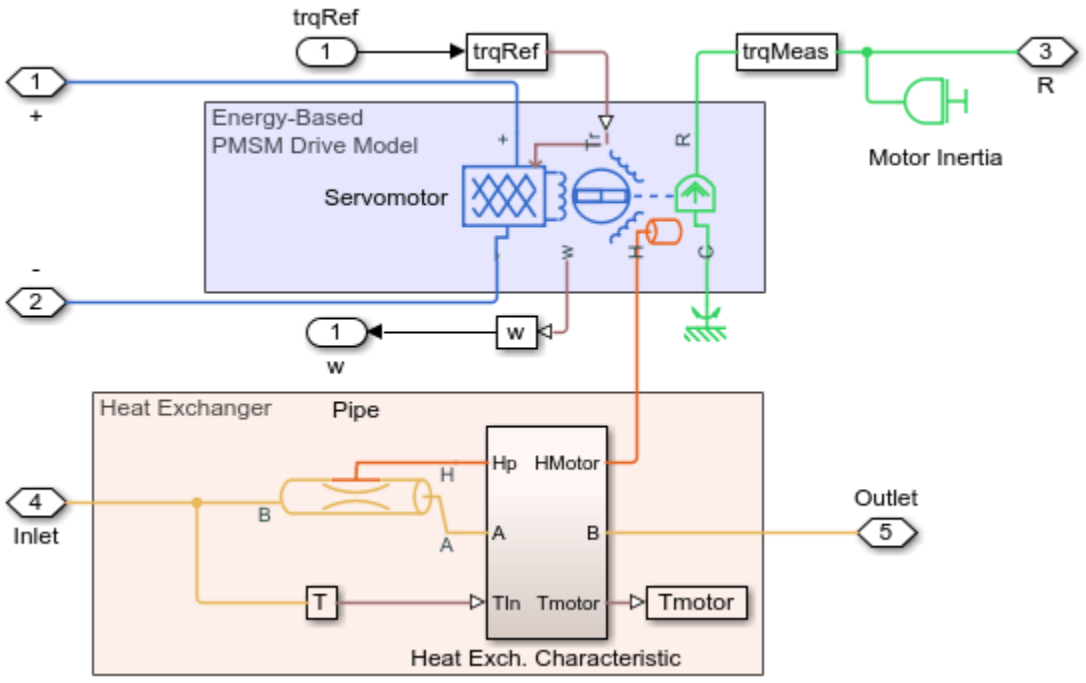
This example shows an electric vehicle model suitable for Hardware-In-the-Loop (HIL) deployment. Energy-based modeling is used to avoid high-frequency switching, and solvers set for fixed-step simulation.

The test run shows the vehicle accelerating to a steady speed up an incline followed by a period of descent during which electrical power is returned to the battery.

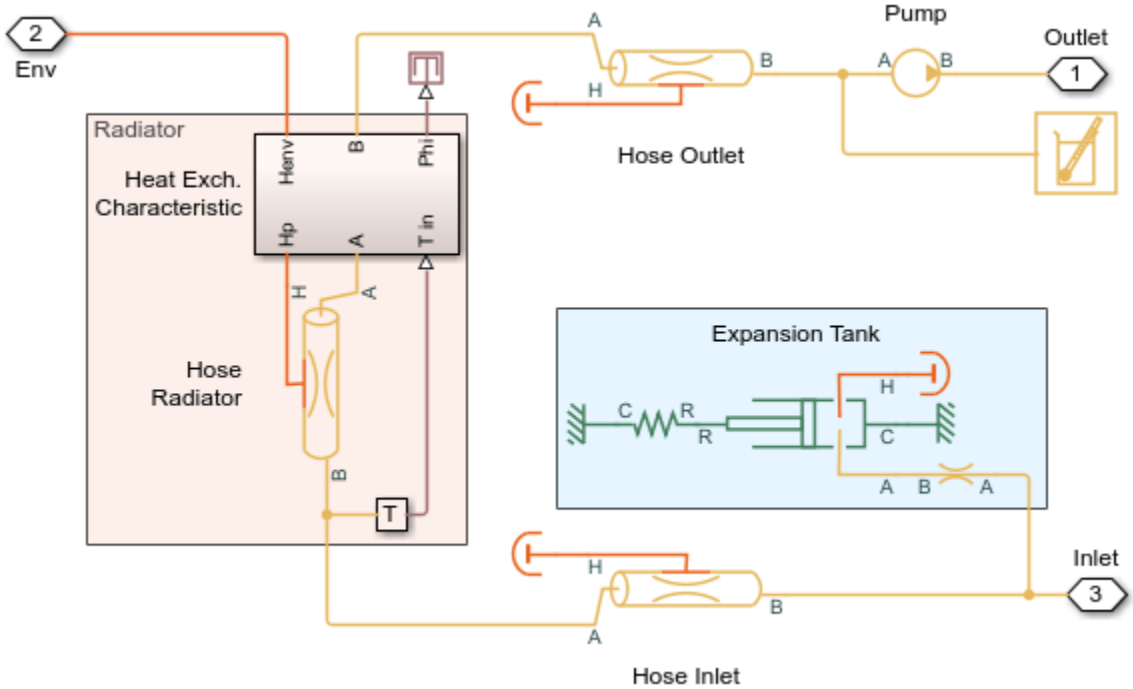
**Model**



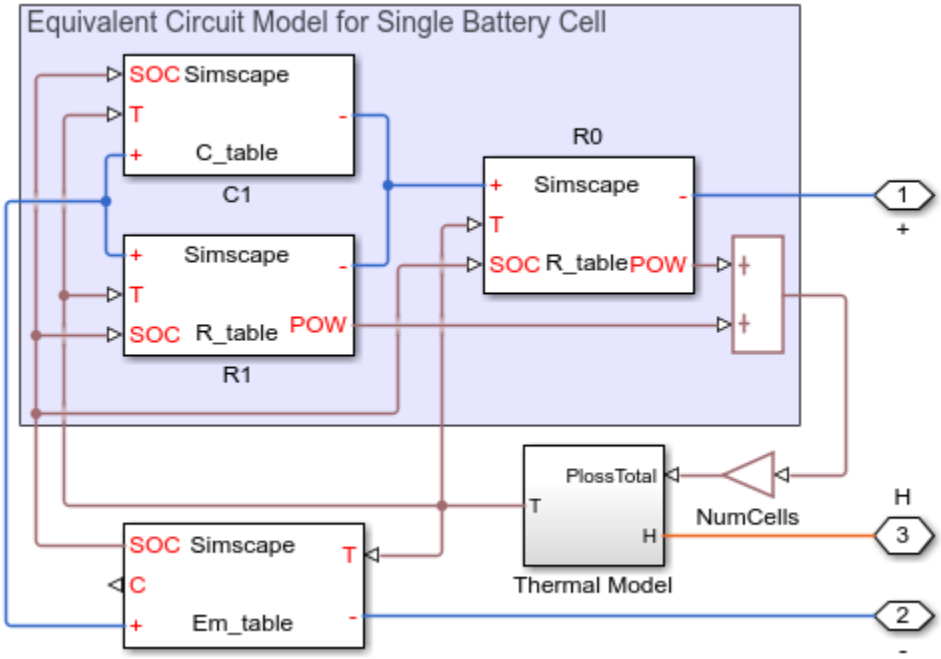
### PMSM Drive Subsystem



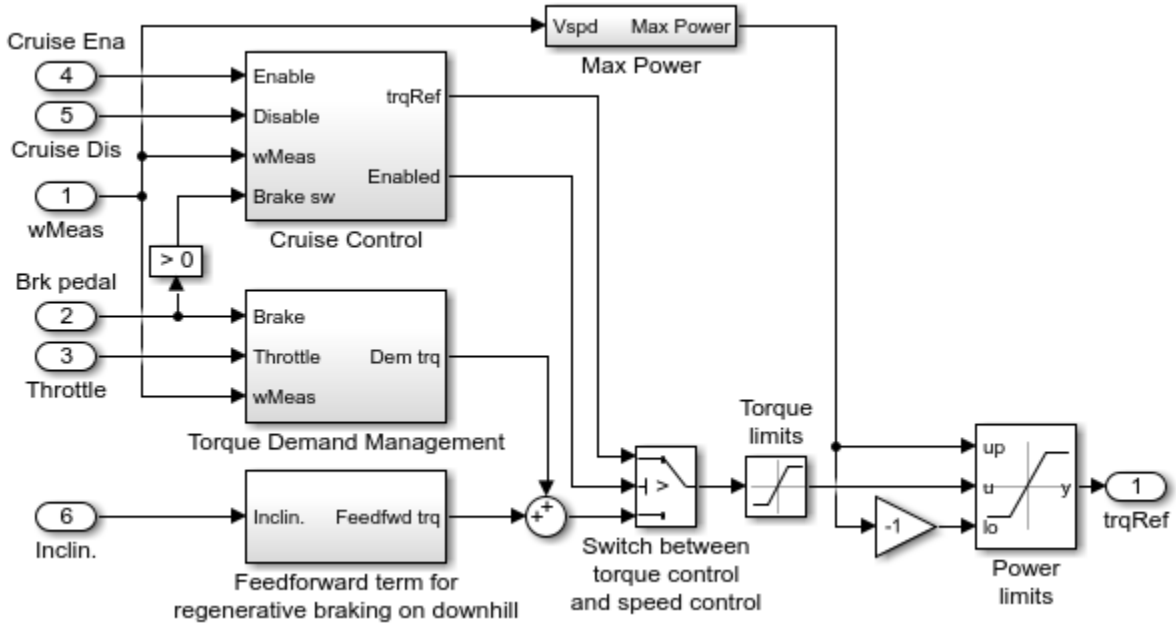

### Cooling System Subsystem



### Electric and Thermal Model Subsystem



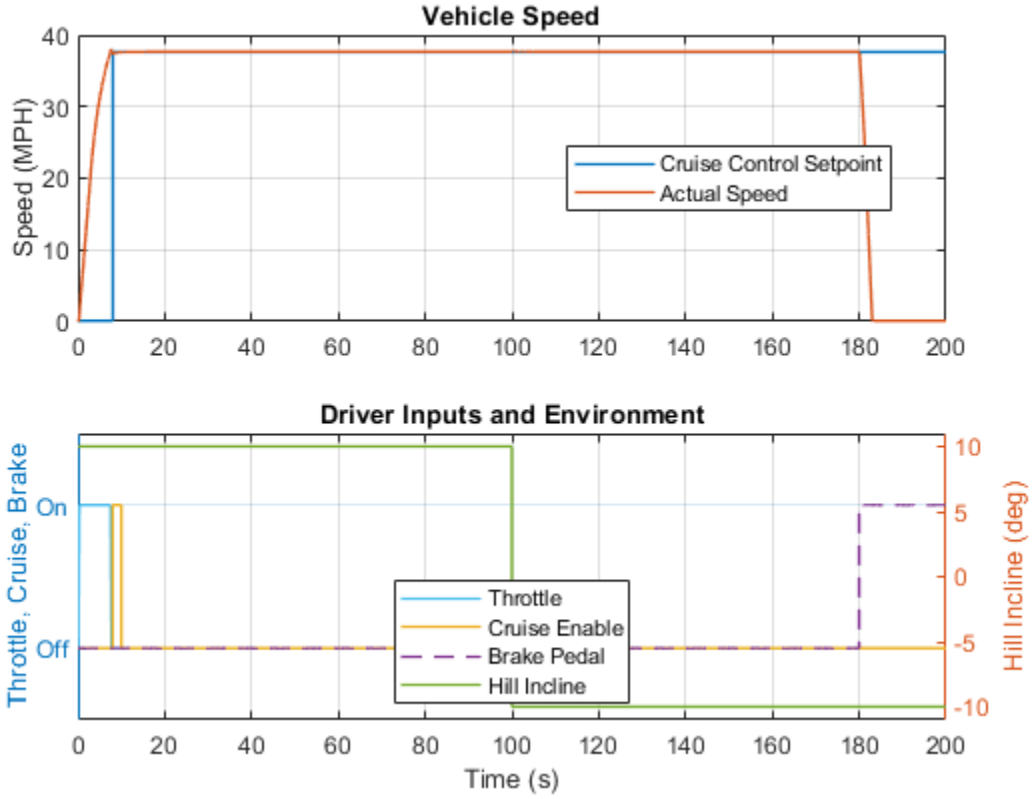

### Vehicle Control Subsystem



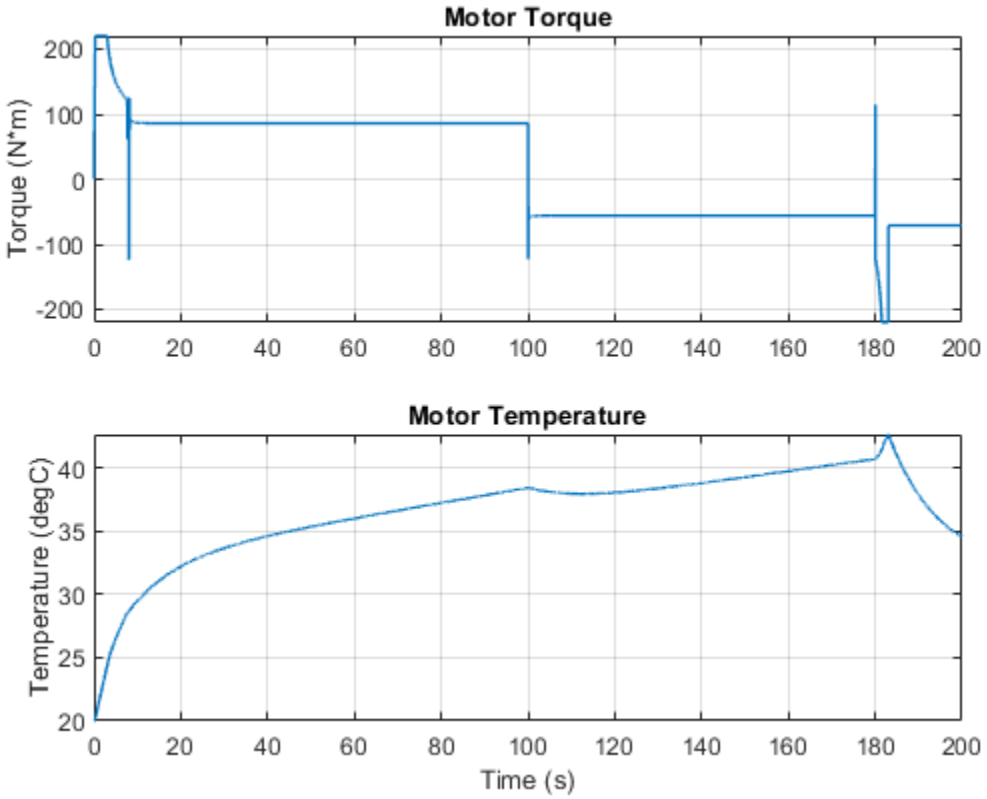

### Simulation Results from Simscape Logging

The plot below shows the behavior of an electric vehicle subject to driver inputs and environmental conditions. The vehicle accelerates until the driver enables cruise control. The vehicle maintains

speed even as the grade of the road changes. As the driver applies the brakes, the vehicle slows down to zero speed.




The plots below shows the torque produced by the PMSM motor in the electric vehicle and its temperature. During the first half of the simulation the motor is accelerating the vehicle to the commanded speed and then continuing to apply torque to push the vehicle up a hill. During the second half of the simulation, the motor acts as a generator as shown by the change in sign of motor torque.




### Results from Real-Time Simulation

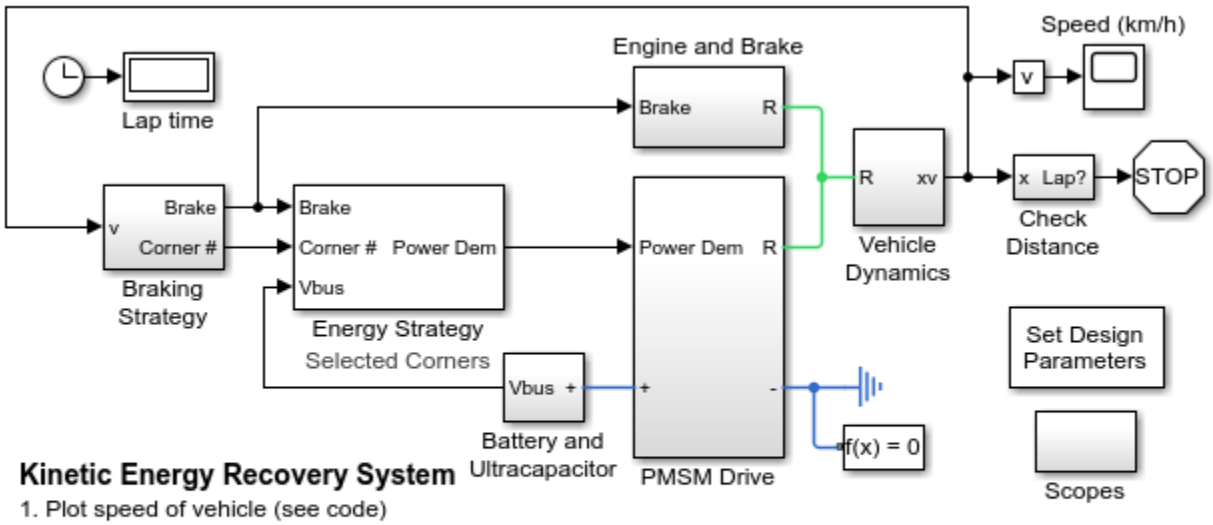
This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 400 microseconds.

# Kinetic Energy Recovery System

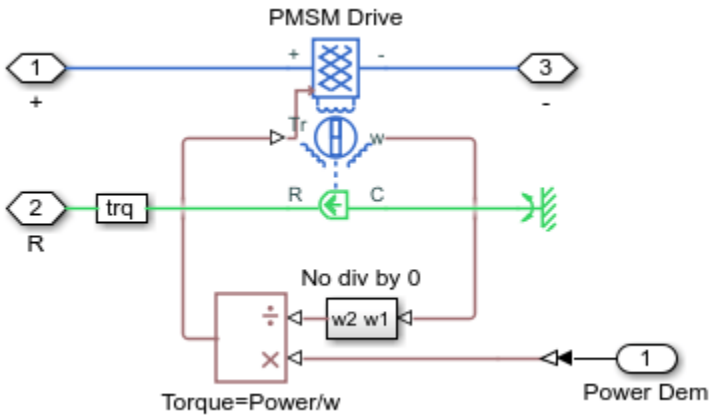
This example shows operation of a Kinetic Energy Recovery System (KERS) on a Formula 1 car. The model permits the benefits to be explored. During braking, energy is stored in a lithium-ion battery and ultracapacitor combination. It is assumed that a maximum of 400KJ of energy is to be delivered in one lap at a maximum power of 60KW. Design parameters are the weight of the battery, ultracapacitor and motor-generator. If these parameters are all set to the very small value of 0.01kg, the lap time is 95.0 seconds, this corresponding to a car with no KERS. With default values set here, approximately 1/4 of a second is saved on the lap time when using any available electrical power when not braking. Applying KERS only to selected corners requires a larger ultracapacitor to show any significant benefit.

This model shows how Simscape™ Electrical™ and Simscape can be used to support system-level design. The KERS performance is a complex trade-off between the masses of the three main components (battery, ultracapacitor and motor-generator), plus the energy-management strategy. The KERS system adds mass which reduces acceleration due to the engine. The stored electrical energy from braking must more than compensate for this. Lithium-ion batteries have a very high energy per unit mass but a poor power per unit mass. Conversely an ultracapacitor has relatively low energy per unit mass, but a very high power per unit mass which suits this particular application.

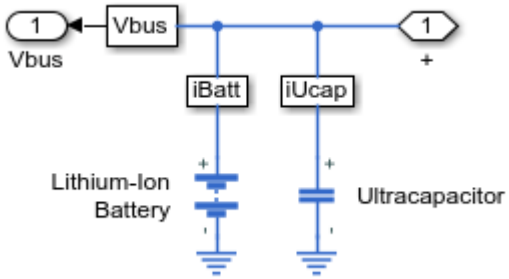
**Model**



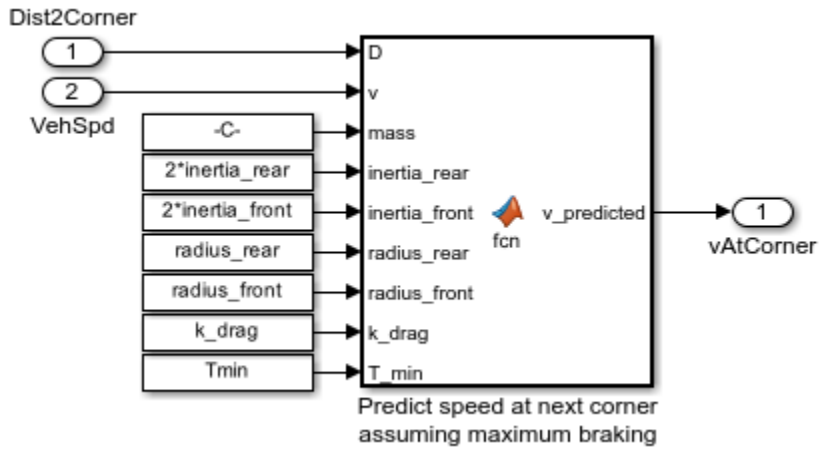
### PMSM Drive Subsystem




### Battery and Ultracapacitor Subsystem

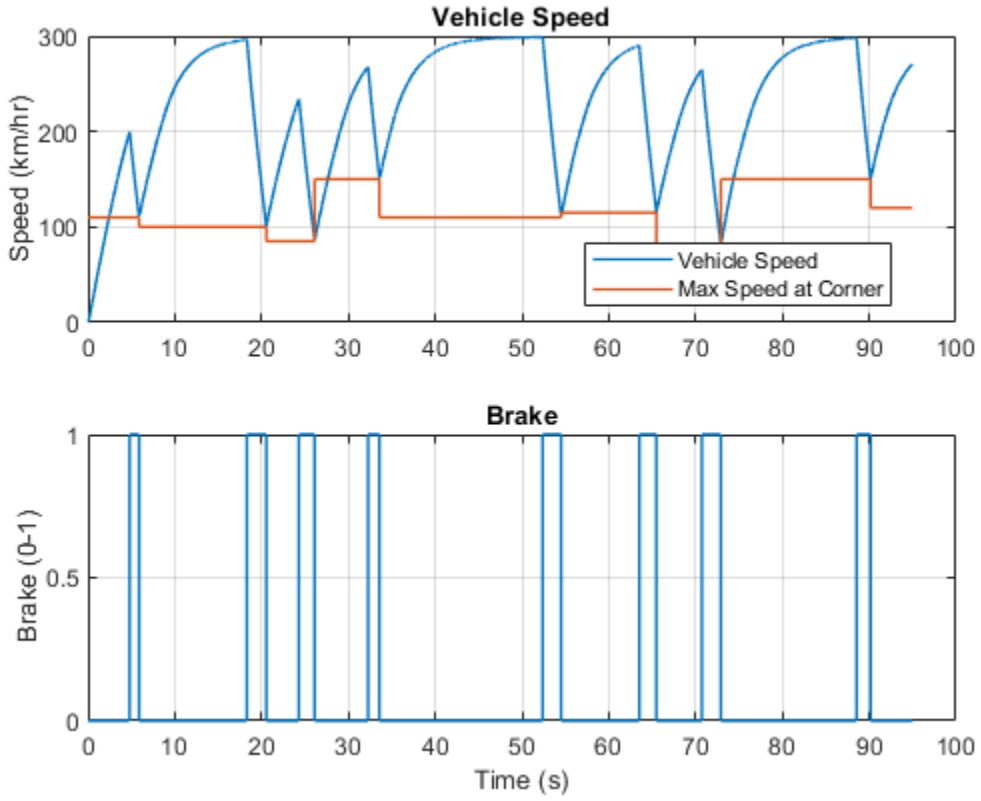



### Predict Speed at Corner If Maximum Braking Used Subsystem

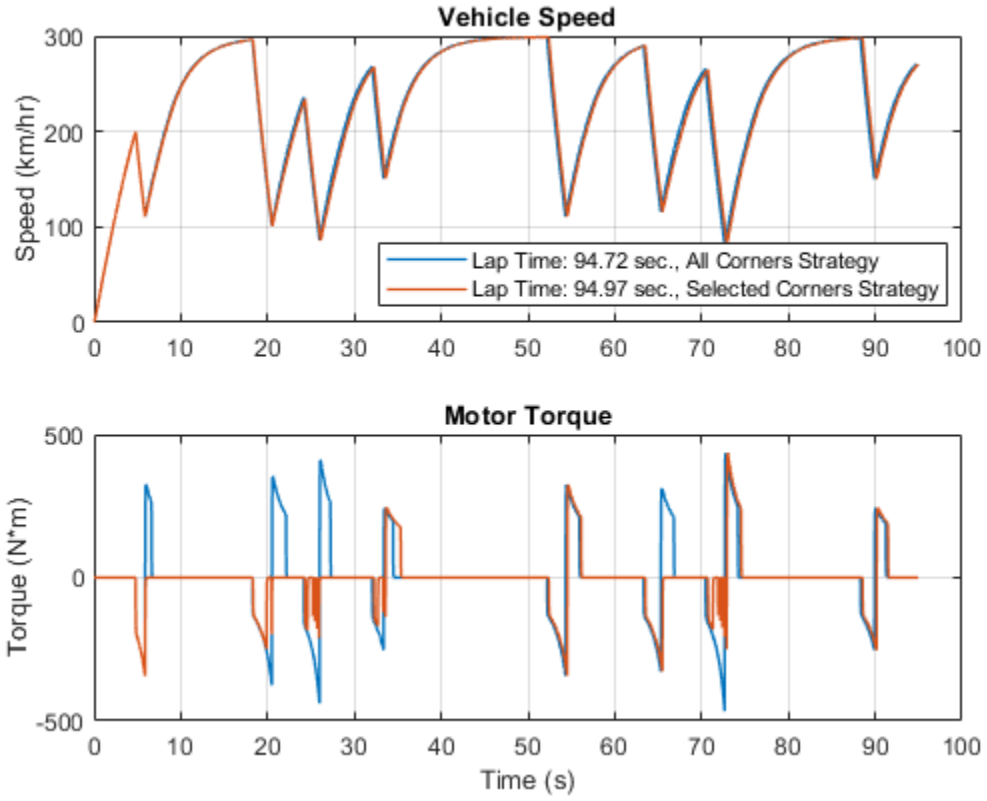


### Simulation Results from Simscape Logging

The plot below shows the vehicle speed during a single lap. The driver knows the maximum speed the vehicle should be travelling at the corners on the track and applies the brakes to achieve that speed in the corner.




The plot below compares the two strategies for using the electrical drive during acceleration. One strategy uses the electrical powertrain during all corners, the other uses it only on selected corners. The difference is easiest to see on the plot of motor torque, where the selected corners strategy shows zero torque from the motor during a number of the corners in the lap (rapid deceleration and acceleration).

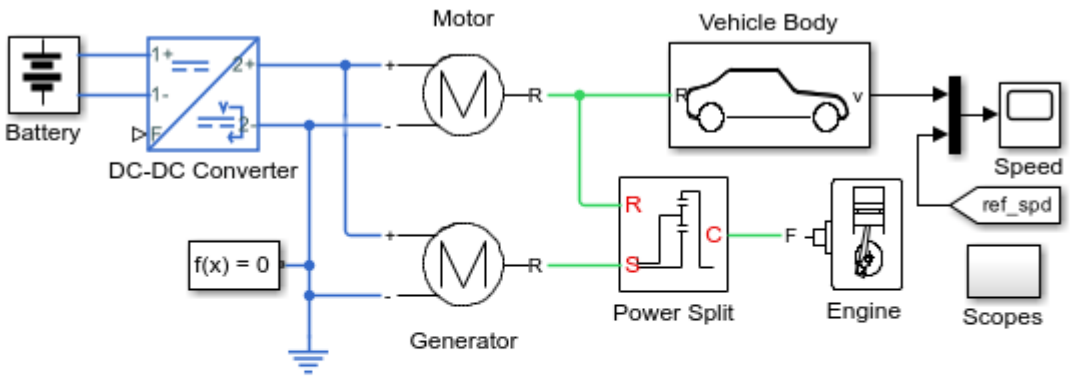

Vehicle Speed
Speed (km/hr)
300
200
100
0
0
10
20
30
40
50
60
70
80
90
100
Lap Time: 94.72 sec., All Corners Strategy
Lap Time: 94.97 sec., Selected Corners Strategy
Motor Torque
Torque (N*m)
500
0
-500
Time (s)

# Power Split Hybrid Vehicle Electrical Network

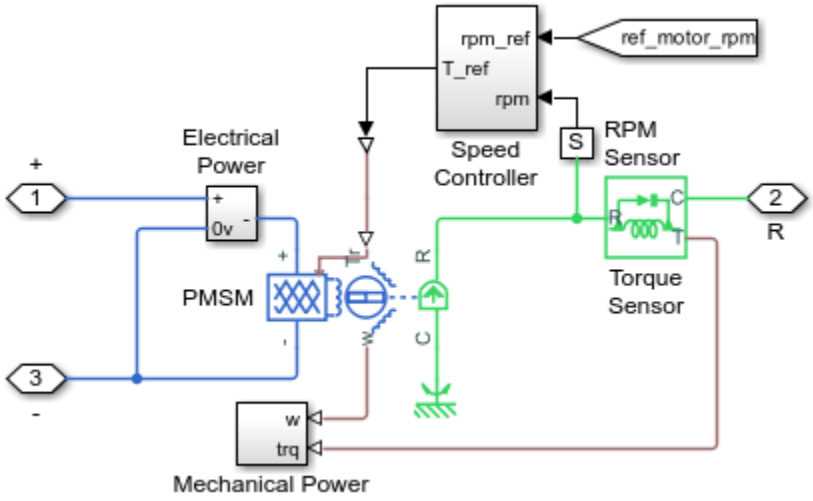
This example shows the basic architecture of a power-split hybrid transmission. The planetary gear, along with the motor and generator, acts like a variable ratio gear. In this test, the vehicle accelerates from 15 m/s to 20 m/s, and then decelerates back to 15 m/s. The power management strategy uses just electrical power to perform the maneuver.

The motor, generator, and DC-DC converter are modeled using Simscape™ Electrical™ library blocks. These blocks use energy-based system-level equations that result in an efficient simulation whilst still capturing conversion losses. As such, the model is suitable for supporting design of the power management strategy. This example can be directly compared with the Simscape Driveline™ sdl_hybrid_power_split example which also includes more accurate representations of the engine, tires and mechanical gears.

### Model

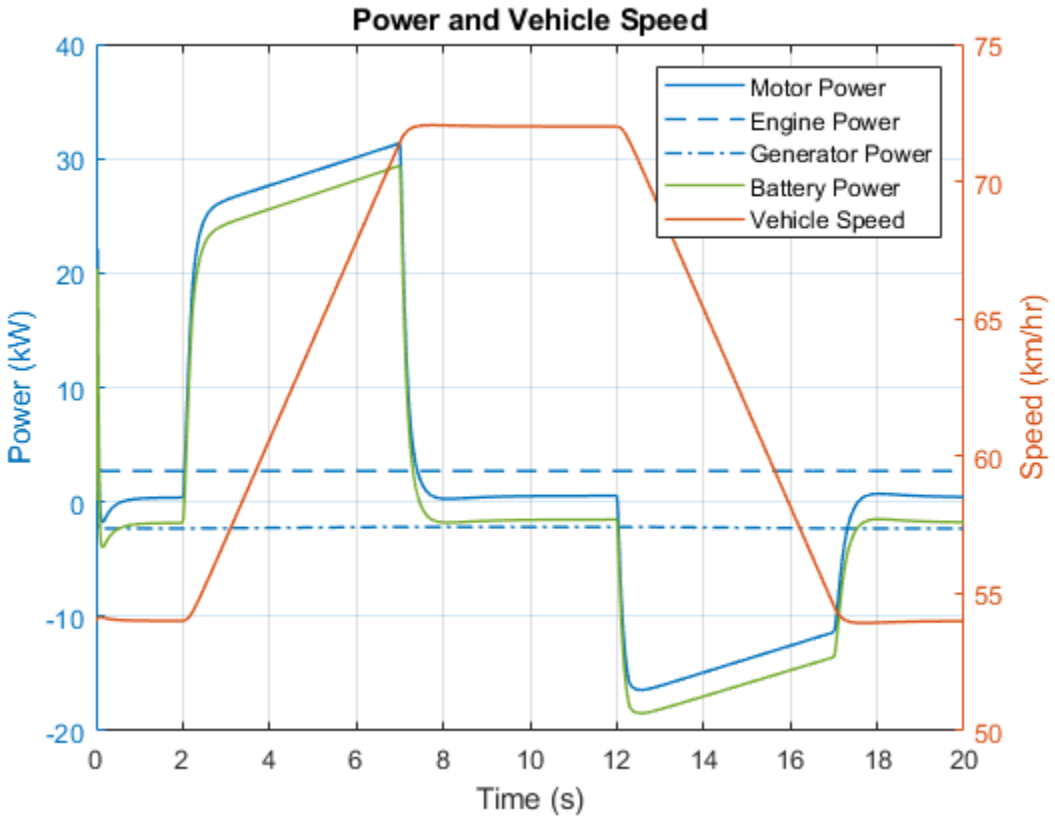


### Motor Subsystem

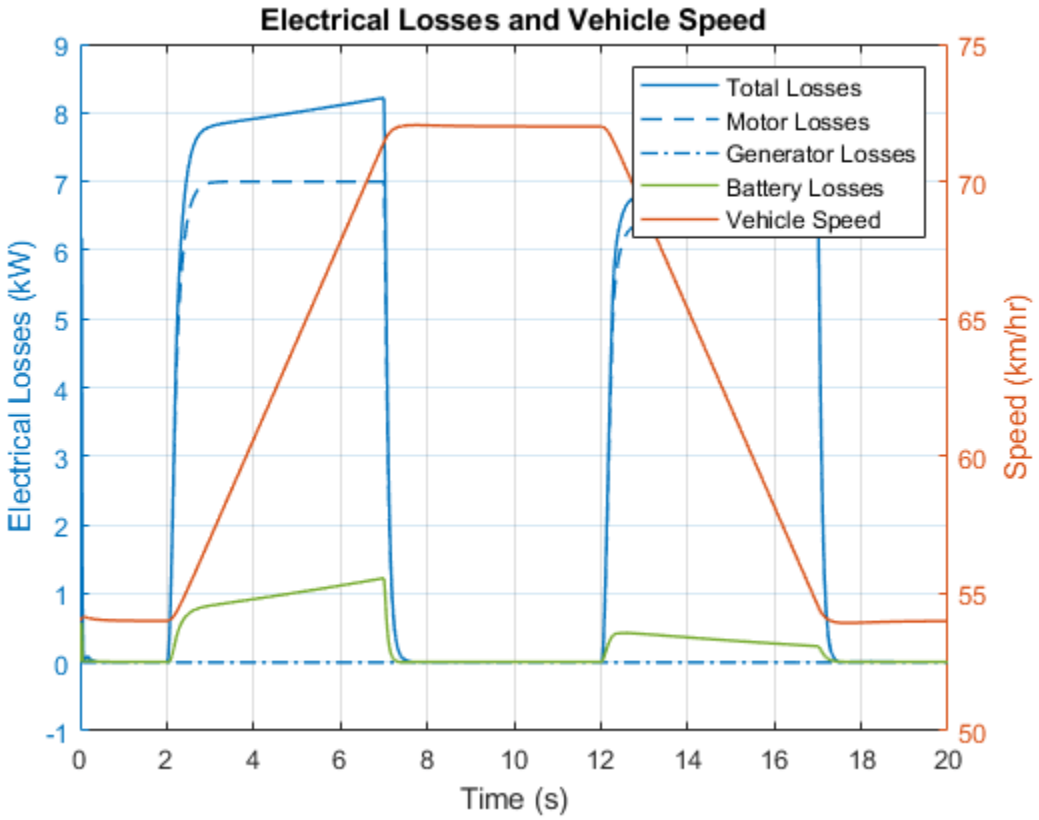



### Simulation Results from Simscape Logging

The plot below shows the flow of power from the engine, motor, and generator as the vehicle accelerates and decelerates. The generator supplies the DC network with a constant flow of power drawn from the engine. The motor draws power from the battery to accelerate the vehicle and then uses regenerative braking to feed that power back to the battery.




The plot below shows the electrical losses from the motor, generator, and battery as the vehicle accelerates and decelerates. The largest losses come from the motor and the battery.


Electrical Losses and Vehicle Speed
Total Losses
Motor Losses
Generator Losses
Battery Losses
Vehicle Speed
Electrical Losses (kW)
Speed (km/hr)
Time (s)
9
8
7
6
5
4
3
2
1
0
-1
75
70
65
60
55
50
0
2
4
6
8
10
12
14
16
18
20

# Import IPMSM Flux Linkage Data from Motor-CAD

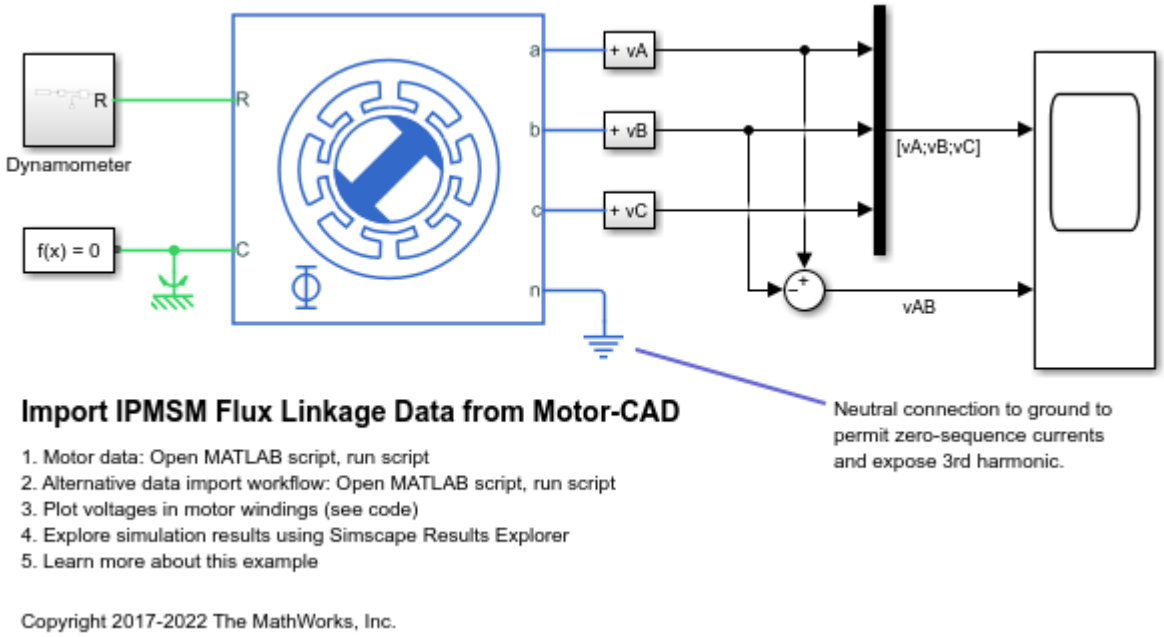
This example shows how you can import a motor design from Motor-CAD into a Simscape™ simulation.

### Flux Linkage Data

This example shows two approaches to import flux linkage from Motor-CAD. The first approach is to export the flux linkage and iron loss data to a MATLAB® data file using the Motor-CAD saturation and loss map export utility. The ee_import_fem_motorcad_sat_loss_map.m script implements this first approach. The second is to extract data using the Motor-CAD ActiveX interface. The ee_import_fem_motorcad_data.m implements this second approach.

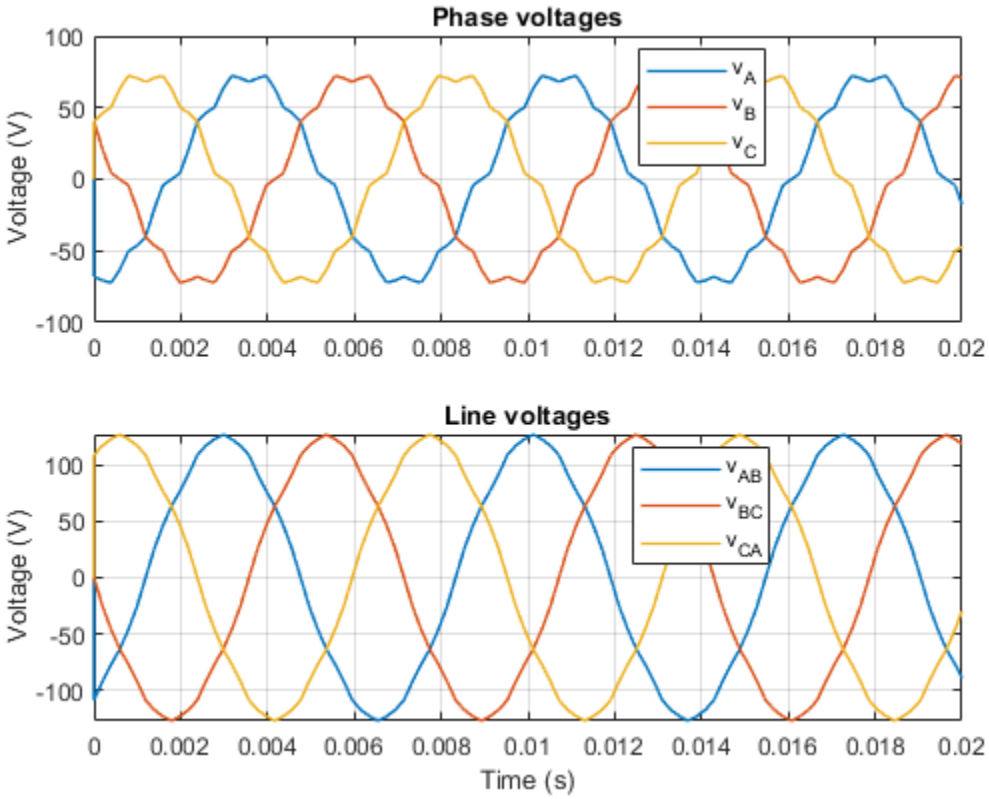
### Model

In this test, each winding phase is connected to a 1 megohm resistor. The neutral port is connected to electrical ground to allow non-zero zero-sequence currents. This allows the third harmonic to show up in the back EMFs. Both A-phase and Park's axes fluxes are exported from Motor-CAD, but you must use the A-phase data for higher harmonics to appear in the simulation results.




### Simulation Results from Simscape Logging

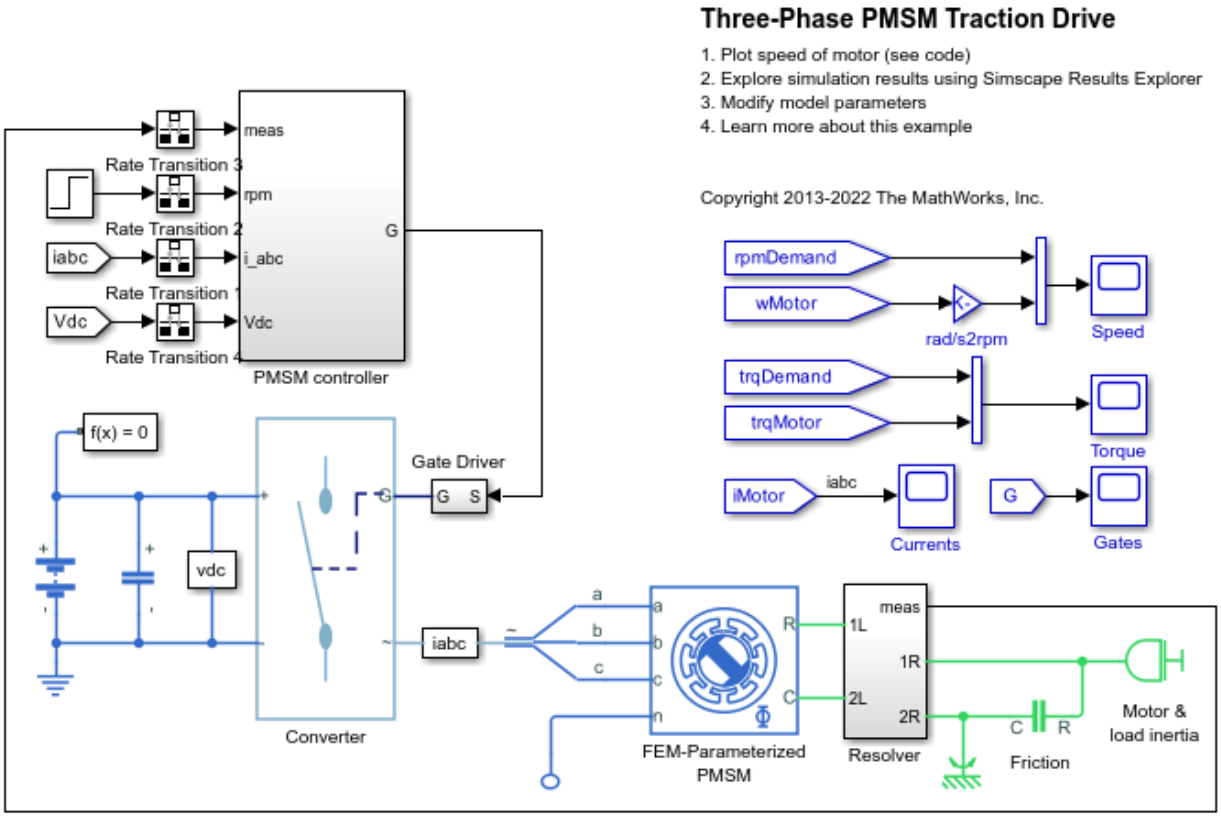
The plot below shows the simulated open-circuit voltages. Notice the nonlinear back EMF profile that would not be captured if using a simplified PMSM model with fixed back EMF constant.


Phase voltages
Voltage (V)
100
50
0
-50
-100
0
0.002
0.004
0.006
0.008
0.01
0.012
0.014
0.016
0.018
0.02
$v_A$
$v_B$
$v_C$
Line voltages
Voltage (V)
100
50
0
-50
-100
0
0.002
0.004
0.006
0.008
0.01
0.012
0.014
0.016
0.018
0.02
Time (s)
$v_{AB}$
$v_{BC}$
$v_{CA}$

# Three-Phase PMSM Traction Drive

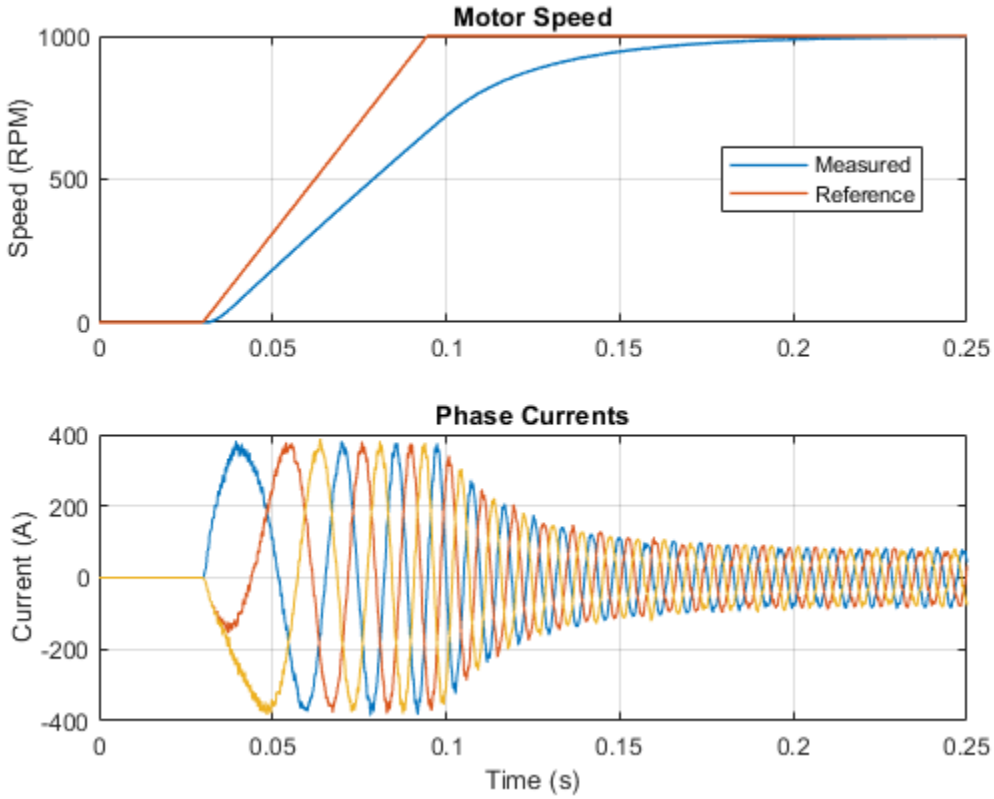
This example shows how to control the rotor speed in a permanent magnet synchronous machine (PMSM) based electrical-traction drive. A high-voltage battery feeds the FEM-Parameterized PMSM block through a controlled three-phase converter. A Rotational Friction block provides the load. The position and speed information are obtained by using a high-fidelity resolver. The PMSM controller subsystem includes a cascade control structure which has an outer speed-control loop and two inner current-control loops. During the 0.25 seconds simulation, the rotor speed demand is ramped-up from 0 to 1000 rpm.

### Model




### Simulation Results from Simscape Logging

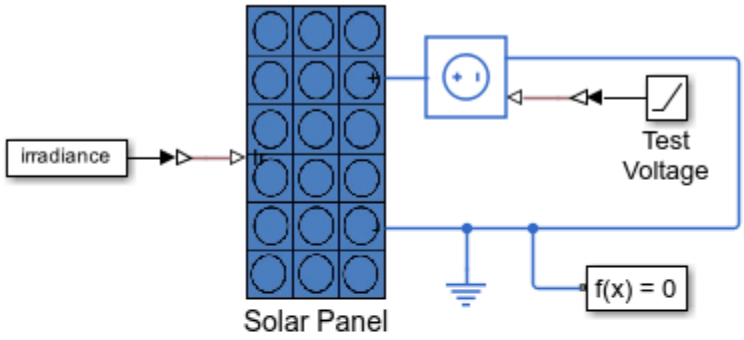
The plot below shows the requested and measured speed for the test, as well as the phase currents in the electric drive.


Motor Speed
Speed (RPM)
1000
500
0
0
0.05
0.1
0.15
0.2
0.25
Measured
Reference
Phase Currents
Current (A)
400
200
0
-200
-400
0
0.05
0.1
0.15
0.2
0.25
Time (s)

# Solar Panel Parameterization Validation

This example shows how to model a solar panel using information from a manufacturer datasheet. The data is imported and used to generate current-voltage and power-voltage curves for the solar panel. The power-voltage curve is useful for designing an inverter because it helps to identify the peak power for a given irradiance level and panel cell temperature

### Model




**Solar Panel Parameterization Validation**

1 Open script that inputs values from the datasheet
2. Plot current, power curves at varying conditions (see code)
3. Explore simulation results using Simscape Results Explorer
4. Learn more about this example

Copyright 2018-2022 The MathWorks, Inc.

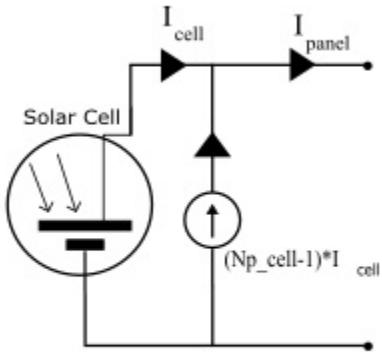
### Solar Panel Subsystem

The Solar Panel subsystem models a solar panel that contains parallel-connected strings of series-connected solar cells. The solar cell strings are modeled by a single Solar Cell block from the Simscape™ Electrical™ library. The number of parallel strings is defined by the workspace variable *Np_cell* and the number series-connected cells in each string is defined by workspace variable *Ns_cell*.

Connecting multiple solar cell strings in parallel can slow simulation because it increases the number of elements in a model. To avoid a reduction in performance, a controlled-current source scales up the current to the fulfill the demand of the requisite number of parallel paths. The current source is modeled by a Current-Controlled Current Source block from the Simscape™ Foundation Library.

The implicit assumptions are that:

- All of the solar cells are identical.
- Temperature and irradiance levels are the same for each cell.

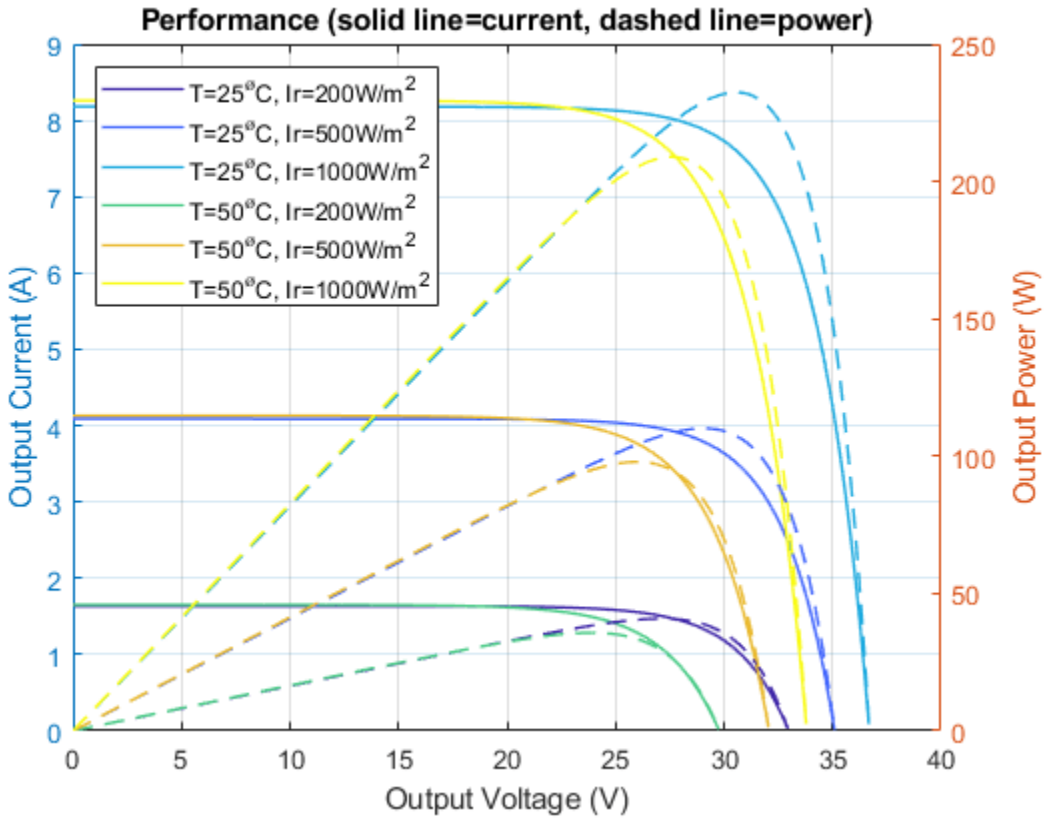



### Simulation Results

The plot shows current and power as a function of voltage for various irradiance and cell temperatures. The table shows the peak power values extracted from the plot. If the datasheet includes a plot like this, then it can be used to validate correct parameterization of the Solar Cell block.

Maximum power operating points

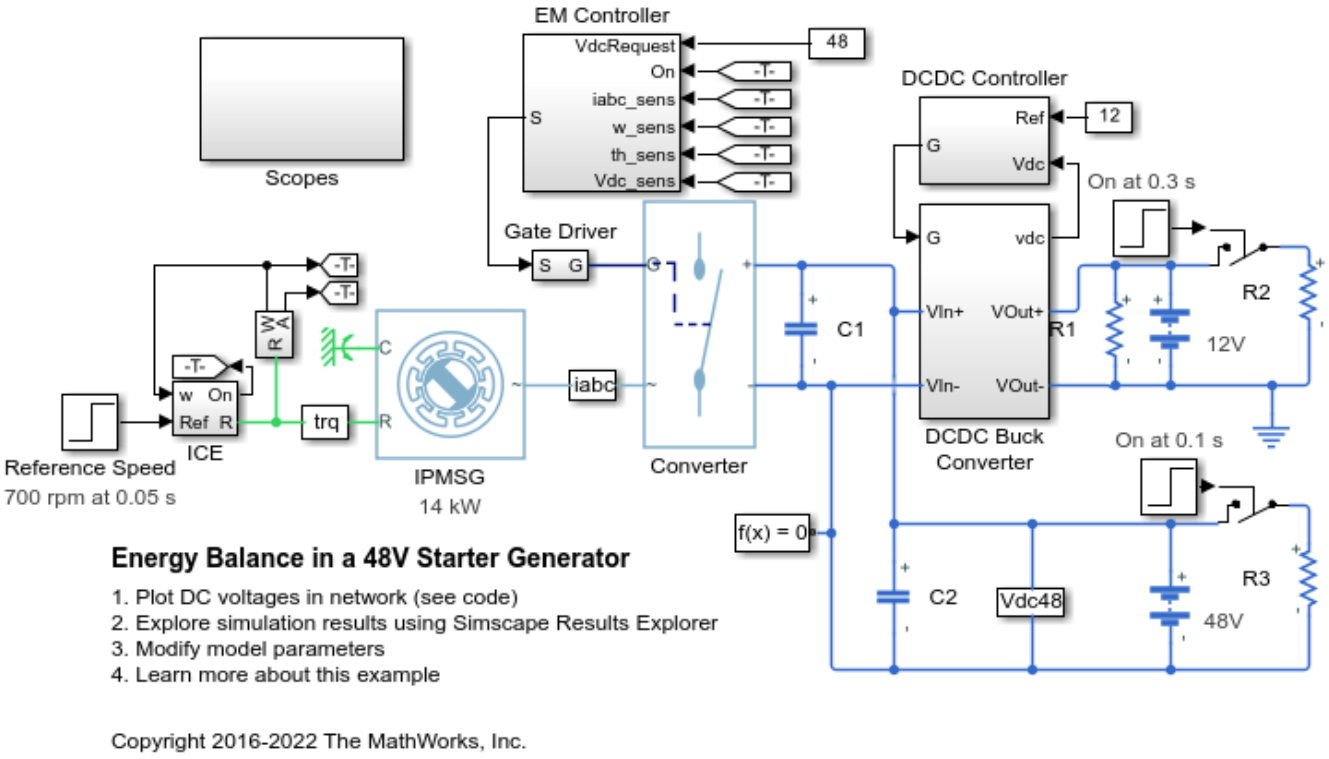
| PanelIrradiance | CellTemperature | MaximumCurrent | MaximumVoltage | MaximumPower |
|---|---|---|---|---|
| 200 | 25 | 1.5067 | 27.11 | 40.847 |
| 500 | 25 | 3.7887 | 29.066 | 110.12 |
| 1000 | 25 | 7.6059 | 30.557 | 232.41 |
| 200 | 50 | 1.4954 | 23.88 | 35.712 |
| 500 | 50 | 3.7665 | 25.992 | 97.899 |
| 1000 | 50 | 7.5762 | 27.576 | 208.92 |


Performance (solid line=current, dashed line=power)
T=25°C, Ir=200W/m2
T=25°C, Ir=500W/m2
T=25°C, Ir=1000W/m2
T=50°C, Ir=200W/m2
T=50°C, Ir=500W/m2
T=50°C, Ir=1000W/m2
Output Current (A)
Output Power (W)
Output Voltage (V)
0
1
2
3
4
5
6
7
8
9
0
50
100
150
200
250
0
5
10
15
20
25
30
35
40

# Energy Balance in a 48V Starter Generator

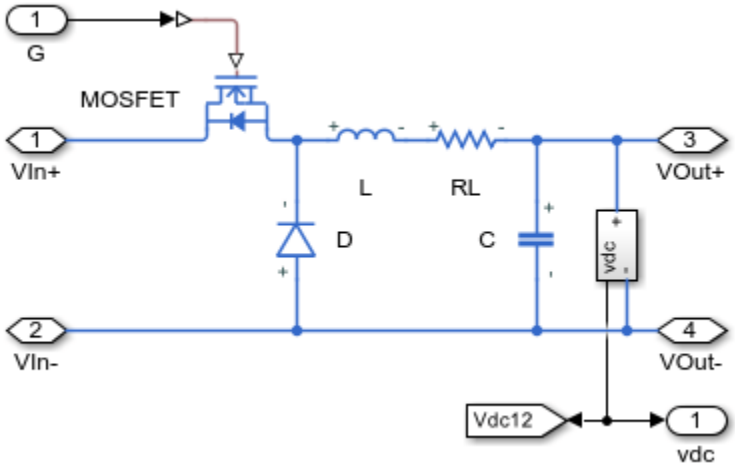
This example shows an interior permanent magnet synchronous machine (IPMSM) used as a starter/generator in a simplified 48V automotive system. The system contains a 48V electric network and a 12V electric network. The internal combustion engine (ICE) is represented by basic mechanical blocks. The IPMSM operates as a motor until the ICE reaches the idle speed and then it operates as a generator. The IPMSM supplies power to the 48V network, which contains the R3 power consumer. The 48V network supplies power to the 12V network which has two consumers: R1 and R2. The total simulation time (t) is 0.5 seconds. At t = 0.05 seconds, the ICE turns on. At t = 0.1 seconds, R3 switches on. At t = 0.3 seconds, R2 switches on and increases the load on the 12V electric network. The EM Controller subsystem includes a multi-rate PI-based cascade control structure which has an outer voltage-control loop and two inner current-control loops. The task scheduling in the Control subsystem is implemented as a Stateflow® state machine. The DCDC Controller subsystem implements a simple PI controller for the DC-DC Buck converter, which feeds the 12V network. The Scopes subsystem contains scopes that allow you to see the simulation results.

### Model



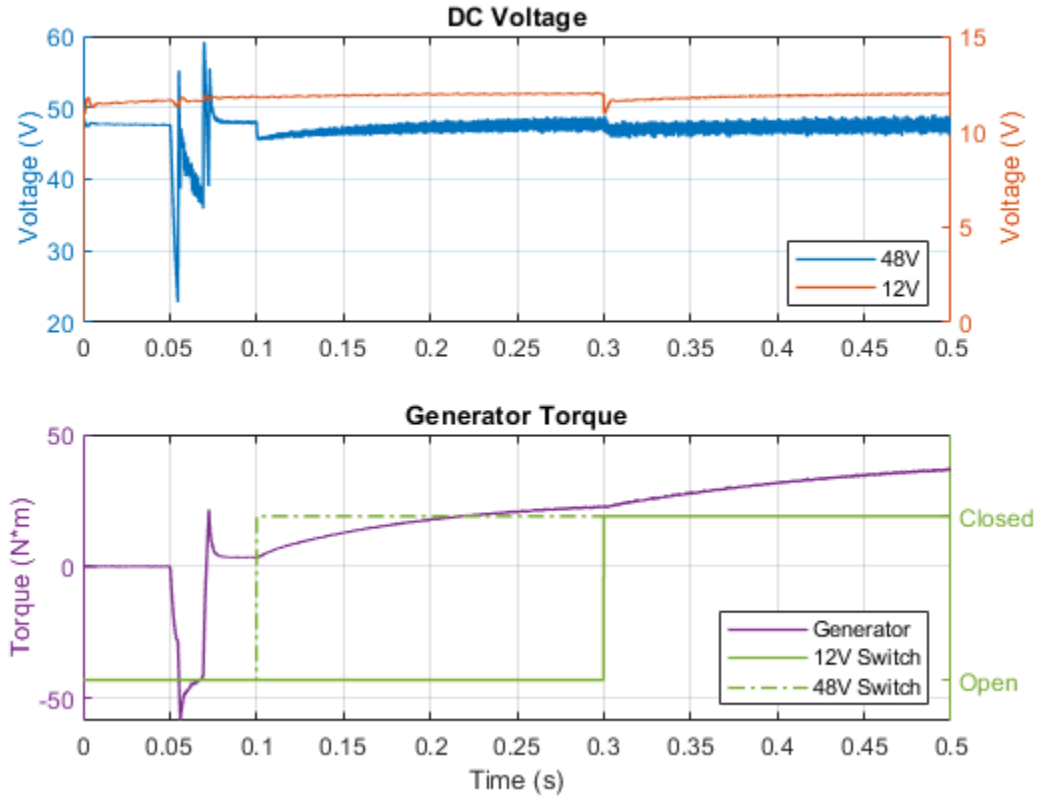

Copyright 2016-2022 The MathWorks, Inc.

### DCDC Buck Converter Subsystem




### Simulation Results from Simscape Logging

The plots below show how the system maintains voltages on the 48V and 12V DC busses as loads in the network change.

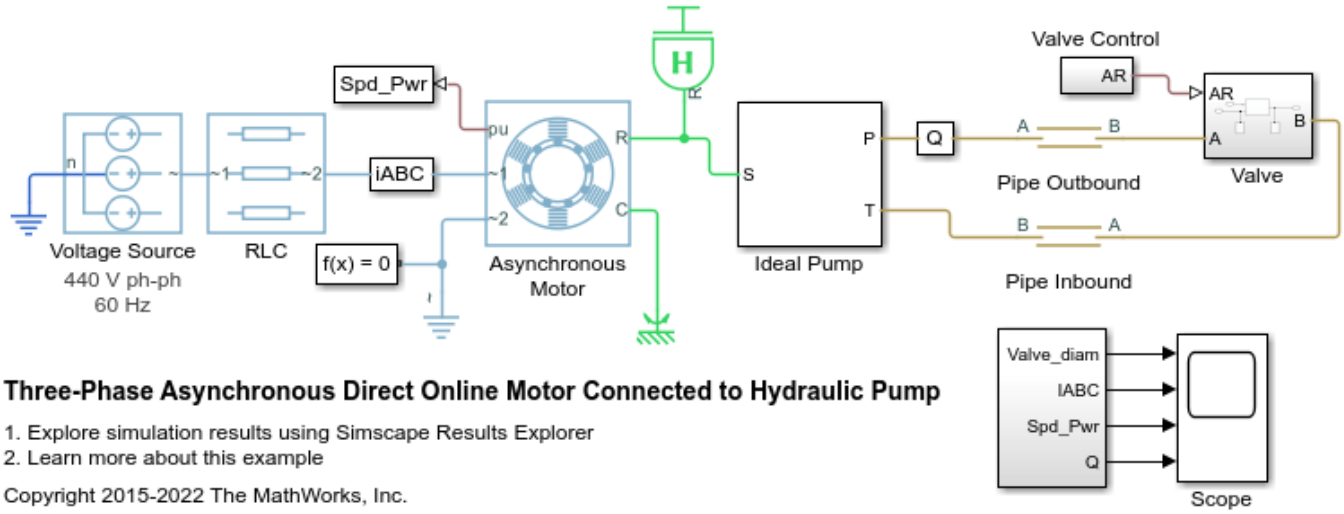


# Three-Phase Asynchronous Direct Online Motor Connected to Hydraulic Pump

In this example, an asynchronous motor is connected to an ideal pump. The diameter of the valve on the hydraulic system affects the speed and current drawn by the motor.

At 0 seconds, the motor is at zero speed. It starts up and at 2.5 seconds, it reaches rated speed. At 9 seconds, the diameter of the valve reduces to 4cm and the motor stalls.

### Model



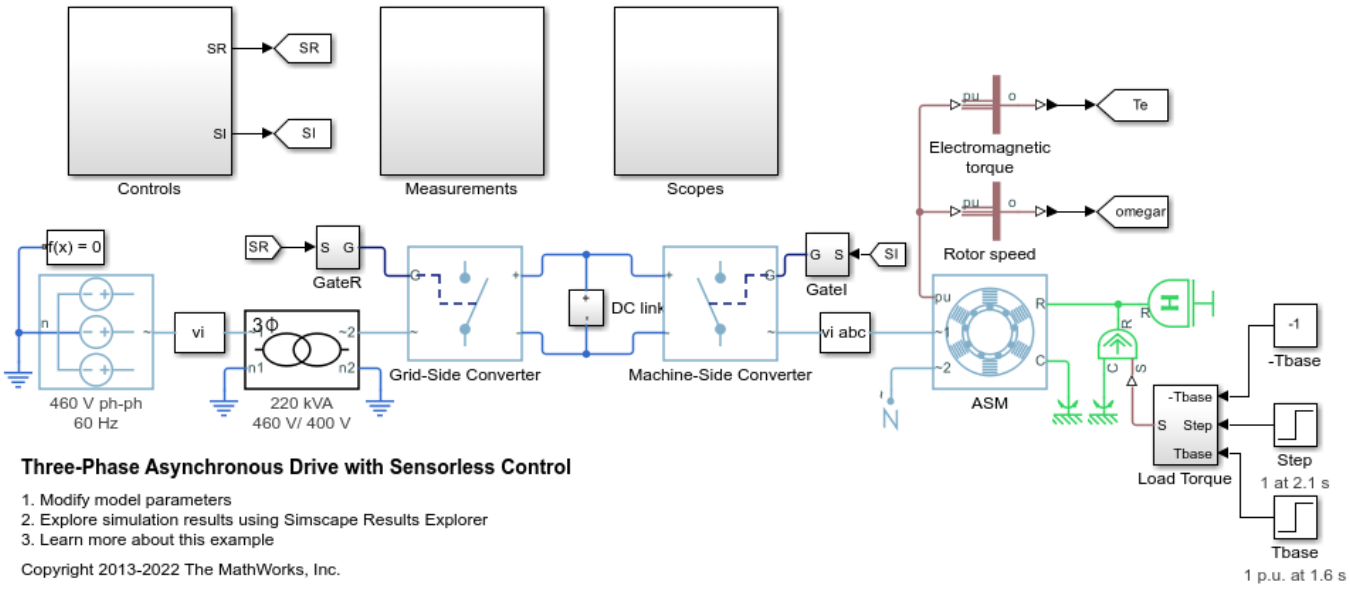

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 500 microseconds.

# Three-Phase Asynchronous Drive with Sensorless Control

This example shows how to control and analyze the operation of an Asynchronous Machine (ASM) using sensorless rotor field-oriented control. The model shows the main electrical circuit, with three additional subsystems containing the controls, measurements, and scopes. The Controls subsystem contains two controllers: one for the Grid-Side Converter (AC/DC) and one for the Machine-Side Converter (DC/AC). The Scopes subsystem contains two time scopes: one for the Grid-Side Converter and one for the ASM. When the model is executed, a Spectrum Analyzer opens and displays frequency data for the A-Phase Supply Current.

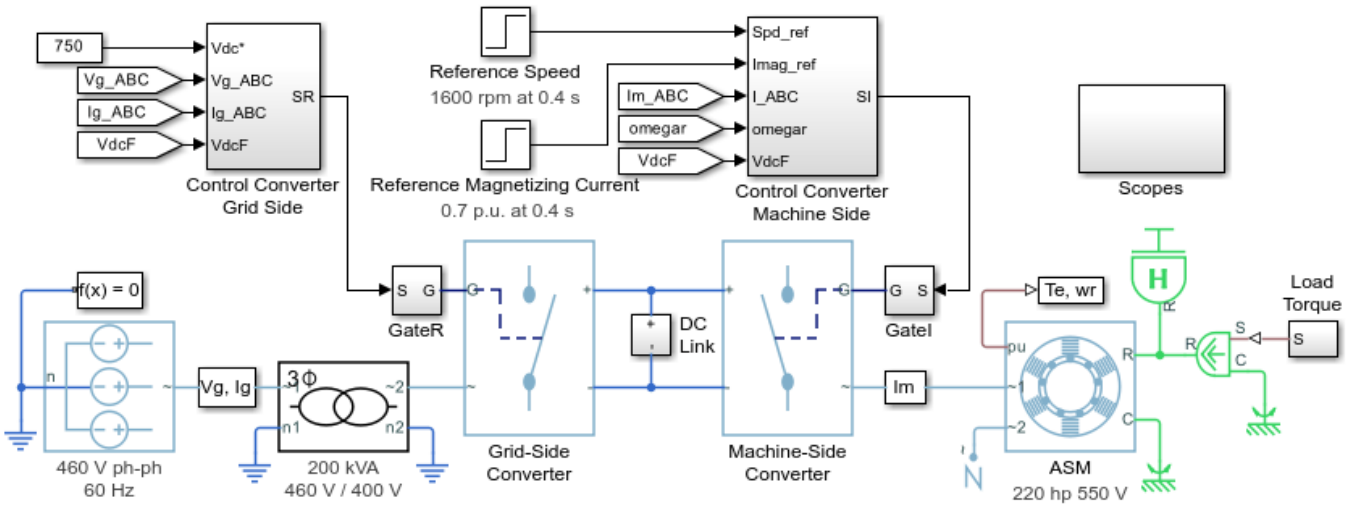
### Model



# Three-Phase Asynchronous Drive with Sensor Control

This example shows how to control and analyze the operation of an Asynchronous Machine (ASM) using sensored rotor field-oriented control. The model shows the main electrical circuit, with three additional subsystems containing the controls, measurements, and scopes. The Controls subsystem contains two controllers: one for the Grid-Side Converter (AC/DC) and one for the Machine-Side Converter (DC/AC). The Scopes subsystem contains two time scopes: one for the Grid-Side Converter and one for the ASM. When the model is executed, a Spectrum Analyzer opens and displays frequency data for the A-Phase Supply Current.

### Model




**Three-Phase Asynchronous Drive with Sensor Control**

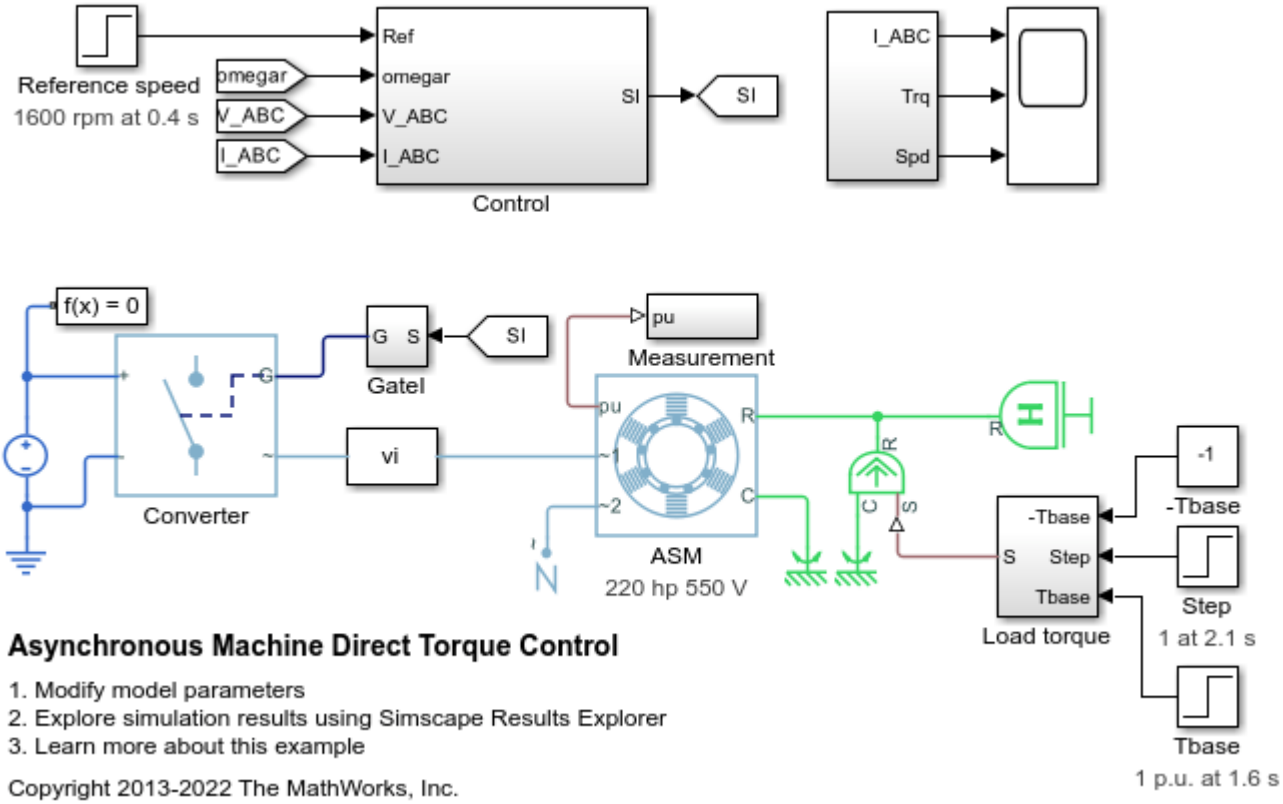
1. Modify model parameters
2. Explore simulation results using Simscape Results Explorer
3. Learn more about this example

Copyright 2013-2022 The MathWorks, Inc.

# Asynchronous Machine Direct Torque Control

This example shows how to control an asynchronous machine (ASM) using the direct-torque control method. A PI-based speed controller supplies the torque reference. The direct-torque controller generates the inverter pulses.

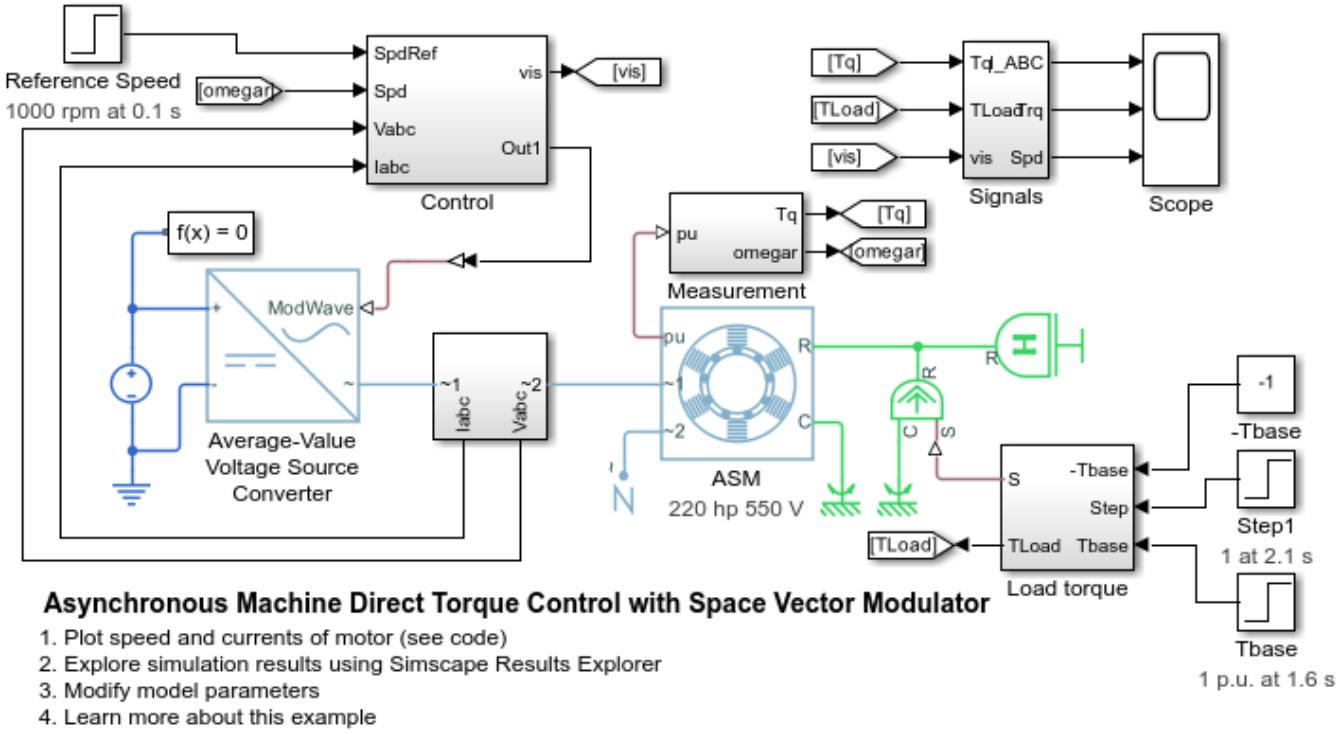
**Model**



# Asynchronous Machine Direct Torque Control with Space Vector Modulator

This example shows how to control an asynchronous machine (ASM) using the direct-torque control method with space vector modulator. A PI-based speed controller supplies the torque reference. The direct-torque controller generates the reference voltages required by the space vector modulator. A DC voltage source feeds the ASM through a controlled average-value voltage source converter.

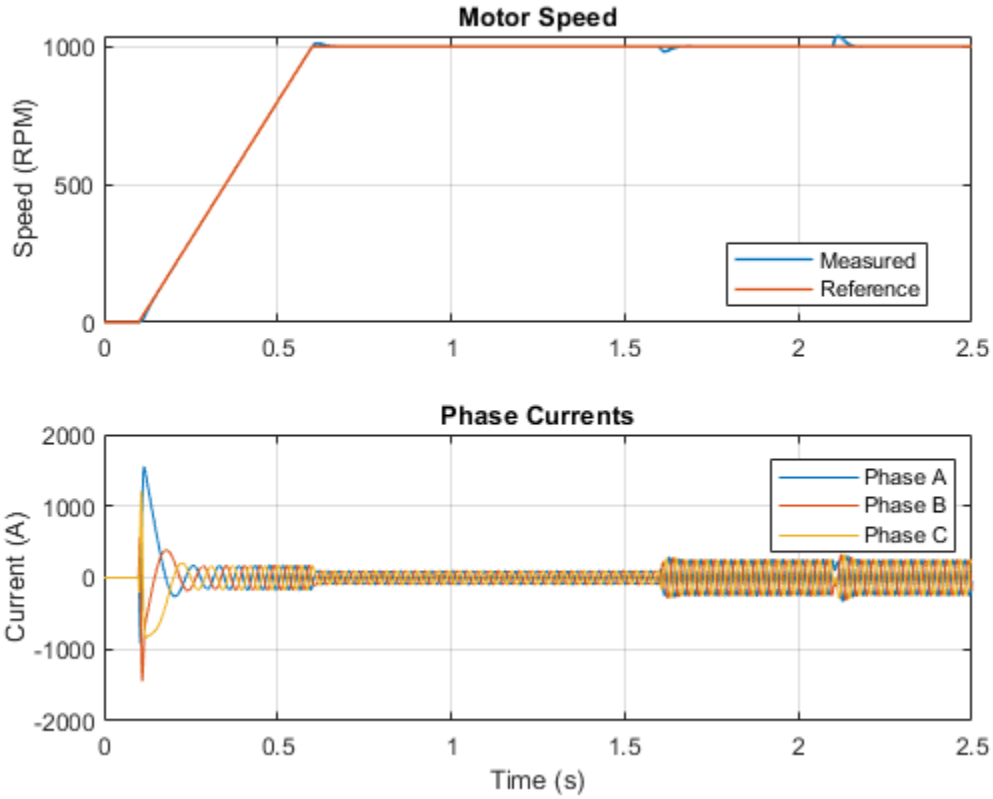
### Model




Copyright 2013-2022 The MathWorks, Inc.

### Simulation Results from Simscape Logging

The plot below shows the requested and measured speed for the test, as well as the phase currents in the electric drive.


Motor Speed
Speed (RPM)
1000
500
0
0
0.5
1
1.5
2
2.5
Measured
Reference
Phase Currents
Current (A)
2000
1000
0
-1000
-2000
Phase A
Phase B
Phase C
Time (s)

# Three-Phase Asynchronous Wind Turbine Generator

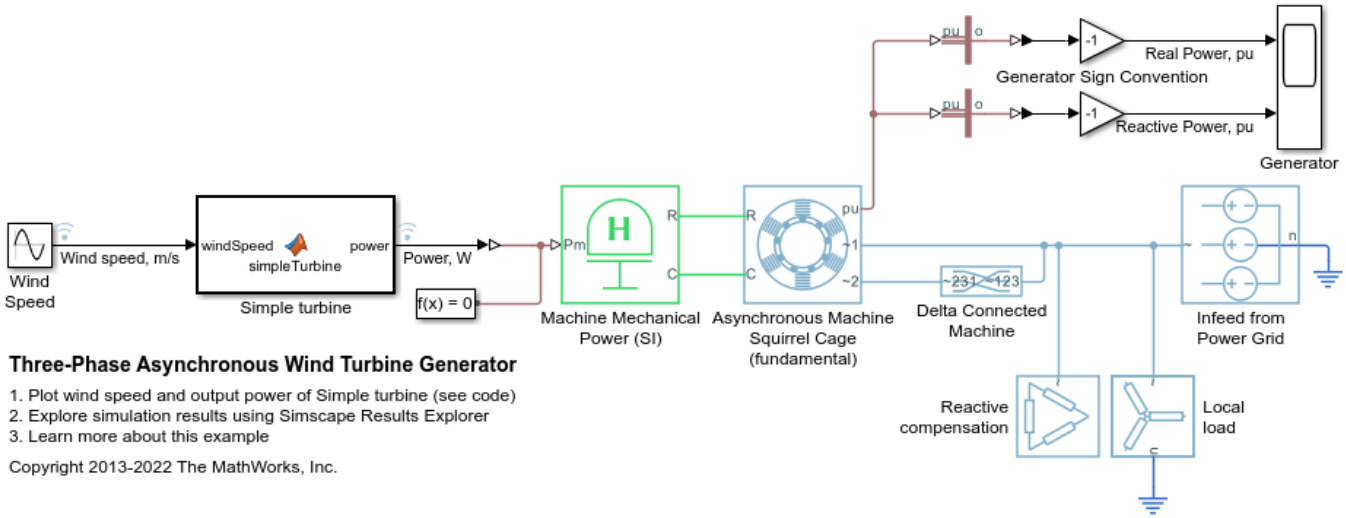
This example shows an induction machine used as a wind turbine generator. The Simple Turbine block converts wind speed to turbine output power by a simple output power versus wind speed characteristic.

When the wind speed is below the cut-in speed or above the cut-out speed, the machine generates zero real power. The machine always consumes reactive power. The Reactive Compensation block offsets the machine's reactive power requirement.

The local load consumes 75kW. The infeed from the power grid meets any wind turbine generation shortfall. When the generator produces more than 75kW, excess power is exported to the grid.

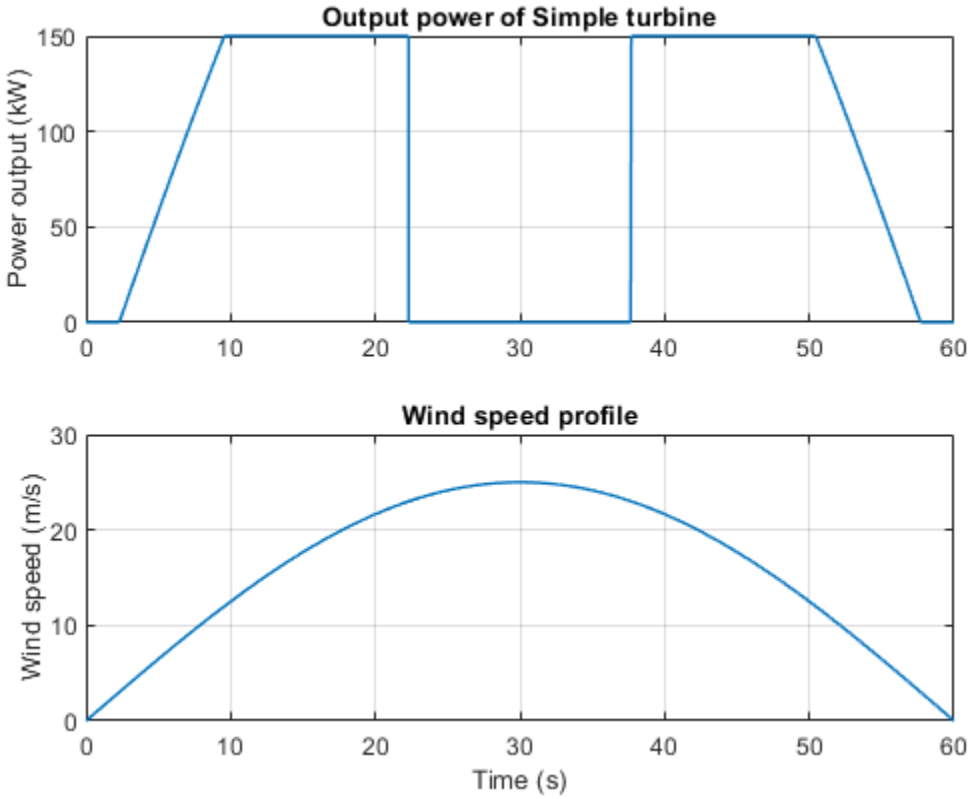
The reactive power compensator is dimensioned to supply 90 kvar when a 440 V phase-to-phase voltage is applied across its terminals.

### Model




### Simulation Results from Simscape Logging

The plot below shows the input wind speed and output power of the Simple turbine block.



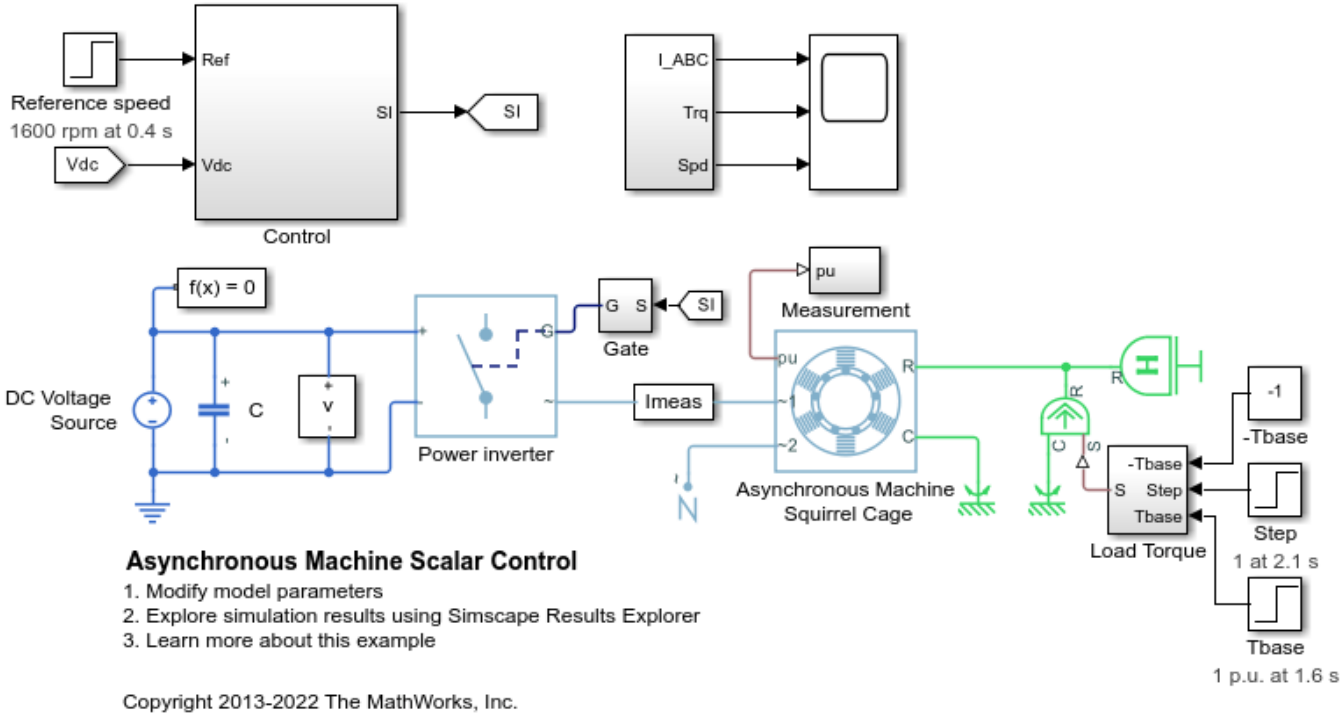

**Results from Real-Time Simulation**

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 100 microseconds.

# Asynchronous Machine Scalar Control

This example shows how to control the rotor speed in an asynchronous machine (ASM) drive using the scalar V/f control method. The converter transforms a reference speed to a reference electrical frequency. The controller generates reference voltages from the reference frequency by maintaining a constant voltage-to-frequency ratio through scalar V/f control.

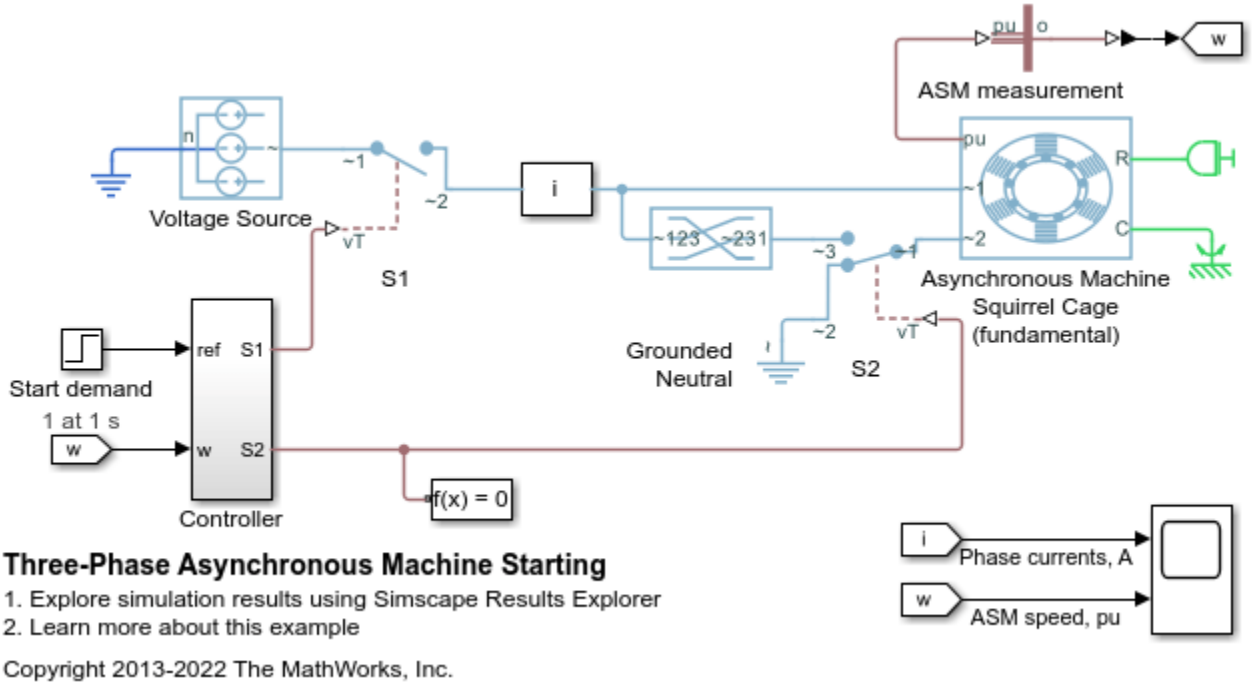
### Model



# Three-Phase Asynchronous Machine Starting

This example shows how to model a wye-delta starting circuit for an induction machine. When the supply is connected to the machine via switch S1, switch S2 is initially off resulting in the machine being connected in a wye configuration. Once the machine is close to synchronous speed, switch S2 is operated thereby reconnecting the machine in a delta configuration. The higher impedance seen by the supply when the motor is in wye configuration reduces the starting current, and causes less disruption to other connected loads.

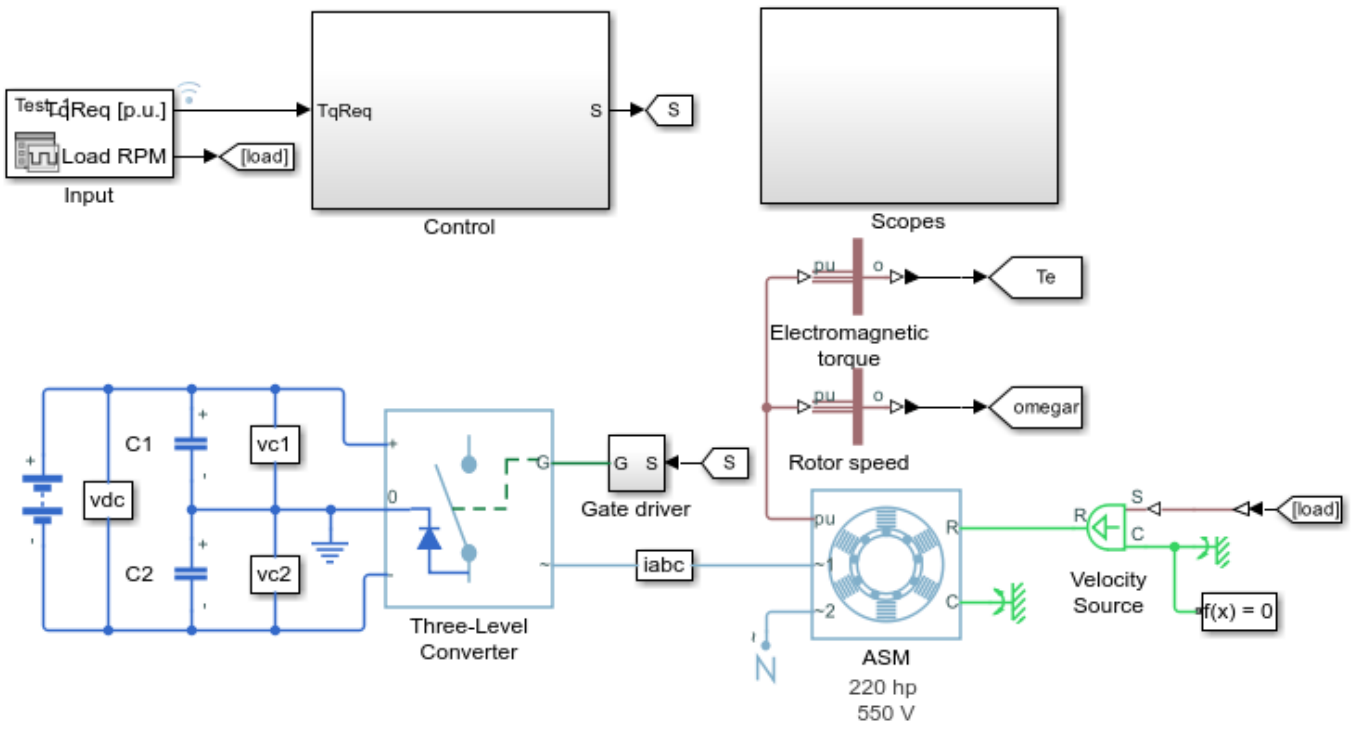
**Model**



# Torque Control in Three-Level Converter-Fed Asynchronous Machine Drive

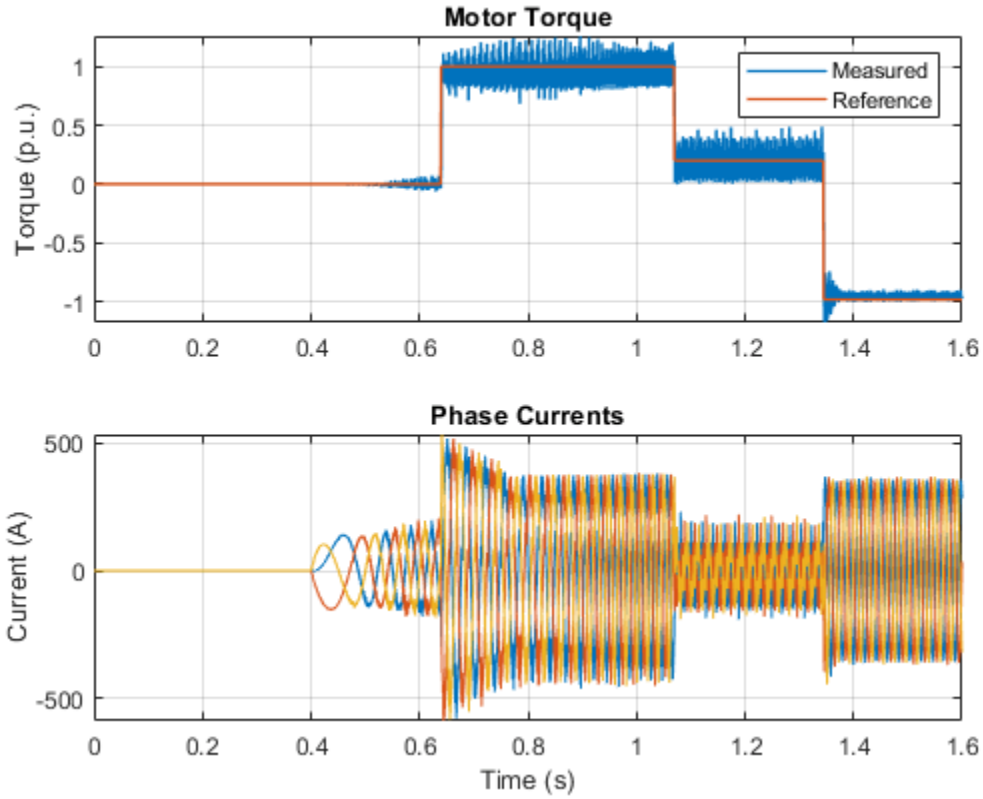
This example shows how to control the torque in an asynchronous machine (ASM) based electrical-traction drive. A high-voltage battery feeds the ASM through a three-phase three-level neutral-point clamped controlled converter. The ASM operates in both motoring and generating modes. An ideal angular velocity source provides the load. The Control subsystem uses the field-oriented control strategy to control the flux and torque. The current control is PI-based. A proportional controller regulates the neutral point voltage. The simulation uses several torque steps in both motor and generator modes. The Scopes subsystem contains scopes that allow you to see the simulation results.

### Model




### Simulation Results from Simscape Logging

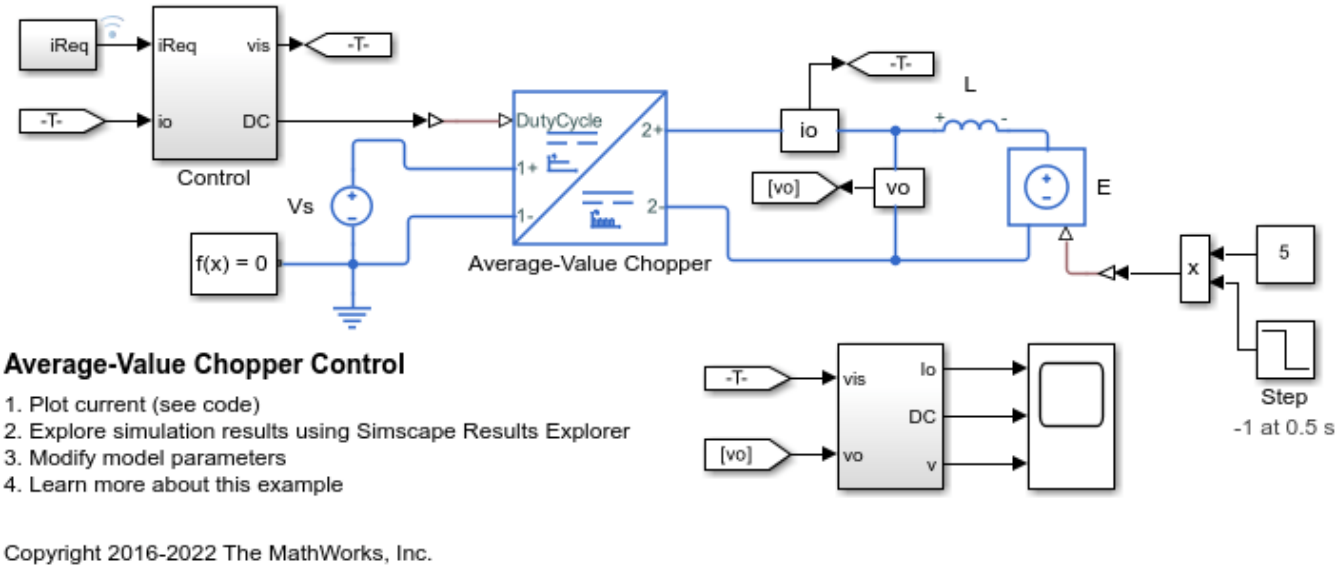
The plot below shows the requested and measured torque for the test, as well as the phase currents in the electric drive.


Motor Torque
Torque (p.u.)
1
0.5
0
-0.5
-1
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
Measured
Reference
Phase Currents
Current (A)
500
0
-500
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
Time (s)

# Average-Value Chopper Control

This example shows how to control a four-quadrant chopper. The Control subsystem implements a simple PI-based control algorithm for controlling the output current. An average-value Chopper model is used to speed up the simulation. The simulation uses both positive and negative references. The total simulation time (t) is 1 s. At t = 0.5 s, the polarity of the load DC source E changes.

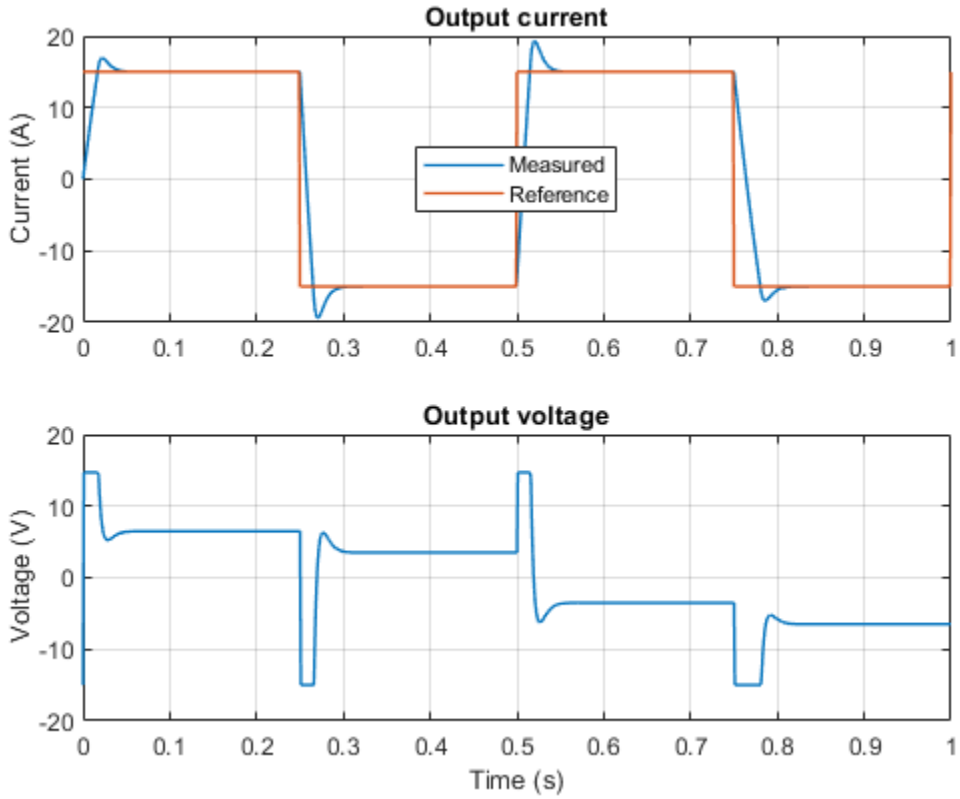
### Model




Copyright 2016-2022 The MathWorks, Inc.

### Simulation Results from Simscape Logging

The plot below shows the requested and measured current for the test and the output voltage in the circuit.



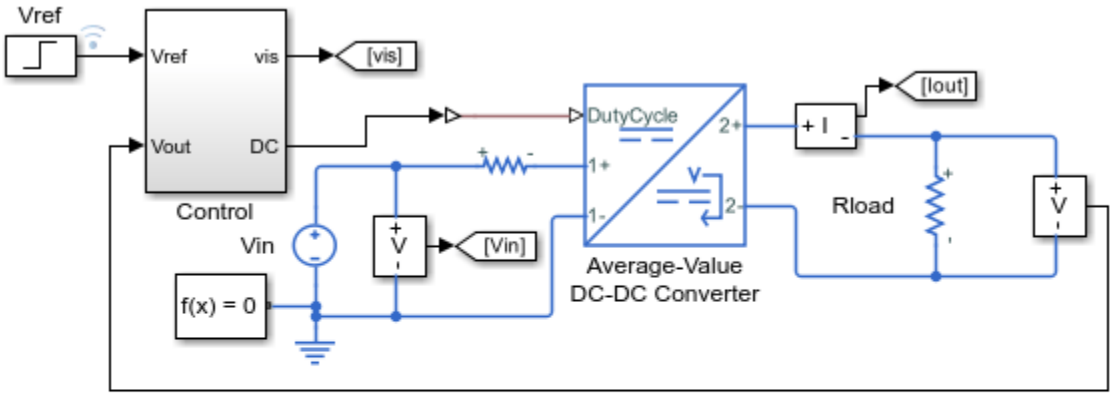

### Results from Real-Time Simulation

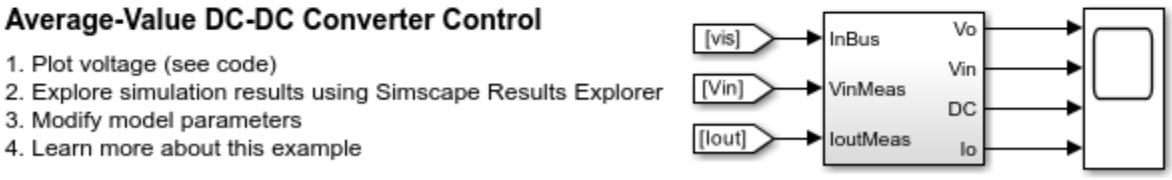
This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 10 microseconds.

# Average-Value DC-DC Converter Control

This example shows how to control the output voltage of a buck-boost converter. To adjust the duty cycle, the Control subsystem uses a PI-based control algorithm. An average-value DC-DC converter model is used to speed up the simulation. The input voltage and the system load are held constant throughout the simulation. The total simulation time (t) is 0.25 s. At t = 0.15 s, the voltage reference changes and the system switches from buck mode to boost mode.

### Model

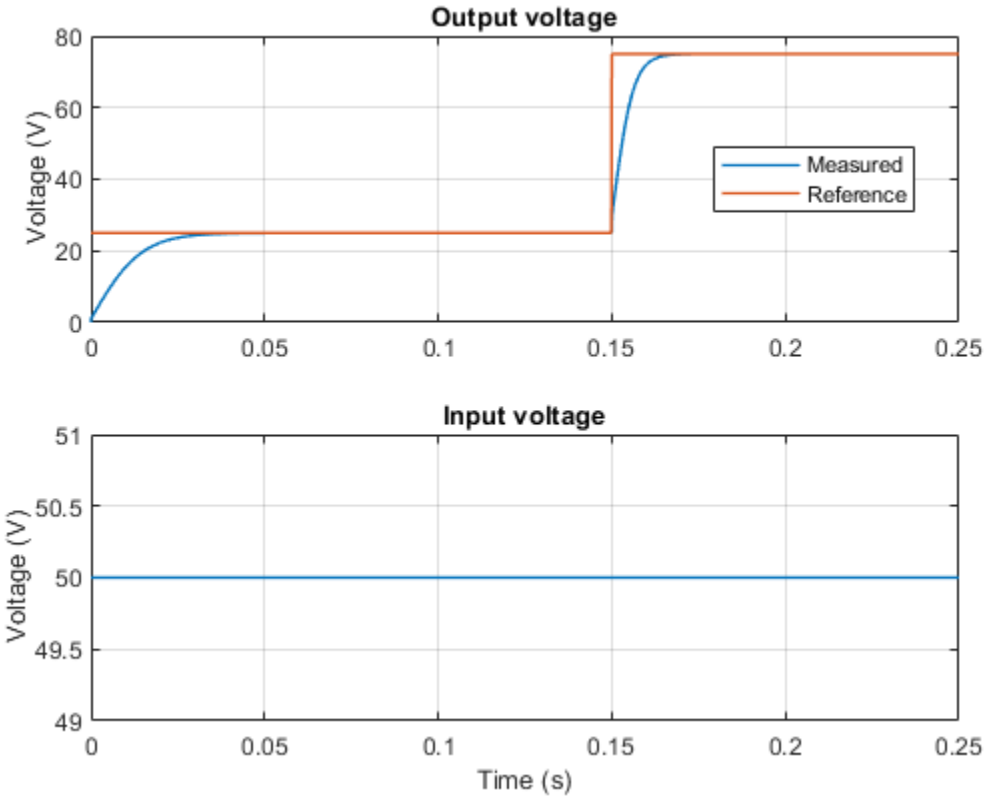



Copyright 2016-2022 The MathWorks, Inc.

### Simulation Results from Simscape Logging

The plot below shows the requested and measured voltage for the test and the input voltage in the circuit.



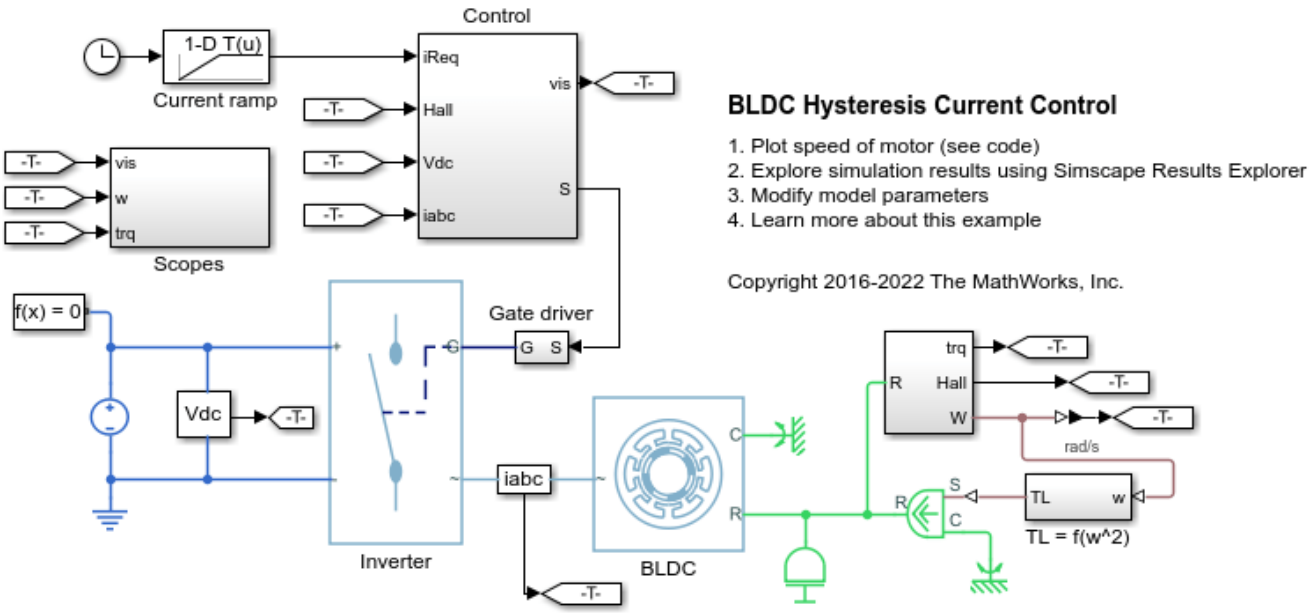

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 10 microseconds.

# BLDC Hysteresis Current Control

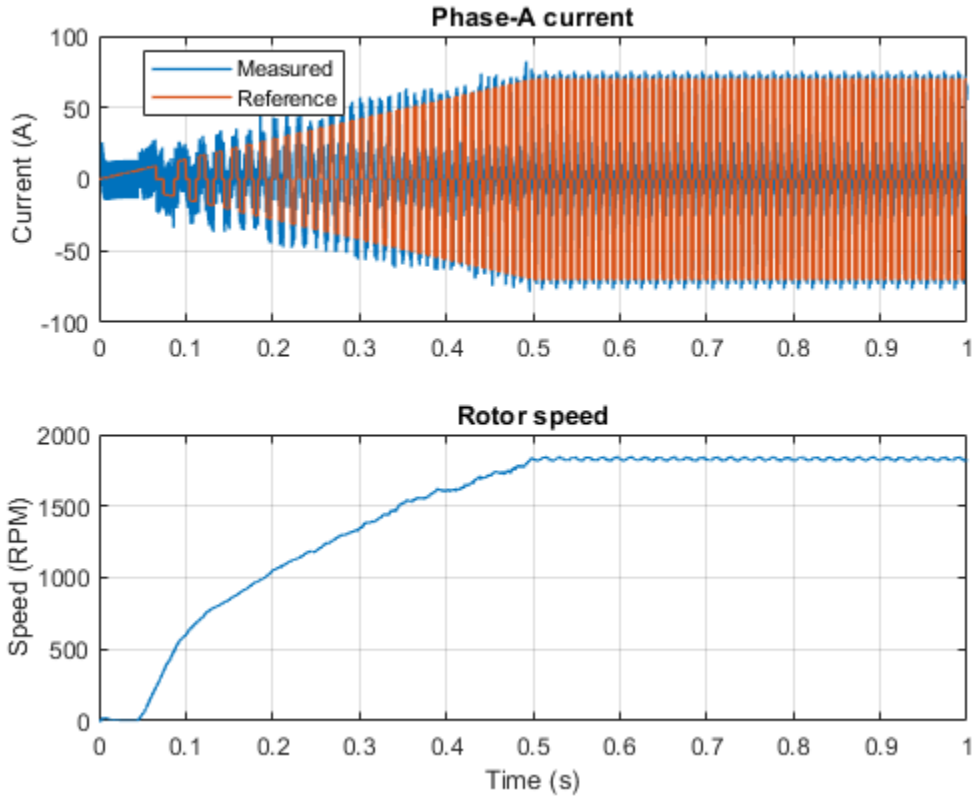
This example shows how to control the currents in a BLDC based electrical drive using hysteresis controllers. A DC voltage source feeds the BLDC through a controlled three-phase inverter. A ramp of current request is provided to the motor controller. The load torque is quadratically dependent on the rotor speed. The Control subsystem implements the hysteresis-based current control strategy. The Scopes subsystem contains scopes that allow you to see the simulation results.

### Model




### Simulation Results from Simscape Logging

The plot below shows the requested and measured current on phase A for the test and the rotor speed in the electric drive.



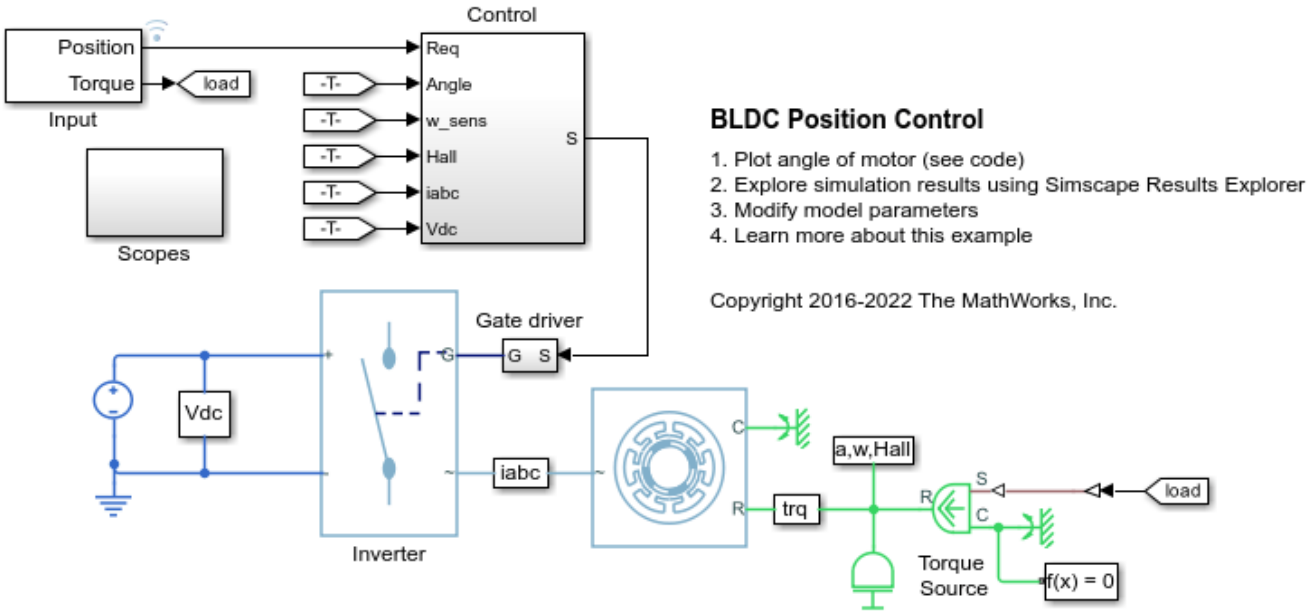

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 100 microseconds.

# BLDC Position Control

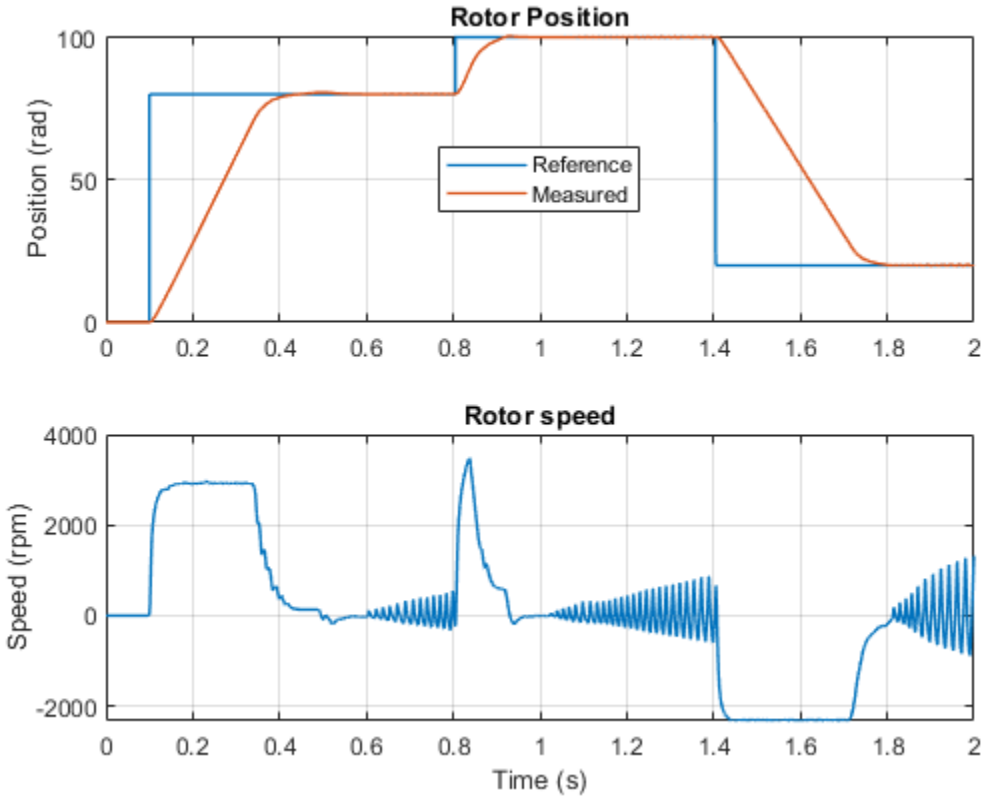
This example shows how to control the rotor angle in a BLDC based electrical drive. An ideal torque source provides the load. The Control subsystem uses a PI-based cascade control structure with three control loops, an outer position control loop, a speed control loop and an inner current control loop. The BLDC is fed by a controlled three-phase inverter. The gate signals for the inverter are obtained from hall signals. The simulation uses step references. The Scopes subsystem contains scopes that allow you to see the simulation results.

### Model




### Simulation Results from Simscape Logging

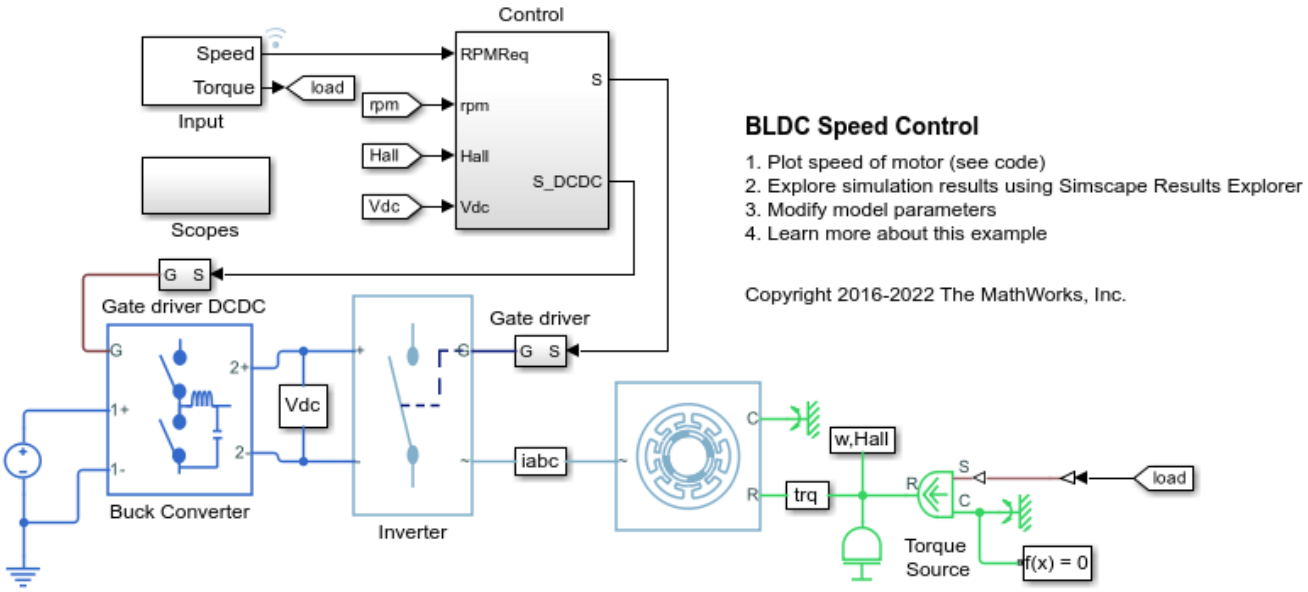
The plot below shows the requested and measured angle for the test and the rotor speed in the electric drive.


Rotor Position
Position (rad)
100
50
0
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
Reference
Measured
Rotor speed
Speed (rpm)
4000
2000
0
-2000
Time (s)

# BLDC Speed Control

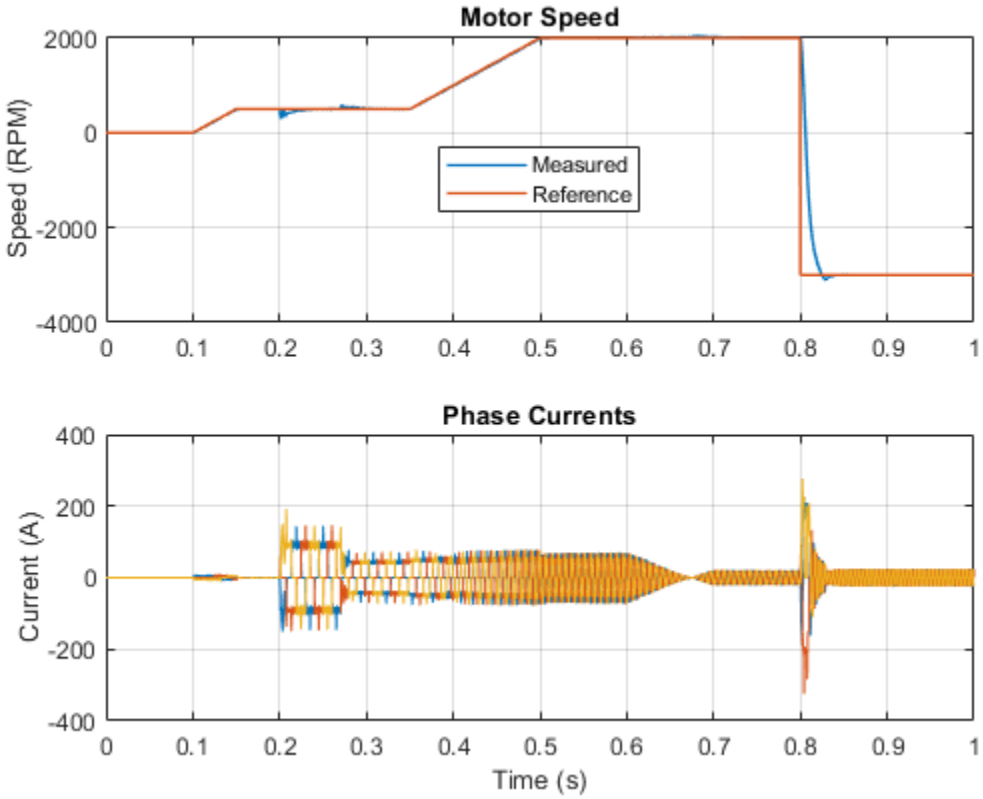
This example shows how to control the rotor speed in a BLDC based electrical drive. An ideal torque source provides the load. The Control subsystem uses a PI-based cascade control structure with an outer speed control loop and an inner dc-link voltage control loop. The dc-link voltage is adjusted through a DC-DC buck converter. The BLDC is fed by a controlled three-phase inverter. The gate signals for the inverter are obtained from hall signals. The simulation uses speed steps. The Scopes subsystem contains scopes that allow you to see the simulation results.

### Model




### Simulation Results from Simscape Logging

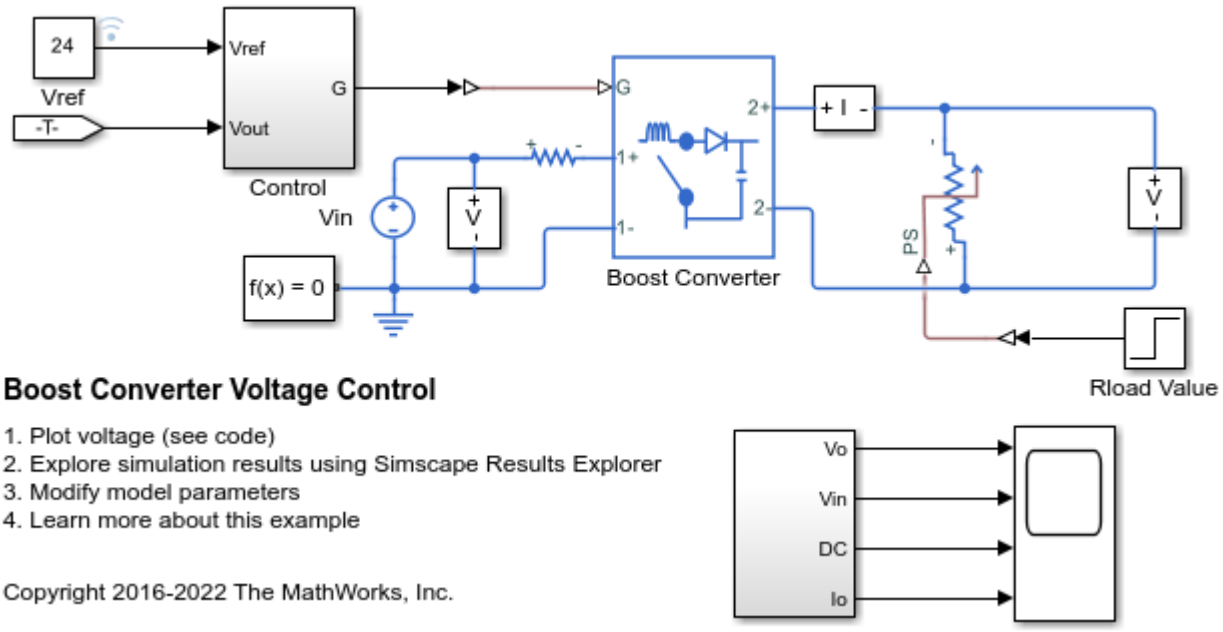
The plot below shows the requested and measured speed for the test and the phase currents in the electric drive.


Motor Speed
Speed (RPM)
2000
0
-2000
-4000
Measured
Reference
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Phase Currents
Current (A)
400
200
0
-200
-400
Time (s)

# Boost Converter Voltage Control

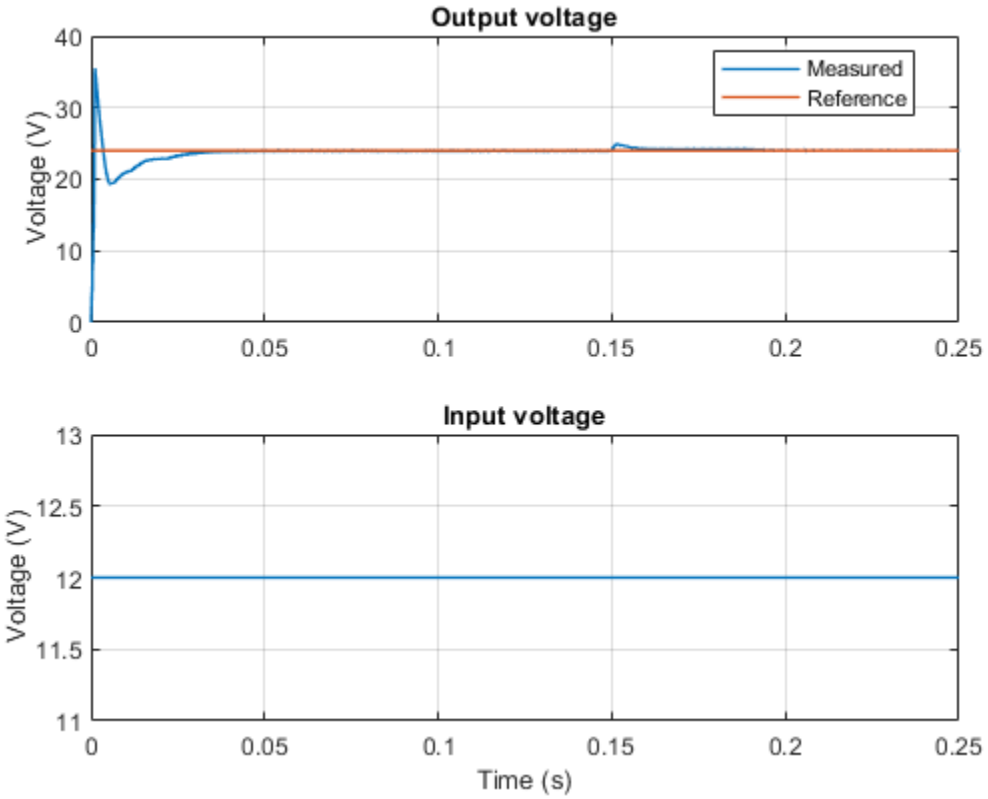
This example shows how to control the output voltage of a boost converter. To adjust the duty cycle, the Control subsystem uses a PI-based control algorithm. The input voltage is considered constant throughout the simulation. A variable resistor provides the load for the system. The total simulation time (t) is 0.25 seconds. At t = 0.15 seconds, the load changes.

### Model




### Simulation Results from Simscape Logging

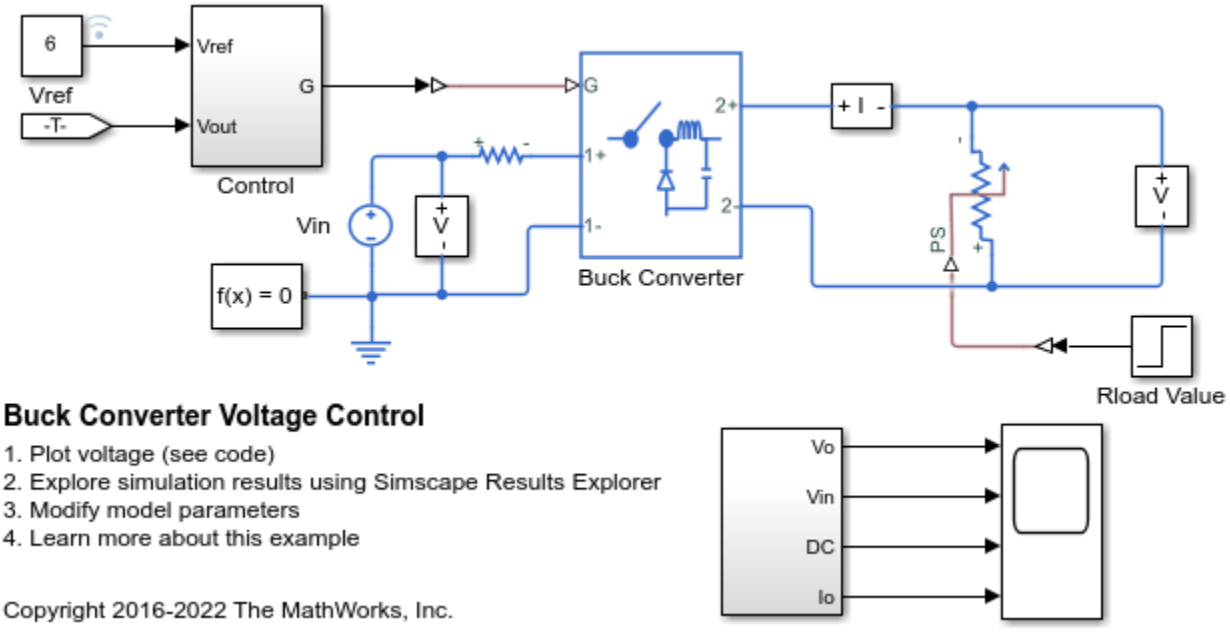
The plot below shows the requested and measured voltage for the test and the input voltage in the circuit.


Output voltage
Measured
Reference
Voltage (V)
40
30
20
10
0
0
0.05
0.1
0.15
0.2
0.25
Input voltage
Voltage (V)
13
12.5
12
11.5
11
0
0.05
0.1
0.15
0.2
0.25
Time (s)

# Buck Converter Voltage Control

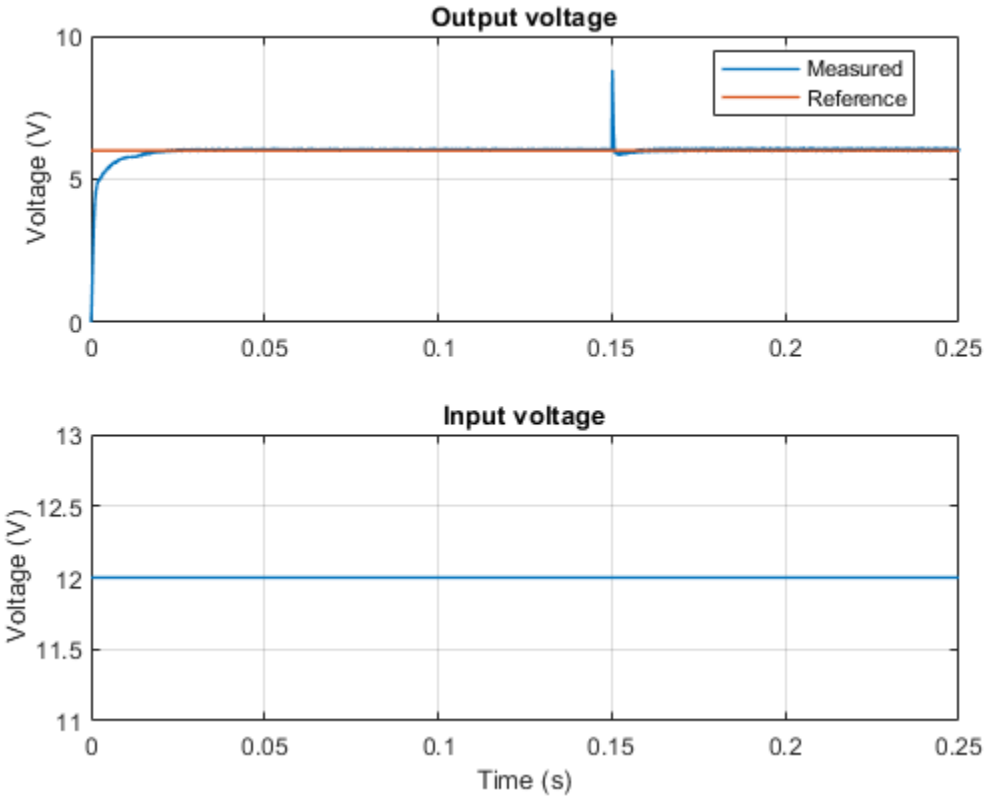
This example shows how to control the output voltage of a buck converter. To adjust the duty cycle, the Control subsystem uses a PI-based control algorithm. The input voltage is considered constant throughout the simulation. A variable resistor provides the load for the system. The total simulation time (t) is 0.25 seconds. At t = 0.15 seconds, the load changes.

### Model




### Simulation Results from Simscape Logging

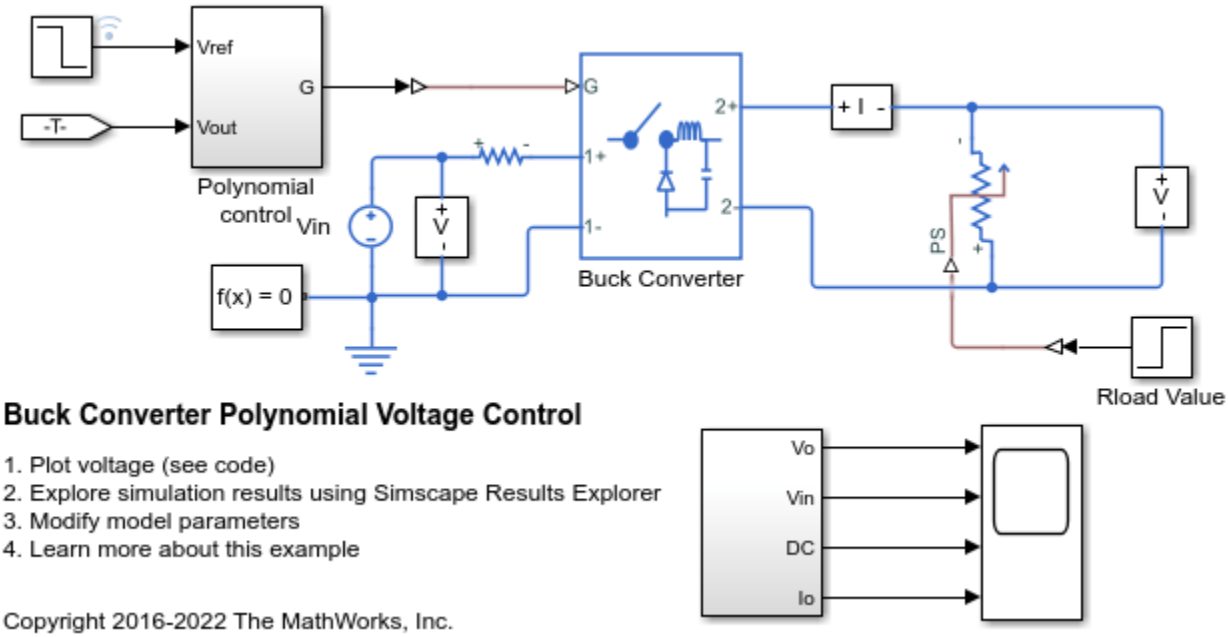
The plot below shows the requested and measured voltage for the test and the input voltage in the circuit.


Output voltage
Measured
Reference
Voltage (V)
10
5
0
0
0.05
0.1
0.15
0.2
0.25
Input voltage
Voltage (V)
13
12.5
12
11.5
11
0
0.05
0.1
0.15
0.2
0.25
Time (s)

# Buck Converter Polynomial Voltage Control

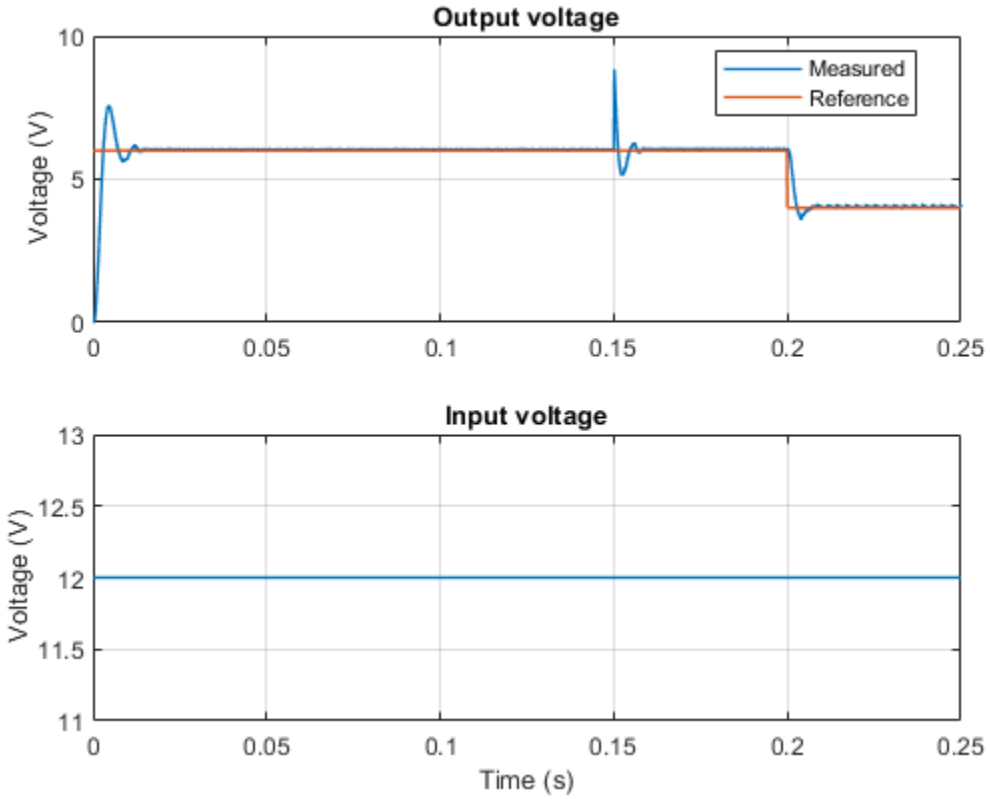
This example shows how to control the output voltage of a buck converter using a polynomial RST controller. The RST controller adjusts the duty cycle. The input voltage is considered constant throughout the simulation. A variable resistor provides the load for the system. The total simulation time (t) is 0.25 seconds. At t = 0.15 seconds, the load is changed. At t = 0.2 seconds, the voltage reference is changed from 6V to 4V.

### Model




### Simulation Results from Simscape Logging

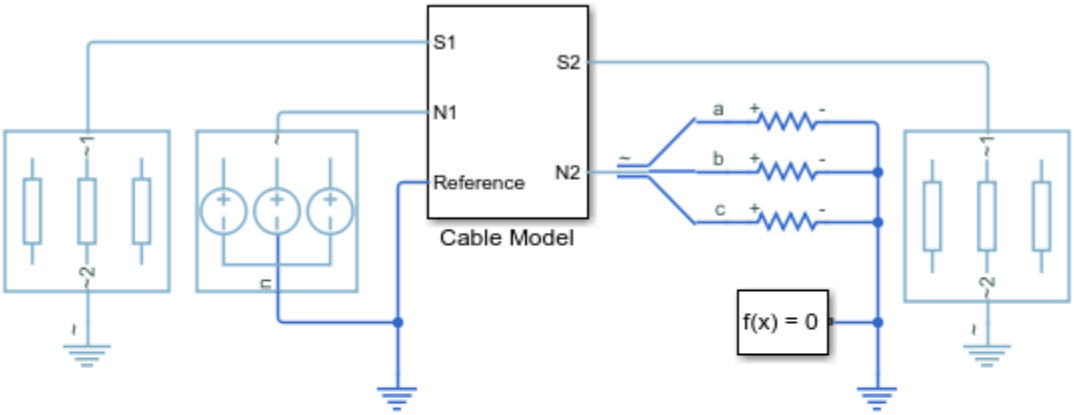
The plot below shows the requested and measured voltage for the test and the input voltage in the circuit.


Output voltage
Measured
Reference
Voltage (V)
10
5
0
0
0.05
0.1
0.15
0.2
0.25
Input voltage
Voltage (V)
13
12.5
12
11.5
11
0
0.05
0.1
0.15
0.2
0.25
Time (s)

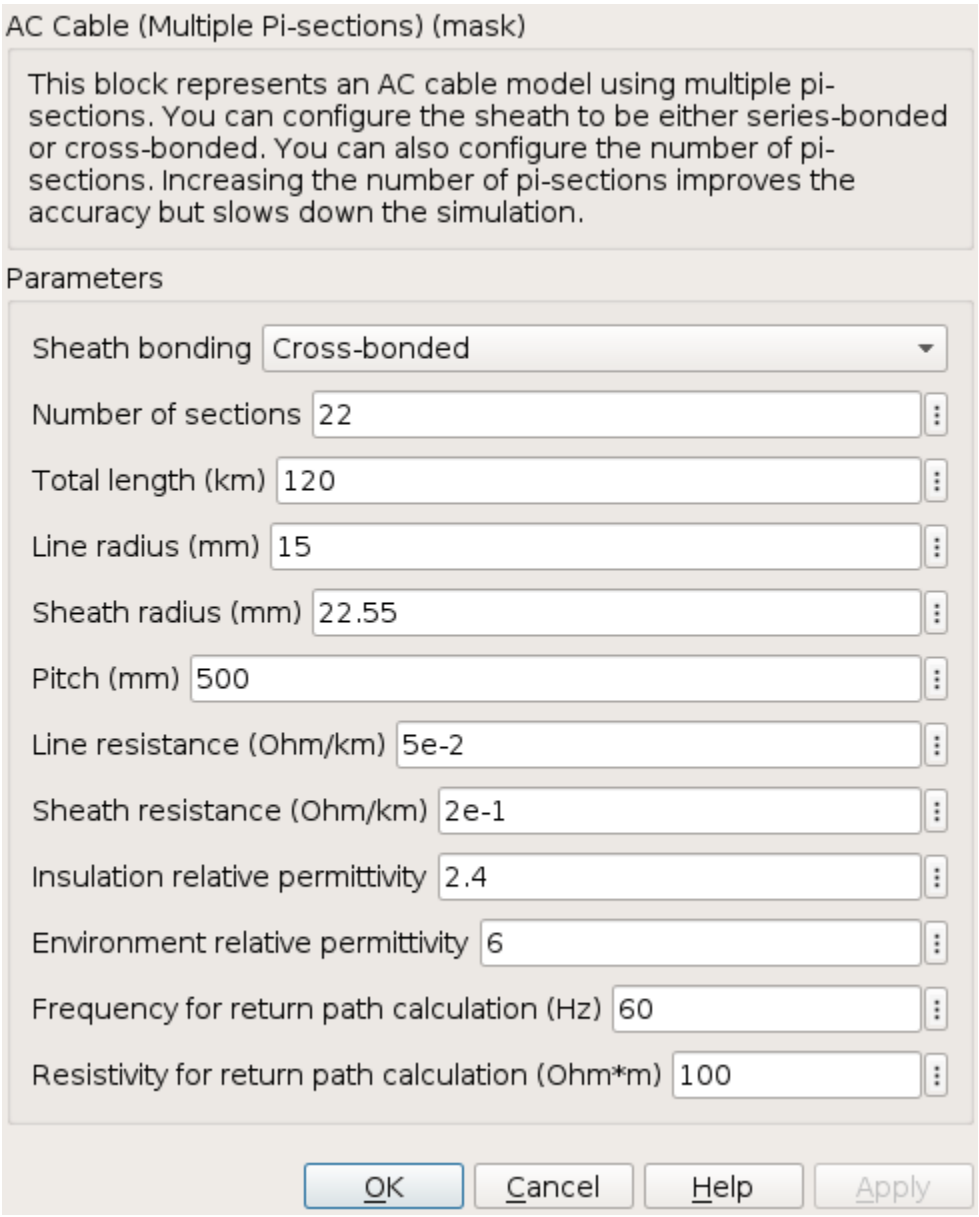
# AC Cable with Bonded Sheaths

This example shows a three-phase cable model comprised of multiple pi-sections. Each phase is enclosed in a conductive sheath. The conductive sheath is connected to ground at either end of the cable through a simple resistance. A high-voltage source provides power to an unbalanced resistive load through the power cable. You can configure the sheath to be either series-bonded or cross-bonded. You can also configure the number of pi-sections. Increasing the number of pi-sections improves the accuracy but slows down the simulation. To facilitate convergence, the voltage source includes an internal impedance.

**Model**

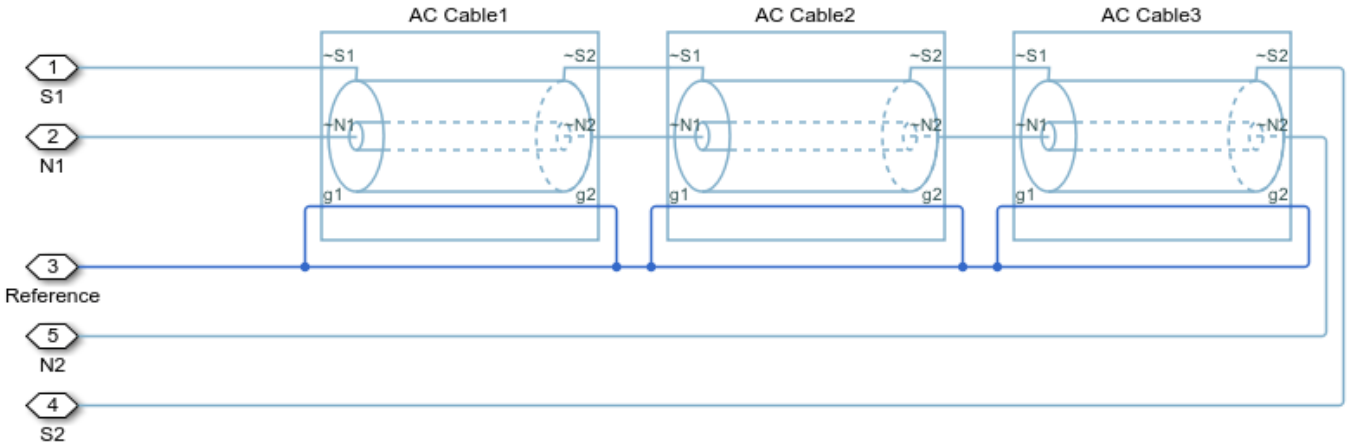


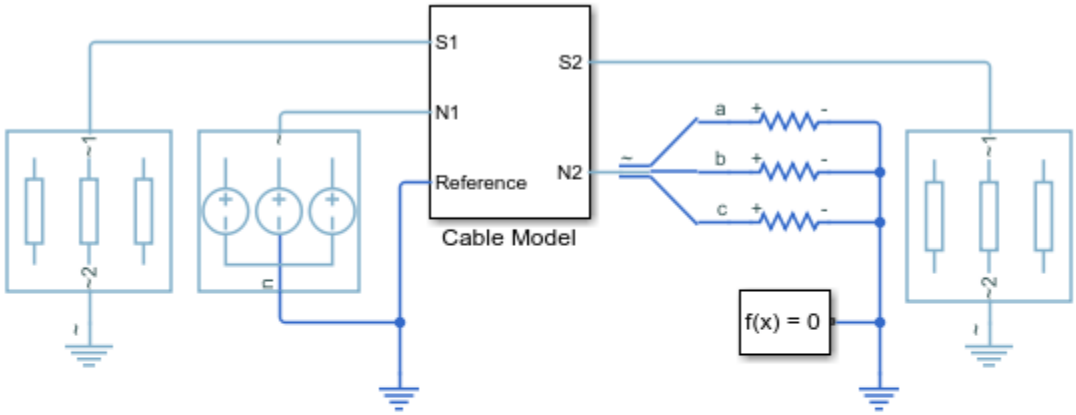
**Cable Model: Parameters**




**Cable Model: Series-bonded**

The figure shows the cable model comprised of three pi-sections with series-bonded sheaths.



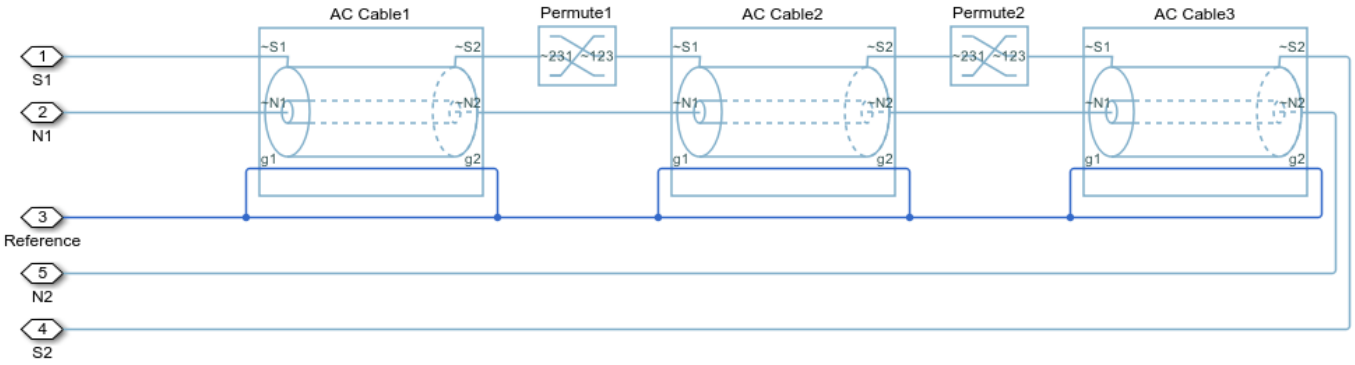

**AC Cable with Bonded Sheaths**

1. Plot sheath voltage per pi-section of cable model (see code)
2. Adjust sheath bonding: Series-bonding, Cross-bonding (open mask, see code)
3. Look under mask to see pi-section connections
4. Compare sheath bonding methods: simulate (see code), plot results (see code)
5. Explore simulation results using Simscape Results Explorer
6. Learn more about this example

Copyright 2017-2022 The MathWorks, Inc.

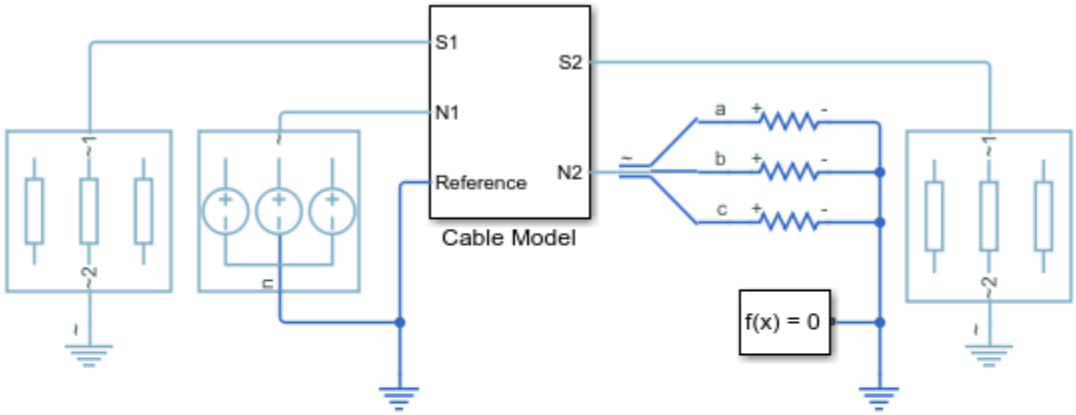
### Cable Model: Cross-bonded

The figure shows the cable model comprised of three pi-sections with cross-bonded sheaths. The sheath cross-bonding is implemented using Phase Permute blocks.




### Simulation Results from Simscape Logging

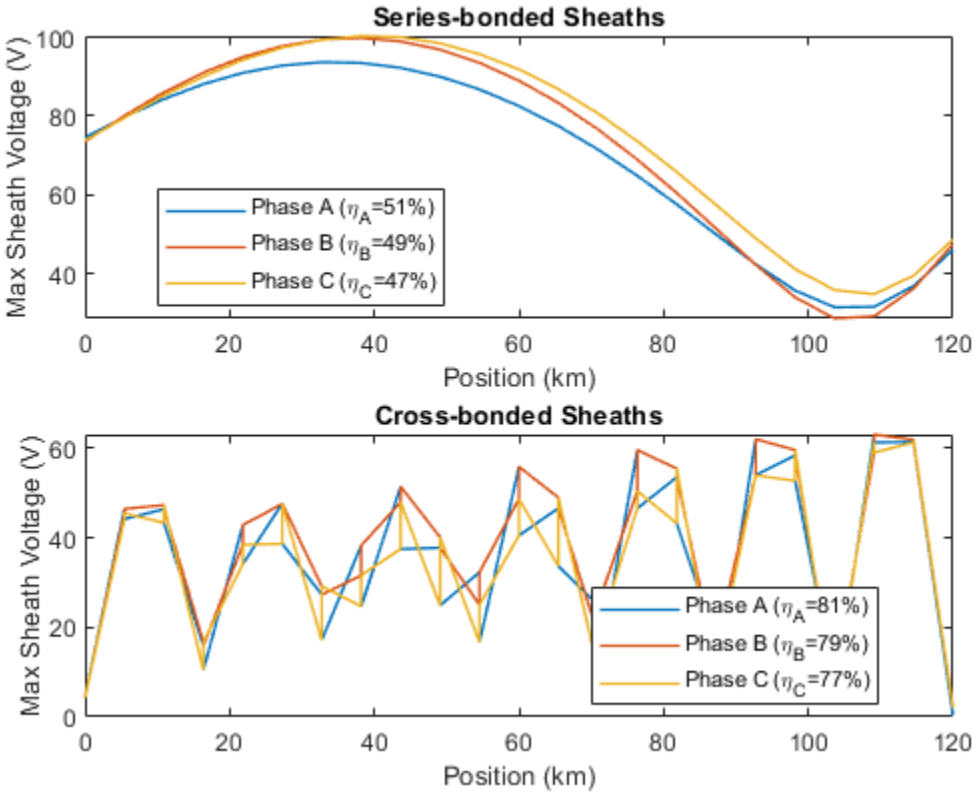
The plots show the behavior of the sheath bonding methods for a cable model comprised of 22 pi-sections. The efficiency with cross-bonded sheaths is higher than the efficiency with series-bonded sheaths.




**AC Cable with Bonded Sheaths**

1. Plot sheath voltage per pi-section of cable model (see code)
2. Adjust sheath bonding: Series-bonding, Cross-bonding (open mask, see code)
3. Look under mask to see pi-section connections
4. Compare sheath bonding methods: simulate (see code), plot results (see code)
5. Explore simulation results using Simscape Results Explorer
6. Learn more about this example

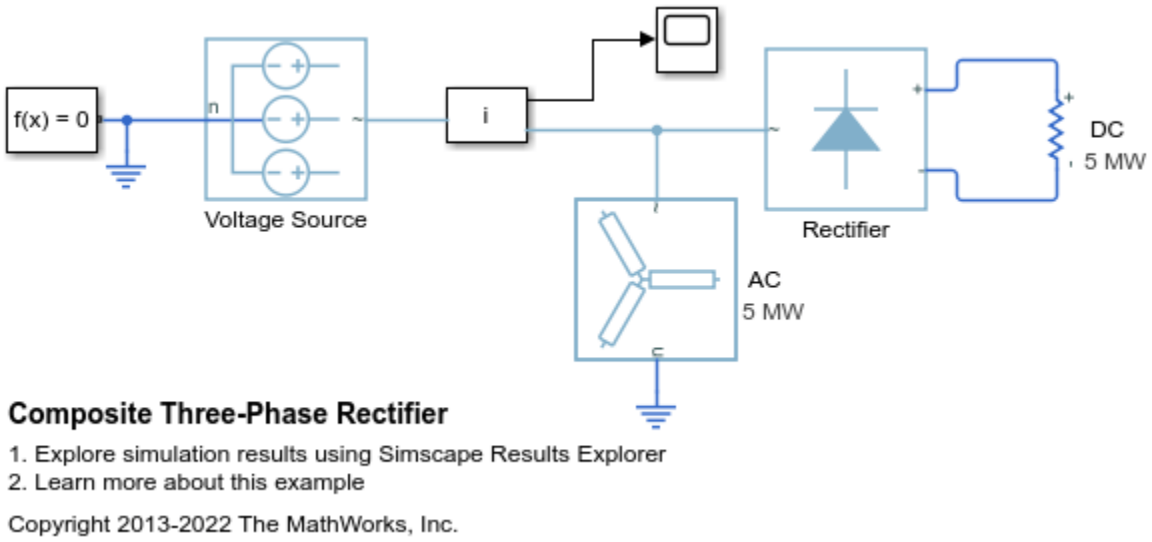
Copyright 2017-2022 The MathWorks, Inc.


Series-bonded Sheaths
Max Sheath Voltage (V)
100
80
60
40
0
20
40
60
80
100
120
Position (km)
Phase A ($\eta_A$=51%)
Phase B ($\eta_B$=49%)
Phase C ($\eta_C$=47%)
Cross-bonded Sheaths
Max Sheath Voltage (V)
60
40
20
0
0
20
40
60
80
100
120
Position (km)
Phase A ($\eta_A$=81%)
Phase B ($\eta_B$=79%)
Phase C ($\eta_C$=77%)

# Composite Three-Phase Rectifier

This is the base model for analysis workflow examples.

### Model




### Results from Real-Time Simulation

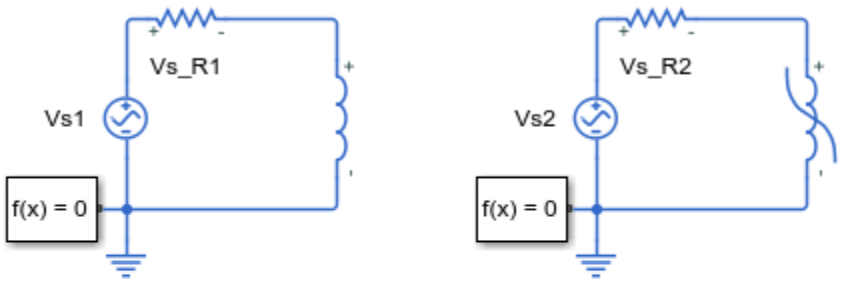
This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 50 microseconds.

# Custom Inductor (B-H Curve)

This example shows a comparison in behavior of a linear and nonlinear inductor. Starting with fundamental parameter values, the parameters for linear and nonlinear representations are derived. These parameters are then used in a Simscape™ model and the simulation outputs compared.

### Open Model

```
modelName = 'ee_custom_inductor';
open_system( modelName );
```




**Custom Inductor (B-H Curve)**

1. Plot comparison graphs (see code)
2. Modify model parameters
3. Explore simulation results using Simscape Results Explorer
4. Learn more about this example

Copyright 2012-2022 The MathWorks, Inc.

### Specification of Parameters

Fundamental parameter values used as the basis for subsequent calculations:

- Permeability of free space, $\mu_0, \mathrm{H/m}$
- Relative permeability of core, $\mu_r$
- Number of winding turns, $N_w$
- Effective magnetic core length, $l_e, \mathrm{m}$
- Effective magnetic core cross-sectional area, $A_e, \mathrm{m^2}$
- Core saturation begins, $B_{sat_{begin}}, \mathrm{T}$
- Core fully saturated, $B_{sat}, \mathrm{T}$

```
mu_0 = pi*4e-7;
mu_r = 3000;
Nw = 100;
le = 0.02;
Ae = 1e-05;
B_sat_begin = 0.75;
B_sat = 1.5;
```

### Calculate Magnetic Flux Density and Magnetic Field Strength Data

Where:

- Magnetic flux density, $B, \mathrm{T}$
- Magnetic field strength, $H, \mathrm{A/m}$

Linear representation:

- $B = \mu_0 \mu_r H$

Nonlinear representation (including coefficient, a):

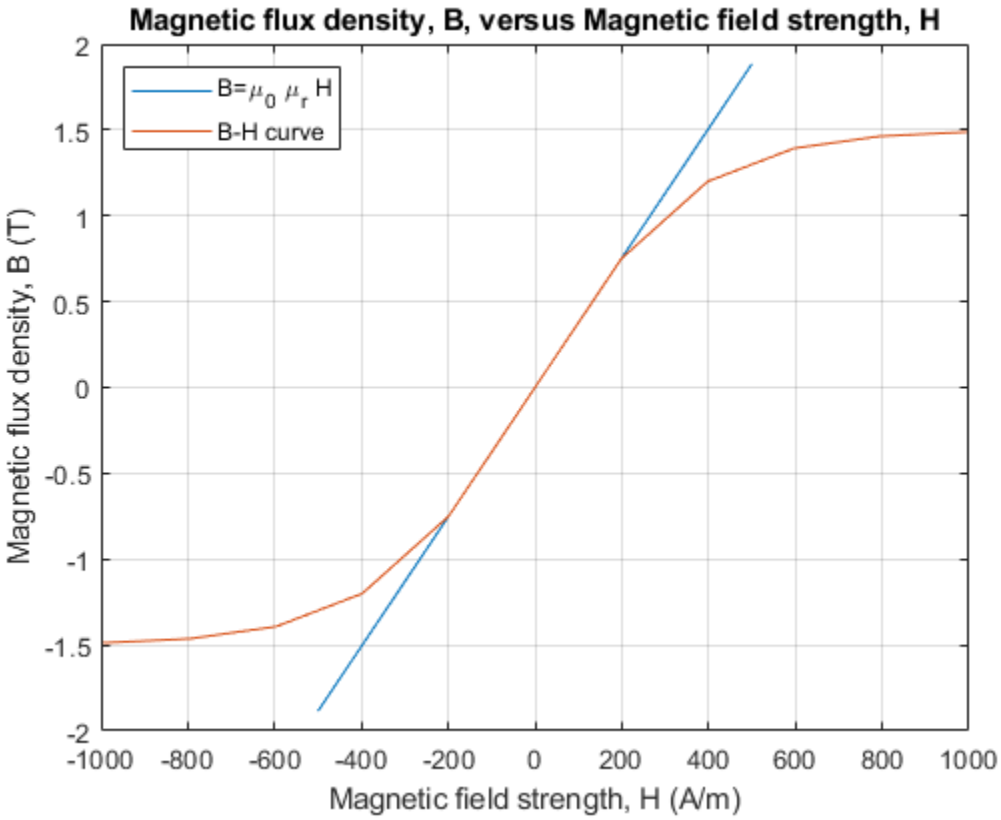
- $B = B_{sat} \tanh(a.H)$

```
% Use linear representation to find value of H corresponding to B_sat_begin
H_sat_begin = B_sat_begin/(mu_0*mu_r);
% Rearrange nonlinear representation to calculate coefficient, a
a = atanh( B_sat_begin/B_sat )/H_sat_begin;

% Linear representation
H_linear = [-500 500];
B_linear = mu_0*mu_r*H_linear;

% Nonlinear representation
H_nonlinear = -5*H_sat_begin:H_sat_begin:5*H_sat_begin;
B_nonlinear = B_sat*tanh(a*H_nonlinear);
```

### Display Magnetic Flux Density Versus Magnetic Field Strength

The linear and nonlinear representations can be overlaid.

```
if ~exist('h1_ee_custom_inductor', 'var') || ...
        ~isgraphics(h1_ee_custom_inductor, 'figure')
    h1_ee_custom_inductor = figure('Name', 'ee_custom_inductor');
end
figure(h1_ee_custom_inductor)
clf(h1_ee_custom_inductor)

plot( H_linear, B_linear, H_nonlinear, B_nonlinear );
grid( 'on' );
title( 'Magnetic flux density, B, versus Magnetic field strength, H' );
xlabel( 'Magnetic field strength, H (A/m)' );
ylabel( 'Magnetic flux density, B (T)' );
legend( 'B=\mu_0 \mu_r H', 'B-H curve', 'Location', 'NorthWest' )
```




### Calculate Magnetic Flux and Current Data

Where:

- Magnetic flux, $\phi, \mathrm{Wb}$
- Current, $I, \mathrm{A}$

Linear representation:

- $L = \mu_0 \mu_r A_e N_w^2 / l_e$
- $\phi = IL/N_w$

Nonlinear representation:

- $I = Hl_e/N_w$
- $\phi = BA_e$

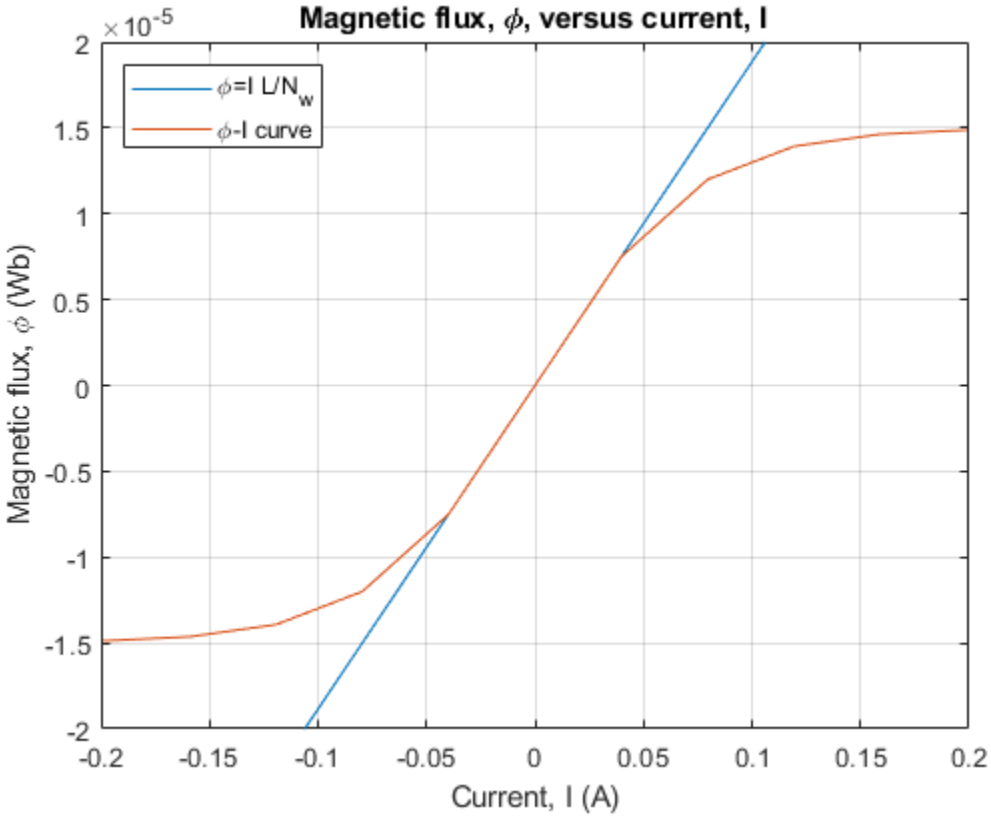
```
% Linear inductance
L_linear = mu_0*mu_r*Ae*Nw^2/le;

% Linear representation
I_linear = [-1.5 1.5];
phi_linear = I_linear.*L_linear/Nw;
```

```
% Nonlinear representation
I_nonlinear = le.*H_nonlinear./Nw;
phi_nonlinear = B_nonlinear.*Ae;
```

### Display Magnetic Flux Versus Current

The linear and nonlinear representations can be overlaid.

```
if ~exist('h2_ee_custom_inductor', 'var') || ...
        ~isgraphics(h2_ee_custom_inductor, 'figure')
    h2_ee_custom_inductor = figure('Name', 'ee_custom_inductor');
end
figure(h2_ee_custom_inductor)
clf(h2_ee_custom_inductor)

plot( I_linear, phi_linear, I_nonlinear, phi_nonlinear );
grid( 'on' );
title( 'Magnetic flux, \phi, versus current, I' );
xlabel( 'Current, I (A)' );
ylabel( 'Magnetic flux, \phi (Wb)' );
xlim([-0.2 0.2]);
ylim([-2e-5 2e-5]);
legend( '\phi=I L/N_w', '\phi-I curve', 'Location', 'NorthWest' );
```



### Use Parameters in Simscape Model

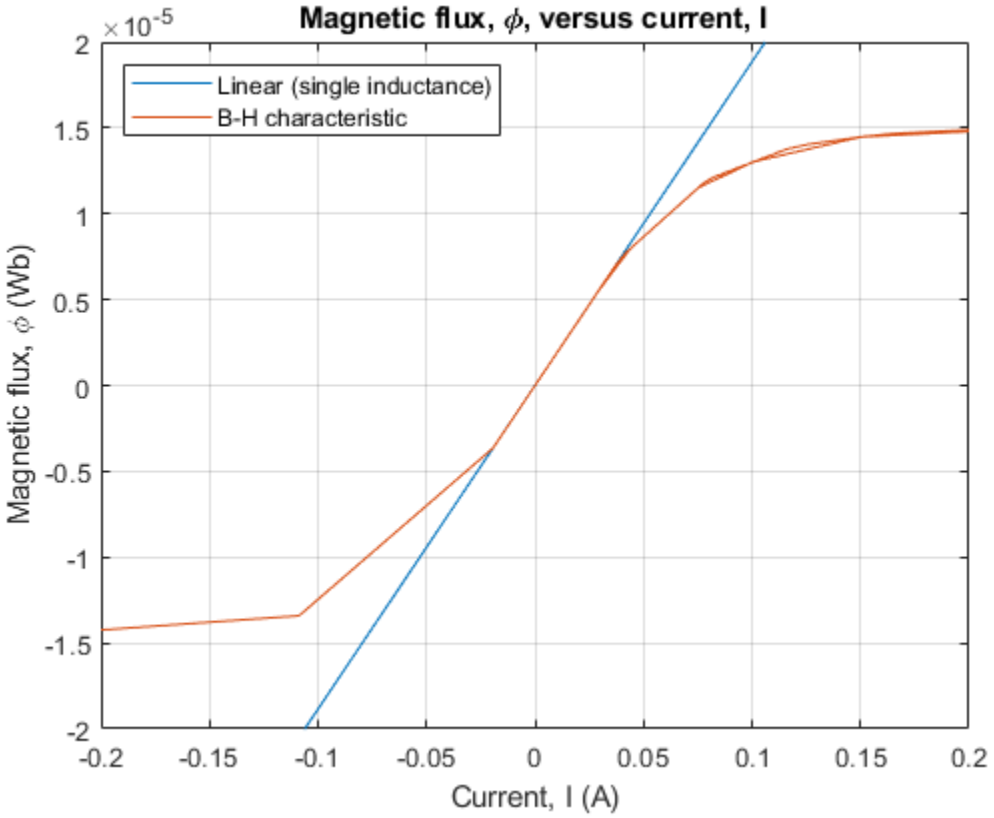
The parameters calculated can now be used in a Simscape model. Once simulated, the model is set to create a Simscape logging variable, simlog_ee_custom_inductor.

```
sim( modelName );
```

### Conclusion

The state variable for both representations is magnetic flux, $\phi$. Current, I, and magnetic flux, $\phi$, can be obtained from the Simscape logging variable, simlog_ee_custom_inductor, for each representation. Overlaying the simulation results from the representations permits direct comparison.

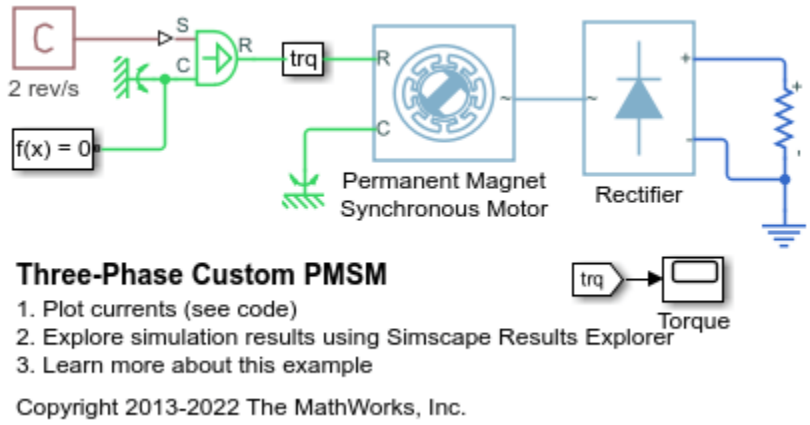
```
if ~exist('h3_ee_custom_inductor', 'var') || ...
        ~isgraphics(h3_ee_custom_inductor, 'figure')
    h3_ee_custom_inductor = figure('Name', 'ee_custom_inductor');
end
figure(h3_ee_custom_inductor)
clf(h3_ee_custom_inductor)

plot( ...
    simlog_ee_custom_inductor.Inductor_linear_magnetic_flux.i.series.values,...
    simlog_ee_custom_inductor.Inductor_linear_magnetic_flux.phi.series.values,...
    simlog_ee_custom_inductor.Inductor_B_H_curve.i.series.values,...
    simlog_ee_custom_inductor.Inductor_B_H_curve.phi.series.values);
grid( 'on' );
title( 'Magnetic flux, \phi, versus current, I' );
xlabel( 'Current, I (A)' );
ylabel( 'Magnetic flux, \phi (Wb)' );
xlim([-0.2 0.2]);
ylim([-2e-5 2e-5]);
legend( 'Linear (single inductance)', 'B-H characteristic', 'Location', 'NorthWest' );
```


Magnetic flux, $\phi$, versus current, I
$\times 10^{-5}$
Linear (single inductance)
B-H characteristic
Magnetic flux, $\phi$ (Wb)
Current, I (A)
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-0.2
-0.15
-0.1
-0.05
0
0.05
0.1
0.15
0.2

# Three-Phase Custom PMSM

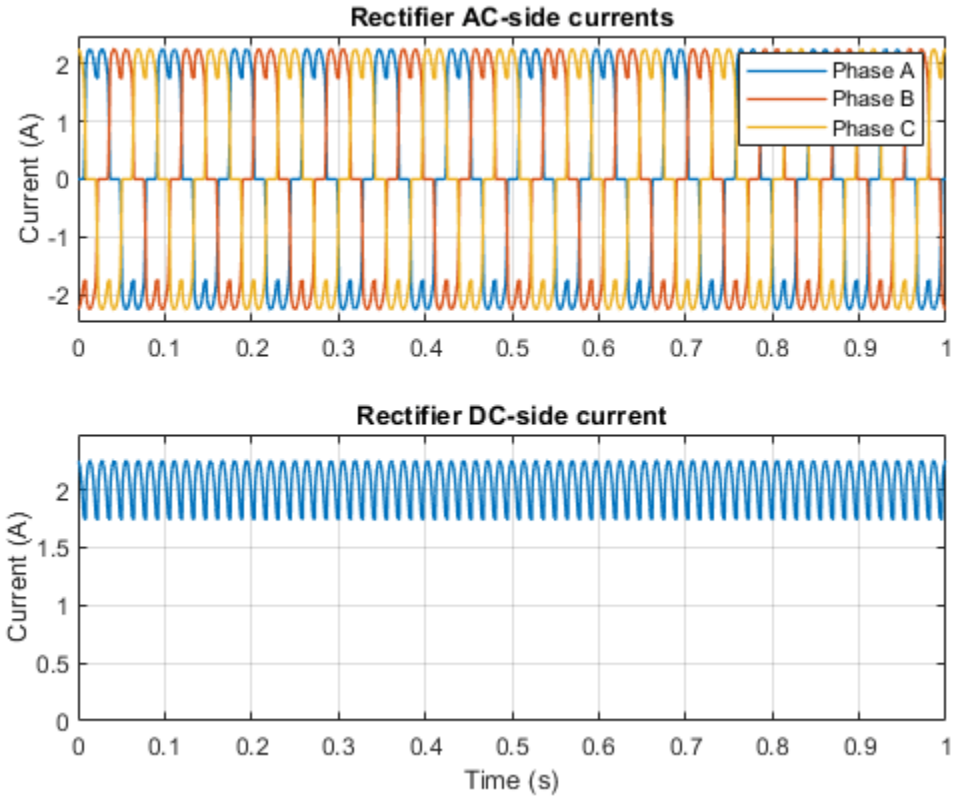
This example shows a custom Simscape™ implementation of a Permanent Magnet Synchronous Machine (PMSM). To view the PMSM source code, double-click on the motor block and then click on the hyperlink 'Source code'. This test circuit shows the PMSM being used as a generator, the rectifier block converting the induced AC back EMFs to a DC voltage which is in turn applied to a resistive load.

### Model




### Simulation Results from Simscape Logging

The first subplot shows the AC currents generated by the Permanent Magnet Synchronous Machine (PMSM). The second subplot shows the DC current through the resistive load.




### Results from Real-Time Simulation

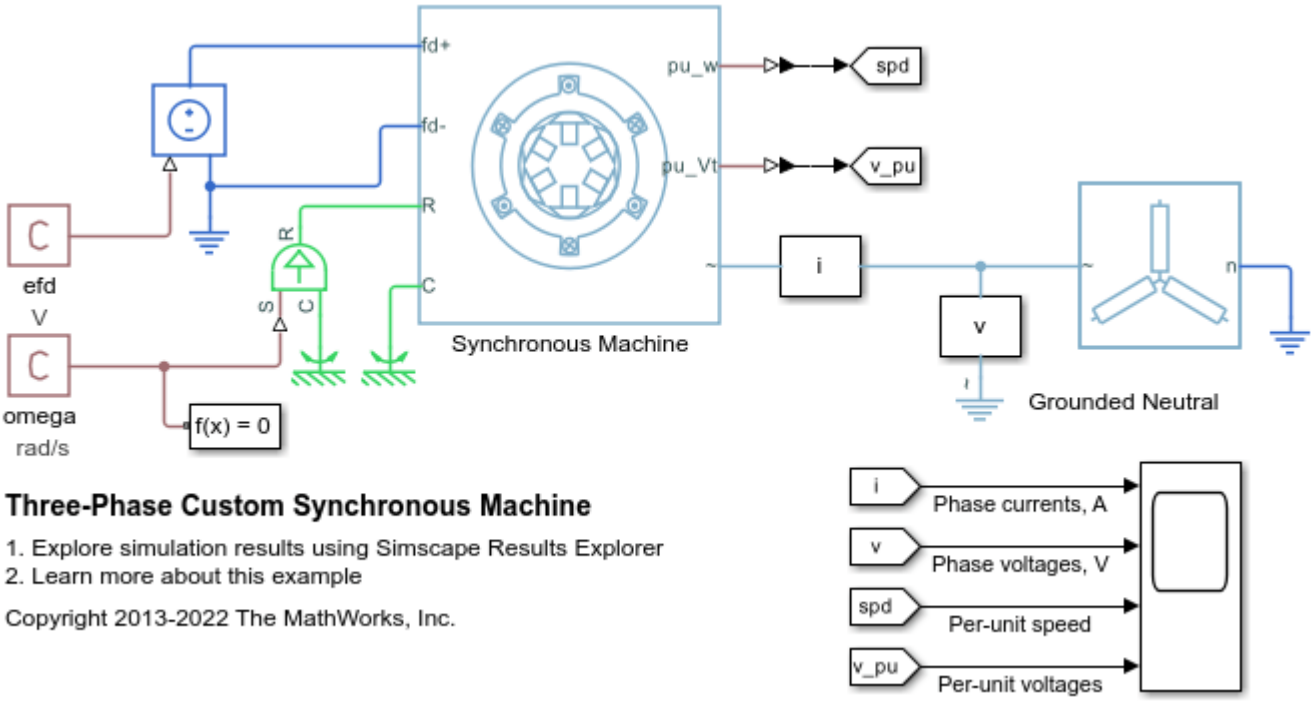
This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 20 microseconds.

# Three-Phase Custom Synchronous Machine

This example shows a custom Simscape™ implementation of a Synchronous Machine (SM). To view the SM source code, double-click on the Synchronous Machine block and then click on the hyperlink 'Source code'.

This test circuit shows the SM being used as a generator, the field has constant voltage applied, and the rotor is turned at a constant speed. The overall model has been initialized to start in periodic steady state to supply the 500MW load.

### Model




### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 70 microseconds.

# Three-Phase Custom Simplified Synchronous Machine

This example shows how simplify the custom Simscape™ synchronous machine model.

### Detailed Machine Equations

Detailed per-unit stator voltage equations include the effect of rate of change of magnetic flux linkage, and the effect of speed variation:

$$e_d = \frac{1}{\omega_{base}} \frac{\mathrm{d}\Psi_d}{\mathrm{d}t} - \Psi_q \omega_r - R_a i_d$$

$$e_q = \frac{1}{\omega_{base}} \frac{\mathrm{d}\Psi_q}{\mathrm{d}t} + \Psi_d \omega_r - R_a i_q$$

### Simplified Machine Equations

Large-scale studies are simulations that include more than 10 generators or motors. To reduce computational requirements of large-scale studies, simplifications are necessary. Simplifications for large-scale studies, include neglecting the following:

- The transformer voltage terms, $\frac{\mathrm{d}\Psi_d}{\mathrm{d}t}$ and $\frac{\mathrm{d}\Psi_q}{\mathrm{d}t}$
- The effect of speed variations. $\omega_r$ is set to 1.

These simplifications yield the following per-unit stator voltage equations:

$$e_d = -\Psi_q - R_a i_d$$

$$e_q = \Psi_d - R_a i_q$$

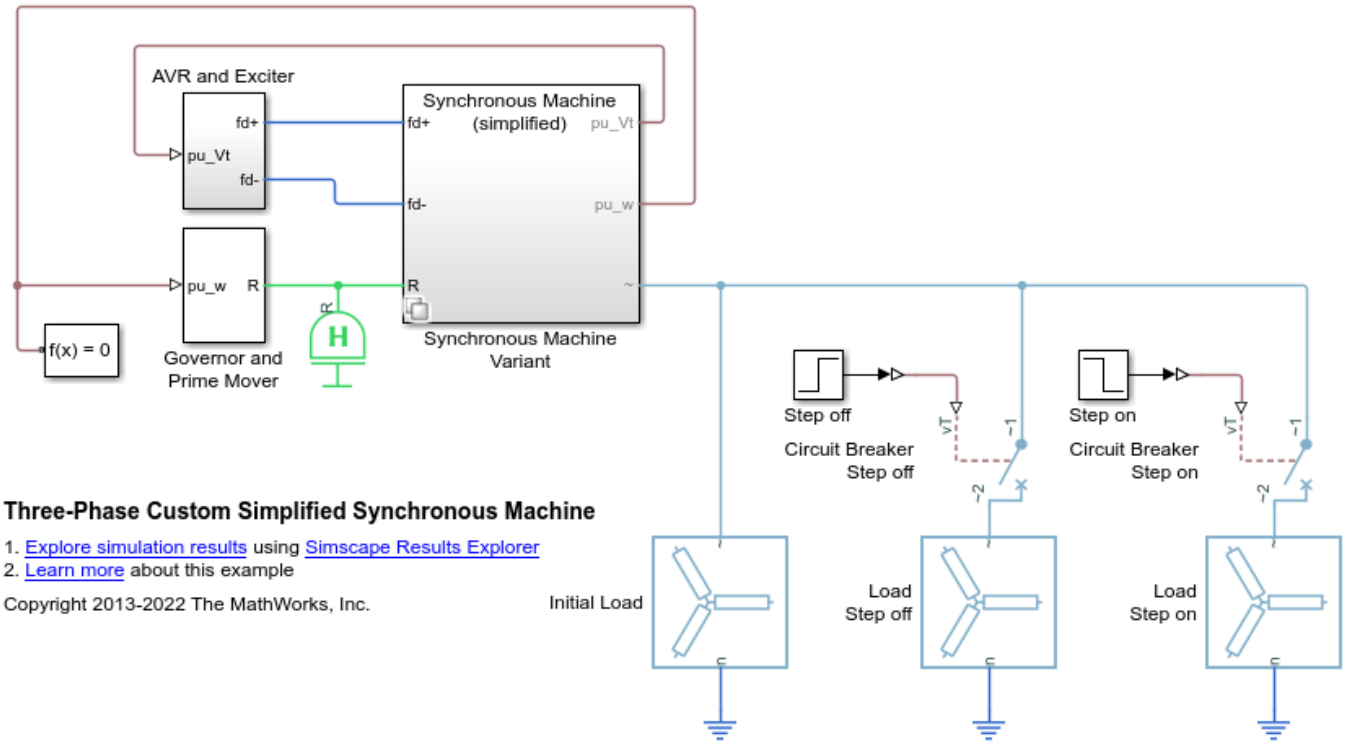
### Simscape Component Comparison

You can use the Comparison Tool to see the differences between the detailed and simplified Simscape components. Type visdiff( 'ThreePhaseExamples.sm', 'ThreePhaseExamples.sm_simplified' ) at the MATLAB® command line to open the Comparison Tool to see the detailed differences on lines 176 and 177 of the files. The detailed Synchronous Machine component is on the left side, and the simplified Synchronous Machine component is on the right side.

### Open Model

Open the model.

```
open_system( 'ee_custom_sm_simplified' );
```




### Define Initial Conditions

Specify values of initial conditions.

```
pu_psid0 = 0.7850;
pu_psiq0 = -0.6216;
pu_psifd0 = 0.9553;
pu_psi1d0 = 0.8269;
pu_psi1q0 = -0.5686;
pu_psi2q0 = -0.5686;
Efd0 = 1.2803;
torque0 = 0.4610;
```

### Define Variants

The example model, ee_custom_sm_simplified, has been configured using Simulink® Variants. To execute the detailed model, set the variable isSimplified to false. To execute the simplified model, set the variable isSimplified to true.

### Execute Simulations and Collect Results

Simulate detailed model.

```
isSimplified = false; %#ok<NASGU>
sim( 'ee_custom_sm_simplified' );
t1 = simlog_ee_custom_sm_simplified.Synchronous_Machine_Variant.Synchronous_Machine.Synchronous_M
v1 = simlog_ee_custom_sm_simplified.Synchronous_Machine_Variant.Synchronous_Machine.Synchronous_M
w1 = simlog_ee_custom_sm_simplified.Synchronous_Machine_Variant.Synchronous_Machine.Synchronous_M
```

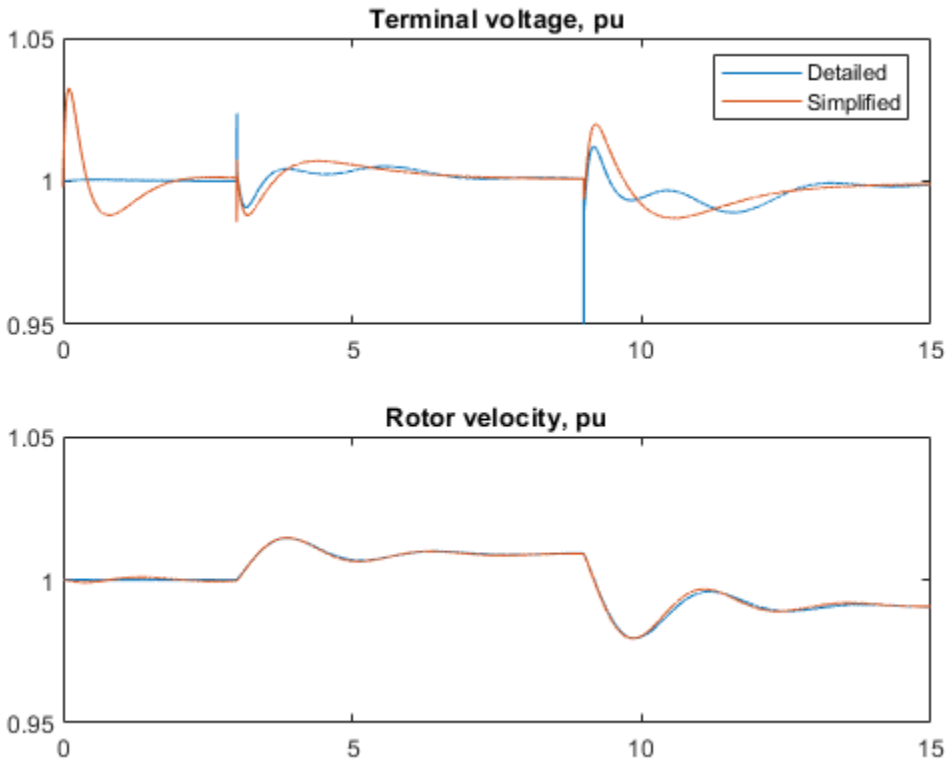
Simulate simplified model.

```
isSimplified = true;
sim( 'ee_custom_sm_simplified' );
t2 = simlog_ee_custom_sm_simplified.Synchronous_Machine_Variant.Synchronous_Machine_simplified.Sy
v2 = simlog_ee_custom_sm_simplified.Synchronous_Machine_Variant.Synchronous_Machine_simplified.Sy
w2 = simlog_ee_custom_sm_simplified.Synchronous_Machine_Variant.Synchronous_Machine_simplified.Sy
```

### Plot Results

The voltage and rotor velocity traces can be overlaid to directly compare differences.

```
figure;
subplot(2,1,1);
plot( t1, v1, t2, v2 );
ylim( [ 0.95 1.05 ] );
legend( 'Detailed', 'Simplified' );
title( 'Terminal voltage, pu' );
subplot(2,1,2);
plot( t1, w1, t2, w2 );
ylim( [ 0.95 1.05 ] );
title( 'Rotor velocity, pu' );
```




### Conclusion

This example shows:

- How to simplify the custom synchronous machine model
- A comparison of Simscape component source code

- A comparison of model simulation outputs

### Results from Real-Time Simulation

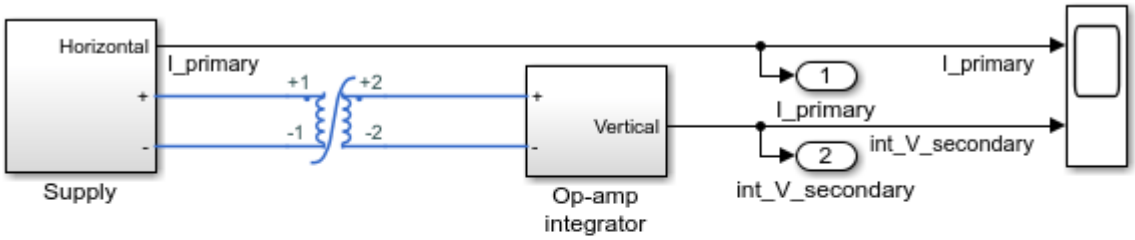
This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 100 microseconds.

# Custom Transformer (B-H Curve)

This example shows calculation and confirmation of a nonlinear transformer core magnetization characteristic. Starting with fundamental parameter values, the core characteristic is derived. This is then used in a Simscape™ model of an example test circuit which can be used to plot the core magnetization characteristic on an oscilloscope. Model outputs are then compared to the known values.

### Open Model

```
modelName = 'ee_custom_transformer';
open_system( modelName );
```




**Custom Transformer (B-H Curve)**

1. Plot comparison graph (see code)
2. Modify model parameters
3. Explore simulation results using Simscape Results Explorer
4. Learn more about this example

Copyright 2012-2022 The MathWorks, Inc.

### Specification of Parameters

Fundamental parameter values used as the basis for subsequent calculations:

- Permeability of free space, $\mu_0, \mathrm{H/m}$
- Relative permeability of core, $\mu_r$
- Number of primary turns, $N_1$
- Number of secondary turns, $N_2$
- Effective magnetic core length, $l_e, \mathrm{m}$
- Effective magnetic core cross-sectional area, $A_e, \mathrm{m^2}$
- Core saturation begins, $B_{sat_{begin}}, \mathrm{T}$
- Core fully saturated, $B_{sat}, \mathrm{T}$

```
mu_0 = pi*4e-7;
mu_r = 3000;
N1 = 100;
N2 = 200;
le = 0.1;
Ae = 1e-4;
```

```
B_sat_begin = 0.6;
B_sat = 1.2;
```

### Calculate Magnetic Flux Density Versus Magnetic Field Strength Characteristic

Where:

- Magnetic flux density, $B, \mathrm{T}$
- Magnetic field strength, $H, \mathrm{A/m}$

Linear representation:

- $B = \mu_0 \mu_r H$

Nonlinear representation (including coefficient, a):

- $B = B_{sat} \tanh(a.H)$

```
% Use linear representation to find value of H corresponding to B_sat_begin
H_sat_begin = B_sat_begin/(mu_0*mu_r);
% Rearrange nonlinear representation to calculate coefficient, a
a = atanh( B_sat_begin/B_sat )/H_sat_begin;

% Nonlinear representation
H_nonlinear = -750:25:750;
B_nonlinear = B_sat*tanh(a*H_nonlinear);
```

### Use Parameters in Simscape Model

The parameters calculated can now be used in a Simscape model. Once simulated, the model is set to output a Simscape logging variable, simlog_ee_custom_transformer, and some signals using output ports, yout. Circuit parameters are:

- Voltage source magnitude, $V_s = 10\mathrm{V}$
- Voltage source frequency, $Freq_{Hz} = 60\mathrm{Hz}$
- Voltage source resistance, $R_{Vs} = 10\Omega$
- Operational amplifier input resistance, $R_1 = 1\mathrm{k}\Omega$
- Operational amplifier feedback resistance, $R_2 = 1\mathrm{M}\Omega$
- Operational amplifier feedback capacitance, $C_2 = 1\mu\mathrm{F}$

```
% Circuit parameters
Vs = 10;
Freq_Hz = 60;
R_Vs = 10;
R_1 = 1e3;
R_2 = 1e6;
C_2 = 1e-6;

% Simulate model
simOut = sim( modelName );
yout = get(simOut, 'yout');
simlog = get(simOut, 'simlog_ee_custom_transformer');
```

```
% Collect internal Simscape logging data for comparison
I_simscape = simlog.Transformer_B_H_curve.L_m.i.series.values;
phi_simscape = simlog.Transformer_B_H_curve.L_m.phi.series.values;

% Collect model output data for comparison (as used for oscilloscope)
I_primary = yout(:,1);
int_V_secondary = yout(:,2);
```

### Calculations on Logging and Output Data

The data needs to be processed to provide the Magnetic field strength and Magnetic flux density data for comparison. Where:

- Magnetomotive force, $F_m, \mathrm{A}$
- Magnetic flux, $\phi, \mathrm{Wb}$
- Operational amplifier input voltage, $V_{in}, \mathrm{V}$
- Operational amplifier output voltage, $V_{out}, \mathrm{V}$

Equations to be used are as follows:

- $F_m = I.N_1$
- $H = F_m/l_e$
- $B = \phi/A_e$
- $V_{out} = \frac{1}{R_1 C_2} \int V_{in}.dt + c$
- $\phi = \frac{1}{N_2} \int v.dt$

```
% Internal logging data
H_simscape = I_simscape.*N1./le;
B_simscape = phi_simscape./Ae;

% Oscilloscope scaling and model output data
H_measured = I_primary.*N1./le;
phi_measured = (int_V_secondary.*R_1.*C_2)./N2;
B_measured = phi_measured./Ae;
```

### Conclusion

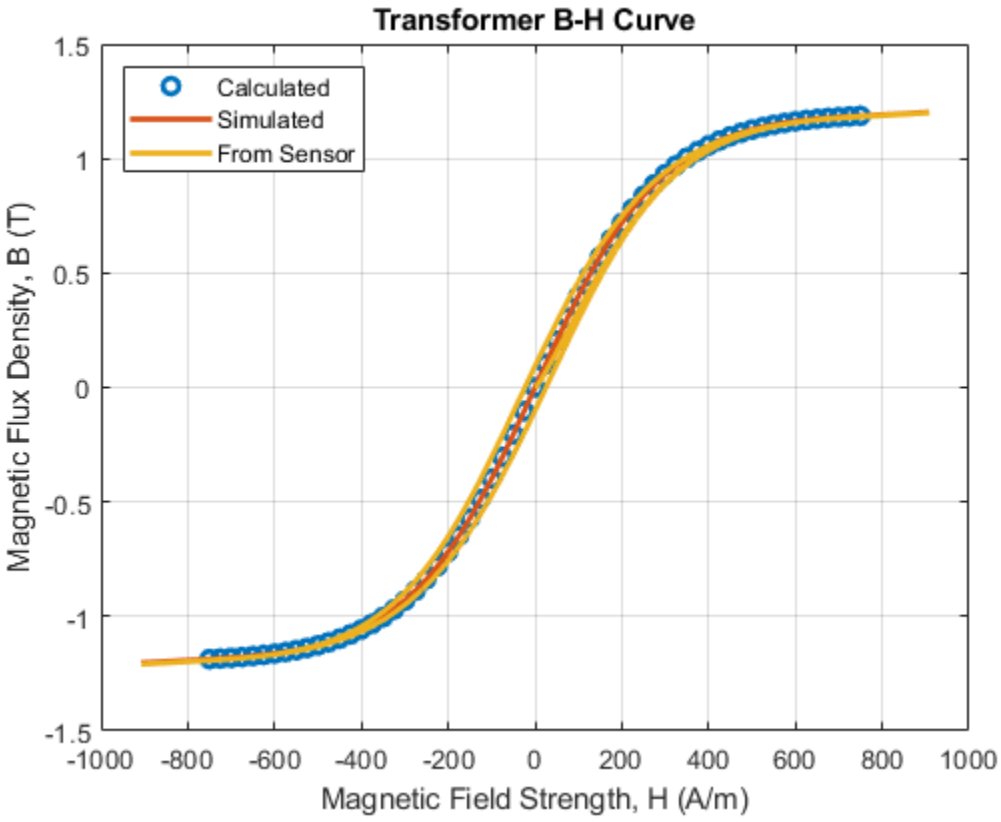
The three characteristics can now be overlaid:

- Defined characteristic: calculated from fundamental parameters
- Characteristic from logging: calculated from internal Simscape logging data
- Characteristic from measurement: obtained by measurement and calculation using electronic test circuit

Due to leakage and parasitic parameters, the characteristic obtained from the electronic test circuit differs to the defined characteristic. However, the test circuit and its parameterization is shown to find the characteristic for the given transformer within suitable tolerances.

```
if ~exist('h1_ee_custom_transformer', 'var') || ...
        ~isgraphics(h1_ee_custom_transformer, 'figure')
    h1_ee_custom_transformer = figure('Name', 'ee_custom_transformer');
```

```
end
figure(h1_ee_custom_transformer)
clf(h1_ee_custom_transformer)

plot( ...
    H_nonlinear,...
    B_nonlinear,...
    'o',...
    H_simscape,...
    B_simscape,...
    H_measured,...
    B_measured,...
    '-',...
    'LineWidth',2);
grid( 'on' );
title( 'Transformer B-H Curve' );
xlabel( 'Magnetic Field Strength, H (A/m)' );
ylabel( 'Magnetic Flux Density, B (T)' );
legend( 'Calculated', 'Simulated',...
    'From Sensor', 'Location', 'NorthWest' );
```




### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real-time target machine with an Intel® 3.5 GHz i7 multi-core CPU. This model can run in real time with a step size of 50 microseconds.

# Three-Phase Custom Transforms

This example shows a comparison of three-phase transforms.

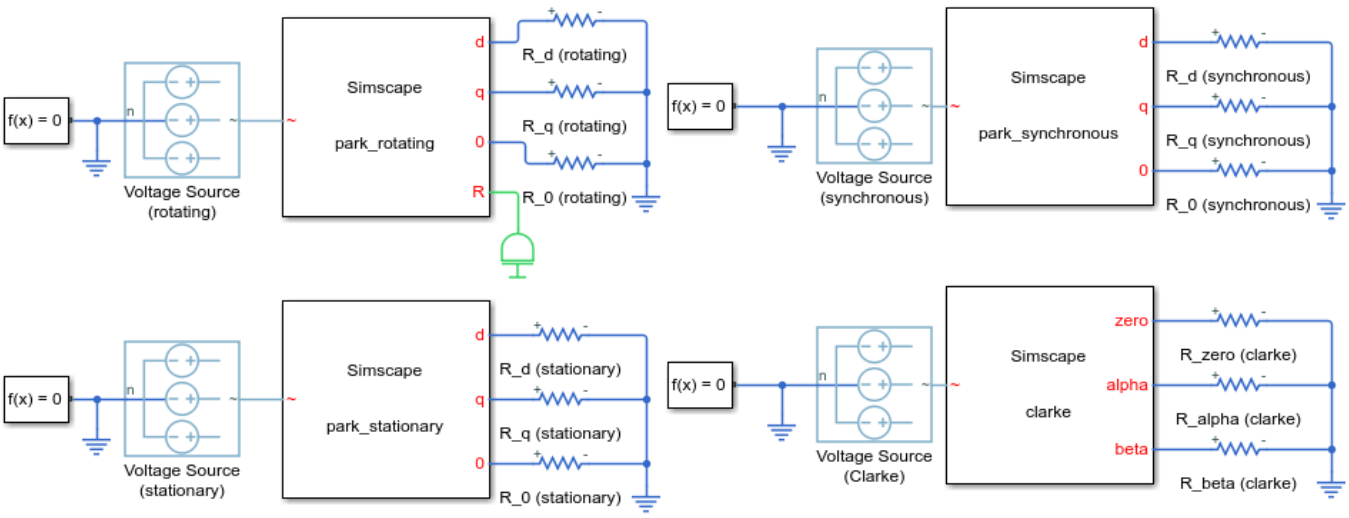
There are four transforms:

- Park transform with rotating reference frame
- Park transform with stationary reference frame
- Park transform with synchronous reference frame
- Clarke transform

For more information on each, double-click the block of interest and read the block mask.

### Open Model and Set Circuit Parameters

```
open_system('ee_custom_transforms');
set_param(find_system('ee_custom_transforms','FindAll', 'on','type','annotation','Tag','ModelFea
```




**Three-Phase Custom Transforms**

1. Explore simulation results using Simscape Results Explorer
2. Learn more about this example

Copyright 2013-2022 The MathWorks, Inc.

### View Results

To view full results, run the simulation, and enter at the MATLAB® command prompt:

```
sscexplore(ee_custom_transforms_simlog)
```

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real-time target machine with an Intel® 3.5 GHz i7 multi?core CPU. This model can run in real time with a step size of 50 microseconds.

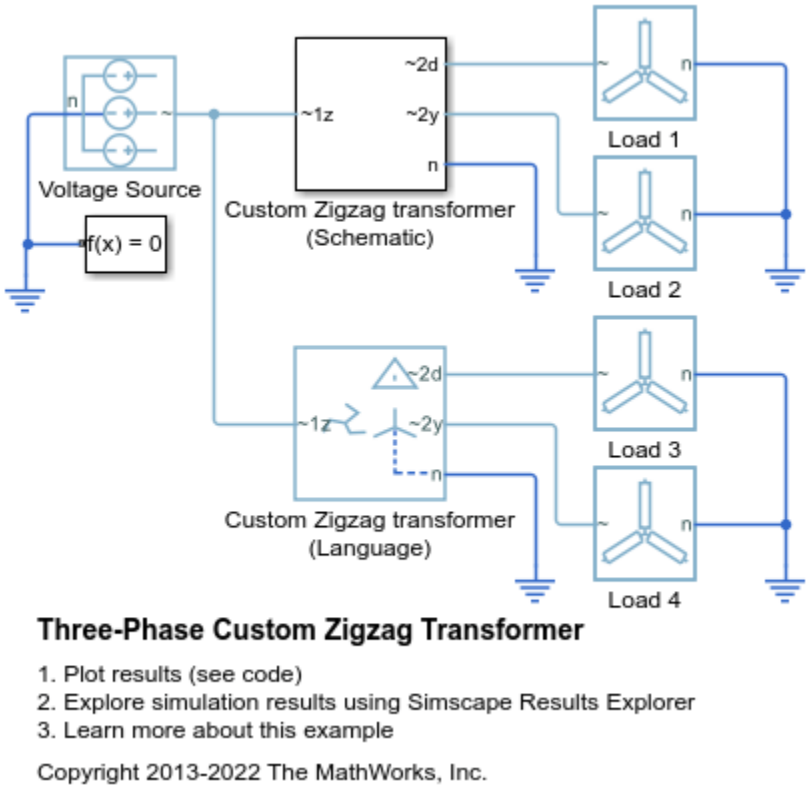
# Three-Phase Custom Zigzag Transformer

This example shows two different ways to create a custom transformer component. The first uses built-in library blocks to lay out the magnetic circuit as a masked Simulink® subsystem. The second builds a custom Simscape™ component using the Simscape language.

To view the Simscape language source code, double-click on the Custom Zigzag transformer (Language) block and then click on the hyperlink 'Source code'.

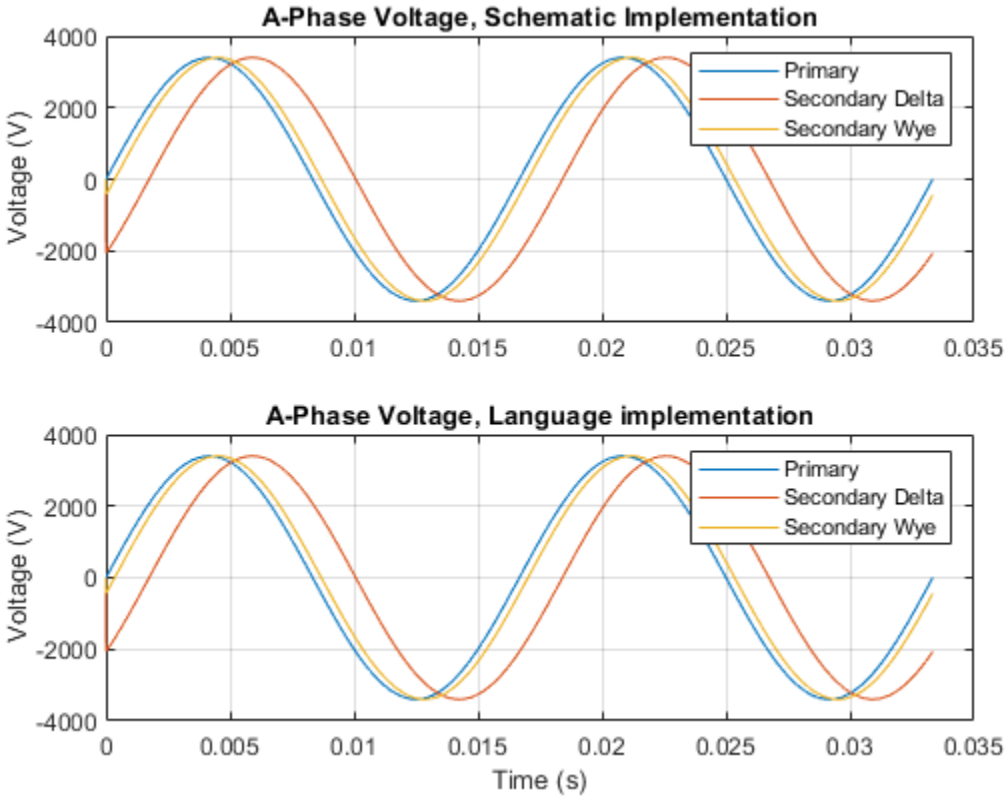
The transformer is implemented with a zigzag primary, 1 o'clock delta secondary and wye secondary winding. The phase offset between the zigzag and wye windings can be specified via a block mask parameter.

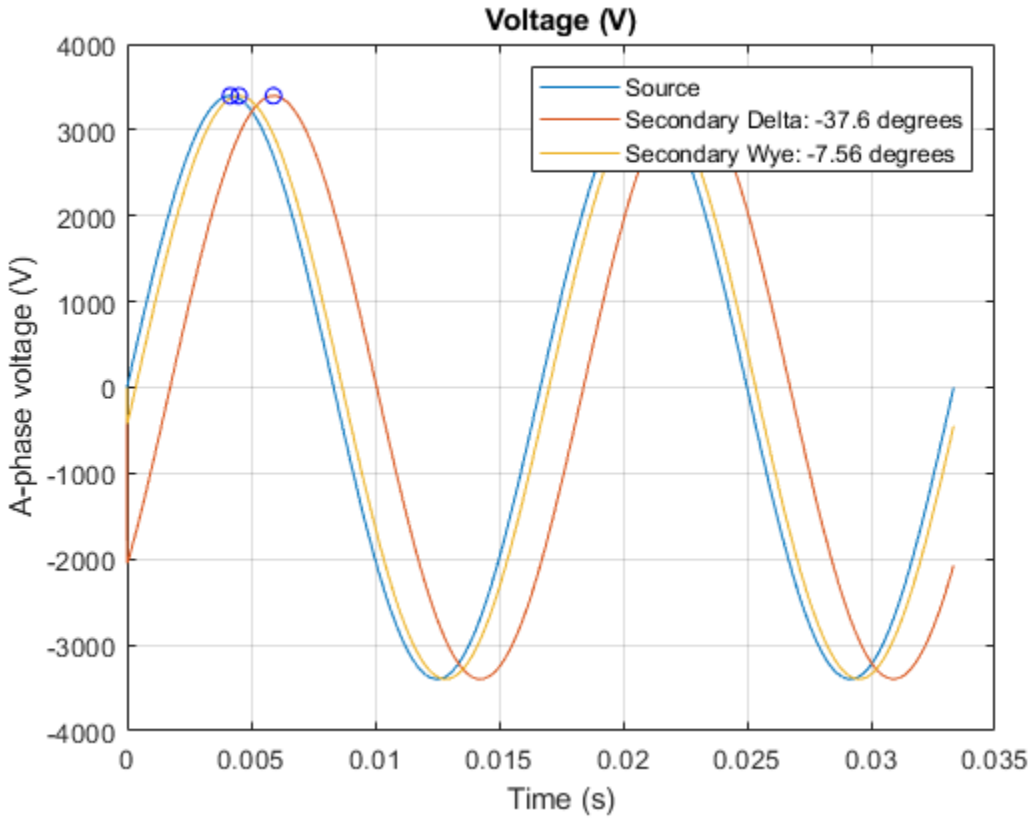
### Model




### Simulation Results from Simscape Logging

The plots below show the results for two different ways to create a custom transformer component. One using built-in library blocks to lay out the magnetic circuit as a masked Simulink® subsystem and other creating a Simscape™ component in Simscape language.

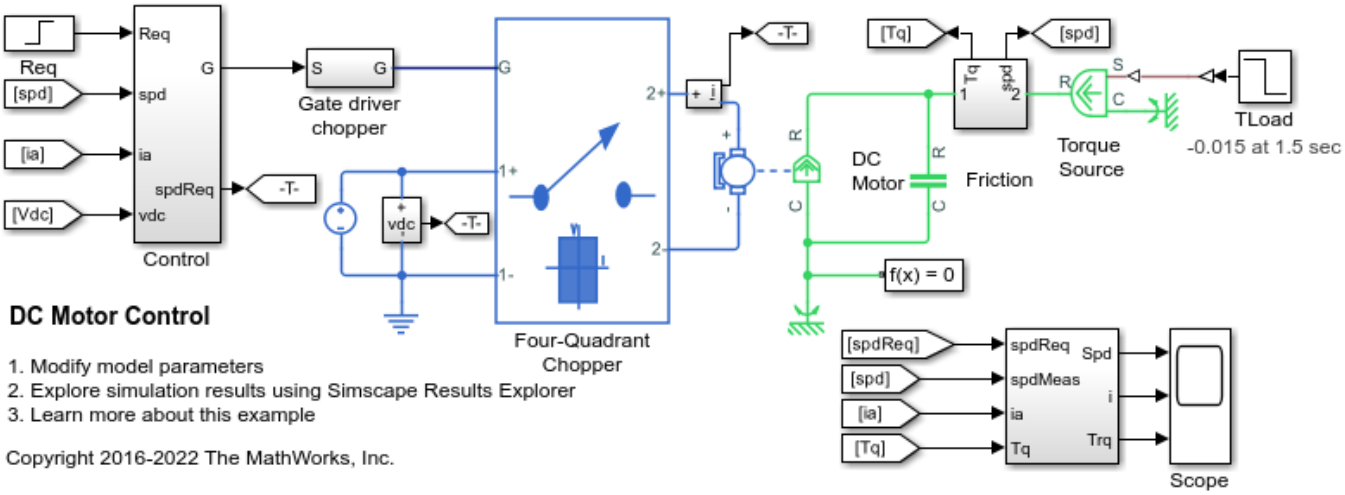

A-Phase Voltage, Schematic Implementation
Voltage (V)
4000
2000
0
-2000
-4000
0
0.005
0.01
0.015
0.02
0.025
0.03
0.035
Primary
Secondary Delta
Secondary Wye
A-Phase Voltage, Language implementation
Voltage (V)
Time (s)
Primary
Secondary Delta
Secondary Wye


Voltage (V)
Source
Secondary Delta: -37.6 degrees
Secondary Wye: -7.56 degrees
A-phase voltage (V)
Time (s)
4000
3000
2000
1000
0
-1000
-2000
-3000
-4000
0
0.005
0.01
0.015
0.02
0.025
0.03
0.035

# DC Motor Control

This example shows a cascade speed-control structure for a DC motor. A PWM controlled four-quadrant Chopper is used to feed the DC motor. The Control subsystem includes the outer speed-control loop, the inner current-control loop, and the PWM generation. The total simulation time (t) is 4 seconds. At t = 1.5 seconds, the load torque increases. At t = 2.5 seconds, the reference speed is changed from 1000 rpm to 2000 rpm.

### Model



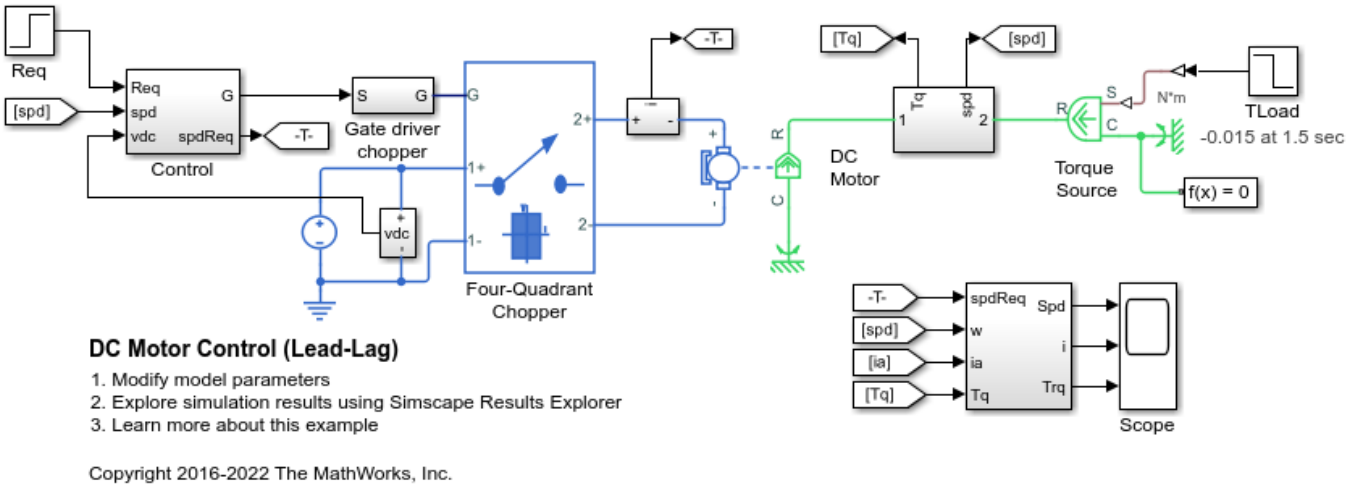

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 50 microseconds.

# DC Motor Control (Lead-Lag)

This example shows a lead-lag speed-control structure for a DC motor. A PWM controlled four-quadrant Chopper is used to feed the DC motor. The Control subsystem includes a lead-lag controller, a constant gain, and the PWM generation. The total simulation time (t) is 4 seconds. At t = 1.5 seconds, the load torque increases. At t = 2.5 seconds, the reference speed is changed from 1000 rpm to 2000 rpm.

### Model



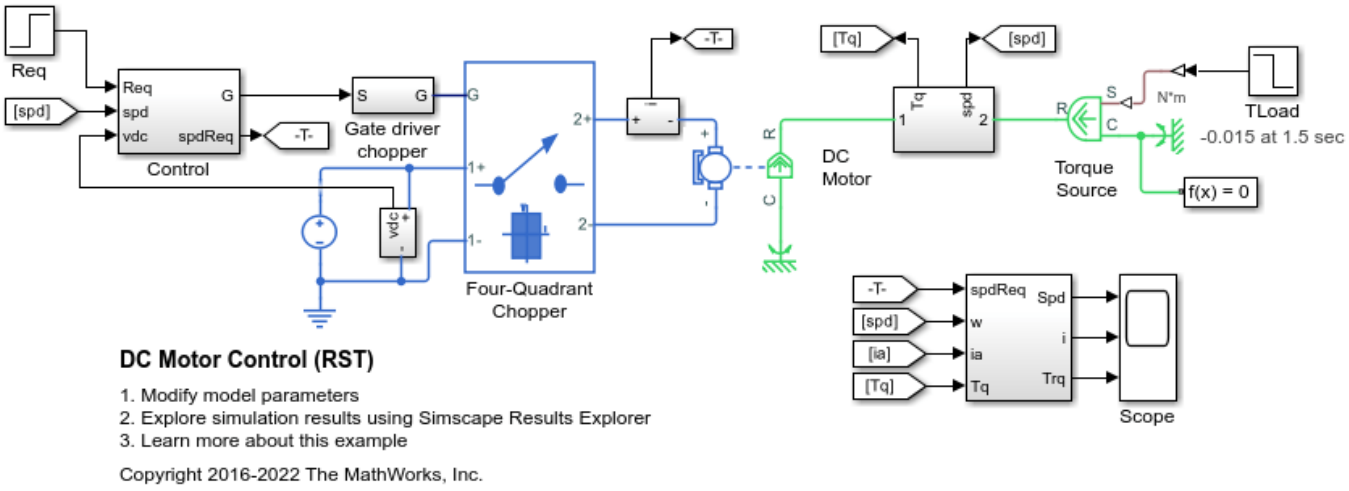

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 50 microseconds.

# DC Motor Control (RST)

This example shows an RST speed-control structure for a DC motor. A PWM controlled four-quadrant Chopper is used to feed the DC motor. The Control subsystem includes the RST controller with control horizon of 30, and the PWM generation. A sensor measures the rotor speed with a delay of 5ms. The total simulation time (t) is 4 seconds. At t = 1.5 seconds, the load torque increases. At t = 2.5 seconds, the reference speed is changed from 1000 rpm to 2000 rpm.

### Model



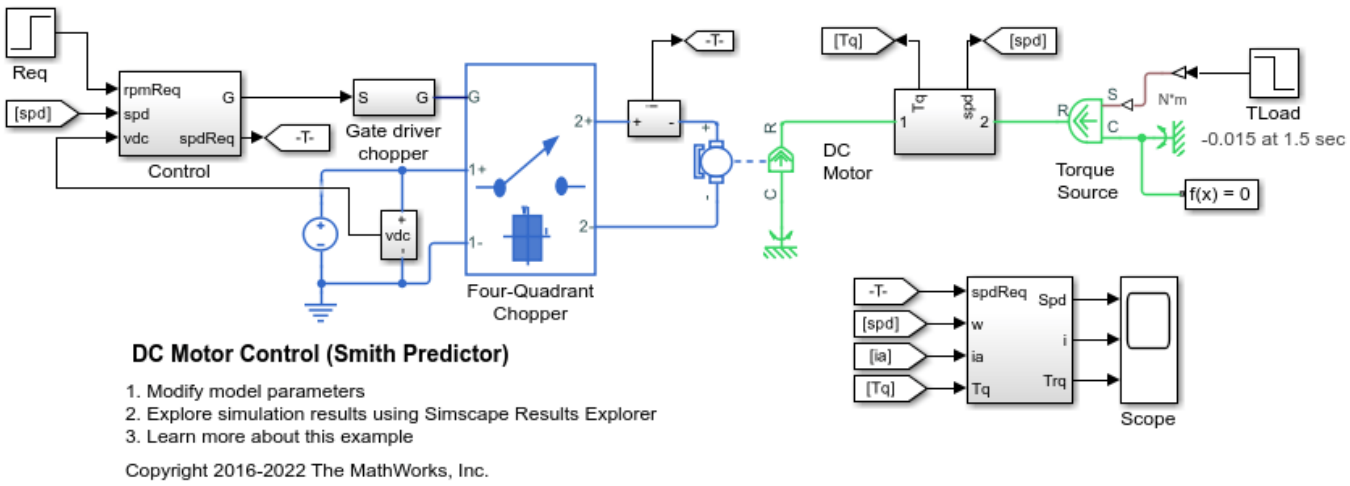

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 50 microseconds.

# DC Motor Control (Smith Predictor)

This example shows a Smith Predictor speed-control structure for a DC motor. A PWM controlled four-quadrant Chopper is used to feed the DC motor. The Control subsystem includes the Smith predictor controller, and the PWM generation. A sensor measures the rotor speed with a delay of 5ms. The total simulation time (t) is 4 seconds. At t = 1.5 seconds, the load torque increases. At t = 2.5 seconds, the reference speed is changed from 1000 rpm to 2000 rpm.

### Model



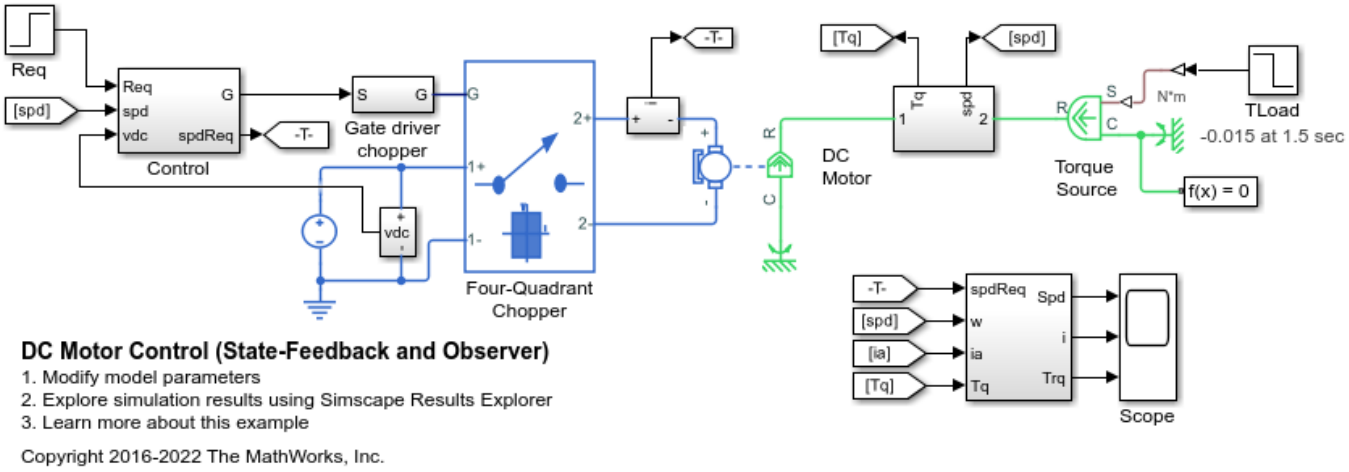

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 50 microseconds.

# DC Motor Control (State-Feedback and Observer)

This example shows a state-feedback speed-control structure for a DC motor. A PWM controlled four-quadrant Chopper is used to feed the DC motor. The Control subsystem includes the state-feedback control loop, and the PWM generation. The state vector includes the rotor speed which is measured, and the dc motor current, which is estimated using an observer. Both the observer and state-feedback controller are synthesized by pole placement using the state-space model of the system. The total simulation time (t) is 4 seconds. At t = 1.5 seconds, the load torque increases. At t = 2.5 seconds, the reference speed is changed from 1000 rpm to 2000 rpm.

### Model



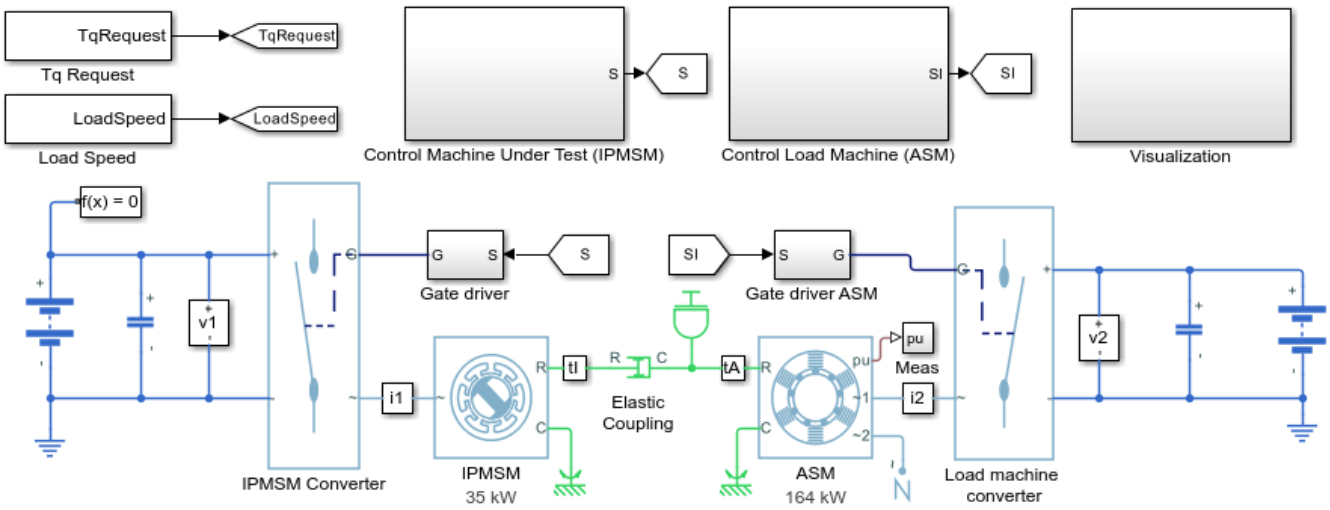

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 50 microseconds.

# Electric Engine Dyno

This example shows how to model an electric vehicle dynamometer test. The test environment contains an asynchronous machine (ASM) and an interior permanent magnet synchronous machine (IPMSM) connected back-to-back through a mechanical shaft. Both machines are fed by high-voltage batteries through controlled three-phase converters. The 164 kW ASM produces the load torque. The 35 kW IPMSM is the electric machine under test. The Control Machine Under Test (IPMSM) subsystem controls the torque of the IPMSM. The controller includes a multi-rate PI-based control structure. The rate of the open-loop torque control is slower than the rate of the closed-loop current control. The task scheduling for the controller is implemented as a Stateflow® state machine. The Control Load Machine (ASM) subsystem uses a single rate to control the speed of the ASM. The Visualization subsystem contains scopes that allow you to see the simulation results.

### Model




**Electric Engine Dyno**

1. Modify model parameters
2. Explore simulation results using Simscape Results Explorer
3. Learn more about this example

Copyright 2016-2022 The MathWorks, Inc.

# Electric Power Assisted Steering

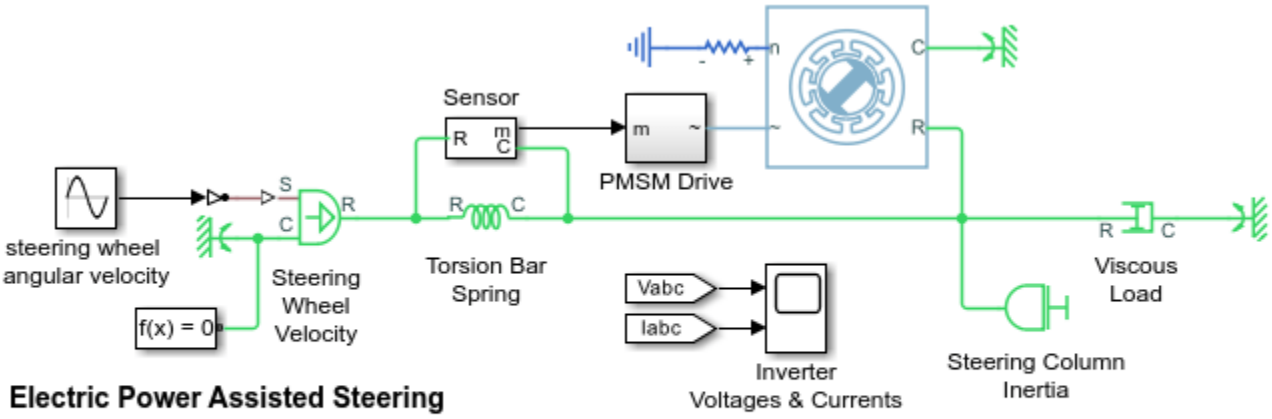
This example shows how to use a Permanent Magnet Synchronous Motor (PMSM) to amplify driver-applied force in a power-assisted automobile steering system.

The steering wheel angular velocity represents the input from the driver. The inputs to the PMSM controller are the angular difference from one end of the steering column to the other and the velocity and angular displacement of the steering load.

The PMSM Inverter is idealized. You can see voltage and current waveforms on the Inverter Voltages & Currents Scope.

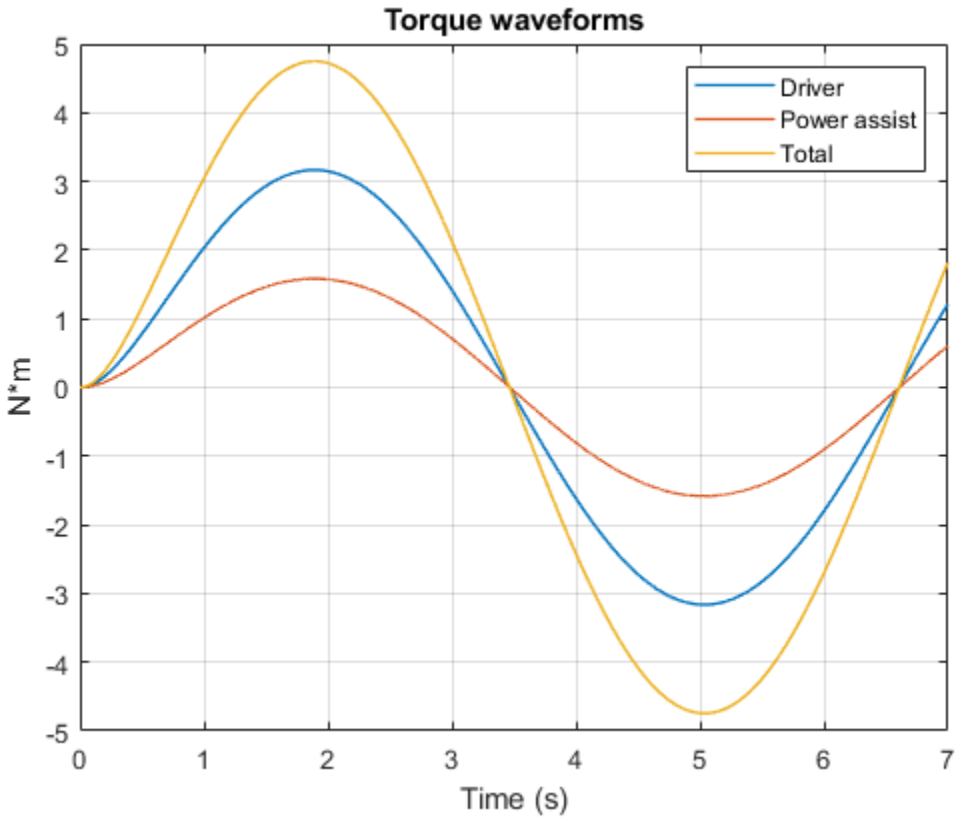
Within the PMSM Driver, the PMSM Controller is a Model block. If you have a Simulink® Coder™ or Embedded Coder® license, you can generate code directly from the PMSM Controller referenced model.

### Model




### Simulation Results from Simscape Logging

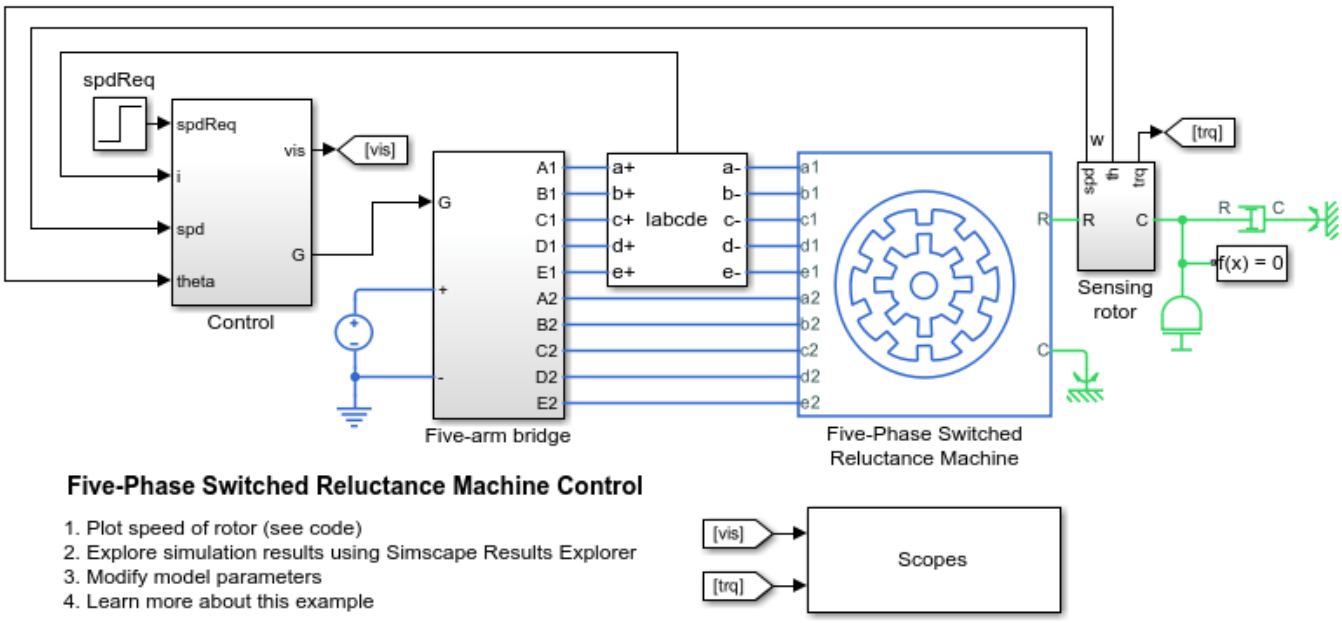
The plot below shows the driver, power assist, and total torque waveforms.


Torque waveforms
Driver
Power assist
Total
N*m
Time (s)
5
4
3
2
1
0
-1
-2
-3
-4
-5
0
1
2
3
4
5
6
7

# Five-Phase Switched Reluctance Machine Control

This example shows how to control the rotor speed in a five-phase switched reluctance machine (SRM) based electrical drive. A DC voltage source feeds the SRM through a controlled five-arm bridge. The converter turn-on and turn-off angles are held constant.

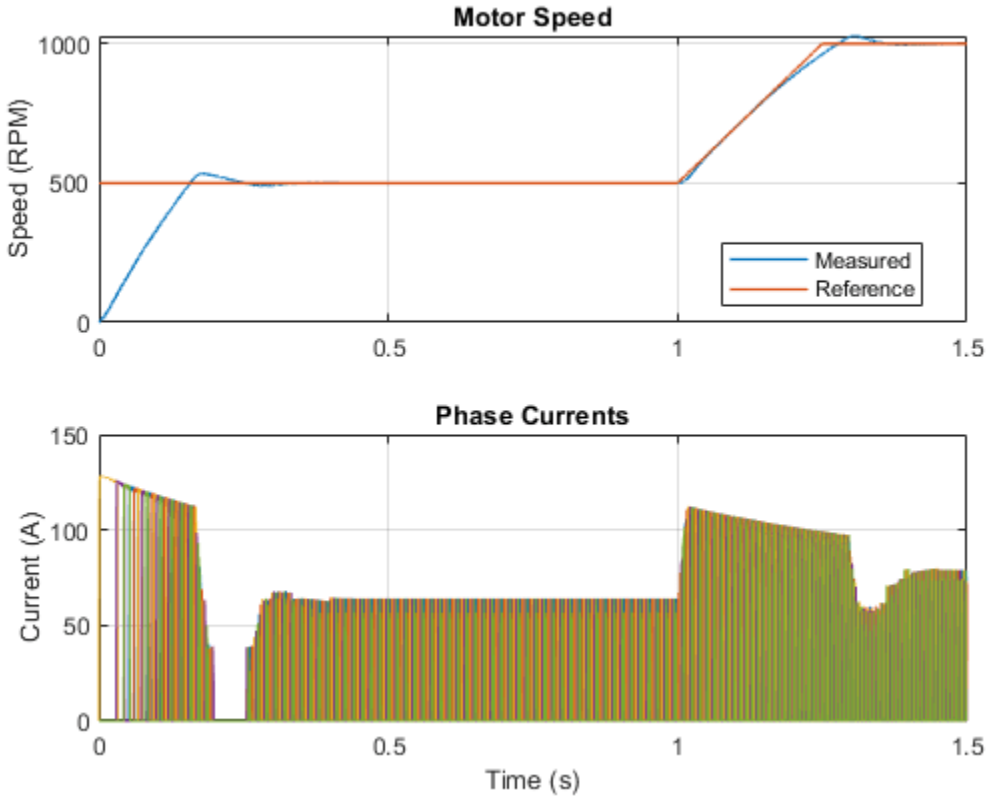
### Model




Copyright 2013-2022 The MathWorks, Inc.

### Simulation Results from Simscape Logging

The plot below shows the requested and measured speed for the test, as well as the phase currents in the electric drive.



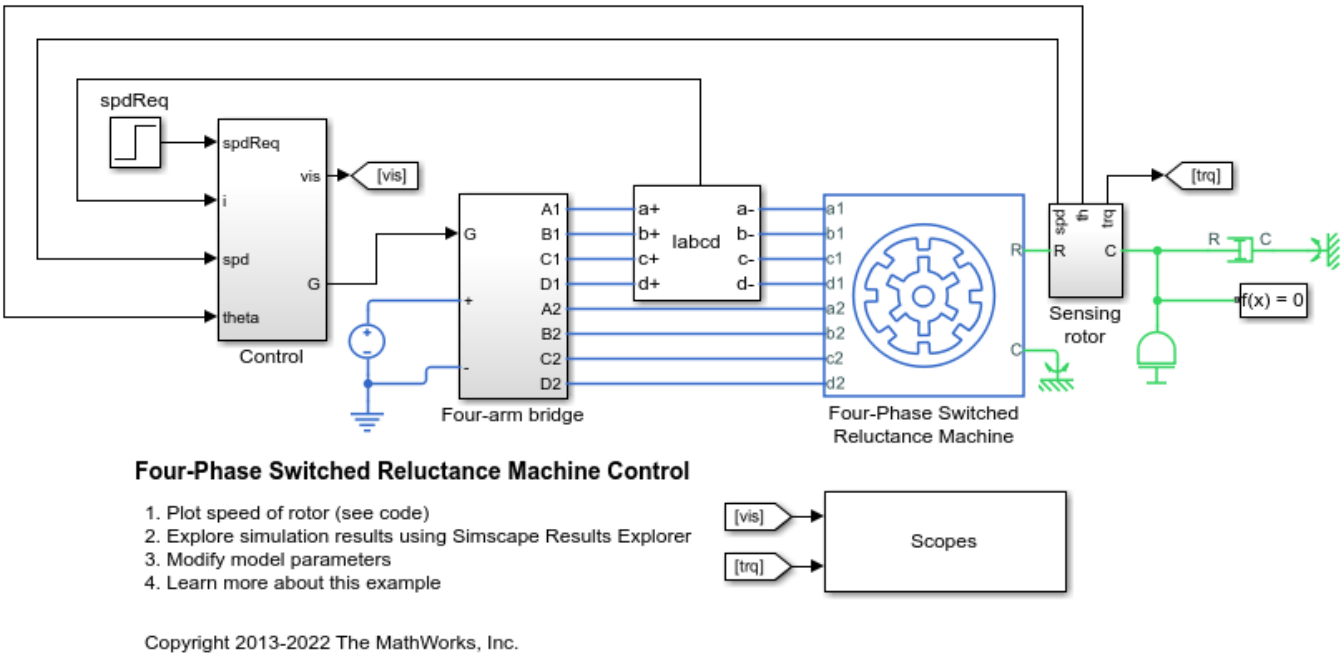

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 100 microseconds.

# Four-Phase Switched Reluctance Machine Control

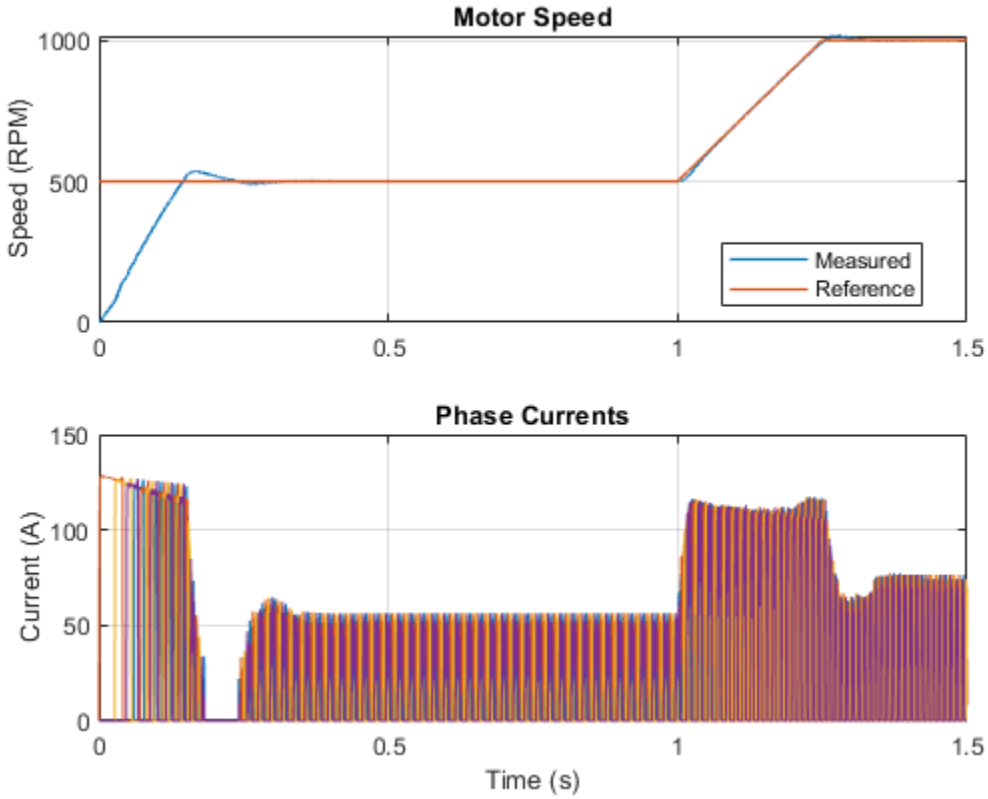
This example shows how to control the rotor speed in a four-phase switched reluctance machine (SRM) based electrical drive. A DC voltage source feeds the SRM through a controlled four-arm bridge. The converter turn-on and turn-off angles are held constant.

### Model




### Simulation Results from Simscape Logging

The plot below shows the requested and measured speed for the test, as well as the phase currents in the electric drive.



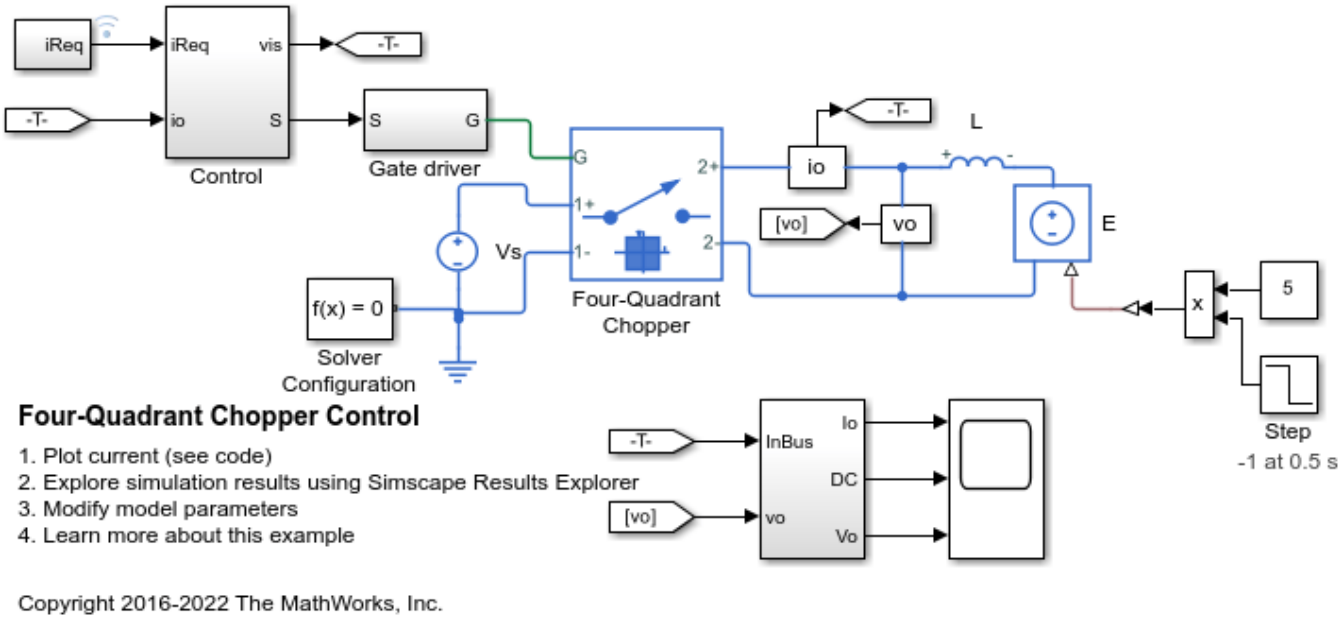

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 100 microseconds.

# Four-Quadrant Chopper Control

This example shows how to control a four-quadrant chopper. The Control subsystem implements a simple PI-based control algorithm for controlling the output current. The simulation uses both positive and negative references. The total simulation time (t) is 1 second. At t = 0.5 seconds, the polarity of the load DC source E is changed.

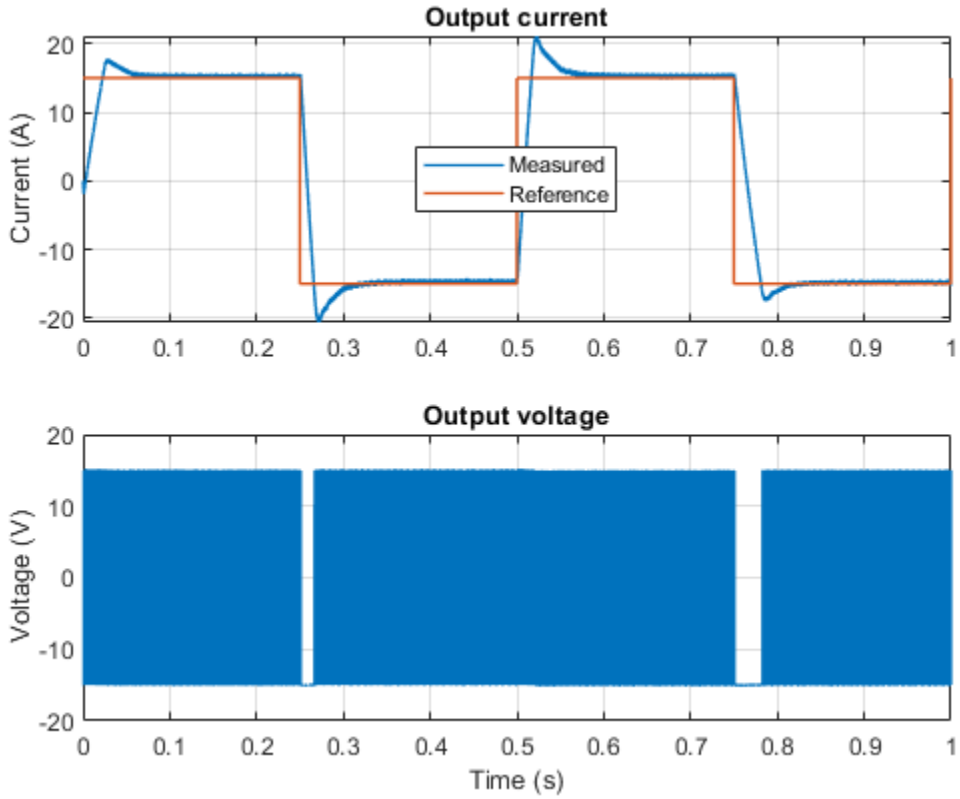
### Model




Copyright 2016-2022 The MathWorks, Inc.

### Simulation Results from Simscape Logging

The plot below shows the requested and measured current for the test and the output voltage in the circuit.



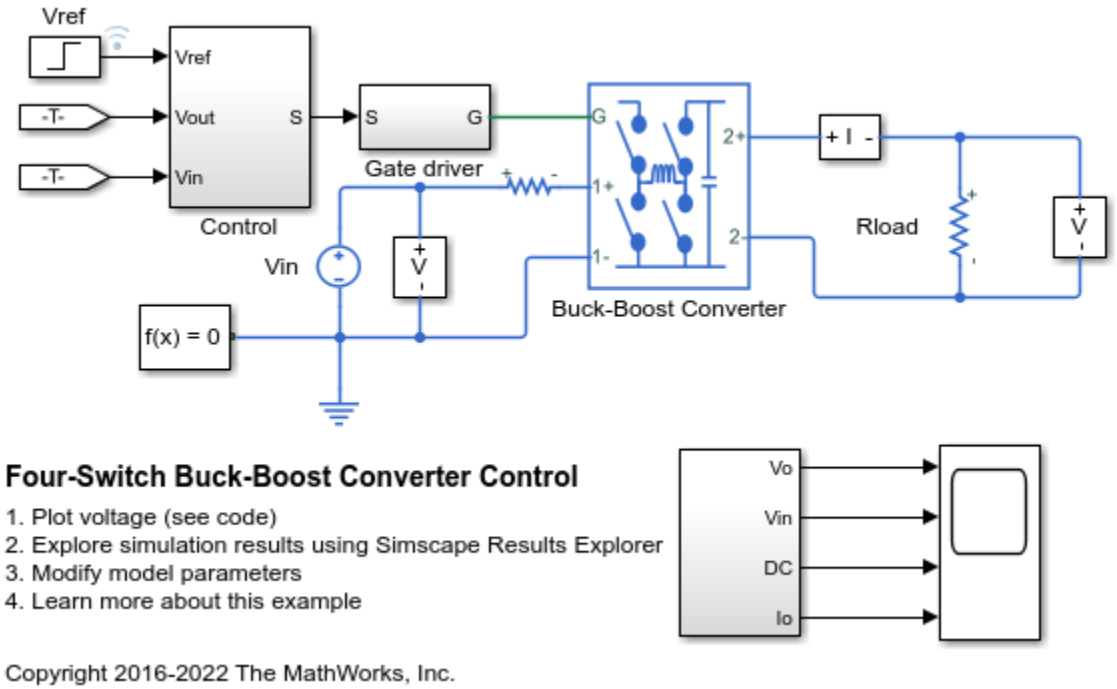

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 50 microseconds.

# Four-Switch Buck-Boost Converter Control

This example shows how to control the output voltage of a four-switch buck-boost converter. To adjust the duty cycle, the Control subsystem uses a PI-based control algorithm. In both the boost and buck modes, one switch controls the duty cycle, one is operated inversely and the other two are kept in fix positions. The input voltage and the system load are considered constant throughout the simulation. The total simulation time (t) is 0.25 seconds. At t = 0.15 seconds, the voltage reference changes and the system switches from buck mode to boost mode.

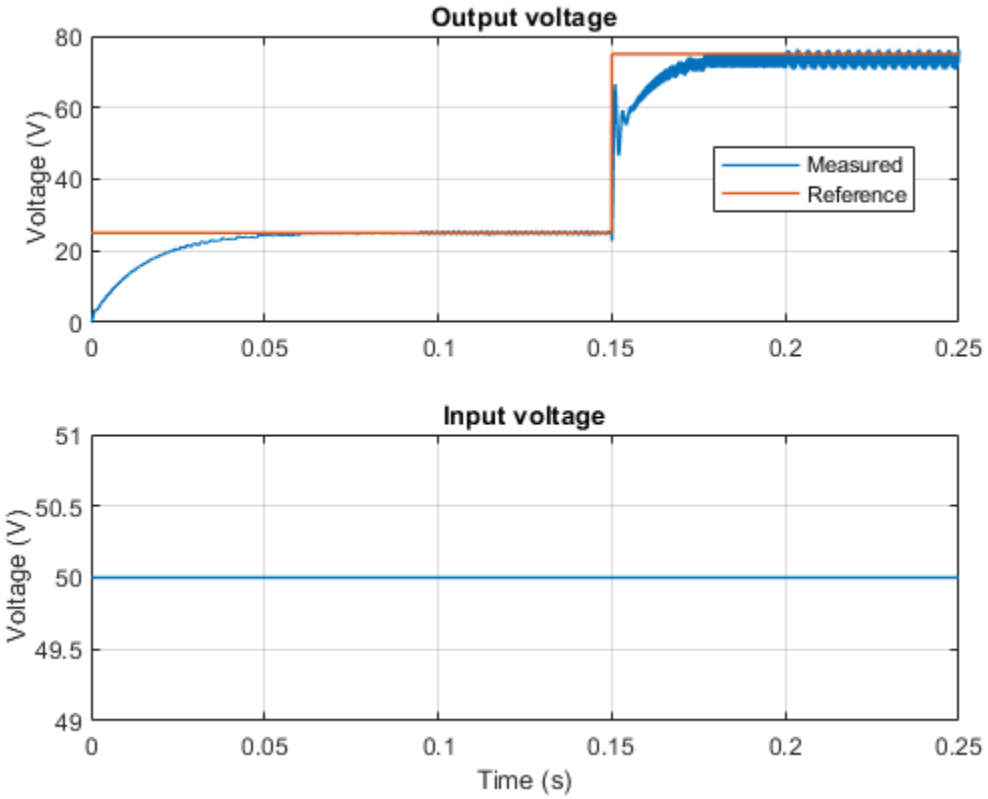
### Model




Copyright 2016-2022 The MathWorks, Inc.

### Simulation Results from Simscape Logging

The plot below shows the requested and measured voltage for the test and the input voltage in the circuit.


Output voltage
Voltage (V)
80
60
40
20
0
0
0.05
0.1
0.15
0.2
0.25
Measured
Reference
Input voltage
Voltage (V)
51
50.5
50
49.5
49
0
0.05
0.1
0.15
0.2
0.25
Time (s)

# Harmonic Analysis of a Three-Phase Rectifier

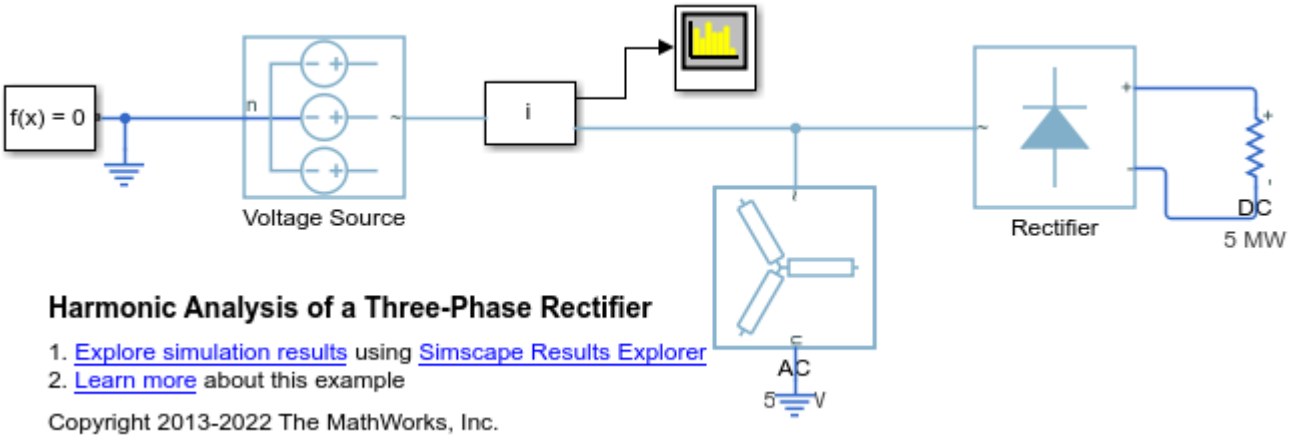
This example shows how to use functions which analyze Simscape™ logging data to get harmonic magnitudes, calculate total harmonic distortion percentage and plot harmonic magnitudes. The model to which this analysis is applied is of a three-phase rectifier. The functions demonstrated are:

- ee_getHarmonics
- ee_calculateThdPercent
- ee_plotHarmonics

### Open Model

Open the model.

```
open_system( 'ee_harmonics_rectifier' );
```




### Specification of Parameters

Where:

- Rated AC voltage, $V_{Rated}\mathrm{V}$
- Rated AC frequency, $F_{Rated}\mathrm{Hz}$
- AC real power load, $P_{AC}\mathrm{W}$
- DC real power load, $P_{DC}\mathrm{W}$
- Total apparent power, $S_{Rated}\mathrm{VA}$

For the test circuit, the AC load is set to consume 5MW, and the DC load is set to consume approximately 5MW.

```
V_Rated = 4160;
F_Rated = 60;
P_AC = 5e6;
P_DC = 5e6;
S_Rated = P_AC + P_DC;
```

### Calculate Source Impedance

Where:

- X/R Ratio, $XR$
- Per-unit impedance, $Z_{pu}$
- Per-unit base impedance, $Z_{base}$
- Per-unit base inductance, $L_{base}$
- Source series resistance, $R_{series}\Omega$
- Source series inductance, $L_{series}\mathrm{H}$

```
XR = 15;
Z_pu = 0.01;
Z_base = ((V_Rated/sqrt(3))^2)/(S_Rated/3);
L_base = Z_base/(2*pi*F_Rated);
R_series = cos(atan(XR))*Z_pu*Z_base;
L_series = sin(atan(XR))*Z_pu*L_base;
```

### Calculate DC Resistance

Where:

- Average DC voltage calculated neglecting losses, $V_{DC}\mathrm{V}$
- DC resistance, $R_{DC}\Omega$

(Consult appendix for derivation of equations)

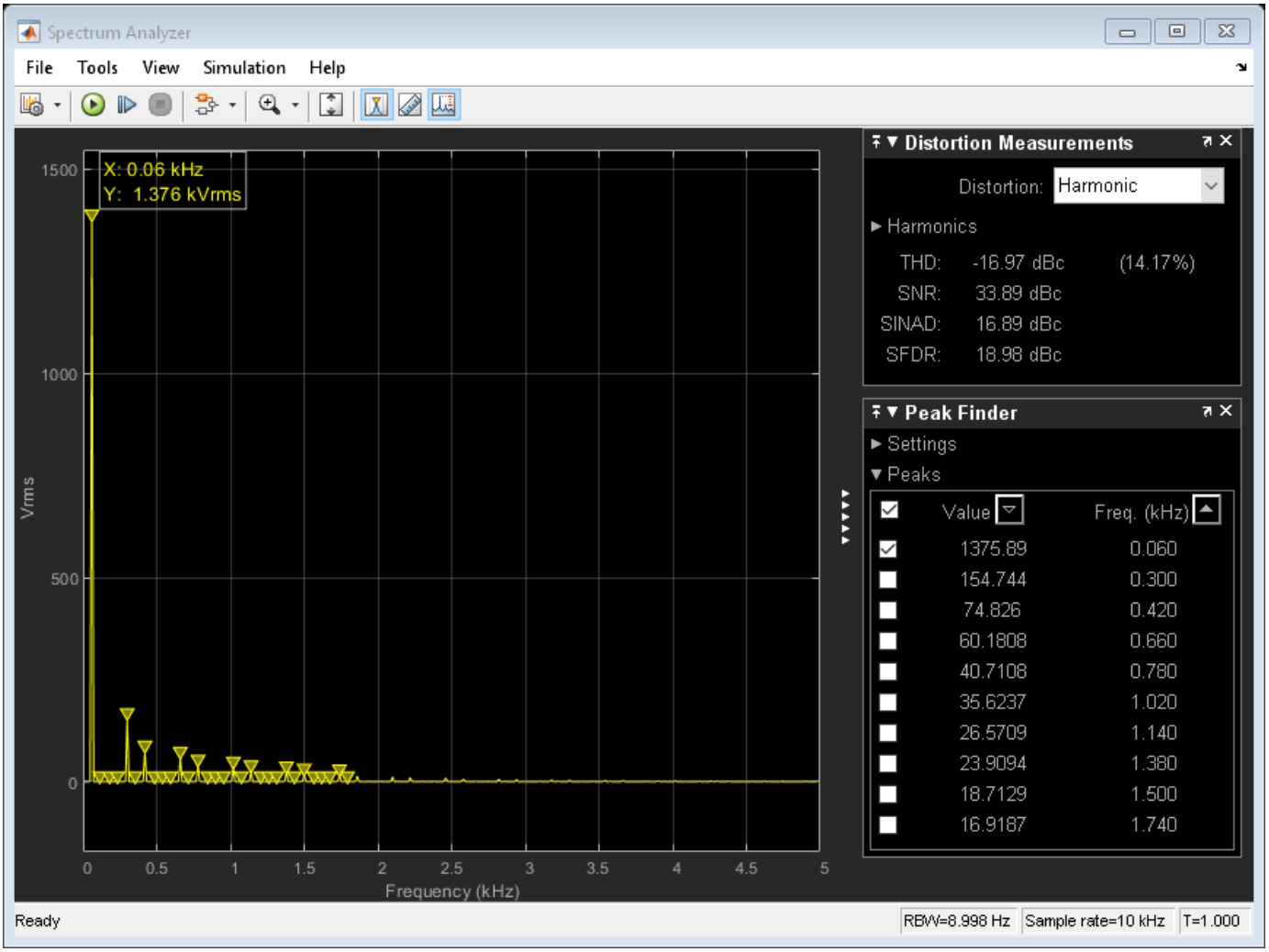
```
V_DC = 3*sqrt(2)*V_Rated/pi;
R_DC = V_DC^2/P_DC;
disp( [ 'DC resistance required to draw ', num2str( P_DC ), ' W on DC side = ', num2str( R_DC ),
```

```
DC resistance required to draw 5000000 W on DC side = 6.3123 Ohm
```

### Use Parameters in Simscape Model

The parameters calculated can now be used in a Simscape model, ee_harmonics_rectifier. Once simulated, the model is set to create a Simscape logging variable, simlog_ee_harmonics_rectifier.

```
sim( 'ee_harmonics_rectifier' );
Voltage_Source_Currents = simlog_ee_harmonics_rectifier.Voltage_Source.I;
```




### Obtain Harmonic Data

Details of harmonic order, harmonic magnitude and fundamental frequency can be obtained from a Simscape logging variable using the ee_getHarmonics function.

```
[ harmonicOrder, harmonicMagnitude, fundamentalFrequency ] = ee_getHarmonics( Voltage_Source_Cur
```

### Calculate Peak Fundamental Value

The peak value of fundamental can be extracted.

```
fundamentalPeak = harmonicMagnitude( harmonicOrder==1 );
disp( [ 'Peak value of fundamental = ', num2str( fundamentalPeak ), ' A' ] );
```

```
Peak value of fundamental = 1945.806 A
```

### Remove Small Harmonics

Find and keep harmonics which are greater than one thousandth of fundamental.

```
threshold = fundamentalPeak ./ 1e3;
aboveThresold = harmonicMagnitude > threshold;
```

```
harmonicOrder = harmonicOrder( aboveThresold )';
harmonicMagnitude = harmonicMagnitude( aboveThresold )';
```

### Display Tabular Harmonic Data

Harmonic data can be contained in a MATLAB® table.

```
harmonicRms = harmonicMagnitude./sqrt(2);
harmonicPercentage = 100.*harmonicMagnitude./harmonicMagnitude( harmonicOrder==1 );
harmonicTable = table( harmonicOrder,...
    harmonicMagnitude,...
    harmonicRms,...
    harmonicPercentage,...
    'VariableNames', {'Order','Magnitude','RMS','Percentage'});
display( harmonicTable );
```

```
harmonicTable =

  10x4 table

    Order    Magnitude     RMS      Percentage
    _____    _________    ______    __________

      1       1945.8      1375.9         100
      5       218.86      154.75       11.248
      7       105.83      74.835        5.439
     11       85.135        60.2       4.3753
     13       57.599      40.729       2.9602
     17       50.417       35.65       2.5911
     19       37.612      26.596        1.933
     23       33.859      23.942       1.7401
     25       26.507      18.743       1.3622
     29       23.979      16.955       1.2323
```

### Total Harmonic Distortion

Calculate Total Harmonic Distortion (THD) percentage from harmonic data using the ee_calculate_ThdPercent function.

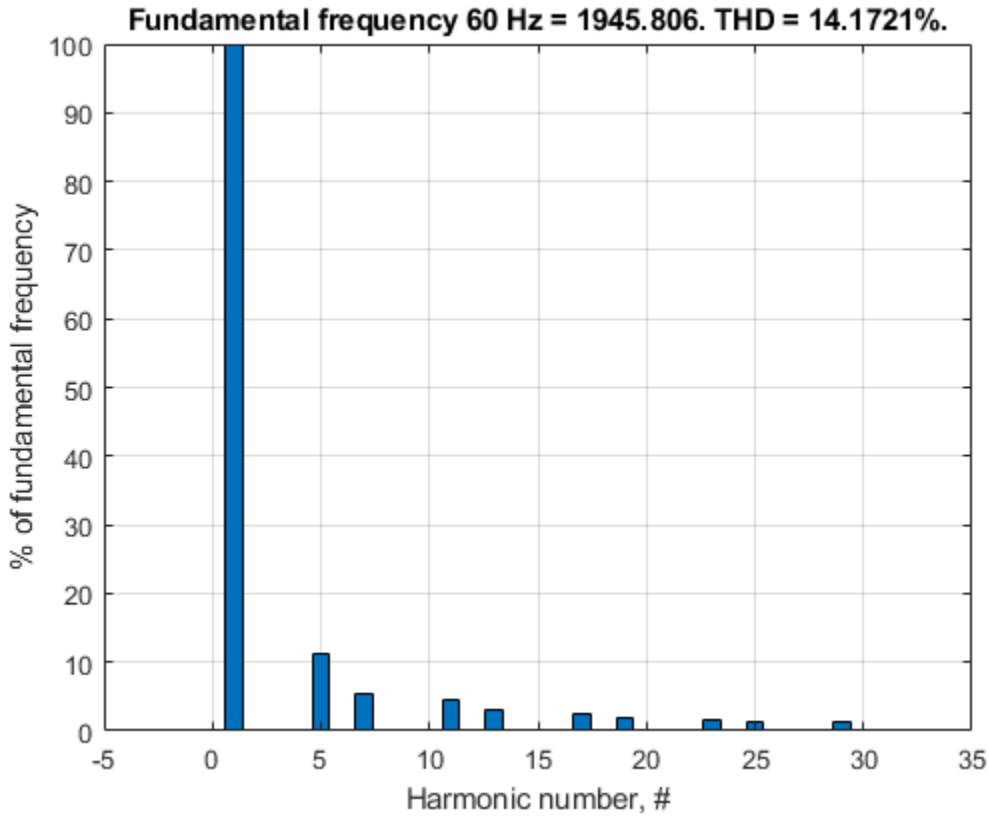
```
thdPercent = ee_calculateThdPercent( harmonicOrder, harmonicMagnitude );
disp( [ 'Total Harmonic Distortion percentage = ' num2str( thdPercent ), ' %' ] );
```

```
Total Harmonic Distortion percentage = 14.1721 %
```

### Plot Harmonics

The harmonic data could be plotted from the harmonic data using the MATLAB bar function. For convenience the ee_plotHarmonics function plots a bar chart directly from the Simscape logging variable.

```
ee_plotHarmonics( Voltage_Source_Currents );
h1_ee_rectifier_thd = gcf;
```




### Conclusion

This example shows how to use three functions which are callable from the MATLAB command line. The functions analyze Simscape logging data to get harmonic magnitudes, calculate total harmonic distortion percentage and plot harmonic magnitudes.

### Appendix - Equations for Calculation of DC Resistance Value

The relationship between peak AC input, $V_P$, and average DC output, $V_{DC}$, of a three-phase rectifier, neglecting losses, can be calculated as follows:

$$V_{DC} = \frac{1}{2\pi/6} \int_{2\pi/3}^{\pi/3} \sqrt{3} V_P \sin(\omega t)\, \mathrm{d}(\omega t)$$

$$V_{DC} = \frac{3\sqrt{3}}{\pi} V_P$$

As the relationship between the rated voltage (line-line RMS), $V_{Rated}$, and peak phase voltage, $V_P$, is:

$$V_P = \frac{\sqrt{2}}{\sqrt{3}} V_{Rated}$$

The relationship between rated AC voltage and average DC voltage is:

$$V_{DC} = \frac{3\sqrt{2}}{\pi} V_{Rated}$$

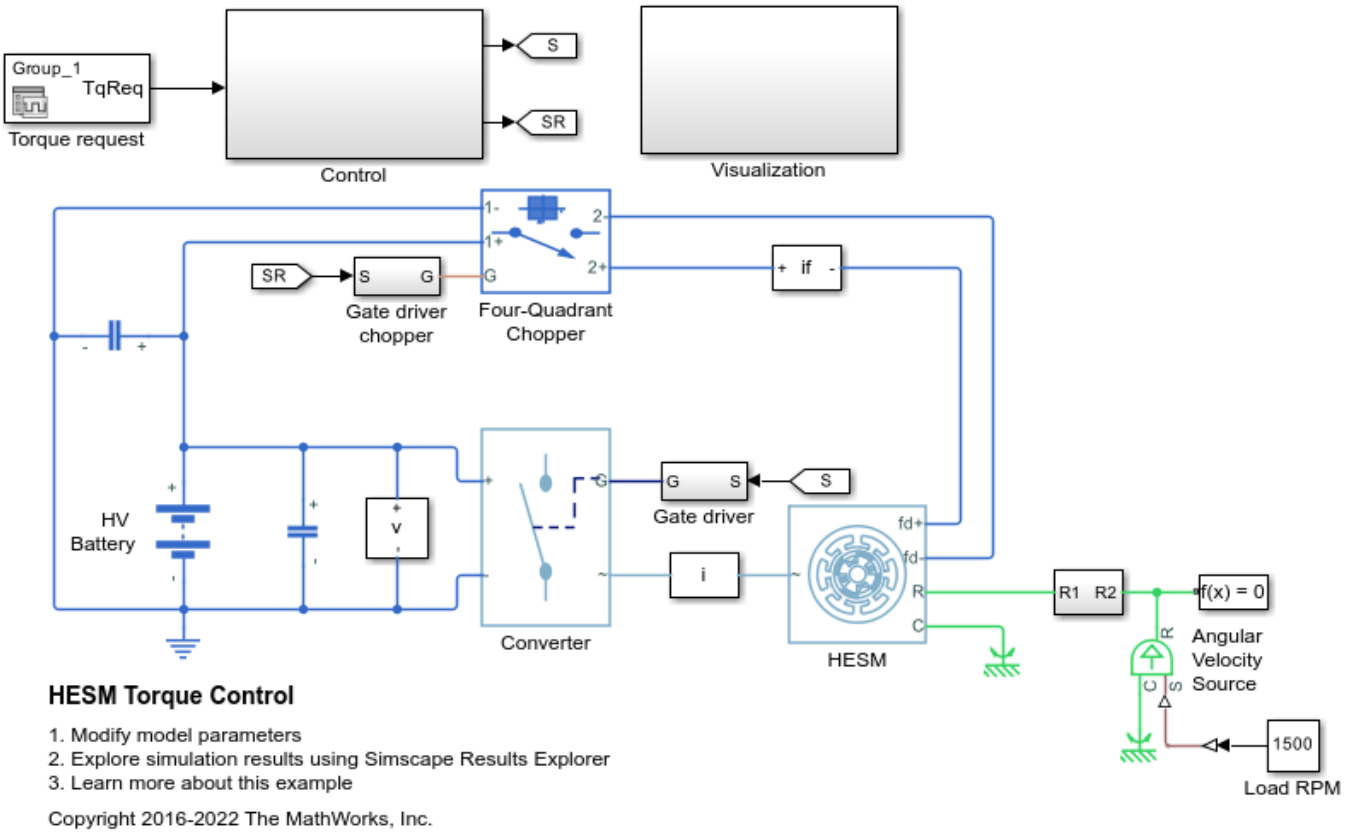
The resistance, $R$ required to draw a particular power, $P$, is:

$$R = \frac{V_{DC}^2}{P}$$

# HESM Torque Control

This example shows how to control the torque in a hybrid excitation synchronous machine (HESM) based electrical-traction drive. Permanent magnets and an excitation winding excite the HESM. A high-voltage battery feeds the SM through a controlled three-phase converter for the stator windings and through a controlled four quadrant chopper for the rotor winding. An ideal angular velocity source provides the load. The Control subsystem uses an open-loop approach to control the torque and a closed-loop approach to control the current. At each sample instant, the torque request is converted to relevant current references. The current control is PI-based. The simulation uses several torque steps in both the motor and generator modes. The Visualization subsystem contains scopes that allow you to see the simulation results.

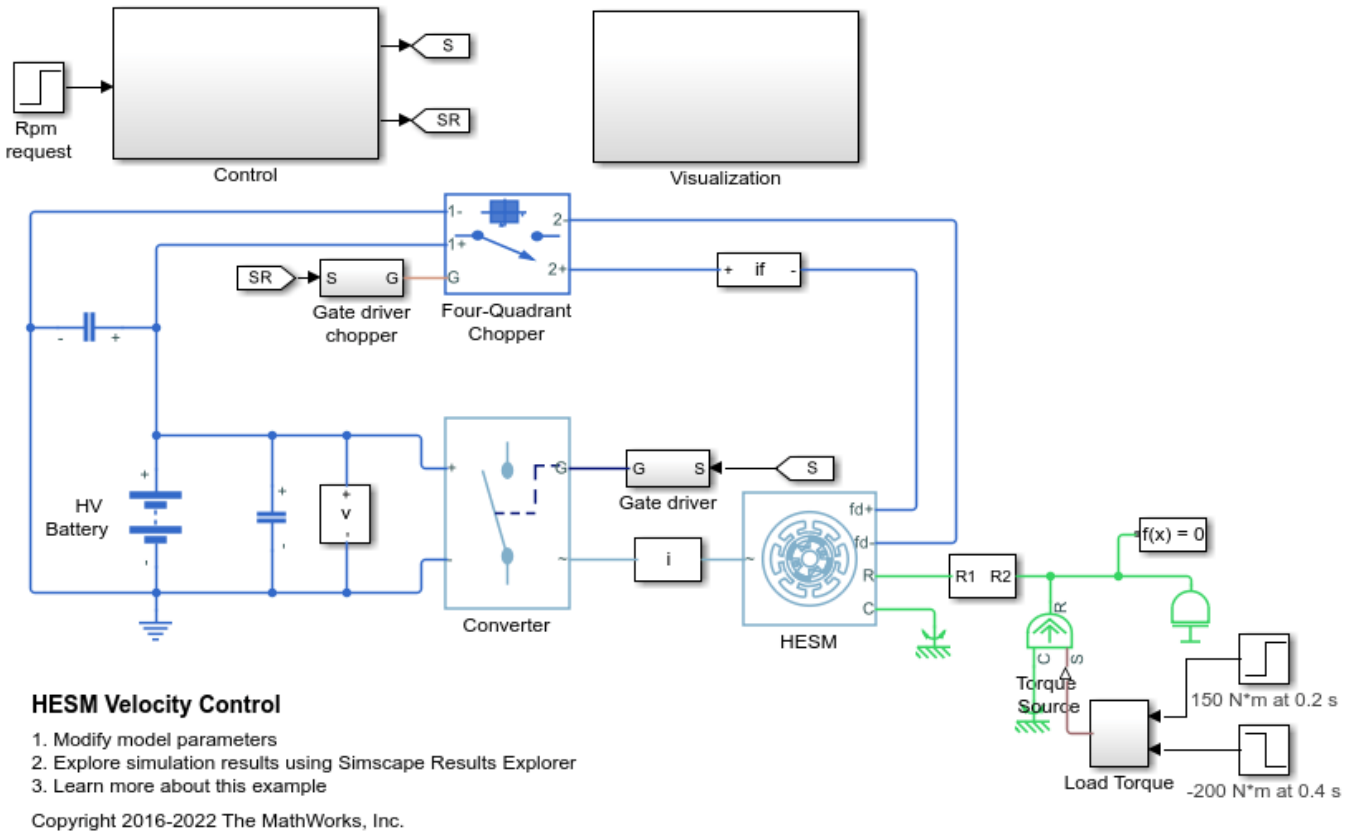
### Model



# HESM Velocity Control

This example shows how to control the rotor angular velocity in a hybrid excitation synchronous machine (HESM) based electrical-traction drive. Permanent magnets and an excitation winding excite the HESM. A high-voltage battery feeds the HESM through a controlled three-phase converter for the stator windings and through a controlled four quadrant chopper for the rotor winding. An ideal torque source provides the load. The Control subsystem includes a multi-rate PI-based cascade control structure. The control structure has an outer angular-velocity-control loop and three inner current-control loops. The Visualization subsystem contains scopes that allow you to see the simulation results.

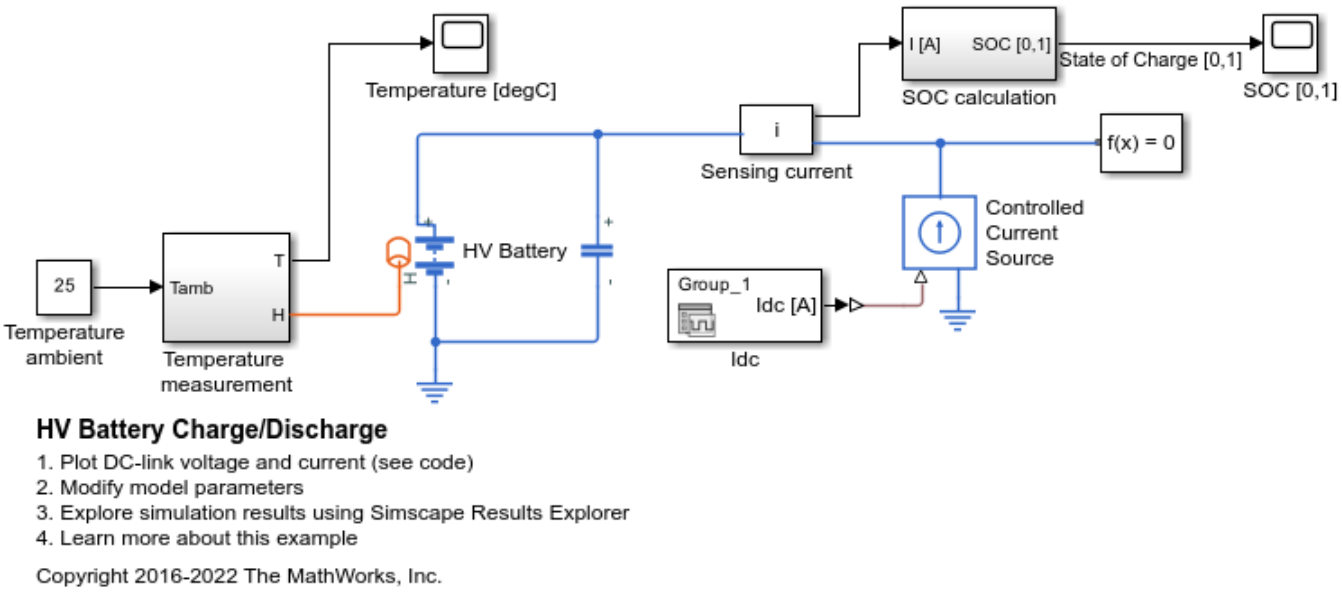
### Model



# HV Battery Charge/Discharge

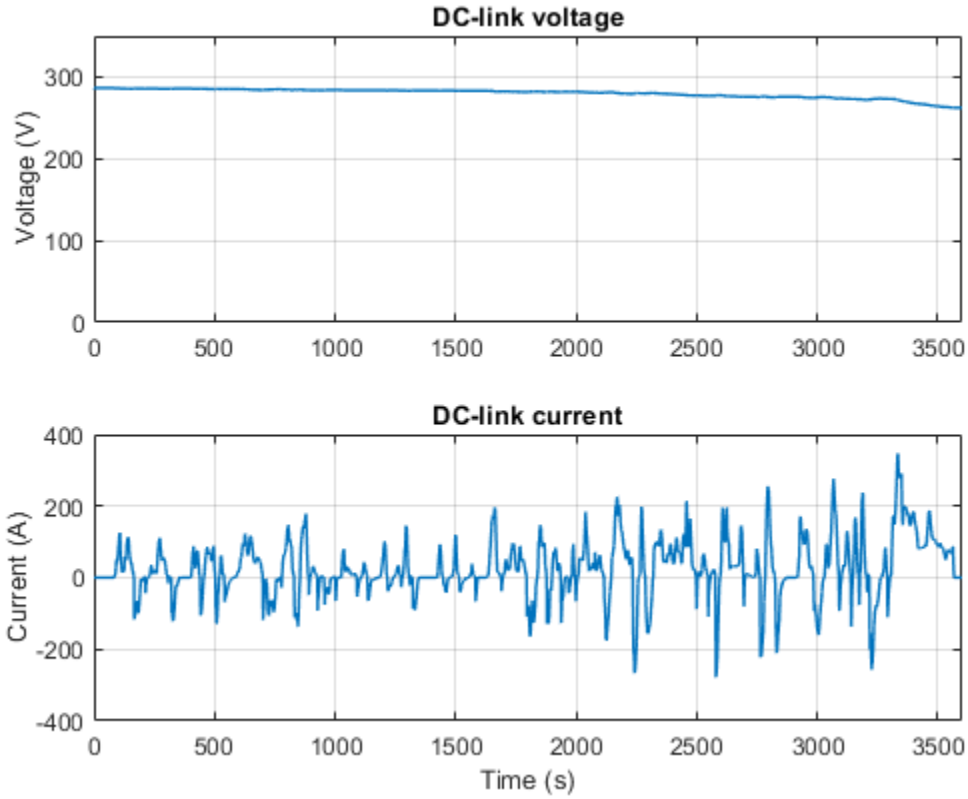
This example shows a high-voltage battery like those used in hybrid electric vehicles. The model uses a realistic DC-link current profile, which originates from a dynamic driving cycle. The total simulation time is 3600 seconds.

### Model




### Simulation Results from Simscape Logging

The plot below shows the battery voltage and output current.




### Results from Real-Time Simulation

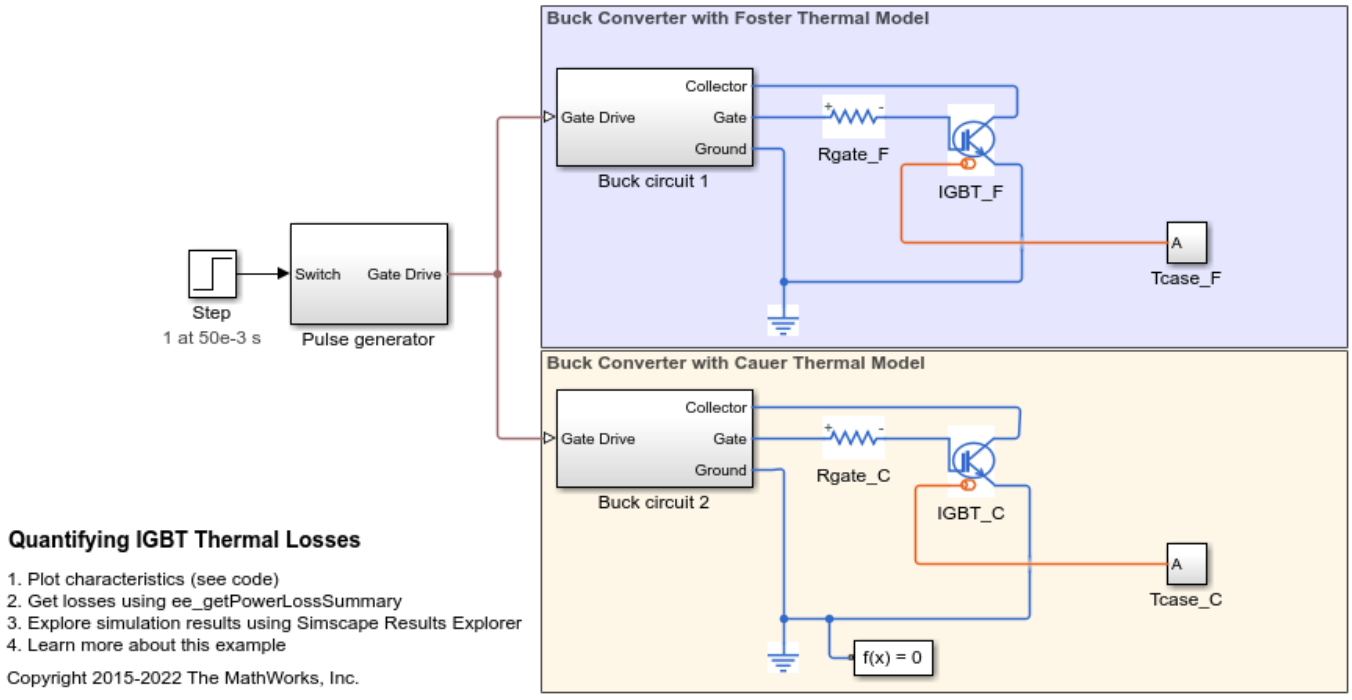
This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 20 microseconds.

# Quantifying IGBT Thermal Losses

This example shows the generation of a temperature profile based on switching and conduction losses in an insulated-gate bipolar transistor (IGBT). There are two buck converters. For one converter, the IGBT attaches to a Foster thermal model. For the other converter, the IGBT attaches to a Cauer thermal model. The parameters for the thermal models are tuned to give roughly equivalent results. At a simulation time of 50ms, the driving frequency changes from 40kHz to 20kHz, which increases the conduction losses and decreases the switching losses. The change in the losses results in a corresponding change in the temperature of the IGBT.

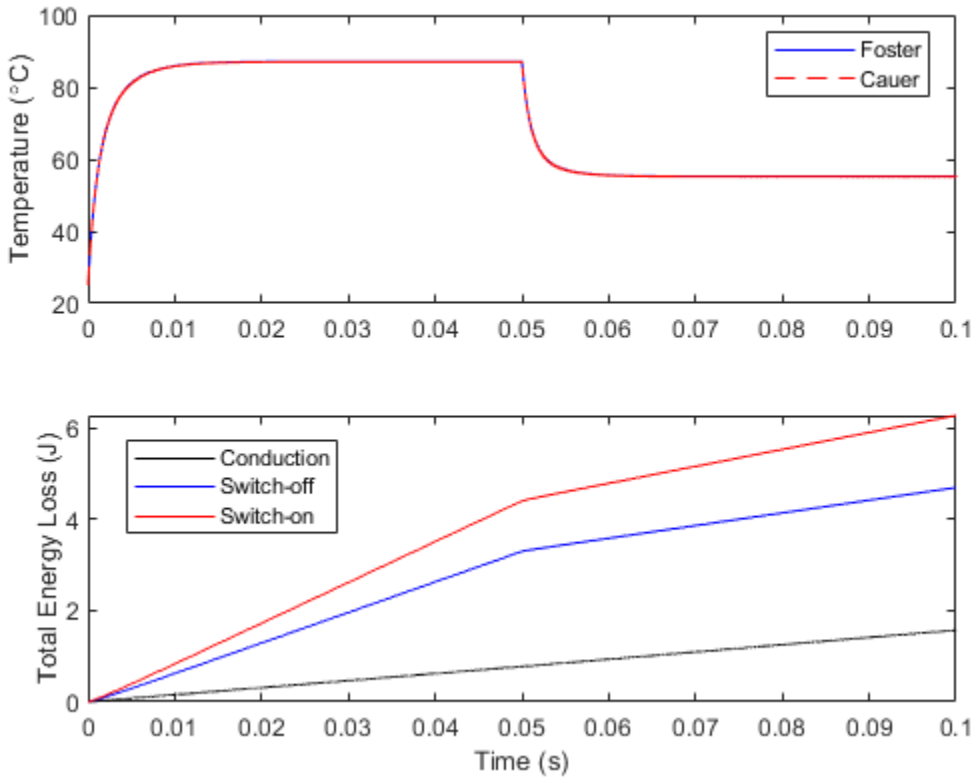
### Model

Both Foster and Cauer thermal networks are connected to a fixed temperature source that represents the heat sink temperature. To obtain meaningful results, you must connect the Foster thermal network to a fixed temperature source. To model a heat sink thermal mass and convection to the environment, you can connect the Cauer model to any other thermal network. In this example the Cauer model is connected to a third first-order thermal network, and the first two thermal states are included in the internal thermal model of the IGBT block. A third-order model is required to match the third-order Foster model implemented. Typically datasheets provide only secord-order junction and case dynamics for a Cauer model.




### Simulation Results from Simscape Logging

The plot below shows the IGBT temperature and energy loss as functions of time for both the Foster and Cauer thermal models. The power loss is the slope of the energy-loss curve.




### Generate a Summary of Losses

The `ee_getPowerLossSummary` utility function reports losses incurred by the circuit components from logged simulation data. The 'Power' column reports conduction losses, and the SwitchingLosses column reports the semiconductor switching losses.

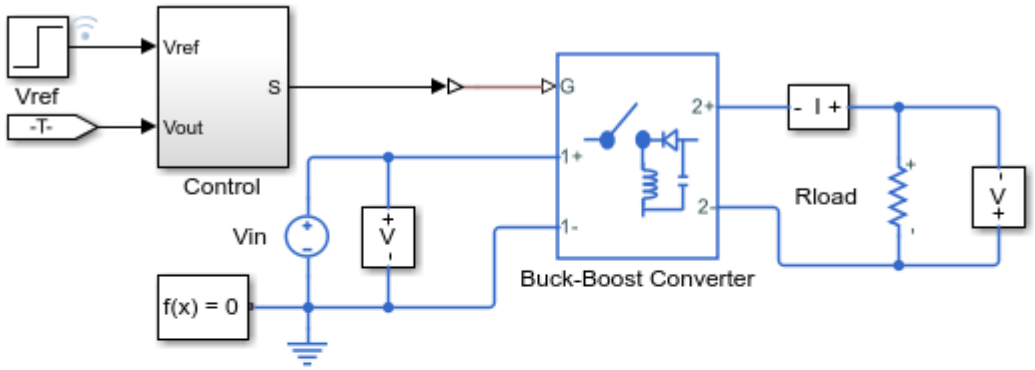
```
ans =

  12x3 table

                    LoggingNode                       Power     SwitchingLosses
    ____________________________________________    _______    _______________

    {'ee_igbt_losses.Buck_circuit_1.Load'     }     7453.8              0
    {'ee_igbt_losses.Buck_circuit_2.Load'     }     7453.8              0
    {'ee_igbt_losses.IGBT_C'                  }     15.704         109.64
    {'ee_igbt_losses.IGBT_F'                  }     15.703         109.68
    {'ee_igbt_losses.Buck_circuit_1.Diode'    }     12.147              0
    {'ee_igbt_losses.Buck_circuit_2.Diode'    }     12.147              0
    {'ee_igbt_losses.Buck_circuit_1.Capacitor'}          0              0
    {'ee_igbt_losses.Buck_circuit_1.Inductor' }          0              0
    {'ee_igbt_losses.Buck_circuit_2.Capacitor'}          0              0
    {'ee_igbt_losses.Buck_circuit_2.Inductor' }          0              0
    {'ee_igbt_losses.Rgate_C'                 }          0              0
```

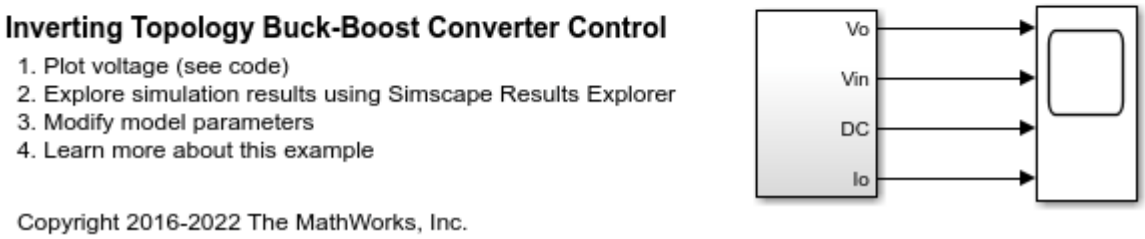
```
{'ee_igbt_losses.Rgate_F'                }         0             0
```

# Inverting Topology Buck-Boost Converter Control

This example shows how to control the output voltage of an inverting topology buck-boost converter. The inverting topology buck-boost converter uses only a single switch and the output voltage is of the opposite polarity than the input. To adjust the duty cycle, the Control subsystem uses a PI-based control algorithm. The input voltage and the system load are considered constant throughout the simulation. The total simulation time (t) is 0.25 seconds. At t = 0.15 seconds, the voltage reference changes and the system switches from buck mode to boost mode.

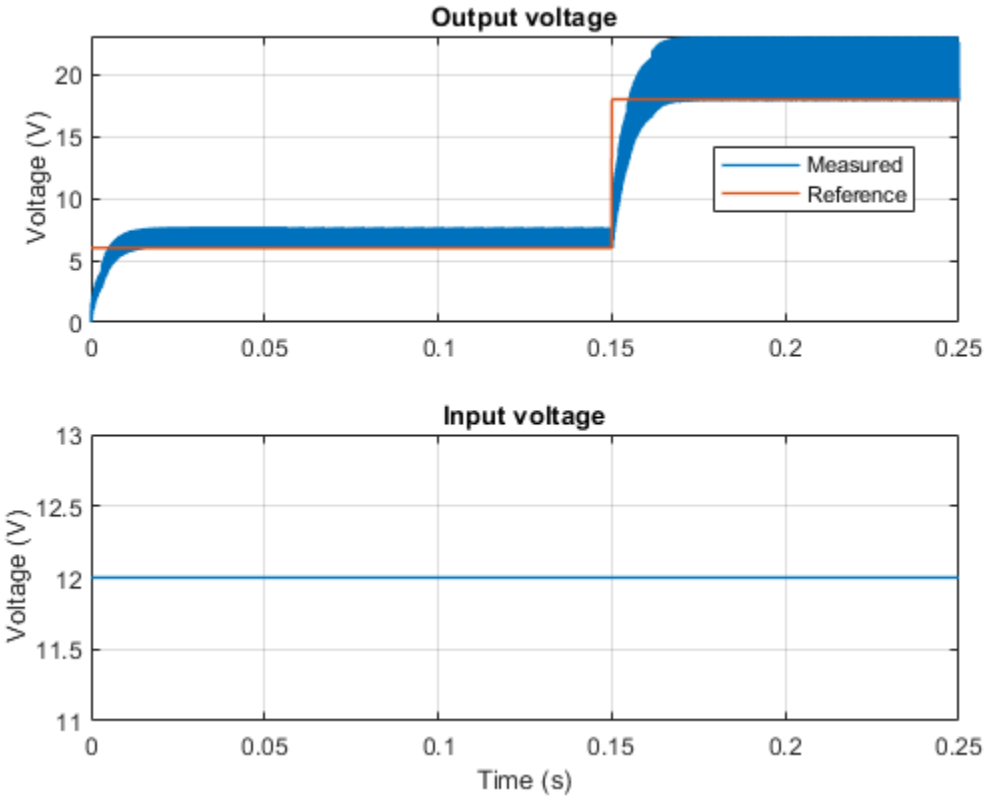
### Model




### Simulation Results from Simscape Logging

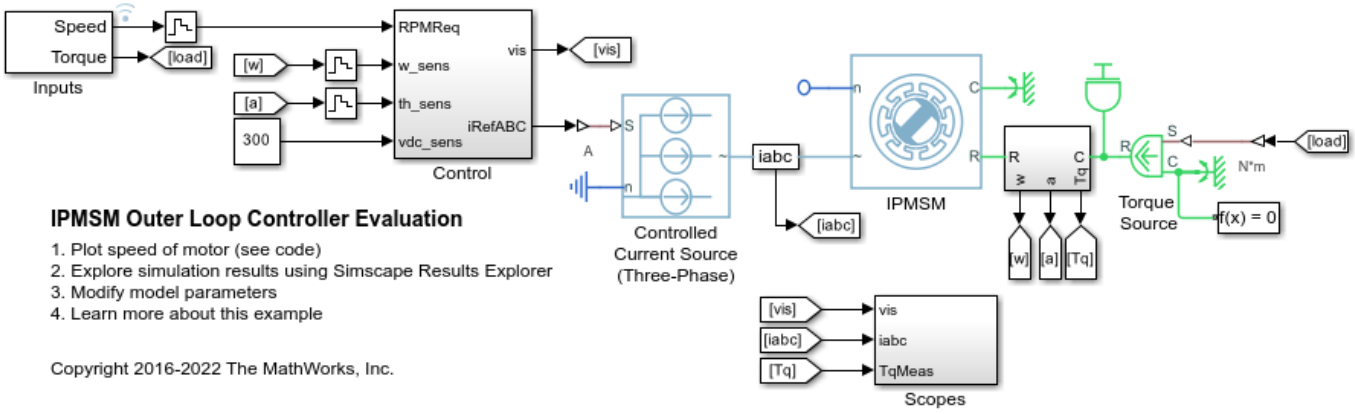
The plot below shows the requested and measured voltage for the test and the input voltage in the circuit.


Output voltage
Voltage (V)
20
15
10
5
0
0
0.05
0.1
0.15
0.2
0.25
Measured
Reference
Input voltage
Voltage (V)
13
12.5
12
11.5
11
0
0.05
0.1
0.15
0.2
0.25
Time (s)

# IPMSM Outer Loop Controller Evaluation

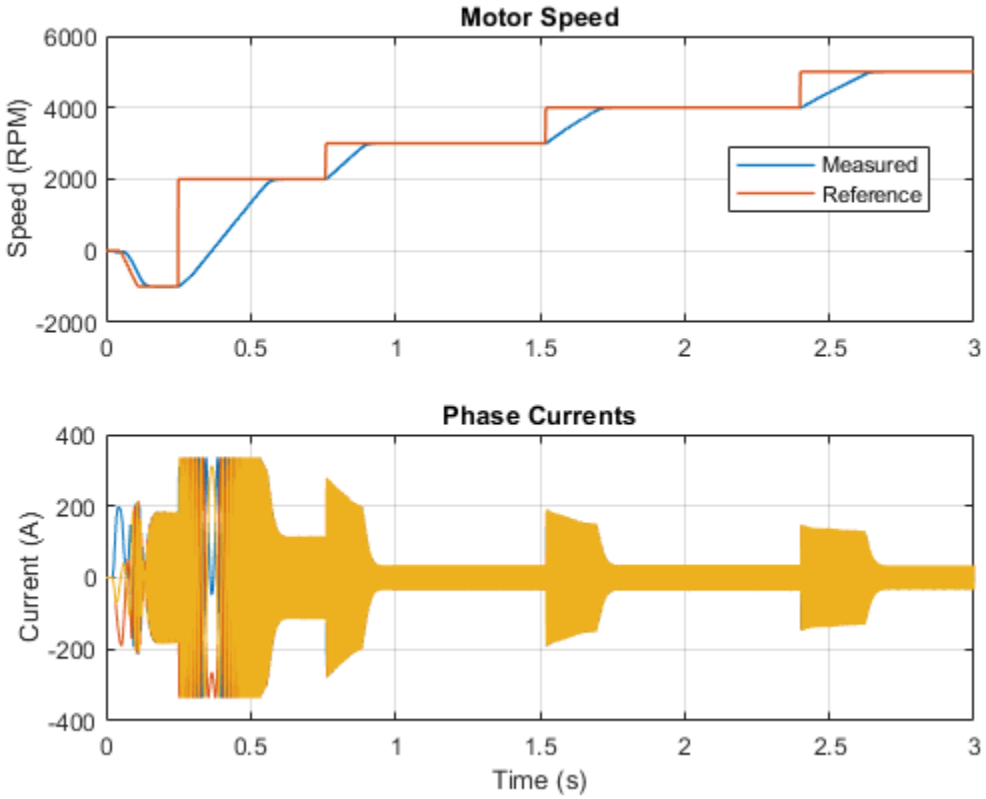
This example shows how to control the rotor angular velocity in an interior permanent magnet synchronous machine (IPMSM) based electrical-traction drive. For evaluation of the outer loop velocity controller, the inner loop current controller is replaced by a three-phase controlled current source. An ideal torque source provides the load. The Scopes subsystem contains scopes that allow you to see the simulation results. The Control subsystem includes a PI-based outer loop controller. During the three-seconds simulation, the angular velocity demand is -1000, 2000, 3000, 4000, and then 5000 rpm.

### Model




### Simulation Results from Simscape Logging

The plot below shows the requested and measured speed for the test, as well as the phase currents in the electric drive.



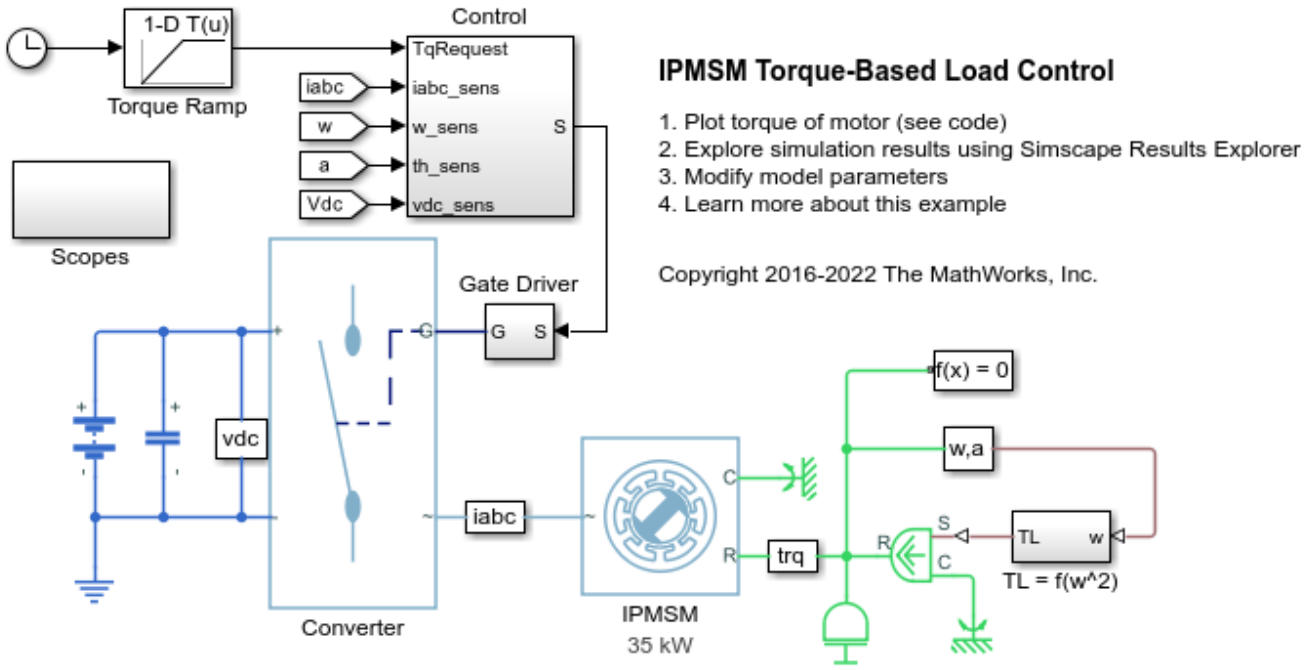

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 100 microseconds.

# IPMSM Torque-Based Load Control

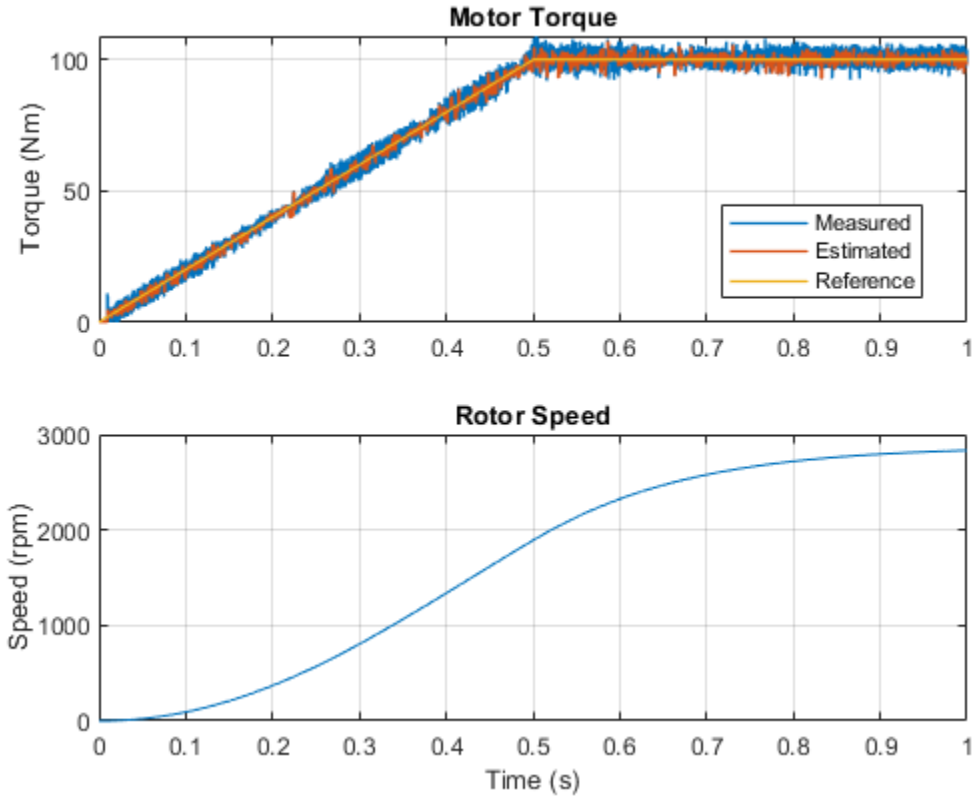
This example shows how to control the torque in an interior permanent magnet synchronous motor (IPMSM) based drive. A high-voltage battery feeds the IPMSM through a controlled three-phase inverter. A ramp of torque request is provided to the motor controller. The load torque is quadratically dependent on the rotor speed. The Control subsystem uses an open-loop approach to control the IPMSM torque and a closed-loop approach to control the current. At each sample instant, the torque request is converted to relevant current references. The current control is PI-based and uses a sample rate that is faster than the rate that is used for torque control. The task scheduling is designed in Stateflow®. The Scopes subsystem contains scopes that allow you to see the simulation results.

### Model




### Simulation Results from Simscape Logging

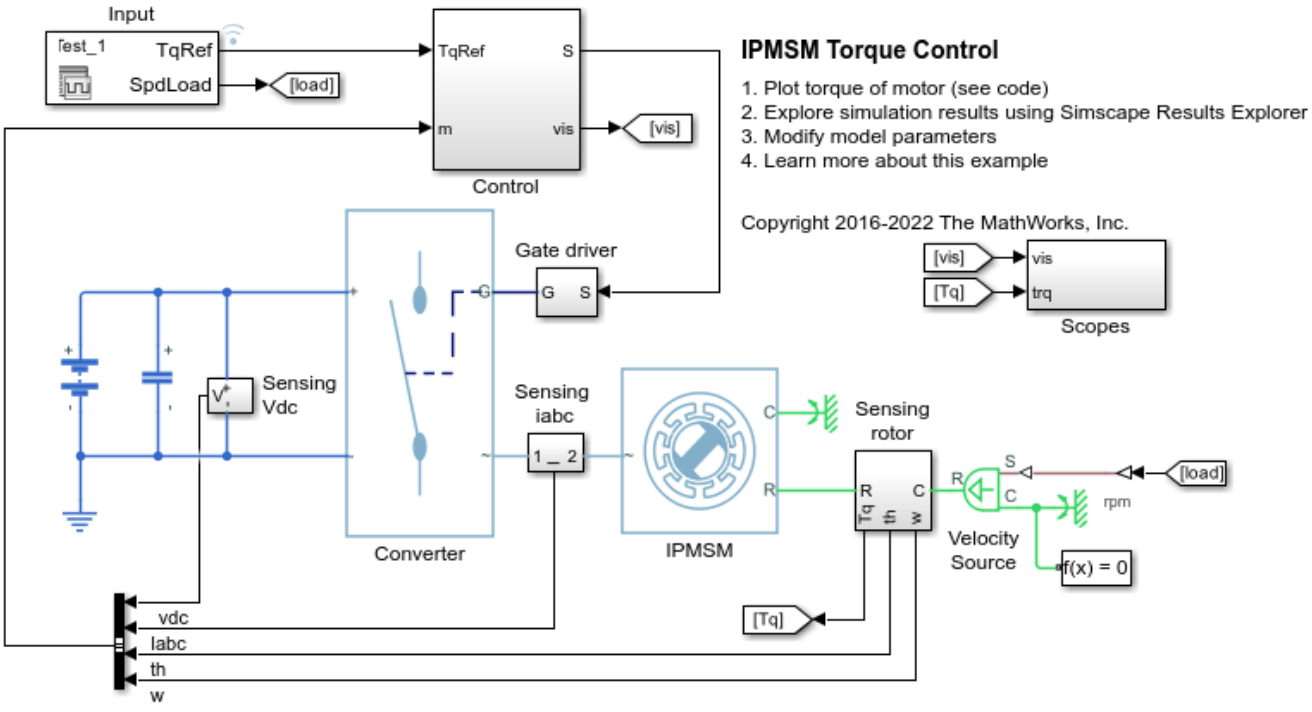
The plot below shows the requested and measured torque for the test, as well as the rotor speed in the electric drive.


Motor Torque
Torque (Nm)
100
50
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Measured
Estimated
Reference
Rotor Speed
Speed (rpm)
3000
2000
1000
0
Time (s)

# IPMSM Torque Control

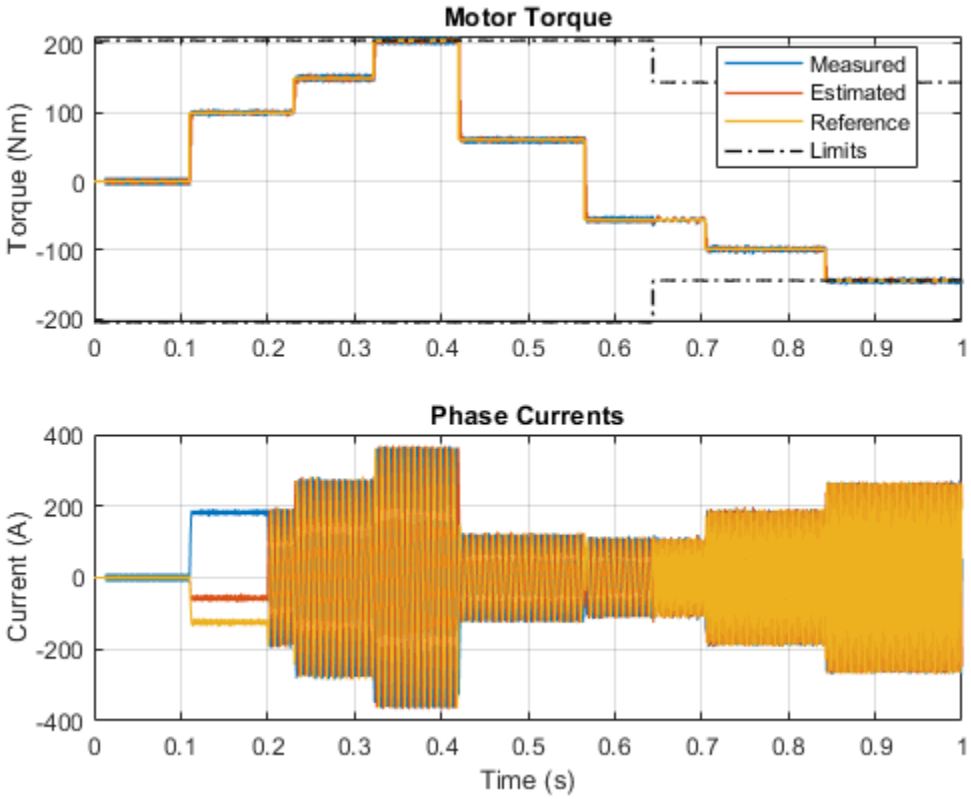
This example shows how to control the torque in an interior permanent magnet synchronous machine (IPMSM) based automotive electrical-traction drive. A high-voltage battery feeds the IPMSM through a controlled three- phase converter. The IPMSM operates in both motoring and generating modes according to the load. An ideal angular velocity source provides the load. The Control subsystem uses an open-loop approach to control the IPMSM torque and a closed-loop approach to control the current. At each sample instant, the torque request is converted to relevant current references. The current control is PI-based and uses a sample rate that is faster than the rate that is used for torque control. The simulation uses several torque steps in both motor and generator modes. The task scheduling is designed in Stateflow®. The Scopes subsystem contains scopes that allow you to see the simulation results.

### Model




### Simulation Results from Simscape Logging

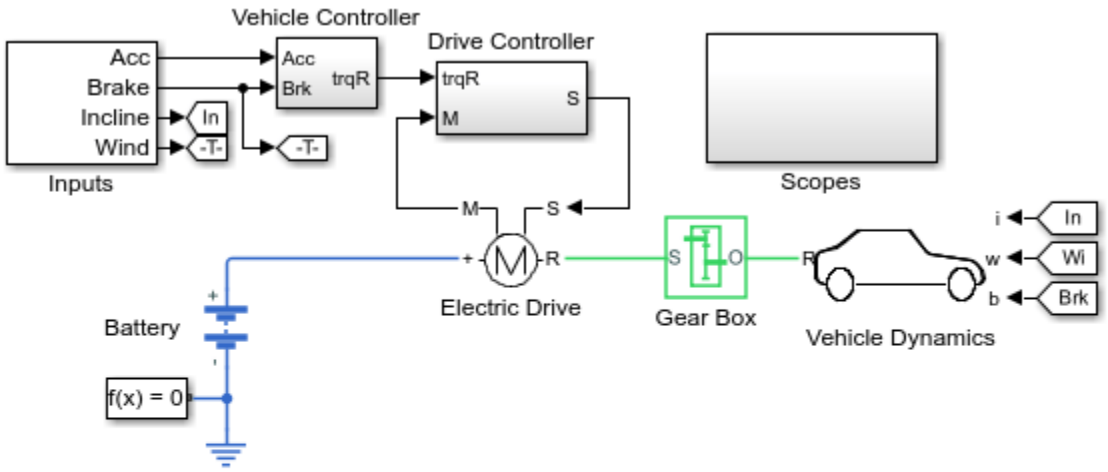
The plot below shows the requested and measured torque for the test, as well as the phase currents in the electric drive.


Motor Torque
Torque (Nm)
200
100
0
-100
-200
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Measured
Estimated
Reference
Limits
Phase Currents
Current (A)
400
200
0
-200
-400
Time (s)

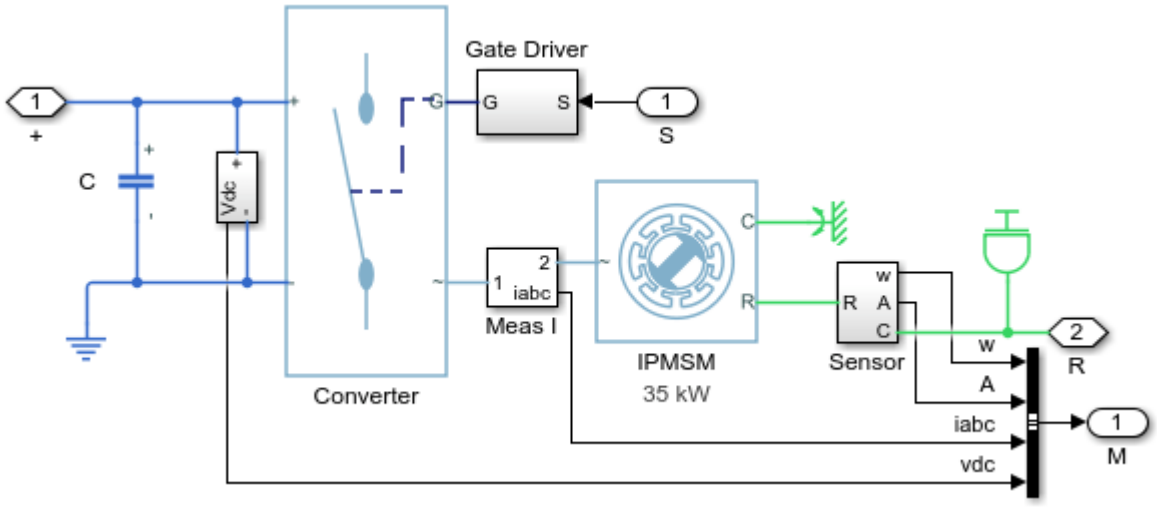
# IPMSM Torque Control in an Axle-Drive EV

This example shows an interior permanent magnet synchronous machine (IPMSM) propelling a simplified axle-drive electric vehicle. A high-voltage battery feeds the IPMSM through a controlled three-phase converter. The IPMSM operates in both motoring and generating modes. The vehicle transmission and differential are implemented using a fixed-ratio gear reduction model. The Vehicle Controller subsystem converts the driver inputs into a relevant torque command. The Drive Controller subsystem controls the torque of the IPMSM. The controller includes a multi-rate PI-based control structure. The rate of the open-loop torque control is slower than the rate of the closed-loop current control. The task scheduling for the controller is implemented as a Stateflow® state machine. The Scopes subsystem contains scopes that allow you to see the simulation results.

**Model**

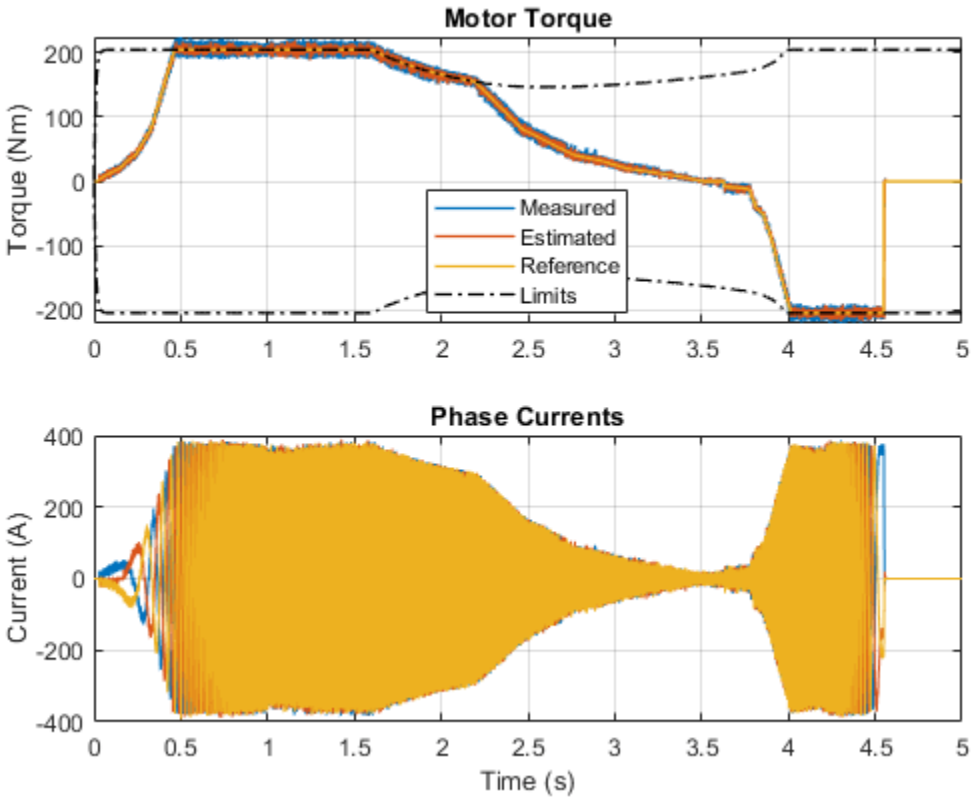


### Electric Drive Subsystem




### Simulation Results from Simscape Logging

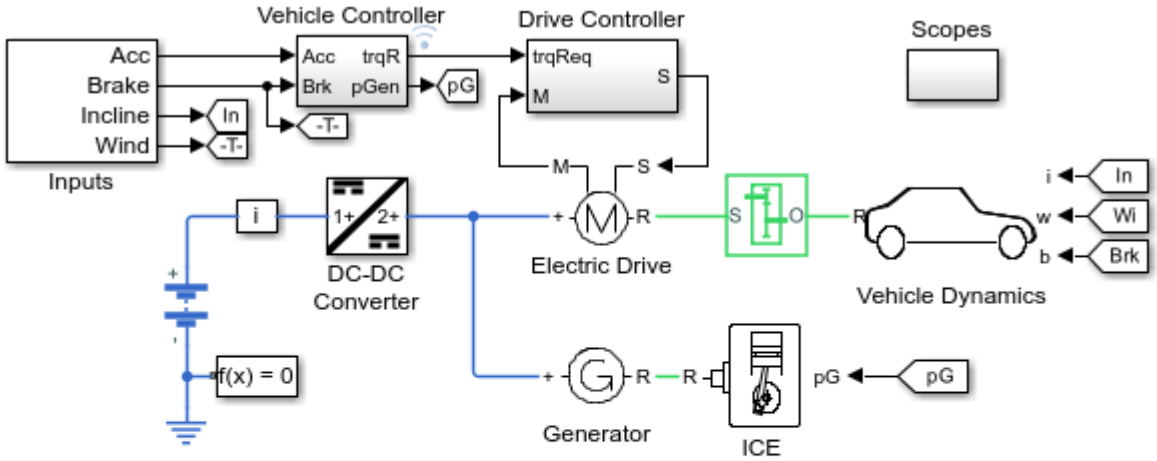
The plot below shows the requested and measured torque for the test, as well as the phase currents in the electric drive.



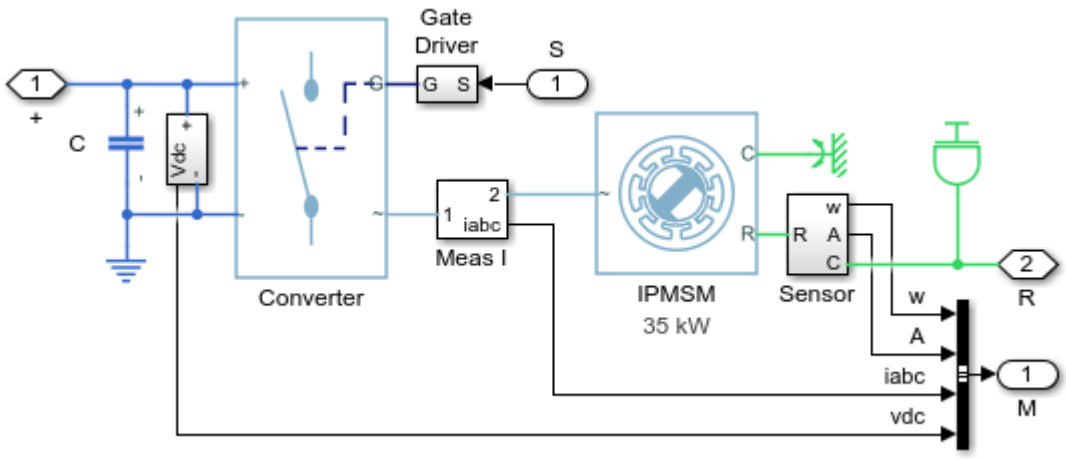
# IPMSM Torque Control in a Series HEV

This example shows an interior permanent magnet synchronous machine (IPMSM) propelling a simplified series hybrid electric vehicle (HEV). An ideal DCDC converter, connected to a high-voltage battery, feeds the IPMSM through a controlled three-phase converter. A combustion engine driven generator charges the high-voltage battery. The vehicle transmission and differential are implemented using a fixed-ratio gear-reduction model. The Vehicle Controller subsystem converts the driver inputs into relevant commands for the IPMSM and generator. The Drive Controller subsystem controls the torque of the IPMSM. The controller includes a multi-rate PI-based control structure. The rate of the open-loop torque control is slower than the rate of the closed-loop current control. The task scheduling for the controller is implemented as a Stateflow® state machine. The Scopes subsystem contains scopes that allow you to see the simulation results.

**Model**

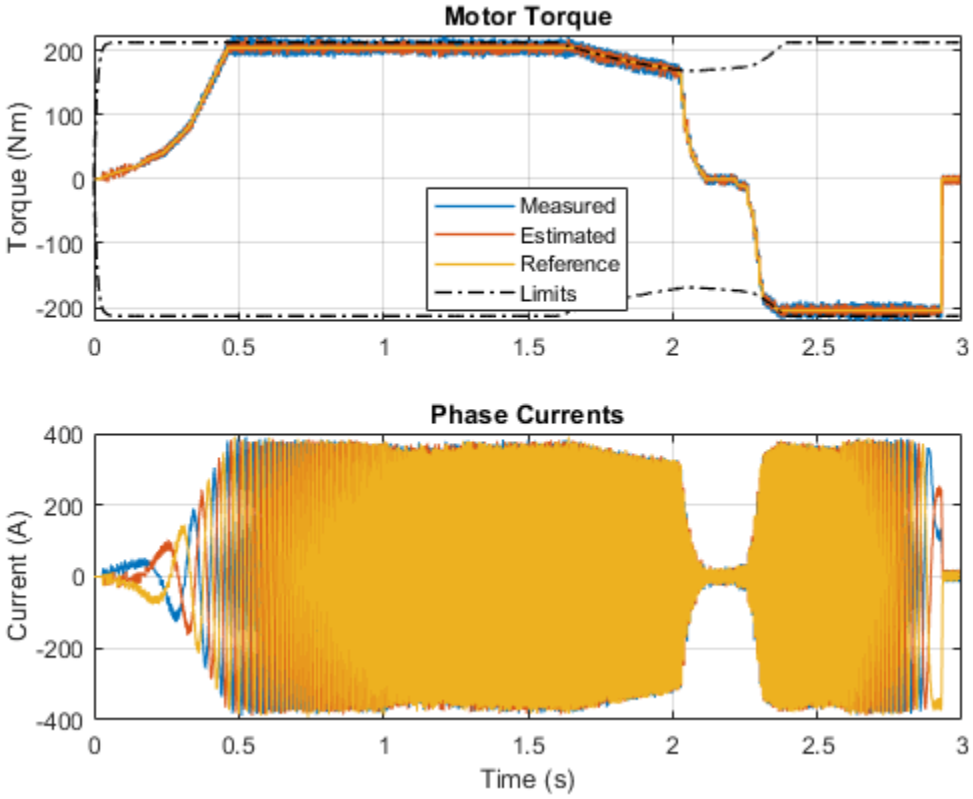


### Electric Drive Subsystem




### Simulation Results from Simscape Logging

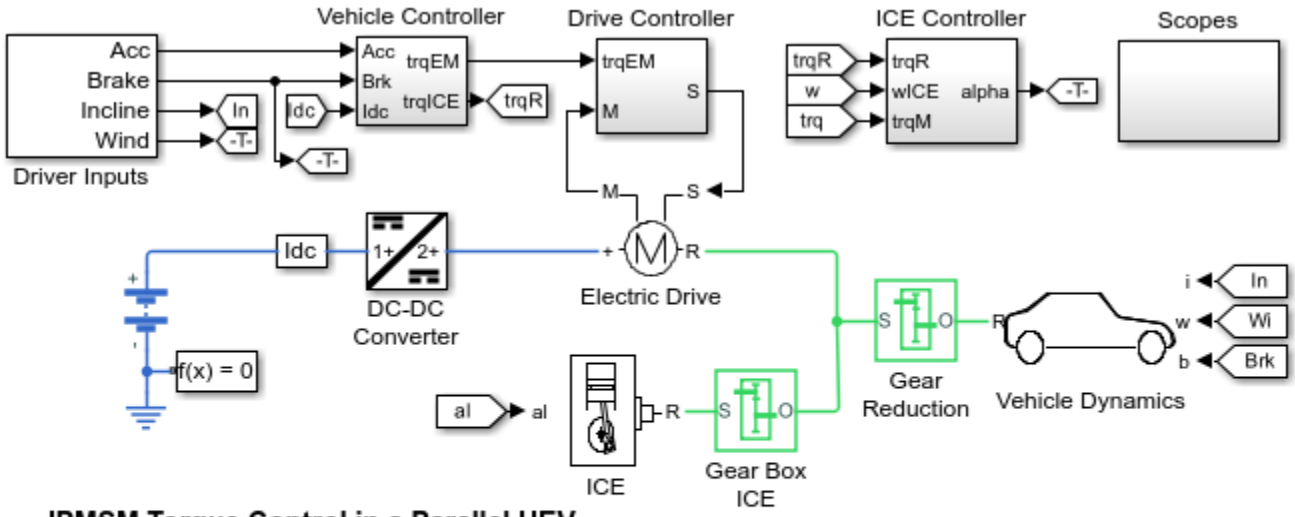
The plot below shows the requested and measured torque for the test, as well as the phase currents in the electric drive.



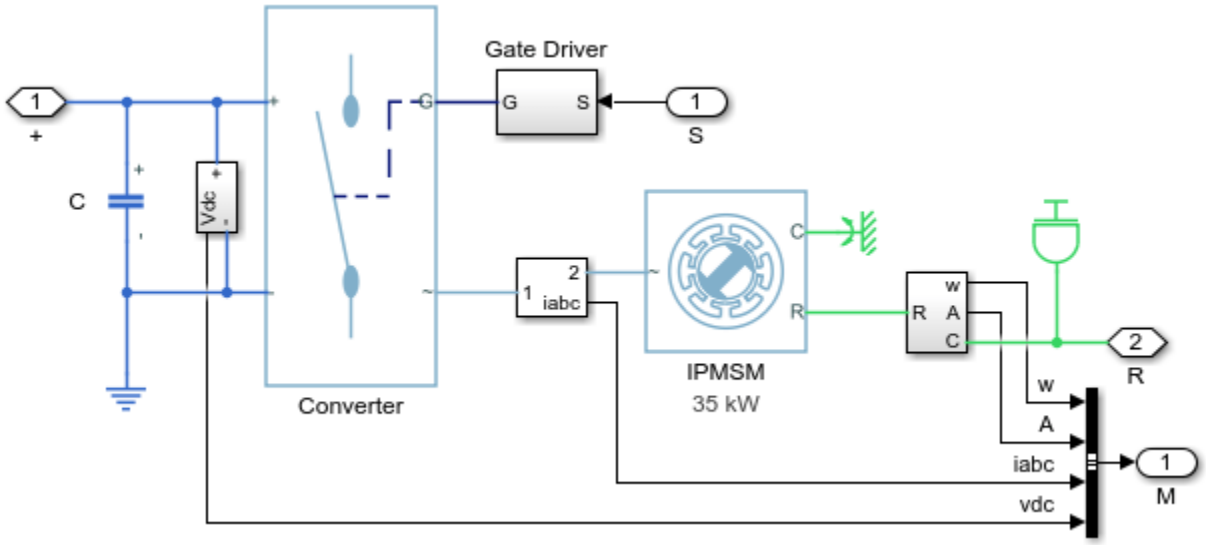
# IPMSM Torque Control in a Parallel HEV

This example shows a simplified parallel hybrid electric vehicle (HEV). An interior permanent magnet synchronous machine (IPMSM) and an internal combustion engine (ICE) provide the vehicle propulsion. The IPMSM operates in both motoring and generating modes. The vehicle transmission and differential are implemented using a fixed-ratio gear-reduction model. The Vehicle Controller subsystem converts the driver inputs into torque commands. The vehicle control strategy is implemented as a Stateflow® state machine. The ICE Controller subsystem controls the torque of the combustion engine. The Drive Controller subsystem controls the torque of the IPMSM. The Scopes subsystem contains scopes that allow you to see the simulation results.

**Model**

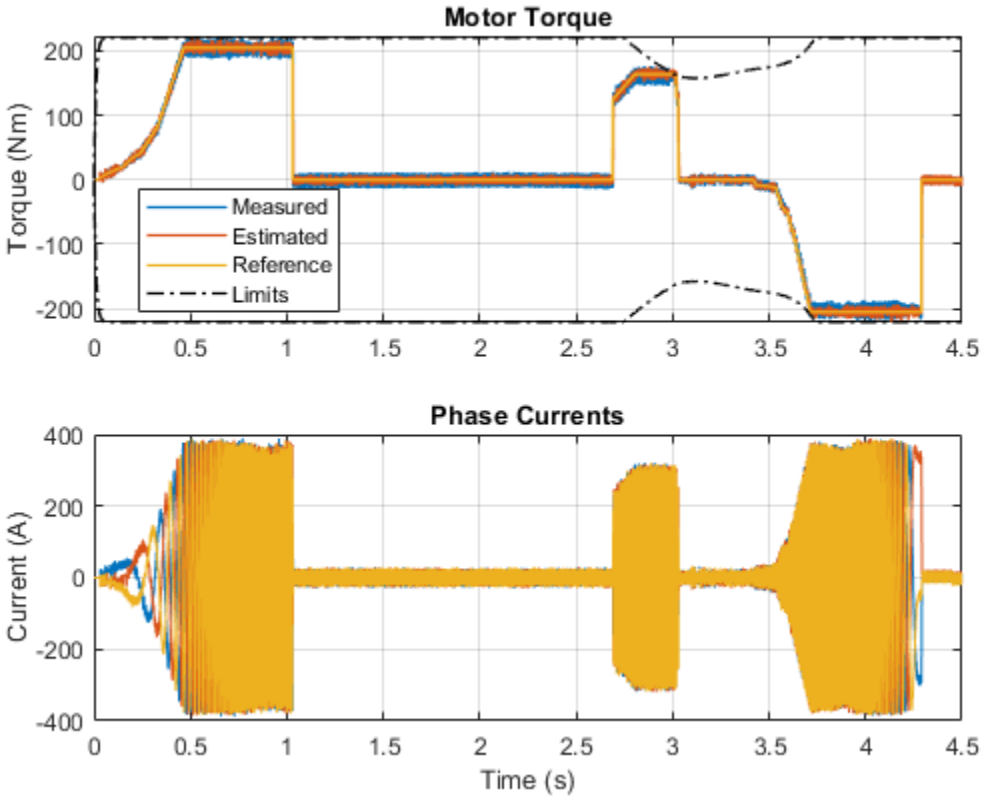


### Electric Drive Subsystem




### Simulation Results from Simscape Logging

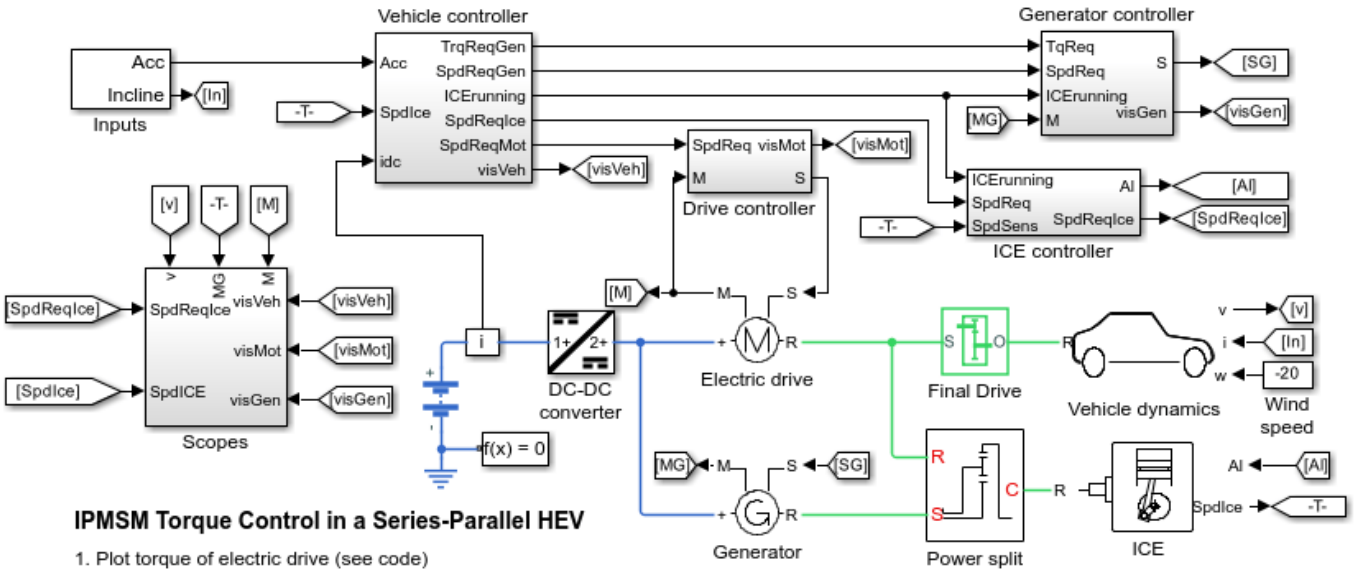
The plot below shows the requested and measured torque for the test, as well as the phase currents in the electric drive.


Motor Torque
Torque (Nm)
200
100
0
-100
-200
Measured
Estimated
Reference
Limits
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
Phase Currents
Current (A)
400
200
0
-200
-400
Time (s)

# IPMSM Torque Control in a Series-Parallel HEV

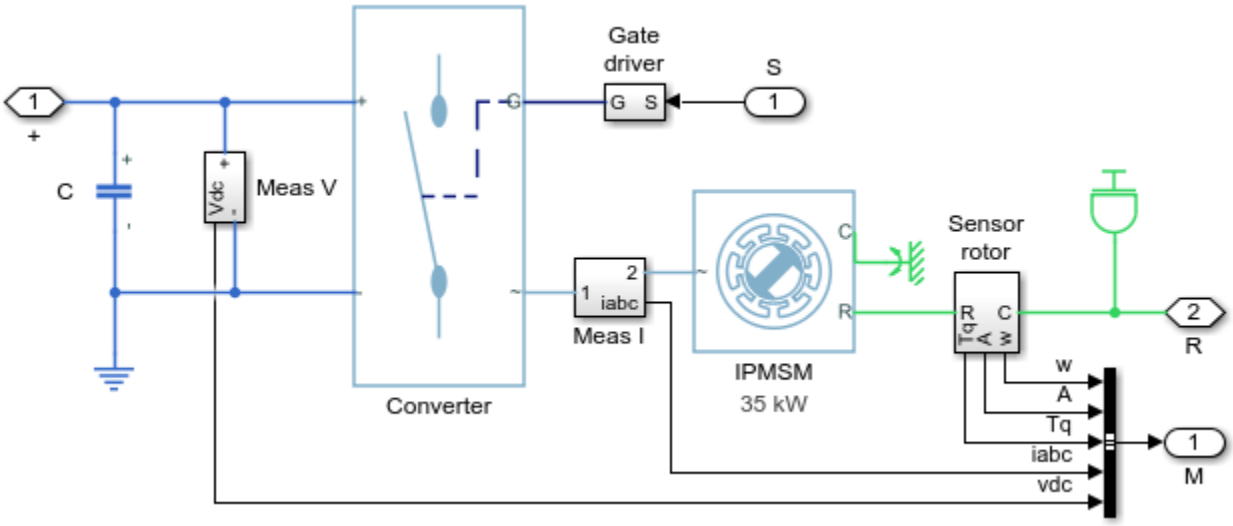
This example shows a simplified series-parallel hybrid electric vehicle (HEV). An interior permanent magnet synchronous machine (IPMSM) and an internal combustion engine (ICE) provide the vehicle propulsion. The ICE also uses electric generator to recharge the high-voltage battery during driving. The vehicle transmission and differential are implemented using a fixed-ratio gear-reduction model. The Vehicle Controller subsystem converts the driver inputs into torque commands. The vehicle control strategy is implemented as a Stateflow® state machine. The ICE Controller subsystem controls the torque of the combustion engine. The Generator Controller subsystem controls the torque of the electric generator. The Drive Controller subsystem controls the torque of the IPMSM. The Scopes subsystem contains scopes that allow you to see the simulation results.

### Model



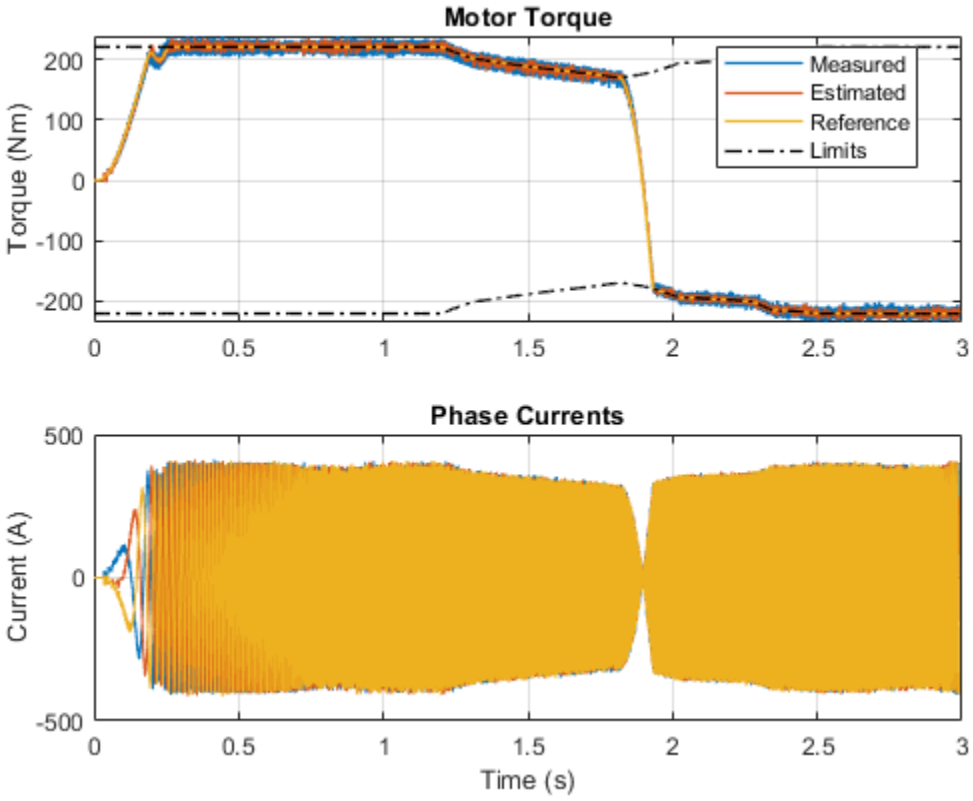

Copyright 2016-2022 The MathWorks, Inc.

### Electric Drive Subsystem




### Simulation Results from Simscape Logging

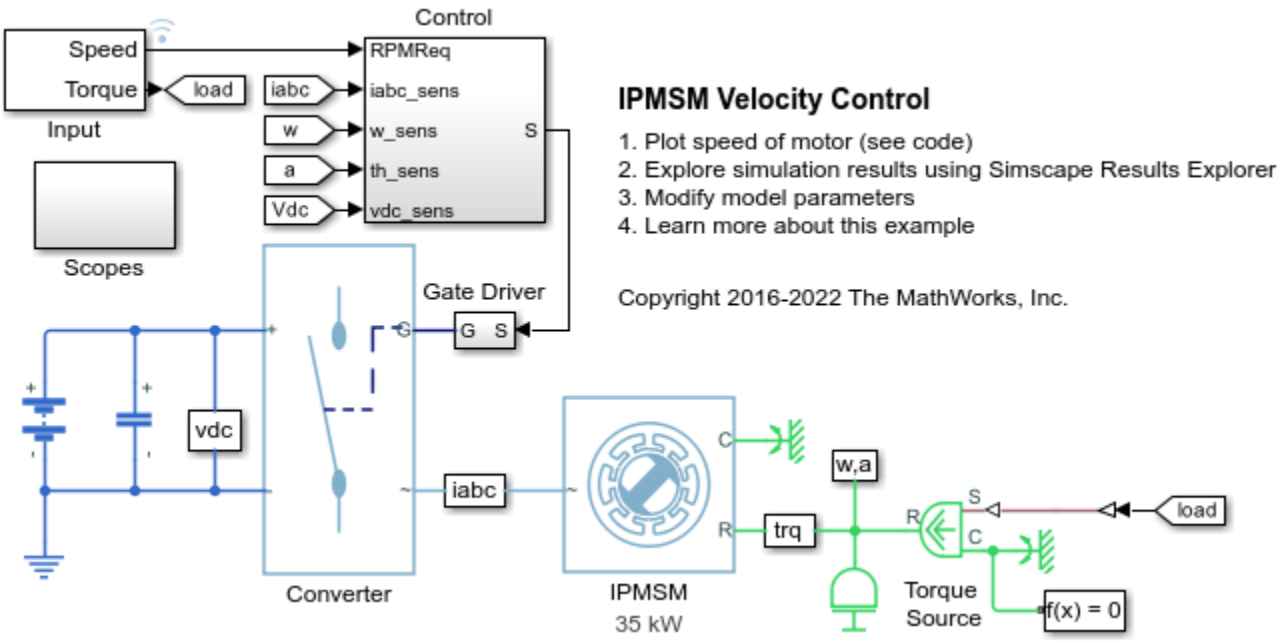
The plot below shows the requested and measured torque for the test, as well as the phase currents in the electric drive.



# IPMSM Velocity Control

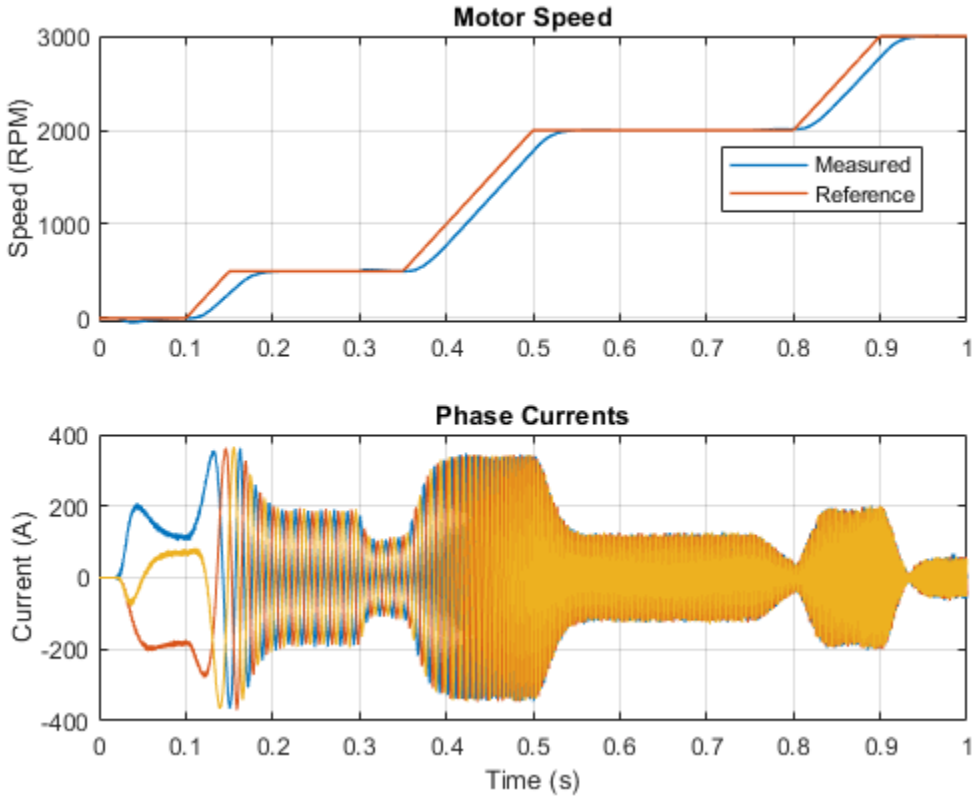
This example shows how to control the rotor angular velocity in an interior permanent magnet synchronous machine (IPMSM) based automotive electrical-traction drive. A high-voltage battery feeds the IPMSM through a controlled three-phase converter. The IPMSM operates in both motoring and generating modes according to the load. An ideal torque source provides the load. The Scopes subsystem contains scopes that allow you to see the simulation results. The Control subsystem includes a multi-rate PI-based cascade control structure which has an outer angular-velocity-control loop and two inner current-control loops. The task scheduling in the Control subsystem is implemented as a Stateflow® state machine. During the one-second simulation, the angular velocity demand is 0 rpm, 500 rpm, 2000 rpm, and then 3000 rpm.

### Model




### Simulation Results from Simscape Logging

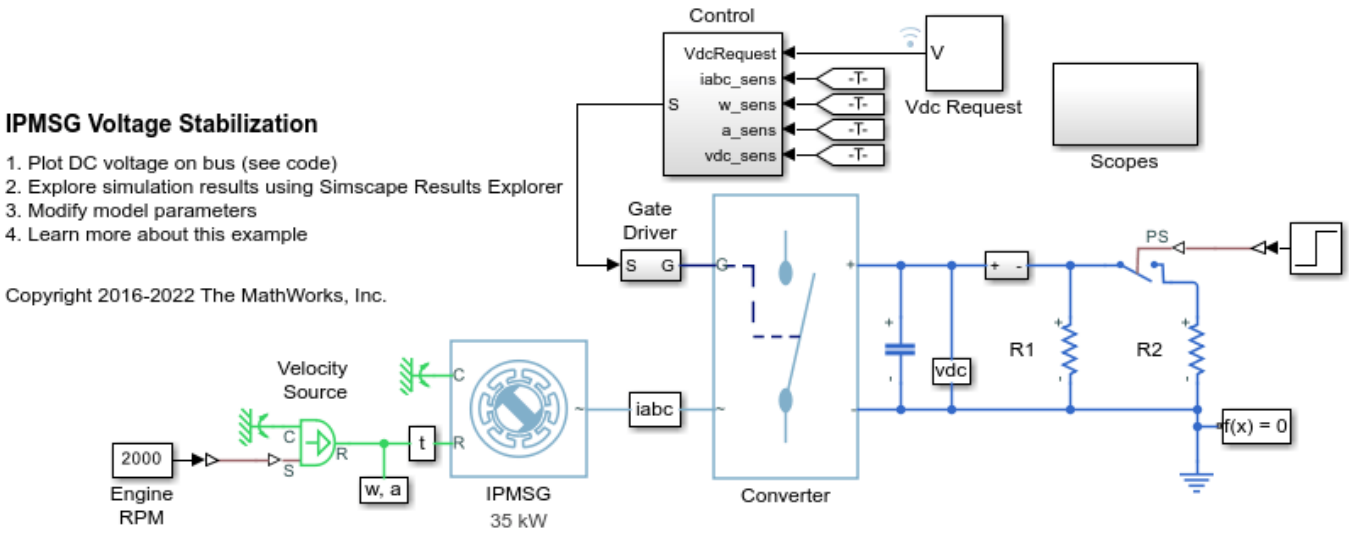
The plot below shows the requested and measured speed for the test, as well as the phase currents in the electric drive.


Motor Speed
Speed (RPM)
3000
2000
1000
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Measured
Reference
Phase Currents
Current (A)
400
200
0
-200
-400
Time (s)

# IPMSG Voltage Stabilization

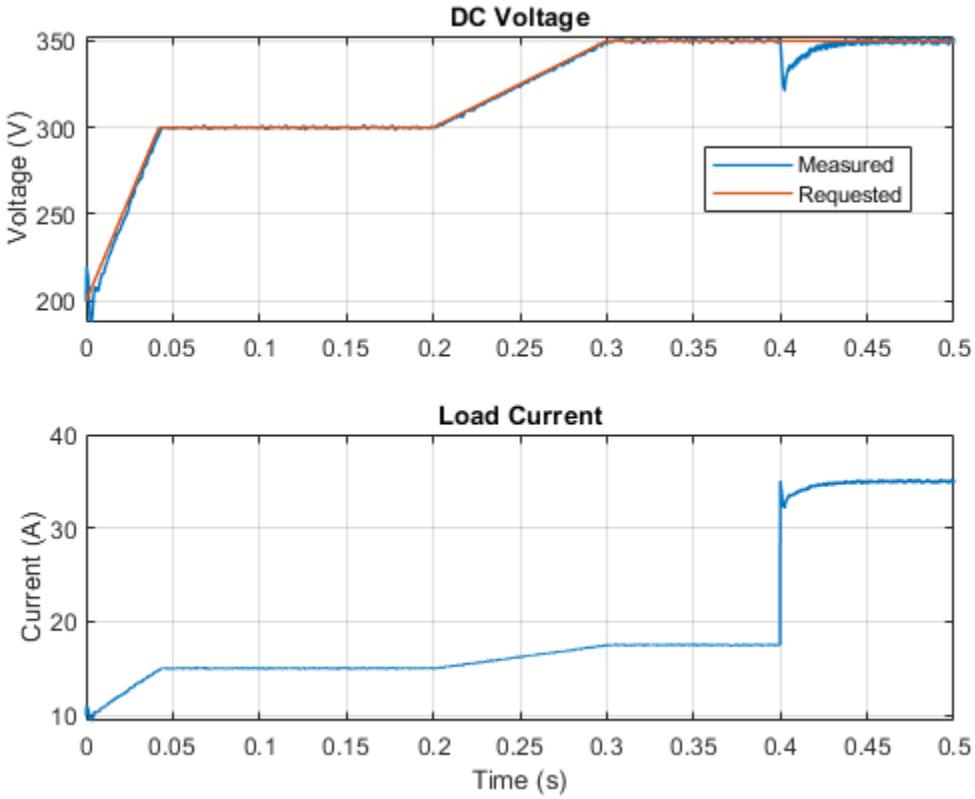
This example shows how to control an Interior Permanent Magnet Synchronous Generator (IPMSG) based low voltage generator system for a hybrid electric vehicle (HEV). The Control subsystem includes a multi-rate PI-based cascade control structure which has an outer voltage-control loop and two inner current-control loops. The task scheduling in the Control subsystem is implemented as a Stateflow® state machine. The Scopes subsystem contains scopes that allow you to see the simulation results. An ideal angular velocity source, which represents a combustion engine, drives the IPMSG. The IPMSG supplies low-voltage power to loads R1 and R2. At t = 0.4 seconds, the switch closes, increasing the load.

### Model




### Simulation Results from Simscape Logging

The plot below shows how well the system maintains the requested DC voltage as the load changes.


DC Voltage
Voltage (V)
350
300
250
200
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5
Measured
Requested
Load Current
Current (A)
40
30
20
10
Time (s)

# Marine Full Electric Propulsion Power System

This example shows a representative marine half-ship electrical power system with base load, hotel load, bow thrusters and electric propulsion.

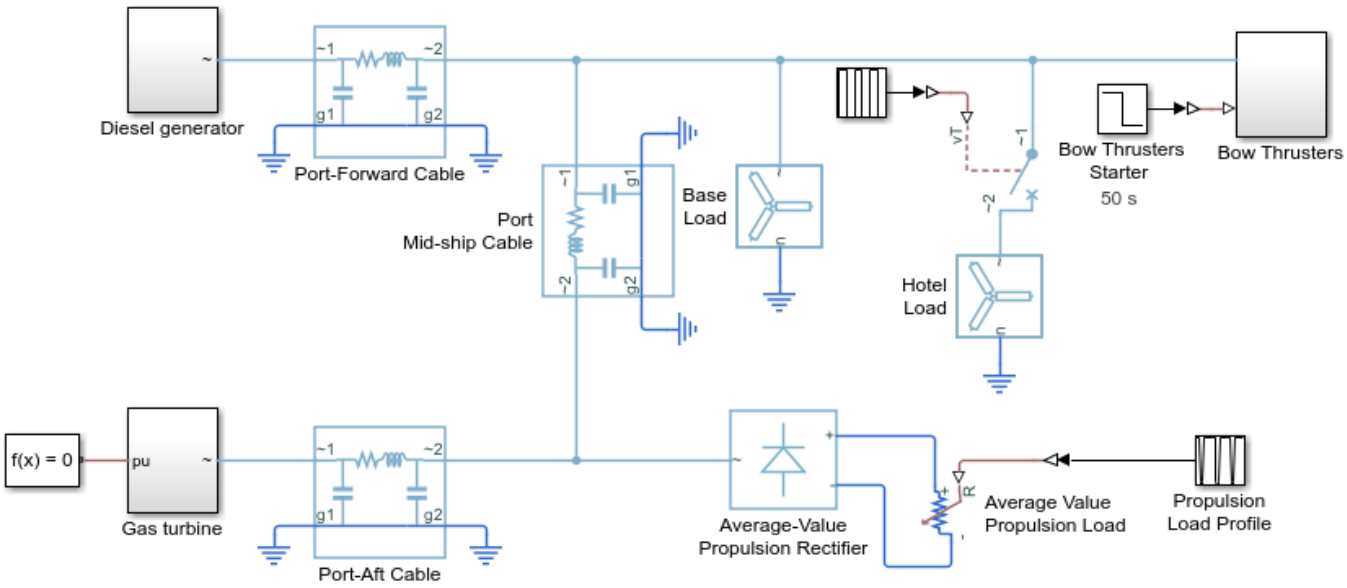
The principal components of the system are:

- 30 MVA Gas Turbine, Round Rotor Alternator
- 5 MVA Diesel Generator, Salient Pole Alternator
- 11.5 MVA Base Load
- 6 MVA Switched Hotel Load
- 20 MVA Average-Value Propulsion Rectifier
- 1 MVA Direct Online Start, Squirrel Cage Bow Thrusters

The sequence of events during the simulation is:

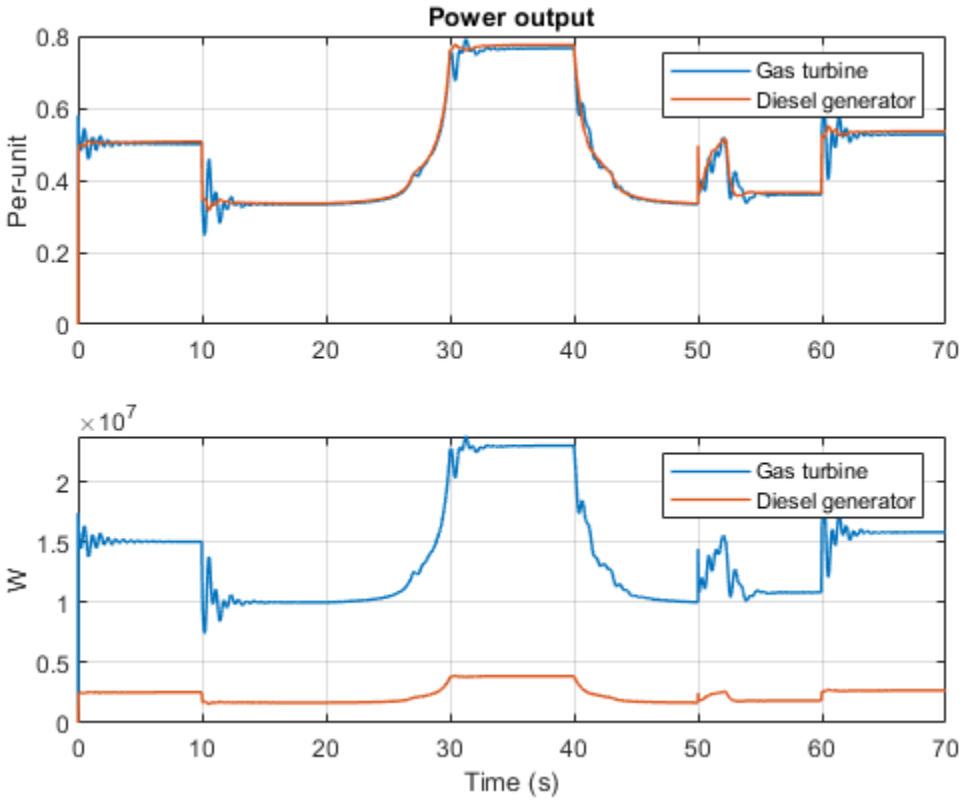
- 10 seconds - Hotel Load disconnected
- 20 seconds - Propulsion begins ramping up
- 30 seconds - Full power ahead
- 40 seconds - Propulsion begins ramping down
- 50 seconds - Bow Thrusters start up
- 60 seconds - Hotel Load connected

### Model



### Simulation Results from Simscape Logging

The plot below shows the diesel generator and gas turbine power outputs.



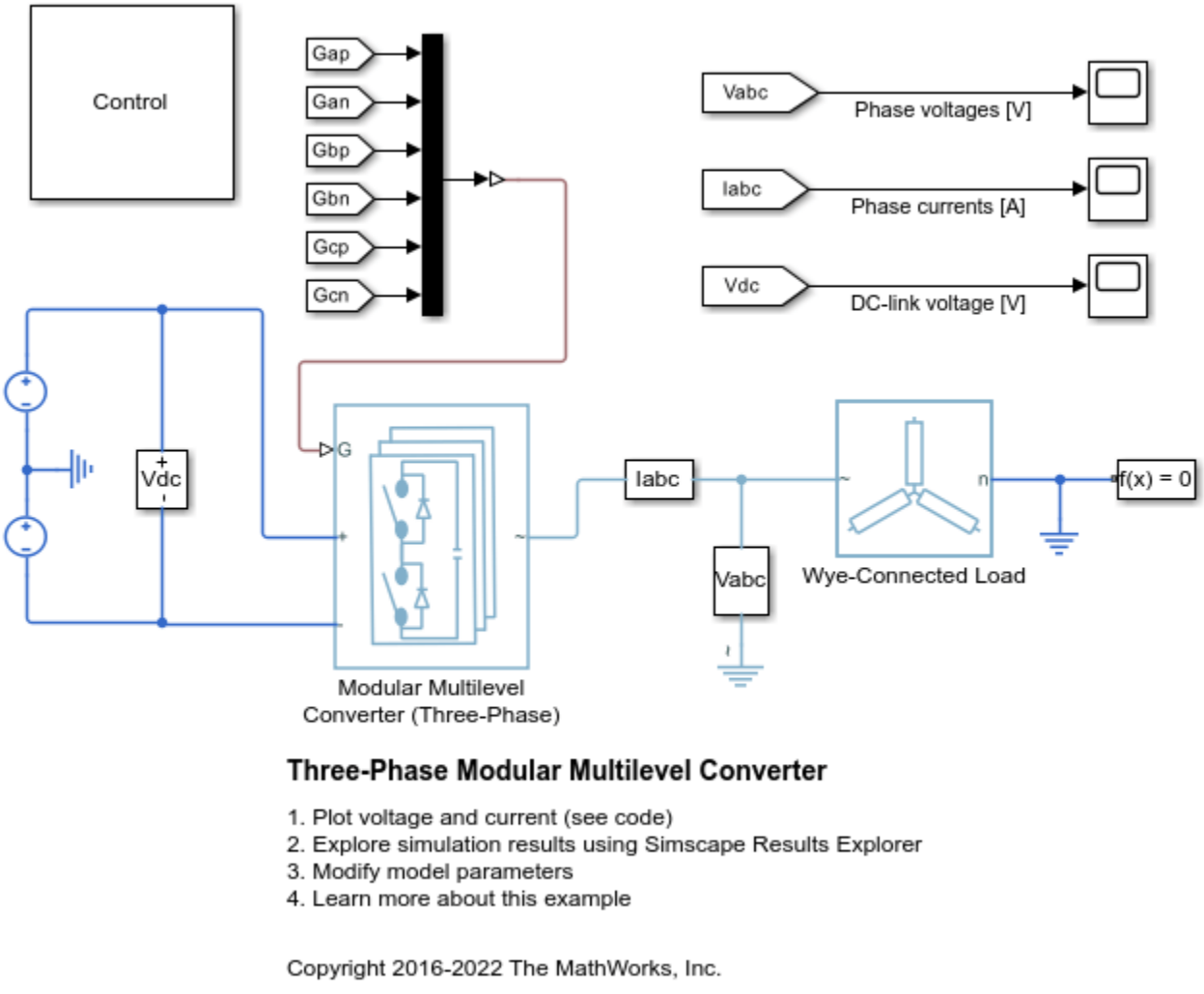

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 500 microseconds.

# Three-Phase Modular Multilevel Converter

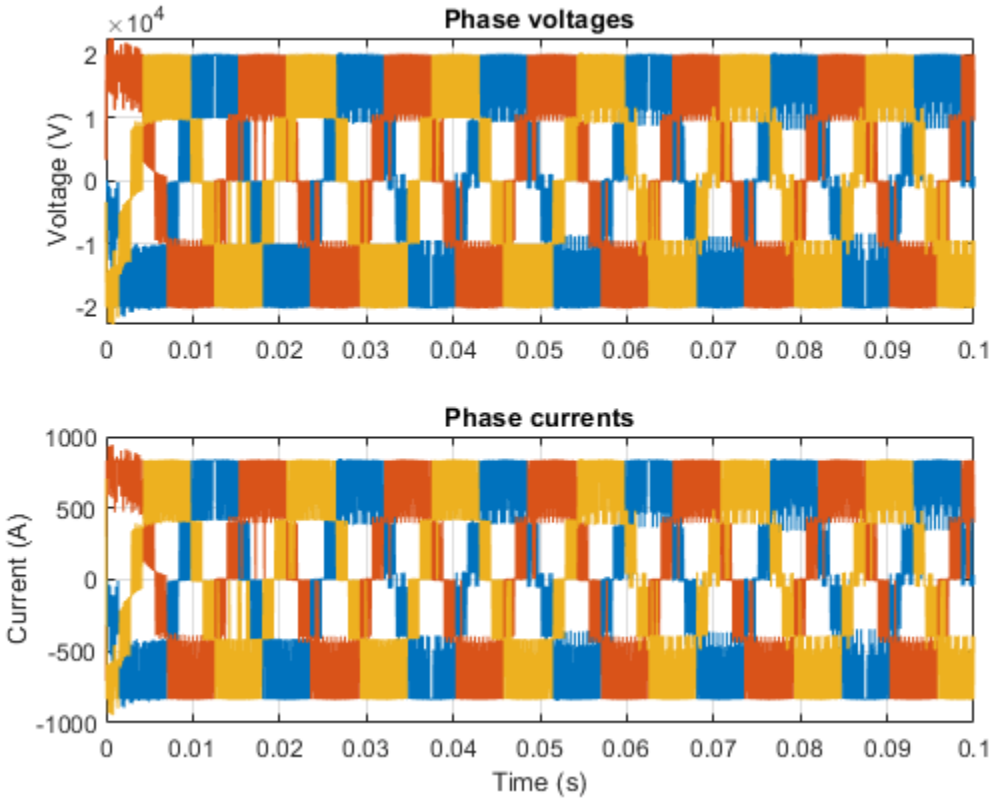
This example shows how to control in open-loop a three-phase modular multilevel converter (MMC). Each MMC arm consists of four half-bridge submodules. A wye-connected series RLC structure provides the load to the system.

### Model




### Simulation Results from Simscape Logging

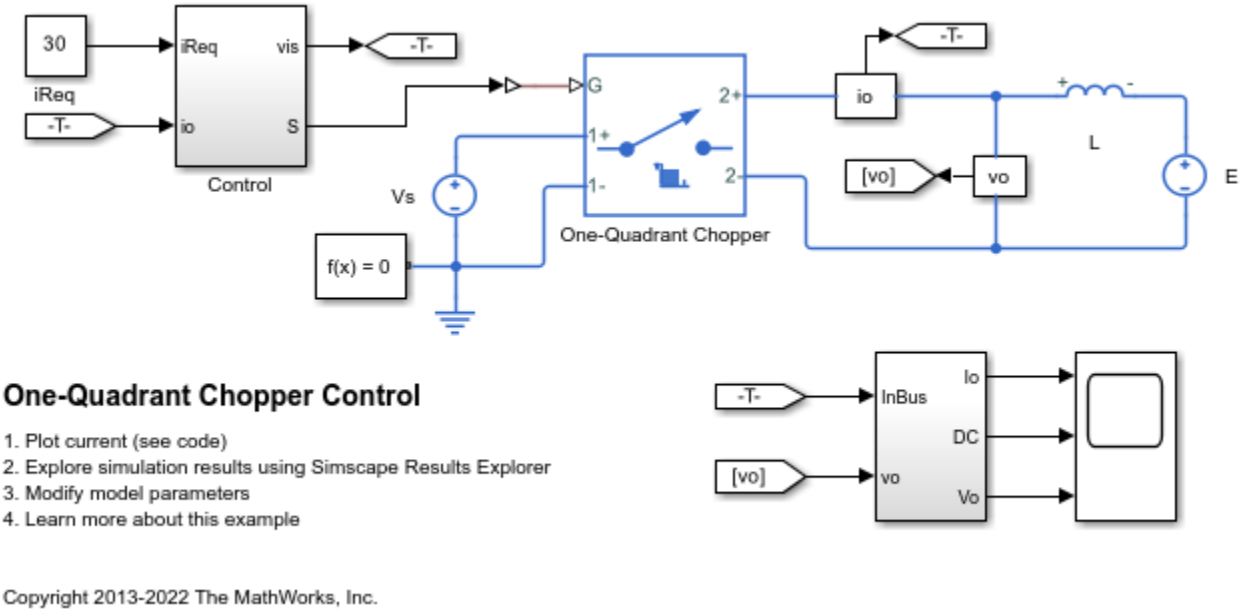
The plot below shows the phase voltages and currents.


Phase voltages
×10^4
Voltage (V)
2
1
0
-1
-2
0
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08
0.09
0.1
Phase currents
Current (A)
1000
500
0
-500
-1000
Time (s)

# One-Quadrant Chopper Control

This example shows how to control a one-quadrant chopper. The Control subsystem implements a simple PI-based control algorithm for controlling the output current.

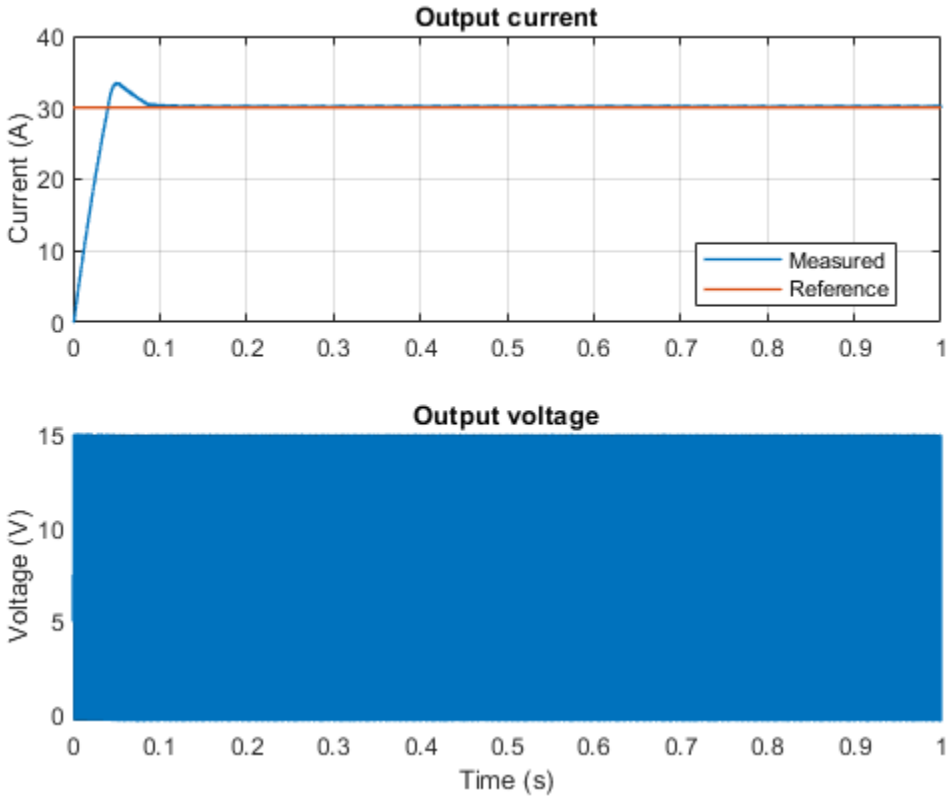
### Model




Copyright 2013-2022 The MathWorks, Inc.

### Simulation Results from Simscape Logging

The plot below shows the requested and measured current for the test and the output voltage in the circuit.



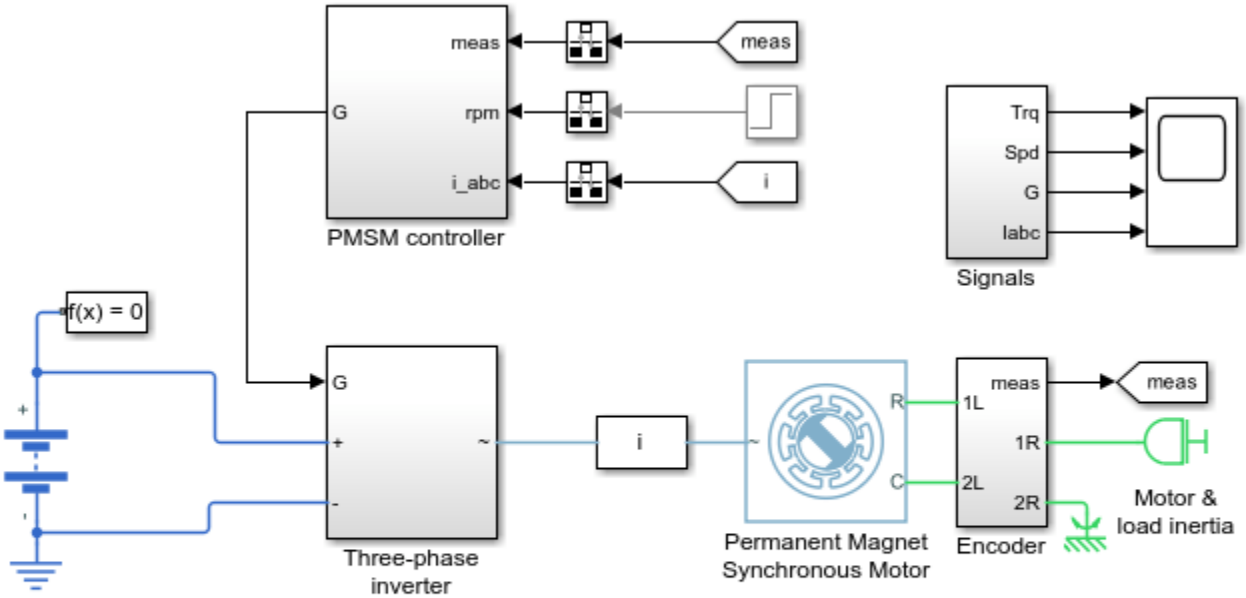

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 10 microseconds.

# Three-Phase PMSM Drive

This example shows a Permanent Magnet Synchronous Machine (PMSM) in wye-wound and delta-wound configuration and an inverter sized for use in a typical hybrid vehicle. The inverter is connected directly to the vehicle battery, but you can also implement a DC-DC converter stage in between. You can use this model to design the PMSM controller, by selecting the architecture and gains to achieve the desired performance. To check the timing of IGBT turn-on and turn-off, you can replace the IGBT devices with the more detailed N-Channel IGBT block. For complete vehicle modeling, you can use the Motor & Drive (System Level) block to abstract the PMSM, inverter, and controller with an energy-based model. The Gmin resistor provides a very small conductance to ground that improves the numerical properties of the model when using a variable-step solver.

**Model**




**Three-Phase PMSM Drive**

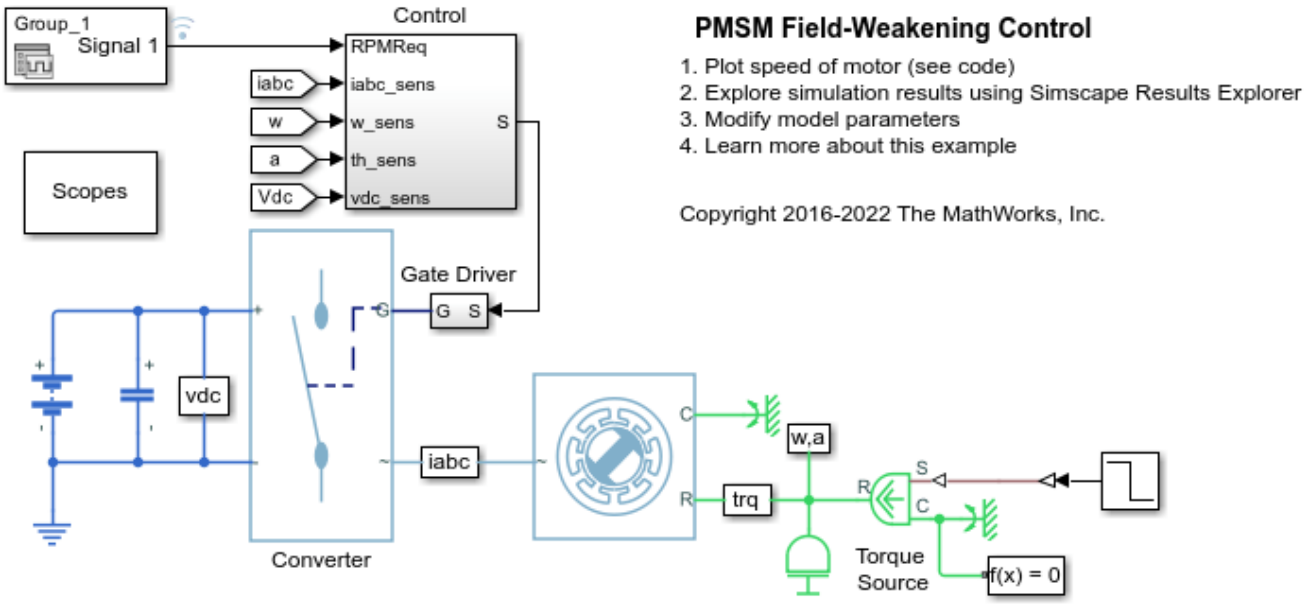
1. Explore simulation results using Simscape Results Explorer2. Motor winding: Wye-wound, Delta-wound (see code)
3. Learn more about this example

Copyright 2013-2022 The MathWorks, Inc.

# PMSM Field-Weakening Control

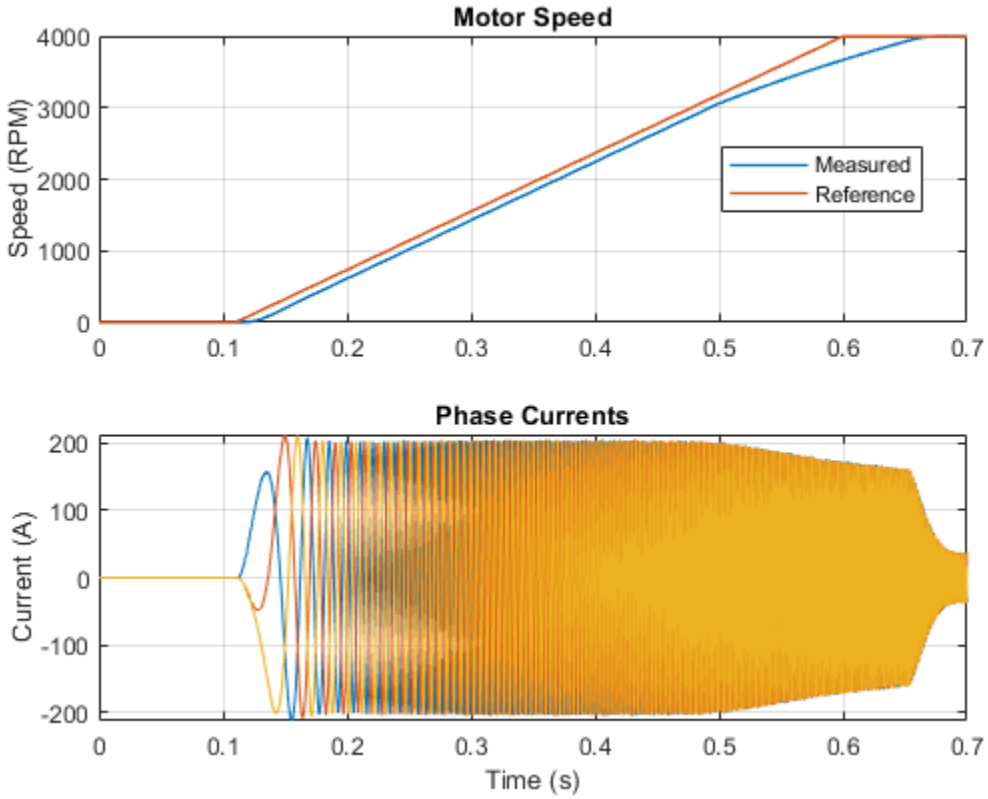
This example shows how to control the rotor angular velocity above the nominal velocity in a permanent magnet synchronous machine (PMSM) based electrical-traction drive. A high-voltage battery feeds the PMSM through a controlled three-phase converter. The Control subsystem includes a multi-rate PI-based cascade control structure which has an outer angular-velocity-control loop and two inner current-control loops. The velocity controller generates a torque reference. A zero d-axis controller converts this torque reference to current references. A field weakening controller adjusts the current references to satisfy the voltage constraints above the nominal velocity. A Stateflow® state machine implements the task scheduling in the Control subsystem. During the 0.7 s simulation, the angular velocity demand ramps up from 0 to 4000 rpm. The Scopes subsystem contains scopes that allow you to see the simulation results.

### Model




### Simulation Results from Simscape Logging

The plot below shows the requested and measured speed for the test, as well as the phase currents in the electric drive.


Motor Speed
Speed (RPM)
4000
3000
2000
1000
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
Measured
Reference
Phase Currents
Current (A)
200
100
0
-100
-200
Time (s)

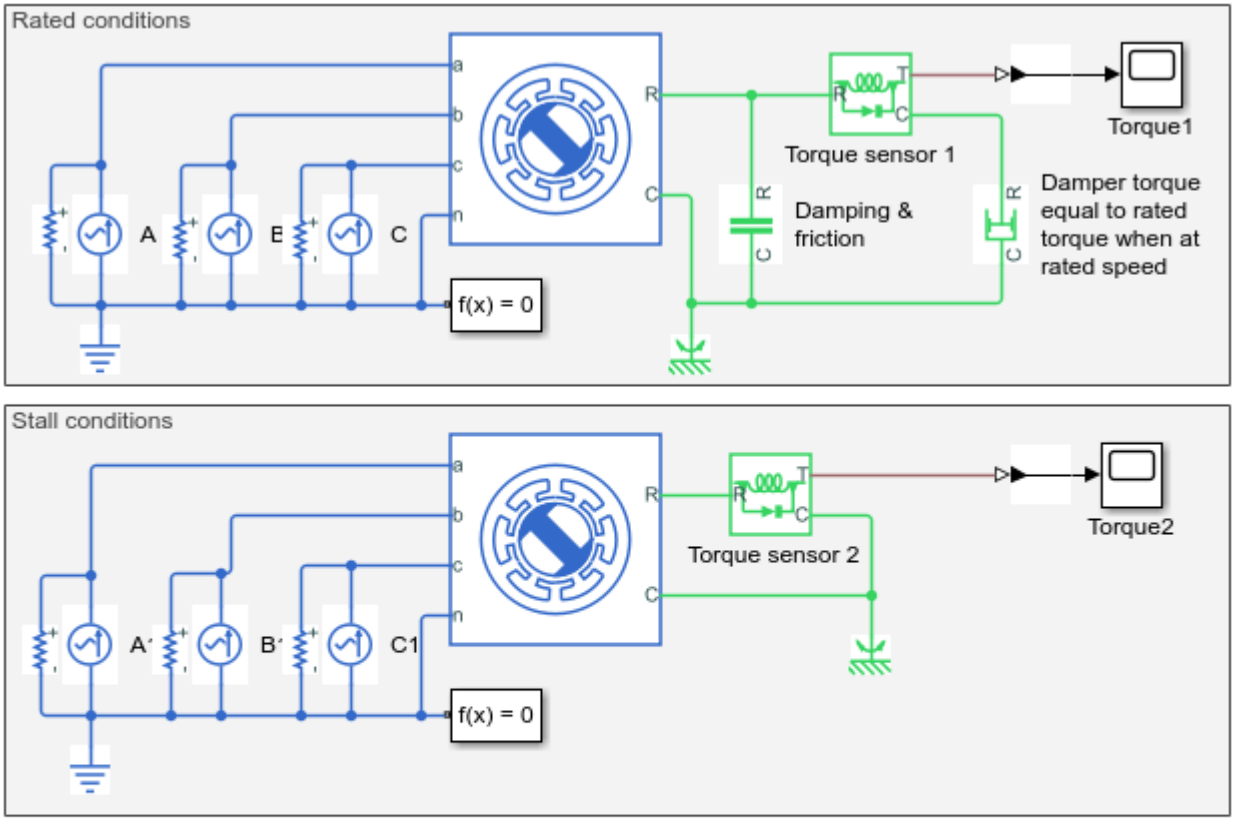
# PMSM Parameterization from Datasheet

This example shows two test harnesses that add confidence that a PMSM is correctly parameterized from a datasheet. It also calculates motor efficiency at rated speed and torque.

### Model

The rated conditions test harness below uses datasheet values for damping and friction plus it adds an additional linear damper that is sized to reach rated torque when rotating at rated speed. The three current sources deliver rated current. This test harness verifies that the motor is capable of delivering the rated torque at rated speed and also determines the motor efficiency.

The stall conditions test harness replicates a stall event by effectively shorting the mechanical R and C connections via the torque sensor. The torque measured by the torque sensor is checked against the datasheet value for the stall torque. An incorrect stall torque could indicate an incorrect value for the torque constant, perhaps because the measurement convention has not been defined on the datasheet.



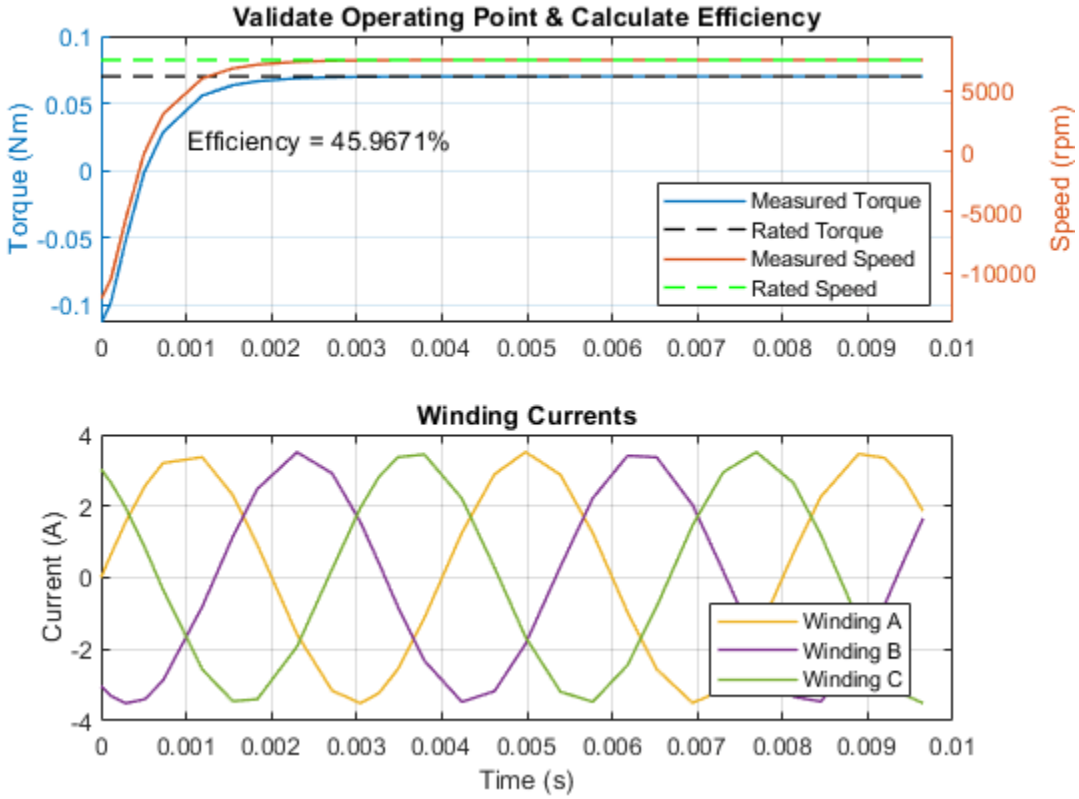

**PMSM Parameterization from Datasheet**

1. Plot torque and calculate efficiency at rated load (see code)
2. Confirm stall torque (see code)
3. Set parameters for model
4. Explore simulation results using Simscape Results Explorer
5. Learn more about this example

Copyright 2013-2022 The MathWorks, Inc.

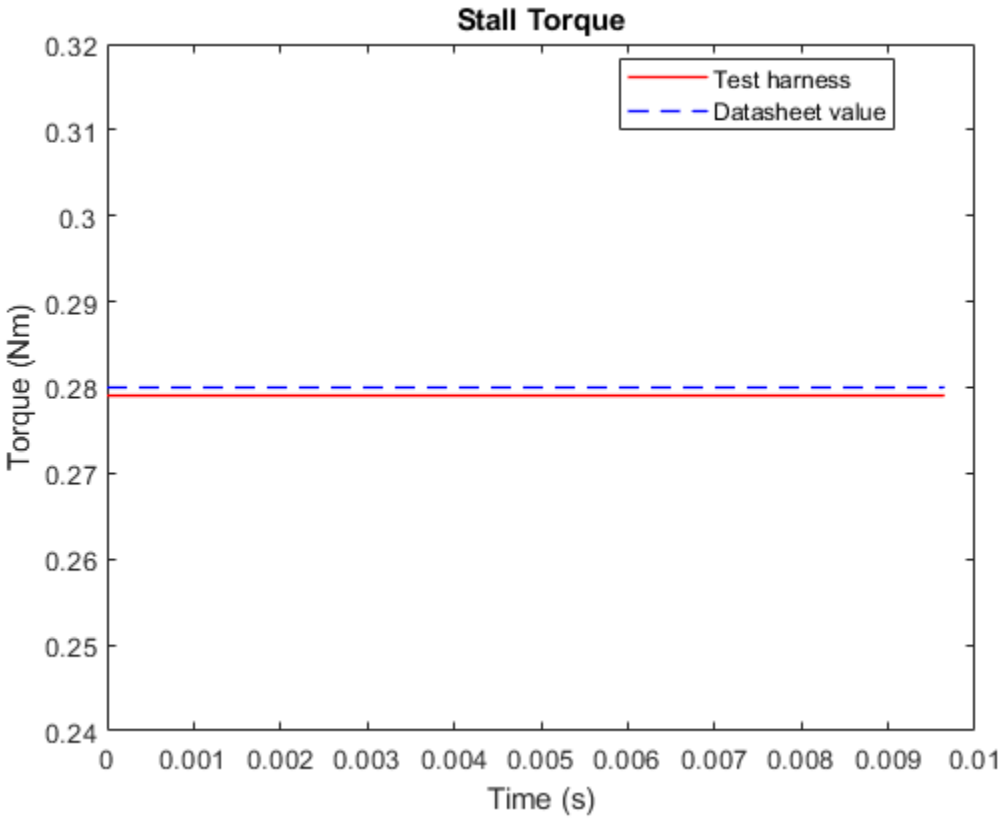
### Rated Conditions

The plot below is generated from the rated conditions test harness and verifies that the motor is capable of delivering the rated torque at rated speed and also determines the motor efficiency.




### Stall Conditions

The plot below confirms that the simulated stall torque is equal to the datasheet stall torque.

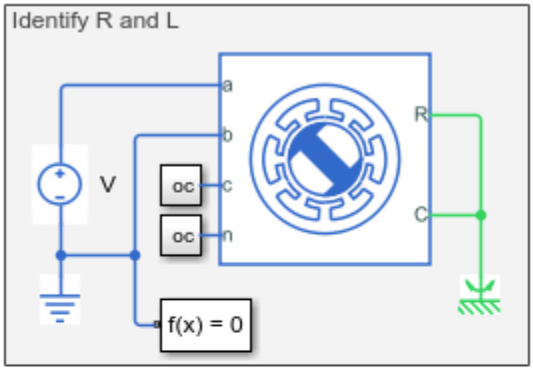

Stall Torque
Test harness
Datasheet value
Torque (Nm)
Time (s)
0.32
0.31
0.3
0.29
0.28
0.27
0.26
0.25
0.24
0
0.001
0.002
0.003
0.004
0.005
0.006
0.007
0.008
0.009
0.01

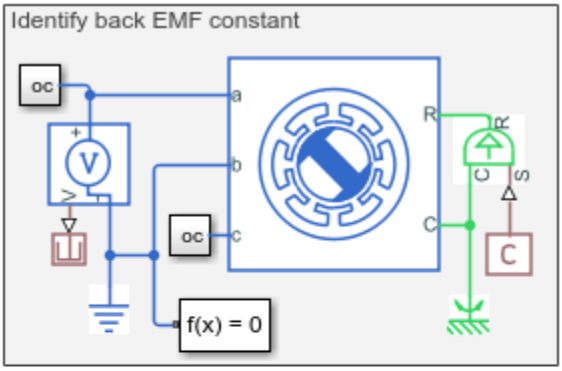
# PMSM Parameterization from Measurements

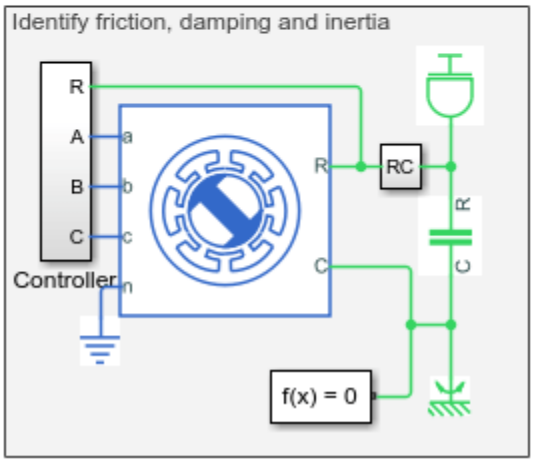
This example shows how to identify PMSM motor parameters from experimental measurements.

### Model

The parameter identification requires three separate tests that are represented here by three models as shown below. In this example we assume known values for the motor parameters and then show that we can reproduce them from the simulated identification tests.



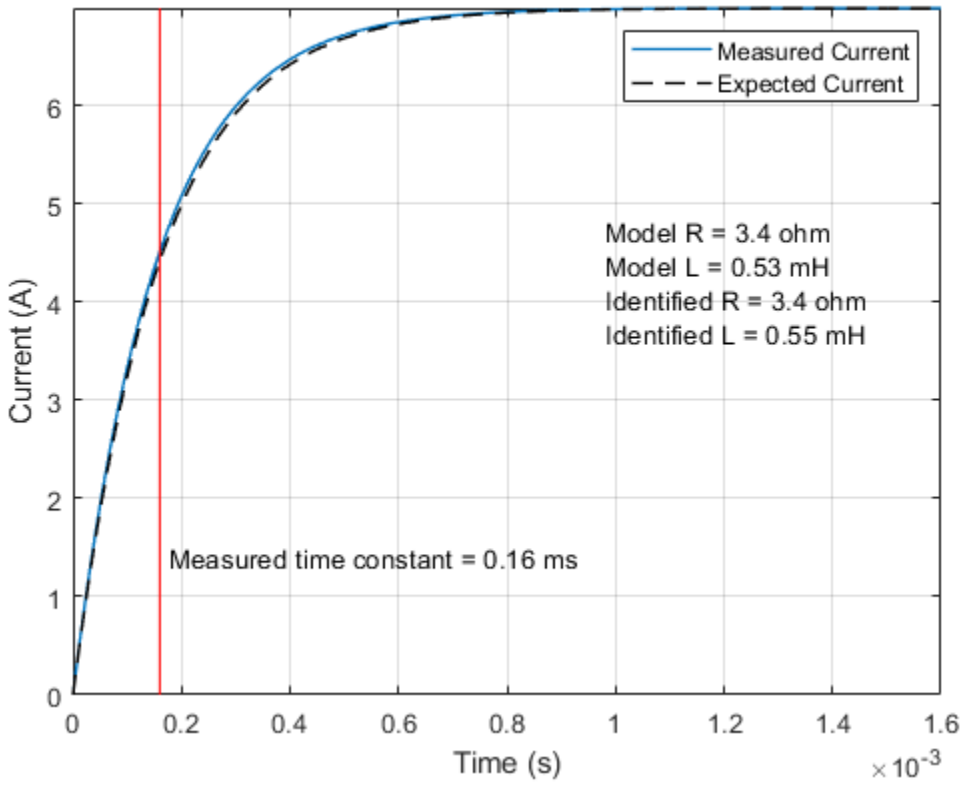

**PMSM Parameterization from Measurements**

1. Identify R and L and plot time constant (see code)
2. Identify back EMF constant (see code)
3. Identify friction, damping and inertia (see code)
4. Set parameters for model
5. Explore simulation results using Simscape Results Explorer
6. Learn more about this example

Copyright 2013-2022 The MathWorks, Inc.

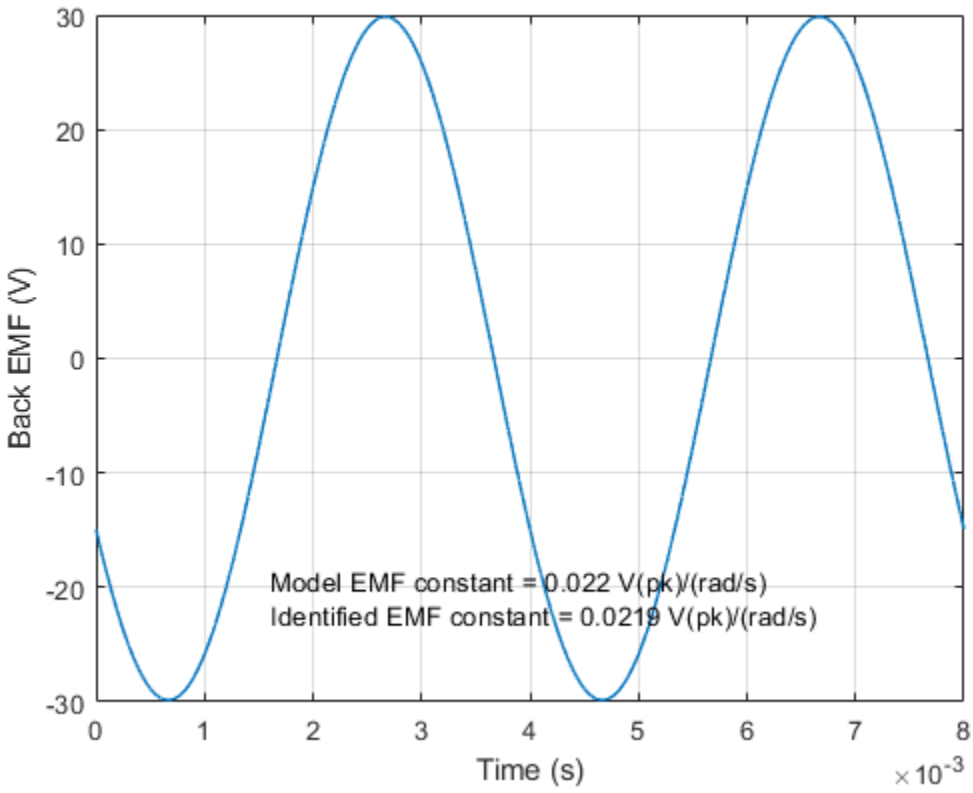
### Locked Rotor Test: Identify R and L

The first step is to lock the rotor and apply a voltage step to one of the stator windings. The resulting first-order time constant is defined by the stator resistance and inductance values, and the steady-state current by the stator resistance.




**Fixed Speed No Load Test: Identify Back EMF Constant**

The second step is to spin the motor in a dynamometer with no electrical load. This permits estimation of the back EMF constant. Note that, when expressed in SI units, the back EMF constant is equal to the torque constant.

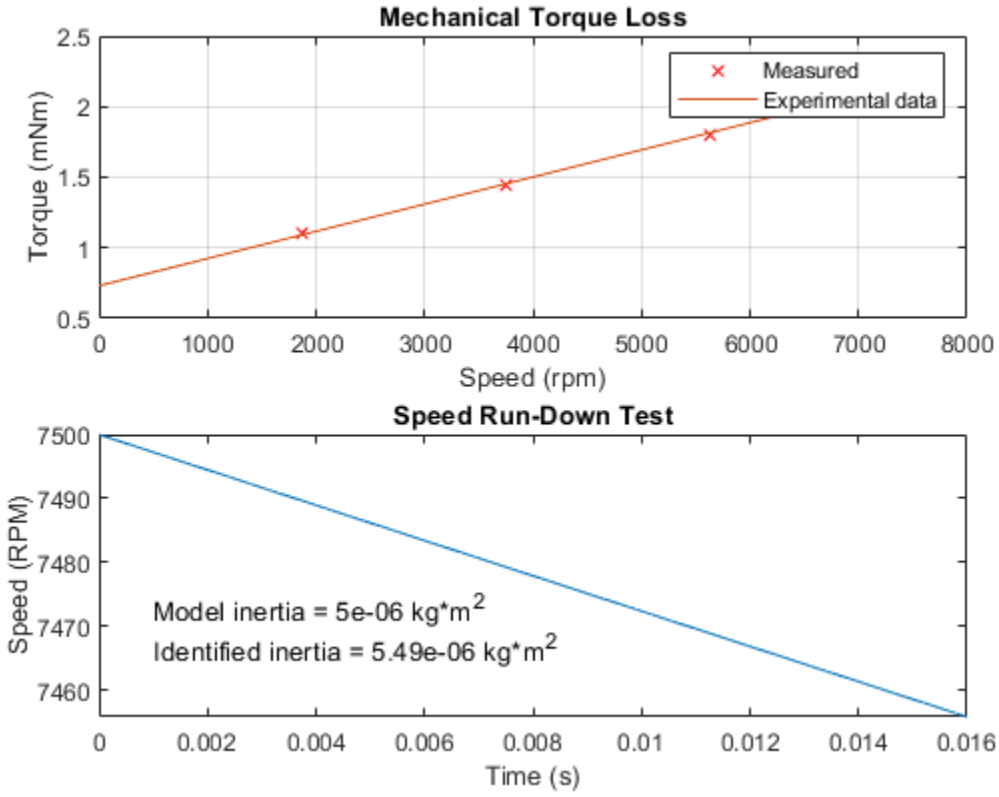



**Controlled Motor Test: Identify Friction, Damping and Inertia**

In this final step, the motor is spun using a motor controller and with no mechanical load. Mechanical torques due to friction and damping are measured by converting stator currents to torques using the already-identified torque constant.

The first plot below shows the mechanical torque required to maintain constant speed at four different speeds. The torque required at the lower speeds is primarily overcoming friction resistance, whereas at higher speeds viscous damping dominates. A straight line is fitted through the points, and the zero-speed intercept gives friction torque and the slope gives the viscous damping coefficient.

The second plot shows a speed run-down test. For this the demanded torque is set to zero, or the motor is unpowered. The gradient gives the deceleration from which the motor inertia can be determined given values for friction and damping torques.


Mechanical Torque Loss
Measured
Experimental data
Torque (mNm)
Speed (rpm)
Speed Run-Down Test
Model inertia = 5e-06 kg*m^2
Identified inertia = 5.49e-06 kg*m^2
Speed (RPM)
Time (s)

# Parameterize a Permanent Magnet Synchronous Motor

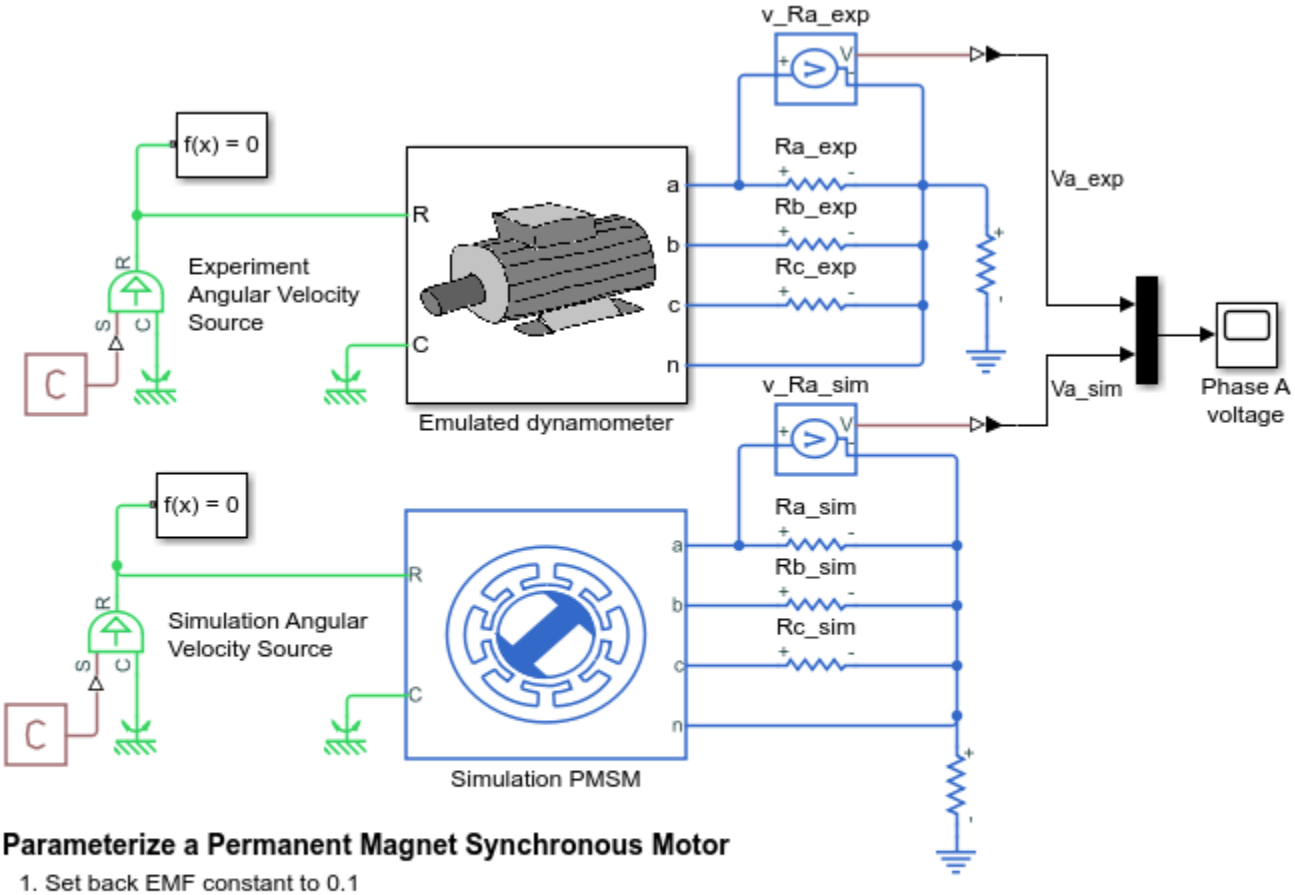
This example shows how to estimate the back EMF and torque constants of a blackbox permanent magnet synchronous motor (PMSM) with an unknown flux linkage. You can use either the back EMF or torque constant to describe the flux linkage and parameterize a Simscape™ Electrical™ PMSM block. This parameterization allows you to accurately replicate the blackbox motor's behaviour in simulation.

In this example, the number of pole pairs, stator resistance and stator inductances of the simulation and blackbox PMSMs are already specified to match.

### Open Model

Open the model.

```
model='ee_pmsm_parameterization';
open_system(model)
set_param(find_system('ee_pmsm_parameterization','FindAll', 'on','type','annotation','Tag','Mode
```




**Parameterize a Permanent Magnet Synchronous Motor**

1. Set back EMF constant to 0.1
2. Set back EMF constant to estimated value
3. Modify example script
4. Explore simulation results using Simscape Results Explorer
5. Learn more about this example

Copyright 2017-2022 The MathWorks, Inc.

### Set Back EMF Constant

Select the back EMF parameterization option for the simulation PMSM and initialize its value to an arbitrary default value of 0.1 V*s/rad.

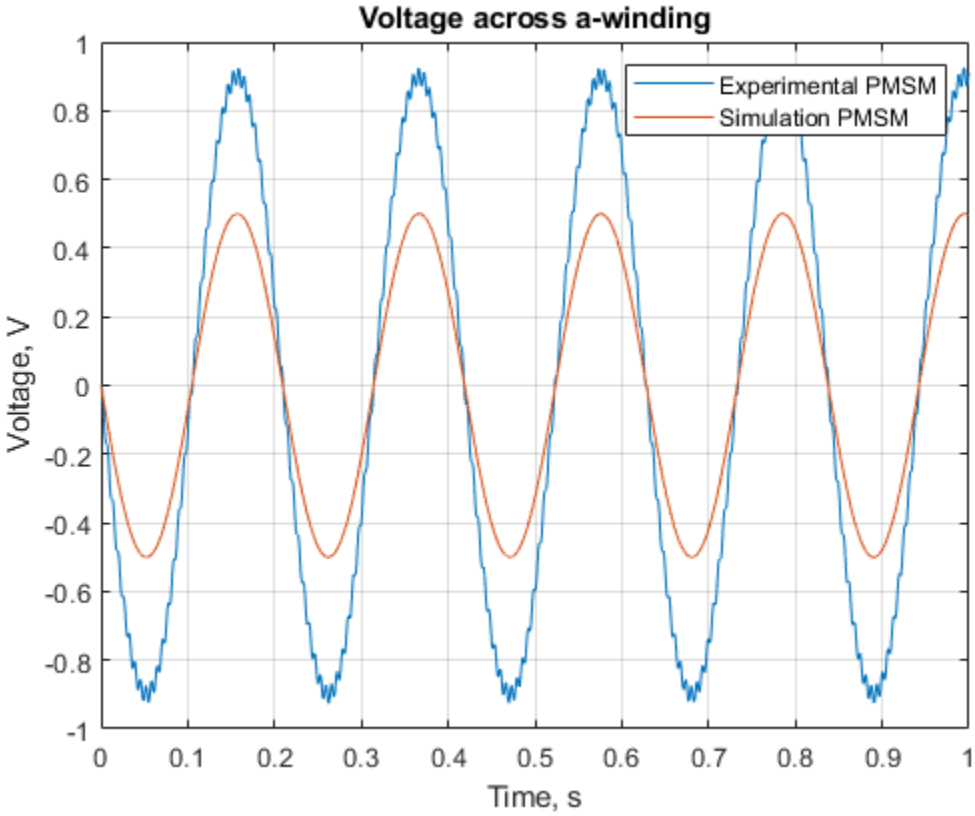
```
Simulation_PMSM = [model,'/Simulation PMSM'];
Ke = 0.1;
set_param(Simulation_PMSM,'pmflux_param','3');
```

Alternatively, you can set the back EMF parameterization option and specify its value from the PMSM block mask by selecting **Specify back EMF** under **Permanent magnet flux linkage parameterization** and specifying **Back EMF constant**.

### Compare Results

Simulate the model and plot the voltage over time across the a-winding for both the experimental and simulation PMSMs. Observe that there is a significant difference in voltage amplitude between the experimental and simulation PMSMs.

```
sim(model);
v_a_exp = simlog_ee_pmsm_parameterization.v_Ra_exp.V;
v_a_sim = simlog_ee_pmsm_parameterization.v_Ra_sim.V;
simscape.logging.plot({v_a_exp,v_a_sim},...
    'names',{'Experimental PMSM','Simulation PMSM'});
title('Voltage across a-winding')
ylabel('Voltage, V');
```




### Calculate Voltage Constant

Calculate the voltage constant $K_e$ of the blackbox experimental PMSM by using the defining equation

$$K_e = \frac{V_{a,pk}}{\omega},$$

where $\omega$ is the angular velocity of the PMSM shaft, and $V_{a,pk}$ is the voltage drop across the a-winding of the motor. This equation is valid only when the current through the windings is very small. The small current restriction is enforced in the model by attaching each of the winding free ends to ground via a very high resistance, effectively making each winding an open-circuit.

$V_{a,pk}$ is calculated by taking the average of the maximum and minimum values of the voltage drop across the a-winding, and dividing the result by two. You can improve the accuracy of this calculation by first filtering out the high frequency noise of the measured voltage signal:

```
window_size = 100;
v_a_exp_filt = filter(1/window_size*ones(1,window_size),1,v_a_exp.series.values);
v_a_exp_peak = (max(v_a_exp_filt)-min(v_a_exp_filt))/2;
```

The angular velocity can be measured directly from the source block:

```
angular_velocity_exp = mean(simlog_ee_pmsm_parameterization.Experiment_Angular_Velocity_Source.w
```

The back EMF constant can now be calculated from the above results:

```
Ke = v_a_exp_peak / angular_velocity_exp;
disp(Ke);
```

```
    0.1803
```

### Change Back EMF Constant

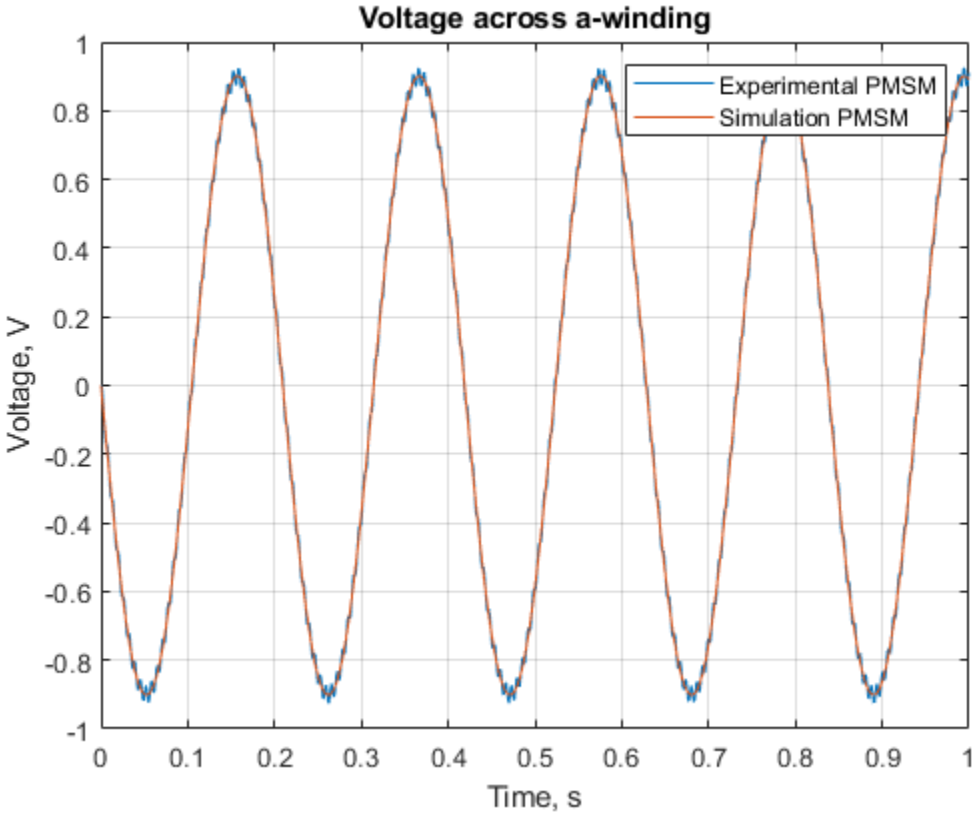
Change the back EMF constant for the simulation motor to the estimated value $K_e$, calculated in the previous section.

```
Simulation_PMSM = [model,'/Simulation PMSM'];
set_param(Simulation_PMSM,'pmflux_param','3');
```

### Compare New Results

Resimulate the model and plot the voltage across the a-winding. The new parameterization of the voltage constant improves the simulation PMSM's ability to replicate the behaviour of the experimental PMSM.

```
sim(model);
v_a_exp = simlog_ee_pmsm_parameterization.v_Ra_exp.V;
v_a_sim = simlog_ee_pmsm_parameterization.v_Ra_sim.V;
simscape.logging.plot({v_a_exp,v_a_sim},...
    'names',{'Experimental PMSM','Simulation PMSM'});
title('Voltage across a-winding')
ylabel('Voltage, V');
```




### Calculate Torque Constant

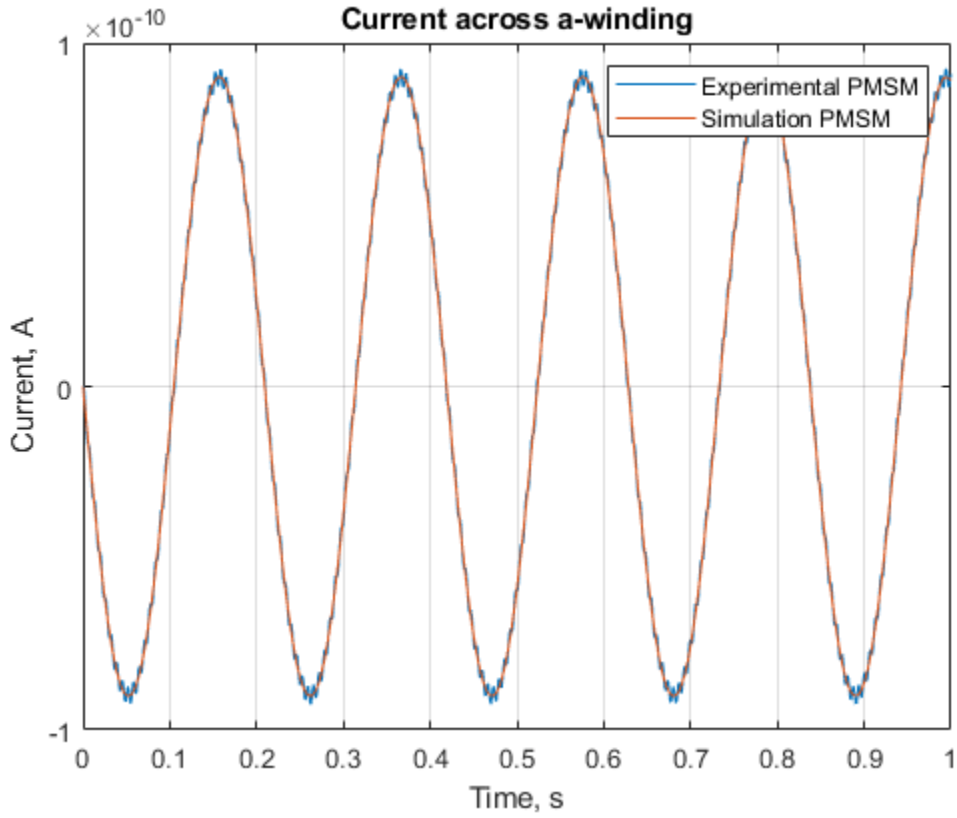
Calculate the torque constant $K_t$ of the blackbox PMSM using the defining equation:

$$K_t = \frac{2}{3}\left(\frac{T}{I_{a,pk}}\right),$$

where $I_{a,pk}$ is the peak current through the a-winding and $T$ is the overall mechanical torque driving the PMSM.

```
i_a_exp = simlog_ee_pmsm_parameterization.Ra_exp.i;
i_a_sim = simlog_ee_pmsm_parameterization.Ra_sim.i;

T = abs(mean(simlog_ee_pmsm_parameterization.Experiment_Angular_Velocity_Source.t.series.values))

window_size = 100;
i_a_exp_filt = filter(1/window_size*ones(1,window_size),1,i_a_exp.series.values);
i_a_exp_peak = (max(i_a_exp_filt)-min(i_a_exp_filt))/2;

Kt = 2/3*T/i_a_exp_peak;
disp(Kt);

simscape.logging.plot({i_a_exp,i_a_sim},...
    'names',{'Experimental PMSM','Simulation PMSM'});
title('Current across a-winding')
ylabel('Current, A');
```

```
0.1797
```




### Compare Flux Linkage Parameterizations

As expected, the measured torque and back EMF constants have approximately the same value (0.18) and are related to the permanent magnet flux linkage $\psi_m$ via the number of pole pairs $N$ of the motor:

$$K_e = K_t = N\psi_m.$$

You can parameterize the permanent magnet flux linkage of the motor by specifying any of $K_e$, $K_t$ or $\psi_m$ in the Simulation PMSM block mask. Back EMF and torque constants are more commonly given than permanent magnet flux linkage on motor datasheets.

### Learn More

See the PMSM block page for more information.

# Push-Pull Buck Converter in Continuous Conduction Mode

This example shows how to control the output voltage of a push-pull buck converter. The current flowing through the inductor is never zero, therefore the DC-DC converter operates in Continuous Conduction Mode (CCM). To convert and maintain the nominal output voltage, the PI Controller subsystem uses a simple integral control. During startup, the reference voltage is ramped up to the desired output voltage.

The converter operates in CCM only if

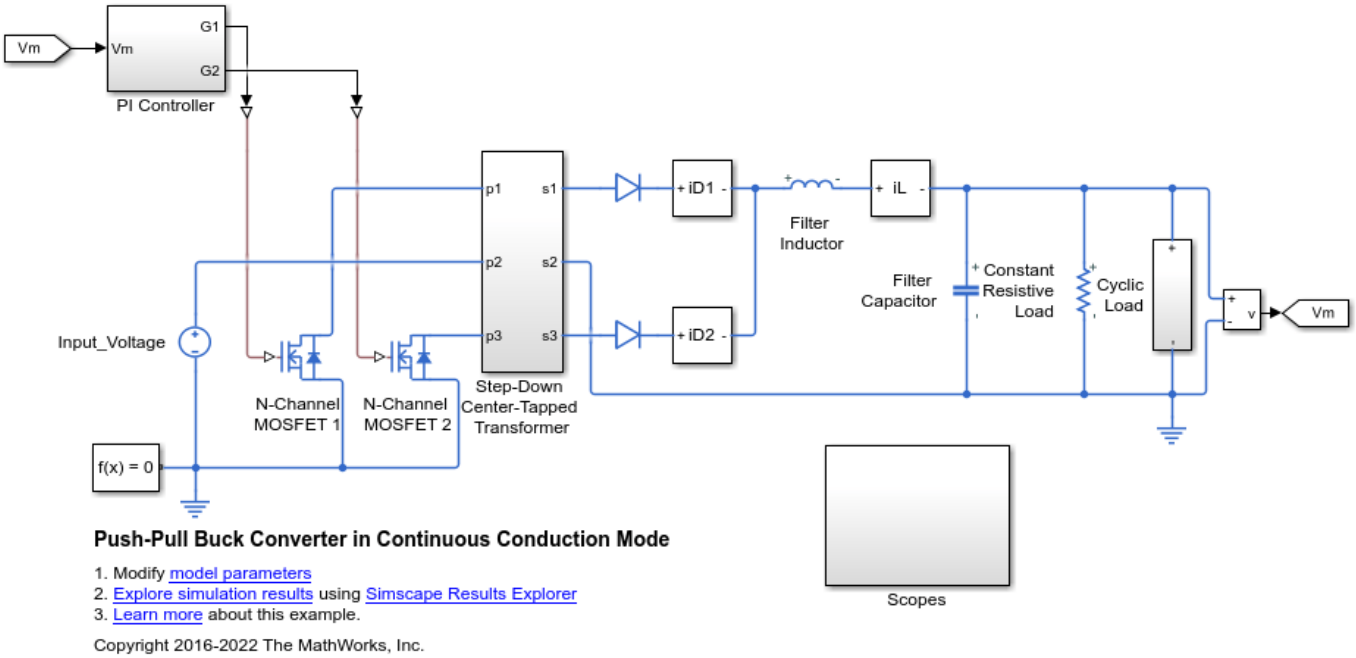
- $K > K_{critical}$ ,

where:

- $K = 2 * L/(R * Tsw)$ .
- $K_{critcal} = 1 - D$ .
- $L$ is the filter inductance.
- $R$ is the load resistance.
- $T_{sw}$ is the switching period for each MOSFET. That is, $T_{sw} = 0.5/f_{sw}$ , where $f_{sw}$ is the switching frequency.
- $D$ is the duty cycle of PWM input to the gate of each MOSFET. That is, $D = T_{on}/T_{sw}$ , where $T_{on}$ is the ON time of the MOSFET.

### Open Model

```
open_system('ee_push_pull_converter_ccm.slx');
```



### Specify the Design Parameters

The system is required to generate and maintain an output voltage of 80 V with a full load power capability of 1000 W. The input voltage is 400 V and the transformer turns ratio is 2. The full load includes a constant load and a cyclic load. The 'ee_push_pull_converter_ccm_data.m' script defines the design parameters as variables in the MATLAB® workspace.

```
Input_Voltage           = 400;                                  %  Input voltage to the push-pull c
Output_Voltage          = 80;                                   %  Desired output voltage from the
Output_Power            = 1000;                                 %  Full load power output [W]
fsw_Hz                  = 40000;                                %  MOSFET switching frequency [Hz]
primary_winding         = 200;                                  %  Number of turns in the primary w
secondary_winding       = 100;                                  %  Number of turns in the secondary
TR                      = primary_winding/secondary_winding;    %  Turns ratio
Kp                      = 0.01;                                 %  Proportional gain for PI control
Ki                      = 20;                                   %  Integral gain for PI controller
del_I                   = 40;                                   %  Peak-peak inductor ripple curren
del_V                   = 1;                                    %  Peak-peak output voltage ripple
share_constload         = 70;                                   %  Percentage of load current drawn
share_cyclicload        = 100-share_constload;                  %  Percentage of load current drawn
cyclic_load_period      = 1/20;                                 %  Cyclic load period
cyclic_load_pul_width   = 50;                                   %  Pulse width of the current pulse
Ts                      = 1e-7;                                 %  Sampling time for the solver
```

### Calculate the Open-Loop Duty Cycle

The duty cycle depends on the input voltage, the turns ratio and the desired output voltage.

```
Duty = Output_Voltage/(Input_Voltage/TR);
```

### Determine the Constant Load Resistance

```
I_fl_average = Output_Power/Output_Voltage;                     %  Full load average current that
R_const = Output_Voltage/I_fl_average;
```

### Calculate the Filter Inductance

Choose the inductance value based on the input and output specifications of the converter. The inductance value depends on the input and output specifications of the converter. For this example, the converter is required to work in CCM for 20-100% of full load power. When, at the lower boundary condition, the power is 20% of full load power, the average load current is 20% of full load average current, I_fl_average. At the end of each cycle at the lower boundary condition, the inductor current goes to zero. The inductor ripple current, del_I, at this point is twice the average output load current, that is 40% of the full load average output current.

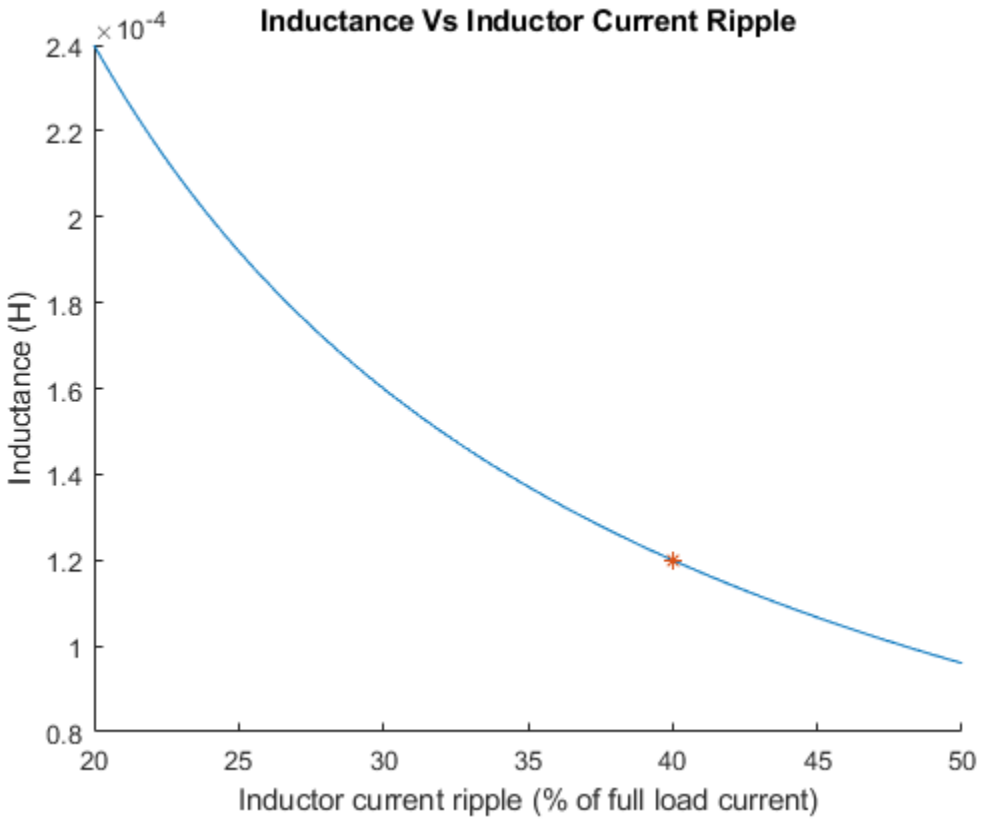
```
L_min = (Input_Voltage/TR)*Duty*(1-Duty)/(2*fsw_Hz*del_I*I_fl_average*...
    0.01);
```

### Plot Inductance Versus Inductor Current Ripple

Generate this plot to see how the filter inductance relates to the inductor ripple current (expressed as a percentage of full load current). For this example, the marker at 40% corresponds to an inductance of 1.2e-04 H.

```
del_I_range = 20:0.1:50;     %  Percentage of full load current (20-50%)
L_range = (Input_Voltage/TR)*Duty*(1-Duty)./(2*fsw_Hz*del_I_range*...
    I_fl_average*0.01);
figure;
```

```
hold on;
plot(del_I_range,L_range);
hold on;
L_del_I = (Input_Voltage/TR)*Duty*(1-Duty)/(2*fsw_Hz*del_I*...
    I_fl_average*0.01);
plot(del_I,L_del_I, '*');
xlabel('Inductor current ripple (% of full load current)');
ylabel('Inductance (H)');
title('Inductance Vs Inductor Current Ripple');
```




### Choose a Filter Capacitance

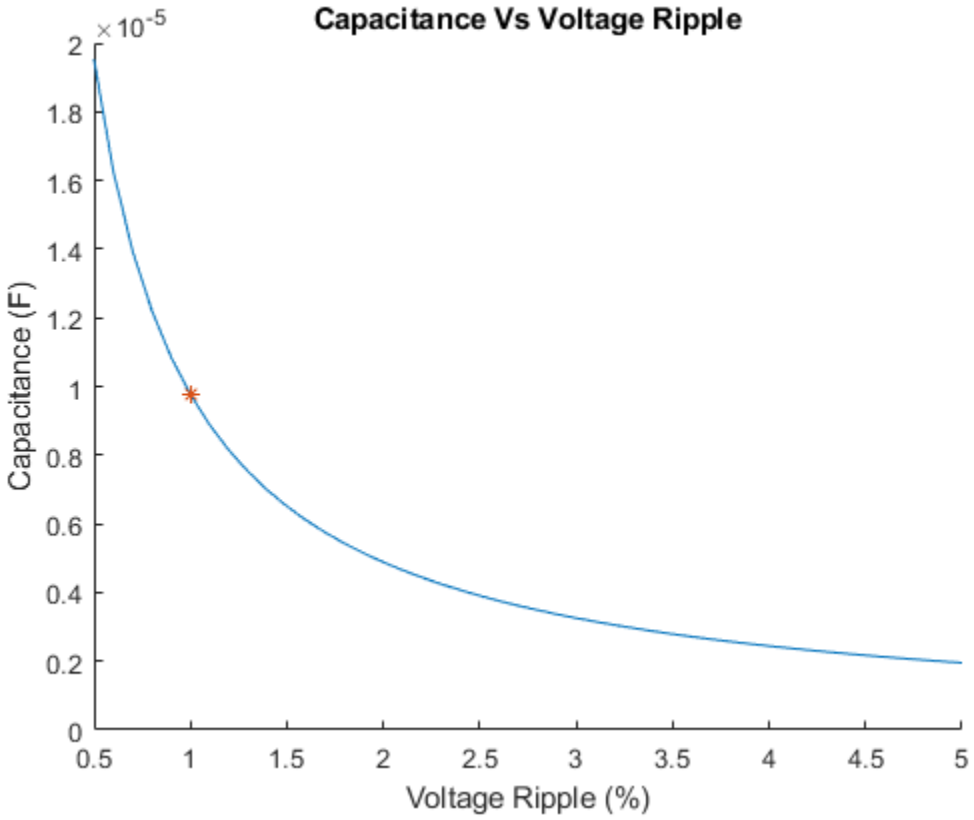
```
C_min = (Input_Voltage/TR)*Duty*(1-Duty)/(8*(2*fsw_Hz)^2*L_min*...
    Output_Voltage*del_V*0.01);
```

### Plot Capacitance Versus Voltage Ripple

Generate this plot to see how capacitance for limiting the output voltage ripple varies depending on the design parameters. For this example, the marker at 1% Output Voltage Ripple corresponds to a capacitance of 9.766e-06 F.

```
del_V_range = 0.5:0.1:5;
C_range = (Input_Voltage/TR-Output_Voltage)*Duty./(8*(2*fsw_Hz)^2*L_min*...
    Output_Voltage*del_V_range*0.01);
figure;
hold on;
plot(del_V_range,C_range);
hold on;
```

```
C = (Input_Voltage/TR-Output_Voltage)*Duty/(8*(2*fsw_Hz)^2*L_min*...
    Output_Voltage*del_V*0.01);
plot(del_V,C,'*');
xlabel('Voltage Ripple (%)');
ylabel('Capacitance (F)');
title('Capacitance Vs Voltage Ripple');
```



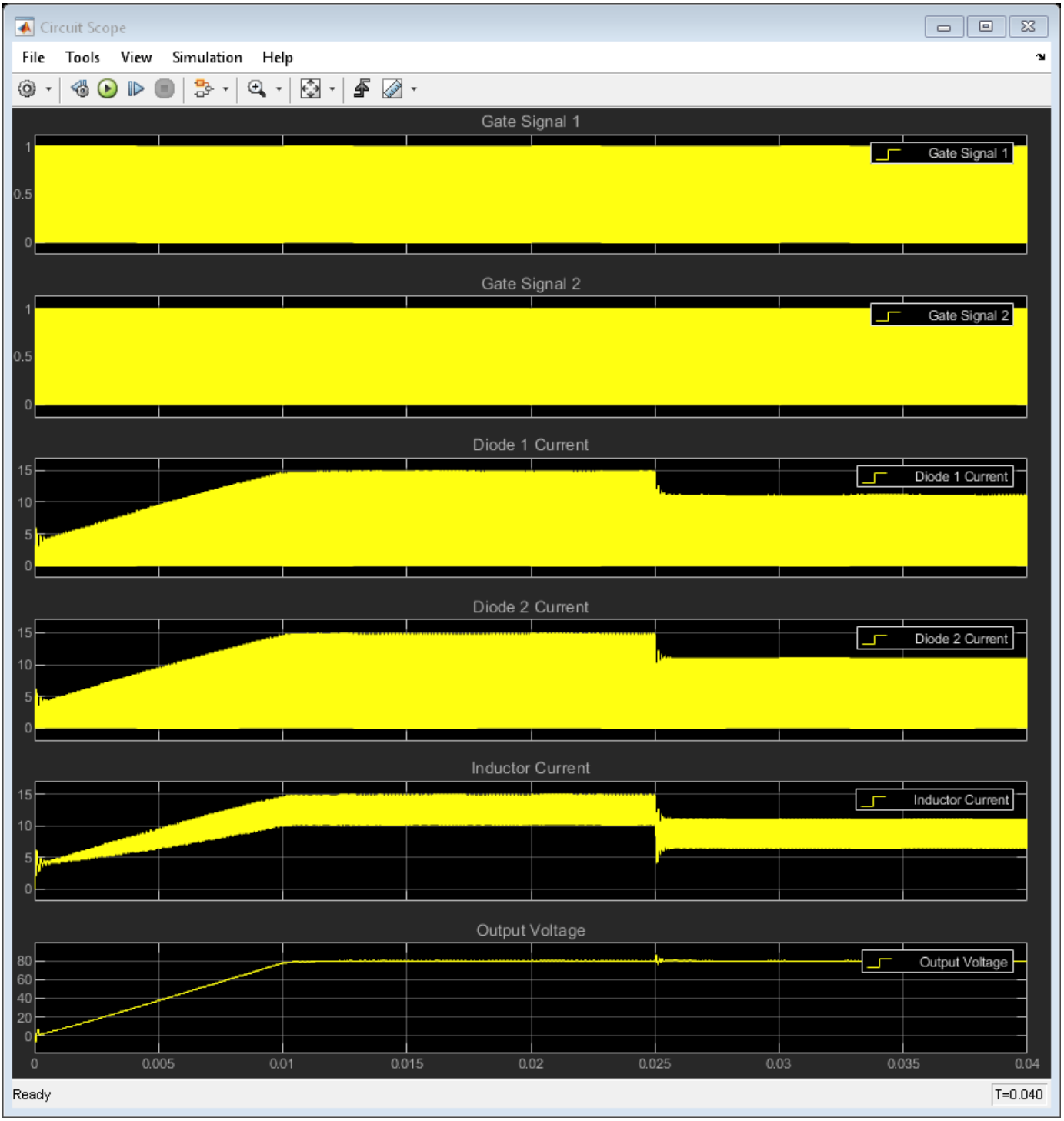

### Run the Simulation

```
sim('ee_push_pull_converter_ccm.slx');
```

### View Simulation Results

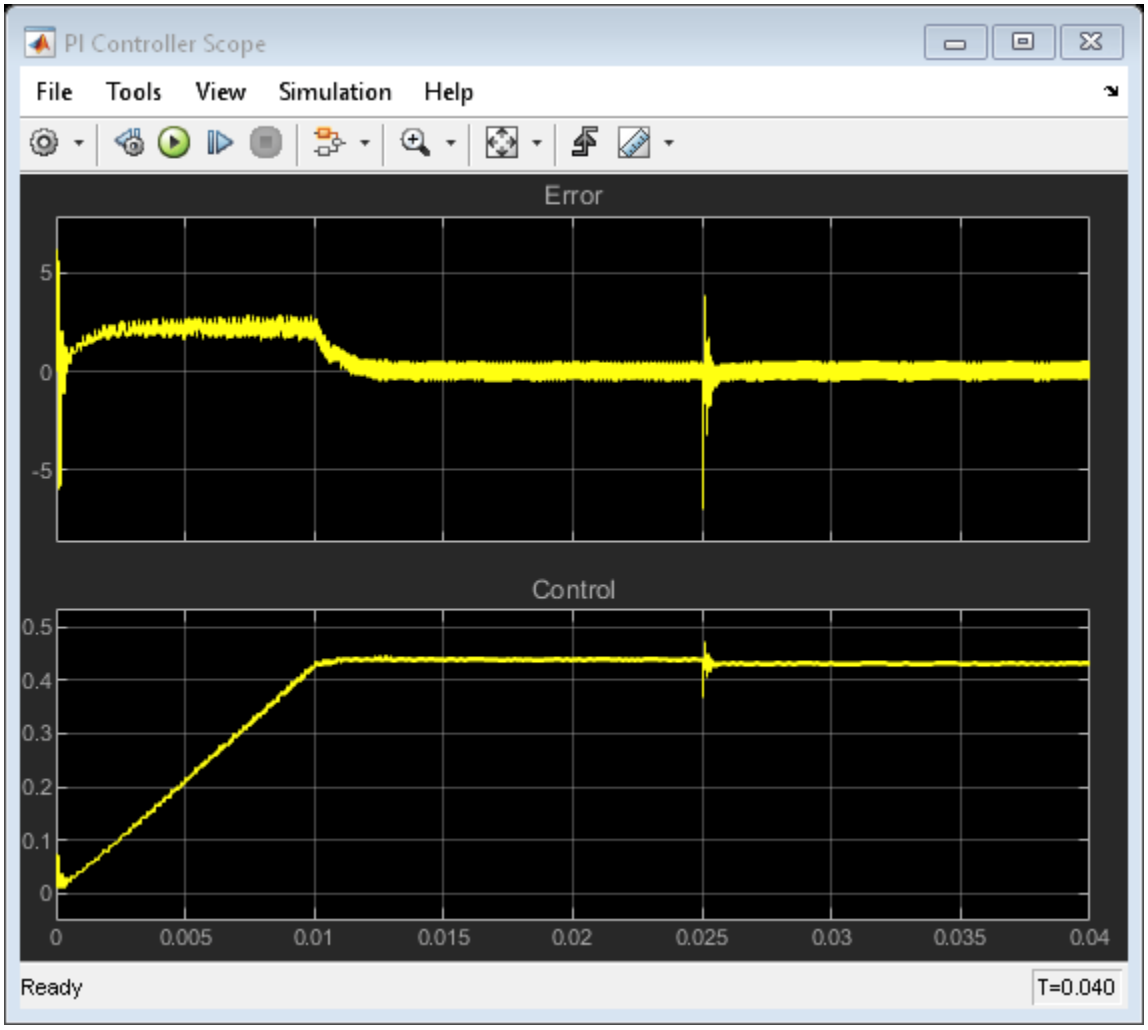
To view summary results during or after the simulation, open the Circuit Scope block from the model window or by entering, at the MATLAB command prompt:

```
open_system('ee_push_pull_converter_ccm/Scopes/Circuit Scope');
```




To view the control and error data during or after the simulation, open the PI Controller Scope block from the model window or, enter:

```
open_system('ee_push_pull_converter_ccm/Scopes/PI Controller Scope');
```




After the simulation, to view logged Simscape™ data using the Simscape Results Explorer, enter:

```
sscexplore(simlog_ee_push_pull_converter_ccm);
%
```

# Push-Pull Buck Converter in Discontinuous Conduction Mode

This example shows how to control the output voltage of a push-pull buck converter. The current flowing through the inductor reaches zero during the switch off cycle of the MOSFETs and therefore the DC-DC converter operates in Discontinuous Conduction Mode (DCM). This mode of conduction is mostly used for low-power applications. To convert and maintain the input DC voltage as a nominal output voltage, the PI Controller subsystem uses a simple integral control. During startup, the reference voltage is ramped up to the desired output voltage.

The converter operates in DCM only if

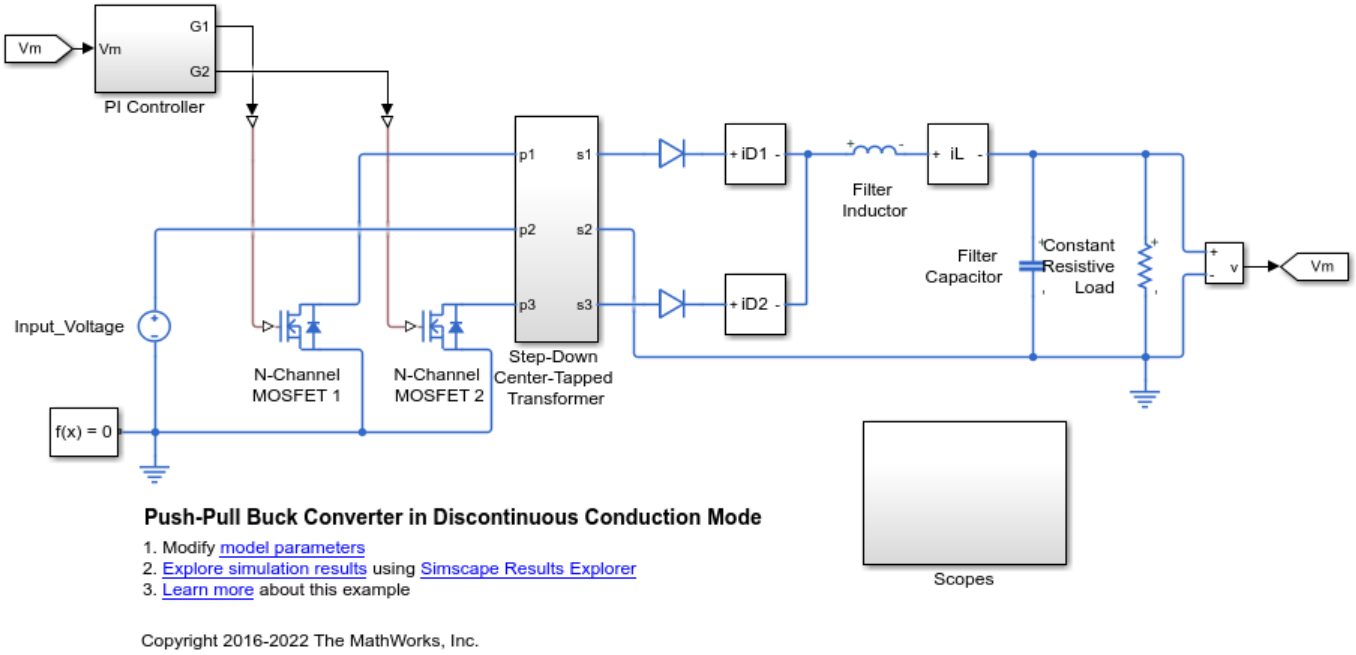
- $K < K_{critical}$ ,

where:

- $K = 2 * L/(R * Tsw)$ .
- $K_{critcal} = 1 - D$ .
- $L$ is the filter inductance.
- $R$ is the load resistance.
- $T_{sw}$ is the switching period for each MOSFET. That is, $T_{sw} = 0.5/f_{sw}$ , where $f_{sw}$ is the switching frequency.
- $D$ is the duty cycle of PWM input to the gate of each MOSFET. That is, $D = T_{on}/T_{sw}$ , where $T_{on}$ is the ON time of the MOSFET.

### Open Model

```
open_system('ee_push_pull_converter_dcm.slx');
```




### Specify the Design Parameters

The system is required to generate and maintain an output voltage of 80 V with a power requirement of 100 W. The input voltage is 400 V and the transformer turns ratio is 2. The load includes a constant resistive load. The 'ee_push_pull_converter_dcm_data.m' script defines the design parameters as variables in the MATLAB® workspace.

```
Input_Voltage           = 400;                                  %  Input Voltage to the push-pull c
Output_Voltage          = 80;                                   %  Desired Output Voltage from the
Output_Power            = 1000;                                 %  Full Load Power Output [W]
fsw_Hz                  = 40000;                                %  MOSFET Switching Frequency [Hz]
primary_winding         = 200;                                  %  Number of turns in the primary w
secondary_winding       = 100;                                  %  Number of turns in the secondary
TR                      = primary_winding/secondary_winding;    %  Turns Ratio
Kp                      = 0.01;                                 %  Proportional Gain for PI Contro
Ki                      = 20;                                   %  Integral Gain for PI Controller
del_V                   = 1;                                    %  Peak-Peak Output Voltage Ripple
K                       = 0.3;                                  %  Denotes trajectory that gives a
Ts                      = 1e-7;                                 %  Sampling time for the solver
```

### Operating Boundary Between Discontinuous Mode of Conduction and Continuous Mode of Conduction

Based on the constraint established by K and the Duty Cycle, the operating mode switches between Discontinuous Conduction Mode (DCM) and Continuous Conduction Mode (CCM). The Output Voltage Ratios of CCM and DCM are:

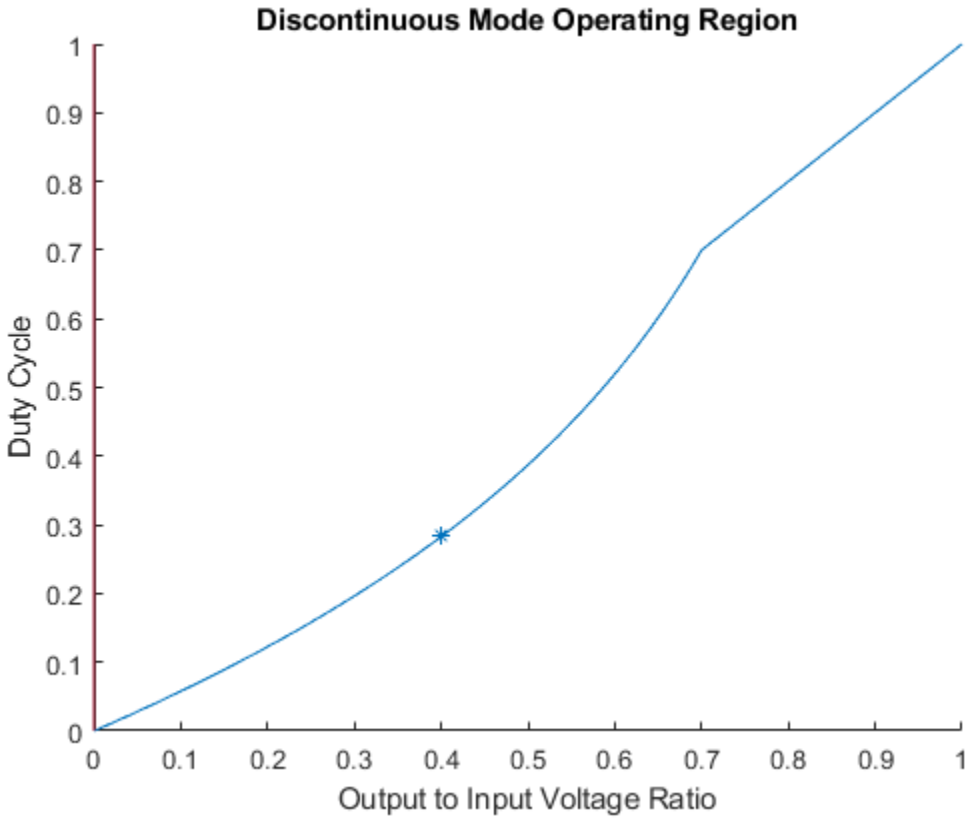
- Continuous Conduction Mode
- $V_{output}/V_{input} = D$
- Discontinuous Conduction Mode

- $V_{output}/V_{Input} = 2/(1+\sqrt{1+4*K/D^2})$

**Calculate the Open-Loop Duty Cycle for DCM**

The operating modes can be visualised for different values of K by generating a plot between the Output Voltage Ratio and the Duty Cycle. From this plot, find the corresponding Duty Cycle needed to achieve the Output Voltage Ratio as specified in the Design Parameters for the particular value of K chosen.

```
figure;
D_range = 0:0.001:1;
Voltage_ratio = zeros(length(D_range));
for i=1:length(D_range)
    K_crit = 1-D_range(i);
    if K < K_crit
        Voltage_ratio(i) = 2/(1+sqrt(1+4*K/D_range(i)^2));
    else
        Voltage_ratio(i) = D_range(i);
    end
end
VR = Output_Voltage/(Input_Voltage/TR);
Duty = sqrt(4*K/((2/VR-1)^2-1));
hold on;
plot(Voltage_ratio,D_range);
hold on;
plot(VR,Duty,'*');
ylabel('Duty Cycle');
xlabel('Output to Input Voltage Ratio');
title('Discontinuous Mode Operating Region');

% The Open-Loop Duty Cycle required would be approximately 28.28% from the
% graph.
```




### Determine the Constant Load Resistance

```
I_average = Output_Power/Output_Voltage;                    %  Average current that flows throug
R_const = Output_Voltage/I_average;
```

### Determine the Filter Inductance

To estimate the inductance required for DCM, use this relationship between K, Resistance, and the Switching Time Period.

```
L_min = (K*R_const)/(2*2*fsw_Hz);
```

Verify the estimated inductance values result in DCM operation. If the value results in CCM operation, choose a different value of K and recalculate. Iterate until you find an inductance value that results in DCM operation.

### Choose a Filter Capacitance

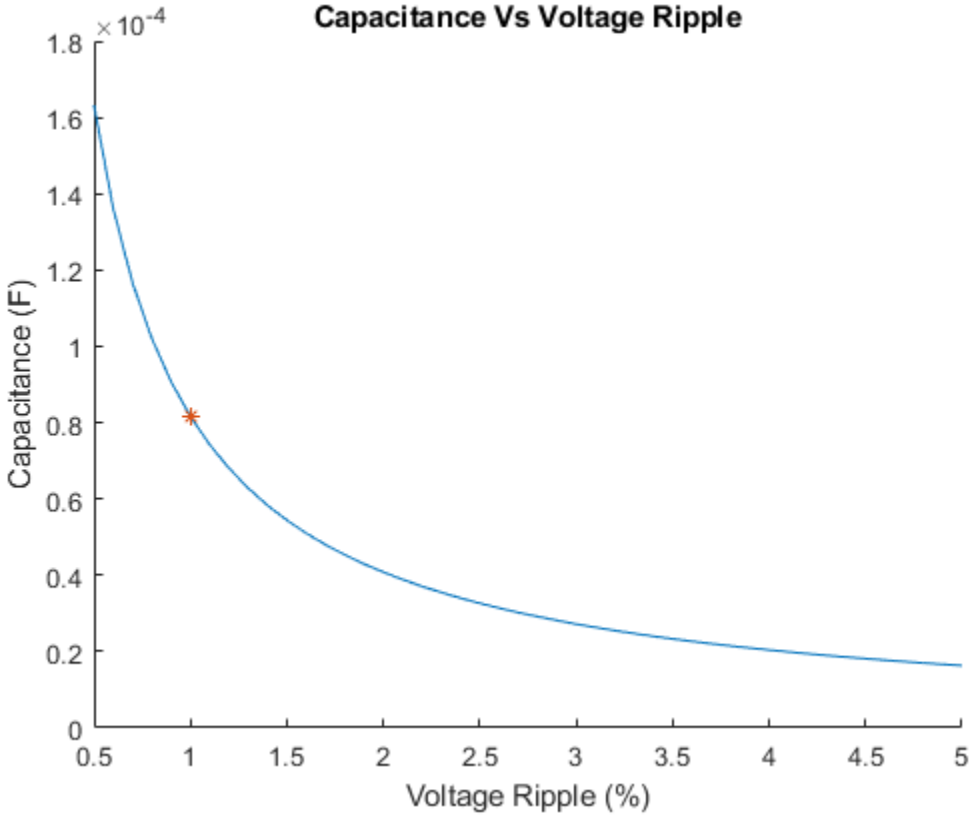
The relation between the capacitance and the Output Voltage Ripple is

```
C_min = (2-(Duty/(Output_Voltage/(Input_Voltage/TR))))^2/(4*R_const*2*fsw_Hz*del_V*0.01);
```

### Plot Capacitance Versus Voltage Ripple

Generate a plot to see the capacitance that is required for limiting the output voltage ripple varies depending on the design parameters. For this example, the marker at 1% Output Voltage Ripple corresponds to a capacitance of 8.157e-06 F.

```
del_V_range = 0.5:0.1:5;
C_range = (2-(Duty/(Output_Voltage/(Input_Voltage/TR))))^2./(4*R_const*2*fsw_Hz*del_V_range*0.01
figure;
hold on;
plot(del_V_range,C_range);
hold on;
plot(del_V,C_min,'*');
xlabel('Voltage Ripple (%)');
ylabel('Capacitance (F)');
title('Capacitance Vs Voltage Ripple');
```



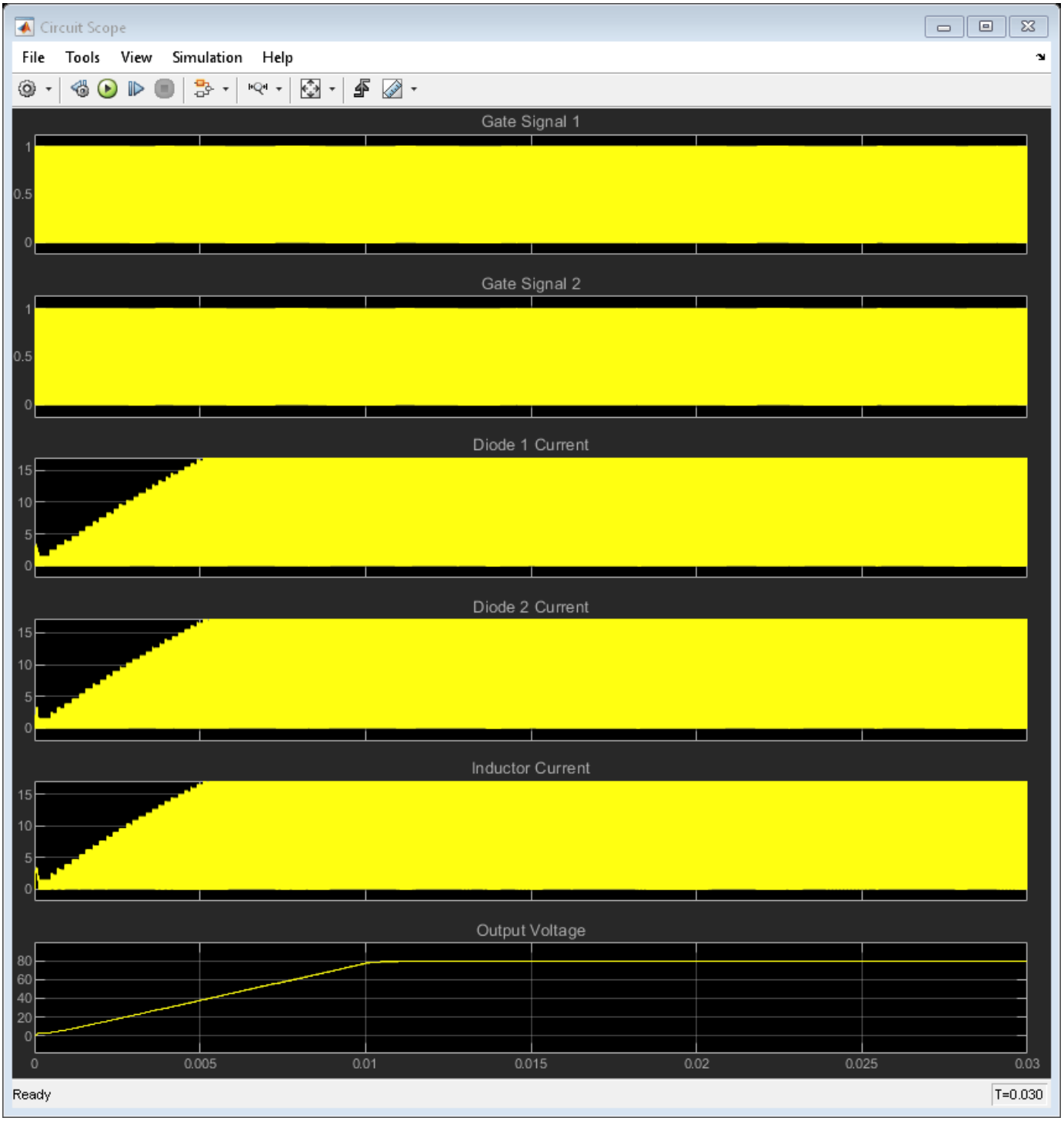

### Run the Simulation

```
sim('ee_push_pull_converter_dcm.slx');
```

### View Simulation Results

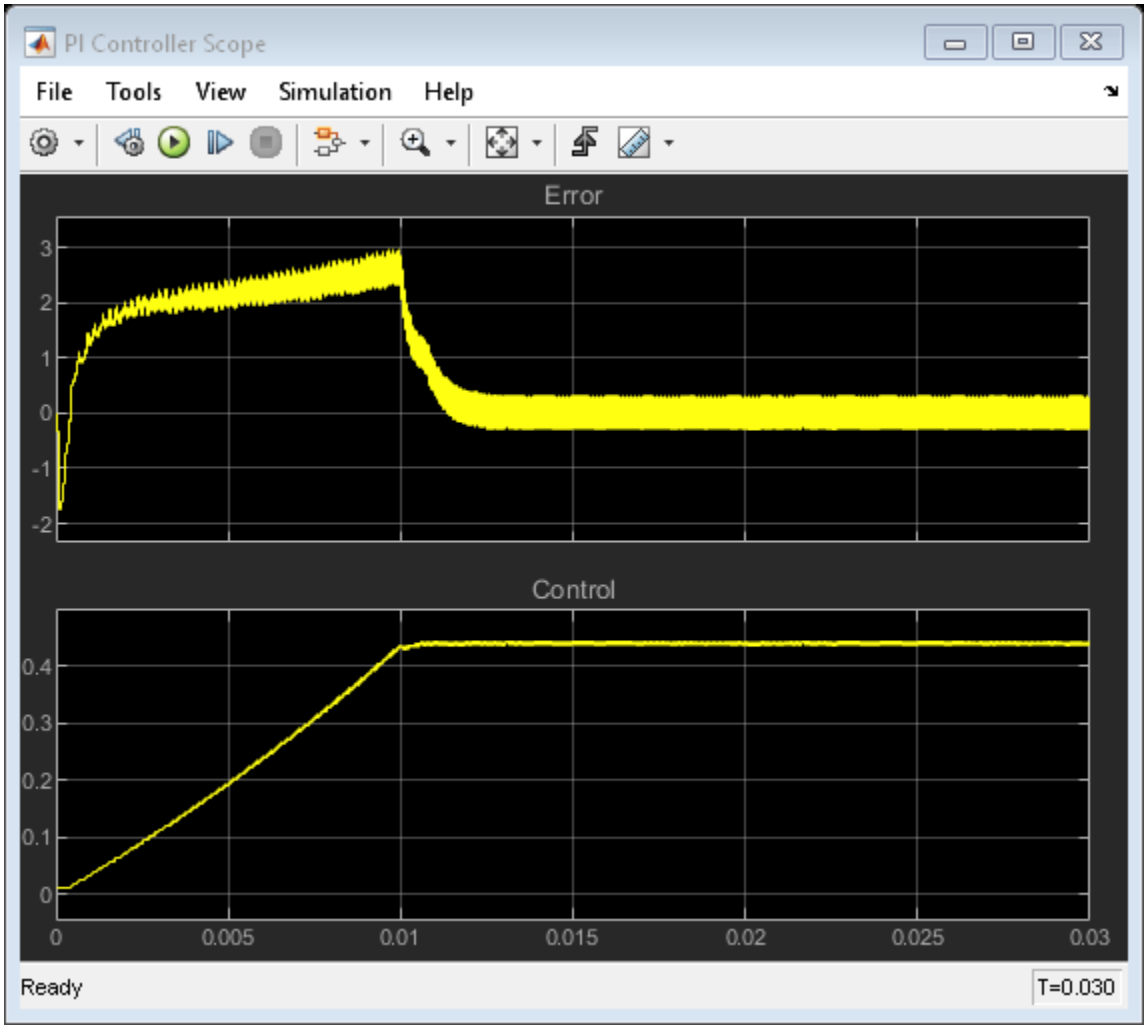
To view summary results during or after the simulation, open the Circuit Scope block from the Scopes subsystem or by entering, at the MATLAB command prompt:

```
open_system('ee_push_pull_converter_dcm/Scopes/Circuit Scope');
```




To view the control and error data during or after the simulation, open the PI Controller Scope block from the Scopes subsystem or, enter:

```
open_system('ee_push_pull_converter_dcm/Scopes/PI Controller Scope');
```



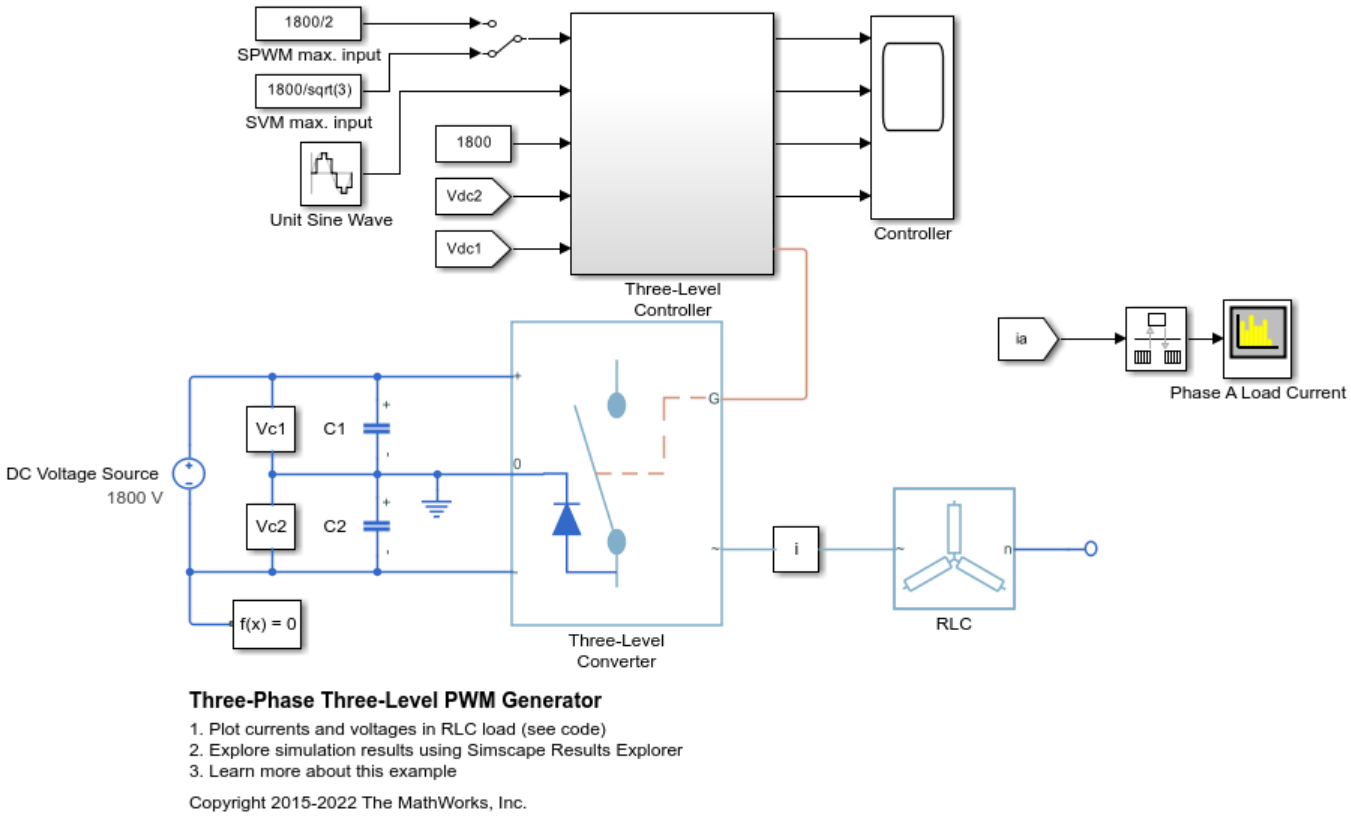

After the simulation, to view logged Simscape™ data using the Simscape Results Explorer, enter:

```
sscexplore(simlog_ee_push_pull_converter_dcm);
%
```

# Three-Phase Three-Level PWM Generator

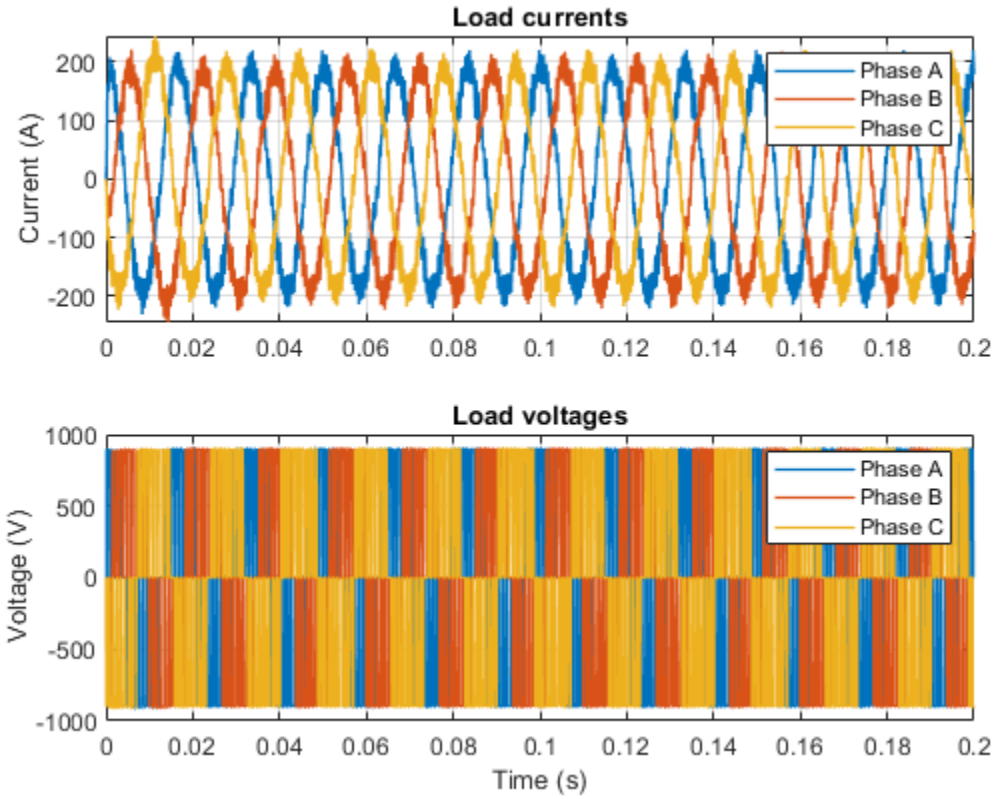
This example shows how to use the PWM Generator (Three-phase, Three-level) to control a Three-Level Converter. The upper and lower supply voltages are input to a Neutral point controller, which balances the DC-link capacitor voltages. Reference AC waveforms are used as inputs to the PWM Generator. There is one time scope for the controller waveforms.

### Model



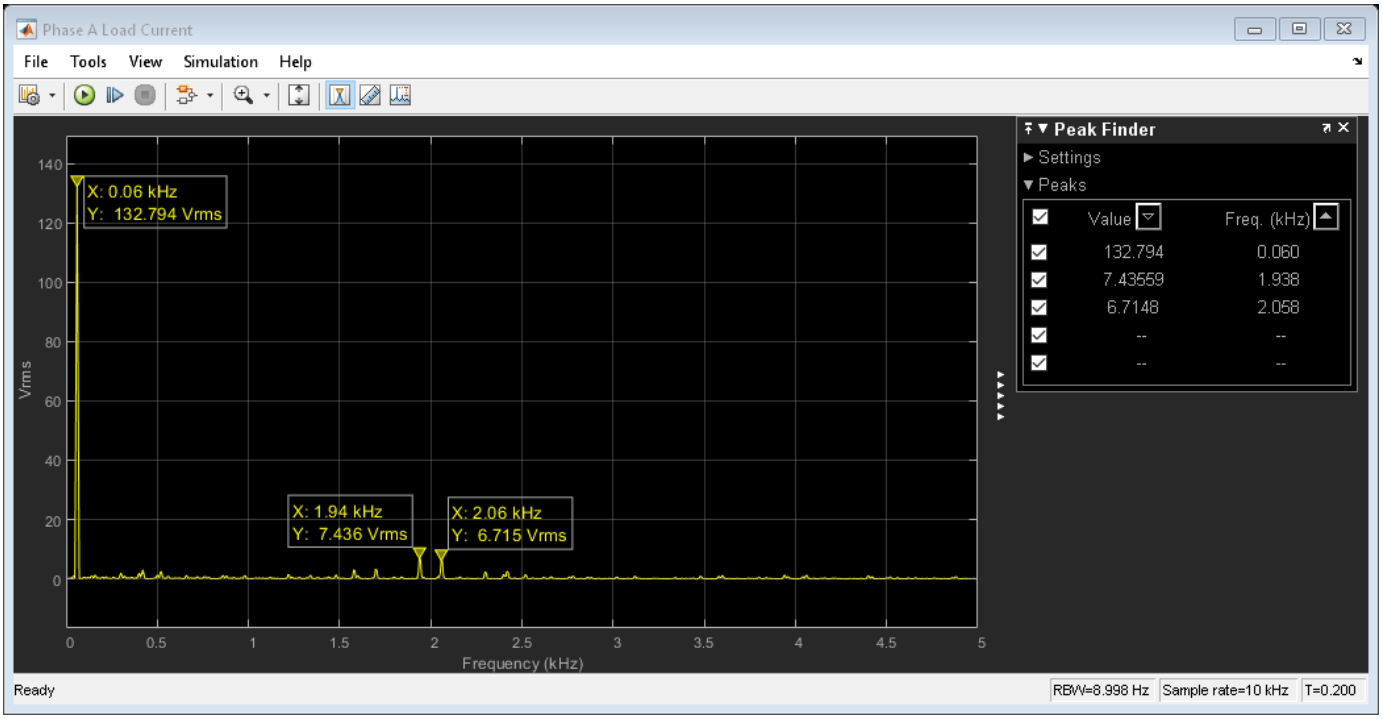

### Simulation Results from Simscape Logging

The plot below shows the load voltages and currents over time.




**Simulation Results from Scopes**

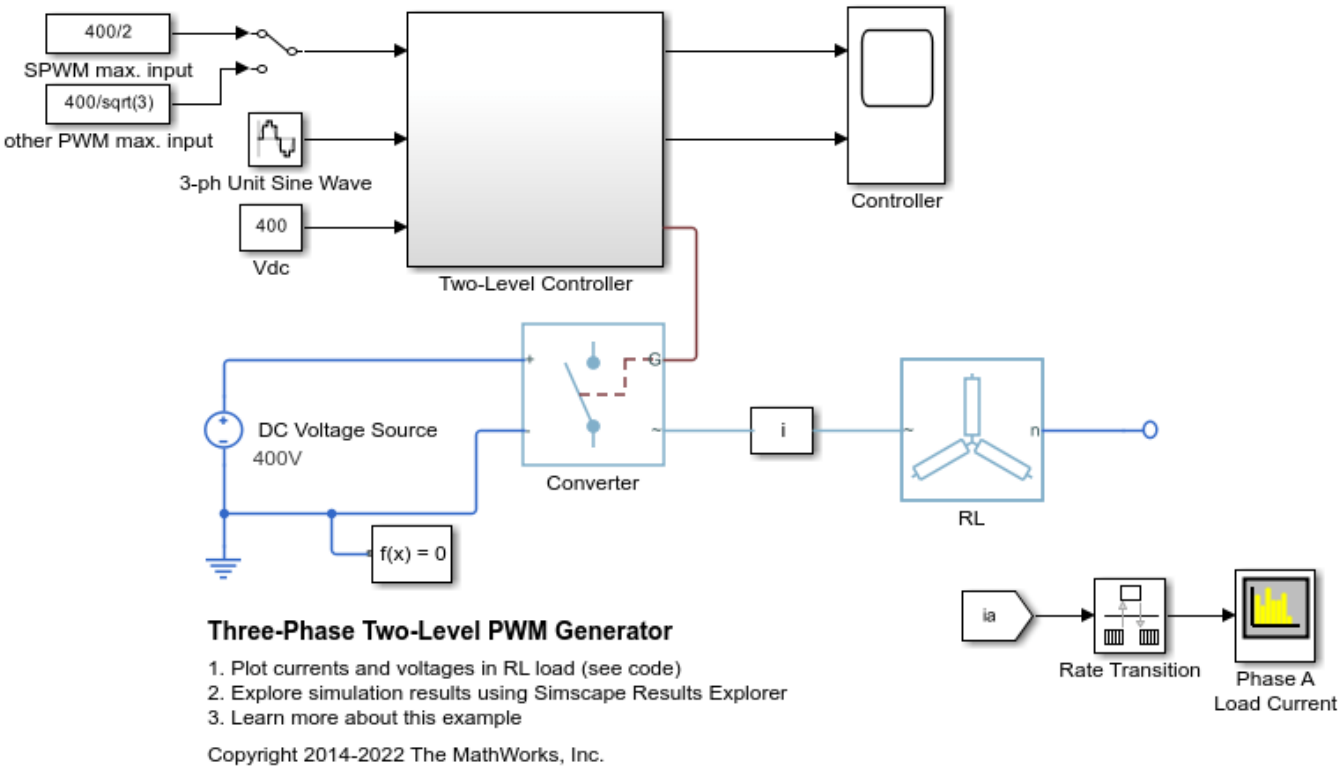
The spectrum plotted in the Spectrum Analyzer displays frequency data for the Phase A Load Current.


Phase A Load Current
File Tools View Simulation Help
Peak Finder
Settings
Peaks
Value
Freq. (kHz)
132.794 0.060
7.43559 1.938
6.7148 2.058
X: 0.06 kHz
Y: 132.794 Vrms
X: 1.94 kHz
Y: 7.436 Vrms
X: 2.06 kHz
Y: 6.715 Vrms
Vrms
Frequency (kHz)
Ready
RBW=8.998 Hz
Sample rate=10 kHz
T=0.200

# Three-Phase Two-Level PWM Generator

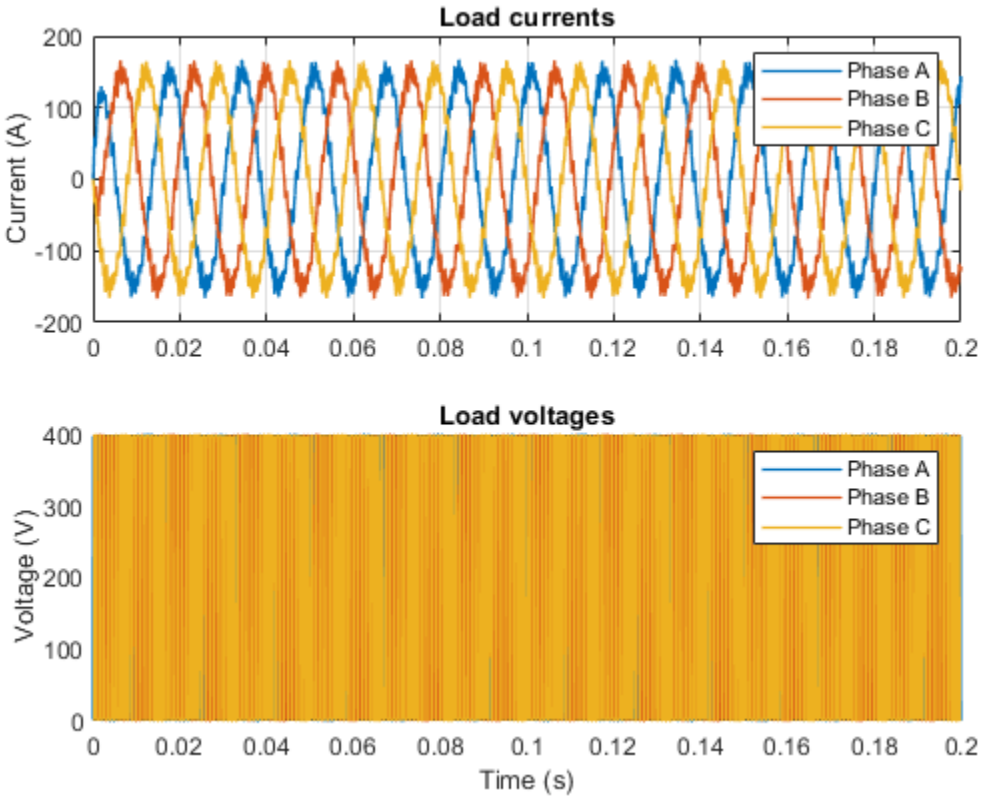
This example shows how to use the PWM Generator (Three-phase, Two-level) to control a Converter. The inputs to the PWM Generator are reference AC waveforms and a DC-link voltage of 400 V. There is one time scope for the controller waveforms.

### Model



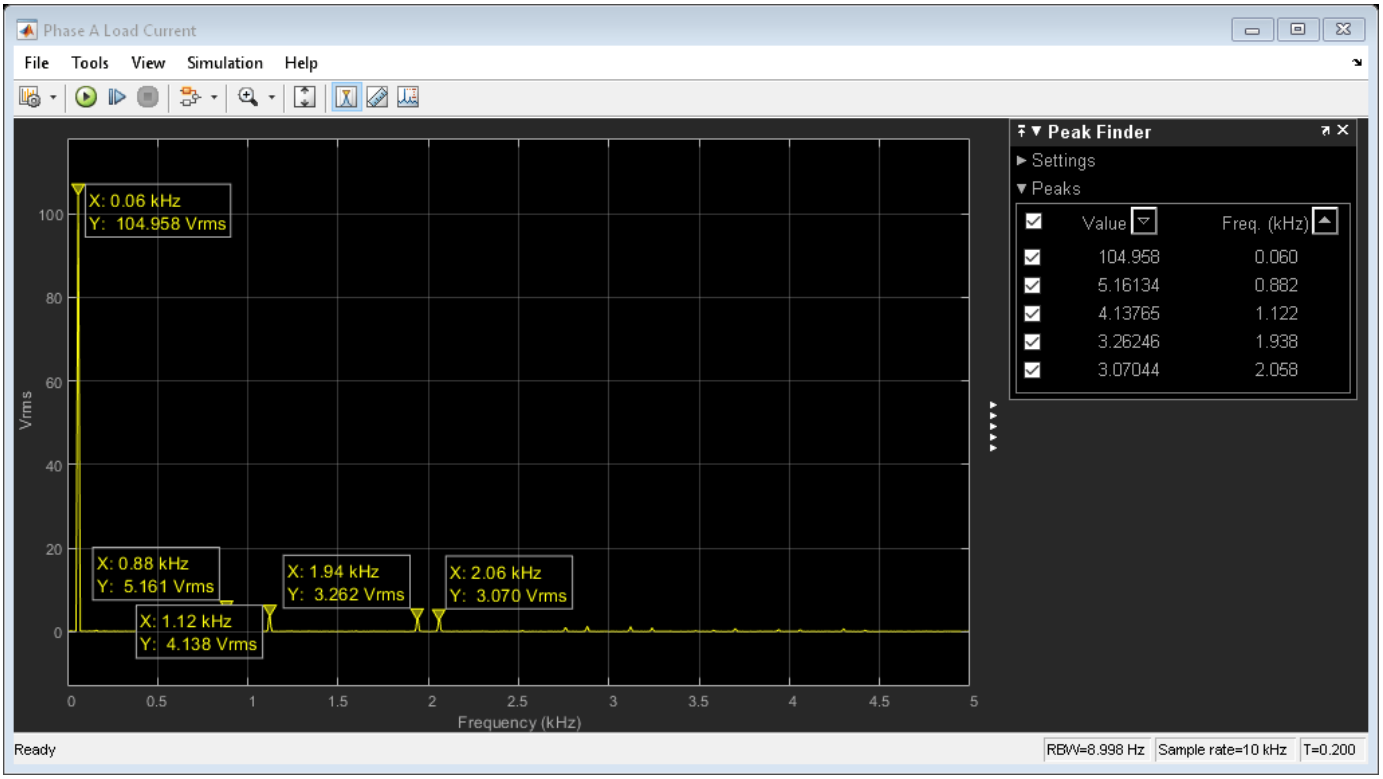

### Simulation Results from Simscape Logging

The plot below shows the load voltages and currents over time.




**Simulation Results from Scopes**

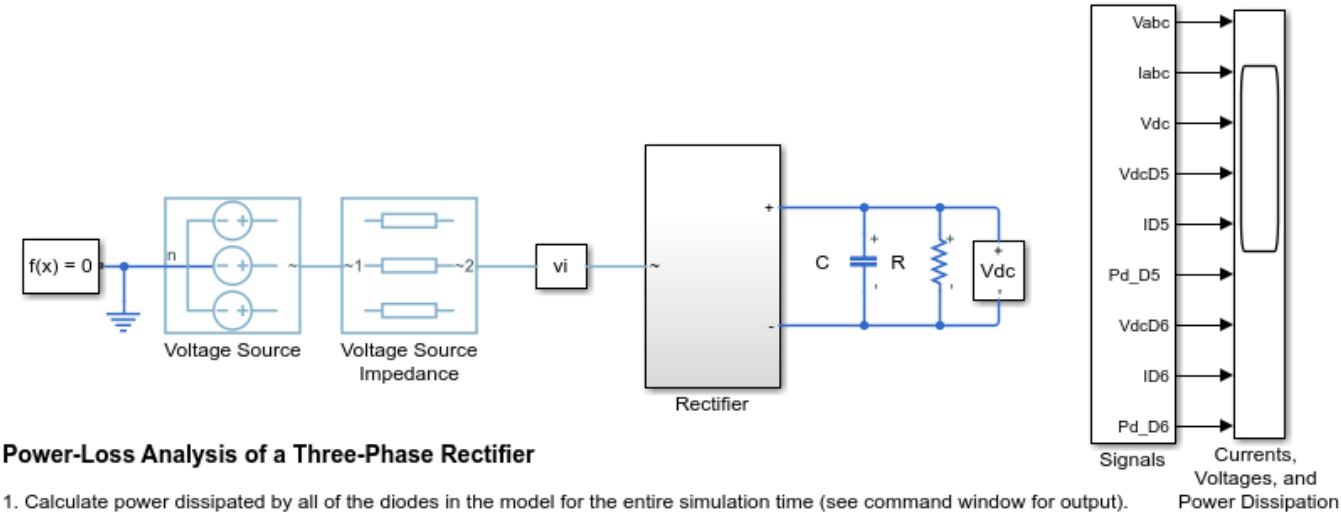
The spectrum plotted in the Spectrum Analyzer displays frequency data for the Phase A Load Current.


Phase A Load Current
File Tools View Simulation Help
Peak Finder
Settings
Peaks
Value
Freq. (kHz)
104.958 0.060
5.16134 0.882
4.13765 1.122
3.26246 1.938
3.07044 2.058
X: 0.06 kHz
Y: 104.958 Vrms
X: 0.88 kHz
Y: 5.161 Vrms
X: 1.12 kHz
Y: 4.138 Vrms
X: 1.94 kHz
Y: 3.262 Vrms
X: 2.06 kHz
Y: 3.070 Vrms
Vrms
Frequency (kHz)
Ready
RBW=8.998 Hz
Sample rate=10 kHz
T=0.200

# Power-Loss Analysis of a Three-Phase Rectifier

This example shows how to perform a power-loss analysis using the ee_getPowerLossSummary function and the power_dissipated variable. The Rectifier contains six Diode blocks. Diodes D1, D2, D4, and D5 each dissipate, on average, 52.19 W over the course of the 0.5 second simulation. Diodes D3 and D6 each dissipate an average of 52.22 W for the same time period because they experience a 340 W transient power-dissipation spike due to DC-load capacitor charging before t = 1e-3 s. For each diode, the average power dissipated when the rectifier is operating in steady-state, t = 0.4 to 0.5 s, is 52.19 W.

### Model




**Power-Loss Analysis of a Three-Phase Rectifier**

1. Calculate power dissipated by all of the diodes in the model for the entire simulation time (see command window for output).
2. Calculate power dissipated by diode D6 for the entire simulation time (see command window for output).
3. Explore simulation results using Simscape Results Explorer.
4. Calculate the maximum power dissipated by diode D6. (see command window for output).
5. Calculate power dissipated by diode D6 between simulation time, t = 1e-3 to 0.5 s (see command window for output).
6. Plot power dissipated by the diodes (see code).
7. Calculate power dissipated by all of the diodes between simulation time, t = 0.4 to 0.5 s (see command window for output).
8. Learn more about the ee_getPowerLossSummary function.

Copyright 2016-2022 The MathWorks, Inc.

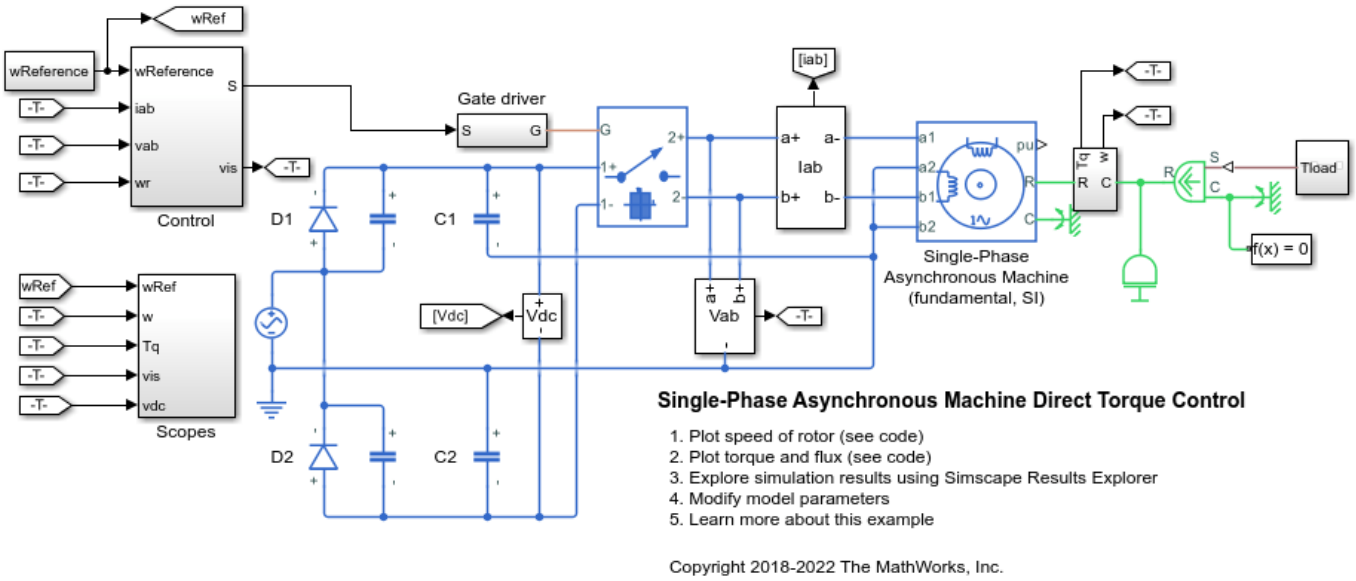
### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 20 microseconds.

# Single-Phase Asynchronous Machine Direct Torque Control

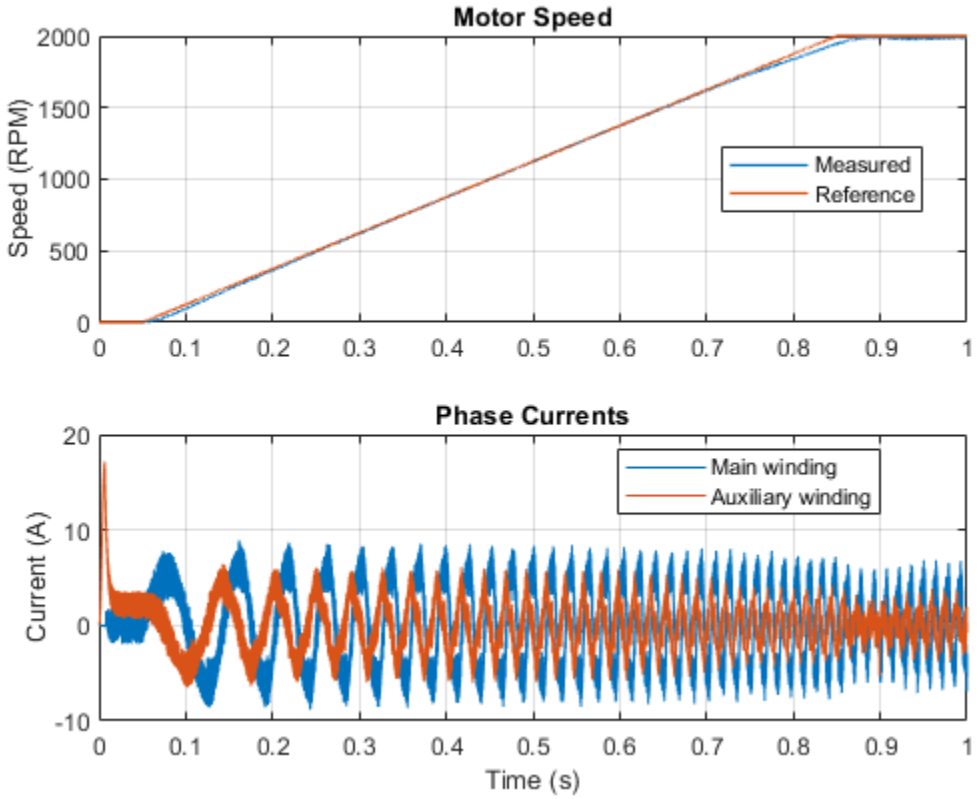
This example shows how to control the rotor speed in a single-phase asynchronous machine (ASM) based electrical drive using direct torque control. An ideal torque source provides the load. The Control subsystem uses a cascade control structure. An outer PI-based speed control loop provides the torque and flux references to the direct torque control algorithm from the inner loop. The single-phase ASM is fed by an H bridge. The Scopes subsystem contains scopes that allow you to see the simulation results.

### Model

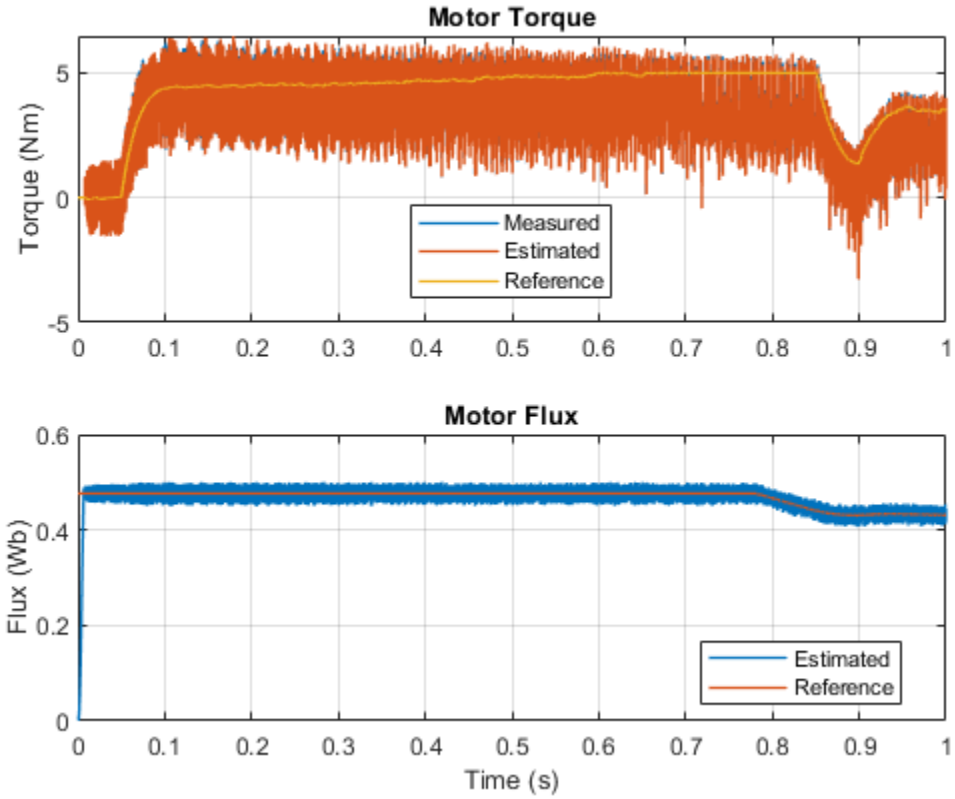



### Simulation Results from Simscape Logging

The plot below shows the requested and measured speed for the test, as well as the stator currents in the electric drive.




The plot below shows the requested and measured torque for the test, as well as the requested and estimated flux in the electric drive.



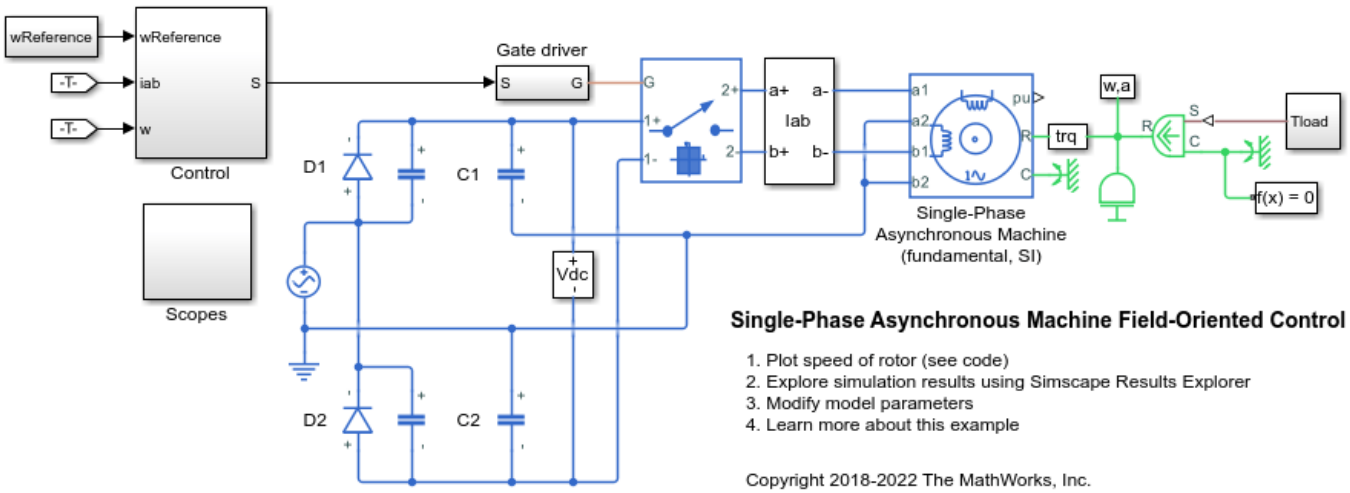

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 20 microseconds.

# Single-Phase Asynchronous Machine Field-Oriented Control

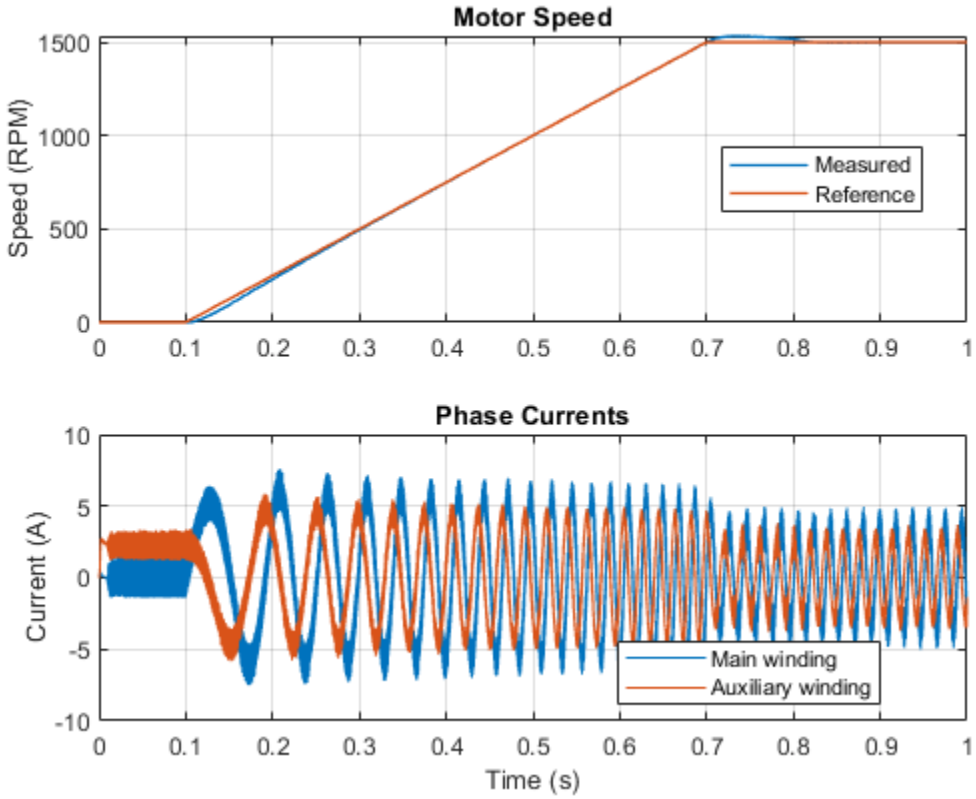
This example shows how to control the rotor speed in a single-phase asynchronous machine (ASM) based electrical drive using field-oriented control. An ideal torque source provides the load. The Control subsystem uses a PI-based cascade control structure with an outer speed control loop and two inner current control loops. The single-phase ASM is fed by an H bridge. The Scopes subsystem contains scopes that allow you to see the simulation results.

### Model




### Simulation Results from Simscape Logging

The plot below shows the requested and measured speed for the test, as well as the phase currents in the electric drive.

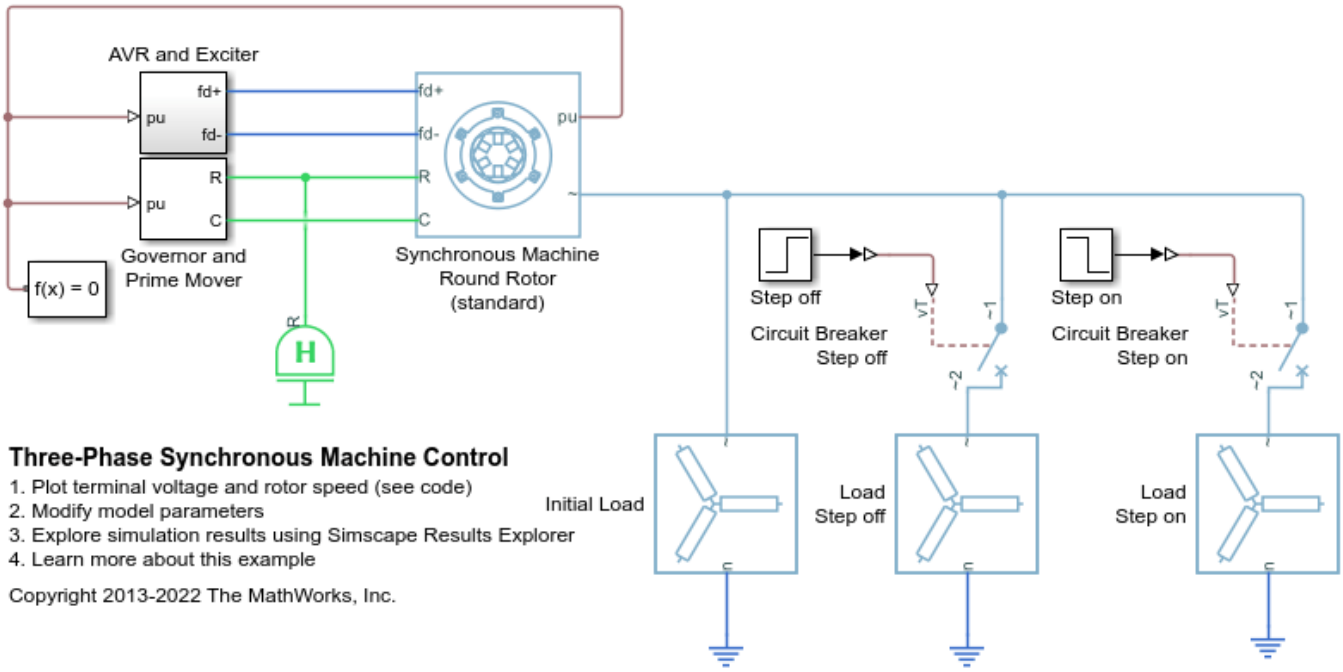

Motor Speed
Speed (RPM)
1500
1000
500
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Measured
Reference
Phase Currents
Current (A)
10
5
0
-5
-10
Main winding
Auxiliary winding
Time (s)

# Three-Phase Synchronous Machine Control

This example shows how to control and initialize a Synchronous Machine (SM). The test circuit shows the SM operating as a generator. The terminal voltage is controlled using an AVR and the speed is controlled using a governor.

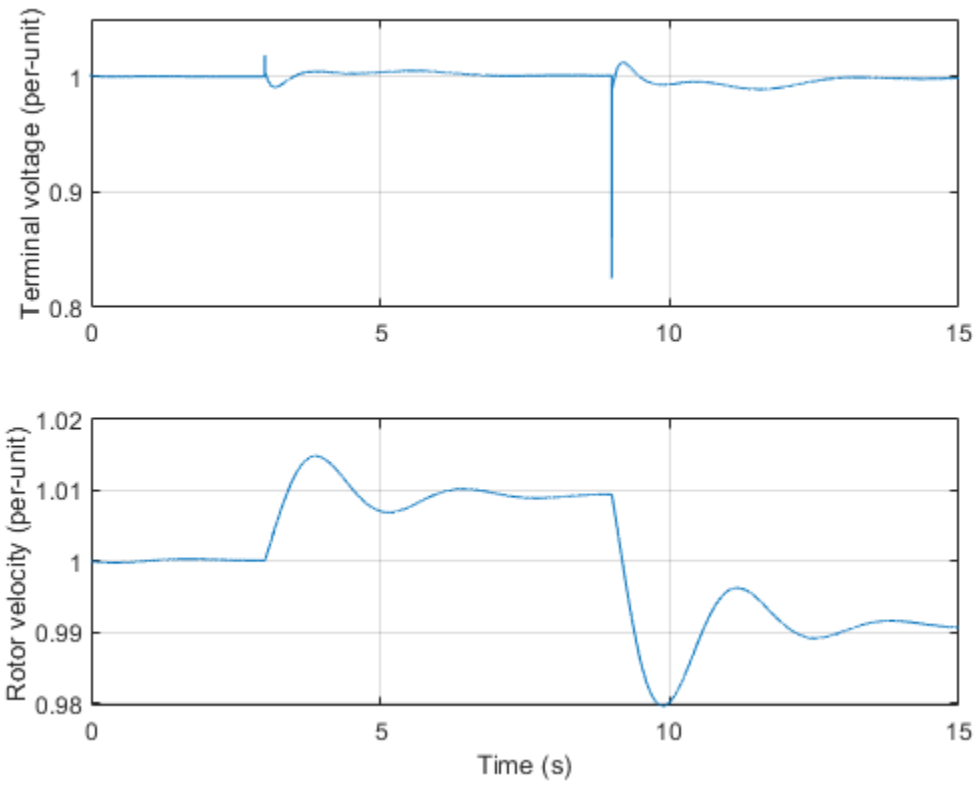
To view the SM machine base values and initial conditions, right-click the Synchronous Machine Round Rotor (standard) block and select 'Electrical' and then 'Display Base Values', 'Display Associated Base Values', or 'Display Associated Initial Conditions'. The overall model is initialized to start in periodic steady state to supply a load of 250 MW/15 Mvar.

### Model




### Simulation Results from Simscape Logging

The plot below shows the terminal voltage and rotor velocity for the Synchronous Machine.



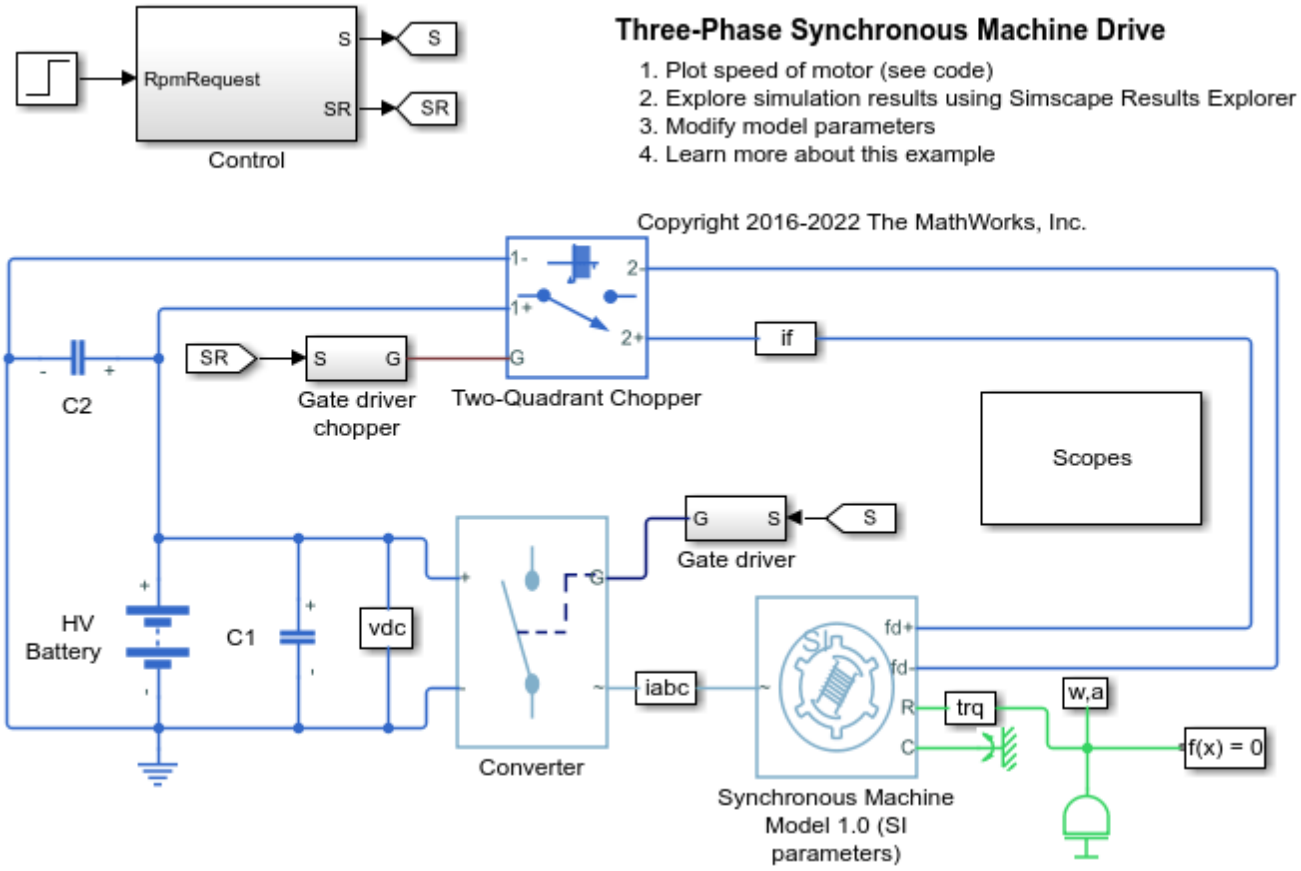

**Results from Real-Time Simulation**

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 100 microseconds.

# Three-Phase Synchronous Machine Drive

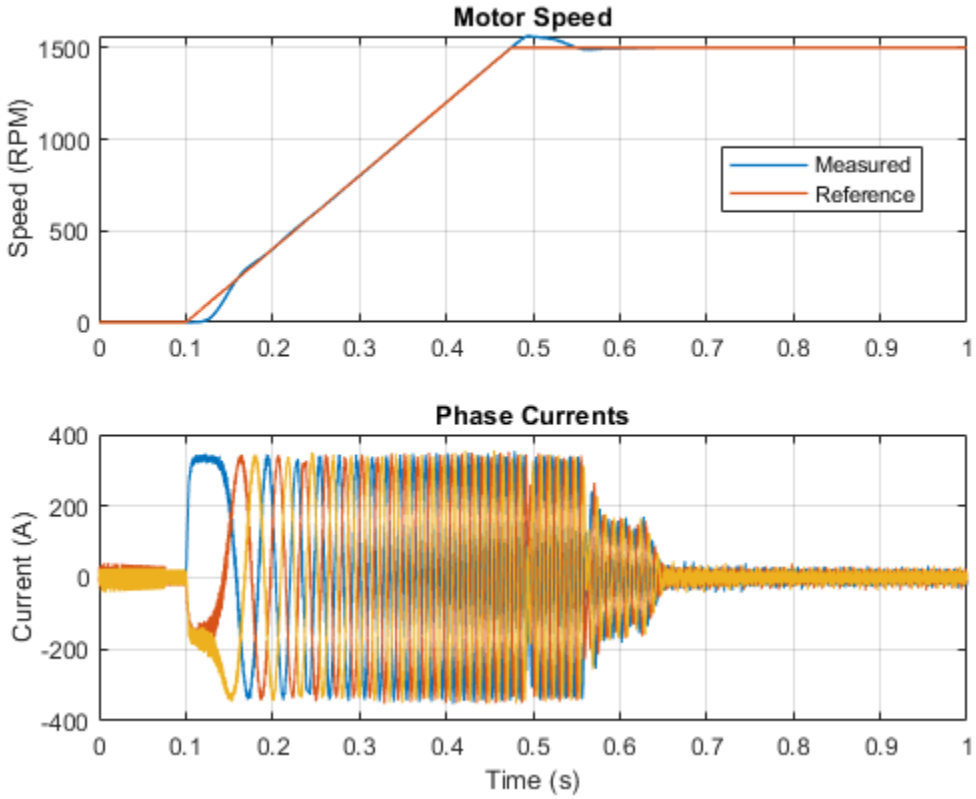
This example shows how to control the rotor speed in a Synchronous Machine (SM) based electrical drive. A high-voltage battery feeds the SM through a controlled three-phase converter for the stator windings and through a controlled two-quadrant chopper for the rotor winding. Use the model to design the SM controller, selecting architecture and gains to achieve desired performance. The Scopes subsystem contains scopes that allow you to see the simulation results.

### Model




### Simulation Results from Simscape Logging

The plot below shows the requested and measured speed for the test, as well as the phase currents in the electric drive.


Motor Speed
Speed (RPM)
1500
1000
500
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Measured
Reference
Phase Currents
Current (A)
400
200
0
-200
-400
Time (s)

# Three-Phase Synchronous Machine Governor Control Design

This example script shows how you can linearize a Simscape™ Electrical™ model to support control system stability analysis and design. It uses example model ee_sm_governor_control_design.

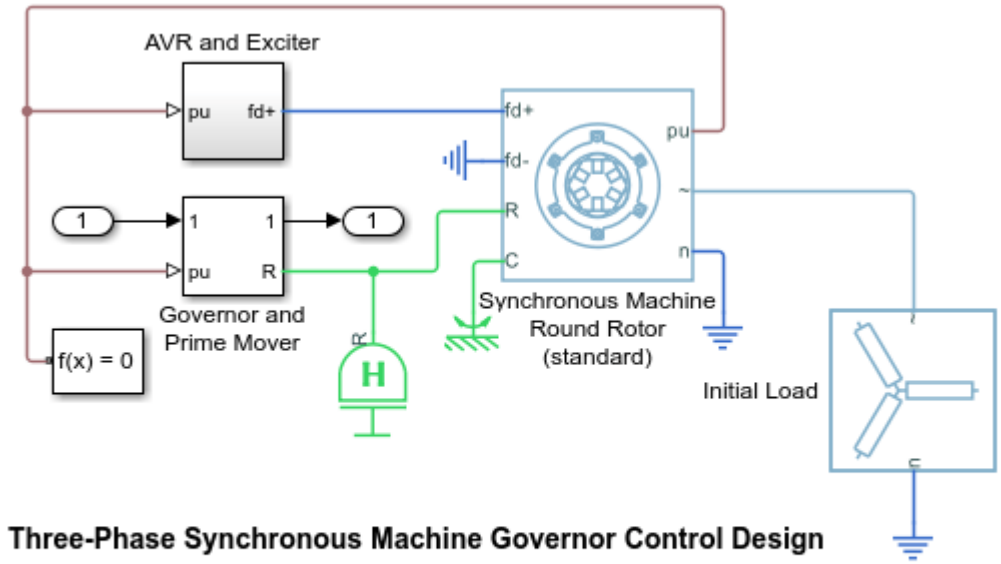
The Synchronous Machine in this example operates as a generator. The machine is initialized to start in periodic steady state to supply a load of 250 MW/15 Mvar.

An alternative and recommended way to linearize Simulink® and Simscape models is to use Simulink Control Design™. Simulink Control Design linearizes your model at operating points you specify. It also returns a state-space model object with state names. If you have Simulink Control Design, open the model ee_sm_governor_control_design. On the Apps tab, under Control Systems, click Model Linearizer. In the Model Linearizer, on the Linear Analysis tab, in the Setup section, select Operating Point > Linearize At. Set the simulation snapshot time to 4 seconds, then click OK. In the Linearize section, click Bode.

### Open Model

Open the model.

```
open_system('ee_sm_governor_control_design')
set_param(find_system('ee_sm_governor_control_design','FindAll', 'on','type','annotation','Tag',
```




### Trim Model

Trim the model by running closed-loop and selecting the state at 4 seconds when generator is in steady-state.

```
assignin('base','ClosedLoop',1); % Close the speed-control feedback loop
[t,x,y] = sim('ee_sm_governor_control_design');
idx = find(t>4,1);
X = x(idx,:); U = y(idx);
```

### Linearize Speed-Control Feedback Loop

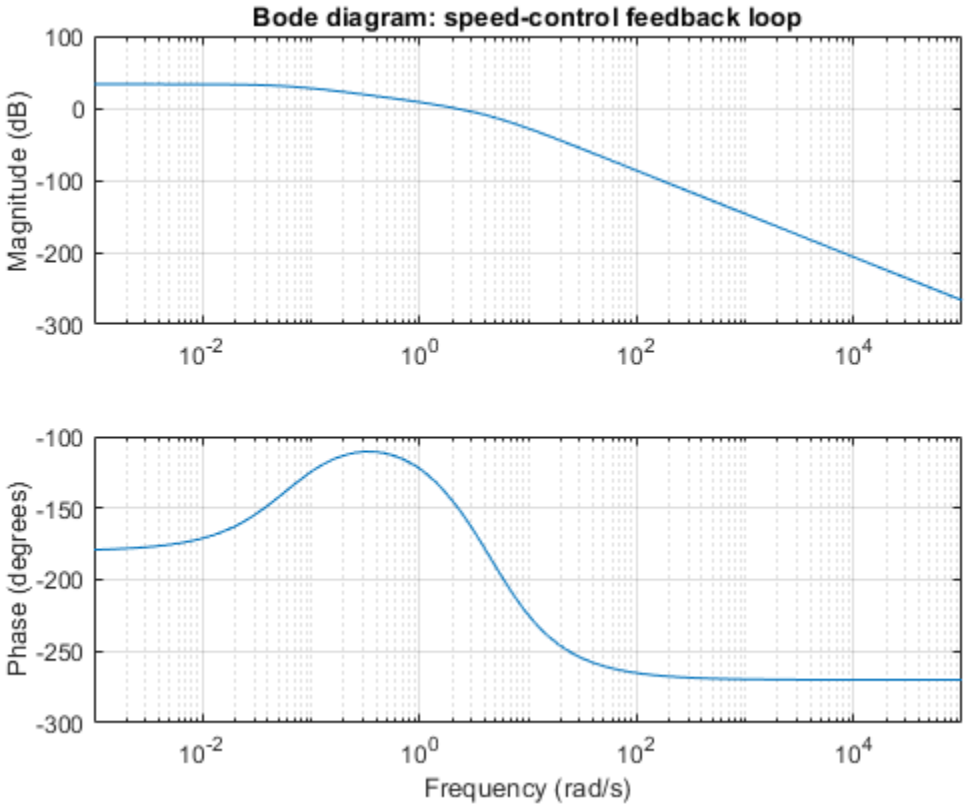
Open the speed-control feedback loop, linearize the model, and close the speed-control feedback loop.

```
assignin('base','ClosedLoop',0); % Open the speed-control feedback loop
[a,b,c,d]=linmod('ee_sm_governor_control_design',X,U);
assignin('base','ClosedLoop',1); % Close the speed-control feedback loop
```

### Bode Diagram

Plot the Bode diagram.

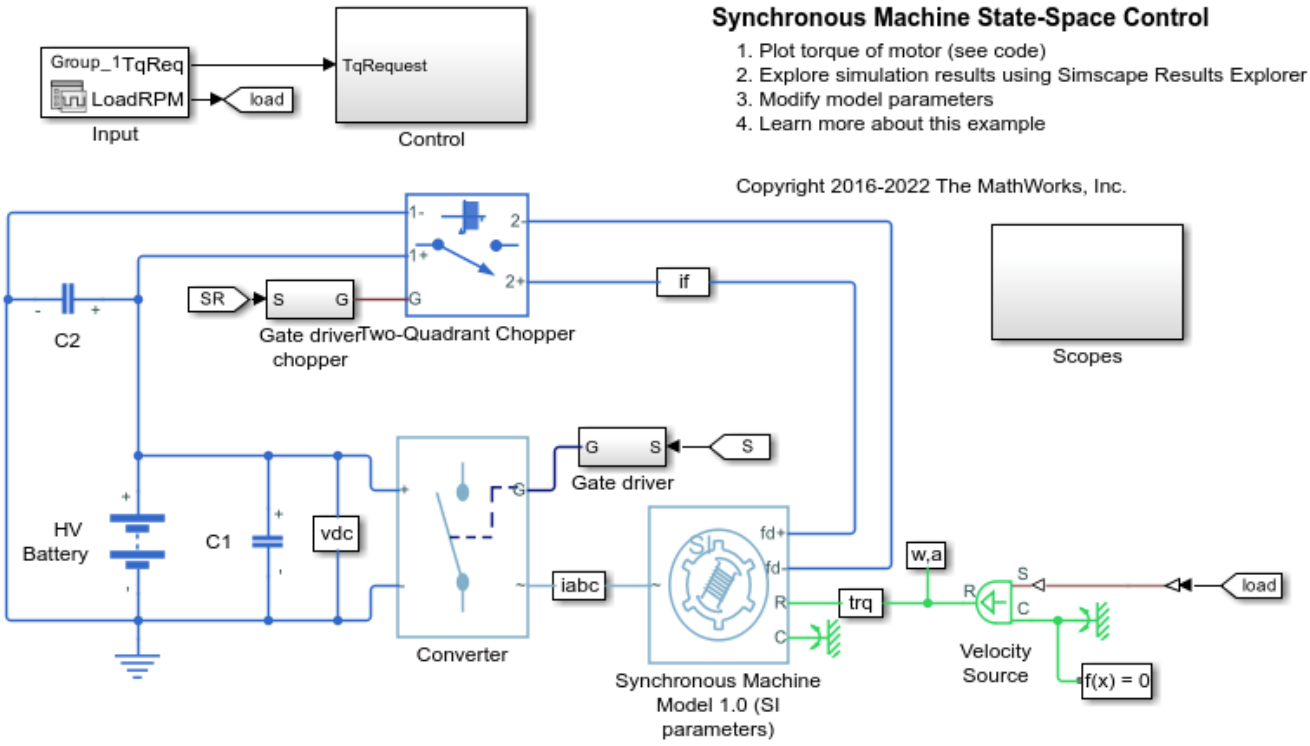
```
c = -c; d = -d; % Negative feedback convention
npts = 100; w = logspace(-3,5,npts); G = zeros(1,npts);
for i=1:npts
    G(i) = c*(1i*w(i)*eye(size(a))-a)^-1*b +d;
end
figure
ax1 = subplot(2,1,1);
semilogx(w,20*log10(abs(G)))
grid on
ylabel('Magnitude (dB)')
title('Bode diagram: speed-control feedback loop')
ax2 = subplot(2,1,2);
semilogx(w,180/pi*unwrap(angle(G)))
ylabel('Phase (degrees)')
xlabel('Frequency (rad/s)')
linkaxes([ax1,ax2],'x')
xlim(w([1 end]))
grid on
```


Bode diagram: speed-control feedback loop
Magnitude (dB)
100
0
-100
-200
-300
$10^{-2}$
$10^{0}$
$10^{2}$
$10^{4}$
Phase (degrees)
-100
-150
-200
-250
-300
$10^{-2}$
$10^{0}$
$10^{2}$
$10^{4}$
Frequency (rad/s)

# Synchronous Machine State-Space Control

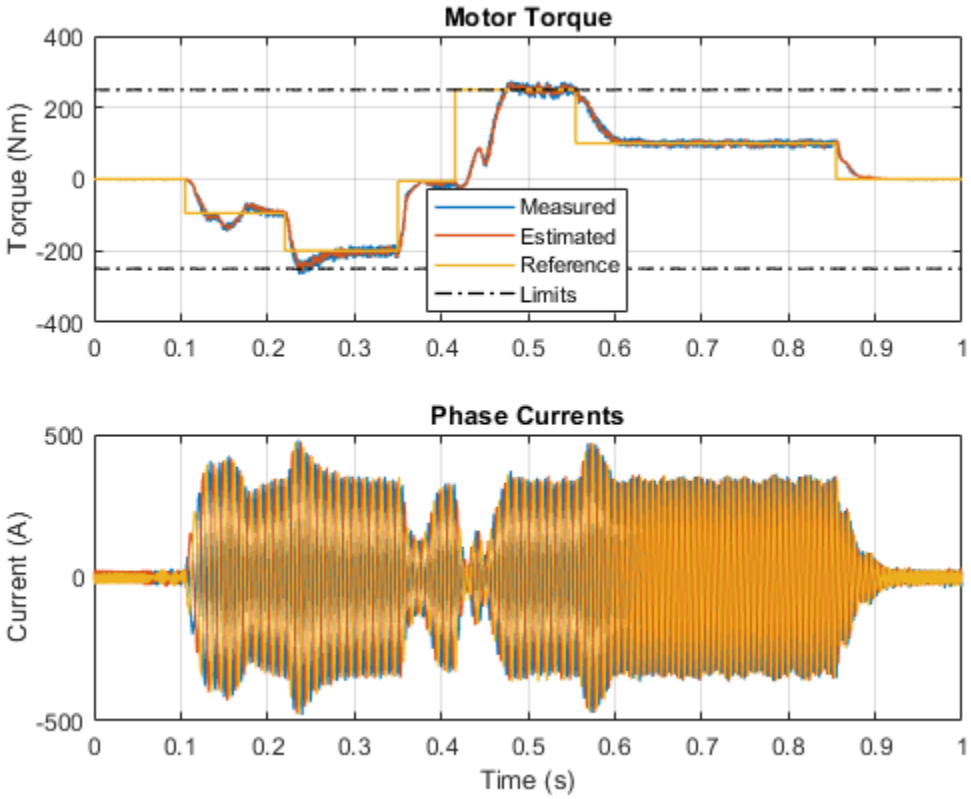
This example shows how to control currents in a synchronous machine (SM) based traction drive using state-space control. A high-voltage battery feeds the SM through a controlled three-phase converter for the stator windings and through a controlled two-quadrant chopper for the rotor winding. An ideal angular velocity source provides the load. The SM operates below the base speed. At each sample instant, the torque request is converted to relevant current references using the zero d-axis control approach. A state-feedback controller controls the currents in the rotor reference frame. A Luenberger observer obtains the velocity-dependent feedforward pre-control terms. The simulation uses several torque steps in both motor and generator modes. The task scheduling is implemented as a Stateflow® state machine. The Scopes subsystem contains scopes that allow you to see the simulation results.

### Model




### Simulation Results from Simscape Logging

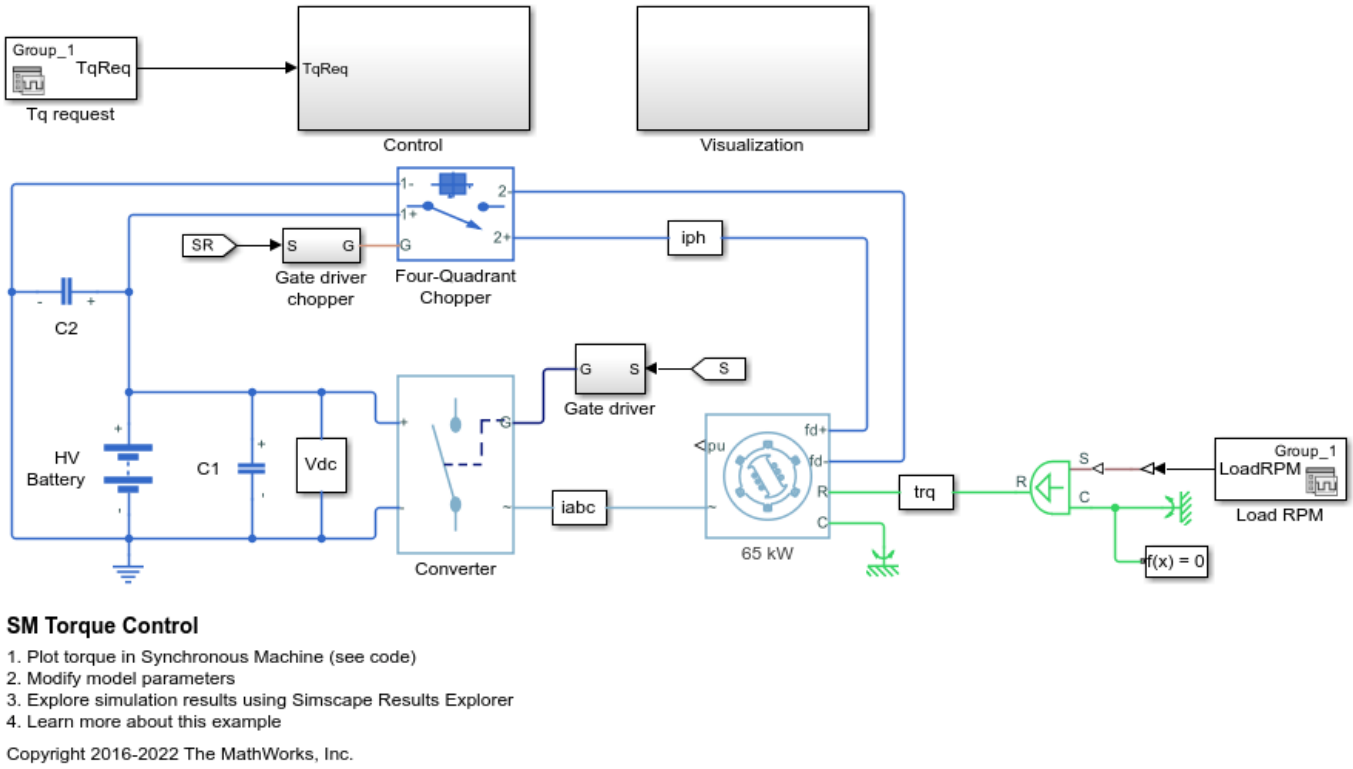
The plot below shows the requested and measured torque for the test, as well as the phase currents in the electric drive.


Motor Torque
Torque (Nm)
400
200
0
-200
-400
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Measured
Estimated
Reference
Limits
Phase Currents
Current (A)
500
0
-500
Time (s)

# SM Torque Control

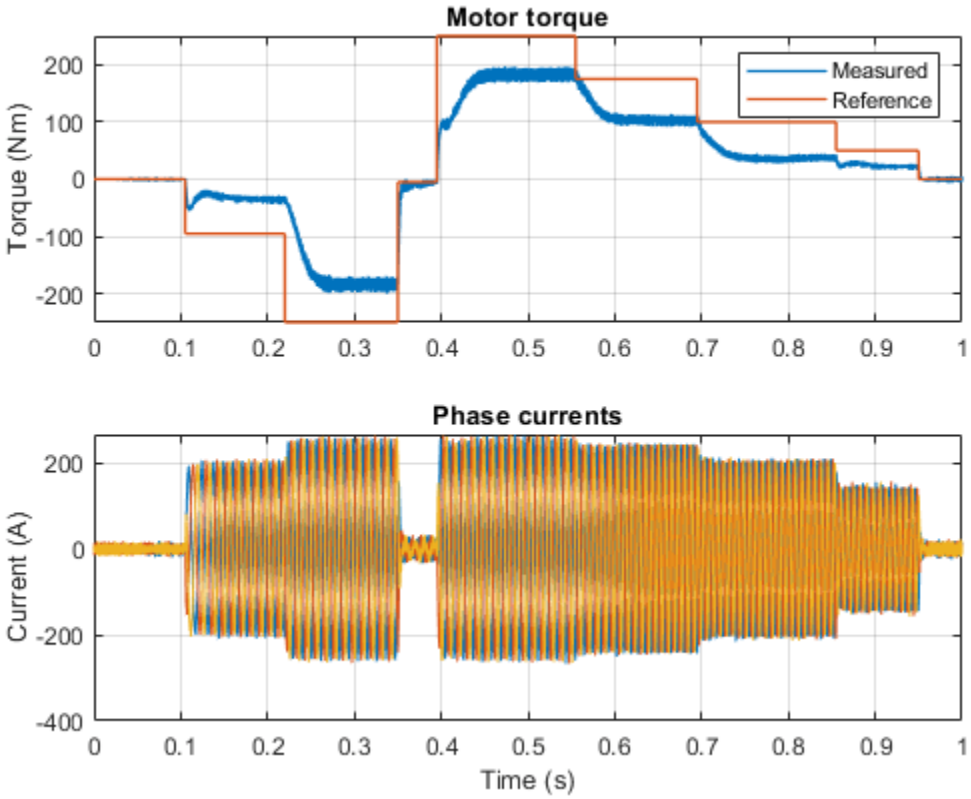
This example shows how to control the torque in a synchronous machine (SM) based electrical-traction drive. A high-voltage battery feeds the SM through a controlled three-phase converter for the stator windings and a controlled four quadrant chopper for the rotor winding. An ideal angular velocity source provides the load. The Control subsystem uses an open-loop approach to control the torque and a closed-loop approach to control the current. At each sample instant, the torque request is converted to relevant current references. The current control is PI-based. The simulation uses several torque steps in both motor and generator modes. The task scheduling is implemented as a Stateflow® state machine. The Visualization subsystem contains scopes that allow you to see the simulation results.

### Model




### Simulation Results from Simscape Logging

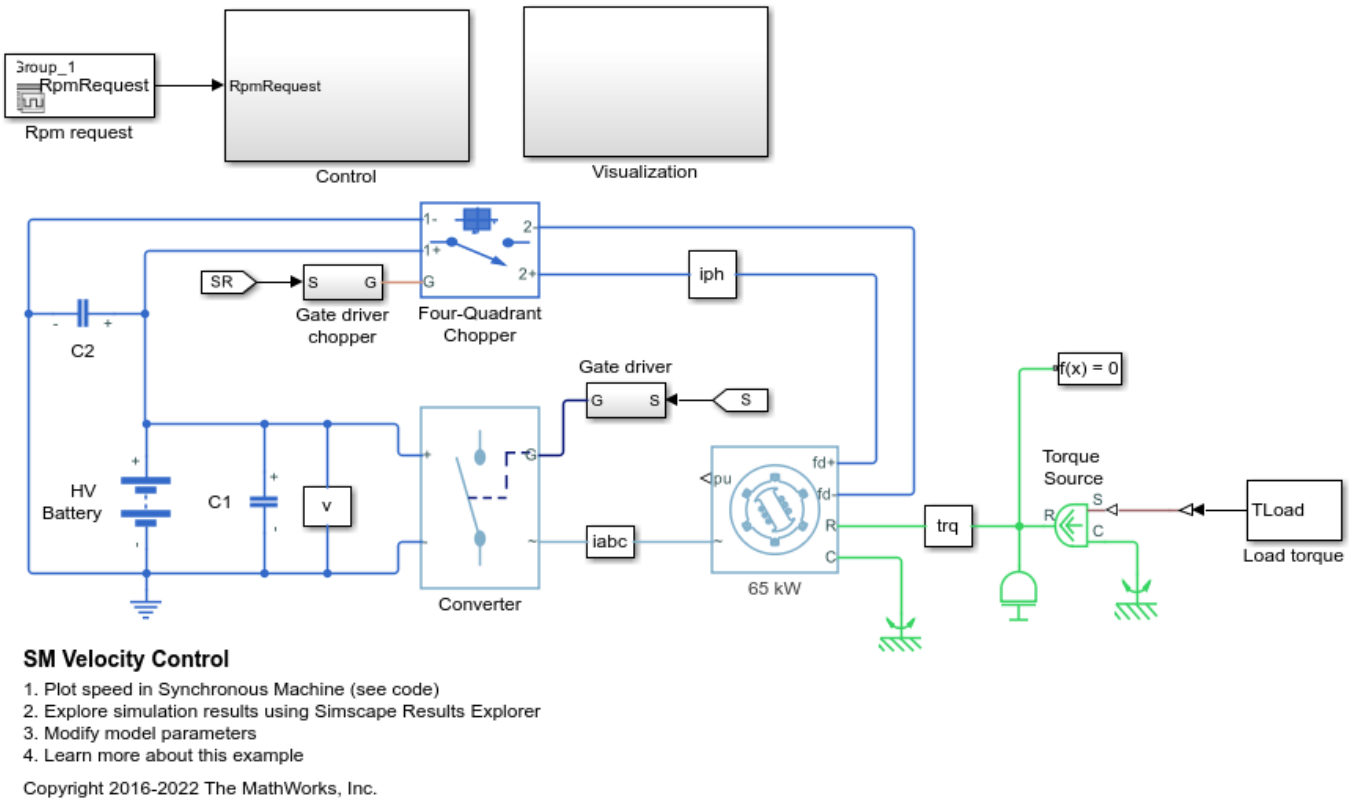
The plot below shows the requested and measured torque for the test, as well as the phase currents in the electric drive.


Motor torque
Measured
Reference
Torque (Nm)
200
100
0
-100
-200
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Phase currents
Current (A)
200
0
-200
-400
Time (s)

# SM Velocity Control

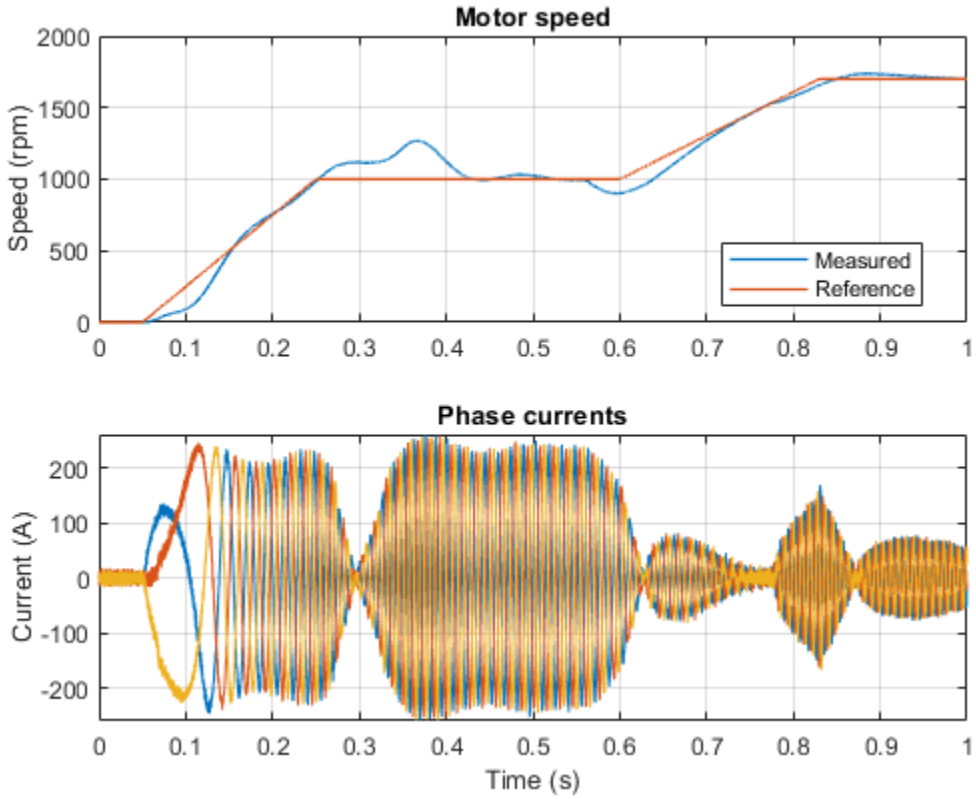
This example shows how to control the rotor angular velocity in a synchronous machine (SM) based electrical-traction drive. A high-voltage battery feeds the SM through a controlled three-phase converter for the stator windings and a controlled four quadrant chopper for the rotor winding. An ideal torque source provides the load. The Control subsystem includes a multi-rate PI-based cascade control structure which has an outer angular-velocity-control loop and three inner current-control loops. The task scheduling in the Control subsystem is implemented as a Stateflow® state machine. The Visualization subsystem contains scopes that allow you to see the simulation results.

### Model




### Simulation Results from Simscape Logging

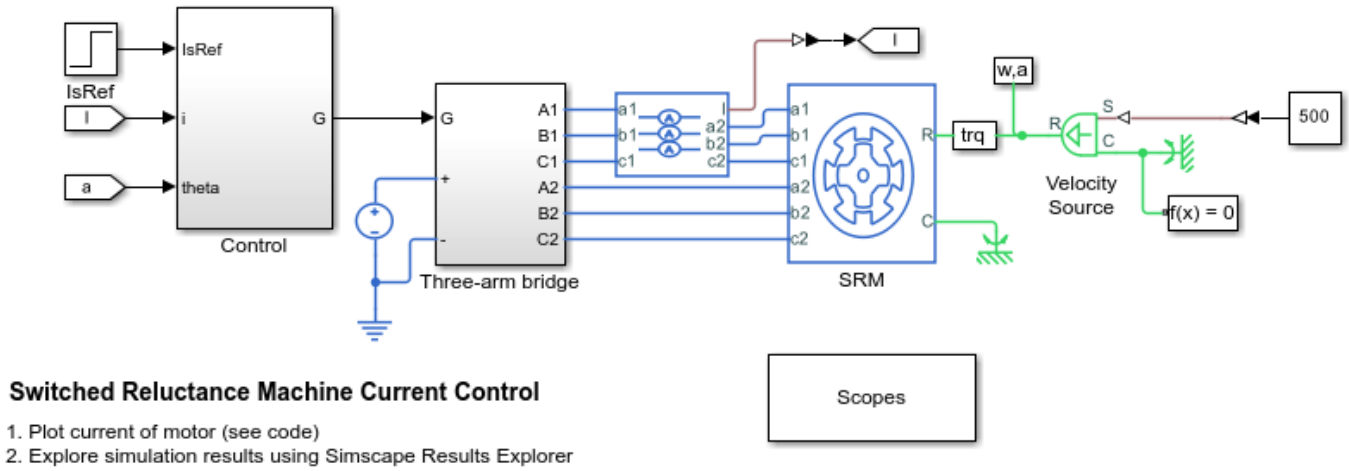
The plot below shows the requested and measured speed for the test, as well as the phase currents in the electric drive.


Motor speed
Speed (rpm)
2000
1500
1000
500
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Measured
Reference
Phase currents
Current (A)
200
100
0
-100
-200
Time (s)

# Switched Reluctance Machine Current Control

This example shows how to control the current amplitude in a switched reluctance machine (SRM) based electrical drive. A DC voltage source feeds the SRM through a controlled three-arm bridge. An ideal angular velocity source provides the load. The converter turn-on and turn-off angles are maintained constant. A PI-based current controller regulates the current amplitude.

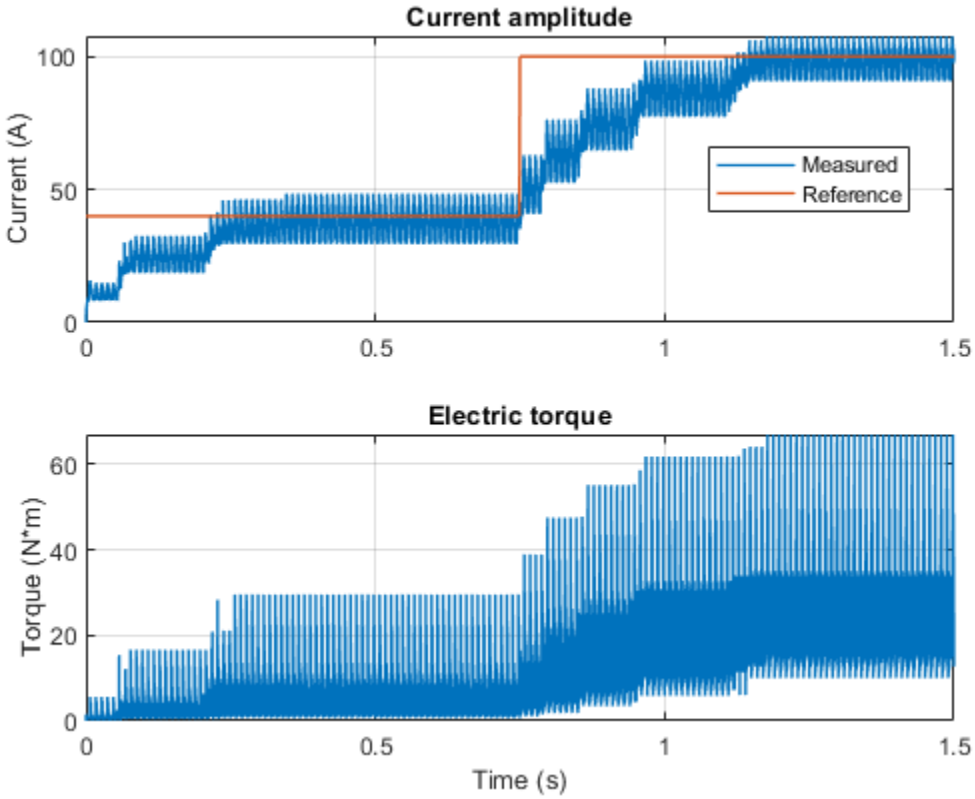
### Model




Copyright 2013-2022 The MathWorks, Inc.

### Simulation Results from Simscape Logging

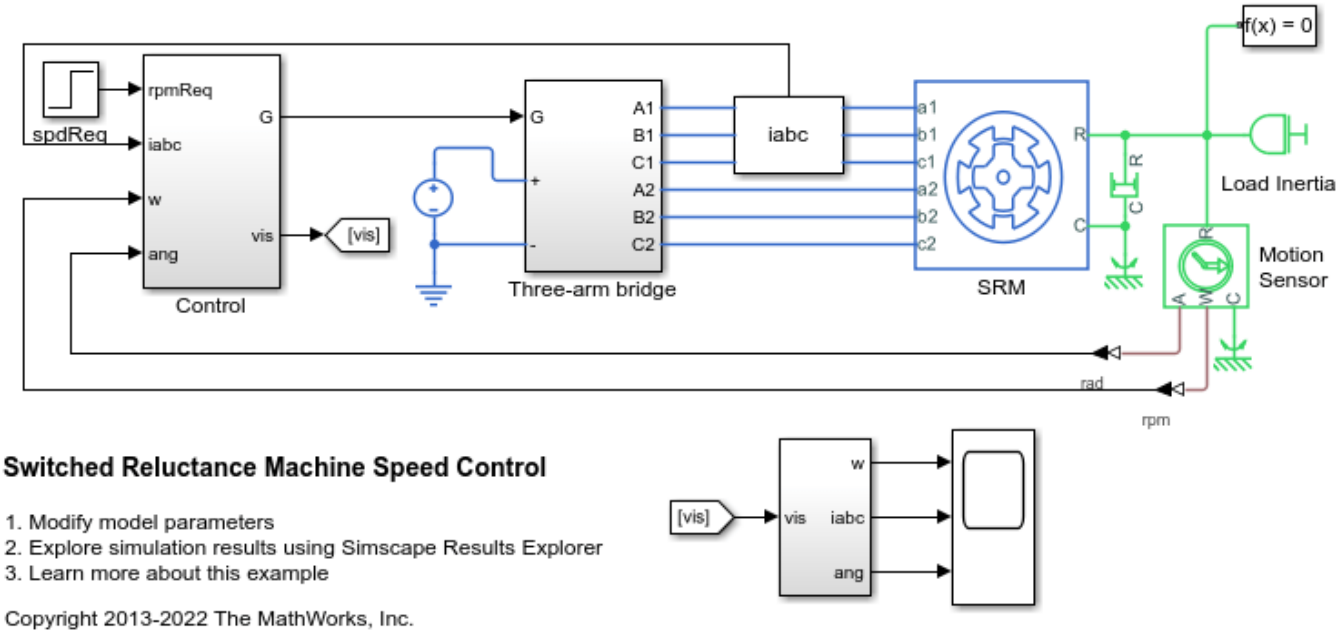
The plot below shows the requested and measured current for the test and the electric torque in the electric drive.


Current amplitude
Current (A)
100
50
0
0
0.5
1
1.5
Measured
Reference
Electric torque
Torque (N*m)
60
40
20
0
0
0.5
1
1.5
Time (s)

# Switched Reluctance Machine Speed Control

This example shows how to control the rotor speed in a switched reluctance machine (SRM) based electrical drive. A DC voltage source feeds the SRM through a controlled three-arm bridge. The converter turn-on and turn-off angles are maintained constant.

### Model



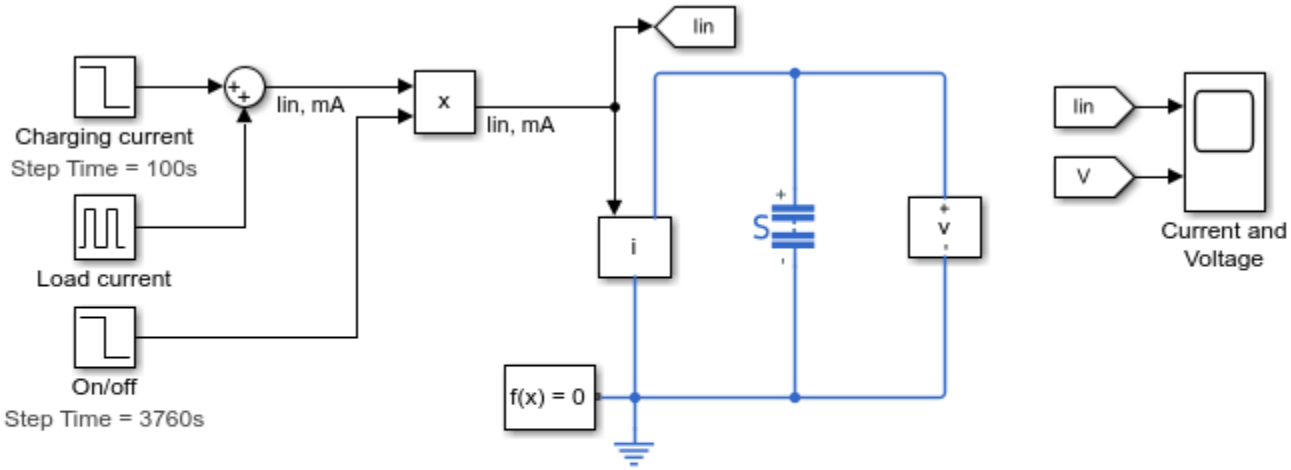

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 80 microseconds.

# Supercapacitor Charging and Discharging Behavior

This example shows the voltage output by a Supercapacitor block as it is charged and then discharged. To charge the Supercapacitor, a current of 100 mA is input to the Supercapacitor for 100 seconds. The Supercapacitor is then rested for one minute. For the next hour, to discharge the Supercapacitor, a load of 50 mA is stepped on for one second in every 50 seconds. The Supercapacitor is then rested until the end of the simulation. The scope displays the Supercapacitor charging/discharging current and voltage.

**Model**




**Supercapacitor Charging and Discharging Behavior**

1. Explore simulation results using Simscape Results Explorer
2. Learn more about this example

Copyright 2016-2022 The MathWorks, Inc.

# Supercapacitor Parameter Identification

This example shows how to identify the parameters of a supercapacitor. Instead of collecting voltage and current waveforms from a real supercapacitor, the example generates voltage and current waveforms by running a simulation of a supercapacitor using parameter values that are already known. Then the example applies a parameter identification methodology [1] to the waveforms.

To assess the accuracy of the methodology, the example compares the identified parameters to the known parameter values. The example also shows how to further refine the parameter values using the Parameter Estimation tool provided by Simulink® Design Optimization™.

To identify the parameters of an actual supercapacitor empirically, you can:

1. Collect voltage and current waveforms from the supercapacitor.
2. Identify parameter values using the waveform data and the methodology described in [1].

To identify the parameters of a modeled supercapacitor, this example:

1. Generates voltage and current waveforms by simulating a model that it configures using known values for supercapacitor parameters.
2. Identifies supercapacitor parameter values using the generated waveform data and the methodology described in [1].
3. Configures and simulates the supercapacitor using the identified supercapacitor parameter values.

To see how the approach works for a real supercapacitor, evaluate the accuracy of the identification methodology by comparing:

- the data generated using known parameter values and the data generated using identified parameter values.
- the known parameter values and identified parameter values.

If the accuracy is not sufficient, you can use the Parameter Estimation tool from Simulink Design Optimization to improve it. Use the identified parameter values as the starting values for the optimization.

### Generate Data

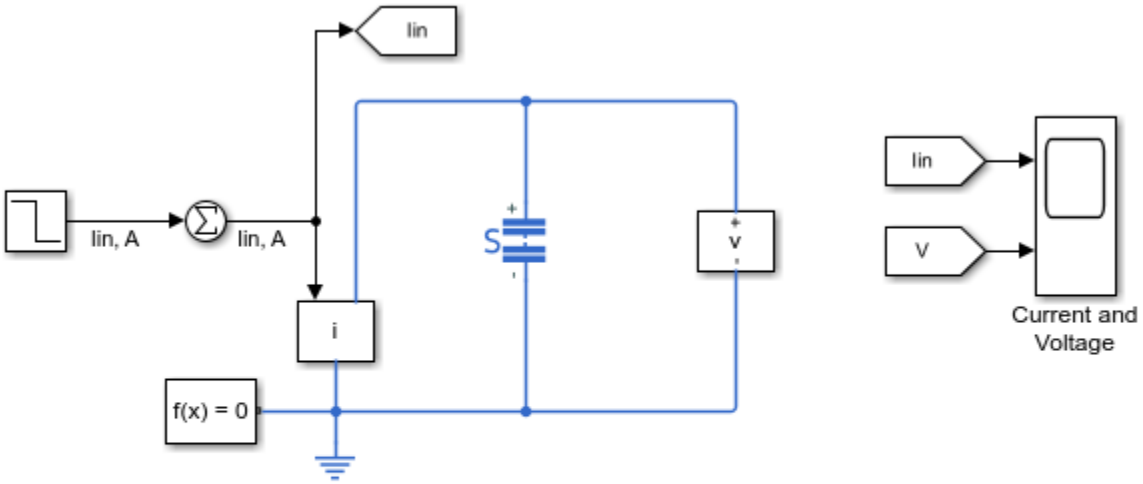
Generate voltage and current waveforms by configuring and simulating a model using known values for the fixed resistances, fixed capacitances, and voltage-dependent capacitor gain parameters of the supercapacitor. Model the input current, configure the physical characteristics and self-discharge resistance of the supercapacitor using realistic values.

Observe the supercapacitor behavior during three distinct phases:

1. Charge with constant current
2. Charge redistribution from immediate to delayed branch
3. Charge redistribution from immediate and delayed branches to long-term branch

```
% Open model
modelName = 'ee_supercapacitor_identification';
```

```
open_system(modelName);
set_param(find_system('ee_supercapacitor_identification','FindAll', 'on','type','annotation','Tag

% Configure the supercapacitor using known parameter values.
%
% Supercapacitor block Cell Characteristics parameters
Kv = 190;                    % Voltage-dependent capacitor gain
R = [2.5e-3 0.9 5.2];        % Fixed resistances, [R1 R2 R3]
C = [270 100 220];           % Fixed capacitances, [C1 C2 C3]

% Store a copy of the known parameter values for an eventual comparison to
% parameter values identified using the methodology described in [1].
Kv_known = Kv;
R_known = R;
C_known = C;

% Specify input current by configuring the Step block.
%
% Step block parameters
stepTime = [40 1900 1917];  % Step time
initialValue = [28 0 0];    % Initial value
finalValue = [0 -25 25];    % Final value

% Specify the physical characteristics of the supercapacitor.
%
% Supercapacitor block Cell Characteristics and Configuration parameters
R_discharge = 9e3;           % Self-discharge resistance
N_series = 1;                % Number of series cells
N_parallel = 1;              % Number of parallel cells

% Specify the simulation duration
stopTime = 2100;

% Simulate and store voltage and current waveform data
sim(modelName);
t = simlog_ee_supercapacitor_identification.Sensing_current.It.i.series.time;
i = simlog_ee_supercapacitor_identification.Sensing_current.It.i.series.values;
v = simlog_ee_supercapacitor_identification.Sensing_voltage.V.V.series.values;
```




**Supercapacitor Parameter Identification**

Identify and validate the supercapacitor parameters (see code) Explore supercapacitor data from the simulation that uses identified parameter values using Simscape Results Explorer Explore all simulation results from the simulation that uses identified parameter values using Simscape Results Explorer Learn more about the Supercapacitor block

Copyright 2016-2022 The MathWorks, Inc.

### Perform Parameter Identification

Using the waveform data from the simulation, apply the methodology described in [1].

### Immediate Branch Parameter Identification

During the first stage of the identification, a fully discharged supercapacitor is charged with constant current. The method assumes that the immediate branch stores all the initial charge because the time constant for the branch is relatively small.

Immediate branch parameters are calculated by taking measurements of the charge characteristic. Once charging current has reached steady-state at $t_1 = 20e^{-3}\mathrm{s}$, measure $v_1$ and use

$$R_i = \frac{v_1}{i_1}$$

where:

- $t_1$ is time at parameter identification event $n = 1$, $\mathrm{s}$
- $v_1$ is terminal voltage at $t_1$, $\mathrm{V}$
- $i_1$ is charging current at $t_1$, $\mathrm{A}$
- $R_i$ is fixed resistance of immediate branch, $\Omega$

Once voltage has increased from $v_1$ by approximately $50e^{-3}\mathrm{V}$, measure $t_2$ and $v_2$ and use

$$C_{i0} = i_1 \frac{t_2 - t_1}{v_2 - v_1}$$

where:

- $t_2$ is time at parameter identification event $n = 2$, $\mathrm{s}$
- $v_2$ is terminal voltage at $t_2$, $\mathrm{V}$
- $C_{i0}$ is fixed capacitance of immediate branch, $\mathrm{F}$

Once voltage has reached rated voltage, measure $v_3$ and $t_3$, and turn off charging current. After $20e^{-3}\mathrm{s}$, once charging current has reached steady-state value of $0\mathrm{A}$ measure $t_4$ and $v_4$ and use

$$t_4 = t_3 + 20e^{-3}$$

$$C_{i1} = \frac{2}{v_4}\left(\frac{i_1(t_4 - t_1)}{v_4} - C_{i0}\right)$$

where:

- $t_3$ is time at parameter identification event $n = 3$, $\mathrm{s}$
- $v_3$ is terminal voltage at $t_3$, $\mathrm{V}$
- $t_4$ is time at parameter identification event $n = 4$, $\mathrm{s}$
- $v_4$ is terminal voltage at $t_4$, $\mathrm{V}$
- $C_{i1}$ is voltage-dependent capacitance coefficient, $\mathrm{F/V}$

```
% Event n=1
t1 = 20e-3;
i1 = interp1(t,i,t1);
v1 = interp1(t,v,t1);

Ri = v1/i1;

% Extract charging data that interp1 can use to find time values rather
% than voltage values
[v3, v_max_idx] = max(v);
[v_charge, v_charge_idx] = unique(v(1:v_max_idx));
t_charge = t(v_charge_idx);

% Event n=2
delta_v = 50e-3;
v2 = v1+delta_v;
t2 = interp1(v_charge,t_charge,v2);
i2 = interp1(t,i,t2);

Ci0 = i2*(t2-t1)/delta_v;

% Event n=3
t3 = t(v_max_idx);

% Event n=4
delta_t = 20e-3;
t4 = t3+delta_t;
v4 = interp1(t,v,t4);
Qtotal = i1*(t4-t1);
Cq = Qtotal/v4;

Ci1 = (2/v4)*(Cq-Ci0);
```

### Delayed Branch Parameter Identification

During the second stage of the identification, charge redistributes from the immediate branch to the delayed branch.

Delayed branch parameters are calculated by taking measurements of the hold charge characteristic.

Once voltage has decreased from $v_4$ by approximately $50e^{-3}\mathrm{V}$, measure $t_5$ and $v_5$ and use

$$\Delta v = v_4 - v_5$$

$$V_{ci} = v_4 - \frac{\Delta V}{2}$$

$$C_{diff} = C_{i0} + C_{i1}V_{ci}$$

$$R_d = \frac{(v_4 - \frac{\Delta V}{2})(t_5 - t_4)}{C_{diff}\Delta V}$$

where:

- $t_5$ is time at parameter identification event $n = 5$, $\mathrm{s}$
- $v_5$ is terminal voltage at $t_5$, $\mathrm{V}$
- $V_{ci}$ is voltage at which the total immediate capacitance is to be calculated, $\mathrm{V}$
- $C_{diff}$ is total immediate capacitance at $V_{ci}$, $\mathrm{F}$
- $R_d$ is delayed branch resistance, $\Omega$

Wait for 300 seconds, measure $t_6$ and $v_6$ and use

$$Q_{total} = i_1(t_4 - t_1)$$

$$C_d = \frac{Q_{total}}{v_6} - (C_{i0} + \frac{C_{i1}}{2}v_6)$$

where:

- $t_6$ is time at parameter identification event $n = 6$, $\mathrm{s}$
- $v_6$ is terminal voltage at $t_6$, $\mathrm{V}$
- $Q_{total}$ is total charge supplied to the supercapacitor, $\mathrm{C}$
- $C_d$ is delayed branch capacitance, $\mathrm{F}$

```
% Event n=5
v5 = v4-delta_v;
discharge_idx = find(i<-20,1,'first');
t_discharge = t(v_max_idx:discharge_idx);
v_discharge = v(v_max_idx:discharge_idx);
t5 = interp1(v_discharge,t_discharge,v5);
delta_t = t5-t4;
Vci = v4-(delta_v/2);
Cdiff = Ci0+Ci1*Vci;
Rd = (v4-(delta_v/2))*delta_t/(Cdiff*delta_v);
```

```
% Event n=6
TypicallyRdTimesCd = 100;
t6 = t5 + 3*TypicallyRdTimesCd;
v6 = interp1(t,v,t6);
Cd = (Qtotal/v6)-(Ci0+((Ci1/2)*v6));
```

### Long-term Branch Parameter Identification

During the third, and final, stage of the identification, charge redistributes from the immediate and delayed branches to the long-term branch.

Long-term branch parameters are calculated by taking measurements of the hold charge characteristic.

Once voltage has decreased from $v_6$ by approximately $50e^{-3}\mathrm{V}$, measure $t_7$ and $v_7$ and use

$$\Delta v = v_6 - v_7$$

$$R_l = \frac{(v_6 - \frac{\Delta V}{2})(t7 - t_6)}{C_{diff}\Delta V}$$

where:

- $t_7$ is time at parameter identification event $n = 7$, $\mathrm{s}$
- $v_7$ is terminal voltage at $t_7$, $\mathrm{V}$
- $R_l$ is long-term branch resistance, $\Omega$

After 30 minutes from the start of the charging/discharging process, measure $t_8$ and $v_8$ and use

$$C_l = \frac{Q_{total}}{v_8} - (C_{i0} + \frac{C_{i1}}{2} v_8) - C_d$$

where:

- $t_8$ is time at parameter identification event $n = 8$, $\mathrm{s}$
- $v_8$ is terminal voltage at $t_8$, $\mathrm{V}$
- $C_l$ is long-term branch capacitance, $\mathrm{F}$

```
% Event n=7
v7 = v6-delta_v;
t7 = interp1(v_discharge,t_discharge,v7);
delta_t = t7-t6;
Vci = v6-(delta_v/2);
Cdiff = Ci0+Ci1*Vci;
Rl = (v6-(delta_v/2))*delta_t/(Cdiff*delta_v); % Rl value large

% Event n=8
t8 = 30*60;
v8 = interp1(t,v,t8);
Cl = (Qtotal/v8)-(Ci0+(v8*Ci1/2))-Cd; % Cl value too large
```

### Collate Time and Voltage Data

Collate the time and voltage data for each parameter identification event in a MATLAB® table:

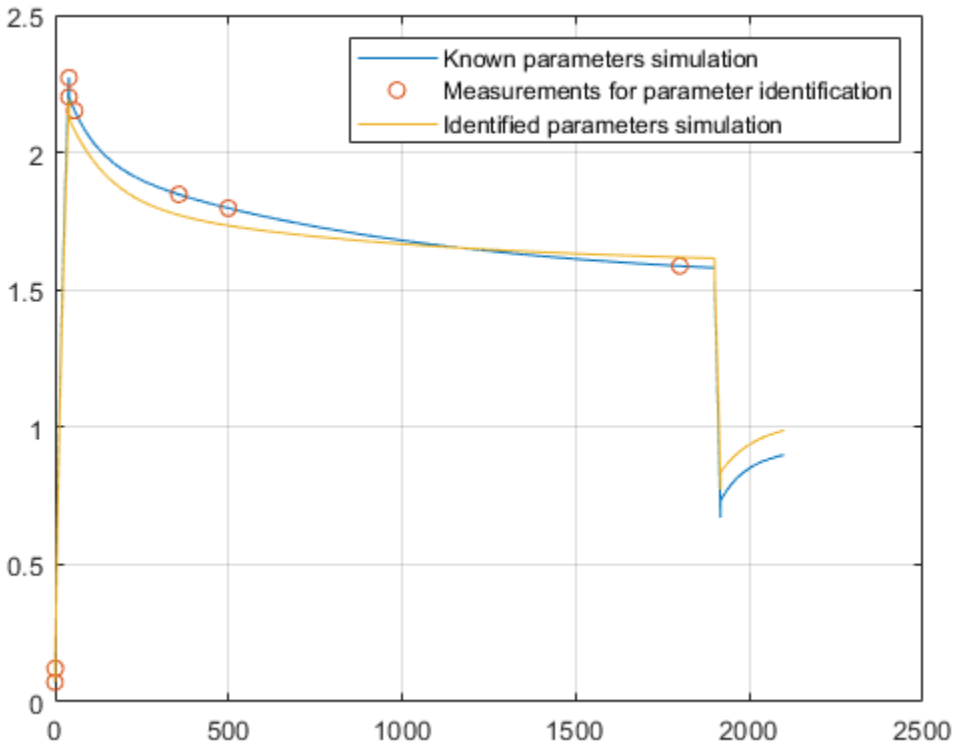
```
DataTable = table((1:8)',...
    [t1 t2 t3 t4 t5 t6 t7 t8]',...
    [v1 v2 v3 v4 v5 v6 v7 v8]',...
    'VariableNames',{'Event','Time','Voltage'}) %#ok<NOPTS>
```

```
DataTable =

  8x3 table

    Event     Time      Voltage
    _____    _______    ________

      1         0.02    0.071799
      2      0.51803      0.1218
      3           40      2.2717
      4        40.02      2.2019
      5       56.675      2.1519
      6       356.67      1.8473
      7       499.28      1.7973
      8         1800      1.5865
```

### Evaluate Accuracy of Identified Parameters

Configure and simulate the model using the identified supercapacitor parameters. Then, to evaluate the accuracy of the identified parameter values, compare the waveform output to the data generated by a simulation that uses known parameters.

```
% Create parameters for supercapacitor block using identified values and
% create copies for easy reference.
Kv = Ci1;
R = [Ri Rd Rl];
C = [Ci0 Cd Cl];
Kv_identified = Kv;
R_identified = R;
C_identified = C;

% Simulate the model and store voltage and current waveforms.
sim(modelName);
t_ = simlog_ee_supercapacitor_identification.Sensing_current.It.i.series.time;
v_ = simlog_ee_supercapacitor_identification.Sensing_voltage.V.V.series.values;

% Plot the data generated from simulating with known parameters on the
% same axis as the data generated from simulating with identified
% parameters.
figure;
plot(t,v,DataTable.Time,DataTable.Voltage,'o',t_,v_);
grid('on');
legend('Known parameters simulation',...
    'Measurements for parameter identification',...
    'Identified parameters simulation');
```




### Optimization

Prerequisites:

1 Simulink Design Optimization license

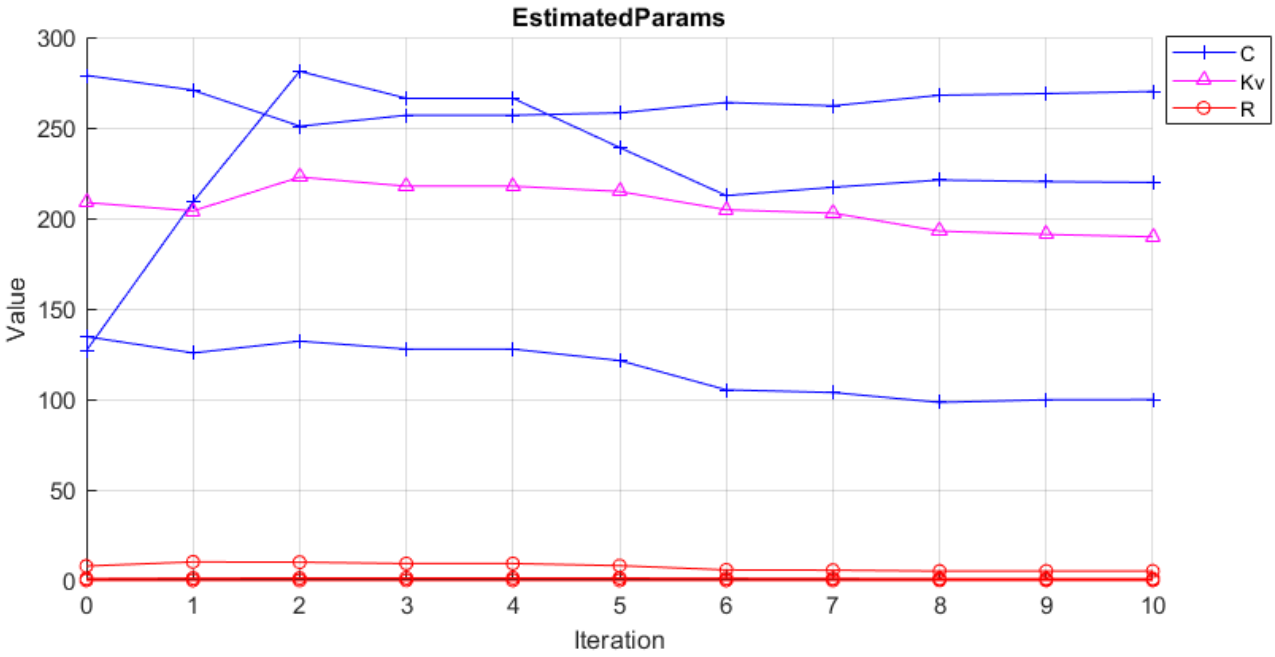
If the traces are not similar enough to meet requirements, then the identified parameter values can serve as initial parameter values for the Parameter Estimation tool from Simulink Design Optimization.

- spetool('ee_supercapacitor_spesession');

### Optimization Results

Display the parameter trajectory created using the **Sum Squared Error** cost function and **Nonlinear least squares** optimization method.

```
openfig('ee_supercapacitor_speresults');
```



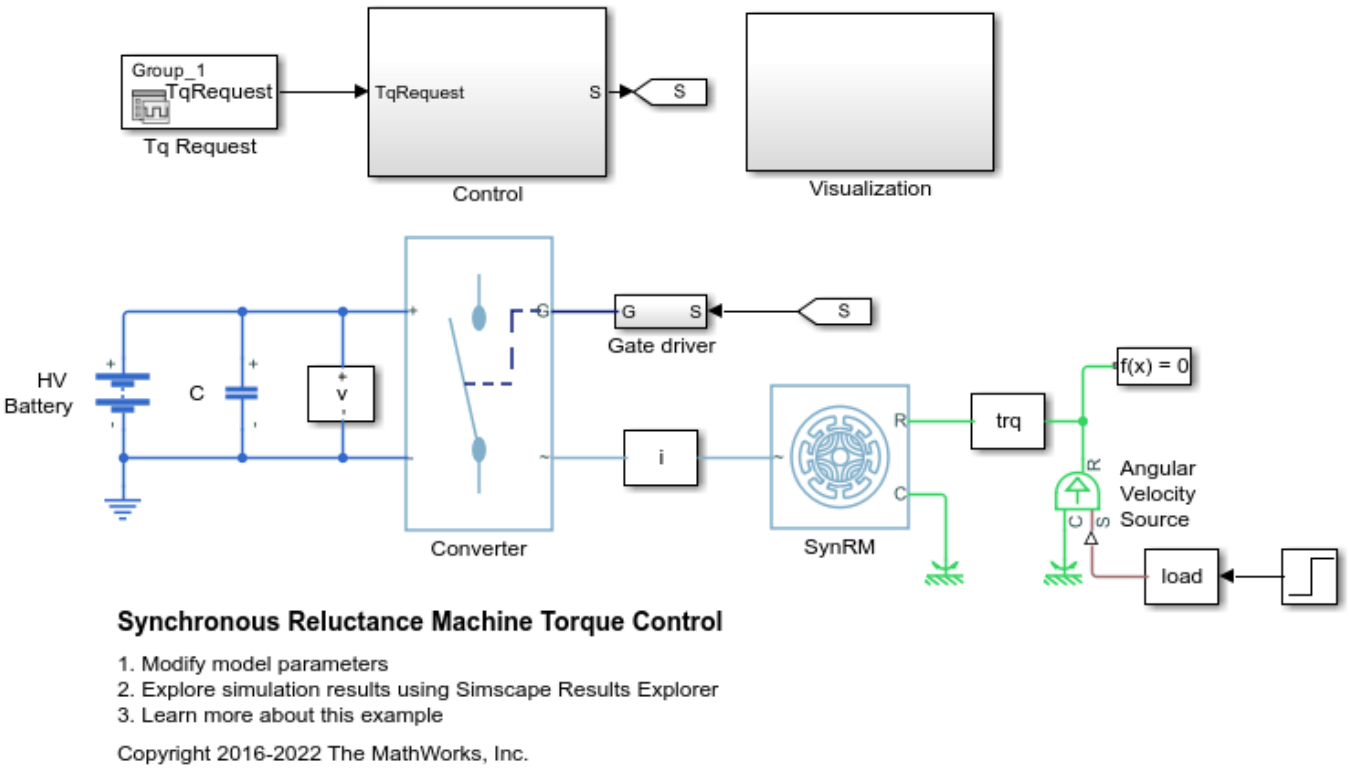

## References

[1] Zubieta, L. and R. Bonert. "Characterization of Double-Layer Capacitors for Power Electronics Applications." IEEE Transactions on Industry Applications, Vol. 36, No. 1, 2000, pp. 199-205.

# Synchronous Reluctance Machine Torque Control

This example shows how to control the torque in a synchronous reluctance machine (SynRM) based electrical drive. A high-voltage battery feeds the SynRM through a controlled three-phase converter. An ideal angular velocity source provides the load. The Control subsystem uses an open-loop approach to control the torque and a closed-loop approach to control the current. At each sample instant, the torque request is converted to relevant current references using the maximum torque per Ampere strategy. The current control is PI-based. The simulation uses torque steps in both the motor and generator modes. The Visualization subsystem contains scopes that allow you to see the simulation results.

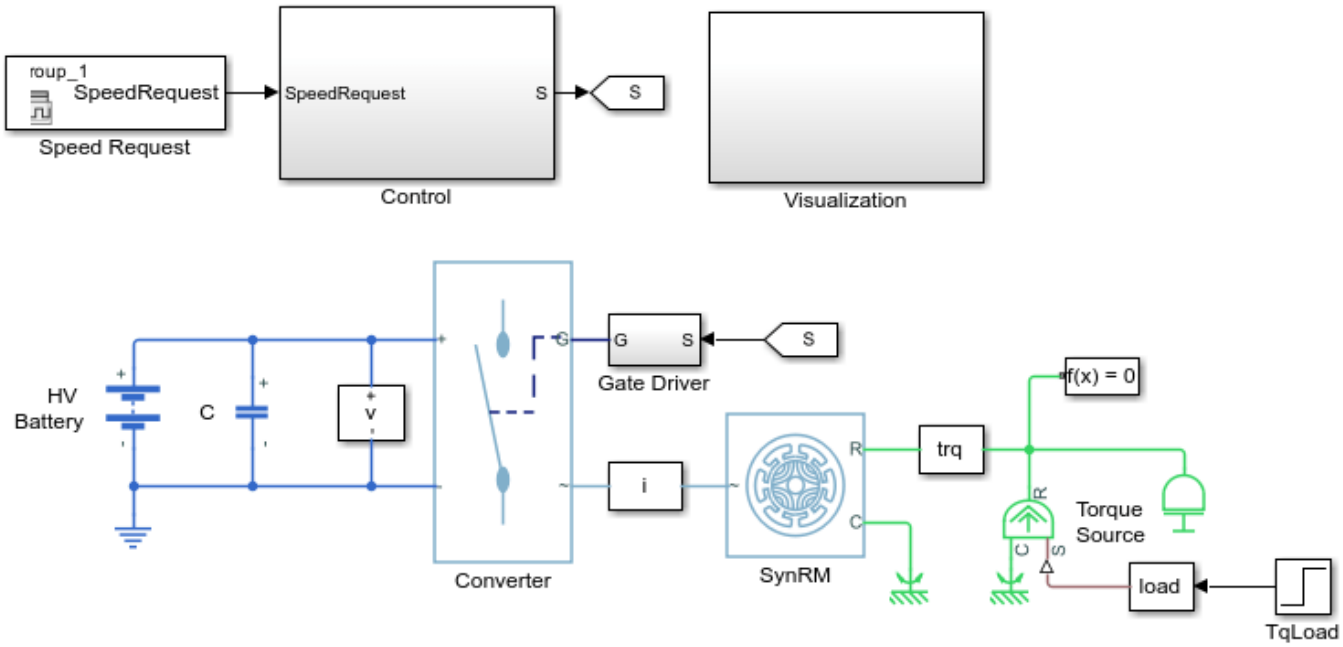
### Model



# Synchronous Reluctance Machine Velocity Control

This example shows how to control the rotor angular velocity in a synchronous reluctance machine (SynRM) based electrical drive. A high-voltage battery feeds the SynRM through a controlled three-phase converter. An ideal torque source provides the load. The Control subsystem includes a multi-rate PI-based cascade control structure. The control structure has an outer angular-velocity-control loop and two inner current-control loops. The Visualization subsystem contains scopes that allow you to see the simulation results.

**Model**




**Synchronous Reluctance Machine Velocity Control**

1. Modify model parameters
2. Explore simulation results using Simscape Results Explorer
3. Learn more about this example.

Copyright 2016-2022 The MathWorks, Inc.

# Tap-Changing Transformer for Automatic Voltage Regulation

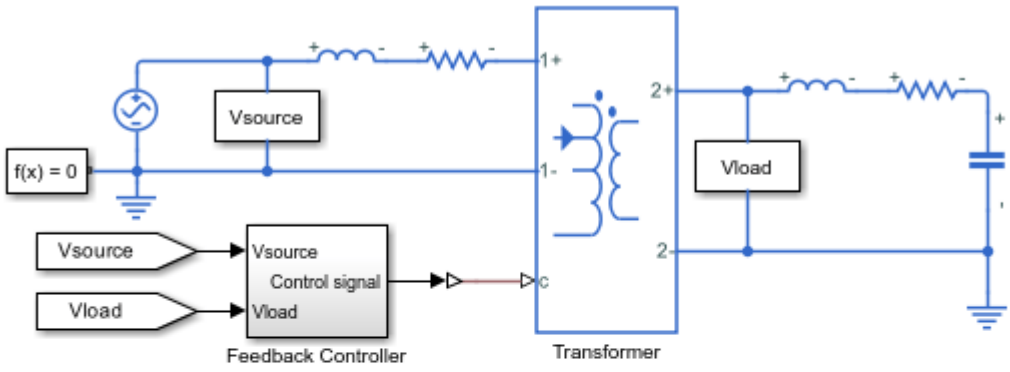
This example shows how to use a single-phase tap-changing transformer to control the voltage across an RLC load. The system contains an AC voltage source that generates a 60 Hz sine wave (located on the left-hand side of the circuit). The root-mean-square value for the voltage generated by this source is 120 V (reference voltage).

The Feedback Controller uses the peak voltages across the AC voltage source and RLC load to generate the control signal provided to the tap-changing transformer.

The tap-changer location is on the transformer's secondary winding. The values defined for the maximum and minimum voltages across the RLC load are 1.01 and 0.99 of the reference voltage, respectively. The tap-changing transformer has 33 positions for the tap setting, between -16 and +16, including the 0 position that corresponds to the nominal tap position. The nominal turns ratio (primary/secondary) is 0.9.

The step voltage per tap change is 1.5% of the transformer's primary voltage. The sample time is 1e-5 seconds.

### Model



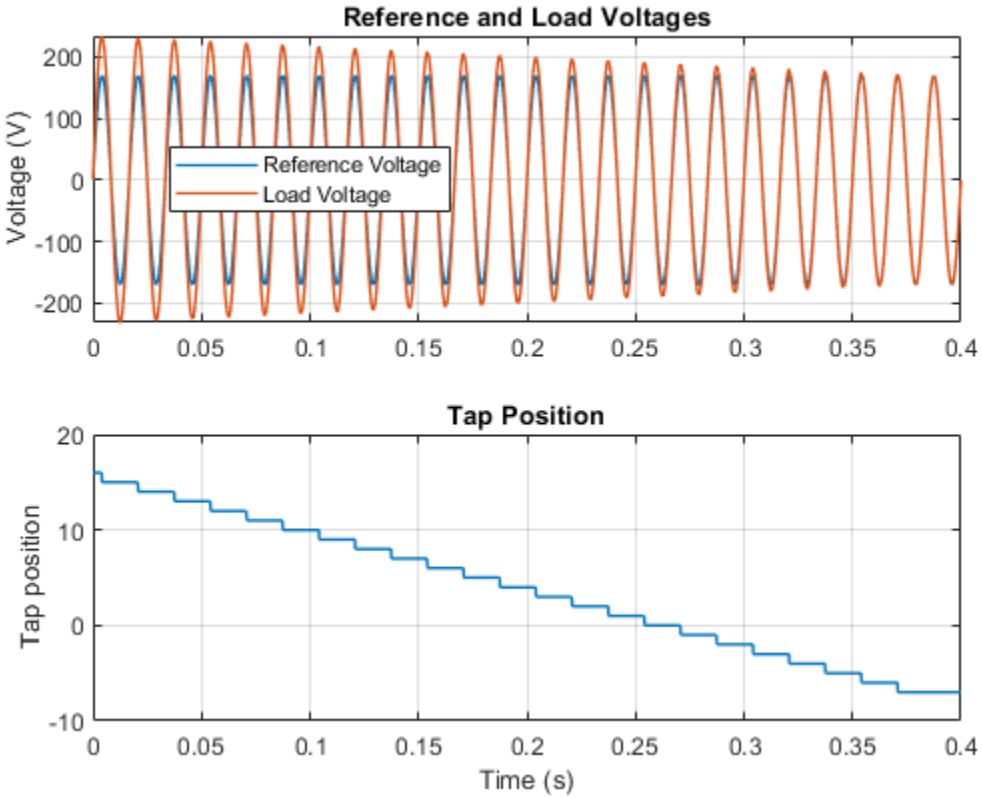

**Tap-Changing Transformer for Automatic Voltage Regulation**

1. Plot voltages and tap position (see code)
2. Set initial tap position: max_tap, min_tap, nom_tap
3. Explore simulation results using Simscape Results Explorer
4. Learn more about this example

Copyright 2017-2022 The MathWorks, Inc.

### Simulation Results from Simscape Logging

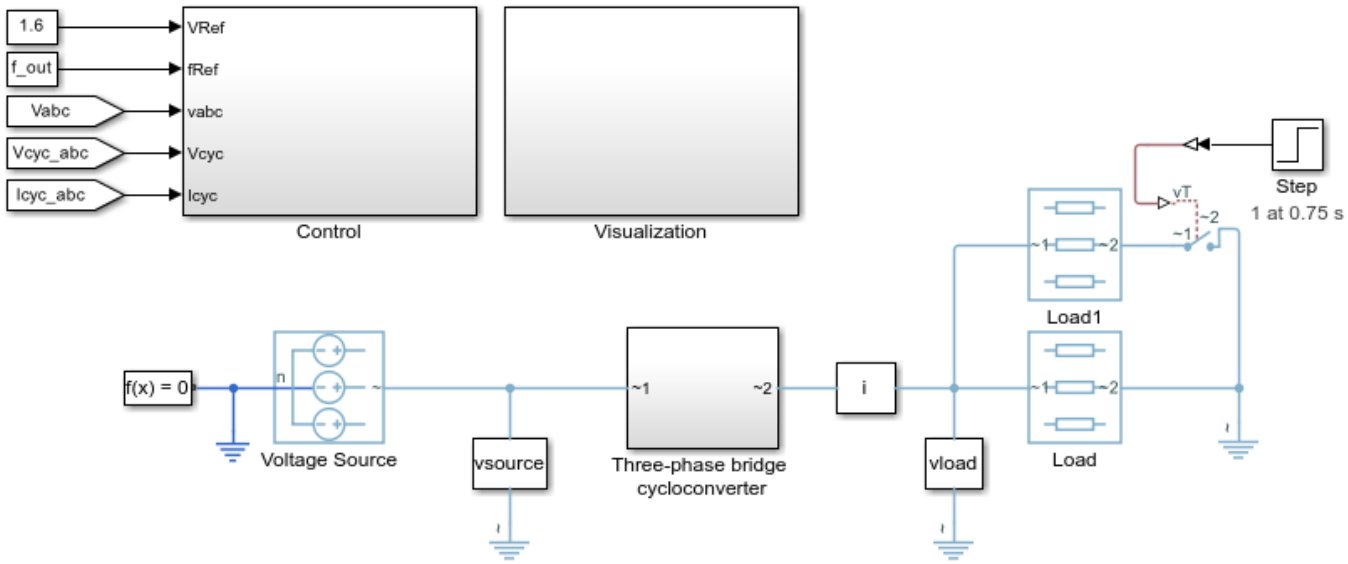
The plot below shows the simulated voltages across the AC voltage source and RLC Load and the evolution of the tap position.


Reference and Load Voltages
Voltage (V)
200
100
0
-100
-200
Reference Voltage
Load Voltage
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
Tap Position
Tap position
20
10
0
-10
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
Time (s)

# Three-Phase Bridge Cycloconverter

This example shows a three-phase bridge cycloconverter. The cycloconverter consists of 36 thyristors and has the capacity to lower the frequency of the input voltage. The Control subsystem implements the cycloconverter RMS voltage control. It also provides pulse generation for the firing of the thyristors. The Visualization subsystem contains scopes that allow you to see the simulation results. The simulation time, t, is 1 second. The load increases when Load1 switches on at t = 0.75 seconds.

**Model**




**Three-Phase Bridge Cycloconverter**

1. Modify model parameters
2. Explore simulation results using Simscape Results Explorer
3. Learn more about this example

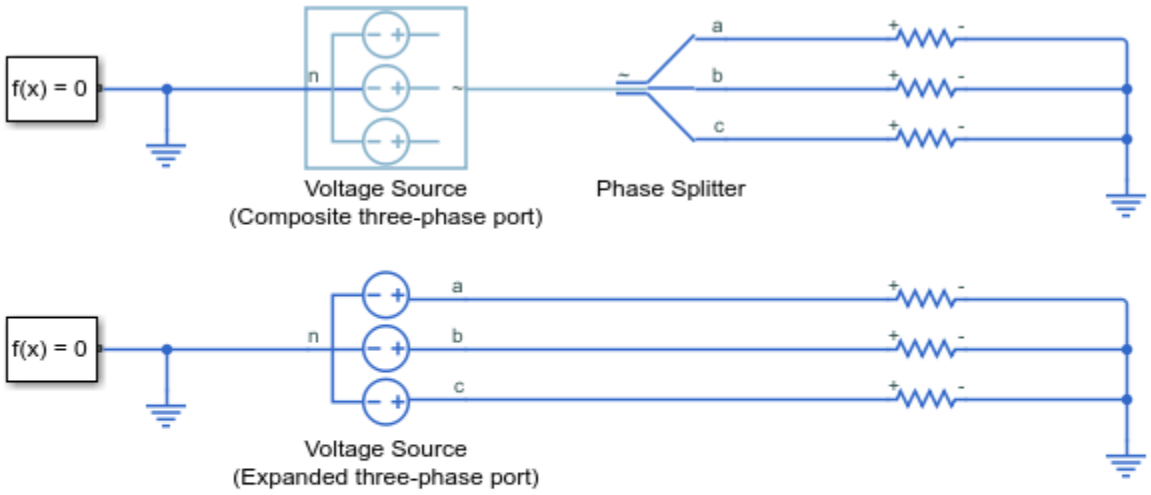
Copyright 2016-2022 The MathWorks, Inc.

# Comparison of Three-Phase Port Types

This example shows a comparison of Composite three-phase ports versus Expanded three-phase ports. The first circuit shows a Voltage Source configured with a Composite three-phase port. A Phase Splitter block provides the interface to the Simscape™ foundation library electrical elements. The second circuit shows a Voltage Source configured with an Expanded three-phase port which can connect directly to the Simscape foundation library electrical elements. The Voltage Source can be changed from Composite to Expanded three-phase ports by use of the Modeling option mask parameter.

Both circuits are equivalent. Both consider instantaneous three-phase voltages and currents. Both give the same results.

**Model**




**Comparison of Three-Phase Port Types**

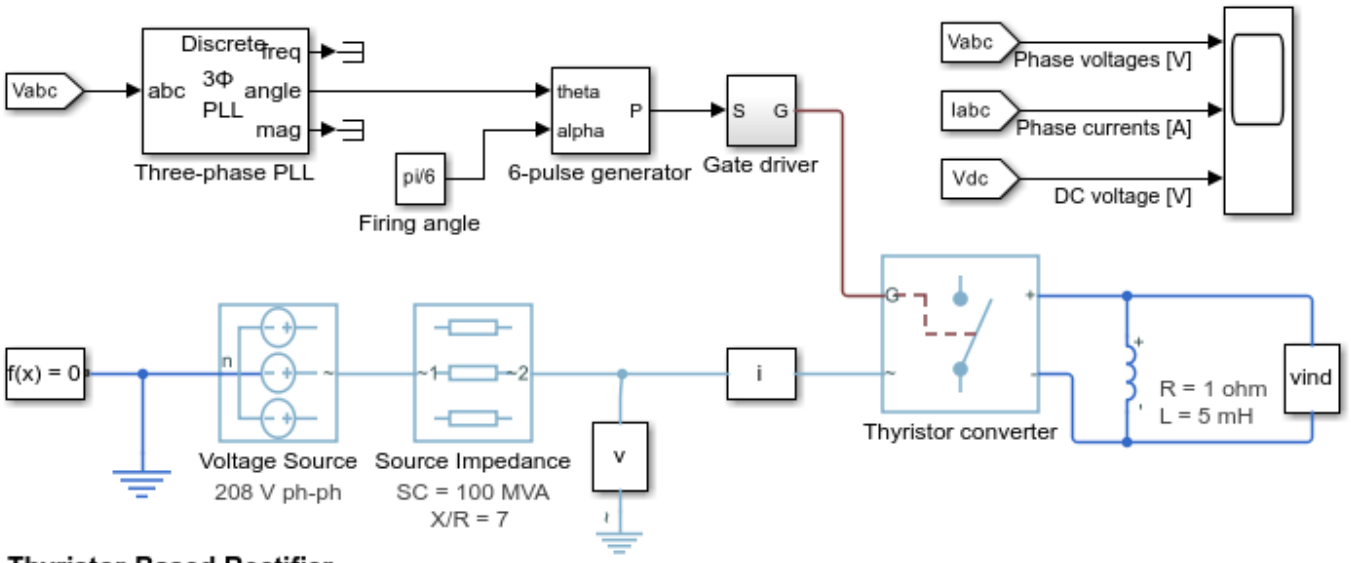
1. Explore simulation results using Simscape Results Explorer
2. Learn more about this example

Copyright 2013-2022 The MathWorks, Inc.

# Thyristor-Based Rectifier

This example shows a thyristor-based rectifier.

**Model**




**Thyristor-Based Rectifier**

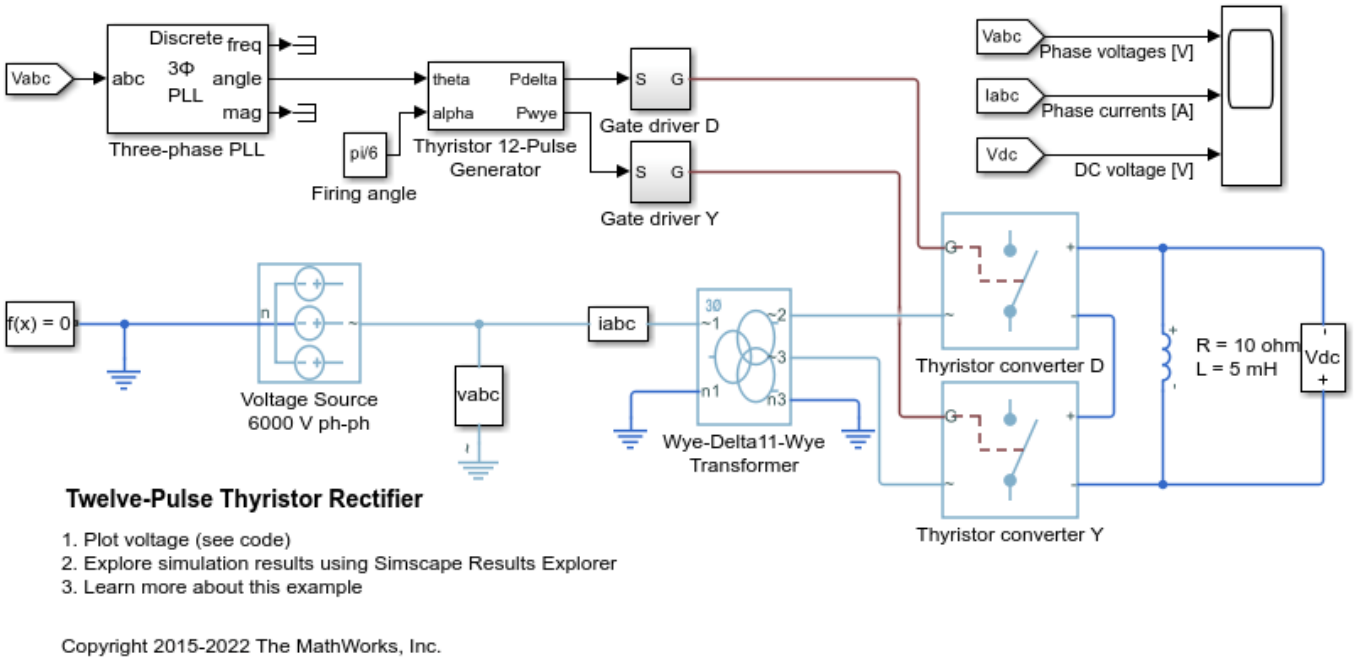
1. Explore simulation results using Simscape Results Explorer
2. Learn more about this example

Copyright 2015-2022 The MathWorks, Inc.

# Twelve-Pulse Thyristor Rectifier

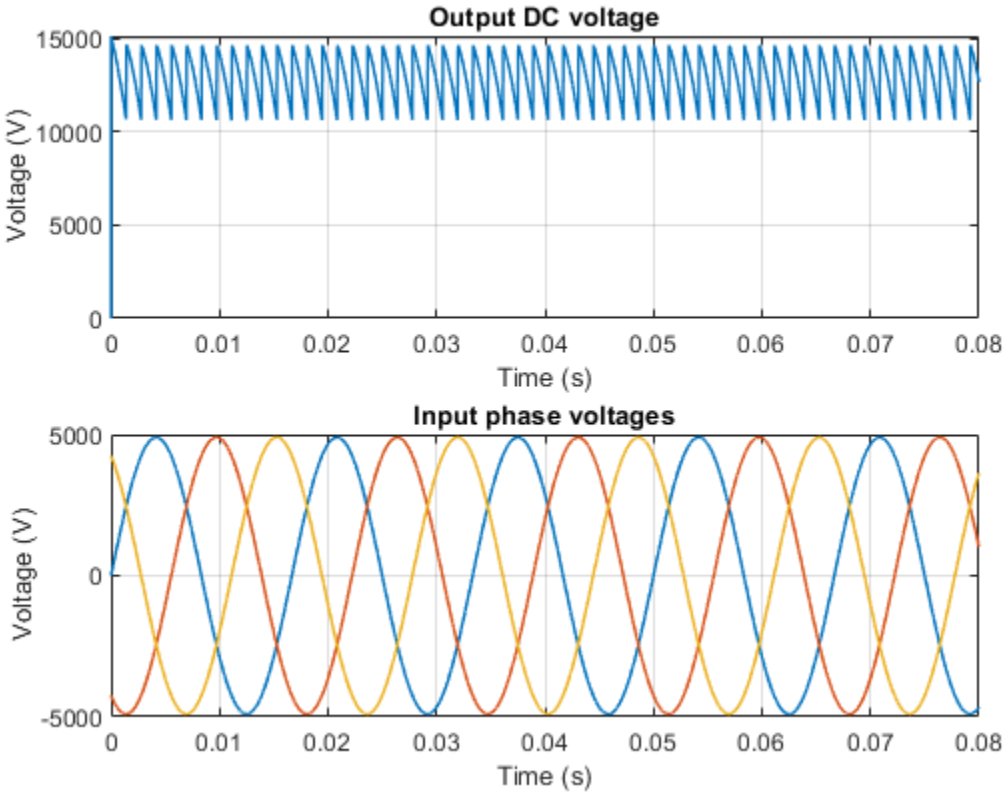
This example shows how to control a twelve-pulse thyristor rectifier. Two thyristor converters are connected to a Wye-Delta-Wye transformer on the input. A Thyristor 12-Pulse Generator block generates the gate signals for the two converters.

### Model




### Simulation Results from Simscape Logging

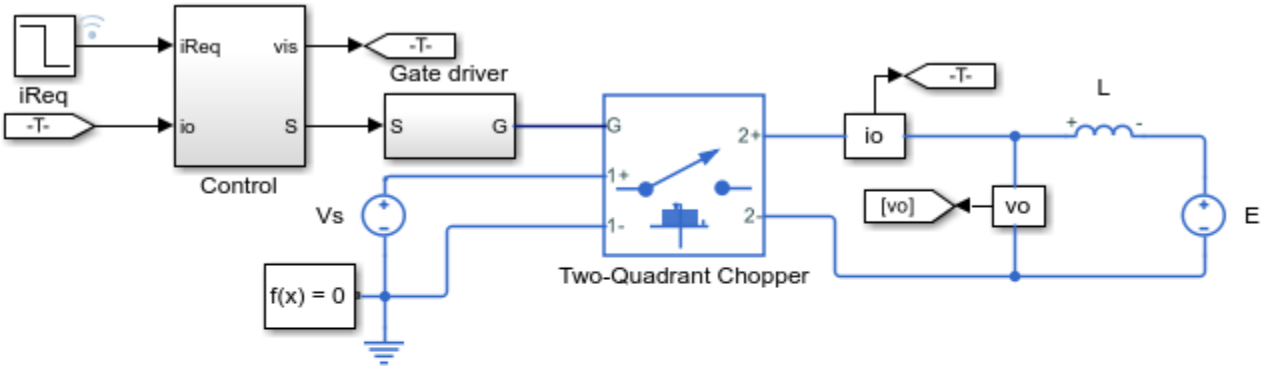
The plot below shows the output DC voltage and the input phase voltages.


Output DC voltage
15000
10000
5000
0
Voltage (V)
0
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08
Time (s)
Input phase voltages
5000
0
-5000
Voltage (V)
0
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08
Time (s)

# First and Second Quadrant Chopper Control

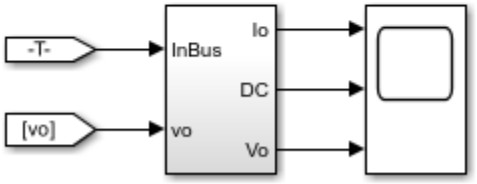
This example shows how to control a two-quadrant chopper. The two-quadrant chopper operates in the first and second quadrants, allowing positive and negative output current. The Control subsystem implements a simple PI-based control algorithm for controlling the output current. The load of the system is considered constant throughout the simulation.

### Model




**First and Second Quadrant Chopper Control**

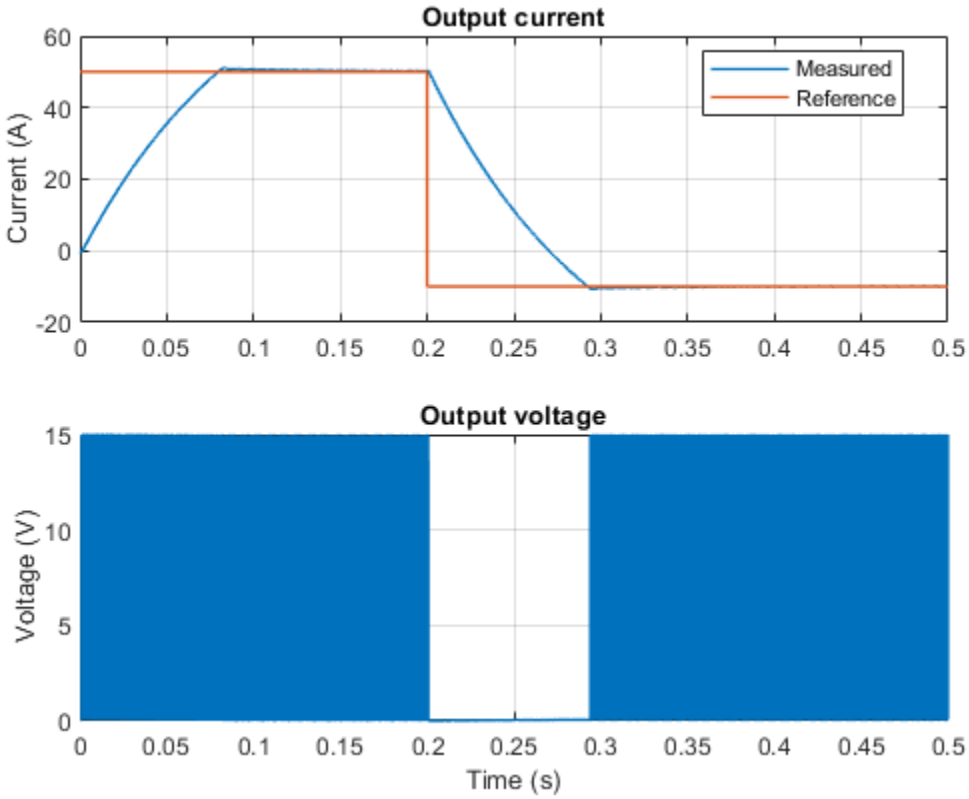
1. Plot current (see code)
2. Explore simulation results using Simscape Results Explorer
3. Modify model parameters
4. Learn more about this example




Copyright 2016-2022 The MathWorks, Inc.

### Simulation Results from Simscape Logging

The plot below shows the requested and measured current for the test and the output voltage in the circuit.



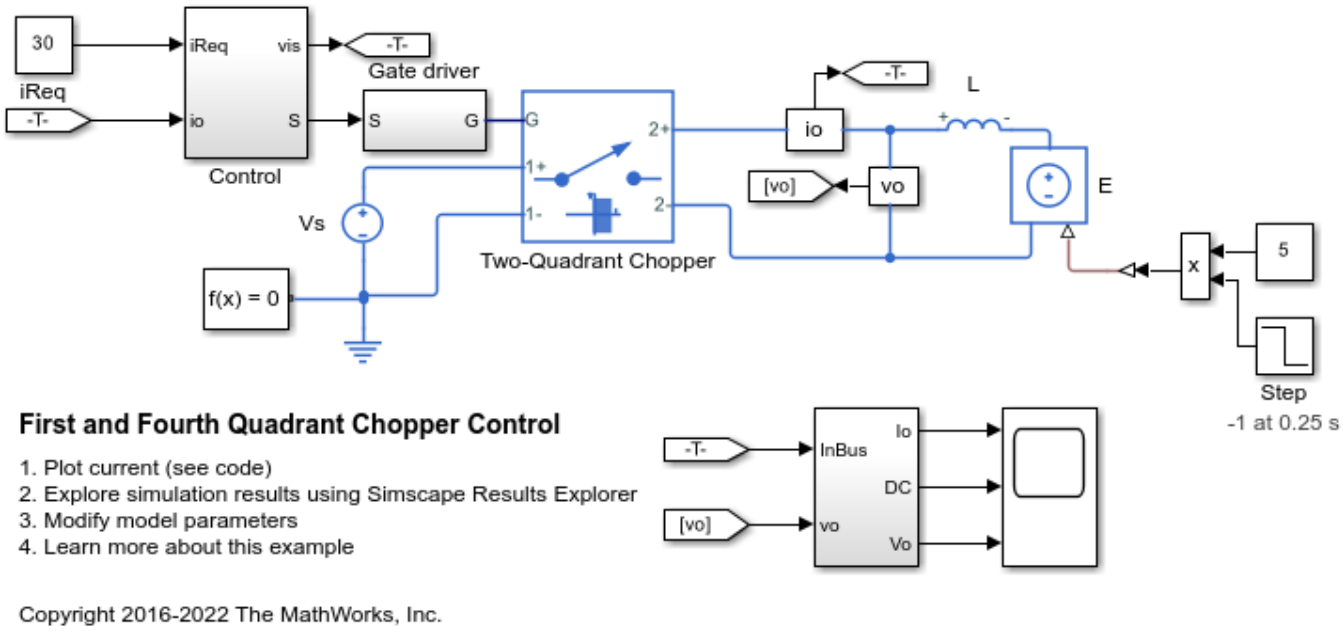

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 50 microseconds.

# First and Fourth Quadrant Chopper Control

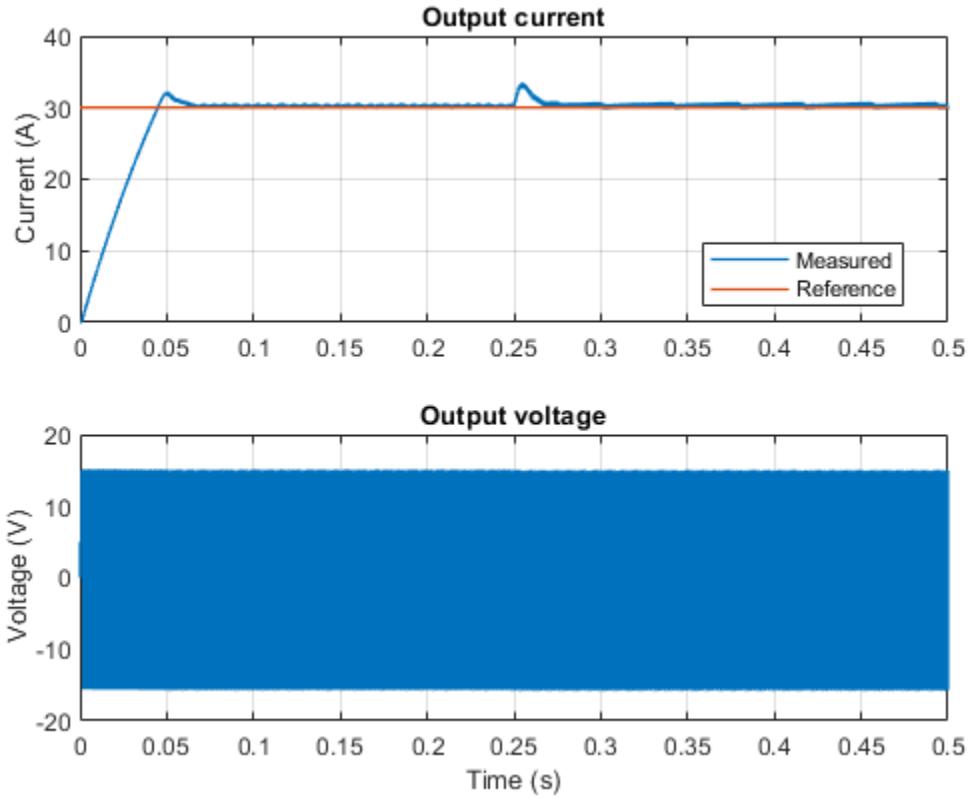
This example shows how to control a two-quadrant chopper. The two-quadrant chopper operates in the first and fourth quadrants, allowing positive and negative output voltage. The Control subsystem implements a simple PI-based control algorithm for controlling the output current. The total simulation time (t) is 0.5 seconds. At t = 0.25 seconds, the polarity of the load DC source E is changed.

### Model




### Simulation Results from Simscape Logging

The plot below shows the requested and measured current for the test and the output voltage in the circuit.



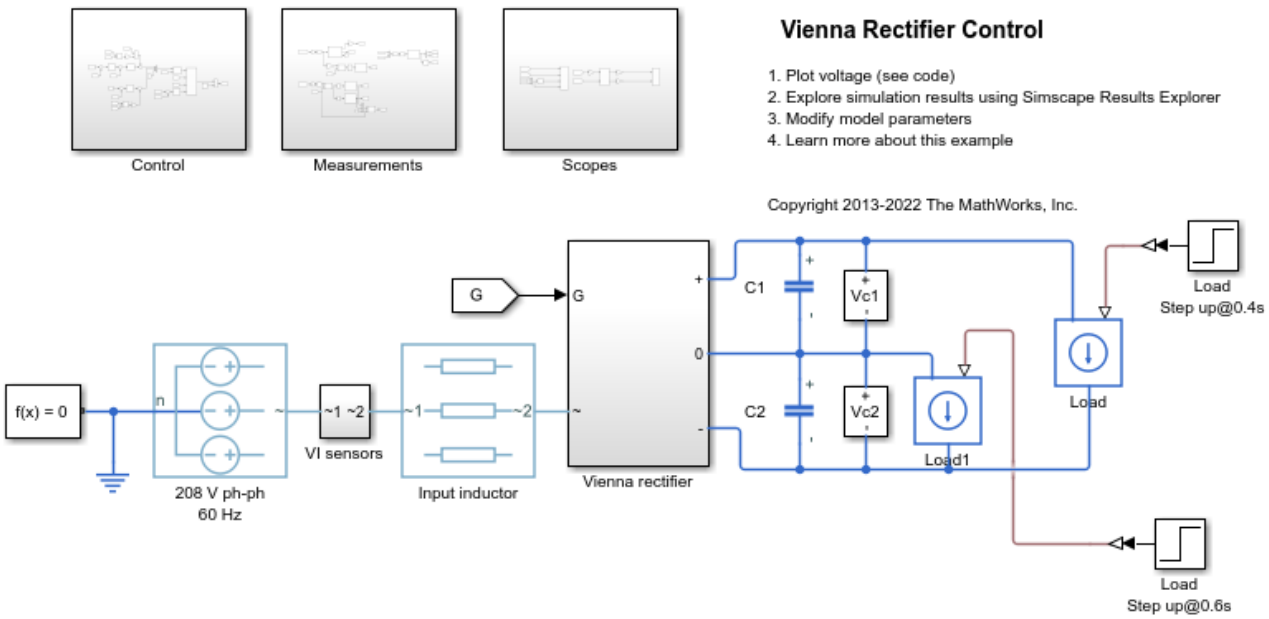

### Results from Real-Time Simulation

This example has been tested on a Speedgoat Performance real–time target machine with an Intel® 3.5 GHz i7 multi–core CPU. This model can run in real time with a step size of 50 microseconds.

# Vienna Rectifier Control

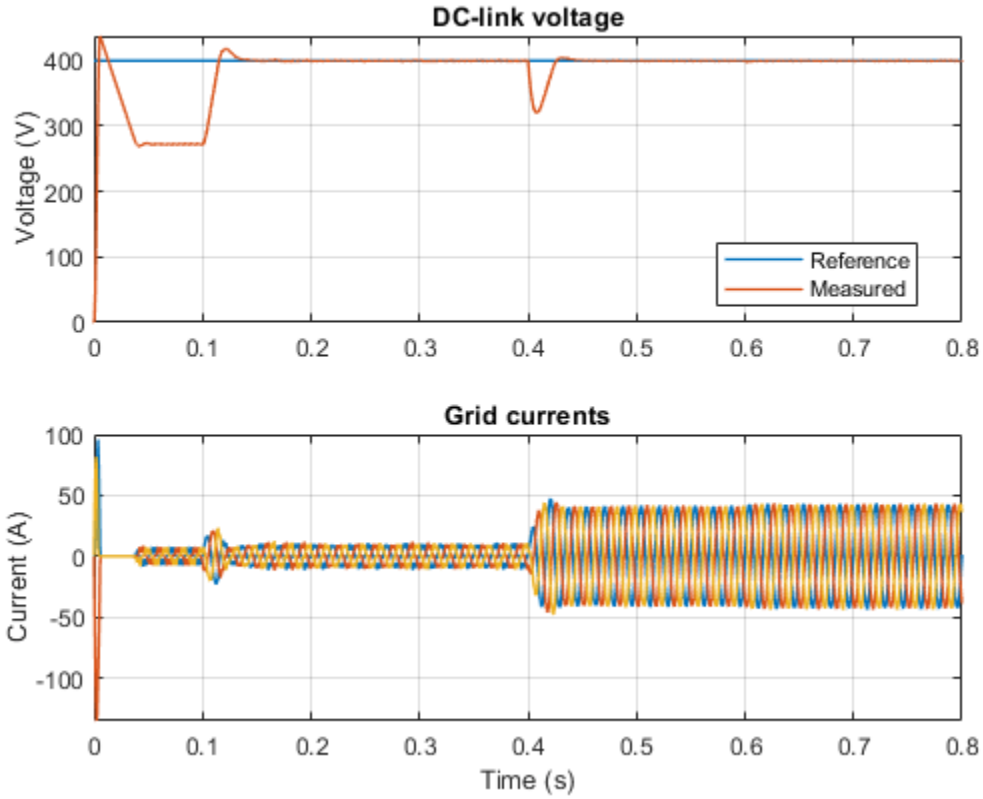
This example shows how to control a Vienna rectifier. The Vienna rectifier subsystem consists of three-phase legs. Each leg has one power switch and six power diodes. The Control subsystem implements a closed-loop control strategy for the Vienna rectifier using space-vector modulation. Total simulation time is 0.1 s. At time 0.1 s, the Vienna Rectifier is engaged. At times 0.4 s and 0.6 s, the load steps up on the DC side.

### Model




### Simulation Results from Simscape Logging

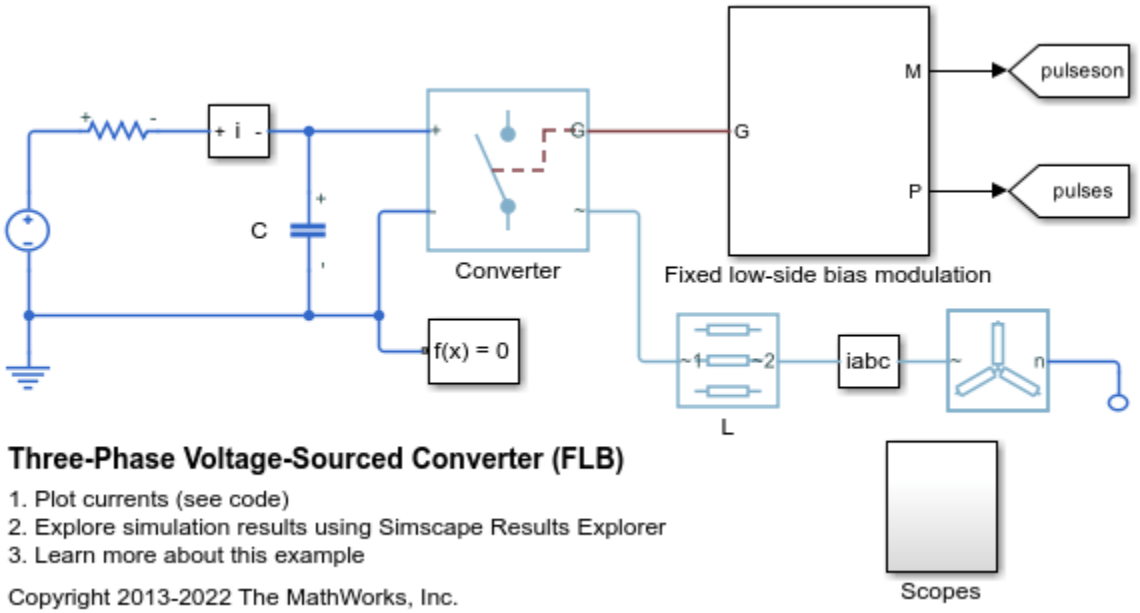
The plot below shows the measured dc-link voltage for the test and the grid currents.


DC-link voltage
Voltage (V)
400
300
200
100
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
Reference
Measured
Grid currents
Current (A)
100
50
0
-50
-100
Time (s)

# Three-Phase Voltage-Sourced Converter (FLB)

This example shows how to model a three-phase voltage-sourced converter that uses Fixed Low-side Bias (FLB) modulation. This modulation scheme minimizes the switching in the converter as at any given time one phase is not being pulse modulated. The trade-off is the need for narrower pulses for a given level of acceptable harmonics. The model can be used to support selection of suitable values for L, C and the pulse modulation scheme parameters.

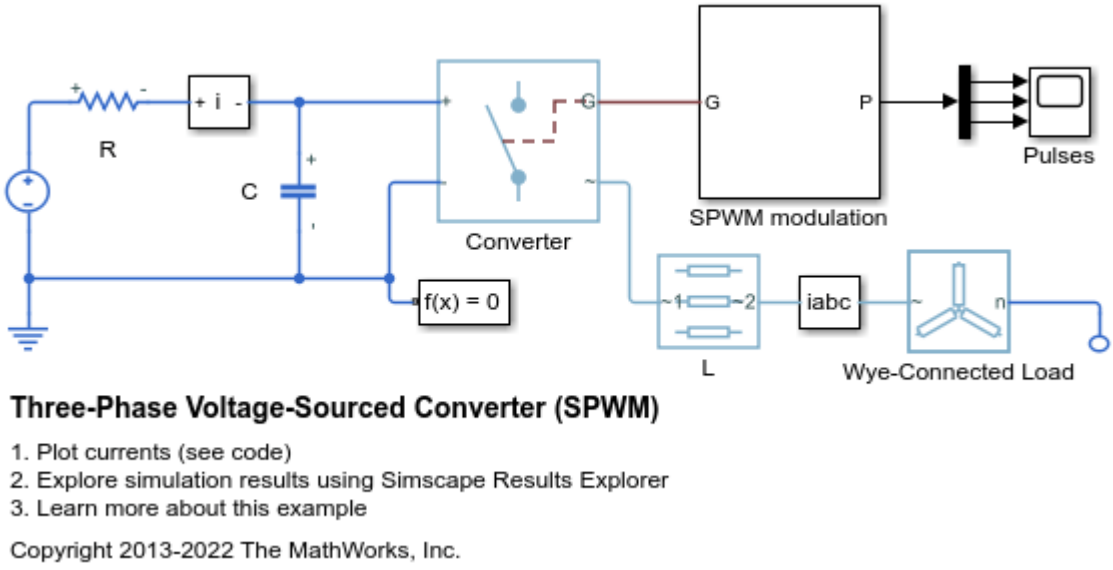
**Model**



# Three-Phase Voltage-Sourced Converter (SPWM)

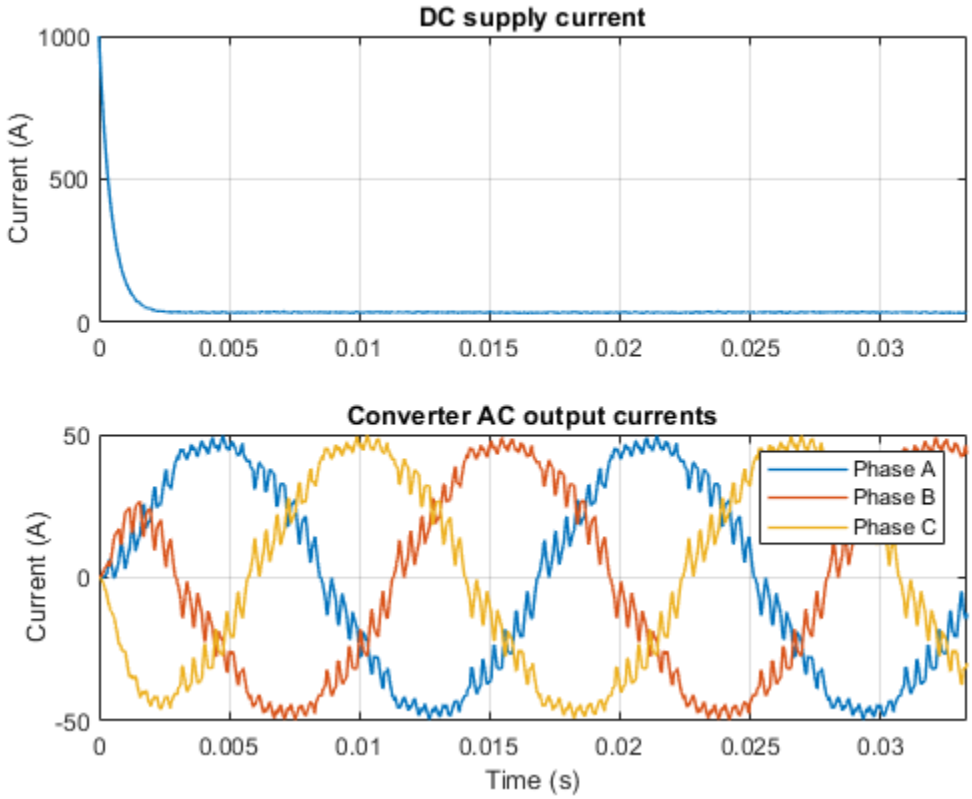
This example shows how to model a three-phase voltage-sourced converter that uses Sinusoidal Pulse-Width Modulation (SPWM). This modulation scheme compares a reference sine wave with a higher-frequency repeating triangle wave in order to generate the pulses. The model can be used to support selection of suitable values for L, C and the pulse modulation scheme parameters.

### Model



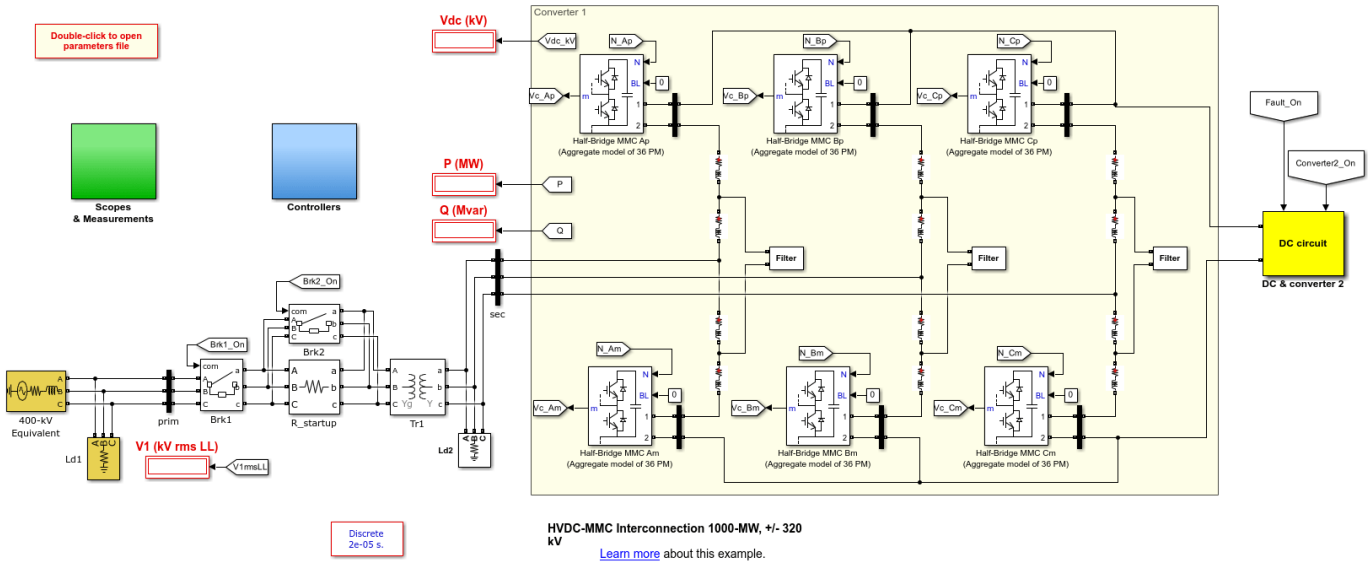

### Simulation Results from Simscape Logging

The plot below shows the DC supply current, as well as the AC phase currents in the Wye-Connected Load.


DC supply current
1000
500
0
Current (A)
0
0.005
0.01
0.015
0.02
0.025
0.03
Converter AC output currents
50
0
-50
Current (A)
Phase A
Phase B
Phase C
0
0.005
0.01
0.015
0.02
0.025
0.03
Time (s)

# HVDC-MMC Interconnection (1000-MW, +/- 320 kV)

This example shows a model of a High Voltage Direct Current (HVDC) interconnection using Voltage-Sourced Converters (VSC) based on the Modular Multi- level Converter (MMC) technology. The simulation is optimized by the use of an aggregate MMC model.




## Description

DC links are becoming the preferred means for power exchange between countries and transmission of power from renewables (hydroelectric plants, off-shore wind power and solar farms) to power grids. Several of these systems are presently in operation, for example the INELFE project (France-Spain interconnection), the Dolwin1 project (offshore wind HVDC link), or planned for future projects such as the Northern Pass (Canada-USA 1090-MW DC link).

The MMC converter is implemented using an aggregate model to simulate the power modules of one arm. With this aggregate model, control system dynamics, converter harmonics and circulating currents phenomena are all well-represented. However, having only one virtual capacitor to represent all capacitors of one arm, the model assumes that capacitor voltages of all power modules are well-balanced, and therefore capacitor voltage-balancing scheme cannot be simulated. The aggregate model runs much faster than a detailed model that would use two switching devices and one capacitor for each individual power module of one arm. This aggregate model is also well-suited for real-time simulation.

## Main Components

1. The electrical grid is modeled using a 400-kV, 50-Hz equivalent and two breakers to energize Converter 1.
2. Converter 1 is implemented using 6 half-bridge MMC blocks, each representing 36 power modules. This custom SPS block uses a switching-function model where only one equivalent module is used to represent all power modules. The control signal is a two-element vector [ Nin, Nbl ], where Nin represents the number of inserted modules and Nbl represents the number of modules in the blocked-state. The output Vc (capacitor voltage) has only one element and gives the average value of the capacitor modules.

3. Inside the DC Circuit subsystem, you will find a simplified model of the cable, as well as the second converter modeled using DC sources and ideal switches. You will also find a switch to apply a fault on the cable.
4. The Controllers subsystem contains the various control systems required to operate the interconnection. It includes the Active & Reactive Power Regulators, the DC Voltage Regulator, a dq current regulator with feed-forward, PLL & measurement subsystem, PWM Generators to control Converter1 half-bridge MMC's. You will also find the Sequencer Area where the various modes of operation of Converter1 are programmed.
5. Scopes & Measurements contains all the scopes used to observe several signals during the simulation. Power and fundamental voltage calculations are also performed in the subsystem

### Simulation

Simulating our SPS model for 10 seconds allows observation of the operation of the interconnection during start-up (capacitor charging), voltage regulation, and power regulation. All parameters required to run the model can be found in the following file: HVDC_MMC_param.m. This file is automatically executed in the MATLAB workspace when the example is open. Run the model and observe the following events:

1. At 0.1s, Breaker 1 is closed and Converter 1 is energized through a resistor to reduce the charging current. Capacitors are being charged and at 1s, the start-up resistor is short- circuited by closing Breaker 2.
2. At 1.5s, Converter 1 is deblocked and the Voltage Regulator is enabled.
3. At 2s, the Voltage Regulator setpoint is ramped to the nominal DC operating voltage of the interconnection: 640 kV (+/-320 kV).
4. At 4s, PQ regulators are enabled and Converter 2 switches are closed.
5. At 4.2s, the Active Power Regulator setpoint is ramped to 1pu (1000 MW).
6. At 7.5s, the Reactive Power Regulator setpoint is ramped to 0.25pu (250 Mvar).

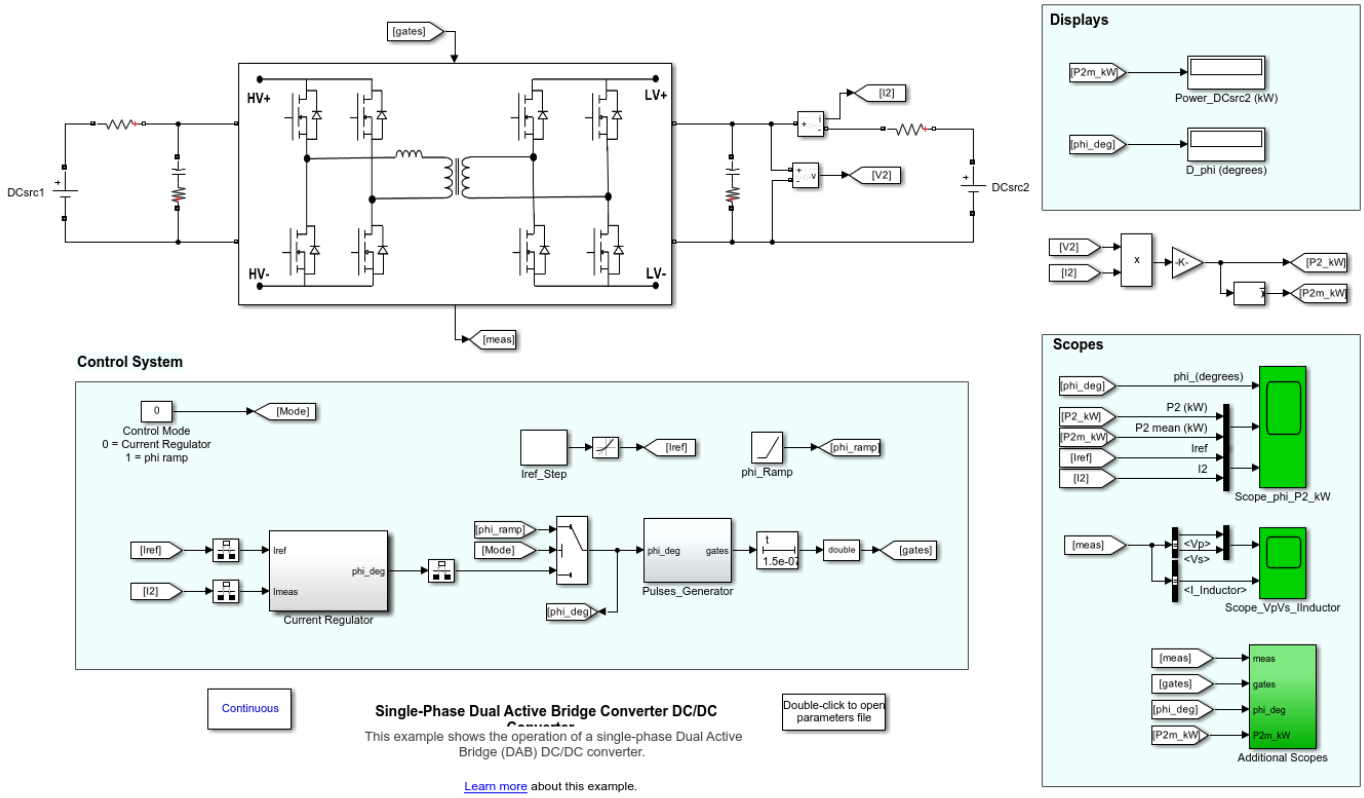
If you set the value of parameter Tfault to 7 (actual default value=9999) in the workspace and re-start the simulation, a DC fault will be applied at the middle of the cable at 7s. The half-bridge MMC will be blocked and the interconnection will be shut down after two cycles (Brk1 will be opened).

### References

1. VSC-HVDC Transmission with Cascaded Two-Level Converters, Bjorn Jacobson, Patrik Karlsson, Gunnar Asplund, Lennart Harnefors, Tomas Jonsson ABB,Sweden, CIGRE 2010 B4-110
2. Setup and Performance of the Real-Time Simulator used for Hardware-in-Loop-Tests of a VSC-Based HVDC scheme for Offshore Applications., O. Venjakob, S. Kubera, R. Hibberts-Caswell, P.A. Forsyth, T.L. Maguire Siemens, Germany & RTDS Technologies, Canada. Paper submitted to the International Conference on Power Systems Transients (IPST2013) in Vancouver, Canada July 18-20, 2013

# Single-Phase Dual Active Bridge Converter DC/DC Converter

This example shows the operation of a single-phase Dual Active Bridge (DAB) DC/DC converter.




### Description

The single-phase Dual Active Bridge (DAB) converter is a highly efficient isolated bidirectional DC-to-DC converter. Its key features are:

- Isolated operation
- Power density (high frequency operation reduces the size of the isolation transformer)
- Bidirectional power flow
- Efficiency (Zero Voltage Switching)

It is used in several applications such as in charging stations and electric vehicle traction. Vehicle-to-Grid (V2G) systems are also using this type of converters. The 10-kW DAB model consists of two MOSFET full-bridges linked by a coupling inductor and a high frequency transformer. The high voltage bridge is connected to an 800V DC source and the low voltage bridge is connected to a 450V DC source. Both bridges are controlled by a pulse generator and a current regulator.

### DAB Operation

Power flow in the DAB is achieved by phase-shifting the pulses of one bridge with respect to the other, while keeping the duty cycle of both bridges at 50%. Note that identical pulses are sent to the

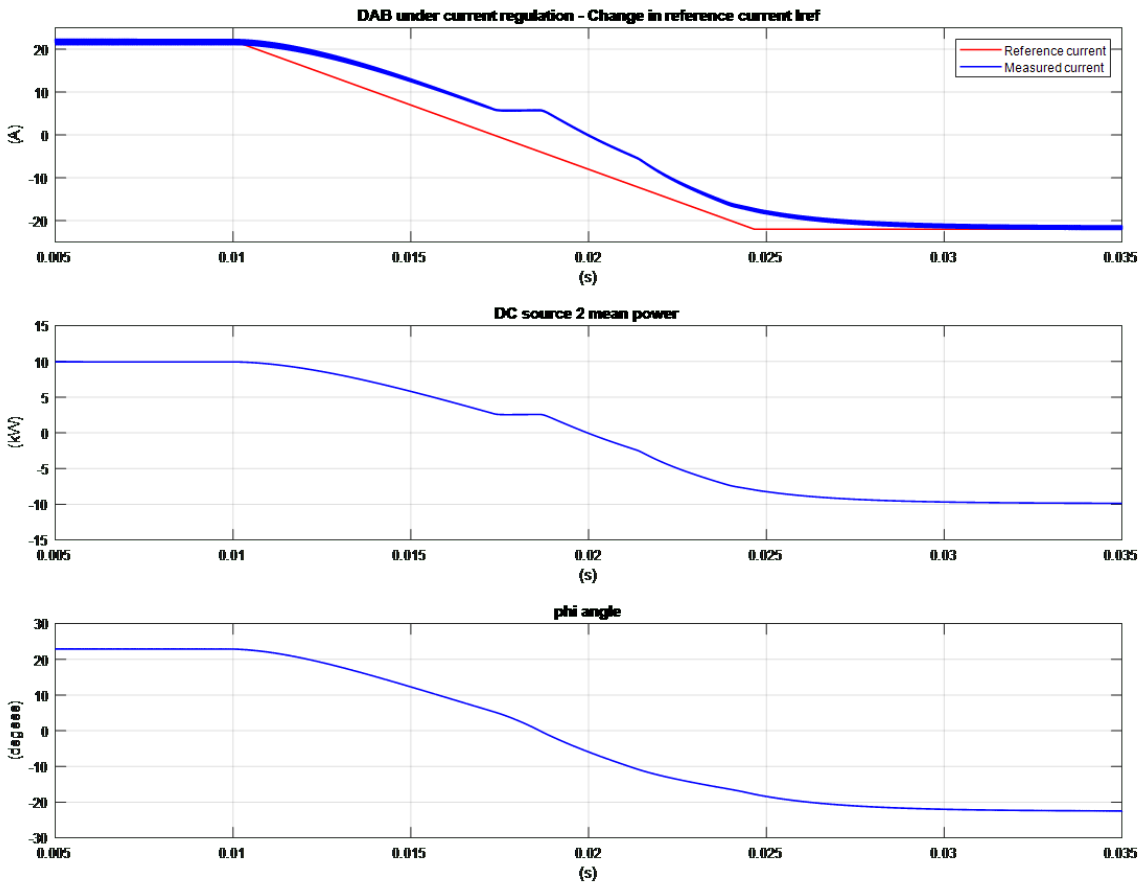
diagonal switches. In our example, the switching frequency is set to 100 kHz and the dead time to 150 ns. The theoretical output power transfer between the two bridges is given by:

$$P = \frac{V_{HV}\, N\, V_{LV}\, \phi\, (\pi - |\phi|)}{2\, \pi^2\, F_{SW}\, L}$$

where:

- $V_{HV}$ is the DC link voltage of the high voltage bridge, in Volts
- $N$ is the primary to secondary transformer voltage ratio
- $V_{LV}$ is the DC link voltage of the low voltage bridge, in Volts
- $\phi$ is the phase shift between the two square waves of the bridges, in Radians
- $F_{SW}$ is the PWM switching frequency, in Hertz
- $L$ is the total inductance (coupling inductor + total transformer leakage reactance seen from the primary) between the two bridges, in Henry.

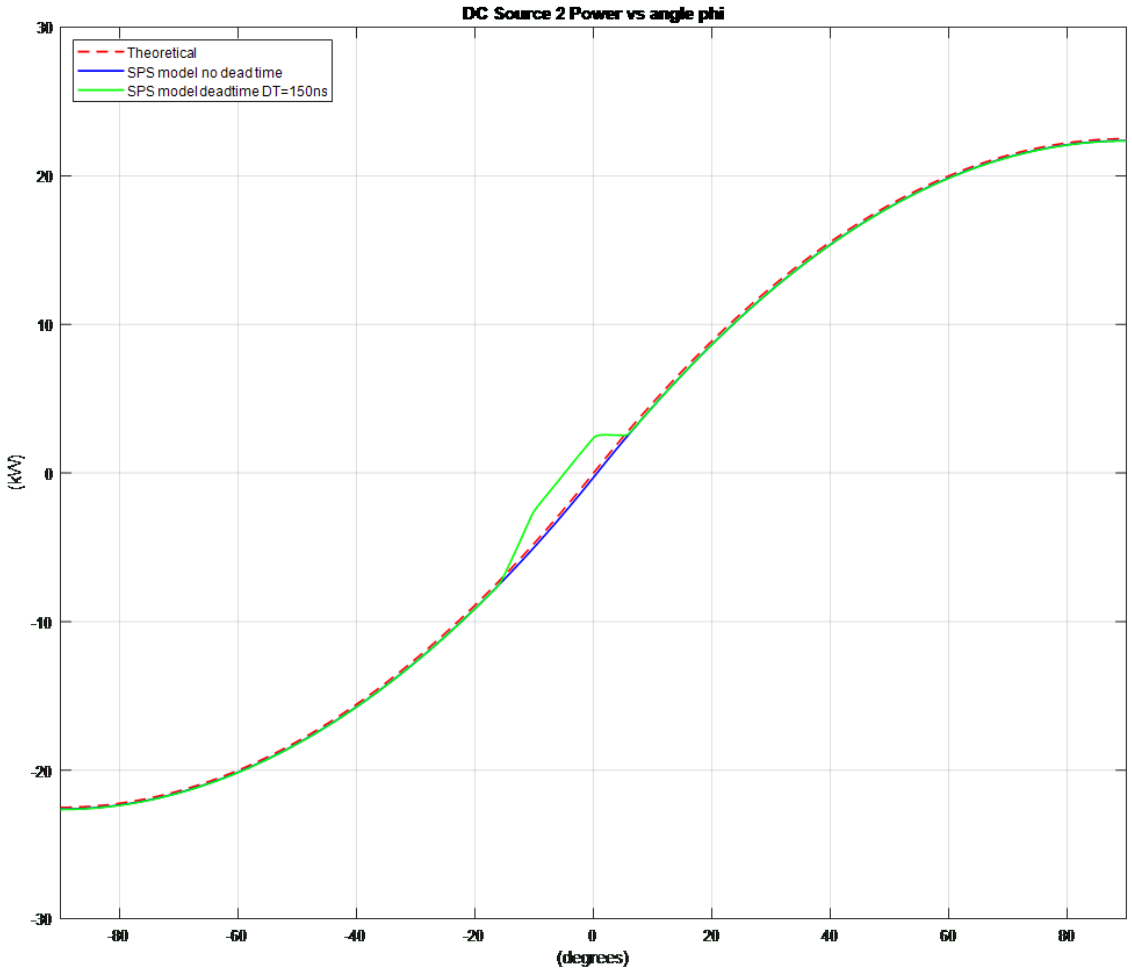
We can see that the power transfer between the two bridges can be controlled by changing the angle of the two square waves of the bridges. For a positive angle $\phi$, power will be transferred from the high-voltage bridge to the low-voltage bridge. Power flow will be reversed for a negative angle $\phi$. The figure below shows the bridges voltages and the resulting coupling inductor current during the four switching states while angle $\phi$ is set to +22 degrees.




## Simulation

The model allows running two simulation:

1. First, activate the current regulation mode by setting the value of the Control Mode block to zero. Run the simulation and observe the DAB current regulator dynamics on Scope phi_P2_kW. The reference current is changed from 22 A to -22 A at 0.01s (at a slew rate of 3000 A/s). In order to follow the new reference, the regulator output will vary from 22.9 degrees to -22.5 degrees.
2. Now set the value of the Control Mode block to 1 to allow direct control of the pulse generator input (phi_deg) to ramp the angle $\phi$ from -100 to 100 degrees. In the Additional Scopes subsystem, open the Scope_P2_Ptheo_kW and run the simulation. You will see the actual power delivered to DC Source 2 compared to the calculated theoretical value while the angle $\phi$ is ramped from -100 to 100 degrees. For an angle $\phi$ ranging from -15 to +5 degrees, the actual power transfer does not match the theoretical value. This is mainly due to the 150 ns dead time. Figure below shows the simulation results as well as a third curve (blue curve) showing the power transfer resulting from a another simulation performed without dead time. This blue curve and the theoretical value curve are almost identical.




## References

1 Texas Instruments TIDA-010054 Bi-directional, Dual Active Bridge Reference Design for Level 3 Electric Vehicle Charging Stations. June 2019. https://www.ti.com/tool/TIDA-010054

2 Rawad Zgheib, Innocent Kamwa, Kamal Al-Haddad, "Comparison between Isolated and Non-Isolated DC/DC Converters for Bidirectional EV Chargers ", IEEE 2017. https://ieeexplore.ieee.org/document/7913285

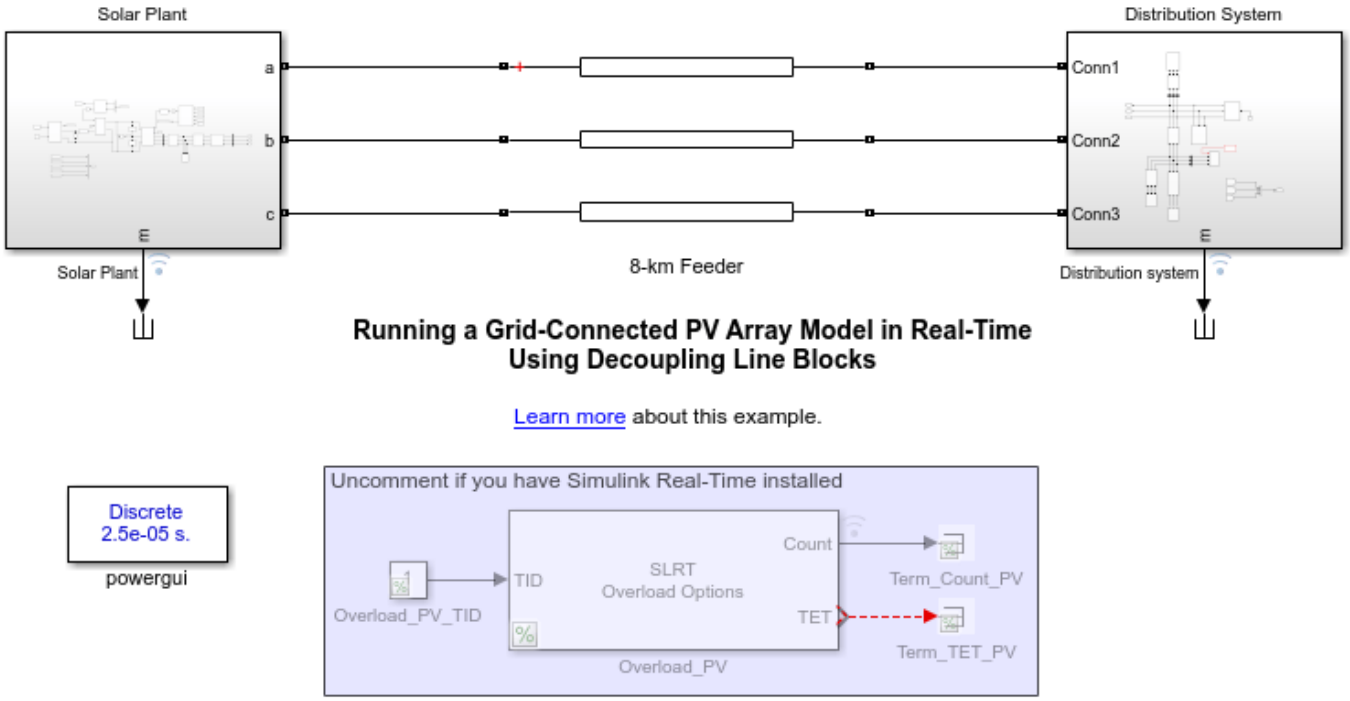
# Running a Grid-Connected PV Array Model in Real-Time Using Decoupling Line Blocks

This example shows how to simulate a grid-connected PV array model in real-time and how Decoupling Line blocks are used to avoid CPU overload by dividing the model into two concurrent tasks on a Speedgoat real-time target machine.

### 1. The Base Model

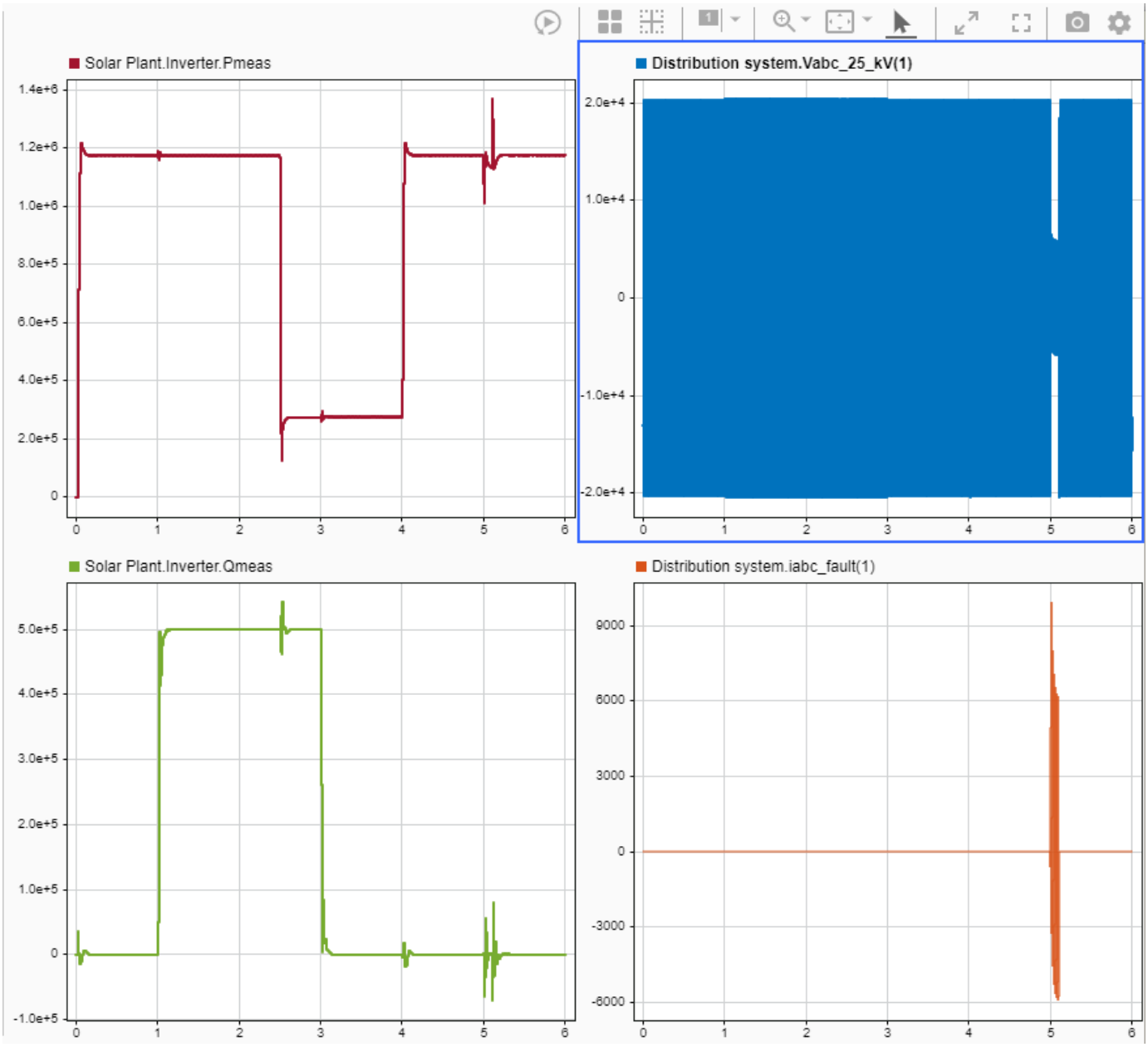
As a base model, we are using a simplified version of the “2-MW PV Farm Connected to a 25-kV Distribution System” on page 8-996 example. This base example is available under the power_PVarray_Grid.slx file:

```
model='power_PVarray_Grid';
open_system(model);
```




We simulate the model and observe simulation signals using the Simulink Data Inspector.

Three events are programmed in the simulation: an irradiance variation, a reactive power reference step, and a two-phase fault on the distribution system:




### 2. Run the Base Model in Real-Time

The code below is a summary of the settings made to the base model to run in real-time.

First we enabled data logging for the number of overload count and the Task Execution Time (TET):

```
set_param([model,'/Overload_PV'],'TETFlag','on');
set_param([model,'/Overload_PV'],'StartupDur','1000');
set_param([model,'/Overload_PV'],'maxOverload','100000');

ph = get_param([model,'/Overload_PV'],'PortHandles');
set_param(ph.Outport(1),'DataLogging','on');
set_param(ph.Outport(1),'DataLoggingNameMode','Custom');
set_param(ph.Outport(1),'DataLoggingName','Overload_Count_PV');
```

```
ph = get_param([model,'/Overload_PV'],'PortHandles');
set_param(ph.Outport(2),'DataLogging','on');
set_param(ph.Outport(2),'DataLoggingNameMode','Custom');
set_param(ph.Outport(2),'DataLoggingName','Overload_TET_PV');
```

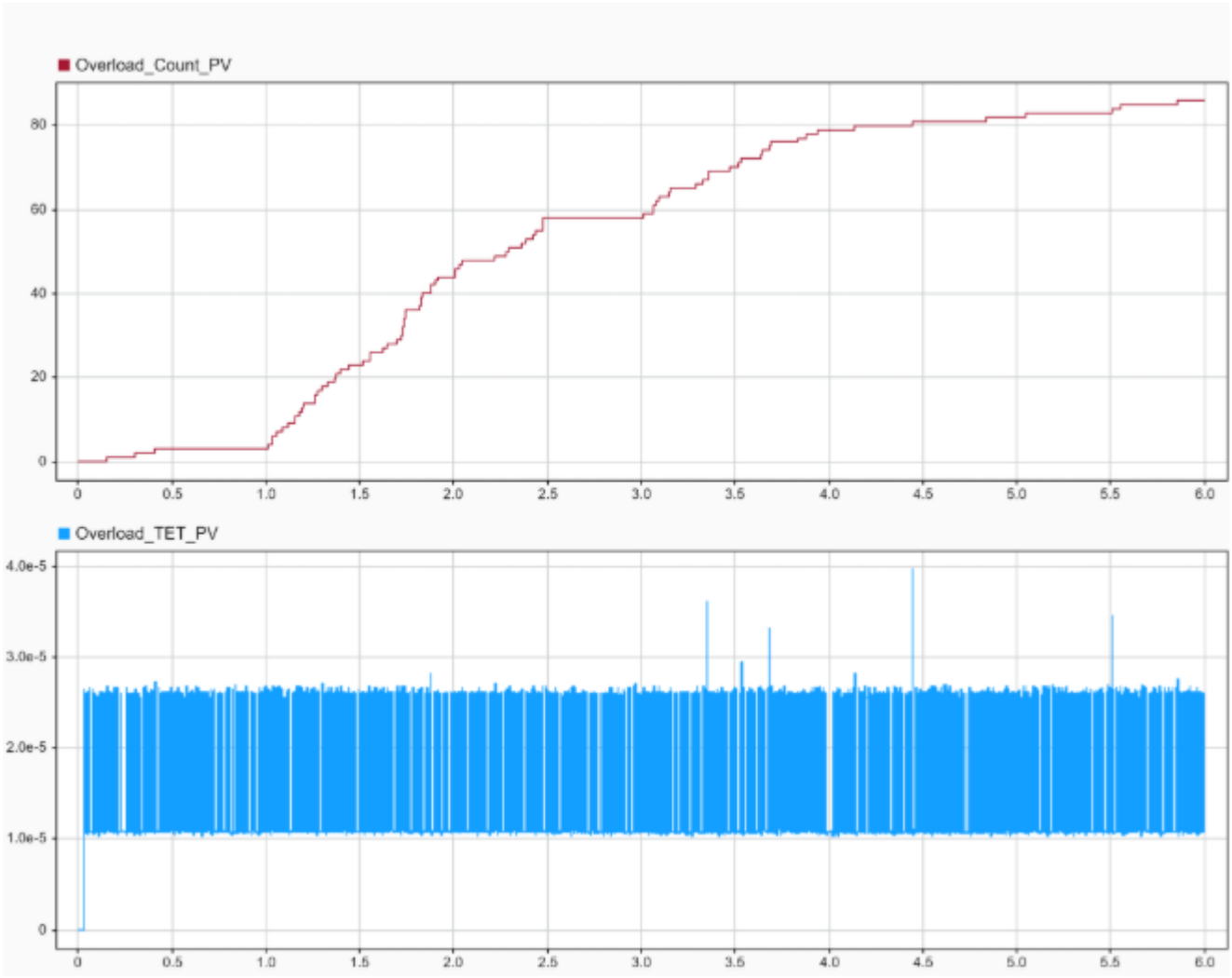
Then we prepared the model for code generation using slrealtime.tlc as system target file:

```
cs = getActiveConfigSet(model);
switchTarget(cs,'slrealtime.tlc',[]);
set_param(model,'SolverType','Fixed-step');
set_param(model,'SolverName','FixedStepDiscrete');
set_param(model,'GenCodeOnly', 'off');
set_param(model,'MaxIdLength', 95);
```

Now, assuming we have a Speedgoat target machine harwired to your host computer we connect, build, load, and run the model on the target with the following commands:

```
tg=slrealtime;
connect(tg);
slbuild(model);
load(tg,model);
pause(5);
tg.start;
```

Using the Data Inspector, we see that the average execution time exceeds the model sample time (2.5e-5sec), producing a large amount of overloads that prevent the model to accurately run in real-time.



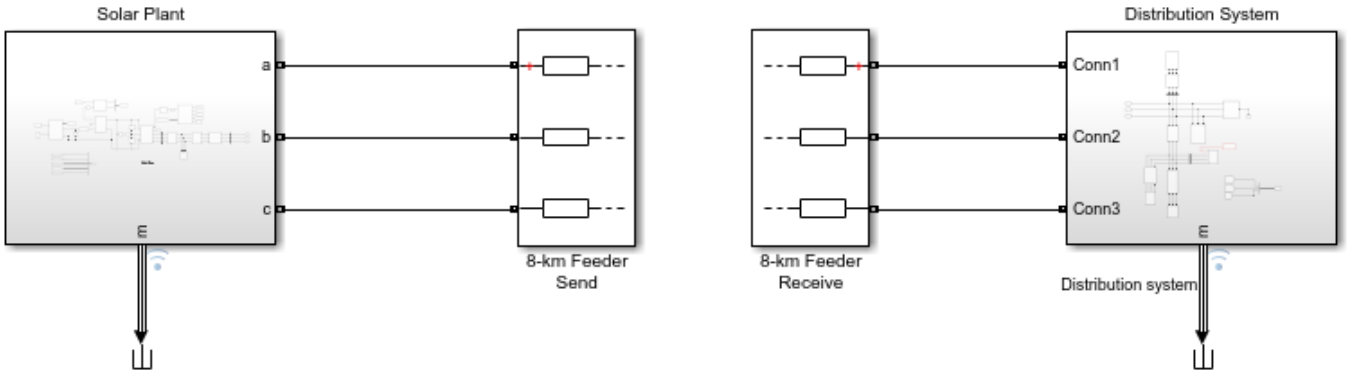

### 3. Split the Base Model with Decoupling Line Blocks

To avoid CPU overloads in real-time, two Decoupling Line blocks can be used to split the model into two separate tasks that can run concurrently on the Speedgoat multi-core target.

We can use the Line Decoupler App to replace the Distributed Parameter Line block, that connects the solar plant to the distribution system, with a pair of Decoupling Line blocks. We can access the App from the command line:

```
DecouplingLineReplace(model)
```

The App lists all the Distributed Parameters Line blocks in the model. We select the 8-km Feeder block in the list, and click **Replace** . The block is automatically replaced by a pair of Decoupling Line blocks that are tuned with the parameters of the original Distributed Parameter Line block.

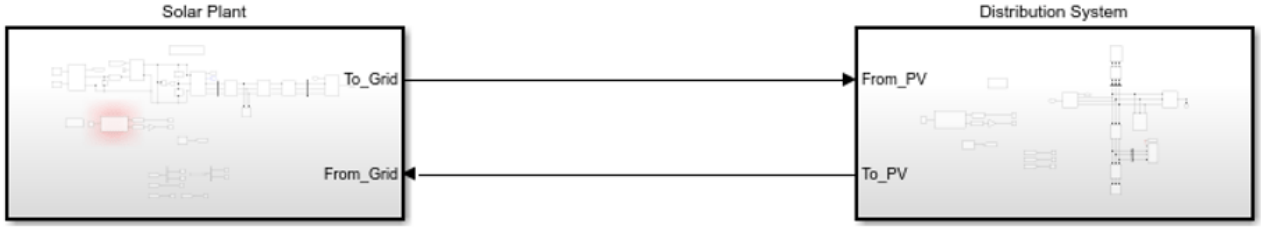


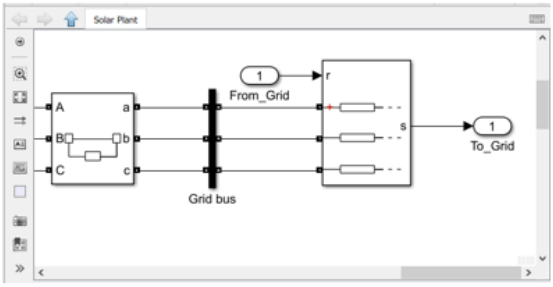

Now that there are no more electrical connections between the two ends of the line, we can move the *8-km Feeder Send* block inside the Solar Plant subsystem, and move the *8-km Feeder Receiving* block inside the Distribution System subsystem. We also move the powergui block and the Overload_PV blocks at top-level of diagram into the Solar Plant subsystem. We make a copy of both blocks into the Distribution System.

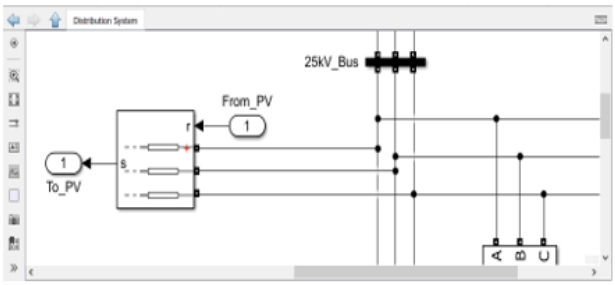
To run the two systems concurrently, we treat them as atomic units:

```
Subsystems = {'/Solar Plant','/Distribution System'};
set_param([model, Subsystems{1}],'TreatAsAtomicUnit','on' );
set_param([model, Subsystems{2}],'TreatAsAtomicUnit','on' );
```

In the Solar Plant susbsystem, we check the **Show send and receiving ports** parameter of the Decoupling Line block. This option replace the internal Goto and From blocks with Inport r and Outport s blocks. This allow us to connect an Inport block and an Outport to the block. We remove the measurement output port. In the Distribution System we do the same modifications. Finally we connect the two subsystems together:



### 4. Prepare the Decoupled Model for Concurrent Execution

The code below is a summary of the settings you need to make to the model to run it in real-time on a Speedgoat target:

- We need to specify the multicore architecture:

```
Simulink.architecture.config(model, 'Convert');
Simulink.architecture.importAndSelect(model,'Multicore');
Simulink.architecture.get_param(model,'ArchitectureName');
set_param(model,'ExplicitPartitioning','on');
```

- We need to define two periodic tasks:

```
for i=1:2
  Simulink.architecture.add('Task', [model, '/CPU/Periodic/Task', num2str(i)]);
  Simulink.architecture.set_param([model, '/CPU/Periodic/Task',num2str(i)], 'Period', 'Ts');
end
```

- We set deterministic data transfer:

```
dt = get_param(model, 'DataTransfer');
dt.DefaultTransitionBetweenSyncTasks = 'Ensure deterministic transfer (maximum delay)';
```

- We map the two periodic tasks to the two subsystems:

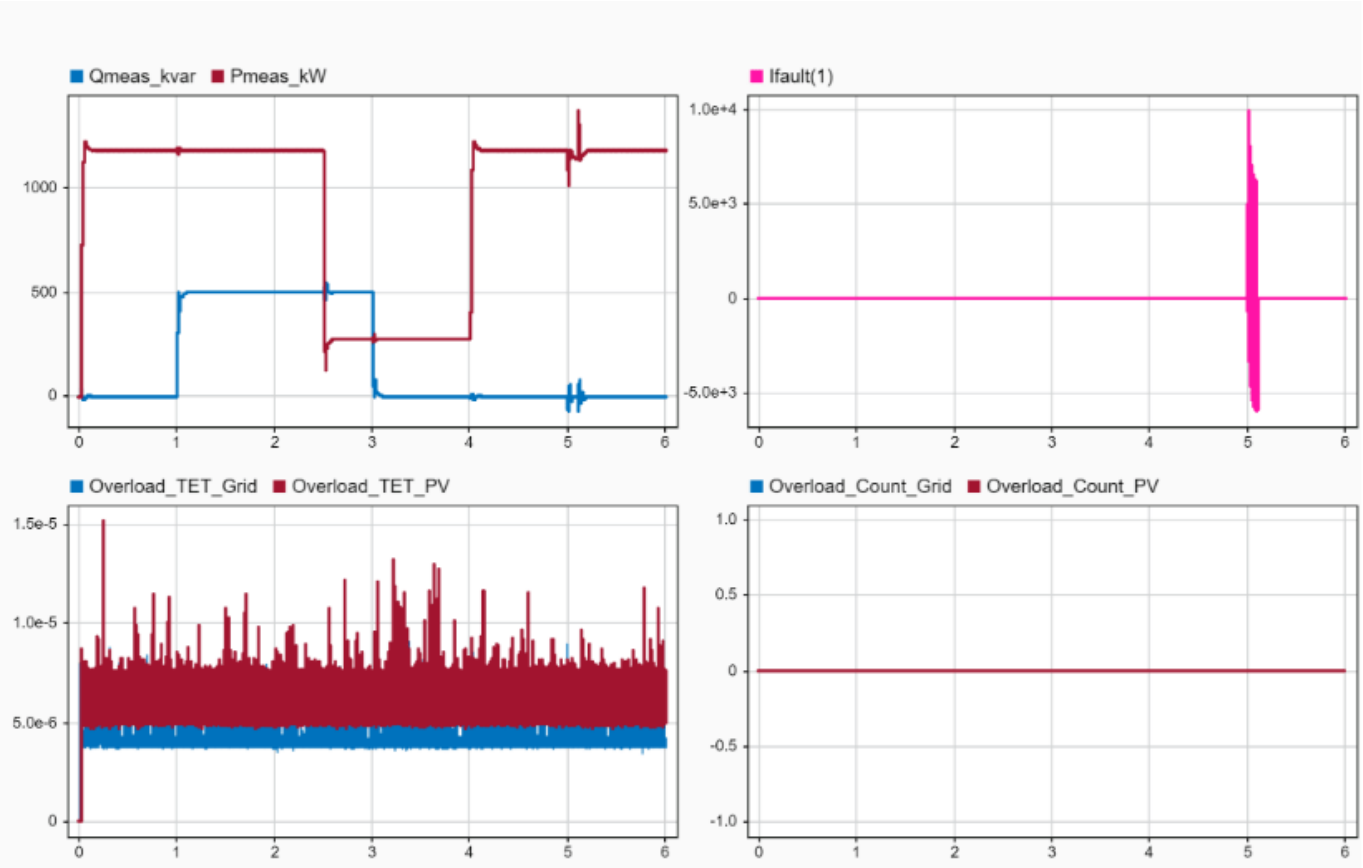
```
for i=1:2
  set_param([model, Subsystems{i}],'TargetArchitectureMapping', [model, '/CPU/Periodic/Task',num2
end
```

### 5. Build, Load, and Run the Decoupled Model on the Target

Now, assuming that our host computer is connected to our Speedgoat target we can build, load and run the decoupled model in real-time:

```
slbuild(model)

tg=slrealtime;
load(tg,model)
pause(5)

tg.start
```

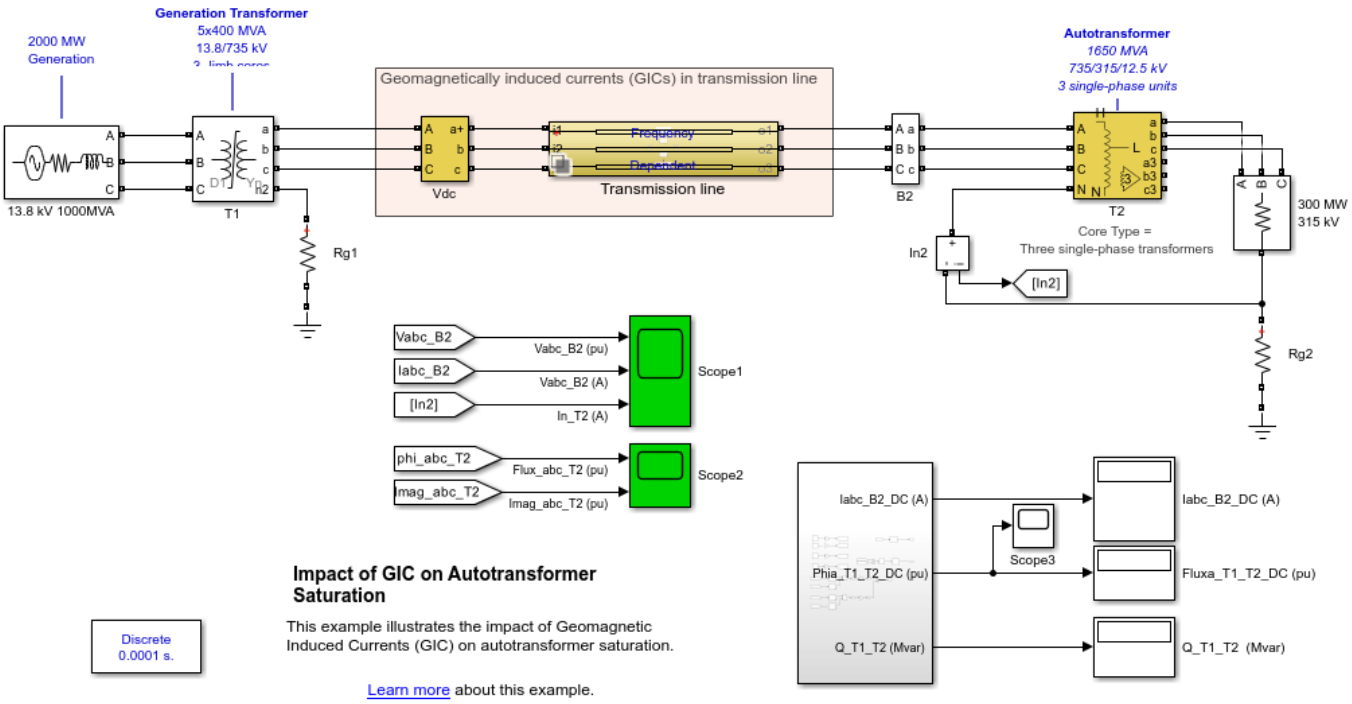
The next figure shows simulation signals in the Data Inspector. We see that the average execution time no longer exceeds the model sample time (subplot on the bottom left).




By using the Decoupling Line block and a multicore target, we can run the model in real-time without CPU overload.

# Impact of GIC on Autotransformer Saturation

This example illustrates the impact of Geomagnetic Induced Currents (GIC) on autotransformer saturation.




### Description

According to [1], GeoMagnetic Disturbances (GMDs) occur when earth is subjected to changes in the energized particle streams emitted by the Sun. Near the Earth's surface, these changes induce currents, known as geomagnetically induced currents (GICs), in long electrical conductor systems such as electric power transmission and distribution lines, communication lines, rail lines, and pipelines. Under strong GIC, transformers become a source of significant current harmonics due to saturation of their magnetic cores.

The example shows a 735-kV transmission network consisting of a 13-8 kV, 2000 MW generating station (10000-MVA short-circuit level), a 200-km, 735-kV transmission line, a 735/315/12.5 kV, a 1650-MVA autotransformer T2 and a 300-MW load. Autotransformer T2 is built with three single-phase cores, whereas generation transformer T1 uses a 3-limb core. A geomagnetic disturbance is producing a uniform 8 V/km electric field aligned with the transmission line. The resulting DC voltages induced on the transmission line are simulated by three DC voltage sources in series with the line (Vdc = 8*200= 1600 V).

As GIC currents are strongly dependent on line resistances, it is important to use a line model yielding an accurate resistance for both AC and DC phenomena. The line is modeled with the Distributed Parameters Line Frequency Dependant block and associated MAT file.

### Simulation

Run simulation for 10 sec in order to reach steady state. While simulating, observe voltage and current waveforms on Scope1 as well as T2 fluxes and magnetization currents on Scope2. DC components of line currents (GICs), DC component of T1 and T2 fluxes, as well as reactive powers absorbed by T1 and T2 are shown on the Display blocks.

Note the large DC component of T2 fluxes (+ 0.67 pu). Transformer T2 is therefore strongly saturated (See fluxes and Imag waveforms on Scope2). Large non sinusoidal magnetization currents of T2 result in strong harmonic voltage and current distortions (Scope1). Also, the 60-Hz component of magnetizing currents results in large reactive power absorption (Q_T2= 424 Mvar). On the opposite, transformer T1 which uses a 3-limb core has a low DC flux offset. The DC flux of T1 is still decreasing very slowly at t=10 s (see Scope3) and in order to reach steady state you would need to simulate during several hundreds of seconds.

### Impact of Autotransformer T2 Core Type

Although it is unlikely to build such a large power autotransformer (1650 MVA) with a single three-phase core, you will now observe impact of using a 3-limb core for T2.

In T2 block menu, change the Core type parameter from Three single-phase units to Three-limb core (core type) and run a new simulation. Observe that the transformer does not saturate anymore. DC fluxes of both T1 and T2 are decreasing very slowly and a simulation of several hundreds of seconds would be required to reach steady state.

In case of a 3-limb core, the three equal DC current components of magnetization currents produce a zero-sequence flux component which must circulate outside of the iron core, through the air and the tank. The high reluctance of this flux return path produces a low DC flux component, thus preventing transformer saturation.

### Impact of Line Model

The three line currents resulting from DC voltages induced along the line produce three equal DC currents (zero-sequence) which will return through the ground below the line. In DC, this ground return resistance is null, except for the grounding resistances at the two substations.

If you use a Distributed Parameter Line with constant RLC parameters, the ground resistance is determined by the positive- and zero-sequence line resistances (r1 and r0 parameters in ohms/km computed at 50 or 60 Hz). For a line of length L, this ground resistance in DC will be the same as the resistance specified at 50 or 60 Hz and will be given by Rline_ground = (r0-r1)/3*L ohms, whereas it should be zero.

To observe the impact of using a Distributed Parameter line (DPL) instead of the Frequency-Dependent (FD) line, set the Variant choice of the Transmission line block to Choice_2 (DPL 735-kV 200 km). Set the T2 Core type back to Three single-phase units. Run simulation and observe that GICs have dramatically decreased from 295 A to ~25 A. Use of the DPL line model gives wrong GIC because the DPL ground resistance at 0 Hz is the same as the ground resistance at 60 Hz whereas it should be zero.

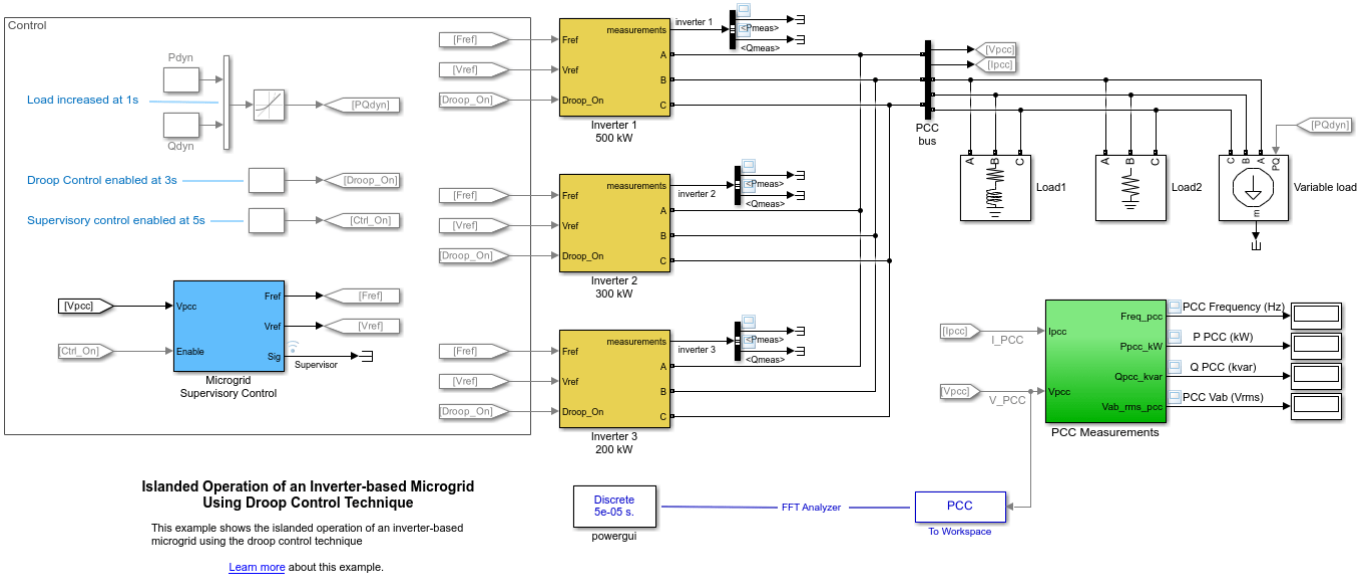
### References

**1** Geomagnetic Disturbance Monitoring, Approach and Implementation, United States Department of Energy, January 2019

2 Impact of GIC on Transformers and the Transmission Network, Dietrich Bonmann, ABB AG Bad Honnef, March 2016

# Islanded Operation of an Inverter-based Microgrid Using Droop Control Technique

This example shows the islanded operation of an inverter-based microgrid using the droop control technique.




## Description

The example illustrate the operation of an inverter-based microgrid disconnected from the main grid (islanded mode), using the droop control technique. The U.S. Department of Energy defines a microgrid as a local energy grid with control capability, which means it can disconnect from the traditional grid and operate autonomously.

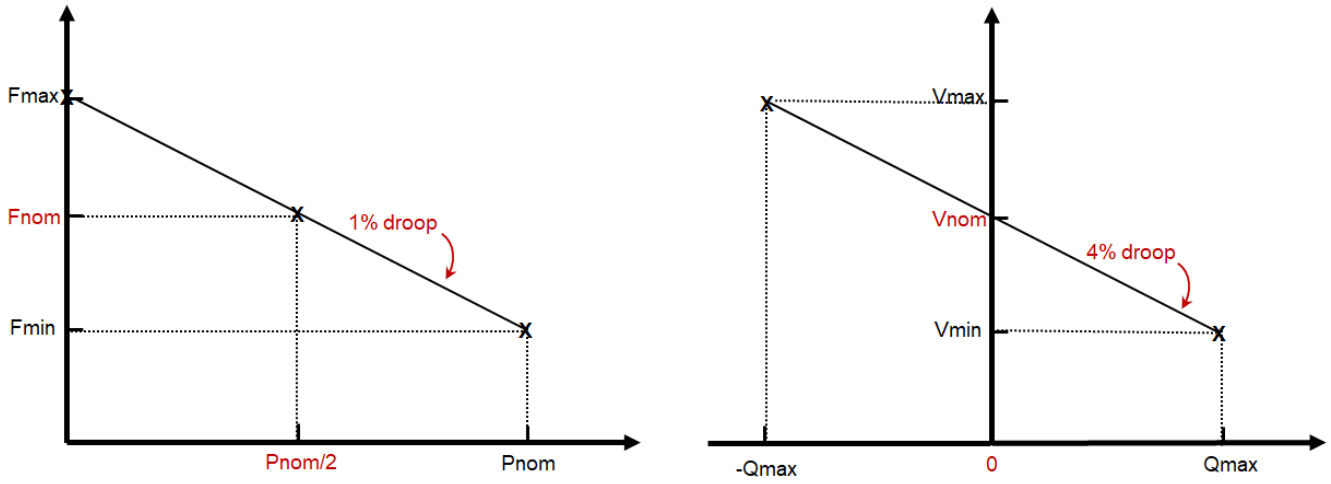
Droop control is a well-established technique to control an autonomous grid. In fact, the Active Power/ Frequency (P/F) and Reactive Power/Voltage (Q/V) droop control mimics the operation of synchronous generators in a transmission system. With the droop control technique, PLL are not required to achieve system-wide synchronization because all inverters reach the same frequency. In addition, power sharing among each inverter can be achieved since each inverter gives power in proportion to its capacity.

The mocrogrid consists of three parallel inverters subsystems, with power ratings of 500 kW, 300 kW and 200 kW respectively, connected to the PCC (Point-of-Common-Coupling) bus. A dynamic load model is used to dynamically change the microgrid total load. The Microgrid Supervisory Control system, when enabled, modifies the inverters P/F and Q/V droop set points in order to bring back the microgrid frequency and voltage at their nominal values (60 Hz and 600 Volts respectively).

Each inverter subsystem contains a three-phase two-level power converter, an LC filter, a 480/600V transformer as well as an ideal DC source to represent the DC link of a typical renewable energy generation system (such as PV array, wind turbine, battery energy storage system). Each subsystem also includes a control system and a PWM generator feeding the inverter.

The main components of the inverter control system are:

**Droop Control:** The Figure shows the droop characteristics of the inverter control.




The droop P/F is set to 1%, meaning that microgrid frequency is allowed to vary from 60.3 Hz (inverter produces no active power) to 59.7 Hz (inverter produces its nominal active power). The droop Q/V is set to 4%, meaning that the microgrid voltage at the PCC bus is allowed to vary from 612 Vrms (inverter produces its full inductive power) to 588 Vrms (inverter produces its full capacitive power). Note that Qmax is specified as half of the nominal active power Pnom.

**Measurements:** Based on the frequency value given by the Droop control, the measurement subsystem computes the active and reactive power generated by the inverter. It also computes the d-q components of the three-phase voltages and currents at the microgrid PCC bus.

**Voltage Regulators:** Reference voltage Vref given by the Droop Control is fed to the Voltage Regulators. The regulators process the measured d-q voltages and reference voltage Vref to generate the reference currents Id_ref and the Iq_ref.

**Current Regulators:** The Id_ref and the Iq_ref reference currents are fed to the Current Regulators. The regulators process the measured and reference currents to produce the required d-q voltages (VdVq_conv) for the inverter. Note that the regulators dynamics benefits from a feed-forward calculation.

**Vref Generation:** VdVq_conv are scaled and transformed to a three-phase signal Vref feeding the PWM modulator generating pulses to the inverter.

### Simulation

At 1 s, the total microgrid load is increased from 450kW/100kvar to 850kW/200kvar. At 3 s, droop control is enabled on all inverters. We can see that the microgrid load is now shared equally among the three inverters. At 5 s, the supervisory control is enabled. The frequency is then being slowly increased to 60Hz and the line voltage to 600V.

To demonstrate the impact of the inverters PWM carrier's initial phase on the PCC bus voltage harmonic content, first open the FFT Analyzer App to perform an FFT analysis of the PCC phase A bus voltage.

In the App, set the **Structure with time** parameter to `PCC` , the **Signal** parameter to `V_PCC`, and the **Dimension** parameter to `1` to analyze the PCC phase A bus voltage. Set the **Zoom on** parameter to

`FFT window`, the **Start time** parameter to `7.9`, and the **Max frequency** parameter to `7000`. Click **Compute FFT**. In the FFT plot, the maximum harmonic occurs around the switching frequency (2700 Hz) and is close to 2%.

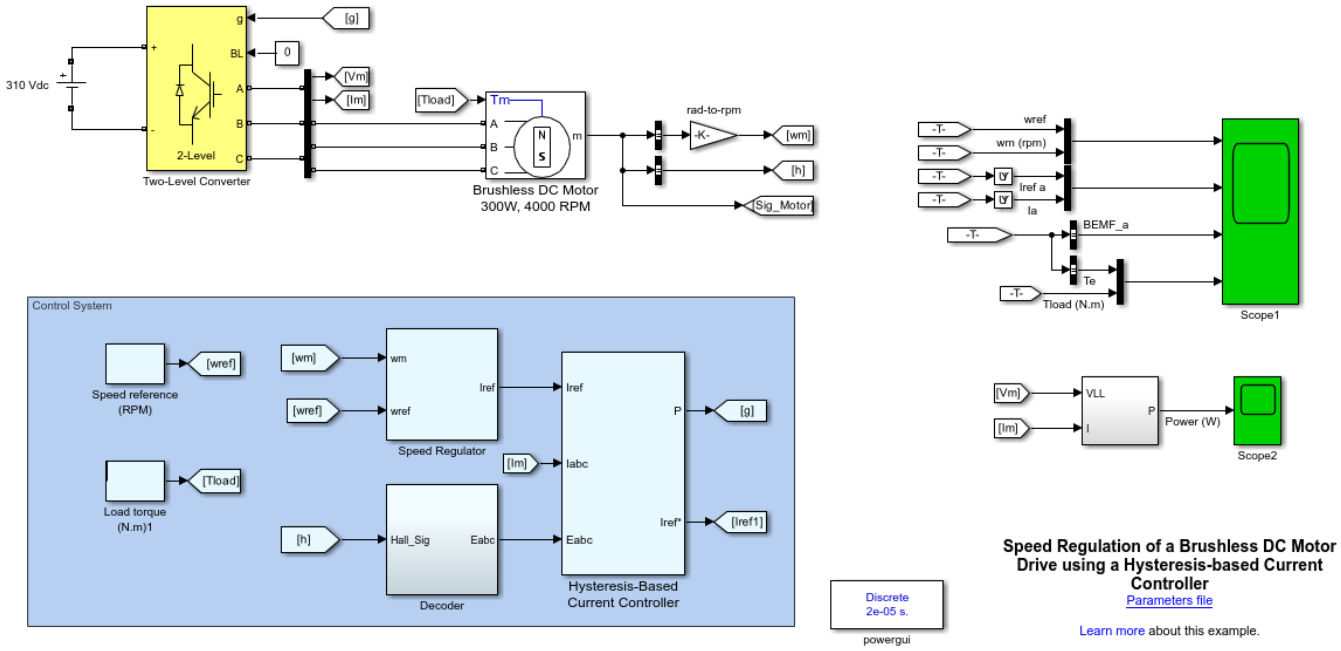
Now, double-click on the Inverter 2 (300 kW) subsystem and change the Carrier initial phase parameter to -90 degrees. Rerun the simulation and again, perform an FFT analysis on the PCC phase A voltage. You should see that this new carrier phase setting significantly reduces the harmonic content around the switching frequency (2700 Hz). This is due to the fact that Inverter 1 carrier phase is set to +90, so switching harmonics are then partially canceled.

### References


1. NREL report: Research Roadmap on Grid-Forming Inverters Yashen Lin, Joseph H. Eto, Brian B. Johnson, Jack D. Flicker, Robert H. Lasseter, Hugo N. Villegas Pico, Gab-Su Seo, Brian J. Pierre, and Abraham Ellis
2. IEEE paper: Dynamic Operation and Control of a Multi-DG Unit Standalone Microgrid Hossein arimi-Davijani, Student Member and Olorunfemi Ojo, Senior Member, 2011

# Speed Regulation of Brushless DC Motor Drive Using a Hysteresis-Based Current Controller

This example shows the speed regulation of a brushless DC motor (BLDC) drive using a hysteresis-based current controller.




### Description

BLDC motors differ from brushed DC motors in that they use electronic commutation (rather than mechanical) to provide power to the windings. In its basic form, a brushless DC motor consists of a trapezoidal back-EMF permanent magnet synchronous motor fed by a three-phase inverter. A position sensor attached to the rotor provides the position signals required to synchronize the stator currents with the back-EMFs so that the motor operates as a synchronous motor at all times.

Because the back-EMFs are of trapezoidal form (with a flat area of 120 degrees), the maximum torque with the lowest ripple is developed if the currents are kept constant during the time intervals when the back-EMFs are also constant. This condition implies a six-step operation for the inverter where only two phases are conducting at a time.

### Electrical Model

The DC bus, modeled by an ideal DC voltage source of 310 V, connects to a three-phase, two-level converter. This converter generates the appropriate three-phase voltages to operate the 300 W, 4000 rpm BLDC motor.

### Control System

The main components of the control system are:

1. Speed Regulator — The regulator compares the actual motor speed to the speed reference. If the motor needs to be accelerated, the regulator increases the reference current magnitude in order to create more torque. If the motor speed is higher than the reference, the regulator reduces the reference current magnitude.
2. Decoder — Based on the Hall sensor signals that detect the rotor position, the decoder outputs three square-wave signals that are synchronized to the motor back-EMFs.
3. Hysteresis-Based Current Controller — Based on the reference current magnitude and the signals provided by the decoder, three current references are generated and compared to the measured motor phase currents. The errors are fed to a hysteresis control with an adjustable band to generate the required gating signals for the three-phase inverter. The switching frequency is variable and dependent on the value of the hysteresis band and the dynamics of the stator circuit.

### Simulation

Run the simulation and observe the waveforms on the Scope blocks. Initially, the motor rotates at 3000 rpm with no load.

At 0.05 s, a load torque of 0.7 N.m is applied to the motor. The control system increases the reference current in order to maintain the motor speed at 3000 rpm.

At 0.1 s, the speed reference is reduced to 1500 rpm. To respect this new speed set point, the control system produces a large negative torque. Notice the reduced frequency of the motor phase currents.

At 0.15 s, a negative load torque of -0.7 N.m is applied to the motor. The motor is now acting as a generator producing 68 Watts.

### Real-Time Simulation

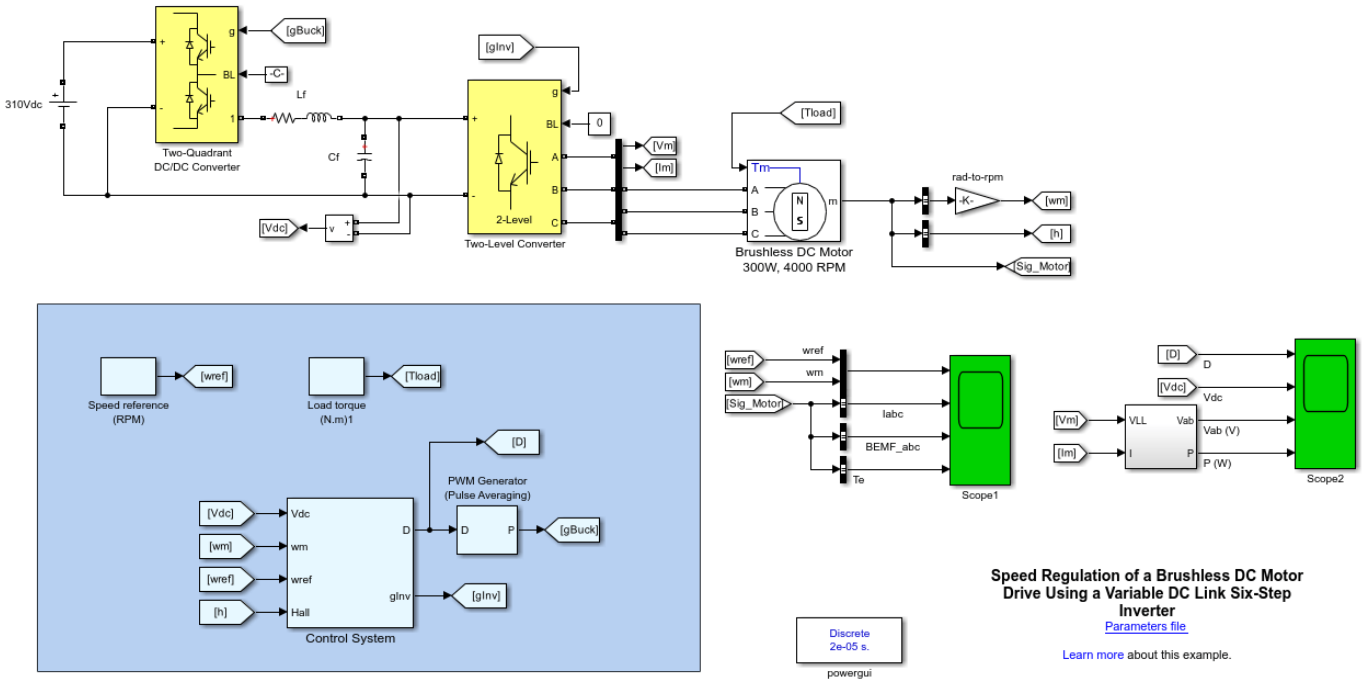
If you have Simulink Real-Time and a Speedgoat target computer, you can run this model in real time.

1. Open the Configuration Parameters window (or press **Ctrl+E** ), click **Code Generation** , and set **System target file** to `slrealtime.tlc` .
2. Connect to the target and, in the **Real-Time** tab, click **Run on Target**.

Your model will then be automatically built, deployed, and executed on the target. Depending on your target streaming bandwidth, you may have to reduce the number of signals transferred in real-time from the target to the host computer.

# Speed Regulation of a Brushless DC Motor Drive Using a Variable DC Link Six-Step Inverter

This example demonstrates the speed regulation of a brushless DC (BLDC) motor drive using a variable DC link six-step inverter.




### Description

In its basic form, BLDC motors consist of a trapezoidal back-EMF permanent magnet synchronous motor fed by a three-phase inverter. A position sensor attached to the rotor provides the position signals required to synchronize the stator currents with the back-EMFs so that the motor operates as a synchronous motor at all times. Because the voltage magnitude is proportional to the motor speed, the speed can then be regulated by varying the DC link voltage connected to the three-phase inverter.

### Electrical Model

A DC/DC converter is fed by an ideal DC source and its output filter allows you to vary the DC link voltage. The DC bus is connected to a three-phase, two-level converter. This converter generates the appropriate three-phase voltage for the operation of the 300 W, 4000 rpm BLDC motor.

### Control System

The main components of the control system are:

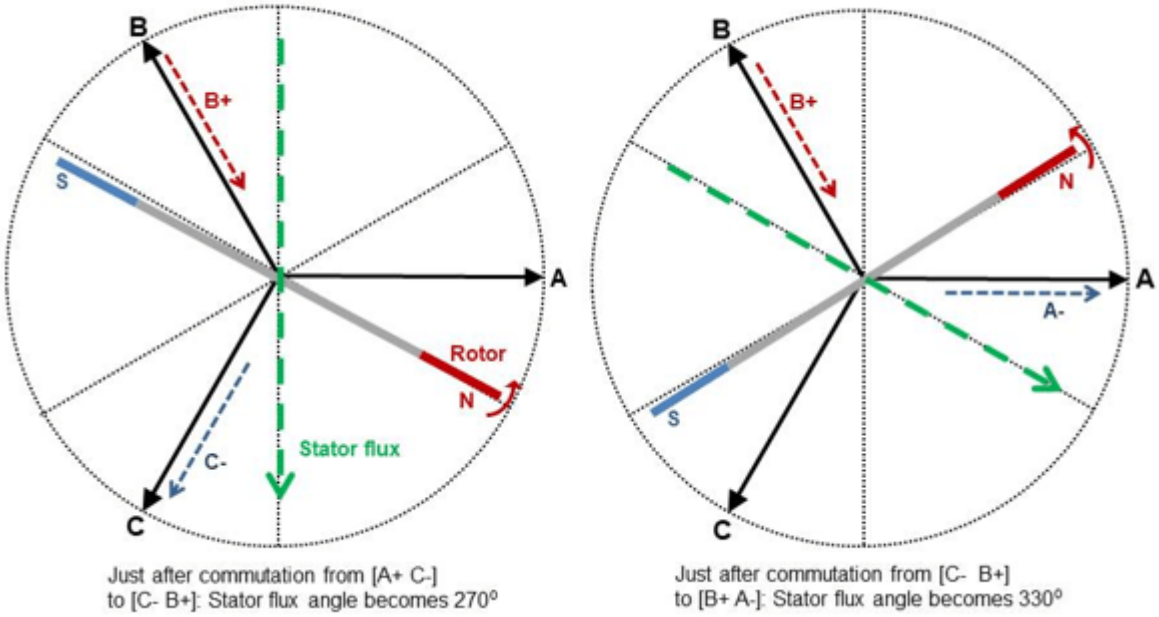
1. Speed Regulator — The regulator compares the actual motor speed to the speed reference and generates the DC voltage reference.
2. Voltage Regulator — The measured DC link voltage is compared to the reference value. The resulting error is fed to the anti-windup PI regulator. To correct the error, the regulator outputs the required duty cycle (D) value to the DC-DC converter.

**3** Commutation Sequences and Pulses Generation — Because the back-EMFs are of trapezoidal form (with a flat area of 120o), the maximum torque with the lowest ripple develops if the currents are kept constant during the time intervals when the back-EMFs are also constant. This condition implies a six-step operation for the inverter where only two phases are conducting at a time. Based on the rotor position (obtained from the Hall Effect sensors) and the motor rotation direction, this block generates appropriate pulses to the three-phase inverter feeding the motor.

This table lists the six possible commutation sequences for the counterclockwise rotation of the motor:

| Rotor alignment (degrees) | Hall sensor [ha h_b hc] | Phase A (V) | Phase B (V) | Phase C (V) | Stator flux position (degrees) |
|---|---|---|---|---|---|
| 30 - 90 | [ 0 1 0 ] | - Vdc | + Vdc | NC | 330 |
| 90 - 150 | [ 0 1 1 ] | - Vdc | NC | + Vdc | 30 |
| 150 - 210 | [ 0 0 1 ] | NC | - Vdc | + Vdc | 90 |
| 210 - 270 | [ 1 0 1 ] | + Vdc | - Vdc | NC | 150 |
| 270 - 330 | [ 1 0 0 ] | + Vdc | NC | - Vdc | 210 |
| 330 - 30 | [ 1 1 0 ] | NC | + Vdc | - Vdc | 270 |

This figure shows two commutation sequences for the counterclockwise rotation of the motor.




For proper operation of a BLDC motor, it is necessary to keep the angle between the stator and rotor flux close to 90°. With six-step control, the motor has a total of six possible stator flux vectors. The stator flux vector must be changed at a certain rotor position. However, with a six-step control technique it is not possible to keep the angle between the rotor flux and the stator flux at 90°. The real angle varies from 60° to 120°.

For the [C- B+] sequence, you can verify from the left image in the diagram that the position of the rotor north pole varies from 60° to 120° ahead of the stator north pole, with repulsion between poles

of same polarity producing a counterclockwise rotation. At the same time, the position of the rotor south pole varies from 120° to 60° behind the stator north pole, with attraction between two opposite poles still producing counterclockwise rotation.

### Simulation

Run the simulation and observe the waveforms on Scope 1. Initially, the motor rotates at 4000 rpm with no load.

At 0.1 s, a load torque of 0.6 N.m is applied to the motor. The control system increases the DC link voltage reference in order to maintain the motor speed at 4000 rpm.

At 0.25 s, the speed reference is reduced to 1000 rpm. The control system significantly decreases the DC link voltage reference in order to respect the new speed reference. Notice the reduced frequency of the motor phase currents.

On Scope 2, you can observe the DC regulator output (D) as well as the DC link voltage variations.

### Real-Time Simulation

If you have Simulink Real-Time and a Speedgoat target computer, you can run this model in real time.

1. Open the Configuration Parameters window (or press **Ctrl+E** ), click **Code Generation** , and set **System target file** to `slrealtime.tlc` .
2. Connect to the target and, in the **Real-Time** tab, click **Run on Target**.

Your model will then be automatically built, deployed, and executed on the target. Depending on your target streaming bandwidth, you may have to reduce the number of signals transferred in real-time from the target to the host computer.

### References

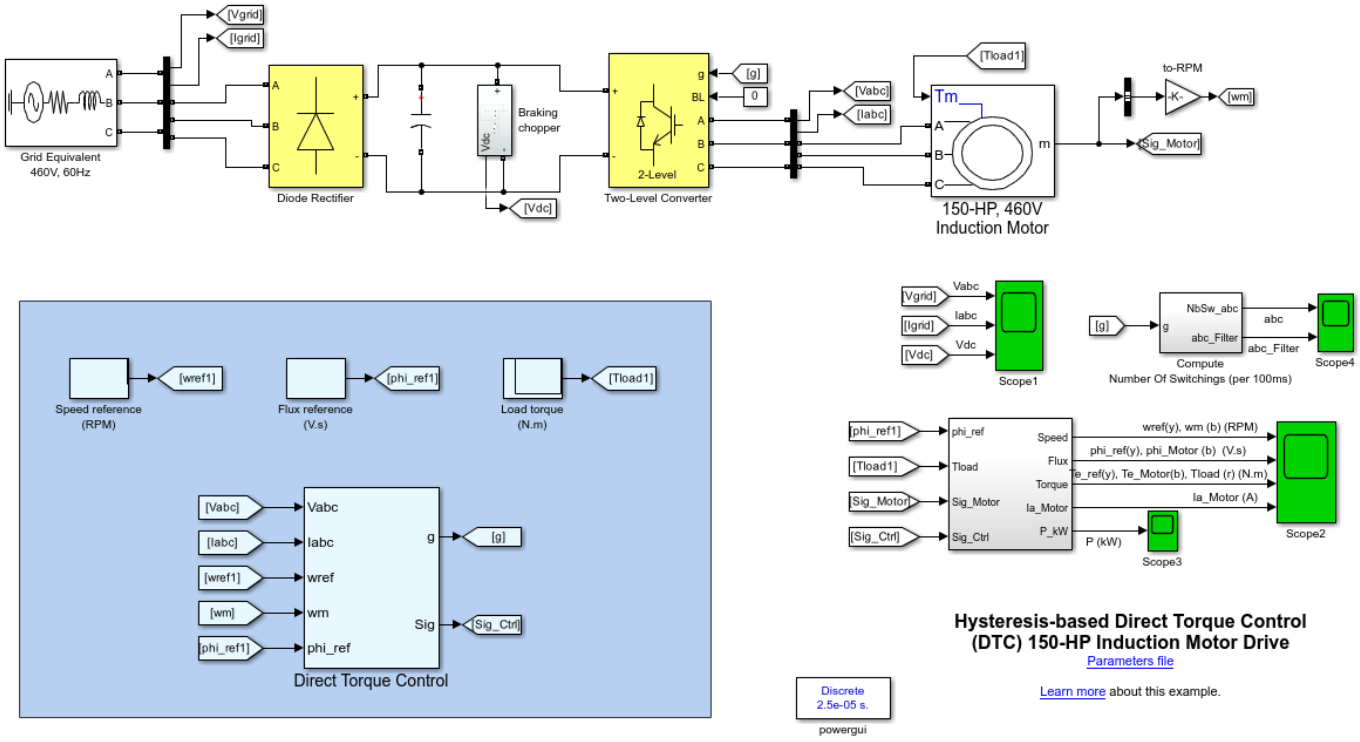
Musil, J. *3-Phase BLDC Drive Using Variable DC Link Six-Step Inverter.* Freescale Czech Systems Laboratories. 2006.

### See Also

Motor Control (Part 1): An Introduction to Brushless DC Motors, Melda Ulusoy, MathWorks https://www.mathworks.com/support/search.html/videos/brushless-dc-motors-introduction-1564728874059.html?fq=asset_type_name:video%20category:physmod/sps/index&page=1

# Direct Torque Control of an Induction Motor Drive

This example demonstrates the speed regulation of a variable-frequency AC drive using a hysteresis-based direct torque control (DTC) technique.




### Electrical Model

The electrical energy is supplied by a three-phase AC/DC diode rectifier connected to a 460 V, 60 Hz grid equivalent. The DC bus is connected to a three-phase, two-level converter. This converter generates the variable voltage and frequency required for variable-speed operation of the 150 HP induction motor. In addition, a braking chopper is connected to the DC bus in order to dissipate the kinetic energy of the motor during deceleration.

An inverter-fed induction motor drive can be controlled through various techniques depending on application, desired performance, and controller design complexity. Commonly used schemes are scalar control (V/Hz control or open loop flux control) or vector control (field-oriented control or direct torque control). This example uses a hysteresis-based direct torque control (DTC) technique.

### Direct Torque Controller

Direct torque control (DTC) is a technique that allows you to instantaneously control the motor magnetic flux and its electromagnetic torque in a decoupled way. Controlling the torque directly permits accurate static and dynamic speed regulation. The main components of the DTC subsystem are:

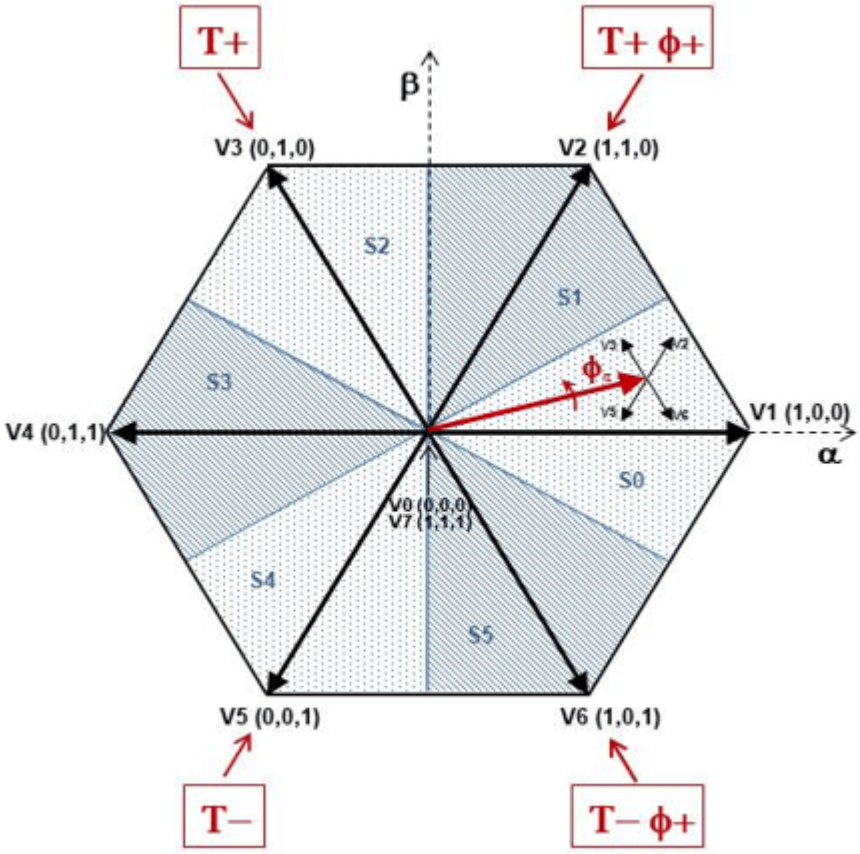
1. Flux and Torque Calculation — The stator flux linkage is estimated by integrating the stator voltages, and torque is calculated based on the estimated flux and the motor currents.

2 Speed Regulator — The regulator compares the actual motor speed with the speed reference and generates the torque reference.

3 Hysteresis Control — The calculated flux magnitude and torque are compared with the reference values. When the resulting flux or torque error crosses either the positive or negative hysteresis band value, a control signal is activated in order to correct the error.

4 Optimal Switching — Pulses to the motor inverter are produced based on the control signals generated by the hysteresis control and the stator flux linkage position.

The figure below illustrates the strategy used to determine the best voltage vector when the flux linkage is located in sector 0.




The image shows four cases:

1 V3 is selected when the electromagnetic torque should be increased and the flux should stay unchanged. Selecting the V3 voltage vector speeds up the flux and thus applies an acceleration torque to the rotor while slightly decreasing the flux magnitude.

2 V2 is selected when the electromagnetic torque should be increased and the flux should be increased. Selecting the V2 voltage vector slightly speeds up the flux and increases its magnitude.

3 V6 is selected when the electromagnetic torque should be reduced and the flux should be increased. Selecting the V6 voltage vector slows down the flux and thus applies a deceleration torque to the rotor while increasing the flux magnitude.

4. V5 is selected when the electromagnetic torque should be reduced and the flux should stay unchanged. Selecting V5 the voltage vector applies a deceleration torque to the rotor and slightly decreases the flux magnitude. Note that voltage vectors V1 and V4 are not selected in sector 0. Using these two vectors would have too much negative impact on the desired flux value. Finally, to keep the torque and the flux unchanged, the null voltage vectors V0 or V7 are selected.

When the flux linkage vector moves to sector 1, the selected voltage vectors become V4 for case 1, V3 for case 2, V1 for case 3, and V6 for case 4, and vectors V2 and V5 are not used. This 60 degree shift in the voltage vectors happens each time the flux linkage vector enters a new sector.

### Simulation

Run the simulation and observe the waveforms on Scope2. Initially, the flux reference is set to 0.9 V.s.

At 0.1 s, the speed reference is set to 1500 RPM and the motor starts to accelerate. The motor speed precisely follows the speed reference, whose maximum rate of change is limited to 1200 rpm/s. The 1500 RPM set point is reached at 1.35 s.

At 1.5 s, a load torque of 500 N.m is applied to the motor. The DTC control operates to maintain the motor speed at 1500 RPM.

At 2 s, the load torque is reduced to 50 N.m and at 2.5 s, the speed reference is reduced to 500 RPM. Observe on Scope 1 that the braking chopper operation dissipates the kinetic energy produced by the motor in order to avoid overvoltage on the DC bus. Finally, at 3.5 s, the flux reference is increased from 0.9 to 1.0 V.s.

### Real-Time Simulation

If you have Simulink Real-Time and a Speedgoat target computer, you can run this model in real time.

1. Open the Configuration Parameters window (or press **Ctrl+E** ), click **Code Generation** , and set **System target file** to `slrealtime.tlc` .
2. Connect to the target and, in the **Real-Time** tab, click **Run on Target**.

Your model will then be automatically built, deployed, and executed on the target. Depending on your target streaming bandwidth, you may have to reduce the number of signals transferred in real-time from the target to the host computer.

### References

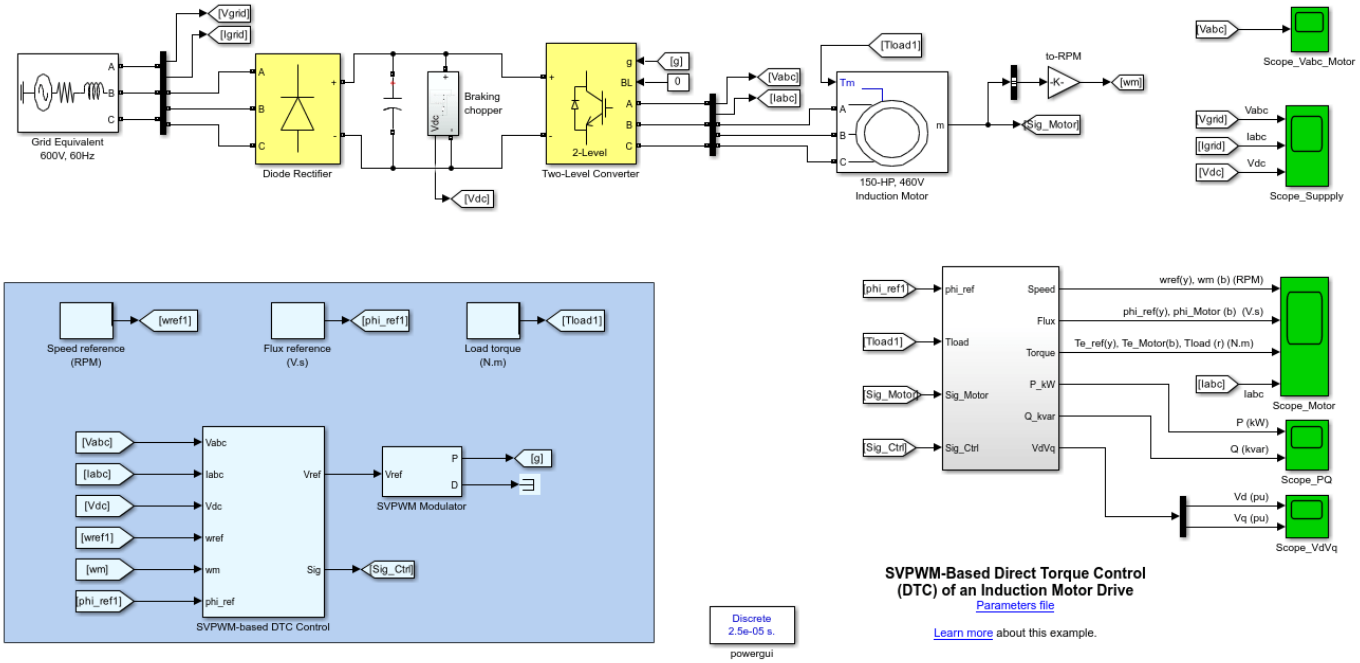
1. M. Cirrincione, M. Pucci, G. Vitale. Power Converters and AC Electrical Drives with Linear Neural Networks. CRC Press, 2012

2. Technical guide No. 1 Direct torque control – the world’s most advanced AC drive technology, ABB 2011

3. T. Wildi, G. Sybille. Électrotechnique (4th edition). Les Presses de l’Université Laval, 2005.

# Direct Torque Control with Space Vector Modulation of an Induction Motor Drive

This example demonstrates the speed regulation of a variable-frequency AC drive using the direct torque control (DTC) technique with space vector modulation.




## Description

### Electrical Model

The electrical energy is supplied by a three-phase AC/DC diode rectifier connected to a 600 V, 60 Hz grid equivalent. The DC bus is connected to a three-phase, two-level converter. This converter generates the variable voltage and frequency required for variable-speed operation of the 150 HP induction motor. In addition, a braking chopper is connected to the DC bus in order to dissipate the kinetic energy of the motor during deceleration.

An inverter-fed induction motor drive can be controlled through various techniques depending on the application, desired performance, and controller design complexity. Commonly used schemes are scalar control (V/Hz control or open loop flux control) or vector control (field-oriented control or direct torque control). In our example, we use the DTC technique with space vector pulse width modulation (SVPWM).

Compared to the classical hysteresis-based DTC, the SVPWM-DTC technique has a fixed switching frequency. Furthermore, this technique significantly reduces the motor torque ripple in steady-state operation. Refer to the Direct Torque Control of an Induction Motor Drive example to see the torque ripple of a motor drive using hysteresis-based DTC control.

### SVPWM-Based DTC Controller

DTC is a control technique that allows you to instantaneously control the motor magnetic flux and its electromagnetic torque in a decoupled way. Controlling the torque directly permits accurate static and dynamic speed regulation.

The main components of the DTC subsystem are:

1. Flux and Torque Calculation — Stator flux linkage is estimated by integrating the stator voltages, and torque is calculated based on the estimated flux and the motor currents.
2. Speed Regulator — The regulator compares the actual motor speed with the speed reference and generates the torque reference.
3. Flux and torque regulators — The calculated flux magnitude and torque are compared with the reference values. The resulting flux and torque errors are fed to anti-windup PI regulators. The output of the flux regulator is the direct-axis reference voltage Vd_ref and the output of the torque regulator is the quadrature-axis reference voltage Vq_ref.
4. Scaling and transformation — Vd_ref and Vq_ref are scaled and transformed to a three-phase signal Vref using the rotating frame reference given by the flux position phi_pha.

The output Vref of the DTC subsystem is fed to a SVPWM modulator that generates pulses to the motor inverter.

### Simulation

Run the simulation and observe waveforms on Scope_Motor. Initially, the flux reference is set to 0.9 V.s.

At 0.1 s, the speed reference is set to 1500 RPM and the motor starts to accelerate. You can see that the motor speed precisely follows the speed reference, whose maximum rate of change is limited to 1200 RPM/s. The 1500 RPM set point is reached at 1.35 s.

At 1.5 s, a load torque of 500 N.m is applied to the motor. The DTC control maintains the motor speed at 1500 RPM.

At 2 s, the load torque is reduced to 50 N.m and at 2.5s, the speed reference is reduced to 500 RPM. Observe on Scope Supply that the braking chopper operation dissipates the kinetic energy produced by the motor in order to avoid overvoltage on the DC bus.

At 3.5 s, the flux reference is increased from 0.9 to 1.0 V.s

### Real-Time Simulation

If you have Simulink Real-Time and a Speedgoat target computer, you can run this model in real time.

1. Open the Configuration Parameters window (or press **Ctrl+E** ), click **Code Generation** , and set **System target file** to `slrealtime.tlc` .
2. Connect to the target and, in the **Real-Time** tab, click **Run on Target**.

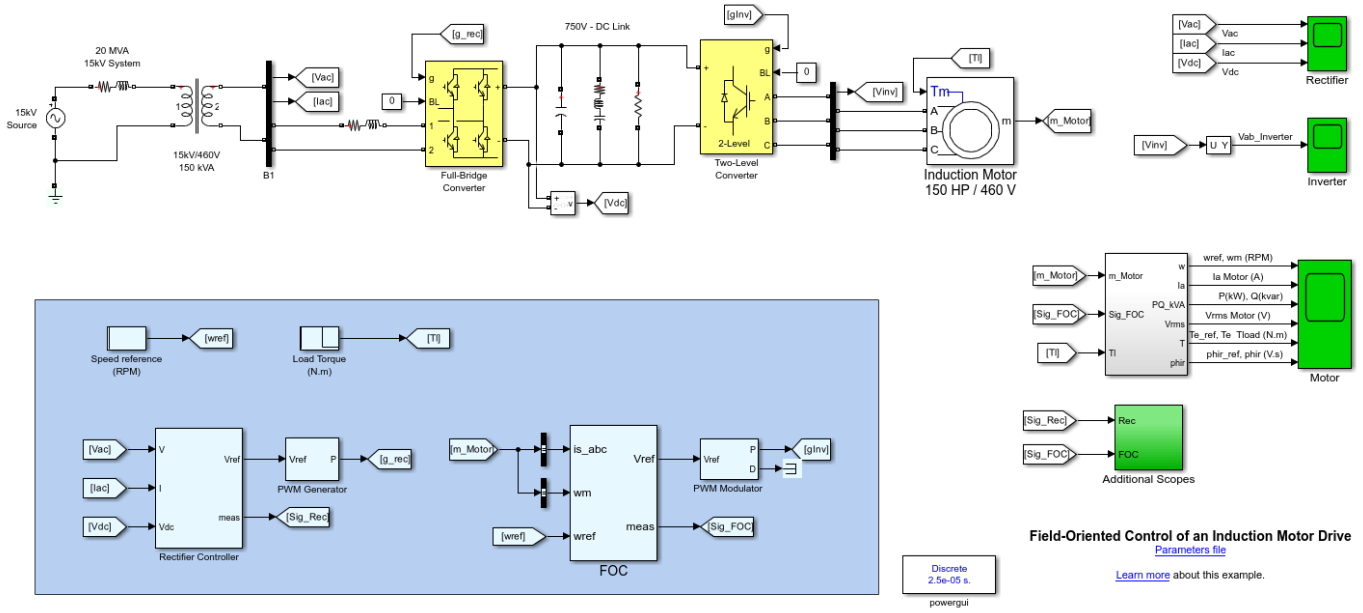
Your model will then be automatically built, deployed, and executed on the target. Depending on your target streaming bandwidth, you may have to reduce the number of signals transferred in real-time from the target to the host computer.

### References

Cirrincione, M., M. Pucci, G. Vitale. Power Converters and AC Electrical Drives with Linear Neural Networks. CRC Press, 2012.

# Field-Oriented Control of an Induction Motor Drive Used in a Ground Transportation System

This example demonstrates the operation of a variable-speed AC drive in a trolley bus traction system.




### Traction System

The electric energy is supplied by a 15 kV, single-phase, 60 Hz catenary through a 15 kV/460 V transformer. The transformer secondary is connected to a pulse-width modulation (PWM) AC/DC full-bridge converter producing a DC voltage of 750 V on the DC link. This voltage is filtered by a filter tuned to the second harmonic (120 Hz). The DC link voltage is connected to a PWM-controlled, three-phase, two-level converter. This converter generates the variable voltage and frequency required for variable-speed operation of the induction motor. The AC/DC converter allows the power flow to reverse during braking period. The kinetic energy of the bus is then converted into electric energy and injected back to the power system. In addition, unity power factor operation is possible with this kind of converter.

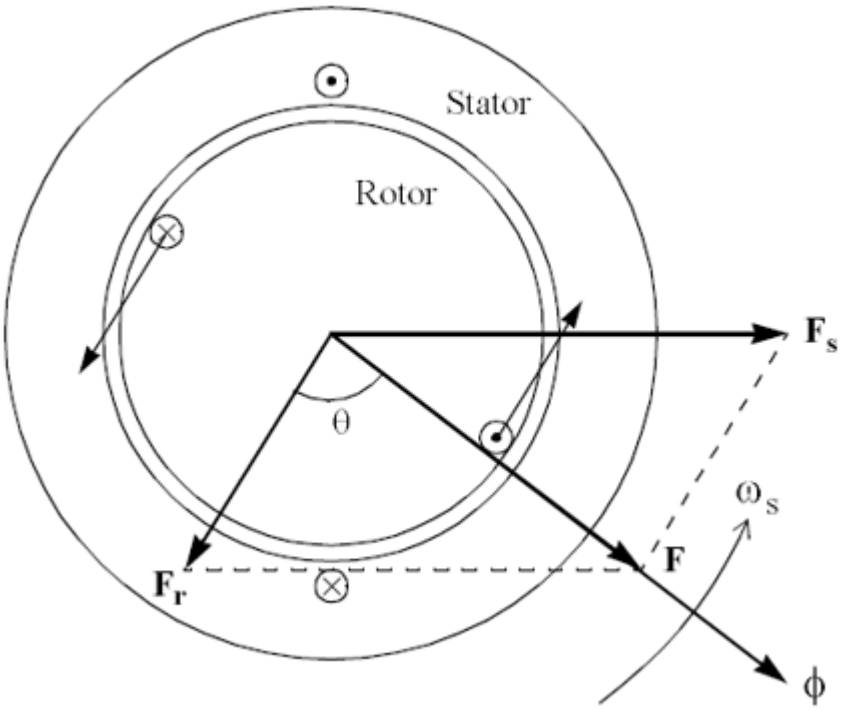
An inverter-fed induction motor drive can be controlled by using various schemes depending on the application, desired performance, and complexity of the controller design. Commonly used schemes are scalar control (V/Hz control or open loop flux control) or vector control (field-oriented control or direct torque control). In these schemes, the motor speed is controlled by changing the supply frequency. In order to develop an optimum torque without magnetic saturation, it is necessary to maintain a constant air gap flux near the nominal value so that the voltage is kept proportional to the frequency. In our example, we use a field-oriented control (FOC) scheme.

### Field-Oriented Control Theory

FOC is a control scheme for induction motors in which a d-q coordinates reference frame locked to the motor flux space vector is used to achieve decoupling between the motor flux and torque.

Consequently, they can be separately controlled by the stator direct-axis and quadrature-axis currents, respectively.

Consider the simplified diagram in the figure that shows the stator and rotor magnetomotive forces (MMFs) $F_s$ and $F_r$ at a given instant.



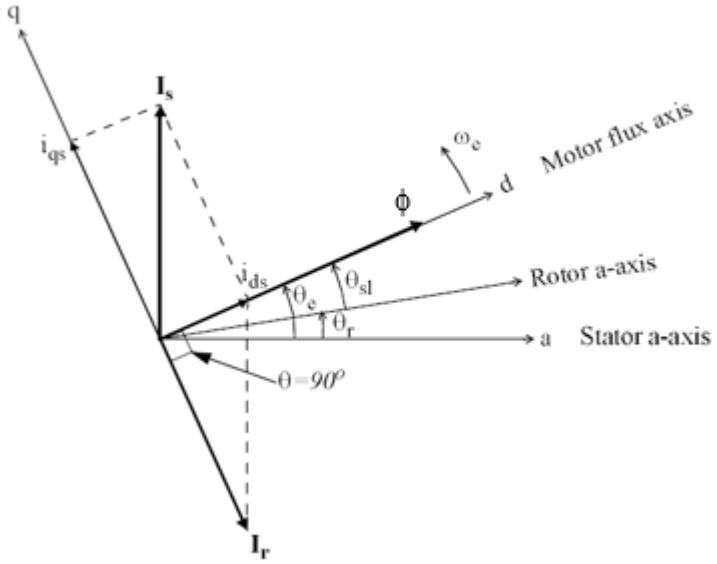

The stator and rotor three-phase windings are represented by two single-turn equivalent windings. The MMFs $F_s$ and $F_r$ rotate at a synchronous speed determined by the supply frequency. The resulting MMF $F$ (the vector sum of $F_s$ and $F_r$) produces the motor flux that rotates at the synchronous speed as shown in the figure.

The torque developed by the motor is given by:

$$T = k\, F_r\, \phi\, sin(\theta)$$

where $F_r$ is the rotor MMF, $\phi$ is the motor flux, $\theta$ is the angle between $F_r$ and $\phi$, and $k$ is a constant that depends on the motor construction. This relation indicates that the developed torque is maximum when $\theta$ is equal to 90 degrees. The challenge of the FOC control scheme is to keep this angle as close as possible to 90 degrees and to maintain the motor flux at its nominal value (except during field-weakening operation).

The figure below shows the FOC principle. The current space vectors $I_s$ and $I_r$ represent the MMFs $F_s$ and $F_r$ in the previous figure. Note that the rotor current space vector $I_r$ is perpendicular to the magnetizing current ids that produces the motor flux.




The reference coordinates are properly oriented. The stator direct-axis current ids is aligned with the motor flux and the stator quadrature-axis current $i_{qs}$ is perpendicular to the motor flux. Consequently, the motor flux and torque can be separately controlled by the stator current components $i_{ds}$ and $i_{qs}$, respectively.

The flux $\phi$ shown in the figure is the total flux produced in the air gap by the stator and rotor current space vectors $\phi = L_m(I_s + I_r)$. In the FOC theory, the rotor flux $\phi_r$ is used instead of the air gap flux $\phi$. The rotor flux $\phi_r$ is very close the air gap flux $\phi$ . The only difference is that $\phi_r$ includes a small flux, $Llr$. $I_r$ is the rotor leakage inductance. $\phi_r$ is controlled in order to stay aligned with the d-axis of the rotating frame so that $\phi_{rd} = \phi_r$ and $\phi_{rq} = 0$.

The motor flux position $\theta_e$ is required to transform the coordinates and is generated from the rotor speed $\omega_m$ and slip frequency $\omega_{sl}$.

$$\theta_e = \int p\omega_m + \omega_{sl} dt$$

where $p$ is the number of pole pairs. Computation of the slip frequency $\omega_{sl}$ and reference currents $i_{ds}ref$ and $i_{qs}ref$ requires that a model of the motor is implemented inside the controller.

$\omega_{sl}$ is evaluated from the stator reference current $i_{qs}ref$ and the motor parameters $L_m$, $Llr$, and $R_r$ as follows:

$$\omega_{sl} = \frac{L_m}{\tau_r}\frac{i_{qs}}{\phi_r} = \frac{L_m R_r}{Ll_r + L_m}\frac{i_{qs}}{\phi_r}$$

where $\phi_r$ is the rotor flux linkage, $L_m$ is the mutual inductance, and $Ll_r$ and $R_r$ are the rotor leakage inductance and resistance. The rotor time constant is:

$$\tau_r = \frac{Ll_r + L_m}{R_r}$$

The rotor flux linkage is calculated as follows:

$$\phi_r = \frac{L_m\, i_{ds}}{1 + \tau_r s}$$

The stator q-axis reference $i_{qs}ref$ is calculated from the torque reference $T_eref$ and the rotor flux linkage $\phi_r$:

$$i_{qs}ref = \frac{2}{3}\,\frac{1}{p}\,\frac{Ll_r + L_m}{L_m}\,\frac{T_eref}{\phi_r}$$

The stator d-axis reference current $i_{ds}ref$ is obtained from the flux linkage reference $\phi_rref$ :

$$i_{ds}ref = \frac{\phi_rref}{L_m}$$

### Field-Oriented Control

The motor speed $\omega_m$ and the speed reference $\omega_{ref}$ are fed to the Speed Regulator block to produce the torque reference $Te_{ref}$. The role of the speed regulator is to keep the motor speed equal to the speed reference in steady state while providing a fast dynamic response during transients.

The calculated $i_{ds}ref$ and $i_{qs}ref$ current references are fed to the current regulators. The regulators process the measured and reference currents to produce the three-phase reference signals, $Vref$. The signals in $Vref$ are connected to the PWM modulator that generates pulses to the motor inverter. The modulator uses the space vector PWM method with pulse averaging and a switching frequency of 2 kHz.

### Rectifier Controller

This controller regulates the DC-link voltage and maintains a unity input power factor. It consists of the following main blocks:

- The PLL & Measurements system is synchronized to the secondary voltage of the transformer. The secondary voltage and current are measured and transformed to d-q coordinates.
- The Current Regulator (inner control loop) consists of two PI controllers for the Id and Iq currents. The controllers' outputs are the Vd and Vq signals that are transformed to the reference signal Vref for the PWM generator. The Iq reference is kept equal to zero to achive the unity power factor. The Id reference is provided by the outer voltage control loop.
- The Voltage Regulator (outer control loop), which regulates the DC-link voltage.

The reference signal Vref is connected to the PWM generator that produces pulses to the full-bridge AC/DC rectifier. The generator uses the carrier-based PWM method with pulse averaging and a switching frequency of 2340 (39*60) Hz.

### Simulation

Run the simulation and observe waveforms on Scope blocks. To simulate an abrupt change of the road slope, at 3.5 seconds the load torque is stepped down from 400 N.m to 40 N.m and the speed reference is stepped down from 1000 rpm to 750 rpm. Because the trolley bus goes down the hill, the kinetic energy is converted into electric energy by the induction motor, which then operates as a

generator. This energy is returned to the power grid through the DC link and the single-phase converter, which then operates as an inverter. Note the active power reverses (around minus 25 kW at 3.5 s) in the P(kW), Q(kVar) trace of the Scope block.

### Real-Time Simulation

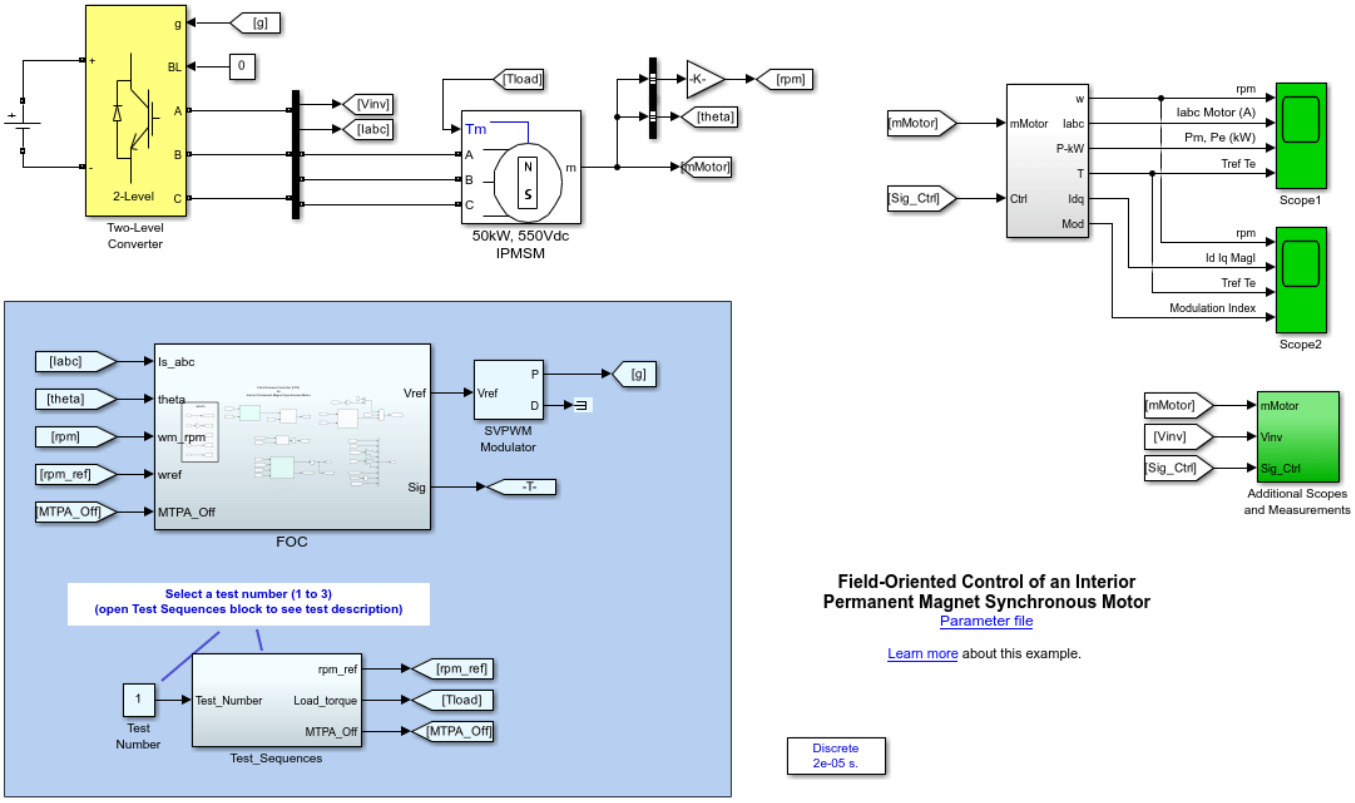
If you have Simulink Real-Time and a Speedgoat target computer, you can run this model in real time.

1. Open the Configuration Parameters window (or press **Ctrl+E** ), click **Code Generation** , and set **System target file** to `slrealtime.tlc` .
2. Connect to the target and, in the **Real-Time** tab, click **Run on Target**.

Your model will then be automatically built, deployed, and executed on the target. Depending on your target streaming bandwidth, you may have to reduce the number of signals transferred in real-time from the target to the host computer.

# Field-Oriented Control of an Interior Permanent Magnet Synchronous Motor

This example shows the wide-speed operation of an interior permanent magnet synchronous motor (IPMSM) drive. The drive uses a field-oriented control system with maximum torque per ampere (MTPA) and field-weakening control strategies.




### Description

IPMSMs are AC synchronous motors with permanent magnets embedded in their steel rotors. Compared to the surface-mounted PMSM motors, IPMSM motors are more robust and can be operated at much higher speed. In addition, an IPMSM motor shows a relatively high magnetic saliency, which allows the motor to benefit from both the magnetic and reluctance torque components.

IPMSM motors are typically controlled using field-oriented control scheme and fed with sinusoidal currents. The example also employs flux-weakening and MTPA control schemes.

### Electrical Model

A DC bus, modeled as an ideal DC source of 550 V, is connected to a three-phase, two-level converter. This converter generates the appropriate three-phase voltages (amplitude and frequency) for the speed regulation of the 50 kW IPMSM motor.

The converter is controlled by a field-oriented control (FOC) controller that generates the voltage references to a space-vector PWM modulator.

### Field-Oriented Control with Field-Weakening and MTPA

FOC is a control scheme in which a d-q coordinates reference frame that is locked to the motor flux space vector is used to achieve decoupling between the motor flux and torque. Consequently, they can be separately controlled by stator direct-axis and quadrature-axis currents, respectively.

The torque is at maximum when the flux produced by the magnets is perpendicular to the stator flux produced by the stator currents. In the field-oriented control scheme, the angle between these two fluxes is maintained at 90° to produce maximum torque.

The torque developed by the motor is given by:

$$T_e = \frac{3}{2} p \left[\lambda i_q + (L_d - L_q) i_d i_q\right]$$

where

- $p$ is the number of pole pairs.
- $\lambda$ is the flux induced by the permanent magnets in the stator windings.
- $L_d$ and $L_q$ are the d-axis and q-axis inductances.
- $i_d$ and $i_q$ are the d-axis and q-axis stator currents.

The equation is expressed in the rotor reference frame (the dq frame) and all quantities in the rotor reference frame are referred to the stator.

To operate the IPMSM motor at a speed higher than its nominal speed, the resulting back-EMF must be reduced in order not to exceed the maximum output voltage of the inverter. This reduction is performed by setting the d-axis stator current to a negative value to reduce the rotor flux linkage. This control strategy is called field-weakening control.

An IPMSM motor shows a relatively high magnetic saliency, which allows the motor to benefit from both the magnetic and reluctance torque components. The Maximum Torque Per Ampere MTPA algorithm calculates the d-axis and q-axis current components values to produce the desired torque while minimizing current magnitude. Additionally, the MTPA ensures that the inverter output does not saturate.

### Field-Oriented Control System

The main components of the FOC system are:

1. Speed Regulator — The regulator compares the actual motor speed to the speed reference. If the motor needs to be accelerated, the regulator increases the reference torque magnitude (Tref) in order to create more torque. On the contrary, if the motor speed is higher than the reference, the regulator reduces Tref. This reference torque value is then fed to the Torque Limiter block to reduce the reference torque as a function of the actual speed and the torque-speed characteristics of the motor.
2. Current measurement d-q conversion — Based on the rotor position (represented by the signal theta in the motor model), the measured three-phase stator currents are converted into their d-q coordinates in the rotor reference frame.

3. Current reference calculation — Based on the reference torque $T_{ref}$, the actual motor speed, the estimated motor parameters, and the available supply voltage, this subsystem determines the optimized reference currents $Id_{ref}$ and Iqref using the MTPA and field-weakening algorithms.
4. Current Regulators — The $Id_{ref}$ and the $Iq_{ref}$ reference currents are fed to the current regulators. The regulators process the measured and reference currents to produce the reference voltages, $V_{ref}$. The regulators' dynamics benefit from a feed-forward calculation of IPMSM currents based on the motor parameters.

The three-phase reference signals are connected to the PWM modulator that generates pulses for the motor inverter. The modulator uses the space vector PWM method with pulse averaging and a switching frequency of 8 kHz.

### Simulation

Specify a test number in the Test Number block and run the simulation. For test 3, specify a stop time of 10 s. You can observe simulation results in both Scope 1 and Scope 2.

**Test 1** : This test shows motor and generator operation in normal speed (1200 rpm) and overspeed mode (2400 rpm).

At 0.4 s, a load torque of 350 N.m. is applied to the motor. At 0.7 s, motor speed ramps to 2400 rpm and the load torque reduces to 150 N.m. The speed regulator performs well for both speed settings.

At 1.0 s, the load torque inverts from + 150 to -150 N.m so that the machine now operates as a generator.

**Test 2** : This test shows the impact of the MTPA control on the motor currents. At 0.4 s, a load torque of 350 N.m. is applied to the motor. At 0.8 s, the MTPA control turns off (Idref = 0). Without the MTPA algorithm, the magnitude of the motor currents increases while producing the same torque value. This results in more stator losses. Test 3: This test shows the wide-speed operation of the IPMSM motor from 0 rpm to 6000 rpm. The motor speed ramps to 6000 rpm while the torque reference is limited in order not to exceed motor ratings and to avoid saturation of the inverter output.

### Real-Time Simulation

If you have Simulink Real-Time and a Speedgoat target computer, you can run this model in real time.

1. Open the Configuration Parameters window (or press **Ctrl+E** ), click **Code Generation** , and set **System target file** to `slrealtime.tlc` .
2. Connect to the target and, in the **Real-Time** tab, click **Run on Target**.

Your model will then be automatically built, deployed, and executed on the target. Depending on your target streaming bandwidth, you may have to reduce the number of signals transferred in real-time from the target to the host computer.

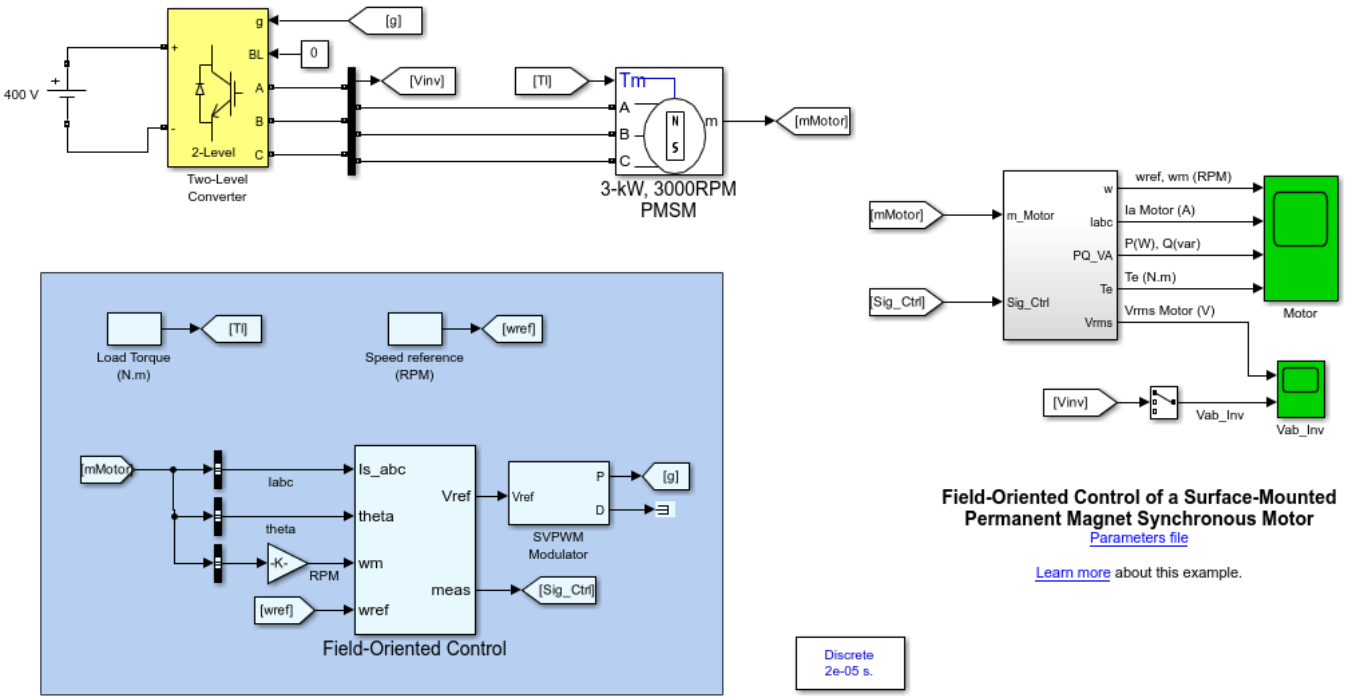
### References

1. Tremblay, Olivier. Development Report: Parameters estimation and vector control of internal permanent magnet synchronous machine, ETS December 2010.
2. Jaszczolt, Christopher. “Understanding permanent magnet motors.” Control Engineering. January 2017.

**3** Cirrincione, M., M. Pucci, G. Vitale. Power Converters and AC Electrical Drives with Linear Neural Networks. CRC Press, 2012.

# Field-Oriented Control of a Surface Mounted Permanent Magnet Synchronous Motor

This example shows the speed regulation of a surface-mounted permanent magnet synchronous motor (PMSM) using field-oriented control (FOC).




### Description

PMSMs are AC synchronous motors with permanent magnets mounted on their rotor surfaces (surfaced-mounted PMSMs) or buried into the rotor (interior-mounted PMSMs). While a BLDC motor has a trapezoidal back-EMF, PMSMs have a sinusoidal back-EMF.

PMSM motors are typically controlled using the field-oriented control scheme and fed with sinusoidal currents.

### Electrical Model

A DC bus, modeled as an ideal DC source of 400 V, is connected to a three-phase, two-level converter. This converter generates the appropriate three-phase voltages for the speed regulation of the 3 kW, 3000 rpm PMSM motor.

The converter is controlled by a FOC controller that generates the voltage references to a space-vector PWM modulator.

### Field-Oriented Control Theory

FOC is a control scheme in which a d-q coordinates reference frame that is locked to the motor flux space vector is used to achieve decoupling between the motor flux and torque. Consequently, they can be separately controlled by stator direct-axis and quadrature-axis currents, respectively.

The torque is at maximum when the flux produced by the magnets is perpendicular to the stator flux produced by the stator currents. In field-oriented control scheme, the angle between these two fluxes is maintained at 90° to produce maximum torque.

The torque developed by the motor is given by:

$$T_e = \frac{3}{2} p \left[\lambda i_q + (L_d - L_q) i_d i_q\right]$$

where

- $p$ is the number of pole pairs.
- $\lambda$ is the flux induced by the permanent magnets in the stator windings.
- $L_d$ and $L_q$ are the d-axis and q-axis inductances.
- $i_d$ and $i_q$ are the d-axis and q-axis stator currents.

Note that the equation is expressed in the rotor reference frame (dq frame) and all quantities in the rotor reference frame are referred to the stator.

Because the permanent magnets of the motor are mounted on the rotor surface, the direct and quadrature-axis inductances have the same value (Ld = Lq in our example). The equation above can then be simplified to:

$$T_e = \frac{3}{2} p \left[\lambda i_q\right]$$

This new equation shows that the direct-axis current component Id has no influence on the torque. The motor torque can then be controlled by the stator Iq component.

### Field-Oriented Control System

Based on the rotor position (represented by the signal theta in the motor model), the measured three-phase stator currents are converted into their d-q coordinates in the rotor reference frame.

The motor speed $\omega_m$ and the speed reference $\omega_{ref}$ are fed to the speed regulator to produce a current reference, $Iq_{ref}$. The role of the speed regulator is to keep the motor speed equal to the speed reference by producing more or less torque to accelerate or decelerate the motor.

The $Iq_{ref}$ current reference and the $Id_{ref}$ current reference (which are set to zero when field weakening is not required) are fed to the current regulators. The regulators process the measured and reference currents to produce the three-phase reference signals. The signals are connected to the PWM modulator that generates pulses for the motor inverter. The modulator uses the space vector PWM method with pulse averaging and a switching frequency of 8 kHz.

### Simulation

Run the simulation and observe waveforms on the scope blocks. The motor rapidely reaches its reference speed of 2000 rpm. At 0.25 s, the motor rotation is reversed by setting the speed reference

to -2800 rpm. To respect this new set point, the control system produces a large negative torque. Notice that the active power transferred back to the DC source during the deceleration. At 0.65 s, a negative load torque of -8 N.m is applied to the motor. In order to maintain the motor speed at -2800 rpm, the control system changes the reference current to produce a negative torque of about -10 N.m.

### Real-Time Simulation

If you have Simulink Real-Time and a Speedgoat target computer, you can run this model in real time.

1. Open the Configuration Parameters window (or press **Ctrl+E** ), click **Code Generation** , and set **System target file** to `slrealtime.tlc` .
2. Connect to the target and, in the **Real-Time** tab, click **Run on Target**.

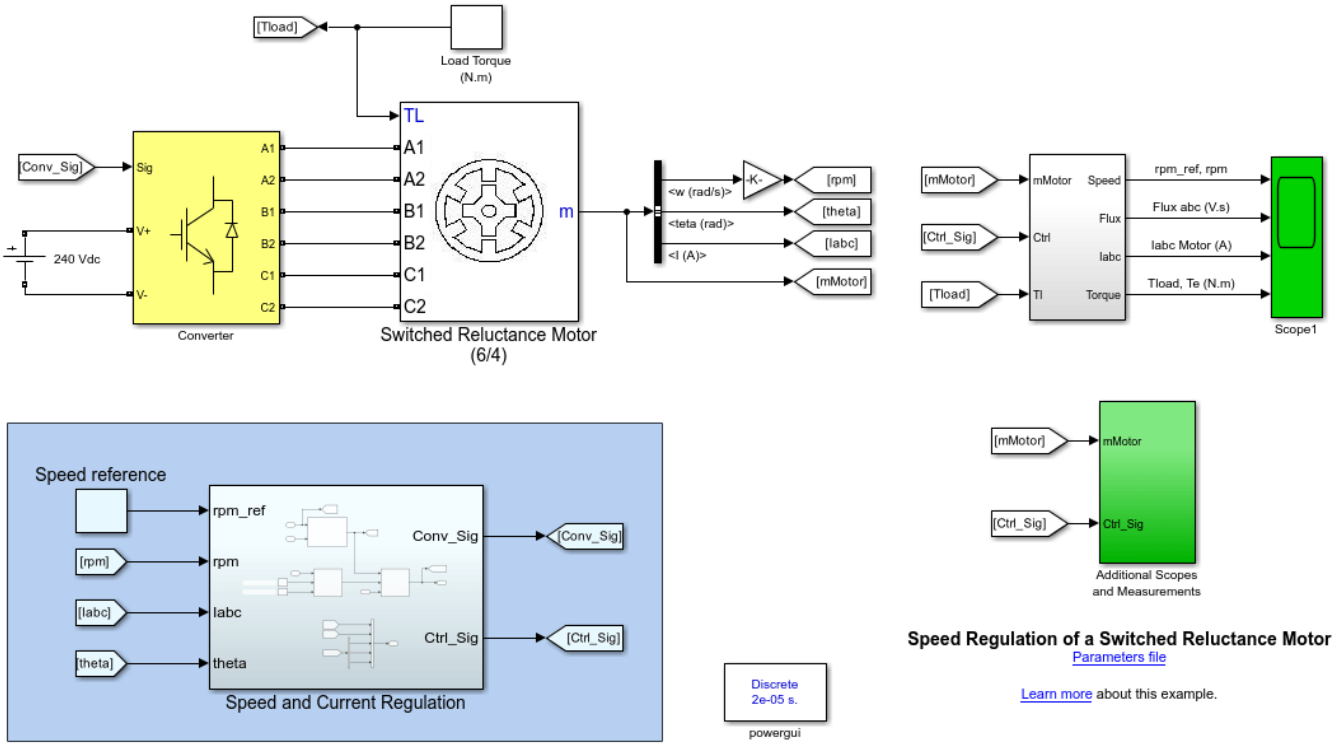
Your model will then be automatically built, deployed, and executed on the target. Depending on your target streaming bandwidth, you may have to reduce the number of signals transferred in real-time from the target to the host computer.

### References

Cirrincione, M., M. Pucci, G. Vitale. Power Converters and AC Electrical Drives with Linear Neural Networks. CRC Press, 2012.

# Speed Regulation of a 6/4 Switched Reluctance Motor

This example shows the speed regulation of a three-phase 6/4 switched reluctance motor (SRM).




### Description

Switched reluctance motors are AC motors that run by reluctance torque. Their rotors have no windings or permanent magnets. Common SRM types are three-phase 6/4 (where 6 is the number of stator poles and 4 is the number of rotor poles), four-phase 8/6, and five-phase 10/8. SRM are connected to power electronic converters that energize appropriate stator phases based on the rotor position.

### Electrical Model

A DC bus, modeled as an ideal DC source of 240 V, is connected to a power electronics converter feeding a three-phase 6/4 SRM. The converter is modeled using three single-phase, full-bridge converters. Full-bridge converters apply positive or negative voltage to the stator windings to energize or de-energize them, respectively.

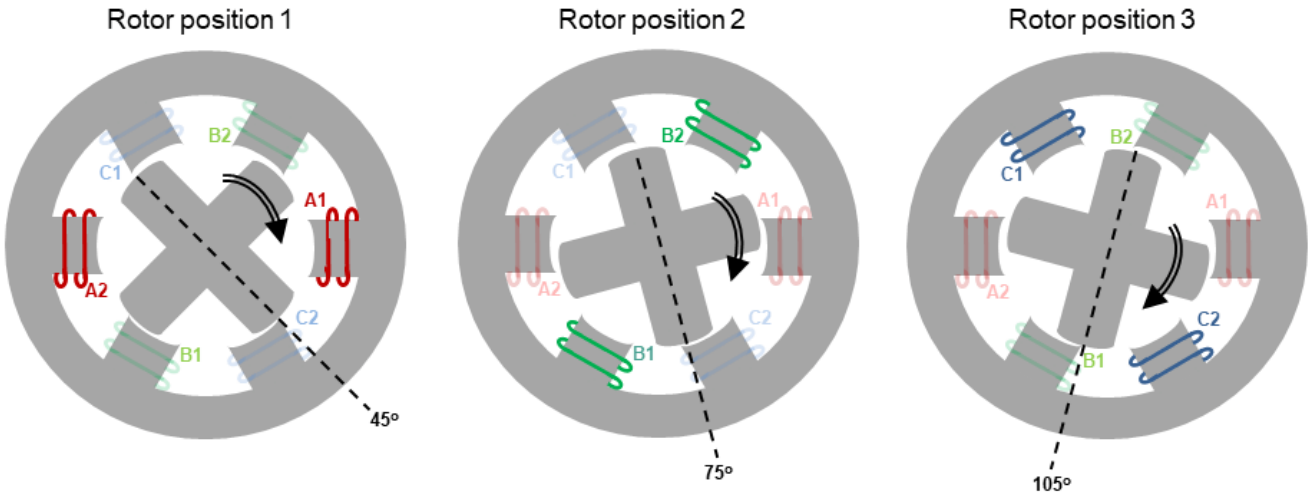
The example uses the generic model type of SRM model. The electrical part is represented by a nonlinear model based on the magnetization characteristic composed of several magnetizing curves and on the torque characteristic computed from the magnetization curves. The magnetization characteristic is calculated using nonlinear functions and the specified motor parameters. You can visualize the magnetization curves (including the linkage flux as a function of stator currents and rotor position) by checking the **Plot magnetization curves** parameter of the Switched Reluctance Motor block and click Apply.

### Control System

The main components of the SRM control system are:

1. Speed Regulator — The regulator compares the actual motor speed to the speed reference. If the motor needs to be accelerated, the regulator increases the current reference (Iref) to create more torque. On the contrary, if the motor speed is higher than the reference, the regulator reduces Iref.
2. Commutation Logic block — Based on the rotor position (represented by the theta signal from the SRM model) and the turn-on and turn-off angles, this block generates the control signals to produce the appropriate commutation sequence for torque production.
3. Current Regulator — Based on the desired reference current, Iref, and the commutation logic signals, a current reference is created for each of the three phases. Each current reference is then compared with the corresponding measured stator current. When the resulting error crosses the positive hysteresis band value, a conduction order is sent to the appropriate full-bridge converter. The converter then applies a positive voltage to the stator winding in order to drive a positive current into the winding. During the free-wheeling period (when there is no pulse), a negative voltage is applied to the windings and the stored energy is returned to the power DC source through the diodes.

The next figure illustrates the commutation sequence for three rotor positions.




In rotor position 1, a positive voltage is applied to windings A1 and A2. The resulting magnetic field creates a reluctance torque that forces the rotor pole to align with the newly energized stator poles.

When the motor reaches rotor position 2, the A1 and A2 windings are de-energized and windings B1 and B2 are energized to keep the rotor turning clockwise. This action keeps the rotor turning clockwise, because the rotor tries to align with B1 and B2 windings.

Finally, when the motor reaches position 3, the B1 and B2 windings are de-energized and windings C1 and C2 are energized. Because the rotor has four poles, this sequence is repeated every 90 degrees.

In this example, the turn-on and turn-off angles (relative to stator windings A1 and A2) are kept constant at 45 degrees and 75 degrees, respectively. The rotor angles when phases A, B, and C are energized are then respectively 45, 75, and 105 degrees with respect to phase A axis.

### Simulation

Run the simulation and observe waveforms on the block named Scope1. The motor moves from zero speed to 1500 rpm with a load torque of 15 N.m. At 0.15 s, the load torque is increased to 75 N.m. The control system increases the reference current in order to maintain the motor speed at 1500 rpm. At 0.3 s, the speed reference is stepped to 2500 rpm. To reach the desired speed, the control system momentarily produces a large torque by increasing the motor currents.

### Real-Time Simulation

If you have Simulink Real-Time and a Speedgoat target computer, you can run this model in real time.

1. Open the Configuration Parameters window (or press **Ctrl+E** ), click **Code Generation** , and set **System target file** to `slrealtime.tlc` .
2. Connect to the target and, in the **Real-Time** tab, click **Run on Target**.

Your model will then be automatically built, deployed, and executed on the target. Depending on your target streaming bandwidth, you may have to reduce the number of signals transferred in real-time from the target to the host computer.

### Reference

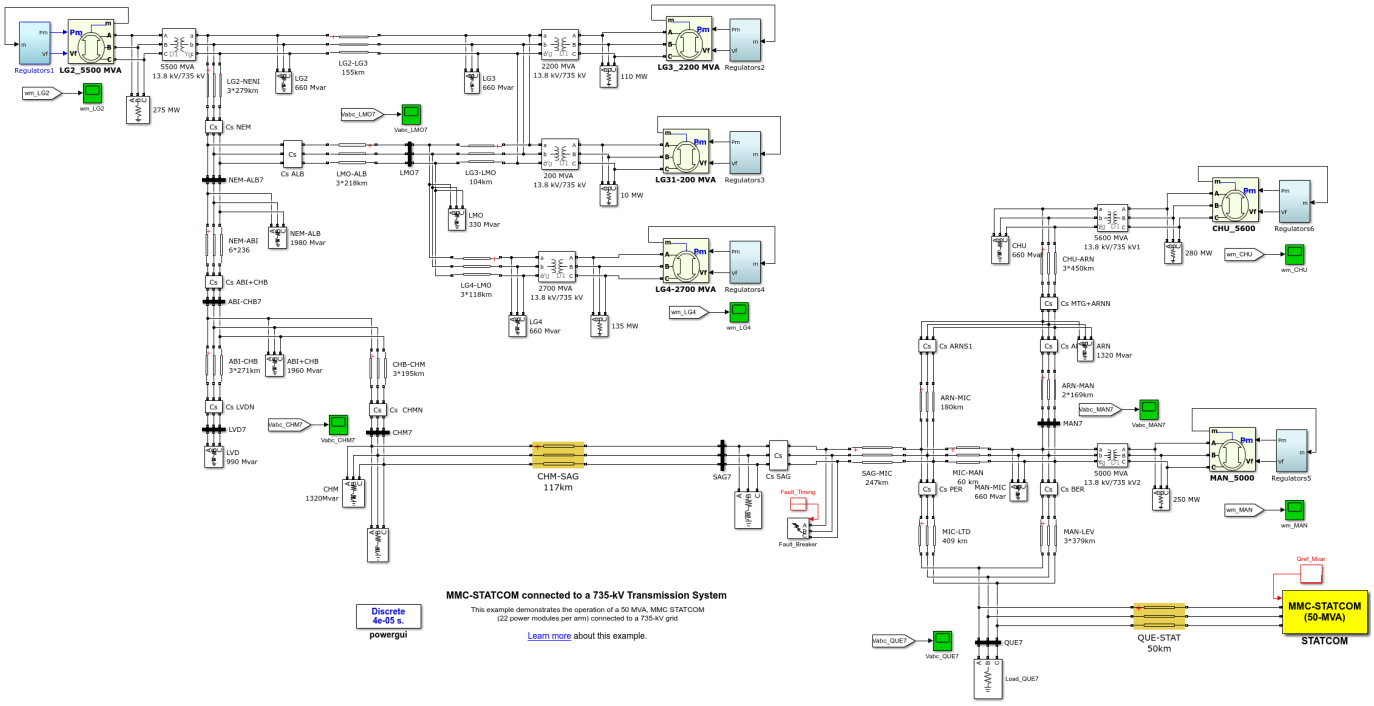
Knight, Andy. Electrical Machines – Switched Reluctance Motors. University of Calgary.

### External Website

Wikipedia. "Switched reluctance motor." Accessed October 28,2020.

# MMC-STATCOM Connected to a 735-kV Transmission System

This example demonstrates the operation of a 50 MVA, MMC STATCOM (22 power modules per arm) connected to a 735 kV grid.




## Description

In electrical grids, shunt compensation is often used for reactive power and voltage control. This example models a shunt compensation device that is increasingly used in modern grids: the Modular-Multi-level (MMC) STATCOM. The MMC-STATCOM is built using the Full-Bridge MMC block to represent a power electronics converter of 22 power modules (PM) per arm. The three MMC arms are delta-connected to an 18kV/735kV power transformer.

## Principles of Operation

The STATCOM can absorb or generate reactive power. The reactive power transfer is done through the phase reactance. The converter generates a voltage in phase with the grid voltage. When the amplitude of the converter voltage is lower than that of the grid voltage, the STATCOM acts like an inductance and absorbs reactive power. When the amplitude of the converter voltage is higher than that of the bus voltage, the STATCOM acts like a capacitor and generates reactive power.

## Transmission System

The 60 Hz power system is a simplified model of a series-compensated 735 kV transmission system. The grid is both series- and shunt-compensated using fixed capacitors and inductors. It comprises the following main components: - Thirty-two three-phase buses - Six power plants. Each power plant is represented by a synchronous machine, an hydraulic turbine and governor, an excitation system, and a power system stabilizer (PSS) - Six three-phase power transformers - Twenty three-phase shunts and loads - Seventeen 735 kV distributed parameters lines. Eleven lines are series-compensated by

capacitors in order to increase the transmission capability (compensation factor = 15% to 42% of the line reactance). - 50 MVA STATCOM with twenty-two cascaded full-bridge converters per arm

### Control System

The STATCOM control system consists of the following main subsystems: - Measurements computes the d-axis and q-axis components of the primary voltages and currents by performing an abc-dq transformation in the synchronous reference provided by a three-phase PLL synchronized on the primary voltages. - Reactive Power Regulator controls the reactive power absorbed or generated by the STATCOM by determining the required reactive current reference value (Iq_ref). - DC Voltage Regulator maintains the average value for all capacitors PM to the specified reference value (Vnom_PM = 1600V). To perform this task, the regulator controls the active power absorbed or generated by the STATCOM by determining the required active current reference value (Id_ref). - Current Regulator generates the converter output voltage signals (Vd,Vq) in order to have measured current values (Id/Iq) equal to the reference values (Id_ref/Iq_ref). - Insertion Indices Computation determines the insertion indices (n) using the nearest-level modulation technique. The indices are calculated based on the current regulator outputs (Vd,Vq), and the number and nominal voltage of the capacitors power modules. - Power Modules Selection selects which power modules will be inserted or removed based on the requested insertion indices and on a voltage balancing algorithm. The voltage balancing algorithm is based on sorting the capacitor voltages and on measuring the polarity of the current flowing into the arm. Double-click on the subsystem to see how the algorithm is implemented.

### Simulation

Run the simulation and observe the waveforms on Scope1. You can see that the simulation starts in steady state and that the STATCOM operates at 0 Mvar. At 0.1 seconds, the set-point Qref value is changed to -30 Mvar (inductive var). At 0.3 seconds, the set-point changes from -30 Mvar to 40 Mvar (capacitive var). You can see that the STATCOM control system reacts very rapidly to modify the inverter output voltage in order to follow the new Qref set-point. You can also program the Fault_Timing block in order to produce a six-cycle, three-phase fault at the SAG7 substation and observe the STATCOM operation.

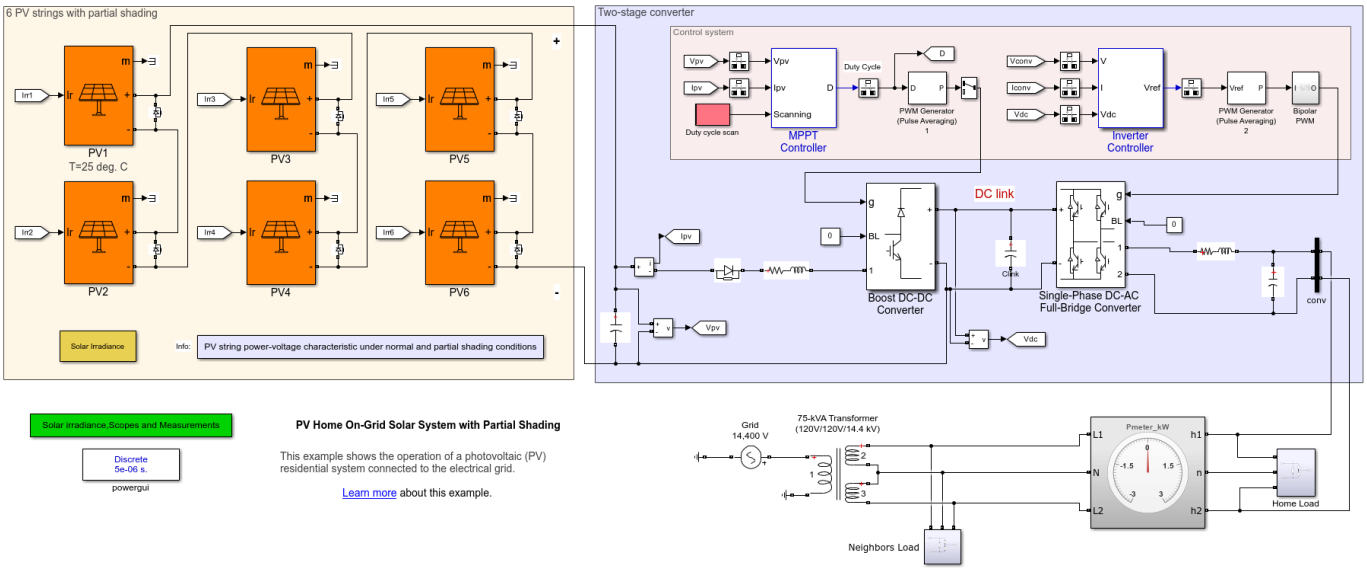
### References

[1] M. Pereira, Member, IEEE, D. Retzmann, Member, IEEE, J. Lottes, M. Wiesinger, G. Wong, SVC PLUS: An MMC STATCOM for Network and Grid Access Applications 2011 IEEE Trondheim PowerTech

[2] Mattia Ricco, Laszlo Mathe, Manel Hammami, Francesco Lo Franco, Claudio Rossi and Remus Teodorescu, A Capacitor Voltage Balancing Approach Based on Mapping Strategy for MMC Applications Electronics Open Access Journal, April 2019

# PV Home On-Grid Solar System

This example shows the operation of a photovoltaic (PV) residential system connected to the electrical grid.




### PV Strings

The PV strings section implements a home installation of six PV array blocks in series that can produce 2400 W of power at a solar irradiance of 1000 W/m2. In the Advanced tab of the PV blocks, the robust discrete model method is selected, and a fixed operating temperature is set to 25 degrees C.

### Two-Stage Converter

The power produced by the PV strings is fed to the house and utility grid using a two-stage converter: a boost DC-DC converter and a single-phase DC-AC full-bridge converter. Both converters are PWM-controlled with a switching frequency of 20 kHz. They are using the Switching Function method with PWM pulse averaging, allowing a simulation sample time of 5 microseconds and good accuracy on harmonics generated.

### Control Systems

- The MPPT Controller: The Maximum Power Point Tracking (MPPT) controller is based on the Perturb and Observe technique with scanning capability. The MPPT system automatically varies the duty cycle of the boost converter to generate the required voltage across the PV string in order to extract maximum power. Under partial shading condition, a duty cycle is initiated to find the global maximum power point (GMPP) among various local maximum power points (LMPP).
- The Inverter Controller: The inverter control maintain the DC link voltage at 400 V while keeping a unity power factor. The controller uses a voltage regulator outer loop and a fast inner loop current regulator to generate the appropriate reference voltage (Vref) for the PWM generator controlling the full-bridge converter.

### Load & Utility Grid

The grid is modeled using a typical pole-mounted transformer and an ideal AC source of 14.4 kVrms. The transformer 240 volt secondary winding is center-tapped and the central neutral wire is grounded. The inverter, the 2500 W residential load as well as the neighbors' load are connected to the 240V secondary winding.

### Simulation

Run the simulation and observe the resulting signals on the various scopes.

(1) At 0.25s, with a solar irradiance of 1000 W/m2 on all PV modules, steady state is reached. The solar system generates 2400 Watts and the DC link is maintained at 400 volts with a small 120-Hz ripple due to the single-phase power extracted from the PV string. The Utility meter indicates that the system takes almost no power from the grid to supply the home total load. (2) At 0.3s, a partial shading condition is created by reducing the irradiance on some PV modules. When steady-state is reached at 0.35s, the MPPT controller has set the boost duty cycle at 0.44, generating a PV string voltage of 225 V. With this voltage, 920 W is extracted from the PV string. As you can see on the PV curve characteristic, the system is operating at a local maximum power point but not at the global maximum power point. (3) At 0.4s, a duty cycle scan of 0.25 seconds is performed by the MPPT controller to find the GMPP point. (4) At 0.7s, the MPPT controller has set the boost duty cycle at 0.58 generating a PV string voltage of 168 V. With this voltage, 1364 W is extracted from the PV string which is the GMPP value. The Utility meter indicates that it takes now around 1100 W (2500 W residential load - 1364 W supplied by PV) from the grid to supply the home total load.

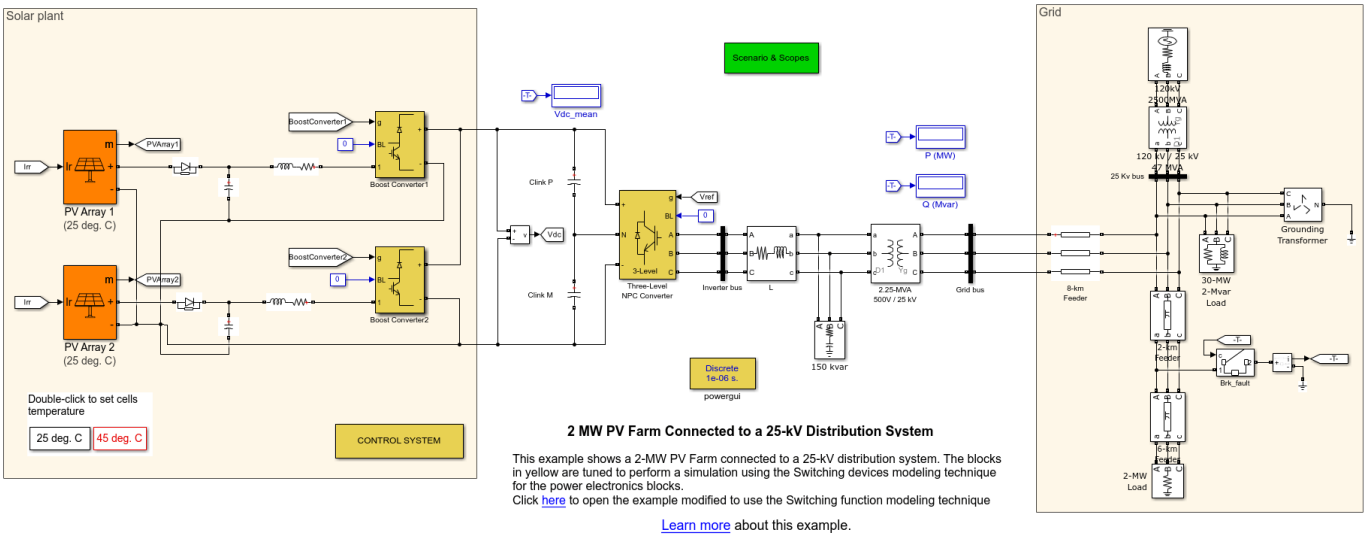
### References

1. https://en.wikipedia.org/wiki/Photovoltaic_system

2. https://www.lg.com/us/business/download/resources/BT00002151/LG400N2W-A5.pdf

# 2-MW PV Farm Connected to a 25-kV Distribution System

This example shows a model of a 2-MW PV farm connected to a 25-kV distribution system.




## Description

The PV farm consists of two PV arrays: PV Array 1 and PV Array 2 can produce respectively 1.5 MW and 500 kW at 1000 W/m2 sun irradiance and at cell temperature of 25 degrees C. Each PV array is connected to a boost converter. Each boost is individually controlled by a Maximum Power Point Trackers (MPPT) system. The MPPTs use the Perturb and Observe technique to vary the voltage across the terminals of the PV array in order to extract the maximum possible power. The outputs of the boost converters are connected to a common DC bus of 1000 V. A three-level NPC converter converts the 1000 V DC to around 500 V AC. The NPC converter is controlled by a DC voltage regulator whose job is to maintain the DC link voltage to 1000V whatever the amount of active power delivered by the PV arrays. In addition, the controller has a reactive power regulator allowing the converter to generate or absorb up to 1 Mvar. A 2.25-MVA 500V/25kV three-phase coupling transformer is used to connect the converter to the grid. The grid model consists of typical 25-kV distribution feeders and a 120-kV equivalent transmission system.

## Simulation

Start the simulation and see the resulting signals on the various scopes.

In the Scenario & Scopes subsystem you can program four various disturbances: 1) Irradiance variation 2) DC link reference voltage step 3) Reactive power set-point variation 4) System fault.

You can simulate the model with the PV cells temperature set to 45 degrees C or to 25 degrees C by double-clicking on the corresponding blocks below the PV Arrays blocks.

The blocks in yellow in the model are tuned to perform a simulation using the Switching devices modeling technique for the power electronics blocks. You can refer to the Please refer to the documentation pages of the Boost Converter and Three-Level NPC Converter blocks for more information on the available modeling techniques of the power electronics blocks.

The same example is simulated using an average model for the boost converters and a switching function model for the three-level NPC converter. This allows to run the model with a larger sample time resulting in a much faster simulation while obtaining almost identical results.

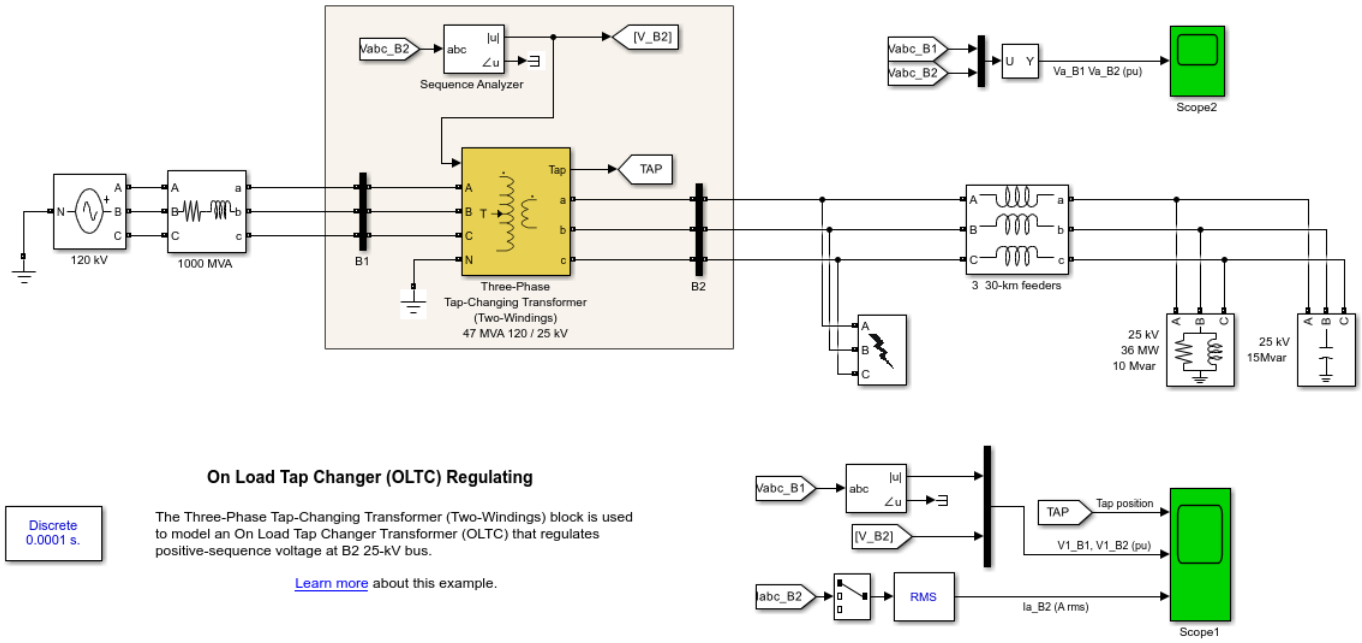
### References

1. Horowitz, Kelsey, Zac Peterson, Michael Coddington, Fei Ding, Ben Sigrin, Danish Saleem, Sara E. Baldwin, et al. 2019. An Overview of Distributed Energy Resource (DER) Interconnection: Current Practices and Emerging Solutions. Golden, CO: National Renewable Energy Laboratory. NREL/TP-6A20-72102

2. Haidar Islam, Saad Mekhilef, Noraisyah Binti,Mohamed Shah, Tey Kok Soon,Mehdi Seyedmahmousian, Ben Horan and Alex Stojcevski. Performance Evaluation of Maximum Power Point Tracking Approaches and Photovoltaic Systems. Energies 2018, 11, 365; doi:10.3390/en11020365.

# On Load Tap Changer (OLTC) Regulating Transformer

This example shows a tap-changing transformer that regulates positive sequence voltage at a 25 kV bus




### Description

The Three-Phase Tap-Changing Transformer (Two-Windings) block is used to model an On Load Tap Changer Transformer (OLTC) that regulates positive-sequence voltage at B2 25-kV bus. The reference voltage is set at 1.0 pu.

### Simulation

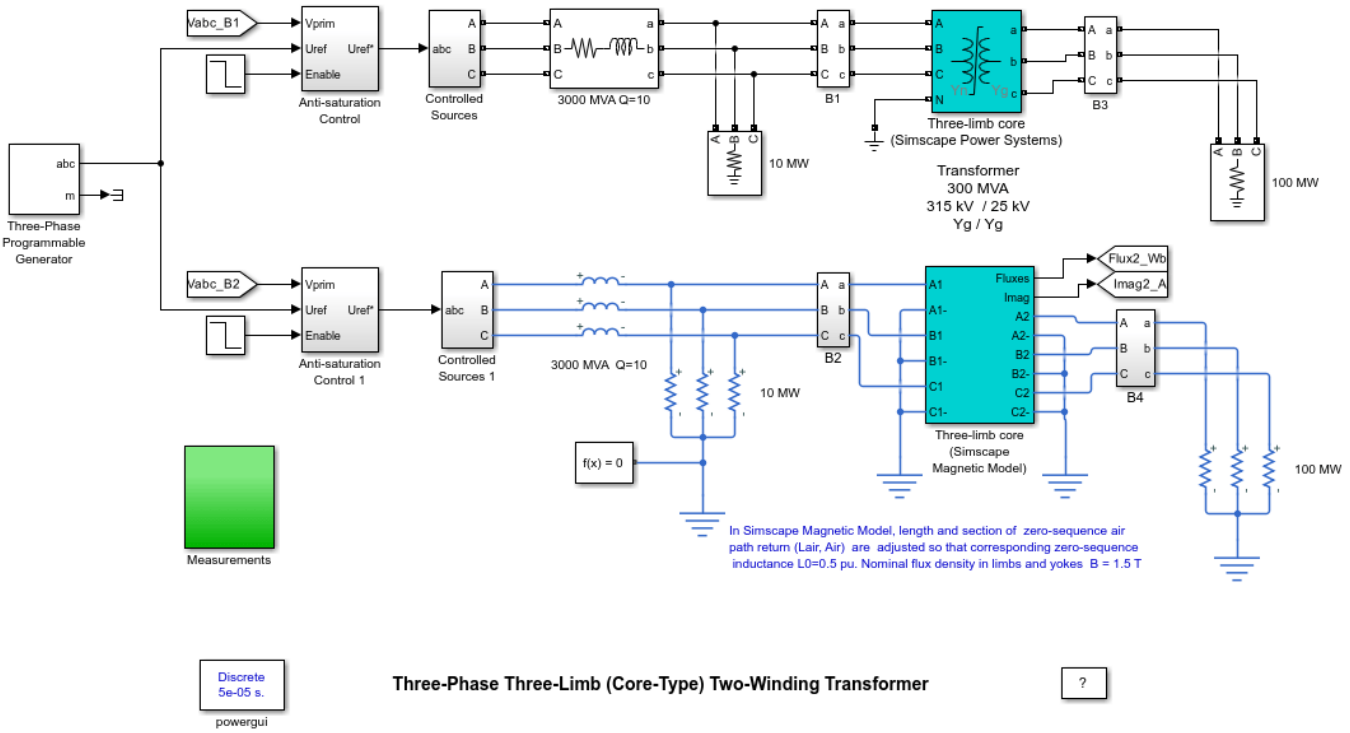
Observe OLTC reaction for the following 3 events:

- 1.1 pu over-voltage in 120 kV network at t= 2 sec.
- 0.95 pu under-voltage in 120 kV network at t= 10 sec.
- 0.2 sec single-phase fault at B2 bus at t= 20 sec. OLTC does not react because under-voltage duration is lower than specified Delay (0.4 sec)

Note: In order to reduce simulation time, the tap selection time (usually a value between 3 sec and 10 sec) has been shortened to 0.5 sec.

# Three-Phase Three-Limb (Core-Type) Two-Winding Transformer

This example illustrates saturation in a Three-Limb Core-Type transformer and compares the Simscape™ Electrical™ Specialized Power Systems model with a Simscape-based physical model.




## Description

The top circuit uses Simscape Electrical Specialized Power Systems blocks to implement a 300 MVA, 315 kV/25 kV, Yg/Yg transformer connected to a 315 kV grid. During asymmetrical voltage conditions, the zero-sequence flux returns outside the core, through an air gap, structural steel and tank. Thus, the natural zero-sequence inductance L0 (without Delta winding) of such a core-type transformer is usually very low (typically 0.5 pu < L0 < 2 pu) compared to a three-phase transformer using three single-phase units (L0 > 100 pu) or with a five-limb shell-type transformer. This low L0 value will affect voltage, current and flux imbalances both during linear and saturated operation.

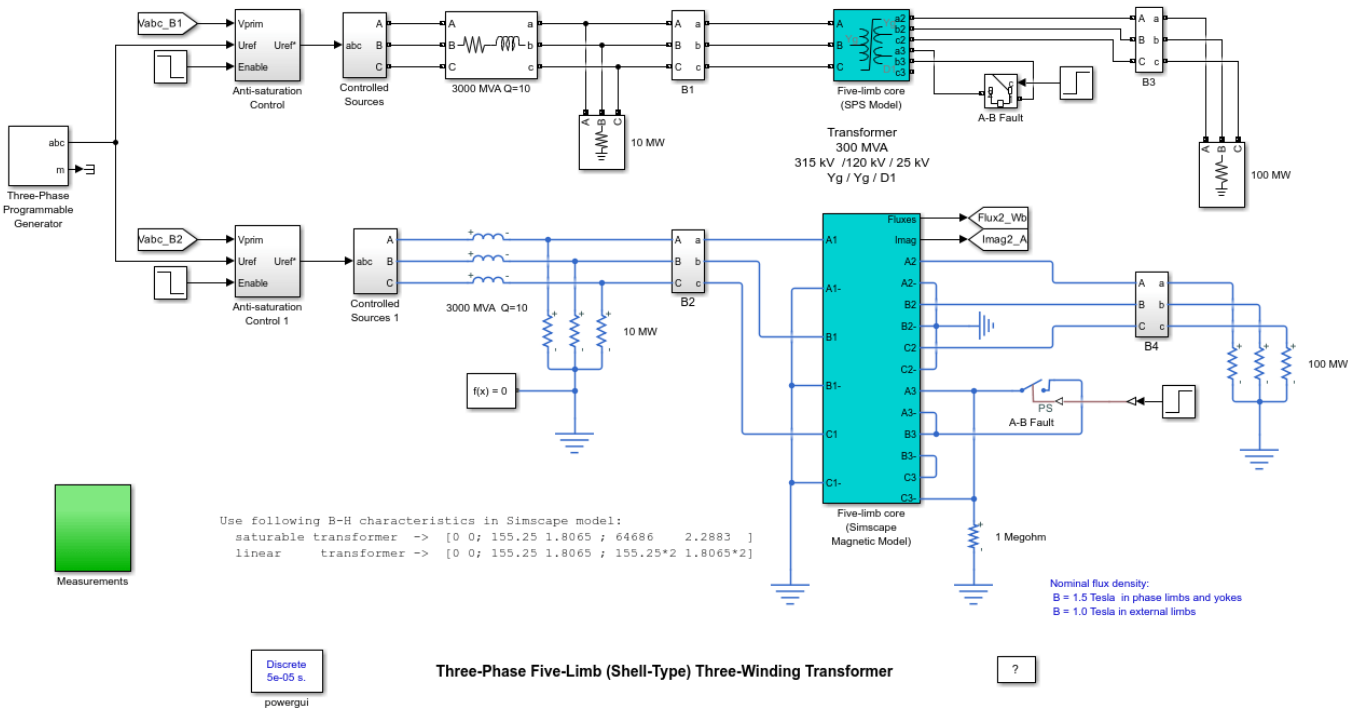
The bottom circuit implements the same circuit using the Simscape electrical and magnetic foundation libraries. Look under the mask of the Simscape transformer to see how the magnetic circuit is implemented. Blue blocks represent steel reluctances of the 3 limbs and 4 yokes with specified nonlinear B-H characteristic. Yellow blocks represent reluctances of air paths modeling the winding leakage inductances. The orange block models reluctance of the zero-sequence flux path through the air. Its average length and section are adjusted in order to get a 0.5 pu zero-sequence inductance (as specified in Specialized Power Systems model).

The Three-Phase Programmable Generator block is used to simulate a 1.7 pu overvoltage on phase A (from 0.2 sec to 0.3 sec), producing saturation of phase A. Transformers in both models start with zero initial fluxes. In order to decrease the time required to reach steady-state, Anti-saturation Control blocks are used to quickly force the average fluxes in the three phases to zero.

Observe that both models produce the same voltage, current, and flux waveforms.

# Three-Phase Five-Limb (Shell-Type) Three-Winding Transformer

This example illustrates saturation in a Five-Limb Shell-Type transformer and compares the Specialized Power Systems model with a Simscape-based physical model.




## Description

The top circuit uses SimPowerSytems blocks to implement a 300 MVA, 315 kV/120 kV/25 kV, Yg/Yg/D transformer connected to a 315 kV grid. During unbalanced voltage conditions, contrary to the three-limb transformer, the zero-sequence flux of the five-limb transformer stays inside the steel core and returns through the two external limbs. The natural zero-sequence inductance (without Delta) is very high (L0 > 100 pu). Therefore, except for small current imbalances due to core asymmetry, the behavior of the five-limb shell-type transformer is very similar to that of a three-phase transformer built with three single-phase units.

The bottom circuit implements the same circuit using the Simscape electrical and magnetic foundation libraries. Look under the mask of the Simscape transformer block to see how the magnetic circuit is implemented. Blue blocks represent steel reluctances of the 5 limbs and 8 yokes with specified nonlinear B-H characteristic. Yellow blocks represent reluctances of air paths modeling the winding leakage inductances.

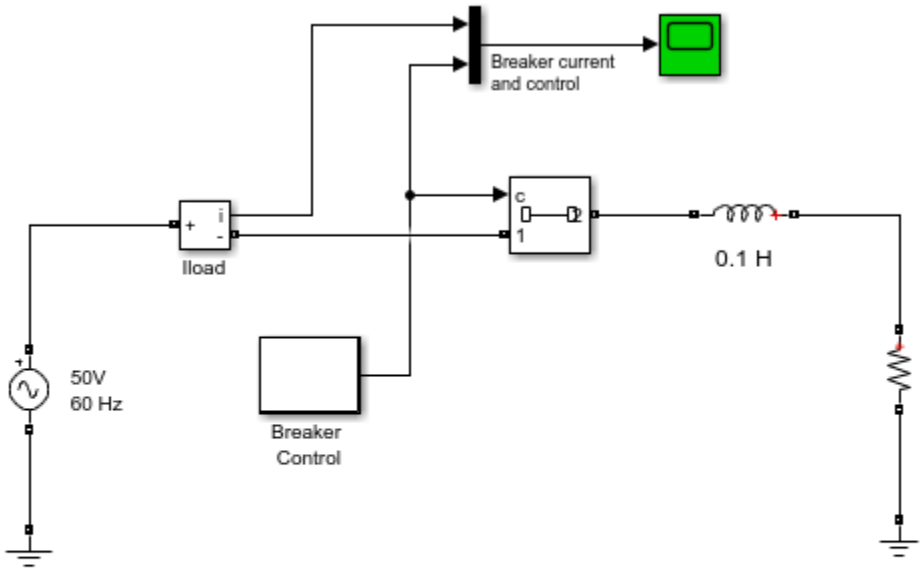
The Three-Phase Programmable Generator block is used to simulate a 1.7 pu overvoltage on phase A (from 0.2 sec to 0.3 sec), producing saturation of phase A. Then, a phase-phase (A-B) fault is applied across the Delta winding at t=0.35 sec. Transformers in both models start with zero initial fluxes. In order to decrease the time required to reach steady-state, Anti-saturation Control blocks are used to quickly force the average fluxes in the three phases to zero.

Observe that voltages, currents, and fluxes produced by the two models are very similar.

# Switching an Inductive Circuit Using a Breaker With No Snubber

This example shows the Ideal Switching device solution method of the Powergui block to model a circuit breaker.

G. Sybille (Hydro-Quebec)



### Description

The use the Ideal Switching device solution method allow to simulate a circuit breaker without snubber and connected in series with an inductive circuit.

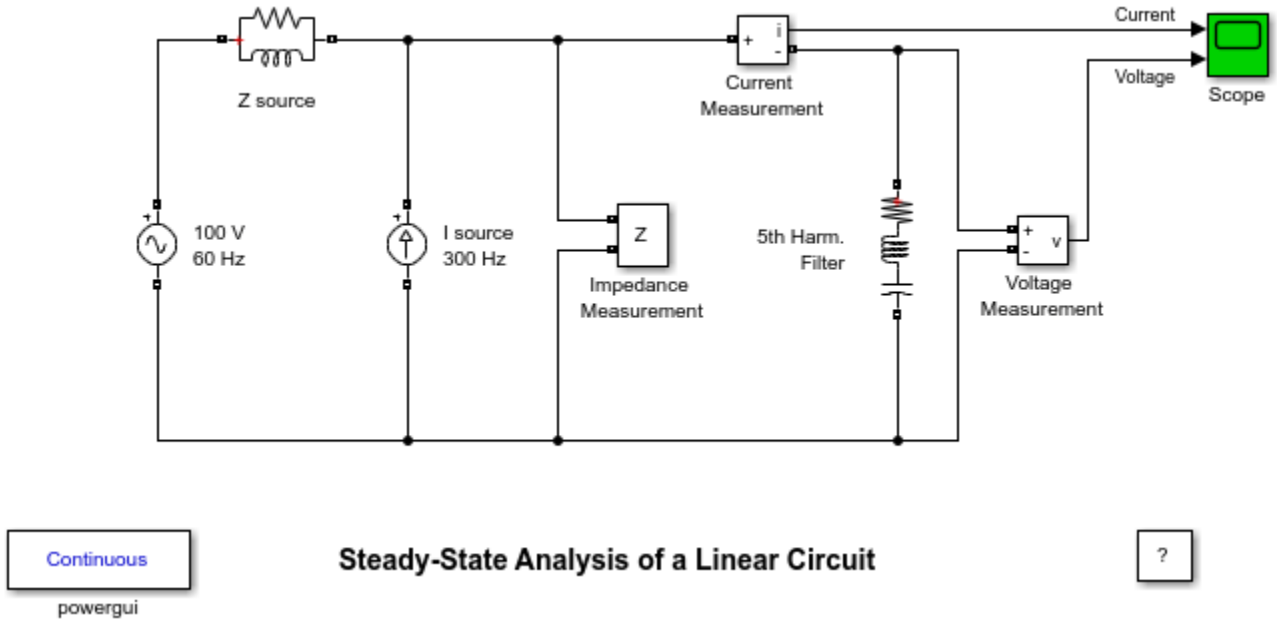
Zoom on breaker current and note the "exact" zero current when the breaker is open.

This solution technique introduced in R2008b is available with continuous solvers. To enable this option, check the "Enable use of ideal switching devices" parameter of the powergui block.

# Steady-State Analysis of a Linear Circuit

This example shows the use of the Powergui and Impedance Measurement blocks to analyze the steady-state operation of a linear electrical circuit

G. Sybille (Hydro-Quebec)




### Description

A 5th harmonic filter is connected at a bus bar fed by a 60 Hz, 100 V inductive source. A 5th harmonic (300 Hz, 1 A) current is injected at the bus bar.

This linear system consists of 3 states (2 inductor currents and 1 capacitor voltage), 2 inputs (Vs, Is) and 2 outputs (Current and Voltage Measurement).

An Impedance Measurement block is used to compute the impedance versus frequency of the circuit.

### Simulation

1. Use the Powergui block to find the steady-state 60Hz and 300 Hz components of voltage and current phasors. The values of the 3 states (phasors and initial values) can be also obtained from the powergui block.

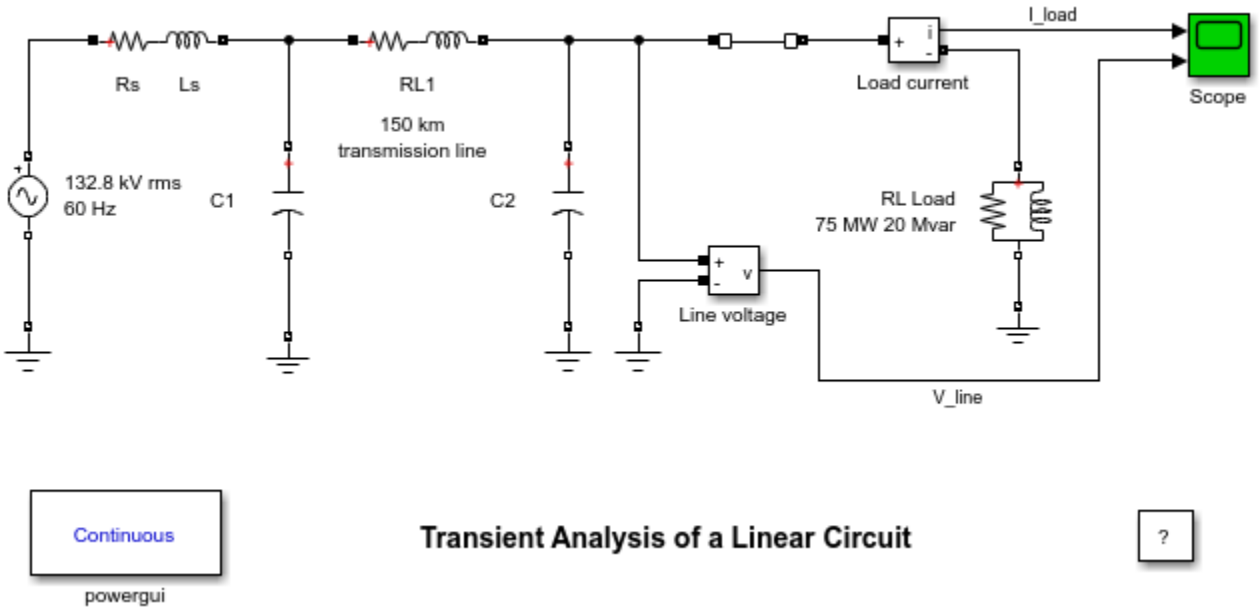
2. Open the scope and start the simulation from the Simulation/Start menu. Notice that the simulation starts in steady-state. Using the Powergui block, select Impedance vs Frequency Measurement. A new window opens.

3. The measurement will be performed for the specified frequency range vector [0: 2:1000] (0 to 1000 Hz by steps of 2 Hz). Click on the Display button. The impedance is displayed in a graphic window. Notice the series resonance at 300 Hz corresponding to the tuned frequency of the filter.

# Transient Analysis of a Linear Circuit

This example shows steady-state and transient simulation of a linear circuit.

H. Le-Huy (Universite Laval, Quebec) and G. Sybille (Hydro-Quebec)




### Description

This circuit is a simplified model of a 230 kV three-phase power system. Only one phase of the transmission system is represented. The equivalent source is modeled by a voltage source (230 kV rms/sqrt(3) or 187.8 kV peak, 60 Hz) in series with its internal impedance (Rs Ls) corresponding to a 3-phase 2000 MVA short circuit level and X/R = 10. (X = 230e3^2/2000e6 = 26.45 ohms or L = 0.0702 H, R = X/10 = 2.645 ohms). The source feeds a RL load through a 150 km transmission line. The line distributed parameters (R = 0.035ohm/km, L = 0.92 mH/km, C = 12.9 nF/km) are modeled by a single pi section (RL1 branch 5.2 ohm; 138 mH and two shunt capacitances C1 and C2 of 0.967 uF). The load (75 MW - 20 Mvar per phase) is modeled by a parallel RLC load block.

A circuit breaker is used to switch the load at the receiving end of the transmission line. The breaker which is initially closed is opened at t = 2 cycles, then it is reclosed at t = 7 cycles. Current and Voltage Measurement blocks provide signals for visualization purpose.

### Simulation

#### 1. Simulation using a continuous solver (ode23tb)

Start the simulation and observe line voltage and load current transients during load switching and note that the simulation starts in steady-state. Use the zoom buttons of the oscilloscope to observe the transient voltage at breaker reclosing.

#### 2. Using the Powergui to obtain steady-state phasors and set initial states

Open the Powergui block and select "Steady State Voltage and Currents" to measure the steady-state voltage and current phasors. Using the Powergui select now "Initial States Setting" to obtain the initial state values (voltage across capacitors and current in inductances). Now, reset all the initial states to zero by clicking the "to zero" button and then "Apply" to confirm changes. Restart the simulation and observe transients at simulation starting. Using the same Powergui window, you can also set selected states to specific values.

**3. Discretizing your circuit and simulating at fixed steps**

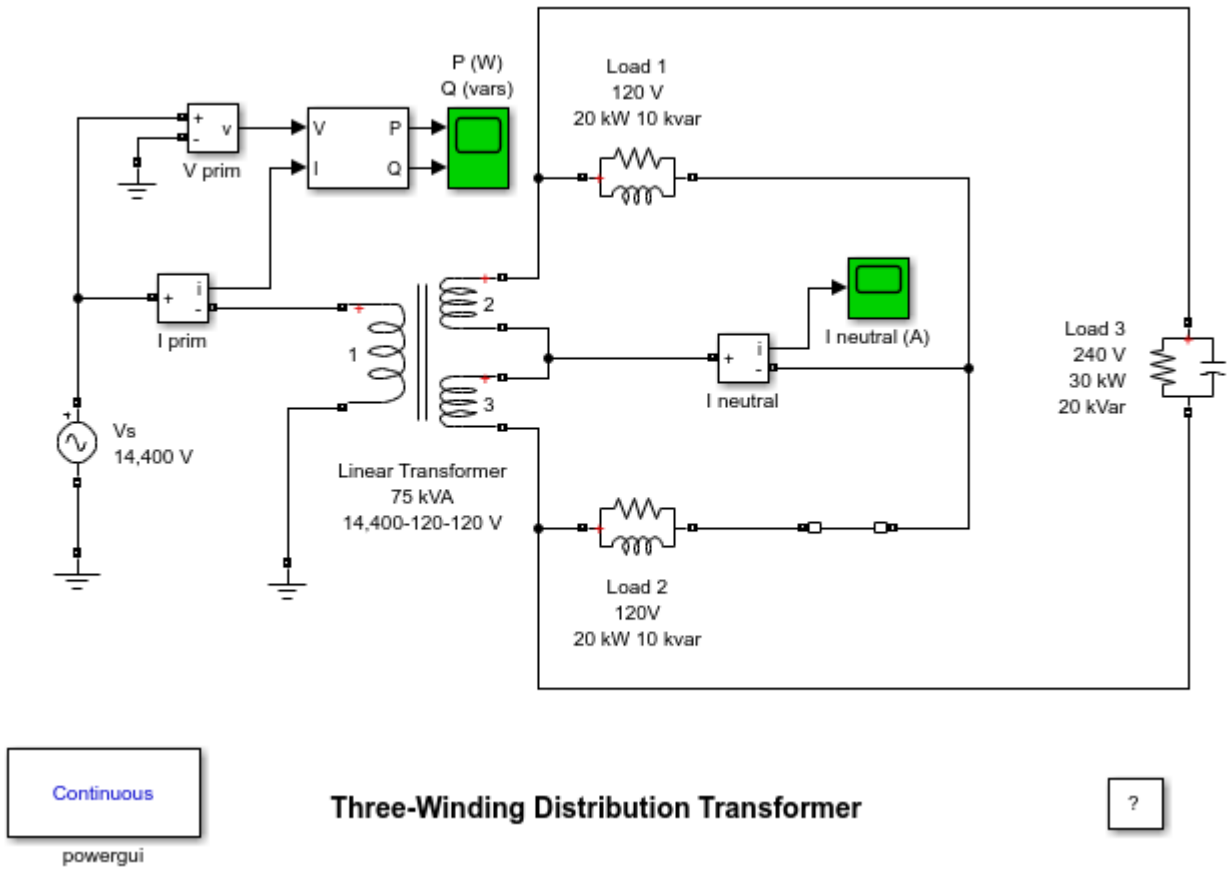
The Powergui block can also be used to discretize your circuit and simulate it at fixed steps. Open the Powergui. Select "Discretize electrical model" and specify a sample time of 50e-6 s. The state-space model will now be discretized using trapezoidal fixed step integration. The precision of results is now imposed by the sample time. Restart the simulation and compare simulation results with the continuous integration method. Vary the sample time of the discrete system and note the impact on precision of fast transients.

**4. Using the phasor simulation method**

You will now use a third simulation technique. The "phasor simulation" method consists to replace the circuit state-space model by a set of algebraic equations evaluated at a fixed frequency and to replace sinusoidal voltage and current sources by phasors (complex numbers). This method allows a fast computation of voltage and current phasors at a selected frequency, disregarding fast transients. It is particularly efficient to study electromechanical transients of generators and motors involving low frequency oscillation modes. Open the Powergui block and select "Phasor simulation". Restart the simulation. Observe that the magnitude of 60 Hz voltage and current is now displayed on the scope. If you double click on the voltage or current measurement block you can choose to output phasor signals in four different formats: Complex, Real/Imag, Magnitude/Angle (in degrees), or just Magnitude (default value). Notice that you cannot send a complex signal to an oscilloscope.

# Three-Winding Distribution Transformer

This example shows the use of the linear transformer to simulate a three-winding distribution transformer rated 75 kVA - 14400/120/120 V.




### Description

The transformer primary is connected to a high voltage source (14,400 V rms). Two identical inductive loads (20 kW -10 kvar) are connected to the two secondaries. A third capacitive load (30 kW -20 kvar) is fed at 240 V.

Initially, the circuit breaker in series with Load 2 is closed, so that the system is balanced.

### Simulation

Open the powergui block to obtain the initial voltage and current phasors in steady state. As loads are balanced the neutral current is practically zero.

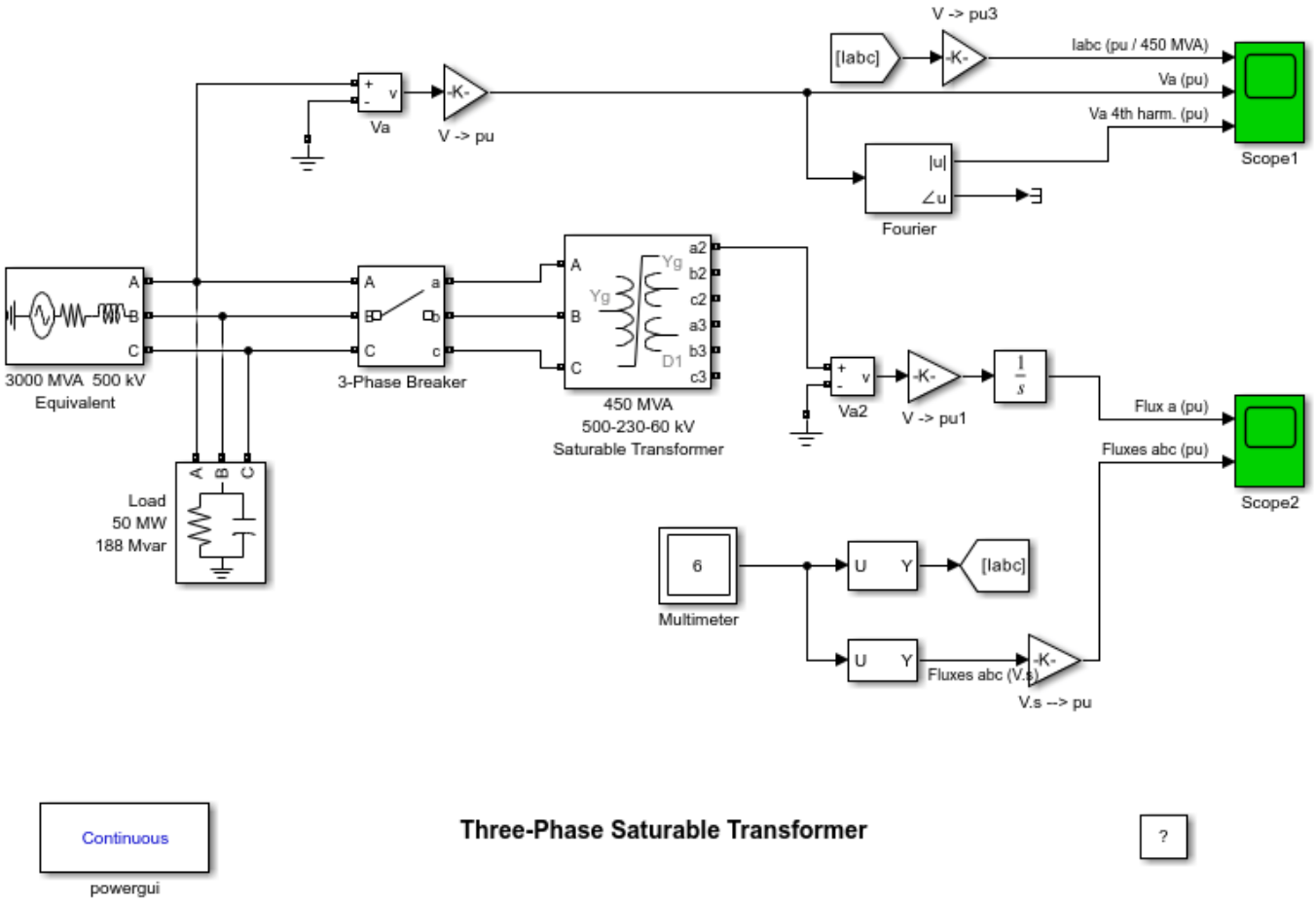
Furthermore, as the inductive reactive power of Load 1 and Load 2 (2*10 kvar) is compensated by the capacitive reactive power of Load 3 (20 kvar), the primary current is almost in phase with voltage The small phase shift (-2.8deg.) is due to the reactive power associated with transformer reactive losses.

Open the two scopes and start the simulation. When the circuit breaker opens, a current starts to flow in the neutral as a result of the load unbalance. When the breaker opens, the active power decreases from 70 kW to 50 kW.

# Three-Phase Saturable Transformer

This example shows the transformer saturation phenomenon.

G. Sybille (Hydro-Quebec)




### Description

A three-phase transformer is energized on a 500 kV network. The transformer rated 450 MVA, 500 kV/230 kV/60 kV consists of three windings connected in Y/Y/Delta.

The power system is simulated by an equivalent circuit consisting of an inductive source (short-circuit power of 3000 MVA) and a parallel RC load. The capacitor reactive power has been selected in order to produce a resonance at 240 Hz (4th harmonic).

The transformer saturation characteristics is approximated by a single slope Xsat of 0.32 pu, corresponding to an air core reactance Xac = 0.40 pu (Xac = Xsat+Xh = 0.32+0.08 = 0.40 pu.) seen from the primary. Three residual fluxes ( -0.8 - 0.4 0.4 pu) are specified for phases A B and C.

The Multimeter and Scope2 blocks are used to monitor extra signals without using measurement blocks. The six signals obtained at the output of the Multimeter are the three currents in the circuit

breaker and the three fluxes inside the core of the saturable transformer. The flux on phase A is also obtained by integrating the phase A voltage at the unloaded output of winding two. The voltage and flux are converted to p.u. with gain blocks using proper scaling. The Fourier block measures the 4th harmonic content of the phase A primary voltage.

In order to allow further signal processing, signals displayed on Scope1 are sampled at 1/60/333 s (333 samples/ cycle) and stored in a variable named 'psbtransfosat_str' (structure with time).

**Simulation**

Start the simulation and observe voltage, current and flux waveforms on Scope1 and Scope2.

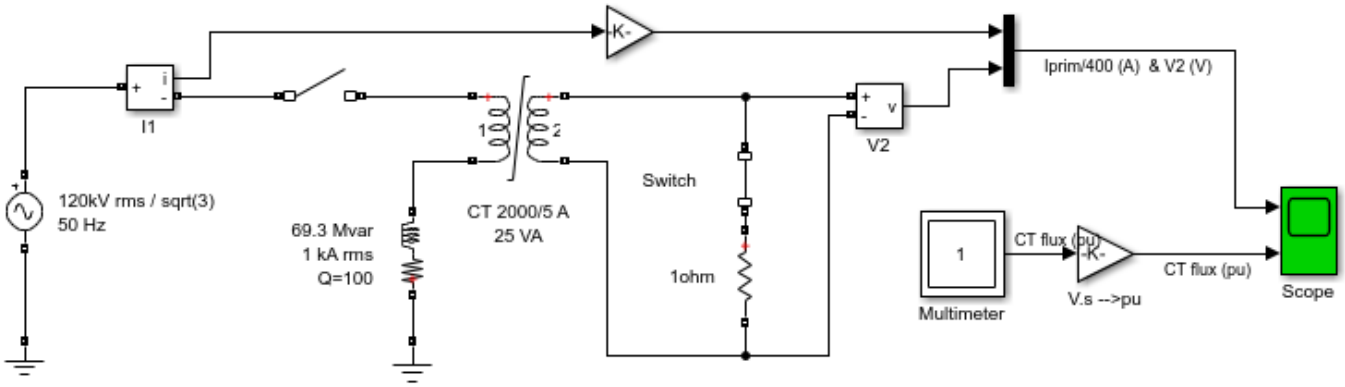
Observe the transformer inrush currents and overvoltage after breaker closing. Because of the 0.8 pu residual flux on phase A and breaker closing at an instant causing maximum flux offset, the flux exceeds 2 pu on phase A. Compare the flux in phase A measured by the multimeter (yellow trace on 2nd input of Scope2) and the flux obtained by integration of the secondary voltage (first trace of Scope2).

Voltage Va contains a high level of 4th harmonic (displayed on 3rd trace of Scope1) due to the 4th harmonic content of current injected into the network resonating at that particular frequency (max. of 0.23 pu at t = 0.32 s). When simulation is completed, open the Powergui and select 'FFT Analysis' to display the 0 - 1000 Hz frequency spectrum of signals saved in the 'psbtransfosat_str' structure. The FFT will be performed on a 2-cycle window starting at 0.32 s (instant of highest 4th harmonic). Select input labeled 'Va (pu)' and click on Display to obtain the Va voltage spectrum. Note that distortion is caused mainly by harmonic orders 2 to 6 (highest content = 23% of 4th harmonic as obtained from the Fourier block).

# Current Transformer Saturation

This example shows the measurement distortion due to saturation of a current transformer (CT).

G. Sybille (Hydro-Quebec)



### Description

A current transformer (CT) is used to measure current in a shunt inductor connected on a 120 kV network. The CT is rated 2000 A / 5 A, 5 VA. The primary winding which consists of a single turn passing through the CT toroidal core is connected in series with the shunt inductor rated 69.3 Mvar, 69.3 kV (120kV/sqrt(3)), 1 kA rms. The secondary winding consisting of 1*2000/5 = 400 turns is short circuited through a 1 ohm load resistance. A voltage sensor connected at the secondary reads a voltage which should be proportional to the primary current. In steady state, the current flowing in the secondary is 1000*5/2000 = 2.5 A (2.5 Vrms or 3.54 Vpeak read by the voltage measurement block V2).

Open the CT dialog box and observe how the CT parameters are specified. The CT is assumed to saturate at 10 pu and a simple 2 segment saturation characteristic is used.

The primary current reflected on the secondary and the voltage developed across the 1 ohm resistance are sent to trace 1 of the Scope block. The CT flux , measured by the Multimeter block is converted in pu and sent to trace 2. (1 pu flux = 0.0125 V *sqrt(2)/ (2*pi*50) = 5.63e-5 V.s)

The switch connected in series with the CT secondary is normally closed. This switch will be used later to illustrate overvoltages produced when CT secondary is left open.

### Simulation

**1. Normal operation**

In this test , the breaker is closed at a peak of source voltage (t = 1.25 cycle). This switching produces no current asymmetry. Start the simulation and observe the CT primary current and secondary voltage (first trace of Scope block). As expected the CT current and voltage are sinusoidal and the measurement error due to CT resistance and leakage reactances is not significant. The flux contains a DC component but it stays lower than the 10 pu saturation value.

**2. CT saturation due to current asymmetry**

Now, change the breaker closing time in order to close at a voltage zero crossing. Use t = 1/50 s. This switching instant will now produce full current asymmetry in the shunt reactor. Restart the simulation. Observe that for the first 3 cycles, the flux stays lower than the saturation knee point (10 pu). The CT voltage output V2 then follows the primary current. However, after 3 cycles, the flux asymmetry produced by the primary current causes CT saturation, thus producing large distortion of CT secondary voltage.

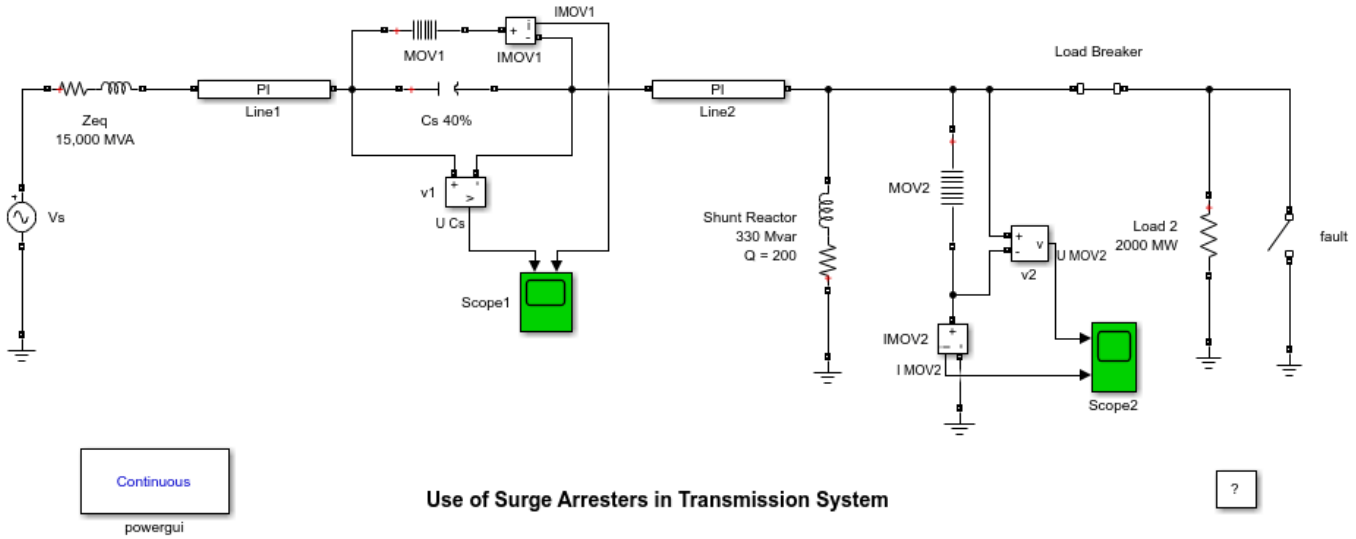
**3. Overvoltage due to CT secondary opening**

Reprogram the primary breaker closing time at t = 1.25/50 s (no flux asymmetry) and change the secondary switch opening time to t = 0.1 s. Restart the simulation and observe the large overvoltage produced when the CT secondary is opened. The flux has a square waveshape chopped at +10 and -10 pu. Large dphi/dt produced at flux inversion generates high voltage spikes (250 V).

# Use of Surge Arresters in Transmission System

This example shows the use of surge arresters in a series and shunt compensated 735 kV AC transmission system

G. Sybille (Hydro-Quebec)




### Description

A 735 kV equivalent transmission systems feeds a load through a 200 km transmission line. The line is series compensated at the middle point and shunt compensated at its receiving end. A fault is applied at the load terminals. The fault is cleared by load breaker opening. For simplification purpose, only one phase of the transmission system is modeled. All parameters correspond to positive-sequence.

The three-phase short circuit level of the transmission system is 15000 MVA. The line is 40% series compensated by a capacitor (26.2 ohms at 60 Hz) and shunt compensated by a 330 Mvar (110 Mvar/ phase) inductor at the load end.

The series capacitor and shunt inductor are both protected by metal oxide varistors (MOV). The series capacitor varistor MOV1 consists of 30 columns protecting the capacitor at 2.5 times its rated voltage (rated voltage is obtained for a 2000 kA line rated current). The corresponding protection voltage (defined at 15 kA = 500 A per column) is 2.5*26.2 *2kA*sqrt(2) = 185 kV.

The shunt inductor is protected by a 2-column arrester (MOV2) at 1.8 p.u. of nominal phase-to-ground voltage (424.4 kVrms). The corresponding protection voltage (defined at 1 kA or 500A /column) is 1.8*424.4*sqrt(2) = 1080 kV.

### Simulation

Start the simulation. Observe the MOV voltages and currents on Scope1 and Scope2.

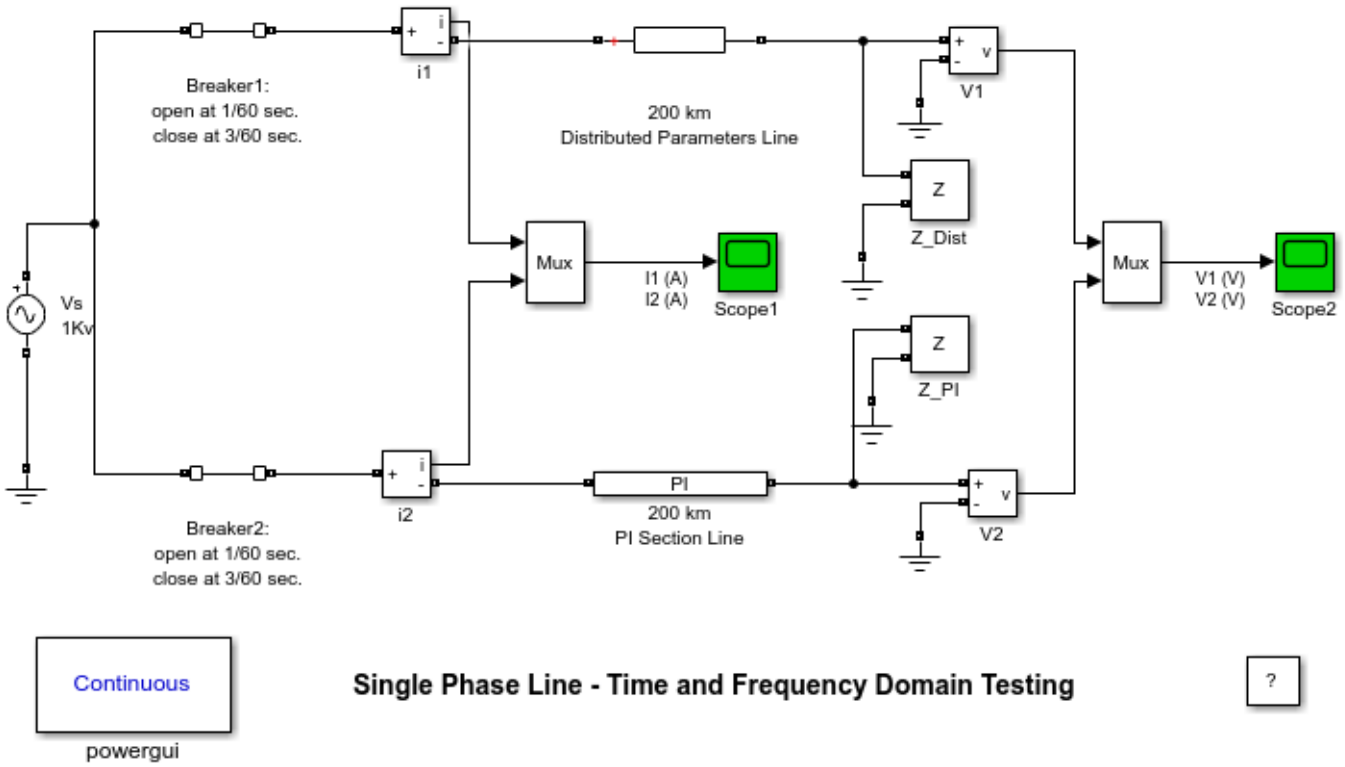
At fault application, the capacitor voltage increases and MOV1 conducts. Peak current reaches 8.4 kA on the 2nd pulse 2 cycles after fault application, the MOV current is symmetric (4.5 kA current pulses at every half cycle).

At fault clearing, the overvoltage produced at bus B2 is limited by the shunt MOV2 . The fast transient produces a 850 A current pulse in the MOV.

# Single Phase Line - Time and Frequency Domain Testing

This example shows time domain and frequency domain performance of distributed parameter line model and PI section line model.

G. Sybille (Hydro-Quebec)




### Description

A 200 km line is connected on a 1 kV, 60 Hz infinite source. The line is deenergized and then reenergized after 2 cycles. The simulation is performed simultaneously with two different line models:

- Distributed parameter line

- PI section line consisting of two 100 km sections.

Currents at the sending end and voltages at the receiving end are compared for the two line models. Impedance Measurement blocks are connected at the open end of both lines in order to compare their frequency responses.

### Simulation

#### 1. Steady-state

Open the Powergui block and select **Steady-State Voltages and Currents** to display the voltage and current phasors. Observe that the values obtained with the two models are the same.

**2. Time domain comparison**

Open the two scopes and start the simulation. Observe the difference in current and voltage waveforms at breaker opening and reclosing. Note the sharp edges of the distributed parameter model. These voltage and current steps which are due to travelling wave reflections at line ends are filtered by the PI model.

**3. Frequency domain comparison**

Open the Powergui block and select **Impedance vs Frequency Measurement**. A new window appears, listing the two Impedance Measurement blocks Z_Dist and Z_PI connected to your circuit. Note also that parameters are set to compute impedance in the 0:2000 Hz frequency range by steps of 2 Hz. Using the Ctrl key, select both Z_Dist and Z_PI in the upper right window. Then, click on the Display button. The two impedances are computed and displayed on the same graph.

Note that the distributed parameter line shows a succession of poles and zeros equally spaced, every 486 Hz. The first pole occurs a 243 Hz, corresponding to frequency f = 1 / (4*T), where :

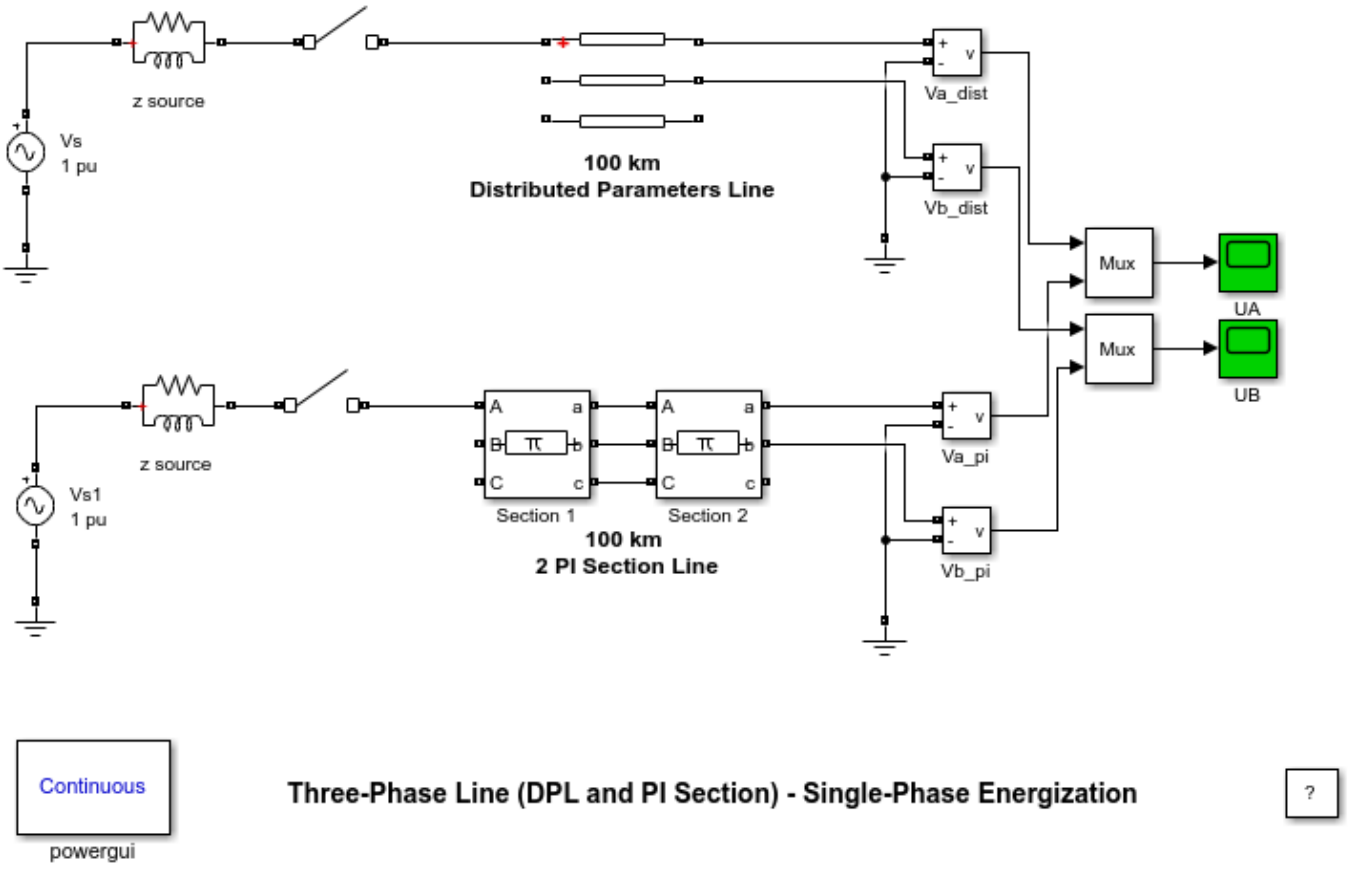
```
T = travelling time = length*sqrt(L*C) = 200*sqrt(2.137e-3*12.37e-9) = 1.028 ms
```

The PI section line only shows two poles because it consists of two PI sections. Impedance comparison shows that a two-PI line gives a good approximation of the distributed line for the 0-350 Hz frequency range.

# Three-Phase Line (DPL and PI Section) - Single-Phase Energization

This example shows transients obtained with a distributed parameter line model and a PI section line model.

G. Sybille (Hydro-Quebec)




### Description

A 100 km, three-phase line is energized on a 735 kV, 30,000 MVA equivalent network. The line is first energized on phase A at t = 1/4 cycle (maximum of source voltage) and then deenergized at t = 4 cycles. Phase B and phase C are left open. Switchings are performed simultaneously on two different line models:

- Distributed parameters line

- Three-phase PI section line consisting of two 50 km sections

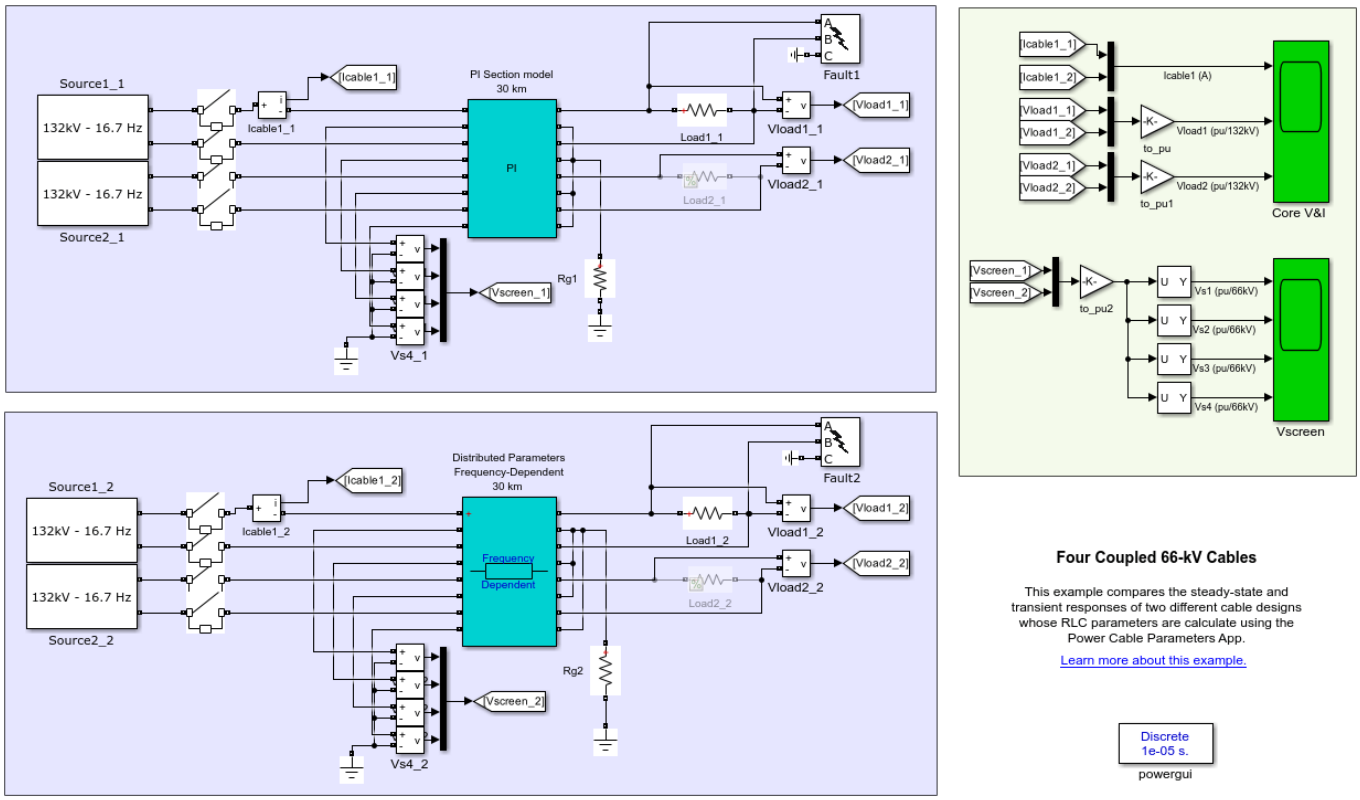
Voltages obtained on phases A and B at the line receiving end with the two models are compared.

### Simulation

Open the two scopes and start the simulation. Compare the voltages obtained on phase A and phase B (induced voltage) with the two models. Zoom on voltages right after energizing and notice the sharp edges obtained with the distributed parameter line (yellow trace). These voltage steps which are due to travelling wave reflections at both ends of the line are filtered by the PI model. Notice also that phases A and B stay charged at peak voltage when the breakers are opened.

# Four Coupled 66-kV Cables

This example compares the steady state and transient responses of two different cable designs whose R L C parameters are calculated using the Power Cable Parameters app.




## Description

A set of four 66-kV 30-km cables is used to transmit power to two 132-kV 50-MW resistive loads at 16.7 Hz. Only Load1 is connected at cable1-cable2 receiving end, while cable3-cable4 receiving end is left open (Load2 commented out). Screen conductors are grounded on the receiving end only.

Cables are buried at 2m depth, and they are arranged in a horizontal flat configuration. To see detailed geometric parameters of this cable system, select «Edit geometry parameters» in the Distributed-Parameters Frequency-Dependent block menu. This will open the Power Cable Parameters App which has been used to compute the R L C 8x8 matrices of this cable system (8 conductors). Conductors 1 3 5 7 are phase conductors, conductors 2 4 6 8 are sheath conductors (screens).

The RLC matrices computed at nominal frequency (16.7 Hz) are used by the PI Section block. The 400 RLC matrices computed for a frequency range 10e-4 to 10e5 Hz are used to fit the Frequency-Dependent cable model (Distributed-Parameters block). Computed RLC matrices and geometric parameters are stored in the DATA structure saved in the Param_4cablesUnderground2cond.mat file (available in Parameters/Load typical menu of the Power Cable Parameters App). These RLC matrices are automatically loaded in your workspace at model opening (see model Properties/ PreLoadFcn callback of the model)

This example requires the Optimization Toolbox™ Software.

### Simulation

In order to compare transient response of the PI and Frequency-Dependent models, the following switching operations are performed:

- t = 3.25 cycles: Energization of the four cables on infinite sources through four circuit breakers on the source side
- t = 5 cycles: 2-cycle fault across Load1 (cable 1 - cable 2)

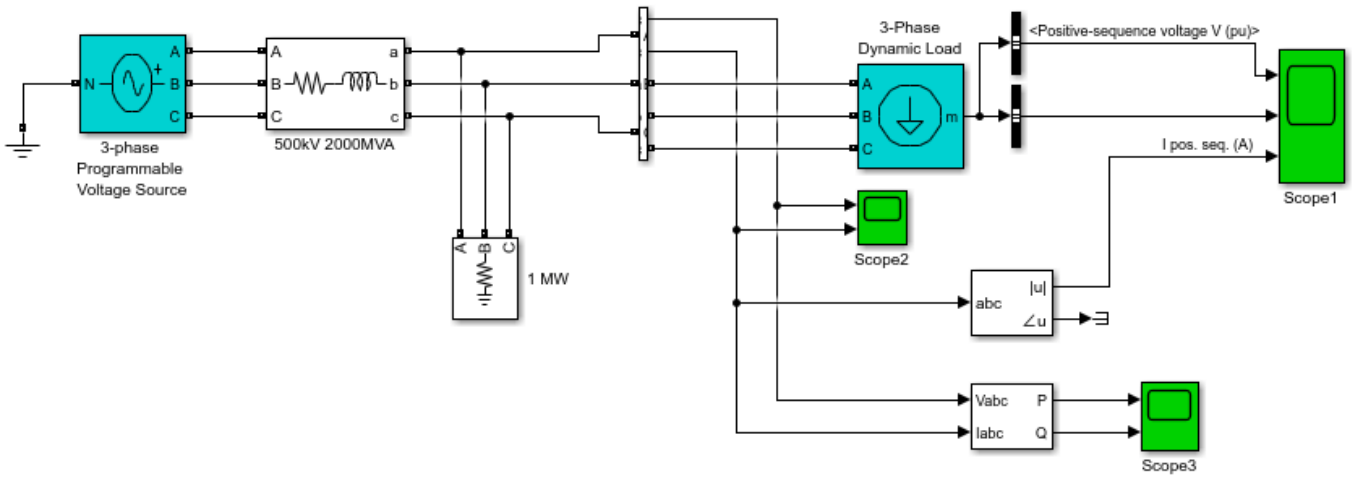
On the two Scopes notice that both models give same steady-state voltages and currents.

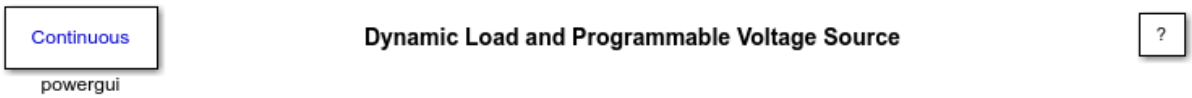
Zoom X around t=0.135 sec to compare phase and screen conductor voltage transients at cable energization. The PI Section is appropriate for steady state, but it does not reproduce the high frequency content of transient voltages. For example, Zooming on Vload2 waveform shows a damped sinusoidal oscillatory response for the PI section (yellow) whereas the Distributed-Parameters Frequency-Dependent model has a damped square-wave type response (blue). The propagation delay of the Distributed-Parameters line is clearly seen (delay ~ 150 us = time difference between the blue and yellow traces when voltage starts to increase at receiving end). The PI section model assuming constant RLC parameters evaluated at 16.7 Hz has a limited frequency response, whereas the Distributed-Parameters Frequency-Dependent model correctly represents a wide frequency range from DC to ~100 kHz.

Observe also voltages induced on screens. As expected, during fault, because of the flat cable configuration, a higher voltage is induced on screen 3 (~ 0.2 pu/66 kV) than on screen 4 (~ 0.1 pu/66 kV). This is because cable 4 is farther from cables 1 and 2 than cable 3.

# Dynamic Load and Programmable Voltage Source

This example shows the use of the 3-Phase Dynamic Load and 3-Phase Programmable Voltage Source blocks




### Description

A dynamic load is connected on a 500 kV, 60 Hz power network . The network is simulated by its Thevenin equivalent (voltage source behind a R-L impedance corresponding to a 3-phase short circuit level of 2000 MVA). The source internal voltage is modulated in order to simulate voltage variation during a power swing. As the dynamic load is a nonlinear model simulated by current sources, it cannot be connected to an inductive network (R-L in series). Therefore, a small resistive load (1 MW) has been added in parallel with the dynamic load.

The dynamic load power is a function of its terminal positive-sequence voltage V. Open the Dynamic Load menu and notice that both exponents np and nq are set to 1 and that the specified minimum voltage Vmin is 0.7 pu . It means that the load active power P and reactive power Q are defined by the following equations:

```
If V > Vmin
      P = Po*(V/Vo);     Q = Qo*(V/Vo)
If V < Vmin
      P = Po*(V/Vo)^2;    Q = Qo*(V/Vo)^2
```

In other words, as long as voltage is higher than 0.7 pu, the load current is constant. When voltage falls below 0.7 pu the load behaves as a constant impedance.

In order to simulate the variation of P and Q as function of voltage, the source internal voltage is controlled by the 3-Phase Programmable Voltage Source block. Open the source menu and notice that the specified type of amplitude variation is a sinusoidal modulation (Amplitude of the modulation =

0.5 pu, Frequency of the modulation = 1 Hz). Therefore, the source positive-sequence voltage varies between 0.5 pu and 1.5 pu. The initial source voltage is 1 pu. Modulation starts at t = 0.2 s and stops after 1 cycle at t = 1.2 s.

### Simulation

#### 1. Initializing the dynamic load

In order to start the simulation in steady-state you have to specify the correct initial voltage Vo (magnitude and phase) corresponding to desired Po and Qo values.

The 'Machine Initialization' utility of Powergui block can be used to find this voltage and initialize the dynamic load. Open the Powergui and select 'Machine Initialization'. Specify the desired active power and reactive powers for the Dynamic Load (50 MW, 25 Mvar): Active Power = 50e6 Reactive Power = 25e6.

Then press the 'Compute and Apply' button. The phasors of AB and BC machine voltages as well as the currents flowing into of phases A and B are updated. The phase-neutral load voltage Uan is also displayed: ( 0.9844 pu - 1.41 degrees). If you now open the Dynamic Load dialog box you will notice that the values of Po, Qo and Vo have been updated.

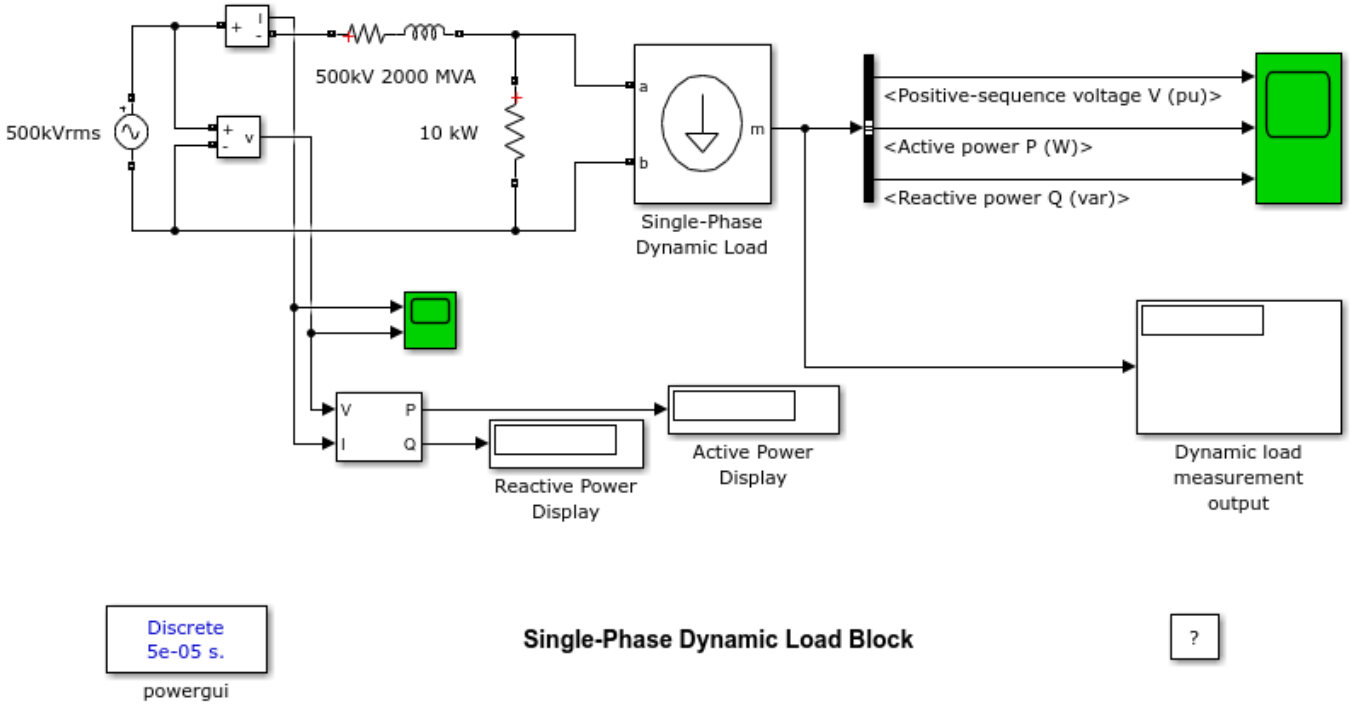
#### 2. Simulating the voltage swing

Start the simulation and observe load voltage, P&Q powers and current on Scope1. Observe that simulation starts in steady state. At t = 0.2 s, when voltage modulation is initiated, P and Q start to increase (trace 2), but, as np and nq are set to 1, the load current (trace 3) stays constant. When voltage falls below 0.7 pu the load behaves as a constant impedance. Therefore load current follows this voltage variation. Observe on Scope2 variations of instantaneous voltages and currents. Also, notice that computed Pand Q displayed on Scope3 are the same as P and Q internal signals returned by the Dynamic Load measurement output.

# Single Phase Dynamic Load Block

This example shows a single-phase dynamic load block built with Simulink® blocks.

Graham Dudgeon, Senior Consultant, The MathWorks, Inc.




### Description

Look under the mask of the custom-built Single-Phase Dynamic Load block and observe the three systems. The 'Measurement System' consist of the voltage measurement blocks which convert the Specialized Power Systems signal into Simulink signal. The 'Simulink System Model', contains the core functionality of the Single-Phase Dynamic Load block. Lastly, the 'Current Injection System' contains the controlled current source blocks which convert Simulink signals into Simscape™ Electrical™ Specialized Power Systems signals.

In general, measurement blocks (voltage/current/impedance) serve as an interface between Specialized Power Systems and Simulink, and controlled source blocks (voltage/current) between Simulink and Specialized Power Systems. This is the recommended template for interfacing custom-built blocks in Simulink, with Simscape Electrical Specialized Power Systems.

### Limitations

1. The Single-Phase Dynamic Load block is modeled as a current source, it cannot be connected to an inductive network, or to another current source. Therefore, a small resistive load (10 Kw) has been added in parallel with the dynamic load.

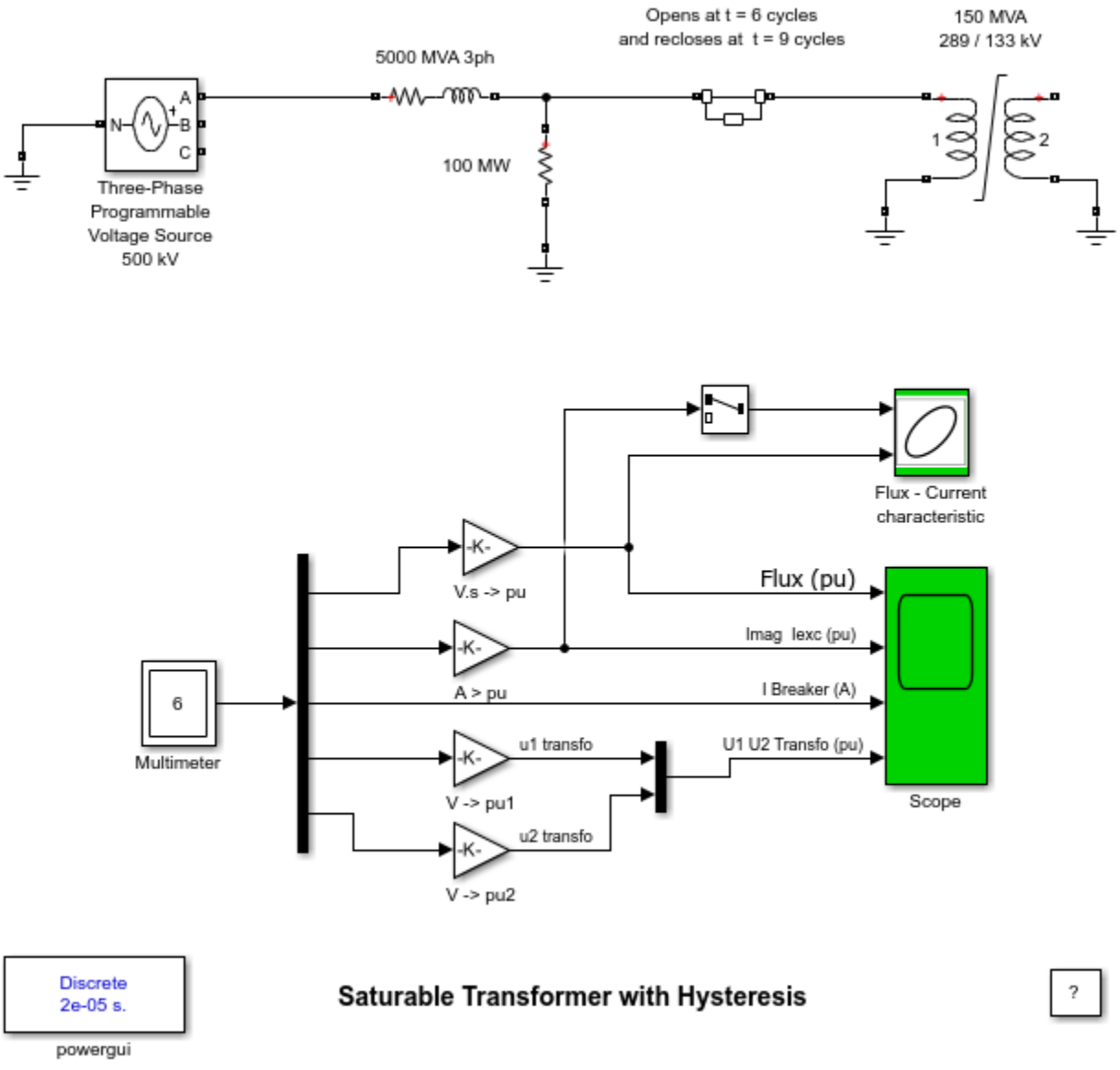
2. The Machine Initialisation tool and the Load Flow tool of Powergui block cannot be used to initialize the Single-Phase Dynamic Load model.

3. The Single-Phase Dynamic Load block cannot be used with the Phasor simulation type mode of the Powergui.

# Saturable Transformer with Hysteresis

This example shows the simulation of hysteresis in a saturable transformer.

S. Casoria (Hydro-Quebec) and P. Brunelle (TransEnergie)




### Description

One phase of a three-phase transformer is connected on a 500 kV, 5000 MVA network. The transformer is rated 500 kV/230 kV, 450 MVA (150 MVA per phase). The flux-current saturation characteristic of the transformer is modeled with the hysteresis or with a simple piecewise nonlinear characteristic.

A Three-Phase Programmable Voltage Source is used to vary the internal voltage of the equivalent 500 kV network. During the first 3 cycles source voltage is programmed at 0.8 pu. Then, at t = 3 cycles (0.05 s) voltage is increased by 37.5% (up to 1.10 pu). In order to illustrate remnant flux and inrush current at transformer energization, the circuit breaker which is initially closed is first opened at t = 6 cycles (0.1 s), then it is reclosed at t = 9 cycles (0.15 s).

The Initial flux phi0 in the transformer is set at zero and source phase angle is adjusted at 90 degrees so that flux remains symmetrical around zero when simulation is started.

A Multimeter block and a Scope block are used to monitor waveforms of flux, magnetization current (not including the eddy currents which are modeled by the Rm resistance), excitation current (including eddy current modeled by Rm), voltages and current flowing into the primary winding. A X-Y Graph block is used to monitor the transformer operating point moving on the flux-current characteristic.

**Simulation**

**1) Simulation of saturation with hysteresis**

Open the transformer menu and select the 'Simulate hysteresis' check box. Using the 'Hysteresis design' tool of the Powergui, load the hysteresis characteristic (load file 'hysteresis.mat'). Notice that flux and excitation current are displayed in pu. Using the 'Parameter units' popup menu, you can convert the pu units to SI units. The saturation characteristics consists of two regions: 1) The main hysteresis loop: In this region there are 2 different flux values for a single current value. 2) The saturated region defined by a simple line segment starting from the maximum point (Is,Fs) of the main loop. The hysteresis loop is defined by the following 3 points marked by red crosses on the main loop: [I=0; Remnant flux(Fr = 0.85 pu)], [Coercive Current(Ic = 0.004 pu); F = 0], [Saturation current(Is = 0.015 pu); Saturation flux(Fs = 1.2 pu)] plus the slope dF/dI at coercive current(F = 0). Using the 'Zoom around hysteresis' checkbox and 'Display' button you can view the whole characteristic or zoom on the hysteresis.

Start the simulation and observe the following phenomena on the two scope blocks:

**From 0 to 0.05 sec:** Voltage and flux peak values are at 0.8 pu. Notice typical square wave of magnetization current. As no remnant flux was specified, magnetization current and flux are symmetrical. Flux travels on inner loops (inside the main loop).

**From 0.05 to 0.1 sec:** Voltage is 1.1 pu. Flux now reaches approximately +1.1pu. A slight flux asymmetry is produced at voltage change. and the flux which varies between +1.14 pu and -1.05 pu now travels on the main loop. Current pulses appear on the magnetization current (yellow trace, Imag), indicating beginning of saturation.

**From 0.1 to 0.15 sec:** At first zero crossing after the breaker opening order, the current is interrupted, and a flux of 0.84 pu stays trapped in the transformer core.

**From 0.15 to 0.2 sec:** The breaker is reclosed at t = 9 cycles, at a zero crossing of source voltage, producing an additional flux offset of approximately 1 pu. The peak flux now reaches 1.85 pu, driving the transformer into the saturated region. Peak excitation current now reaches 0.81 pu.

**2) Simulation of saturation with a piecewise nonlinear characteristic**

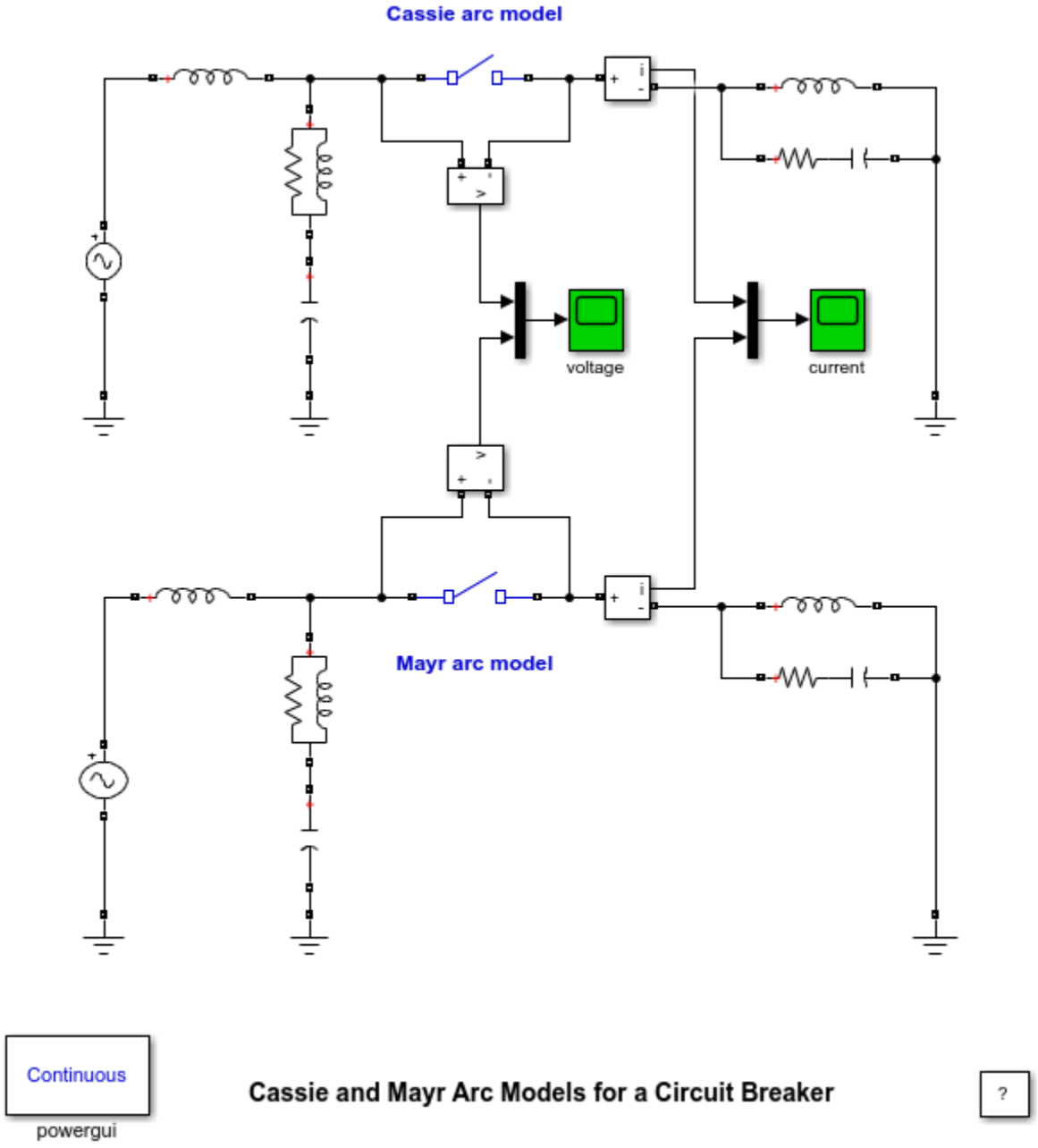
Open the transformer menu and deselect 'Simulate hysteresis'. The saturation will now be simulated by a piecewise nonlinear single-valued characteristic defined by 7 points. Current/Flux pairs (in pu) are: [0 0; 0.0 0.85; 0.015 1.2; 0.03 1.35; 0.06 1.5; 0.09 1.56; 0.12 1.572]. The saturated region is the

same, but the hysteresis loop is not simulated. Note that this single valued characteristics still allows specification of a remnant flux (keep phi0=0 flux as with hysteresis saturation model). Start simulation and compare waveforms.

# Cassie and Mayr Arc Models for a Circuit Breaker

This example shows the use of the Cassie and Mayr arc model for simulating high voltage circuit breaker interruption

P.H. Schavemaker, Power Systems Laboratory, Delft University of Technology, the Netherlands.
P.H.Schavemaker@its.tudelft.nl



### Description

Two identical circuits are displayed: one with a Cassie and one with a Mayr arc model. The circuit is a simple representation of a circuit breaker interrupting a short-line fault. At the source side of the circuit breaker a circuit is present for reproducing a (2-parameter IEC) transient recovery voltage, while the RLC circuit at the line side represents a short transmission line that is short circuited.

A stiff solver is recommended for solving this problems because of the nonlinear behavior of the arc.

### Simulation

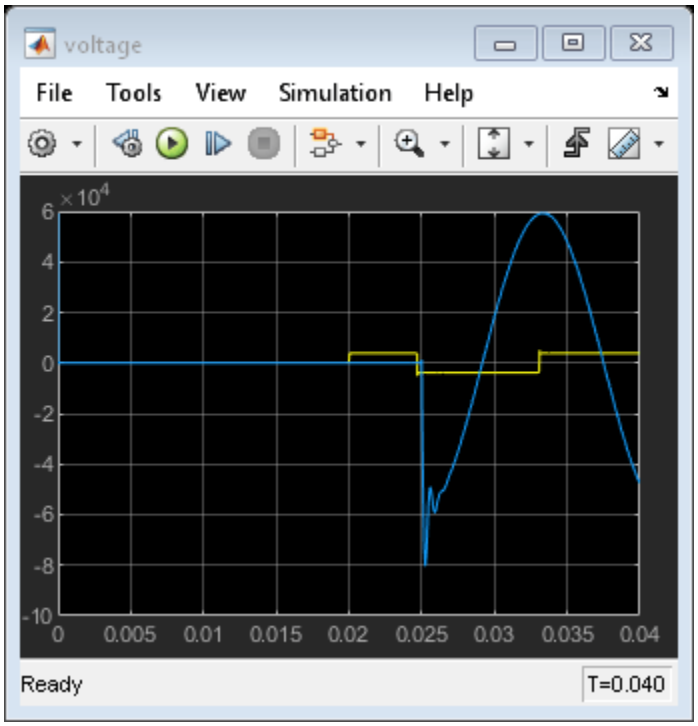



**Figure 2:** Voltage comparisons for the Cassie and Mayr arc models.

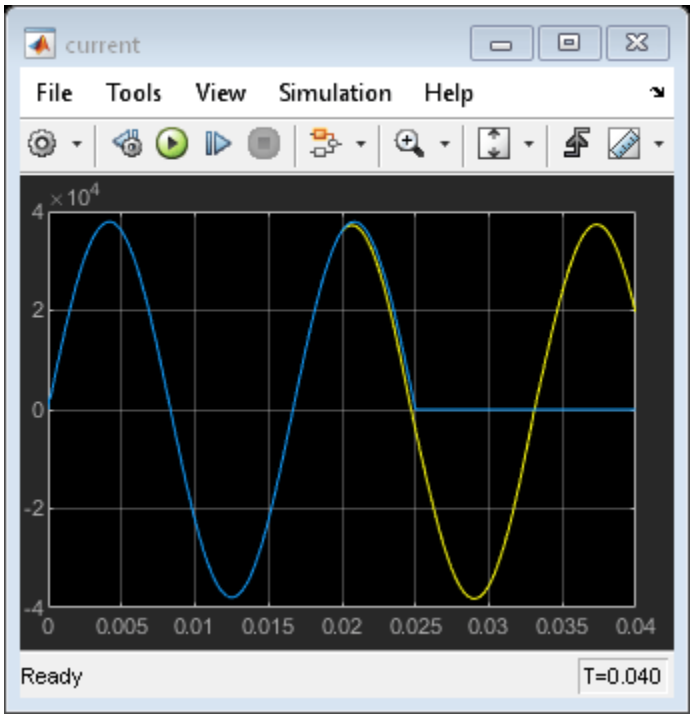



**Figure 3:** Current comparisons for the Cassie and Mayr arc models.

The following observations become clear from the simulation: The arc behaves as a non-linear resistance; therefore both arc current and voltage pass the zero value at the same time. As the power input into the arc channel is zero at that time, the current zero crossing is the place where the interruption takes place.

The Cassie arc model has a constant arc voltage in the high current interval, while the Mayr arc model shows an increasing arc voltage (the extinction peak) close to the current zero crossing.

The Cassie arc model fails to interrupt the short circuit current, while the Mayr arc model interrupts and shows a small post-arc current of 0.6A immediately after the current zero crossing. This current is caused by the transient recovery voltage building up over the non-infinite arc resistance.

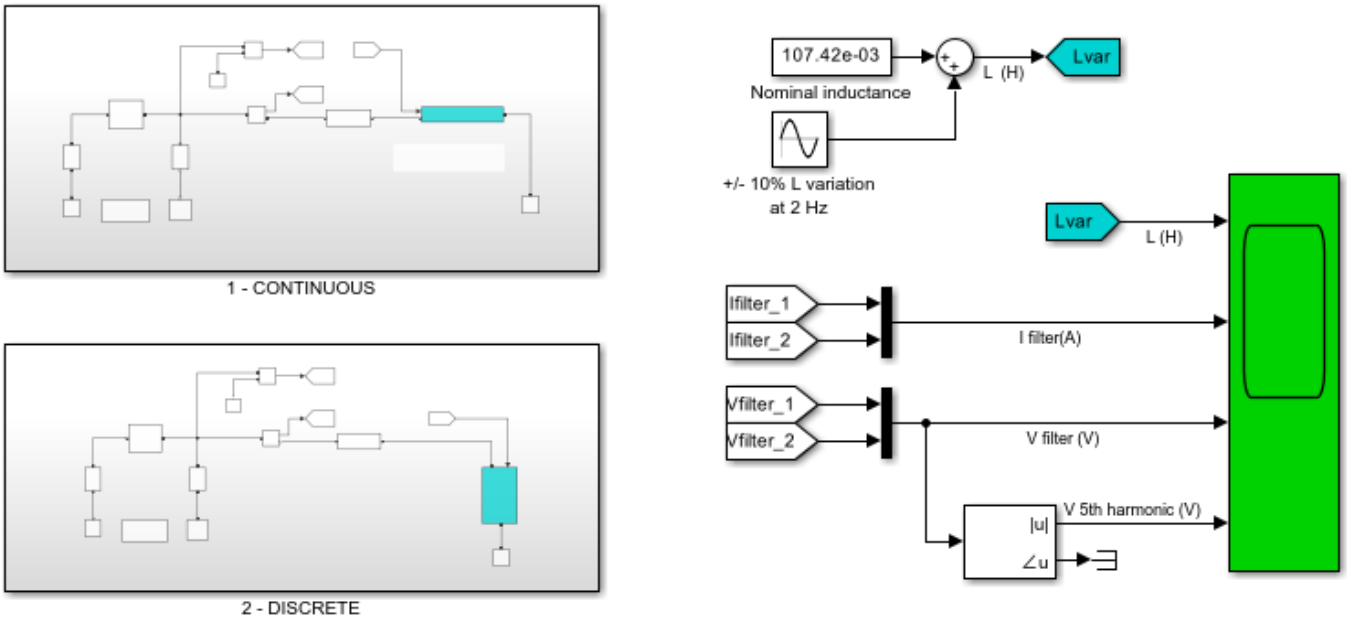
In case of the Mayr arc model (the successful interruption), the transient recovery voltage (TRV) builds up over the breaker. This voltage consists of the 2-parameter TRV and the high frequency voltage oscillation of the line side, which you can observe by zooming in.

### Additional Notes

This model cannot be run in accelerator mode due to modeling constraints.

# Variable Inductance Modeling

The example compares two models of a variable inductor.




**Variable Inductance Modeling**

Learn more about this example.

### Description

The variable inductor is used in a series LC filter tuned at the 5th harmonic. This filter is connected to a RL 60-Hz voltage source in parallel with a 300-Hz current source.

Inductance is varied sinusoidally +/- 10 % around its nominal value. Compare filter voltage and current of continuous and discrete models

# Interfacing Simulink Models with Simscape Electrical Specialized Power Systems

This example shows the interfaces between Simulink® and Simscape™ Electrical™ Specialized Power Systems to design a custom synchronous machine model. It also shows an example of a customized Synchronous Machine block

### Introduction

In this example, we use the Synchronous machine blocks as an example to show the interfaces between Simulink and Simscape Electrical Specialized Power Systems. There are two similar circuits shown in parallel. The top circuit model shows the starting of a a synchronous machine using blocks from the Specialized Power Systems library. The circuit below it is identical in all respects, but for a custom-built model of the Synchronous machine.

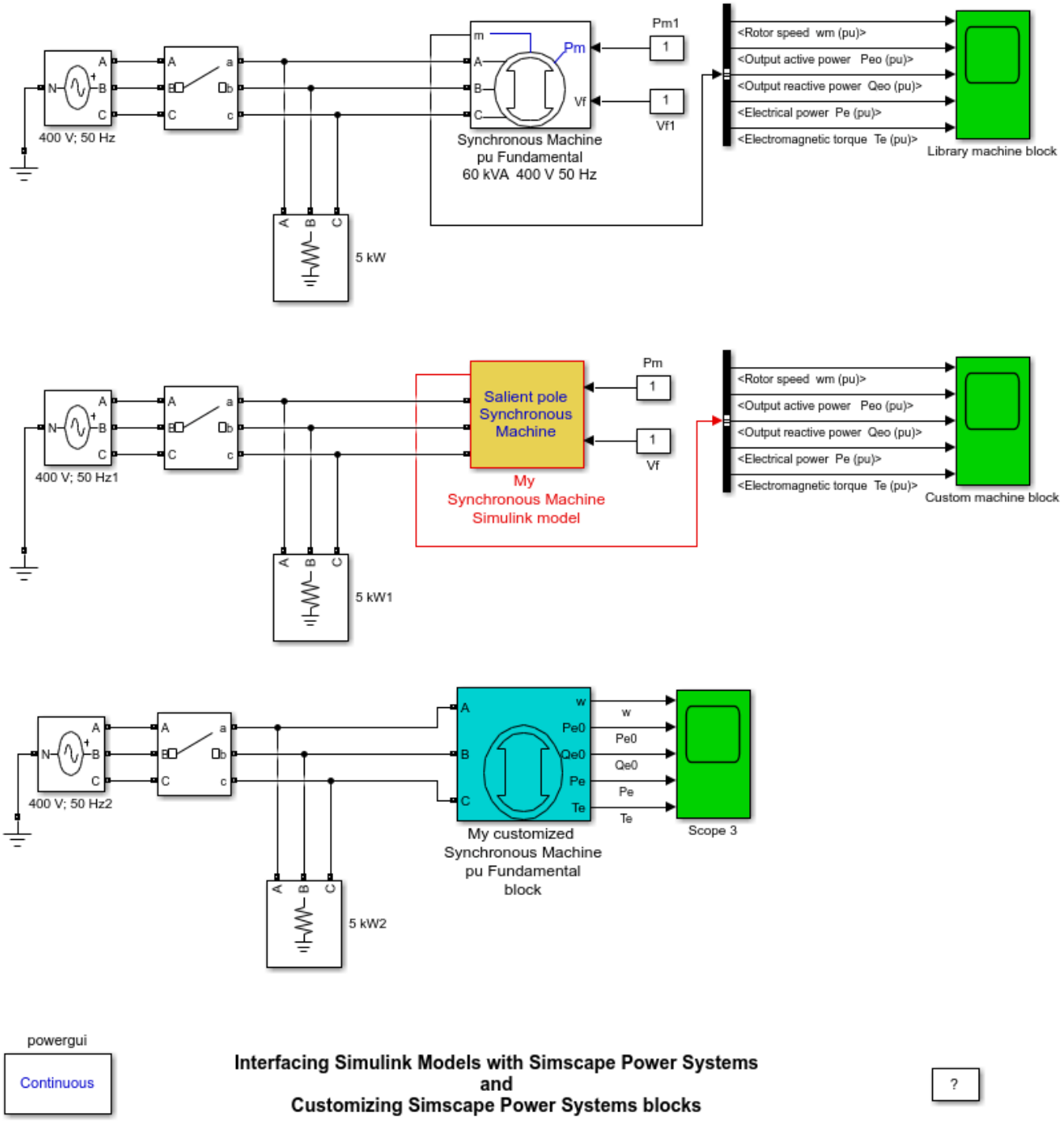



**Figure 1:** Starting of the synchronous motor. The top model is modeled using Specialized Power Systems library blocks while the middle model is a custom-made model built using Simulink. The bottom model is a customized version of the Synchronous Machine block of powerlib, using the power_customize function of powergui.

### Description

Click to look under the mask of the official Synchronous machine block, and find the core functionality of the block modeled in the 'Source' and the 'SM_mechanics' subsystems. It is important to note that these subsystems are built up from either Simulink elements or basic Specialized Power Systems blocks. Users may need to change this representation to add their own functionality to the block, and meet their custom needs.

Now look under the mask of the custom-built Synchronous machine block (in the middle circuit), and observe the three subsystems. The 'Measurement System' in green, contains the voltage measurement blocks which convert the Specialized Power Systems' signals into Simulink signals. The 'Simulink System Model', contains the core functionality of the Synchronous machine block, comprising the 'Source' and the 'SM_mechanics' subsystems, borrowed from the standard shipped block. It is here that users may customize and manipulate the extracted Simulink signals to any degree. Lastly, the 'Current Injection System', contains the controlled current source blocks which convert Simulink signals into Specialized Power Systems' signals.

In general, measurement blocks (voltage/current/impedance) serve as an interface between Simscape Electrical Specialized Power Systems and Simulink, and controlled source blocks (voltage/current) between Simulink and Simscape Electrical Specialized Power Systems. This is the recommended template for interfacing custom-built blocks in Simulink, with Simscape Electrical Specialized Power Systems.

### Simulation

Run the example to see that there is no differences between the two circuits.

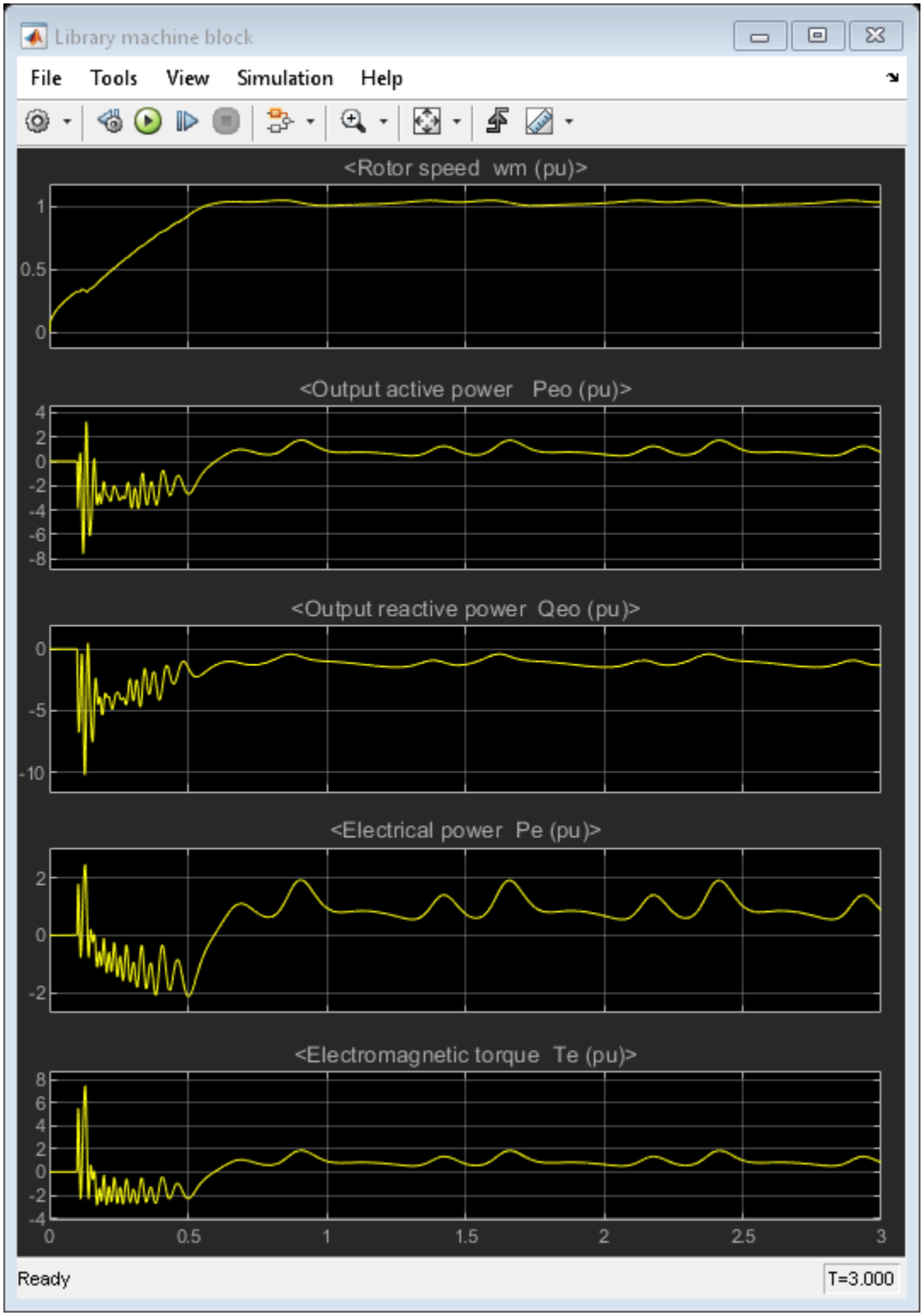



**Figure 2:** Results obtained from the Specialized Power Systems model.

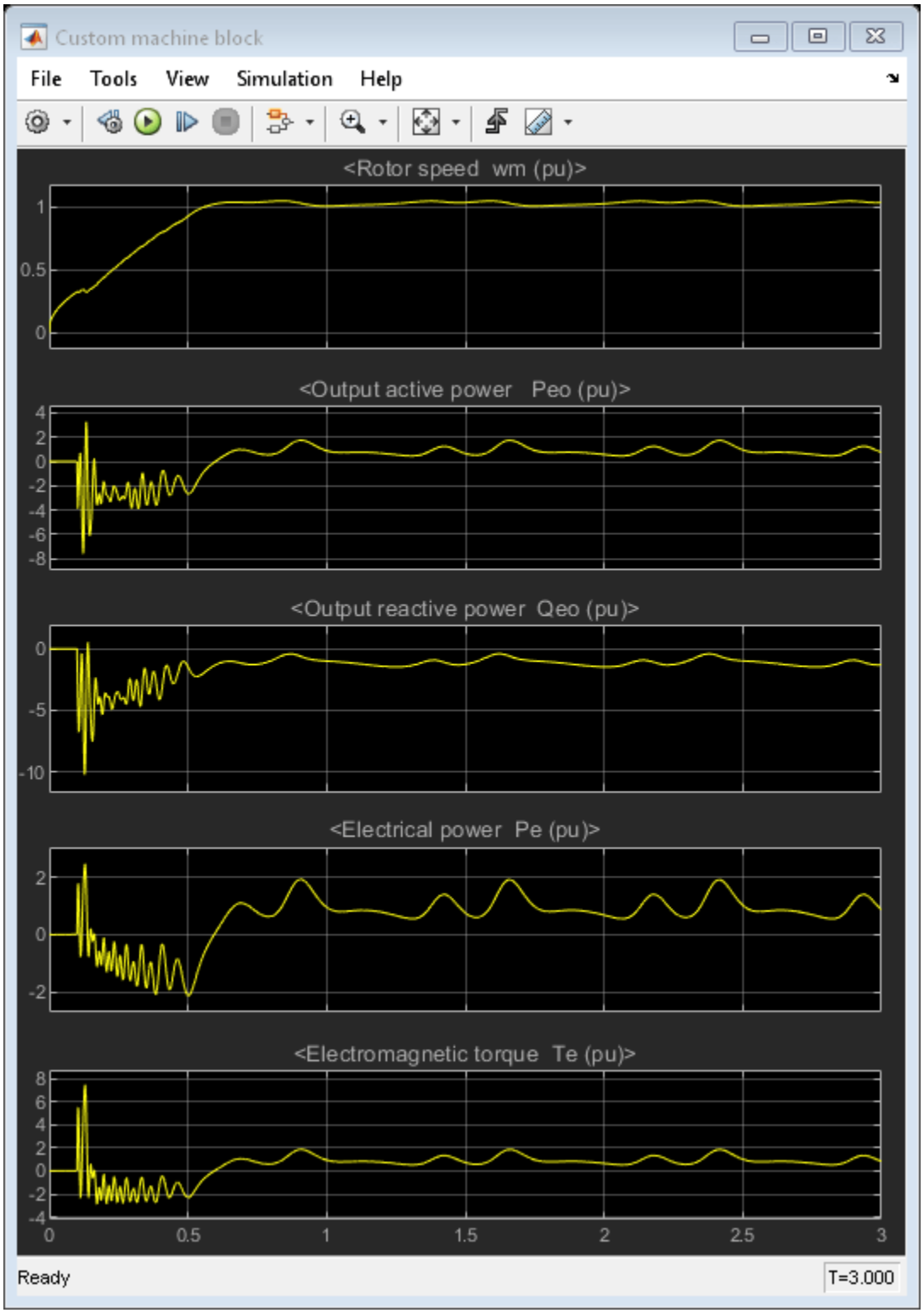



**Figure 3:** Results obtained from the custom-made Simulink model.

### Conclusion

This model presents a model of the Synchronous machine block, but users may just as well employ and edit any standard block in Specialized Power Systems. Extending this further, using the recommended template, it is even possible to create any custom block not provided with Specialized Power Systems, in Simulink, and then interface it seamlessly with a Simscape Electrical Specialized Power Systems circuit.

# Interfacing Simscape Models with Simscape Electrical Specialized Power Systems

This example shows the interface between Simscape™ and Simscape Electrical™ Specialized Power Systems to design a breaker model.

### Introduction

There are two similar circuits shown in parallel. The top circuit model illustrates tripping of a circuit breaker protecting a 240 V residential distribution circuit. The Circuit Breaker model is built with Simscape Electrical Specialized Power Systems blocks. The circuit below is identical in all respects, but for a Simscape model of the Circuit Breaker.

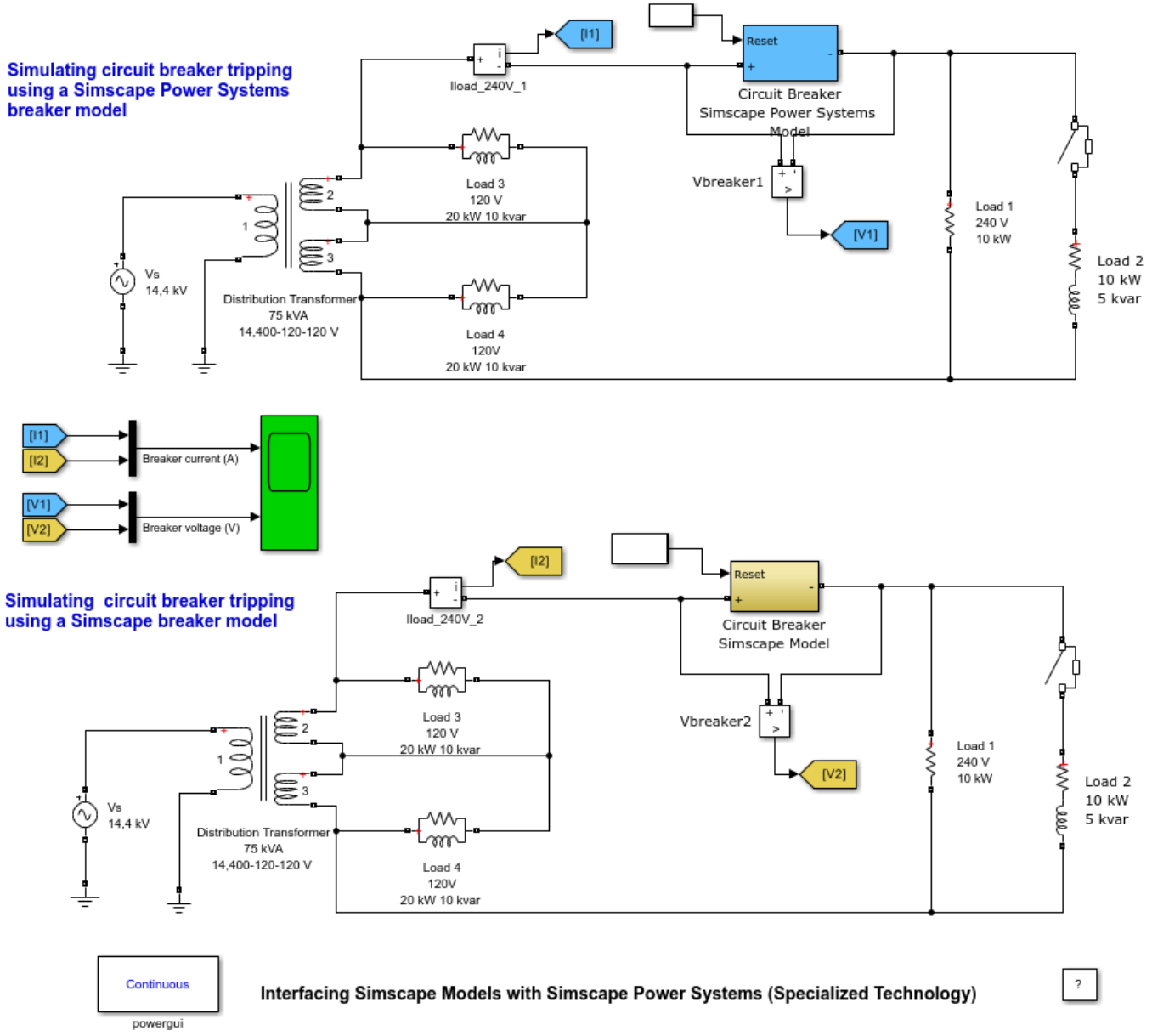



**Figure 1:** In the top model the Circuit Breaker is modeled using two Specialized Power Systems blocks (Ideal Switch and Current Measurement) and associated logic implemented with Simulink® blocks while the lower model uses two Simscape blocks (Switch and Current Sensor).

### Description

The top circuit illustrates tripping of a circuit breaker in a 240 V residential distribution circuit. Initially, the 240 V circuit feeds a 10 kW load. Then, at t=0.05 s, a second load (10 kW, 5 kvar) is switched on, causing current to exceed the trip setting (100 A). Current is interrupted at its first zero-crossing after the trip current has been exceeded. Finally, Load 1 is reconnected by resetting the circuit breaker at t= 0.2 s.

The bottom circuit is using the Simscape Interface Element block of powerlib to connect a Simscape circuit breaker model to the Specialized Power Systems circuit. The Simscape breaker block is a modified version of the breaker model found in ssc_circuitbreaker model.

**Simulation**

On running the model we can observe that the two different implementations of the circuit breaker device produce the same circuit breaker current and voltage.

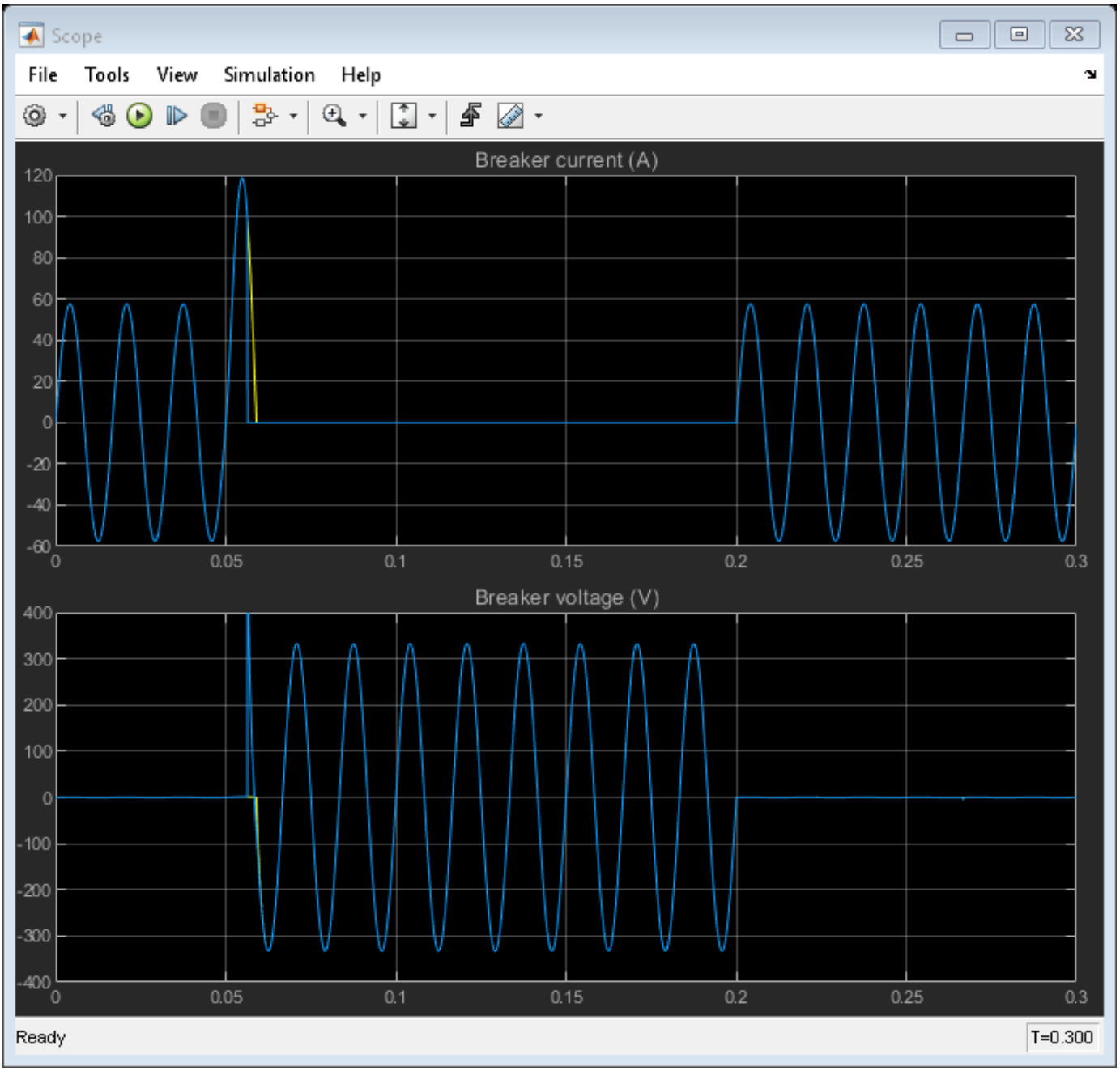


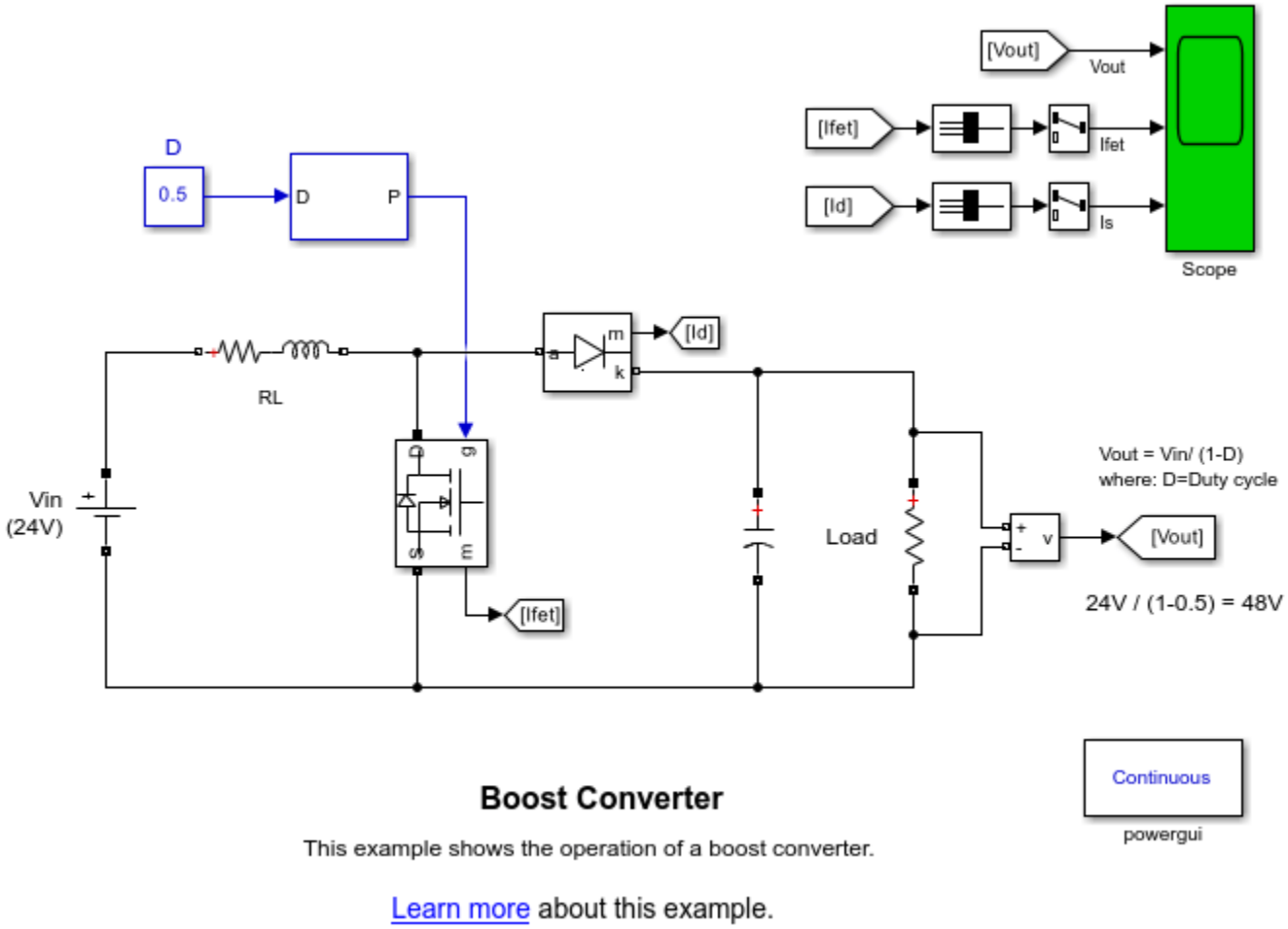
**Figure 2:** Results obtained from the Specialized Power Systems Breaker model are superimposed with results from Simscape Breaker model. Note that the breaker current is interrupted at zero-crossing after the 100 A trip current setting has been exceeded.

### Conclusion

Using the Simscape Interface blocks of powerlib, it is possible to create any custom block not provided with Specialized Power Systems, in Simscape, and then interface it seamlessly with a Simscape Electrical Specialized Power Systems circuit.

# Boost Converter

This example shows the operation of a boost converter.




### Description

A boost converter is a DC/DC power converter which steps up voltage from its input (source) to its output (load). In continuous conduction mode (current through the inductor never falls to zero), the theoretical transfer function of the boost converter is:

$$Vout/Vin = 1/(1 - D)$$

where $D$ is the duty cycle.

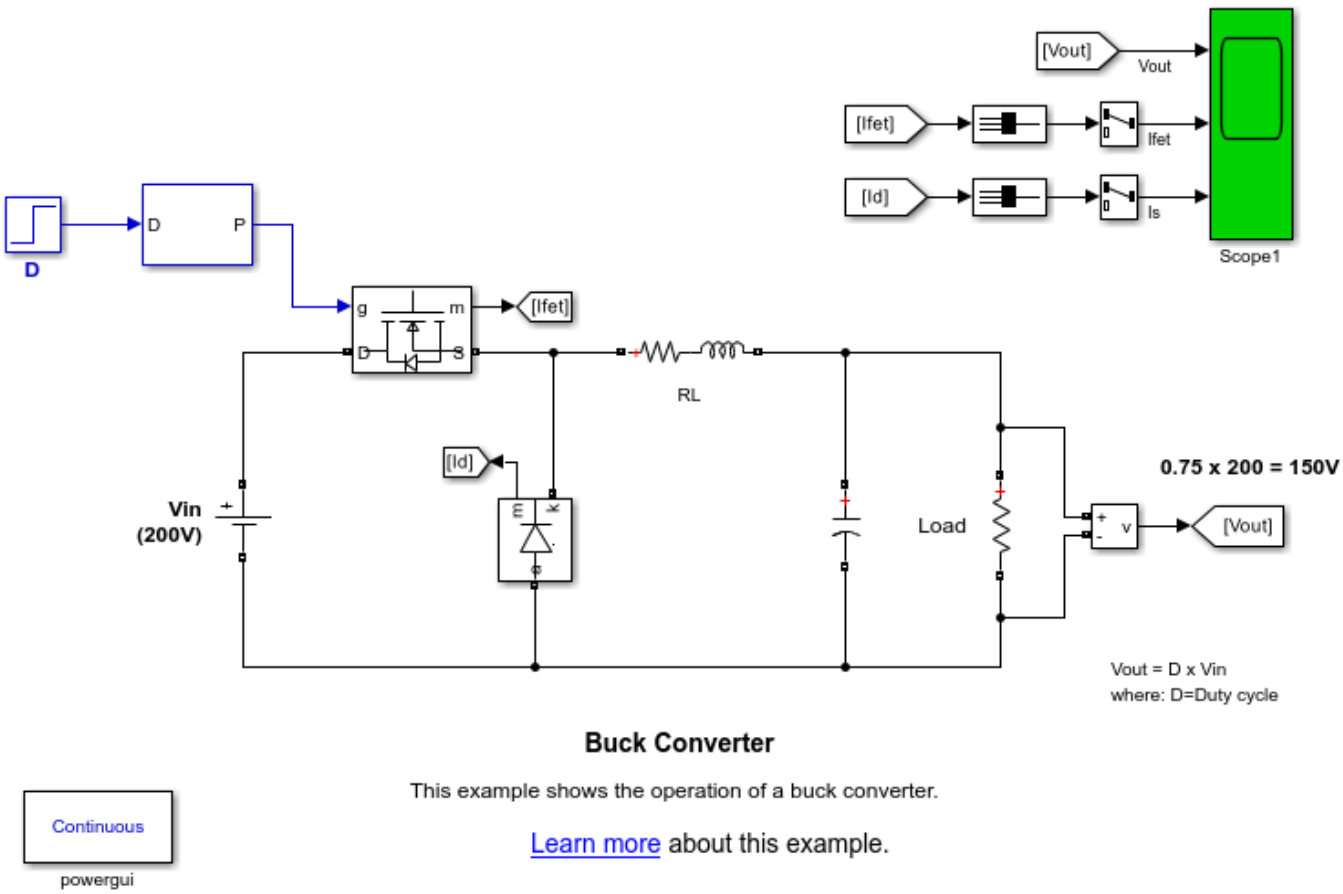
In this example, the converter is feeding an RC load from a 24 V source and the PWM frequency is set to 20 kHz.

### Simulation

Run the simulation and observe waveforms on Scope. Verify that the mean value of the load voltage (Vout) is very close to the theoretical value of: Vout = 24 / (1 - 0.5) = 48 V.

# Buck Converter

This example shows the operation of a buck converter.




### Description

A buck converter is a DC/DC power converter which steps down voltage from its input (source) to its output (load). In continuous conduction mode (current through the inductor never falls to zero), the theoretical transfer function of the buck converter is:

$$Vout/Vin = D$$

where $D$ is the duty cycle.

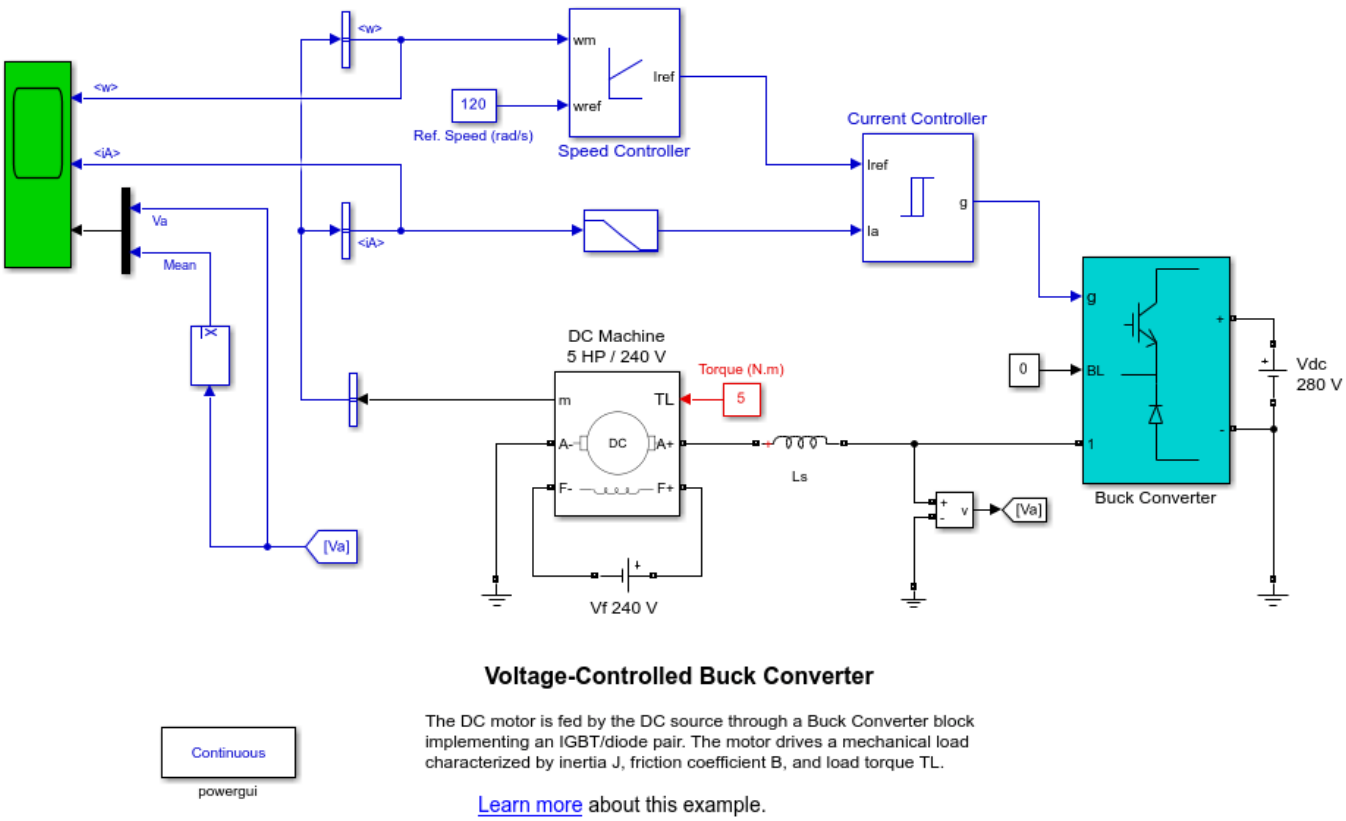
In this example, the converter is feeding an RC load from a 200 V source and the PWM frequency is set to 10 kHz.

### Simulation

Run the simulation and observe waveforms on Scope1. At 3.5 ms, the duty cycle is increased from 0.5 to 0.75. Verify that the mean value of the load voltage (Vout) at the end of the simulation is very close to the theoretical value of: 0.75 * 200 = 150 V.

# Voltage-Controlled Buck Converter

This example shows the operation of a voltage-controlled buck converter.



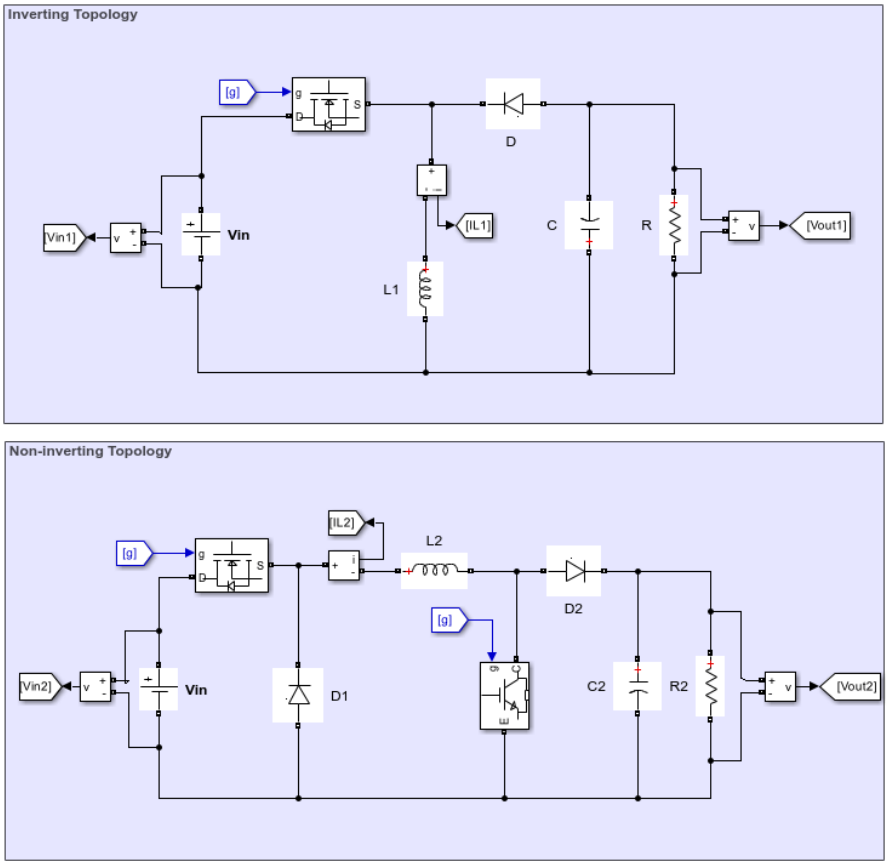

### Description

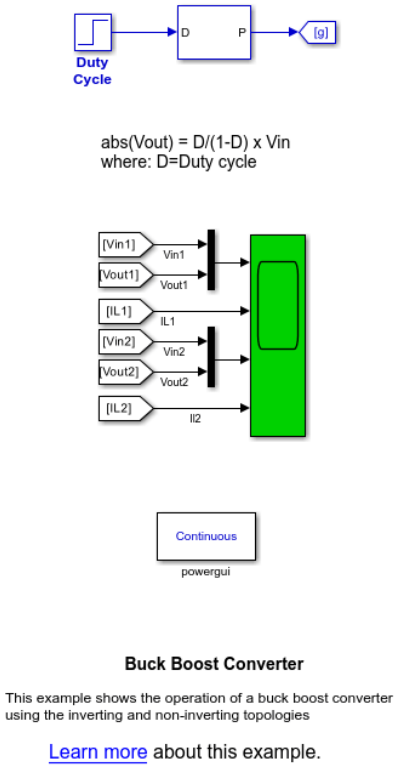
The DC motor is fed by the DC source through a Buck Converter block implementing an IGBT/diode pair. The motor drives a mechanical load characterized by inertia J, friction coefficient B, and load torque TL.

The hysteresis current controller compares the sensed current with the reference and generates the trigger signal for the IGBT thyristor to force the motor current to follow the reference. The speed control loop uses a proportional-integral controller which produces the reference for the current loop.

# Buck Boost Converter

This example shows the operation of buck boost converters using the inverting and non-inverting topologies.




### Description

The buck boost converter is a DC/DC converter with the output voltage magnitude that is either greater than or less than the input voltage magnitude. It is comparable to a flyback converter where an inductor is used in place of a transformer. The theoretical transfer function of the buck boost converter is:

$$abs(Vout) = D/(1-D) * Vin$$

where $D$ is the duty cycle.

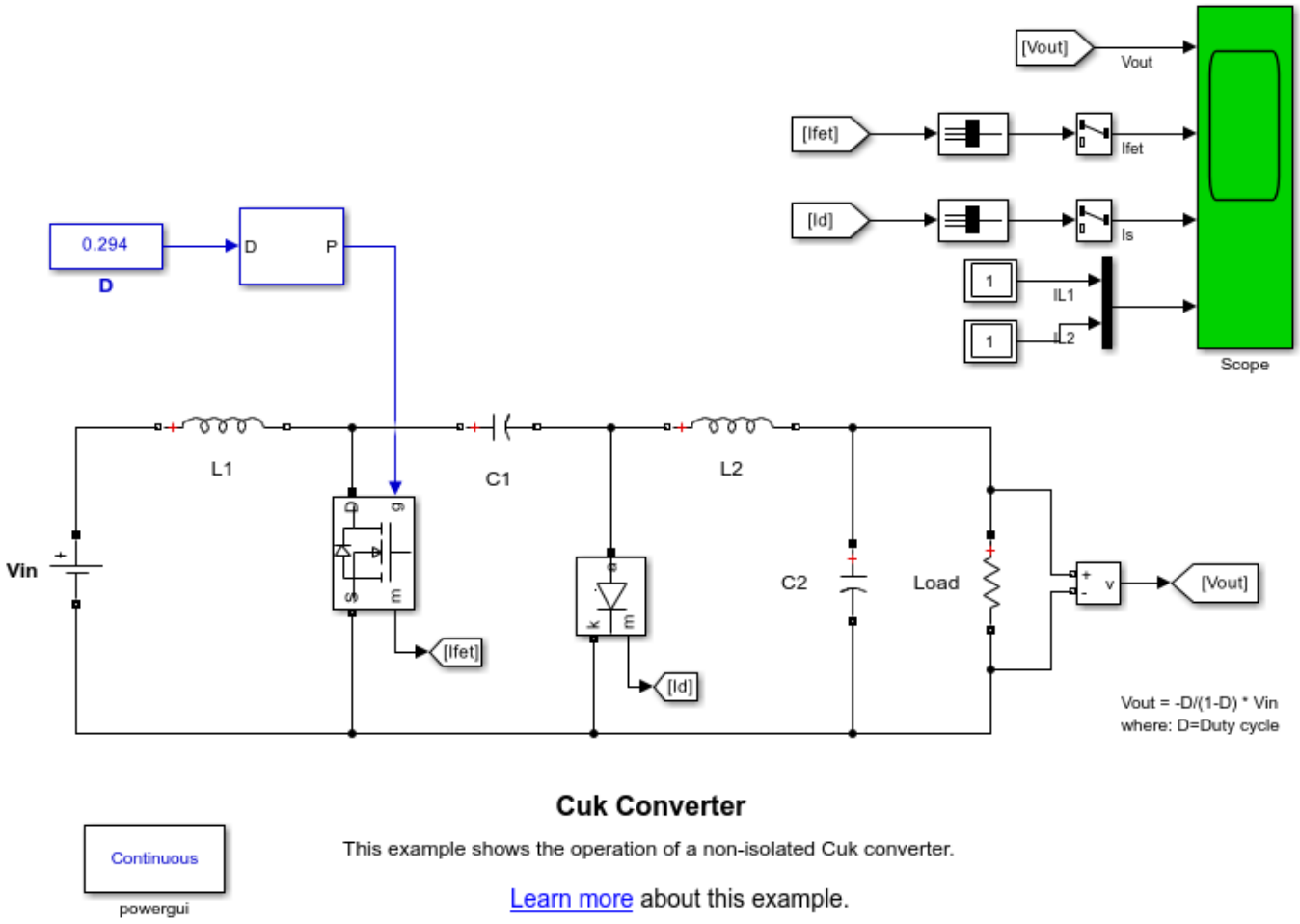
The inverting buck-boost topology produces an output voltage that is of the opposite polarity as the input voltage. The output voltage is determined by the duty cycle of the MOSFET transistor.

The non-inverting topology, also named the 4-switch topology, produces an output voltage that is of the same polarity as the input voltage. In the buck mode, the output voltage is determined by the

operation of the MOSFET and diode D1. In the boost mode, the output voltage is determined by the operation of the IGBT and diode D2.

# Cuk Converter

This example shows the operation of a non-isolated Cuk converter.




### Description

The non-isolated Cuk converter is a DC/DC power converter that, like a buck-boost converter, can produce an output voltage (Vout) magnitude that is either greater or less than the input voltage (Vin) magnitude. However, it is an inverter converter, so the output voltage is of opposite polarity with respect to the input voltage. In the Cuk converter topology, the capacitor C1 acts as the primary means of storing and transferring energy from the input to the output. The advantage of this converter is that both the input current (IL1) and the current feeding the output stage (IL2) are reasonably ripple-free (unlike the buck-boost converter where both these currents are highly discontinuous). Assuming a constant voltage across C1, the theoretical transfer function of the Cuk converter is:

$$Vout/Vin = -D/(1-D)$$

where $D$ is the duty cycle.

In this example, the converter is feeding a 50-W load from a 12 V source and the PWM frequency is set to 50 kHz.

### Simulation

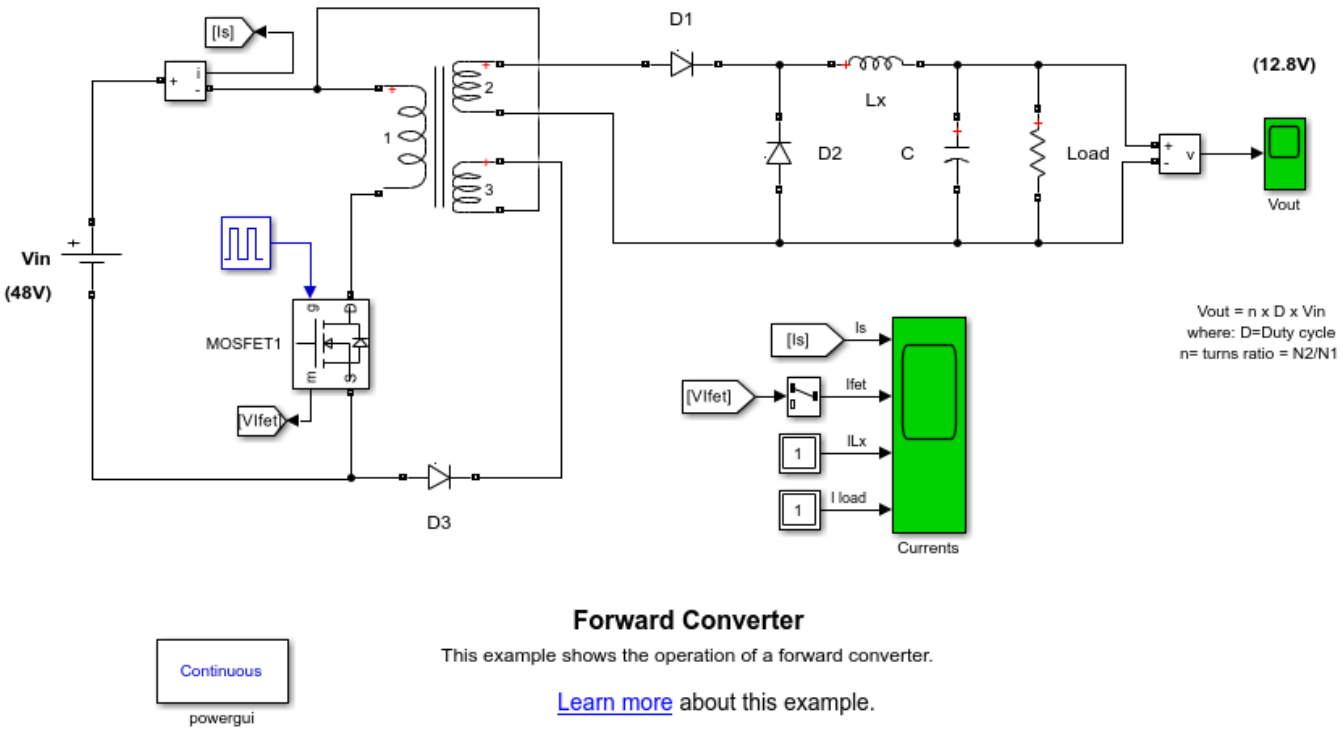
Run the simulation and observe waveforms on Scope. Verify that the mean value of the load voltage (Vout) is very close to the theoretical value of: Vout = -0.294/ (1 - 0.294) * 12 = -5 V.

### Reference

Ned Mohan, Tore M. Undeland, Power Electronics: Converters, Applications, and Design, John Wiley & Sons.

# Forward Converter

This example shows the operation of a forward converter.




## Description

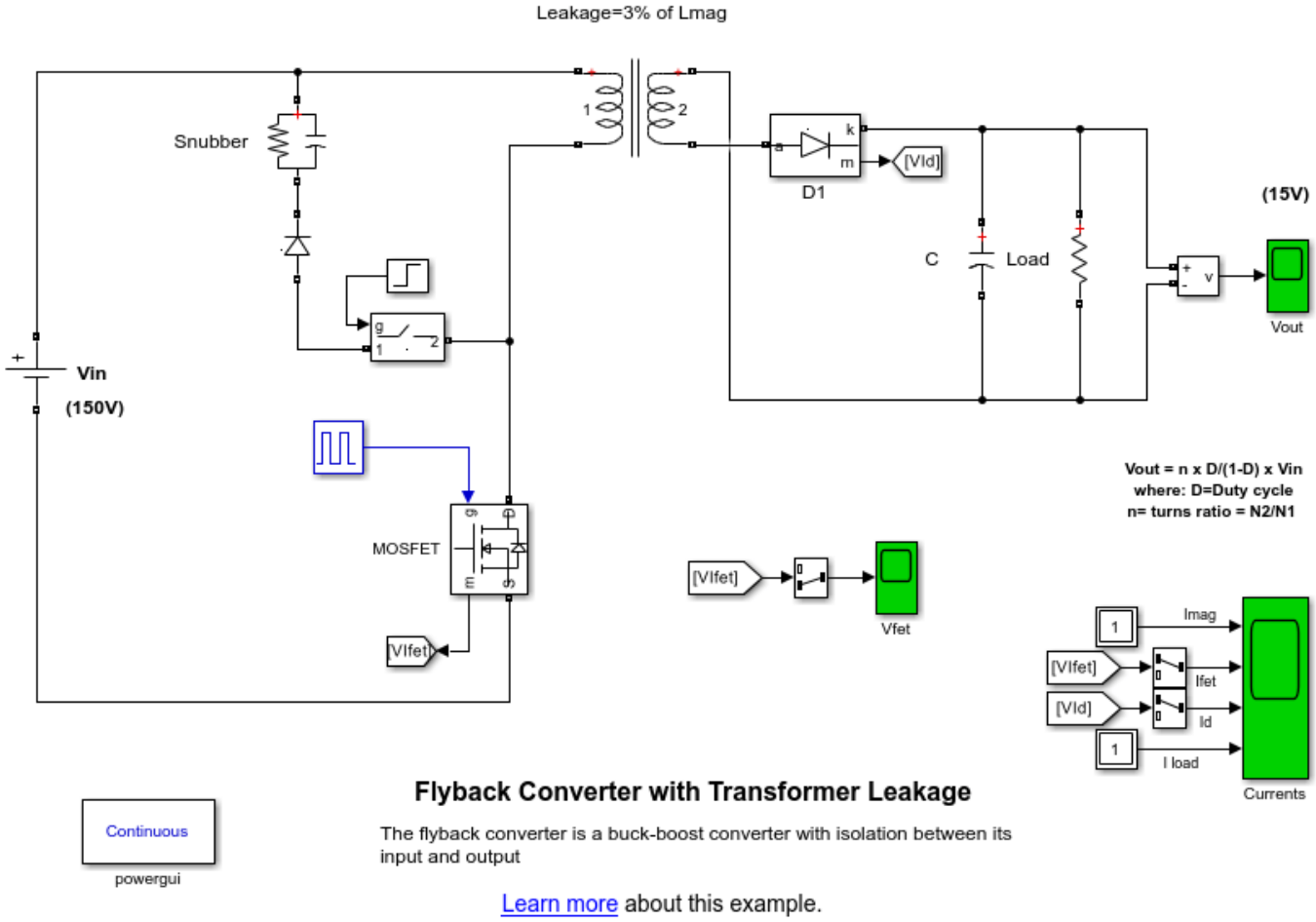
The forward converter is a DC/DC converter that uses a transformer in series with the switching device (MOSFET in this example) to isolate the source from the load and to increase or decrease the input voltage depending on the transformer turns ratio. The theoretical transfer function of the forward converter is:

$$Vout = n * D * Vin$$

where $D$ is the duty cycle, and $n$ is the turns ratio of the transformer $(N_2/N_1)$

# Flyback Converter with Transformer Leakage

This example shows the operation of a flyback converter.




### Description

The flyback converter is a buck-boost converter with isolation between its input and output. The inductor used in the buck-boost configuration is replaced by a transformer for isolation and storing energy before sending to the output via the output capacitor.

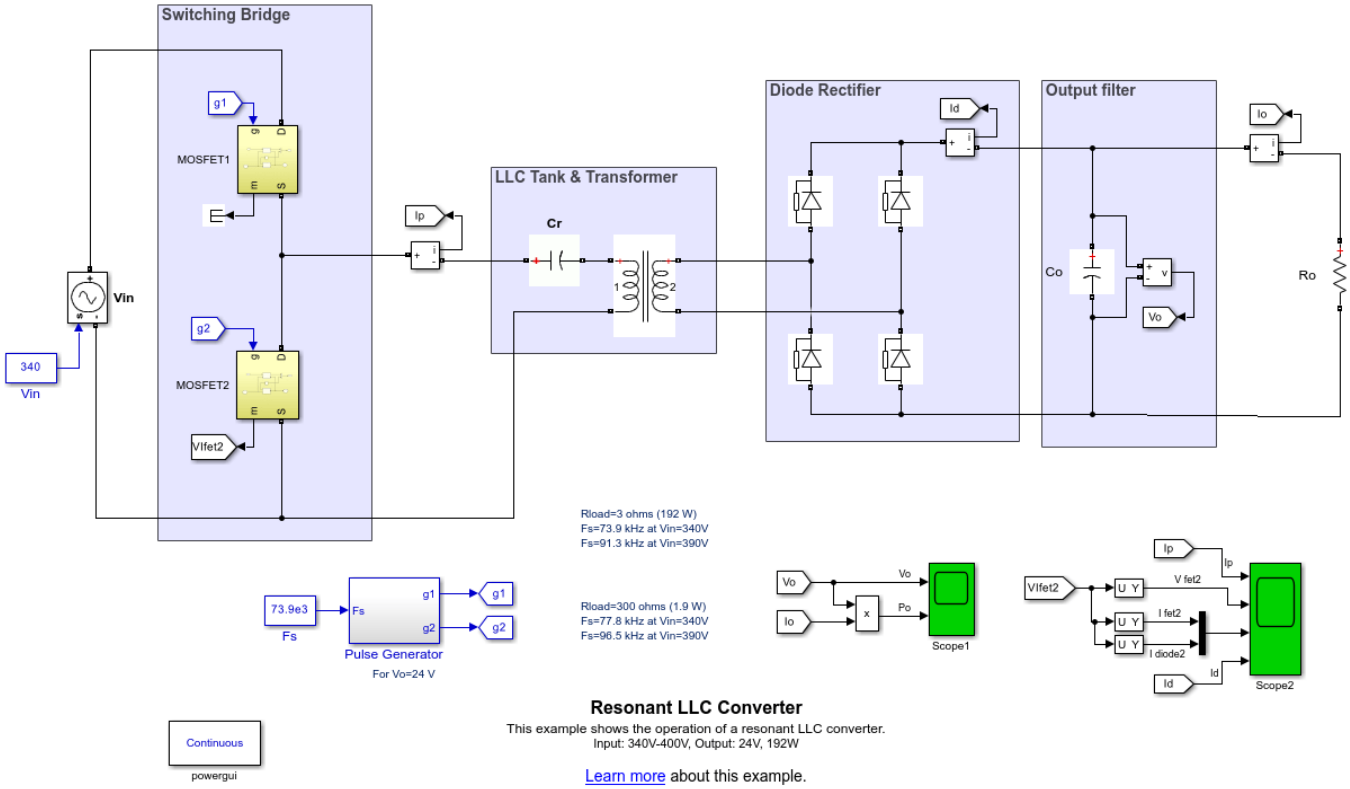
The theoretical transfer function of the flyback converter is:

$$Vout = n * D/(1 - D) * Vin$$

where $D$ is the duty cycle, and $n$, the turns ratio, is equal to $N2/N1$.

# Resonant LLC Converter

This example shows the operation of a resonant LLC converter.




### Description

The LLC converter is a DC/DC converter based on a resonant circuit which allows soft-switching operation. The LLC resonant circuit reduces switching loss through zero-voltage switching (ZVS). Unlike the SLR converter, the LLC converter can keep the output voltage regulated even under light load condition.

The converter consists of a switching bridge (half-bridge MOSFET), an LLC resonant circuit, a full-bridge rectifier and an output filter (Co). The resonant circuit consists of the primary leakage inductance of a transformer (Llkp), its secondary leakage inductance (Llks), its magnetization inductance (Lm) and a capacitor Cr. The resonant frequency of this LLC circuit is given by:

$$w_0 = 1/sqrt(L_r * C_r) = 2 * pi * f_0$$

where $Lr = Llkp + Lm/(N^2 x Llks)$ and $N$ is the transformer winding ratio.

### Simulation

For a given load (3 or 30 ohms) and/or input voltage (340V or 390V), run several simulations using the appropriate switching frequency (Fs) values in order to maintain the output voltage at 24V (signal Vo on Scope1).

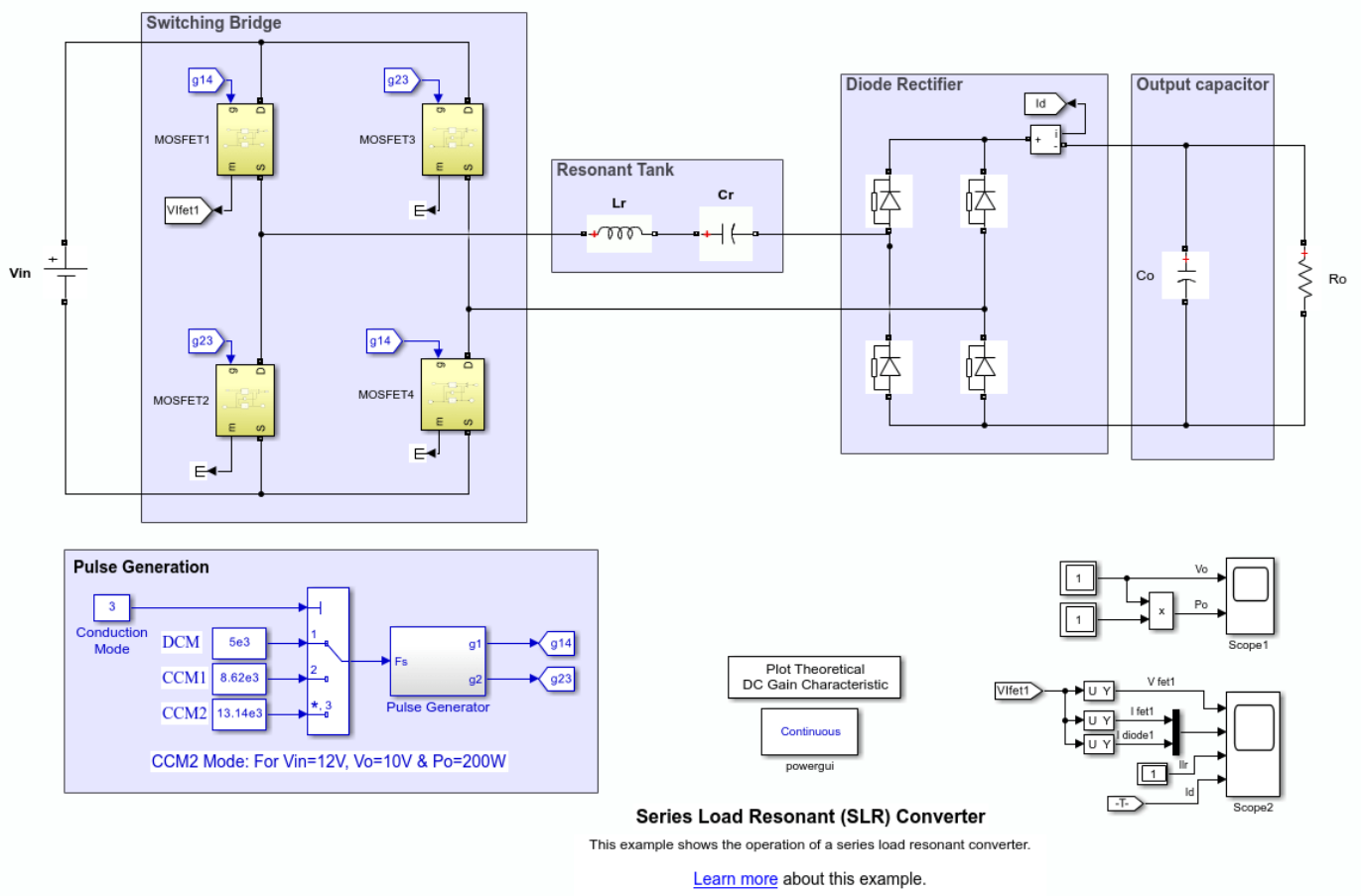
Also, observing waveforms on Scope2, you will see that the current lags the voltage applied to the resonant circuit which allows the MOSFETs to be turned on with zero voltage.

### References

H. Choi; 'Half-Bridge LLC Resonant Converter Design Using FSFR-Series Fairchild Power Switch', Fairchild Semiconductors, AN-4151 S. Abdel-Rahman; "Resonant LLC Converter: Operation and Design", Infineon AN 2012-09

# Series Load Resonant (SLR) Converter

This example shows the operation of a series load resonant converter.




### Description

The Series Load Resonant converter is a DC/DC converter based on a resonant circuit which allows soft-switching operation. In a soft-switching converter, switching occurs when voltage and/or current values are zero, thus significantly improving converter's efficiency. A switch is said to have zero-voltage switching (ZVS) / zero-current switching (ZCS) when the voltage/current is zero as the switch changes state.

The converter consists of a full-bridge MOSFET bridge, a resonant tank represented by the Lr inductor and Cr capacitor block, a full-bridge rectifier and an output filter (Co). The resonant frequency of the tank is given by:

$$w_r = 1/sqrt(L_r * C_r) = 2 * pi * f_r$$

The switching frequency $f_s$ range determines the three possible operating modes of the SLR converter:

- Discontinuous Conduction Mode (DCM), where $f_s < f_r/2$, leading to a ZVS and ZCS at turn-off and a ZCS at turn-on.
- Continuous Conduction Mode (CCM1) where $f_r/2 < f_s < f_r$, leading to a ZVS and ZCS at turn-off.
- Continuous Conduction Mode (CCM2) where $f_s > f_r$, leading to a ZVS at turn-on.

### Simulation

In this example the resonant frequency $f_r$ = 11.25 kHz. Select a switching frequency $f_s$ = 13.14 kHz (conduction mode 3 = CCM2) and run the simulation. Observe waveforms on Scope 1 and Scope 2. For a 0.5 Ohm load, you should get a 10V output for a 12V input voltage. Now try the other conduction modes and/or change the load from 0.5 Ohm to 5 Ohms. You will see that the output will no longer be 10V due to the DC gain characteristic of the resonant circuit for a given load.

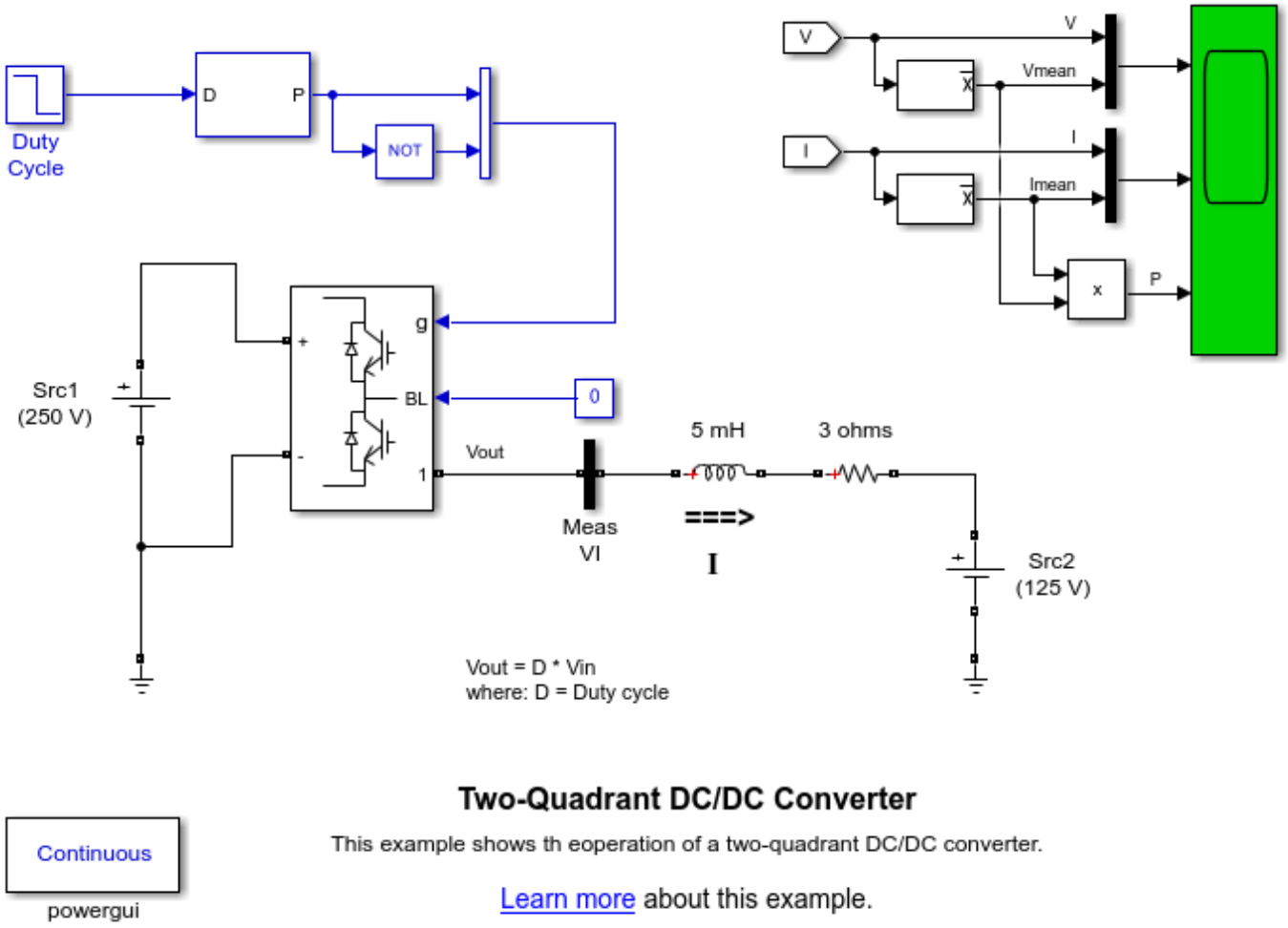
Observe the DC gain characteristic plot (0.5 Ohm traces), you will note that the experimental gain values are slightly shifted relative to the theoretical gain values. This is due to the fact that the output filter Co is not taken into account in the theoretical gain calculation. In addition, as the DC gain characteristic plot suggests (5 Ohms trace), it will be difficult to regulate the output of the SLR converter at light load.

### Reference

T.Taufik, M.McCarthy, S.Watkins, M.Anwari; "Performance Study of Series Loaded Resonant Converter Using Super Barrier Rectifiers", IEEE, TENCON 2009.

# Two-Quadrant DC/DC Converter

This example shows the operation of a two-quadrant DC/DC converter.

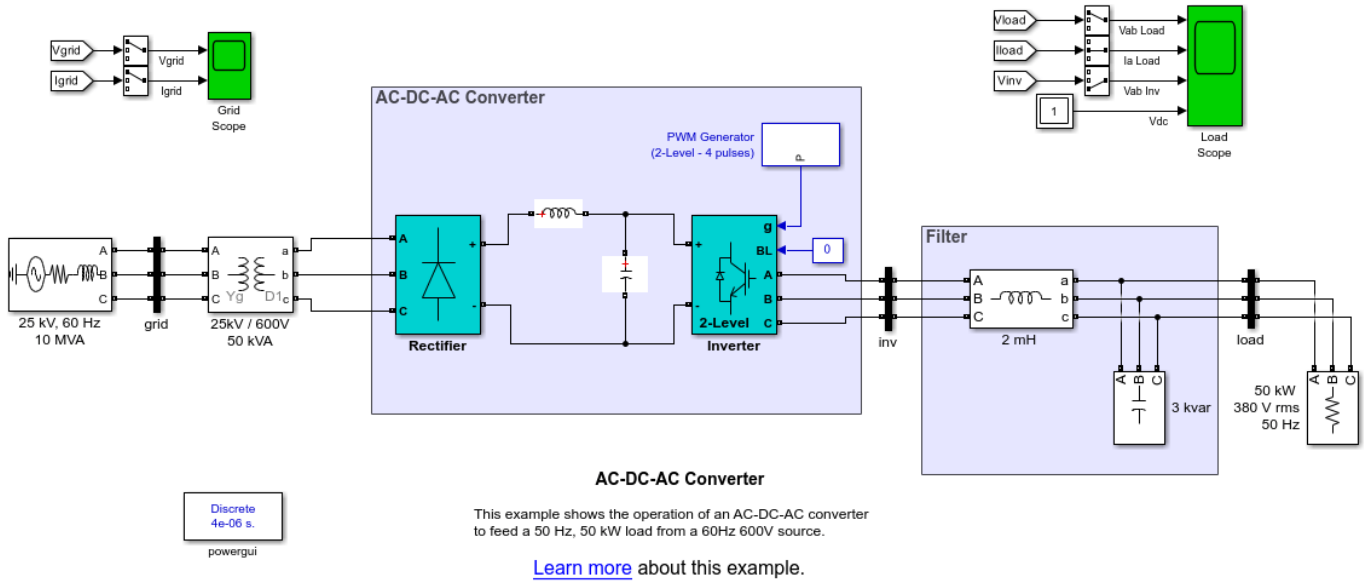



### Description

The example illustrates the operation of a DC-DC converter that converts DC voltage from one voltage level to another one, of same polarity. The converter can operate in both the directions (current can be reversed).

# AC-DC-AC Converter

This example shows the operation of an AC-DC-AC converter.

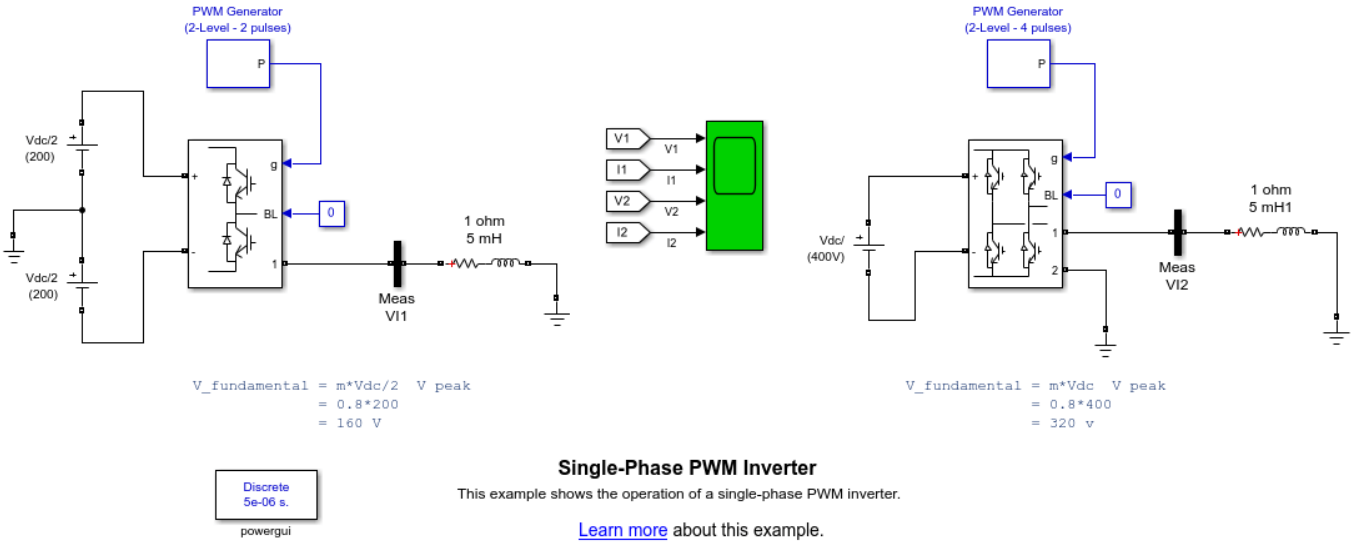



### Description

A 60 Hz, voltage source feeds a 50 Hz, 50 kW load through an AC-DC-AC converter. The 600V, 60 Hz voltage obtained at the secondary of the Wye/Delta transformer is first rectified by a six pulse diode bridge. The filtered DC voltage is applied to an IGBT two-level inverter generating 50 Hz. The IGBT inverter uses Pulse Width Modulation (PWM) at a 2 kHz carrier frequency. The circuit is discretized at a sample time of 4 us.

# Single-Phase PWM Inverter

This example shows the operation of a single-phase PWM inverter.




### Description

The system consists of two independent circuits illustrating single-phase PWM voltage-sourced inverters.

The Half-Bridge Converter block and the Full-Bridge converter block are modeling simplified model of an IGBT/Diode pair where the forward voltages of the forced-commutated device and diode are ignored.

The converters are controlled in open loop with the PWM Generator blocks. The two circuits use the same DC voltage (Vdc = 400V), carrier frequency (1620 Hz) and modulation index (m = 0.8).

In order to allow further signal processing, signals displayed on the Scope block are stored in a variable named ScopeDataForFFT, in structure with time format.

### Simulation

Run the simulation and observe the current into the loads and the voltage generated by the PWM inverters.

Once the simulation is completed, open the Powergui and select FFT Analysis to display the 0 - 5000 Hz frequency spectrum of signals saved in the ScopeDataForFFT structure. The FFT will be performed on a 2-cycle window starting at t = 0.07 - 2/60 (last 2 cycles of recording). Click on Display and observe the frequency spectrum of last 2 cycles.

The fundamental component of V inverter is displayed above the spectrum window. Compare the magnitude of the fundamental component of the inverter voltage with the theoretical values given in the circuit. Compare also the harmonic contents in the inverter voltage.

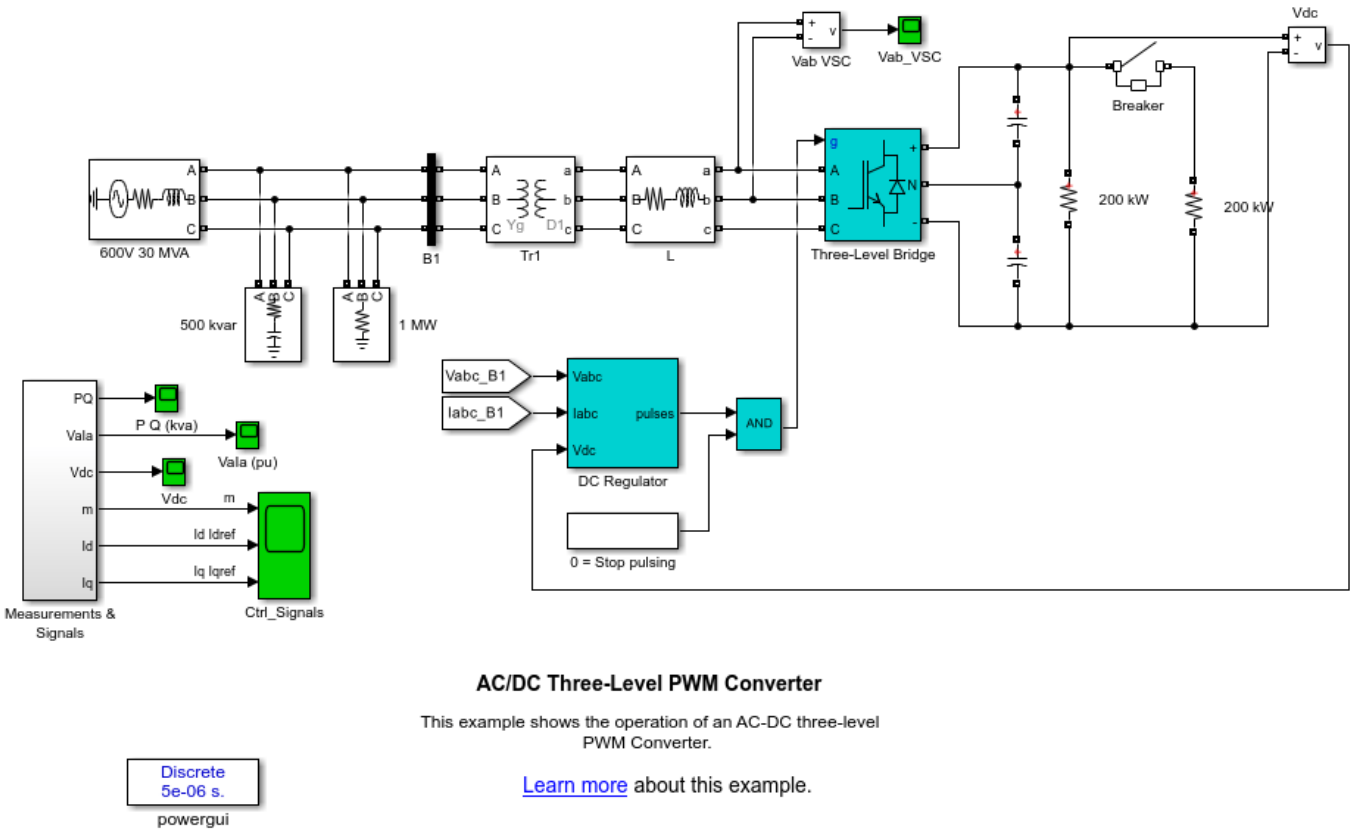
The half-bridge inverter generates a bipolar voltage (-200V or +200V) . Harmonics occur around the carrier frequency (1620 Hz +- k*60 Hz), with a maximum of 103% at 1620 Hz.

The full-bridge inverter generates a monopolar voltage varying between 0 and +400V for one half cycle and then between 0 and -400V for the next half cycle. For the same DC voltage and modulation index, the fundamental component magnitude is twice the value obtained with the half-bridge. Harmonics generated by the full-bridge are lower and they appear at twice the carrier frequency (maximum of 40% at 2*1620+-60 Hz). As a result, the current obtained with the full-bridge is smoother.

# AC/DC Three-Level PWM Converter

This example shows the operation of an AC-DC three-level PWM Converter.

P. Giroux ; G. Sybille Hydro-Quebec (IREQ)




**AC/DC Three-Level PWM Converter**

This example shows the operation of an AC-DC three-level PWM Converter.

Discrete 5e-06 s.
powergui

Learn more about this example.

### Description

1) Converter rating: 500 Volts DC, 500 kW

2) AC Supply: three-phase, 600 V, 30 MVA, 60 Hz system

3) Voltage-sourced Converter (VSC): - Three-level, three-phase IGBT bridge (modeled using the "Three-Level Bridge" block) controlled by a PWM modulator (carrier frequency of 1620 Hz) - DC Link: 2 capacitors of 75000 uF

4) Controller: The DC regulator uses two PI regulators to control the DC voltage while maintaining a unity input power factor for the AC supply.

### Simulation Parameters

- Discrete (no continuous states)

- Two sample times: Ts_Power= 5 us (used by PSB for the simulation of the power system + converters) Ts_Control= 100 us (used for the simulation of the DC regulator)

- Initial conditions are set at the start of the simulation (by automatically loading the file power_3levelVSC.mat prior to the simulation). This file has been generated by running an initial simulation to steady-state for an integer number of cycles of 60 Hz. The final states (both Specialized Power Systems and Simulink® controllers states) were saved in a structure with time, called xInit. This variable was saved in "power_3levelVSC.mat" file.

**Simulation**

- In the simulation (lasting 200 ms), we can observe the following signals: 1) the DC voltage (Vdc Scope) 2) the primary voltage and current (phase A) of the AC supply (VaIa Scope), 3) the device currents of leg A of the IGBT bridge (double-click on the "Measurements & Signals" block to look inside the "Ia_Devices" Scope). 4) the phase-to-phase AC voltage generated by the VSC (Vab_VSC Scope)

- At t=50 ms, a 200-kW load is switched-in. We can see that the dynamic response of the DC regulator to this sudden load variation (200 kW to 400 kW) is satisfactory. The DC voltage is back to 500 V within 1.5 cycle and the unity power factor on the AC side is maintained.

- At t=100 ms, a "Stop Pulsing" signal is activated (pulses normally sent to the converter are blocked). We can see that the DC voltage drops to 315 V. A drastic change in the primary current waveform can also be observed. When the pulses are blocked, the Three-Level Bridge block operation becomes similar to a three-phase diode bridge.

**Regenerate Initial Conditions**

The initial states required to start in steady state have been saved in the "power_3levelVSC.mat" file. When you start simulation, the StartFcn callback (in the Model Properties/Callbacks) automatically loads into your workspace the contents of this .mat file ("xInit" variable).

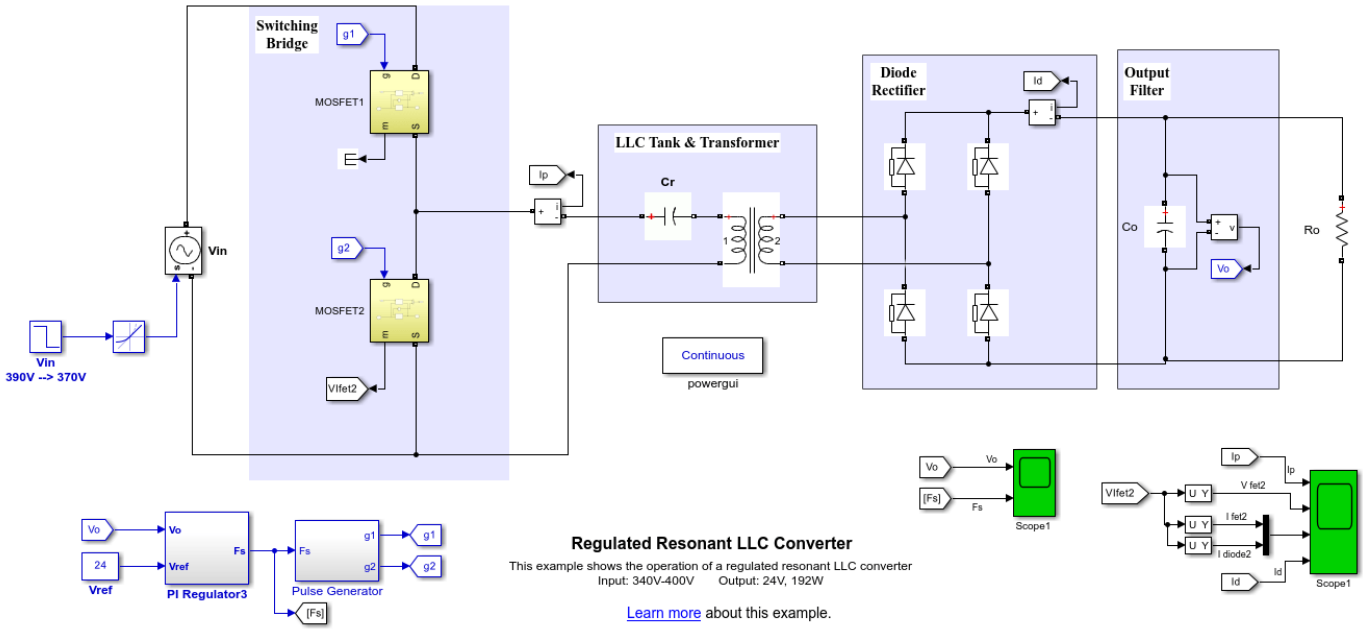
If you modify this model, or change parameter values of power components, the initial conditions stored in the "xInit" variable will no longer be valid and Simulink will issue an error message. To regenerate the initial conditions for your modified model, follow the steps listed below:

1. In the Simulation/Configuration Parameters/Data Import/Export Parameters menu, uncheck the "Initial state" parameter and check the "Final states" parameter.
2. Double click on the Breaker block and temporarily disable the breaker closing by multiplying the "Switching time(s)" parameter by 100.
3. Double click on the Timer block labeled "0=Stop pulsing". Temporarily disable pulse blocking by multiplying the " Time(s)" parameter by 100.
4. Change the Simulation Stop Time to 0.5 second. Note that in order to generate initial conditions coherent with the 60 Hz voltage source phase angle, the Stop Time must an integer number of 60 Hz cycles.
5. Start simulation. When simulation is completed, verify that steady state has been reached by looking at waveforms displayed on the Ctrl_Signals scope. The final states which have been saved in the "xFinal" structure with time can be used as initial states for future simulations. Executing the next two commands copies these final conditions in "xInit" and saves this variable in a new file (myModel_init.mat). `* >> xInit=xFinal; * >> save myModel_init xInit`
6. In the File/Model Properties/Callbacks/StartFcn window, change the name of the initialization file from "power_3levelVSC" to "myModel_init". Next time you start a simulation with this model, the variable xInit saved in the myModel_init.mat file will be loaded in your workspace.

7. In the Simulation/Configuration Parameters menu, check "Initial state".
8. Start simulation and verify that your model starts in steady-state.
9. Double click on the Breaker block and reset the "Switching time(s)" parameter back to 0.05 s (remove the 100 multiplication factor).
10. Double click on the Timer block labeled "0=Stop pulsing". Re-enable the pulse blocking at t=0.1 s by removing the 100 multiplication factor.
11. Change the Simulation Stop time back to 0.2 s.
12. Save your Model.

# Regulated Resonant LLC Converter

This example shows the operation of a regulated resonant LLC converter.




### Description

The LLC converter is a DC/DC converter based on a resonant circuit which allows soft-switching operation.

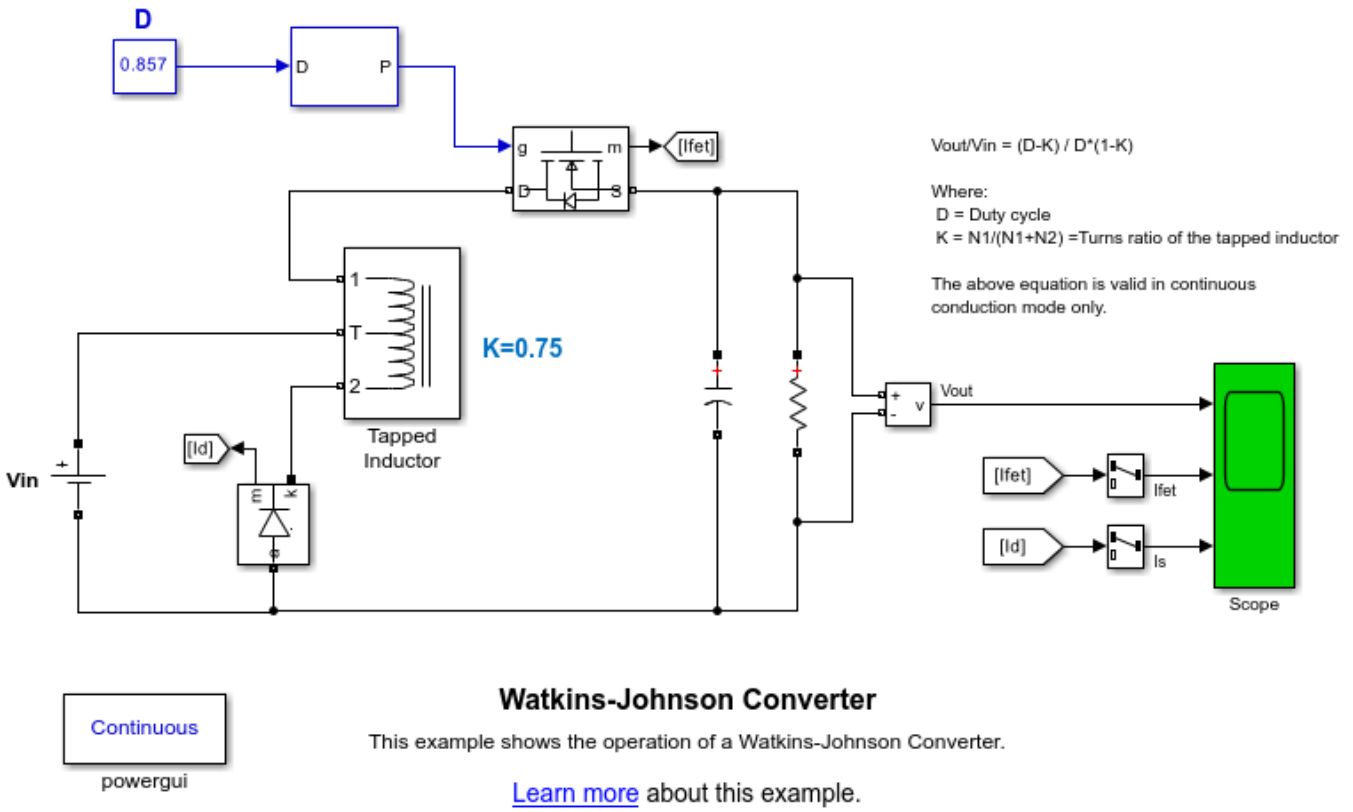
The converter output voltage is regulated using a PI regulator. The output of the regulator is the switching frequency (Fs) which will keep the converter output voltage at the desired voltage set point (Vref). You can change the regulator gains by double-clicking on the PI Regulator block.

### Simulation

Start the simulation and observe on Scope1, the behavior of the regulated converter under a input voltage step (340V to 390V). Also, observing waveforms on Scope2, you will see that the current lags the voltage applied to the resonant circuit which allows the MOSFETs to be turned on with zero voltage.

# Watkins-Johnson Converter

This example shows the operation of a Watkins-Johnson Converter.




### Description

The Watkins-Johnson (WJ) converter employs a tapped-inductor giving an extra-degree of freedom compared to the classical buck converter. The theoretical transfer function of the WJ converter is:

$$Vout/Vin = D - K/D * (1 - K)$$

where $D$ is the Duty cycle and $K = N1/(N1 + N2)$ is the turns ratio of the tapped inductor,

Note that the above equation is valid in continuous conduction mode only. For a giving conversion ratio, the WJ converter has the advantage of employing a larger duty cycle than the traditional classical buck converter. In our example, the tapped inductor is built using the SPS mutual inductance. By double-clicking on the Tapped Inductor block, you can specify the value in henrys and the turns ratio of the tapped inductor.

### Simulation

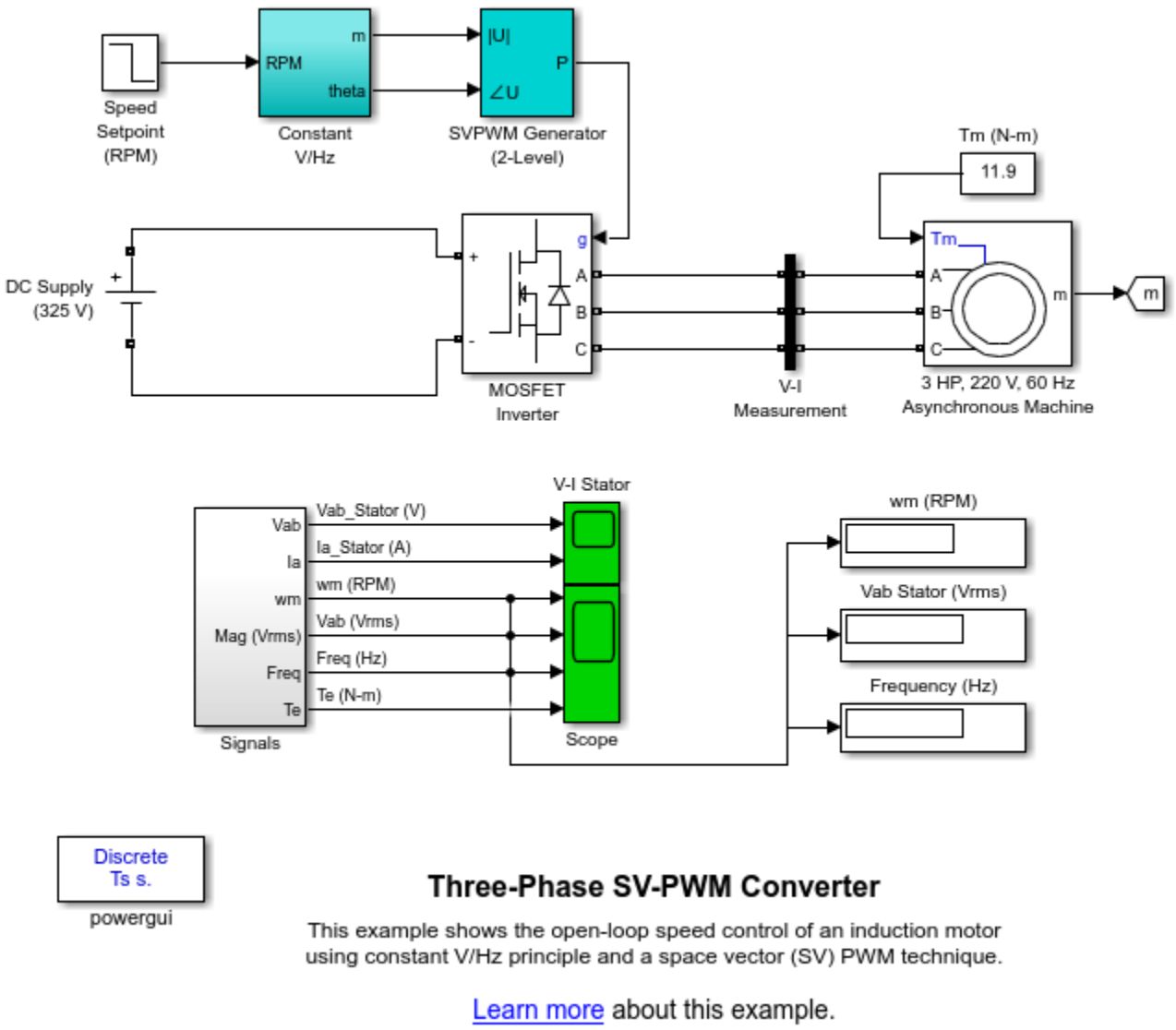
Run the simulation and observe waveforms on Scope. The mean value of the load voltage (Vout) should be very close to the theoretical value of:

```
Vout = 24 * (0.857-0.75) /(0.855*(1-0.75) ) = 12.01 V
```

# Three-Phase SV-PWM Converter

This example shows the open-loop speed control of an induction motor using constant V/Hz principle and a space vector (SV) PWM technique.

Pierre Giroux, Hydro-Quebec (IREQ)




### Description

A 3-phase squirrel-cage motor rated 3 HP, 220 V, 60 Hz, 1725 rpm is fed by a 3-phase MOSFET inverter connected to a DC voltage source of 325 V. The inverter is modeled using the "Universal Bridge" block and the motor by the "Asynchronous Machine" block. Its stator leakage inductance Lls is set to twice its actual value to simulate the effect of a smoothing reactor placed between the

inverter and the machine. The load torque applied to the machine's shaft is constant and set to its nominal value of 11.9 N.m.

The firing pulses to the inverter are generated by the "Space-Vector PWM modulator" block of the SPS library. The chopping frequency is set to 1980 Hz and the input reference vector to "Magnitude-Angle".

Speed control of the motor is performed by the "Constant V/Hz" block. The magnitude and frequency of the stator voltages are set based on the speed setpoint. By varying the stator voltages magnitude in proportion with frequency, the stator flux is kept constant.

**Simulation**

Start the simulation. Since the initial states have been automatically loaded, the simulation should start in steady-state. The initial motor speed should be 1720 RPM and the rms value of the stator voltages should be 220V@60Hz.

At 0.1s, the speed setpoint is changed from 1725 to 1300 RPM. You can observe the system dynamic looking inside Scope 1. When the motor reaches a constant speed of 1275 RPM, the stator voltage rms value is down to 165.8V and the frequency to 45.2 Hz.

Stator voltage (phase AB) and phase A current waveforms can be observed in the "V-I Stator" Scope. You can do a FFT of these two quantities using the powergui FFT Analysis.

**Regenerate Initial Conditions**

The initial states required to start this model in steady state with a 1725 rpm reference speed and a 11.9 N.m load torque have been saved in the "power_svpwm_init.mat" file. When you open this model, the InitFcn callback (in the Model Properties/Callbacks) automatically loads into your workspace the contents of this .mat file ("xInitial" variable).

If you modify this model, or change parameter values of power components, the initial conditions stored in the "xInitial" variable will no longer be valid and Simulink® will issue an error message. To regenerate the initial conditions for your modified model, follow the steps listed below:

1. In the Simulation/Configuration Parameters/Data Import/Export Parameters menu, uncheck the "Initial state" parameter.

2. Double click on the Step block labeled "Speed Setpoint (RPM)" and temporarily disable the change of reference speed by multiplying the Step time by 100.

3. Change the Simulation Mode from "Normal" to "Accelerator".

4. Start simulation. When simulation is completed, verify that steady state has been reached by looking at waveforms displayed on the scopes. The final states which have been saved in the "xFinal" structure with time can be used as initial states for future simulations. Executing the next two commands copies these final conditions in "xInitial" and saves this variable in a new file (myModel_init.mat).

```
>> xInitial=xFinal;
```

```
>> save myModel_init xInitial
```

5. In the File -> Model Properties -> Callbacks -> InitFcn window, change the name of the initialization file from "power_svpwm_init" to "myModel_init". Next time you open this model, the variable xInitial saved in the myModel_init.mat file will be loaded in your workspace.

6. In the Simulation -> Configuration Parameters menu, check "Initial state".

7. Start simulation and verify that your model starts in steady-state.

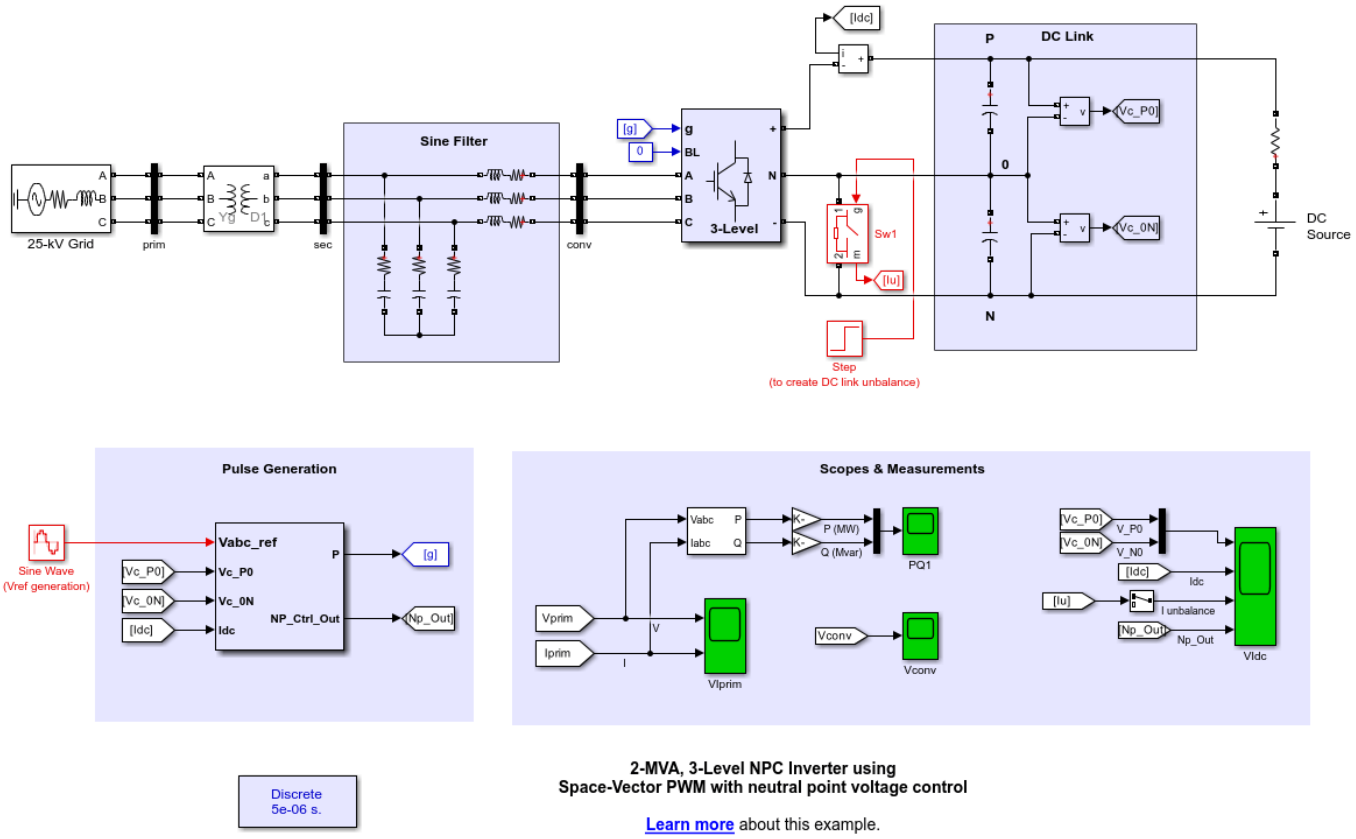
8. Double click on the Step block labeled "Speed Setpoint (RPM)" and re-enable the change of reference speed at t=0.1 s (remove the 100 multiplication factor in the Step time).

9. Change the Simulation Mode back to Normal.

10. Save your model.

# Three-Level NPC Inverter Using Space-Vector PWM with Neutral-Point Voltage Control

This example shows the operation of a 2-MVA, 3-Level NPC inverter using Space-Vector Pulse-Width-Modulation (SVPWM) technique with neutral-point voltage control.




## Description

From an ideal 2400-Volt DC source, a 2-MVA, three-phase 3-level inverter delivers power to a 25-kV distribution system. A sine filter is used on the secondary side of the distribution transformer in order mitigate high-frequency harmonics generated by the inverter. The Sw1 ideal switch is used to create a major unbalance on the DC link.

The inverter is controlled in open-loop using an SVPWM 3-Level Generator block. The PWM switching frequency is set to 1620 Hz and the neutral-point voltage control gain is set to 0.04.

## Simulation

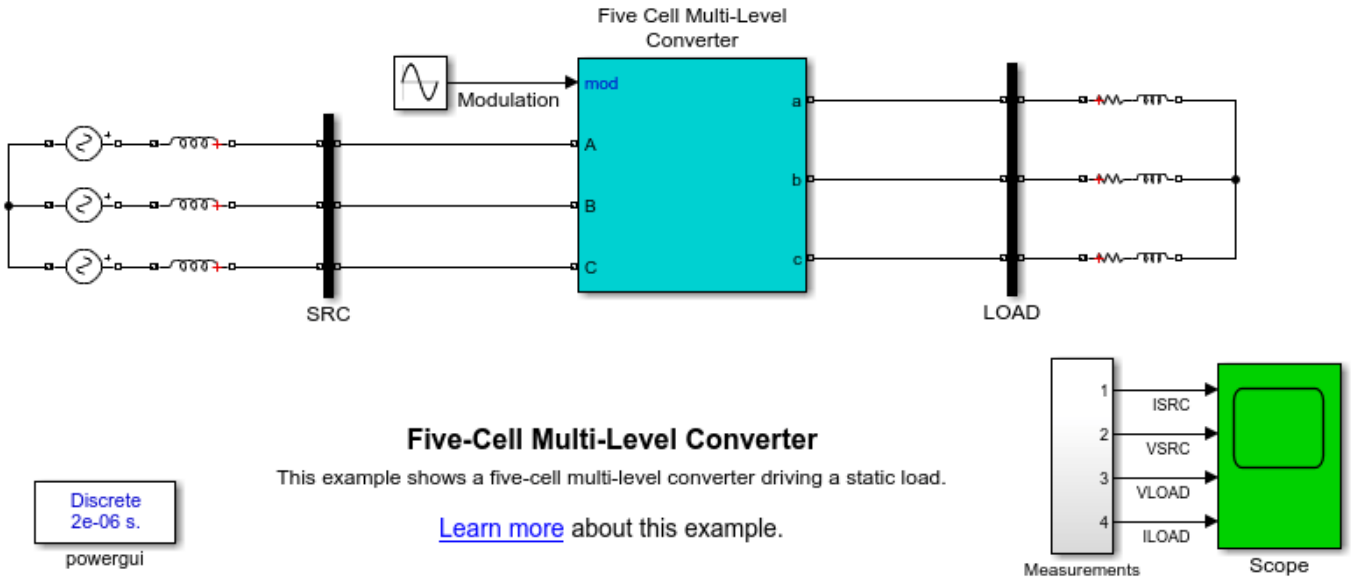
Run the simulation and observe waveforms on the VIdc Scope. We will notice that the neutral-point voltage control perform correctly when a large unbalance is created at 0.2 sec. Now double-click on the SVPWM generator block and set the proportional gain of the neutral-point voltage control to zero.

Re-run the simulation and note that without neutral-point voltage control, the DC link unbalance increases rapidly.

# Five-Cell Multi-Level Converter

This example shows a five-cell multi-level converter driving a static load.

Graham Dudgeon, Senior Consultant, The MathWorks, Inc.




### Description

The system consists of a three-phase five-cell multi-level converter (MLC) constructed from cells containing diode rectifiers and IGBT H-bridge inverters. The MLC is supplied by an ideal 60Hz three-phase source and drives a static resistive/inductive load at 100Hz. The architecture of the converter is taken from [1]. The zig-zag input transformers for each cell are phase-shifted by 15 degrees relative to each other to ameliorate source harmonics below the 23rd harmonic. The H-bridge PWM carrier waves are phase shifted by 45 degrees relative to each other to produce the output voltage 'stacking' effect. The modular nature of a MLC is clearly seen by looking under the component mask.

### Simulation

Start the simulation. Observe the voltage stacking effect from Scope block. Using the FFT Analysis tool in the powergui, confirm that the source current ISRC has no significant harmonics under the 23rd harmonic.

You can adjust the modulation waveforms to change output voltage amplitude and frequency.

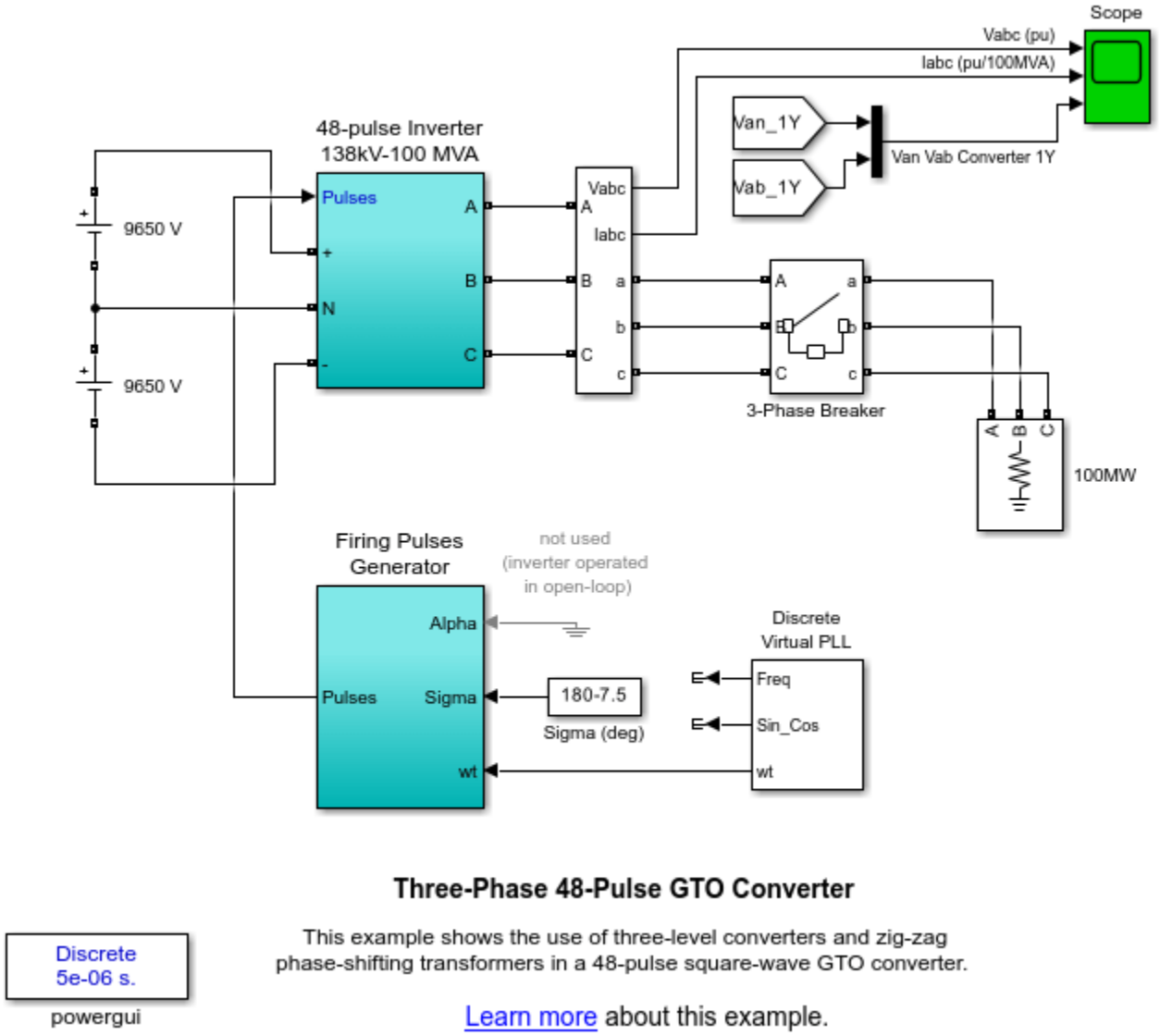
### References

[1] Wen, J. and Smedley, K.M., 'Synthesis of Multi-Level Converters Based on Single- and/or Three-Phase Converter Building Blocks', IEEE® Transactions on Power Electronics, Vol. 23, No. 3, May 2008.

# Three-Phase 48-Pulse GTO Converter

This example shows the use of three-level converters and zig-zag phase-shifting transformers in a 48-pulse square-wave GTO converter.

P. Giroux and G. Sybille (Hydro-Quebec)




### Description

In this example, ideal switches and zig-zag phase shifting transformers are used to build a GTO-type 100 MVA, 138 kV voltage source inverter. This type of converter is used in high-power (up to 200 MVA) Flexible AC Transmission Systems (FACTS) which are used to control power flow on transmission grids. It can be used, for example, to build a model of shunt or series static compensator

(STATCOM or SSSC) or, using two such converters, a combination of shunt and series devices known as Unified Power Flow Controller (UPFC).

The inverter described in this example is a harmonic neutralized, 48-pulse GTO converter described in reference. It consists of four 3-phase, 3-level inverters and four phase-shifting transformers. Open the "48-pulse inverter" subsystem. Notice that the DC bus (Vdc = +/-9650 V) is connected to the four 3-phase inverters.The four voltages generated by the inverters are applied to secondary windings of four zig-zag phase-shifting transformers connected in Wye (Y) or Delta (D). The four transformer primary windings are connected in series and the converter pulse patterns are phase shifted so that the four voltage fundamental components sum in phase on the primary side.

Each 3-level inverter generates three square-wave voltages which can be +Vdc, 0, -Vdc. The duration of the +Vdc or -Vdc level (Sigma) can be adjusted between 0 and 180 degrees from the Sigma input of the Firing Pulse Generator block. Each inverter uses a Three-Level Bridge block where the specified power electronic devices are Ideal Switches. In this model, each leg of the inverter uses 3 ideal switches to obtain the 3 voltage levels (+Vdc, 0, - Vdc). This simple model simulates the behavior of a physical inverter where each leg consists of 4 GTOs, 4 antiparallel diodes and 2 neutral clamping diodes. Despite this simplified switch arrangement, the model still requires 4 pulses per arm as in the physical model. The pulse pattern sent to each leg of a 3-phase inverter is described inside the Firing Pulse Generator.

You can also select GTO/Diodes pairs instead of Ideal Switches as power electronic devices. It would allow you to specify forward voltage drops for GTOs and diodes and to observe currents flowing in GTOs and diodes by means of the Multimeter block.

The phase shifts produced by the secondary delta connections (-30 degrees) and by the primary zig-zag connections (+7.5 degrees for transformers 1Y and 1D, and -7.5 degrees for transformers 2Y and 2D) allows to neutralize harmonics up to 45th harmonic, as explained below:

The 30-degree phase-shift between the Y and D secondaries cancels harmonics 5+12n (5, 17, 29, 41, ...) and 7+12n (7, 19, 31, 43, ...). In addition, the 15-degree phase shift between the two groups of transformers (1Y and 1D leading by 7.5 degrees, 2Y and 2D lagging by +7.5 degrees) allows cancellation of harmonics 11+24n (11, 35, ...) and 13+24n (13, 37, ...). Considering that all the 3n harmonics are not transmitted by the Y and D secondaries, the first harmonics which are not cancelled by the transformers are 23rd, 25th, 47th and 49th. By choosing an appropriate conduction angle for the 3-level inverters (sigma = 180 - 7.5 = 172.5 degrees), the 23rd and 25th can be minimized. The first significant harmonics are therefore the 47th and 49th. This type of inverter generates an almost sinusoidal waveform consisting of 48-steps.

The inverter is operated in open loop at constant DC voltage, therefore, the voltage angle (alpha) which is normally kept close to zero is not used. You can look at the STATCOM (Detailed Model) example that shows the operation of a 48-pulse GTO STATCOM in closed-loop.

Initially, the inverter operates at no load. Then, at t = 0.025 s, a 100 MVA resistive load is connected at the 138-kV terminals.

**Simulation**

Run the simulation and observe the following waveforms on the Scope block:

Voltages generated by the inverter (trace 1), load currents (trace 2), phase-neutral voltage and phase-phase voltage of one of the four inverters (1Y) superimposed on trace 3. When the inverter is operating at no load, you can observe the three 48-step voltage waveform. When the load is switched

on the voltage becomes smoother because harmonics are filtered by the transformer leakage reactances.

Once the simulation is completed, open the Powergui and select "FFT Analysis" to display the 0-4000 Hz frequency spectrum of signals saved in the two "psb48pulse_str" structure. Select signal labeled 'Vabc (pu)'. The FFT will be performed on a 1-cycle window of phase A voltage starting at t = 0.025-1/60 s (inverter operating at no load). Click on Display and observe the frequency spectrum.

The fundamental component of Voltage (in pu) as well as THD are displayed above the spectrum window. Notice that the first significant harmonics are 47th and 49th (approx. 2%). Notice also that 23rd and 25th are reduced below 0.3%. In order to appreciate the efficiency of harmonic neutralization, you can also observe the frequency spectrum of phase-phase voltage generated by each individual inverter. Select input labeled "Van Vab Converter 1Y" and signal number 2 and click on Display. Observe that THD in the 0 - 4000 Hz frequency range is 25%.

You can also run another simulation by specifying different values of sigma at the input of the pulse generator. You can verify that, in order to cancel a particular harmonic n in the phase-phase voltage of each individual converter, the Sigma value (in degrees) is given by:

$$Sigma = 180 * (1 - 1/n)$$

Verify also that choosing Sigma = 180 degrees is equivalent to using 2-level converters and that the voltage waveform is degraded to 24 pulses.

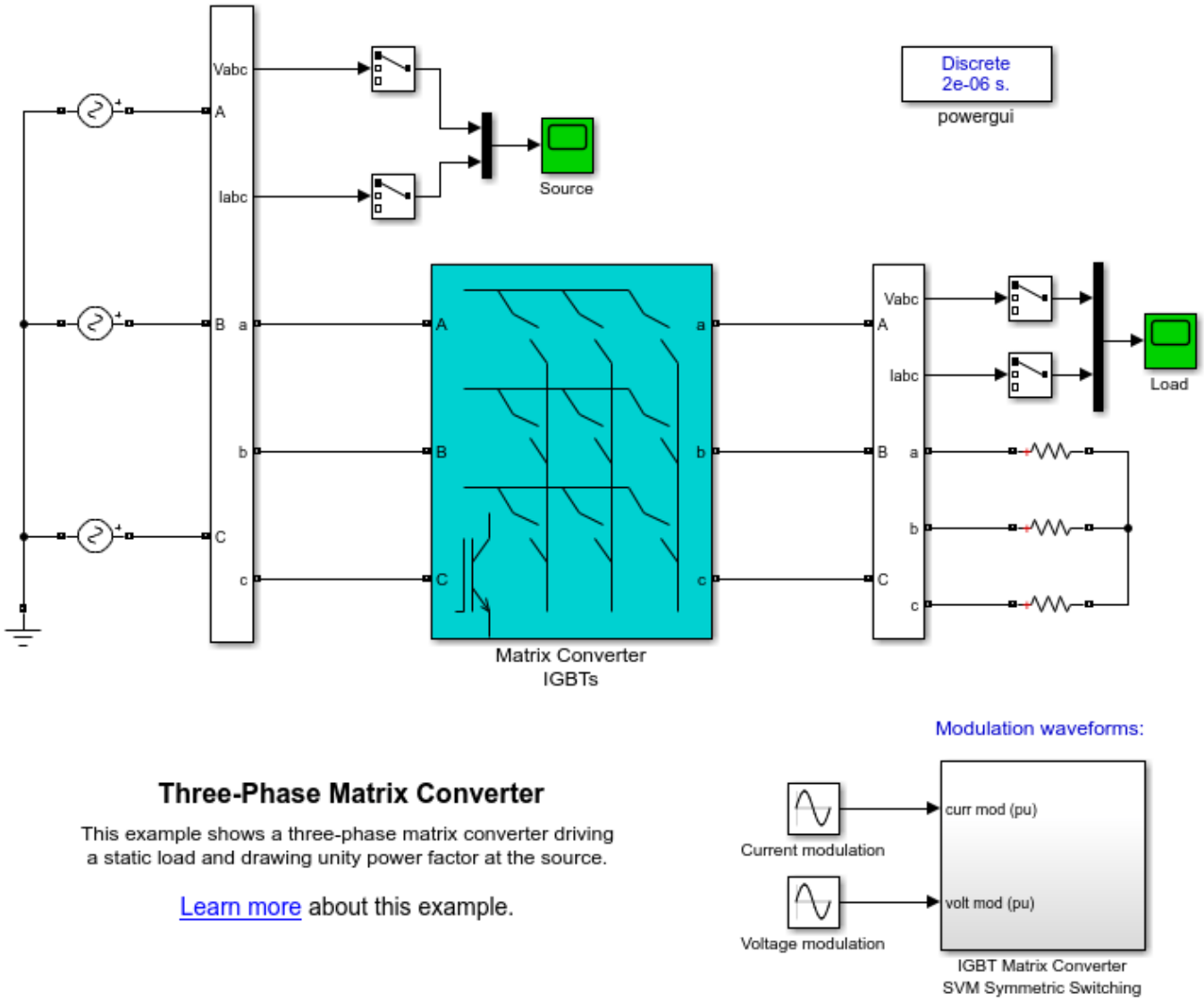
### Reference

Narain G. Hingorani and Laszlo Gyuyi, "Understanding FACTS", IEEE® Press, 2000

# Three-Phase Matrix Converter

This example shows a three-phase matrix converter driving a static load and drawing unity power factor at the source.

Graham Dudgeon, Senior Consultant, The MathWorks, Inc.




### Description

The system consists of a three-phase matrix converter (MC) constructed from 9 back-to-back IGBT switches. The MC is supplied by an ideal 60Hz three-phase source and drives a static resistive load at 60Hz. The switching algorithm is based on an indirect space-vector modulation described in [1] which considers the MC as a rectifier and inverter connected via a DC link with no energy storage. Indirect space-vector modulation allows direct control of input current and output voltage and hence allows

the power factor of the source to be controlled. The switching algorithm utilizes a symmetric switching sequence described in [2].

No filters are included in the model so that the construction of the waveforms can be clearly seen.

### Simulation

Start the simulation. Observe from Scope 'Source' that the MC draws unity power factor.

### References

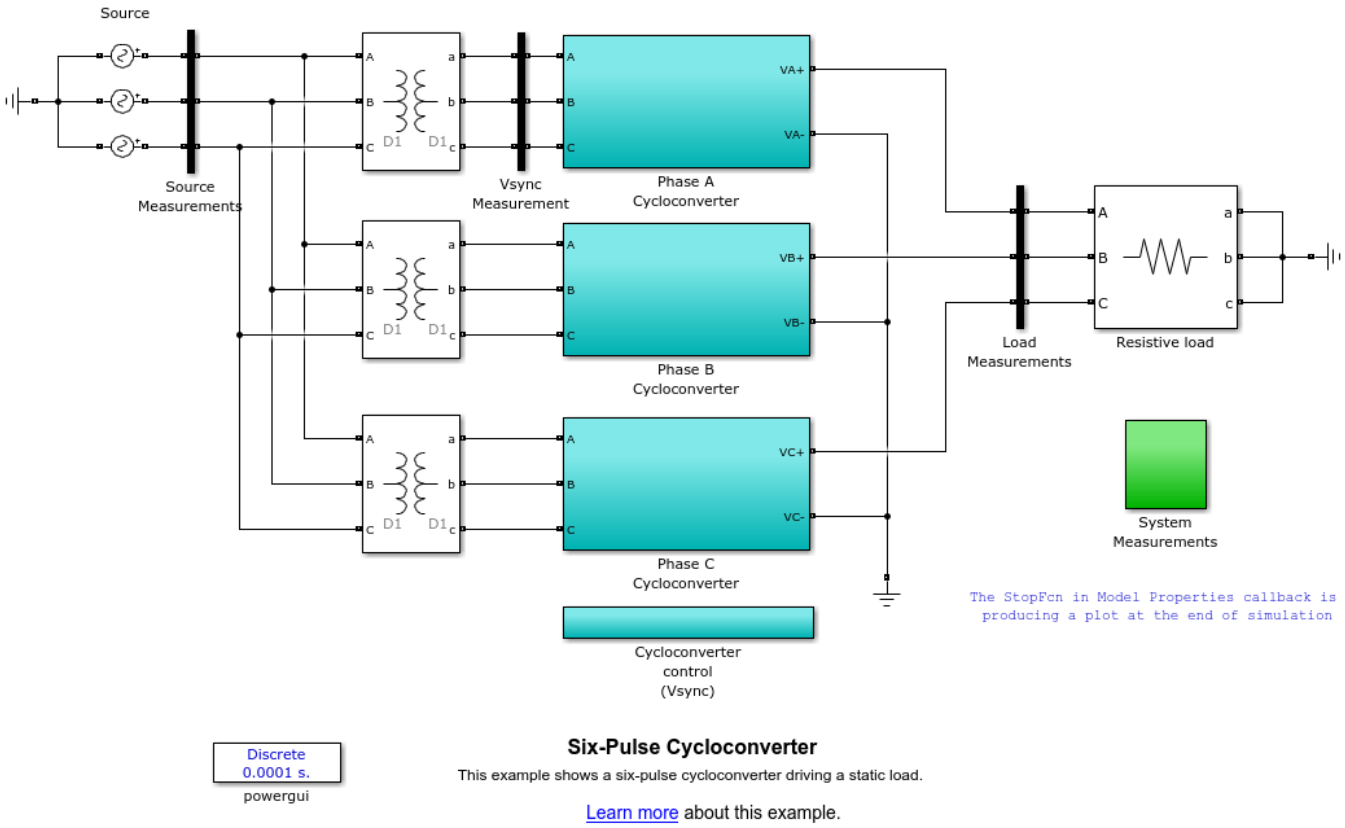
[1] Huber, L. and Borojevic, D., 'Space Vector Modulated Three-Phase to Three-Phase Matrix Converter with Input power Factor Correction', IEEE® Transactions on Industry Applications, Vol. 31, No. 6, November/December 1995.

[2] Prasad, V.H., 'Analysis and Comparison of Space Vector Modulation Schemes For Three-Leg and Four-leg Voltage Source Inverters', Masters Thesis, Electrical Engineering, Virginia Tech, May 15, 1997.

# Six-Pulse Cycloconverter

This example shows a six-pulse cycloconverter driving a static load.

Graham Dudgeon, Senior Consultant, The MathWorks, Inc. Igor Braverman, Senior Consultant, The MathWorks, Inc.




### Description

The system consists of three-phase to single-phase cycloconverter modules suitably arranged to implement a three-phase to three-phase architecture. Each module has a positive and negative cycle thyristor bridge connected in anti-parallel. To avoid circulating currents when one bridge is conducting, the anti-parallel bridge is blocked. A 5 degree 'dead zone' exists between the positive and negative cycle commutation. You can modify the dead zone value in **File -> Model Properties -> Callbacks -> InitFcn**. The output voltage waveforms are constructed by suitably modifying the firing angle of the thyristor bridges.

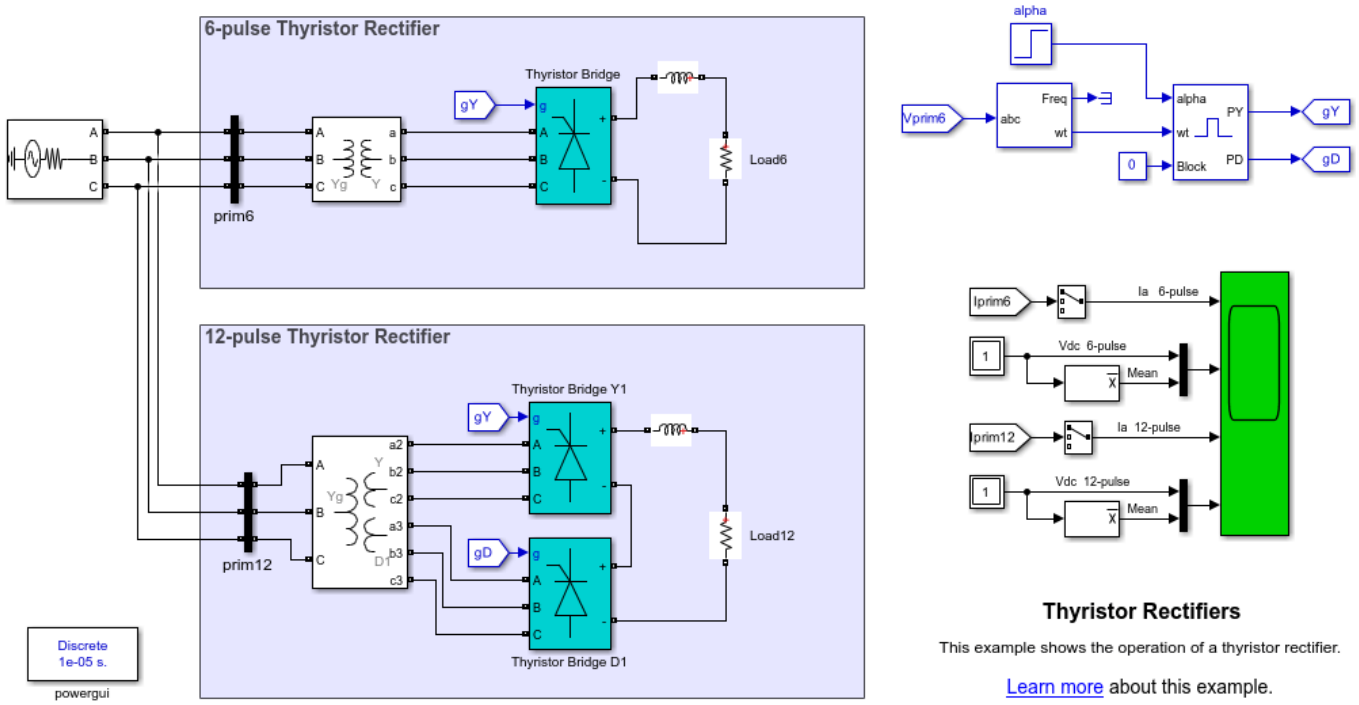
### Simulation

Start the simulation. Observe the load voltages from Scope 'Vabc_load' in System Measurements subsystem, and note that a 10Hz output has been constructed from a 60Hz input. A MATLAB® script executes on completion of the simulation to more clearly show the construction of phase-A output voltage from the three-phase input. The user may modify this execution as desired by editing **File ->**

**Model Properties -> Callbacks -> StopFcn**. The user can adjust the output voltage magnitude and frequency by modifying 'Output voltage (V peak)' and 'Output frequency (Hz)' respectively.

# Thyristor Rectifiers

This example shows the operation of thyristor rectifiers.




### Description

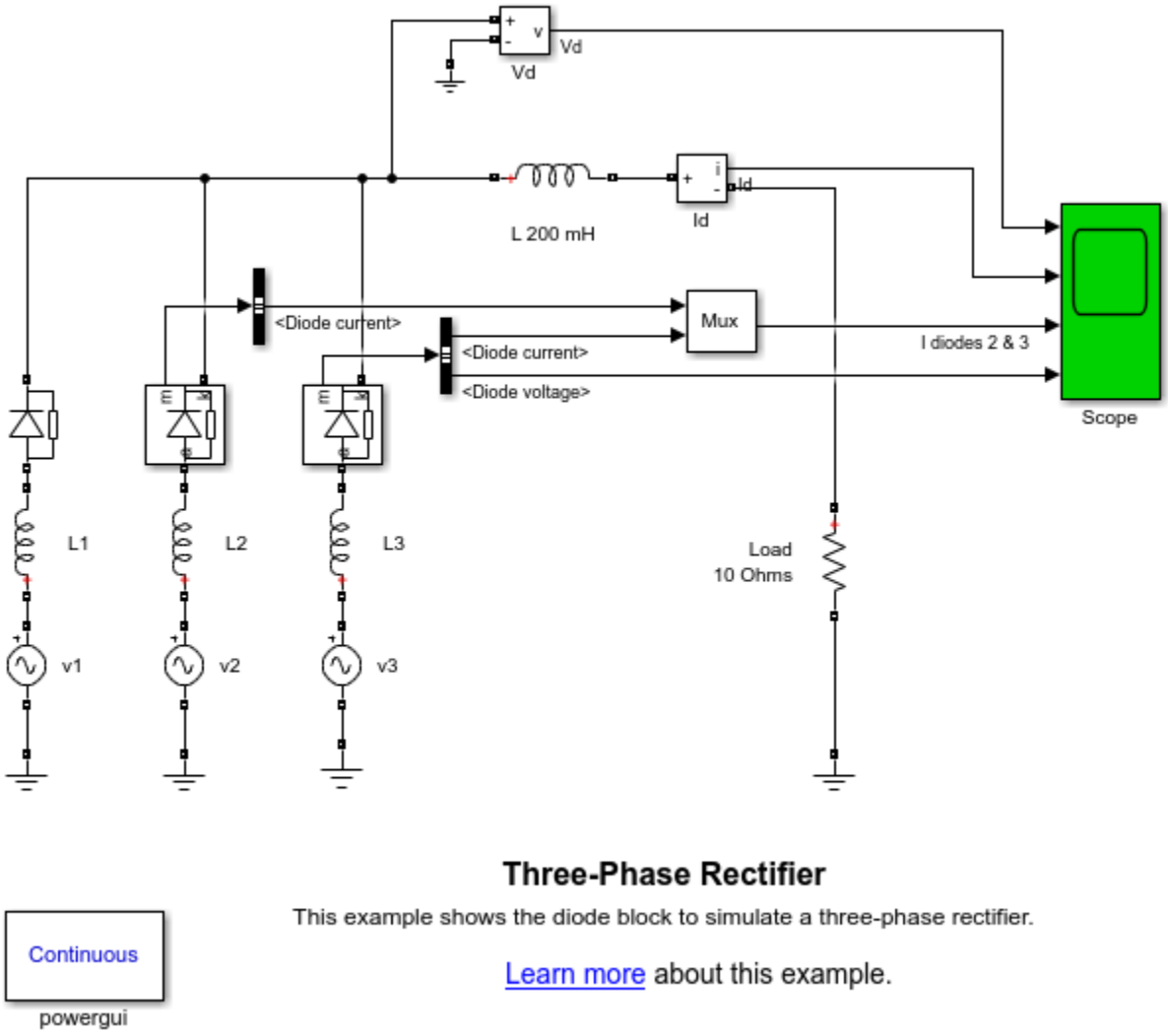
This example compares the operation of a six-pulse thyristor rectifier to a twelve-pulse thyristor rectifier.

The twelve-pulse bridge is represented by two six-pulse bridge connected in series, with their AC connections connected to a transformer with two secondaries (one wye, one delta) that produces a 30 deg. phase shift between the two bridges. This transformer configuration cancels many of the characteristic harmonics produced by the six-pulse bridges.

# Three-Phase Rectifier

This example shows how to use the diode block to simulate a three-phase rectifier.

G. Sybille (Hydro-Quebec)




### Description

A 10 ohm load is fed in DC through a three-phase rectifier from an inductive source (5 mH; 120 V rms). The rectified current is filtered by a 200mH inductance.

Diodes are connected in parallel with RC snubbers (1000 ohms-0.1 uF) The measurement output of Diode2 and Diode3 is used to observe the diode voltage and current.

### Simulation

#### 1. Simulation with continuous integration method

Check the simulation parameters in the Simulation -> Configuration Parameters menu. The Variable-step auto (Automatic solver selection) solver is selected. Note that the Simulation type parameter of the Powergui block is not accessible, and it is set to Continuous. Start the simulation and observe waveforms on the four-trace Scope block. Note that Simulink® selected the Ode23tb solver to simulate the model.

After a transient period, the load current Id stabilizes at 12.7 A. Note that most of the third harmonic which can be seen in the rectified voltage Vd (mean value 127 V) is filtered out by the smoothing reactor.

Also observe the commutation period due to the source inductances. The 3.3 degrees overlap is clearly seen on trace 3 of the scope showing currents in diodes 2 and 3.

Zoom on the Diode3 voltage (trace 4) during the conduction period. The voltage magnitude is the sum of the specified forward voltage (0.8 V) and the voltage developed across the diode resistance (0.01 ohm).

**2. Simulation with discretized system**

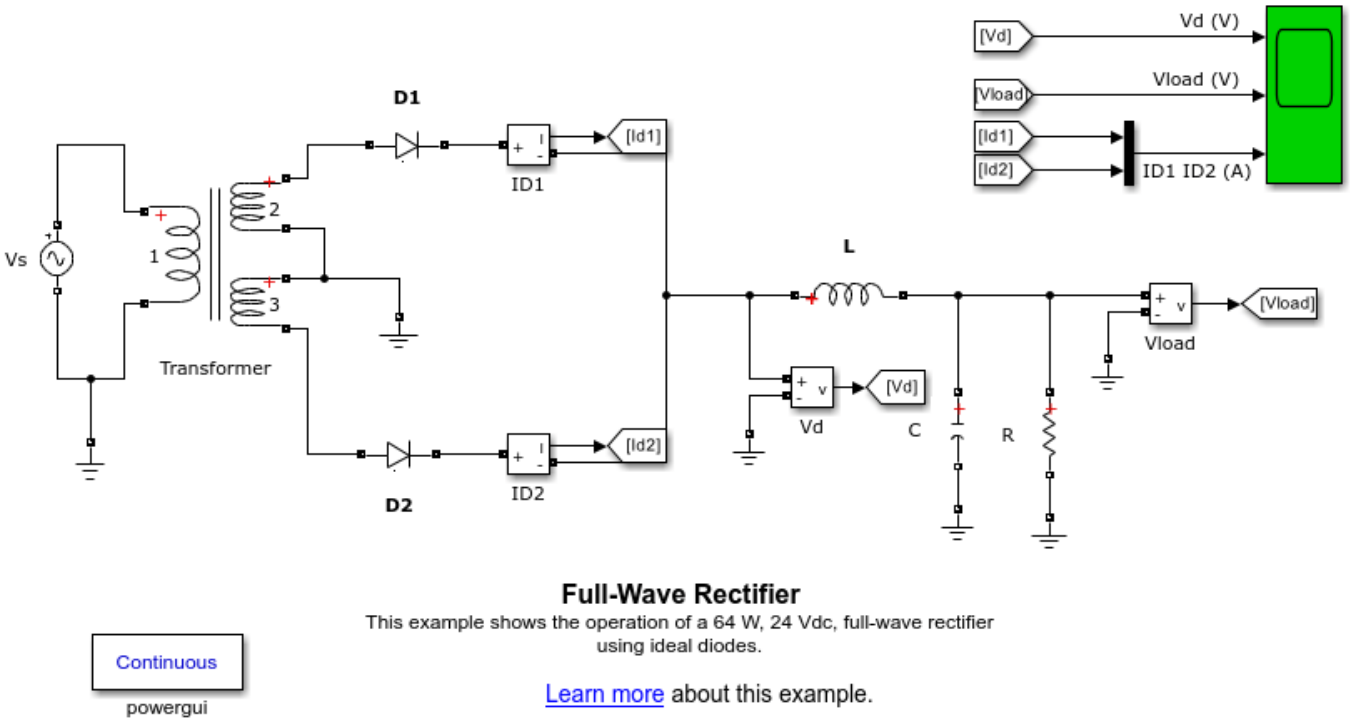
Open the Configuration parameters of the model and select the Fixed-step auto (Automatic solver selection) solver. Restart the simulation. Note that the Simulation type parameter of the Powergui is automatically changed to Discrete, with a sample time of 50e-6 usec. Note that Simulink selected the FixedStepDiscrete solver to simulate the model.

Compare waveforms with those obtained with the continuous integration algorithm.

# Full-Wave Rectifier

This example shows the operation of a full wave rectifier using ideal diodes.

G. Sybille (Hydro-Quebec)




### Description

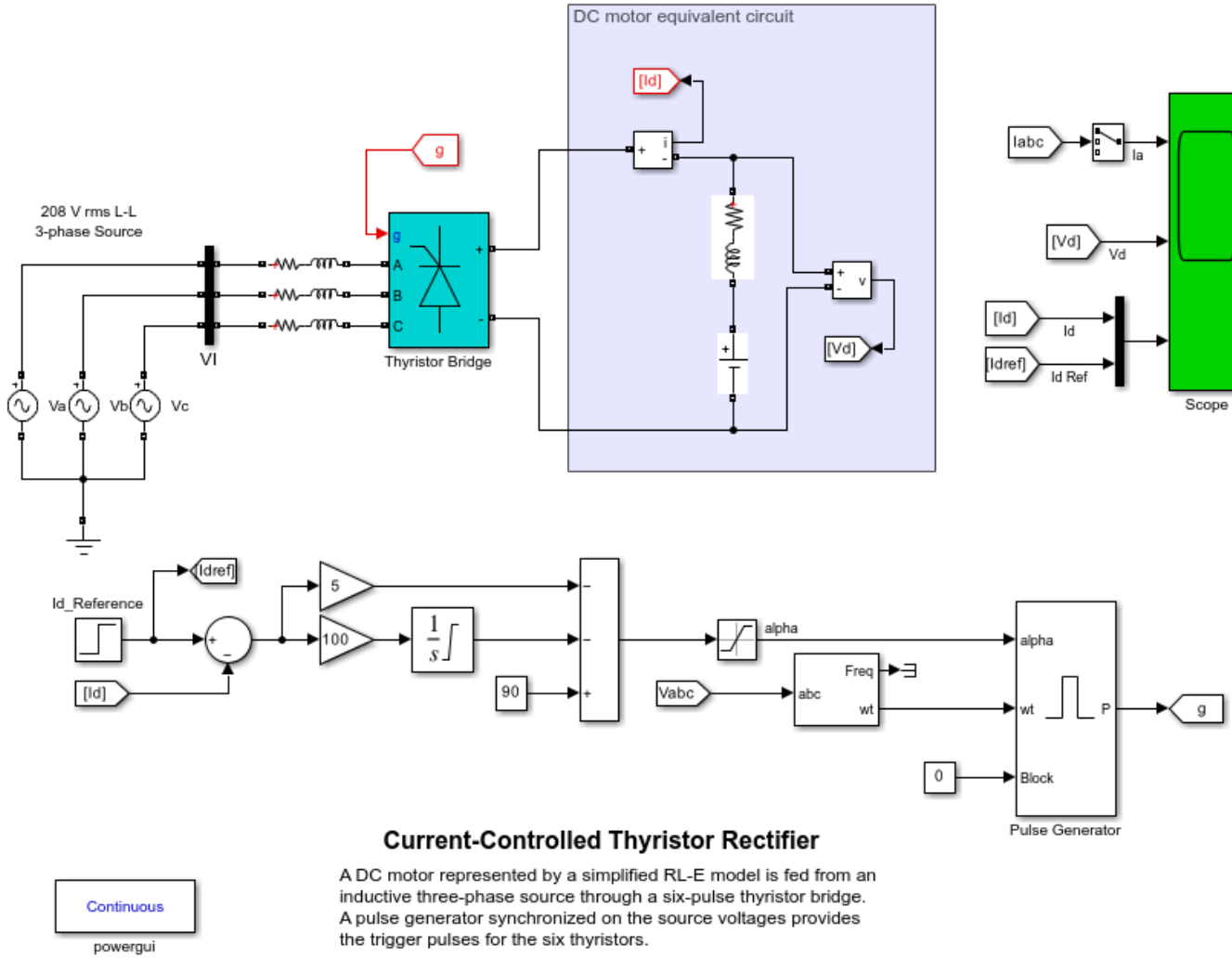
A convenient way of disabling snubbers of all switches in the model is to select the **Disable snubbers in switching devices** option in powergui Preferences tab. Alternatively, you can individually disable the snubbers of selected switches by specifying Rs = inf in their block menus.

In order to simulate diodes as ideal switches, the **Disable Ron resistances in switching devices** and **Disable forward voltages in switching devices** options in the powergui Preferences tab are also selected.

The elimination of the snubbers reduces the circuit stiffness so that you can use a non-stiff solver such as ode45. This solver produces correct results and good simulation speed.

# Current-Controlled Thyristor Rectifier

This example shows the operation of a current-controlled thyristor rectifier.




### Description

A DC motor represented by a simplified RL-E model is fed from an inductive three-phase source through a six-pulse thyristor bridge. A pulse generator synchronized on the source voltages provides the trigger pulses for the six thyristors.

The converter output current is controlled by a PI current regulator built with Simulink® blocks. A step signal is applied to the reference input to test the dynamic response of the current regulator.

In order to allow further signal processing, signals displayed on the Scope block are stored in a variable named ScopeData1 (in structure with time format).

### Simulation

Start the simulation and observe current and voltage waveforms on the Scope block.

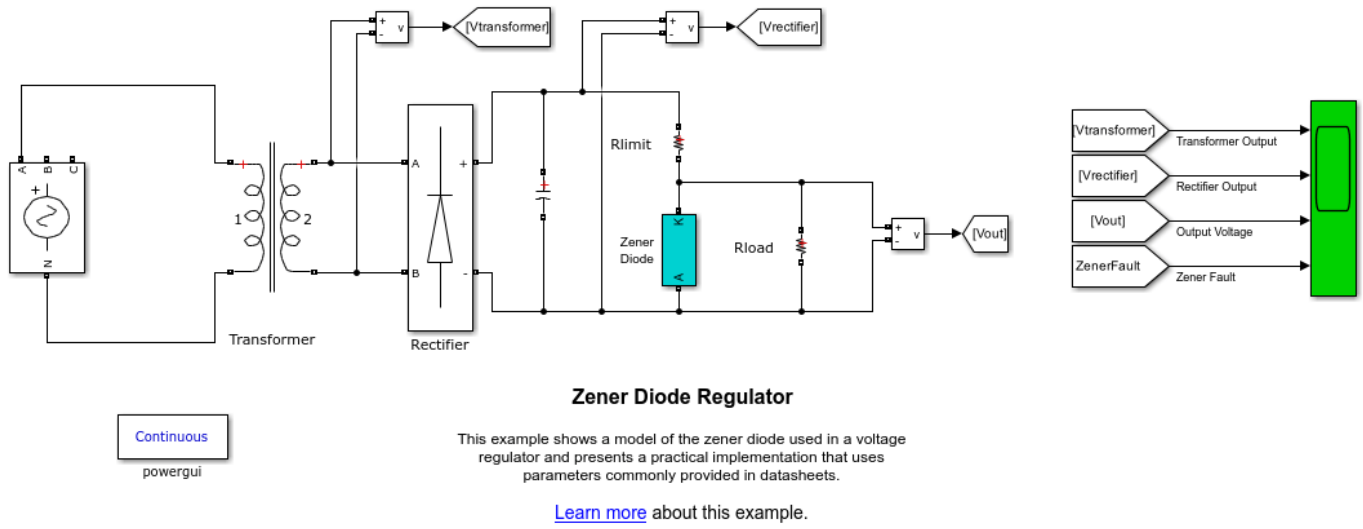
When simulation is completed, open the Powergui and select **FFT Analysis** to display the 0 - 2000 Hz frequency spectrum of signals saved in the ScopeData1 structure. The FFT will be performed on a 2-cycle window starting at t = 0.08 - 2/60 (last 2 cycles of recording). Input labeled Ia should be already selected. Click on Display and observe the frequency spectrum of the last two cycles of Ia. Harmonic currents (6n+/-1) are displayed in % of the fundamental component.

The value of the fundamental current (magnitude = 27.48 A peak) and the Total Harmonic Distortion (THD) are also displayed. Harmonics in the 0 - 2000 Hz range contribute to 29% in THD. If you specify a 0-4000 Hz frequency range, notice that contributions of harmonics above 2000 Hz are negligible in the current THD.

Now select the input labeled Vd and display its spectrum. The DC voltage contains harmonics (6n) and its DC component is 145.4 V.

# Zener Diode Regulator

This example shows a model of the zener diode used in a voltage regulator.




### Description

The Zener diode block modeled in this example presents a practical implementation that uses parameters commonly provided in datasheets. These parameters are

- (1) Zener Voltage Vz
- (2) Dynamic Impedance Zzt
- (3) Knee Impedance Zzk
- (4) Max Continuous Current Izm
- (5) Forward Voltage Drop Vf
- (6) On Resistance Ron

This block can effectively model three regions of operation of the zener diode I-V characteristics - forward-biased, reverse-biased before breakdown and reverse-biased after breakdown. Beyond the maximum reverse continuous current Izm, the zener is assumed to burn up and is treated as an open circuit.

The implementation of the zener diode can be seen by looking under the mask of the block, and is based on [1].

Zener diodes are commonly employed in applications as voltage regulators. The circuit shows an AC source fed to a step-down transformer. The output of the transformer is then rectified using a diode bridge and smoothened using a capacitive filter. The zener diode then acts to regulate the output voltage to the zener voltage 10V. The input current into the zener is limited by the resistor Rlimit to permissible values.

The programmable voltage source is setup to increase its output voltage at 0.1s. As the source output increases, so does the voltage applied at the input of the zener. However, the zener can regulate the

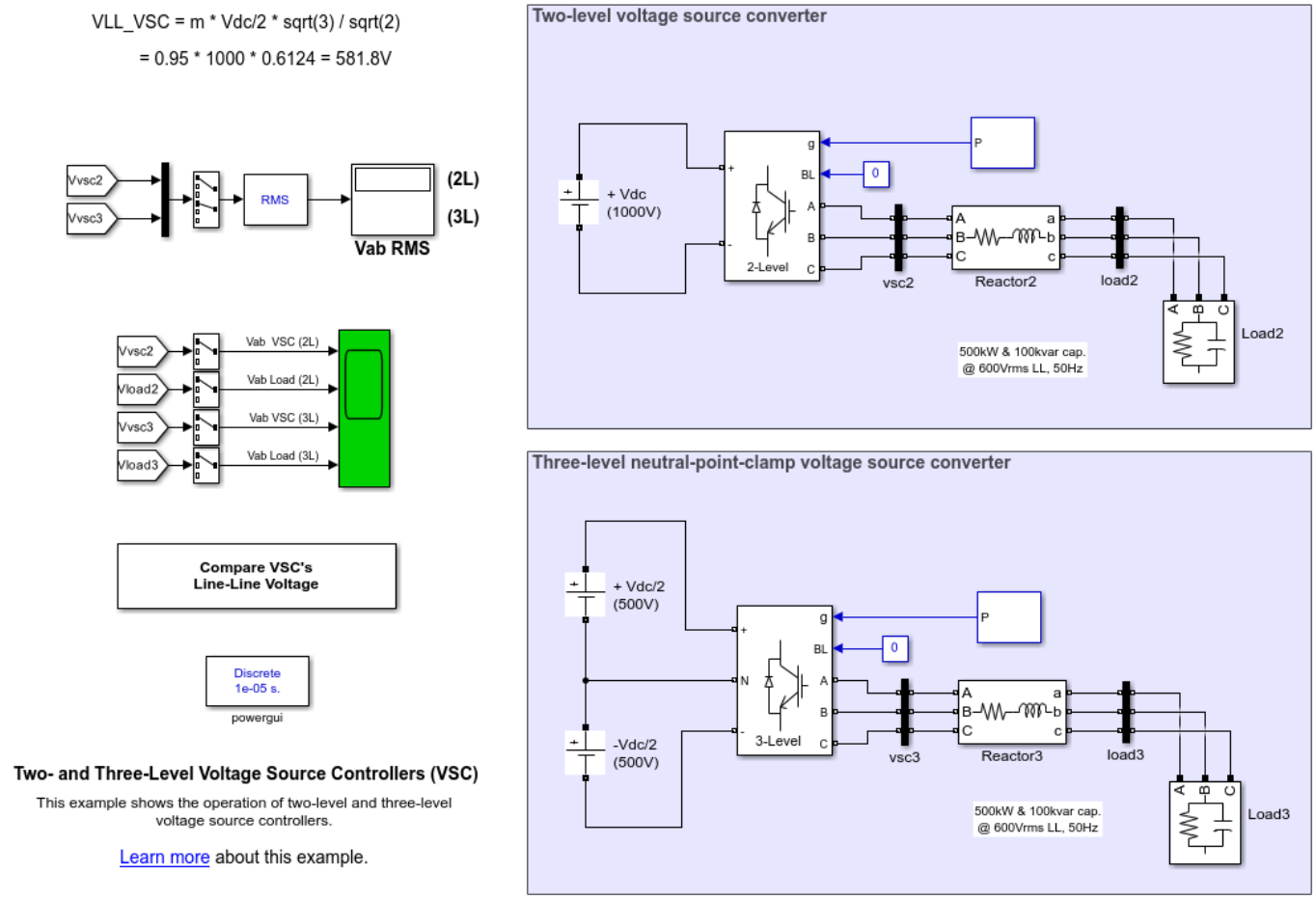
output only as long as its input current is below the maximum specified value. This current increases as we increase the source voltage and the zener ultimately fails at about 0.112s. With the zener acting as an open-circuit at fault, voltage regulation is lost and the output of the capacitive filter gets applied to the load.

### References

1. Wong. S, Hu. C-M, "SPICE macro model for the simulation of zener diode I-V characteristics", Circuits and Devices Magazine, IEEE® Volume 7, Issue 4, Jul 1991 Page(s): 9 - 12, 52

# Two- and Three-Level Voltage Source Controllers (VSC)

This example shows the operation of two-level and three-level voltage source controllers.



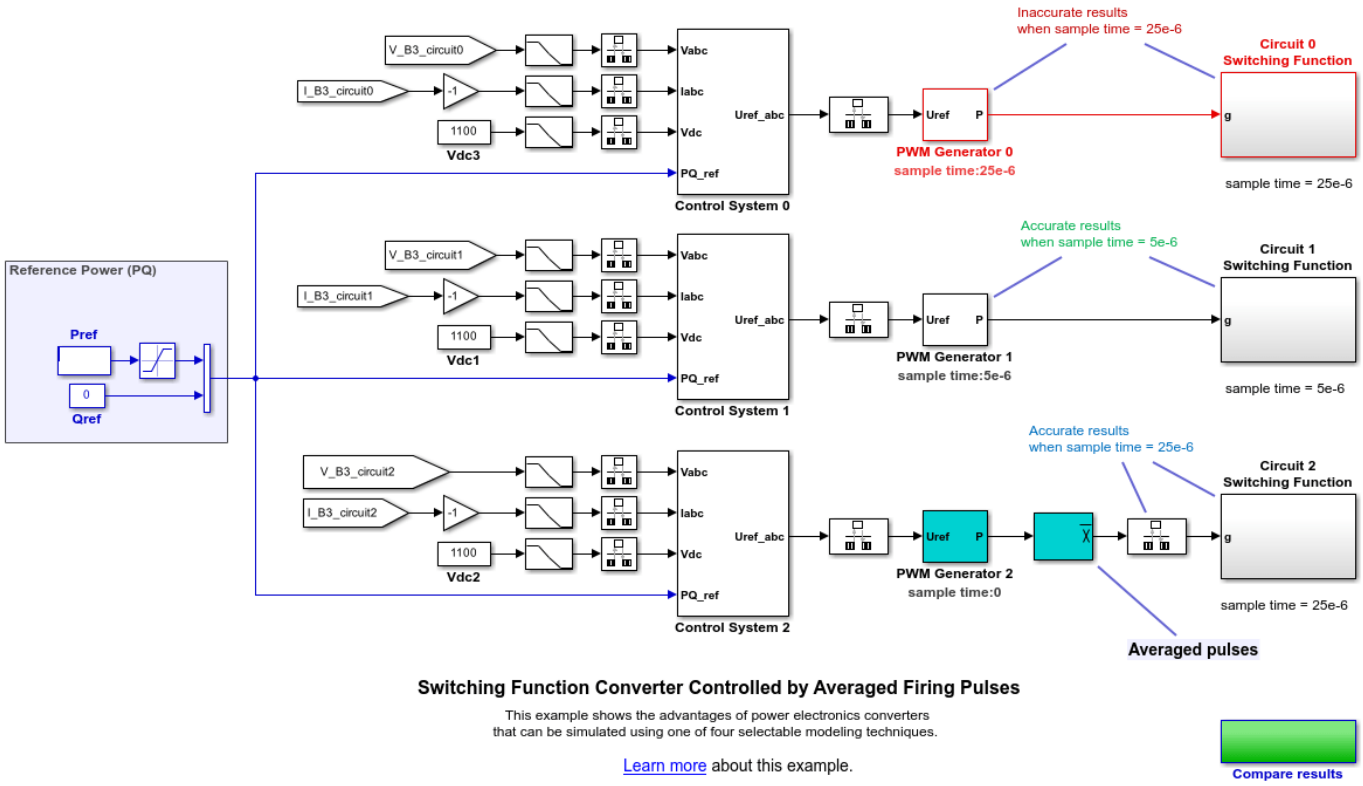

## Description

The upper circuit represents a two-level voltage source converter represented by a six pulse IGBT bridge with inverse-parallel diodes. The voltage at the AC terminals of each phase is switched between the positive and negative potential values of the DC terminals.

The lower circuit represents a three-level voltage source converter with neutral-point clamp. This topology can synthesize three voltage levels from positive DC value, zero (neutral), and negative DC value at the AC terminal of each phase. In this neutral-point clamped configuration, the clamping diode valves are replaced by IGBT valves, to allow more control on the generated AC waveforms.

# Switching Function Converter Controlled by Averaged Firing Pulses

This example shows the advantages of power electronics converters that can be simulated using one of four selectable modeling techniques




## Description

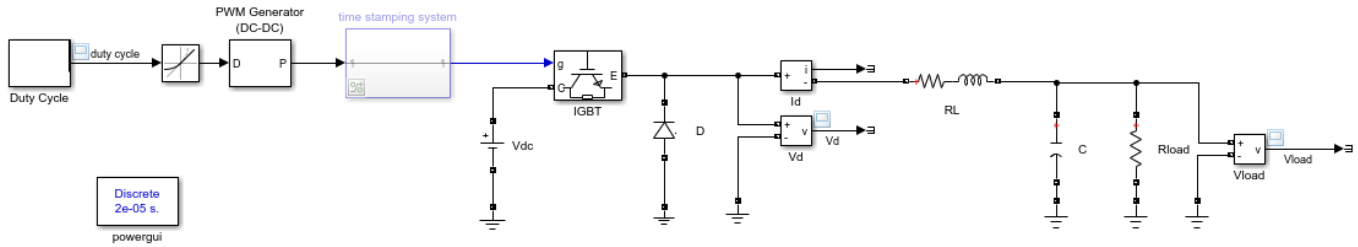
The converter in the circuit 0 is modeled by a switching function controlled by firing pulses produced by a PWM generator operating at 25 microseconds. The circuit 1 also uses a switching function converter and a PWM generator but is sampled at 5 microseconds. The converter in the circuit 2 is modeled by a switching function controlled by firing pulses averaged over a period of 25 microseconds. In this case, the PWM generator is in continuous mode.

## Simulation

Run the simulation and look at the simulation results to observe that good performance is obtained with the use of averaged pulses at higher sample time period, when compared to the performance obtained with regular pulses at lower sampling time.

# Buck Converter - Increased Accuracy and Simulation Speed Using Interpolation

This example shows how to use the interpolation method on the powergui to preserve model accuracy for simulations with larger time steps.




**Buck Converter - Increased Accuracy and Simulation Speed Using Interpolation**

This example shows how to use the interpolation method on the powergui to preserve model accuracy for simulations with larger time steps.

Learn more about this example.

### Description

This example shows a DC-DC buck converter feeding an RC load from a 200 V source. The PWM frequency is set at 5 kHz and duty cycle varies between 0.1 and 0.8. With this 5 kHz PWM frequency, the sample time needed for a 0.5% resolution on duty cycle using a standard discretization method (Tustin or Backward Euler) is Ts = 1e-6 sec (1 MHz sampling frequency = 200 x PWM_freq -> resolution = 1/200 = 0.5 %). The example shows that using interpolation allows you to run the model with a much larger time step (Ts = 20e-6 sec) while preserving model accuracy. The example also demonstrates the concept of time-stamped gate signals in Specialized Power Systems switching devices.

### Simulation

By default, sample time is initialized to 20e-6 sec (inside the Model Properties -> PreloadFcn Callback). Open the powergui block and on the Preferences tab, make sure that the "Discrete solver" parameter is set to Tustin, and that the "Interpolate switching events" option is enabled. Also make sure that the "Use time-stamped gate signals" option is disabled. The Simulation Data Inspector is enabled and the Vload signal is logged.

1) Perform a first simulation with interpolation in service. Note that the subsystem named Time-Stamping System is commented through so that the pulse signal of the PWM Generator block is passed directly to the IGBT block.

2) Now, in the powergui block, enable the "Use time-stamped gate signals" option and uncomment the Time-Stamping System. This subsystem is now computing delays on and delays off for the pulse signal. Three signals (the pulse, the delay on, and the delay off signals) are now passed to the IGBT block. Run the simulation and verify that the simulation results are the same.

3) Now disable interpolation and specify Ts = 1e-6 in the command window. Comment through the Time-Stamping System, and disable the "Use time-stamped gate signals" option. Perform a third simulation run.

4) In the powergui Solver tab, set "Simulation type" to Continuous. Perform a fourth simulation run with the continuous model.

5) Using the Data Inspector, compare the four simulation runs. Vload voltage obtained during the second and third runs (interpolation with Ts = 20e-6 sec and no interpolation with Ts = 1e-6 sec) is very close to the continuous simulation results.

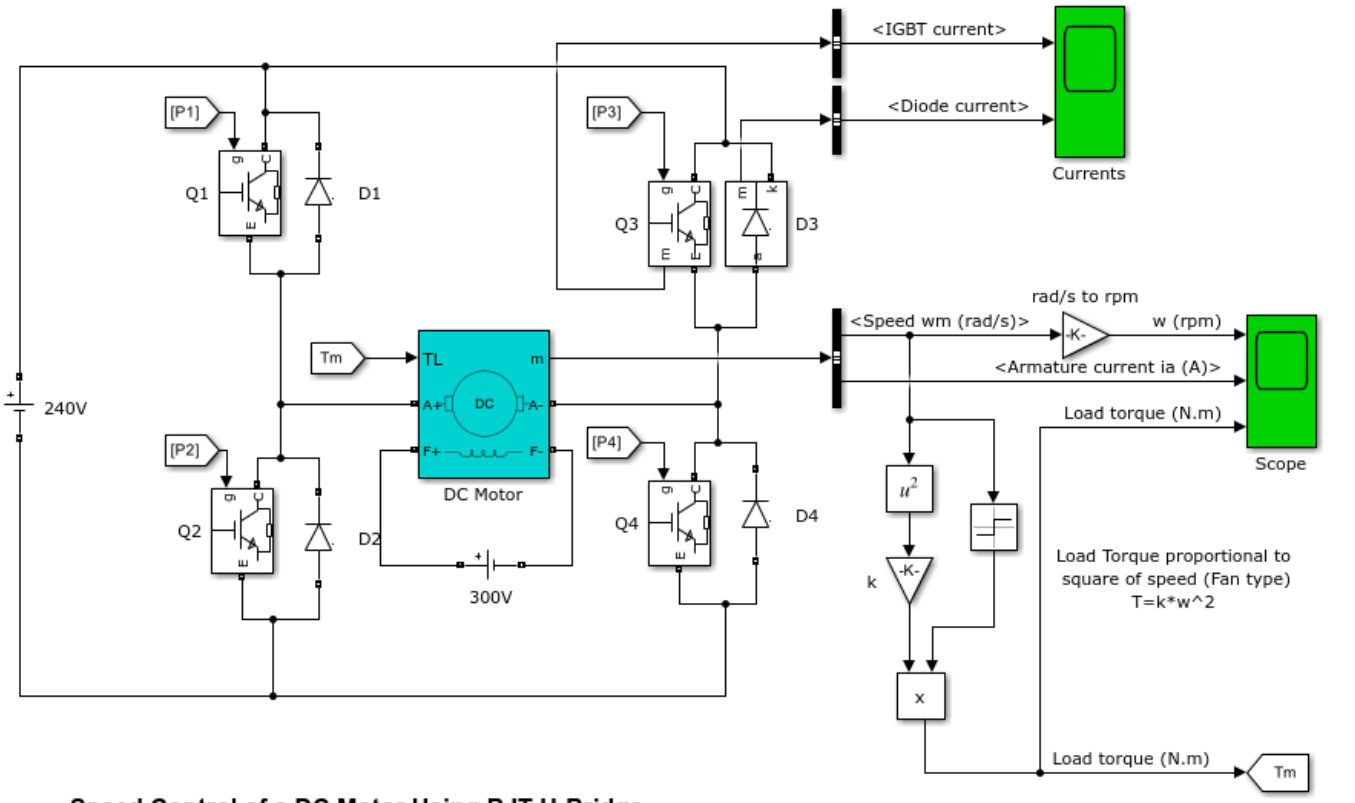
6) Notice that the interpolation solver matches the continuous solver and is even more accurate than the standard discrete solver.

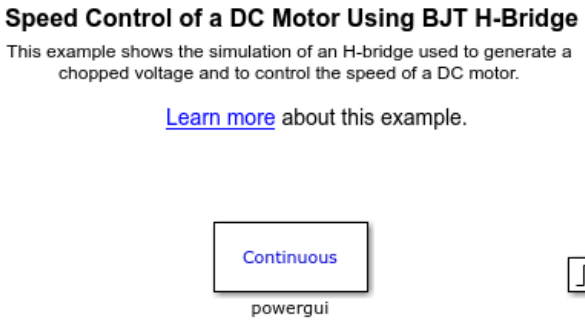
7) Compare simulation speeds of discrete models (interpolation with Ts = 20e-6 sec and no interpolation with Ts = 1e-6 sec). To achieve significant simulation times, increase simulation stop time to 0.5 sec. The Diagnostic Viewer displays the simulation time at the end of each simulation run. The speed increase obtained with the interpolation method is approximately 4X.

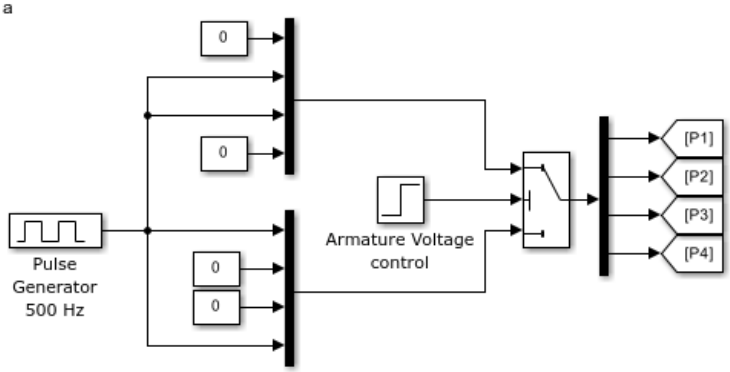
# Speed Control of a DC Motor Using BJT H-Bridge

This example shows the simulation of an H-bridge used to generate a chopped voltage and to control the speed of a DC motor.

Gilbert Sybille (Hydro-Quebec)




## Description

The Bipolar Junction Transistor (BJT) when used for power switching applications, operates as an IGBT. When it is conducting (BJT operating in the saturated region), a forward voltage Vf is developed

between collector and emitter (in the range of 1 V). Therefore, the IGBT block can be used to model the BJT device.

The IGBT block does not simulate the gate current controlling the BJT or IGBT. The switch is controlled by a Simulink® signal (1/0). The DC motor uses the preset model (5 HP 24V 1750 rpm). It simulates a fan type load (where Load torque is proportional to square of speed). The armature mean voltage can be varied from 0 to 240 V when the duty cycle (specified in the Pulse Generator block) is varied from 0 to 100%.

The H-bridge consists of four BJT/Diode pairs (BJT simulated by IGBT models). Two transistors are switched simultaneously: Q1 and Q4 or Q2 and Q3. When Q1 and Q4 are fired, a positive voltage is applied to the motor and diodes D2-D3 operate as free-wheeling diodes when Q1 and Q4 are switched off. When Q2 and Q3 are fired, a negative voltage is applied to the motor and diodes D1-D4 operate as free-wheeling diodes when Q2 and Q3 are switched off.

**Simulation**

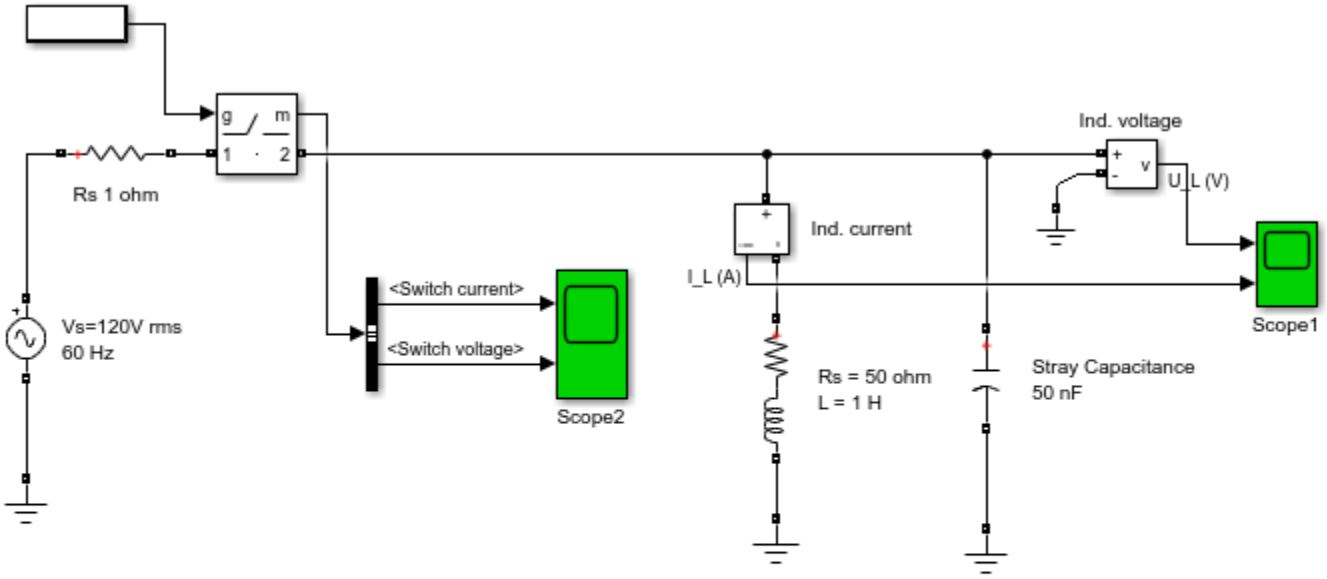
The motor starts in the positive direction with a duty cycle of 75% (mean DC voltage of 180V). At t= 0.5 sec., the armature voltage is suddenly reversed and the motor runs in the negative direction.

'Scope' shows motor speed, armature current and load torque and 'Currents' shows currents flowing in BJT Q3 and diode D3.

# Inductive Current Chopping

This example shows the effect of current chopping in an inductive circuit.

G. Sybille (Hydro-Quebec)



**Inductive Current Chopping**

This example shows the effect of current chopping in an inductive circuit.

Learn more about this example.

### Description

The Ideal Switch is used to switch off and on an inductor (L = 1H, R = 50 ohms) on a 120 Vrms, 60 Hz source. The inductor has a stray capacitance C = 50 nF. The switch resistance is 1e-3 ohm. The switch is initially closed (initial state = 1). It is open at time t = 36ms and then reclosed at time t = 0.15 s.

### Simulation

Start the simulation and observe the inductor voltage U_L and inductor current I_L on Scope1. At the opening time, t = 36 ms, the current in the inductor is I_L= 0.187 A. The current chopping in the inductor produces a high frequency overvoltage (711 Hz) across the inductor. The maximum voltage can be calculated from the following equation:

```
U_Lmax = I_L*sqrt(L/C) = 0.187*sqrt(1/50e-9) = 836 V
```

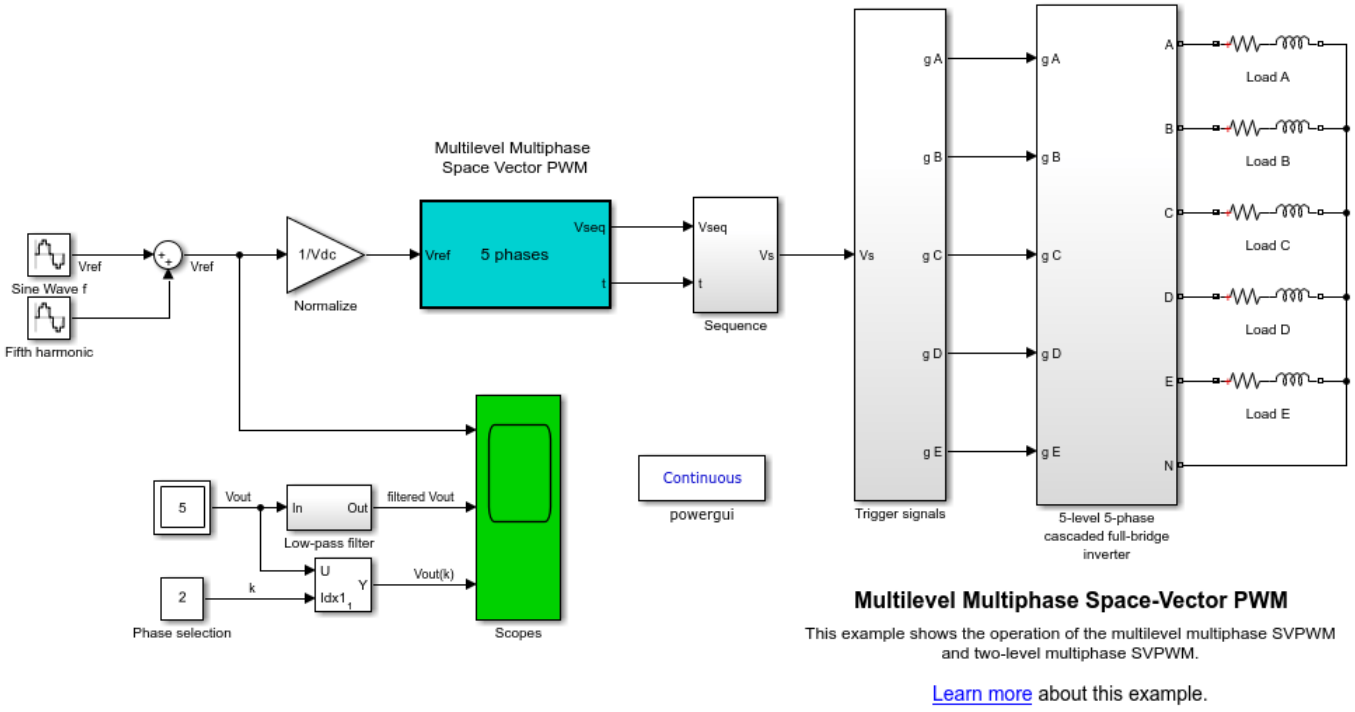
At the reclosing time, t = 0.15 s, the source voltage is zero. Therefore, a full offset is observed in the inductor current. The switch current spike observed on trace 1 of Scope2 at switch reclosing (t =

0.15 s) is due to the capacitor discharge through the source resistance. If you zoom on this current spike you will observe a current step of 30 A (30V capacitor voltage / 1 ohm source resistance), followed by a fast discharge time constant (RC = 1*50e-9 = 50e-9 s).

# Multilevel Multiphase Space-Vector PWM

This example shows the operation of the multilevel multiphase SVPWM and two-level multiphase SVPWM.

Oscar Lopez, Jacobo Alvarez, Jesus Doval-Gandoy and Francisco Freijedo, Electronics Technology Department University of Vigo, Spain. olopez@uvigo.es




### Description

This example shows the operation of the Multilevel multiphase SVPWM and Two-level multiphase SVPWM. It includes a five-level five-phase inverter feeding a passive load. Detailed information about the modulation algorithm, its Simulink® implementation, and the simulated case can be found in [1].

1. The 'Sine Wave f' and 'Fifth harmonic' blocks generate an unbalanced five-phase reference voltage with a fifth harmonic.

2. The 'Normalize' block normalize the reference voltage

3. The 'Multilevel multiphase SVPWM' performs the multilevel multiphase space-vector PWM (SVPWM) algorithm presented in [1]. This block makes use of the block 'Two-level multiphase SVPWM', also described in [1], that can be used alone with two-level multiphase converters. Both blocks require the specification of the number of phases and their outputs are a matrix with the switching vector sequence and a vector with the switching times.

4. The 'Sequence' block provides the time sequence of switching vectors.

5. The 'Trigger signals' generates the proper trigger signals from output levels specified in each switching vector.

6. The '5-level 5-phase cascaded full-bridge inverter' block is the ideal model of the multilevel voltage-source converter.

7. The loads are resistances with a series connected inductances. Load neutral is connected to the inverter neutral.

### Simulation

The following signals can be observed:

'Vref' is the sampled voltage reference for each phase.

'filtered Vout' is the output voltage of 'Vout' phase after been filtered with a low-pass filter.

'Vout(k)' is the switched output voltage of phase 'k'. The phase shown in the scope can be selected by means of the 'Phase selection' block.

The filtered output voltage 'filtered Vout' of the converter follows the reference voltage 'Vref'. Therefore, the modulation is working properly.

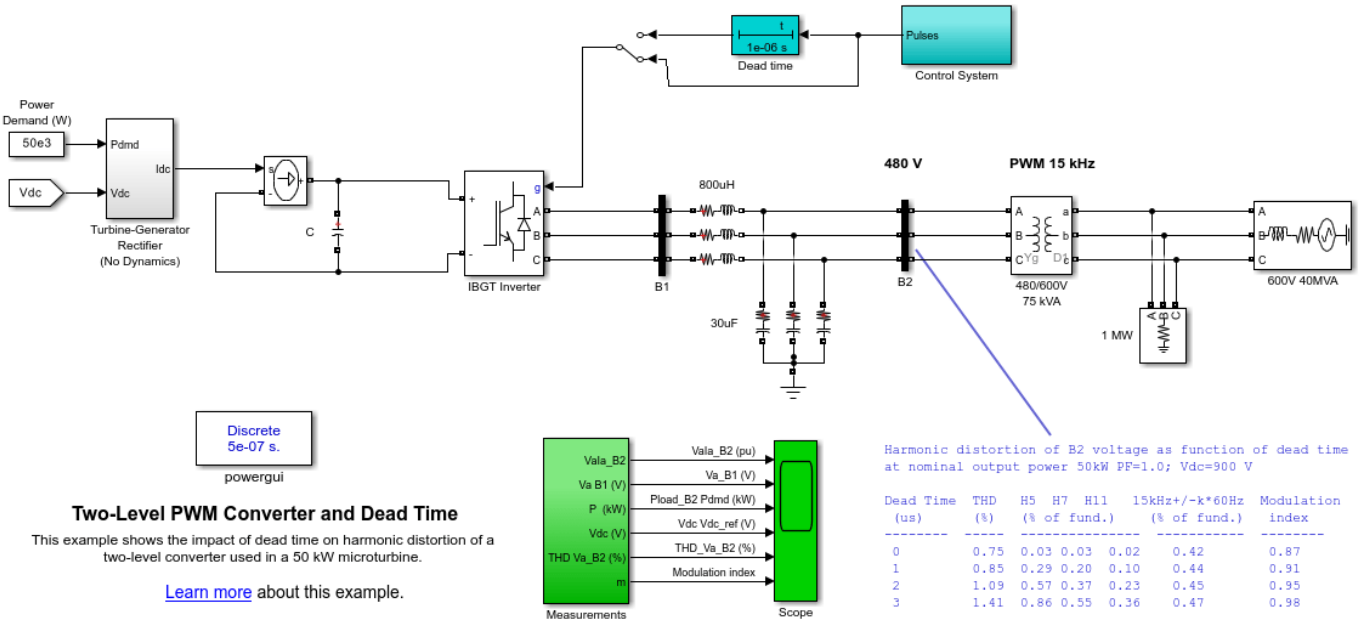
The output voltage of the inverter 'Vout' has five levels in phases b, c, and d (k=2,3,4) and three levels in phase a and e (k=1,5).

### References

[1] Oscar Lopez, Jacobo Alvarez, Jesus Doval-Gandoy and Francisco D. Freijedo "Multilevel Multiphase Space Vector PWM algorithm", IEEE® Transactions on Industrial Electronics, vol. 55, no. 5, pp. 1933-1942, May 2008, doi:10.1109/TIE.2008.918466.

# Two-Level PWM Converter and Dead Time

This example shows the impact of dead time on harmonic distortion of a two-level converter used in a 50 kW microturbine.




## Description

The circuit represents the DC to AC conversion unit of a 480 V, 50 kW microturbine connected to a 600 V power grid. As the purpose of this example is to illustrate the high frequency harmonics generated by the inverter, the slow dynamics of the gas-fired turbine and permanent magnet generator are not represented. The Turbine-Generator-Rectifier group is modeled as a simple DC current source injecting the requested DC power into the DC bus. The inverter uses a PWM, two-level IGBT converter (using the Universal Bridge block). The SPWM modulator uses a carrier frequency of 15 kHz. The control system uses two regulators: an inner current loop controlling the current at bus B2 and an outer DC voltage regulator controlling the DC bus voltage. LC filters are used to reduce harmonic voltages produced at the 480 V bus B2 (harmonic frequencies around multiples of 15 kHz). Note that 1 ohm resistances are connected in series with the 30 uF capacitors of the shunt filters. These resistances are used to damp non characteristic low frequency harmonics resulting from the resonance introduced by the LC filters (around 1.9 kHz).

In a two-level voltage-sourced converter (VSC) using ideal switches, the two pulses sent to the upper and lower IGBT of each arm could be complementary. However, in practical VSCs the turn-off of semiconductor switches is delayed because of the storage effect. Therefore, a time delay of a few microseconds (storage time + safety margin) is required to allow complete extinction of the IGBT which is switched off before switching on the other IGBT. Otherwise, a short-circuit could result on the DC bus. The dead time is modeled by introducing a On/Off Delay block at the pulse input of the converter block. The delay specified in that block is applied on the rising edges of pulses.

In order to get an acceptable accuracy with a 15 kHz switching frequency, a sample time Ts_Power= 0.5 microseconds is used to discretize the circuit. The discrete control system uses a much larger sample time (Ts_Control=50 microseconds)

### Simulation

#### 1. Simulation with zero dead time

Start the simulation and observe voltages, current, power and control signals on the Scope block. The simulation has been automatically initialized to start in steady state. The quadrature-axis current Iq_ref is set to zero inside the current controller in order to generate power at unity power factor. Therefore, voltage and current at bus B2 (trace 1) are in phase. The DC voltage (trace 4) is regulated at 900 V. On trace 3, you can observe the power demand (set at nominal power of 50 kW) and the measured output power at bus B2 (49 kW, because of 1 kW losses in converter and filters).

Once the simulation is completed, open the Powergui and select "FFT Analysis" to display the frequency spectrum of signals saved in the ScopeData structure (variable specified in the Scope block). Make sure that input VaIa_B2 (pu), and signal number 1 are selected. The FFT will be performed on the last cycle of phase A voltage at bus B2. Then click Display to observe the 0-50000 Hz frequency spectrum. As expected, harmonics are observed mainly around multiples of the switching frequency (15 kHz). The Total Harmonic Distortion (THD) is displayed above the spectrum (THD=0.75%). THD is also measured during simulation by the "Discrete THD" block (trace 5 of Scope1)

#### 2. Impact of dead time on harmonic distortion

Now move the manual switch to its upper position to apply a dead time of 1 microsecond to the firing pulses. Repeat simulation and FFT analysis for dead times up to 3 microseconds. You will note an increase of THD when the dead time is increased. Results from frequency analysis are summarized in the table shown on the model for dead times comprised between zero and 3 microseconds. This table indicates that the amplitude of the main characteristic frequencies (around 15 kHz) does not vary significantly when the dead time is increased. The THD increase is caused mainly by the introduction of low frequency harmonics (mainly 5th, 7th and 11th ). For a 3 us dead time the THD has increases from 0.74 % to 1.75 %. The table also indicates the modulation index increases when the dead time increases, indicating a less efficient use of the DC voltage. With a 900 V DC voltage, if dead time would be increased above 3 us, the modulation index would be higher than 1and additional distortion would be introduced because of overmodulation. You can check harmonic distortion for different power outputs and DC operating voltages by specifying new values in the "Power Demand" constant block and in the "Vdc_ref" constant block (inside the Control System).

### Regenerate Initial Conditions

The initial states required to start this model in steady state have been saved in the "power_microturbineDT.mat" file. When you open this model, the InitFcn callback (in the Model Properties/Callbacks) automatically loads into your workspace the contents of this .mat file ("xInitial" variable).

If you modify this model, or change parameter values of power components, the initial conditions stored in the "xInitial" variable will no longer be valid and Simulink® will issue an error message. To regenerate the initial conditions for your modified model, follow the steps listed below:

1. In the Simulation/Configuration Parameters/Data Import/Export Parameters menu, uncheck the "Initial state" parameter.

2. Change the Simulation Stop Time to 1 second. Note that in order to generate initial conditions coherent with the 60 Hz voltage source phase angle, the Stop Time must an integer number of 60 Hz cycles.

3. Change the Simulation Mode from "Normal" to "Accelerator".

4. Start simulation. When simulation is completed, verify that steady state has been reached by looking at waveforms displayed on the scope. The final states which have been saved in the "xFinal" structure with time can be used as initial states for future simulations. Executing the next two commands copies these final conditions in "xInitial" and saves this variable in a new file (myModel_init.mat).

```
>> xInitial=xFinal;
```

```
>> save myModel_init xInitial
```

5. In the File-> Model Properties -> Callbacks -> InitFcn window, change the name of the initialization file from "power_microturbineDT.mat" to "myModel_init.mat". Next time you open this model, the variable xInitial saved in the myModel_init.mat file will be loaded in your workspace.

6. In the Simulation -> Configuration Parameters menu, check "Initial state".

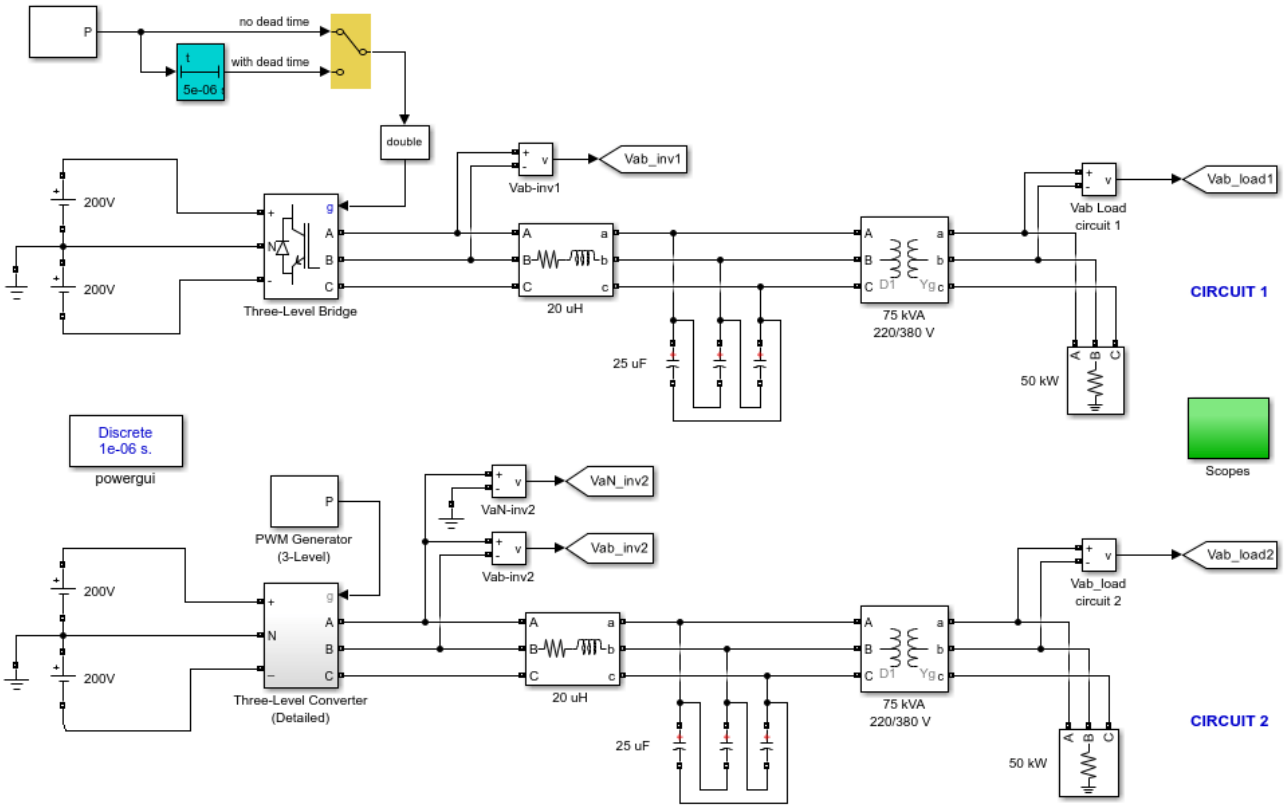
7. Start the simulation and verify that your model starts in steady-state.

8. Change the Simulation Stop Time and Simulation Mode back to their original values (0.2 seconds, Normal).

9. Save your model.

# Neutral Point Clamp Inverter and Dead Time

This example shows the impact of dead time and semiconductor failure inside a three-level PWM converter.




**Neutral Point Clamp Inverter and Dead Time**

This example shows dead time and semiconductor failure inside a 50 kW, 380 V, 50 Hz, three-phase, three-level PWM converter.

Learn more about this example.

**Harmonic Distortion of Load Voltage**

| Dead time (us) | Vab (Vrms) | THD (%) | H5 (%) | H7 (%) | 8 kHz (%) |
|---|---|---|---|---|---|
| 0 | 380 | 1.27 | 0.21 | 0.13 | 0.74 |
| 5 | 359 | 1.99 | 1.25 | 0.73 | 0.81 |

## Description

This model represents two identical circuits modeling a 50kW, 380V, 50Hz, three-phase, three-level inverter. The IGBT inverter uses SPWM technique, (8kHz carrier frequency) to convert DC power from a +/-200Vdc source to 220V AC, 50Hz. The inverter feeds a 50kW resistive load through a 75kVA 220/380V transformer. L-C filters are used at the converter output to filter out harmonic frequencies Fh generated mainly around multiples of the 8kHz switching frequency (Fh=n*8000 +/- k*50Hz). The 12 inverter pulses required by the inverter are generated by the PWM Generator block. The system operates in open loop at a constant modulation index.

Circuit 1 uses the Three-Level Bridge block to model the inverter. Circuit 2 uses individual IGBT and diode blocks. The dead time is simulated with the On/Off Delay block and it is applied on the circuit 1 only.

### Simulation

#### Comparing simulation results with the Three-Level Bridge block and the user-built converter - no dead time

1. Set the Manual Switch block to its upper position to disable dead time effect on circuit 1. 2. Start the simulation and observe the inverter and load voltages of circuit 1 and circuit 2 which are superimposed on Scope1. 3. Once the simulation is completed, open the Powergui and select FFT Analysis to display the frequency spectrum of signals saved in the ScopeData1 structure. 4. The FFT will be performed on the last cycle of the selected signal. You can analyze either inverter voltages (select input Vab_inv_1_2) or load voltages (select input Vab_load_1_2). Select Signal number 1 or Signal number 2 to analyze voltages of circuit 1 or circuit 2 and click on Display to observe the 0-25000 Hz frequency spectrum.

As expected, at the inverter output, harmonics are observed around multiples of the switching frequency (8 kHz). At the load terminals, these high frequency harmonics are considerably reduced by LC filters. Also, note that the LC filters creates a resonance which produces additional harmonics around 4.4 kHz. The fundamental component and Total Harmonic Distortion (THD) are displayed above the spectrum graph. For both circuits THD is 1.27% on the load side and fundamental voltage is 380 V rms (537.6 V peak).

#### Impact of dead time on fundamental voltage and harmonic distortion

In a three-level voltage-sourced converter (VSC) using ideal switches, the two pairs of pulses sent to each arm could be complementary. For example, for phase A, IGBT1 is complementary of IGBT3 and IGBT2 is complementary of IGBT4. However, in practical VSCs the turn-off of semiconductor switches is delayed because of the storage effect. Therefore, a time delay of a few microseconds (storage time + safety margin) is required to allow complete extinction of the IGBT which is switched off before switching on the other IGBT. Otherwise, a short-circuit could result on the DC bus.

5. Set the Manual Switch block to its lower position. 6. Run the simulation and compare the two load voltage waveforms on Scope1 and their fundamental values on the two Display blocks. 7. Do the FFT analysis with the Powergui block. You will note that the dead time decreases the fundamental component and increases distortion.

Results from frequency analysis are summarized in the table shown in the model. This table indicates that the amplitude of the main characteristic frequencies (around 8 kHz) increases slightly (from 0.74% to 0.82 %) when the dead time is used. The THD increase is caused mainly by the introduction of non characteristic low frequency harmonics (mainly 5th and 7th).

#### Simulating a diode failure fault inside the three-level converter

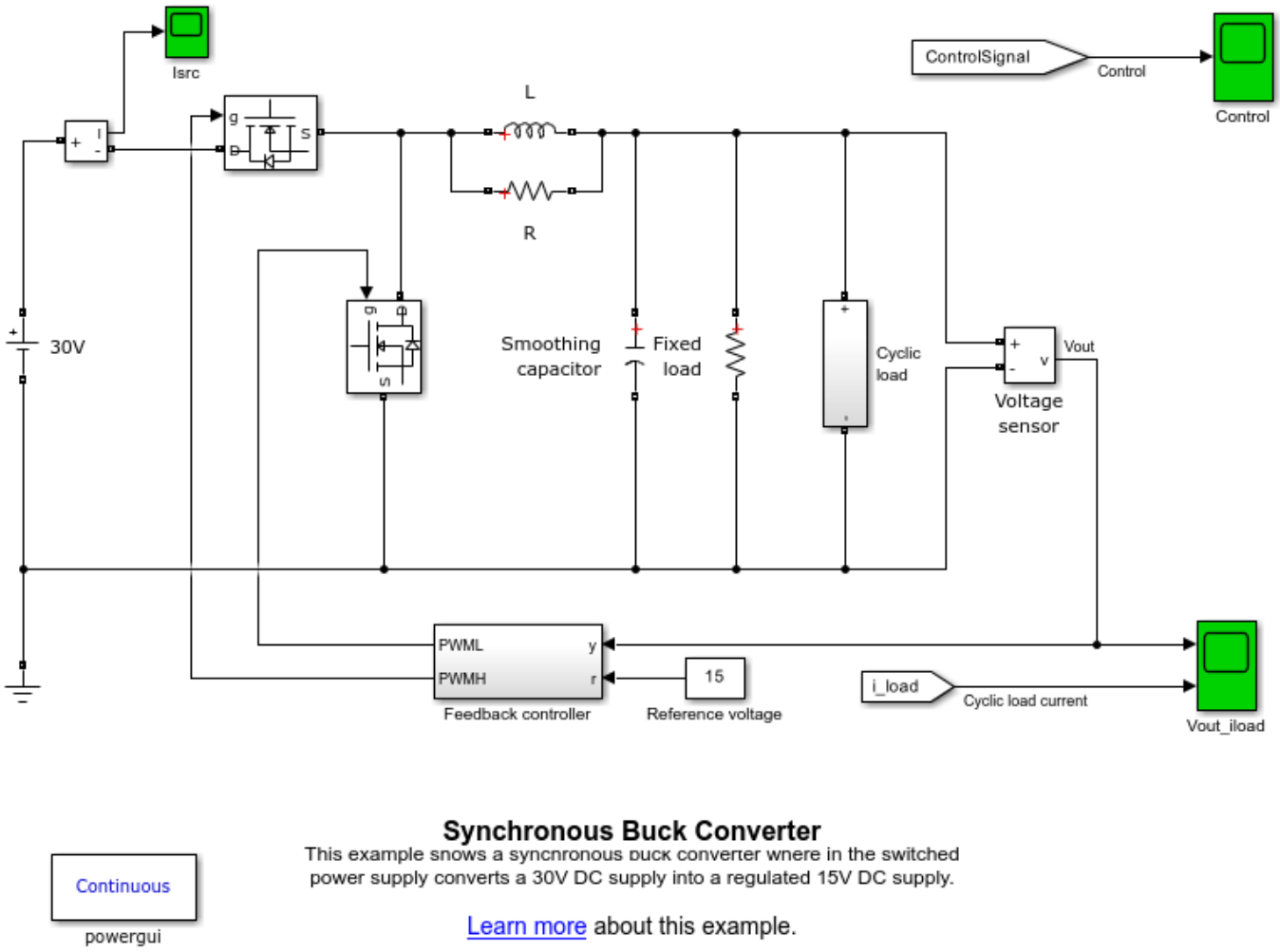
8. Set the manual switch to its upper position to disable the dead time and for the two inverters to generate identical waveforms. 9. Open the Three-Level Converter (Detailed) subsystem. Note that an ideal switch block labeled D5 Open is connected in series with the neutral clamping diode D5 of phase A. 10. Open the Stair Generator block menu and in the Time parameter, change the 100 multiplier to 1. The block will now order a switch opening at t=0.25 s, thus simulating a diode failure in open circuit. 11. Start simulation and compare circuit 1 and circuit 2 waveforms on Scope1.

Diode D5 current and VaN voltage of inverter 2 are displayed on Scope2. When D5 fails open, its clamping effect to neutral is disabled and VaN voltage oscillates between +200V and -200 V (instead of +200V and zero). This unsymmetrical operation of phase A inverter arm introduces low frequency, odd and even harmonics.

# Synchronous Buck Converter

This example shows a synchronous buck converter.

P. Giroux (Hydro-Quebec)

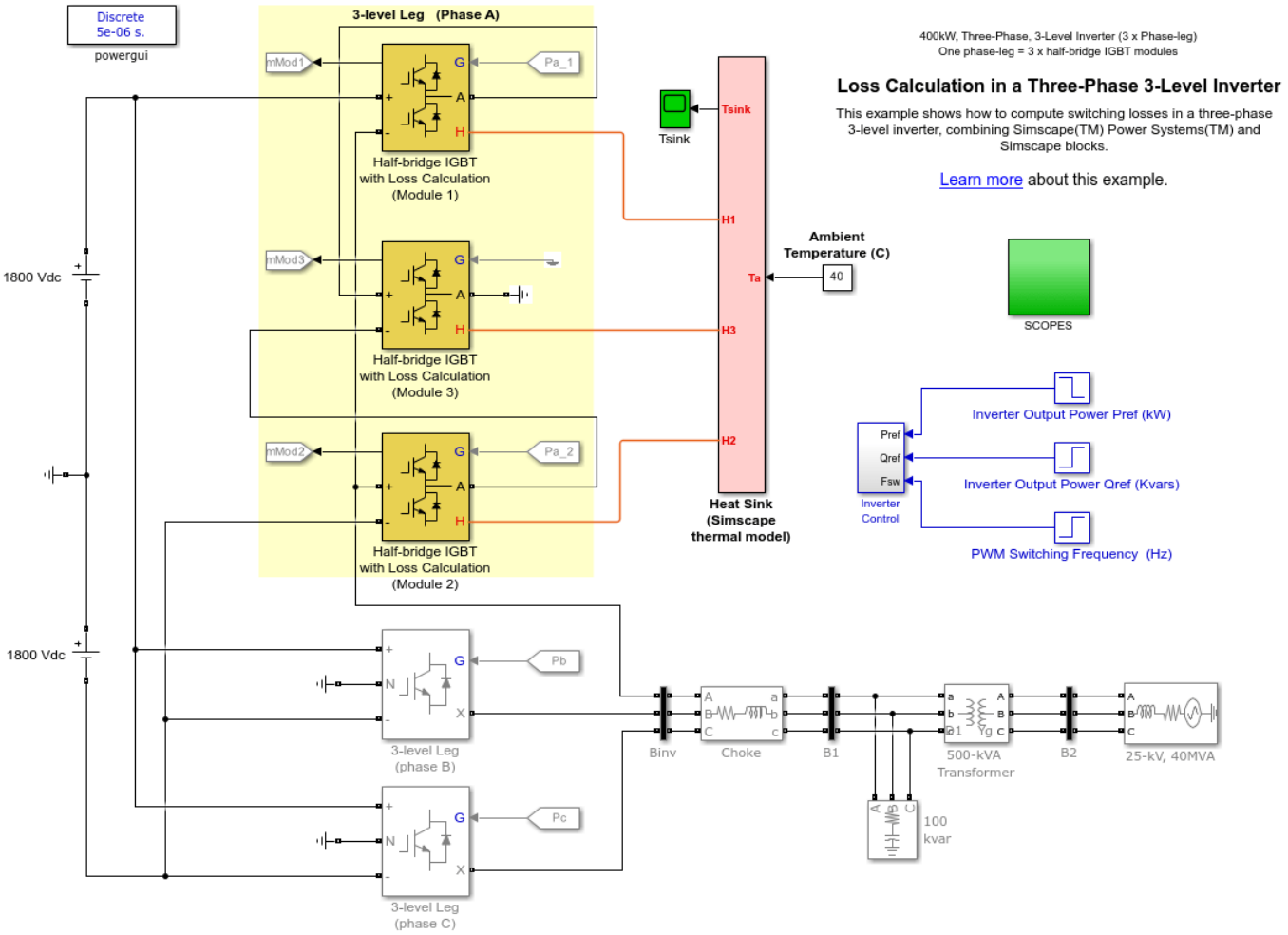



### Description

This switched power supply converts a 30V DC supply into a regulated 15V DC supply. The model can be used to size the inductance L and smoothing capacitor C, as well as to design the feedback controller. By selecting between continuous, discrete time, and fixed-point PI controllers, the impact of discretization and using limited numerical precision can be explored.

# Loss Calculation in a Three-Phase 3-Level Inverter

This example shows how to compute switching losses in a three-phase 3-level inverter, combining Specialized Power Systems and Simscape blocks.




### General

From a +/- 1800 volts DC source, a 400-kW, three-phase 3-level inverter delivers variable power to a distribution power system. The inverter output is connected to the 25-kV, 40 MVA, 50-Hz system through a 2200 V / 25 kV transformer. The inverter topology is based on the model described in [1]. Each 3-level leg of the inverter comprises three commercial half-bridge IGBT modules. The IGBT pulsing of module 3 is not required since only the anti-parallel diodes are operating as neutral clamping diodes. The control system contains two PI controllers (one PQ regulator and one current regulator) to generate the required inverter pulses to achieve the reference output power.

The Phase-A leg is implemented using three Half-bridge IGBT with Loss Calculation blocks. Both switching and conduction losses are calculated and injected into a thermal network. The simulation illustrates the achievable output power versus switching frequency for the three-phase, 3-level inverter.

### The Half-bridge IGBT With Loss Calculation Block

The half-bridge is modeled by two IGBT/Diode blocks. The upper and lower IGBT/Diode blocks are pulsed from an external pulse generator. The loss calculations are based on the specifications found on the manufacturer's data sheets.

The IGBTs losses are computed as follows:

- Turn-on loss: Pre-switching value of the voltage across the device, post-switching value of the current flowing into the device, and the junction temperature are used to determine the energy losses with the help of a 3-D lookup table. This energy is converted into a power pulse which is injected into the thermal network.
- Turn-off loss: Pre-switching value of the current flowing into the device, post-switching value of the voltage across the device, and the junction temperature are used to determine the energy losses with the help of a 3-D lookup table. This energy is converted into a power pulse which is injected into the thermal network.
- Conduction loss: Value of the current (Ic) flowing in the device and its junction temperature determine what would be the saturation voltage (Vce) across the IGBT using a 2-D look-up table. This Vce is then multiplied by Ic to obtain the losses which are injected into the thermal network.

The diode's losses are computed as follows:

- Reverse recovery loss: Pre-switching value of the current flowing into the device, post switching value of the voltage across the device, and the junction temperature are used to determine the energy losses with the help of a 3-D lookup table. This energy is converted into a power pulse which is injected into the thermal network.
- Conduction loss: Value of the current (If) flowing in the device and its junction temperature determine what would be the on-state voltage (Vf) across the diode using a 2-D look-up table. This Vf is then multiplied by If to obtain the losses which are injected into the thermal network.

The thermal capacitance of the device junction as well as its junction-to-case thermal resistance are represented by a one-cell Cauer network modeled with a Simulink® State-space block.

### The Thermal Network

Simscape blocks from the thermal foundation library are used to build a two-cell Cauer network based on the thermal capacitances (case and heat sink) and resistances (case-to-sink and sink-to-ambient). For the sake of simplicity, the two-cell Cauer network uses arbitrary values for the thermal capacitances in order to reduce the time required to simulate the thermal phenomena.

### Demonstration

Run the simulation and observe the following operating points:

- From t=0 sec to t=5 sec: the inverter outputs 372 kW (power factor = 0.85) using a switching frequency of 850 Hz. The converter total losses are 2.7 kW and the highest junction temperature (125 C) is observed on IGBT1 of Module 1 (or IGBT2 of Module 2). See Tj(Celsius) Scope block inside the Additional Scopes & Measurements block.

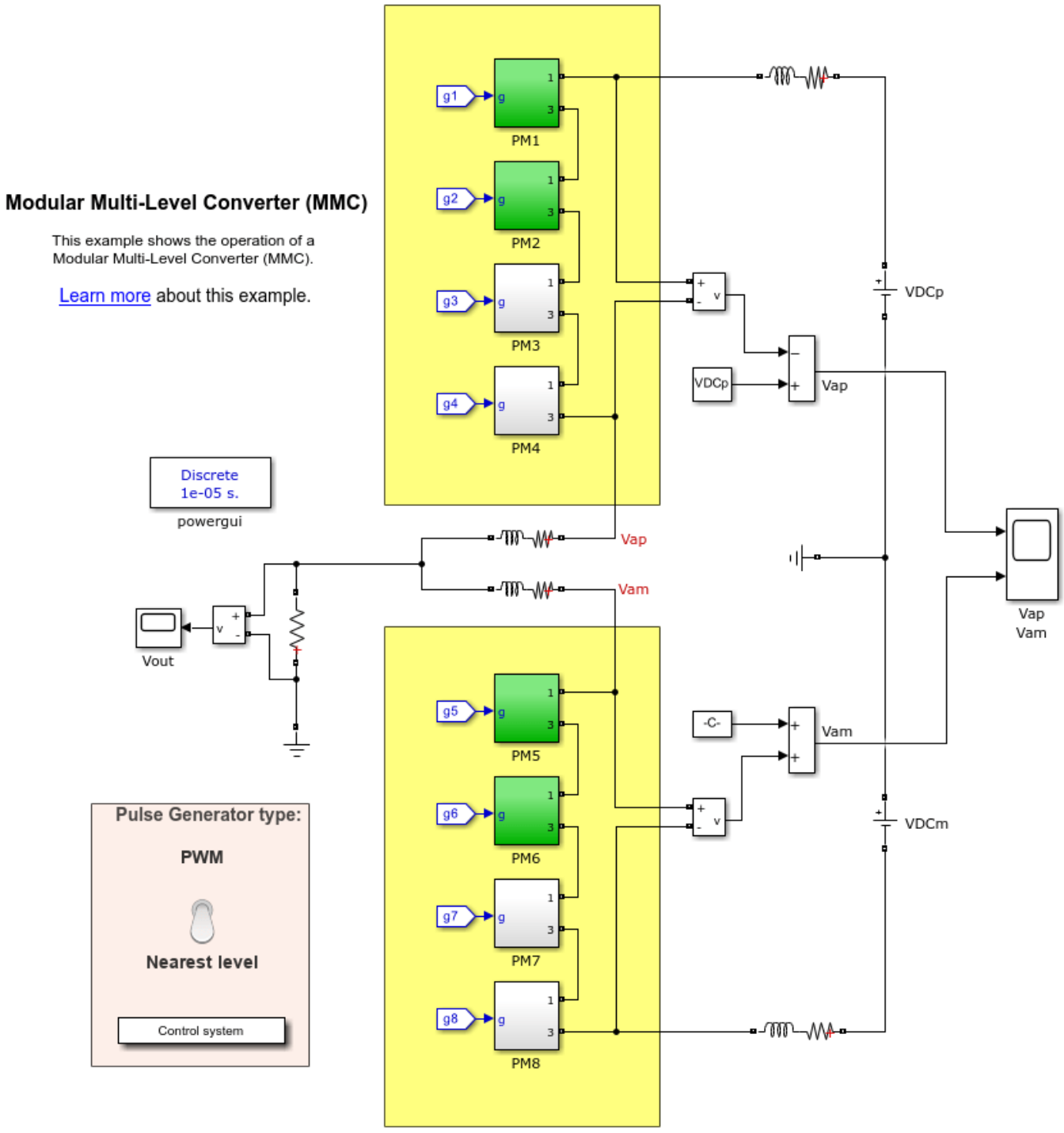
- From t=5 sec to t=12 sec, the inverter outputs 210 kW (power factor = 0.85) using a switching frequency of 1850 Hz. The converter total losses are 2.7 kW and the highest junction temperature (125 C) is still observed on IGBT1 of Module 1.

### Reference

[1] Raffael Schnell, Manager Application, ABB Switzerland, "High-Voltage Phase-Leg Modules for Medium Voltage Drives and Inverters".

# Modular Multi-Level Converter (MMC)

This example shows the operation of a Modular Multi-Level Converter (MMC).



### Description

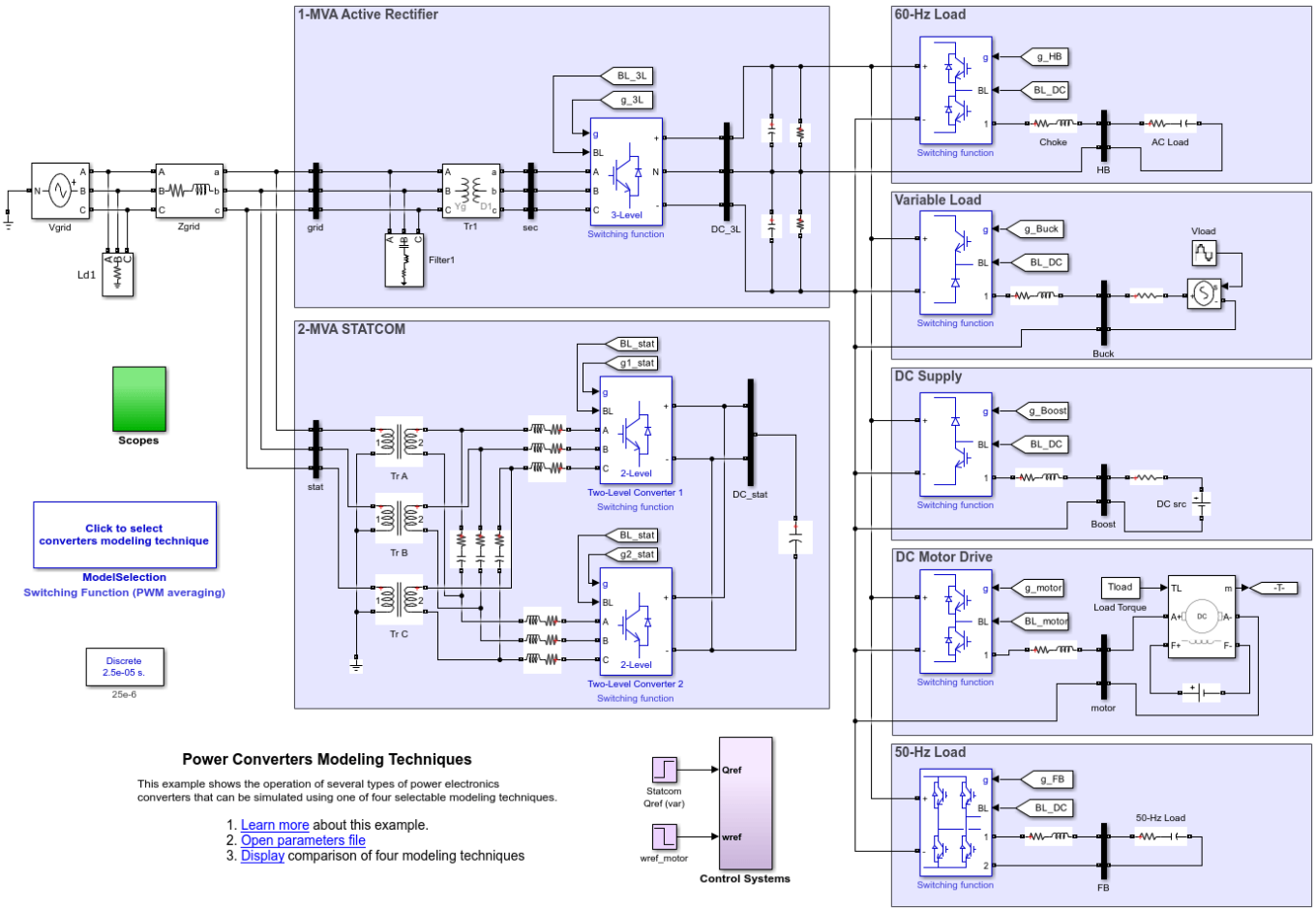
This example uses the half-bridge arm blocks to model an MMC consisting of eight power modules. The control system allows you to choose between two types of pulse generators - PWM and Nearest level. You can toggle the switch to see the impact on the output signals of the converter.

The use of MATLAB® functions within PM1 to PM8 subsystems allows coloring the modules in red when they are activated during the simulation. The use of these functions also allows us to slow down Simulink® in order to observe the activation and deactivation of the modules.

Note that only one half-bridge block can be used to model the positive half-arm and/or negative half-arms of the converter.

# Power Converters Modeling Techniques

This example shows the operation of several types of power electronics converters that can be simulated using one of four selectable modeling techniques




## Modeling Techniques Description

You can run this simulation using one of the following modeling techniques:

1. Switching devices: The converter is modeled with IGBT/diode pairs controlled by firing pulses produced by a PWM generator.
2. Switching function: The converter is modeled by a switching function controlled by firing pulses produced by a PWM generator.
3. Switching function (PWM averaging): The converter is modeled by a switching function controlled by firing pulses produced by a PWM generator with pulse averaging. The g input then expects signals values between 0 and 1.
4. Average model (Uref-controlled): The converter is modeled using a switching-function model directly controlled by the reference voltage. A PWM generator is not required.

5 Average model (Uref-controlled, no rectifier mode): The converter is modeled using voltage sources directly controlled by the reference voltage. A PWM generator is not required, and the rectifier mode is not simulated.

Technique 1 is the most accurate, while technique 5 yields to the fastest simulation. Technique 3 (with firing pulses averaged), as well as techniques 4 and 5 are well-suited for real-time simulation.

### Power Converters Description

The simulation allows you to observe operation of several types of power electronics converters:

1-MVA Active Rectifier: This active rectifier will produce the main DC supply (+/-500V) used by several other converters. The rectifier consists of a three-phase, 3-level NPC converter and a closed-loop control system. It can take or give back power to the grid in order to maintain the specified DC level.

60-Hz Load: The load is modeled using a half-bridge converter controlled by a PWM generator having a carrier frequency of 33*60.

DC Variable Load: The load variation is achieved using a buck converter and a variable DC source at the converter output.

DC Supply: A boost converter transfers power (125 kW) from a 500V DC source to the main DC supply.

DC Motor Drive: The drive consists of a speed-regulated 200-HP motor, a Two-Quadrant DC-DC Converter and a control system.

50-Hz Load: The load is modeled using a full-bridge converter controlled by a PWM generator having a carrier frequency of 1650 Hz and a modulation index of 0.9.

2-MVA STATCOM: This distribution STATCOM consists of two three-phase, 2-level converters (twin topology) and a closed-loop control system. It can generate or absorb 2 Mvars from the grid.

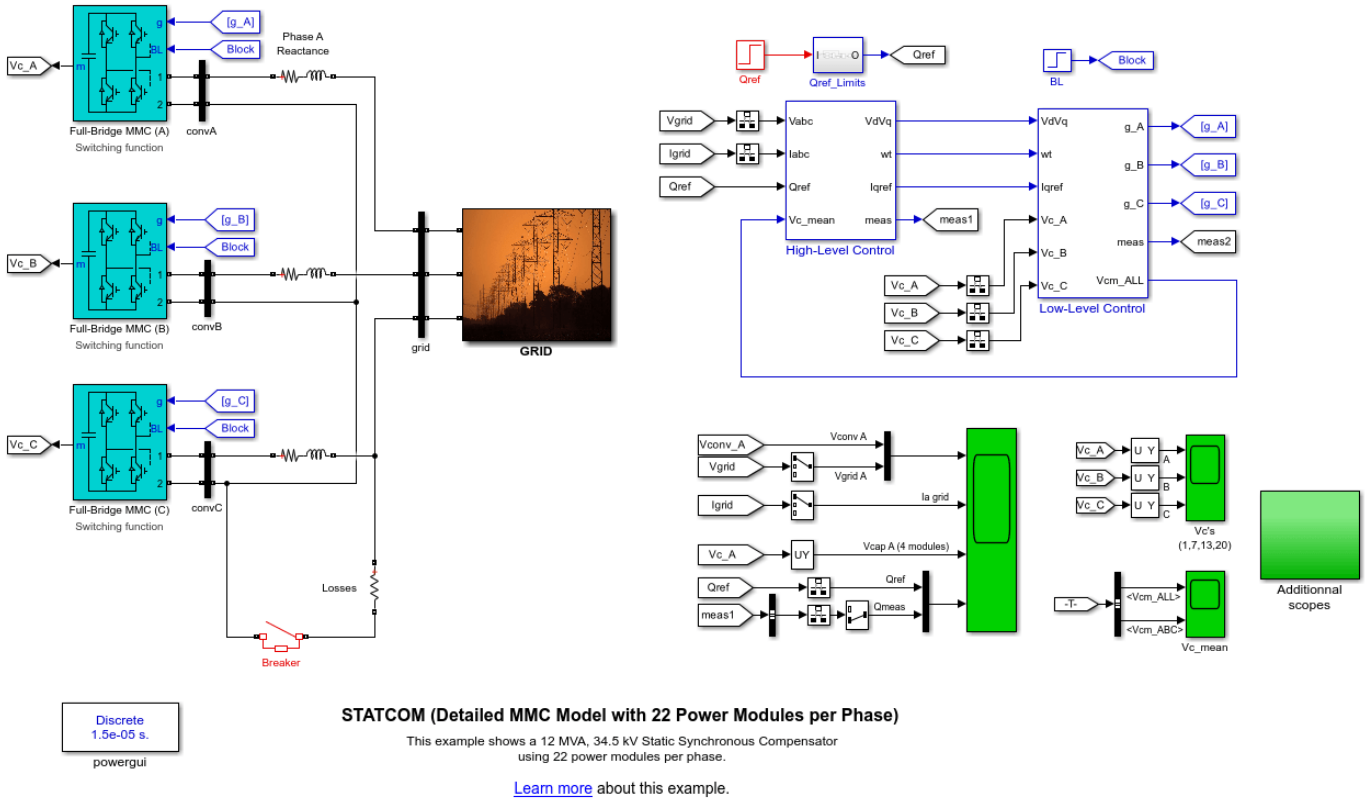
### Simulation

During the simulation, the DC variable load will vary from 125 kW to 350 kW at 5 Hz. At 0.5 s, the DC motor speed setpoint is changed from 1200 to 800 RPM. At 0.6 s, the STATCOM reference (Qref) is changed from -1 Mvar to +1.5 Mvar. Run the simulation and observe the resulting signals on the various scopes. Select a different modeling technique and rerun the simulation, comparing results with previous runs.

# STATCOM (Detailed MMC Model with 22 Power Modules per Phase)

This example shows a 12 MVA, 34.5 kV Static Synchronous Compensator using 22 power modules per phase

Pierre Giroux (Hydro-Quebec)




## Description

In electrical grids, shunt compensation is often used for reactive power and voltage control. This example models a shunt compensation device that is increasingly used in modern grids: the Modular-Multi-level (MMC) STATCOM. The MMC-STATCOM is built using the Full-Bridge MMC block to represent a power electronics converter of 22 modules per phase. In order to speed-up simulation while keeping simulation fidelity, the Switching Function model is selected.

## Principle of Operation

The STATCOM can absorb or generate reactive power. The reactive power transfer is done through the phase reactance. The converter generates a voltage in phase with the grid voltage. When the amplitude of the converter voltage is lower than that of the grid voltage, the STATCOM acts like an inductance absorbing reactive power. When the amplitude of the converter voltage is higher than that of the bus voltage, the STATCOM acts like a capacitor generating reactive power.

### Simulation 1: Dynamic Response

Run the simulation and observe waveforms on Scope1. You can see that the simulation starts in steady state and that the STATCOM operates in inductive mode following its set-point Qref (-5 Mvar ). At 0.1 second, the set-point changes from -5 to +10 Mvar. The STATCOM control system reacts very rapidly to modify the inverter output voltage in order to generate 10 Mvar of reactive power (capacitive mode).

### Simulation 2: Low-Level Control - Capacitor Voltage Balancing

In order to analyze the operation of the low-level control (DC voltage balancing) run the simulation with the following changes: 1. Set the simulation time to 2 seconds. 2. Set the initial value of Qref (red Step block) to 10e6. With this modification, the setpoint will not change during the simulation. 3. Set the Switching time of the red Breaker block to 1 second. This will produce a power unbalance at 1s by switching-in a small resistive load on phase C only. 4. Run the simulation and observe some of the capacitor voltages on the Vc's & Vc_mean Scopes, located inside the green Additional scopes subsystem. You can see that the low-level control system performs well to maintain the capacitor voltages balanced. 5. Double-click on the blue Low-Level Control block and turn-off the Phase Balancing regulator of the DC voltage balancing system. Run the simulation and notice that the STATCOM DC voltages balance between phases is lost when the load is switched-in on phase C. Note: The individual capacitor voltages balancing control generates a small voltage in phase/180 degrees phase-shift with the current flowing into the arm in order to produce/absorb active power from each capacitor. This control system will then not work properly if the current (capacitive or inductive) flowing into the capacitors is too low to produce sufficient active power to charge/discharge them. For this reason, the Qref_Limits subsystem does not permit Qref values between -0.75 and +0.75 Mvar. See reference [2] for more details on this control. Other voltage balancing approaches, such as a sorting algorithm, do not have this limitation. This method is used in the example power_Grid_STATCOM.

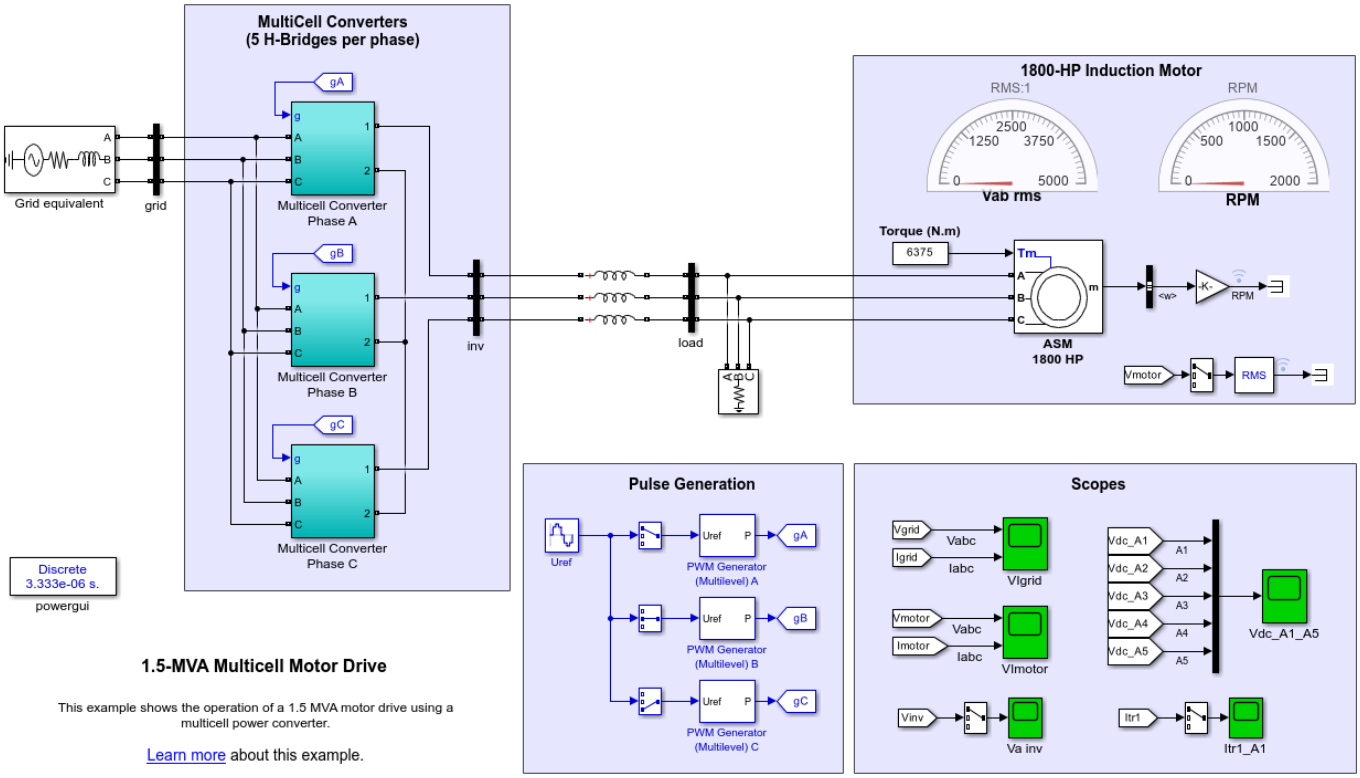
### References

[1] M. Pereira, Member, IEEE, D. Retzmann, Member, IEEE, J. Lottes, M. Wiesinger, G. Wong, SVC PLUS: An MMC STATCOM for Network and Grid Access Applications 2011 IEEE Trondheim PowerTech

[2] Hirofumi Akagi ; Shigenori Inoue ; Tsurugi Yoshii, Control and Performance of a Transformerless Cascade PWM STATCOM with Star Configuration IEEE Transactions on Industry Applications (Volume: 43, Issue: 4 , July-august 2007)

# 1.5-MVA Multicell Motor Drive

This example shows the operation of a 1.5-MVA motor drive using a multicell power converter.




### Description

A Full-Bridge MMC with External DC Link block is used to feed a 1800-HP induction motor. this block is programmed to model five H-bridges in series. Phase-shifting carrier-based PWM technique is used to control the H-bridges.

On the rectifier side, a phase-shifting transformer and a three-phase diode bridge is used to feed each individual DC link. Using phase-shifting transformer allows significantly reducing current harmonics injected into the distribution grid.

In order to cancel harmonics produced by the five six-pulse rectifier units in each phase, voltages applied to the AC side of individual rectifiers must be phase shifted by 60/5 = 12 degrees with each others. These phase shifts are specified in the PhaseShift parameter defined in the model pre-load function (PhaseShift = [12,24,36,48,60])

### Simulation

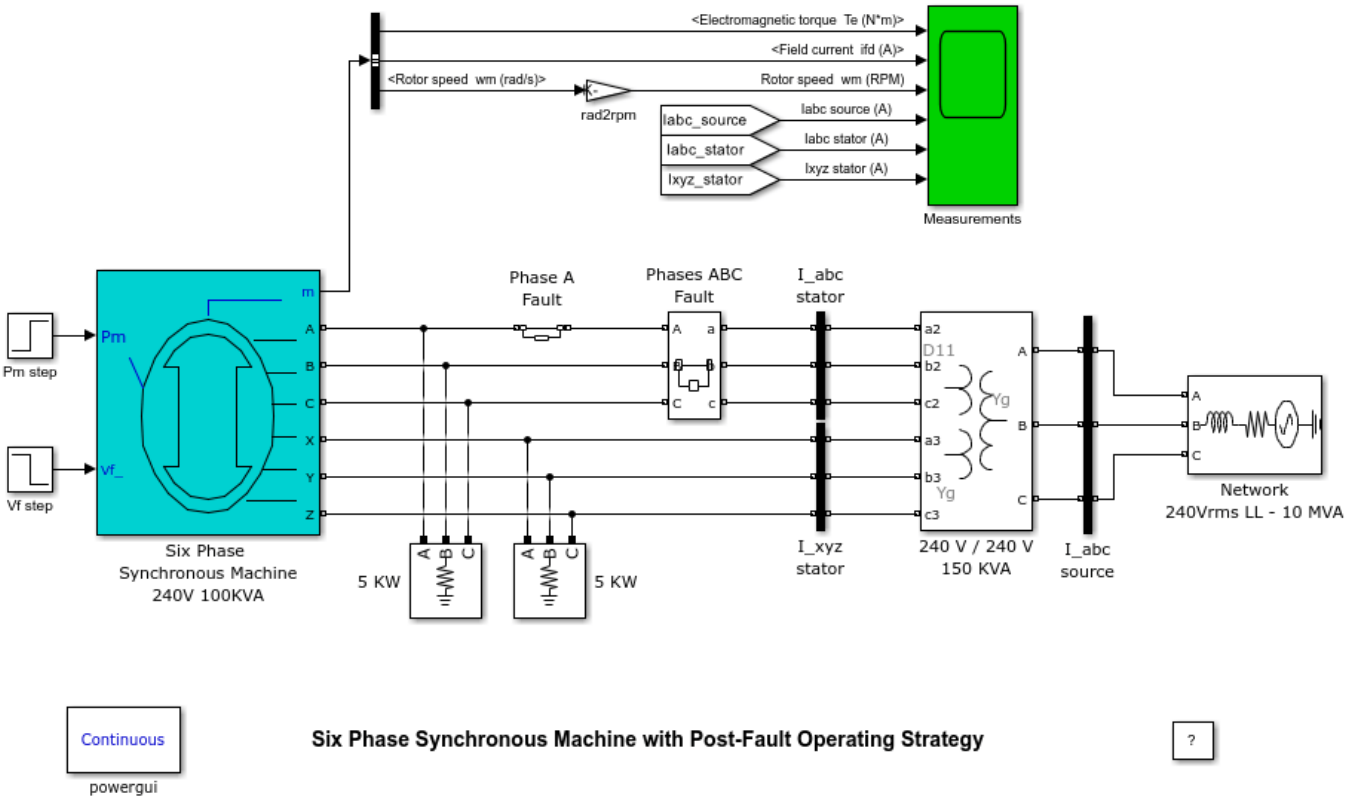
Run the simulation and observe waveforms on the various scopes. The harmonics cancellation process using phase-shifting transformers works fairly well.

Now, set PhaseShift=ones(1,5) in the MATLAB® workspace to program the same phase-shifting to all transformers. Rerun the simulation. You will see that the currents injected into the grid now have very high harmonics content.

# Six-Phase Synchronous Machine with Post-Fault Operating Strategy

This example shows the use of a 100 KVA Six-Phase Synchronous Machine under normal and fault operation.

J-F. Doyon and Louis-A. Dessaint (Ecole de Technologie Superieure, Montreal)




### Description

A 100KVA, 240V, 1800 RPM six-phase synchronous motor is supplied by a three-windings, three-phase transformer connected to a 240V network with a short-circuit level of 10 MVA. Full load is initially applied to the shaft's machine and the excitation voltage is set to 0.6V, which represents the voltage value viewed from the stator so that the machine can produce its nominal phase current at full load.

### Simulation

1. The machine starts under normal operation (all healthy phases).

2. At t = 4 seconds, phase A is disconnected. The Ia and Ib currents become dephased by a 180 degrees value to ensure a sum of currents in the stator equal to zero. Such a behavior creates a lot of ripple on the electromagnetic torque. This ripple is eliminated as soon as the ABC stator is disconnected from the circuit.

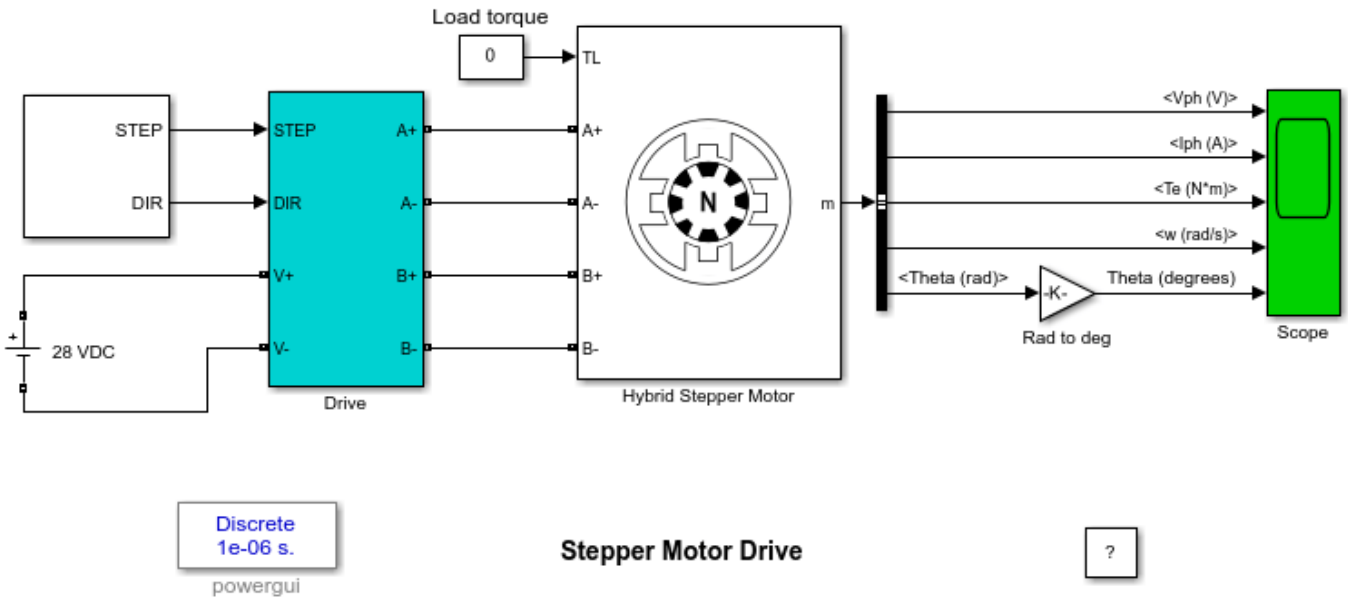
3. 50 ms later, the ABC stator is entirely disconnected from the circuit to compensate the phase A loss. Also, the load is reduced to half its nominal value and the excitation voltage Vf is decreased to 0.4V so the remaining XYZ stator keeps a nominal phase current value.

4. The system quickly stabilizes.

# Stepper Motor Drive

This example shows a model of a hybrid stepper motor drive.

Hoang Le-Huy, Laval University




### Description

This example presents a stepper motor drive using the Hybrid Two-Phase model selected among the options on the dialog window. The motors parameters are those of a small stepper motor (size 23). The motor phases are fed by two H-bridge MOSFET PWM converters. The DC bus is represent by a 28-V DC voltage source. The motor currents are independently regulated by two hysteresis-based controllers that generate the MOSFETs drive signals by comparing the measured currents with their references. The ripple in the current waveforms is controlled by the hysteresis band of the comparators. The switching frequency is variable and dependent on the motor parameters.

In this example, single-phase-on excitation scheme is used because of its simplicity. Square-wave current references are generated using the current amplitude and the step frequency parameters specified in the dialog window. The movement of the stepper drive is controlled by the STEP and DIR signals received from Signal Builder block.

### Simulation

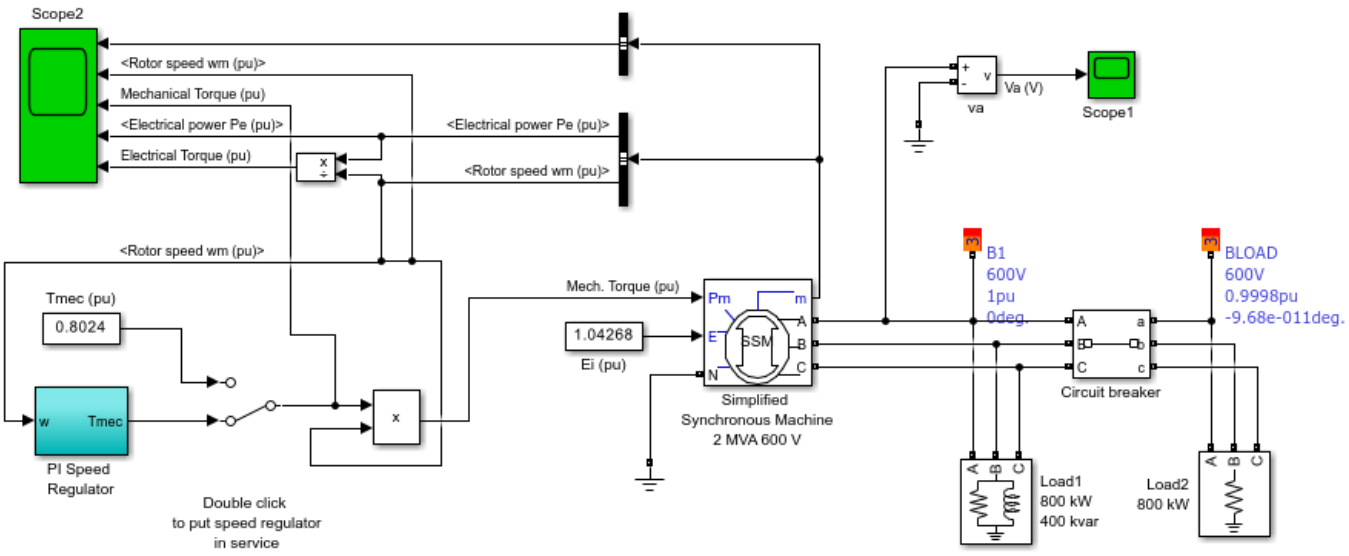
The current amplitude and the stepping rate are selected in the dialog mask to be 2A and 500 step/s, respectively. The STEP signal from the Signal Builder block controls the movement of the stepper drive. A positive value (1.0) will make the motor rotating and a zero value will stop the rotation. The DIR signal controls the rotation direction. A positive value (1.0) will impose the positive direction while a zero value will impose the reverse direction. The stepper motor drive operation is illustrated by the main waveforms (voltages, currents, torque, speed and position) displayed on the Scope block. The simulation is done using a fixed-step solver with a sampling time of 1 us, providing acceptable

accuracy for the PWM. If a high PWM accuracy is required, a smaller time step can be used but the simulation will be slower.

# Simplified Synchronous Machine - Speed Regulation

This example shows a load shedding test on a 2000 kVA, 600 V alternator.

G. Sybille (Hydro-Quebec)



## Description

A three-phase, four-wire alternator rated 2000 kVA, 1600 kW, 0.8 power factor, 600 V, 1800 rpm is connected to a 1600 kW, 400 kvar inductive load. The stator neutral point is grounded. The internal impedance of the generator (Zg = 0.0036 + j*0.16 pu) represents the armature winding resistance Ra and direct axis transient reactance X'd. The total inertia constant of the generator and prime mover is H = 0.6 s, corresponding to J = 67.5 kg.m^2. Speed regulation is modeled with Simulink® blocks implementing a PI regulator. The machine is excited with a constant voltage.

A three-phase breaker is used to switch out a 800 kW resistive load. The breaker is initially closed and it is opened at t = 0.2 s, resulting in a 50% load shedding.

## Simulation

### 1. Initializing the machine to start in steady state

Open the powergui and select 'Machine Initialization'. A new window appears. Note that the machine 'Bus type' is set to 'Swing Bus'; this is because the circuit contains no voltage source imposing the reference angle. The initialization is then performed with the machine controlling the voltage and the angle at its terminals. The desired terminal voltage is initialized to 600 V (machine nominal voltage) and the phase angle of UAN angle to 0 degree.

Press the 'Compute and Apply' button. Once the initialization is solved the three line-to-line machine voltages as well as the three machine output currents are updated. The machine active and reactive powers as well as the required mechanical power Pmec and field voltage E are also displayed. You should read the following values: P = 1600kW, Q = 400 kvar, Pmec = 1604.7kW (0.8024 pu), field voltage E = 1.0247 pu. The Machine initialization tool also prompts you with a warning to set initial condition for mechanical power to 0.8024 pu. This is because the Pm input of the machine is not connected to a constant block or to a library block (HTG or STG). Note that the mechanical torque (Tmec block) is set at 0.8024 pu. Disregard this message. Note also that the constant block Ei connected at the E input of the machine is automatically updated (E = 1.04268 pu).

**2. Simulation at constant torque - No speed regulator**

Make sure that the speed regulator is not in service (Manual Switch is in the upper position) . Start the simulation and observe signals on Scope2. The three Iabc currents should start with steady-state sinusoidal waveforms. Observe that when the circuit breaker opens at t = 0.2 s, the electrical power (trace 4) drops from 0.8 pu to 0.4 pu and that the machine starts to accelerate.The rate of speed increase is dN/dt = 1/2H = 0.833 pu speed/ pu torque / s. As the net electromechanical torque is now Tmec-Telec = 0.8 - 0.4 = 0.4 pu, the speed increases at a rate of 0.833*0.4 = 0.33 pu/s. At t = 1.2s, the expected speed increase is therefore 0.33 pu. In fact, the speed measured at t = 1.2 s is slightly higher than the theoretical value (1.38 pu as compared to 1.33 pu). This is because the electrical torque (trace 5) decreases as speed is increasing, resulting in a net acceleration torque higher than 0.4 pu.

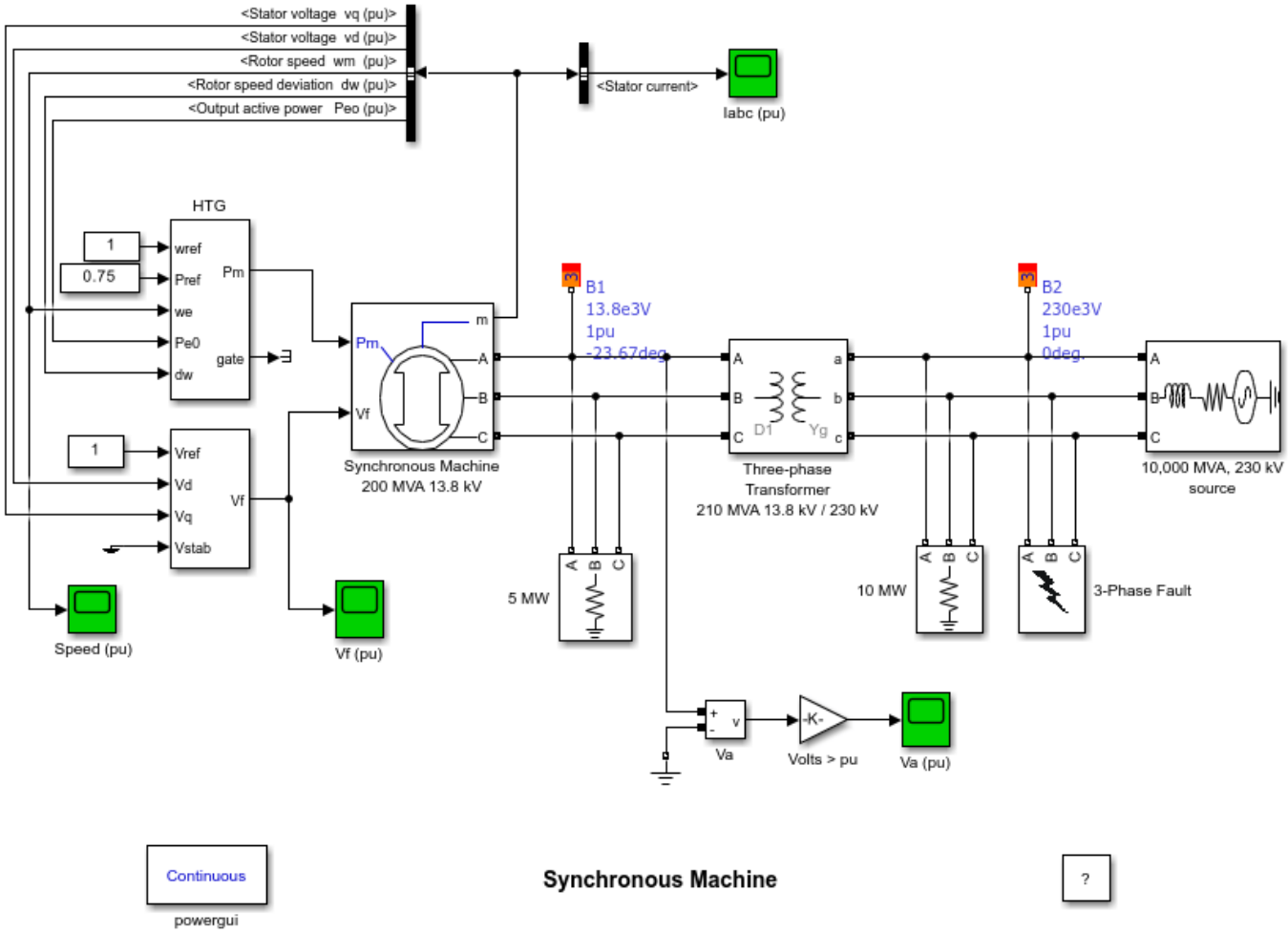
**3. Simulation with speed regulator**

Now double click on the Manual Switch block in order to put the speed regulator in service. Restart the simulation and observe the dynamic response of the speed regulator on Scope 2. Notice that in order to maintain speed at its reference value (1 pu), the speed regulator has reduced the mechanical torque to 0.4 pu.

# Synchronous Machine

This example shows the use of the Machine Initialization tool of Powergui to initialize machine currents.

Louis-A. Dessaint and R. Champagne (Ecole de Technologie Superieure, Montreal)




## Description

A three-phase generator rated 200 MVA, 13.8 kV, 112.5 rpm is connected to a 230 kV, 10,000 MVA network through a Delta-Wye 210 MVA transformer. At t = 0.1 s, a three-phase to ground fault occurs on the 230 kV bus. The fault is cleared after 6 cycles (t = 0.2 s).

## Simulation

**1.** Open the Powergui and select 'Machine Initialization'. A new window appears. The machine 'Bus type' is initialized as 'PV generator', indicating that the initialization is performed with the machine controlling the active power and its terminal voltage. The desired terminal voltage parameter is set to 13800 and the active Power to 150e6.*

Press the 'Compute and Apply' button. The phasors of AB and BC machine voltages as well as the currents flowing out of phases A and B are updated. The machine reactive power, mechanical power and field voltage requested to supply the electrical power is also be displayed: Q = 3.4 Mvar; Pmec = 150.32 MW (0.7516 pu); field voltage Ef = 1.291 pu.

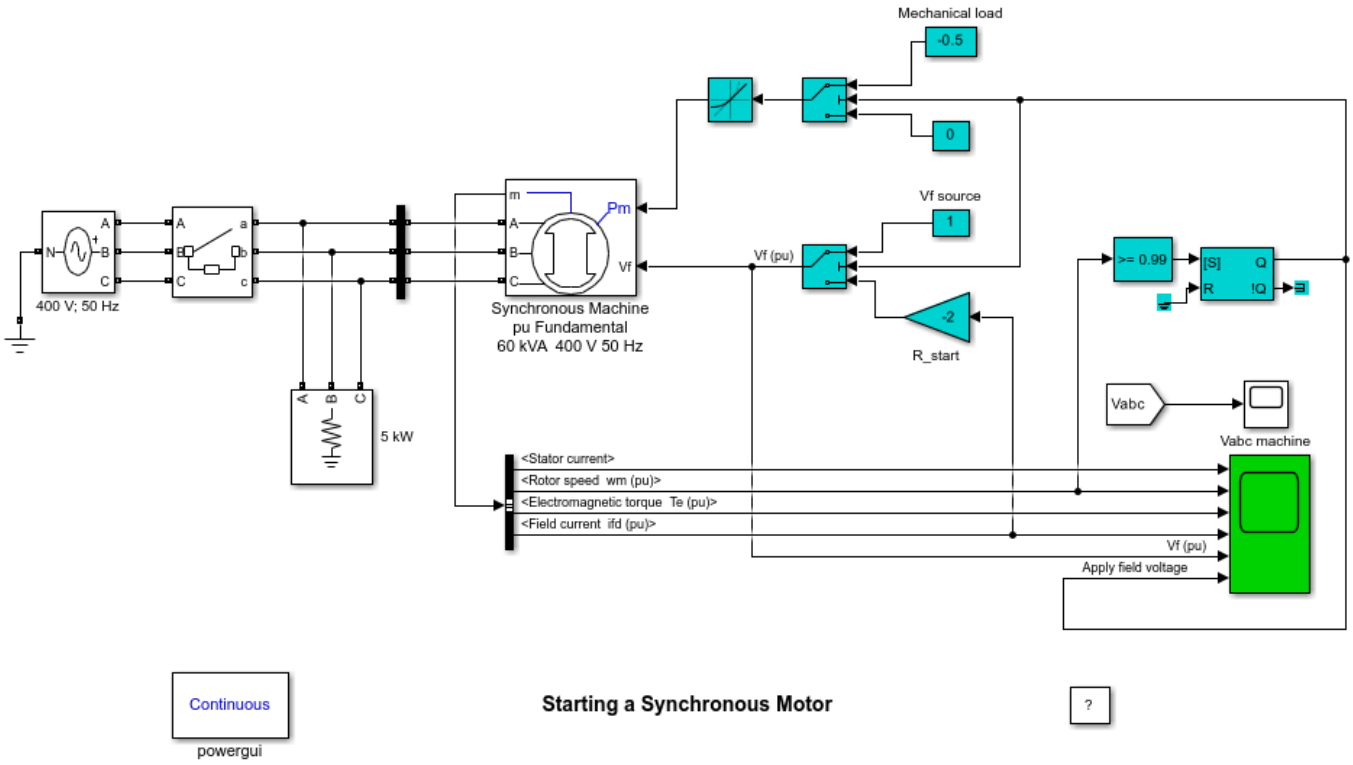
**2.** In order to start the simulation in steady state with the HTG and excitation system connected, these two blocks are initialized according to the values calculated by the Machine Initialization tool. This initialization is automatically performed as long as you connect at the Pm and Vf inputs of the machine either Constant blocks or regulation blocks from the machine library (HTG, STG, or Excitation System). Open the HTG block menu and note that the initial mechanical power was set to 0.7516 pu (150.32 MW) by the tool. Open the Excitation System block menu and note that the initial terminal voltage and field voltage have been set respectively to 1.0 and 1.29071 pu.

**3.** Open the 4 scopes and restart the simulation. Observe that the terminal voltage Va is 1.0 p.u. at the beginning of the simulation. It falls to about 0.4 pu during the fault and returns to nominal quickly after the fault is cleared. This quick response in terminal voltage is due to the fact that the Excitation System output Vf can go as high as 11.5 pu which it does during the fault. The speed of the machine increases to 1.01 pu during the fault then it oscillates around 1 p.u. as the governor system regulates it. The speed takes much longer than the terminal voltage to stabilize mainly because the rate of valve opening/closing in the governor system is limited to 0.1 pu/s.

# Starting a Synchronous Motor

This example shows the starting procedure for a synchronous motor.

Richard Gagnon (Hydro-Quebec)




### Description

When a synchronous motor is started, the excitation DC voltage is not applied to the field winding. The motor is started in induction machine mode with currents induced in the damper and field windings. A resistor is connected across the field winding in order to produce an acceptable field current and to limit voltage induced across the field winding. Then when speed reaches a preset value near synchronous speed, the field winding is connected to the DC voltage source and the motor synchronizes on the system frequency.

In the synchronous machine model, the field winding terminals are not available. Instead, a Simulink® signal representing the field voltage must be applied at the Vf input of the machine. Therefore if the Vf input is left unconnected, a zero field voltage is applied on the rotor. In other words, the field winding is short-circuited. In this example, the field current (idf) and a gain block (R_start) are used to implement the resistance connected across the field winding.

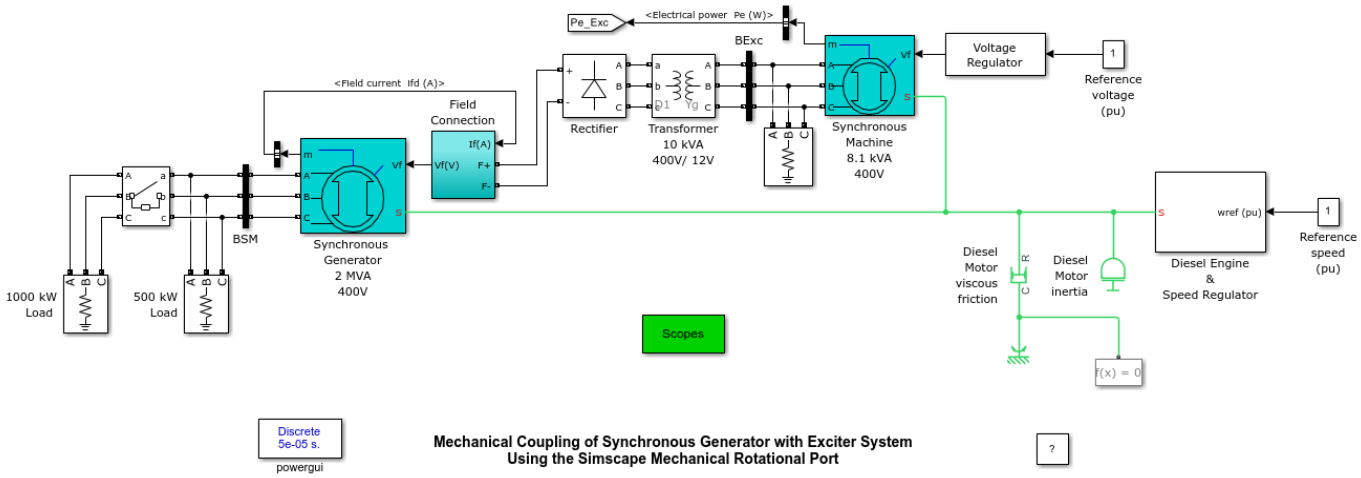
This model illustrates the starting procedure of a 60-kVA 400-V 50Hz synchronous motor. The motor is started at no load by closing the circuit breaker at t=0.1s. A 2 pu resistor is initially connected across the field winding. When the rotor speed reaches 0.99 pu, the "R_start" resistor is disconnected from the field terminals and it is replaced by the "Vf source" (1 pu). At the same time, the mechanical

power is ramped from zero to 50% of the nominal mechanical power (Pm = -0.5 pu) in one second. The motor locks into step at synchronous speed (1 pu) at approximately t = 1.3 s.

# Mechanical Coupling of Synchronous Generator with Exciter System Using the Simscape Mechanical Rotational Port

This example shows a machine mechanical coupling of an excitation system using Simscape™ mechanical rotational ports.

Gilbert Sybille, IREQ. Jean-Francois Doyon, ETS.




## Description

In large alternators, the excitation system is provided by a small synchronous machine connected to the shaft of the synchronous generator. Current rectification is performed by a rotating diode bridge mounted on the generator shaft, thus avoiding slip rings for providing DC power to the synchronous generator field.

The synchronous generator is driven by a diesel motor with speed regulation. The mechanical coupling of the generator, the exciter system, and the diesel motor is done by using the Simscape mechanical rotational ports of the Synchronous Machine blocks.

This model is very similar to the power_SM_exciter model. The only difference is that the two synchronous Machine blocks and the diesel motor use a mechanical rotational port to connect together and represents the mechanical shaft.

## Simulation

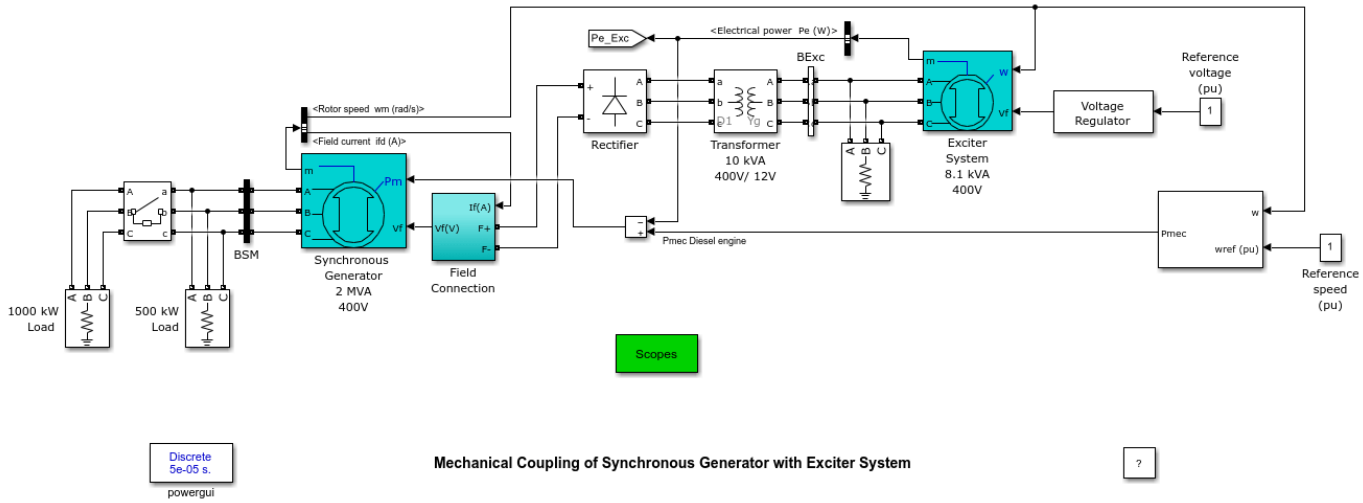
Run the model and compare the simulation results with the ones obtained with power_SM_exciter model. They should be identical.

Note that the inertia and viscous friction are specified for every system (in the mask of the Synchronous Machine blocks, and directly to the mechanical shaft for the diesel motor). This is not the case for the power_SM_Excitation model where the total inertia and viscous friction of the three systems need to be specified in the Synchronous Machine block representing the generator.

# Mechanical Coupling of Synchronous Generator with Exciter System

This example shows a Machine mechanical coupling of an excitation system.

Gilbert Sybille, IREQ.




## Description

In large alternators, the excitation system is provided by a small synchronous machine connected on the same shaft as the main synchronous generator. Current rectification is performed by a rotating diode bridge mounted on the synchronous machine shaft, thus avoiding slip rings for providing DC power to the synchronous generator field. Mechanical coupling of the Synchronous generator and the Exciter is done by using speed as mechanical input for the Exciter machine.

The Exciter is a small synchronous machine rated 8.1 kVA, 400 V, 50Hz, 1500 RPM. A 400V / 12 V transformer is used to adapt the 400 V output voltage of the exciter to the rectifier. In real life this transformer would not be used. Instead, the Exciter output voltage would rather be 12 V. The output of the rectifier bridge is connected directly to the synchronous generator field terminals. Filtering is not required because of the large field inductance.

The synchronous generator is a 2 MVA, 400V, 50 Hz, 1500 rpm machine driven by a diesel motor. A nominal field current (Ifn) of 100A specified in the mask parameters allows using the real voltage applied to the rotor (not the field voltage seen from stator). It results in a nominal field voltage of 9.2837 V. The field terminal voltage of the synchronous generator is measured inside the "Field Connections" subsystem. This subsystem is required to interface the input Vf of the synchronous machine and the real field terminals. It uses a current source driven by the DC current output of the bridge which also corresponds to the DC field current. The voltage appearing across this current source corresponds to the field voltage which must be applied to the Vf Synchronous Machine input.

Voltage regulation of the generator is performed by controlling the field voltage of the exciter. The Diesel engine provides the total mechanical power required by the main synchronous machine and the exciter.

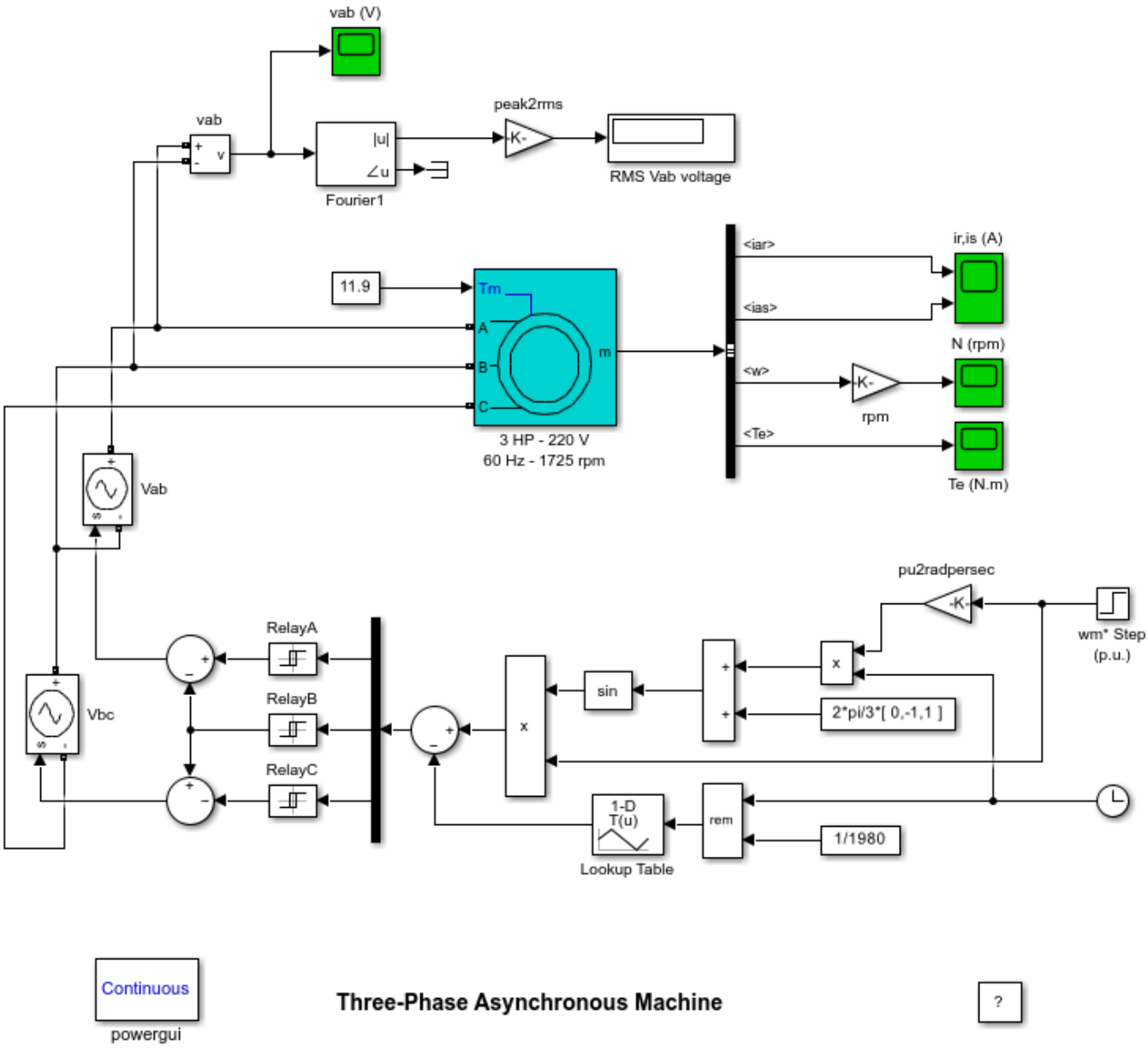
### Simulation

Run the model and observe signals on the Synchronous Generator Scope block. From 0 to 3 sec the synchronous machine delivers 500 kW (25 % of rating). At t=3 sec, an additional 1000 kW is switched on by closing the circuit breaker. In order to keep voltage at 1 pu, the excitation current increases from 112 A to 185 A (trace 1). The field voltage (trace 2) contains a 300 Hz ripple, but this ripple does not appear in the field current because of the large field inductance. Trace 3 shows that after load switching the synchronous machine terminal voltage resumes to its nominal value after a 3 second transient. Traces 4 and 5 show the mechanical output power of the diesel engine and the speed of the engine-generator set. Speed regulation maintains 1 pu speed and nominal frequency (50 Hz output voltage).

# Three-Phase Asynchronous Machine

This example shows the asynchronous machine in an open-loop speed control on a 3 HP 220 V industrial motor.

Louis-A. Dessaint and R. Champagne (Ecole de Technologie Superieure, Montreal)




### Description

A three-phase motor rated 3 HP, 220 V, 1725 rpm is fed by a sinusoidal PWM inverter. The base frequency of the sinusoidal reference wave is 60 Hz while the triangular carrier wave's frequency is

set to 1980 Hz. The PWM inverter is built entirely with standard Simulink® blocks. Its output goes through Controlled Voltage Source blocks before being applied to the Asynchronous Machine block's stator windings. The machine's rotor is short-circuited. Its stator leakage inductance Lls is set to twice its actual value to simulate the effect of a smoothing reactor placed between the inverter and the machine. The load torque applied to the machine's shaft is constant and set to its nominal value of 11.9 N.m.

The motor is started from stall. The speed setpoint is set to 1.0 pu, or 1725 rpm. This speed is reached after 0.9 s.

**Simulation**

Take a look at the simulation parameters. The Maximum time step has been limited to 10 microseconds. This is required due to the relatively high switching frequency (1980 Hz) of the inverter.

Observe that the rotor and stator currents are quite noisy despite the use of a smoothing reactor. The noise introduced by the PWM inverter is also observed in the electromagnetic torque waveform Te. However, the motor's inertia prevents this noise from appearing in the motor's speed waveform.

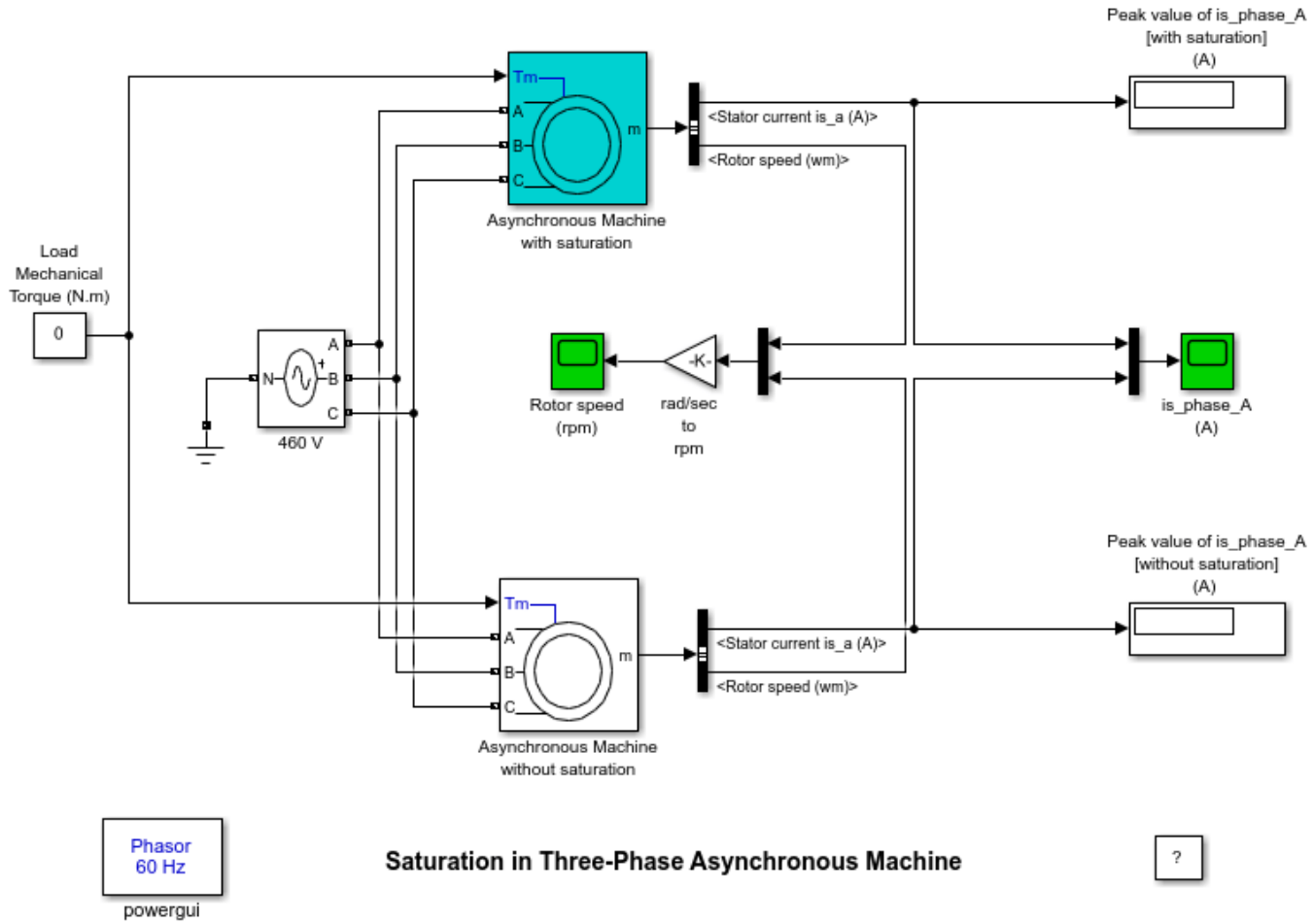
The RMS value of the fundamental component of the line voltage at the machine's stator terminals is measured with a Fourier block.

Finally, observe the PWM inverter's output. Use the zoom on the vab oscilloscope to zoom in on the waveform.

# Saturation in Three-Phase Asynchronous Machine

This example shows the effects of saturation in a three-phase asynchronous motor at various operating conditions.

Jean-Nicolas Paquin and Louis-A. Dessaint (Ecole de Technologie Superieure, Montreal)




### Description

The model shows two identical asynchronous motors rated 50 HP, 460 V, 1800 rpm. In order to compare the linear and saturated model, saturation is implemented in the top motor only. The stator is connected to an ideal three-phase source and load torque is kept constant. Simulation is performed in phasor mode.

### Simulation

**1.** At no load (Tm =0), (three-phase source), simulate and observe the steady-state stator current for different values of stator voltage ranging from 0.5*nominal_voltage (230 Vrms L-L) to 1.5*nominal voltage (690 Vrms L-L). For the saturated motor, you should observe that each operating point lies on the I_versus_V saturation curve specified by the Saturation Parameters. Obviously, a voltage value

below the saturation voltage (first point supplied in the Saturation Parameters) will produce the same current for both motors.

Observe the impact of saturation for different torque values ranging from zero to nominal torque (197 N.m).

Note that during motor start up the 60-Hz component of current observed on the 'is_phase_A' scope model is higher for the saturated model.

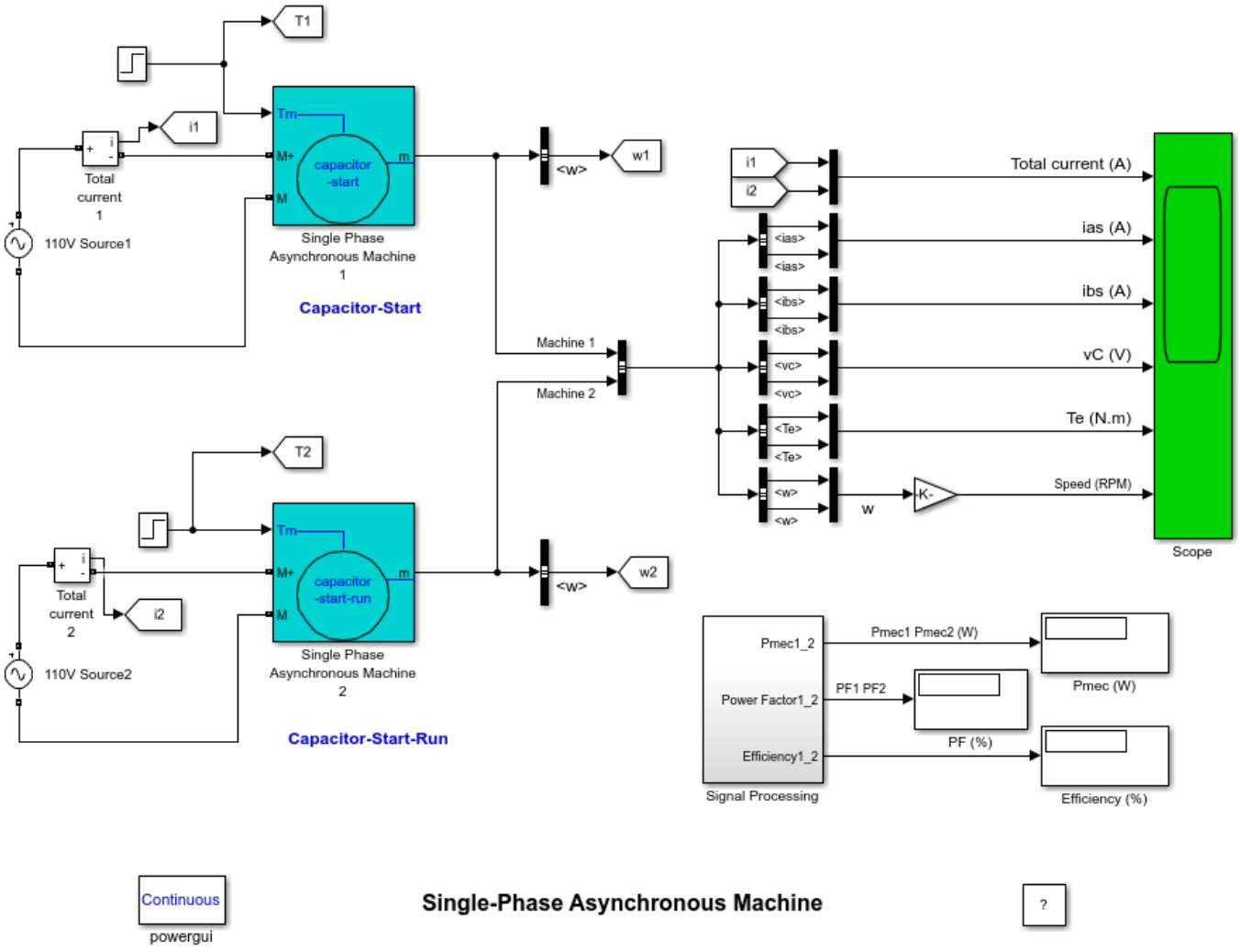
**2.** The locked rotor test is generally used to determine the approximate values of the rotor parameters. In that specific case where the slip is unitary, the impedance of the rotor branch becomes so low that the magnetizing current is negligible compared to the rotor and stator currents. Therefore, the magnetizing branch can be neglected and it can be assumed that the saturation has no influence.

To simulate a locked rotor, specify an infinite inertia (J=inf in the machines' dialog boxes). Simulate with 460V source voltage (nominal voltage) and then 690V (1.5*nominal voltage). Observe both machines' stator currents in steady-state. You should observe stator currents for saturated and unsaturated motors, very close to each other. This shows that when the rotor of an asynchronous motor is blocked, saturation is negligible.

# Single-Phase Asynchronous Machine

This example shows the operation of a single phase asynchronous motor in Capacitor-Start and Capacitor-Start-Run operation modes.

H. Ouquelle and Louis-A.Dessaint (Ecole de technologie superieure, Montreal)




### Description

This model uses two single-phase asynchronous motors respectively in Capacitor-Start and Capacitor-Start-Run modes, in order to compare their performance characteristics, such as torque, torque pulsation, efficiency and power factor. The two motors are rated 1/4 HP, 110 V, 60 Hz, 1800 rpm.and they are fed by a 110V single phase power supply. They have identical stator windings (main and auxiliary) and rotor squirrel cages.

Motor 1 motor operates in capacitor-start mode. Its auxiliary winding, in series with a 255 uF starting capacitor, is disconnected when its speed reaches 75% of nominal speed. The starting capacitor is used to provide a high starting torque.

Motor 2 operates in capacitor-start-run mode. This operation mode uses two capacitors: The run and start capacitors. During the starting period, the auxiliary winding is also connected in series with a 255 uF capacitor but, after the disconnection speed has been reached, the auxiliary winding stays connected in series with a 21.1 uF run capacitor. This capacitor value is optimized to mitigate torque pulsations. The motor operates efficiently with a high power factor.

The two motors are first started at no load, at t=0. Then at t=2 sec, once the motors have reached their steady-state regime, a 1 N.m torque (nominal torque) is suddenly applied on the shaft.

**Simulation**

Start the simulation. The Scope block displays the following signals for the capacitor-start motor (yellow traces) and capacitor-run motor (magenta traces) : total current (main +auxiliary winding), main winding current, auxiliary winding current, capacitor voltage, rotor speed and electromagnetic torque. The mechanical power, power factor and efficiency of motor 1 and motor 2 are computed inside the Signal Processing subsystem and displayed on 3 Display blocks.

During the starting period, as long as the disconnect switch remains closed (from t=0 to t=0.48 s), all waveforms are identical. After opening of the switch, differences are observed as explained below.

**1. Capacitor-Start:**

Observe the 120 Hz torque pulsations which produce 120 Hz mechanical vibrations of the rotor and decrease the motor efficiency. Peak to peak torque ripple is about 3 N, or 300% of rated load when the motor is operating at no load. Observe that the starting capacitor remains charged at its peak voltage when the auxiliary winding is switched off.

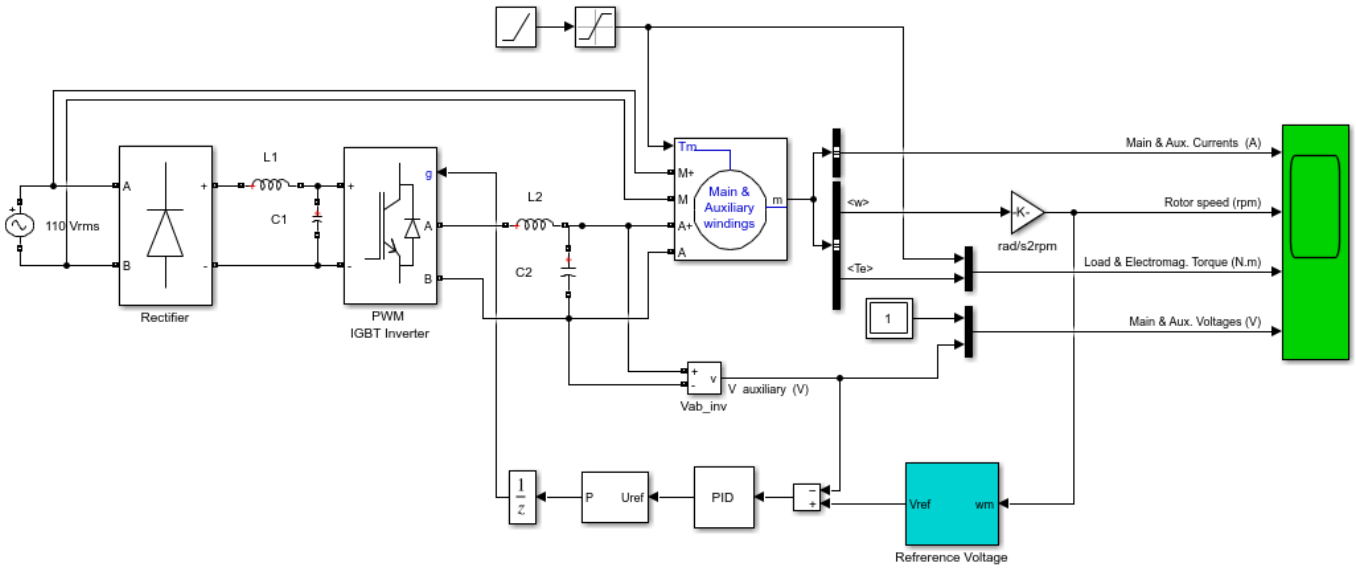
**2. Capacitor-Start-Run:**

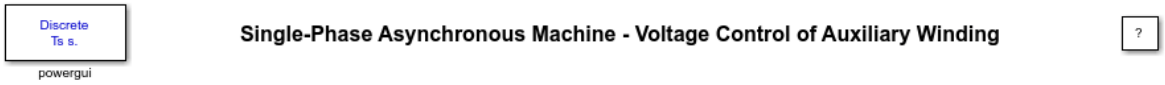
Observe that the torque pulsations are substantially reduced. The run capacitor value has been optimized to minimize torque pulsations at full load. Torque pulsations magnitude is 2 N.m peak to peak (200 % of rated torque) at no load, while it is only 0.04 N.m peak to peak (4 % of rated torque) at full load. Power factor and efficiency at full load (respectively 90 % and 75%) are higher than with the capacitor-start motor (respectively 61 % and 74 %).

# Single-Phase Asynchronous Machine - Voltage Control of Auxiliary Winding

This example shows a drive acting on the auxiliary winding of a single-phase synchronous machine.

H.Ouquelle and Louis-A.Dessaint (Ecole de technologie superieure, Montreal)




### Description

The motor's main winding is directly connected to the 110 V utility supply. The auxiliary winding is fed by a H-bridge IGBT inverter connected to a rectifier.

The reference values of the magnitude and phase of voltage applied to the auxiliary winding are computed by the Reference Voltage block [1]. By varying the auxiliary winding voltage magnitude and phase, the torque ripples are alleviated at all operating points. The voltage control loop uses a proportional-integral-derivative (PID) controller.

### Simulation

Start the simulation. The machine starts at no load and then at t=1.0 sec, once the machine has reached its steady state, the load torque is increased to its nominal value (1 N.m) in 0.5 sec.

Observe that torque ripples are substantially mitigated at all operating points. Also observe the auxiliary and winding currents. These currents are in quadrature and their magnitude ratio is equal to the ratio N of number of turns of auxiliary and main windings.

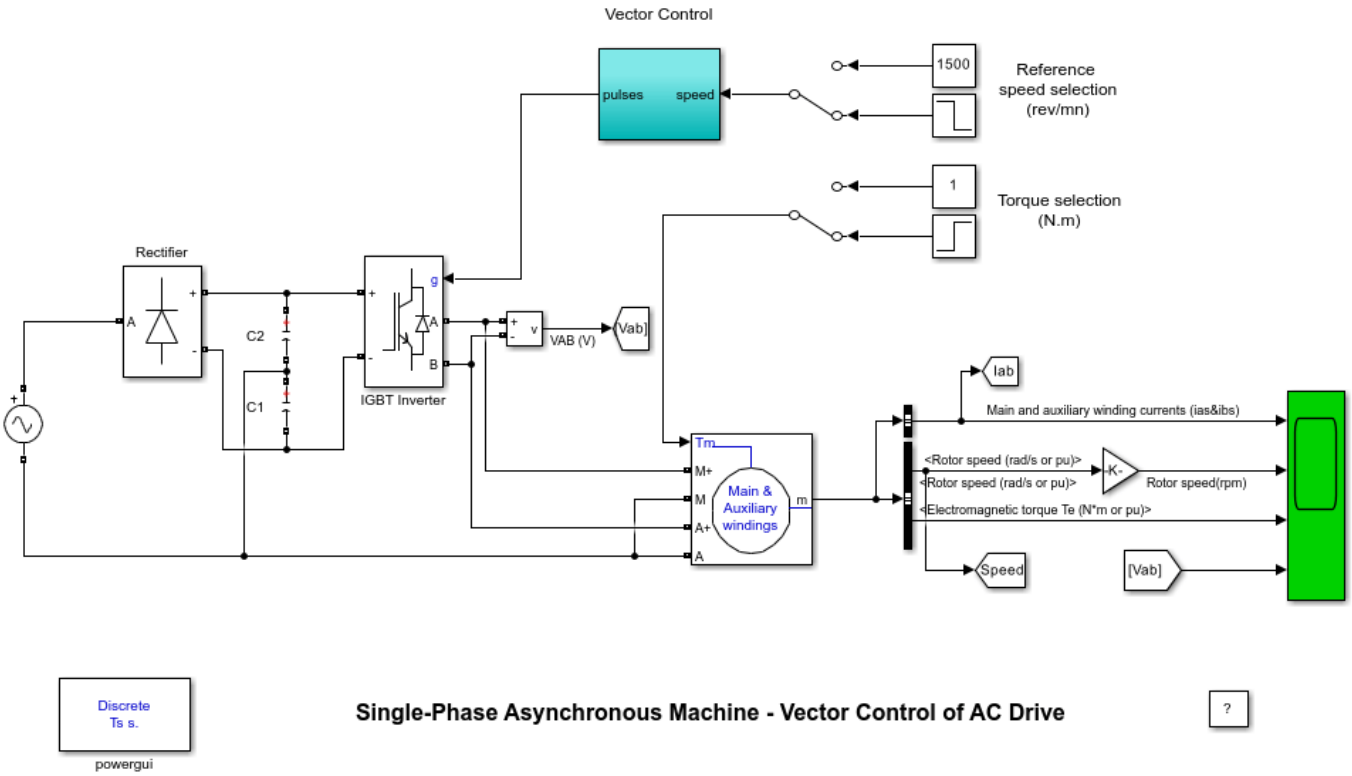
### Reference

[1] COLLINS, E.R., PUTTGEN, H.B., and SYLE, W.E.'Single-Phase Induction Motor Adjustable Speed Drive: Drive Phase Angle Control Of the Auxiliary Winding Supply'. IEEE® Industrial Application Society annual meeting conference records, pp.246-252

# Single-Phase Asynchronous Machine - Vector Control of AC Drive

This example shows the vector control of an AC drive.

H.Ouquelle and Louis-A.Dessaint (Ecole de technologie superieure, Montreal)




### Description

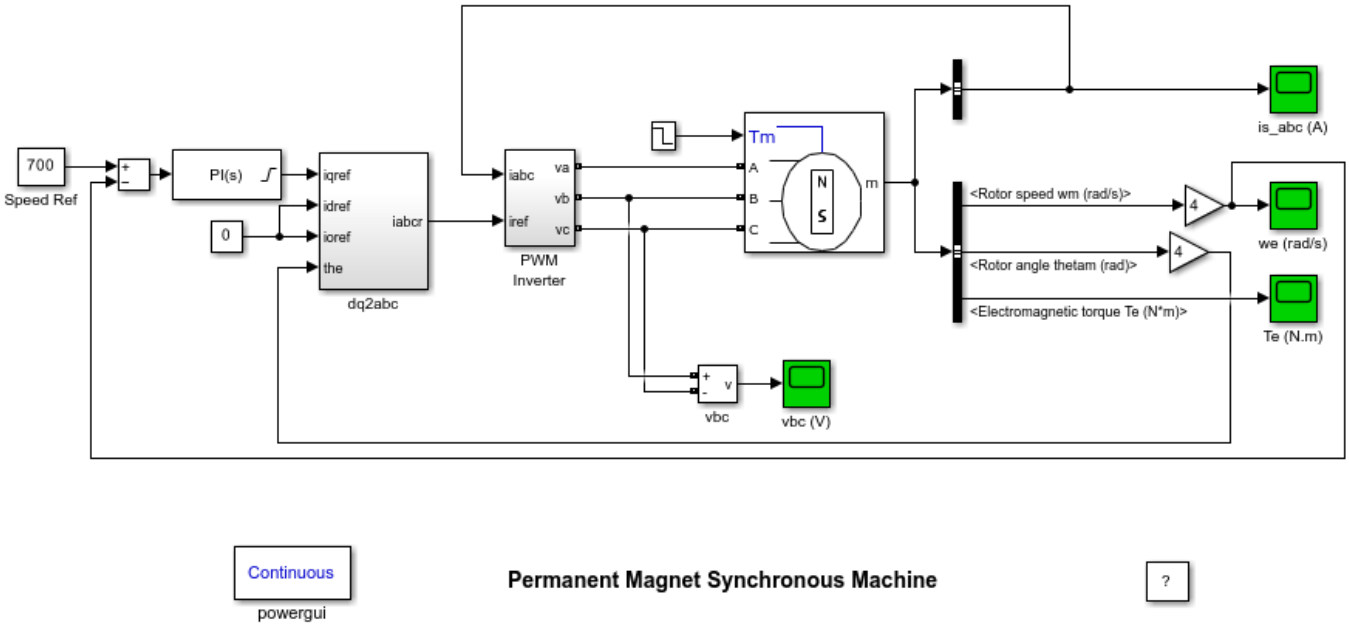
A single phase asynchronous machine rated 1/4 HP, 110 V, 60Hz is fed by a current-controlled PWM inverter which is built using a Universal Bridge block. The speed control loop uses a proportional-integral-derivative regulator to produce the quadrature-axis current reference iq* which controls the motor torque. The motor flux is controlled by the direct-axis current reference id*. The block DQ-AB is used to convert id* and iq* into current references ia*, and ib* for the current regulator. Current and Voltage Measurement blocks provide signals for visualization purpose.

### Simulation

Start the simulation. Observe on the scope the motor currents, voltage, speed, and torque during the starting. Double click on the two Manual Switch blocks to switch from the constant wref=1500 rpm and TL blocks to the Step blocks. (Reference speed wref changed from 1500 to 750 rpm at t = 2 s and load torque changed from 0 to 1 N.m at t= 1s). Restart the simulation and observe the drive response to successive changes in speed reference and load torque.

# Permanent Magnet Synchronous Machine

This example shows the Permanent Magnet Synchronous Machine in a closed-loop speed and current control on a 1.1 kW, 3000 rpm industrial motor.




### Description

A three-phase motor rated 1.1 kW, 220 V, 3000 rpm is fed by a PWM inverter. The PWM inverter is built entirely with standard Simulink® blocks. Its output goes through Controlled Voltage Source blocks before being applied to the PMSM block's stator windings. The load torque applied to the machine's shaft is originally set to its nominal value (3 N.m) and steps down to 1 N.m at t = 0.04 s.

Two control loops are used. The inner loop regulates the motor's stator currents. The outer loop controls the motor's speed.

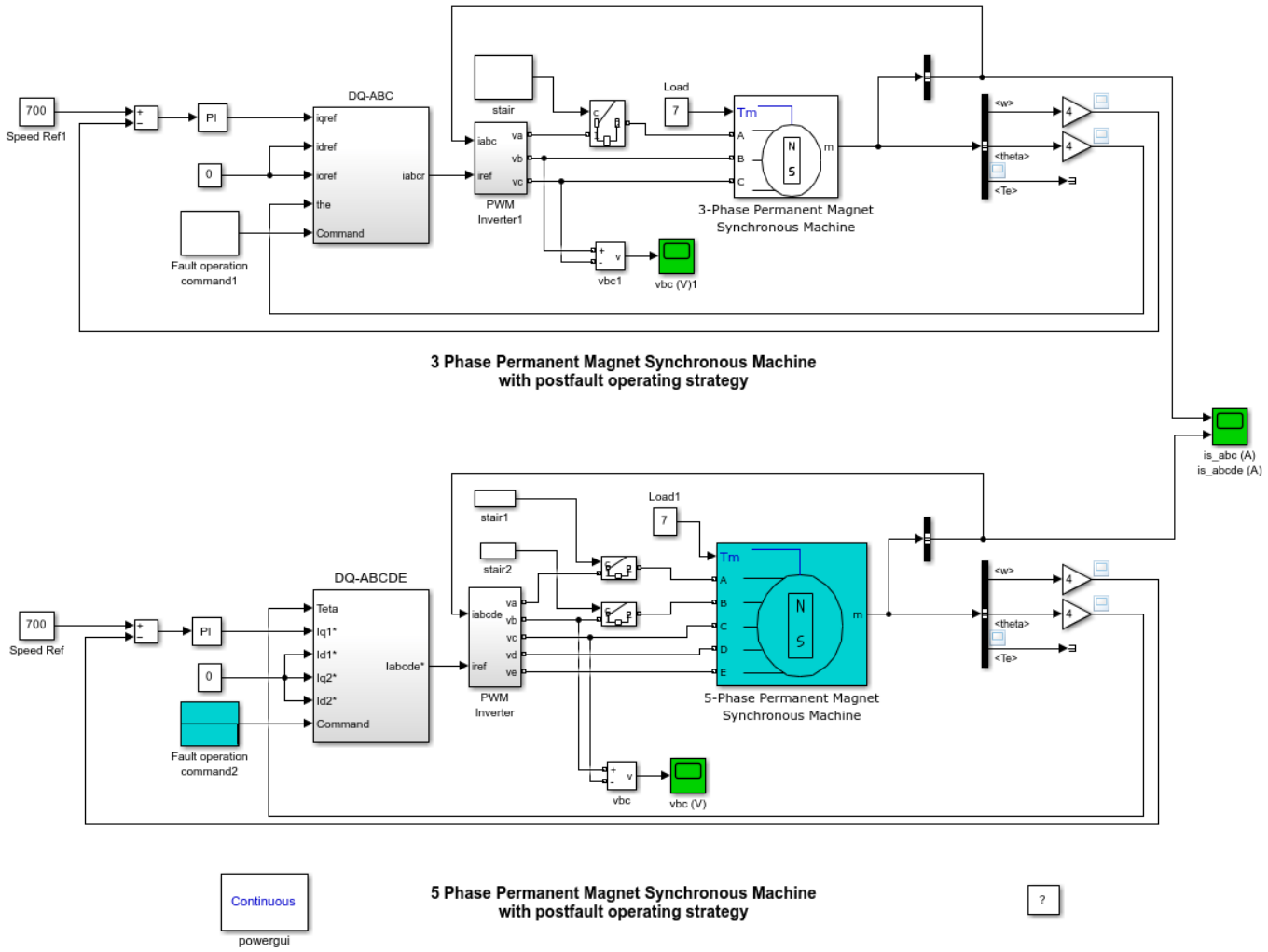
### Simulation

Observe that the stator currents are quite "noisy," which is to be expected when using PWM inverters. Also, the amplitude of these currents decreases at t = 0.04 s, when the load is decreased. The noise introduced by the PWM inverter is also observed in the electromagnetic torque waveform Te. However, the motor's inertia prevents this noise from appearing in the motor's speed waveform.

# Five-Phase Permanent Magnet Synchronous Machine

This example shows the use of the Five-Phase PMSM and the Three-Phase PMSM in a closed-loop speed and current control on two 4.4 kW industrial motors..

J-F. Doyon and Louis-A. Dessaint (Ecole de Technologie Superieure, Montreal)




## Description

A five-phase motor and a three-phase motor both rated 4.4 kW are each fed by a PWM inverter. The PWM inverter outputs goes through Controlled Voltage Source blocks before being applied to the PMSM block's stator windings. The load torque applied to each machine's shaft is fixed to 7 N.m. Two control loops are used. The inner loop regulates the motor's stator currents. The outer loop controls the motor's speed.

1. Each machine starts under normal operation (all healthy phases).

2. At t = 0.06 second, phase ''A'' of both machines is disconnected.

3. At t = 0.09 second, each controller's current references are varied to compensate for the phase loss.

4. At t = 0.12 second, phase ''B'' is also disconnected from the five-phase machine.

5. At t = 0.15 second, the controller's current references of the five-phase machine are varied to compensate for the two phases loss.

**Simulation**

Observe that due to the change of current references during the two successive faults, the five-phase machine currents are increased to keep the electromagnetique torque at the same value and to prevent large ripple. Also, good speed regulation is maintained.

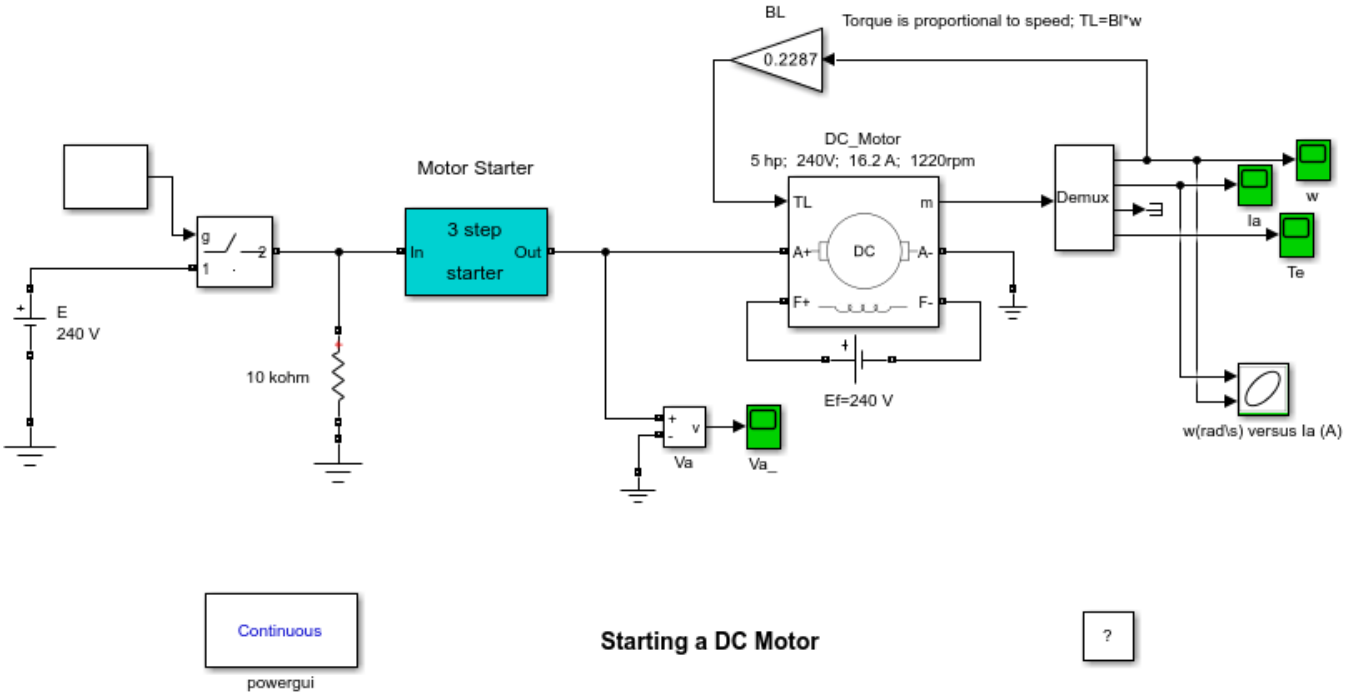
Less good results are observed with the three-phase machine. It can be seen that the electromagnetic torque has a lot of ripple which propagates to the regulated speed.

In summary the five-phase machine can be much more fault-tolerant than the three-phase machine.

# Starting a DC Motor

This example shows the starting of a 5 HP 240V DC motor with a three-step resistance starter.

G. Sybille (Hydro-Quebec)




**Description**

Induced EMF: Eo = 240-16.2*0.6 = 230.3 V

Pe = 230.3*16.2 = 3731 W = 5.0 HP

Field current: If = 240/240 =1 A

Eo = w*Laf*If ---> w = (Eo/La*If) speed w = 230.3/1.8 = 127.7 rad/s = 1220 r/min

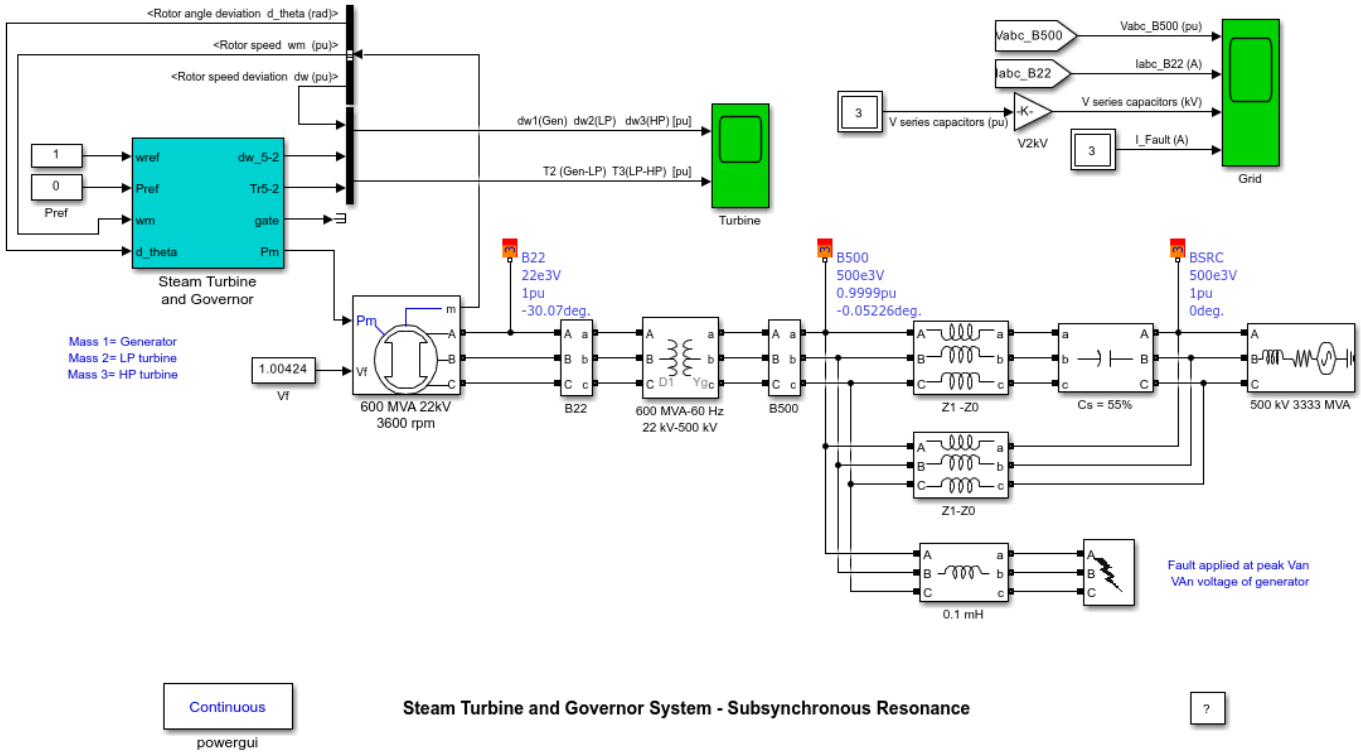
Nominal torque: Te = Pe/w = 3731/127.7 = 29.2 N.m

(Krause et al., Reference Analysis of Electric Machinery, pp. 89-92)

# Steam Turbine and Governor System - Subsynchronous Resonance

This example shows sub-synchronous resonance (SSR) in Steam Turbine and Governor on a series-compensated network.

R. Champagne and L. Dessaint (Ecole de Technologie Superieure, Montreal)




## Description

This system is an IEEE® benchmark [1] used to study sub-synchronous resonance and particularly torque amplification after a fault on a series-compensated power system. It consists in a single generator (600 MVA/22kV/60 Hz/3600 rpm) connected to an infinite bus via two transmission lines, one of which is 55% series-compensated. The sub-synchronous mode introduced by the compensation capacitor after a three-phase fault has been applied and cleared excites the oscillatory torsional modes of the multi-mass shaft and the torque amplification phenomenon can be observed. The mechanical system is modeled by 3-masses : mass 1 = generator; mass 2 = low pressure turbine (LP); mass 3 = high pressure turbine(HP).

## Simulation

**1.** In order to start the simulation in steady-state we initialized the synchronous machine with the 'Machine Initialization' tool. Open the Powergui and select "Machine initialization" . The machine "Bus type" should be already initialized as "PV generator", indicating that the initialization will be performed with the machine controlling its active power and terminal voltage. Specify the desired values by entering the following parameters:

*Terminal voltage U AB (Vrms) = 22000, Active power (Watts) = 0 *

Then press the "Compute and Apply" button.

The phasors of AB and BC machine voltages and currents flowing in phases A and B are updated. The SM reactive power, mechanical power and field voltage are displayed : Q = 1.48 Mvar, field voltage Ef = 1.0041 pu; If you open the SM block menu you will observe that the initial conditions have been updated.

**2.** In order to start the simulation in steady state with the STG connected, this Simulink® block must also be initialized. This initialization is automatically performed as long as you connect at the Pm and Vf inputs of the machine either Constant blocks or regulation blocks from the machine library (HTG, STG, or Excitation System). Open the STG block menu and note that the initial mechanical power and generator rotor angle have been automatically set to Pm0=2.7e-8 pu (corresponding to very low resistive losses in stator windings at zero output active power) and th0 = -120.1 degrees. The Vf constant block connected at the Vf excitation input of the synchronous machine has been automatically set to 1.0046 pu .

**3.** Open the Scope connected to the STG model. Run the simulation. Observe the speed deviations on the top axis of the Scope and the torques transmitted between the shaft's masses on the bottom axis. A peak torque of over 4 pu is observed for T2 between mass 1 (Generator) and mass 2 (LP turbine) . The frequency of the oscillations observed in the torques and speed deviations is about 27 Hz. The peak values we obtained here agree with those given in [1].

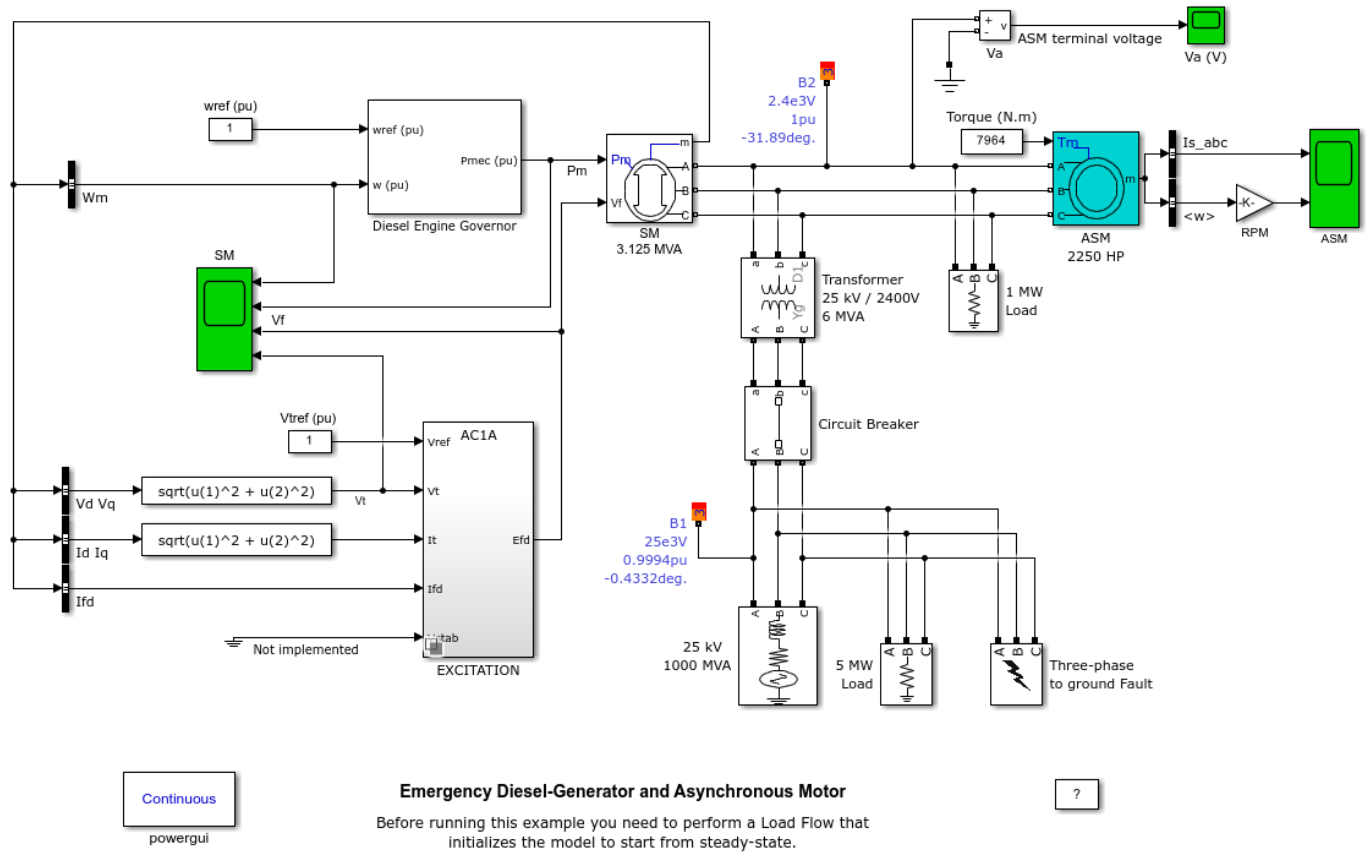
### Reference

[1] IEEE Sub-synchronous resonance working group, "Second benchmark model for computer simulation of sub-synchronous resonance," IEEE Transactions on Power Apparatus and Systems, vol. PAS-104, no. 5, 1985, pp. 1057-1066.

# Emergency Diesel-Generator and Asynchronous Motor

This example shows the Machine Load Flow tool of Powergui block to initialize an induction motor/diesel-generator system.

G. Sybille (Hydro-Quebec), Tarik Zabaiou (ETS)




## Circuit Description

A plant consisting of a resistive and motor load is fed at 2400 V from a distribution 25 kV network through a 6 MVA 25/2 kV Wye-Delta transformer and from an emergency synchronous generator/diesel engine unit. The 25 kV network is modeled by a R-L equivalent source with a short-circuit level of 1000 MVA and with a 5 MW load. A three-phase to ground fault occurs on the 25 kV system, causing opening of the 25kV circuit breaker.

## Demonstration

**1.** To start the simulation in steady-state, the synchronous machine and the asynchronous motor need to be initialized by the Load Flow tool of powergui. The Load flow parameters of the machine and motor are defined in the Load Flow tab of the two blocks:

For the synchronous machine: The "Generator type" parameter is set to "PV", indicating that the load flow is performed with the machine controlling its active power and terminal voltage. The "Active power generation P" parameter is set to 0.

For the asynchronous motor: The "Mechanical Power" parameter is set to 1.492e+006 W (2000 HP).

**2.** In the Powergui menu, select 'Load Flow'. A new window appears. A summary of the load flow settings is displayed in a table.

**3.** Press the 'Compute' button to solve the load flow. The table now display the actual machines active and reactive powers.

**4.** Press the 'Apply' button to apply the load flow solution to the model.

**5.** Open the SM and ASM blocks and note that the initial values have been updated by the Load Flow tool. The value of the constant block connected to the torque input of the asynchronous motor has also been automatically set to 7964 N.

**6.** Open the Diesel Engine Governor block. Note that the initial value of mechanical power has been set to 0.00027 pu (844 W) by the Load Flow tool.

**7.** Open the EXCITATION block and note in the Initial values tab that the initial values of terminal voltage and field voltage are set respectively to 1.0 and 1.4273 pu.

**8.** Right-click the EXCITATION block then select the type of excitation system you want to simulate.

Note that the initial values of all models have been already presaved with the same initial values. For the ST2A model, an additional line representing the initial value of terminal current It0 is set to 0.2739 pu.

**9.** Start the simulation. In the scopes check that the simulation start in steady-state.

### Simulation Results

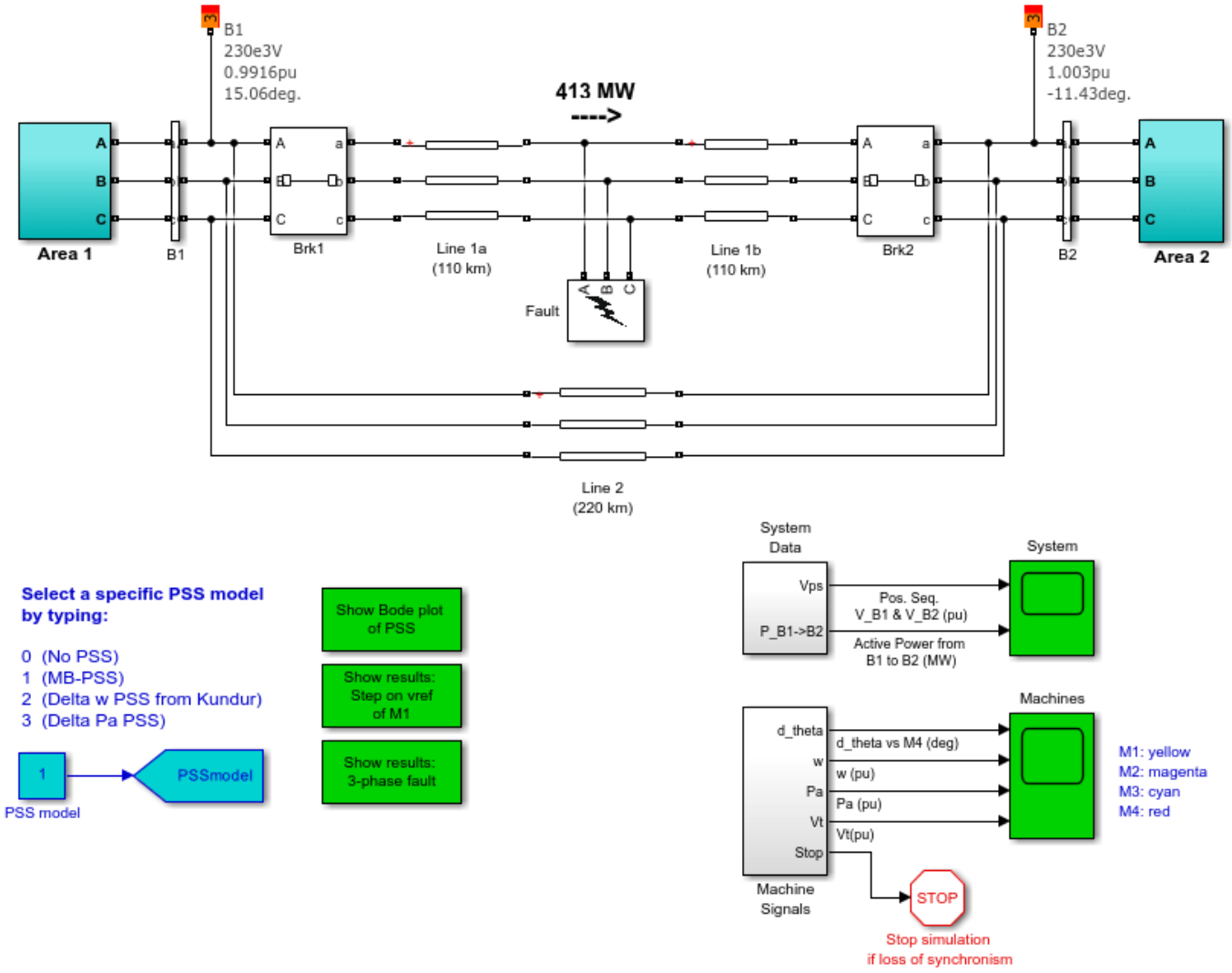
Simulation results obtained with the different excitation systems exhibit good stability when the fault is cleared. However, the ST1A and AC1A models provide better stability. The stabilization of the terminal voltage Vt is obtained in less than 2 seconds with the ST1A model and less than 3 seconds with the AC1A model. The results obtained with the AC4A and DC2A models are less efficient: the system takes longer to stabilize, the stabilization of terminal voltage Vt is obtained after 6 seconds. Note that the field voltage Vf reaches its limit without saturating in most of the models.

After fault clearing and islanding, and for all excitations models, the SM mechanical power increases from its initial value of 0 pu to the final value of 0.80 pu required by the resistive and motor load (2.49 MW). The motor speed decreases transiently from 1789 rpm to 1635 rpm, then it recovers close to its normal value.

# Performance of Three PSS for Interarea Oscillations

This example shows three Power System Stabilizers (PSS) models using Kundur's four-machine two-area test system.

I. Kamwa (Hydro-Quebec)




### Comparison of Three Power System Stabilizer (PSS) Using Kundur's Four-Machine Two-Area Test System

Three PSS are compared using the same settings for all machines:

1) MB-PSS with simplified settings: IEEE® type PSS4B according to IEEE Std 421.5

2) Conventional Delta w PSS from P. Kundur (Ref. [1], pp. 814-815, and Ref. [2] )

3) Conventional Acceleration Power (Delta Pa) PSS

### Description

The test system consists of two fully symmetrical areas linked together by two 230 kV lines of 220 km length. It was specifically designed in [1,2] to study low frequency electromechanical oscillations in large interconnected power systems. Despite its small size, it mimics very closely the behavior of typical systems in actual operation. Each area is equipped with two identical round rotor generators rated 20 kV/900 MVA. The synchronous machines have identical parameters [1,2], except for inertias which are H = 6.5s in area 1 and H = 6.175s in area 2 [1]. Thermal plants having identical speed regulators are further assumed at all locations, in addition to fast static exciters with a 200 gain [1,2]. The load is represented as constant impedances and split between the areas in such a way that area 1 is exporting 413MW to area 2. Since the surge impedance loading of a single line is about 140 MW [1], the system is somewhat stressed, even in steady-state. The reference load-flow with M2 considered the slack machine is such that all generators are producing about 700 MW each. The results can be seen by opening the Powergui and selecting Machine Initialization. They are slightly different from [1], because the load voltage profile was improved (made closer to unity) by installing 187 Mvar more capacitors in each area. In addition, transmission and generation losses may vary depending on the detail level in line and generator representation.

### Simulation

#### 1. Small-signal analysis of the systems

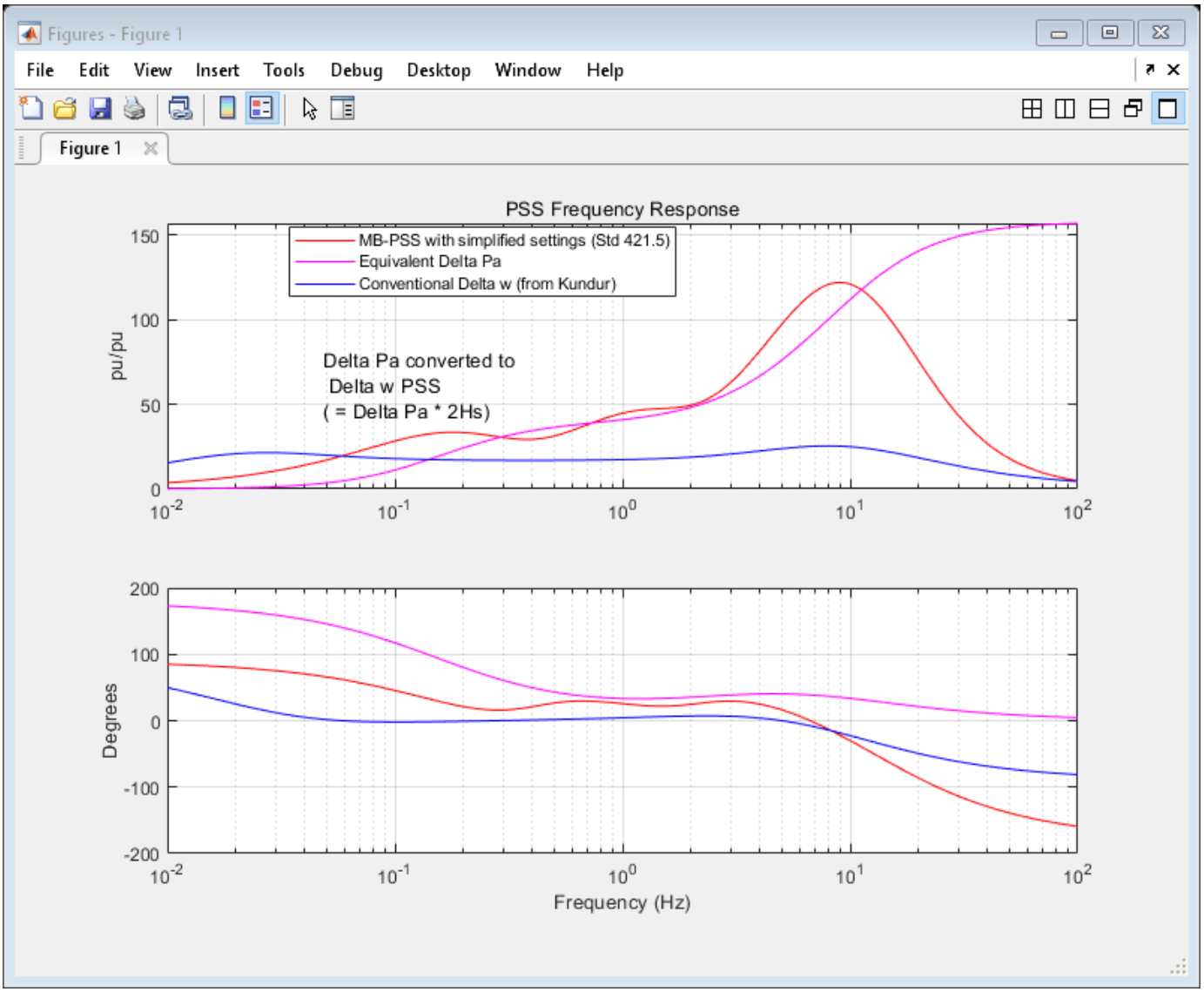
For an initial understanding of the network behavior, we can simulate its open-loop responses (PSSmodel = 0) to a 5%-magnitude pulse, applied for 12 cycles at the voltage reference of M1. This test is activated by opening the timer controlling the voltage reference of M1 and changing the multiplication factor of the transition times vector from 100 to 1. Similarly, the line fault should be deactivated by changing from 1 to 100 the multiplication factor of the transition times vector in the "Fault" device and line breakers "Brk1" and "Brk2". After starting the simulation, the signals responses are visualized by opening the "Machine" and "System" scopes on the main diagram. All signals show undamped oscillations leading to unstability. A modal analysis of acceleration powers of the four machines shows three dominant modes:

- (1) An interarea-mode (fn = 0.64Hz , z = -0.026) involving the whole area 1 against area 2: this mode is clearly observable in the tie-line power displayed in "System" scope
- (2) Local mode of area 1 (fn = 1.12Hz, z = 0.08) involving this area's machines against each other
- (3) Local mode of area 2 (fn = 1.16Hz, z = 0.08) involving machine M3 against M4 (i.e.: the smaller the inertia, the greater the local natural frequency)

If one of the two tie-lines is removed by setting the breakers "Brk1" and "Brk2" in an open position, it is possible to achieve another steady-state stable equilibrium point with the same generation and load patterns. This is called a post-contingency network, easy to initialize using the Machine Initialization tool of the Powergui. A modal analysis of this network's responses to the same 5%-magnitude pulse, applied for 12 cycles at the voltage reference of M1 reveals that, while the two local modes remain basically unchanged in both frequency and damping (fn=1.10Hz, z=0.09 in area 1 and fn=1.15Hz, z=0.08 in area 2), the interarea mode shifts to a much lower frequency with still a negative damping (i.e.: unstable): (fn = 0.44Hz, z = -0.015).

### 2. PSS Tuning

The MB-PSS settings were easily selected by varying the center frequency and gain of each band so as to achieve a nearly flat phase response between 0.1 Hz and 5 Hz. The Delta w PSS settings are from Kundur [1], with two changes: a gain increase from 20 to 30 and the addition of a 15-ms transducer time constant. To see the frequency responses of these PSSs, click on the Show Bode Plot of the PSS icon on the main diagram.

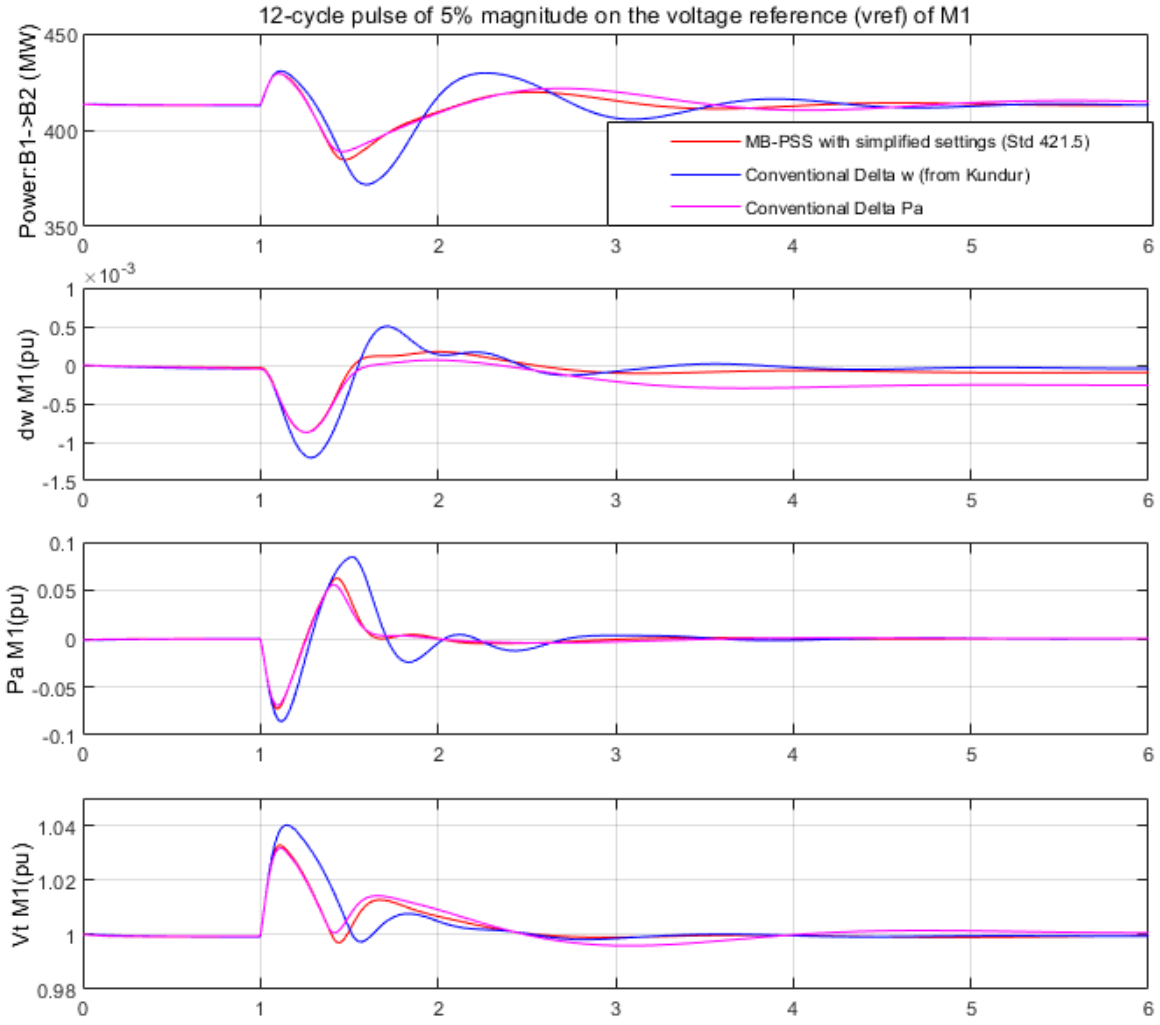



This figure confirms that the MP-BSS phase is effectively flat around 20-40 degrees in the frequency range of interest. The Delta w PSS has an overall poor phase shape, especially around 1-2 Hz, which makes it unable to cope with faster local or intermachine modes in multi-unit power plants. By contrast, the Delta Pa PSS has a good combination of strong gain and phase advance above 0.3 Hz, although it is unpractical at low frequency where it shows a 180 deg phase advance, which actually has a destabilizing effect despite the rather small low-frequency gain. Finally, even though the low-frequency shaping of the Delta w PSS is satisfactory in overall, its DC rejection (washout) is not

efficient enough, providing five time less attenuation than the MB-PSS (zoom on the magnitude plot to see this).

**3. Small-Signal Performance Assessment (12-cycle pulse on voltage reference of G1)**

To simulate the small-signal closed-loop system responses, the transition times vector of the breakers and fault devices on the main diagram should be disabled by multiplying it by 100 as above (section 1). Then the timer block controlling the voltage reference of M1 is activated in the same way by removing any multiplication by 100, i.e. by changing 100 to 1. Select the PSS to be simulated by setting PSSmodel = 1,2,3. Start the simulation and record the variables you want to compare latter. Most of the useful variables such as the machine speed and terminal voltage are stored in matrices W and Vt (look inside the "Machines" and "System" blocks on the main diagram for other variables.) This process was repeated for the three PSSs. To see the comparisons, double click on icon Show results: Step on vref of M1. The figure contains four plots: the top plot shows the power transfer from area 1 to 2; the second is the M1 speed, then the M1 acceleration power and the bottom plot is the M1 terminal voltage.

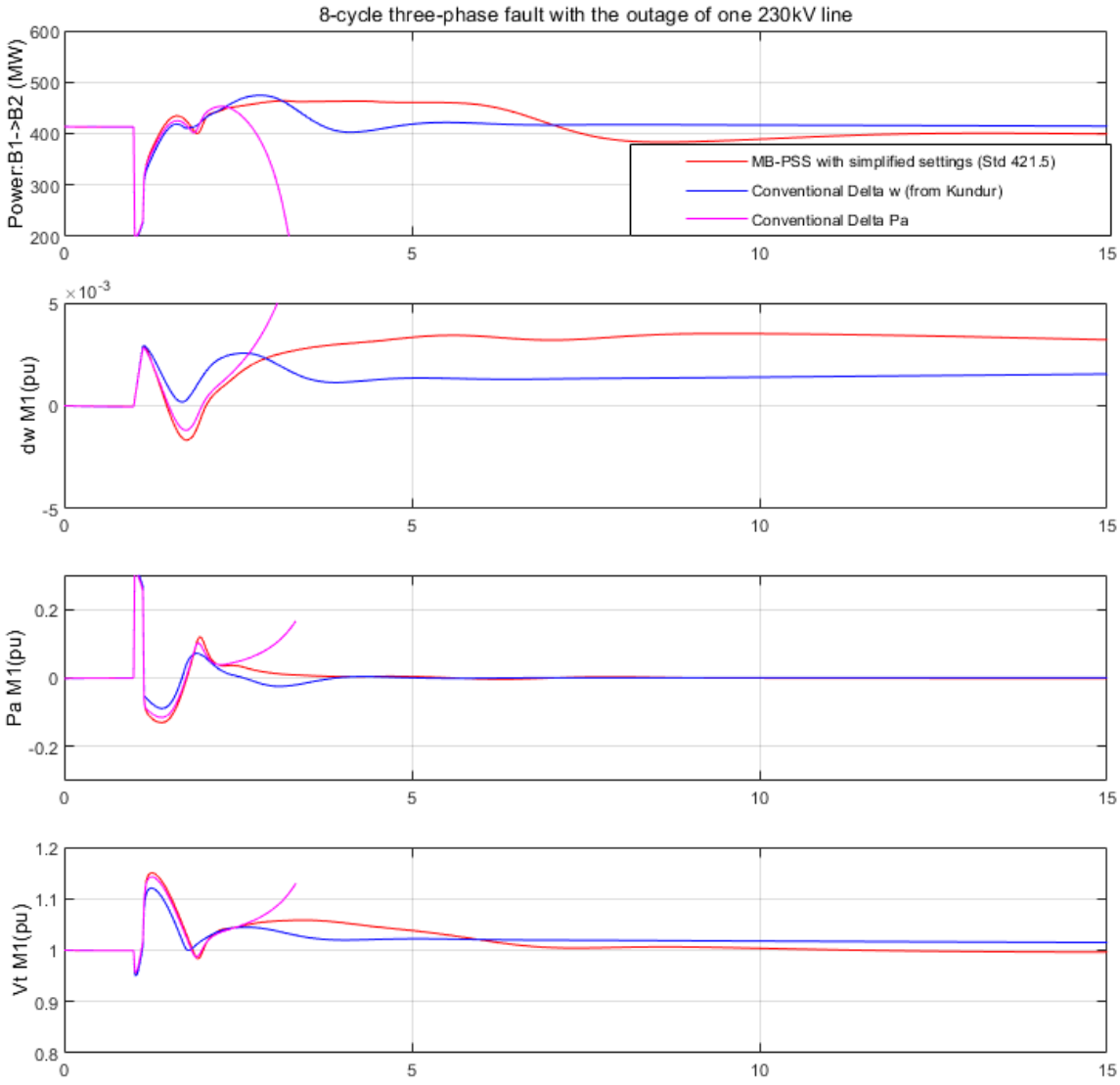



All PSSs do a good job stabilizing the naturally unstable system. However, it is clear that the MB-PSS is superior to the other two PSSs, providing significantly more damping to all modes, especially with respect to the Delta w PSS, whose poor phase/gain shaping above 0.5 Hz highlighted above shows all its limitations. Damping performance of the interarea-mode with two ties: MB-PSS: (fn = 0.50Hz, z = 0.30); Delta w: (fn = 0.64Hz, z = 0.25); Delta Pa: (fn = 0.35Hz, z = 0.30); Damping performance of the interarea-mode with one tie: MB-PSS: (fn = 0.40Hz, z = 0.36); Delta w: (fn = 0.43Hz, z = 0.35); Delta Pa: (fn = 0.23Hz, z = 0.14).

**4. Large Signal Performance and Robustness Assessment (8-cycle, three-phase fault with line outage)**

In assessing a PSS, small-signal performance is not enough. Good performance during large perturbations and good robustness with respect to changing operating conditions are other criteria of an equal importance. The system responses to a three-phase fault cleared in 8 cycles by opening the breaker "Brk1" and "Brk2" can be simulated following the same procedure as above. Without one tie-

line, the system can reach a stable operating point in steady-state, although not every PSS is able to ensure a smooth transition into this new highly stressed operating point. For comparisons, open the icon Show results: 3-phase fault.




You will see that with the Delta Pa PSS, the system lost its synchronism while the MB-PSS and the Delta W PSS succeed in maintaining stability. The latter are both very effective in damping the oscillation of the power transfer. However, the closed-loop oscillation frequency of the MB-PSS is lower while the Delta w PSS is too slow on recovering the terminal voltage: this is a bad side effect of an inefficient washout. In addition, the power acceleration is more damped with the MB-PSS than any other PSS.

### Reference

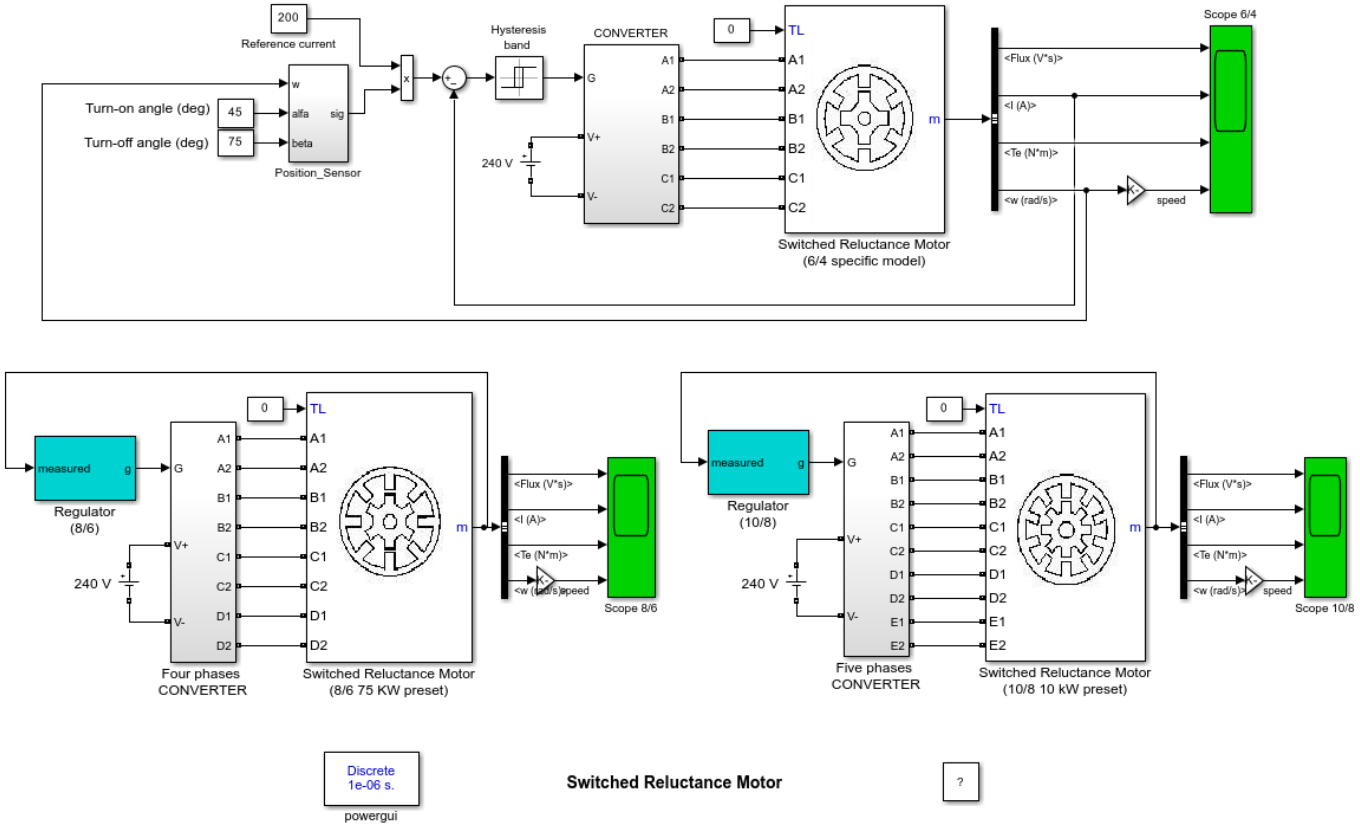
[1] P. Kundur, Power System Stability and Control, McGraw-Hill, 1994, Example 12.6, p. 813

[2] Klein, Rogers, Moorty and Kundur: "Analytical investigation of factors influencing PSS performance," IEEE Trans. on EC, Vol. 7 , No 3, September 1992, pp.382-390

# Switched Reluctance Motor

This example shows a current-controlled 60-kW 6/4 SRM drive using the SRM specific model based on measured magnetization curves. 8/6 and 10/8 preset models are also presented with same control strategy.

Hoang Le-Huy, Laval University




### Description (6/4 specific model)

The SRM is fed by a three-phase asymmetrical power converter having three legs, each of which consists of two IGBTs and two free-wheeling diodes. During conduction periods, the active IGBTs apply positive source voltage to the stator windings to drive positive currents into the phase windings. During free-wheeling periods, negative voltage is applied to the windings and the stored energy is returned to the power DC source through the diodes. The fall time of the currents in motor windings can be thus reduced. By using a position sensor attached to the rotor, the turn-on and turn-off angles of the motor phases can be accurately imposed. These switching angle can be used to control the developed torque waveforms. The phase currents are independently controlled by three hysteresis controllers which generate the IGBTs drive signals by comparing the measured currents with the references. The IGBTs switching frequency is mainly determined by the hysteresis band.

### Simulation (6/4 specific model)

In this example, a DC supply voltage of 240 V is used. The converter turn-on and turn-off angles are kept constant at 45 deg and 75 deg, respectively, over the speed range. The reference current is 200

A and the hysteresis band is chosen as +-10 A. The SRM is started by applying the step reference to the regulator input. The acceleration rate depends on the load characteristics. To shorten the starting time, a very light load was chosen. Since only the currents are controlled, the motor speed will increase according to the mechanical dynamics of the system. The SRM drive waveforms (phase voltages, magnetic flux, windings currents, motor torque, motor speed) are displayed on the scope. As can be noted, the SRM torque has a very high torque ripple component which is due to the transitions of the currents from one phase to the following one. This torque ripple is a particular characteristic of the SRM and it depends mainly on the converter s turn-on and turn-off angles. In observing the drive's waveforms, we can remark that the SRM operation speed range can be divided into two regions according to the converter operating mode: current-controlled and voltage-fed.

**Current-controlled mode**

From stand still up to about 3000 rpm, the motor's emf is low and the current can be regulated to the reference value. In this operation mode, the average value of the developed torque is approximately proportional to the current reference. In addition to the torque ripple due to phase transitions, we note also the torque ripple created by the switching of the hysteresis regulator. This operation mode is also called constant torque operation.

**Voltage-fed mode**

For speeds above 3000 rpm, the motor's emf is high and the phase currents cannot attain the reference value imposed by the current regulators. The converter operation changes naturally to voltage-fed mode in which there is no modulation of the power switches. They remain closed during their active periods and the constant DC supply voltage is continuously applied to the phase windings. This results in linear varying flux waveforms as shown on the scope. In voltage-fed mode, the SRM develops its 'natural' characteristic in which the average value of the developed torque is inversely proportional to the motor speed. Since the hysteresis regulator is inactive in this case, only torque ripple due to phase transitions is present in the torque waveforms.

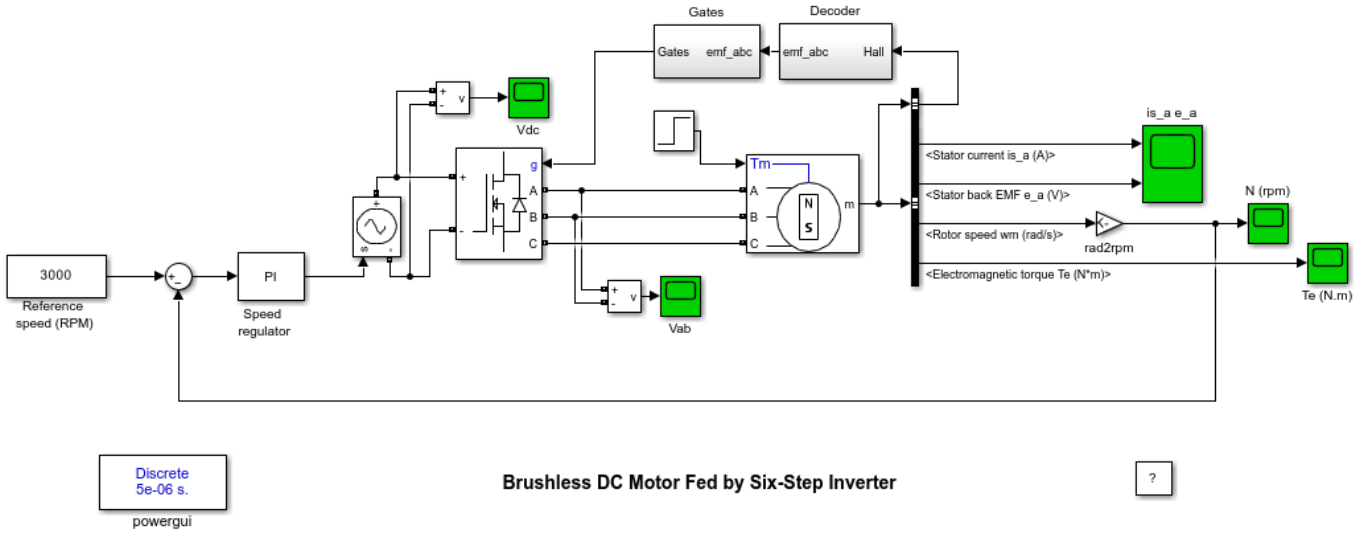
**Optimization of the Torque Characteristic - Adaptive Switching Angle**

In SRM drives, both the average torque and torque ripple are affected by the turn-on and turn-off angles and by the current waveforms in the motor phases. And these characteristics change as a function of the motor speed. In many applications, electric vehicle drives for instance, it is highly desirable to have highest torque/ampere ratio and lowest torque ripple and this over a widest speed range possible. The SRM torque characteristic can be optimized by applying appropriated pre-calculated turn-on and turn-off angles in function of the motor current and speed. The optimum values of optimum angles can be stored in a 2-D lookup table.

# Brushless DC Motor Fed by Six-Step Inverter

This example shows the use of a Six-Step Switch-on mode for a trapezoidal PMSM motor rated 1kW, 3000 rpm and speed regulated.

Olivier Tremblay, Louis-A. Dessaint (Ecole de technologie superieure, Montreal)




### Description

A three-phase motor rated 1 kW, 500 Vdc, 3000 rpm is fed by a six step voltage inverter. The inverter is a MOSFET bridge of the Specialized Power Systems library. A speed regulator is used to control the DC bus voltage. The inverter gates signals are produced by decoding the Hall effect signals of the motor. The three-phase output of the inverter are applied to the PMSM block's stator windings. The load torque applied to the machine's shaft is first set to 0 and steps to its nominal value (3 N.m) at t = 0.1 s.

Two control loops are used. The inner loop synchronises the inverter gates signals with the electromotive forces. The outer loop controls the motor's speed by varying the DC bus voltage.

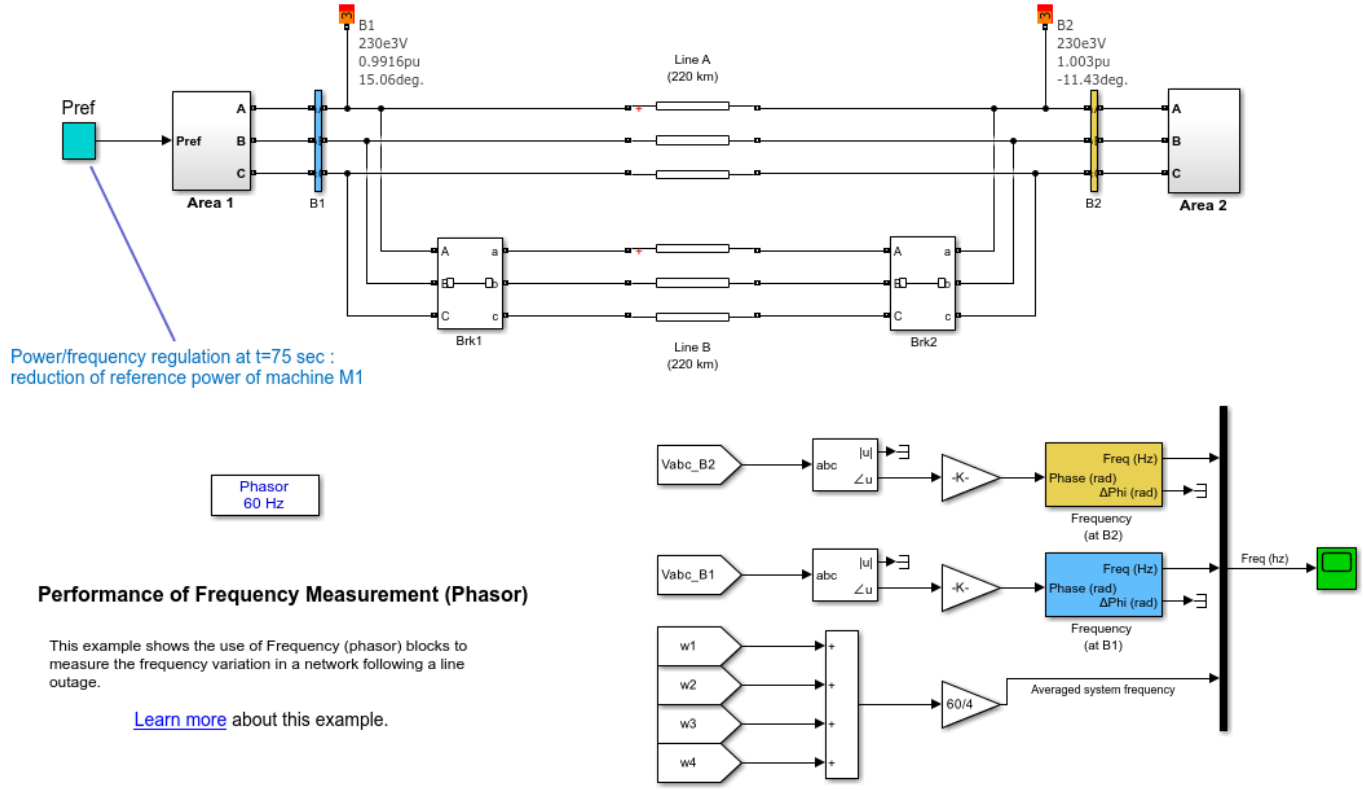
### Simulation

Observe the sawtooth shape of the motor currents. That's caused by the DC bus which applies a constant voltage during 120 electrical degrees to the motor inductances. The initial current is high and decreases during the acceleration to the nominal speed. When the nominal torque is applied, the stator current increases to maintain the nominal speed. The sawtooth waveform is also observed in the electromagnetic torque signal Te. However, the motor's inertia prevents this noise from appearing in the motor's speed waveform.

Change the "Back EMF flat area" of the motor from 120 to 0 and observe the waveform of the electromotive force e_a.

# Performance of Frequency Measurement (Phasor)

This example shows the use of two Frequency (Phasor) blocks to measure the frequency variation in a network following a line outage.



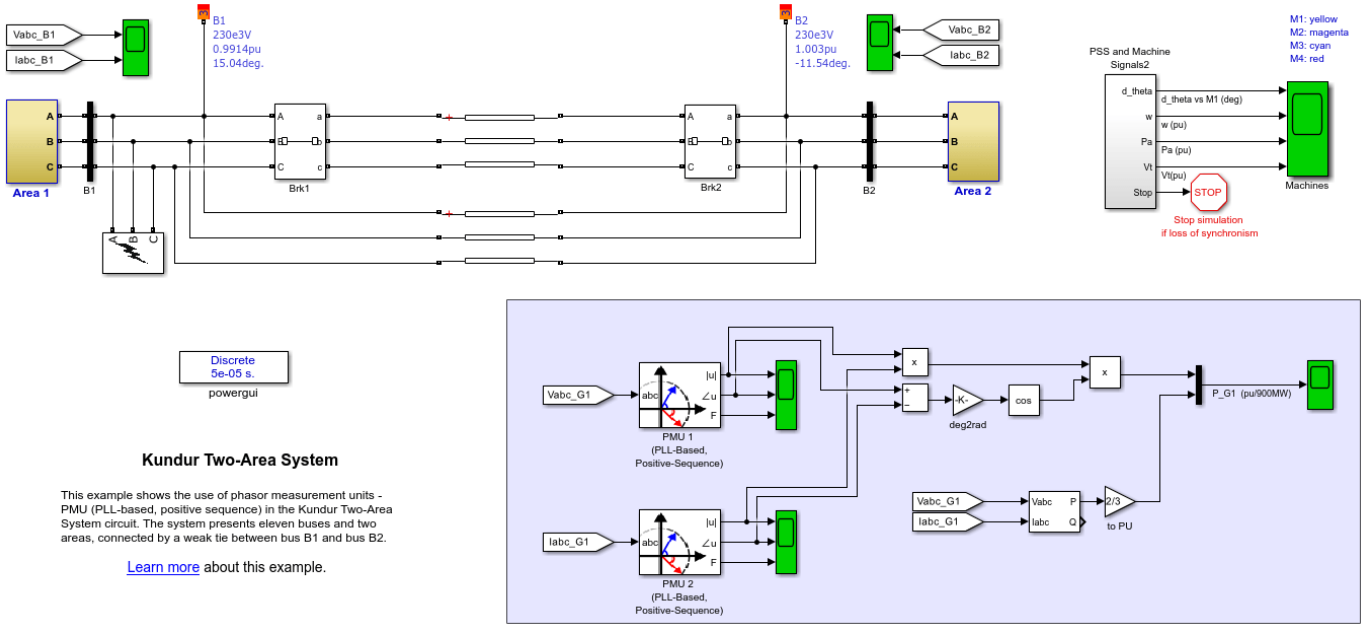

## Description

The network that is represented corresponds to the Kundur's Four-Machine Two-Area Test System, well known to study low frequency electromechanical oscillations in large interconnected power systems.

In this example, two Frequency (phasor) blocks are used to measure the frequency variation at the B1 bus (Area 1) and B2 bus (Area 2) after the transmission line B is disconnected from the network. The two measured frequencies are compared to the average system frequency. We can observe that the frequency tends to stabilize to 60.14 Hz around 75 sec, where a simple frequency regulation is performed (in open-loop) by reducing the reference power of the machine M1 in the Area 1.

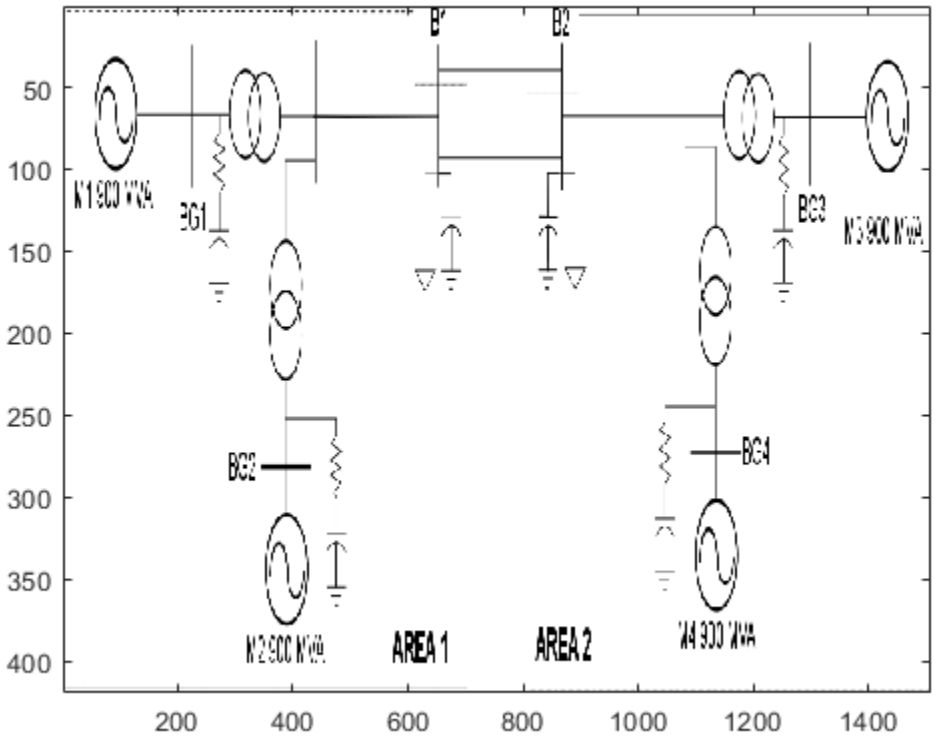
# PMU (PLL-based, Positive-Sequence) Kundur's Two-Area System

This example shows the use of the PMU (PLL-Based, Positive-Sequence) block within the Kundur's Two-Area System.




## Description

The Kundur's Two-Area System used in this example can be found on page 813 in the textbook 'Power System Stability and Control', written by P. Kundur [1]. Figure below shows basic topology.




The system presents eleven buses and two areas, connected by a weak tie between bus B1 and bus B2.

Two loads are applied to the system at buses B1 and B2. In addition, two shunt capacitors are also connected to bus B1 and bus B2 as shown in the topology above.

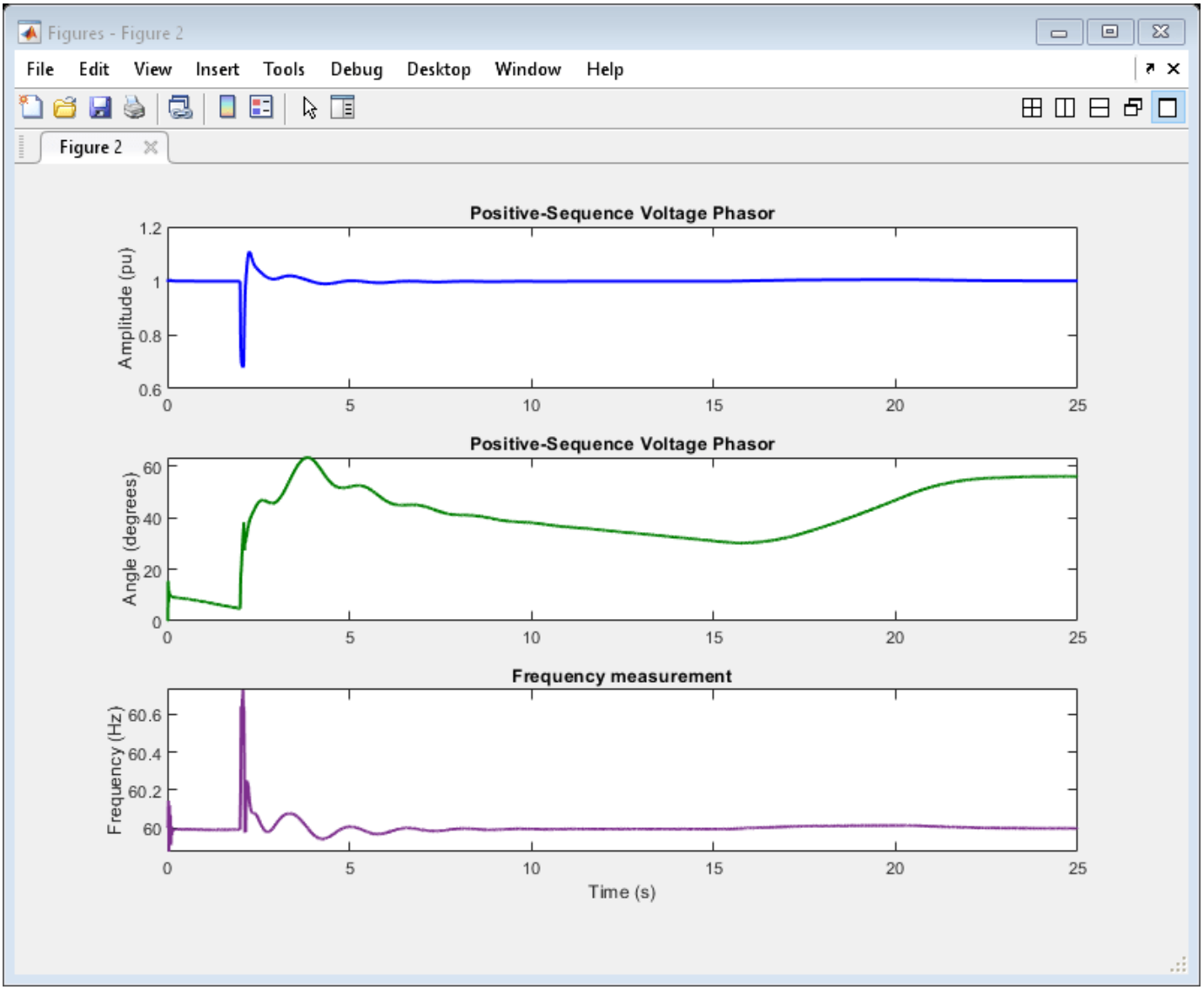
**Simulation**

In Model Configuration Parameters, select 'Fixed-step' as simulation type and 'Discrete' as solver.

PMU sample time is Ts_PMU = 1/60/64 = 20.42 us. However, the network model sample time Ts = 50 us.

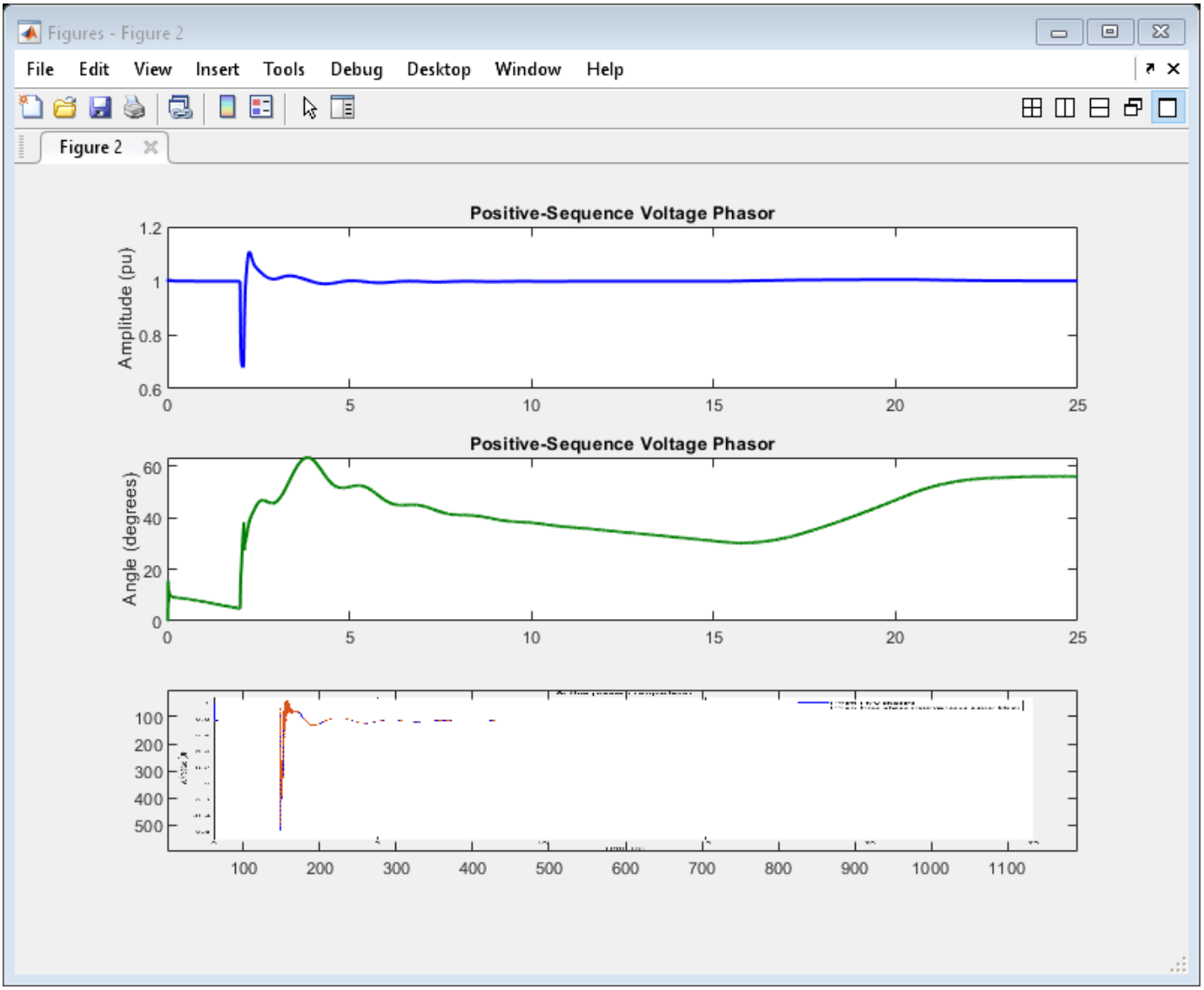
Open the scopes and start the simulation. Zoom on voltage and current phasors right after the fault occurs and notice the transient.

A three-phase fault occurs at 2 seconds of simulation. After 0.10 seconds, the three-phase fault is cleared and a new steady-state is presented until the end of the simulation at 25 seconds, as shown in the plot below.




Change the reporting rate factor and the sampling rate. Compare the shape of the waveforms shown on the scope.

Finally, compare the active power computed from the PMU phasors with the three-phase instantaneous power block. The plot below shows how the waveforms match, particularly during fault condition.

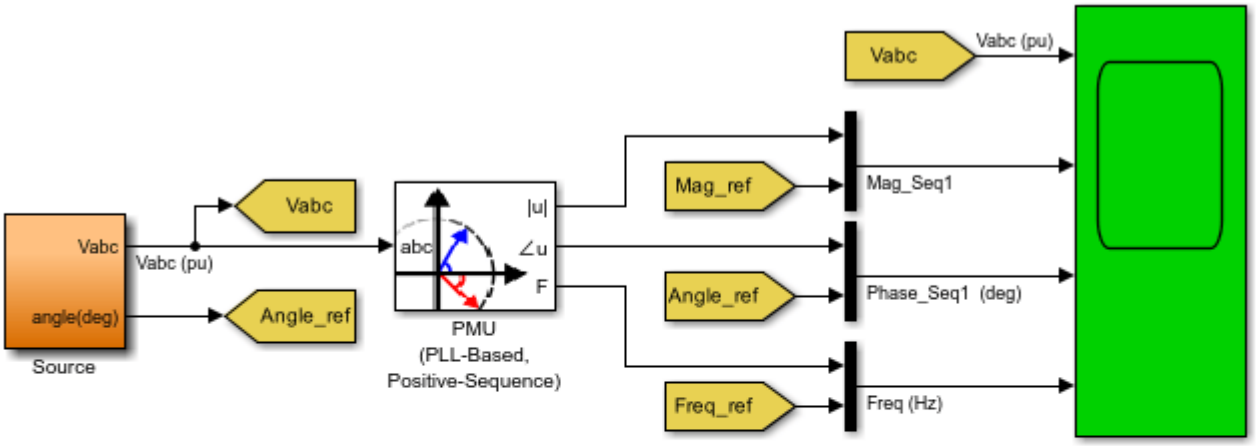



## References

[1] P. Kundur, N. J. Balu, and M. G. Lauby, Power system stability and control, vol. 7. McGraw-hi

# PMU (PLL-based, Positive-Sequence) Benchmark

This example shows the use of a phasor measurement unit - PMU (PLL-based, Positive-Sequence) in a benchmark circuit.




**PMU (PLL-Based, Positive-Sequence) Benchmark**

This example shows the PMU block that computes the positive-sequence component of a three-phase signal containing a series of events in the input signal (unbalanced condition, harmonics, frequency modulation).

Learn more about this example.

### Description

The PMU (PLL-Based, Positive-Sequence) benchmark model computes the positive-sequence component of a three-phase signal containing a series of events in the input signal (unbalanced condition, harmonics, frequency modulation).

It shows that the PMU (PLL-Based, Positive-Sequence) block outputs accurate magnitude, phase and frequency even if the input signal varies during the simulation.

### Simulation

In Model Configuration Parameters, select "Variable-step" as simulation type and "Auto (Automatic solver selection)" as the solver.

Sequence of simulated events:

1. t = 0: Positive-sequence voltage = 1 pu, phase 0 deg

2. 0.2 s < t < 0.4 s: Positive-sequence voltage falls to 0 pu

3. t > 0.5 s: Sinusoidal frequency modulation (58/62 Hz Freq = 1 Hz)

4. 1 s < t < 1.5 s: Adding unbalanced condition (0.4 pu of negative-sequence and 0.3 pu of zero-sequence, no harmonics)

5. t = 1.75 s: Phase shift of -90 deg

6. t > 2 s: Adding harmonic condition without unbalance (20% 3rd + 30% 4th)

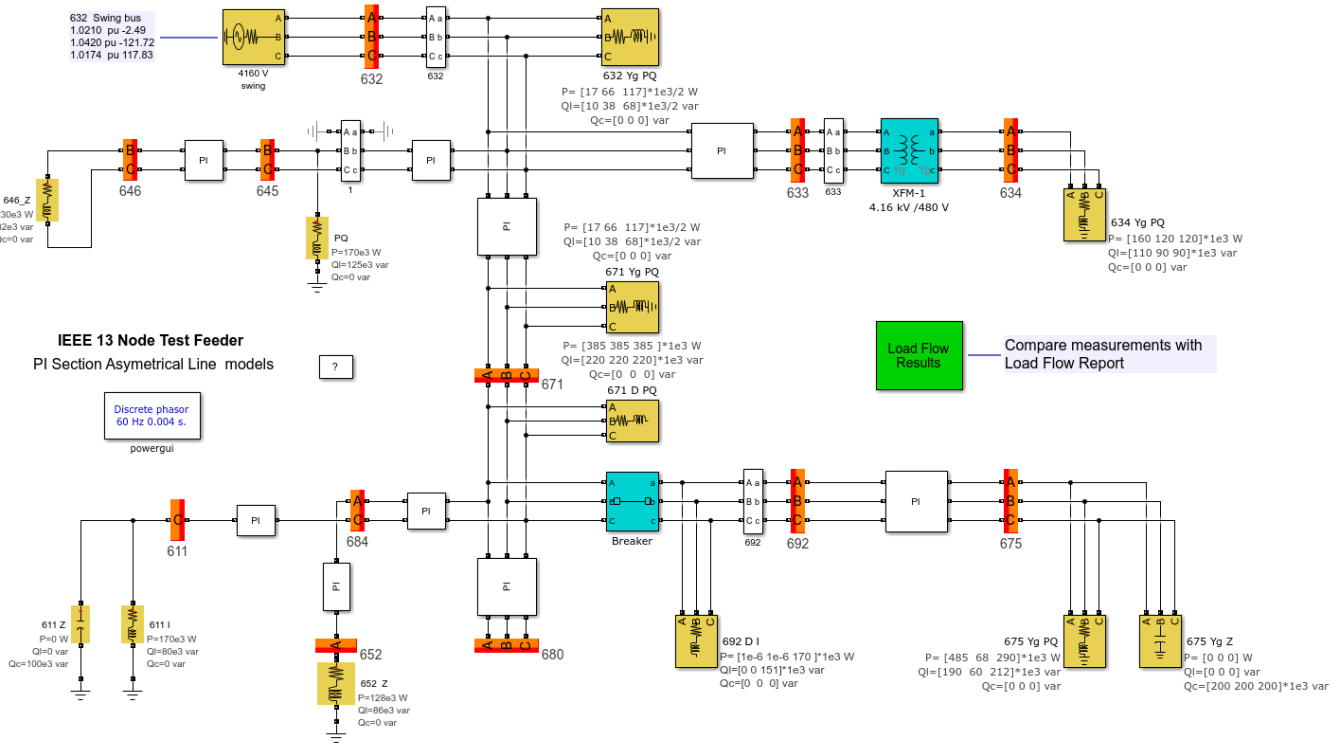
Change the reporting rate factor and the sampling rate. Compare the shape of the waveforms shown on the scope.

Finally, compare the reference signals (magenta) and the PMU signals (yellow) of frequency, magnitude and phase.

# IEEE 13 Node Test Feeder

This example shows the use of the Load Flow tool of Powergui to initialize the IEEE 13 Node Test Feeder circuit.

G. Sybille (Hydro-Quebec)




## Description

Twelve Load Flow Bus blocks are used to compute an unbalanced load flow on a model representing the IEEE 13 Node Test Feeder circuit, originally published by the IEEE Distribution System Analysis Subcommittee Report. Note that the model does not include the regulating transformer between nodes 650 and 632 of the reference test model.

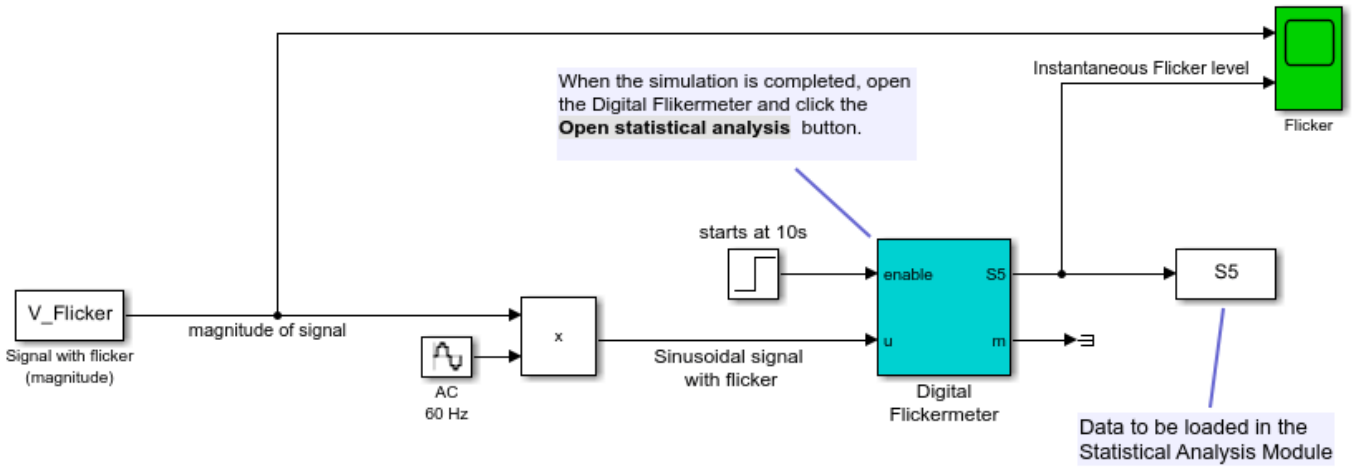
## Simulation

Open the Load Flow Tool of Powergui and press the Compute button. Press the 'Apply' button to apply the load flow solution to the model in order to start the simulation in steady-state. Note that you can view the individual bus voltage magnitude and phase angle values in the corresponding 'Load flow Bus' block's 'Load Flow' tab or you can specify them as block annotations for convenience.

Press the 'Report' button to get a report that shows a load flow summary and detailed load flow results at each bus. Partial load flow report results are reproduced in the Power-Flow Result subsystem in the model (The Green block). Start simulation and check that it starts in steady state, with expected power flow.

# Flickermeter Statistical Analysis Module

This example shows a method to compute short-term flicker severity of a phase-to-ground voltage taken from one feeder of a given remote community.



### Description

The statistical analysis of instantaneous flicker level is performed on a phase-to-ground voltage taken from one feeder of a given network. The analyzed signal comes from a phasor simulation of 650 sec of duration of a detail model of a remote community (not discussed here).

We can observe in the analyzed signal (Signal with flicker) that a large and relatively slow voltage fluctuation is superimposed on small voltage fluctuations induced by distributed thermal storage systems. It is apparent that the slow moving voltage fluctuation did not induce significant flicker compared to the small voltage fluctuations.

The magnitude of the phasor signal is used to regenerate a 60 Hz sinewave voltage that is connected at the input of the Digital Flickermeter block. Doing so, the block can compute the instantaneous flicker level present in the voltage. The signal is saved in the S5 workspace variable. The statistical analysis module of the flickermeter is used to analyze the S5 signal.

### Simulation

1. Run the simulation. When completed, click the Open statistical analysis button of the flickermeter block.

2. In the Statistical Analysis of Instantaneous Flicker Level window, load the S5 signal from workspace. The instantaneous flicker level signal is then displayed in the upper diagram.

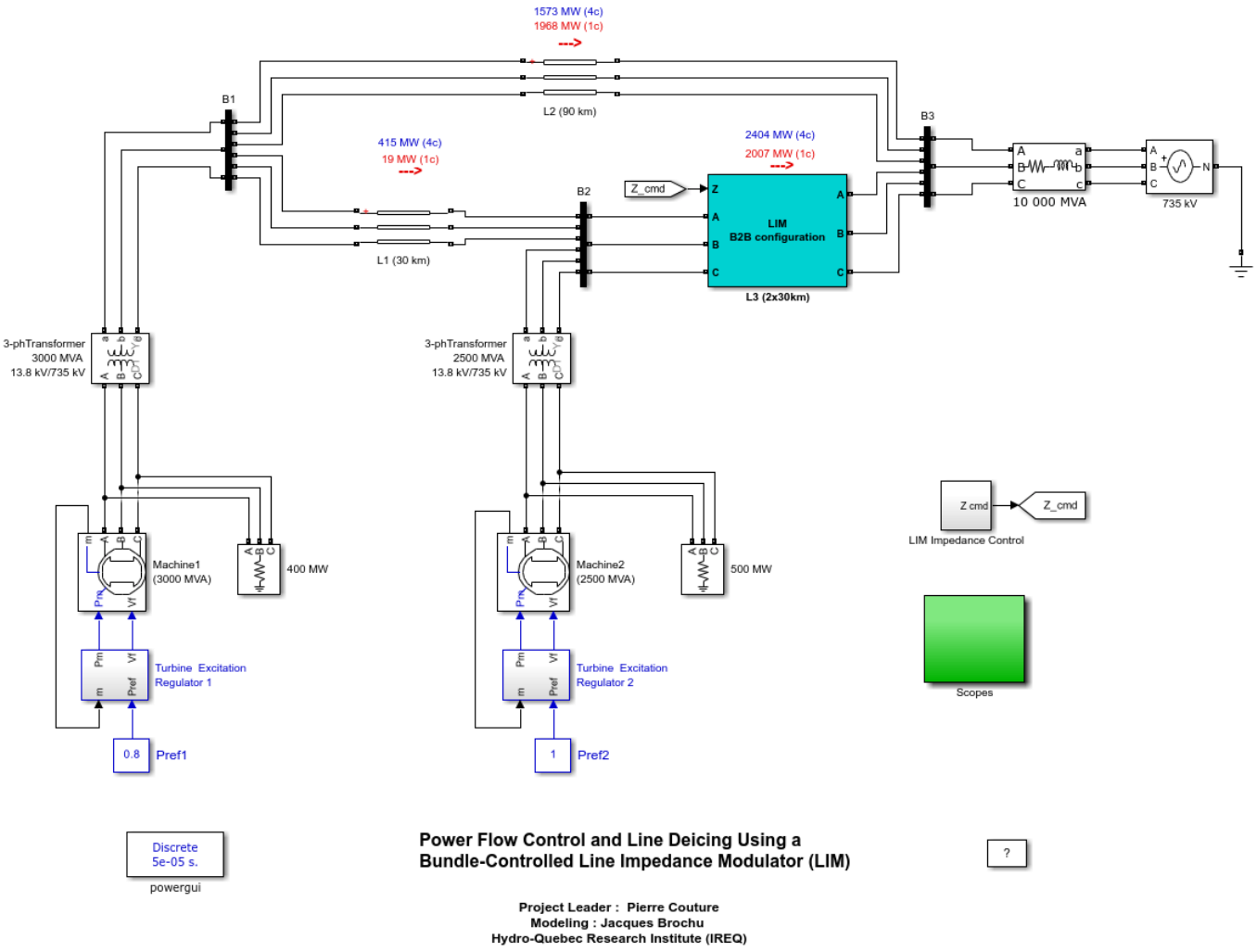
3. In the analysis section, set the start time value to 50 seconds to exclude from the analysis the initial spike in amplitude in the instantaneous flicker level.

4. Click the Compute button.

The cumulative probability function (CFP) is then displayed in the lower diagram. The short-term flicker severity value Pst is displayed along with the P0s, P1s, P3s, P10s, and P50s percentiles corresponding to the flicker levels exceeded for 0.1%, 1%, 3%, 10% and 50% of the time during the observed period.

# Power Flow Control and Line Deicing Using a Bundle-Controlled Line Impedance Modulator (LIM)

This example shows how power flow control and deicing of a transmission line could be implemented using the Line Impedance Modulator (LIM) technology.




## Technology Description

The Bundle-Controlled Line Impedance Modulator (LIM) is a distributed FACTS device that has the capability of increasing the impedance of high-voltage transmission lines. Using switches connected in series with each sub-conductors of a bundle, the LIM also allows concentrating each phase current into one sub-conductor at a time. It is then possible to de-ice each sub-conductor by the Joule effect, one after the other. This example shows how power flow control and line deicing could be implemented with a LIM. More information is available in the papers in reference.

### Demonstration

This example shows two generators and a 10,000 MVA equivalent power system interconnected by three 735-kV transmission lines. Lines L1 and L2 are conventional 30- and 90-km long transmission lines. L3 is a 60-km long line with two switching modules installed at its mid-point. Line L3 with the two line segments forms a back-to-back LIM. Power outputs of generators 1 and 2 are respectively set at 2400 and 2500 MW. Given their respective loads, connected at the 13.8 kV for the sake of simplicity, they each supply 2000 MW to bus B1 and B2 respectively. Line L2 being much longer than L3, its normal power flow is only 1573 MW when the LIM's switches are all closed. Power flows on L1 and L3 are 415 and 2404 MW respectively. The impedance control command transmitted to the LIM is produced by a signal generator inside the LIM impedance Control block. The Z cmd signal ramps between 0.5 and 3 s from its minimum value 1.0 (when all four subconductors are used) to its maximum value 1.642 pu (when only one subconductor per bundle is used) . It then varies in steps after t= 4 s. As shown inside the LIM subsystem, a look-up table associates 58 combinations of 24 switch states to the requested impedance command. As explained in [6], these combinations have been selected to maintain negative- and zero-sequence currents at a level smaller or equal than observed when all switches are closed. The 58 switch combinations used in this example represent a very small subset of the 33752 switch combinations provided by a pair of BCL segments. Each line segment is represented by a 14-conductor Exact-Pi section . Line resistances, inductances and capacitances are provided in the power_LineImpedanceModulator_init.m file. The 14x14 impedance and admittance matrices are automatically loaded in the workspace (see File/Model Properties/ Callbacks/PreloadFcn).

Run this example and observe the following sequence of events in Scope 1.

- At t=0 s, all the LIM's switches are closed. Power flows in each line are annotated in blue in the example next to each transmission line.
- At t=0.5 s, the impedance signal Zcmd ramps from 1 to 1.642 pu as shown by the yellow trace. For each values of Zcmd, the look-up table provides the corresponding switch combinations. The switch combinations transmitted to the switching modules are sampled here every 0.1 s. This gives the discretized impedance signal Zdisc (magenta).
- At t=3 s, the LIM impedance is maximum. Note that power flows in L2 (magenta) and L3 (blue) are almost equal. Power flows at this point are annotated in red in the example.
- At t=4 s, the LIM impedance is set to 1 pu which closes all switches. As a result, power flows in the lines vary abruptly within 1 cycle. This perturbation forces the synchronous generators governors to react and stabilize the power flows back to the initial values prevailing at t=0 s.
- At t=5 s, the LIM impedance signal returns to 1.642 pu which again induces a power flow perturbation.
- At 6.3 s, the LIM impedance is reduced down to 1 pu in three large steps.

### Power Flow Control

As shown in the example, when the switches of the LIM are operated, impedance of line L3 progressively increases up to the point where only one switch remains closed per switching module. With only one conductor in service per bundle power flows in line L2 and L3 are nearly equals even though L2 is 50% longer than L3. Power flows in line L1 becomes almost zero. This shows that LIMs have the capability of reducing power flows of overloaded transmission lines. The step changes beyond t=4 s shows that LIMs can also quickly vary the line impedance if required. Scope 2 shows the bus B2 sequence voltages and the sequence currents flowing out of B2 toward line L3. This is the line current of the back-to-back LIM. It can be seen that for all switch combinations used, the negative- and zero-sequence voltages and currents remain lower than the initial values obtained with all switches closed. Hence LIMs can be operated to produce power ramping or power step without

increasing negative- and zero-sequence levels. Scope 3 shows that the voltages across the switches of the phase A switching module located on the bus B2 side. It can be seen that transient voltages remain within 35 kV which allows the use of medium voltage switching devices. The maximum rms voltage in steady-state, visible at t = 3.9s, is 13.4 kV.

### Line Deicing

Scope 3 also shows the switch currents of the phase A switching module located on the bus B2 side. With all switches closed, switch currents are initially 465 A rms. With all but one switch opened at t=3.9 s, it can be seen that the switch current of subconductor 2 reaches 1533 A rms. Hence, although the power flow in line L3 has been reduced by 17%, the subconductor current has increased by a 3.3 factor. Such a current is large enough for simultaneously deicing by the Joule effect three subconductors (one per phase) in both 30-km BCL segments. Once a first subconductor is de-iced in each bundle of both BCL segments, three other switch combinations can be used to completely de-ice the transmission line. Note that the switch combination table provided in the example is appropriate for power flow control. Other switch combination tables would be used to force specific subconductor deicing sequence and avoid bundle rotation. Note that if line currents had initially been too low for reaching a deicing level, it could have been possible to open the switches of one 30-km segment only. The smaller impedance increase provided by one BCL segment would then lead to a higher current in the sub-conductors to de-ice. Also, accordingly to the concept of the smart power grid where each line would be equipped with switching modules [4-5], impedance of line L2 could be increased to divert even more current into line L3.

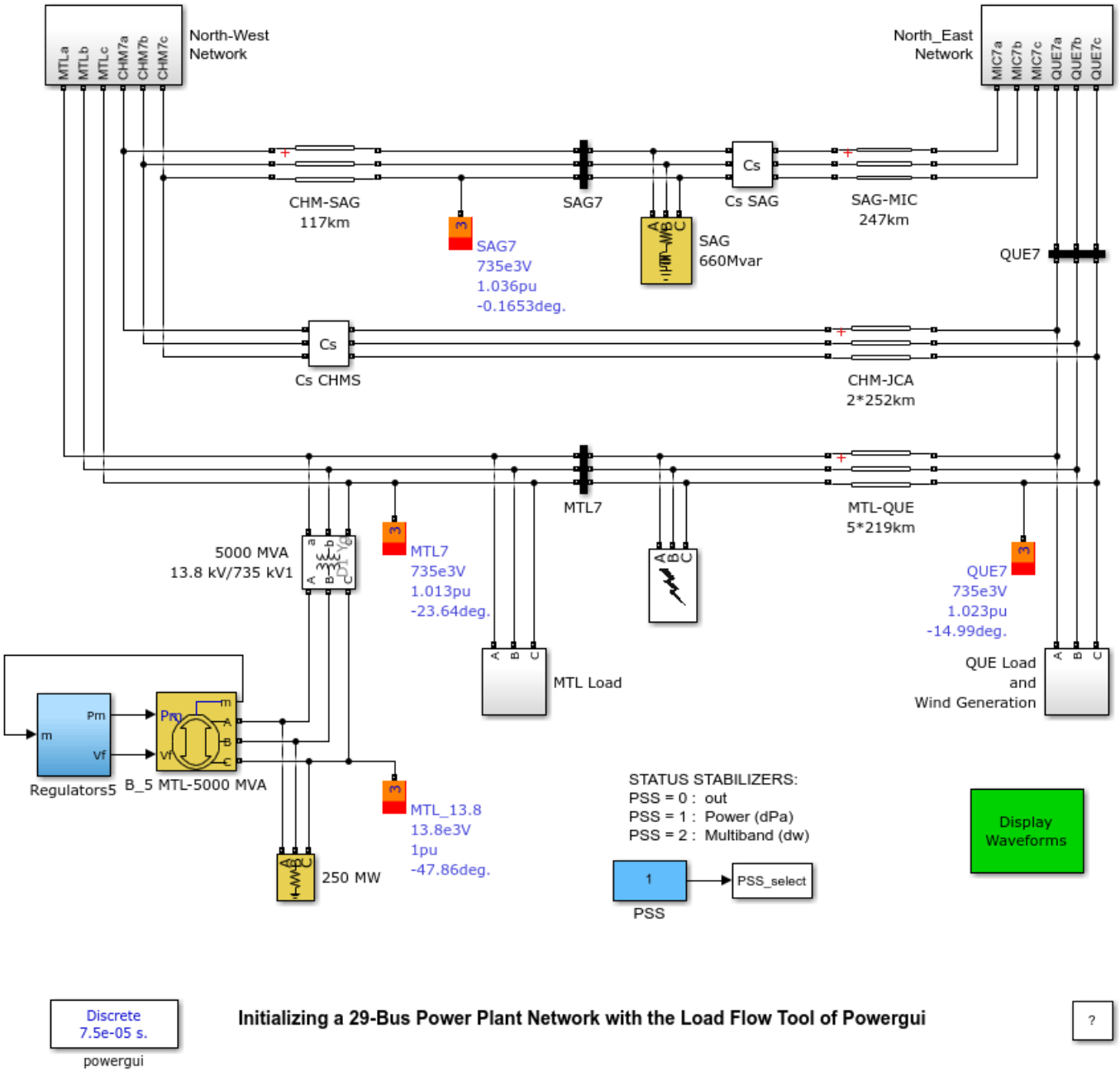
### References


1. [1] P. Couture, J. Brochu, G. Sybille, P. Giroux and A. O. Barry, "Power flow and stability control using an integrated HV bundle-controlled line-impedance modulator", IEEE Trans. Power Del., vol. 25, no. 4, pp. 2940-2949, Oct. 2010.
2. [2] P. Couture, "Smart Power Line and Photonic de-icer concepts for transmission-line capacity and reliability improvement" Cold Reg. Sci. Technol. 65 (2011), Jan., 13-22.
3. [3] P. Couture, "Switching modules for the extraction/injection of power (without ground or phase reference) from a bundled HV line," IEEE Trans. Power Delivery, vol. 19, No3, pp. 1259-1266, July 2004.
4. [4] P. Couture, "Switching apparatus, control system and method for varying an impedance of a phase line", Patent pending, PCT/CA2011/00850, July 22, 2011.
5. [5] P. Couture, J. Brochu, B. Francoeur, R. Morin, D. H. Nguyen, K. Slimani, A. Turgeon and P. Van Dyke, "Smart Power Line (SPL) experimental research project," CIGRE 2014.
6. [6] J. Brochu and P. Couture, "Load Flow Modeling of the Integrated Bundle-controlled Line Impedance Modulator," IEEE Trans. Power Delivery, accepted and currently available on IEEExplore.

# Initializing a 29-Bus, 7-Power Plant Network With the Load Flow Tool of Powergui

This example shows the use of the Load Flow tool of Powergui to initialize a 29-bus 735 kV network with detailed modeling of power plants using hydraulic turbines, speed regulation, excitation systems and power system stabilizers.

G. Sybille (Hydro-Quebec)



### Description

The model shows a 735-kV transmission network with detailed modeling of seven 13.8 kV power plants (total available generation =26200 MVA) including hydraulic turbines, speed regulation, excitation systems and power system stabilizers. The 735-kV transmission network is both series and shunt compensated using fixed capacitors and inductors. The load is lumped at two buses (MTL7 and QUE7).

The MTL Load subsystem connected to the MTL7 bus consists of four types of load blocks connected on the 25 kV distribution system through 735 kV /230 kV and 230 kV/ 25 kV transformers.

The QUE Load and Wind Generation subsystem uses a 6000 MW load (constant Z and constant PQ) connected on the 120 kV bus. A 9 MW wind farm using an asynchronous generator is connected to the 120 kV bus through a 25-kV feeder and a 25 kV/120 KV transformer.

The model is discretized using a 50 usec. sample time. A six-cycle three-phase fault is programmed at MTL7 bus. Look at waveforms in the scopes (inside the Display Waveforms green subsystem) and notice that simulation starts in steady state.

### Simulation

In the Powergui menu, select 'Load Flow'. A new window appears. A summary of the load flow settings is displayed in a table. Press the 'Compute' button to solve the load flow. The table now displays the actual active and reactive powers of the load flow blocks of the model.

Press the 'Apply' button to apply the load flow solution to the model in order to start the simulation in steady-state. Note that the Load flow Bus block displays the bus voltage magnitude and phase angle as block annotations.

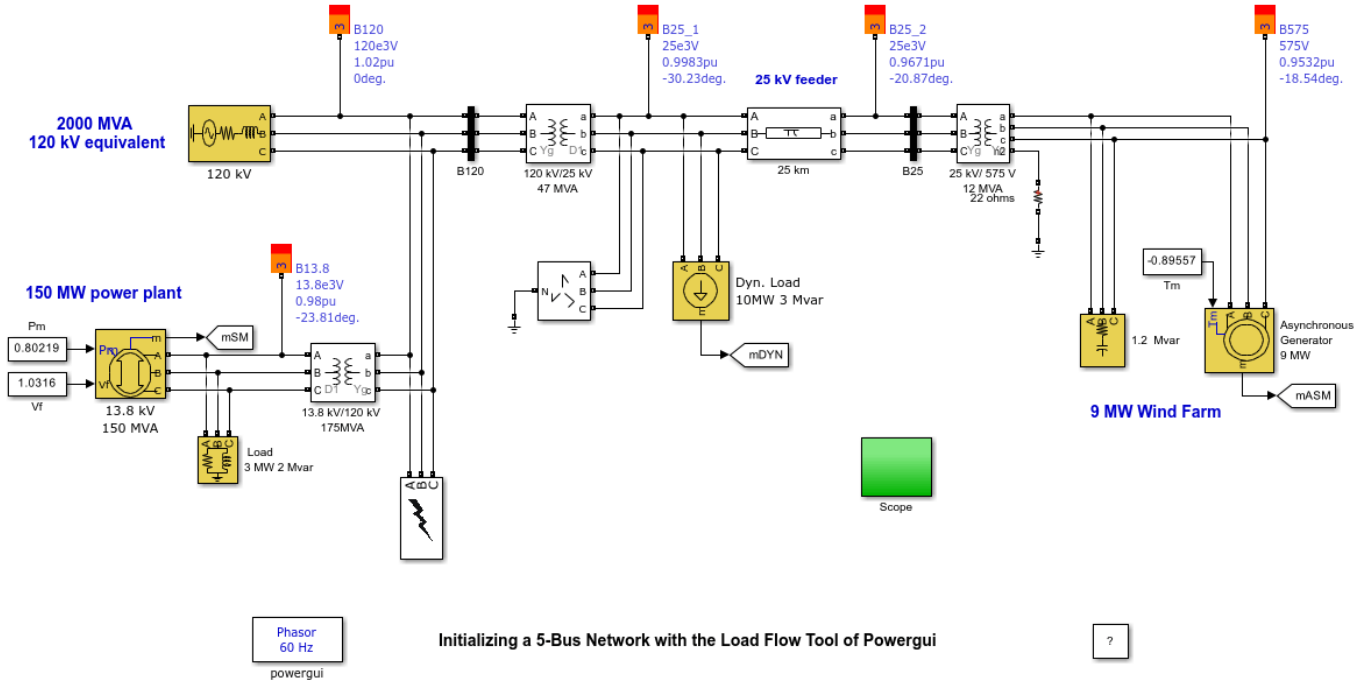
Press the 'Report' button to get a report that shows a load flow summary and detailed load flow results at each bus.

Finally, start simulation and check that it starts in steady state.

# Initializing a 5-Bus Network with the Load Flow Tool of Powergui

This example shows the use of the Load Flow tool of Powergui to initialize a simple 5-bus network consisting of a wind farm connected to a 25-kV distribution feeder and exporting power to a 120 kV network.

G. Sybille (Hydro-Quebec)




### Description

The model shows a 9 MW wind farm using asynchronous generators and exporting power to a 120 kV network through a 25-kV distribution feeder. The 120 kV network is modeled by a simple inductive voltage source (short circuit power of 1200 MVA) using the Three-Phase Source block. A 150 MW power plant using a 13.8 kV synchronous generator is connected at the 120 kV bus through a 13.8 kV/ 120 kV transformer.

### Simulation

The five Load Flow Bus blocks are used to specify the bus base voltages and to specify the voltage at PV bus and the voltage and angle of the swing bus.

The Load flow parameters are defined in the Load Flow tab of the Synchronous and Asynchronous machine blocks, Three-Phase Source block, Three-Phase Dynamic Load block, and the Three-Phase RLC Load blocks.

Press the 'Apply' button to apply the load flow solution to the model in order to start the simulation in steady-state. Note that the Load flow Bus block displays the bus voltage magnitude and phase angle as block annotations.

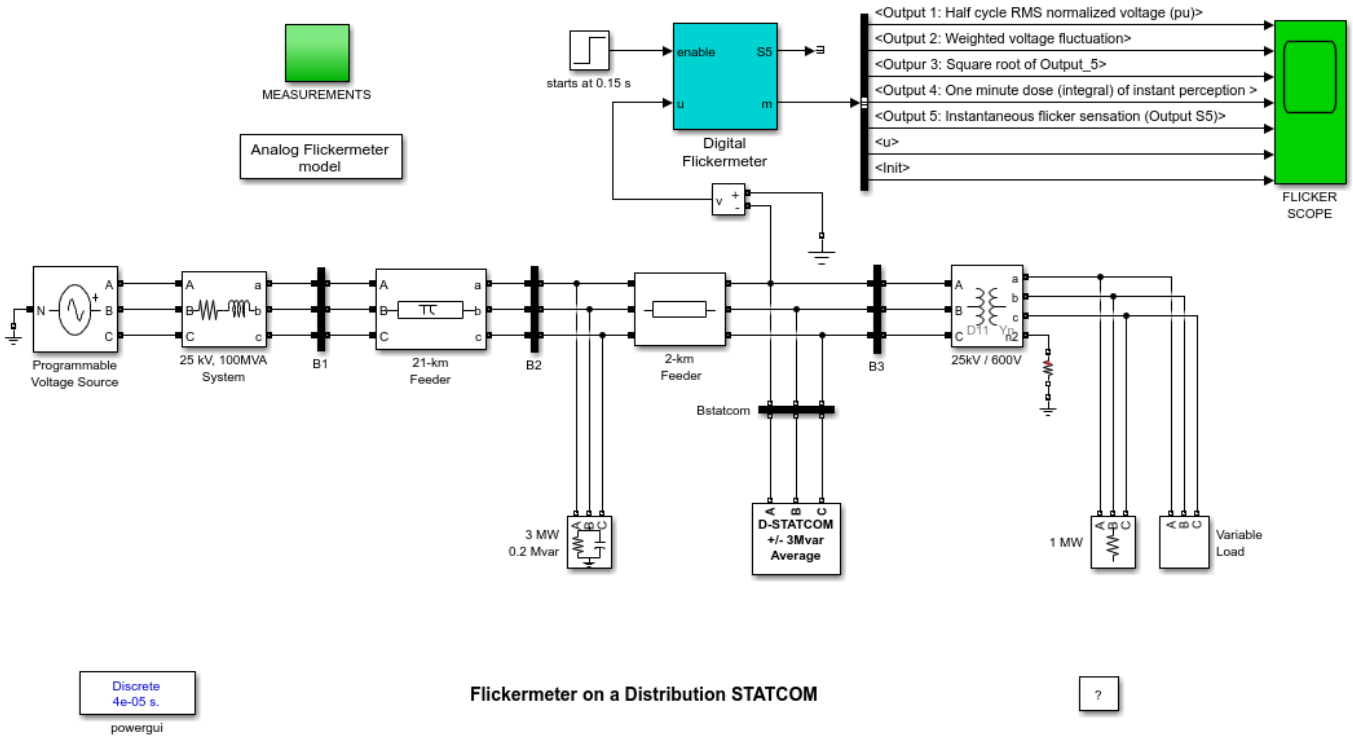
Press the 'Report' button to get a report that shows a load flow summary and detailed load flow results at each bus.

Finally, start simulation and check that it starts in steady state.

# Flickermeter on a Distribution STATCOM

This example shows a flickermeter model designed according to functional specifications of the international standard IEC 6100-4-15 [1].

Silvano Casoria (Hydro-Quebec)




## Description

A flickermeter device is used to measure the instantaneous flicker sensation on terminal voltage of a STATCOM unit.

The flickermeter model is designed according to functional specifications of the international standard IEC 6100-4-15 [1]. The flickermeter is a standardized instrument for measuring the flicker obtained by simulation and by statistical analysis of the response of the lamp-eye-brain chain to the input voltage fluctuations. Reference [2] gives a detailed description of the standard block diagram. The statistical analysis (module 5 in standard block diagram) is not modeled in the block.

In order to minimize the initial transient response, the initial conditions are defined for the different transfer functions. A fixed initialization period (Tini = 0.3 s) is required between the instant the model is enabled (input ON = 1) and the computation of the results. The initial conditions calculation are evaluated in the mask initialization section of the block.

The model should be enabled (input ON > 0) after the analyzed signal (input U) is stable. The flickermeter internal outputs of the various modules are available in output m.

In practice output S (instantaneous flicker sensation) stabilizes within 2 seconds after the initialization period is started (Init signal in output m) and Output_4 (One minute dose of instant perception signal in output m) stabilizes after 2 minutes.

Module 1 contains a signal generator for checking the flickermeter setting in the field and a circuit for normalizing (in pu) the RMS value of the input voltage at network frequency (50 or 60 Hz). The generator adds a modulating voltage (fluctuating sinusoidal or triangular signal) to the fundamental (120 Vrms/60 Hz or 230 Vrms/50 Hz). Different relative amplitude (in %) and frequency of the fluctuation are defined in the standard. They should produce at Output_5 an instantaneous flicker sensation of 1.

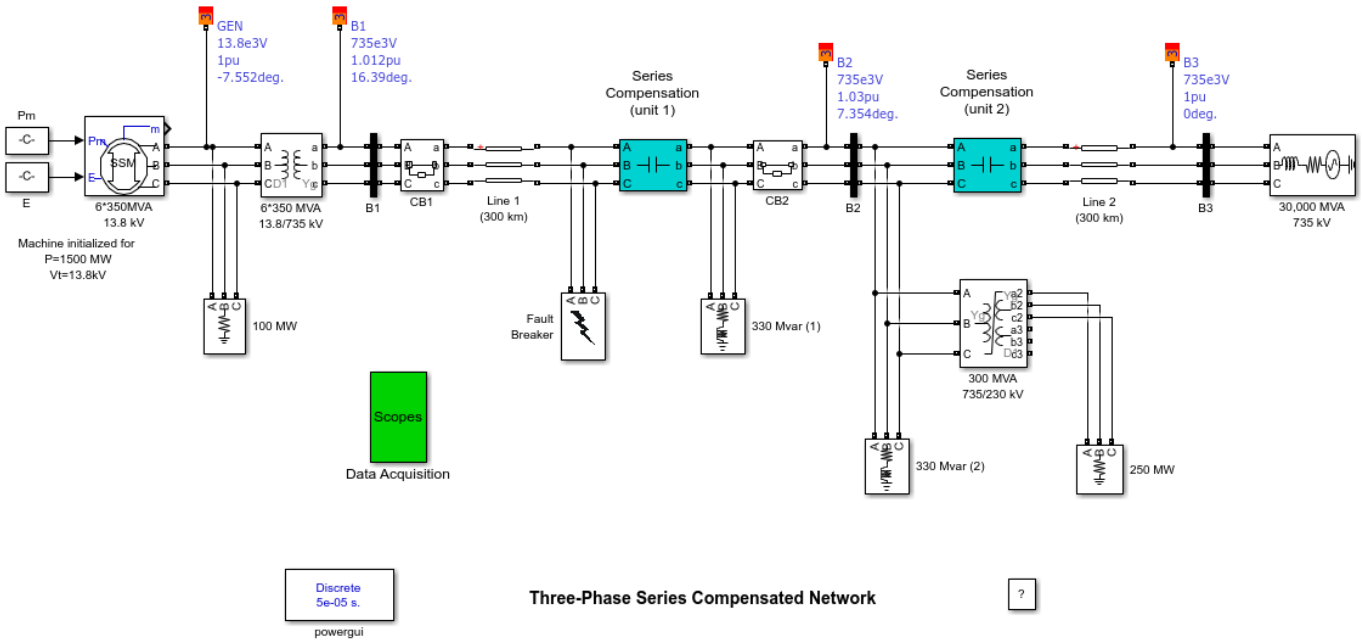
### References

[1] EC Standard 61000-4-15 Ed. 1.1 (2003) Electromagnetic compatibility (EMC)-Part4 Testing and measurement techniques-Section15 Flickermeter: Functional and design specifications [2] Bertola A.; Lazaroiu, G.C.; Roscia, M.; Zaninelli, D.;"A MATLAB®-Simulink® flickermeter model for power quality studies", IEEE® PES 11th International Conference on Harmonics and Quality of Power, September 2004, Lake Placid, USA.

# Three-Phase Series Compensated Network

This example shows the use of three-phase blocks to study transients on a series-compensated 735-kV transmission system

G. Sybille (Hydro-Quebec)




## Description

A three-phase, 60 Hz, 735 kV power system transmitting power from a power plant consisting of six 350 MVA generators to an equivalent network through a 600 km transmission line. The transmission line is split in two 300 km lines connected between buses B1,B2, and B3. In order to increase the transmission capacity, each line is series compensated by capacitors representing 40% of the line reactance. Both lines are also shunt compensated by a 330 Mvar shunt reactance. The shunt and series compensation equipment are located at the B2 substation where a 300 MVA 735/230 kV transformer with a 25 kV tertiary winding feeds a 230 kV, 250 MW load. The series compensation subsystems are identical for the two lines. For each line, each phase of the series compensation module contains the series capacitor , a metal oxide varistor (MOV) protecting the capacitor, and a parallel gap protecting the MOV. When the energy dissipated in the MOV exceeds a threshold level of 30 MJ, the gap simulated by a circuit breaker is fired. CB1 and CB2 are the two line circuit breakers .

The generators are simulated with a Simplified Synchronous Machine block. Universal transformer blocks (two-windings and three-windings) are used to model the two transformers. Saturation is implemented on the transformer connected at bus B2. Voltages and currents are measured in B1, B2, and B3 blocks. These blocks are Three-phase V-I Measurement blocks where voltage and current signals are sent to the Data Acquisition block through Goto blocks.

## Fault and Line Switching

You now study the transient performance of this circuit when a line-to-ground and three-phase-to-ground faults are applied on line 1. The fault and the two line circuit breakers CB1 and CB2 are

simulated with blocks from the three-phase library. Open the dialog boxes of CB1 and CB2. See how the initial breaker status and switching times are specified. A line-to-ground fault is applied on phase A at t = 1cycle. The two circuit breakers which are initially closed are then open at t = 5 cycles, simulating a fault detection and opening time of 4 cycles. The fault is eliminated at t = 6 cycles, one cycle after line opening.

### Simulation

Notice that this system contains the Powergui block. In addition, when you start the system the 'power_3phseriescomp' model, the sampling time Ts = 50e-6 is automatically set in your workspace. The system is therefore be discretized using a 50 microseconds sample time.

### Line-to-Ground Fault

Double click the Data Acquisition block and open the three scopes. Start the simulation. As the system has already been initialized (1500 MW generation at the 13.8 kV bus) with the Load Flow utility of the Powergui, the simulation starts in steady state. At t = 1 cycle a line-to-ground fault is applied and the fault current reaches 10 kA . During the fault, the MOV conducts at every half cycle and the energy dissipated in the MOV builds up to 13 MJ.

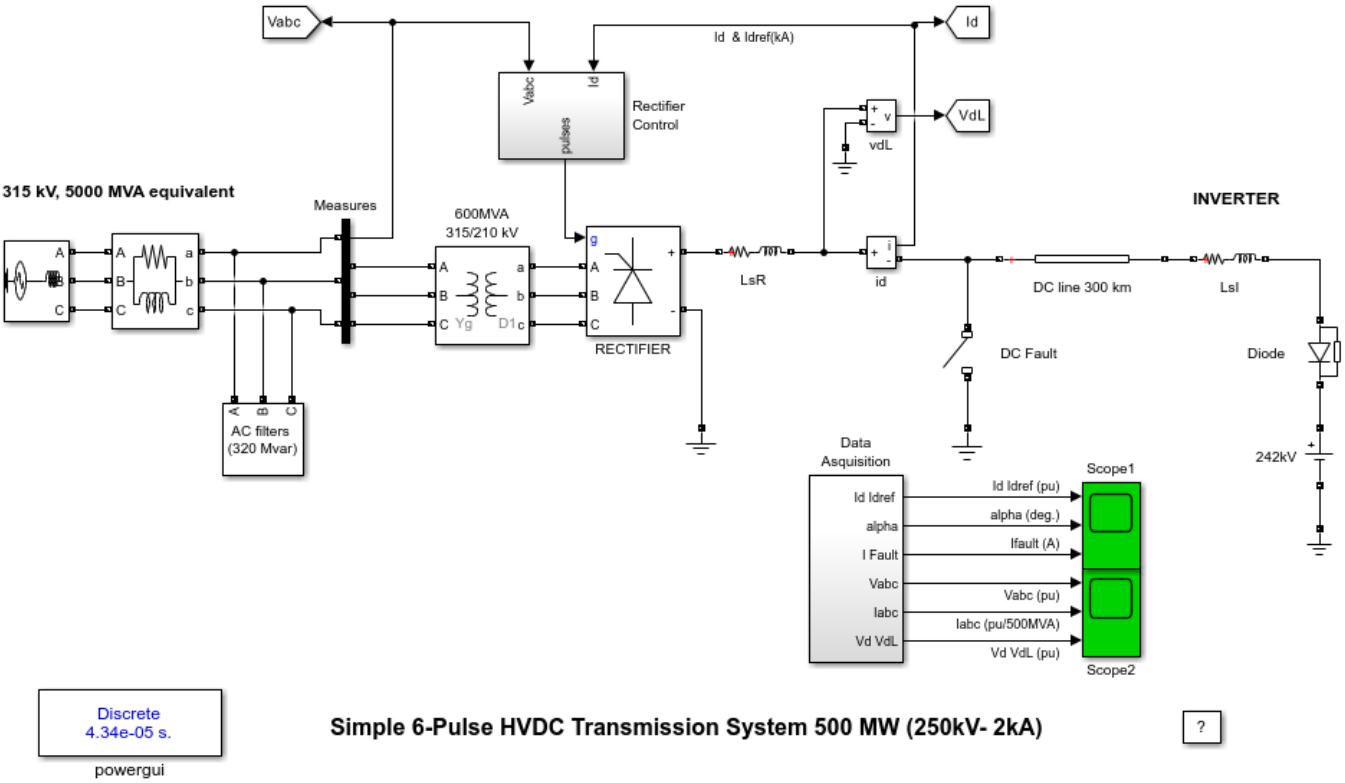
At t = 5 cycles the line protection relays (not simulated) open breakers CB1 and CB2 and the energy stays constant at 13 MJ. As the maximum energy does not exceed the 30 MJ threshold level, the gap is not fired. After breaker opening the fault current drops to a small value and the line and series capacitance start to discharge through the fault and the shunt reactance. The fault current extinguishes at the first zero crossing after the opening order given to the fault breaker (t = 6 cycles). Then, the series capacitor stops discharging and its voltage oscillates around 220 kV .

### Three-Phase-to-Ground Fault

Change the fault type to a three-phase-to-ground fault by checking Phases A, B, and C in the Fault Breaker block. Restart the simulation. Notice that during the fault the energy dissipated in the MOV builds up faster that in the case of a line-to-ground fault. The energy reaches the 30 MJ threshold level after 3 cycles, one cycle before opening of the line breakers. As a result, the gap is fired and the capacitor voltage quickly discharges to zero through the damping circuit.

# Simple 6-Pulse HVDC Transmission System

This example shows the steady-state and transient performance of a simple 500 MW (250 kV-2kA) HVDC transmission system.




### Description

A 500 MW (250 kV, 2 kA) DC interconnection is used to transmit power from a 315 kV, 5000 MVA AC network. The network is simulated by a LLR damped equivalent (impedance angle of 80 degrees at 60 Hz and 3rd harmonic). The converter transformer and the rectifier are modelled respectively with the Universal Transformer and Universal Bridge blocks The converter is a 6-pulse rectifier. It is connected to a 300 km distributed parameter line through a 0.5 H smoothing reactor LsR. The inverter is simulated by a simple DC voltage source in series with a diode (to force unidirectional conduction) and smoothing reactor LsI. The reactive power required by the converter is provided by a set of filters (C bank plus 5th, 7th and high pass filters; total 320 Mvar). Open the AC filter subsystem to see the filter topology. A circuit breaker facilitates the application of a DC line fault on the rectifier side.

Voltages sent to the synchronization system are filtered by 2nd order band pass filters. The whole control system is discretized (Sample time = 1/360/64 = 43.4 us).

The DC line current at the output of the rectifier is compared with a reference. The PI regulator tries to keep the error at zero and outputs the alpha firing angle required by the synchronizing unit. Inputs 3 and 4 of the current regulator allow to bypass the regulator action and to impose the alpha firing angle.

### Simulation

Notice that the system is discretized (sample time 1/360/64 = 43.4 us). Setting, the sample time in to zero, will change to continuous integration for the power system.

The system is programmed to start and reach a steady state. Then, a step is applied on the reference current to observe the dynamic response of the regulator. Finally a DC fault is applied on the line.

Start the simulation and observe the following events on Scope1 :

**0 < t < 0.3 s**

Trace 1 shows the reference current (magenta) and the measured Id current (yellow). The reference current is set to 0.5 pu (1 kA). The DC current starts from zero and reaches a steady-state in 0.1 s. Trace 2 shows the alpha firing angle required to obtain 0.5 pu of current (30 degrees).

**0.3 < t < 0.5 s**

At t = 0.3 s, the reference current is increased from 0.5 pu (1 kA) to the nominal current 1pu (2 kA). The current regulator responds in approximately 0.1 s (6 cycles). The alpha angle decreases from 30 degrees to 15 degrees.

**0.5 < t < 0.55 s**

At t = 0.5 s, a DC fault is applied on the line. The fault current ( trace 3) increases to 5 kA and the Id current increases to 2 pu (4 kA) in 10 ms. Then, the fast regulator action lowers the current back to its reference value of 1 pu.

**0.55 < t <0.57 s**

At t = 0.55 s, the alpha angle is forced by the protection system (not simulated) to reach 165 degrees when the Forced_alpha input of the current regulator goes high (1). The rectifier thus passes in inverter mode and sends the energy stored in the line back to the 345 kV network. As a result, the arc current producing the fault rapidly decreases. The fault is cleared at t = 0.555 s when the fault current zero crossing is reached.

**0.57 < t < 0.8 s**

At t = 0.57 s, the regulator is released and it starts to regulate the DC current again. The steady-state 1 pu current is reached at t = 0.75 s.

### Frequency analysis of AC and DC voltages and currents

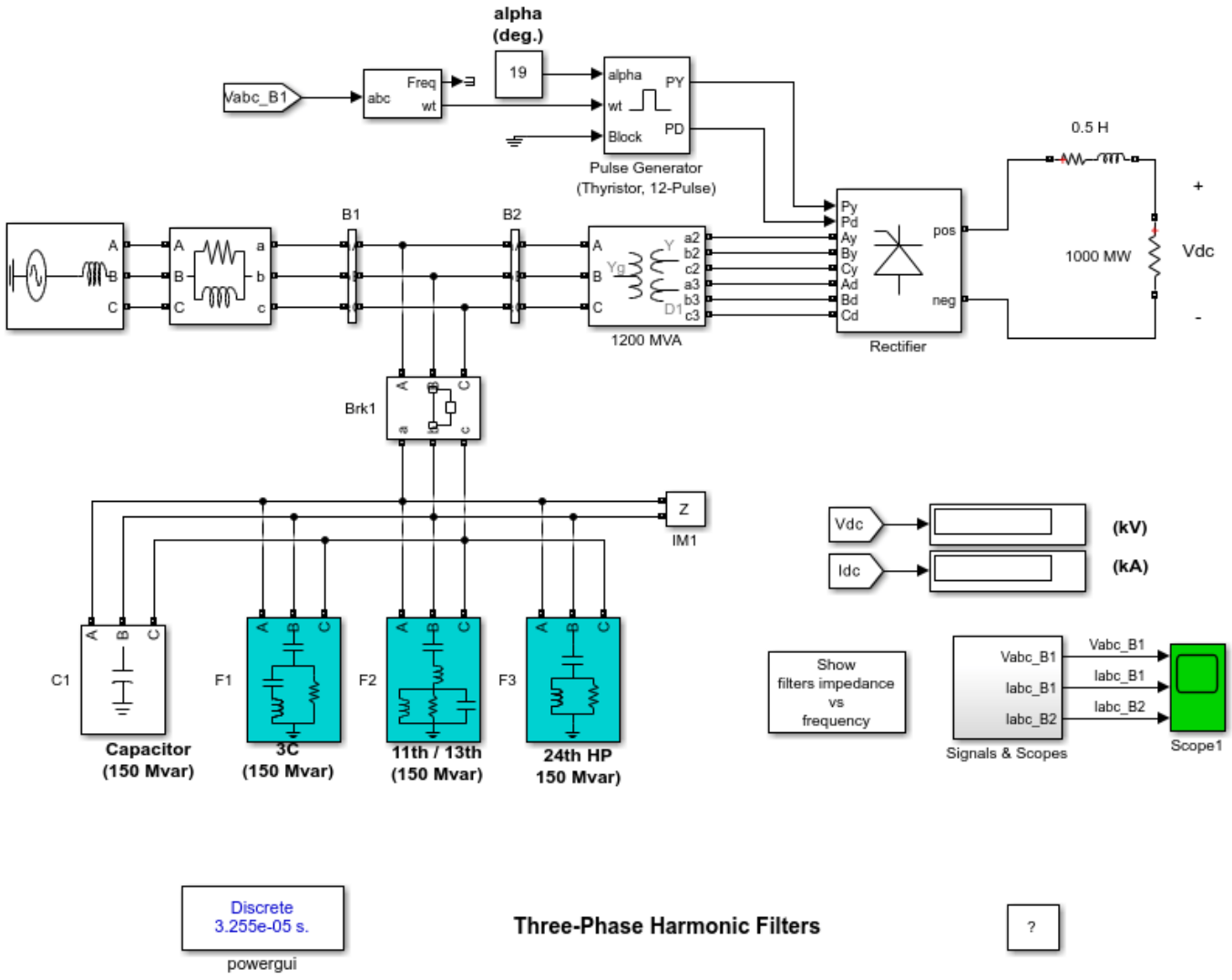
In order to allow further signal processing, signals displayed on Scope2 have been saved in a variable named 'psbvdc_str'(structure with time). These signals are: AC voltages(input 1), AC currents(input 2) and DC voltages on rectifier and line side of the smoothing reactor(input 3).

Open the Powergui and select 'FFT Analysis'. In the FFT window select structure 'psbhvdc_str'. The 0 - 2000 Hz FFT will be performed on a 2-cycle window starting at t = 0.8 - 2/60 (last 2 cycles of recording). Select input labeled 'Iabc'. By default Phase a current (signal number 1) is selected. Press on 'Display' and observe the frequency spectrum of AC current. Harmonic currents (order 6n+/-1 , n = 1,2,3, ...for a 6-pulse converter) are displayed in % of the fundamental component. If you analyze AC voltage (input Vabc) you should notice that the highest harmonics generated by the converter (5th and 7th) are filtered out by the two filters tuned at the 5th and 7th harmonics. Finally, select input

labeled 'Vd VdL (pu)' and then input 1 or input 2. You will obtain harmonic content (order 6n) of the DC voltage Vd (rectifier side) or VdL (line side) expressed in % of the DC component.

# Three-Phase Harmonic Filters

This example shows three-phase harmonic filters in HVDC installations.




### Description

In HVDC installations, AC harmonic shunt filters are used to:

1) reduce harmonic voltages and currents in the power system,

2) supply the reactive power consumed by the converter. To illustrate these concepts, a 1000-MW (500 kV, 2kA) HVDC rectifier is simulated.

The HVDC rectifier is built up from two 6-pulse thyristor bridges connected in series. The converter is connected to the system with a 1200-MVA Three-Phase transformer (three windings). A 1000-MW

resistive load is connected to the DC side through a 0.5 H smoothing reactor. The filters set is made of the following four components of the powerlib/Elements library:

- one capacitor banks (C1) of 150 Mvar modeled by a "Three-Phase Series RLC Load",

- three filters modeled using the "Three-Phase Harmonic Filter"

(1) one C-type high-pass filter tuned to the 3rd (F1) of 150 Mvar

(2) one double-tuned filter 11/13 th (F2) of 150 Mvar

(3) one high-pass filter tuned to the 24th (F3) of 150 Mvar

The total Mvar rating of the filters set is then 600 Mvar. A three-phase circuit breaker (Brk1) is used to connect the filters set on the AC bus.

### Simulation

#### Time-domain simulation

Run a first simulation with an alpha firing angle of 19 degrees. You should get a DC voltage level of 500 kV. Now, look inside Scope1. Compare the currents flowing into Bus B1 (Iabc_B1, axis 2) with those flowing into Bus B2 (Iabc_B2, axis 3). You can see that the harmonic filters almost eliminate the harmonics generated by the converter. If you use the FFT tool of the powergui, you will find that the harmonic filters reduce the THD of the current injected in the system from 9% to 0.7%. You can also perform other simulations with various values of alpha. Notice the impact on the DC level and on the generated harmonics.

#### Frequency response

You will now plot the impedance vs frequency of the harmonic filters:

1) Disconnect the filters from the AC bus. To do so, double-click on the breaker Brk1, select 'open' for initial status of breakers and click on the OK button.

2) Open the powergui and select "Impedance vs Frequency Measurement".

3) Click on the Display/Save button. Specialized Power Systems will compute and display the filters frequency response.

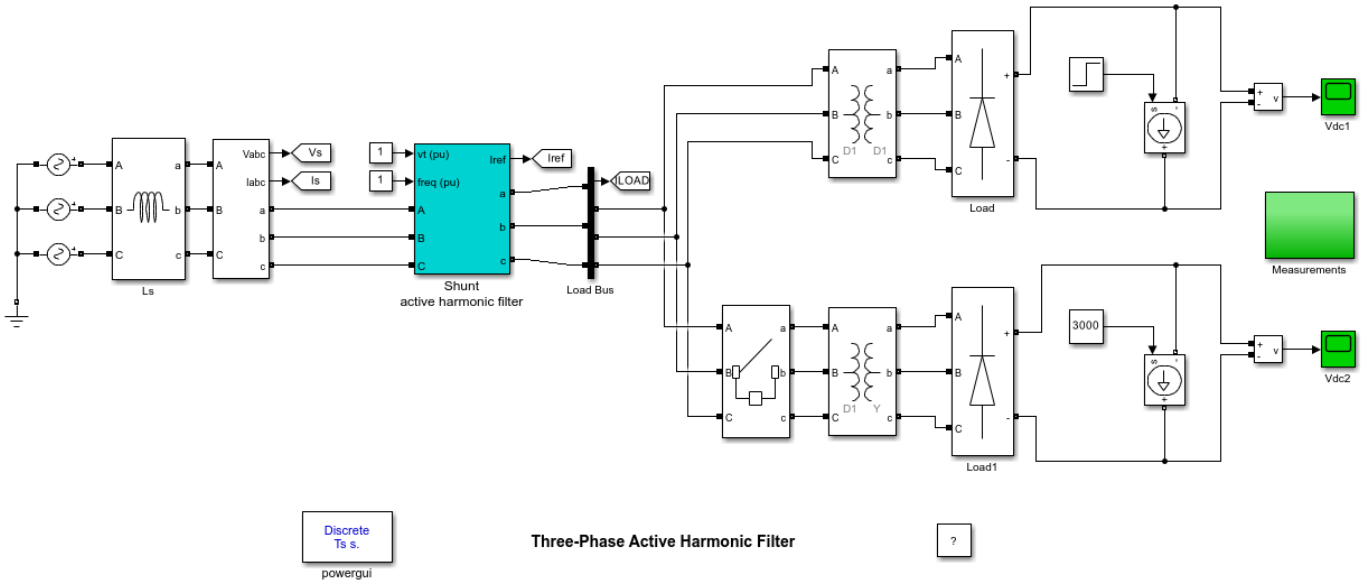
4) Double-click on the block "Show filters impedance vs frequency". A second figure, showing the pre-computed filters frequency response will be displayed. The impedance data of the two figures should be identical.

5) If you zoom the figure (using the "Tool" menu), you should find an impedance of 417 ohms capacitive (-90deg.) at 60 Hz. This value confirms that the total reactive power of the filters at 60 Hz is: Qc =V^2/Xc = 500e3^2/417 = 600 Mvar.

# Three-Phase Active Harmonic Filter

This example shows the use of a shunt active harmonic filter (AHF) to minimize the harmonic content propagated to the source from a non-linear load.

Graham Dudgeon, Senior Consultant, The MathWorks, Inc.




## Description

The circuit models a standard shunt AHF with IGBT inverter and series inductor on the AC side and DC capacitor energization. The load consists of two diode rectifiers which are phase-shifted by 30 degrees. The Delta-Y connected rectifier is connected after 10 cycles to change the load from 6-pulse to 12-pulse.

The AHF uses a PLL to generate a reference sinusoidal source current which is in-phase and has the same RMS gain as the load current. The current error between the load current and the reference current is generated by the IGBT Bridge through hysteresis switching. The AHF aims to inject this current error at the point of common coupling in order to match the source current as closely as possible with the reference current.

## Simulation

Start the simulation. You can observe phase A reference and source current on scope 'Iref_Isource' and phase A source and load current on scope 'Isource_Iload'. If you look under the mask of the AHF, further responses specific to the operation of the AHF are available.

At t=0s, only the delta/delta connected diode rectifier is in-circuit. The DC current is set at 2000A. Observe from 'Iref_Isource' that the AHF captures the reference current within 1 cycle.

FFT Analysis tool of Powergui show an FFT analysis of the load and source phase A current on the 3rd cycle. It is seen that the AHF has effectively reduced the THD from 22.41% to 0.69%.

At t=5/60s, the DC current is increased from 2000A to 3000A. Observe that the AHF effectively responds to this change in load and captures the new reference current within one cycle.

At t=10/60s, the delta/wye diode rectifier is connected, thus producing a 12-pulse load. Observe that the AHF again captures the new reference current within one cycle. FFT Analysis tool of Powergui shows the load and source phase A current on the 13th cycle. It is seen that the AHF has effectively reduced the THD from 4.80% to 0.93%.

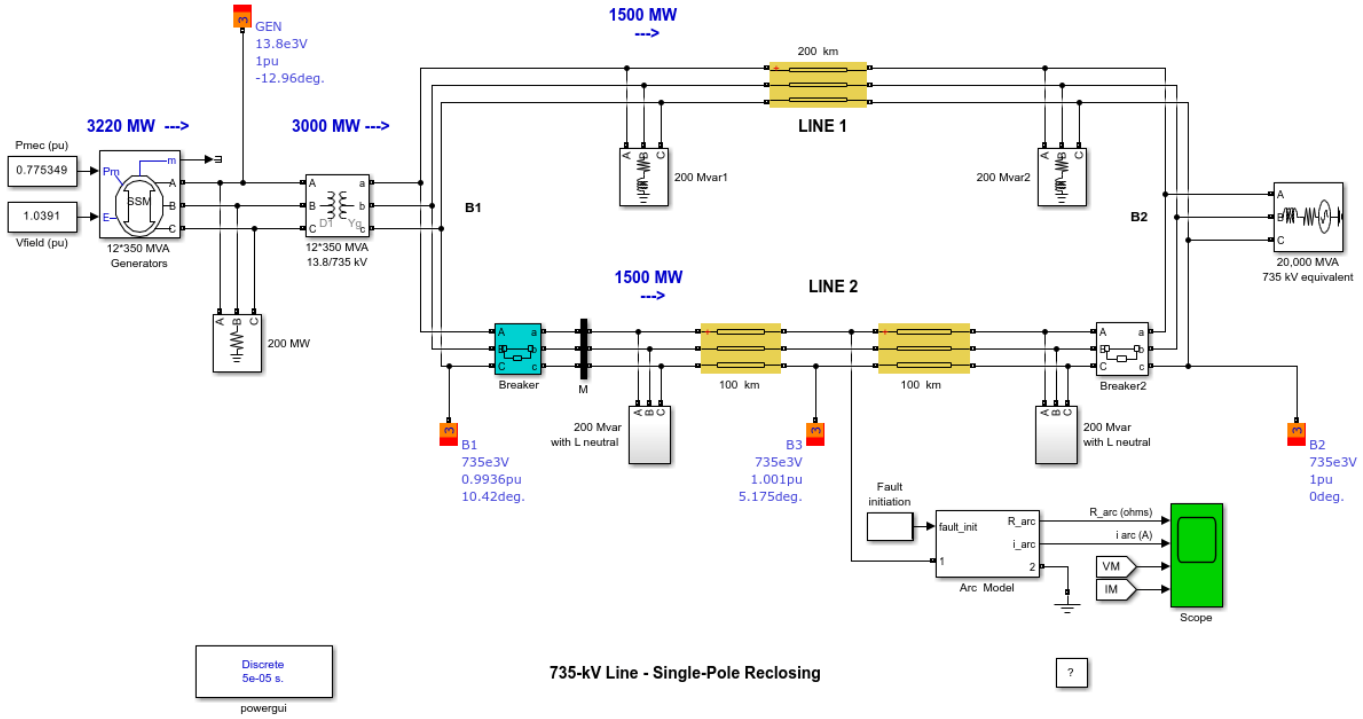
**Notes**

The circuit has been discretized with a 2us time step. This time step may be increased but a decrease in accuracy of the simulation will be observed.

# Three-Phase Line - Single-Pole Reclosing

This example shows the use of three-phase blocks to study phase-to-ground fault and single-pole reclosing of a 735-kV transmission line

G. Sybille (Hydro-Quebec)




## Description

Two 735 kV parallel lines, 200 km long, transmit 3000 MW of power from a generation plant (12 generators of 350 MVA) to to an equivalent network having a short circuit level of 20 GVA.

The generation plant is simulated with a simplified synchronous machine (sub-transient reactance of 0.22 pu). The machine is connected to the transmission network through a 13.8 kV/ 735 kV Wye-Delta transformer.

The line models are distributed parameter lines. The lines are assumed to be transposed and their parameters R,L,C /km are specified in positive- and zero-sequence components. Each line is shunt compensated by two shunt reactors of 200 Mvars each, connected at line ends. Applying single pole reclosing at this voltage level is made possible by the use of neutral inductances for the two shunt reactances of line 2. Otherwise, the 'secondary arc current' which is induced into the fault ( mainly because of the capacitive coupling between the two sound phases and the faulted phase) would be too high to allow arc extinction after opening of line breakers on the faulted phase. If you open the two shunt reactance blocks of line 2, observe how the optimum neutral inductance is computed.

## Fault and Line Switching

A phase-to-ground fault is applied at the middle of line 2. In order to apply the fault along the line, this line is simulated in two sections of 100 km. As soon as the fault is detected by the protection

relays (not simulated here) , an opening command is sent to the two line breakers of the faulted phase. The breaker are kept open during a certain 'dead time' , usually around 0.5 s, during which the arc normally extinguishes, then the two breakers are reclosed

When the two line breakers are tripped on the faulted phase, the fault current is interrupted but a small current will continue to flow through the arc. If this secondary arc current is too large (typically above 50 A), the arc cannot extinguish and the breaker will reclose on the fault.

**Arc Model**

The arc is modeled by a fixed or non-linear resistance R = f(Iarc_rms). The arc extinguishes when its rms current falls below a threshold value (typically 50 A) defined in the arc model block. Open the Arc Model block and look at the arc parameters. The mean arc resistance is programmed as an exponential function of the rms current. The mean arc resistance increases when the rms arc current decreases so that the time for arc current to decay below the threshold value is shortened . (With the specified parameters, Rarc = 0.1ohm and 30 ohms respectively for currents of 1kA and 100A). The Arc Model block is a masked block. Use 'Look under mask' to see how the arc model is implemented.

Open the blocks simulating the two line breakers. See how the line opening/reclosing sequence is programmed. The fault is applied at t = 1 cycle. Then, the opening command is sent to both breakers at t = 4 cycles (3 cycles detection + opening time). The two breakers are reclosed at t = 34 cycles after a dead time of 30 cycles, during which the arc creating the fault should extinguish.

**Simulation**

Start the simulation and observe the voltage and current waveforms on the 4-trace oscilloscope.

Observe the three phase-to-ground voltages and currents at sending end of line 2 and the current flowing into the fault. The machine has been initialized to deliver 3220 MW at 1 pu voltage so that a net power of 3000 MW is flowing into Line 1 and Line 2. The line currents flowing into each line are therefore 15pu/100 MVA as observed on trace 2.

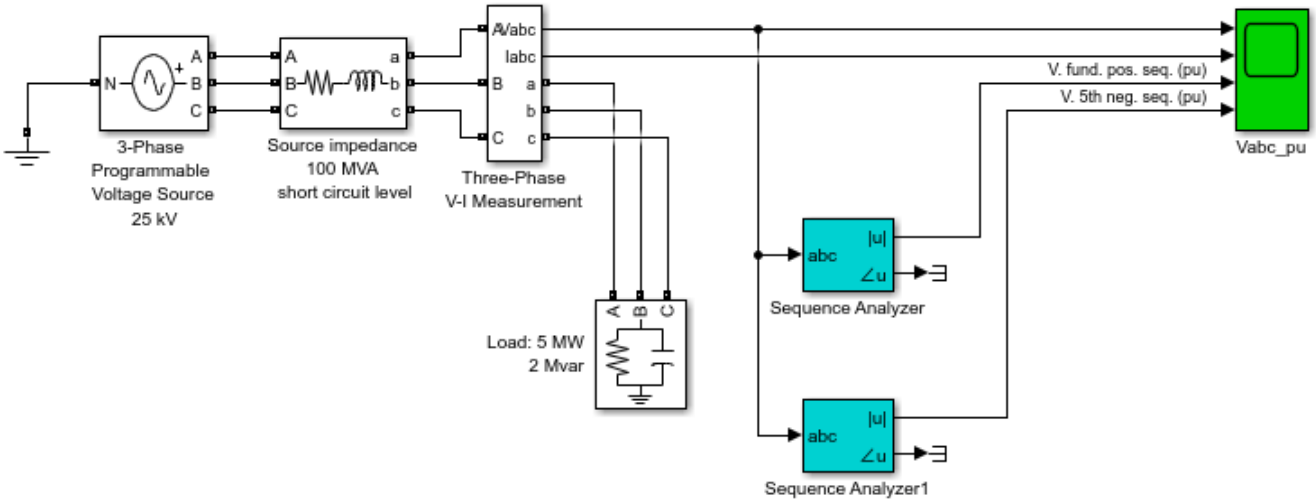
The fault current (I_arc) is measured in amperes. It reaches 22 kA during the first cycle, then it drops to a very small value after 3 cycles when the two line breakers open. Use Y-zoom to look at the secondary arc current. It contains a slowly decaying DC component and a fundamental component (12 A peak). As its rms value is below 50 A, the arc extinguishes at the first current zero crossing.

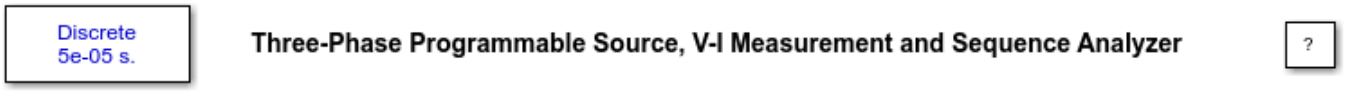
Now, open the Arc Model block menu and change the arc model to a fixed resistance (0.1 ohm). Restart the simulation. Notice that the DC component of the arc current prevents arc extinction, so that the line is now reclosed on a fault. You can also suppress the neutral reactance in the two shunt reactance blocks and see the impact on the secondary arc current.

# Three-Phase Programmable Source, V-I Measurement and Sequence Analyzer

This example shows the use of the 3-Phase Programmable Voltage Source, 3-Phase V-I Measurement, and Sequence Analyzer blocks.

G. Sybille (Hydro-Quebec)




### Description

A 25kV, 100 MVA short-circuit level, equivalent network feeds a 5MW, 2 Mvar capacitive load.

The internal voltage of the source is controlled by the Discrete 3-phase Programmable Voltage Source block.

Open the source dialog box and look at the parameters controlling the voltage. A positive-sequence of 1.0 pu, 0 degrees is specified. At t = 0.05 s a step of 0.5 pu is applied on positive-sequence voltage magnitude, then, at t = 0.1 s, 0.08 pu of 5th harmonic in negative sequence is added on the 1.5 pu voltage.

In order to start simulation in steady state, the three voltage sources are initialized with a positive-sequence voltage of 25 kV, 0 degree, 60 Hz.

The Three-Phase V-I Measurement block allows to monitor the three load voltages and currents. Open its dialog box and see how this block allows to output the voltages and currents in pu.

Two Discrete 3-phase Sequence Analyzer blocks are used to monitor the positive-sequence of the three fundamental voltages Va, Vb, Vc and the negative-sequence component of the 5th harmonic.

The whole system, (power network, programmable source, and sequence analyzers) is discretized at a 50 us sample time.

### Simulation

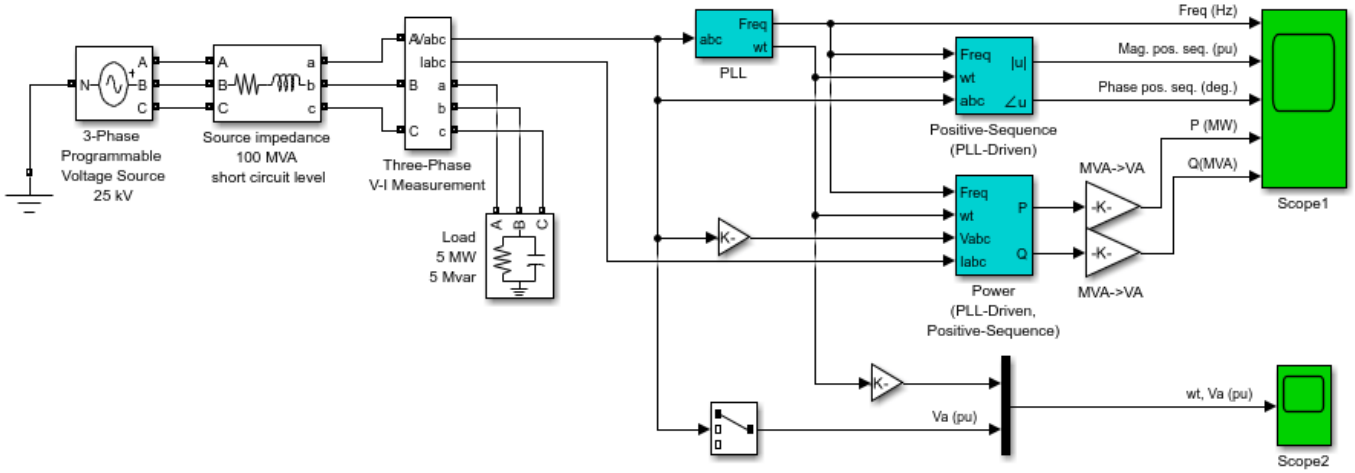
Choose Simulation/Start and observe the voltage and current waveforms.

The simulation starts in steady state with 1 pu voltage. At t = 0.05 sec, the internal voltage is increased to 1.5 pu. At t = 0.10 sec. 0.08 pu of 5th harmonic internal voltage is added.

Observe the fundamental component (pos. seq.) and the 5th harmonic component (neg. seq.) measured by the two sequence analyzers. The 0.08 pu 5th harmonic internal voltage is amplified at 0.14 pu at the load terminals. As the sequence analyzers use Fourier analysis, their response time is one cycle.

# Three-Phase Programmable Source, PLL, Voltage and Power Measurement

This example shows the use of the 3-Phase Programmable Voltage Source, PLL and Variable-Frequency Positive-Sequence Voltage and Power Measurement blocks.




Discrete
5e-05 s.

**Three-Phase Programmable Source, PLL, Voltage and Power Measurement**

?

### Description

A 25kV, 100 MVA short-circuit level, equivalent network feeds a 5 MW, 5 Mvar capacitive load. The internal voltage of the source is controlled by the Discrete 3-Phase Programmable Voltage Source block.

Open the programmable source dialog box and look at the parameters controlling the voltage and frequency. A 60 Hz, positive-sequence of 1.0 pu, 45 degrees is specified.

At t = 0.5 s, a sinusoidal modulation of the frequency (amplitude 3 Hz, frequency 0.4 Hz) is started. The modulation stops at t =3 s so that a full cycle of modulation can be observed.

The Three-Phase V-I Measurement block is used to monitor the three load voltages and currents. Open its dialog box and see how this block allows to output the three voltages and currents in p.u.

The Discrete 3-Phase PLL block measures the frequency and generates a signal (wt output) locked on the variable frequency system voltage. The PLL drives two measurement blocks taking into account the variable frequency: one block computing the fundamental value of the positive-sequence load voltage and another one computing the load active and reactive powers. These two blocks and the PLL are initialized in order to start in steady state.

The whole system, (power network, PLL and measurement blocks) is discretized at a 50 us sample time.

### Simulation

Start the simulation.

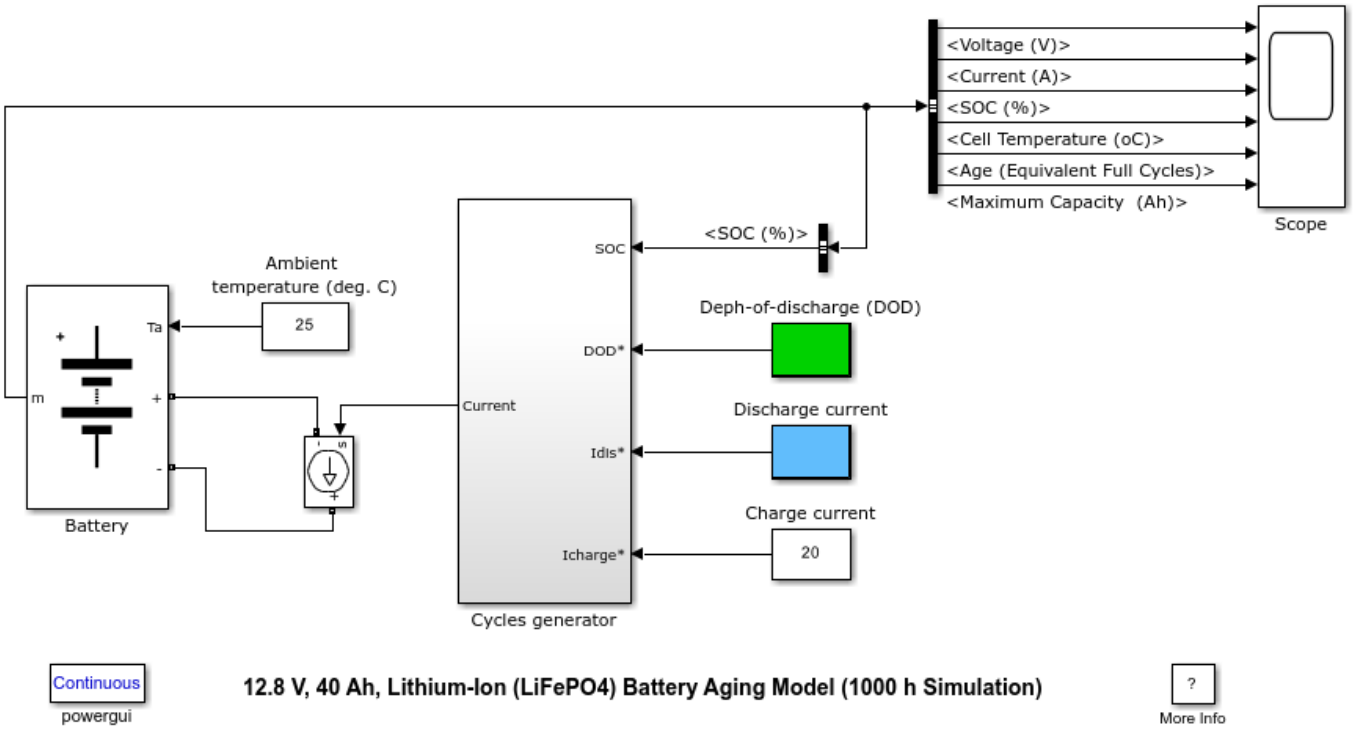
Observe on Scope1 block how the PLL tracks the changing system frequency (trace 1). As expected, the system frequency varies between 57 Hz and 63 Hz with a maximum rate of change of 7.5 Hz/s. Traces 2 and 3 show the variations of magnitude and phase of positive-sequence voltage as the frequency is changing. Traces 4 and 5 show the variations of active and reactive powers.

Observe on Scope2 that the (wt) ramp output of the PLL stays locked on zero crossings of phase A voltage.

# 12.8 V, 40 Ah, Lithium-Ion (LiFePO4) Battery Aging Model (1000 h Simulation)

This example shows the impact of aging (due to cycling) on a 12.8 V, 40 Ah, Lithium-Ion battery module.

Souleman Njoya M., Louis-A. Dessaint (Ecole de technologie superieure, Montreal)




### Circuit Description

This demo illustrates the effect of aging (due to cycling) on the performance of a 12.8 V, 40 Ah Lithium-Ion battery model. The battery is submitted for 1000 hours, to several discharge-charge cycles at ambient temperature of 25 degrees C , and at various depths of discharge (DOD) and discharge rates. As observed from the Scope, the impact of DOD and discharge rate on the battery life is as expected. As the DOD or discharge rate increases, the battery ages rapidly, which quickly reduces the battery capacity.

### Simulation

The demonstration shows the life cycle performance of a 12.8 V, 40 Ah, Lithium-Ion battery during shallow and deep discharges, at various discharge rates. At first, the battery is cycled to a DOD of 20 % at 0.5 C discharge rate. Afterwards, the DOD and discharge rate are increased to 80 % and 2 C, respectively. Start the Simulation and open the Scope to view all signals.

At t = 0 s, the battery cycling starts with a discharge current of 20 A (0.5 C-rate), at ambient temperature of 25 degrees C. The initial SOC is 100 %. The discharge goes on until the SOC reaches

80 % (DOD of 20%). Afterwards, the battery is charged to 100 % SOC with a charge current of 20 A. As this cycle is repeated, the battery age increases whereas its capacity decreases. The internal cell temperature also increases from 25 degrees C to 33 degrees C.

At t = 200 h, the battery is now discharged to 20 % SOC (DOD of 80 %) and charged to 100 % SOC. This cycle is repeated for another 200 hours. As observed, the battery age starts to increase rapidly.

At t = 400 h, the cycling DOD is brought back to 20 % for another 200 hours. This slows down the aging process of the battery.

At t = 600 h, the discharge current is increased to 80 A (2 C-rate). This causes the internal cell temperature to rise from 33 degrees C to 43 degrees C. As this cycle is repeated, the battery ages more rapidly, which quickly reduces the battery capacity.

At t = 800 h, the discharge current is brought back to 0.5 C-rate for another 200 hours, which slows down the aging process of the battery.

**References**

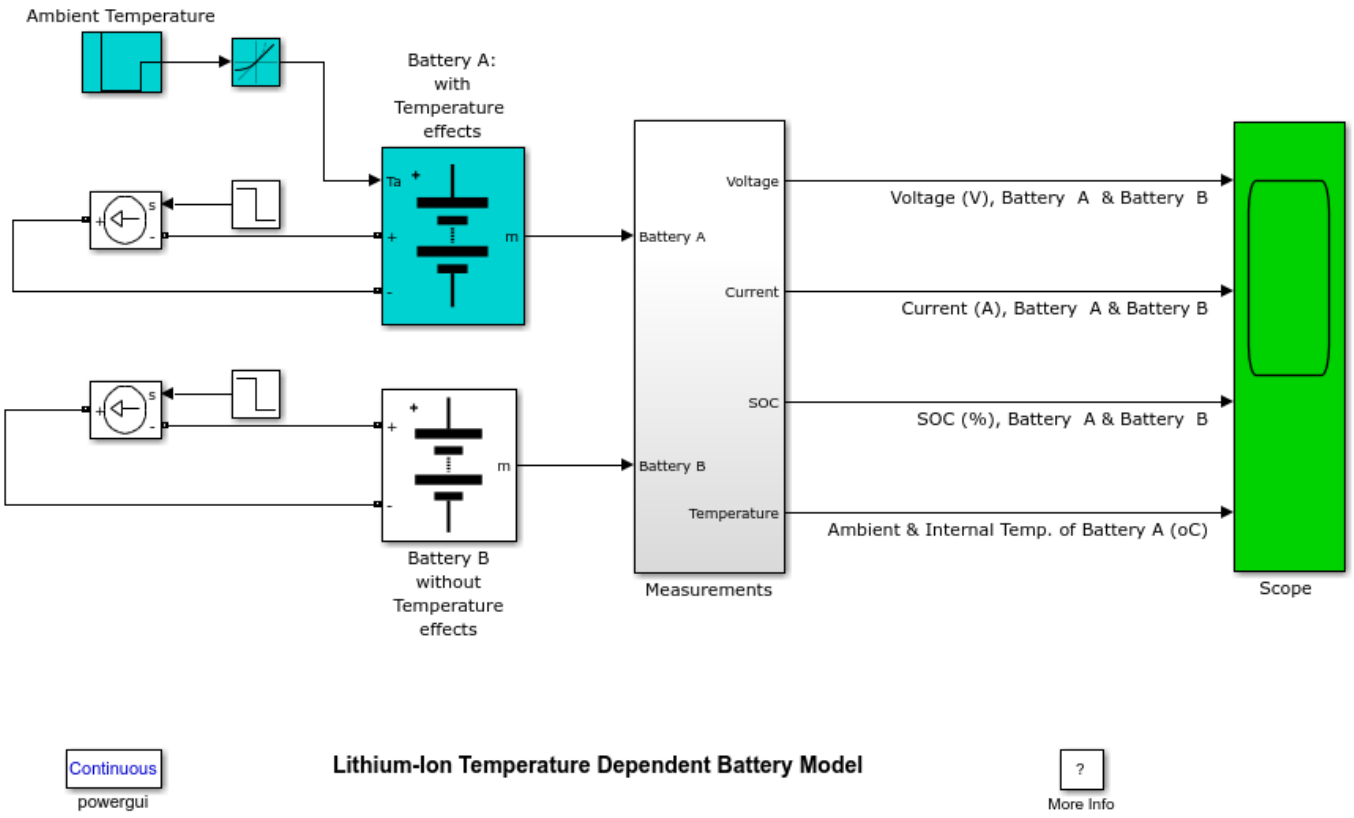
1. O. Tremblay, L.-A. Dessaint, A.-I. Dekkiche, A Generic Battery Model for the Dynamic Simulation of Hybrid Electric Vehicles, 2007 IEEE® Vehicle Power and Propulsion Conference, September 9-13, 2007, Arlington/Texas, USA.

2. N. Omar, M. A. Monem, Y. Firouz, J. Salminen, J. Smekens, O. Hegazy, H. Gaulous, G. Mulder, P Van den Bossche, T. Coosemans, J. Van Mierlo, Lithium iron phosphate based battery - Assessment of the aging parameters and development of cycle life model, Applied Energy, Volume 113, January 2014, Pages 1575-1585.

# Lithium-Ion Temperature Dependent Battery Model

This example shows the impact of temperature on a 7.2 V, 5.4 Ah, Lithium-Ion battery module.

Souleman Njoya M., Louis-A. Dessaint (Ecole de technologie superieure, Montreal)




### Circuit Description

This demo illustrates the effect of temperature on the performance of a 7.2 V, 5.4 Ah Lithium-Ion battery model. The model (which includes the impact of cell/ambient temperature on the voltage, capacity and resistance) is submitted to a variable ambient temperature during a discharge and charge process. Its performance is compared to the case where the impact of temperature is neglected. As observed from the Scope, the temperature dependent battery model performs close to reality. As the cell/internal temperature increases/decreases due to charge(or discharge) heat losses and ambient temperature variations, the output voltage and capacity also increase/decrease.

### Simulation

The demonstration shows the performance of the temperature dependent Lithium-Ion battery model (Battery A) when the ambient temperature is varied from 20 degrees C to -20 degrees C and then to 0 degrees C. Battery B represents the case where the effect of temperature is neglected. Start the Simulation and open the Scope to view all signals.

At t = 0 s, the Battery A and B are discharged with 2 A at ambient temperature of 20 degrees C.

At t = 150 s, the internal temperature has increased to its steady state value of 29.2 degrees due to heat losses from the discharge process. This causes the output voltage of Battery A to slightly increase, while battery B output voltage continues to decrease.

At t = 1000 s, the ambient temperature is decreased to -20 degrees C. This causes the output voltage of Battery A to greatly decrease as the internal temperature decreases rapidly. Also the SOC of Battery A decreases due to the reduction of battery capacity. The battery B output voltage continues to decrease slowly to its steady state voltage.

At t = 2000 s, the ambient temperature is increased from -20 degrees C to 0 degrees C. As the internal temperature increases, the output voltage of Battery A increases. Also, as the capacity increases, the SOC of Battery A increases. The Battery B output voltage remains constant to its steady state value.

At t = 2500 s, the Battery A and B are charged with 3 A at ambient temperature of 0 degrees C. This causes the internal temperature to increase due to heat losses during the charge process, which increases the charging voltage of Battery A. Afterwards, Battery A and B continue to charge up until fully charged.

**References**

1. O. Tremblay, L.-A. Dessaint, A.-I. Dekkiche, A Generic Battery Model for the Dynamic Simulation of Hybrid Electric Vehicles, 2007 IEEE Vehicle Power and Propulsion Conference, September 9-13, 2007, Arlington/Texas, USA.

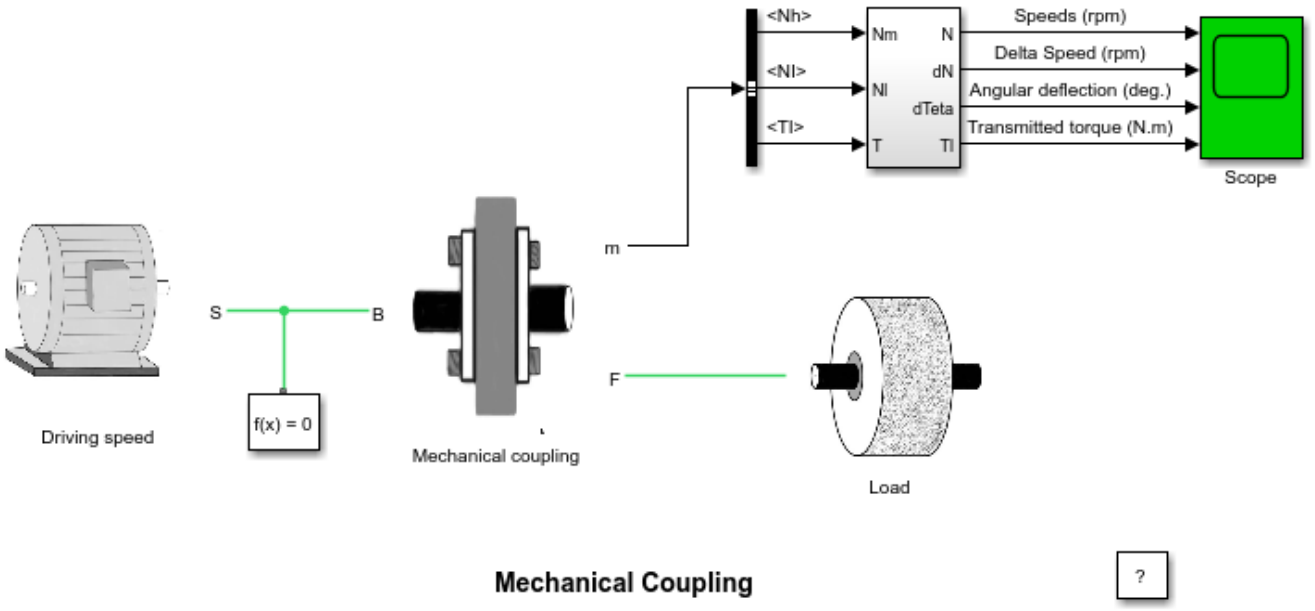
3. Cong Zhu, Xinghu Li, Lingjun Song, Liming Xiang, Development of a theoretically based thermal model for lithium ion battery pack, Journal of Power Sources, Volume 223, 1 February 2013, Pages 155-164.

2. L.H. Saw, K. Somasundaram, Y. Ye, A.A.O. Tay, Electro-thermal analysis of Lithium Iron Phosphate battery for electric vehicles, Journal of Power Sources, Volume 249, 1 March 2014, Pages 231-238.

# Shaft Coupling (Simscape model)

This example shows a Simscape™ model of mechanical coupling shaft.

Ernesto Vilchez-Ynca, Louis-A. Dessaint (Ecole de technologie superieure, Montreal)




### Description

The model outputs the transmitted torque through the shaft regarding the speed difference between the driving side and the loaded side of the shaft.

The shaft has a stiffness of 17190 N.m and an internal damping factor of 600 N.m.s. This shaft is designed to have 0.1 degree of angular deflection for a 30 N.m load torque.

The shaft is driven by a variable speed source and is connected to a load. The load has an inertia of 0.35 kg.m2 and a viscous friction term of 0.006 N.m.s

### Simulation

Start the simulation. You can observe the driving and the load speeds, the speed difference, the angular deflection and the transmitted torque values on the scope.

At t = 0 s, the driving speed starts climbing to 1750 rpm with a 500 rpm/s acceleration ramp. The angular deflection jumps to about 0.06 degree and the shaft transmits about 18.5 Nm to the load in order to accelerate it. At t = 0.2 s, the driving and load speeds tend to equalize. During the acceleration phase, the angular deflection increases slowly in order to transmit a higher torque to compensate the viscous friction increase.

At t = 3.5 s, the driving speed settles at 1750 rpm. This reduces the angular deflection and also the transmitted torque which settles around 1.1 Nm to compensate the viscous friction of the load.

At t = 5 s, the driving speed lowers towards 0 rpm with a -500 rpm/s deceleration ramp. The angular deflection becomes negative and so does the transmitted torque in order to decelerate the load.

During the deceleration phase, the angular deflection increases in order to transmit a higher deceleration torque to compensate the reduction of viscous friction.

At t = 8.5 s, the driving speed stabilizes at 0 rpm. This causes the angular deflection to reduce to 0 degree, the transmitted torque becomes null and the load stops.

**Notes**

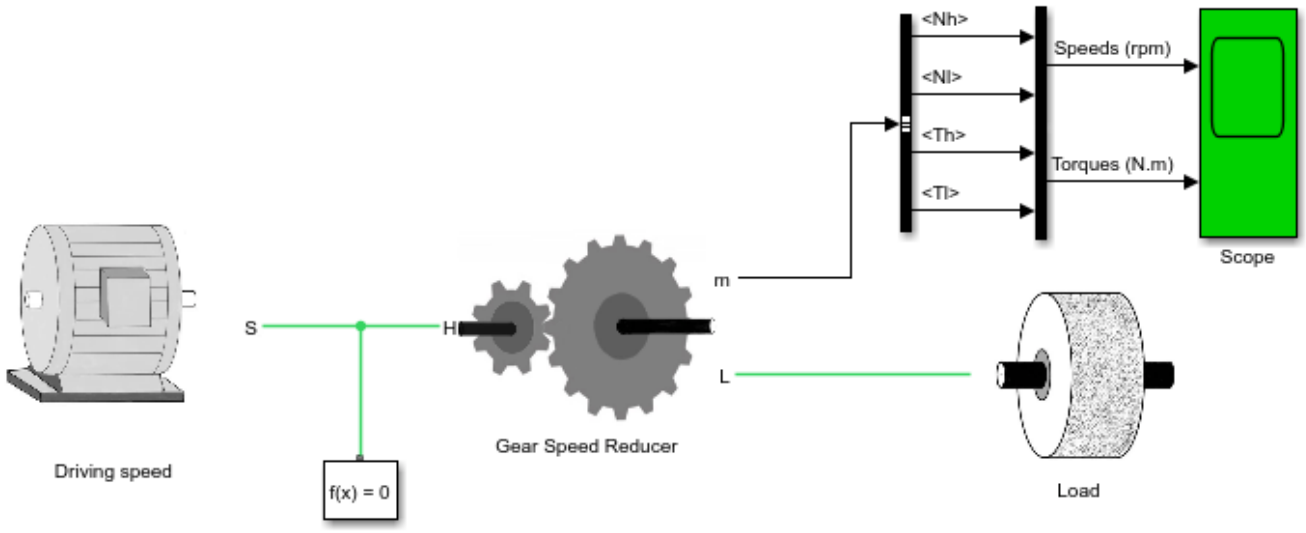
1) The shaft has been discretized with a 10 us time step.

2) In order to reduce the number of points stored in the scope memory, a decimation factor of 10 is used.

# Gear Speed Reducer (Simscape model)

This example shows a Simscape™ model representing a gear speed reducer device connected between a high-speed shaft and a low-speed shaft.

Ernesto Vilchez-Ynca, Louis-A. Dessaint (Ecole de technologie superieure, Montreal)




**Gear Speed Reducer (Simscape model)**

### Description

The speed reducer is driven by a variable speed source and is connected to a load. The load has an inertia of 30 kg.m2 and a viscous friction term of 0.5 N.m.s.

The reduction device has a reduction ratio of 10 and it's inertia with respect to the high-speed side is 0.0005 kg.m2. The reduction ratio being quite low, the efficiency is high and worth 0.95.

The high-speed shaft has a stiffness of 17190 N.m and an internal damping factor of 600 N.m.s. This shaft is designed to have 0.1 degree of angular deflection for a 30 N.m load torque. The low-speed shaft, having a higher torque to transmit, has a stiffness of 171900 N.m and an internal damping factor of 6000 N.m.s. This shaft is designed to have 0.1 degree of angular deflection for a 300 N.m load torque.

### Simulation

Start the simulation. You can observe the driving (high-speed) and load speeds (low-speed), the torque transmitted by the high-speed shaft and the torque transmitted by the low-speed shaft on the scope.

At t = 0 s, the driving speed starts climbing to 1750 rpm with a 500 rpm/s acceleration ramp. This causes the transmitted torque of the high-speed shaft to jump to about 18 N.m. Because of the

reduction device, the torque transmitted to the load by the low-speed shaft is a lot bigger and is worth about 170 N.m.

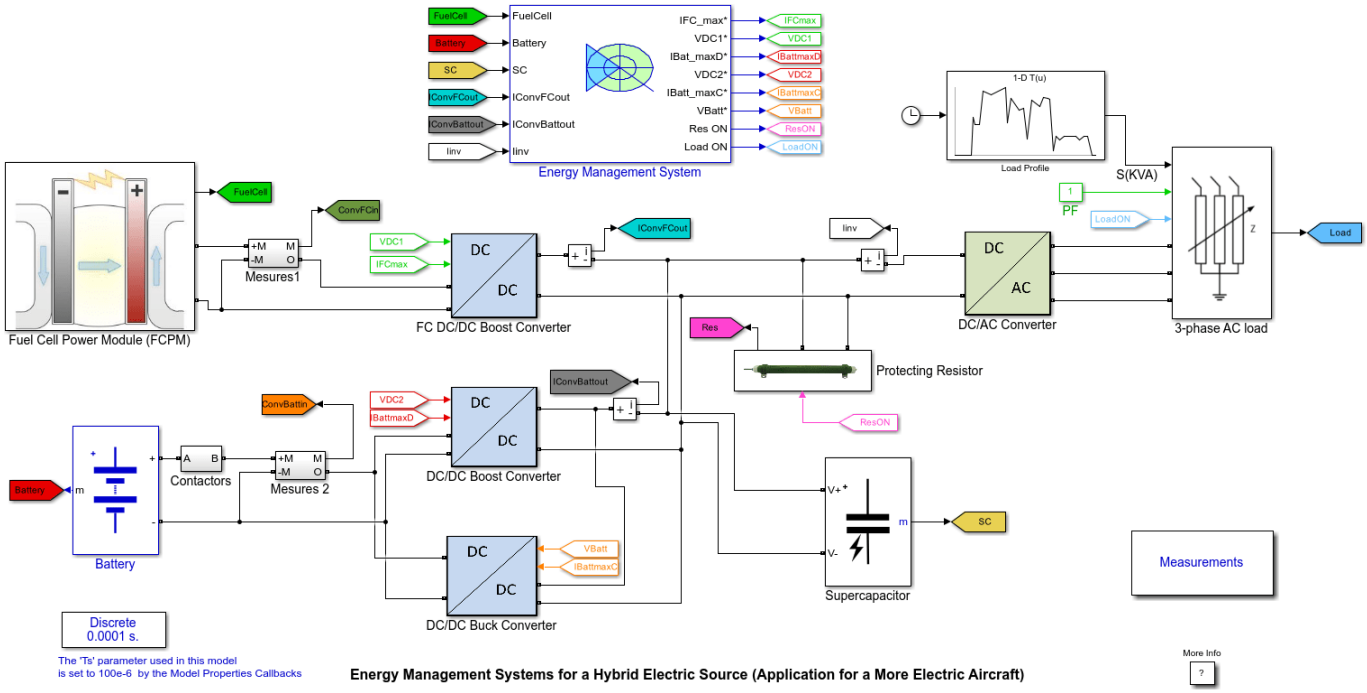
During the accelerating phase, both torques keep increasing in order to compensate the viscous friction of the load. Notice that the load accelerates with a ramp of +50 rpm/s due to the reduction ratio of the speed reducer.

At t = 3.5 s, the driving speed settles at 1750 rpm. Since no more accelerating torque is needed, the input and output torques decrease and stabilize to 0.965 N.m and 9.16 N.m respectively at t = 4 s. The load speed is now equal to 175 rpm.

# Energy Management Systems for a Hybrid Electric Source (Application for a More Electric Aircraft)

This example shows energy management systems for a fuel cell hybrid electric source.

Souleman Njoya M., Louis-A. Dessaint (Ecole de technologie superieure, Montreal) and Susan Liscouet-Hanke (Bombardier Aerospace)




## Circuit Description

This example illustrates a simulation model of a fuel cell based emergency power system of More Electric Aircraft (MEA). As the landing-gear and flight control systems become more electric in MEA, the peak electrical load seen by the conventional emergency power system (ram air turbine or air-driven generator) increases. Consequently, there is a potential risk of overloading the ram air turbine (RAT)/air-driven generator (ADG) at lower aircraft speeds, where the produced power is nearly zero. A more robust emergency power system is needed to ensure a safe landing of MEA. This model presents an alternative emergency power system based on fuel cells, lithium-ion batteries and supercapacitors. The demo also features different energy management systems for a fuel cell hybrid electric source.

The fuel cell hybrid power system is designed based on a representative emergency flight profile of a Bombardier aircraft and consists of the following:

- A 12.5 kW (peak), 30-60 V PEM (proton exchange membrane) fuel cell power module (FCPM), with nominal power of 10 kW.
- A 48 V, 40 Ah, Li-ion battery system.
- A 291.6 V, 15.6 F, supercapacitor system (six 48.6v cells in series)

- A 12.5 kW fuel cell DC/DC boost converter, with regulated output voltage and input current limitation.
- Two DC/DC converters for discharging (4 kW boost converter) and charging (1.2 kW buck converter) the battery system. These converters are also output voltage regulated with current limitation. Normally, a single bidirectional DC/DC converter can also be used to reduce the weight of the power system.
- A 15 kVA, 270 V DC in, 200 V AC, 400 Hz inverter system.
- A 3 phase AC load with variable apparent power and power factor, to emulate the MEA emergency load profile.
- A 15 kW protecting resistor to avoid overcharging the supercapacitor and battery systems.
- An energy management system, which distributes the power among the energy sources according to a given energy management strategy. Five types of energy management strategies are implemented, which are:

1. The state machine control strategy
2. The Classical PI control strategy
3. The frequency decoupling and state machine control strategy
4. The equivalent consumption minimization strategy (ECMS)
5. The external energy maximization strategy (EEMS)

### Demonstration

The demonstration shows the performance of the fuel cell hybrid emergency power system during a five minutes emergency landing scenario. In this scenario, the fuel cell hybrid power system supplies the essential loads during the following events:

- Instantly when the main generators are lost (this is normally assumed by the Avionic and APU battery system till the RAT/ADG is fully deployed).
- Emergency hydraulic pump start-ups.
- Motion of Flaps/Slats and gear down.
- Taxiing and passengers evacuation (also normally assumed by the Avionic and APU battery system as the RAT/ADG becomes unavailable).

Depending on the type of energy management strategy selected, the energy management system controls the power of each energy source devices through the reference signals (output voltage and maximum current) of the fuel cell and battery DC/DC converters. Double click on the **Energy Management System** block and select for example the **State Machine Control Strategy**. Start the simulation. Double click on the **Measurements** block. Open the **Power** scope (showing the power distribution referred to the 270 V dc bus) together with the **Fuel Cell**, **Battery**, **SuperCap** and **Load** scopes. The following explains what happens during this simulated emergency landing scenario:

At t = 0 s, the essential loads are supplied by the main generators and the fuel cell hybrid power system is turned ON to prepare for an unlikely emergency landing situation.

At t = 5 s, the fuel cell begins to recharge the battery with its optimal power (around 1 kW).

At t = 40 s, all generators are lost. The fuel cell hybrid power system takes over the essential loads. At this time the extra load power required is instantly supplied by the supercapacitor due to its fast dynamics, while the fuel cell power increases slowly.

At t = 45 s, the supercapacitor is discharged below the required DC bus voltage (270 V) and the battery starts providing power to regulate the bus voltage back to 270 V.

At t = 48.5 s, the DC bus or supercapacitor voltage reaches 270 V and the battery reduces its power slowly to zero. The fuel cell provides the total load power and continues to recharge the supercapacitor.

At t = 60 s, an emergency hydraulic pump is started, and the supercapacitor provides the extra transient load power, while the fuel cell power increases slowly.

At t = 61.5 s, the battery comes online to regulate the DC bus voltage to 270 V and helps the fuel cell by providing the extra load power required.

At t = 70 s, the fuel cell reaches its maximum power (the FCPM power was limited to 9 kW due to its DC/DC converter input voltage range) and the extra load power is provided by the battery.

At t = 110 s, the battery also reaches its maximum power (4 kW) and the supercapacitor provides the extra load power.

At t = 125 s, the load power reduces below the fuel cell maximum power. Due to the slow fuel cell dynamics, the extra fuel cell power during transients is transferred to the supercapacitor.

At t = 126 s, the DC bus voltage reaches 270 V and the battery power drops to zero.

At t = 130 s, a second emergency hydraulic pump is turned ON and the fuel cell hybrid power system behavior is similar to when the first hydraulic pump was turned ON.

At t = 170 s, the load power reduces below the fuel cell maximum power and the extra fuel cell power is transferred to both the battery and supercapacitor.

At t = 180 s, the load is suddenly increased due to the motion of Flaps/Slats and landing gears. Once again, the supercapacitor responds quickly by providing the extra load power.

At t = 185 s, the battery discharges to regulate the DC bus voltage and helps the fuel cell with the extra load power required.

At t = 235 s, the aircraft has landed and the load power decreases suddenly. The extra fuel cell energy is stored in the battery and supercapacitor.

At t = 250 s, the aircraft is taxiing and the fuel cell supplies nearly the total load power required.

At t = 330 s, the passengers have been evacuated and the load power reduces to zero. The fuel cell reduces its power slowly to its optimal power and recharges the battery.

**Notes**

1. In order to reduce the amount of memory used, a decimation factor of 100 is used for all scopes, except for the **Load** scope (which uses a decimation factor of 10).

2. Average-value models of DC/DC and DC/AC converters are used to speed up the simulation.

3. Select a different energy management strategy in the **Energy Management System** block and compare its performance in terms of hydrogen consumption, storage (battery/supercapacitor) energy used and overall efficiency.

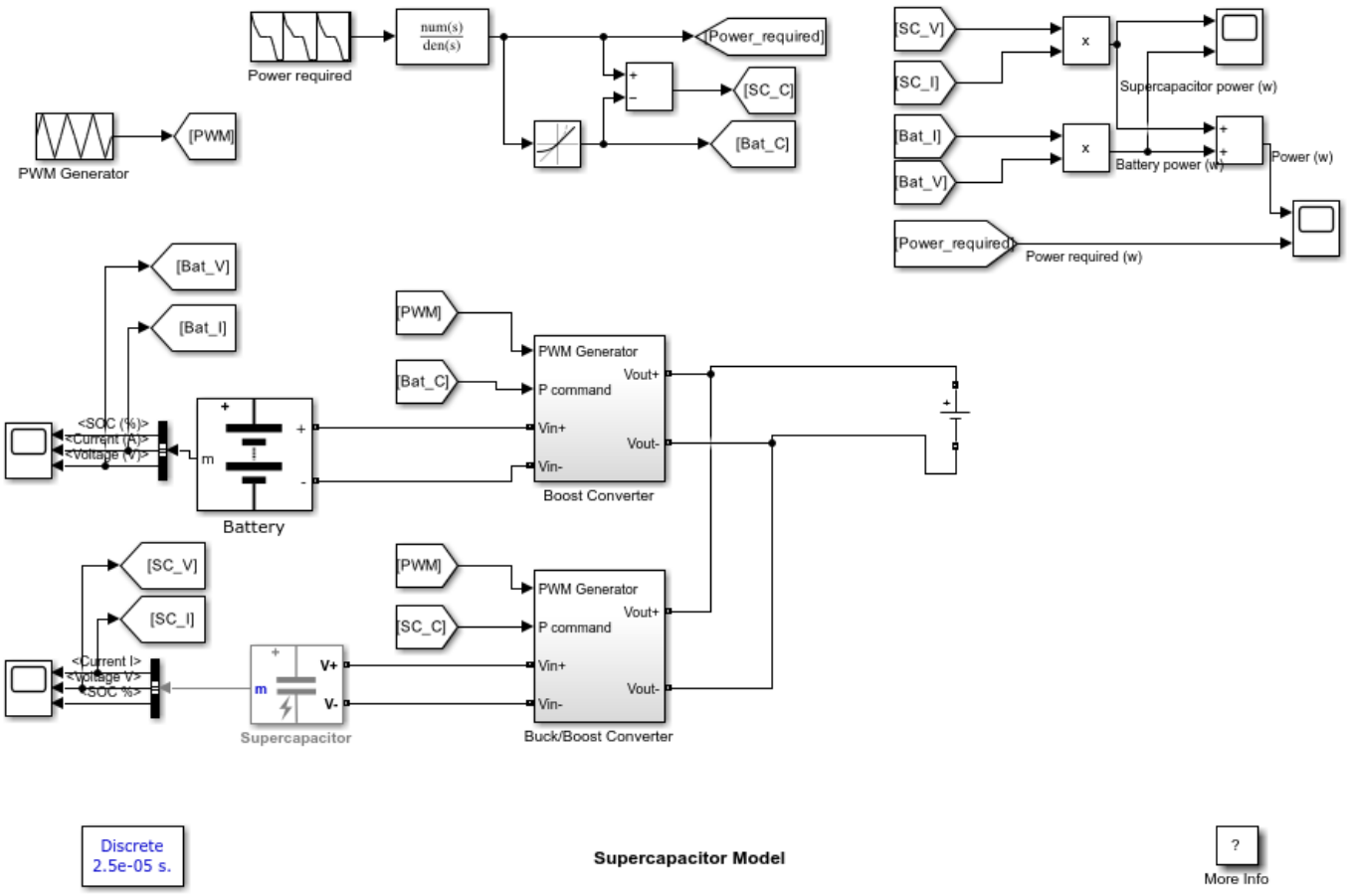
### References

1. S. Njoya Motapon, L.A. Dessaint and K. Al-Haddad, "A Comparative Study of Energy Management Schemes for a Fuel Cell Hybrid Emergency Power System of More Electric Aircraft," IEEE Transactions on Industrial Electronics, 2013 (IEEE Early access).

# Supercapacitor Model

This example shows the supercapacitor model during charge and discharge.

Pierre Clement Blaud, L.-A. Dessaint (Ecole de technologie superieure, Montreal)



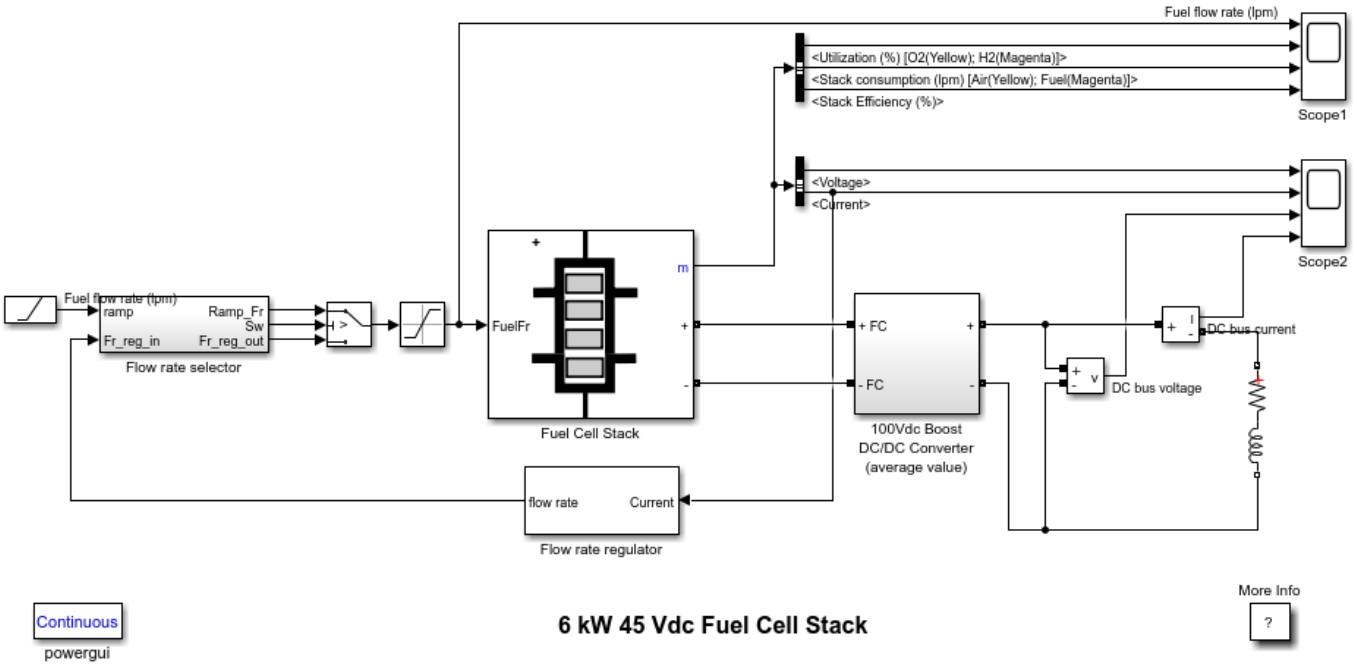

## Description

The circuit illustrates a simple hybridization of a supercapacitor with a battery. The supercapacitor is connected to a Buck/Boost converter and the battery is connected to a Boost converter. Power of the battery is limited by a rate limiter block, therefore the transient power is supplied to the DC bus by the supercapacitor.

# 6 kW 45 Vdc Fuel Cell Stack

This example shows the Proton Exchange Membrane (PEM) Fuel Cell Stack model feeding an average value 100Vdc DC/DC converter.

Olivier Tremblay, Njoya Motapon Souleman, Louis-A. Dessaint (Ecole de technologie superieure, Montreal)




### Description

The nominal Fuel Cell Stack voltage is 45Vdc and the nominal power is 6kW. The converter is loaded by an RL element of 6kW with a time constant of 1 sec. During the first 10 secs, the utilization of the hydrogen is constant to the nominal value (Uf_H2 = 99.56%) using a fuel flow rate regulator. After 10 secs, the flow rate regulator is bypassed and the rate of fuel is increased to the maximum value of 85 lpm in order to observe the variation in the stack voltage. That will affect the stack efficiency, the fuel consumption and the air consumption.

Fuel cell voltage, current, DC/DC converter voltage and DC/DC converter current signals are available on the Scope2. Fuel flow rate, Hydrogen and oxygen utilization, fuel and air consumption, and efficiency are available on the Scope1.

### Simulation

At t = 0 s, the DC/DC converter applies 100Vdc to the RL load (the initial current of the load is 0A). The fuel utilization is set to the nominal value of 99.56%. The current increases to the value of 133A. The flow rate is automatically set in order to maintain the nominal fuel utilization. Observe the DC bus voltage (Scope2) which is very well regulated by the converter. The peak voltage of 122Vdc at the beginning of the simulation is caused by the transient state of the voltage regulator.

At t = 10 s, the fuel flow rate is increased from 50 liters per minute (lpm) to 85 lpm during 3.5 s reducing by doing so the hydrogen utilization. This causes an increasing of the Nernst voltage so the fuel cell current will decrease. Therefore the stack consumption and the efficiency will decrease (Scope1).

**Notes**

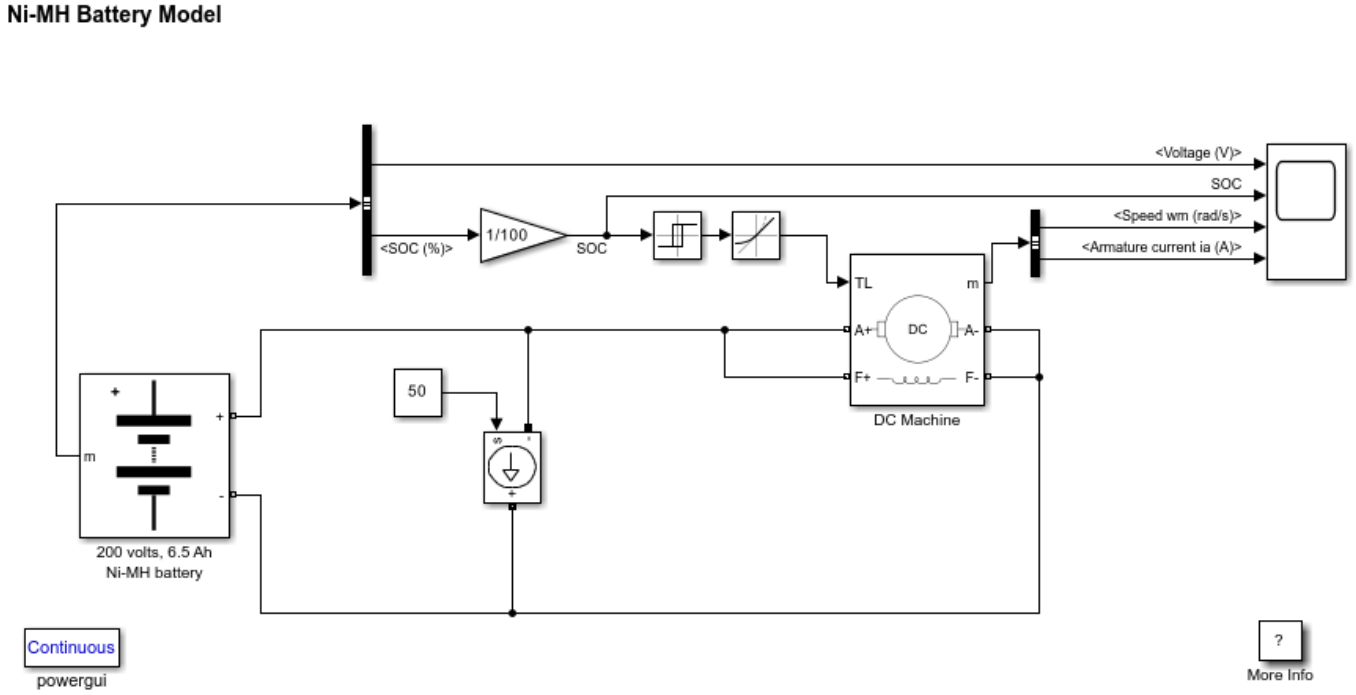
1) You can vary more than 7 signals by checking the checkbox in the "Signal variation" tab of the Simulink® mask. Refer to documentation for more information.

2) A simplified version of the model allows simulating the fuel cell with less parameters using only the V-I curves. Select "Model detailed Level" to "Simplified". Note that the signal variation is not available in this mode; hence the Fuel flow rate input is not available anymore. Also the utilization, the stack consumption and the efficiency are 0 for the "Simplified" mode.

# Ni-MH Battery Model

This example shows a 200 V, 6.5 Ah Ni-MH battery model during charge and discharge process.

Olivier Tremblay, Louis-A. Dessaint (Ecole de technologie superieure, Montreal)




### Description

The battery is connected to a constant load of 50 Amps. The DC machine is connected in parallel with the load and operates at no load torque. When the State-Of-Charge (SOC) of the battery goes under 0.4 (40%), a negative load torque of 200 Nm is applied to the machine so it acts as a generator to recharge the battery. When the SOC goes over 80%, the load torque is removed so only the battery supplies the 50 amps load.

Battery voltage, SOC, Motor speed and Motor current signals are available at the output of the block.

### Simulation

At t = 0 s, the DC machine is started with the battery power. The speed increases to 120 rad/s. The battery is also discharged by the constant DC load of 50 amps. At t = 280 s, the SOC drops under 40%. A mechanical torque of -200 Nm is applied to the machine so it acts as a generator and provides a current of 100 amps. Hence, 50 amps goes to the load and 50 amps goes to recharge the battery. At t = 500 s, the SOC goes over 80%. The mechanical torque is removed and the machine operates free. And the cycle restarts.

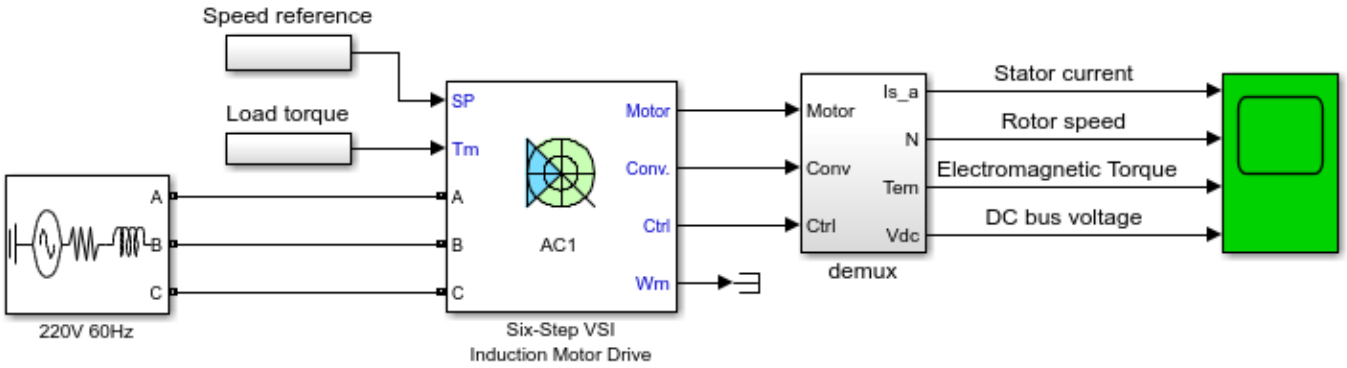
### References

1. O. Tremblay, L.-A. Dessaint, A.-I. Dekkiche, "A Generic Battery Model for the Dynamic Simulation of Hybrid Electric Vehicles", 2007 IEEE® Vehicle Power and Propulsion Conference, September 9-13, 2007, Arlington/Texas, USA

# AC1 - Six-Step VSI Induction 3HP Motor Drive

This example shows the AC1 Six-Step VSI Induction Motor Drive during speed regulation.

H.Blanchette, L.-A. Dessaint (Ecole de technologie superieure, Montreal)



### Description

The induction motor is fed by a voltage source inverter, which is built using the Universal Bridge Block. The DC bus voltage is produced by a thyristor rectifier and regulated using a PI controller in order to maintain a constant volts per hertz ratio. A braking chopper limits the DC bus voltage increase during motor deceleration or when the load torque tends to accelerate the motor. The motor drives a mechanical load characterized by inertia J, friction coefficient B, and load torque TL.

Motor current, speed, and torque signals are available at the output of the block.

### Simulation

A speed reference step from 0 to 1800 rpm is applied at t = 0. The speed set point doesn't go instantaneously at 1800 rpm but follows the acceleration ramp. The motor reaches steady state at t = 1.3 s.

At t = 2 s, an accelerating torque is applied on the motor's shaft. You can observe a speed increase. Since the rotor speed is higher than the synchronous speed, the motor is working in the generator mode. The braking energy is transferred to the DC link and the bus voltage tends to increase. However the over voltage activates the braking chopper which causes the voltage to reduce. In this example, the braking resistance is not big enough to avoid a voltage increase but the bus is maintained within tolerable limits.

At t = 3 s, the torque applied to the motor's shaft steps from -11 N.m to +11 N.m .You can observe a DC bus voltage and speed drop. At this point, the DC bus controller switches from braking to motoring mode.

At t = 4 s, the load torque is removed completely and the electromagnetic stabilizes around zero shortly after.

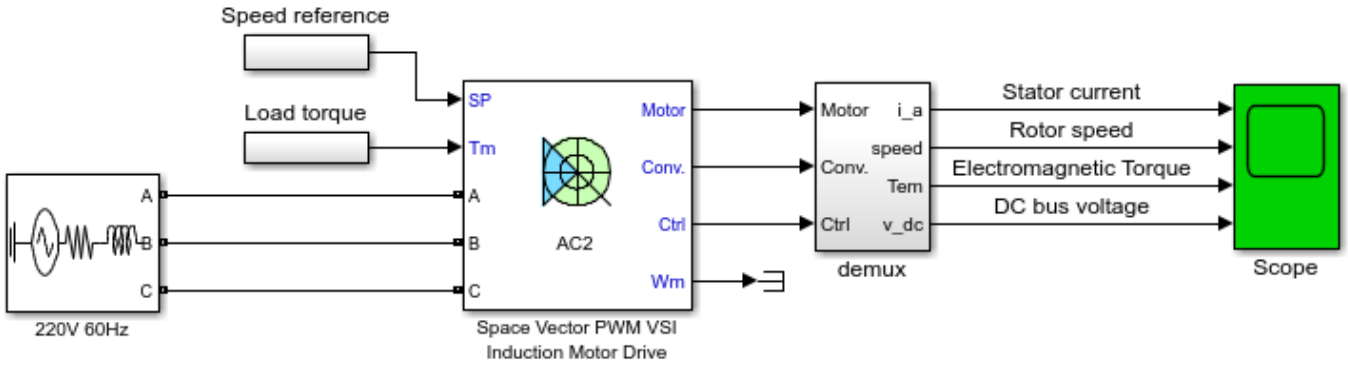
**Notes**

1) The power system has been discretised with a 20us time step

# AC2 - Space Vector PWM VSI Induction 3HP Motor Drive

This example shows a PWM VSI induction motor drive with a braking chopper for a 3HP AC motor.

H.Blanchette, L.-A. Dessaint (Ecole de technologie superieure, Montreal)



### Description

The induction motor is fed by a PWM inverter, which is built using a Universal Bridge Block. The speed controller consists in a PI regulator that produces a slip compensation, which is added to the rotor speed in order to derive the commanded stator voltage frequency. A constant volts per hertz ratio is also applied to the motor. The motor drives a mechanical load characterized by inertia J, friction coefficient B, and load torque TL.

Motor current, speed, and torque signals are available at the output of the block.

### Simulation

Start the simulation. You can observe the motor stator current, the rotor speed, the electromagnetic torque and the DC bus voltage on the scope. The speed set point and the torque set point are also shown.

At time t = 0 s, the speed set point is 1000 rpm. As shown in Figure 0-10, the speed follows precisely the acceleration ramp.

At t = 0.5 s, the full load torque is applied to the motor shaft while the motor speed is still ramping to its final value. This forces the electromagnetic torque to increase to a high value and then to stabilize at 11 N.m once the speed ramping is completed and the motor has reached 1000 rpm.

At t = 1 s, the speed set point is changed to 1500 rpm and the electromagnetic torque reaches again a high value so that the speed ramps precisely at 1800 rpm/s up to 1500 rpm under full load.

At t = 1.5 s, the mechanical load passed from 11 N.m to -11 N.m , which causes the electromagnetic torque to stabilize at approximately at -11 N.m shortly after. Note that the DC bus voltage increases since the motor is in the braking mode. This increase is limited by the action of the braking chopper.

**Notes**

1) The power system has been discretised with a 2 us time step. The speed controller uses a 100 us sample time and the space vector modulator uses a 20 us sample time in order to simulate a microcontroller control device.

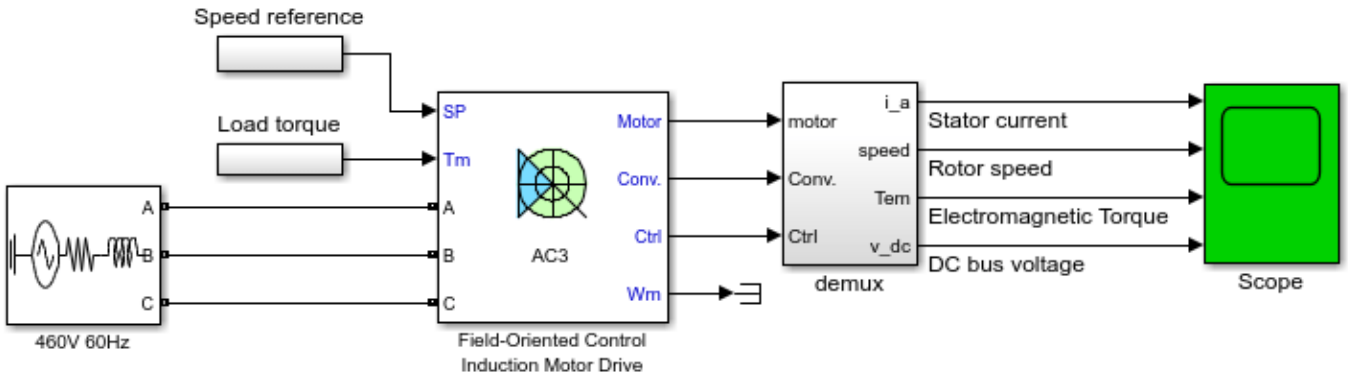
2) In order to reduce the number of points stored in the scope memory, a decimation factor of 25 is used.

3) A simplified version of the model using average-value inverter can be used by selecting 'Average' in the 'Model detail level' menu of the graphical user-interface. The time step can then be increased up to the control system sample time value.This can be done by typing 'Ts = 20e-6' in the workspace in the case of this example. See also ac2_example_simplified model.

# AC3 - Field-Oriented Control Induction 200 HP Motor Drive

This example shows the Field-Oriented Control Induction Motor Drive during speed regulation.

H.Blanchette, L.-A. Dessaint (Ecole de technologie superieure, Montreal)



### Description

This circuit uses the AC3 block of Specialized Power Systems library. It models a field-oriented control (FOC) induction motor drive with a braking chopper for a 200HP AC motor.

The induction motor is fed by a PWM voltage source inverter, which is built using a Universal Bridge Block. The speed control loop uses a PI controller to produce the flux and torque references for the FOC controller. The FOC controller computes the three reference motor line currents corresponding to the flux and torque references and then feeds the motor with these currents using a three-phase current regulator.

Motor current, speed, and torque signals are available at the output of the block.

### Simulation

Start the simulation. You can observe the motor stator current, the rotor speed, the electromagnetic torque and the DC bus voltage on the scope. The speed set point and the torque set point are also shown.

At time t = 0 s, the speed set point is 500 rpm. Observe that the speed follows precisely the acceleration ramp.

At t = 0.5 s, the full load torque is applied to the motor shaft while the motor speed is still ramping to its final value. This forces the electromagnetic torque to increase to the user-defined maximum value (1200 N.m) and then to stabilize at 820 N.m once the speed ramping is completed and the motor has reached 500 rpm.

At t = 1 s, the speed set point is changed to 0 rpm. The speed decreases down to 0 rpm by following precisely the deceleration ramp even though the mechanical load is inverted abruptly, passing from 792 N.m to - 792 N.m, at t = 1.5 s. Shortly after, the motor speed stabilizes at 0 rpm.

Finally, note how well the DC bus voltage is regulated during the whole simulation period.

**Notes**

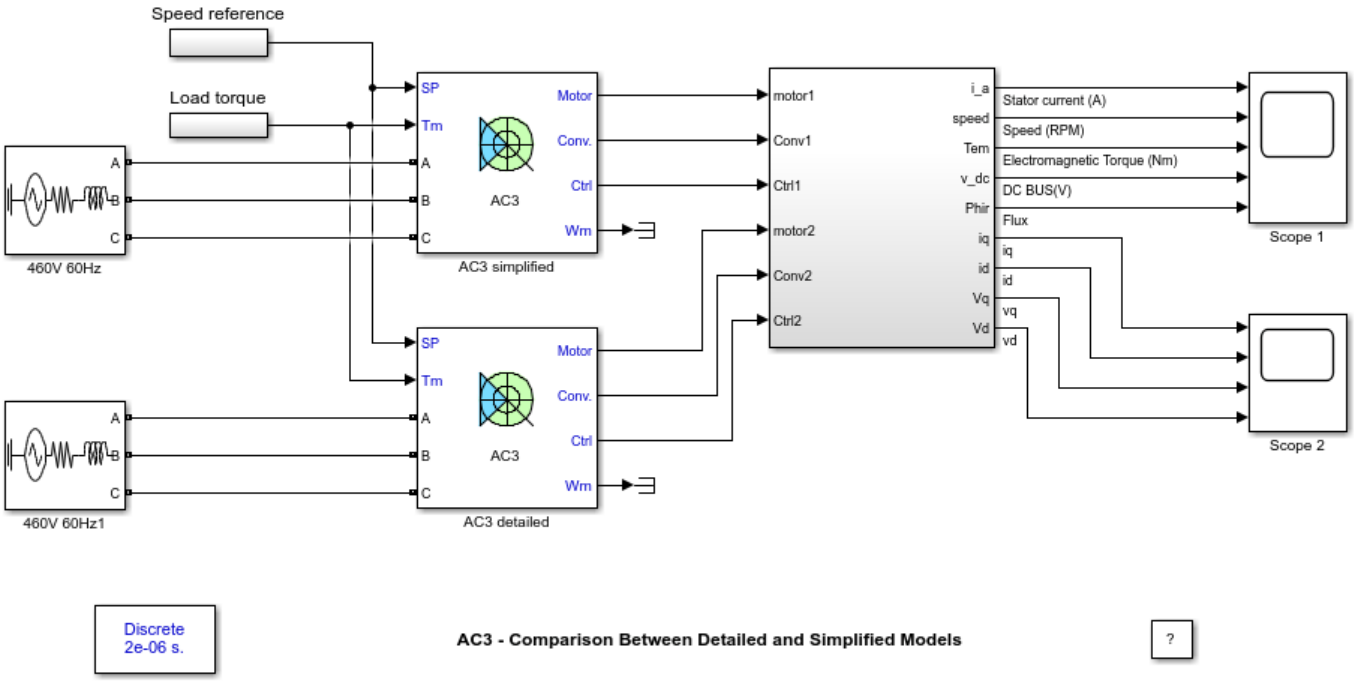
1) The power system has been discretised with a 2 us time step. The speed controller uses a 100 us sample and the vector controller uses a 20us is sample time in order to simulate a microcontroller control device.

2) A simplified version of the model using average-value inverter can be used by selecting 'Average' in the 'Model detail level' menu of the graphical user-interface. The time step can then be increased up to 60 us. This can be done by typing 'Ts = 60e-6' in the workspace and by changing the speed controller sampling time to 120e-6 and the vector controller sampling time to 60e-6. See also ac3_example_simplified model.

# AC3 - Comparison Between Detailed and Simplified Models

This example shows the AC3 detailed model and the AC3 average model during speed regulation. The comparison is performed for normal condition and for inverter saturation condition.

O. Tremblay, L.-A. Dessaint (Ecole de technologie superieure, Montreal)




### Description

This circuit uses two AC3 blocks of the Specialized Power Systems electric drives library. The AC3 block models a field-oriented control (FOC) induction motor drive with a braking chopper for a 200 HP AC motor. The first AC3 block is set to average value model and the second AC3 block is set to detailed model.

For the detailed model, the induction motor is fed by a PWM voltage source inverter, which is built using a Universal Bridge Block. The average value model uses ideal voltages and currents sources to feed the induction motor. The speed control loop uses a PI regulator to produce the flux and torque references for the FOC controller. The FOC controller computes the three reference motor line currents corresponding to the flux and torque references and then feeds the motor with these currents using a three-phase current regulator.

Motor currents, voltages, speed, and torque signals are available at the output of the block.

### Simulation

Start the simulation. You can observe the motor stator current, the rotor speed, the electromagnetic torque, the DC bus voltage and the magnitude of the rotor flux on the first scope. The speed set point and the torque set point are also shown. On the second scope, the dq currents and voltages are displayed. Note that all signals are multiplexed to compare the two models.

At time t = 0 s, the speed set point is 4000 rpm. Observe that the speed follows precisely the acceleration ramp.

At t = 0.5 s, the full load torque is applied to the motor shaft while the motor speed is still ramping.

At t = 0.65 s, the inverter is saturated due to the limited DC bus voltage. You can observe loss of current tracking which decreases the motor torque.

At t = 1.5 s, the speed set point is changed to -4000 rpm.

At t = 2 s, the deceleration ramp reaches the motor speed. The inverter comes back to normal mode.

At t = 2.5 s, the mechanical load passes from 792 N.m to -792 N.m.

At t = 3.45 s, the inverter is saturated due to insufficient DC bus voltage. You can observe loss of current tracking which decreases the motor torque.

Finally, notice that the results of the average-value model are similar to those of the detailed model except that the higher frequency signal components are not represented with the average-value converter.

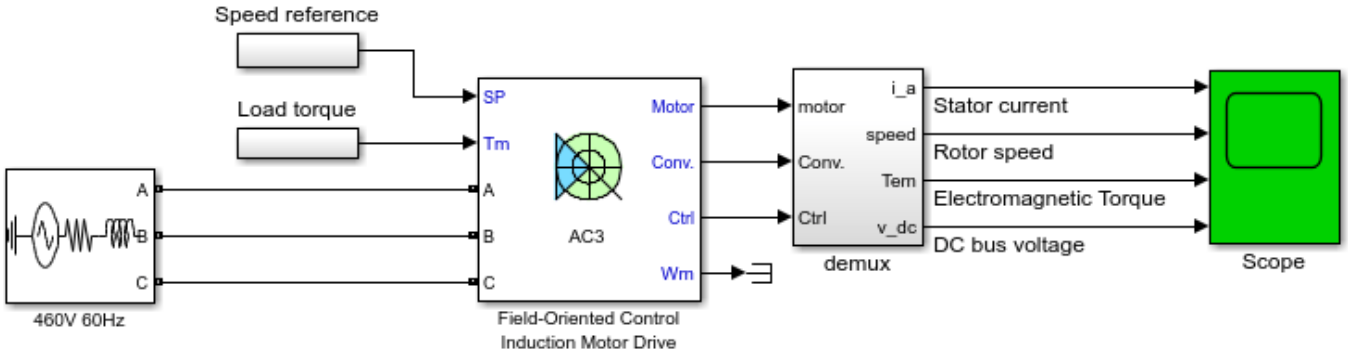
**Notes**

To evaluate the speed gain of the average value model, see ac3_example_simplified and compare the simulation speed with ac3_example.

# AC3 - Sensorless Field-Oriented Control Induction Motor Drive

This example shows the Sensorless Field-Oriented Control Induction Motor Drive during speed regulation.

Souleman Njoya M., Louis-A. Dessaint (Ecole de technologie superieure, Montreal))



### Description

This circuit uses a modified version of the AC3 block of Specialized Power Systems library. It models a sensorless field-oriented control (FOC) induction motor drive with a braking chopper for a 200HP AC motor. The motor speed is estimated from terminal voltages and currents based on the MRAS (Model Referencing Adaptive System) technique [1]. Consequently, the speed sensor (necessary in AC3) is no more required.

The induction motor is fed by a PWM voltage source inverter, which is built using a Universal Bridge Block. The speed control loop uses a PI controller to produce the flux and torque references for the FOC controller. The FOC controller computes the three reference motor line currents corresponding to the flux and torque references and then feeds the motor with these currents using a three-phase current regulator.

Motor current, speed (reference, real and estimated), and torque signals are available at the output of the block.

### Simulation

Start the simulation. You can observe the motor stator current, the rotor speed, the electromagnetic torque and the DC bus voltage on the scope. The speed set point and the torque set point are also shown.

At time t = 0 s, the speed set point is 500 rpm. Observe that the speed follows precisely the acceleration ramp.

At t = 0.5 s, the full load torque is applied to the motor shaft while the motor speed is still ramping to its final value. This forces the electromagnetic torque to increase to the user-defined maximum value (1200 N.m) and then to stabilize at 820 N.m once the speed ramping is completed and the motor has reached 500 rpm.

At t = 1 s, the speed set point is changed to 0 rpm. The speed decreases down to 0 rpm by following precisely the deceleration ramp even though the mechanical load is inverted abruptly, passing from 792 N.m to - 792 N.m, at t = 1.5 s. Shortly after, the motor speed stabilizes at 0 rpm.

Finally, note how well the DC bus voltage is regulated during the whole simulation period.

**Notes**

1) The power system has been discretised with a 2 us time step. The speed controller uses a 100 us sampling time and the vector controller uses a 20us sampling time in order to simulate a microcontroller control device.

2) A simplified version of the model using average-value inverter can be used by selecting 'Average' in the 'Model detail level' menu of the graphical user-interface. The time step can then be increased up to 40 us. This can be done by typing 'Ts = 40e-6' in the workspace and by changing the speed controller sampling time to 120e-6 and the vector controller sampling time to 40e-6. See also ac3_sensorless_simplified model.

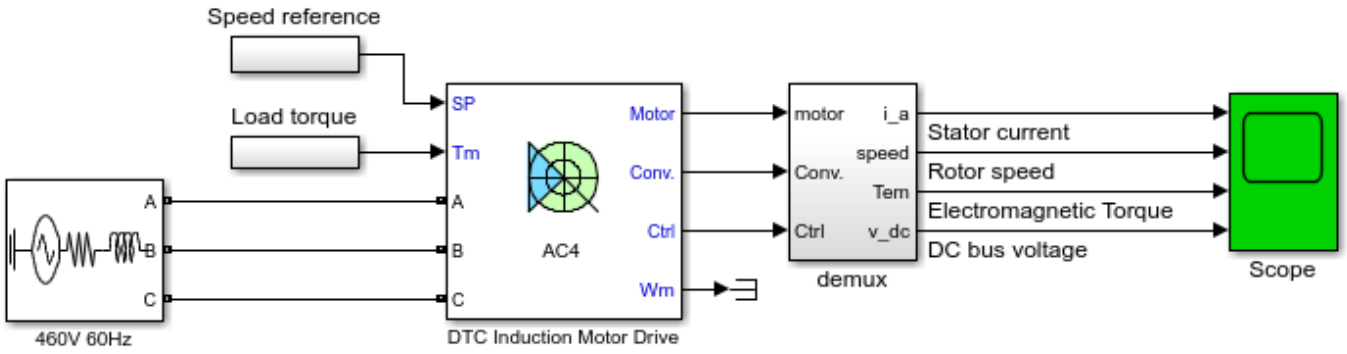
**References**

1. Bose, Bimal K., "Modern Power Electronics And AC Drives", Prentice-Hall, Inc., Upper Saddle River, NJ 07458, 2002.

# AC4 - DTC Induction 200 HP Motor Drive

This example shows the AC4 DTC Induction Motor Drive during speed regulation.

H.Blanchette, L.-A. Dessaint (Ecole de technologie superieure, Montreal)



### Description

This circuit uses the AC4 block of Specialized Power Systems library. It models a direct torque control (DTC) induction motor drive with a braking chopper for a 200HP AC motor.

The induction motor is fed by a PWM voltage source inverter which is built using a Universal Bridge Block. The speed control loop uses a proportional-integral controller to produce the flux and torque references for the DTC block. The DTC block computes the motor torque and flux estimates and compares them to their respective reference. The comparators outputs are then used by an optimal switching table which generates the inverter switching pulses.

Motor current, speed, and torque signals are available at the output of the block.

### Simulation

Start the simulation. You can observe the motor stator current, the rotor speed, the electromagnetic torque and the DC bus voltage on the scope. The speed set point and the torque set point are also shown.

At time t = 0 s, the speed set point is 500 rpm. Observe that the speed follows precisely the acceleration ramp.

At t = 0.5 s, the full load torque is applied to the motor shaft while the motor speed is still ramping to its final value. This forces the electromagnetic torque to increase to the user-defined maximum value (1200 N.m) and then to stabilize at 820 N.m once the speed ramping is completed and the motor has reached 500 rpm.

At t = 1 s, the speed set point is changed to 0 rpm. The speed decreases down to 0 rpm by following precisely the deceleration ramp even though the mechanical load is inverted abruptly, passing from 792 N.m to - 792 N.m, at t = 1.5 s. Shortly after, the motor speed stabilizes at 0 rpm.

Finally, note how well the DC bus voltage is regulated during the whole simulation period.

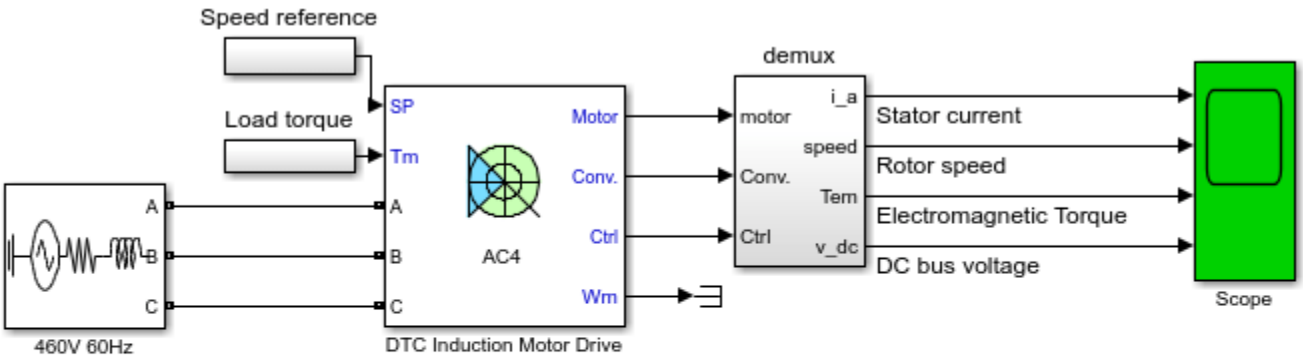
**Notes**

The power system has been discretised with a 2 us time step. The speed controller uses a 140 us sample and the DTC controller uses a 20 us sample time in order to simulate a microcontroller control device.

# AC4 - Space Vector PWM-DTC Induction 200 HP Motor Drive

This example shows a Space Vector PWM DTC Induction Motor Drive during speed regulation.

C.Semaille, O. Tremblay, L.-A. Dessaint (Ecole de technologie superieure, Montreal)



### Description

This circuit uses a modified version of the AC4 block of the Specialized Power Systems electric drives library. It models a direct torque control (DTC) induction motor drive with space vector pulse width modulation. The particularity of this modified version is that the DTC is no longer based on hysteresis regulation that implies switching at variable frequency but on a fixed frequency PMW inverter. As in AC4 it uses a braking chopper for a 200HP AC motor.

The induction motor is fed by a PWM voltage source inverter which is built using a Universal Bridge Block. The speed control loop uses a proportional-integral controller to produce the flux and torque references for the DTC block. The DTC block computes the motor torque and flux estimates and compares them to their respective reference. The torque and flux are then controlled by independent PI regulators that compute a reference voltage vector. The voltage source inverter is then controlled by the space vector modulation method in order to output the desired reference voltage.

Motor current, speed, and torque signals are available at the output of the block.

### Simulation

Start the simulation. You can observe the motor stator current, the rotor speed, the electromagnetic torque and the DC bus voltage on the scope. The speed set point and the torque set point are also shown.

At time t = 0 s, the speed set point is 500 rpm. Observe that the speed follows precisely the acceleration ramp.

At t = 0.5 s, the full load torque is applied to the motor shaft while the motor speed is still ramping to its final value. This forces the electromagnetic torque to increase to the user-defined maximum value (1200 N.m) and then to stabilize at 820 N.m once the speed ramping is completed and the motor has reached 500 rpm.

At t = 1 s, the speed set point is changed to 0 rpm. The speed decreases down to 0 rpm by following precisely the deceleration ramp even though the mechanical load is inverted abruptly, passing from 792 N.m to - 792 N.m, at t = 1.5 s. Shortly after, the motor speed stabilizes at 0 rpm.

Finally, note how well the DC bus voltage is regulated during the whole simulation period.

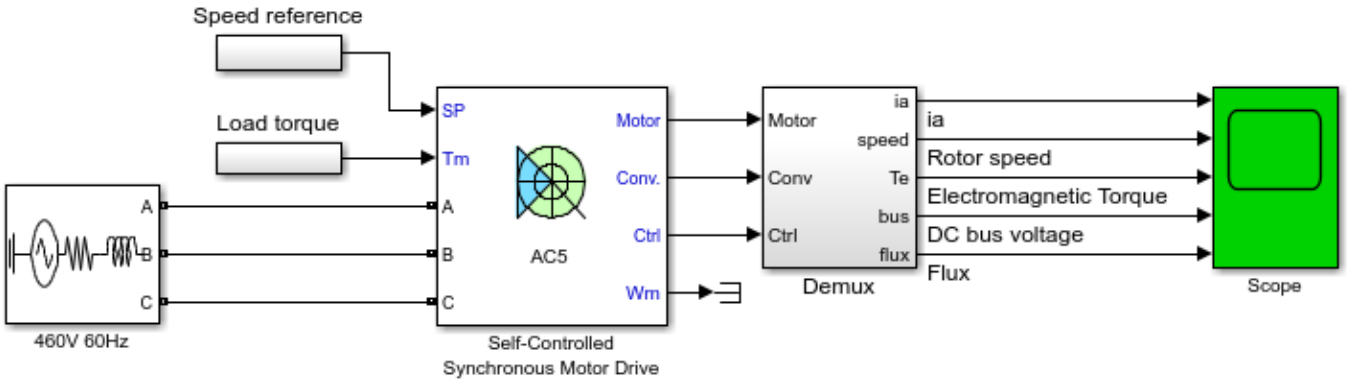
**Notes**

The power system has been discretised with a 2us time step. The speed controller uses a 140 us sample and the DTC controller uses a 20 us sample time in order to simulate a microcontroller control device. The inverter switching frequency is set to 5 kHz.

# AC5 - Self-Controlled Synchronous 200 HP Motor Drive

This example shows the AC5 Self-Controlled Synchronous motor drive during speed regulation.

H.Blanchette, L.-A. Dessaint (Ecole de technologie superieure, Montreal)



### Description

This circuit uses the AC5 block of Specialized Power Systems library. It models a self-controlled synchronous motor drive with active front-end rectifier for a 200HP motor.

The synchronous motor is fed by a PWM voltage source inverter, which is built using a Universal Bridge Block. The speed control loop uses a PI regulator to produce the flux and current references for the vector controller block. The vector controller computes the three reference motor line currents corresponding to the torque reference and feeds the motor with these currents using a three-phase current regulator. The vector controller also computes the flux estimate and compares it with the desired value in order to generate the field excitation voltage.

Since the field flux dynamics of the synchronous machine are relatively slow, it is advisable to establish first the field flux to its nominal value before feeding the stator with three-phase currents. In this example, a high magnetization voltage of 600V is applied to the rotor field during the first 0.2 s of the simulation in order to speed up the rotor field increase. Once the field flux has reached its nominal value of 1.0 weber, the three-phase current regulator associated to the motor stator is switched on.

Motor current, speed, and torque signals are available at the output of the block.

### Simulation

Start the simulation. You can observe the motor stator current, the rotor speed, the electromagnetic torque and the DC bus voltage and the motor magnetic flux on the scope. The speed set point and the torque set point are also shown.

At time t = 1.5 s, the speed set point is 200 rpm. Observe that the speed follows precisely the acceleration ramp and that the stator current amplitude and frequency increase gradually.

At t = 3.0 s, a resistive torque of the nominal value is applied to the motor shaft. Recall that this type of torque tends to decelerate the motor. This explains why the motor speed slightly undershoots. Then, the motor reaches 200 rpm.

At t = 4.0 s, the speed set point is changed to 0 rpm. This forces the motor to produce a lower electric torque. The speed decreases down to 0 rpm following precisely the deceleration ramp. At t = 6.0 s, the speed setpoint reaches 0 rpm.

At t = 5.5 s, the sign of the load torque applied to the motor shaft is reversed. Observe the corresponding small overshoot in the motor speed and the stabilization of the electric torque at its nominal value.

Finally, note how well the DC bus voltage is regulated during the whole simulation period.

**Notes**

1) The power system has been discretised with a 2us time step. The speed controller uses a 140 us sample and the vector controller uses a 20 us sample time in order to simulate a microcontroller control device.

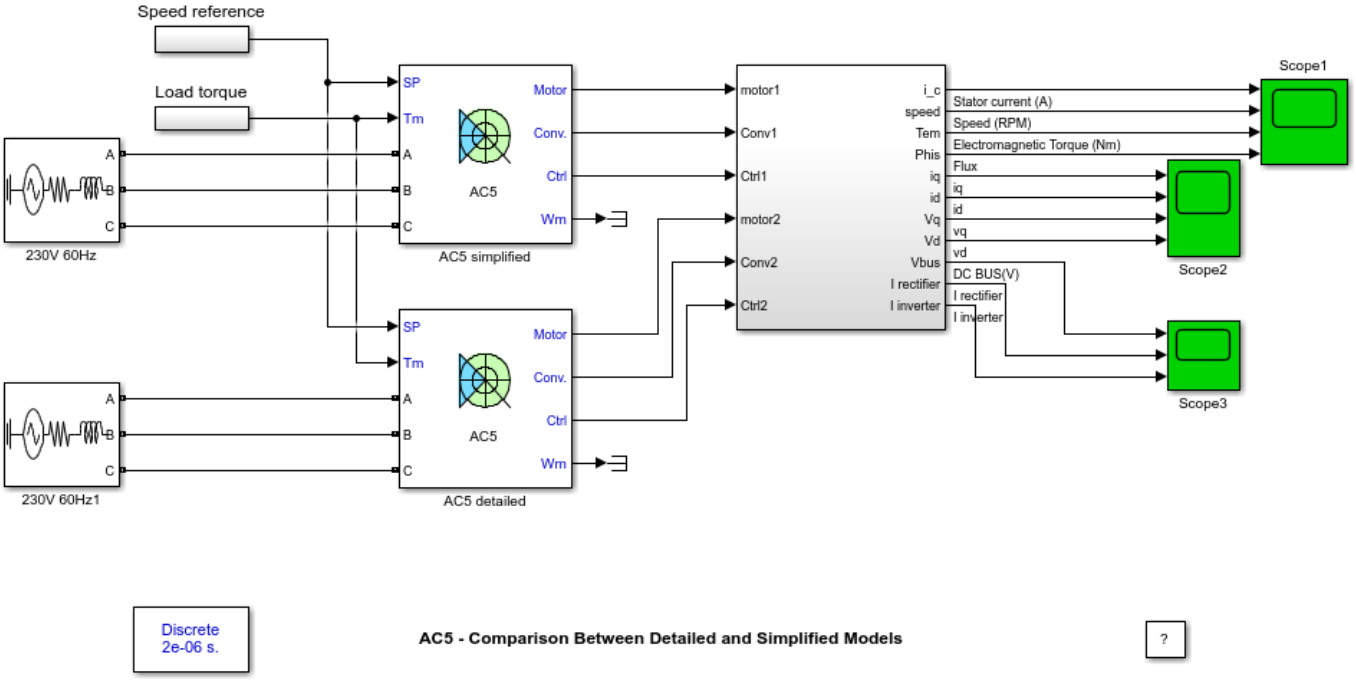
2) The torque sign convention of the synchronous machine is different from the one of the asynchronous and PM synchronous machines. That is, the synchronous machine is in the motor operation mode when the electric torque is negative and in the generator operation mode when the electric torque is positive.

3) A simplified version of the model using average-value inverter and rectifier can be used by selecting 'Average' in the 'Model detail level' menu of the graphical user-interface. The time step can then be increased up to 50 us. This can be done by typing 'Ts = 50e-6' in the workspace and by changing the speed controller sampling time to 150e-6, the DC bus controller sampling time to 50e-6 and the vector controller sampling time to 50e-6. See also ac5_example_simplified model.

# AC5 - Comparison Between Detailed and Simplified Models

This example shows the AC5 detailed model and the AC5 average model during speed regulation. The comparison is performed for normal condition and for inverter-rectifier saturation condition.

O. Tremblay, L.-A. Dessaint (Ecole de technologie superieure, Montreal)




## Description

This circuit uses two AC5 blocks of Specialized Power Systems electric drives library. The AC5 block models a self-controlled synchronous motor drive with active front-end rectifier for a 200 HP motor.

The first AC5 block is set to average value model and the second AC5 block is set to detailed model.

For the detailed model, the synchronous motor is fed by a PWM voltage source inverter, which is built using a Universal Bridge Block. The average value model uses ideal voltages and currents sources to feed the synchronous motor. The speed control loop uses a PI regulator to produce the flux and current references for the vector controller block. The vector controller computes the three reference motor line currents corresponding to the torque reference and feeds the motor with these currents using a three-phase current regulator. The vector controller also computes the flux estimate and compares it with the desired value in order to generate the field excitation voltage.

Since the field flux dynamics of the synchronous machine are relatively slow, it is advisable to establish first the field flux to its nominal value before feeding the stator with three-phase currents. In this example, a high magnetization voltage of 600V is applied to the rotor field during the first 0.2 s of the simulation in order to speed up the rotor field increase. Once the field flux has reached its nominal value of 1.0 weber, the three-phase current regulator associated to the motor stator is switched on.

Motor currents, voltages, speed, and torque signals are available at the output of the block.

### Simulation

Start the simulation. You can observe the motor stator current, the rotor speed, the electromagnetic torque and the magnitude of the rotor flux on the first scope. The speed set point and the torque set point are also shown. On the second scope, the dq currents and voltages are displayed. Finally, on the third scope, the DC bus voltage, the rectifier current and the inverter current are shown. Note that all signals are multiplexed to compare the two models.

At time t = 1.5 s, the speed set point is 1750 rpm. Observe that the speed follows precisely the acceleration ramp and that the stator current amplitude and frequency increase gradually.

At t = 3.0 s, a resistive torque of -792 N.m is applied to the motor shaft while the motor speed is still ramping. Observe the rectifier saturation on the third scope.

At t = 3.5 s, the inverter is saturated due to the limited DC bus voltage. You can observe loss of current tracking which decreases the motor torque.

At t = 5 s, the speed set point is changed to -1750 rpm.

At t = 6 s, the deceleration ramp reaches the motor speed. The inverter and the rectifier come back to normal mode.

At t = 6.5 s, the mechanical load passes from -792 Nm to 792 Nm.

At t = 9.5 s, observe on the third scope the rectifier saturation.

At t = 10.5 s, the inverter is saturated due to the limited DC bus voltage. You can observe loss of current tracking which decreases the motor torque.

Finally, note how the simplified model reacts well compared to the detailed model.

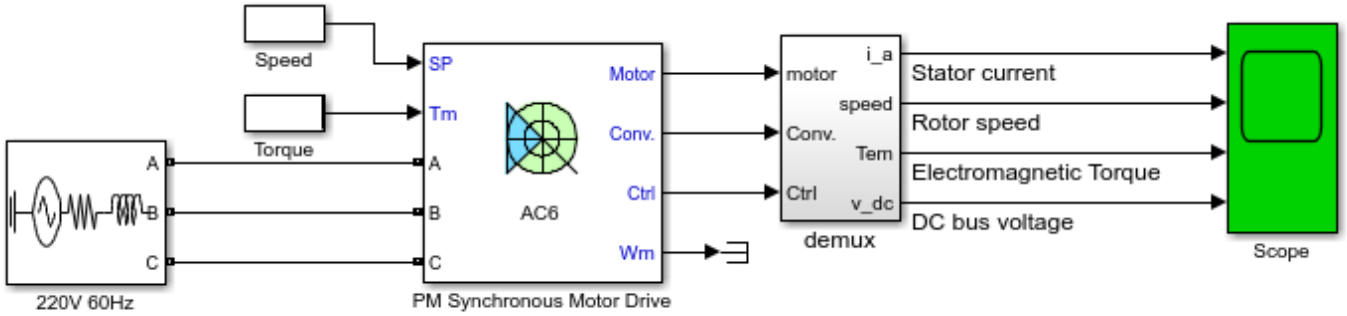
### Notes

To evaluate the speed gain of the average value model, see ac5_example_simplified and compare the simulation speed with ac5_example.

# AC6 - PM Synchronous 3HP Motor Drive

This example shows the AC6 PM Synchronous Motor Drive during speed regulation.

H.Blanchette, L.-A. Dessaint (Ecole de technologie superieure, Montreal)



### Description

This circuit uses the AC6 block of Specialized Power Systems library. It models a permanent magnet (PM) synchronous motor drive with a braking chopper for a 3HP motor.

The PM synchronous motor is fed by a PWM voltage source inverter, which is built using a Universal Bridge Block. The speed control loop uses a PI regulator to produce the flux and torque references for the vector control block. The vector control block computes the three reference motor line currents corresponding to the flux and torque references and then feeds the motor with these currents using a three-phase current regulator.

Motor current, speed, and torque signals are available at the output of the block.

### Simulation

Start the simulation. You can observe the motor stator current, the rotor speed, the electromagnetic torque and the DC bus voltage on the scope. The speed set point and the torque set point are also shown.

At time t = 0 s, the speed set point is 300 rpm. Observe that the speed follows precisely the acceleration ramp.

At t = 0.5 s, the full load torque is applied to the motor. You can observe a small disturbance in the motor speed, which stabilizes very quickly.

At t = 1 s, the speed set point is changed to 0 rpm. The speed decreases down to 0 rpm following precisely the deceleration ramp.

At t = 1.5 s., the mechanical load passes from 11 Nm to -11 Nm. The motor speed stabilizes very quickly after a small overshoot.

Finally, note how well the DC bus voltage is regulated during the whole simulation period.

**Notes**

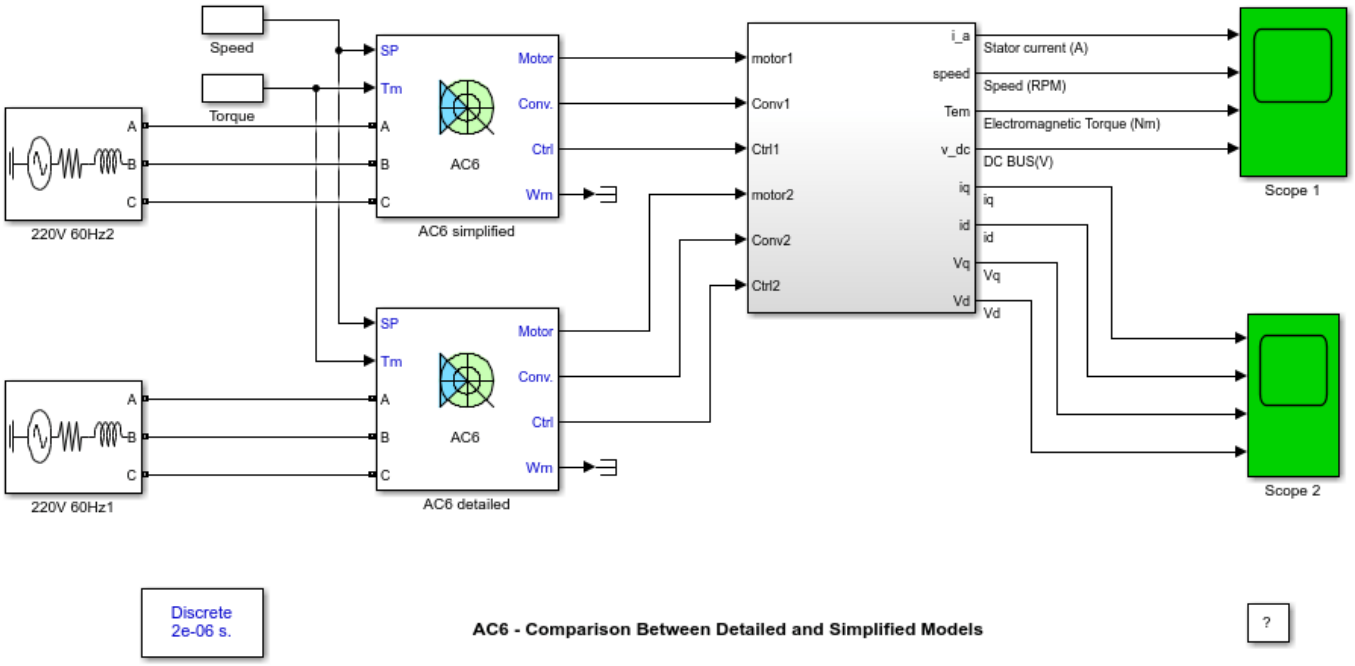
1) The power system has been discretised with a 2 us time step. The speed controller uses a 140 us sample and the vector controller uses a 20 us sample time in order to simulate a microcontroller control device.

2) A simplified version of the model using average-value inverter can be used by selecting 'Average' in the 'Model detail level' menu of the graphical user-interface. The time step can then be increased up to 75 us. This can be done by typing 'Ts = 75e-6' in the workspace, by setting Vector control sample time to 75e-6 and by setting Speed controller sample time to 150e-6 in the case of this example. See also ac6_example_simplified model.

# AC6 - Comparison Between Detailed and Simplified Models

This example shows the AC6 detailed model and the AC6 average model during speed regulation. The comparison is performed for normal condition and for inverter saturation condition.

O. Tremblay, L.-A. Dessaint (Ecole de technologie superieure, Montreal)




### Description

This circuit uses two AC6 blocks of Specialized Power Systems library. It models a permanent magnet (PM) synchronous motor drive with a braking chopper for a 3HP motor. The first block is set to average value model and the second is set to detailed model.

For the detailed model, the PM synchronous motor is fed by a PWM voltage source inverter, which is built using a Universal Bridge Block. The average value model uses ideal voltages and currents sources to feed the PM synchronous motor. The speed control loop uses a PI regulator to produce the flux and torque references for the vector control block. The vector control block computes the three reference motor line currents corresponding to the flux and torque references and then feeds the motor with these currents using a three-phase current regulator.

Motor currents, voltages, speed, and torque signals are available at the output of the block.

### Simulation

Start the simulation. You can observe the motor stator current, the rotor speed, the electromagnetic torque and the DC bus voltage on the first scope. The speed set point and the torque set point are also shown. On the second scope, the dq currents and voltages are displayed. Note that all signals are multiplexed to compare the two models.

At time t = 0 s, the speed set point is 2400 rpm and the full load torque is applied to the motor. Observe that the speed follows precisely the acceleration ramp.

At t = 0.7 s, the inverter is saturated due to insufficient inverter voltage. You can observe loss of current tracking which decreases the motor torque.

At t = 1.2 s, the speed set point is changed to -2400 rpm.

At t = 1.5 s., the deceleration ramp reaches the motor speed. The inverter comes back to normal mode.

At t = 2 s., the mechanical load passes from 11 Nm to -11 Nm.

At t = 3.15 s., the inverter is saturated due to insufficient inverter voltage. You can observe loss of current tracking which decreases the motor torque.

Finally, note how the simplified model reacts well compared to the detailed model.

**Notes**

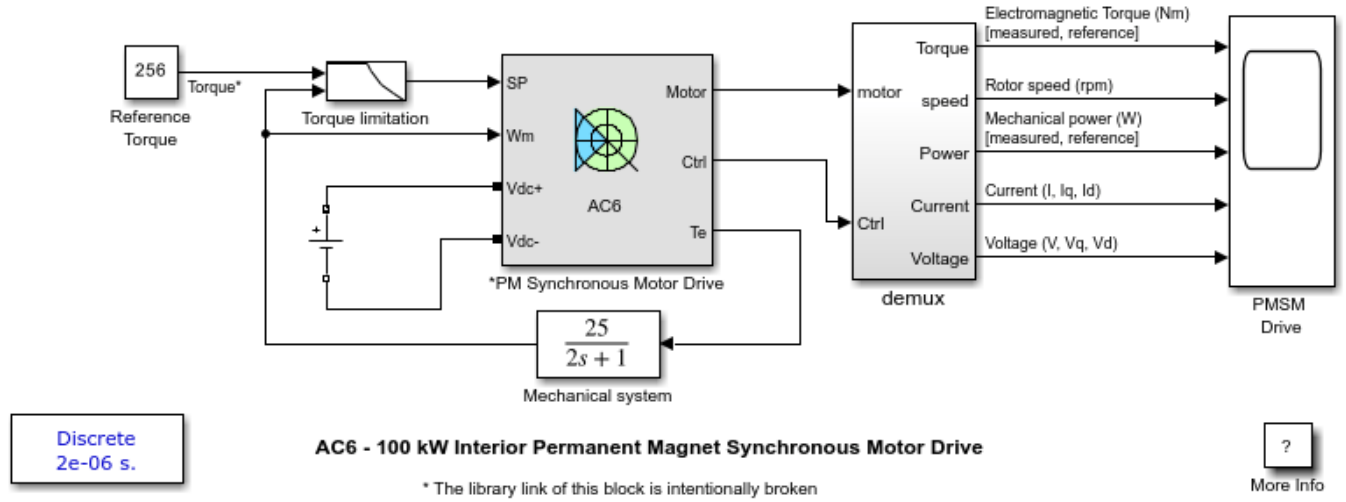
To evaluate the speed gain of the average value model, see ac6_example_simplified and compare the simulation speed with ac6_example.

# AC6 - 100 kW Interior Permanent Magnet Synchronous Motor Drive

This example shows Vector Control for an interior Permanent Magnet Synchronous Motor (PMSM) during torque regulation.

In the ac6_example model, it was assumed that the PMSM had its permanent magnets mounted on the surface of the rotor. This type of PMSM has therefore a uniform air gap and no saliency, hence Ld = Lq. It is assumed that the PMSM has an interior permanent magnets rotor. The impact of the buried-magnet configuration is rotor saliency that makes Lq > Ld and introduces a reluctance torque term into the PMSM torque equation. To take advantage of the reluctance torque, the Id current component is no longer set to zero has it is for the PMSM with surface mounted permanent magnets.

Olivier Tremblay, Louis-A. Dessaint (Ecole de Technologie Superieure, Montreal).




The 'Ts' parameter used in this model
is set to 2e-6 by the Model Properties Callbacks

### Description

This circuit uses a modified version of the AC6 block of the Specialized Power Systems electric drives library. It models a flux weakening vector control for a 100 kW, 12500 rpm, salient pole PMSM powered by a 288 Vdc source. The mechanical system is represented externally. That's why the input of the motor is the speed and the output is the electromagnetic torque.

The **PM Synchronous Motor Drive** is composed of four main parts: The electrical motor, the Three-phase Inverter, the VECT controller and the Speed Controller.

- The electrical motor is a 288 Vdc, 100 kW PMSM. This motor has 8 pole and the magnets are buried (salient rotor's type).
- The Three-phase Inverter is a voltage source inverter, controlled by PWM. This block is built using the Universal Bridge Block.
- The VECT controller block computes the three reference motor line currents corresponding to the flux and torque references and then generates a corresponding PWM using a three-phase current

regulator. When the nominal flux is required, an optimal control is used in order to minimise the line current amplitude for the required torque. When a flux weakening is needed, the amplitude and the phase of the current are changed to extend the torque-speed operating range.

- The Speed Controller is used in torque regulation mode. The normalized flux value is computed with the speed of the machine in order to perform a flux weakening control.

The Torque limitation block is used to prevent the limitation due to the torque-speed characteristic of this motor for a 288 Vdc source. When the internal machine's voltage reaches the inverter voltage (because the desired torque is too high for the motor's speed), the inverter becomes in saturation mode (the desired current cannot flow anymore into motor). After this point, there will be a loss of current tracking which will decrease the motor current. This block is used to reduce the reference torque as a function of the motor's speed and the torque-speed characteristic in order to never operate in inverter saturation mode.

Motor torque, speed, power, currents and voltages signals are available at the output of the block.

### Simulation

Start the simulation. You can observe the motor torque (electromagnetic and reference), the rotor speed, the mechanical power (electromagnetic and reference), the stator currents (magnitude, Iq and Id), and the stator voltages (magnitude, Vq and Vd)

- At t = 0 s, the torque set point is set to 256 Nm (the nominal torque of the motor). The electromagnetic torque reaches rapidly the reference.
- At t = 0.104 s, the rotor speed exceeds the nominal speed of 3000 rpm. Hence, a flux weakening is performed in order to limit the back electromotive force (BEMF) of the motor; therefore the Id current component is increased (negatively). Also the reference torque is limited (due to the torque-speed characteristic of the motor) to prevent inverter saturation, causing a decrease in the Iq current component. Note that the magnitude of the current is constant; only the angle changes.

Now change the **Reference Torque** to 100 Nm and observe the results:

- At t = 0 s, the torque set point is set to 100 Nm. The current amplitude is optimal for this torque.
- At t = 0.28 s, the rotor speed exceeds the nominal speed of 3000 rpm. Hence, a flux weakening is performed in order to limit the back electromotive force (BEMF) of the motor; therefore the Id current component is increased (negatively).
- At t = 1.06 s, the reference torque is limited (due to the torque-speed characteristic of the motor) to prevent inverter saturation, causing a decrease in the Iq current component. The magnitude of the current is maintained at a constant value, but the phase of current changes.

Note that the electromagnetic torque follows precisely the reference torque even in the flux weakening region.

### Notes

1) The power system has been discretised with a 2 us time step. The speed controller uses a 140 us sample and the vector controller uses a 20 us sample time in order to simulate a micro controller control device.

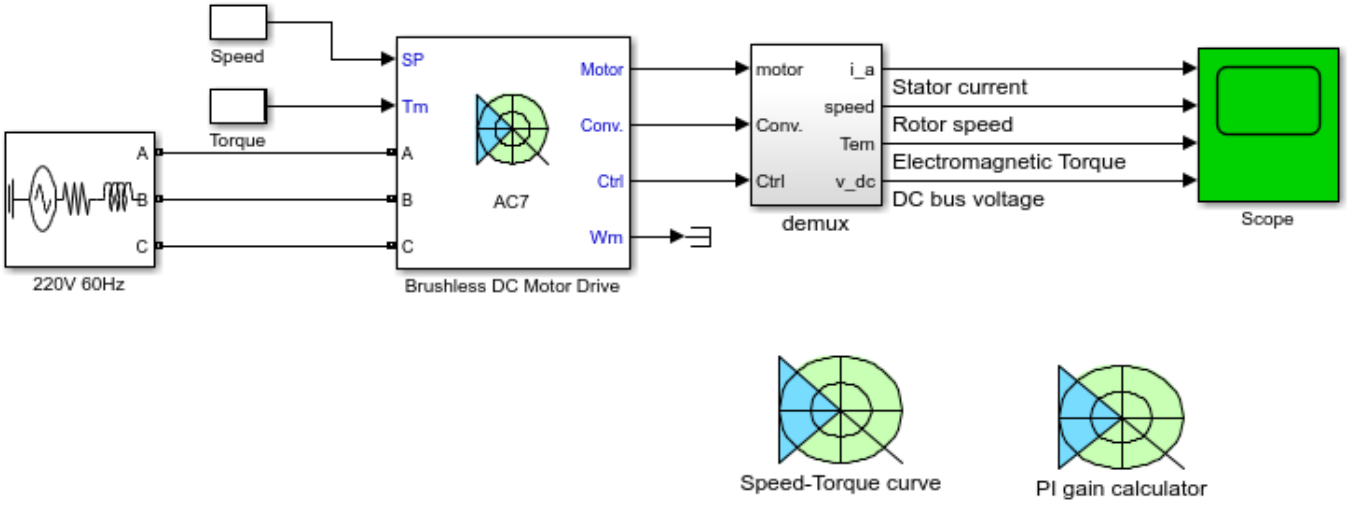
### See Also

- AC6 - PM Synchronous 3HP Motor Drive

# AC7 - Brushless DC Motor Drive During Speed Regulation

This example shows the AC7 Brushless DC Motor Drive during speed regulation.

O.Tremblay, L.-A. Dessaint (Ecole de technologie superieure, Montreal)




### Description

This circuit uses the AC7 block of Specialized Power Systems library. It models a brushless DC motor drive with a braking chopper for a 3HP motor.

The permanent magnet synchronous motor (with trapezoidal back-EMF) is fed by a PWM voltage source inverter, which is built using a Universal Bridge Block. The speed control loop uses a PI regulator to produce the torque reference for the current control block. The current control block computes the three reference motor line currents, in phase with the back electromotive forces, corresponding to the torque reference and then feeds the motor with these currents using a three-phase current regulator.

Motor current, speed, and torque signals are available at the output of the block.

### Simulation

Start the simulation. You can observe the motor stator current, the rotor speed, the electromagnetic torque and the DC bus voltage on the scope. The speed set point and the torque set point are also shown.

At time t = 0 s, the speed set point is 300 rpm. Observe that the speed follows precisely the acceleration ramp.

At t = 0.5 s, the full load torque is applied to the motor. You can observe a small disturbance in the motor speed, which stabilizes very quickly.

At t = 1 s, the speed set point is changed to 0 rpm. The speed decreases down to 0 rpm following precisely the deceleration ramp.

At t = 1.5 s., the mechanical load passes from 11 Nm to -11 Nm. The motor speed stabilizes very quickly after a small overshoot.

Finally, note how well the DC bus voltage is regulated during the whole simulation period.

**Notes**

1) The power system has been discretised with a 2 us time step. The speed controller uses a 140 us sample and the current controller uses a 20 us sample time in order to simulate a microcontroller control device.

2) To calculate automatically the speed regulator gains, double-click on the PI gain calculator icon. Enter the mechanical parameters of the machine and click on the "Enter specification" checkbox. Enter the desired specifications and click on the "Calculate Kp and Ki" checkbox.

If you don't know the specifications, uncheck the "Enter specification" checkbox and enter the electrical parameters of the machine. Click on the "Machine natural frequency" checkbox to obtain the open-loop machine response. After, click on the "Enter specification" checkbox to enter a response time faster than the open loop response time and click on the "Calculate Kp and Ki" checkbox.

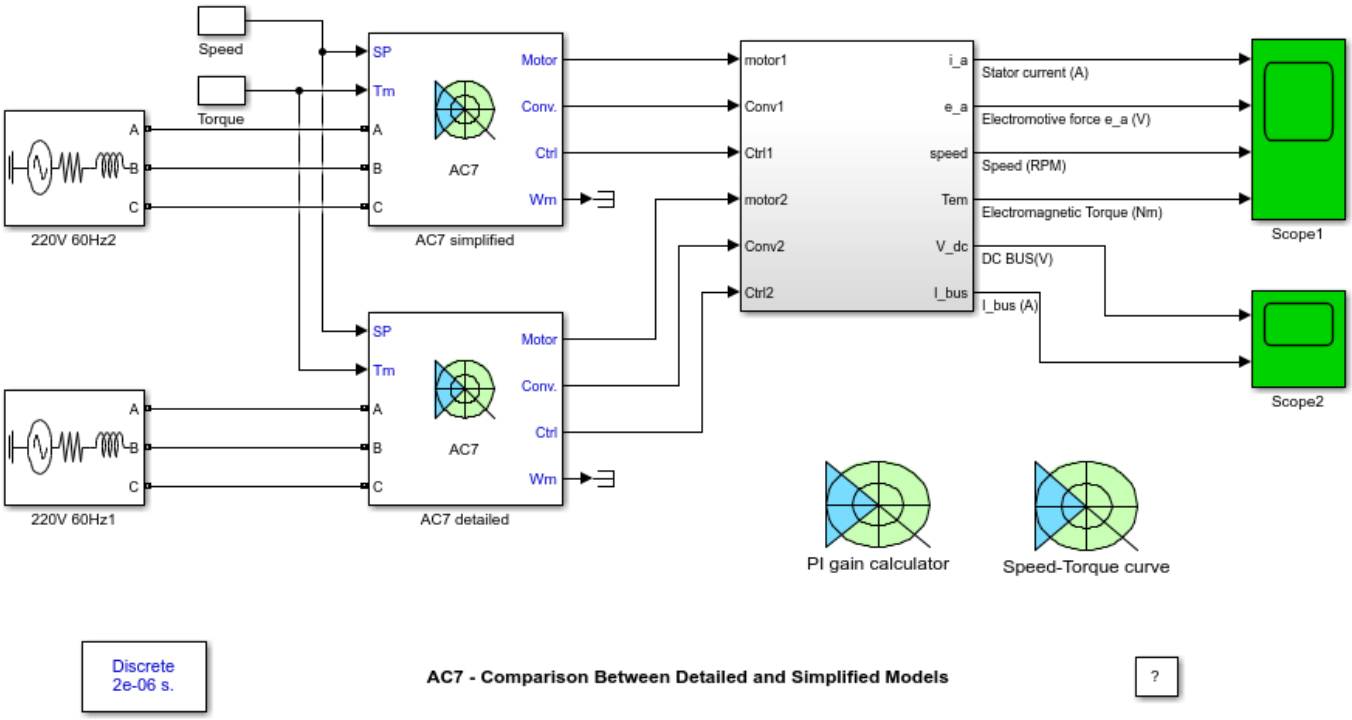
3) To obtain the operation regions of the drive system, double-click on the Speed-Torque curve icon. Enter the machine parameters, the DC bus voltage, the output torque limit and the acceleration speed ramp and click on the "Plot Speed-Torque curve" checkbox.

4) A simplified version of the model using average-value inverter can be used by selecting 'Average' in the 'Model detail level' menu of the graphical user-interface. The time step can then be increased up to 40 us. This can be done by typing 'Ts = 40e-6' in the workspace, by setting current control sample time to 40e-6 and by setting Speed controller sample time to 120e-6 in the case of this example. See also ac7_example_simplified model.

# AC7 - Comparison Between Detailed and Simplified Models

This example shows the AC7 detailed model and the AC7 average model during speed regulation. The comparison is performed for normal condition and for inverter saturation condition.

O.Tremblay, L.-A. Dessaint (Ecole de technologie superieure, Montreal)




### Description

This circuit uses two AC7 blocks from the Specialized Power Systems library. It models a brushless DC motor drive with a braking chopper for a 3HP motor. The first block is set to average value model and the second is set to detailed model.

For the detailed model, the permanent magnet (PM) synchronous motor (with trapezoidal back-EMF) is fed by a PWM voltage source inverter, which is built using a Universal Bridge Block. The average value model uses ideal voltages and currents sources to feed the PM synchronous motor. The speed control loop uses a PI regulator to produce the torque reference for the current control block. The current control block computes the three reference motor line currents, in phase with the back electromotive forces, corresponding to the torque reference and then feeds the motor with these currents using a three-phase current regulator.

Motor currents, voltages, speed, and torque signals are available at the output of the block.

### Simulation

Start the simulation. You can observe the motor stator current, the back EMF, the rotor speed and the electromagnetic torque on the first scope. The speed set point and the torque set point are also

shown. On the second scope, the DC bus voltage and the bus current are displayed. Note that all signals are multiplexed to compare the two models.

At time t = 0 s, the speed set point is 2400 rpm and the full load torque is applied to the motor. Observe that the speed follows precisely the acceleration ramp.

At t = 0.6 s, the inverter is saturated due to insufficient inverter voltage. You can observe loss of current tracking which decreases the motor torque. This saturation point occurs when the speed is about 1200 rpm. That's the same point as predicted by the operation region during acceleration (when load torque is 11 Nm) given by the Speed-Torque curve tool (double-click on the 'Speed-Torque curve' tool to check the operating regions).

At t = 1.2 s, the speed set point is changed to -2400 rpm.

At t = 1.6 s., the deceleration ramp reaches the motor speed. The inverter comes back to normal mode.

At t = 2 s., the mechanical load passes from 11 Nm to -11 Nm.

At t = 3 s., the inverter is saturated due to insufficient inverter voltage. You can observe loss of current tracking which decreases the motor torque.

Finally, note how the simplified model reacts well compared to the detailed model.

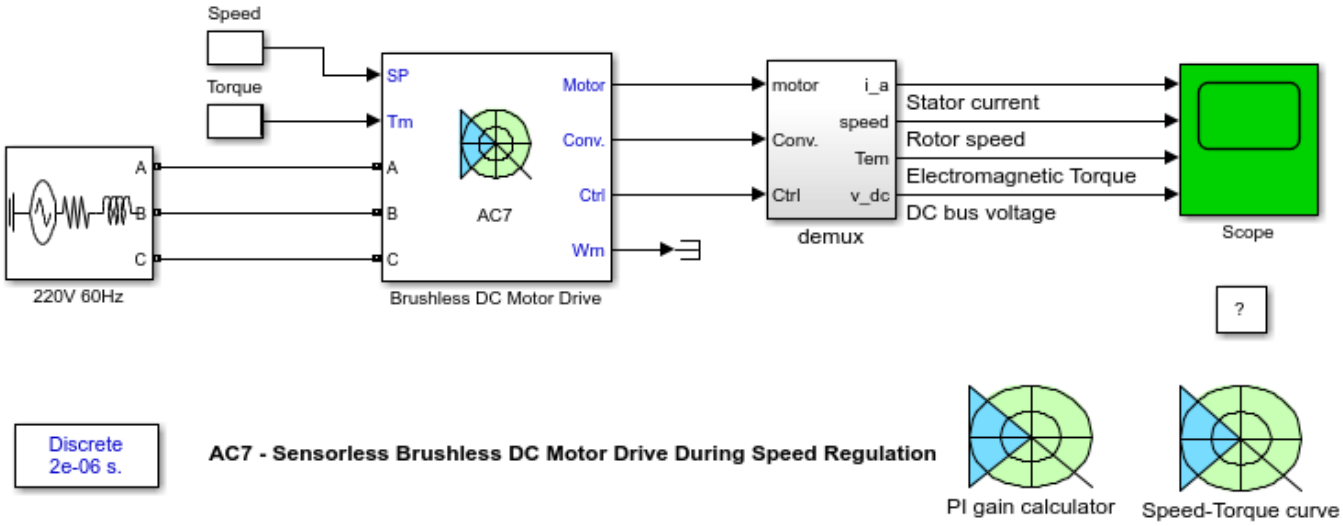
**Notes**

To evaluate the speed gain of the average value model, see ac7_example_simplified and compare the simulation speed with ac7_example.

# AC7 - Sensorless Brushless DC Motor Drive During Speed Regulation

This example shows the AC7 Sensorless Brushless DC Motor Drive during speed regulation.

Souleman Njoya M., Louis-A. Dessaint (Ecole de technologie superieure, Montreal))




### Description

This circuit uses a modified version of the AC7 block of Specialized Power Systems library. It models a sensorless brushless DC motor drive with a braking chopper for a 3HP motor. The AC7 which requires speed and hall sensors is made sensorless. The motor speed and position are estimated from terminal voltages and currents using a back-emf observer [1]. The commutations signals (equivalent to hall effect signals) are generated from the rotor position every 60 electrical degrees.

The permanent magnet synchronous motor (with trapezoidal back-EMF) is fed by a PWM voltage source inverter, which is built using a Universal Bridge Block. The speed control loop uses a PI regulator to produce the torque reference for the current control block. The current control block computes the three reference motor line currents, in phase with the back electromotive forces, corresponding to the torque reference and then feeds the motor with these currents using a three-phase current regulator.

Motor current, speed (reference, real and estimated), and torque signals are available at the output of the block.

### Simulation

Start the simulation. You can observe the motor stator current, the rotor speed, the electromagnetic torque and the DC bus voltage on the scope. The speed set point and the torque set point are also shown.

At time t = 0 s, the speed set point is 300 rpm. Observe that the speed follows precisely the acceleration ramp.

At t = 0.5 s, the full load torque is applied to the motor. You can observe a small disturbance in the motor speed, which stabilizes very quickly.

At t = 1 s, the speed set point is changed to 0 rpm. The speed decreases down to 0 rpm following precisely the deceleration ramp.

### Notes

1) The power system has been discretised with a 2 us time step. The speed controller uses a 140 us sample and the current controller uses a 20 us sample time in order to simulate a microcontroller control device.

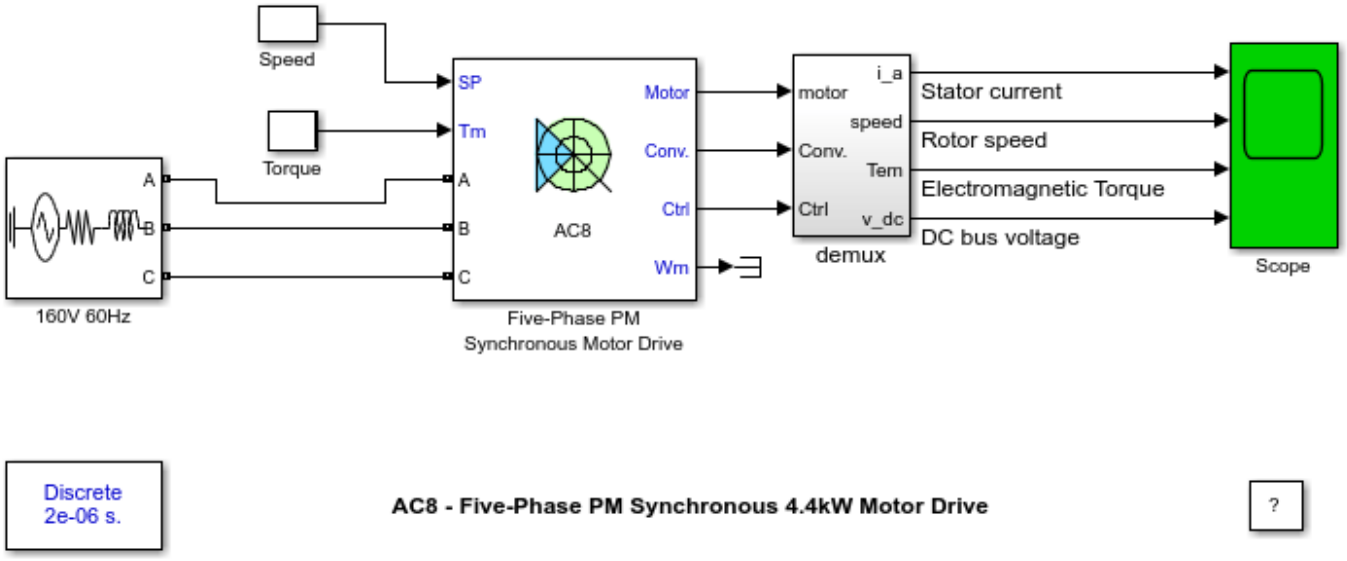
### References

1. TS Kim, BG Park, DM Lee, JS Ryu, DS Hyun, "A New Approach to Sensorless Control Method for Brushless DC Motors", International Journal of Control, Automation, and Systems, vol. 6, no. 4, pp. 477-487, August 2008.

# AC8 - PM Synchronous 3HP Motor Drive

This example shows the AC8 Five-Phase PM Synchronous Motor Drive during speed regulation.

J-F. Doyon, L.-A. Dessaint (Ecole de technologie superieure, Montreal)




### Description

This circuit uses the AC8 block of Specialized Power Systems library. It models a five-phase permanent magnet (PM) synchronous motor drive with a braking chopper for a 4.4kW five-phase motor.

The PM synchronous motor is fed by a PWM voltage source inverter, which is built using a Universal Bridge Block. The speed control loop uses a PI regulator to produce the flux and torque references for the vector control block. The vector control block computes the five reference motor line currents corresponding to the flux and torque references and then feeds the motor with these currents using a five-phase current regulator.

Motor current, speed, and torque signals are available at the output of the block.

### Simulation

Start the simulation. You can observe the motor stator current, the rotor speed, the electromagnetic torque and the DC bus voltage on the scope. The speed set point and the torque set point are also shown.

At time t = 0 s, the speed set point is 300 rpm. Observe that the speed follows precisely the acceleration ramp.

At t = 0.5 s, the full load torque is applied to the motor. You can observe a small disturbance in the motor speed, which stabilizes very quickly.

At t = 1 s, the speed set point is changed to 0 rpm. The speed decreases down to 0 rpm following precisely the deceleration ramp.

At t = 1.5 s., the mechanical load passes from 11 Nm to -11 Nm. The motor speed stabilizes very quickly after a small overshoot.

Finally, note how well the DC bus voltage is regulated during the whole simulation period.

**Notes**

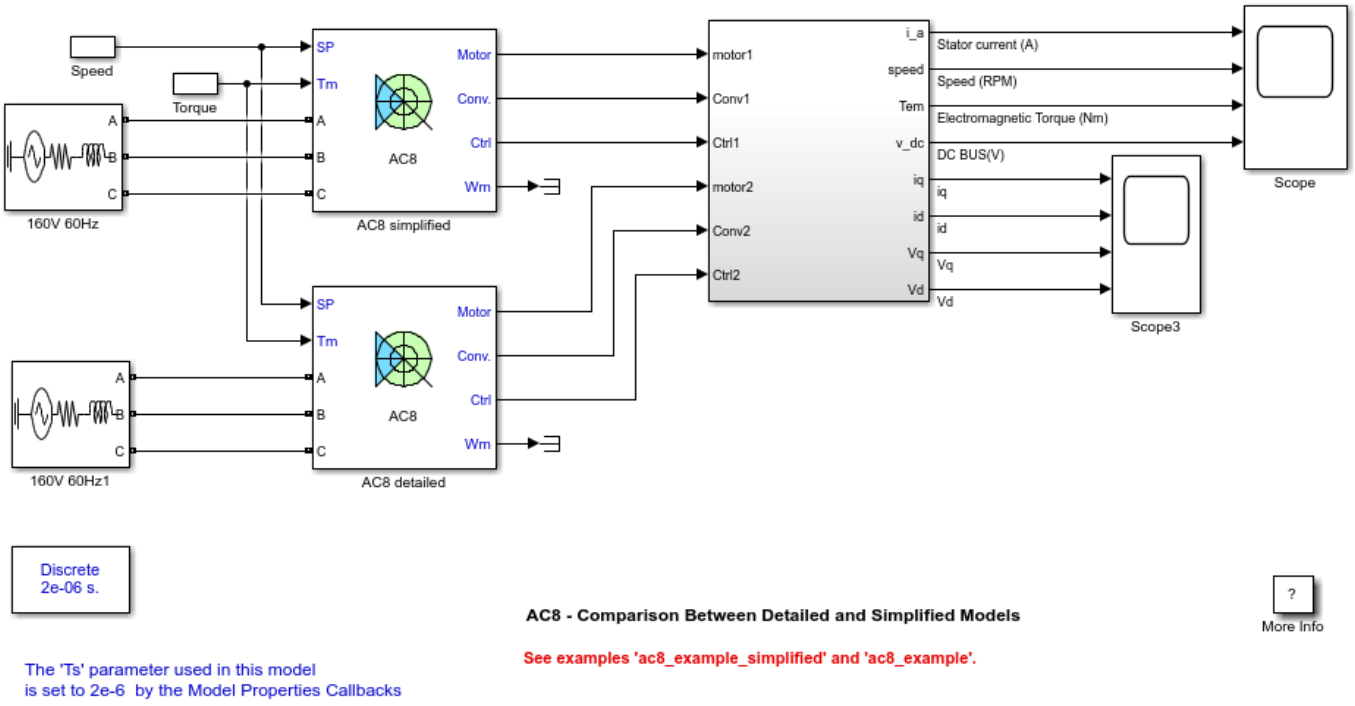
1) The power system has been discretized with a 2 us time step. The speed controller uses a 80 us sample and the vector controller uses a 8 us sample time in order to simulate a micro-controller control device.

2) A simplified version of the model using average-value inverter can be used by selecting 'Average' in the 'Model detail level' menu of the graphical user-interface. The time step can then be increased up to 35 us. This can be done by typing 'Ts = 35e-6' in the workspace, by setting Vector control sample time to 35e-6 and by setting Speed controller sample time to 70e-6 in the case of this example. See also ac8_example_simplified model.

# AC8 - Comparison Between Detailed and Simplified Models

This example shows the AC8 detailed model and the AC8 average model during speed regulation.

J-F. Doyon, L.-A. Dessaint (Ecole de technologie superieure, Montreal)




### Description

This circuit uses two AC8 blocks of Specialized Power Systems library. It models a five-phase permanent magnet (PM) synchronous motor drive with a braking chopper for a 4.4kW motor. The first block is set to average value model and the second is set to detailed model.

For the detailed model, the five-phase PM synchronous motor is fed by a PWM voltage source inverter, which is built using a Universal Bridge Block. The average value model uses ideal voltages and currents sources to feed the PM synchronous motor. The speed control loop uses a PI regulator to produce the flux and torque references for the vector control block. The vector control block computes the five reference motor line currents corresponding to the flux and torque references and then feeds the motor with these currents using a five-phase current regulator.

Motor currents, voltages, speed, and torque signals are available at the output of the block.

### Simulation

Start the simulation. You can observe the motor stator current, the rotor speed, the electromagnetic torque and the DC bus voltage on the first scope. The speed set point and the torque set point are also shown. On the second scope, the dq currents and voltages are displayed. Note that all signals are multiplexed to compare the two models.

At time t = 0 s, the speed set point is 900 rpm and a 25 Nm load torque is applied to the motor. Observe that the speed follows precisely the acceleration ramp.

At t = 0.45 s, the speed set point is reached by both drives.

At t = 1.2 s, the speed set point is changed to -900 rpm.

At t = 2 s., the mechanical load passes from 25 Nm to -25 Nm. You can observe a small disturbance in the motor speed, which stabilizes very quickly.

At t = 2.1 s., the speed set point is reached by both drives.

Finally, note how the simplified model reacts well compared to the detailed model.

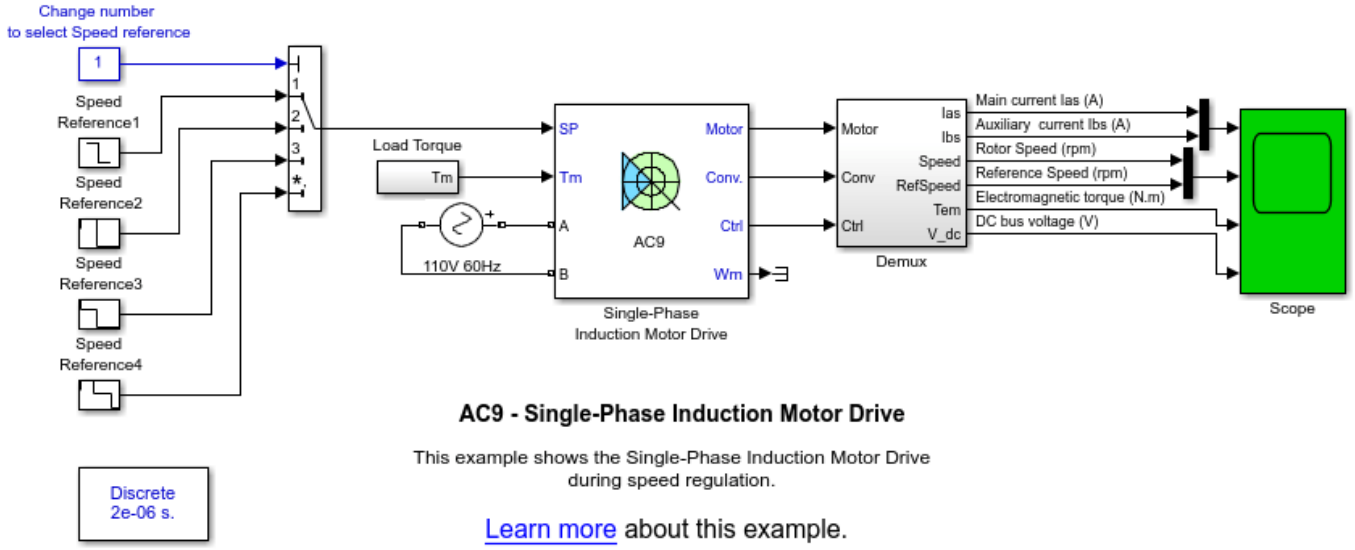
**Notes**

To evaluate the speed gain of the average value model, see ac8_example_simplified and compare the simulation speed with ac8_example.

# AC9 - Single-Phase Induction Motor Drive

This example shows the Single-Phase Induction Motor Drive during speed regulation.

E. Vilchez, L.-A. Dessaint (Ecole de technologie superieure, Montreal)




### Description

This circuit uses the AC9 block of Specialized Power Systems library. It models a vector-controlled single-phase machine drive for a 1/4 HP, 110 V, 60Hz single phase AC motor with its auxiliary and main windings accessible (Main & auxiliary windings operation mode).

The single-phase induction motor is fed by a two-leg inverter, which is built using a Universal Bridge Block. The speed control loop uses a PI controller to produce the flux and torque references for the block controller. In this, two vector control strategies are implemented: Field oriented control (FOC) and direct torque control (DTC) and the modulation is hysteresis-based.

Motor current, speed, and torque signals are available at the output of the block.

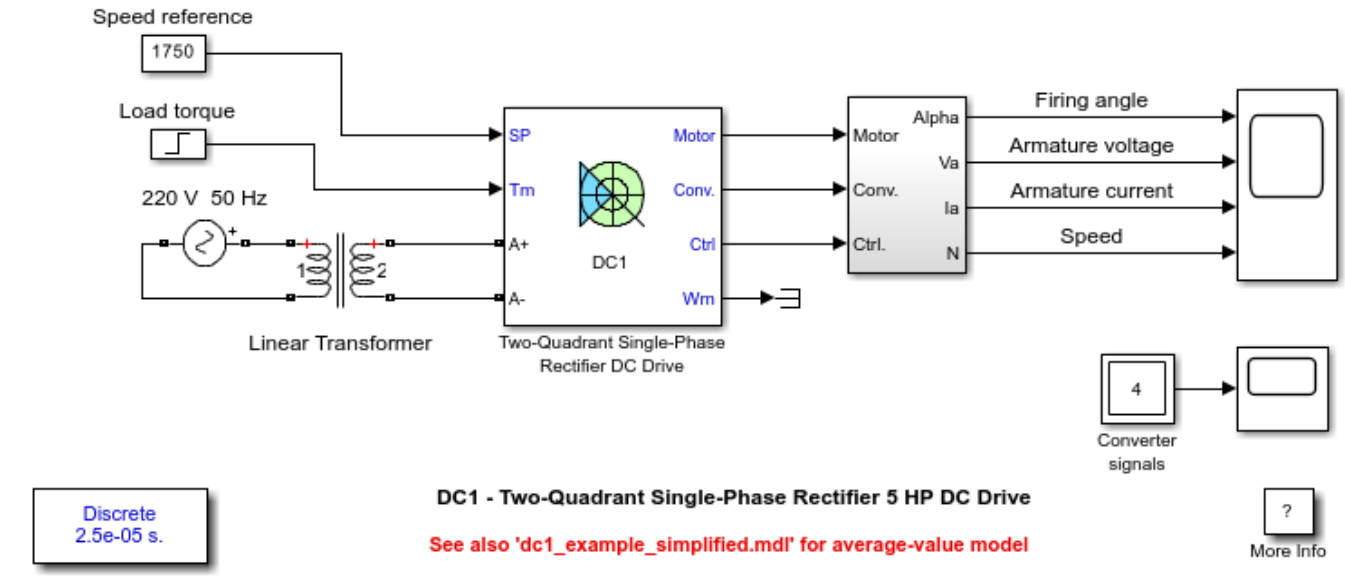
### Simulation

Select one of the four speed references provided. Start the simulation. You can observe the main and auxiliary windings stator currents, the rotor speed, the electromagnetic torque and the DC bus voltage on the Scope block. The speed set point is also shown. Restart the simulation and observe the drive response to successive changes in speed reference and control type (FOC or DTC)

# DC1 - Two-Quadrant Single-Phase Rectifier 5 HP DC Drive

This example shows the DC1 two-quadrant single-phase rectifier DC drive during speed regulation.

C.Semaille, Louis-A. Dessaint (Ecole de technologie superieure, Montreal)




### Description

This circuit uses the DC1 block of Specialized Power Systems. It models a two-quadrant single-phase rectifier drive for a 5 HP DC motor.

The 5 HP DC motor is separately excited with a constant 150 V DC field voltage source. The armature voltage is provided by a single-phase rectifier controlled by two PI regulators. The rectifier is fed by a 220 V AC 50 Hz voltage source followed by a linear transformer to boost the voltage up to a sufficient value.

The regulators control the firing angle of the rectifier thyristors. The first regulator is a speed regulator, followed by a current regulator. The speed regulator outputs the armature current reference (in p.u.) used by the current controller in order to obtain the electromagnetic torque needed to reach the desired speed. The speed reference change rate follows acceleration and deceleration ramps in order to avoid sudden reference changes that could cause armature over-current and destabilize the system. The current regulator controls the armature current by computing the appropriate thyristor firing angle. This generates the rectifier output voltage needed to obtain the desired armature current.

A 150 mH smoothing inductance is placed in series with the armature circuit to reduce armature current oscillations.

### Simulation

Start the simulation. You can observe the motor armature voltage and current, the rectifier firing angle and the motor speed on the scope. The current and speed references are also shown.

The speed reference is set at 1750 rpm at t = 0 s. Initial load torque is 15 N.m.

Observe that the motor speed follows the reference ramp accurately (+250 rpm/s) and reaches steady state around t = 8.5 s. The armature current follows the current reference very well, and the firing angle stays below 90 degrees, the converter being in rectifier mode (first quadrant operating mode). The lower limit of the firing angle has been set to 20 degrees.

At t = 8.75 s, the load torque passes from 15 N.m to 20 N.m. The motor speed recovers fast and is back at 1750 rpm at t = 10 s. The current reference rises to about 17.5 A to generate a higher electromagnetic torque to maintain the needed speed. As observed before, the armature current follows its reference perfectly.

### Notes

1) The power system has been discretized with a 25 us time step. The control system (regulators) uses a 100 us time step in order to simulate a microcontroller control device.

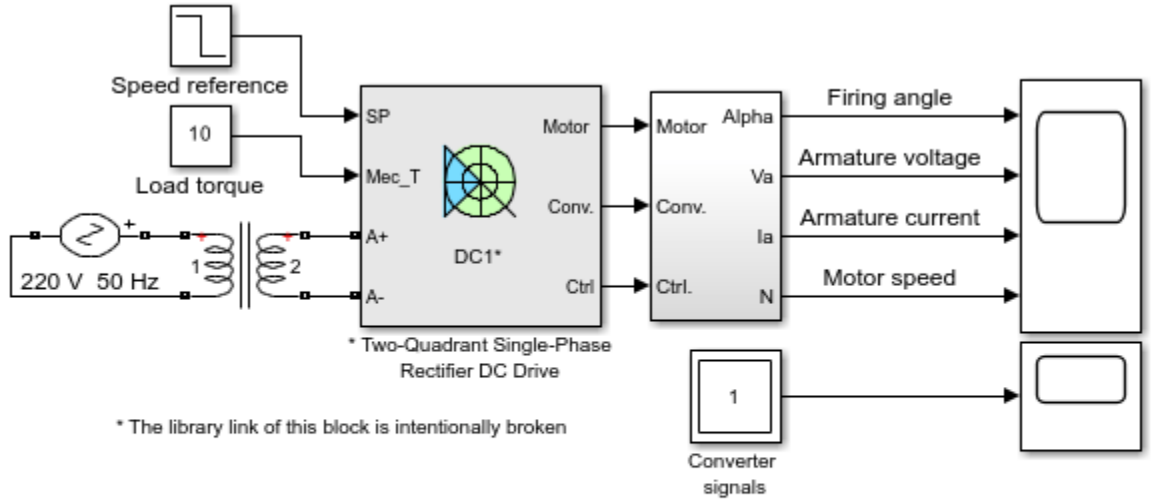
2) In order to reduce the number of points stored in the scope memory, a decimation factor of 20 is used.

3) A simplified version of the model using an average-value rectifier can be used by selecting 'Average' in the 'Model detail level' menu of the graphical user-interface. The time step can then be increased up to the control system sample time value.This can be done by typing 'Ts = 100e-6' in the workspace in the case of this example. See also dc1_example_simplified model.

# DC1 - Two-Quadrant Single-Phase Rectifier 5 HP DC Drive with Regenerative Braking System

This example shows a two quadrant single-phase rectifier DC drive with regenerative braking system.

C.Semaille, Louis-A. Dessaint (Ecole de technologie superieure, Montreal)




### Description

This circuit is based on the DC1 block of Specialized Power Systems. It models a two-quadrant single-phase rectifier drive for a 5 HP DC motor. A braking unit block has been added in order to simulate regenerative braking (quadrant IV operating mode).

The 5 HP DC motor is separately excited with a constant 150 V DC field voltage source. The armature voltage is provided by a single-phase rectifier controlled by two PI regulators. The rectifier is fed by a 220 V AC 50 Hz voltage source followed by a linear transformer to boost the voltage up to a sufficient value.

The regulators control the firing angle of the rectifier thyristors. The first regulator is a speed regulator, followed by a current regulator. The speed regulator outputs the armature current reference (in p.u.) used by the current controller in order to obtain the electromagnetic torque needed to reach the desired speed. The speed reference change rate follows acceleration and deceleration ramps in order to avoid sudden reference changes that could cause armature over-current and destabilize the system. The current regulator controls the armature current by computing the appropriate thyristor firing angle. This generates the rectifier output voltage needed to obtain the desired armature current.

The braking unit block is managed by a finite state machine with two states (normal operating mode and braking operating mode). When the system passes in braking mode, armature switches are activated and allow reversal of the armature current flow. This generates a reverse electromagnetic braking torque for fast speed deceleration. The reversal of the current flow is initiated when the armature current flowing through the switches equals 0 A. This avoids destructive arcs in the switches during commutation.

A 150 mH smoothing inductance is placed in series with the armature circuit to reduce current oscillations.

### Simulation

Start the simulation. You can observe the motor voltage, current, speed and rectifier firing angle on the scope. The speed and current references are also shown.

The initial speed reference is set to 800 rpm. Load torque is 10 N.m. Observe that the motor speed follows the acceleration reference ramp accurately (+350 rpm/s) and reaches steady-state after about 3.5 s. The armature current follows the current reference very well and stays below nominal current. During this phase, the average firing angle value stays below 90 degrees, the thyristor bridge being in rectifier mode (first quadrant operating mode).

At t = 4 s, the speed reference drops to 200 rpm and the system passes in braking mode. The armature switches are activated when the armature current reaches 0 A and reversal of the current flow through the motor takes place (the current flow direction through the bridge is of course unchanged). Observe again that the motor speed follows the deceleration ramp as wanted. The deceleration ramp has been set to a high value (-1250 rpm/s) to clearly show the effect of the braking electromagnetic torque. During this period of time, the bridge works in inverter mode (second quadrant operating mode).

At t = 4.5 s, motor speed is slightly lower than speed reference and the armature current flow through the motor is reversed back to normal. The bridge operates in rectifier mode and motor speed reaches 200 rpm around t = 5.5 s.

Notice the current overshoot during commutation. This is due to the sudden voltage reversal at the bridge output caused by armature switching. The bridge output voltage cannot follow instantaneously this voltage reversal. This sudden voltage difference between bridge and motor creates the current overshooting. However, the overshoot peak is of reasonable value and is not damageable.

### Notes

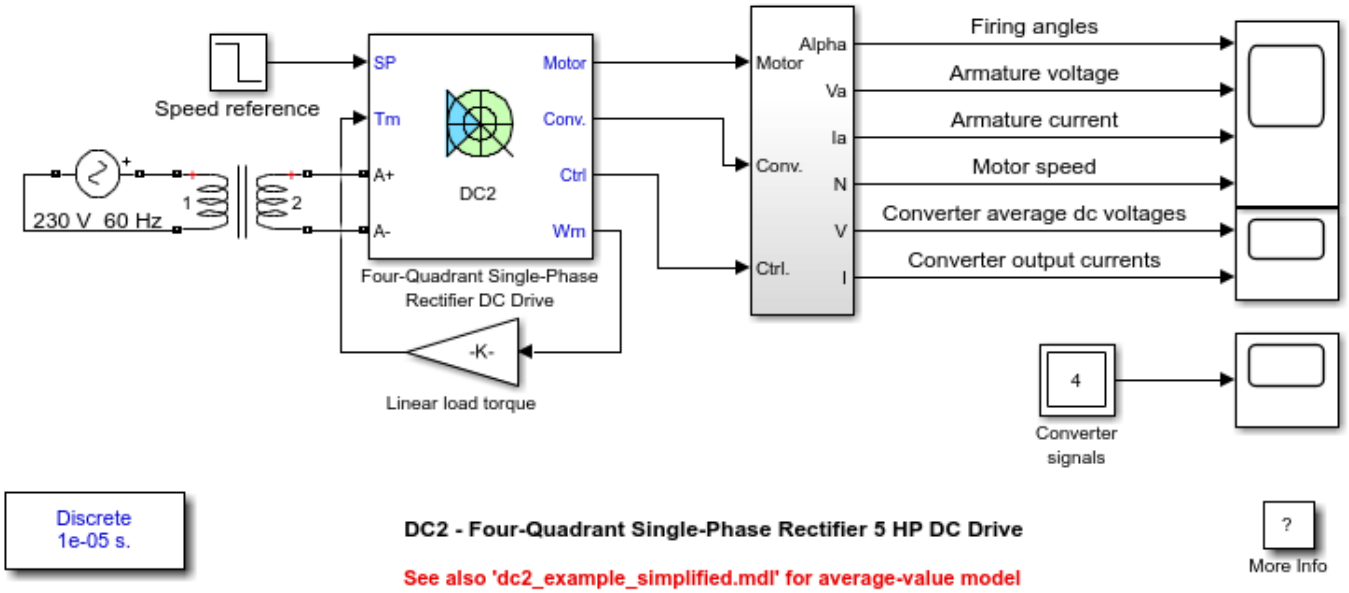
1) The power system has been discretized with a 10 us time step. The control system (regulators and braking unit) uses a 100 us time step in order to simulate a microcontroller control device.

2) In order to reduce the number of points stored in the scope memory, a decimation factor of 20 is used.

# DC2 - Four-Quadrant Single-Phase Rectifier 5 HP DC Drive

This example shows the DC2 four-quadrant single-phase rectifier DC drive with circulating current during speed regulation.

C.Semaille, Louis-A. Dessaint (Ecole de technologie superieure, Montreal)




### Description

This circuit uses the DC2 block of Specialized Power Systems. It models a four-quadrant single-phase rectifier (dual-converter topology) drive for a 5 HP DC motor.

The 5 HP DC motor is separately excited with a constant 150 V DC field voltage source. The armature voltage is provided by two single-phase anti-parallel connected converters controlled by two PI regulators. This allows bidirectional current flow through the DC motor armature circuit and thus four-quadrant operation. The converters are fed by a 230 V AC 60 Hz voltage source followed by a linear transformer to boost the voltage up to a sufficient value.

The regulators control the firing angles of both converter thyristors. The first regulator is a speed regulator, followed by a current regulator. The speed regulator outputs the armature current reference (in p.u.) used by the current controller in order to obtain the electromagnetic torque needed to reach the desired speed. The speed reference change rate follows acceleration and deceleration ramps in order to avoid sudden reference changes that could cause armature over-current and destabilize the system. The current regulator controls the armature current by computing the appropriate thyristor firing angles. This generates the converter output voltages needed to obtain the desired armature current.

Both converters operate simultaneously and the two firing angles are controlled so that their sum gives 180 degrees. This produces opposite average voltages at the converter dc output terminals and

thus identical average voltages at the DC motor armature, the converters being connected in anti-parallel. One converter is working in rectifier mode while the other is in inverter mode.

The circulating current produced by the instantaneous voltage difference at the terminal of both converters is limited by 80 mH inductors connected between these terminals. A 50 mH smoothing inductance is placed in series with the armature circuit to reduce armature current oscillations.

**Simulation**

Start the simulation. You can observe the motor armature voltage and current, the converter firing angles and the motor speed on the scope. The current and speed references are also shown. A second scope allows you to visualize the converter average output voltages and output currents.

During this simulation, the motor is coupled to a linear load, which means that the mechanical torque produced by the load is proportional to the speed.

The speed reference is set at 1200 rpm at t = 0 s. Observe that the firing angles are symmetrical around 90 degrees and that the converter average output DC voltages are of opposite signs. The armature current is supplied by converter 1, and the total current in this converter is the sum of load current and circulating current. Converter 2 simply carries the circulating current.

Observe that the motor speed follows the reference ramp accurately (+250 rpm/s) and reaches steady state after 5.5 s. The armature current follows the current reference very well and stabilizes around 12 A.

At t = 6 s, speed reference drops to -600 rpm. The current reference decreases to reduce the electromagnetic torque, and the load torque causes the motor to decelerate. Around t = 10.4 s, the armature current becomes negative and the electromagnetic torque reverses in order to brake the motor down to 0 rpm, the load torque being insufficient to decelerate the motor. At t = 10.8 s, the motor reaches 0 rpm and the load torque becomes negative. The electromagnetic torque now produces an accelerating torque to allow the motor to follow the negative speed ramp (-250 rpm/s). The armature current is now provided by converter 2, converter 1 only handling the circulating current.

At t = 13.2 s, speed stabilizes at -600 rpm.

**Notes**

1) The power system has been discretized with a 10 us time step. The control system (regulators) uses a 100 us sample time in order to simulate a microcontroller control device.

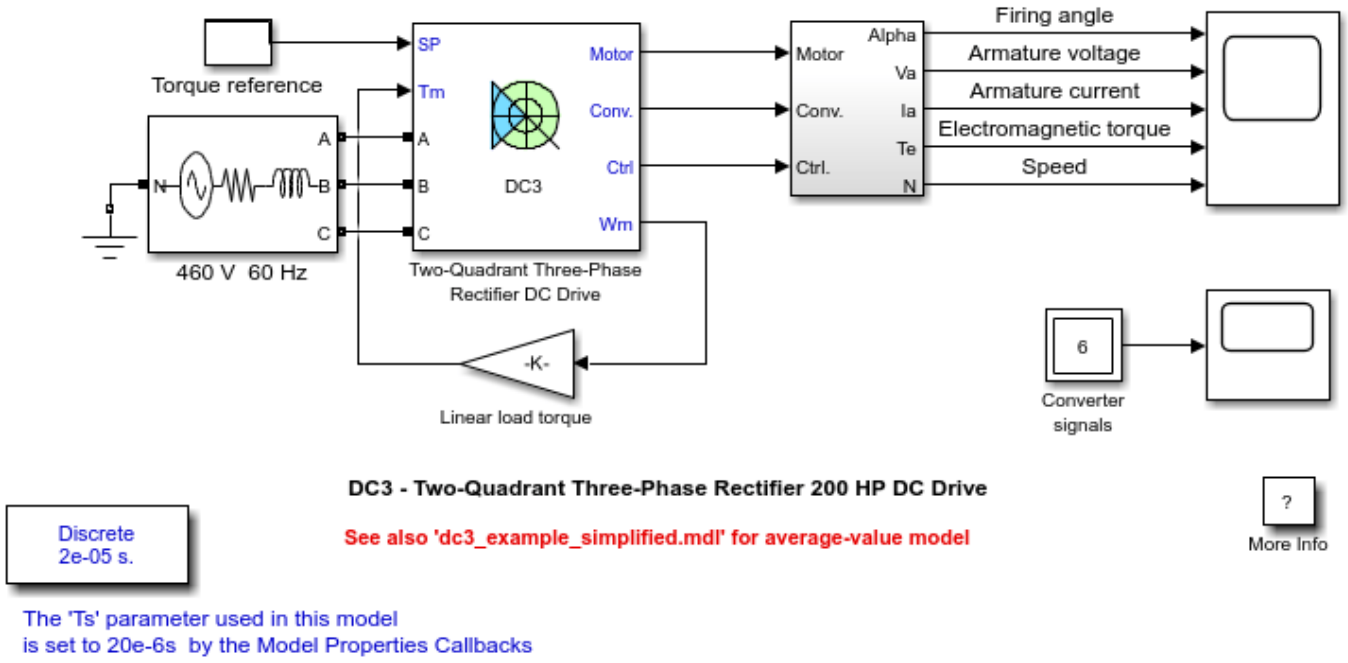
2) In order to reduce the number of points stored in the scope memory, a decimation factor of 20 is used.

3) A simplified version of the model using average-value rectifiers can be used by selecting 'Average' in the 'Model detail level' menu of the graphical user-interface. The time step can then be increased up to the control system sample time value.This can be done by typing 'Ts = 100e-6' in the workspace in the case of this example. See also dc2_example_simplified model.

# DC3 - Two-Quadrant Three-Phase Rectifier 200 HP DC Drive

This example shows the DC3 two-quadrant three-phase rectifier DC drive during torque regulation.

C.Semaille, Louis-A. Dessaint (Ecole de technologie superieure, Montreal)




### Description

This circuit uses the DC3 block of Specialized Power Systems. It models a two-quadrant three-phase rectifier drive for a 200 HP DC motor.

The 200 HP DC motor is separately excited with a constant 310 V DC field voltage source. The armature voltage is provided by a three-phase rectifier controlled by two PI regulators. The rectifier is fed by a 460 V AC 60 Hz voltage source.

The regulators control the firing angle of the rectifier thyristors. The first regulator is a speed regulator, followed by a current regulator. Since we are here in torque regulation mode, the speed regulator is disabled and only the current regulator is used. The current regulator controls the armature current by computing the appropriate thyristor firing angle. This generates the rectifier output voltage needed to obtain the desired armature current and thus the desired electromagnetic torque.

The current controller takes two inputs. The first one is the current reference (in p.u). This current reference is computed from the torque reference provided by the user. The second input is the armature current flowing through the machine.

A 15 mH smoothing inductance is placed in series with the armature circuit to reduce armature current oscillations.

### Simulation

Start the simulation. You can observe the motor armature voltage and current, the rectifier firing angle, the electromagnetic torque and the motor speed on the scope. The current and torque references are also shown.

The motor is coupled to a linear load, which means that the mechanical torque of the load is proportional to the speed.

The initial torque reference is set to 0 N.m and the armature current is null. No electromagnetic torque is produced, and the motor stays still.

At t = 0.05 s, the torque reference jumps to 800 N.m. This causes the armature current to rise to about 305 A. Notice that the armature current follows the reference quite accurately, with fast response time and small overshooting. The 15 mH smoothing inductance keeps the current oscillations quite small. Observe also that the average firing angle value stays below 90 degrees, the converter being in rectifier mode.

The electromagnetic torque produced by the armature current flow causes the motor to accelerate. The speed rises and starts to stabilize around t = 5 s at about 1450 rpm, the sum of the load and viscous friction torques beginning to equalize the electromagnetic torque.

At t = 5 s, the torque reference is set to 400 N.m and the armature current jumps down to about 155 A. This causes the load torque to decelerate the motor.

At t = 10 s speed starts to stabilize around 850 rpm.

### Notes

1) The power system has been discretized with a 20 us time step. The control system (regulators) uses a 100 us time step in order to simulate a microcontroller control device.

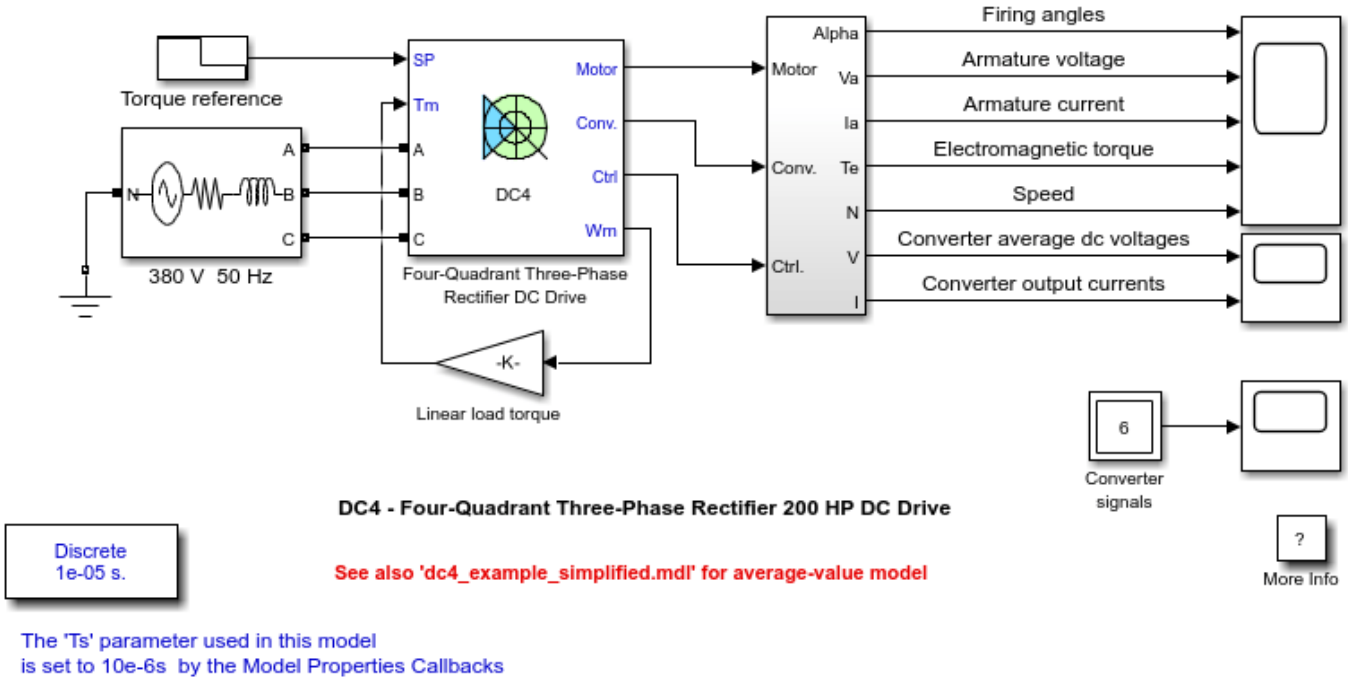
2) In order to reduce the number of points stored in the scope memory, a decimation factor of 20 is used.

3) A simplified version of the model using an average-value rectifier can be used by selecting 'Average' in the 'Model detail level' menu of the graphical user-interface. The time step can then be increased up to the control system sample time value.This can be done by typing 'Ts = 100e-6' in the workspace in the case of this example. See also dc3_example_simplified model.

# DC4 - Four-Quadrant Three-Phase Rectifier 200 HP DC Drive

This example shows the DC4 four-quadrant three-phase rectifier DC drive with circulating current during torque regulation.

C.Semaille, Louis-A. Dessaint (Ecole de technologie superieure, Montreal)




### Description

This circuit uses the DC4 block of Specialized Power Systems. It models a four-quadrant three-phase rectifier (dual-converter topology) drive for a 200 HP DC motor.

The 200 HP DC motor is separately excited with a constant 310 V DC field voltage source. The armature voltage is provided by two three-phase anti-parallel connected converters controlled by two PI regulators. This allows bidirectional current flow through the DC motor armature circuit and thus four-quadrant operation. The converters are fed by a 380 V AC 50 Hz voltage source.

The regulators control the firing angles of both converter thyristors. The first regulator is a speed regulator, followed by a current regulator. Since we are here in torque regulation mode, the speed regulator is disabled and only the current regulator is used. The current regulator controls the armature current by computing the appropriate thyristor firing angles. This generates the rectifier output voltages needed to obtain the desired armature current and thus the desired electromagnetic torque.

Both converters operate simultaneously and the two firing angles are controlled so that their sum gives 180 degrees. This produces opposite average voltages at the converter dc output terminals and thus identical average voltages at the DC motor armature, the converters being connected in anti-parallel. One converter is working in rectifier mode while the other is in inverter mode.

The circulating current produced by the instantaneous voltage difference at the terminal of both converters is limited by 5 mH inductors connected between these terminals. No smoothing inductance is placed in series with the armature circuit, the armature current oscillations being quite small due to the three-phase voltage source.

**Simulation**

Start the simulation. You can observe the motor armature voltage and current, the converter firing angles, the electromagnetic torque and the motor speed on the scope. The current and torque references are also shown. A second scope allows you to visualize the converter average output voltages and output currents.

The motor is coupled to a linear load, which means that the mechanical torque of the load is proportional to the speed.

The initial torque reference is set to 0 N.m and the armature current is null. No electromagnetic torque is produced and the motor stays still.

At t = 0.2 s, the torque reference jumps to 600 N.m. This causes the armature current to rise to about 180 A. The armature current is supplied by converter 1, and the total current in this converter is the sum of load current and circulating current. Converter 2 simply carries the circulating current. Notice that the armature current follows the reference current quite accurately, with fast response time and small overshooting. Observe also that the firing angles are symmetrical around 90 degrees and that the converter average output DC voltages are equal but of opposite signs.

The electromagnetic torque produced by the armature current flow causes the motor to accelerate. The speed rises and starts to stabilize around t = 4 s at about 560 rpm, the sum of the load and viscous friction torques beginning to equalize the electromagnetic torque.

At t = 4 s, the torque reference is set to 0 N.m and the load torque causes the motor to decelerate. Notice that the four reactors keep the current oscillations quite small.

At t = 8 s, the torque reference is set to -300 N.m. The armature current jumps down to -90 A and is now delivered by converter 2 while converter 1 only handles the circulating current. Converter 2 is now working in rectifier mode and converter 1 in inverter mode.

The negative electromagnetic torque produced allows the motor to accelerate in the negative speed plane.

At t = 12 s, speed starts to stabilize around -290 rpm.

**Notes**

1) The power system has been discretized with a 10 us time step. The control system (regulators) uses a 100 us sample time in order to simulate a microcontroller control device.

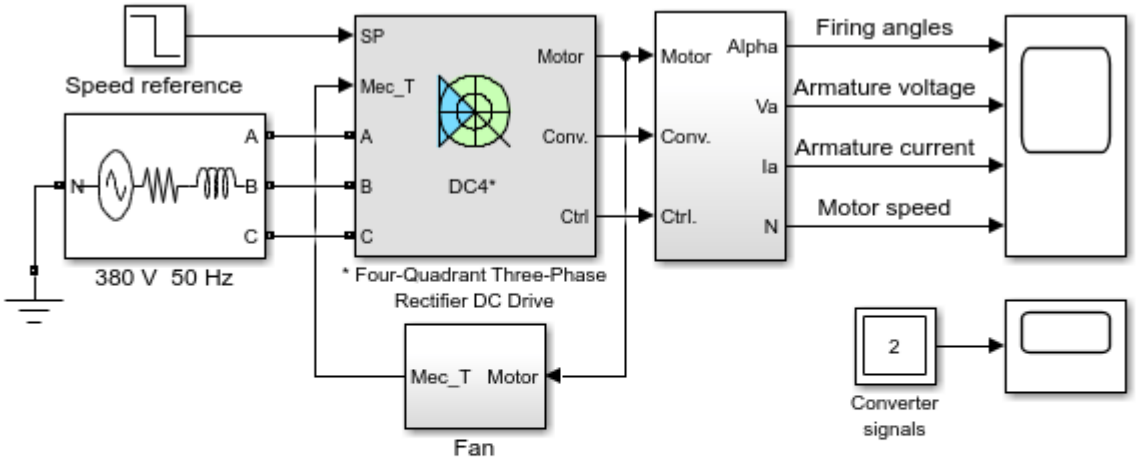
2) In order to reduce the number of points stored in the scope memory, a decimation factor of 20 is used.

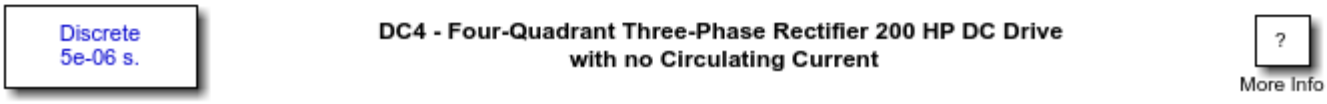
3) A simplified version of the model using average-value rectifiers can be used by selecting 'Average' in the 'Model detail level' menu of the graphical user-interface. The time step can then be increased up to the control system sample time value.This can be done by typing 'Ts = 100e-6' in the workspace in the case of this example. See also dc4_example_simplified example.

# DC4 - Four-Quadrant Three-Phase Rectifier 200 HP DC Drive with No Circulating Current

This example shows a four-quadrant three-phase rectifier DC drive with no circulating current.

C.Semaille, Louis-A. Dessaint (Ecole de technologie superieure, Montreal)




### Description

This circuit is based on the DC4 block of Specialized Power Systems. It models a four-quadrant three-phase rectifier (dual-converter topology) drive with no circulating current for a 200 HP DC motor.

The 200 HP DC motor is separately excited with a constant 310 V DC field voltage source. The armature voltage is provided by two three-phase anti-parallel connected converters controlled by two PI regulators. This allows bidirectional current flow through the DC motor armature circuit and thus four-quadrant operation. The converters are fed by a 380 V AC 50 Hz voltage source.

The regulators control the firing angles of both converter thyristors. The first regulator is a speed regulator, followed by a current regulator. The speed regulator outputs the armature current reference (in p.u.) used by the current controller in order to obtain the electromagnetic torque needed to reach the desired speed. The speed reference change rate follows acceleration and deceleration ramps in order to avoid sudden reference changes that could cause armature over-current and destabilize the system. The current regulator controls the armature current by computing the appropriate thyristor firing angles. This generates the converter output voltages needed to obtain the desired armature current.

Compared with the original DC4 block, this circuit models a four-quadrant drive with no circulating current. During this regulation process, the flow of circulating current is completely inhibited by automatic control of the firing pulses. By enabling only one of the two firing pulses needed for the two thyristor converters, only one converter operates at a time and carries the load current. The other converter is temporarily blocked. The firing pulse control is completely managed by the "bridge driver" module. By sensing the reference and load currents, this module determines when converter crossover has to take place by enabling the appropriate converter's firing pulses.

The two firing angles are controlled so that their sum gives 180 degrees. This allows smoother bridge transition. Since circulating current is inhibited, no more inductors are needed to limit the value of this current. However, a 10 mH smoothing inductance is placed in series with the armature circuit to reduce armature current oscillations.

**Simulation**

Start the simulation. You can observe the motor armature voltage and current, the converter firing angles and the motor speed on the scope. The current and speed references are also shown. A second scope has been added inside the main block to allow you to visualize the converter output currents. The "block_1" and "block_2" signals, controlled by the "bridge driver" module are also visible.

During this simulation the motor is coupled to a fan. The mechanical torque of this type of load is proportional to the square of the speed.

The initial speed reference is set to 1184 rpm, nominal speed. Observe that the motor speed follows the acceleration reference ramp accurately (+320 rpm/s) and reaches steady-state after about 4 s. The slow acceleration is due to the high inertia of the load.

The armature current follows the current reference very well and stabilizes around 330 A. During the acceleration phase, the armature current rises progressively (and thus also the electromagnetic torque produced) the mechanical torque opposed by the load rising with the speed. Only converter 1 is working (Block_1 is low) and converter 2 is inhibited (Block_2 is high). Consequently the output current of converter 1 is equal to the load current and converter 2 outputs no current. Notice that the 10 mH smoothing inductance keeps the armature current oscillations quite small.

At t = 4.5 s, the speed reference drops to -600 rpm and the armature current lowers with the speed to reduce the electromagnetic torque in order to decelerate following the negative speed ramp (-320 rpm/s).

Around t = 5.3 s, the armature current reaches 0 A and bridge crossover takes place to let the armature current become negative. Converter 1 is disabled and converter 2 enabled (Block_1 becomes high and Block_2 becomes low). To avoid simultaneous conduction of both converters (and thus to avoid circulating current) during the crossover, the incoming converter is enabled a few milliseconds after disabling Converter 1. Converter 2 then outputs the load current and the output current of converter 1 is nul. Notice that the current waveforms keep smooth during bridge transition. This negative current now generates a braking torque to keep the fan slowing down.

At t = 8.2 s, the speed reaches 0 rpm and the armature current now generates an accelerating torque to allow the fan to accelerate in the negative speed plane.

At t =11 s, speed and armature current stabilize around -600 rpm and 90 A respectively.

**Notes**

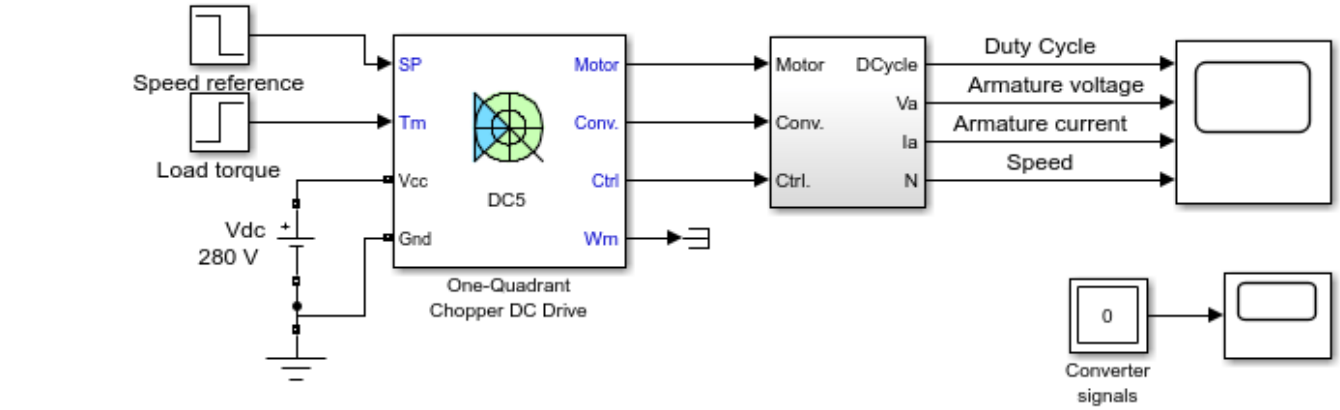
1) The power system has been discretized with a 5 us time step. The control system (regulators and bridge driver module) uses a 100 us time step in order to simulate a microcontroller control device.

2) In order to reduce the number of points stored in the scope memory, a decimation factor of 10 is used.

# DC5 - One-Quadrant Chopper 5 HP DC Drive

This example shows the DC5 one-quadrant chopper DC drive during speed regulation.

C.Semaille, Louis-A. Dessaint (Ecole de technologie superieure, Montreal)



### Description

This circuit uses the DC5 block of Specialized Power Systems. It models a one-quadrant chopper (buck converter) drive for a 5 HP DC motor.

The 5 HP DC motor is separately excited with a constant 150 V DC field voltage source. The armature voltage is provided by an IGBT buck converter controlled by two PI regulators. The buck converter is fed by a 280 V DC voltage source.

The first regulator is a speed regulator, followed by a current regulator. The speed regulator outputs the armature current reference (in p.u.) used by the current controller in order to obtain the electromagnetic torque needed to reach the desired speed. The speed reference change rate follows acceleration and deceleration ramps in order to avoid sudden reference changes that could cause armature over-current and destabilize the system. The current regulator controls the armature current by computing the appropriate duty ratio of the IGBT 5 kHz pulses (Pulse Width Modulation). This generates the average armature voltage needed to obtain the desired armature current. In order to limit the amplitude of the current oscillations, a smoothing inductance is placed in series with the armature circuit.

### Simulation

Start the simulation. You can observe the motor armature voltage and current, the IGBT pulses and the motor speed on the scope. The current and speed references are also shown.

The speed reference is set at 500 rpm at t = 0 s. Initial load torque is 15 N.m.

Observe that the motor speed follows the reference ramp accurately (+250 rpm/s) and reaches steady state around t = 2.5 s. The armature current follows the current reference very well, with fast response time and small ripples. Notice that the current ripple frequency is 5 kHz.

At t = 2.5 s, the load torque passes from 15 N.m to 20 N.m. The motor speed recovers fast and is back at 500 rpm at t = 3 s. The current reference rises to about 16.7 A to generate a higher electromagnetic torque to maintain the speed reference. As observed before, the armature current follows its reference perfectly.

At t = 3 s, the speed reference jumps down to 350 rpm. The armature current lowers in order for the speed to decrease following the negative speed slope (-250 rpm/s) with the help of the load torque.

At t = 4 s, the speed stabilizes around its reference.

**Notes**

1) The power system has been discretized with a 1us time step. The speed and current controllers use a 100 us and 20 us sampling time respectively in order to simulate a microcontroller control device.

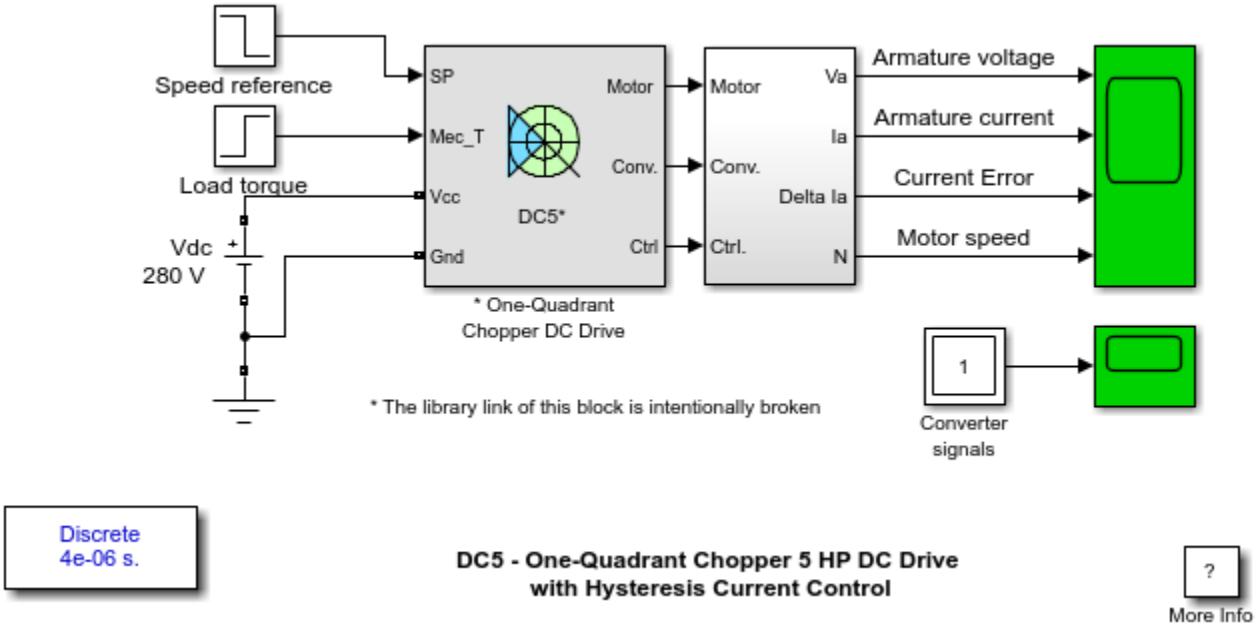
2) In order to reduce the number of points stored in the scope memory, a decimation factor of 25 is used. Some transitions may thus not appear on the scope. To view detailed simulation results, reduce the decimation factor to 1.

3) A simplified version of the model using an average-value converter can be used by selecting 'Average' in the 'Model detail level' menu of the graphical user-interface. The time step can then be increased up to the smallest control system sample time value. This can be done by typing 'Ts = 20e-6' in the workspace in the case of this example. See also dc5_example_simplified model.

# DC5 - One-Quadrant Chopper 5 HP DC Drive with Hysteresis Current Control

This example shows a one-quadrant chopper DC drive with hysteresis current control during speed regulation.

C.Semaille, Louis-A. Dessaint (Ecole de technologie superieure, Montreal)




### Description

This circuit is based on the DC5 block of Specialized Power Systems. It models a one-quadrant chopper (buck converter) drive for a 5 HP DC motor.

The 5 HP DC motor is separately excited with a constant 150 V DC field voltage source. The armature voltage is provided by an IGBT buck converter controlled by two regulators. The buck converter is fed by a 280 V DC voltage source.

The first regulator is a PI speed regulator, followed by a hysteresis current regulator. The speed regulator outputs the armature current reference (in p.u.) used by the current controller in order to obtain the electromagnetic torque needed to reach the desired speed. The speed reference change rate follows acceleration and deceleration ramps in order to avoid sudden reference changes that could cause armature over-current and destabilize the system. The current regulator controls the armature current by delivering the correct pulses to the IGBT device in order to keep the armature current inside a user-defined hysteresis band. The switching frequency of the IGBT device is limited by the motor inductance and an external inductance placed in series with the armature circuit.

### Simulation

Start the simulation. You can observe the motor armature voltage and current, the IGBT pulses and the motor speed on the scope. The current and speed references are also shown.

The speed reference is set at 500 rpm at t = 0 s. Initial load torque is 15 N.m.

Observe that the motor speed follows the reference ramp accurately (+250 rpm/s) and reaches steady-state around t = 2.5 s. The armature current follows the current reference very well and stays limited inside it's hysteresis band.

At t = 2.5 s, the load torque passes from 15 N.m to 20 N.m. The motor speed recovers fast and is back at 500 rpm at t = 3 s. The current reference rises to about 16.7 A to generate a higher electromagnetic torque to maintain the speed reference. As observed before, the armature current follows its reference perfectly.

At t = 3 s, the speed reference jumps down to 350 rpm. The armature current lowers in order for the speed to decrease following the negative speed slope (-250 rpm/s) with the help of the load torque.

At t = 4 s, the speed stabilizes around its reference.

**Notes**

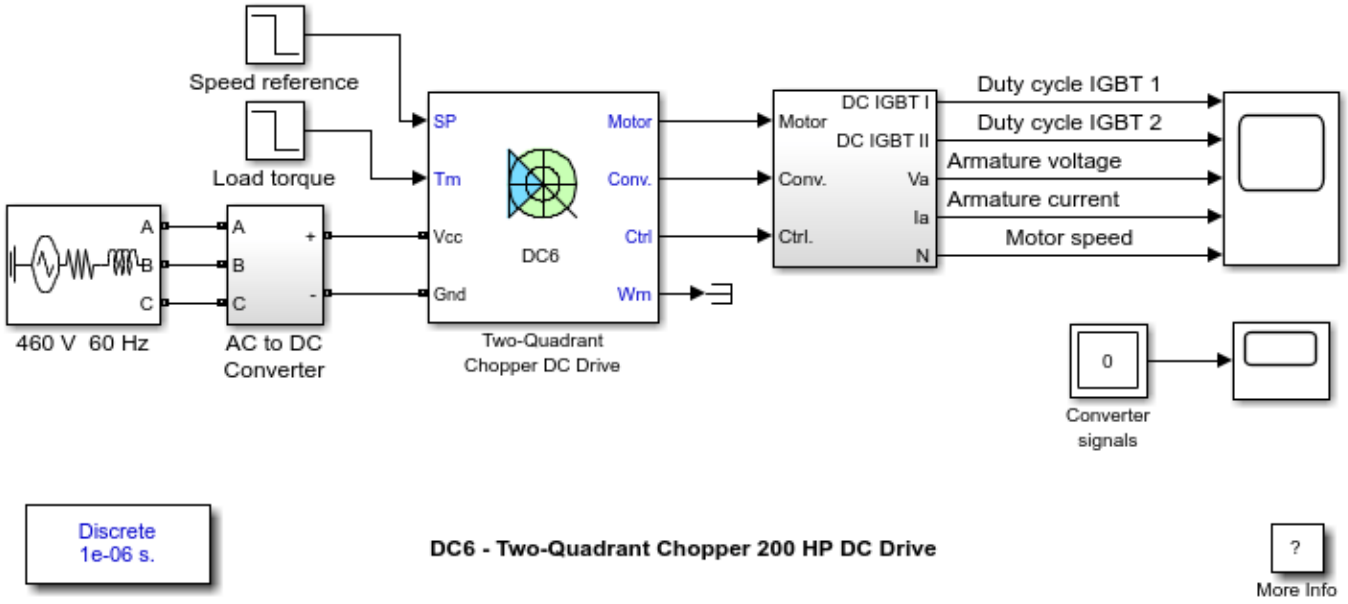
1) The power system has been discretized with a 4 us time step. The control system (regulators) uses a 100 us time step in order to simulate a microcontroller control device.

2) In order to reduce the number of points stored in the scope memory, a decimation factor of 5 is used.

# DC6 - Two-Quadrant Chopper 200 HP DC Drive

This example shows the DC6 two-quadrant chopper DC drive during speed regulation.

C.Semaille, Louis-A. Dessaint (Ecole de technologie superieure, Montreal)




### Description

The 200 HP DC motor is separately excited with a constant 150 V DC field voltage source. The armature voltage is provided by an IGBT buck-boost converter controlled by two PI regulators. The buck-boost converter is fed by a 630 V DC bus obtained by rectification of a 460 V AC 60 Hz voltage source. In order to limit the DC bus voltage during dynamic braking mode, a braking chopper has been added between the diode rectifier and the DC6 block.

The first regulator is a speed regulator, followed by a current regulator. The speed regulator outputs the armature current reference (in p.u.) used by the current controller in order to obtain the electromagnetic torque needed to reach the desired speed. The speed reference change rate follows acceleration and deceleration ramps in order to avoid sudden reference changes that could cause armature over-current and destabilize the system. The current regulator controls the armature current by computing the appropriate duty ratios of the 5 kHz pulses of the two IGBT devices (Pulse Width Modulation).For proper system behaviour, the two IGBT devices have opposite instantaneous pulse values.This generates the average armature voltage needed to obtain the desired armature current. In order to limit the amplitude of the current oscillations, a smoothing inductance is placed in series with the armature circuit.

### Simulation

Before starting the simulation, set the initial bus voltage to 630 V via the GUI block ('Initial States Setting' button and 'Cbus' variable).

Start the simulation. You can observe the motor armature voltage and current, the two IGBT pulses and the motor speed on the scope. The current and speed references are also shown.

The speed reference is set at 400 rpm at t = 0 s. Initial load torque is 814 N.m.

Observe that the motor speed follows the reference ramp accurately (+250 rpm/s) and reaches steady state around t = 2 s. The armature current follows the current reference very well, with fast response time and small ripples. Notice that the current ripple frequency is 5 kHz.

At t = 2.1 s, the load torque passes from 814 N.m to 100 N.m. The motor speed recovers fast and is back at 400 rpm at t = 2.75 s. The current reference lowers to about 40 A to generate a smaller electromagnetic torque, the load torque being reduced. As observed before, the armature current follows its reference perfectly.

At t = 2.75 s, the speed reference jumps down to 100 rpm. In order for the motor to decelerate following the negative speed ramp, the armature current reverses down to -160 A to generate a braking electromagnetic torque (dynamic braking mode). This causes the DC bus voltage to increase. The braking chopper limits the voltage value.

At t = 3.4 s, the motor speed reaches 100 rpm and the current reverses back to 40 A.

At t = 4 s, the speed stabilizes around its reference.

**Notes**

1) The power system has been discretized with a 1 us time step. The speed and current controllers use a 100 us and 20 us sampling time respectively in order to simulate a microcontroller control device.

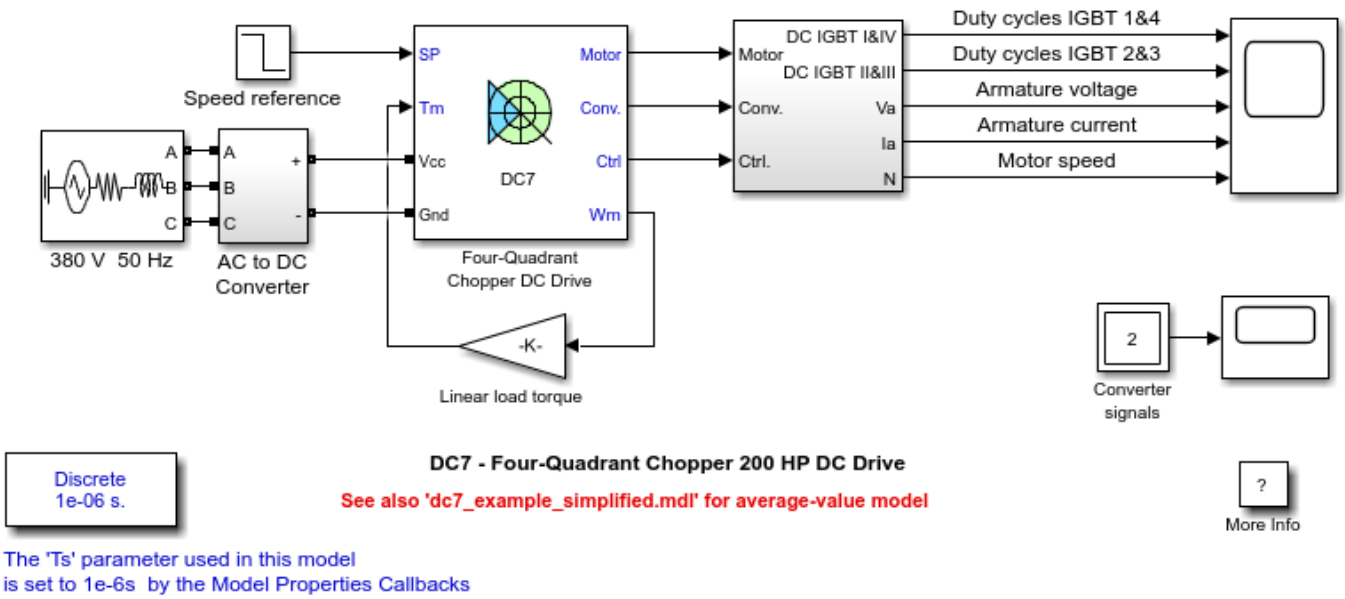
2) In order to reduce the number of points stored in the scope memory, a decimation factor of 25 is used. Some transitions may thus not appear on the scope. To view detailed simulation results, reduce the decimation factor to 1.

3) A simplified version of the model using an average-value converter can be used by selecting 'Average' in the 'Model detail level' menu of the graphical user-interface. The time step can then be increased up to the smallest control system sample time value. This can be done by typing 'Ts = 20e-6' in the workspace in the case of this example. See also dc6_example_simplified model.

# DC7 - Four-Quadrant Chopper 200 HP DC Drive

This example shows the DC7 four-quadrant chopper DC drive during speed regulation.

C.Semaille, Louis-A. Dessaint (Ecole de technologie superieure, Montreal)




### Description

The 200 HP DC motor is separately excited with a constant 150 V DC field voltage source. The armature voltage is provided by an IGBT converter controlled by two PI regulators. The converter is fed by a 515 V DC bus obtained by rectification of a 380 V AC 50 Hz voltage source. In order to limit the DC bus voltage during dynamic braking mode, a braking chopper has been added between the diode rectifier and the DC7 block.

The first regulator is a speed regulator, followed by a current regulator. The speed regulator outputs the armature current reference (in p.u.) used by the current controller in order to obtain the electromagnetic torque needed to reach the desired speed. The speed reference change rate follows acceleration and deceleration ramps in order to avoid sudden reference changes that could cause armature over-current and destabilize the system. The current regulator controls the armature current by computing the appropriate duty ratios of the 5 kHz pulses of the four IGBT devices (Pulse Width Modulation). For proper system behaviour, the instantaneous pulse values of IGBT devices 1 and 4 are opposite to those of IGBT devices 2 and 3. This generates the average armature voltage needed to obtain the desired armature current. In order to limit the amplitude of the current oscillations, a smoothing inductance is placed in series with the armature circuit.

### Simulation

Before starting the simulation, set the initial bus voltage to 515 V via the GUI block ('Initial States Setting' button and 'Cbus' variable).

Start the simulation. You can observe the motor armature voltage and current, the four IGBT pulses and the motor speed on the scope. The current and speed references are also shown.

The motor is coupled to a linear load, which means that the mechanical torque of the load is proportional to the speed.

The speed reference is set at 500 rpm at t = 0 s. Observe that the motor speed follows the reference ramp accurately (+400 rpm/s) and reaches steady state around t = 1.3 s.

The armature current follows the current reference very well, with fast response time and small ripples. Notice that the current ripple frequency is 5 kHz.

At t = 2 s, speed reference drops to -1184 rpm. The current reference decreases to reduce the electromagnetic torque and causes the motor to decelerate with the help of the load torque.

At t = 2.2 s, the current reverses in order to produce a braking electromagnetic torque (dynamic braking mode). This causes the DC bus voltage to increase.

At t = 3.25 s, the motor reaches 0 rpm and the load torque reverses and becomes negative. The negative current now produces an accelerating electromagnetic torque to allow the motor to follow the negative speed ramp (-400 rpm/s). At t = 6.3 s, the speed reaches -1184 rpm and stabilizes around its reference.

**Notes**

1) The power system has been discretized with a 1 us time step. The speed and current controllers use a 100 us and 20 us sampling time respectively in order to simulate a microcontroller control device.

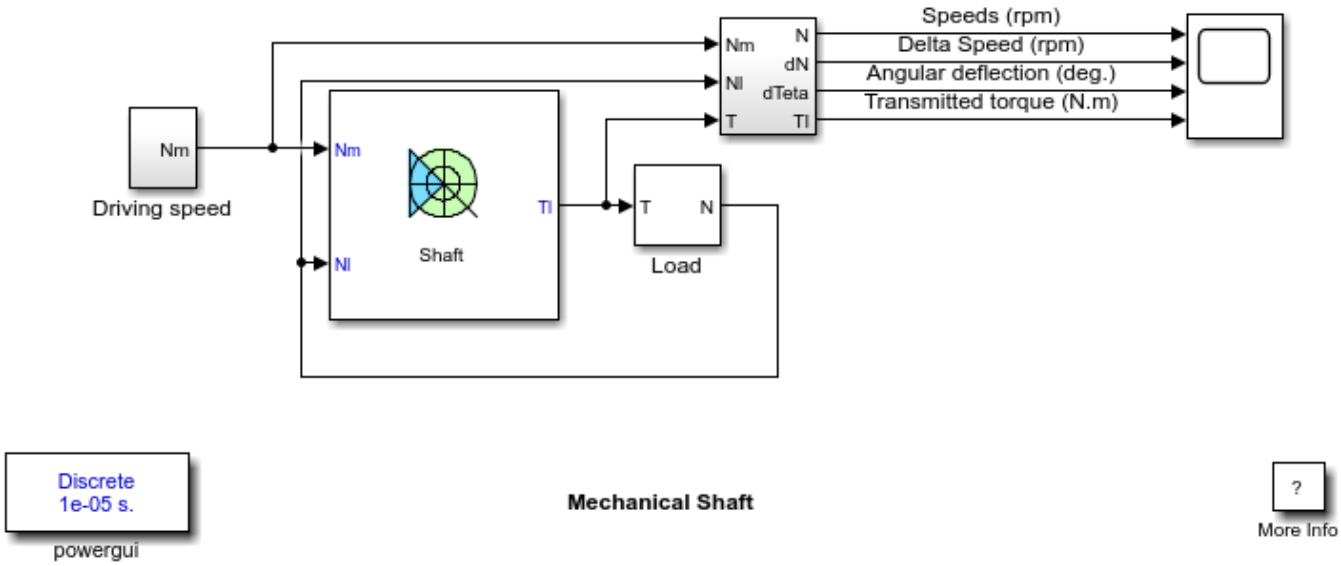
2) In order to reduce the number of points stored in the scope memory, a decimation factor of 25 is used. Some transitions may thus not appear on the scope. To view detailed simulation results, reduce the decimation factor to 1.

3) A simplified version of the model using an average-value converter can be used by selecting 'Average' in the 'Model detail level' menu of the graphical user-interface. The time step can then be increased up to the smallest control system sample time value. This can be done by typing 'Ts = 20e-6' in the workspace in the case of this example. See also dc7_example_simplified model.

# Mechanical Shaft

This example shows the mechanical shaft model.

C.Semaille, Louis-A. Dessaint (Ecole de technologie superieure, Montreal)




### Description

The model outputs the transmitted torque through the shaft regarding the speed difference between the driving side and the loaded side of the shaft.

The shaft has a stiffness of 17190 N.m and an internal damping factor of 600 N.m.s. This shaft is designed to have 0.1 degree of angular deflection for a 30 N.m load torque.

The shaft is driven by a variable speed source and is connected to a load. The load has an inertia of 0.35 kg.m2 and a viscous friction term of 0.006 N.m.s

### Simulation

Start the simulation. You can observe the driving and the load speeds, the speed difference, the angular deflection and the transmitted torque values on the scope.

At t = 0 s, the driving speed starts climbing to 1750 rpm with a 500 rpm/s acceleration ramp. The angular deflection jumps to about 0.06 degree and the shaft transmits about 18.5 Nm to the load in order to accelerate it. At t = 0.2 s, the driving and load speeds tend to equalize. During the acceleration phase, the angular deflection increases slowly in order to transmit a higher torque to compensate the viscous friction increase.

At t = 3.5 s, the driving speed settles at 1750 rpm. This reduces the angular deflection and also the transmitted torque which settles around 1.1 Nm to compensate the viscous friction of the load.

At t = 5 s, the driving speed lowers towards 0 rpm with a -500 rpm/s deceleration ramp. The angular deflection becomes negative and thus the transmitted torque in order to decelerate the load. During

the deceleration phase, the angular deflection increases in order to transmit a higher deceleration torque to compensate the reduction of viscous friction.

At t = 8.5 s, the driving speed stabilizes at 0 rpm. This causes the angular deflection to reduce to 0 degree, the transmitted torque becomes nul and the load stops.

**Notes**

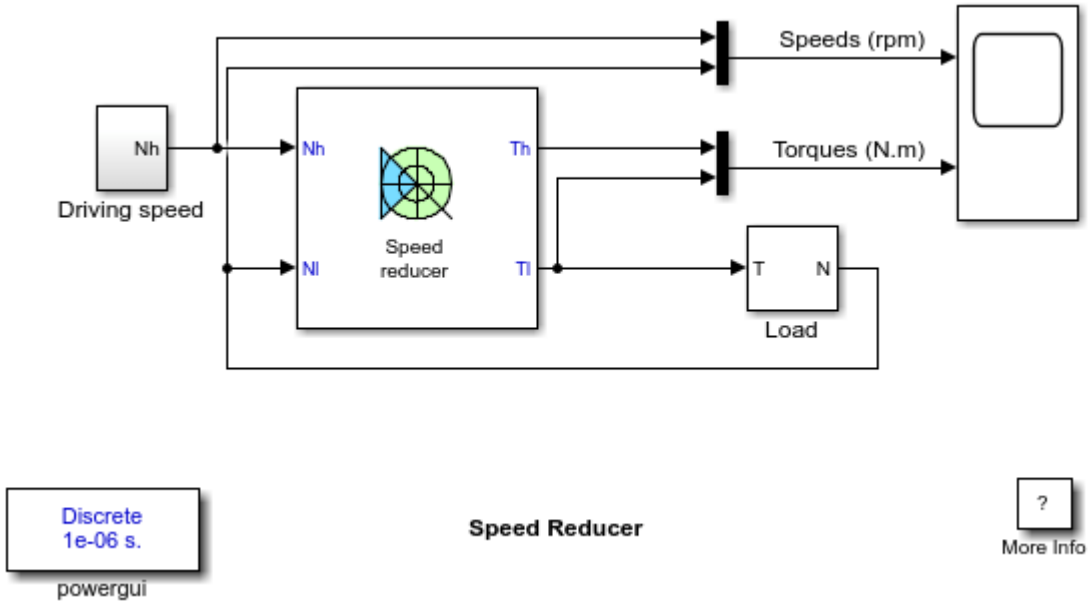
1) The shaft has been discretized with a 10 us time step.

2) In order to reduce the number of points stored in the scope memory, a decimation factor of 10 is used.

# Speed Reducer

This example shows the speed reducer model connected between a high-speed shaft and a low-speed shaft.

C.Semaille, Louis-A. Dessaint (Ecole de technologie superieure, Montreal)




### Description

The speed reducer is driven by a variable speed source and is connected to a load. The load has an inertia of 30 kg.m2 and a viscous friction term of 0.5 N.m.s.

The reduction device has a reduction ratio of 10 and it's inertia with respect to the high-speed side is 0.0005 kg.m2. The reduction ratio being quite low, the efficiency is high and worth 0.95.

The high-speed shaft has a stiffness of 17190 N.m and an internal damping factor of 600 N.m.s. This shaft is designed to have 0.1 degree of angular deflection for a 30 N.m load torque. The low-speed shaft, having a higher torque to transmit, has a stiffness of 171900 N.m and an internal damping factor of 6000 N.m.s. This shaft is designed to have 0.1 degree of angular deflection for a 300 N.m load torque.

### Simulation

Start the simulation. You can observe the driving (high-speed) and load speeds (low-speed), the torque transmitted by the high-speed shaft and the torque transmitted by the low-speed shaft on the scope.

At t = 0 s, the driving speed starts climbing to 1750 rpm with a 500 rpm/s acceleration ramp. This causes the transmitted torque of the high-speed shaft to jump to about 18 N.m. Because of the reduction device, the torque transmitted to the load by the low-speed shaft is a lot bigger and is worth about 170 N.m.

During the accelerating phase, both torques keep increasing in order to compensate the viscous friction of the load. Notice that the load accelerates with a ramp of +50 rpm/s due to the reduction ratio of the speed reducer.

At t = 3.5 s, the driving speed settles at 1750 rpm. Since no more accelerating torque is needed, the input and output torques decrease and stabilize respectively to 0.965 N.m and 9.16 N.m at t = 4 s. The load speed is now equal to 175 rpm.

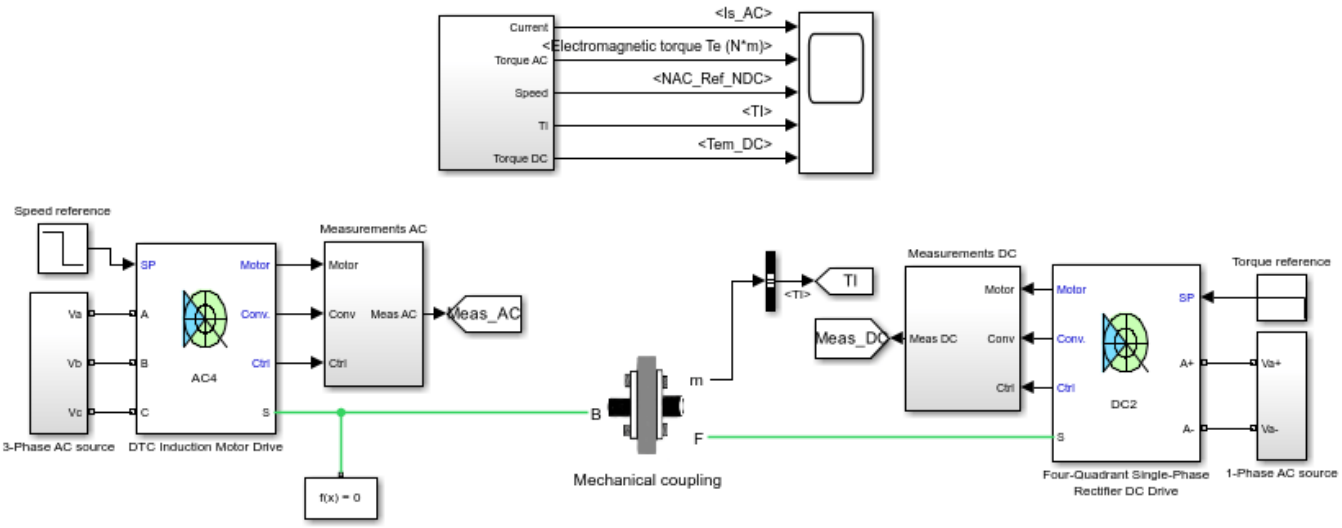
**Notes**

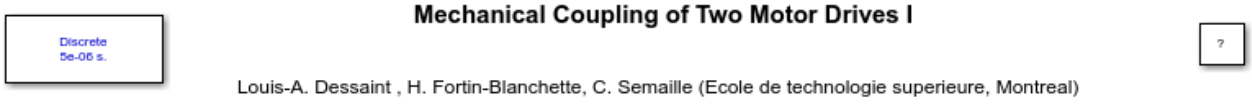
1) The speed reducer has been discretized with a 1 us time step.

# Mechanical Coupling of Two Motor Drives I

This example shows mechanical coupling of two motor drives.

Louis-A. Dessaint , H. Fortin-Blanchette, C. Semaille (Ecole de technologie superieure, Montreal)

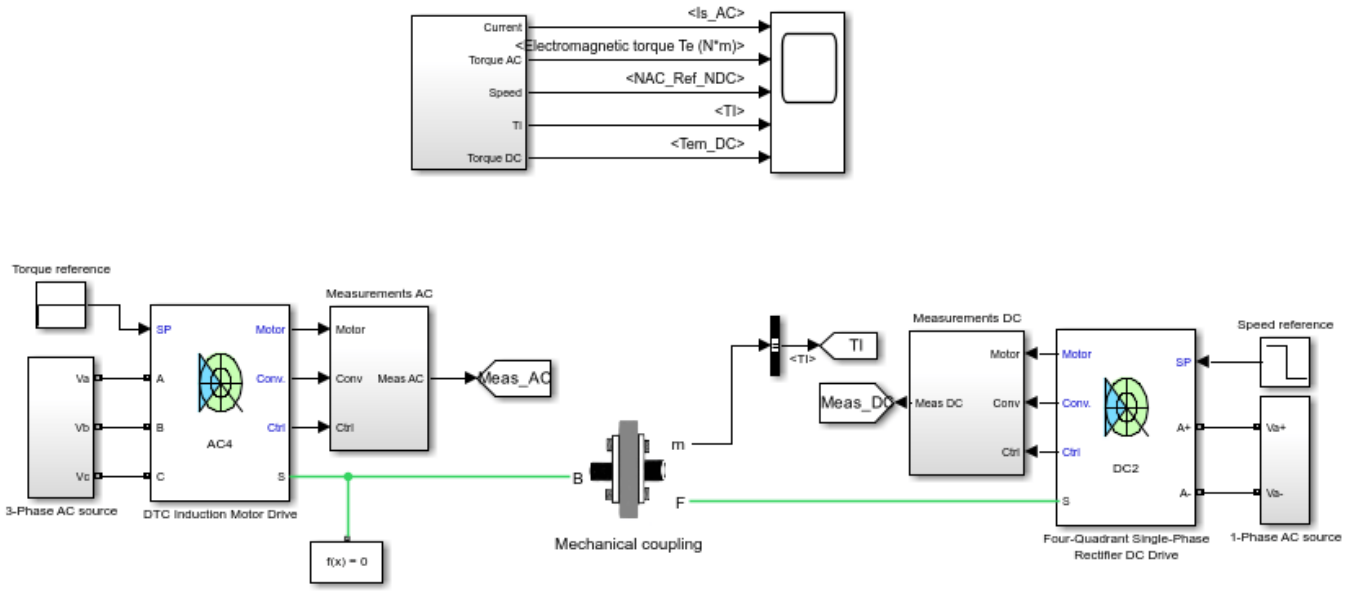


# Mechanical Coupling of Two Motor Drives II

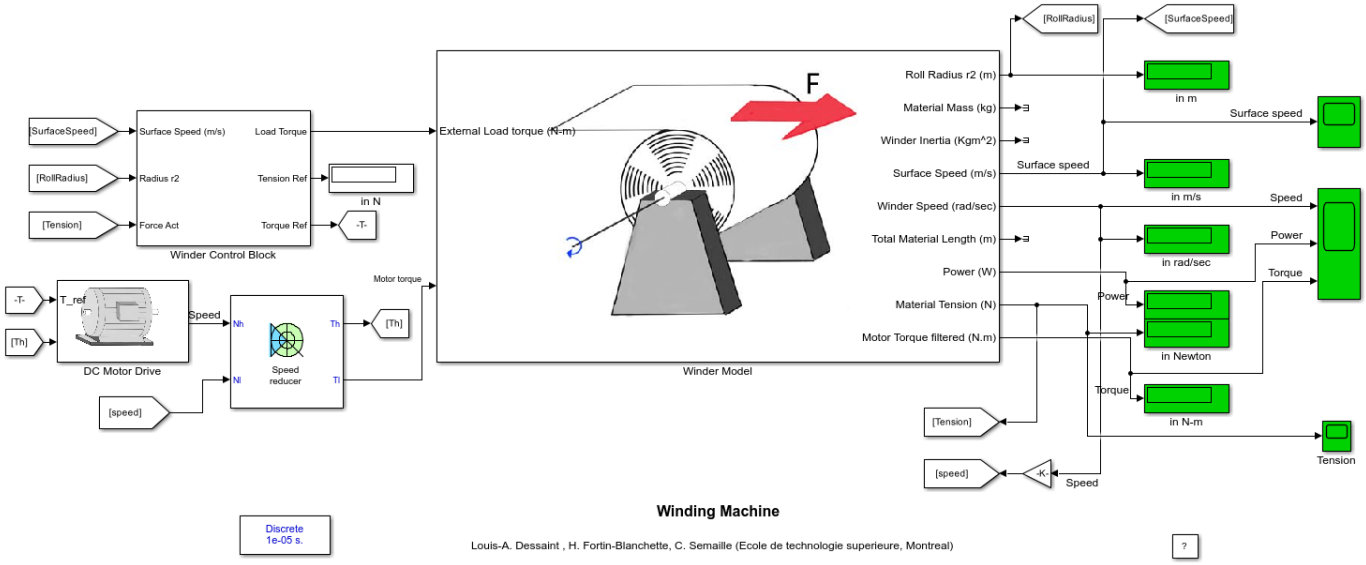
This example shows mechanical coupling of two motor drives.

Louis-A. Dessaint , H. Fortin-Blanchette, C. Semaille (Ecole de technologie superieure, Montreal)



# Winding Machine

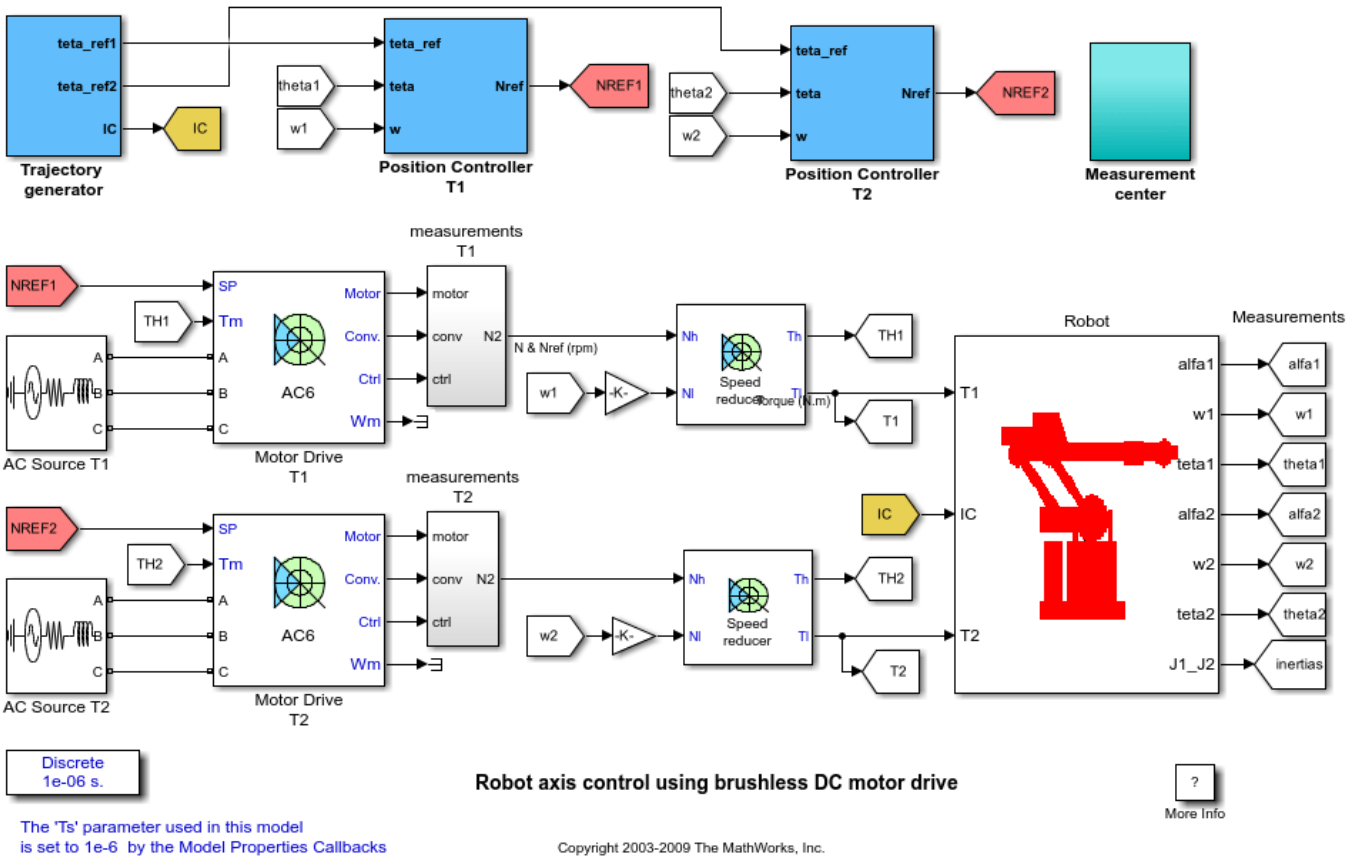
This example shows a model of a winding machine.



# Robot Axis Control Using Brushless DC Motor Drives

This example shows robot axis control with brushless DC motor drive.

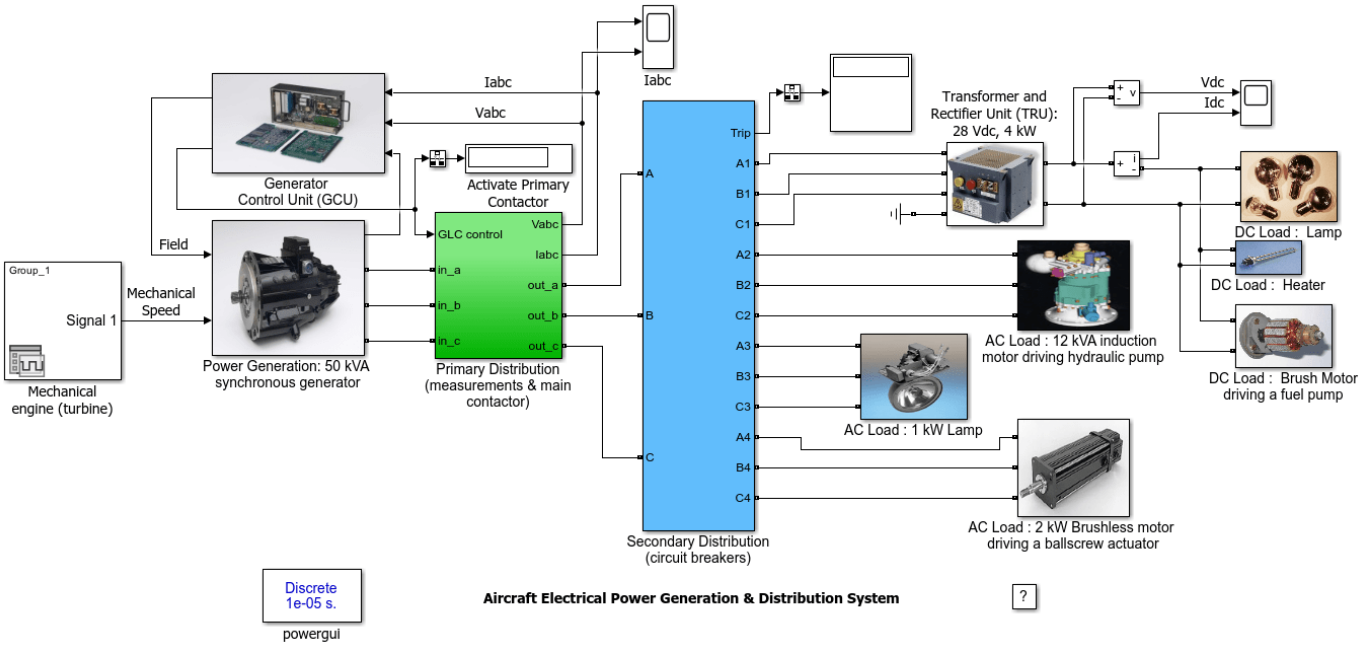
H. Le-Huy Laval University



# Aircraft Electrical Power Generation and Distribution

This example shows an aircraft electrical power generation and distribution system. The AC power frequency is variable and depends of the engine speed

Olivier Tremblay, Louis-A. Dessaint (Ecole de technologie superieure, Montreal)




## Description

The system is composed of six main sections.

The first section represents the generator mechanical drive and is modeled by a simple signal builder, which provides the mechanical speed of the engine shaft.

The second section represents the power AC generator. It is composed of a modified version of the simplified synchronous machine. The mechanical input of the modified machine of 50 kW is the engine speed. The Generator Control Unit regulates the voltage of the generator to 200 volts line to line.

The third section represents the Primary Distribution system. It is composed of three current and voltage sensors. There is also a 3-phase contactor controlled by the Generator Control Unit. Finally, a parasitic resistive load is required to avoid numerical oscillations.

The fourth section represents the secondary Power Distribution system. It is represented by 4 circuit breakers with adjustable current trip.

The fifth section represents the AC loads. There is a 4 kW Transformer And Rectifier Unit (which supplies 28 Vdc), a 12 kW induction machine (motor driving a pump), a 1 kW resistive load (lamps) and a 3 hp simplified (using an average value inverter) brushless DC drive (motor driving a ballscrew actuator).

Finally, the last section represents the DC loads. There are two resistive loads (heater and lamp) and a 300 W DC brush motor (motor driving a fuel pump).

### Simulation

Start the simulation. You can observe the AC and the DC measurements at the top level of the schematic. There are also scopes inside the AC Power Generation, the Induction motor, the Brushless DC motor and the DC brush motor.

At time t = 0 s, the engine accelerates from 0 rpm to 12000 rpm in 0.4 second.

At time t = 0.3 s, the speed reaches the threshold of 9000 rpm. The Generator Control Unit activates the primary contactor, which enables the AC power on the aircraft. All the resistive loads are now online. The DC bus voltage increases to 28 Vdc. The induction machine and the DC brush motor accelerate to nominal speed (the two mechanical loads are proportional to the speed of the motors).

At time t = 1.4 s, the Brushless DC motor starts accelerating to the set point of 500 rpm. Observe that the speed follows precisely the acceleration ramp.

At time t = 1.9 s, the speed set point of the Brushless drive is changed to -500 rpm. You can see that the main current decreases, because the motor acts as a generator. The current flows from the motor to the breaking chopper inside the drive. Open the drive scope to observe the increasing of the DC bus voltage to the breaking chopper activation voltage (290 Vdc).

At time t = 2 s, the engine speed decelerates from 12 000 rpm to 10 000 rpm in 1 second. Observe the speed of the induction machine, which decelerates according to the reduction of the generator frequency.

At time t = 3 s, the engine speed accelerates from 10 000 rpm to 18 000 rpm in 1.5 second.

At time t = 3.5 s, there is a manual breaker trip on the Transformer And Rectifier Unit. This cause a reduction of the main current. Observe the voltage and the current on the DC monitoring scope.

At time t = 4.5 s, the engine speed decelerates from 18 000 rpm to 0 rpm in 1.5 second.

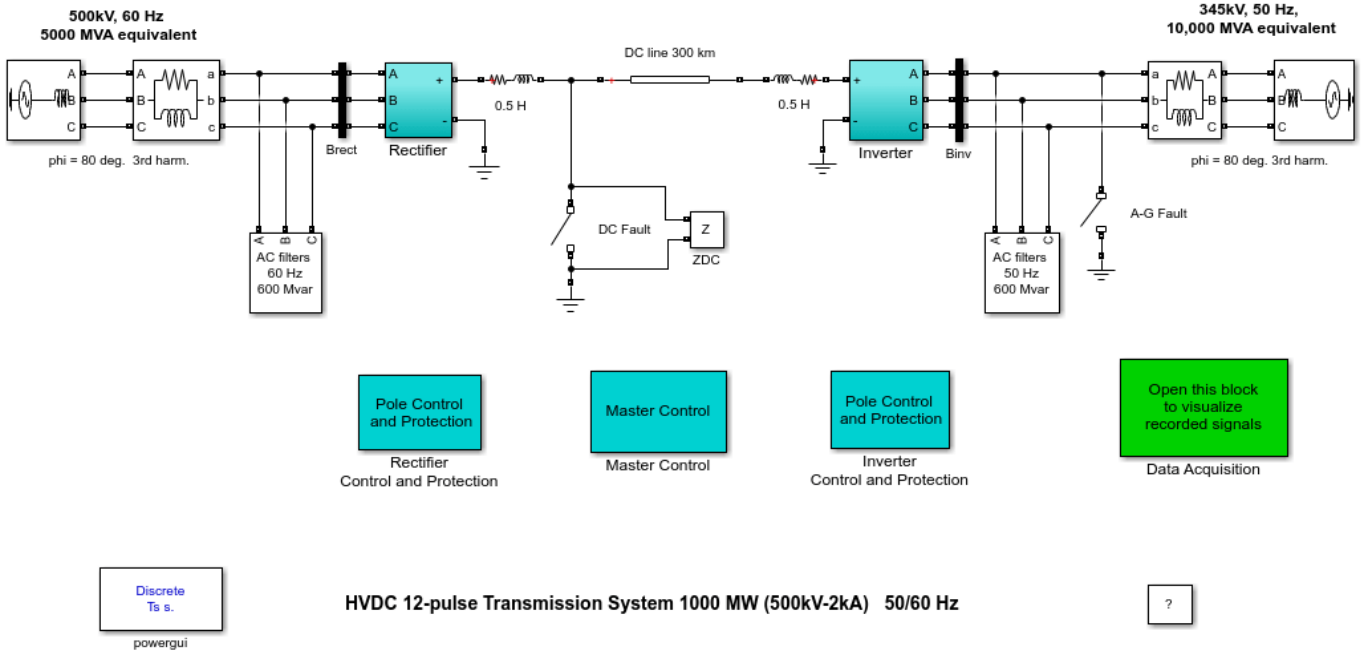
At time t = 5.26 s, the speed reaches the threshold of 8900 rpm. So the GCU de-activates the primary contactor.

Finally, note how well the AC voltage is regulated during the whole simulation period.

# Thyristor-Based HVDC Transmission System (Detailed Model)

This example shows the steady-state and transient performance of a 12-pulse, 1000 MW (500 kV-2kA) 50/60 Hz HVDC transmission system.

Silvano Casoria (Hydro-Quebec)




## Description

A 1000 MW (500 kV, 2kA) DC interconnection is used to transmit power from a 500 kV, 5000 MVA, 60 Hz network to a 345 kV, 10 000 MVA, 50 Hz network.

The rectifier and the inverter are 12-pulse converters using two 6-pulse thyristor bridges connected in series. The rectifier and the inverter are interconnected through a 300 km distributed parameter line and two 0.5 H smoothing reactors. The transformer tap changers are not simulated and fixed taps are assumed. Open the two transformer blocks in the Rectifier and Inverter subsystems to see the factors applied on the primary voltage: 0.90 on rectifier side and 0.96 on inverter side. Reactive power required by the converters is provided by a set of capacitor banks plus 11th, 13th and high pass filters for a total of 600 Mvar on each side. Two circuit breakers are used to apply faults on the inverter AC side and rectifier DC side.

DC Protection functions are implemented in each converter. At the rectifier the DC fault protection will detect and force the delay angle into the inverter region so to extinguish the fault current. At the inverter the commutation failure prevention control will detect AC faults and reduce the maximum delay angle limit in order to decrease the risk of commutation failure. The Low AC voltage detection blocks will lock the DC fault protection when a drop in the AC voltage is detected. The Master Control block initiates the starting and stopping of the converters as well as the ramping up and down of the current references. The power system and the control system are both discretized for a sample time Ts=50 us. Notice that the "Model initialization" function of the model automatically sets Ts = 50e-6 in

your MATLAB® workspace. A description of the control systems is provided in the HVDC Transmission System Case Study of the User's Manual.

**Simulation**

The system is programmed to start and reach a steady state. Then steps are applied on the reference current of the rectifier and on the inverter reference voltage in order to observe the dynamic response of the regulators. Finally, a stop sequence is initiated to bring the DC power down before blocking the converters. Start the simulation. Open the RECTIFIER and INVERTER scopes (in the Data Acquisition subsystem) and observe the DC line voltage on trace 1 (1pu = 500 kV) and the DC line current (reference and measured values) on trace 2 (1pu = 2kA).

**Start-up and Stop**

In the Master Control, the converters are deblocked and started by ramping the rectifier and inverter reference current. At t = 0.02 s (i.e. when the converters at deblocked), the reference current is ramped to reach the minimum value of 0.1 pu in 0.3 s (0.33 pu/s). At the end of this first ramp (t = 0.32 s) the DC line is charged at its nominal voltage and DC voltage reaches steady-state. At t= 0.4 s, the reference current is ramped from 0.1 pu to 1 pu (2kA) in 0.18 s (5 pu/s). At the end of this starting sequence (t=0.58 s) , the DC current reaches steady state. The RECTIFIER then controls the current and the INVERTER controls the voltage. In steady-state, the alpha firing angles (trace 3) are 16.5 degrees and 143 degrees respectively on the RECTIFIER and INVERTER sides. The extinction angle gamma (minimum value) is measured at the INVERTER and shown in trace 4. In steady-state, the minimum value is between 22 and 24 degrees. The control mode of operation (an integer between 0 to 6) is shown in trace 4 (0= blocked; 1=Current control; 2=Voltage control; 3=Alpha minimum limitation; 4=Alpha maximum limitation; 5=Forced or constant alpha; 6=Gamma control). At t = 1.4 s the Stop sequence is initiated by ramping down the current to 0.1 pu. At t = 1.6 s a Forced-alpha at the Rectifier extinguishes the current and at the Inverter the Forced-alpha brings down the DC voltage. At t = 1.7 s the pulses are blocked in both converters.

**Step response of current and voltage regulators**

Verify in the Master Control that the "Enable Ref. Current Step" switch is in the upper position. This switch is used to apply a step on the reference voltage. Also verify that the reference voltage step is enabled in the Inverter Control. At t=0.7 s, a -0.2 pu step is first applied on the reference current (decrease from 1 pu to 0.8 pu ) and at t=0.8 s, the reference current is reset to its 1 pu original value.The current stabilizes in approximately 0.1 seconds. Steps are also applied on the reference voltage of the inverter (-0.1 pu / +0.1 pu at t=1.0 s / 1.1s).

**DC line fault at the rectifier**

Deactivate the steps applied on the current reference and on the voltage reference in the Master Control and in the inverter control respectively by setting the switches in lower position. In the DC Fault block, change to 1 the 100 multiplication factor in the Switching times so that a fault is now applied at t = 0.7 s. Reduce the Simulation stop time from 2 to 1.4 s. The DC Fault protection (DCPROT) in the rectifier is activated by default. Open the FAULT scope to observe the DC fault current. Restart the simulation.

At fault application the DC current quickly increases to 2.3 pu and the DC voltage falls to zero at the rectifier. This DC voltages drop is seen by the Voltage Dependent Current Order Limiter (VDCOL) which reduces the reference current to 0.3 pu at the rectifier. A DC current still continues to circulate in the fault. Then, at t = 0.77 s, the rectifier alpha firing angle is forced to 166 degrees by the DC protection because a DC voltage drop is detected (VdL< 0.5 pu for more than 70 ms). The rectifier now operates in inverter mode. The DC line voltage becomes negative and the energy stored in the

line is returned to the AC network, causing rapid extinction of the fault current at its next zero-crossing. Then, alpha is released at t = 0.87 s and the normal DC voltage and current recover in approximately 0.4 s.

**AC line-to-ground fault at the inverter**

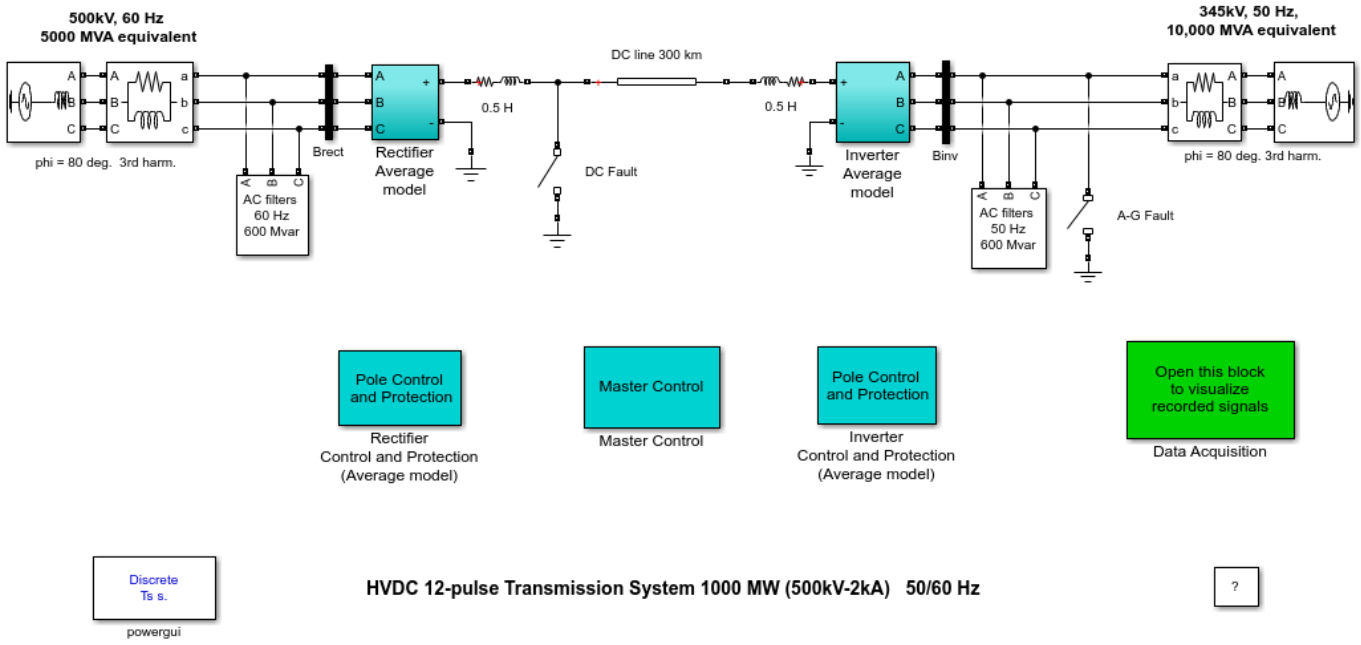
In the DC Fault block, change the multiplication factor of 1 in the Switching times to 100, so that the DC fault is now eliminated. In the A-G Fault block, change to 1 the 100 multiplication factor in the Switching times so that a 6 cycles line-to-ground fault is now applied at t = 0.7 s. The Low AC voltage detection (LACVD) subsystem in the rectifier and inverter protections and the Commutation Failure Prevention Control (CFPREV) in the inverter protection are activated by default. Restart the simulation.

Notice the 120 Hz oscillations in the DC voltage and currents during the fault. When the fault is cleared at t = 0.8 s, the VDCOL operates and reduces the reference current to 0.3 pu. The system recovers in approximately 0.35 s after fault clearing. The LACVD detects the fault and locks the DC Fault protection that should not detect a DC fault even if the DC line voltage dips. Look at the CFPREV output (A_min_I) which decreases the maximum delay angle limit in order to increase the commutation margin during and after the fault. Now deactivate the CFPREV protection by deselecting the "ON State" in the CFPREV dialog box. Restart the simulation and observe the difference in recovery time of the DC transmission. Note that a commutation failure now occurs during the recovery. A commutation failure is the result of a failure of the incoming valve to take over the direct current before the commutation voltage reverses its polarity. The symptoms are a zero DC voltage across the affected bridge causing an increase of the DC current at a rate determined mainly by the DC circuit inductance.

# Thyristor-Based HVDC Transmission System (Average Model)

This example shows the steady-state and transient performance of a 12-pulse, 1000 MW (500 kV-2kA) 50/60 Hz HVDC transmission system.

Silvano Casoria (Hydro-Quebec)




### Detailed Versus Average Model

When modeling line-commutated converter based HVDC systems you can use two types of model, depending on the range of frequencies to be represented:

**The detailed model** such as the one presented in the power_hvdc12pulse model includes detailed representation of the converter unit with its power electronic thyristor bridge and converter transformer. This model is well suited for observing harmonics and control system dynamic and transient performance.

**The average model** such as the one presented here where the converter unit (bridge and transformer) is represented by an equivalent voltage source generating the bridge average DC voltage and AC sources generating the fundamental component of the currents flowing into the network. This model does not represent harmonics, but the dynamics resulting from control system and power system interaction is preserved. This model allows using a time step that is higher than the one of the control system or the network. Optimal performance however is obtained by using the same time step of the control system regulators.

### Description

A 1000 MW (500 kV, 2kA) DC interconnection is used to transmit power from a 500 kV, 5000 MVA, 60 Hz network to a 345 kV, 10 000 MVA, 50 Hz network. The rectifier and the inverter are average models of 12-pulse converters representing two 6-pulse thyristor bridges connected in series. The

rectifier and the inverter are interconnected through a 300 km distributed parameter line and two 0.5 H smoothing reactors.

The converter transformer linear representation is part of the average model block. The transformer tap changers are not simulated and fixed taps are assumed as model inputs. At the rectifier the tap ratio (Nprimary/Nsecondary) is 0.9 (pu) and at the inverter it is 0.96 (pu). Reactive power required by the converters is provided by a set of capacitor banks plus 11th, 13th and high pass filters for a total of 600 Mvar on each side. Note that since no harmonics are generated by the average converter model the Mvar could be provided entirely by capacitor banks.

Two circuit breakers are used to apply faults on the inverter AC side and rectifier DC side. Note that since thyristor valves are not present the 12-pulse Firing Control block is no longer necessary in the average model.

DC Protection functions are implemented in each converter. At the rectifier, the DC fault protection will detect and force the delay angle into the inverter region so to extinguish the fault current. At the inverter, the commutation failure prevention control will detect AC faults and reduce the maximum delay angle limit in order to decrease the risk of commutation failure. Note that the commutation failure phenomenon is not possible in the average model.

To assist the user in identifying conditions that may produce this phenomenon an indication from the model is provided ( CF_alarm signal available from the Bus Selector block of the rectifier and inverter models). The Low AC Voltage Detection blocks will lock the DC fault protection when a drop in the AC voltage is detected. The Master Control block initiates the starting and stopping of the converters as well as the ramping up and down of the current references.

A description of the control systems is provided in the HVDC Transmission System Case Study of the User's Manual. The firing angle order (alpha_ord) output of the controller is an input of the average model.

The power system and the control system are both discretized for a sample time Ts=50 us. The "Model initialization" section of the model automatically sets Ts = 50e-6 in your MATLAB® workspace. It also sets the average model time step Ts_avg equal to Ts.

### Simulation

The system is programmed to start and reach a steady state. Then steps are applied on the reference current of the rectifier and on the inverter reference voltage in order to observe the dynamic response of the regulators. Finally, a stop sequence is initiated to bring the DC power down before blocking the converters.

Start the simulation, open the RECTIFIER and INVERTER scopes (in the Data Acquisition subsystem) and observe the DC line voltage on trace 1 (1pu = 500 kV) and the DC line current (reference and measured values) on trace 2 (1pu = 2 kA).

### Start-up and Stop

In the Master Control, the converters are unblocked and started by ramping the rectifier and inverter reference current.

At t = 0.02 s (i.e. when the converters at unblocked), the reference current is ramped to reach the minimum value of 0.1 pu in 0.3 s (0.33 pu/s). At the end of this first ramp (t = 0.32 s) the DC line is charged at its nominal voltage and DC voltage reaches steady-state.

At t= 0.4 s, the reference current is ramped from 0.1 pu to 1 pu (2kA) in 0.18 s (5 pu/s). At the end of this starting sequence (t=0.58 s), the DC current reaches steady state. The RECTIFIER then controls the current and the INVERTER controls the voltage.

In steady-state, the alpha firing angles (trace 3) are 17.7 degrees and 144.5 degrees respectively on the RECTIFIER and INVERTER sides. Note that in the detailed model these traces (16.5 degrees for the rectifier and 143 degrees for the inverter) are not the measured firing delay angles but the corresponding orders from the control regulators. In the detailed model, the firing angles are smaller because the regulators must advance the firing orders by two time steps in order to compensate for the delays introduced by interfacing of input AC voltages and output firing pulses of the 12-pulse Firing Control block. The extinction angle gamma value is an output of the average model. It is used at the INVERTER and shown in trace 5. In steady-state, its value is 23 degrees.

The control mode of operation (an integer between 0 to 6) is shown in trace 4 (0= blocked; 1=Current control; 2=Voltage control; 3=Alpha minimum limitation; 4=Alpha maximum limitation; 5=Forced or constant alpha; 6=Gamma control).

At t = 1.4 s the Stop sequence is initiated by ramping down the current to 0.1 pu.

At t = 1.6 s a Forced-alpha at the Rectifier extinguishes the current and at the Inverter the Forced-alpha brings down the DC voltage.

At t = 1.7 s the pulses are blocked in both converters.

**Step response of current and voltage regulators**

Verify in the Master Control that the "Enable Ref. Current Step" switch is in the upper position. This switch is used to apply a step on the reference voltage. Also verify that the Ref. Voltage Step is enabled in the Inverter Control. Start the simulation.

At t=0.7 s, a -0.2 pu step is first applied on the reference current (decrease from 1 pu to 0.8 pu ) and at t=0.8 s, the reference current is reset to its 1 pu original value. The current stabilizes in approximately 0.1 seconds. Steps are also applied on the reference voltage of the inverter (-0.1 pu / +0.1 pu at t=1.0 s / 1.1s).

**DC line fault at the rectifier**

Deactivate the steps applied on the current reference and on the voltage reference in the Master Control and in the inverter control respectively by setting the switches in lower position.

The DC Fault protection (DCPROT) in the rectifier is activated by default. In the DC Fault block, change to 1 the 100 multiplication factor in the Switching times so that a fault is now applied at t = 0.7 s.

Reduce the Simulation stop time from 2 to 1.4 s. Open the FAULT scope to observe the DC fault current. Restart the simulation.

At fault application the DC current quickly increases to 2.63 pu and the DC voltage falls to zero at the rectifier. This DC voltages drop is seen by the Voltage Dependent Current Order Limiter (VDCOL) which reduces the reference current to 0.3 pu at the rectifier. A DC current still continues to circulate in the fault.

At t = 0.77 s, the rectifier alpha firing angle is forced to 166 degrees by the DC protection because a DC voltage drop is detected (VdL< 0.5 pu for more than 70 ms). The rectifier now operates in

inverter mode. The DC line voltage becomes negative and the energy stored in the line is returned to the AC network, causing rapid extinction of the fault current at its next zero-crossing.

Alpha is released at t = 0.82 s and the normal DC voltage and current recover in approximately 0.4 s.

**AC line-to-ground fault at the inverter**

In the DC Fault block, change the multiplication factor of 1 in the Switching times to 100, so that the DC fault is now disabled. In the A-G Fault block, change to 1 the 100 multiplication factor in the Switching times so that a 6 cycles line-to-ground fault is now applied at t = 0.7 s.

The Low AC voltage detection (LACVD) subsystem in the rectifier and inverter protections and the Commutation Failure Prevention Control (CFPREV) in the inverter protection are activated by default.

Restart the simulation.

Note that the 120 Hz oscillations in the DC voltage and currents which are normally observed during single-phase fault with a detailed model do not exist in the average model. Indeed, only positive-sequence fundamental components of AC quantities are significant in the average model. The VDCOL would not operate during the fault. The system recovers in approximately 0.2 s after fault clearing (see the measured DC power Pd).

Abnormal inverter operation resulting from a commutation failure malfunction (CF) due to AC faults are not correctly represented by the average model equations. To assist the user in identifying such a condition an alarm signal (CF_alarm) is set whenever the onset of a CF is predicted.

Look at the CF_alarm signal at the inverter, triggered at t = 0.73 s. Open the CF_alarm block (inside the HVDC_CONV_AVG block of the inverter model) to examine the logic.

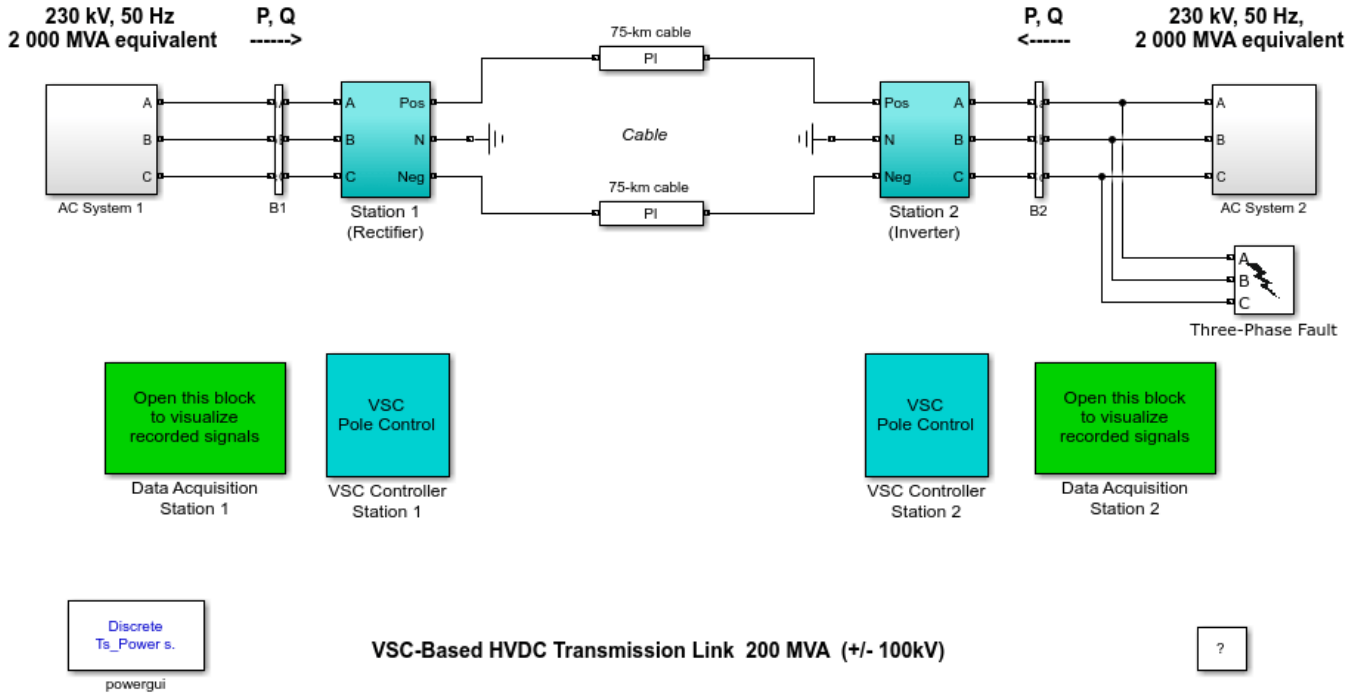
Look at the A_min_I signal of the PROTECTION INVERTER scope. This signal monitors the Commutation Failure Prevention (CFPREV) output of the Inverter Protection block. The A_min signal is used to decrease the delay angle limit in order to increase the commutation margin during and after the fault.

Finally, deactivate the CFPREV protection by deselecting the "ON State" in the CFPREV dialog box. Restart the simulation and observe the difference in recovery time of the DC transmission.

# VSC-Based HVDC Transmission System (Detailed Model)

This example shows a VSC-based HVDC transmission link 200 MVA (+/- 100kV).

Silvano Casoria (Hydro-Quebec)




### Description

A 200 MVA (+/- 100 kV DC) forced-commutated Voltage-Sourced Converter (VSC) interconnection is used to transmit power from a 230 kV, 2000 MVA, 50 Hz system to another identical AC system. The rectifier and the inverter are three-level Neutral Point Clamped (NPC) VSC converters using close IGBT/Diodes. The Sinusoidal Pulse Width Modulation (SPWM) switching uses a single-phase triangular carrier wave with a frequency of 27 times fundamental frequency (1350 Hz). Along with the converters, the station includes on the AC side: the step down Yg-D transformer, the AC filters, the converter reactor; and on the DC side: the capacitors, the DC filters. The transformer tap changers and saturation characteristics are not simulated. The 40 Mvar shunt AC filters are 27th and 54th high-pass tuned around the two dominating harmonics. The 0.15 p.u. converter reactor with the 0.15 p.u. transformer leakage reactance permits the VSC output voltage to shift in phase and amplitude with respect to the AC system Point of Common Coupling (PCC) (bus B1 for station 1 and B2 for station 2) and allows control of converter active and reactive power output. The reservoir DC capacitors are connected to the VSC terminals. They have an influence on the system dynamics and the voltage ripple on the DC side. The high-frequency blocking filters are tuned to the 3rd harmonic, i.e. the main harmonic present in the positive and negative pole voltages. The rectifier and the inverter are interconnected through a 75 km cable (i.e. 2 pi sections) and two 8 mH smoothing reactors. A circuit breaker is used to apply a three-phase to ground fault on the inverter AC side. A Three-Phase Programmable Voltage Source block is used in station 1 system to apply voltage sags.

The discrete control system generates the three sinusoidal modulating signals that are the reference value of the bridge phase voltages. The amplitude and phase of the modulating signals can be calculated to control either: the reactive and real AC power flow at the PCC, or the reactive power flow at the PCC and the pole to pole DC voltage. It would also be possible to control the AC voltage amplitude at the PCC, but this option is not included in our model. A description of the control system is provided in the "VSC-Based HVDC Link" case study of the User's Manual. The power system and the control system are both discretized for a sample time Ts_Power=7.406e-6 s and Ts_control=74.06e-6 s respectively. They are multiples of the carrier period. Notice that the "Model initialization" function of the model automatically sets these two sample times in your MATLAB® workspace.

**Simulation**

Two simulations will permit to examine the system response to:

1) Steps on the regulators references, and

2) Minor and severe perturbations on the AC sides.

**Steady-state - Step response of power (P & Q) and DC voltage regulators**

The system is programmed to start and reach a steady state. Steps are then applied sequentially on: the reference active and reactive power of the rectifier; the reference DC voltage of the inverter. The dynamic response of the regulators is observed. Start the simulation. Open the BUS B1 STATION_1 and DC_SIDE_STATION_2 scopes (in the respective Data Acquisition subsystems). Examine in station 1: the active power on trace 2 (1 p.u. = 200 MW) and the reactive power (reference and measured values) on trace 3 (1 p.u. =200 Mvar); in station 2: the DC voltage (reference and measured values) on trace 2 (1 p.u. = 200 kV).

At t = 1.5 s, a -0.1 p.u. step is first applied to the reference active power (decrease from 1 p.u. to 0.9 pu). The power stabilizes in approximately 0.3 seconds. Steps are also applied to the reference reactive power of the rectifier (from 0 to -0.1 p.u.) at t = 2.0 s and on the reference DC voltage of the inverter (decrease from 1 p.u. to 0.95 p.u.) at t = 2.5 s. Note the regulators dynamics and how they are more or less mutually affected. The control design attempts to decouple the active and reactive power responses.

**AC side perturbations**

Deactivate the steps applied on the three references by changing the multiplication factors to 100 in the Step times. In the "Three-Phase Programmable Voltage Source" inside AC system 1 subsystem, change the Time variation setting to "Amplitude". Check that the source is now programmed for a step of -0.1 p.u on voltage magnitude at t = 1.5 s, for a duration of 7 cycles. In the "Three-Phase Fault" block change to 1 the multiplication factor in the Transition times. A 6 cycles three-phase fault will be applied at t = 2.1 s in station 2 PCC (Bus B2). Restart the simulation.

After the AC voltage sag in station 1, the active and reactive power deviation from the pre-disturbance is less than 0.09 p.u. and 0.2 p.u. respectively. The recovery time is less than 0.3 s and steady state is reached again. A second perturbation follows. During the severe three-phase fault at station 2, the transmitted DC power is almost halted and the DC voltage tends to increase (1.2 p.u.) since the DC side capacitance is being excessively charged. A special function (DC Voltage Control Override) in the Active Power Control (in station 1) attempts to limit the DC voltage within a fixed range (see the controller mask). The system recovers well after the fault within 0.5 s. You can observe overshoot in the active power (1.33 p.u. in station 1) and damped oscillations (around 10 Hz) in the reactive power.

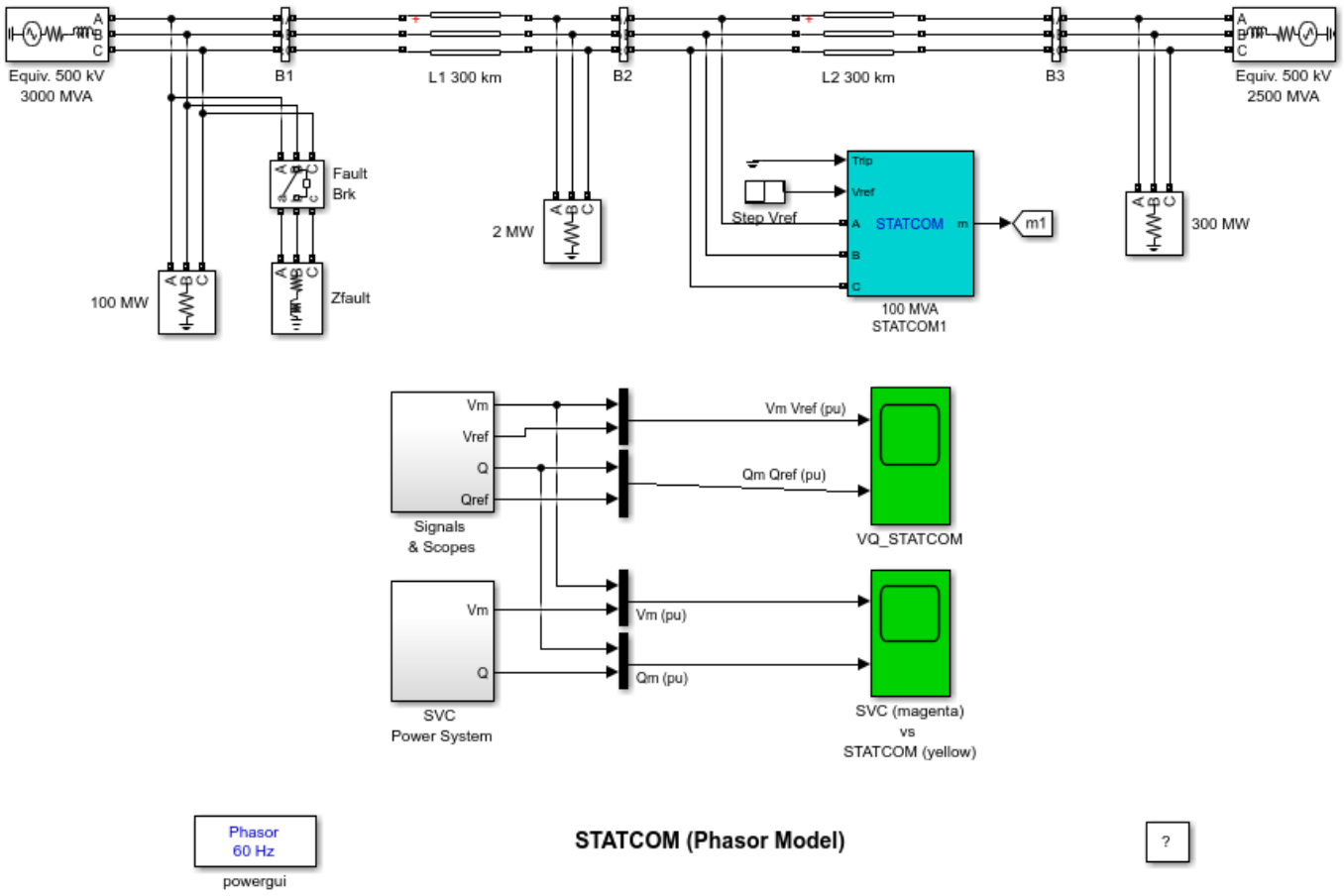
### Impact of the DC voltage balance control

Open the VOLTAGE_BALANCE_CONTROL_STATION_2 scope. Finally, open the control dialog box in Station 2 and verify that the DC Voltage Balance box is activated. The DC voltage balance control objective is to minimize the voltage unbalance (Udc_0_mean signal = sum of the positive and negative pole voltages). A way of producing an unbalance is to use unequal capacitance values in the positive and negative poles (for example, Cp divided by 2). Observe the Udc_0_mean signal first with the DC balance control activated and then deactivated. Note that this function response is relatively slow.

# STATCOM (Phasor Model)

This example shows a Static Synchronous Compensator (STATCOM) used for midpoint voltage regulation on a 500-kV transmission line.

Pierre Giroux and Gibert Sybille (Hydro-Quebec)




## Description

The Static Synchronous Compensator (STATCOM) is one of the key FACTS devices. Based on a voltage-sourced converter, the STATCOM regulates system voltage by absorbing or generating reactive power. Contrary to a thyristor-based Static Var Compensator (SVC), STATCOM output current (inductive or capacitive) can be controlled independent of the AC system voltage.

The power grid consists of two 500-kV equivalents (respectively 3000 MVA and 2500 MVA) connected by a 600-km transmission line. When the STATCOM is not in operation, the "natural" power flow on the transmission line is 930 MW from bus B1 to B3. The STATCOM is located at the midpoint of the line (bus B2) and has a rating of +/- 100MVA. This STATCOM is a phasor model of a typical three-level PWM STATCOM. If you open the STATCOM dialog box and select "Display Power data", you will see that our model represents a STATCOM having a DC link nominal voltage of 40 kV with an equivalent capacitance of 375 uF. On the AC side, its total equivalent impedance is 0.22 pu on 100

MVA. This impedance represents the transformer leakage reactance and the phase reactor of the IGBT bridge of an actual PWM STATCOM.

### Simulation

#### 1. STATCOM Dynamic Response

We will now verify the dynamic response of our model. Open the STATCOM dialog box and select "Display Control parameters". Verify that the "Mode of operation" is set to "Voltage regulation" and that "External control of reference voltage Vref" is selected. Also, the "droop" parameter should be set to 0.03 and the "Vac Regulator Gains" to 5 (proportional gain Kp) and 1000 (integral gain Ki). Close the STATCOM dialog block and open the "Step Vref" block (the red timer block connected to the "Vref" input of the STATCOM). This block should be programmed to modify the reference voltage Vref as follows: Initially Vref is set to 1 pu; at t=0.2 s, Vref is decreased to 0.97 pu; then at t=0.4 s, Vref is increased to 1.03; and finally at 0.6 s, Vref is set back to 1 pu. Also, make sure that the fault breaker at bus B1will not operate during the simulation (the parameters "Switching of phase A, B and C" should not be selected).

Run the simulation and look at the "VQ_STATCOM" scope. The first graph displays the Vref signal (magenta trace) along with the measured positive-sequence voltage Vm at the STATCOM bus (yellow trace). The second graph displays the reactive power Qm (yellow trace) absorbed (positive value) or generated (negative value) by the STATCOM. The signal Qref (magenta trace) is not relevant to our simulation because the STATCOM is in "Voltage regulation" and not in "Var Control".

Looking at the Qm signal we can determine that the closed-loop time constant of the system is about 20 ms. This time constant depends primarily on the power system strength at bus B2 and on the programmed Vac Regulator gains of the STATCOM. To see the impact of the regulator gains, multiply the two gains of the Vac Regulator Gains by two and rerun the simulation. You should observe a much faster response with a small overshoot.

Looking at the Vm and Vref signals, you can see that the STATCOM does not operate as a perfect voltage regulator (Vm does not follow exactly the reference voltage Vref). This is due to the regulator droop (regulating slope) of 0.03 pu. For a given maximum capacitive/inductive range, this droop is used to extend the linear operating range of the STATCOM and also to ensure automatic load sharing with other voltage compensators (if any). Set the droop parameter to 0 and the voltage regulator gains back to 5 (Kp) and 1000 (Ki). If you then run a simulation, you will see that the measured voltage Vm now follows perfectly the reference voltage Vref.

#### 2. STATCOM compared to a SVC under fault condition

We will now compare our STATCOM model with a SVC model having the same rating (+/- 100 MVA). If you double-click on the "SVC Power System" (the magenta block), you will see a SVC connected to a power grid similar to the power grid on which our STATCOM is connected. A remote fault will be simulated on both systems using a fault breaker in series with a fault impedance. The value of the fault impedance has been programmed to produce a 30% voltage sag at bus B2. Before running the simulation, you will first disable the "Step Vref" block by multiplying the time vector by 100. You will then program the fault breaker by selecting the parameters "Switching of phase A, B and C" and verify that the breaker is programmed (look at the "Transition times" parameter) to operate at t=0.2 s for a duration of 10 cycles. Check also that the fault breaker inside the "SVC Power System" has the same parameters. Finally, set the STATCOM droop back to its original value (0.03 pu).

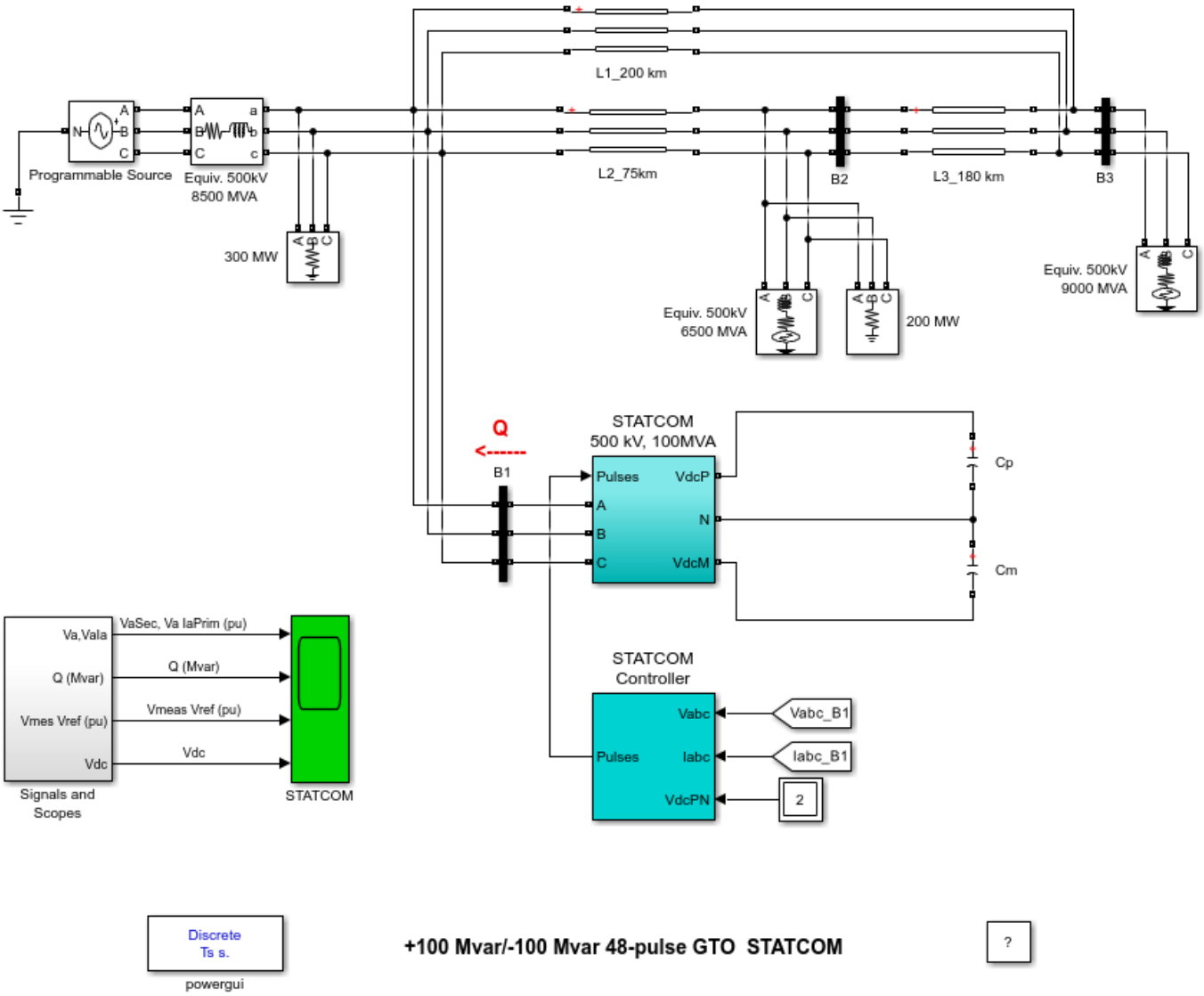
Run the simulation and look at the "SVC vs STATCOM" scope. The first graph displays the measured voltage Vm on both systems (magenta trace for the SVC). The second graph displays the measured reactive power Qm generated by the SVC (magenta trace) and the STATCOM (yellow trace). During

the 10-cycle fault, a key difference between the SVC and the STATCOM can be observed. The reactive power generated by the SVC is -0.48 pu and the reactive power generated by the STATCOM is -0.71 pu. We can then see that the maximum capacitive power generated by a SVC is proportional to the square of the system voltage (constant susceptance) while the maximum capacitive power generated by a STATCOM decreases linearly with voltage decrease (constant current). This ability to provide more capacitive power during a fault is one important advantage of the STATCOM over the SVC. In addition, the STATCOM will normally exhibits a faster response than the SVC because with the voltage-sourced converter, the STATCOM has no delay associated with the thyristor firing (in the order of 4 ms for a SVC).

# STATCOM (Detailed Model)

This example shows the operation of a +100 Mvar/-100 Mvar 48-pulse GTO STATCOM.

P. Giroux ; G. Sybille (Hydro-Quebec)




## Description

A 100-Mvar STATCOM regulates voltage on a three-bus 500-kV system. The 48-pulse STATCOM uses a Voltage-Sourced Converter (VSC) built of four 12-pulse three-level GTO inverters. Look inside the STATCOM block to see how the VSC inverter is built. The four sets of three-phase voltages obtained at the output of the four three-level inverters are applied to the secondary windings of four phase-shifting transformers (-15 deg., -7.5 deg., 7.5 deg., +7.5 deg. phase shifts). The fundamental components of voltages obtained on the 500 kV side of the transformers are added in phase by the

serial connection of primary windings. Please refer to the "power_48pulsegtoconverter" example to get details on the operation of the VSC.

During steady-state operation the STATCOM control system keeps the fundamental component of the VSC voltage in phase with the system voltage. If the voltage generated by the VSC is higher (or lower) than the system voltage, the STATCOM generates (or absorbs) reactive power. The amount of reactive power depends on the VSC voltage magnitude and on the transformer leakage reactances. The fundamental component of VSC voltage is controlled by varying the DC bus voltage. In order to vary the DC voltage, and therefore the reactive power, the VSC voltage angle (alpha) which is normally kept close to zero is temporarily phase shifted. This VSC voltage lag or lead produces a temporary flow of active power which results in an increase or decrease of capacitor voltages.

One of the three voltage sources used in the 500 kV system equivalents can be varied in order to observe the STATCOM dynamic response to changes in system voltage. Open the "Programmable Voltage Source" menu and look at the sequence of voltage steps which are programmed.

### Simulation

#### Dynamic response of the STATCOM

Run the simulation and observe waveforms on the STATCOM scope block. The STATCOM is in voltage control mode and its reference voltage is set to Vref=1.0 pu. The voltage droop of the regulator is 0.03 pu/100 VA.Therefore when the STATCOM operating point changes from fully capacitive (+100 Mvar) to fully inductive (-100 Mvar) the STATCOM voltage varies between 1-0.03=0.97 pu and 1+0.03=1.03 pu.

Initially the programmable voltage source is set at 1.0491 pu, resulting in a 1.0 pu voltage at SVC terminals when the STATCOM is out of service. As the reference voltage Vref is set to 1.0 pu, the STATCOM is initially floating (zero current). The DC voltage is 19.3 kV. At t=0.1s, voltage is suddenly decreased by 4.5 % (0.955 pu of nominal voltage). The SVC reacts by generating reactive power (Q= +70 Mvar) in order to keep voltage at 0.979 pu. The 95% settling time is approximately 47 ms. At this point the DC voltage has increased to 20.4 kV. Then, at t=0.2 s the source voltage is increased to1.045 pu of its nominal value. The SVC reacts by changing its operating point from capacitive to inductive in order to keep voltage at 1.021 pu. At this point the STATCOM absorbs 72 Mvar and the DC voltage has been lowered to 18.2 kV. Observe on the first trace showing the STATCOM primary voltage and current that the current is changing from capacitive to inductive in approximately one cycle. Finally, at t=0.3 s the source voltage in set back to its nominal value and the STATCOM operating point comes back to zero Mvar.

If you look inside the "Signals and Scopes" subsystem you will have access to other control signals. Notice the transient changes on alpha angle when the DC voltage is increased or decreased in order to vary reactive power. The steady state value of alpha (0.5 degrees) is the phase shift required to maintain a small active power flow compensating transformer and converter losses.

### How To Regenerate Initial Conditions

The initial states required to start this example in steady state have been saved in the "power_statcom_gto48p.mat" file. When you open this example, the InitFcn callback (in the Model Properties/Callbacks) automatically loads into your workspace the contents of this .mat file ("xInitial" variable).

If you modify this model, or change parameter values of power components, the initial conditions stored in the "xInitial" variable will no longer be valid and Simulink® will issue an error message. To regenerate the initial conditions for your modified model, follow the steps listed below:

1. In the Simulation/Configuration Parameters/Data Import/Export Parameters menu, uncheck the "Initial state" parameter and check the "Final states" parameter.

2. In the Programmable Voltage Source menu, disable the source voltage steps by setting the "Time variation of " parameter to "none".

3. Make sure that the Simulation Stop Time is 0.4 second. Note that in order to generate initial conditions coherent with the 60 Hz voltage source phase angles, the Stop Time must an integer number of 60 Hz cycles.

4. Start simulation. When simulation is completed, verify that steady state has been reached by looking at waveforms displayed on the scope. The final states which have been saved in the "xFinal" structure with time can be used as initial states for future simulations. Executing the next two commands copies these final conditions in "xInitial" and saves this variable in a new file (myModel_init.mat).

```
>> xInitial=xFinal;
```

```
>> save myModel_init xInitial
```

5. In the File/Model Properties/Callbacks/InitFcn window, change the line "xInitial = init_power_statcom_gto48p" to "load myModel_init.mat". Next time you open this model, the variable xInitial saved in the myModel_init.mat file will be loaded in your workspace.

6. In the Simulation/Configuration Parameters menu, check "Initial state".

7. Start simulation and verify that your model starts in steady-state.

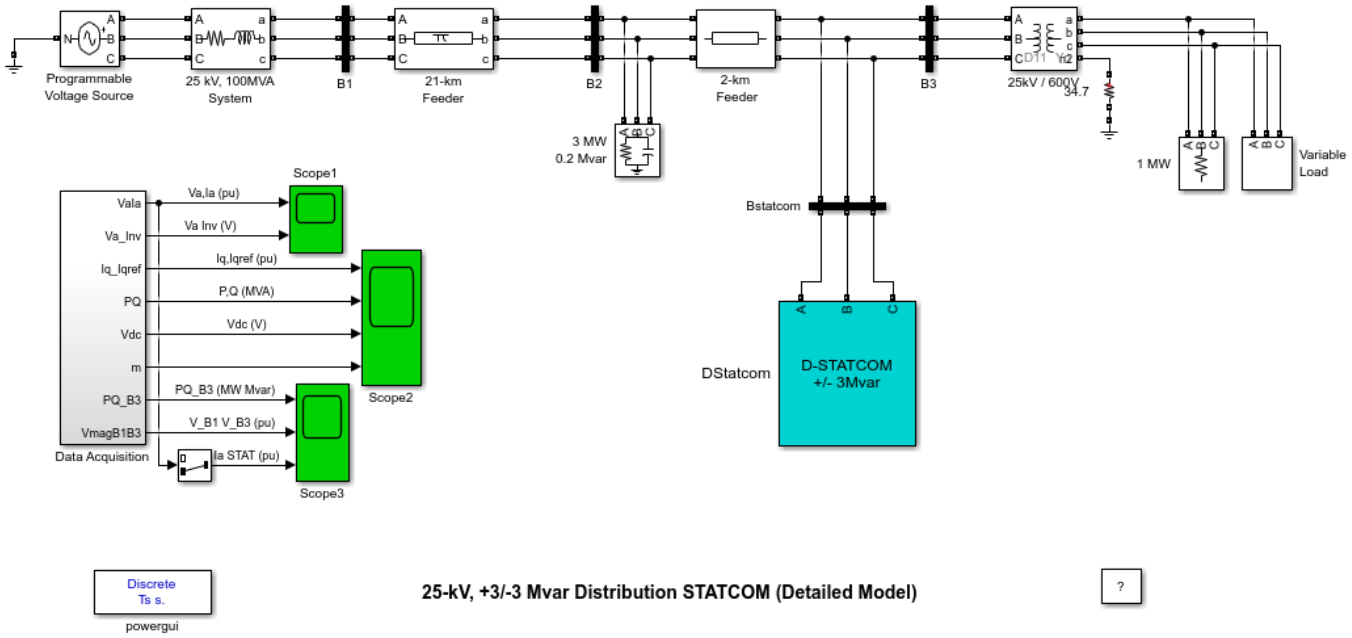
8. In the Programmable Voltage Source menu, set the "Time variation of" parameter back to "Amplitude".

9. Save your model.

# D-STATCOM (Detailed Model)

This example shows a detailed model of a distribution STATCOM.

Pierre Giroux ; Gilbert Sybille (Hydro-Quebec, IREQ)




### Detailed Versus Average Model

When modeling VSC-based energy conversion systems in Specialized Power Systems, you can use two types of models, depending on the range of frequencies to be represented : the detailed model and the average model.

**The detailed model (discrete)** such as the one presented in this example. The detailed model includes detailed representation of power electronic IGBT converters. In order to achieve an acceptable accuracy with the 1680 Hz switching frequency used in this example, the model must be discretized at a relatively small time step (5 microseconds). This model is well suited for observing harmonics and control system dynamic performance over relatively short periods of times (typically hundreds of milliseconds to one second).

**The average model (discrete)** such as the one presented in the power_dstatcom_avg model in the FACTS library of examples. In this type of model the IGBT Voltage-Sourced Converters (VSC) are represented by equivalent voltage sources generating the AC voltage averaged over one cycle of the switching frequency. This model does not represent harmonics, but the dynamics resulting from control system and power system interaction is preserved. This model allows using much larger time steps (typically 50 microseconds), thus allowing simulations of several seconds.

Alternatively, a third type of model can be used for simulating on larger time frames: the **phasor model**. This type of model is not available for the D-STATCOM , but it is available for the STATCOM, a similar device, in the power_statcom.mdl model.

### Description

A Distribution Static Synchronous Compensator (D-STATCOM) is used to regulate voltage on a 25-kV distribution network. Two feeders (21 km and 2 km) transmit power to loads connected at buses B2 and B3. A shunt capacitor is used for power factor correction at bus B2. The 600-V load connected to bus B3 through a 25kV/600V transformer represents a plant absorbing continuously changing currents, similar to an arc furnace, thus producing voltage flicker. The variable load current magnitude is modulated at a frequency of 5 Hz so that its apparent power varies approximately between 1 MVA and 5.2 MVA, while keeping a 0.9 lagging power factor. This load variation will allow you to observe the ability of the D-STATCOM to mitigate voltage flicker.

The D-STATCOM regulates bus B3 voltage by absorbing or generating reactive power. This reactive power transfer is done through the leakage reactance of the coupling transformer by generating a secondary voltage in phase with the primary voltage (network side). This voltage is provided by a voltage-sourced PWM inverter. When the secondary voltage is lower than the bus voltage, the D-STATCOM acts like an inductance absorbing reactive power. When the secondary voltage is higher than the bus voltage, the D-STATCOM acts like a capacitor generating reactive power.

The D-STATCOM consists of the following components:

- **a 25kV/1.25kV coupling transformer** which ensures coupling between the PWM inverter and the network.
- **a voltage-sourced PWM inverter** consisting of two IGBT bridges. This twin inverter configuration produces less harmonics than a single bridge, resulting in smaller filters and improved dynamic response. In this case, the inverter modulation frequency is 28*60=1.68 kHz so that the first harmonics will be around 3.36 kHz.
- **LC damped filters** connected at the inverter output. Resistances connected in series with capacitors provide a quality factor of 40 at 60 Hz.
- **a 10000-microfarad capacitor** acting as a DC voltage source for the inverter
- **a voltage regulator** that controls voltage at bus B3
- **a PWM pulse generator** using a modulation frequency of 1.68 kHz
- **anti-aliasing filters** used for voltage and current acquisition.

The D-STATCOM controller consists of several functional blocks:

- **a Phase Locked Loop (PLL)**. The PLL is synchronized to the fundamental of the transformer primary voltages.
- **two measurement systems**. Vmeas and Imeas blocks compute the d-axis and q-axis components of the voltages and currents by executing an abc-dq transformation in the synchronous reference determined by sin(wt) and cos(wt) provided by the PLL.
- **an inner current regulation loop**. This loop consists of two proportional-integral (PI) controllers that control the d-axis and q-axis currents. The controllers outputs are the Vd and Vq voltages that the PWM inverter has to generate. The Vd and Vq voltages are converted into phase voltages Va, Vb, Vc which are used to synthesize the PWM voltages. The Iq reference comes from the outer voltage regulation loop (in automatic mode) or from a reference imposed by Qref (in manual mode). The Id reference comes from the DC-link voltage regulator.
- **an outer voltage regulation loop**. In automatic mode (regulated voltage), a PI controller maintains the primary voltage equal to the reference value defined in the control system dialog box.
- **a DC voltage controller** which keeps the DC link voltage constant to its nominal value (Vdc=2.4 kV).

The electrical circuit is discretized using a sample time Ts=5 microseconds. The controller uses a larger sample time (32*Ts= 160 microseconds).

### Simulation

#### 1. D-STATCOM dynamic response

During this test, the variable load will be kept constant and you will observe the dynamic response of a D-STATCOM to step changes in source voltage. Check that the modulation of the Variable Load is not in service (Modulation Timing [Ton Toff]= [0.15 1]*100 > Simulation Stop time). The Programmable Voltage Source block is used to modulate the internal voltage of the 25-kV equivalent. The voltage is first programmed at 1.077 pu in order to keep the D-STATCOM initially floating (B3 voltage=1 pu and reference voltage Vref=1 pu). Three steps are programmed at 0.2 s, 0.3 s, and 0.4 s to successively increase the source voltage by 6%, decrease it by 6% and bring it back to its initial value (1.077 pu).

Start the simulation. Observe on Scope1 the phase A voltage and current waveforms of the D-STATCOM as well as controller signals on Scope2. After a transient lasting approximately 0.15 sec., the steady state is reached. Initially, the source voltage is such that the D-STATCOM is inactive. It does not absorb nor provide reactive power to the network. At t = 0.2 s, the source voltage is increased by 6%. The D-STATCOM compensates for this voltage increase by absorbing reactive power from the network (Q=+2.7 Mvar on trace 2 of Scope2). At t = 0.3 s, the source voltage is decreased by 6% from the value corresponding to Q = 0. The D-STATCOM must generate reactive power to maintain a 1 pu voltage (Q changes from +2.7 MVAR to -2.8 MVAR). Note that when the D-STATCOM changes from inductive to capacitive operation, the modulation index of the PWM inverter is increased from 0.56 to 0.9 (trace 4 of Scope2) which corresponds to a proportional increase in inverter voltage. Reversing of reactive power is very fast, about one cycle, as observed on D-STATCOM current (magenta signal on trace 1 of Scope1).

#### 2. Mitigation of voltage flicker

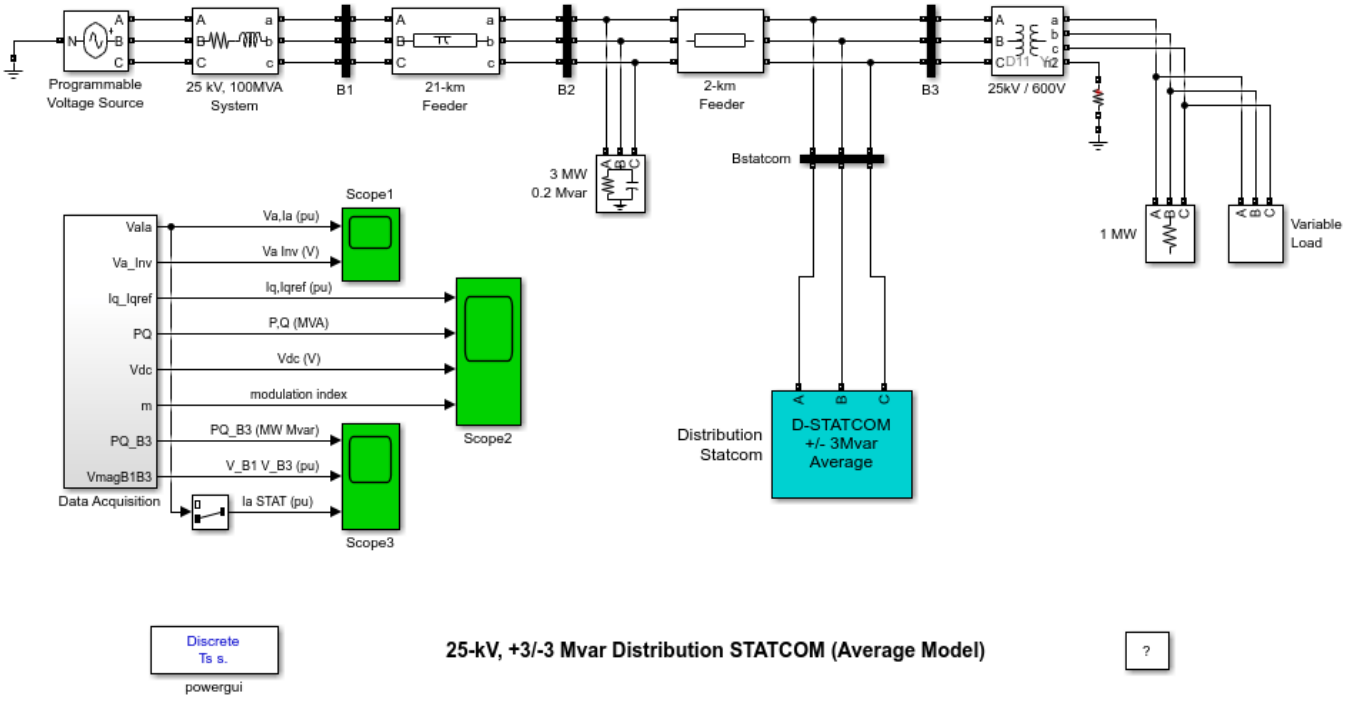
During this test, voltage of the Programmable Voltage Source will be kept constant and you will enable modulation of the Variable Load so that you can observe how the D-STATCOM can mitigate voltage flicker. In the Programmable Voltage Source block menu, change the "Time Variation of" parameter to "None". In the Variable Load block menu, set the Modulation Timing parameter to [Ton Toff]= [0.15 1] (remove the 100 multiplication factor). Finally, in the D-STATCOM Controller, change the "Mode of operation" parameter to "Q regulation" and make sure that the reactive power reference value Qref (2nd line of parameters) is set to zero. In this mode, the D-STATCOM is floating and performs no voltage correction.

Run the simulation and observe on Scope3 variations of P and Q at bus B3 (1st trace) as well as voltages at buses B1 and B3 (trace 2). Without D-STATCOM, B3 voltage varies between 0.96 pu and 1.04 pu (+/- 4% variation). Now, in the D-STATCOM Controller, change the "Mode of operation" parameter back to "Voltage regulation" and restart simulation. Observe on Scope 3 that voltage fluctuation at bus B3 is now reduced to +/- 0.7 %. The D-STATCOM compensates voltage by injecting a reactive current modulated at 5 Hz (trace 3 of Scope3) and varying between 0.6 pu capacitive when voltage is low and 0.6 pu inductive when voltage is high.

# D-STATCOM (Average Model)

This example shows an average model of a distribution STATCOM.

Pierre Giroux ; Gilbert Sybille (Hydro-Quebec, IREQ)




### Detailed Versus Average Model

When modeling VSC-based energy conversion systems in Specialized Power Systems, you can use two types of models, depending on the range of frequencies to be represented : the detailed model and the average model.

**The detailed model** such as the one presented in the power_dstatcom_pwm model in the FACTS library of examples. The detailed model includes detailed representation of power electronic IGBT converters. In order to achieve an acceptable accuracy with the 1680 Hz switching frequency used in this example, the model must be discretized at a relatively small time step (5 microseconds). This model is well suited for observing harmonics and control system dynamic performance over relatively short periods of times (typically hundreds of milliseconds to one second).

**The average model** such as the one presented in this example. In this type of model the IGBT Voltage-Sourced Converters (VSC) are represented by equivalent voltage sources generating the AC voltage averaged over one cycle of the switching frequency. This model does not represent harmonics, but the dynamics resulting from control system and power system interaction is preserved. This model allows using much larger time steps (typically 40-50 microseconds), thus allowing simulations of several seconds.

Alternatively, a third type of model can be used for simulating on larger time frames: the **phasor model**. This type of model is not available for the D-STATCOM , but it is available for the STATCOM, a similar device, in the power_statcom model.

### Description

A Distribution Static Synchronous Compensator (D-STATCOM) is used to regulate voltage on a 25-kV distribution network. Two feeders (21 km and 2 km) transmit power to loads connected at buses B2 and B3. A shunt capacitor is used for power factor correction at bus B2. The 600-V load connected to bus B3 through a 25kV/600V transformer represents a plant absorbing continuously changing currents, similar to an arc furnace, thus producing voltage flicker. The variable load current magnitude is modulated at a frequency of 5 Hz so that its apparent power varies approximately between 1 MVA and 5.2 MVA, while keeping a 0.9 lagging power factor. This load variation will allow you to observe the ability of the D-STATCOM to mitigate voltage flicker.

The D-STATCOM regulates bus B3 voltage by absorbing or generating reactive power. This reactive power transfer is done through the leakage reactance of the coupling transformer by generating a secondary voltage in phase with the primary voltage (network side). This voltage is provided by a voltage-sourced PWM inverter. When the secondary voltage is lower than the bus voltage, the D-STATCOM acts like an inductance absorbing reactive power. When the secondary voltage is higher than the bus voltage, the D-STATCOM acts like a capacitor generating reactive power.

The D-STATCOM consists of the following components:

- **a 25kV/1.25kV coupling transformer** which ensures coupling between the PWM inverter and the network.
- **a voltage-sourced PWM inverter**. In this example, the PWM inverter is replaced on the AC side with three equivalent voltage sources averaged over one cycle of the switching frequency (1.68 kHz). Harmonics generated by the inverter are therefore not visible with this average model. On the DC side, the inverter is modeled by a current source charging the DC capacitor. The DC current Idc is computed so that the instantaneous power at the AC inputs of the inverter remains equal the instantaneous power at the DC output (Va*Ia + Vb*Ib + Vc*Ic = Vdc*Idc).
- **LC damped filters connected** at the inverter output. Resistances connected in series with capacitors provide a quality factor of 40 at 60 Hz.
- **a 10000-microfarad capacitor** acting as a DC voltage source for the inverter
- **a voltage regulator** that controls voltage at bus B3
- **anti-aliasing filters** used for voltage and current acquisition.

The D-STATCOM controller consists of several functional blocks:

- **a Phase Locked Loop (PLL)**. The PLL is synchronized to the fundamental of the transformer primary voltages.
- **two measurement systems**. Vmeas and Imeas blocks compute the d-axis and q-axis components of the voltages and currents by executing an abc-dq transformation in the synchronous reference determined by sin(wt) and cos(wt) provided by the PLL.
- **an inner current regulation loop**. This loop consists of two proportional-integral (PI) controllers that control the d-axis and q-axis currents. The controllers outputs are the Vd and Vq voltages that the PWM inverter has to generate. The Vd and Vq voltages are converted into phase voltages Va, Vb, Vc which are used to synthesize the PWM voltages. The Iq reference comes from the outer voltage regulation loop (in automatic mode) or from a reference imposed by Qref (in manual mode). The Id reference comes from the DC-link voltage regulator.
- **an outer voltage regulation loop**. In automatic mode (regulated voltage), a PI controller maintains the primary voltage equal to the reference value defined in the control system dialog box.

- **a DC voltage controller** which keeps the DC link voltage constant to its nominal value (Vdc=2.4 kV).

The electrical circuit is discretized using a sample time Ts=40 microseconds. The controller uses a larger sample time (4*Ts=160 microseconds).

### Simulation

**1. D-STATCOM dynamic response**

During this test, the variable load will be kept constant and you will observe the dynamic response of a D-STATCOM to step changes in source voltage. Check that the modulation of the Variable Load is not in service (Modulation Timing [Ton Toff]= [0.15 1]*100 > Simulation Stop time). The Programmable Voltage Source block is used to modulate the internal voltage of the 25-kV equivalent. The voltage is first programmed at 1.077 pu in order to keep the D-STATCOM initially floating (B3 voltage=1 pu and reference voltage Vref=1 pu). Three steps are programmed at 0.2 s, 0.3 s, and 0.4 s to successively increase the source voltage by 6%, decrease it by 6% and bring it back to its initial value (1.077 pu).

Start the simulation. Observe on Scope1 the phase A voltage and current waveforms of the D-STATCOM as well as controller signals on Scope2. After a transient lasting approximately 0.15 sec., the steady state is reached. Initially, the source voltage is such that the D-STATCOM is inactive. It does not absorb nor provide reactive power to the network. At t = 0.2 s, the source voltage is increased by 6%. The D-STATCOM compensates for this voltage increase by absorbing reactive power from the network (Q=+2.7 Mvar on trace 2 of Scope2). At t = 0.3 s, the source voltage is decreased by 6% from the value corresponding to Q = 0. The D-STATCOM must generate reactive power to maintain a 1 pu voltage (Q changes from +2.7 MVAR to -2.8 MVAR). Note that when the D-STATCOM changes from inductive to capacitive operation, the modulation index of the PWM inverter is increased from 0.56 to 0.9 (trace 4 of Scope2) which corresponds to a proportional increase in inverter voltage. Reversing of reactive power is very fast, about one cycle, as observed on D-STATCOM current (magenta signal on trace 1 of Scope1).

**2. Mitigation of voltage flicker**

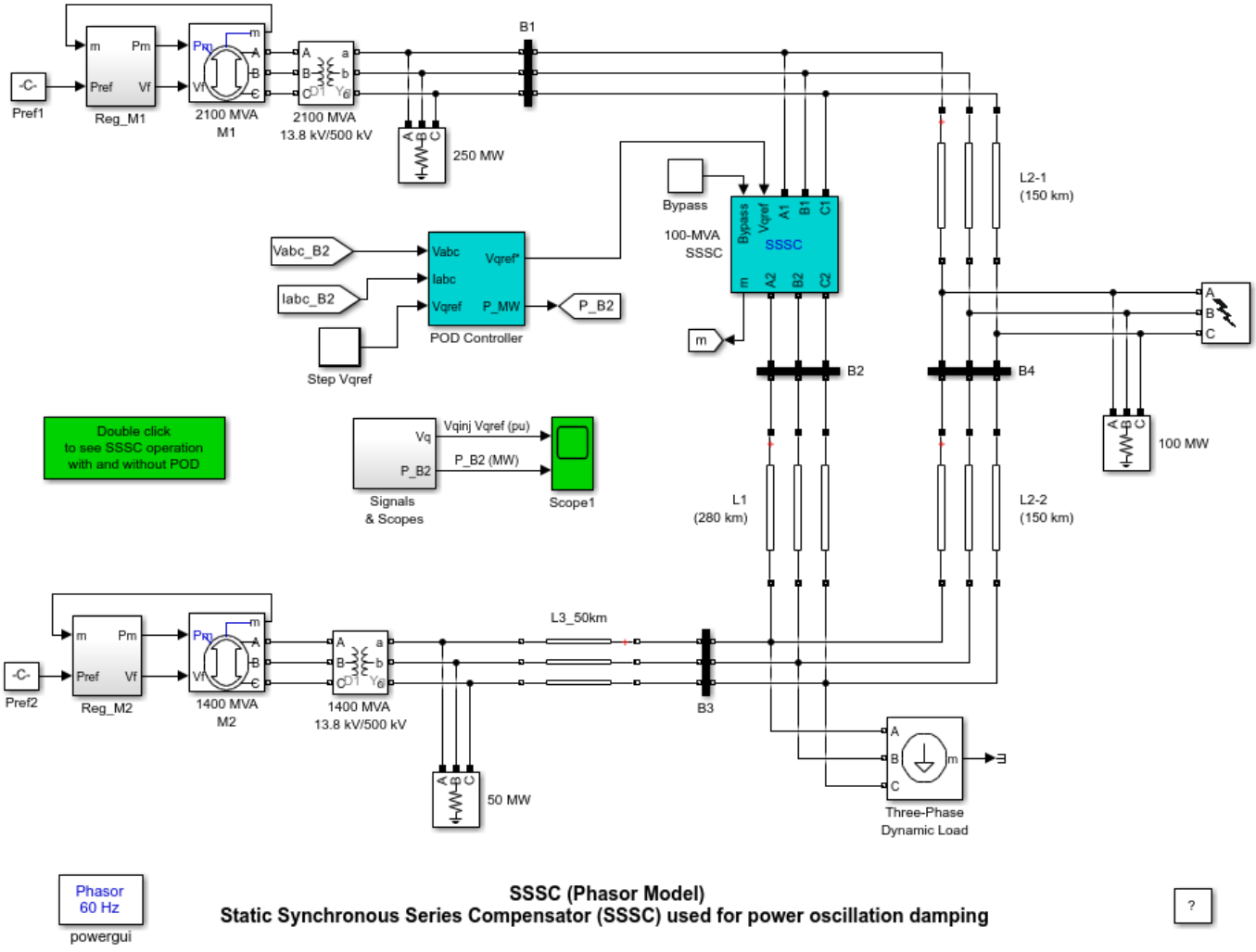
During this test, voltage of the Programmable Voltage Source will be kept constant and you will enable modulation of the Variable Load so that you can observe how the D-STATCOM can mitigate voltage flicker. In the Programmable Voltage Source block menu, change the "Time Variation of" parameter to "None". In the Variable Load block menu, set the Modulation Timing parameter to [Ton Toff]= [0.15 1] (remove the 100 multiplication factor). Finally, in the D-STATCOM Controller, change the "Mode of operation" parameter to "Q regulation? and make sure that the reactive power reference value Qref (2nd line of parameters) is set to zero. In this mode, the D-STATCOM is floating and performs no voltage correction.

Run the simulation and observe on Scope3 variations of P and Q at bus B3 (1st trace) as well as voltages at buses B1 and B3 (trace 2). Without D-STATCOM, B3 voltage varies between 0.96 pu and 1.04 pu (+/- 4% variation). Now, in the D-STATCOM Controller, change the "Mode of operation" parameter back to "Voltage regulation" and restart simulation. Observe on Scope 3 that voltage fluctuation at bus B3 is now reduced to +/- 0.7 %. The D-STATCOM compensates voltage by injecting a reactive current modulated at 5 Hz (trace 3 of Scope3) and varying between 0.6 pu capacitive when voltage is low and 0.6 pu inductive when voltage is high.

# SSSC (Phasor Model)

This example shows a Static Synchronous Series Compensator (SSSC) used for power oscillation damping.

Pierre Giroux and Gibert Sybille (Hydro-Quebec)




## Description

The Static Synchronous Series Compensator (SSSC), one of the key FACTS devices, consists of a voltage-sourced converter and a transformer connected in series with a transmission line. The SSSC injects a voltage of variable magnitude in quadrature with the line current, thereby emulating an inductive or capacitive reactance. This emulated variable reactance in series with the line can then influence the transmitted electric power. The SSSC is used to damp power oscillation on a power grid following a three-phase fault.

The power grid consists of two power generation substations and one major load center at bus B3. The first power generation substation (M1) has a rating of 2100 MVA, representing 6 machines of 350

MVA and the other one (M2) has a rating of 1400 MVA, representing 4 machines of 350 MVA. The load center of approximately 2200 MW is modeled using a dynamic load model where the active & reactive power absorbed by the load is a function of the system voltage.The generation substation M1 is connected to this load by two transmission lines L1 and L2. L1 is 280-km long and L2 is split in two segments of 150 km in order to simulate a three-phase fault (using a fault breaker) at the midpoint of the line. The generation substation M2 is also connected to the load by a 50-km line (L3). When the SSSC is bypassed, the power flow towards this major load is as follows: 664 MW flow on L1 (measured at bus B2), 563 MW flow on L2 (measured at B4) and 990 MW flow on L3 (measured at B3).

The SSSC, located at bus B1, is in series with line L1. It has a rating of 100MVA and is capable of injecting up to 10% of the nominal system voltage. This SSSC is a phasor model of a typical three-level PWM SSSC. If you open the SSSC dialog box and select "Display Power data", you will see that our model represents a SSSC having a DC link nominal voltage of 40 kV with an equivalent capacitance of 375 uF. On the AC side, its total equivalent impedance is 0.16 pu on 100 MVA. This impedance represents the transformer leakage reactance and the phase reactor of the IGBT bridge of an actual PWM SSSC.The SSSC injected voltage reference is normally set by a POD (Power Oscillation Damping) controller whose output is connected to the Vqref input of the SSSC. The POD controller consists of an active power measurement system, a general gain, a low-pass filter, a washout high-pass filter, a lead compensator, and an output limiter. The inputs to the POD controller are the bus voltage at B2 and the current flowing in L1. Look under mask to see how the controller is built.

### Simulation

#### 1. SSSC Dynamic Response

We will first verify the dynamic response of our model. Open the "Step Vqref" block (the red timer block connected to the "Vqref" input of the POD Controller).This block should be programmed to modify the reference voltage Vqref as follows: Initially Vqref is set to 0 pu; at t=2 s, Vqref is set to -0.08 pu (SSSC inductive); then at t=6 s, Vqref is set to 0.08 pu (SSSC capacitive). Double-click on the POD Controller block and set the POD status parameter to "off". This will disable the POD controller. Also, make sure that the fault breaker will not operate during the simulation (the parameters "Switching of phase A, B and C" should not be selected).

Run the simulation and look at Scope1. The first graph displays the Vqref signal (magenta trace) along with the measured injected voltage by the SSSC. The second graph displays the active power flow (P_B2) on line L1, measured at bus B2. We can see that the SSSC regulator follows very well the reference signal Vqref. Depending on the injected voltage, the power flow on line varies from 575 to 750 MW. In a real system the reference signal Vqref would typically be changed much more gradually in order to avoid the oscillation we see on the transmitted power (P_B2 signal). Double-click on the SSSC block and select "Display Control parameters". Modify the "Maximum rate of change for Vqref (pu/s)" parameter from 3 to 0.05. Rerun the simulation. The power oscillation on the active power should now be very small.

#### 2. SSSC damping power oscillation

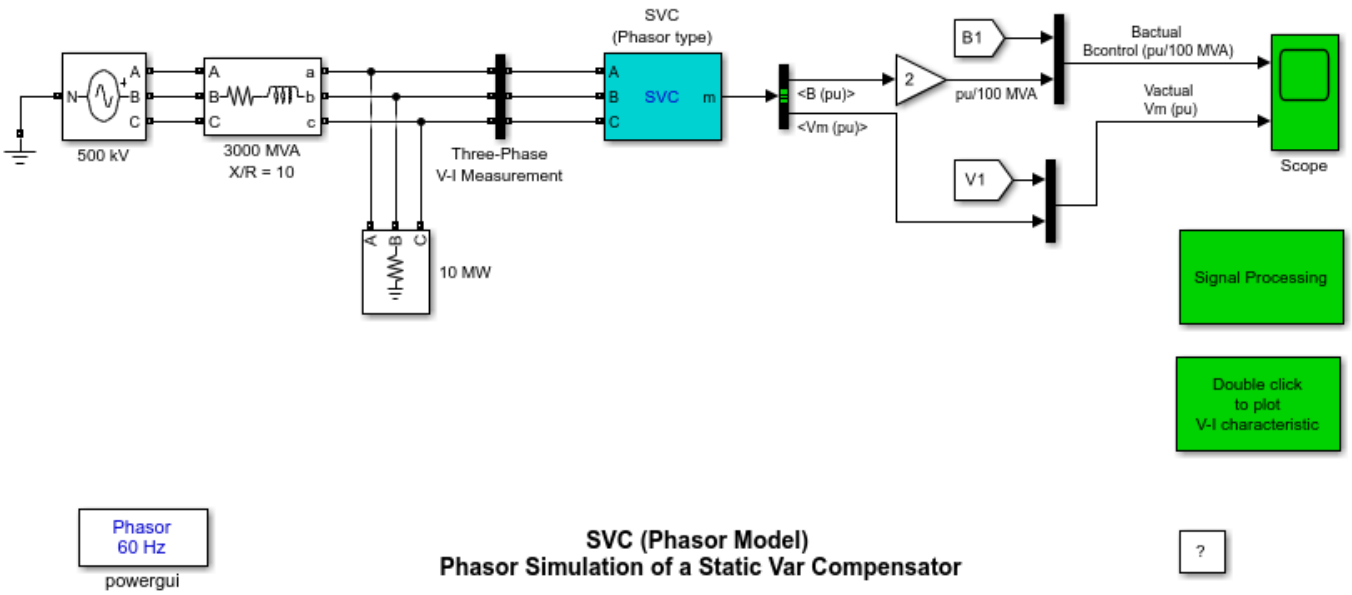
We will now compare the operation of our SSSC with and without POD control. Open the "Step Vqref" block and multiply by 1000 the time vector in order to disable the Vqref variations. Double-click on the fault breaker and select the parameters "Switching of phase A, B and C" to simulate a three-phase fault. The transition times should be set as follows: [ 20/60 30/60]+1; this means that the fault will be applied at 1.33 s and will last for 10 cycles. Run a simulation and observe the power oscillation on the L1 line (second graph on Scope1) following the three-phase fault.

Now, you will run a second simulation with the POD controller in operation. Double-click on the POD Controller block and set the POD status parameter to "on". Start the simulation. Looking again at the second graph on Scope1(P_B2 signal), we can see that the SSSC with a POD controller is a very effective tool to damp power oscillation. To see a figure showing a comparison of the SSSC operation with and without POD control, double-click on the blue block on the bottom right of the model.

# SVC (Phasor Model)

This example shows steady-state and dynamic performance of the static var compensator model.

G. Sybille (Hydro-Quebec)




### Description

A static var compensator (SVC) is used to regulate voltage on a 500 kV, 3000 MVA system. When system voltage is low the SVC generates reactive power (SVC capacitive). When system voltage is high it absorbs reactive power (SVC inductive). The SVC is rated +200 Mvar capacitive and 100 Mvar inductive. The Static Var Compensator block is a a phasor model representing the SVC static and dynamic characteristics at the system fundamental frequency.

To see the SVC control parameters, open the SVC dialog box and select "Display Control parameters". The SVC is set in voltage regulation mode with a reference voltage Vref=1.0 pu. The voltage droop is 0.03 pu/ 200MVA, so that the voltage varies from 0.97 pu to 1.015 pu when the SVC current goes from fully capacitive to fully inductive. Double click now on the blue block to display the SVC V-I characteristic.

The actual SVC positive-sequence voltage (V1) and susceptance (B1) are measured inside the 'Signal Processing' subsystem, using the complex voltages Vabc and complex currents Iabc returned by the Three-Phase V-I Measurement block.

### 1. Dynamic Response of the SVC

The Three-Phase Programmable Voltage Source is used to vary the system voltage and observe the SVC performance. Initially the source is generating nominal voltage. Then, voltage is successively decreased (0.97 pu at t = 0.1 s), increased (1.03 pu at t = 0.4 s) and finally returned to nominal voltage (1 pu at t = 0.7 s).

Start the simulation and observe the SVC dynamic response to voltage steps on the Scope. Trace 1 shows the actual positive-sequence susceptance B1 and control signal output B of the voltage regulator. Trace 2 shows the actual system positive-sequence voltage V1 and output Vm of the SVC measurement system.

The SVC response speed depends on the voltage regulator integral gain Ki (Proportional gain Kp is set to zero), system strength (reactance Xn) and droop (reactance Xs). If the voltage measurement time constant and average time delay Td due to valve firing are neglected, the system can be approximated by a first order system having a closed loop time constant :

```
Tc= 1/(Ki*(Xn+Xs))
```

With given system parameters (Ki = 300; Xn = 0.0667 pu/200 MVA; Xs = 0.03 pu/200 MVA), Tc = 0.0345 s. If you increase the regulator gain or decrease the system strength, the measurement time constant and the valve firing delay Td will no longer be negligible and you will observe an oscillatory response and eventually unstability.

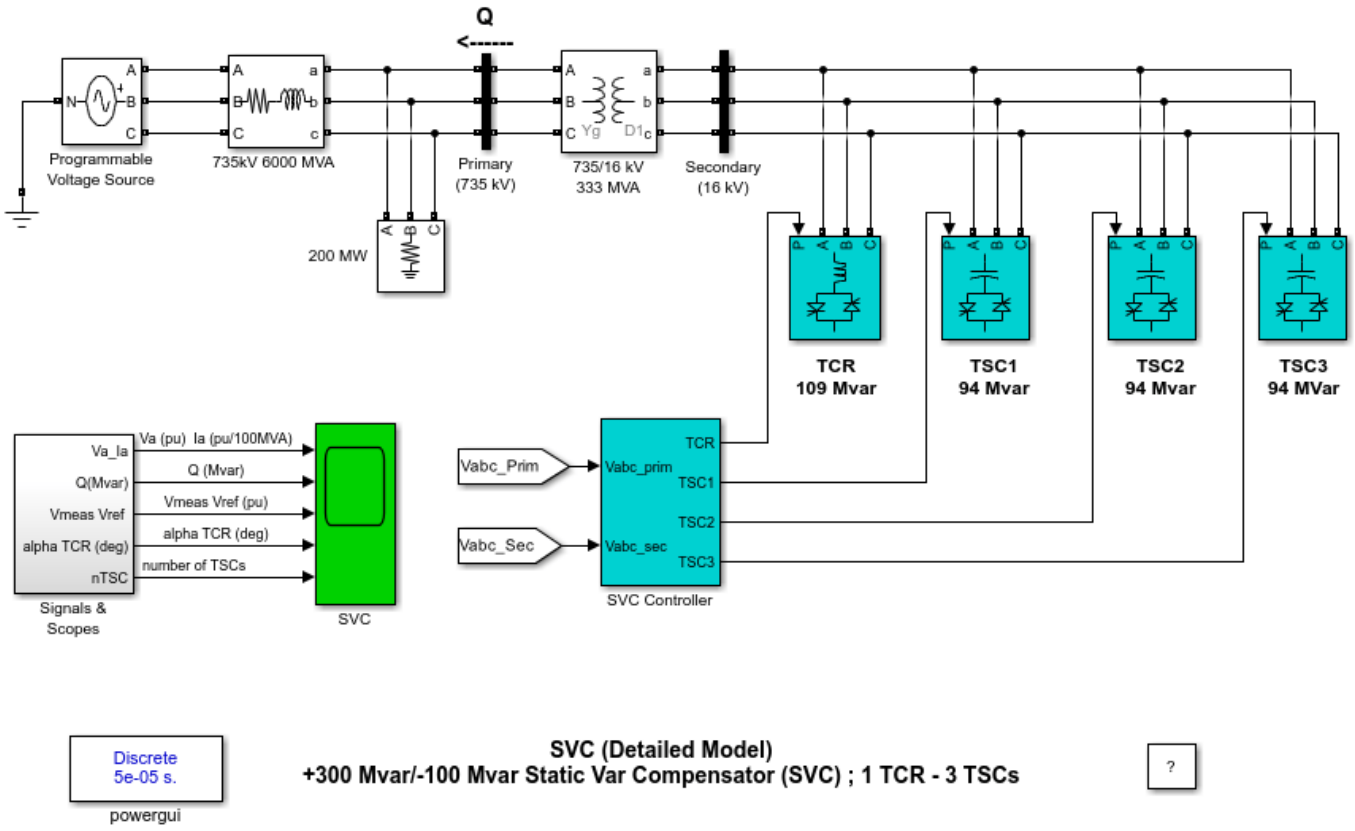
### 2. Measurement of Steady-State V-I Characteristic

In order to measure the SVC steady-state V-I characteristic, you will now program a slow variation of the source voltage. Open the Programmable Voltage Source menu and change the "Type of Variation" parameter to "Modulation". The modulation parameters are set to apply a sinusoidal variation of the positive-sequence voltage between 0.75 and 1.25 pu in 20 seconds. Change the stop time to 20 s and restart simulation. When simulation is completed, double click the blue block. The theoretical V-I characteristic is displayed (in red) together with the measured characteristic (in blue).

# SVC (Detailed Model)

This example shows the operation of a +300 Mvar/-100 Mvar Static Var Compensator (SVC).

Pierre Giroux and Gibert Sybille (Hydro-Quebec)







### Description

A 300-Mvar Static Var Compensator (SVC) regulates voltage on a 6000-MVA 735-kV system. The SVC consists of a 735kV/16-kV 333-MVA coupling transformer, one 109-Mvar thyristor-controlled reactor bank (TCR) and three 94-Mvar thyristor-switched capacitor banks (TSC1 TSC2 TSC3) connected on the secondary side of the transformer. Switching the TSCs in and out allows a discrete variation of the secondary reactive power from zero to 282 Mvar capacitive (at 16 kV) by steps of 94 Mvar, whereas phase control of the TCR allows a continuous variation from zero to 109 Mvar inductive. Taking into account the leakage reactance of the transformer (15%), the SVC equivalent susceptance seen from the primary side can be varied continuously from -1.04 pu/100 MVA (fully inductive) to +3.23 pu/100 Mvar (fully capacitive). The SVC controller monitors the primary voltage and sends appropriate pulses to the 24 thyristors (6 thyristors per three-phase bank) in order to obtain the susceptance required by the voltage regulator.

Use Look under Mask to see how the TCR and TSC subsystems are built. Each three-phase bank is connected in delta so that, during normal balanced operation, the zero-sequence tripplen harmonics (3rd, 9th... ) remain trapped inside the delta, thus reducing harmonic injection into the power system.

The power system is represented by an inductive equivalent (6000 MVA short circuit level) and a 200-MW load. The internal voltage of the equivalent can be varied by means of programmable source in order to observe the SVC dynamic response to changes in system voltage. Open the voltage source menu and look at the sequence of voltage steps which are programmed.

### Simulation

#### Dynamic response of the SVC

Run the simulation and observe waveforms on the SVC scope block. The SVC is in voltage control mode and its reference voltage is set to Vref=1.0 pu. The voltage droop of the regulator is 0.01 pu/100 VA (0.03 pu/300MVA). Therefore when the SVC operating point changes from fully capacitive (+300 Mvar) to fully inductive (-100 Mvar) the SVC voltage varies between 1-0.03=0.97 pu and 1+0.01=1.01 pu.

Initially the source voltage is set at 1.004 pu, resulting in a 1.0 pu voltage at SVC terminals when the SVC is out of service . As the reference voltage Vref is set to 1.0 pu, the SVC is initially floating (zero current). This operating point is obtained with TSC1 in service and TCR almost at full conduction (alpha=96 degrees). At t=0.1s voltage is suddenly increased to 1.025 pu. The SVC reacts by absorbing reactive power (Q=-95 Mvar) in order to bring the voltage back to 1.01 pu. The 95% settling time is approximately 135 ms. At this point all TSCs are out of service and the TCR is almost at full conduction (alpha = 94 degrees) . At t=0.4 s the source voltage is suddenly lowered to 0.93 pu. The SVC reacts by generating 256 Mvar of reactive power, thus increasing the voltage to 0.974 pu. At this point the three TSCs are in service and the TCR absorbs approximately 40% of its nominal reactive power (alpha =120 degrees). Observe on the last trace of the scope how the TSCs are sequentially switched on and off. Each time a TSC is switched on the TCR alpha angle changes suddenly from 180 degrees (no conduction) to 90 degrees (full conduction). Finally, at t=0.7 s the voltage is increased to 1.0 pu and the SVC reactive power is reduced to zero.

#### Misfiring of TSC1

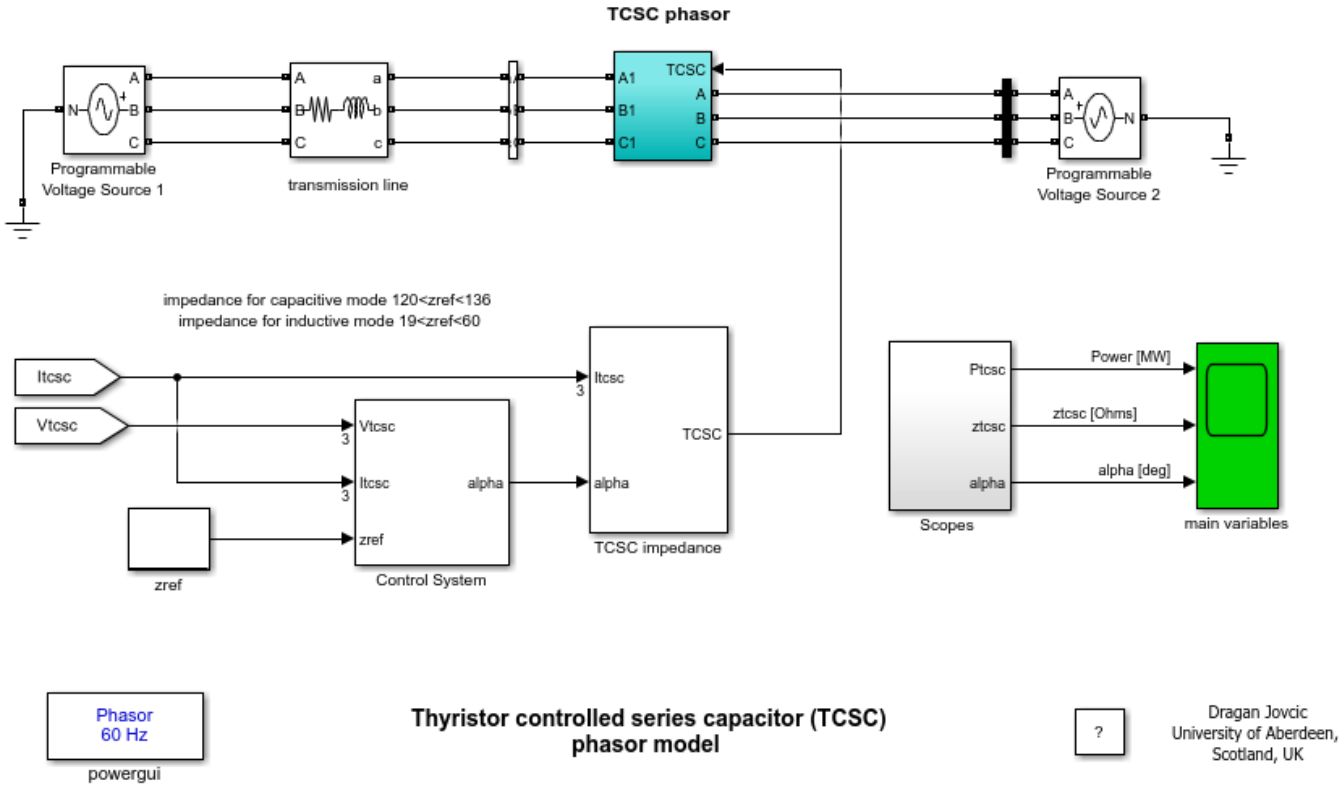
Each time a TSC is switched off a voltage remains trapped across the TSC capacitors. If you look at the 'TSC1 Misfiring' scope inside the "Signals and Scope" subsystem you can observe the TSC1 voltage (first trace) and the TSC1 current (second trace) for branch AB. The voltage across the positive thyristor (thyristor conducting the positive current) is shown on the 3rd trace and the pulses sent to this thyristor are shown on the 4th trace. Notice that the positive thyristor is fired at maximum negative TSC voltage, when the valve voltage is minimum. If by mistake the firing pulse is not sent at the right time, very large overcurrents can be observed in the TSC valves.

Look inside the SVC Controller block how a misfiring can be simulated on TSC1. A Timer block and a OR block are used to add pulses to the normal pulses coming from the Firing Unit. Open the Timer block menu and remove the 100 multiplication factor. The timer is now programmed to send a misfiring pulse lasting one sample time at time t= 0.121 s. Restart simulation. Observe that the misfiring pulse is sent when the valve voltage is maximum positive immediately after the TSC has blocked. This thyristor misfiring produces a large thyristor overcurrent (18 kA or 6.5 times the nominal peak current). Also, immediately after the thyristor has blocked, the thyristor voltage reaches 85 kV (3.8 times the nominal peak voltage). In order to prevent such overcurrents and overvoltages, thyristor valves are normally protected by metal oxide arresters (not simulated here).

# TCSC (Phasor Model)

This example shows the Thyristor Controlled Series Capacitor (TCSC) phasor test system.

Dragan Jovcic (University of Aberdeen, Scotland, UK)




### Phasor TCSC Model

This phasor tests system is similar to the TCSC thyristor-based tests system in the library. The phasor model however uses the equivalent impedances at the fundamental frequency, neglecting all transients, and therefore it is not as accurate as the thyristor model. Nevertheless, the phasor model is much simpler and the speed of simulation is increased. By comparing the responses with the detailed model we can observe very good matching of all variables in steady-state. Some small discrepancies are caused by the thyristor resistance and other TCSC losses which are not included in the phasor model.

### Description

A TCSC is placed on a 500kV, long transmission line, to improve power transfer. Without the TCSC the power transfer is around 110MW, as seen during the first 0.5s of the simulation when the TCSC is bypassed. The TCSC is modeled as a voltage source using equivalent impedance at fundamental frequency in each phase. The nominal compensation is 75%, i.e. assuming only the capacitors (firing angle of 90deg). The natural oscillatory frequency of the TCSC is 163Hz, which is 2.7 times the fundamental frequency. The test system is described in [1].

The TCSC can operate in capacitive or inductive mode, although the latter is rarely used in practice. Since the resonance for this TCSC is around 58deg firing angle, the operation is prohibited in firing angle range 49deg - 69deg. Note that the resonance for the overall system (when the line impedance is included) is around 67deg. The capacitive mode is achieved with firing angles 69-90deg. The impedance is lowest at 90deg, and therefore power transfer increases as the firing angle is reduced. In capacitive mode the range for impedance values is approximately 120-136 Ohm. This range corresponds to approximately 490-830MW power transfer range (100%-110% compensation). Comparing with the power transfer of 110 MW with an uncompensated line, TCSC enables significant improvement in power transfer level.

To change the operating mode (inductive/capacitive/manual) use the toggle switch in the control block dialog. The inductive mode corresponds to the firing angles 0-49deg, and the lowest impedance is at 0deg. In the inductive operating mode, the range of impedances is 19-60 Ohm, which corresponds to 100-85 MW range of power transfer level. The inductive mode reduces power transfer over the line. A constant firing angle can also be applied and the same limits will apply as above.

### TCSC Control

When TCSC operates in the constant impedance mode it uses voltage and current feedback for calculating the TCSC impedance. The reference impedance indirectly determines the power level, although an automatic power control mode could also be introduced.

A separate PI controller is used in each operating mode. The capacitive mode also employs a phase lead compensator. Each controller further includes an adaptive control loop to improve performance over a wide operating range. The controller gain scheduling compensates for the gain changes in the system, caused by the variations in the impedance.

The TCSC is simulated as a controllable voltage source in each phase. The voltage magnitude is the product of equivalent complex impedance and the line current. The expression for the TCSC impedance is given in [1].

### Simulation

Run the simulation and observe waveforms on the main variables scope block. The TCSC is in the capacitive impedance control mode and the reference impedance is set to 128 Ohm. For the first 0.5s, the TCSC is bypassed (assuming a circuit breaker), and the power transfer is 110 MW. At 0.5s TCSC begins to regulate the impedance to 128 Ohm and this increases power transfer to 610MW. Note that the TCSC starts with alpha at 90deg to enable lowest switching disturbance on the line.

### Dynamic Response

At 2.5s a 5% change in the reference impedance is applied. The response indicates that TCSC enables tracking of the reference impedance and the settling time is around 500ms. At 3.3s a 4% reduction in the source voltage is applied, followed by the return to 1p.u. at 3.8s. It is seen that the TCSC controller compensates for these disturbances and the TCSC impedance stays constant. The TCSC response time is 200ms-300ms. Note that the shape of transient response is inaccurate with phasor models and the thyristor based model should be used for studying transients.

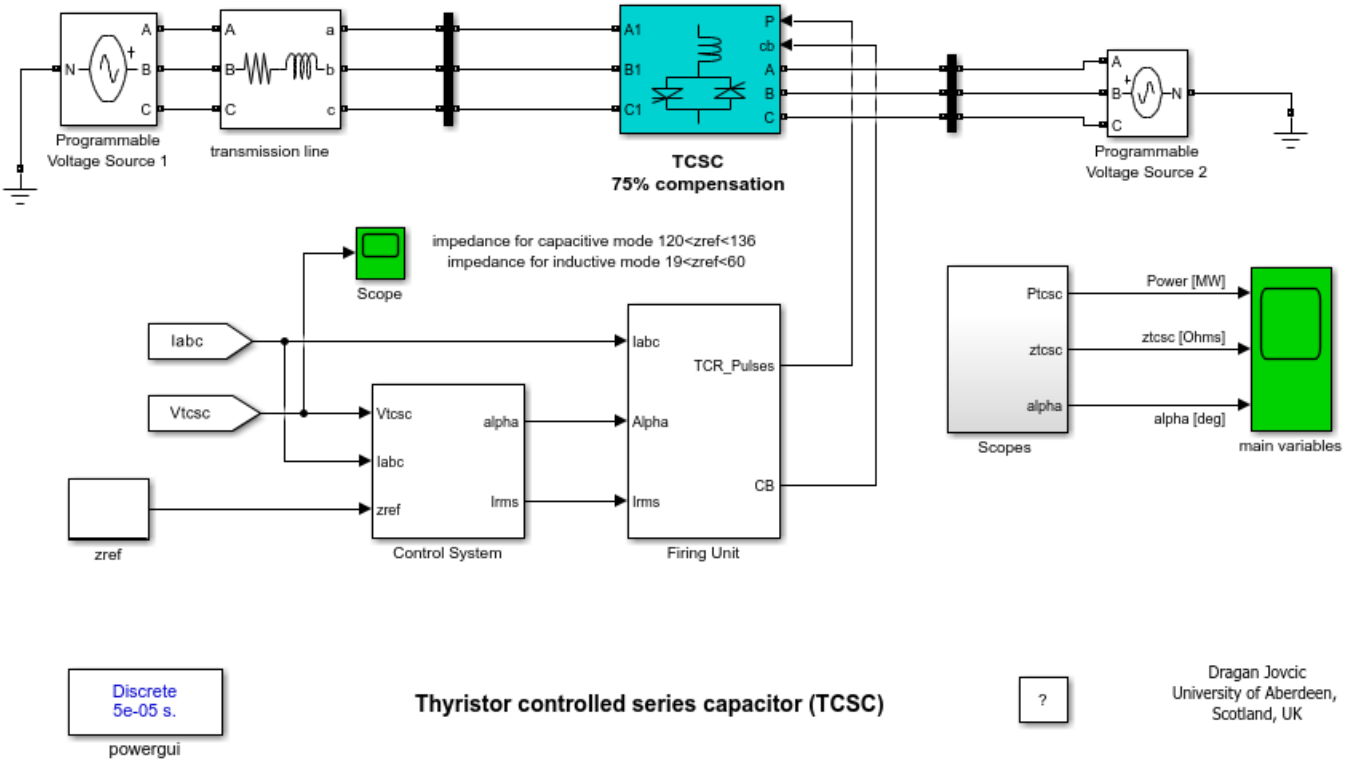
### Reference

[1] D.Jovcic, G.N.Pillai "Analytical Modelling of TCSC Dynamics" IEEE® Transactions on Power Delivery, vol 20, Issue 2, April 2005, pp. 1097-1104

# TCSC (Detailed Model)

This example shows the Thyristor Controlled Series Capacitor (TCSC) test system.

Dragan Jovcic (University of Aberdeen, Scotland, UK)




## Description

A TCSC is placed on a 500kV, long transmission line, to improve power transfer. Without the TCSC the power transfer is around 110MW, as seen during the first 0.5s of the simulation when the TCSC is bypassed. The TCSC consists of a fixed capacitor and a parallel Thyristor Controlled Reactor (TCR) in each phase. The nominal compensation is 75%, i.e. using only the capacitors (firing angle of 90deg). The natural oscillatory frequency of the TCSC is 163Hz, which is 2.7 times the fundamental frequency. The test system is described in [1].

The TCSC can operate in capacitive or inductive mode, although the latter is rarely used in practice. Since the resonance for this TCSC is around 58deg firing angle, the operation is prohibited in firing angle range 49deg - 69deg. Note that the resonance for the overall system (when the line impedance is included) is around 67deg. The capacitive mode is achieved with firing angles 69-90deg. The impedance is lowest at 90deg, and therefore power transfer increases as the firing angle is reduced. In capacitive mode the range for impedance values is approximately 120-136 Ohm. This range corresponds to approximately 490-830MW power transfer range (100%-110% compensation). Comparing with the power transfer of 110 MW with an uncompensated line, TCSC enables significant improvement in power transfer level.

To change the operating mode (inductive/capacitive/manual) use the toggle switch in the control block dialog. The inductive mode corresponds to the firing angles 0-49deg, and the lowest impedance is at 0deg. In the inductive operating mode, the range of impedances is 19-60 Ohm, which corresponds to 100-85 MW range of power transfer level. The inductive mode reduces power transfer over the line. A constant firing angle can also be applied and the same limits will apply as above.

### TCSC Control

When TCSC operates in the constant impedance mode it uses voltage and current feedback for calculating the TCSC impedance. The reference impedance indirectly determines the power level, although an automatic power control mode could also be introduced.

A separate PI controller is used in each operating mode. The capacitive mode also employs a phase lead compensator. Each controller further includes an adaptive control loop to improve performance over a wide operating range. The controller gain scheduling compensates for the gain changes in the system, caused by the variations in the impedance.

The firing circuit uses three single-phase PLL units for synchronisation with the line current. Line current is used for synchronisation, rather than line voltage, since the TCSC voltage can vary widely during the operation.

### Simulation

Run the simulation and observe waveforms on the main variables scope block. The TCSC is in the capacitive impedance control mode and the reference impedance is set to 128 Ohm. For the first 0.5s, the TCSC is bypassed using the circuit breaker, and the power transfer is 110 MW. At 0.5s TCSC begins to regulate the impedance to 128 Ohm and this increases power transfer to 610MW. Note that the TCSC starts with alpha at 90deg to enable lowest switching disturbance on the line.

### Dynamic Response

At 2.5s a 5% change in the reference impedance is applied. The response indicates that TCSC enables tracking of the reference impedance and the settling time is around 500ms. At 3.3s a 4% reduction in the source voltage is applied, followed by the return to 1p.u. at 3.8s. It is seen that the TCSC controller compensates for these disturbances and the TCSC impedance stays constant. The TCSC response time is 200ms-300ms.

Note: Using the Simulink® Accelerator™ can speed up the execution of this model by approximately 9 times.

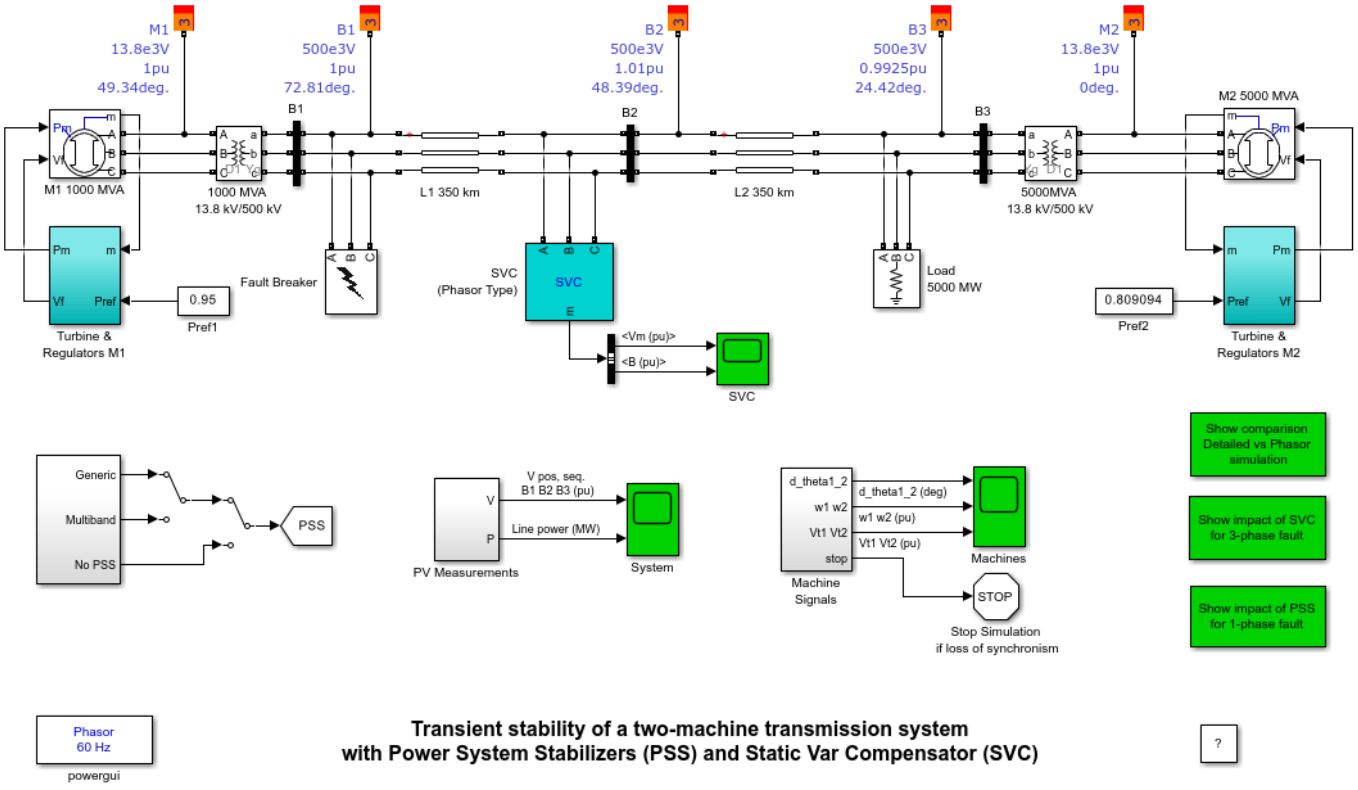
### Reference

[1] D.Jovcic, G.N.Pillai "Analytical Modelling of TCSC Dynamics" IEEE® Transactions on Power Delivery, vol 20, Issue 2, April 2005, pp. 1097-1104

# SVC and PSS (Phasor Model)

This example shows the use of the phasor solution for transient stability analysis of multi-machine systems. It analyzes transient stability of a two-machine transmission system with Power System Stabilizers (PSS) and Static Var Compensator (SVC).

Gilbert Sybille (Hydro-Quebec)




## Description

A 1000 MW hydraulic generation plant (machine M1) is connected to a load center through a long 500 kV, 700 km transmission line. The load center is modeled by a 5000 MW resistive load . The load is fed by the remote 1000 MW plant and a local generation of 5000 MW (machine M2). The system has been initialized so that the line carries 950 MW which is close to its surge impedance loading (SIL = 977 MW). In order to maintain system stability after faults, the transmission line is shunt compensated at its center by a 200-Mvar Static Var Compensator (SVC). Notice that this SVC model is a phasor model valid only for transient stability solution. The SVC does not have a Power Oscillation Damping (POD) unit. The two machines are equipped with a Hydraulic Turbine and Governor (HTG), Excitation system and Power System Stabilizer (PSS). These blocks are located in the two 'Turbine and Regulator' subsystems.Two types of stabilizers can be selected : a generic model using the acceleration power (Pa= difference between mechanical power Pm and output electrical power Peo) and a Multi-band stabilizer using the speed deviation (dw). The stabilizer type can be selected by specifying a value (0=No PSS 1=Pa PSS or 2= dw MB PSS) in the PSS constant block.

In this example we apply faults on the 500 kV system and observe the impact of the PSS and SVC on system stability.

### Simulation

***Note:*** *Before starting the example, open the Powergui block and notice that 'Phasor simulation' has been checked. The phasor solution is much faster than the 'standard' detailed solution. In this solution method, the network differential equations are replaced by a set of algebraic equations at a fixed frequency, thus reducing dramatically the simulation time. This allows transient stability studies of multi-machine systems, as illustrated below.*

**1. Initialization**

Note that the system has already been initialized to start in steady-state. If you are familiar with the Load Flow procedure you can skip this item and proceed to step 2.

Open the mask of the M1 1000 MVA and M2 5000 MVA Synchronous Machine blocks:

In the Load Flow tab of machine M1, the 'Generator type' parameter is set to 'PV', indicating that the load flow will be performed with the machine controlling its active power and its terminal voltage. The 'Active power generation' parameter is set to 950e6 W and the terminal voltage is defined by the Load Flow Bus block labeled M1 and connected to the machine terminals.

The 'Generator type' parameter of machine M2 is set to 'swing', indicating that the machine will be used as a swing bus for balancing the power.

In the Powergui menu, select 'Load Flow'. A new window appears. A summary of the load flow settings is displayed in a table. Press the 'Compute' button to solve the load flow. The table now display the actual machine active and reactive powers.

Press the 'Apply' button to apply the load flow solution to the model.

Look in the hydraulic turbine and governor (HTG) and Excitation system contained in the two Regulator subsystems to note that the initial mechanical power and field voltage have been automatically initialized by the Load Flow. The reference mechanical powers and reference voltages for the two machines have also been updated in the two constant blocks connected at the HTG and excitation system inputs: Pref1=0.95 pu (950 MW), Vref1=1pu; Pref2=0.8091 pu (4046 MW), Vref2=1 pu.

**2. Single-phase fault - Impact of PSS - No SVC**

Open the SVC dialog box and notice that the SVC is set to operate in 'Var control (fixed susceptance)' mode with Bref = 0. Setting Bref to zero is equivalent to putting the SVC out of service. Verify also that the two PSS (Pa type) are in service (value=1 in the PSS constant block) Start the simulation and observe signals on the 'Machines' scope. For this type of fault the system is stable without SVC. After fault clearing, the 0.8 Hz oscillation is quickly damped. This oscillation mode is typical of inter-area oscillations in a large power system. First trace on the 'Machine' scope shows the rotor angle difference d_theta1_2 between the two machines. Power transfer is maximum when this angle reaches 90 degrees. This signal is a good indication of system stability. If d_theta1_2 exceeds 90 degrees for a too long period of time, the machines will loose synchronism and the system goes unstable. Second trace shows the machine speeds. Notice that machine 1 speed increases during the fault because during that period its electrical power is lower than its mechanical power. By simulating over a long period of time (50 seconds) you will also notice that the machine speeds oscillate together at a low frequency (0.025 Hz) after fault clearing. The two PSS (Pa type) succeed to damp the 0.8 Hz mode but they are not efficient for damping the 0.025 Hz mode. If you select instead the Multi-Band PSS (value=2 in the PSS constant block) you will notice that this stabilizer type succeeds to damp both the 0.8 Hz mode and the 0.025 Hz mode.

You will now repeat the test with the two PSS out of service (value=0 in the PSS constant block). Restart simulation. Notice that the system is unstable without PSS. You can compare results with and without PSS by double clicking on the 2nd blue block on the right side. You can also compare the results obtained with the two solution methods 'Detailed' and 'Phasor' by double-clicking on the first blue block on the right side.

*Note: This system is naturally unstable without PSS, even for small disturbances. For example, if you remove the fault (by deselecting phase A in the Fault Breaker) and apply a Pref step of 0.05 pu on machine 1, you will see the instability slowly building up after a few seconds.*

**3. Three-phase fault - Impact of SVC - two PSS in service**

You will now apply a 3-phase fault and observe the impact of the SVC for stabilizing the network during a severe contingency. Put the two PSS (Pa type) in service (value=1 in the PSS constant block. Reprogram the 'Fault Breaker' block in order to apply a 3-phase-to-ground fault. Verify that the SVC is in fixed susceptance mode with Bref = 0. Start the simulation. By looking at the d_theta1_2 signal, you should observe that the two machines quickly fall out of synchronism after fault clearing. In order not to pursue unnecessary simulation, the Simulink® 'Stop' block is used to stop the simulation when the angle difference reaches 3*360degrees.

Now open the SVC block menu and change the SVC mode of operation to 'Voltage regulation'. The SVC will now try to support the voltage by injecting reactive power on the line when the voltage is lower than the reference voltage (1.009 pu). The chosen SVC reference voltage corresponds to the bus voltage with the SVC out of service. In steady state the SVC will therefore be 'floating' and waiting for voltage compensation when voltage departs from its reference set point.

Restart simulation and observe that the system is now stable with a 3-phase fault. You can compare results with and without SVC by double clicking on the 3rd blue block on the right side.

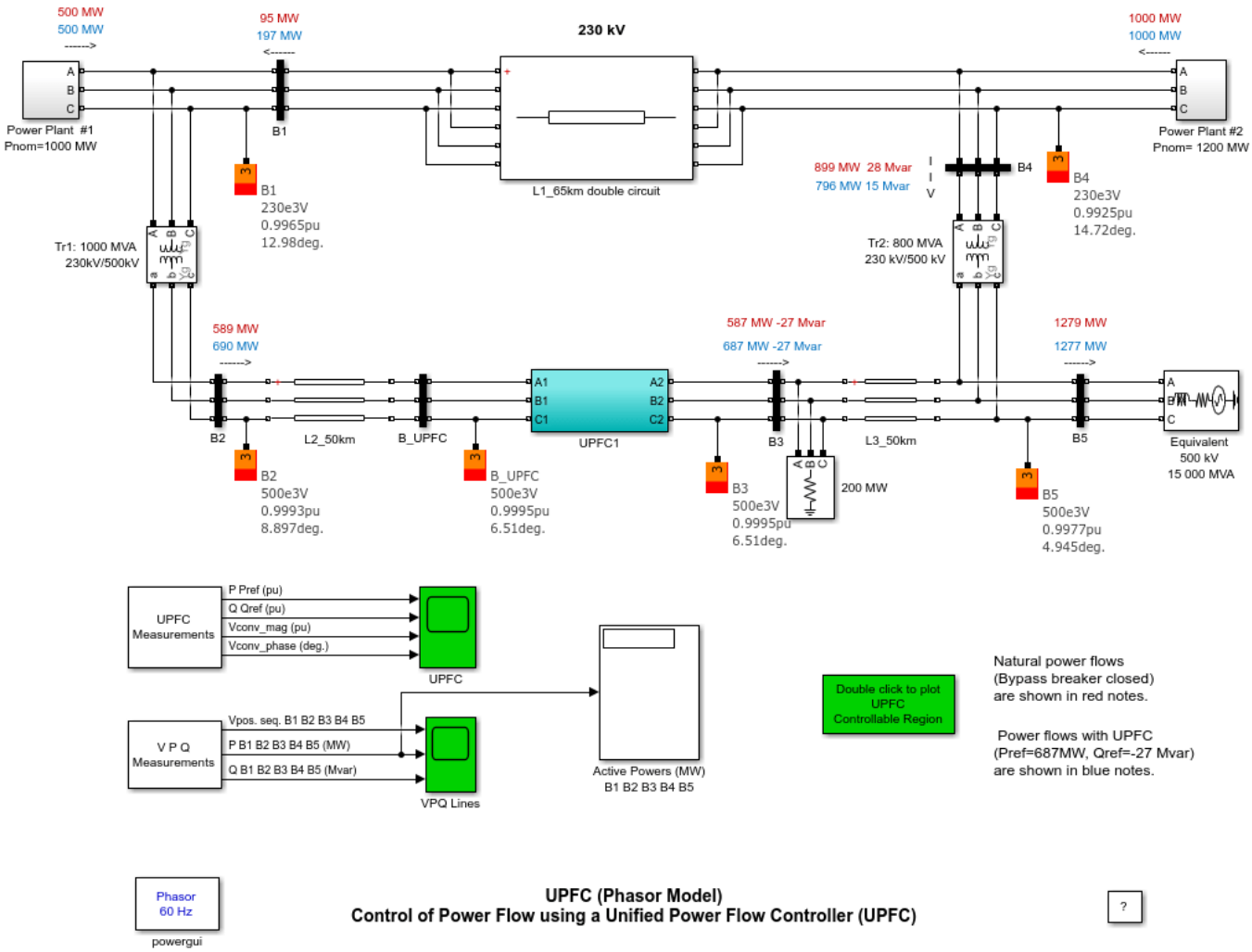
**Reference**

[1] D.Jovcic, G.N.Pillai "Analytical Modeling of TCSC Dynamics" IEEE® Transactions on Power Delivery, vol 20, Issue 2, April 2005, pp. 1097-1104

# UPFC (Phasor Model)

This example shows an Unified Power Flow Controller (UPFC) used to relieve power congestion on a 500/230 kV grid.

Gibert Sybille and Pierre Giroux (Hydro-Quebec)




## Description

A UPFC is used to control the power flow in a 500 kV /230 kV transmission system. The system, connected in a loop configuration, consists essentially of five buses (B1 to B5) interconnected through transmission lines (L1, L2, L3) and two 500 kV/230 kV transformer banks Tr1 and Tr2. Two power plants located on the 230-kV system generate a total of 1500 MW which is transmitted to a 500-kV 15000-MVA equivalent and to a 200-MW load connected at bus B3. The plant models include a speed regulator, an excitation system as well as a power system stabilizer (PSS). In normal operation, most of the 1200-MW generation capacity of power plant #2 is exported to the 500-kV equivalent through three 400-MVA transformers connected between buses B4 and B5. We are considering a contingency case where only two transformers out of three are available (Tr2= 2*400 MVA = 800 MVA).

Using the load flow option of the powergui block, the model has been initialized with plants #1 and #2 generating respectively 500 MW and 1000 MW and the UPFC out of service (Bypass breaker closed). The resulting power flow obtained at buses B1 to B5 is indicated by red numbers on the circuit diagram. The load flow shows that most of the power generated by plant #2 is transmitted through the 800-MVA transformer bank (899 MW out of 1000 MW), the rest (101 MW), circulating in the loop. Transformer Tr2 is therefore overloaded by 99 MVA. The example illustrates how the UPFC can relieve this power congestion.

The UPFC located at the right end of line L2 is used to control the active and reactive powers at the 500-kV bus B3, as well as the voltage at bus B_UPFC. It consists of a phasor model of two 100-MVA, IGBT-based, converters (one connected in shunt and one connected in series and both interconnected through a DC bus on the DC side and to the AC power system, through coupling reactors and transformers). Parameters of the UPFC power components are given in the dialog box. The series converter can inject a maximum of 10% of nominal line-to-ground voltage (28.87 kV) in series with line L2. The blue numbers on the diagram show the power flow with the UPFC in service and controlling the B3 active and reactive powers respectively at 687 MW and -27 Mvar.

**Simulation**

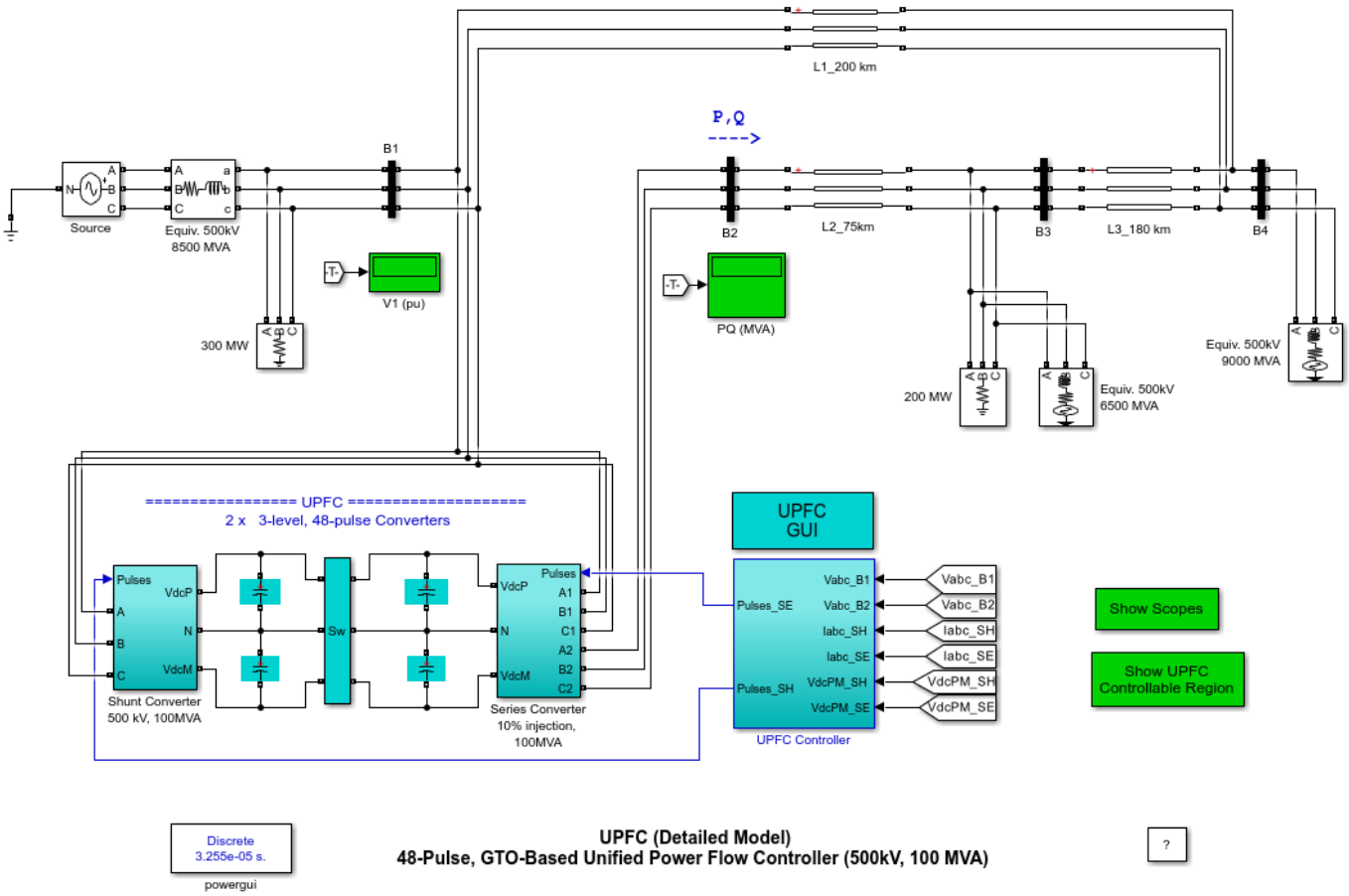
The UPFC reference active and reactive powers are set in the blocks labeled "Pref(pu)" and "Qref(pu)". Initially the Bypass breaker is closed and the resulting natural power flow at bus B3 is 587 MW and -27 Mvar. The Pref block is programmed with an initial active power of 5.87 pu corresponding to the natural flow. Then, at t=10s, Pref is increased by 1 pu (100 MW), from 5.87 pu to 6.87 pu, while Qref is kept constant at -0.27 pu.

Run the simulation and look on the UPFC Scope how P and Q measured at bus B3 follow the reference values. At t=5 s, when the Bypass breaker is opened the natural power is diverted from the Bypass breaker to the UPFC series branch without noticeable transient. At t=10 s, the power increases at a rate of 1 pu/s. It takes one second for the power to increase to 687 MW. This 100 MW increase of active power at bus B3 is achieved by injecting a series voltage of 0.089 pu with an angle of 94 degrees. This results in an approximate 100 MW decrease in the active power flowing through Tr2 (from 899 MW to 796 MW), which now carries an acceptable load. See the variations of active powers at buses B1 to B5 on the VPQ Lines Scope.

# UPFC (Detailed Model)

This example shows a detailed model of a 48-Pulse, GTO-based unified power flow controller (500 kV, 100 MVA).

Pierre Giroux ; Gilbert Sybille (Hydro-Quebec, IREQ)




## Description

A Unified Power Flow Controller (UPFC) is used to control the power flow in a 500 kV transmission system. The UPFC located at the left end of the 75-km line L2, between the 500 kV buses B1 and B2, is used to control the active and reactive powers flowing through bus B2 while controlling voltage at bus B1. It consists of two 100-MVA, three-level, 48-pulse GTO-based converters, one connected in shunt at bus B1 and one connected in series between buses B1 and B2. The shunt and series converters can exchange power through a DC bus. The series converter can inject a maximum of 10% of nominal line-to-ground voltage (28.87 kV) in series with line L2.

This pair of converters can be operated in three modes:

- **Unified Power Flow Controller (UPFC)** mode, when the shunt and series converters are interconnected through the DC bus. When the disconnect switches between the DC buses of the shunt and series converter are opened, two additional modes are available:

- Shunt converter operating as a **Static Synchronous Compensator (STATCOM)** controlling voltage at bus B1
- Series converter operating as a **Static Synchronous Series Capacitor (SSSC)** controlling injected voltage, while keeping injected voltage in quadrature with current.

The mode of operation as well as the reference voltage and reference power values can be changed by means of the "UPFC GUI" block.

The principle of operation of the harmonic neutralized converters is explained in another example entitled "Three-phase 48-pulse GTO converter". This power_48pulsegtoconverter model is accessible in the Power Electronics Models library of examples. When the two converters are operated in UPFC mode, the shunt converter operates as a STATCOM. It controls the bus B1 voltage by controlling the absorbed or generated reactive power while also allowing active power transfer to the series converter through the DC bus. The reactive power variation is obtained by varying the DC bus voltage. The four three-level shunt converters operate at a constant conduction angle (Sigma= 180-7.5 = 172.5 degrees), thus generating a quasi-sinusoidal 48-step voltage waveform. The first significant harmonics are the 47th and the 49th.

When operating in UPFC mode, the magnitude of the series injected voltage is varied by varying the Sigma conduction angle, therefore generating higher harmonic contents than the shunt converter. As illustrated in this example, when the series converter operates in SSSC mode it generates a "true" 48-pulse waveform.

The natural power flow through bus B2 when zero voltage is generated by the series converter (zero voltage on converter side of the four converter transformers) is P=+870 MW and Q=-70 Mvar. In UPFC mode, both the magnitude and phase angle and the series injected voltage can be varied, thus allowing control of P and Q. The UPFC controllable region is obtained by keeping the injected voltage to its maximum value (0.1 pu) and varying its phase angle from zero to 360 degrees. To see the resulting P-Q trajectory, double click the "Show UPFC Controllable Region". Any point located inside the PQ elliptic region can be obtained in UPFC mode.

### Simulation

#### 1. Power control in UPFC mode

Open the UPFC GUI block menu. The GUI allows you to choose the operation mode (UPFC, STATCOM or SSSC) as well as the Pref/Qref reference powers and/or Vref reference voltage settings . Also, in order to observe the dynamic response of the control system, the GUI allows you to specify a step change of any reference value at a specific time.

Make sure that the operation mode is set to "UPFC (Power Flow Control)". The reference active and reactive powers are specified in the last two lines of the GUI menu. Initially, Pref= +8.7 pu/100MVA (+870 MW) and Qref=-0.6 pu/100MVA (-60 Mvar). At t=0.25 sec Pref is changed to +10 pu (+1000MW). Then, at t=0.5 sec, Qref is changed to +0.7 pu (+70 Mvar). The reference voltage of the shunt converter (specified in the 2nd line of the GUI) will be kept constant at Vref=1 pu during the whole simulation (Step Time=0.3*100> Simulation stop time (0.8 sec). When the UPFC is in power control mode, the changes in STATCOM reference reactive power and in SSSC injected voltage (specified respectively in 1st and 3rd line of the GUI) as are not used.

Run the simulation for 0.8 sec. Open the "Show Scopes" subsystem. Observe on traces 1 and 2 of the UPFC scope the variations of P and Q. After a transient period lasting approximately 0.15 sec, the steady state is reached (P=+8.7 pu; Q=-0.6 pu). Then P and Q are ramped to the new settings (P= +10 pu Q=+0.7 pu). Observe on traces 3 and 4 the resulting changes in P Q on the three

transmission lines. The performance of the shunt and series converters can be observed respectively on the STATCOM and SSSC scopes. If you zoom on the first trace of the STATCOM scope, you can observe the 48-step voltage waveform Vs generated on the secondary side of the shunt converter transformers (yellow trace) superimposed with the primary voltage Vp (magenta) and the primary current Ip (cyan). The dc bus voltage (trace 2) varies in the 19kV-21kV range. If you zoom on the first trace of the SSSC scope, you can observe the injected voltage waveforms Vinj measured between buses B1 and B2.

**2. Var control in STATCOM mode**

In the GUI block menu, change the operation mode to "STATCOM (Var Control)". Make sure that the STATCOM references values (1st line of parameters, [T1 T2 Q1 Q2]) are set to [0.3 0.5 +0.8 -0.8 ]. In this mode, the STATCOM is operated as a variable source of reactive power. Initially, Q is set to zero, then at T1=0.3 sec Q is increased to +0.8 pu (STATCOM absorbing reactive power) and at T2=0.5 sec, Q is reversed to -0.8 pu (STATCOM generating reactive power).

Run the simulation and observe on the STATCOM scope the dynamic response of the STATCOM. Zoom on the first trace around t=0.5 sec when Q is changed from +0.8 pu to -0.8 pu. When Q=+0.8 pu, the current flowing into the STATCOM (cyan trace) is lagging voltage (magenta trace), indicating that STATCOM is absorbing reactive power. When Qref is changed from +0.8 to -0.8, the current phase shift with respect to voltage changes from 90 degrees lagging to 90 degrees leading within one cycle. This control of reactive power is obtained by varying the magnitude of the secondary voltage Vs generated by the shunt converter while keeping it in phase with the bus B1 voltage Vp. This change of Vs magnitude is performed by controlling the dc bus voltage. When Q is changing from +0.8 pu to -0.8 pu, Vdc (trace 3) increases from 17.5 kV to 21 kV.

**3. Series voltage injection in SSSC mode**

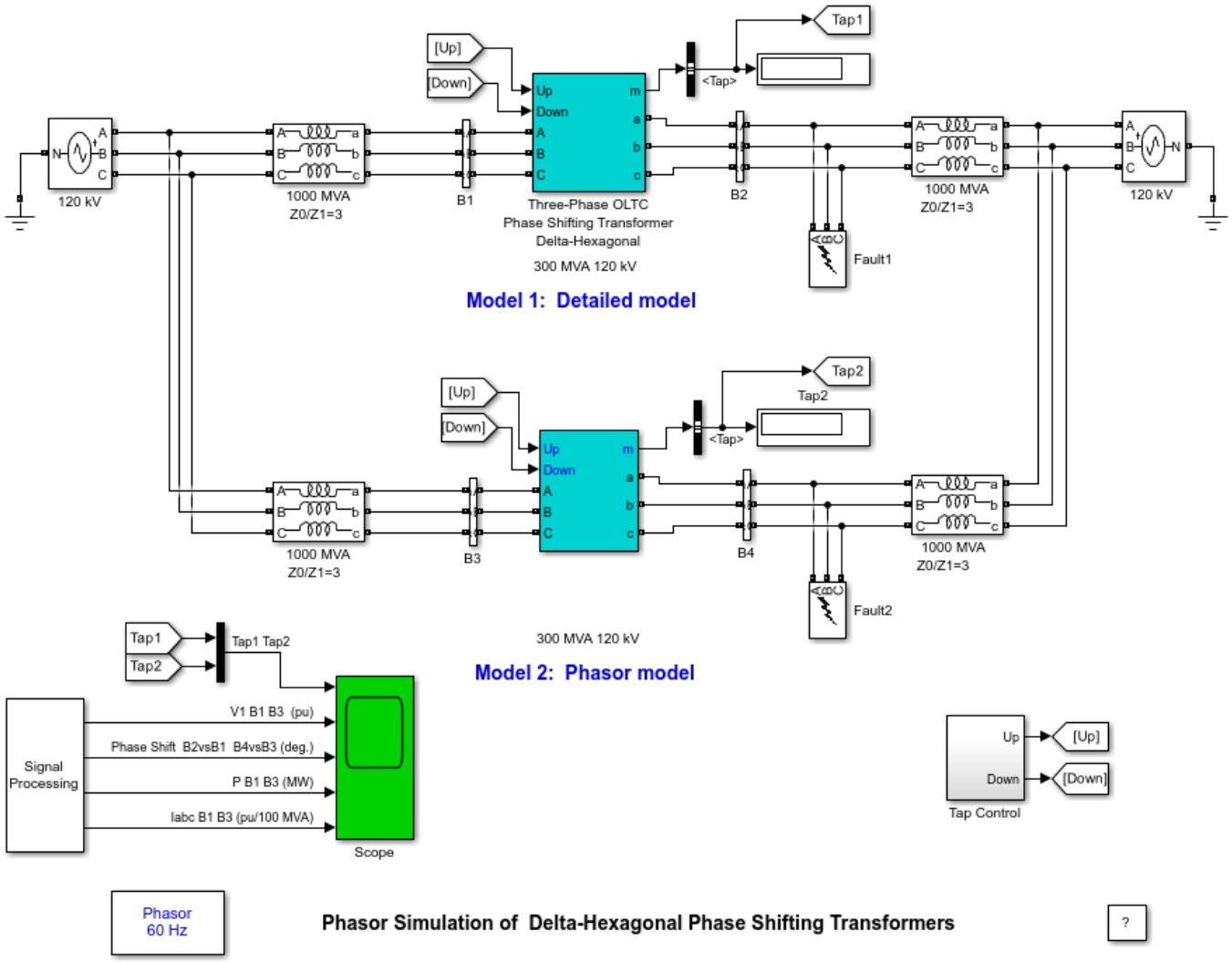
In the GUI block menu change the operation mode to "SSSC (Voltage injection)". Make sure that the SSSC references values (3rd line of parameters) [Vinj_Initial Vinj_Final StepTime] ) are set to [0.0 0.08 0.3 ]. The initial voltage is set to 0 pu, then at t=0.3 sec it will be ramped to 0.8 pu.

Run the simulation and observe on the SSSC scope the impact of injected voltage on P and Q flowing in the 3 transmission lines. Contrary to the UPFC mode, in SSCC mode the series inverter operates with a constant conduction angle (Sigma= 172.5 degrees). The magnitude of the injected voltage is controlled by varying the dc voltage which is proportional to Vinj (3rd trace). Also, observe the waveforms of injected voltages (1st trace) and currents flowing through the SSSC (2nd trace). Voltages and currents stay in quadrature so that the SSSC operates as a variable inductance or capacitance.

# OLTC Phase Shifting Transformer (Phasor Model)

This example shows the operation of two models of delta-hexagonal Phase Shifting Transformer (PST) using On Load Tap Changers (OLTC).

Gilbert Sybille (Hydro-Quebec)




### Description

Two 120 kV 1000 MVA networks are interconnected through a phase shifting transformer (PST). The phase shift can be varied on load by means of On Load Tap Changers (OLTC).

### Simulation

### Open loop control of power transfer

In order to observe impact of phase shift on power transfer, the phase shift is increased from zero to 32.2 degrees lagging (tap +5), then phase shift is reduced to zero and increased again up to 32.2

degrees leading. This is performed by sending 5 pulses to the "Up" input, and then, 10 pulses to the "Down" input ". As the tap selection is a relatively slow mechanical process (3 sec per tap as specified in the "Tap selection time" parameter of the block menus), the simulation Stop time is set to 50 s.

Start simulation and observe PST operation on the Scope. Results obtained with the two models are superimposed on five traces.

- Trace 1 shows the tap position.
- Trace 2 shows a superposition of positive-sequence voltages measured at bus B1 (yellow) and bus B3 (magenta).
- Trace 3 shows the phase shifts of positive-sequence voltages measured at output terminals (abc) with respect to input terminals (ABC).
- Trace 4 compares the active power measured at bus B1 (yellow) and bus B3 (magenta).
- Trace 5 compares the phase currents at bus B1 (yellow) and bus B3 (magenta).

When simulation starts the OLTCs are at position 0 (zero phase shift). As the two networks are symmetrical with both internal angles set at 0 degree, there is no current flowing. Then, phase shift is increased and bus B2 (or B4) is lagging bus B1 (or B3). As B2 is lagging the internal voltage of the source located on right side, power flows from right to left. Power measured from left to right is therefore negative for positive tap positions. The maximum power is obtained at tap +5, or -5 when the phase shift is respectively -32.2 degrees and +32.2 degrees. The active power can be computed from P= V1.V2*sin(psi)/ (X1 +X2 + Xpst) , where: V1=V2=internal voltages= 1.0 pu ; X1= X2 = network reactances = 1pu /1000 MVA Xpst= PST leakage reactance at tap 5. The PST leakage reactance varies with tap position (from zero at tap zero to 0.15 pu at maximum tap (10)). The positive-sequence impedance of the phasor model is available as a signal at its measurement output "m". The reactance obtained at tap 5 is Xpst=0.1067 pu/ 300 MVA. The total reactance expressed in pu/100 MVA is X= 0.1 +0.1 + 0.1067/3 =0.2356 pu/100 MVA. The expected active power at tap 5 is P= 1*sin(32.2deg)/0.2356 = 2.26 pu/100 MVA or 226 MW, which corresponds well with the measured value on trace 4 (224 MW). Because of the voltage developed across the PST leakage reactance, the phase shift measured between PST input and output voltages (trace 3) is lower than the expected value. For example, 27.2 degrees is obtained at tap 5, instead of the 32.2 degrees theoretical value computed at no load. The phase shift variation depends on load current.

**Initializing the phasor model**

For the phasor model to start initialized at t=0, the current sources used in the model must be initialized with current values corresponding to steady state. Suppose that you want to start with the initial tap position 5. First, in the two block menus, "set Initial tap" parameter to 5. Then disconnect the signals connected to the "Up" and "Down" inputs of the two models, so that the taps stay at position 5. If you start simulation you will notice a transient in the phasor model signals at t=0 because the model is not initialized. Use the "Steady-State Voltages and Currents" option of the powergui to obtain the initial current flowing in the detailed model at bus B4. The phase A output current identified "B2/Ia" is 1129.4 A rms, 169 degrees. This current converted to per unit based on PST rating is 1129/1443 = 0.7824 pu/ 300MVA. Specify [ 0.7824 169] in the "Initial pos. seq. output current" parameter. If you now restart simulation, you should observe no transient at t=0.

**Operation under unbalanced conditions**

The phasor model is valid for unbalanced conditions. If you check "Phase A Fault" in the two fault breakers, a single phase fault will be applied at t=5s. The currents measured at buses B1 and B3 should be identical. (For example at tap position +5: Ia=3.48 pu, Ib=2.25 pu Ic=2.10 pu ).

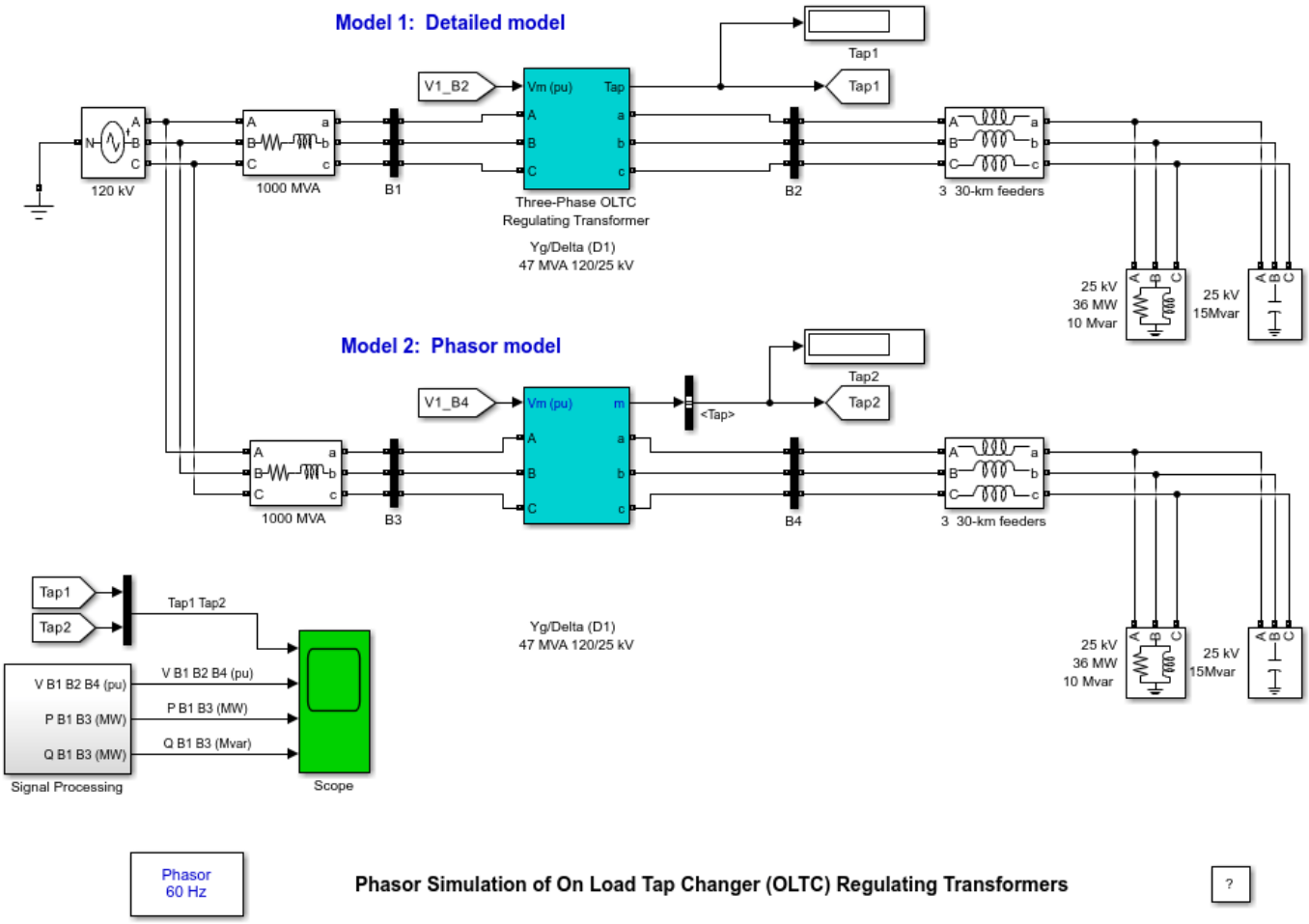
**Simulation with phasor model only**

In order to appreciate the gain in simulation speed provided by the phasor model, delete the detailed PST model and replace it with a duplicate of the phasor model. Reconnect control signals to the "Up" and "Down" inputs. Restart simulation. The model runs approximately 5 times faster, mainly because the OLTC switches of the detailed model are not simulated.

# OLTC Regulating Transformer (Phasor Model)

This example shows the operation of two models of on load tap changer (OLTC) regulating transformer.

Gilbert Sybille (Hydro-Quebec)




## Description

A 25 kV distribution network consisting of three 30-km distribution feeders connected in parallel supplies power to a 36 MW /10 Mvar load (0.964 PF lagging) from a 120 kV, 1000 MVA system and a 120kV/25 kV OLTC regulating transformer. Reactive power compensation is provided at load bus by a 15 Mvar capacitor bank. The same circuit is duplicated in order to compare the performance of two different models of OLTC transformers:

- Model 1 is a detailed model where all OLTC switches and transformer characteristics are represented. This model can be used with either continuous or discrete 'simulation type' modes of Powergui to get detailed wave shapes or with the phasor simulation method to observe variations of phasor voltages and currents. To simulate Model 1 in continuous or discrete you need to delete Model 2 from the example.

- Model 2 is a simplified phasor model where the transformer and OLTC are simulated by current sources. This model can be used only with the phasor 'simulation type' mode of Powergui. It is much faster to execute and it should be the preferred model for transient stability studies, when several such devices are used in the same system.

Both OLTC transformer models implement a three-phase regulating transformer rated 47 MVA, 120 kV/25 kV, Wye/ Delta, with the OLTC connected on the high voltage side (120 kV). The OLTC transformers are used to regulate system voltage at 25 kV buses B2 and B4.

Voltage regulation is performed by varying the transformer turn ratio. This is obtained by connecting on each phase, a tapped winding (regulation winding) in series with each 120/sqrt(3) kV winding. Nine (9) OLTC switches allow selection of 8 different taps (tap positions 1 to 8, plus tap 0 which provides nominal 120kV/25 kV ratio). A reversing switch included in the OLTC allows reversing connections of the regulation winding so that it is connected either additive (positive tap positions) or subtractive (negative tap positions). For a fixed 25 kV secondary voltage, each tap provides a voltage correction of +/-0.01875 pu or +/-1.875% of nominal 120 kV voltage. Therefore, a total of 17 tap positions, including tap 0, allow a voltage variation from 0.85 pu (102 kV) to 1.15 pu (138 kV) by steps of 0.01875 pu (2.25 kV).

The positive-sequence voltages measured at buses B2 and B4 are provided as inputs to the voltage regulators (input 'Vmeas' of the transformer blocks). Open the two transformer block menus and look at their parameters. The voltage regulators are in service ('Voltage regulator' parameter = 'on'). The reference voltage is set to 1.04 pu. In order to start simulation with 25-KV voltages close to 1.04 pu at buses B2 and B4, the initial tap positions are set at -4, so that the transformers are boosting the voltage by a factor 1/(1-4*0.01875)=1.081.

The detailed model is built with a fixed number of taps (8). Note that the phasor model provides more flexibility as it allows selection of primary and secondary winding connections (wye or Delta) as well as changing the number of taps and using the OLTC either on primary or secondary side.

Look under mask to see how the transformer models are built. The detailed model is built from three Multi-Winding Transformer blocks and three OLTC subsystems which contain switches performing tap selection and reversal of the regulation winding. The tap transition is performed by temporarily short-circuiting two adjacent transformer taps through resistors (5 ohm resistances and 60 ms transition time as specified in the block menu). The phasor model is built with current sources emulating the transformer impedance which depends on winding resistances, leakage reactances and tap position. Both models use a voltage regulator that generates pulses at the 'Up' or 'Down' outputs and orders a tap change either in the positive or negative direction. The voltage regulation depends on the specified dead band (DB = two times the voltage step or 0.0375 pu). This means that the maximum voltage error at buses B2 and B4 should be 0.01875 pu. As long as the maximum tap number is not reached (-8 or +8), voltage should stay in the range : ( Vref-DB/2< V<1.04+DB/2 ) = (1.021< V< 1.059).

**Simulation**

As tap selection is a relatively slow mechanical process (4 sec per tap as specified in the 'Tap selection time' parameter of the block menus), the simulation Stop time is set to 2 minutes (120 s). The Three-Phase Programmable Voltage Source is used to vary the 120 kV system voltage in order to observe the OLTC performance. Initially, the source is generating its nominal voltage. Then, voltage is successively decreased (0.95 pu at t = 10 s) and increased (1.10 pu at t = 50 s).

Start the simulation and observe OLTC operation on the Scope.

- Trace 1 shows the tap position.
- Trace 2 shows a superposition of positive-sequence voltages at 120 kV bus B1 (yellow ), at 25 kV bus B2 (magenta) and bus B4 (cyan).
- Traces 3 and 4 show the active and reactive powers measured on 120 kV side (buses B1 and B3).

When simulation starts the OLTCs are at position -4 and the resulting voltage at bus B2 and B4 is 1.038 pu. At t=10 s, the source internal voltage is suddenly lowered to 0.95 pu , so that the 25 kV voltages drop to 0.986 pu, outside of the permitted voltage range (1.021< V<1.059). The voltage regulator then orders further voltage boosting and the OLTC stabilizes at tap=-6 (V=1.025 pu) At t=50 s, the source internal voltage is suddenly increased to 1.10 pu , so that the 25 kV voltages now reach to 1.19 pu. The voltage regulator then starts to decrease voltage by moving taps in the upward direction and the OLTCs stabilize at tap=+1 (V=1.043 pu).

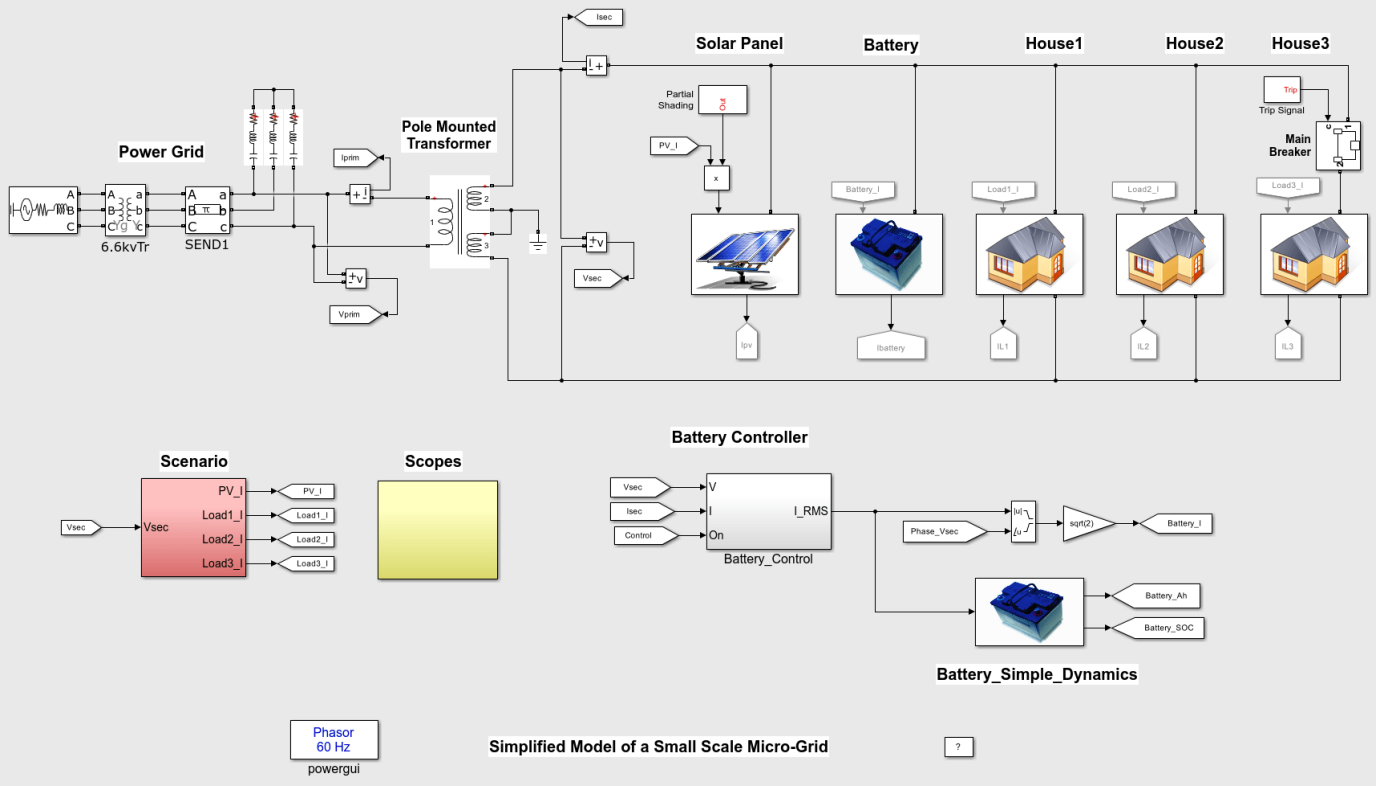
**Simulation with phasor model only**

In order to appreciate the gain in simulation speed provided by the phasor model, delete the detailed transformer model and replace it with a duplicate of the phasor model. Restart simulation. The model runs approximately 2.5 times faster, mainly because the OLTC switches of the detailed model are not simulated.

**Note:** The voltage glitches observed with the phasor model when the source voltage is stepped down (t= 10s) and up (t=50 s) can be ignored. They are caused by the first order transfer functions (one cycle time constant) which are used inside the model to break algebraic loops.

# Simplified Model of a Small Scale Micro-Grid

This example shows the behavior of a simplified model of a small-scale micro grid during 24 hours on a typical day. The model uses Phasor solution provided by Specialized Power Systems in order to accelerate simulation speed.




## Description

The micro-grid is a single-phase AC network. Energy sources are an electricity network, a solar power generation system and a storage battery.

The storage battery is controlled by a battery controller. It absorbs surplus power when there is excess energy in the micro-network, and provides additional power if there is a power shortage in the micro-network. Three ordinary houses consume energy (maximum of 2.5 kW) as electric charges.

The micro-array is connected to the power network via a transformer mounted on a post which lowers the voltage of 6.6 kV to 200 V.

The solar power generation and storage battery are DC power sources that are converted to single-phase AC. The control strategy assumes that the microarray does not depend entirely on the power supplied by the power grid, and the power supplied by the solar power generation and storage are sufficient at all times.

## Simulation

From 20h to 4h, the solar power generation is 0 W. It reaches the peak amount (5 kW) from 14h to 15h.

As a typical load change in ordinary houses, the amount of electric power load reaches peak consumption at 9h (6,500 W), 19h, and 22h (7,500 W).

From 0h to 12h and from 18h to 24h, battery control is performed by battery controller. The battery control performs tracking control of the current so that active power which flows into system power from the secondary side of the pole transformer is set to 0. Then, the active power of secondary side of the pole mounted transformer is always around zero.

The storage battery supplies the insufficient current when the power of the micro-grid is insufficient and absorbs surplus current from the micro-grid when its power is surpasses the electric load.

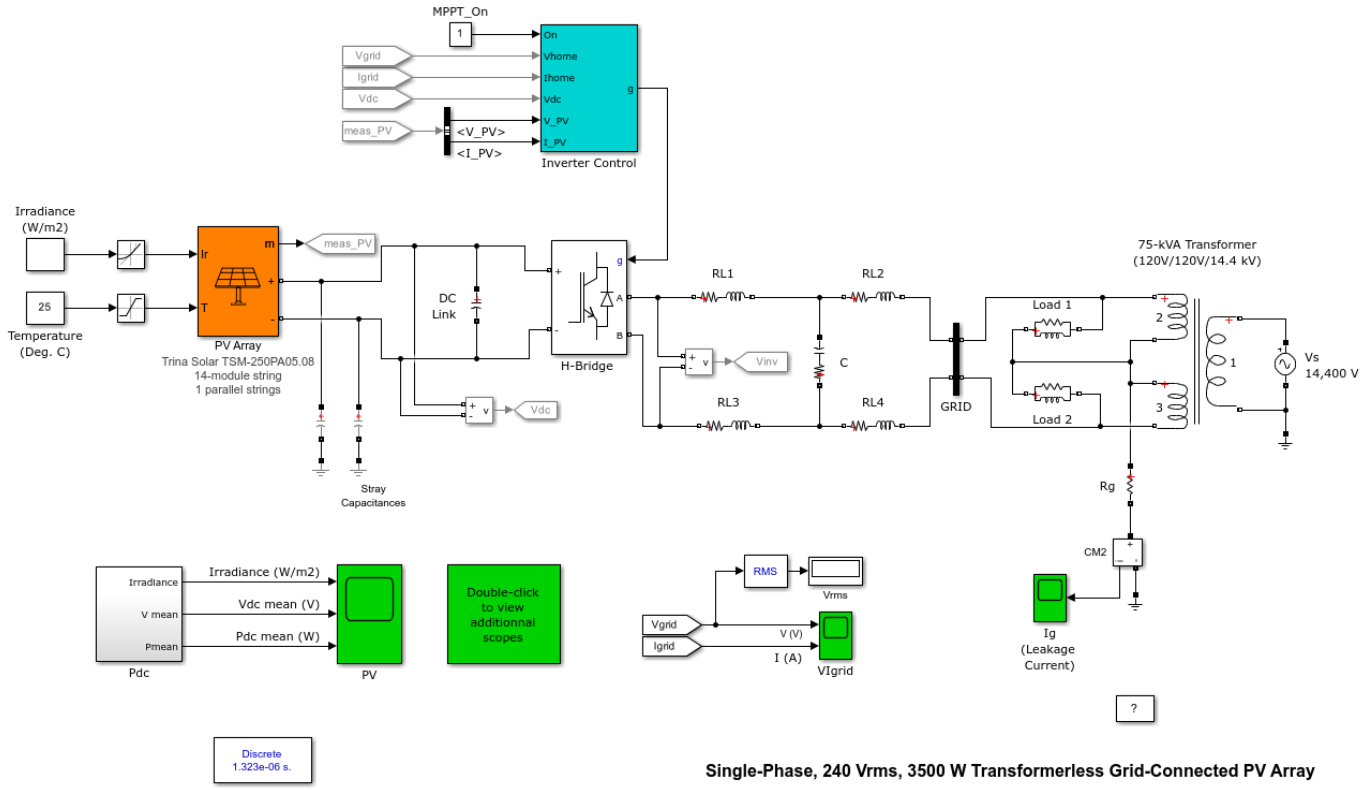
From 12h to 18h, battery control is not performed. SOC (State Of Charge) of the storage battery is fixed to a constant and does not change since charge or discharge of the storage

battery are not performed by the battery controller. When there is a power shortage in the micro-grid, the system power supplies insufficient power. When there is a surplus power in the micro-grid, surplus power is returned to the system power.

At 8h, electricity load No. 3 of an ordinary house is set to OFF for 10 sec by the breaker. A spike is observed in the active power on the secondary side of the pole transformer and the electric power of the storage battery.

# Single-Phase, 240 Vrms, 3500 W Transformerless Grid-Connected PV Array

This example shows the operation of a typical transformerless photovoltaic (PV) residential system connected to the electrical utility grid.




### PV Array

The SPS PV array model implements a PV array built of series- and parallel-connected PV modules. It allows modeling a variety of preset PV modules available from NREL System Advisor Model (Jan. 2014) as well as a user-defined PV module. The PV array block has two inputs that allow you to supply varying sun irradiance (input Ir in W/m^2) and temperature (input T in degrees C) data.

In our example, the PV array consists of one string of 14 Trina Solar TSM-250 modules connected in series. At 25 degrees C and with a solar irradiance of 1000 W/m2, the string can produce 3500 W.

Two small capacitors, connected on the + and - terminals of the PV array, are used to model the parasitic capacitance between the PV modules and the ground.

### One-phase DC/AC Converter

The inverter is modeled using a PWM-controlled single-phase full-bridge IGBT module ( H-bridge). The topology of the grid-side filter is the classical LCL configuration with the inductors split equally between the line and the neutral branches.

### Inverter Control

The control system contains five major Simulink®-based subsystems:

- MPPT Controller: The Maximum Power Point Tracking (MPPT) controller is based on the 'Perturb and Observe' technique. This MPPT system automatically varies the VDC reference signal of the inverter VDC regulator in order to obtain a DC voltage which will extract maximum power from the PV string.
- VDC Regulator: Determine the required Id (active current) reference for the current regulator.
- Current Regulator: Based on the current references Id and Iq (reactive current), the regulator determines the required reference voltages for the inverter. In our example, the Iq reference is set to zero.
- PLL & Measurements: Required for synchronization and voltage/current measurements.
- PWM Generator: Use the PWM bipolar modulation method to generate firing signals to the IGBTs. In our example, the PWM carrier frequency is set to 3780 Hz (63*60).

### Load & Utility Grid

The grid is modeled using a typical pole-mounted transformer and an ideal AC source of 14.4 kVrms. The transformer 240V secondary winding is center-tapped and the central neutral wire is grounded via a small resistance Rg. The residential load (10 kW / 4 kvar @ 240 Vrms) is equally distributed between the two "hot" (120 V) terminals.

### Simulation

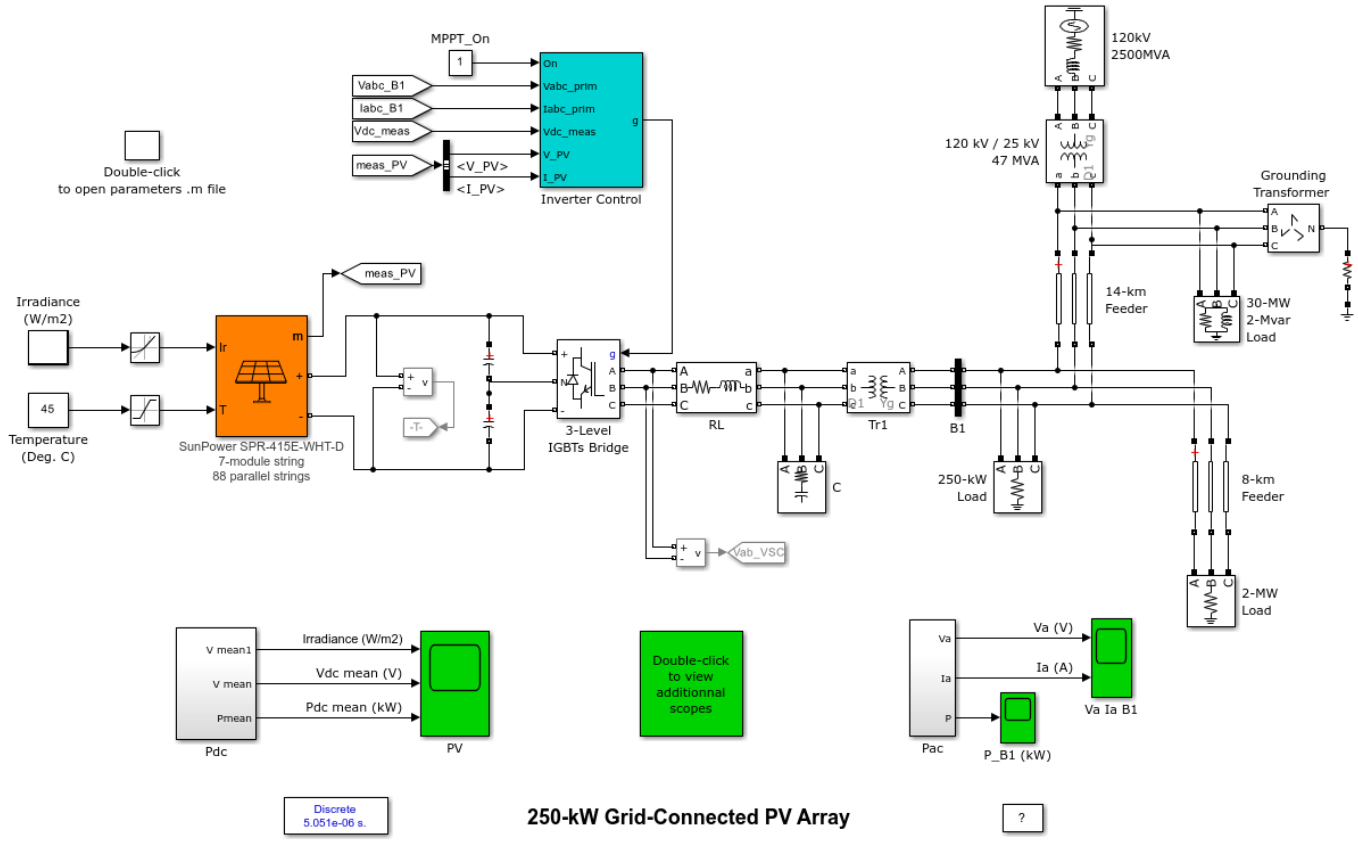
Run the simulation and observe the resulting signals on the various scopes.

The initial input irradiance to the PV array model is 250 W/m2 and the operating temperature is 25 degrees C. When steady-state is reached (around t=0.25 sec.), we get a PV voltage (Vdc_mean) of 424.5 V and the power extracted (Pdc_mean) from the array is 856 W. At t=0.4 sec, sun irradiance is rapidly ramped up from 250 W/m^2 to 750 W/m^2. Due to the MPPT operation, the control system increases the VDC reference to 434.2 V in order to extract maximum power from the PV string (2624 W). These values correspond well to the expected values. To confirm that, use the Plot button of the PV Array menu to plot the I-V and P-V characteristics of the PV string based on the manufacturer specifications.

If you look at the leakage current (Ig scope), you will notice that there are no current flowing through the stray capacitance of the PV modules. This is due to PWM method used and the filter topology. Now, if you select the PWM unipolar modulation method (using the Inverter control menu) and repeat the simulation, you will see a significant leakage current in the system.

# 250-kW Grid-Connected PV Array

This example shows a detailed model of a 250-kW PV array connected to a 25-kV grid via a three-phase converter.




### PV Array

The PV array consists of 86 parallel strings. Each string has 7 SunPower SPR-415E modules connected in series. Note that the model menu allows you to plot the I-V and P-V characteristics of the selected module or of the whole array.

### Three-phase DC/AC Converter

The converter is modeled using a 3-level IGBT bridge PWM-controlled. The inverter choke RL and a small harmonics filter C are used to filter the harmonics generated by the IGBT bridge. A 250-kVA 250V/25kV three-phase transformer is used to connect the inverter to the utility distribution system.

### Inverter Control

The control system contains five major Simulink®-based subsystems:

- MPPT Controller: The Maximum Power Point Tracking (MPPT) controller is based on the 'Perturb and Observe' technique. This MPPT system automatically varies the VDC reference signal of the inverter VDC regulator in order to obtain a DC voltage which will extract maximum power from the PV array.

- VDC Regulator: Determine the required Id (active current) reference for the current regulator.
- Current Regulator: Based on the current references Id and Iq (reactive current), the regulator determines the required reference voltages for the inverter. In our example, the Iq reference is set to zero.
- PLL & Measurements: Required for synchronization and voltage/current measurements.
- PWM Generator: Generate firing signals to the IGBTs based on the required reference voltages. In our example, the carrier frequency is set to 1980 Hz (33*60).

### Utility Grid

The grid is modeled as a typical North American distribution grid. It included two 25-kV feeders, loads, grounding transformer and an equivalent 120-kV transmission system.

### Simulation

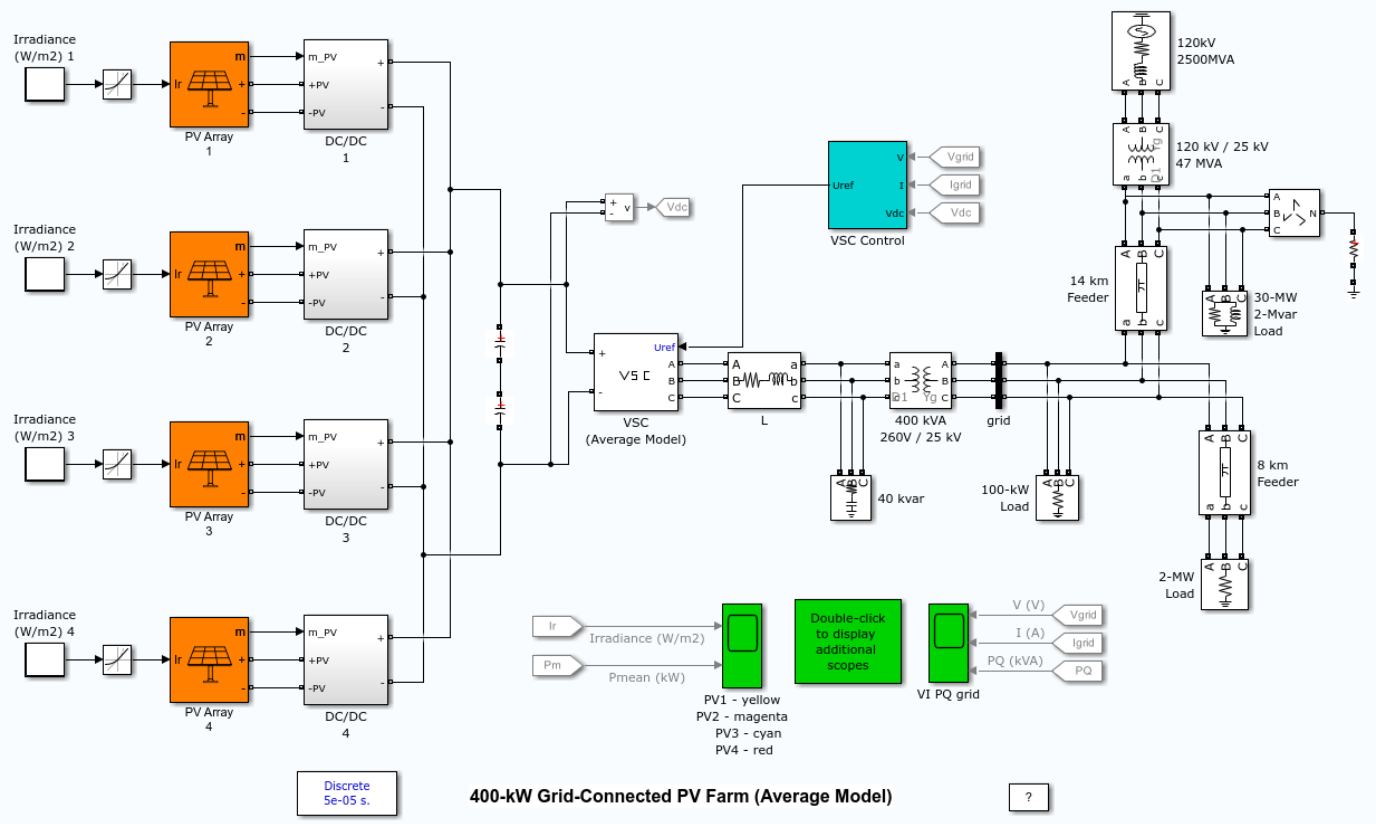
Run the simulation and observe the resulting signals on the various scopes.

The initial input irradiance to the PV array model is 1000 W/m2 and the operating temperature is 45 degrees C. When steady-state is reached (around t=0.15 sec.), we get a PV voltage (Vdc_mean) of 481 V and the power extracted (Pdc_mean) from the array is 236 kW. These values correspond very well to the expected values from the PV module manufacturer specifications.

At t=0.3 sec, sun irradiance is rapidly ramped down from 1000 W/m^2 to 200 W/m^2. Due to the MPPT operation, the control system reduces the VDC reference to 464 V in order to extract maximum power from the PV array (46 kW).

# 400-kW Grid-Connected PV Farm (Average Model)

This example shows an average model of a small PV farm (400 kW) connected to a 25-kV grid using two-stage converter.




## Description

The PV farm consists of four PV arrays delivering each a maximum of 100 kW at 1000 W/m2 sun irradiance. A single PV array block consist of 64 parallel strings where each string has 5 SunPower SPR-315E modules connected in series.

Each PV array is connected to a DC/DC converter (average model). The outputs of the boost converters are connected to a common DC bus of 500 V. Each boost is controlled by individual Maximum Power Point Trackers (MPPT). The MPPTs use the "Perturb and Observe" technique to vary the voltage across the terminals of the PV array in order get the maximum possible power.

A three-phase Voltage Source Converter (VSC) converts the 500 V DC to 260 V AC and keeps unity power factor. A 400-kVA 260V/25kV three-phase coupling transformer is used to connect the converter to the grid. The grid model consists of typical 25-kV distribution feeders and 120-kV equivalent transmission system.

In the average model the boost and VSC converters are represented by equivalent voltage sources generating the AC voltage averaged over one cycle of the switching frequency. Such a model does not represent harmonics, but the dynamics resulting from control system and power system interaction is

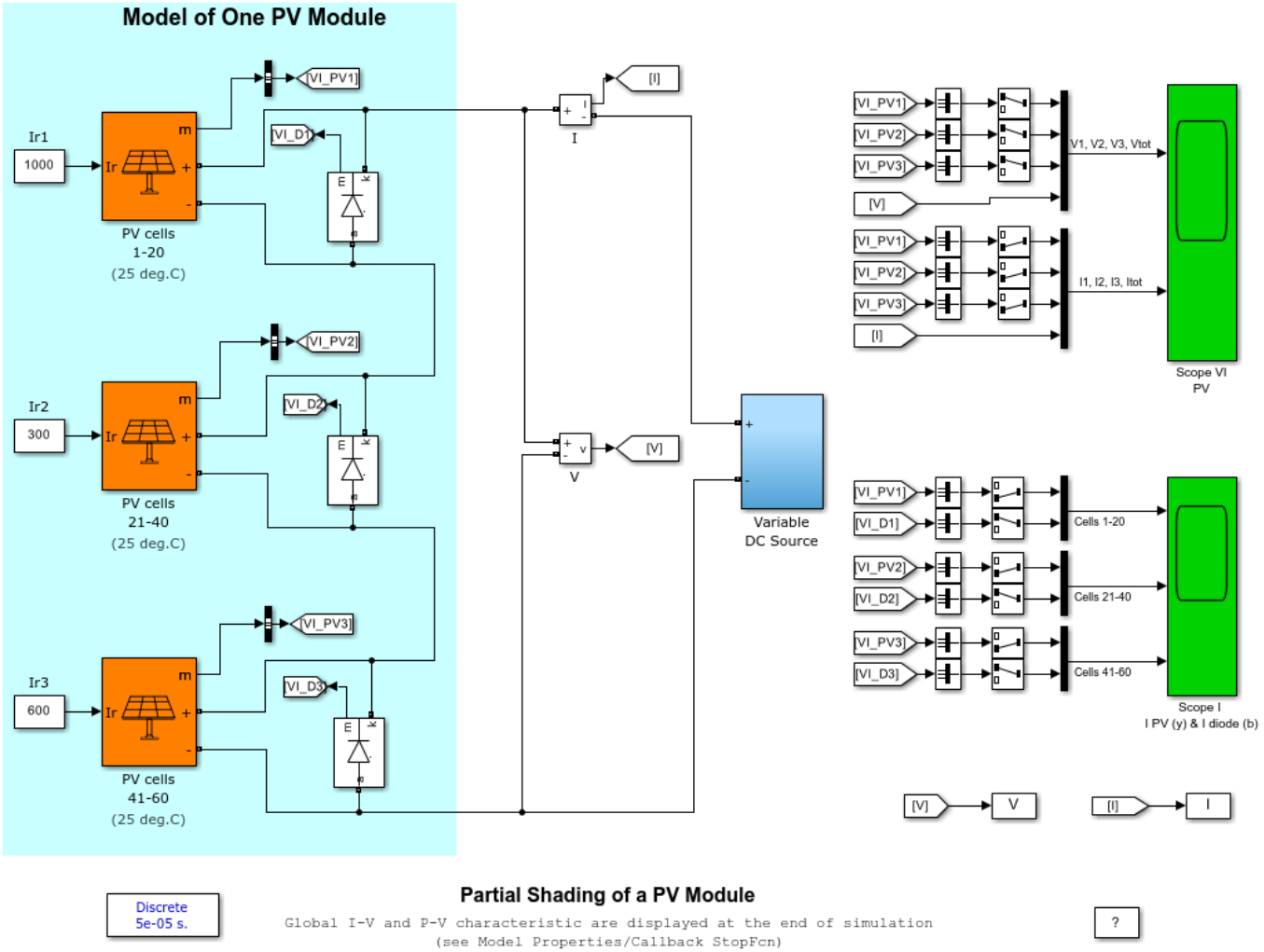
preserved. This model allows using much larger time steps (50 us), resulting in a much faster simulation.

### Simulation

Start the simulation and see the resulting signals on the various scopes. This three-second simulation allows us to observe the operation of each individual PV Array system under varying irradiances.

# Partial Shading of a PV Module

This example shows partial shading of a 250-W PV module consisting of 60 cells connected in series.




### Description

The PV module is connected to a variable DC voltage source for measuring its I-V and P-V characteristics. It is modeled as three strings of 20 series-connected cells in parallel with bypass diodes that allow current flow when cells are shaded or damaged. Standard irradiance of 1000 W/m2 is applied on the first string of 20 cells while partial shading is applied on strings 2 (cells 21-40) and string 3 (cells 41-60), resulting in respective irradiances of 300 W/m2 and 600 W/m2.

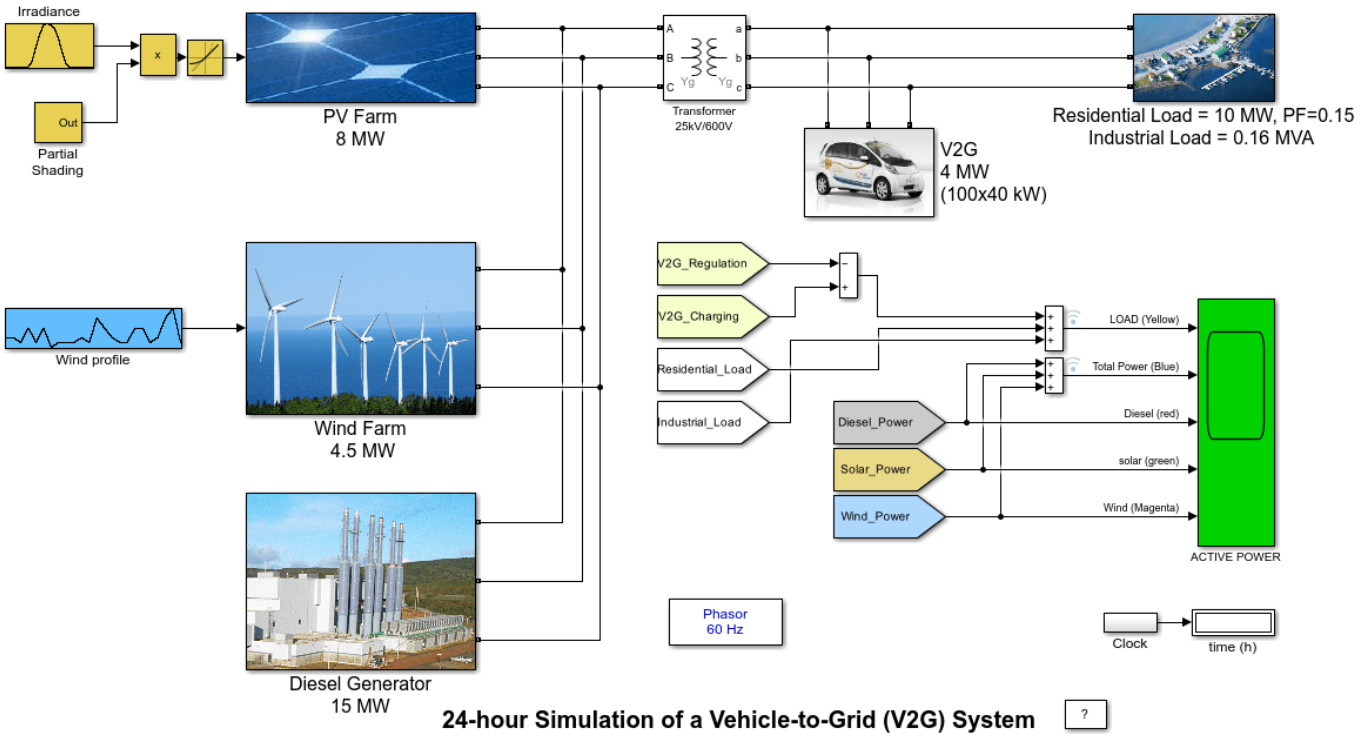
### Simulation

Simulate the model. The global I-V and P-V characteristics are plotted at the end of simulation. Note that the P-V curve exhibits three maxima. When this PV module is connected to a voltage-sourced converter, this may be challenging for the Maximum Power Point Tracking (MPPT) algorithm to converge on the highest peak. The Global Maximum Power Point (GMPP) (Pm= 104 W) indicated by a

red circle on the figure is 34% lower than the expected maximum power (250/3*(1 +0.3 +0.6) = 158 W).

# 24-hour Simulation of a Vehicle-to-Grid (V2G) System

This example shows a vehicle-to-grid system used to regulate the frequency on a microgrid when events occur during a full day. The phasor mode of Specialized Power Systems allows a fast simulation of a 24 hour scenario.




### Description

The microgrid is divided into four important parts: A diesel generator, acting as the base power generator; A PV farm combined with a wind farm, to produce renewable energy; a V2G system installed next to the last part of the system which is the load of the grid. The size of the microgrid represents approximately a community of a thousand households during a low consumption day in spring or fall. There are 100 electric vehicles in the base model which means that there is a 1:10 ratio between the cars and the households. This is a possible scenario in a foreseeable future.

### Diesel Generator

The diesel generator balances the power consumed and the power produced. We can determine the frequency deviation of the grid by looking at the rotor speed of its synchronous machine.

### Renewable Energy

There are two sources of renewable energy in this microgrid. First, a PV farm produces energy proportional to three factors: the size of the area covered by the PV farm, the efficiency of the solar panels and the irradiance data. Second, a simplified model of a wind farm produces electrical power following a linear relationship with the wind. When the wind reaches a nominal value, the wind farm produces the nominal power. The wind farm trips from the grid when the wind speed exceeds the maximum wind value, until the wind gets back to its nominal value.

### Vehicle-to-Grid

The V2G has two functions: Controls the charge of the batteries connected to it and uses the available power to regulate the grid when an event occurs during the day. The block implements five different car-user profiles:

- Profile #1: People going to work with a possibility to charge their car at work
- Profile #2: People going to work with a possibility to charge their car at work but with a longer ride
- Profile #3: People going to work with no possibility to charge their car at work
- Profile #4: People staying at home
- Profile #5: People working on a night shift

### Load

The load is composed of residential load and an asynchronous machine that is used to represents the impact of an industrial inductive load (like a ventilation system) on the microgrid. The residential load follows a consumption profile with a given power factor. The asynchronous machine is controlled by a square relation between the rotor speed and the mechanical torque.

### Scenario

The simulation lasts 24 hours. The solar intensity follows a normal distribution where the highest intensity is reached at midday. The wind varies greatly during the day and has multiple peaks and lows. The residential load follows a typical pattern similar to a normal household consumption. The consumption is low during the day and increases to a peak during the evening, and slowly decreases during the night. Three events will affect the grid frequency during the day:

- The kick-off of the asynchronous machine early at the third hour
- A partial shading at noon affecting the production of solar power
- A wind farm trip at 22h when the wind exceeds the maximum wind power allowed

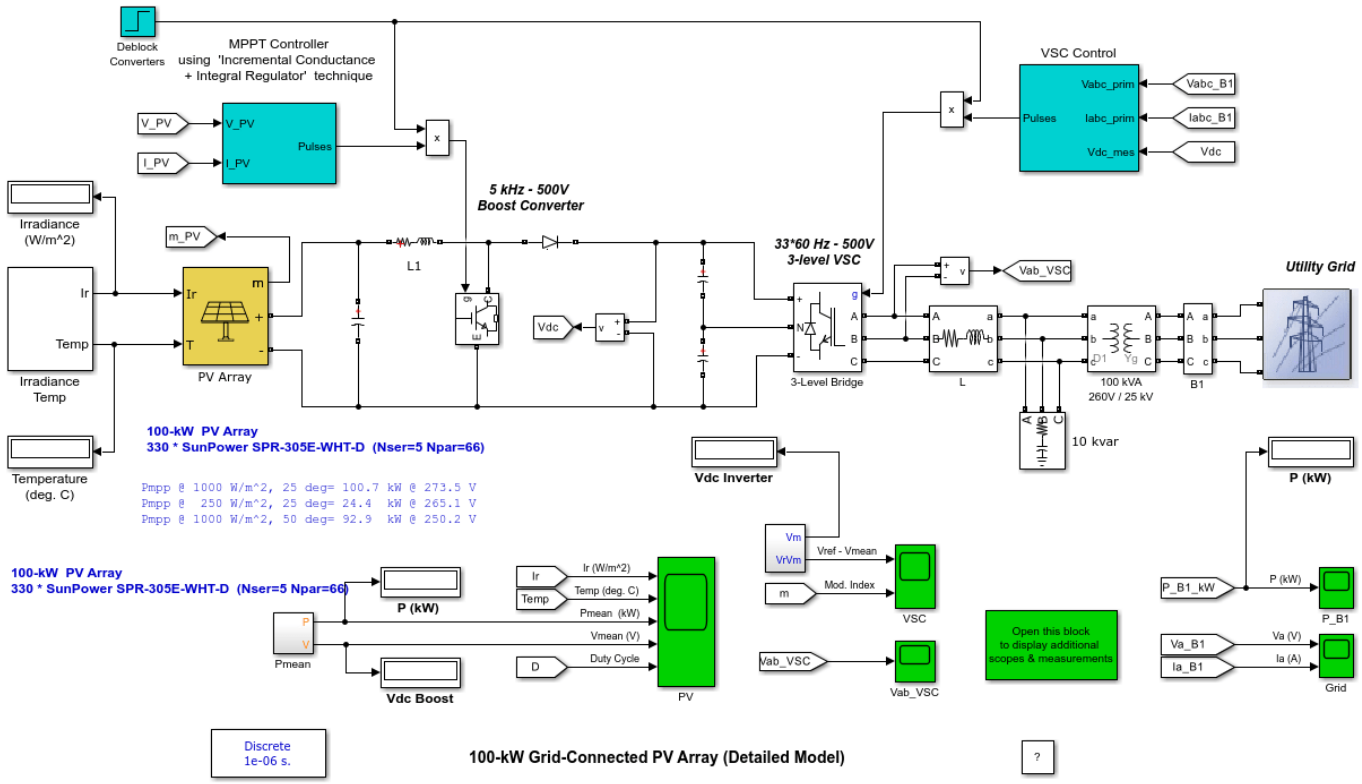
### Simulation

Run the example model and observe the power profile (generation and consumption) during the entire day. You can see the contribution of each generation system and the impact of the Vehicle-to-Grid system on peak regulation.

# Detailed Model of a 100-kW Grid-Connected PV Array

This example shows a detailed model of a 100-kW array connected to a 25-kV grid via a DC-DC boost converter and a three-phase three-level VSC.

Pierre Giroux, Gilbert Sybille (Hydro-Quebec, IREQ) Carlos Osorio, Shripad Chandrachood (The MathWorks)




## Description

A 100-kW PV array is connected to a 25-kV grid via a DC-DC boost converter and a three-phase three-level Voltage Source Converter (VSC). Maximum Power Point Tracking (MPPT) is implemented in the boost converter by means of a Simulink® model using the 'Incremental Conductance + Integral Regulator' technique.

Another example (see power_PVarray_grid_avg model) uses average models for the DC_DC and VSC converters. In this average model the MPPT controller is based on the 'Perturb and Observe' technique.

The detailed model contains the following components:

- **PV array** delivering a maximum of 100 kW at 1000 W/m^2 sun irradiance.
- **5-kHz DC-DC boost converter** increasing voltage from PV natural voltage (273 V DC at maximum power) to 500 V DC. Switching duty cycle is optimized by a MPPT controller that uses the 'Incremental Conductance + Integral Regulator' technique. This MPPT system automatically varies the duty cycle in order to generate the required voltage to extract maximum power.

- **1980-Hz 3-level 3-phase VSC**. The VSC converts the 500 V DC link voltage to 260 V AC and keeps unity power factor. The VSC control system uses two control loops: an external control loop which regulates DC link voltage to +/- 250 V and an internal control loop which regulates Id and Iq grid currents (active and reactive current components). Id current reference is the output of the DC voltage external controller. Iq current reference is set to zero in order to maintain unity power factor. Vd and Vq voltage outputs of the current controller are converted to three modulating signals Uabc_ref used by the PWM Generator. The control system uses a sample time of 100 microseconds for voltage and current controllers as well as for the PLL synchronization unit. Pulse generators of Boost and VSC converters use a fast sample time of 1 microsecond in order to get an appropriate resolution of PWM waveforms.
- **10-kvar capacitor bank** filtering harmonics produced by VSC.
- **100-kVA 260V/25kV three-phase coupling transformer**.
- **Utility grid** (25-kV distribution feeder + 120 kV equivalent transmission system).

The 100-kW PV array uses 330 SunPower modules (SPR-305E-WHT-D). The array consists of 66 strings of 5 series-connected modules connected in parallel (66*5*305.2 W= 100.7 kW).

The 'Module' parameter of the PV Array block allows you to choose among various array types of the NREL System Advisor Model (https://sam.nrel.gov/).

The manufacturer specifications for one module are:

- Number of series-connected cells : 96
- Open-circuit voltage: Voc= 64.2 V
- Short-circuit current: Isc = 5.96 A
- Voltage and current at maximum power : Vmp =54.7 V, Imp= 5.58 A

The PV array block menu allows you to plot the I-V and P-V characteristics for one module and for the whole array.

The PV array block has two inputs that allow you varying sun irradiance (input 1 in W/m^2) and temperature (input 2 in degrees C). The irradiance and temperature profiles are defined by a Signal Builder block which is connected to the PV array inputs.

### Simulation

Run the model and observe the following sequence of events on Scopes.

Simulation starts with standard test conditions (25 degrees C, 1000 W/m^2).

From t=0 sec to t= 0.05 sec, pulses to Boost and VSC converters are blocked. PV voltage corresponds to open-circuit voltage (Nser*Voc=5*64.2=321 V, see Vmean trace on PV scope). The three-level bridge operates as a diode rectifier and DC link capacitors are charged above 500 V (see Vmean trace on VSC scope ).

At t=0.05 sec, Boost and VSC converters are de-blocked. DC link voltage is regulated at Vdc=500V. Duty cycle of boost converter is fixed (D= 0.5 as shown on PV scope).

Steady state is reached at t=0.25 sec. Resulting PV voltage is therefore V_PV = (1-D)*Vdc= (1-0.5)*500=250 V (see Vmean trace on PV scope). The PV array output power is 96 kW (see Pmean trace on PV scope) whereas specified maximum power with a 1000 W/m^2 irradiance is 100.7 kW. Observe on Scope Grid that phase A voltage and current at 25 kV bus are in phase (unity power

factor). At t=0.4 sec MPPT is enabled. The MPPT regulator starts regulating PV voltage by varying duty cycle in order to extract maximum power. Maximum power (100.4 kW) is obtained when duty cycle is D=0.454.

At t=0.6 sec, PV array mean voltage =274 V as expected from PV module specifications (Nser*Vmp=5*54.7= 273.5 V).

From t=0.6 sec to t=1.1 sec, sun irradiance is ramped down from 1000 W/m^2 to 250 W/m^2. MPPT continues tracking maximum power.

At t=1.2 sec when irradiance has decreased to 250 W/m^2, duty cycle is D=0.461. Corresponding PV voltage and power are Vmean= 268 V and Pmean=24.3 kW. Note that the MMPT continues tracking maximum power during this fast irradiance change.

From t=1.2 sec to t=2.5 sec sun irradiance is restored back to 1000 W/m^2 and then temperature is increased to 50 degrees C. in order to observe impact of temperature increase. Note that when temperature increases from 25 degrees C to 50 degrees C, the array output power decreases from 100.7 kW to 93 kW.

### References

For details on various MPPT techniques, refer to the following paper:

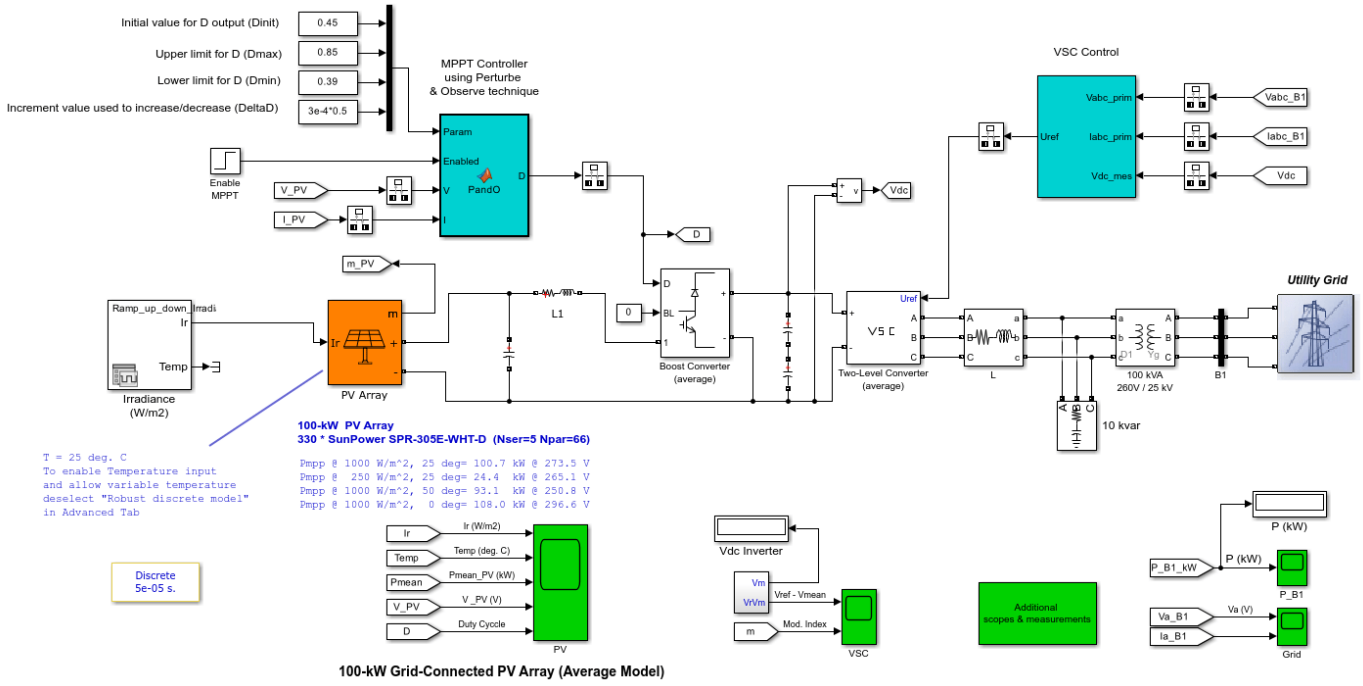
Moacyr A. G. de Brito, Leonardo P. Sampaio, Luigi G. Jr., Guilherme A. e Melo, Carlos A. Canesin "Comparative Analysis of MPPT Techniques for PV Applications", 2011 International Conference on Clean Electrical Power (ICCEP).

The module characteristics were extracted from NREL System Advisor Model (https://sam.nrel.gov/).

# Average Model of a 100-kW Grid-Connected PV Array

This example shows an average model of a 100-kW array connected to a 25-kV grid via a DC-DC boost converter and a three-phase three-level VSC.

Pierre Giroux, Gilbert Sybille (Hydro-Quebec, IREQ) Carlos Osorio, Shripad Chandrachood (The MathWorks)




## Description

A 100-kW PV array is connected to a 25-kV grid via a DC-DC boost converter and a three-phase three-level Voltage Source Converter (VSC). Maximum Power Point Tracking (MPPT) is implemented in the boost converter by means of a Simulink® model using the 'Perturb & Observe' technique.

Another example (see power_PVarray_grid_det model) uses detailed models for the DC_DC and VSC converters. In this detailed model the MPPT controller is based on the 'Incremental Conductance + Integral Regulator' technique.

The average model contains the following components.

- **PV array** delivering a maximum of 100 kW at 1000 W/m^2 sun irradiance.
- **DC-DC boost converter** (orange blocks)
- **3-level 3-phase VSC** (blue blocks).
- **100-kVA 260V/25kV three-phase coupling transformer**.
- **Utility grid**

Please refer to the power_PVarray_grid_det model for a complete description of the PV array, converters and connection to the grid.

The main difference between the detailed model and this average model is in the way that DC-DC boost converter and three-phase VSC are modeled. In this average model the boost and VSC converters are represented by equivalent voltage sources generating the AC voltage averaged over one cycle of the switching frequency. Such a model does not represent harmonics, but the dynamics resulting from control system and power system interaction is preserved. This model allows using much larger time steps than the detailed model (50 microseconds vs 1 microsecond), resulting in a much faster simulation. Note that in the average model the PV-array model contains an algebraic loop. This algebraic loop is required to get an iterative and accurate solution of the PV model when large sample times are used. This algebraic loop is easily solved by Simulink.

The 'Perturb and Observe' MPPT algorithm is implemented in the MPPT Control MATLAB® Function block.

The 100-kW PV array consists of 66 strings of 5 series-connected 305.2-W modules connected in parallel (66*5*305.2 W= 100.7 kW). Manufacturer specifications for one module are:

- Number of series-connected cells : 96
- Open-circuit voltage: Voc= 64.2 V
- Short-circuit current: Isc = 5.96 A
- Voltage and current at maximum power : Vmp =54.7 V, Imp= 5.58 A

The PV array block has two inputs that allow you varying sun irradiance (input 1 in W/m^2) and temperature (input 2 in degrees Celsius). The irradiance and temperature profiles are defined by a Signal Builder block which is connected to the PV array inputs.

**Simulation**

Run the model and observe the following sequence of events on Scopes.

Simulation starts with standard test conditions (25 degrees Celsius, 1000 W/m^2).

From t=0 sec to t= 0.3 sec, duty cycle of boost converter is fixed (D= 0.5 as shown on PV scope). Resulting PV voltage is therefore V= (1-D)*Vdc= (1-0.5)*500=250 V (see V_PV trace on PV scope). The PV array output power is 96 kW (see Pmean trace) whereas specified maximum power with a 1000 W/m^2 irradiance is 100.7 kW. Observe on Grid scope that phase A voltage and current at 25 kV bus are in phase (unity power factor).

At t=0.3 sec MPPT is enabled. The MPPT regulator starts regulating PV voltage by varying duty cycle in order to extract maximum power. Maximum power (100.7 kW) is obtained when duty cycle is D=0.453.

From t=0.3 sec to t=0.5 sec, the PV array operates at standard test conditions (25 degrees C, 1000 W/m^2). Duty cycle D varies between 0.450 and 0.459. PV voltage = 273.5 V (Nser*Vmp=5*54.7= 273.5 V) and mean power = 100.7 kW as expected from PV module specifications .

From t=0.5 sec to t=1.0 sec, sun irradiance is ramped down from 1000 W/m^2 to 250 W/m^2. It can be seen that this type of MPPT controller tracks maximum power only while irradiance stays constant.

From t=1.0 sec to t=1.5 sec when irradiance stays constant and equal to 250 W/m^2, duty cycle D varies between 0.466 and 0.474. Corresponding PV voltage and power are V_PV= 265 V and Pmean=24.4 kW.

From t=1.5 sec to t=6.0 sec sun irradiance is restored back to 1000 W/m^2 and then temperature is varied between 50 and 0 degrees C. in order to observe impact of temperature. Note that maximum PV output power (107.5 kW) is obtained at minimum temperature (0 degrees C).

**References**

For details on various MPPT techniques, refer to the following paper:

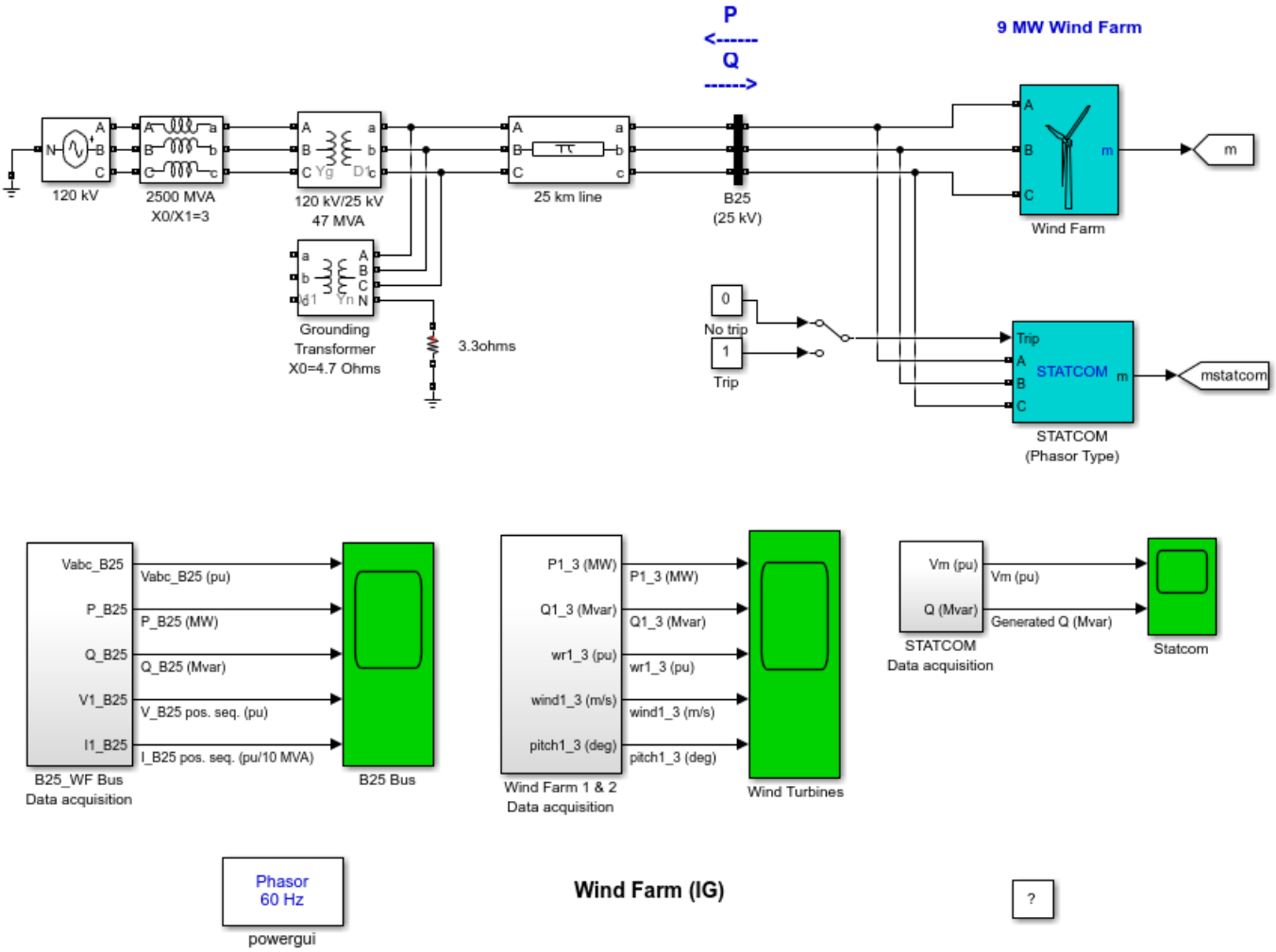
Moacyr A. G. de Brito, Leonardo P. Sampaio, Luigi G. Jr., Guilherme A. e Melo, Carlos A. Canesin "Comparative Analysis of MPPT Techniques for PV Applications", 2011 International Conference on Clean Electrical Power (ICCEP).

The module characteristics were extracted from NREL System Advisor Model (https://sam.nrel.gov/).

# Wind Farm (IG)

This example shows phasor simulation of a 9-MW wind farm using Induction Generators (IG) driven by variable-pitch wind turbines.

Richard Gagnon (Hydro-Quebec)




This example uses an initial state vector to start the simulation from steady-state. When you make changes to the model (add, delete, rename blocks, etc), the initial state vector needs to be regenerated or disabled, otherwise Simulink signals an error when the simulation is started.
To disable the use of initial state vector for this model, go to the Data Import/Export section of the Model Configuration Parameters tool and uncheck the ***Initial state*** option under the 'Load from workspace' section.
To regenerate the initial state vector after a change to the model, follow the instructions given in the initialization file for this model.
Type open('init_power_wind_ig') at the matlab prompt to view this file.

### Description

A wind farm consisting of six 1.5-MW wind turbines is connected to a 25-kV distribution system exports power to a 120-kV grid through a 25-km 25-kV feeder. The 9-MW wind farm is simulated by three pairs of 1.5 MW wind-turbines. Wind turbines use squirrel-cage induction generators (IG). The stator winding is connected directly to the 60 Hz grid and the rotor is driven by a variable-pitch wind

turbine. The pitch angle is controlled in order to limit the generator output power at its nominal value for winds exceeding the nominal speed (9 m/s). In order to generate power the IG speed must be slightly above the synchronous speed. Speed varies approximately between 1 pu at no load and 1.005 pu at full load. Each wind turbine has a protection system monitoring voltage, current and machine speed.

Reactive power absorbed by the IGs is partly compensated by capacitor banks connected at each wind turbine low voltage bus (400 kvar for each pair of 1.5 MW turbine). The rest of reactive power required to maintain the 25-kV voltage at bus B25 close to 1 pu is provided by a 3-Mvar STATCOM with a 3% droop setting.

Open the "Wind Farm" block and look at "Wind Turbine 1". Open the turbine menu and look at the two sets of parameters specified for the turbine and the generator. Each wind turbine block represents two 1.5 MW turbines. Open the turbine menu, select "Turbine data" and check "Display wind-turbine power characteristics". The turbine mechanical power as function of turbine speed is displayed for wind speeds ranging from 4 m/s to 10 m/s. The nominal wind speed yielding the nominal mechanical power (1pu=3 MW) is 9 m/s. The wind turbine model and the statcom model (from the FACTS library) are phasor models that allow transient stability type studies with long simulation times. In this example, the system is observed during 20 s.

The wind speed applied to each turbine is controlled by the "Wind 1" to "Wind 3" blocks . Initially, wind speed is set at 8 m/s, then starting at t=2s for "Wind turbine 1", wind speed is rammed to 11 m/s in 3 seconds. The same gust of wind is applied to Turbine 2 and Turbine 3, respectively with 2 seconds and 4 seconds delays. Then, at t=15 s a temporary fault is applied at the low voltage terminals (575 V) of "Wind Turbine 2".

## Simulation

### Turbine response to a change in wind speed

Start simulation and observe the signals on the "Wind Turbines" scope monitoring active and reactive power, generator speed, wind speed and pitch angle for each turbine. For each pair of turbine the generated active power starts increasing smoothly (together with the wind speed) to reach its rated value of 3 MW in approximately 8s. Over that time frame the turbine speed will have increased from 1.0028 pu to 1.0047 pu. Initially, the pitch angle of the turbine blades is zero degree. When the output power exceed 3 MW, the pitch angle is increased from 0 deg to 8 deg in order to bring output power back to its nominal value. Observe that the absorbed reactive power increases as the generated active power increases. At nominal power, each pair of wind turbine absorbs 1.47 Mvar. For a 11m/s wind speed, the total exported power measured at the B25 bus is 9 MW and the statcom maintains voltage at 0.984 pu by generating 1.62 Mvar (see "B25 Bus" and "Statcom" scopes).

### Operation of protection system

At t=15 s, a phase to phase fault is applied at wind turbine 2 terminals, causing the turbine to trip at t=15.11 s. If you look inside the "Wind Turbine Protections" block you will see that the trip has been initiated by the AC Undervoltage protection. After turbine 2 has tripped, turbines 1 and 3 continue to generate 3 MW each.

### Impact of STATCOM

You will now observe the impact of the "STATCOM". First, open the "Three-Phase Fault" block menu and disable the phase to phase fault. Then put the "STATCOM" out of service by double clicking the "Manual Switch" block connected to the "Trip" input of the "STATCOM". Restart simulation. Observe on " B25 Bus" scope that because of the lack of reactive power support, the voltage at bus "B25" now

drops to 0.91pu. This low voltage condition results in an overload of the IG of "Wind Turbine 1". "Wind Turbine 1" is tripped at t=13.43 s. If you look inside the "Wind Turbine Protections" block you will see that the trip has been initiated by the AC Overcurrent protection.

**Regenerate Initial Conditions**

This example is set-up with all states initialized so that the simulation starts in steady-state. The initial conditions have been saved in the "power_wind_ig.mat" file. When you open this model, the InitFcn callback (in the Model Properties/Callbacks) automatically loads into your workspace the contents of this .mat file ("xInitial" variable).

If you modify this model, or change parameter values of power components, the initial conditions stored in the "xInitial" variable will no longer be valid and Simulink® will issue an error message. To regenerate the initial conditions for your modified model, follow the steps listed below:

1. In the Configuration Parameters pane, uncheck the "Initial state" parameter and check "Final States" parameter.

2. Open the "Wind Farm" subsystem and in the Timer blocks labeled "Wind1" and "Wind2", Wind3" temporarily disable the changes of wind speed by multiplying the "Time(s)" vector by 100.

3. In the "Wind Farm" subsystem, double click the "Three-Phase Fault" block and disable the AB to ground fault (deselect "Phase A Fault" and "Phase B Fault").

4. Start simulation. When Simulation is completed, verify that steady state has been reached by looking at waveforms displayed on the scopes. The final states which have been saved in the "xFinal" array can be used as initial states for future simulations. Executing the next two commands copies these final conditions in "xInitial" and saves this variable in a new file (myModel_init.mat).

```
>> xInitial=xFinal;

>> save myModel_init xInitial
```

5. In the InitFcn window of Model Properties pane, replace the first line of initialization commands with "load myModel_init". Next time you open this model, the variable xInitial saved in the myModel_init.mat file will be loaded in your workspace.

6. In the Configuration Parameters pane, check "Initial state".

7. Start simulation and verify that your model starts in steady-state.

8. Open the "Wind Farm" subsystem and in the Timer blocks labeled "Wind1", "Wind2" and Wind3" re-enable the changes of wind speed respectively a t=2 s , t=4 s and t=6 s (remove the 100 multiplication factors).

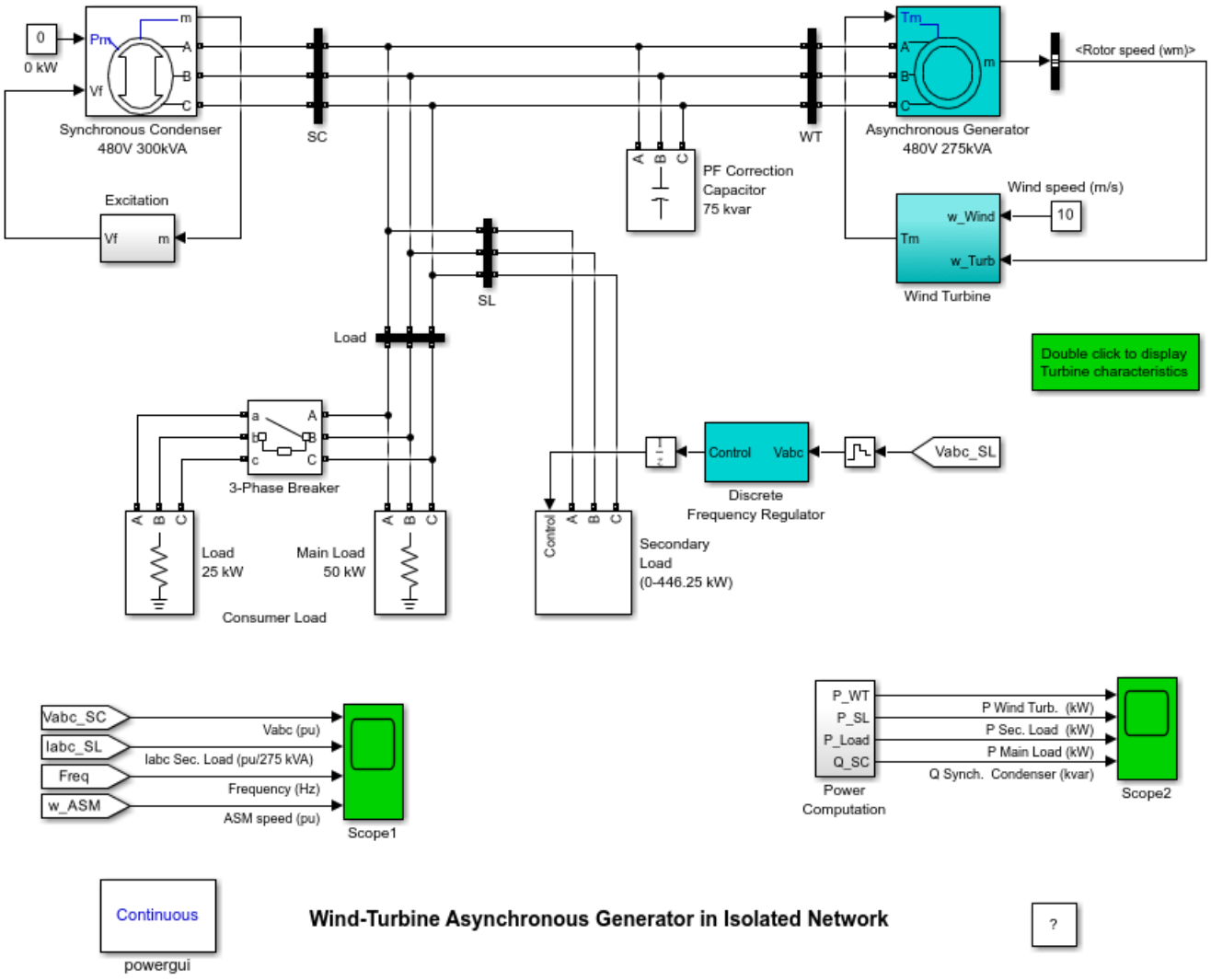
9. In the "Wind Farm" subsystem, re-enable the AB to ground fault in the "Three-Phase Fault" block (check "Phase A Fault" and "Phase B Fault")

10. Save your model.

# Wind-Turbine Asynchronous Generator in Isolated Network

This example shows a wind turbine asynchronous generator in an isolated network.

R. Reid, B. Saulnier, R. Gagnon; Hydro-Quebec (IREQ)




This example uses an initial state vector to start the simulation from steady-state. When you make changes to the model (add, delete, rename blocks, etc), the initial state vector needs to be regenerated or disabled, otherwise Simulink signals an error when the simulation is started.
To disable the use of initial state vector for this model, go to the Data Import/Export section of the Model Configuration Parameters tool and uncheck the ***Initial state*** option under the 'Load from workspace' section.
To regenerate the initial state vector after a change to the model, follow the instructions given in the initialization file for this model.
Type open('init_power_windgen') at the matlab prompt to view this file.

### Description

A generic model of the High-Penetration, No Storage, Wind-Diesel (HPNSWD) system is presented in this example [1]. This technology was developed by Hydro-Quebec to reduce the cost of supplying electricity in remote northern communities [2]. The optimal wind penetration (installed wind

capacity/peak electrical demand) for this system depends on the site delivery cost of fuel and available wind resource. The first commercial application of HPNSWD technology was commissioned in 1999 by Northern Power Systems (Vermont, USA) on St. Paul Island, Alaska [3]. The HPNSWD system presented in this example uses a 480 V, 300 kVA synchronous machine, a wind turbine driving a 480 V, 275 kVA induction generator, a 50 kW customer load and a variable secondary load (0 to 446.25 kW).

At low wind speeds both the induction generator and the diesel-driven synchronous generator are required to feed the load. When the wind power exceeds the load demand, it is possible to shut down the diesel generator. In this all-wind mode, the synchronous machine is used as a synchronous condenser and its excitation system controls the grid voltage at its nominal value. A secondary load bank is used to regulate the system frequency by absorbing the wind power exceeding consumer demand.

The Wind Turbine block uses a 2-D Lookup Table to compute the turbine torque output (Tm) as a function of wind speed (w_Wind) and turbine speed (w_Turb). When you opened this example, the Pm (w_Wind, w_Turb) characteristics was automatically loaded in your workspace (psbwindgen_char array). To display the turbine characteristics, double click on the block located below the Wind Turbine block.

The Secondary Load block consists of eight sets of three-phase resistors connected in series with GTO thyristor switches. The nominal power of each set follows a binary progression so that the load can be varied from 0 to 446.25 kW by steps of 1.75kW. GTOs are simulated by ideal switches.

The frequency is controlled by the Discrete Frequency Regulator block. This controller uses a standard three-phase Phase Locked Loop (PLL) system to measure the system frequency. The measured frequency is compared to the reference frequency (60 Hz) to obtain the frequency error. This error is integrated to obtain the phase error. The phase error is then used by a Proportional-Differential (PD) controller to produce an output signal representing the required secondary load power. This signal is converted to an 8-bit digital signal controlling switching of the eight three-phase secondary loads. In order to minimize voltage disturbances, switching is performed at zero crossing of voltage.

**Simulation**

For the example, the wind speed (10m/s) is such that the wind turbine produces enough power to supply the load. The diesel generator (not simulated) is stopped and the synchronous machine operates as a synchronous condenser with its mechanical power input (Pm) set at zero. The example illustrates the dynamic performance of the frequency regulation system when an additional 25 kW customer load is switched on.

Start simulation and observe voltages, currents, powers, asynchronous machine speed and system frequency on the two scopes. Initial conditions (xInitial vector) have been automatically loaded in your workspace so that simulation starts in steady state.

As the asynchronous machine operates in generator mode, its speed is slightly above the synchronous speed (1.011 pu). According to turbine characteristics, for a 10 m/s wind speed, the turbine output power is 0.75 pu (206 kW). Because of the asynchronous machine losses, the wind turbine produces 200 kW. As the main load is 50 kW, the secondary load absorbs 150 kW to maintain a constant 60 Hz frequency. At t=0.2 s, the additional load of 25 kW is switched on. The frequency momentarily drops to 59.85 Hz and the frequency regulator reacts to reduce the power absorbed by the secondary load in order to bring the frequency back to 60 Hz. Voltage stays at 1 pu and no flicker is observed.

### Regenerate Initial Conditions

This example is set-up with all states initialized so that the simulation starts in steady-state. The initial conditions have been saved in the "power_windgen.mat" file. When you open this model, the InitFcn callback (in the Model Properties/Callbacks) automatically loads into your workspace the contents of this .mat file ("xInitial" variable).

If you modify this model, or change parameter values of power components, the initial conditions stored in the "xInitial" variable will no longer be valid and Simulink® will issue an error message. To regenerate the initial conditions for your modified model, follow the steps listed below:

1. In the Configuration Parameters pane, uncheck the "Initial state" parameter and check "Final States" parameter.

2. Double click the 3-Phase Breaker block and disable breaker switching (deselect "Switching of phase X" parameters for phases A, B and C").

3. Change the Simulation Stop Time to 20 s. Note that in order to generate initial conditions coherent with the 60 Hz frequency, the Stop Time must an integer number of 60 Hz cycles.

4. Start simulation. When Simulation is completed, verify that steady state has been reached by looking at waveforms displayed on the scopes. The final states which have been saved in the "xFinal" array can be used as initial states for future simulations. Executing the next two commands copies these final conditions in "xInitial" and saves this variable in a new file (myModel_init.mat).

```
>> xInitial=xFinal;

>> save myModel_init xInitial
```

5. In the InitFcn window of Model Properties pane, replace the first line of initialization commands with "load myModel_init". Next time you open this model, the variable xInitial saved in the myModel_init.mat file will be loaded in your workspace.

6. In the Configuration Parameters pane, check "Initial state".

7. Start simulation and verify that your model starts in steady-state.

8. Double click the 3-Phase Breaker block and re-enable breaker switching (check "Switching of phase X" parameters for phases A, B and C").

9. Change the Simulation Stop Time back to 5 s.

10. Save your model.

### References

[1] R. Gagnon, B. Saulnier, G. Sybille, P. Giroux; "Modeling of a Generic High-Penetration No-Storage Wind-Diesel System Using MATLAB®/Power System Blockset" 2002 Global Windpower Conference, April 2002, Paris, France

[2] B. Saulnier, A.O. Barry, B. Dube, R. Reid; "Design and Development of a Regulation and Control System for the High-Penetration No-Storage Wind/Diesel Scheme" European Community Wind Energy Conference 88, 6-10 june 1988, Herning, Denmark

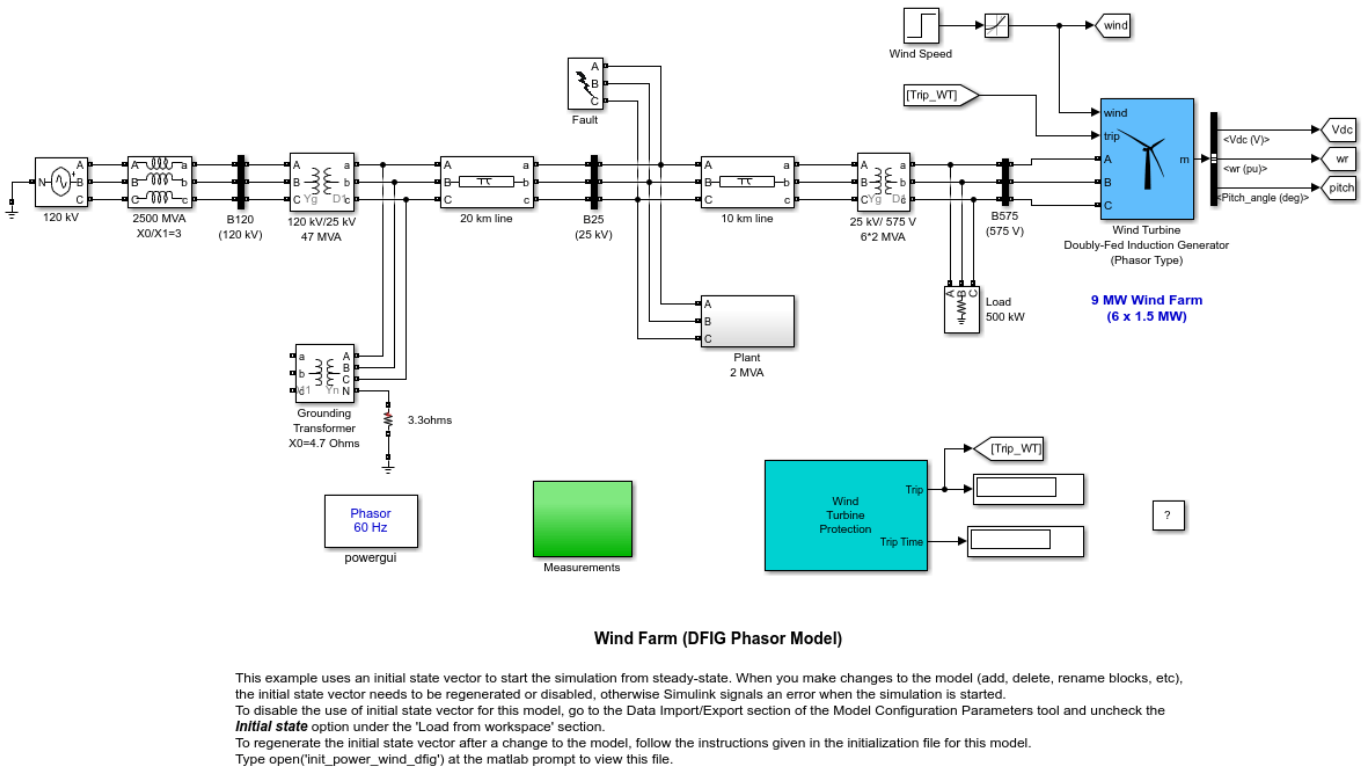
[3] L. Mott (NPS), B. Saulnier (IREQ) " Commercial Wind-Diesel Project, St. Paul Island, Alaska" 14th Prime Power Diesel Inter-Utility Conference, May 28-June 2, Winnipeg, Manitoba, Canada

# Wind Farm (DFIG Phasor Model)

This example shows phasor simulation of a 9 MW wind farm using Doubly-Fed Induction Generator (DFIG) driven by a wind turbine.

Richard Gagnon, Bernard Saulnier, Alain Forcione (Hydro-Quebec)

Note: This example uses a generic model of a DFIG wind turbine. The model is useful for education and academic works.




## Description

A 9-MW wind farm consisting of six 1.5 MW wind turbines connected to a 25-kV distribution system exports power to a 120-kV grid through a 30-km, 25-kV feeder. A 2300V, 2-MVA plant consisting of a motor load (1.68 MW induction motor at 0.93 PF) and of a 200-kW resistive load is connected on the same feeder at bus B25. Both the wind turbine and the motor load have a protection system monitoring voltage, current and machine speed. The DC link voltage of the DFIG is also monitored.

Wind turbines use a doubly-fed induction generator (DFIG) consisting of a wound rotor induction generator and an AC/DC/AC IGBT-based PWM converter. The stator winding is connected directly to the 60 Hz grid while the rotor is fed at variable frequency through the AC/DC/AC converter. The DFIG technology allows extracting maximum energy from the wind for low wind speeds by optimizing the turbine speed, while minimizing mechanical stresses on the turbine during gusts of wind. The optimum turbine speed producing maximum mechanical energy for a given wind speed is proportional to the wind speed. For wind speeds lower than 10 m/s the rotor is running at subsynchronous speed . At high wind speed it is running at hypersynchronous speed. Open the

turbine menu, select "Turbine data" and check "Display wind-turbine power characteristics". The turbine mechanical power as function of turbine speed is displayed for wind speeds ranging from 5 m/s to 16.2 m/s. The DFIG is controlled in order to follow the red curve. Turbine speed optimization is obtained between point B and point C on this curve. Another advantage of the DFIG technology is the ability for power electronic converters to generate or absorb reactive power, thus eliminating the need for installing capacitor banks as in the case of squirrel-cage induction generators.

The wind-turbine model is a phasor model that allows transient stability type studies with long simulation times. In this example, the system is observed during 50 s.

Open the wind turbine block menu and look at the four sets of parameters specified for the turbine, the generator and the converters (grid-side and rotor-side). The 6-wind-turbine farm is simulated by a single wind-turbine block by multiplying the following three parameters by six, as follows:

1. The nominal wind turbine mechanical output: 6*1.5e6 watts, specified in the Turbine data menu

2. The generator rated power: 6*1.5/0.9 MVA (6*1.5 MW at 0.9 PF), specified in the Generator data menu

3. The nominal DC bus capacitor: 6*10000 microfarads, specified in the Converters data menu

Also, notice in the Control parameters menu that the "Mode of operation" is set to " Voltage regulation". The terminal voltage will be controlled to a value imposed by the reference voltage (Vref = 1 pu) and the voltage droop (Xs = 0.02 pu).

**Simulation**

**1. Turbine response to a change in wind speed**

Open the "Wind Speed" step block specifying the wind speed. Initially, wind speed is set at 8 m/s, then at t = 5s, wind speed increases suddenly at 14 m/s. Start simulation and observe the signals on the "Wind Turbine" scope monitoring the wind turbine voltage, current, generated active and reactive powers, DC bus voltage and turbine speed. At t = 5 s, the generated active power starts increasing smoothly (together with the turbine speed) to reach its rated value of 9 MW in approximately 15 s. Over that time frame the turbine speed will have increased from 0.8 pu to 1.21 pu. Initially, the pitch angle of the turbine blades is zero degree and the turbine operating point follows the red curve of the turbine power characteristics up to point D. Then the pitch angle is increased from 0 deg to 0.76 deg in order to limit the mechanical power. Observe also the voltage and the generated reactive power. The reactive power is controlled to maintain a 1 pu voltage. At nominal power, the wind turbine absorbs 0.68 Mvar (generated Q = -0.68 Mvar) to control voltage at 1pu. If you change the mode of operation to "Var regulation" with the "Generated reactive power Qref " set to zero, you will observe that voltage increases to 1.021 pu when the wind turbine generates its nominal power at unity power factor.

**2. Simulation of a voltage sag on the 120-kV system**

You will now observe the impact of a voltage sag resulting from a remote fault on the 120-kV system. First, in the wind speed step block, disable the wind speed step by changing the Final value from 14 to 8 m/s. Then open the 120-kV voltage source menu. In the parameter "Time variation of", select " Amplitude". A 0.15 pu voltage drop lasting 0.5 s is programmed to occur at t = 5 s. Make sure that the control mode is still in Var regulation with Qref = 0. Start simulation and open the "Grid" scope. Observe the plant voltage and current as well as the motor speed. Note that the wind farm produces 1.87 MW. At t = 5 s, the voltage falls below 0.9 pu and at t = 5.22 s, the protection system trips the plant because an undervoltage lasting more than 0.2 s has been detected (look at the protection

settings and status in the "Plant" subsystem). The plant current falls to zero and motor speed decreases gradually, while the wind farm continues generating at a power level of 1.87 MW. After the plant has tripped, 1.25 MW of power (P_B25 measured at bus B25) is exported to the grid.

Now, change the wind turbine control mode to "Voltage regulation" and repeat the test. You will notice that the plant does not trip anymore. This is because the voltage support provided by the 5 Mvar reactive power generated by the wind-turbines during the voltage sag keeps the plant voltage above the 0.9 pu protection threshold. The plant voltage during the voltage sag is now 0.93 pu.

**3. Simulation of a fault on the 25-kV system**

Finally, you will now observe impact of a single phase-to-ground fault occurring on the 25-kV line at B25 bus. First disable the 120-kV voltage step. Now open the "Fault" block menu and select "Phase A Fault". Check that the fault is programmed to apply a 9-cycle single-phase to ground fault at t = 5 s.

You should observe that when the wind turbine is in "Voltage regulation" mode, the positive-sequence voltage at wind-turbine terminals (V1_B575) drops to 0.8 pu during the fault, which is above the undervoltage protection threshold (0.75 pu for a t > 0.1 s). The wind farm therefore stays in service. However, if the "Var regulation" mode is used with Qref = 0, the voltage drops under 0.7 pu and the undervoltage protection trips the wind farm. We can now observe that the turbine speed increases. At t= 40 s the pitch angle starts to increase in order to limit the speed.

**Regenerate Initial Conditions**

This example is set-up with all states initialized so that the simulation starts in steady-state. The initial conditions have been saved in the "power_wind_dfig.mat" file. When you open this model, the InitFcn callback (in the Model Properties/Callbacks) automatically loads into your workspace the contents of this .mat file ("xInitial" variable).

If you modify this model, or change parameter values of power components, the initial conditions stored in the "xInitial" variable will no longer be valid and Simulink® will issue an error message. To regenerate the initial conditions for your modified model, follow the steps listed below:

1. In the Configuration Parameters pane, uncheck the "Initial state" parameter and check "Final States" parameter.

2. Double click the Step block labeled "Wind Speed (m/s)" and temporarily disable the change of wind speed by multiplying the Step time by 100.

3. Double click the Breaker block and make sure that no fault is applied (Phase A, B and C checkboxes not selected).

4. Double click the 120 kV voltage source block and make sure that the "Time variation of" parameter is set to "None".

5. Start simulation. When Simulation is completed, verify that steady state has been reached by looking at waveforms displayed on the scopes. The final states which have been saved in the "xFinal" array can be used as initial states for future simulations. Executing the next two commands copies these final conditions in "xInitial" and saves this variable in a new file (myModel_init.mat).

```
>> xInitial=xFinal;

>> save myModel_init xInitial
```

6. In the InitFcn window of Model Properties pane, replace the line "xInitial = init_power_wind_dfig;" with "load myModel_init.mat". Next time you open this model, the variable xInitial saved in the myModel_init.mat file will be loaded in your workspace.

7. In the Configuration Parameters pane, check "Initial state".

8. Start simulation and verify that your model starts in steady-state.

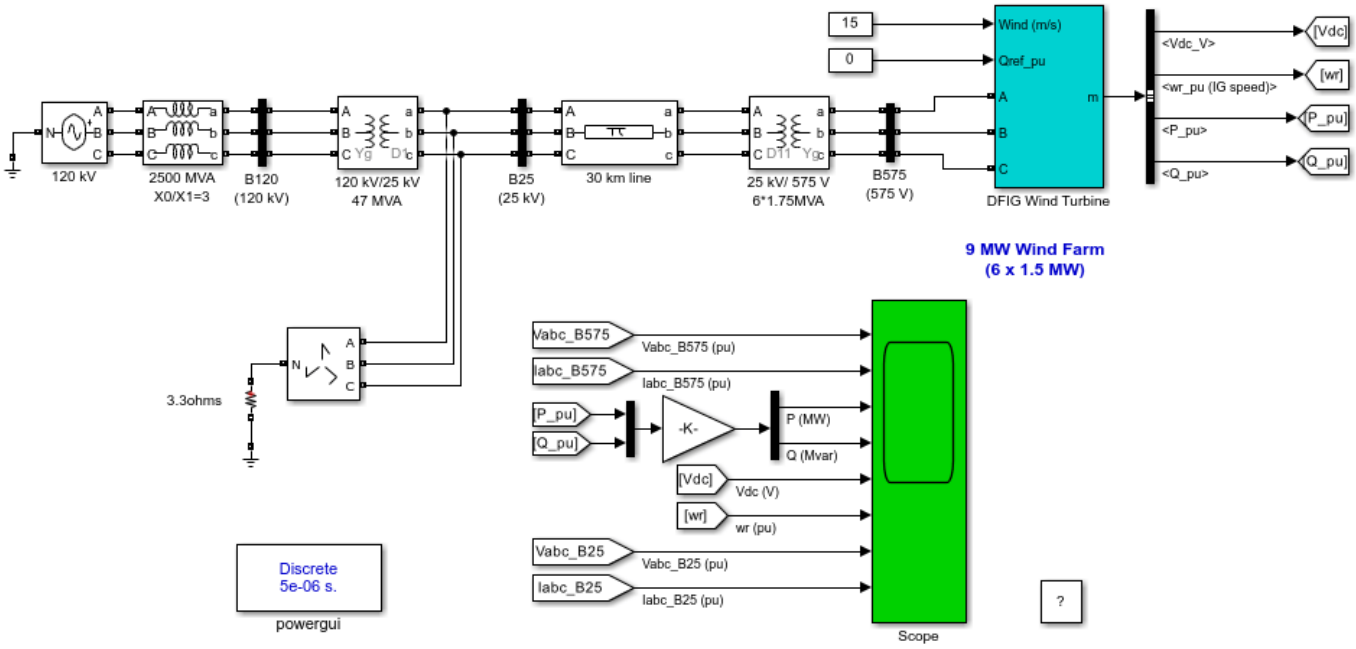
9. Double click the Step block labeled "Wind Speed (m/s)" and re-enable the change of wind speed at t=5 s (remove the 100 multiplication factor).

10. Save your model.

# Wind Farm - DFIG Detailed Model

This example shows a 9 MW wind farm using a detailed model of a Doubly-Fed Induction Generator (DFIG) driven by a wind turbine.

Richard Gagnon (Hydro-Quebec)




**Wind Farm - DFIG Detailed Model**

This example uses an initial state vector to start the simulation from steady-state. When you make changes to the model (add, delete, rename blocks, etc), the initial state vector needs to be regenerated or disabled, otherwise Simulink signals an error when the simulation is started.
To disable the use of initial state vector for this model, go to the Data Import/Export section of the Model Configuration Parameters tool and uncheck the ***Initial state*** option under the 'Load from workspace' section.
To regenerate the initial state vector after a change to the model, follow the instructions given in the initialization file for this model.
Type open('init_power_wind_dfig_det') at the MATLAB prompt to view this file.

### 1. Simulation Methods of the DFIG

Depending on the range of frequencies to be represented, three simulation methods are currently available in Specialized Power Systems to model VSC based energy conversion systems connected on power grids.

**The detailed model (discrete)** such as the one presented in this example. The detailed model includes detailed representation of power electronic IGBT converters. In order to achieve an acceptable accuracy with the 1620 Hz and 2700 Hz switching frequencies used in this example, the model must be discretized at a relatively small time step (5 microseconds). This model is well suited for observing harmonics and control system dynamic performance over relatively short periods of times (typically hundreds of milliseconds to one second).

**The average model (discrete)** such as the one presented in the power_wind_dfig_avg model in the Renewable Energy examples library. In this type of model the IGBT Voltage-sourced converters (VSC) are represented by equivalent voltage sources generating the AC voltage averaged over one cycle of

the switching frequency. This model does not represent harmonics, but the dynamics resulting from control system and power system interaction is preserved. This model allows using much larger time steps (typically 50 microseconds), thus allowing simulations of several seconds.

**The phasor model (continuous)** such as the one presented in the "power_wind_dfig" model in the Renewable Energy examples library. This model is better adapted to simulate the low frequency electromechanical oscillations over long periods of time (tens of seconds to minutes). In the phasor simulation method, the sinusoidal voltages and currents are replaced by phasor quantities (complex numbers) at the system nominal frequency (50 Hz or 60 Hz).This is the same technique which is used in transient stability software.

### 2. Description

A 9 MW wind farm consisting of six 1.5 MW wind turbines connected to a 25 kV distribution system exports power to a 120 kV grid through a 30 km, 25 kV feeder.

Wind turbines using a doubly-fed induction generator (DFIG) consist of a wound rotor induction generator and an AC/DC/AC IGBT-based PWM converter. The stator winding is connected directly to the 60 Hz grid while the rotor is fed at variable frequency through the AC/DC/AC converter. The DFIG technology allows extracting maximum energy from the wind for low wind speeds by optimizing the turbine speed, while minimizing mechanical stresses on the turbine during gusts of wind.

In this example the wind speed is maintained constant at 15 m/s. The control system uses a torque controller in order to maintain the speed at 1.2 pu. The reactive power produced by the wind turbine is regulated at 0 Mvar.

Look under the mask of "DFIG Wind Turbine" block to see how the model is built. The sample time used to discretize the model (Ts= 50 microseconds) is specified in the Initialization function of the Model Properties.

Open the "DFIG Wind Turbine" block menu to see the data of the generator, the converter, the turbine, the drive train and the control systems. In the Display menu select "Turbine data for 1 wind turbine", check "Display wind turbine power characteristics" and then click Apply. The turbine Cp curves are displayed in Figure 1. The turbine power, the tip speed ratio lambda and the Cp values are displayed in Figure 2 as function of wind speed. For a wind speed of 15 m/s, the turbine output power is 1 pu of its rated power, the pitch angle is 8.7 deg and the generator speed is 1.2 pu.

### 3. Simulation

In this example you will observe the steady-state operation of the DFIG and its dynamic response to voltage sag resulting from a remote fault on the 120-kV system. Open the "120 kV" block modeling the voltage source and see how a six-cycle 0.5 pu voltage drop is programmed at t=0.03 s

Start simulation. Observe voltage and current waveforms on the Scope block. At simulation start the "xInitial" variable containing the initial state variables is automatically loaded (from the "power_wind_dfig_det.mat" file specified in the Model Properties) so that the simulation starts in steady state.

Initially the DFIG wind farm produces 9 MW. The corresponding turbine speed is 1.2 pu of generator synchronous speed. The DC voltage is regulated at 1150 V and reactive power is kept at 0 Mvar. At t=0.03 s the positive-sequence voltage suddenly drops to 0.5 p.u. causing an oscillation on the DC bus voltage and on the DFIG output power. During the voltage sag the control system tries to regulate DC voltage and reactive power at their set points (1150 V, 0 Mvar). The system recovers in approximately 4 cycles.

### 4. Regenerate Initial Conditions

This example is set-up with all states initialized so that the simulation starts in steady-state. Otherwise, due to the long time constants of the electromechanical part of the wind turbine model and to its relatively slow regulators you would have to wait for tens of seconds before reaching steady-state. The initial conditions have been saved in the "power_wind_dfig_det.mat" file. When you start simulation, the InitFcn callback (in the Model Properties/Callbacks) automatically loads into your workspace the contents of this .mat file ("xInitial" variable specified in the "Initial state" parameter in the Configuration Parameters pane).

If you modify this model, or change parameter values of power components, the initial conditions stored in the "xInitial" variable will no longer be valid and Simulink® will issue an error message. To regenerate the initial conditions for your modified model, follow the steps listed below:

1. In the Configuration Parameters pane, uncheck the "Initial state" parameter and check "Final States" parameter.

2. In the 120 kV Three-phase Voltage Source menu, disable the source voltage step by setting the "Time variation of " parameter to "none".

3. In order to shorten the time required to reach steady-state, temporarily decrease the inertia of the turbine-generator group. Open the DFIG Wind Turbine menu and in the Drive train data and Generator data, divide the H inertia constants by 10.

4. Change the Simulation Stop Time to 5 seconds. Note that in order to generate initial conditions coherent with the 60 Hz voltage source phase angles, the Stop Time must be an integer number of 60 Hz cycles.

5. Change the Simulation Mode from "Normal" to "Accelerator".

6. Start simulation. When Simulation is completed, verify that steady state has been reached by looking at waveforms displayed on the Scope block. The final states which have been saved in the "xFinal" structure with time can be used as initial states for future simulations. Executing the next two commands copies these final conditions in "xInitial" and saves this variable in a new file (myModel_init.mat).

```
>> xInitial=xFinal;
```

```
>> save myModel_init xInitial
```

7. In the InitFcn window of Model Properties pane, replace the first line of initialization commands with "load myModel_init". Next time you start a simulation with this model, the variable xInitial saved in the myModel_init.mat file will be loaded in your workspace.

8. In the Configuration Parameters pane, check "Initial state".

9. In the Wind Turbine Generator and Drive train data, reset the inertia constants H back to their original values.

10. Start simulation and verify that your model starts in steady-state.

11. In the 120 kV Three-phase voltage source menu, set the "Time variation of" parameter back to "Amplitude".

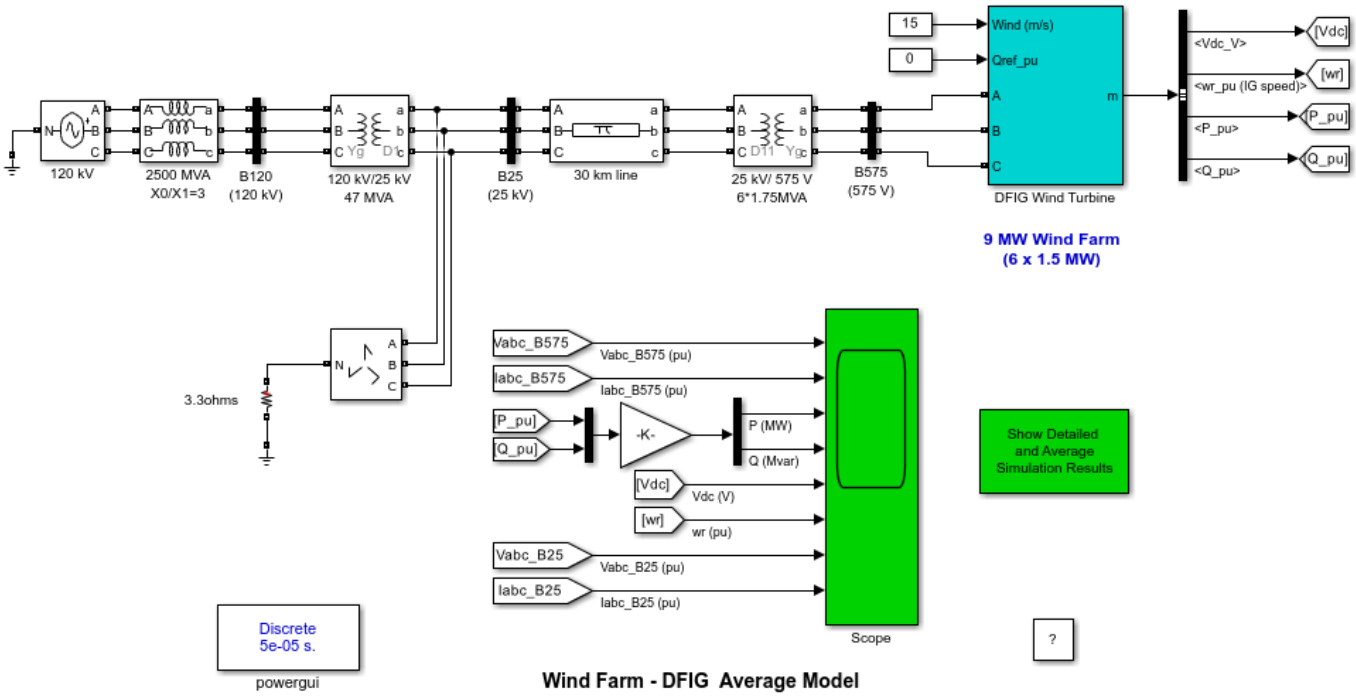
12. Change the Simulation Stop Time and Simulation Mode back to their original values (0.2 seconds, Normal).

13. Save your model.

# Wind Farm - DFIG Average Model

This example shows a 9 MW wind farm using an average model of a Doubly-Fed Induction Generator (DFIG) driven by a wind turbine.

Richard Gagnon (Hydro-Quebec)




This example uses an initial state vector to start the simulation from steady-state. When you make changes to the model (add, delete, rename blocks, etc), the initial state vector needs to be regenerated or disabled, otherwise Simulink signals an error when the simulation is started.
To disable the use of initial state vector for this model, go to the Data Import/Export section of the Model Configuration Parameters tool and uncheck the ***Initial state*** option under the 'Load from workspace' section.
To regenerate the initial state vector after a change to the model, follow the instructions given in the initialization file for this model.
Type open('init_power_wind_dfig_avg') at the MATLAB prompt to view this file.

### 1. Simulation Methods of the DFIG

Depending on the range of frequencies to be represented, three simulation methods are currently available in Specialized Power Systems to model VSC based energy conversion systems connected on power grids.

**The detailed model (discrete)** such as the one presented in the power_wind_dfig_det model in the Renewable Energy examples library. The detailed model includes detailed representation of power electronic IGBT converters. In order to achieve an acceptable accuracy with the 1620 Hz and 2700 Hz switching frequencies used in this example, the model must be discretized at a relatively small time step (5 microseconds). This model is well suited for observing harmonics and control system dynamic performance over relatively short periods of times (typically hundreds of milliseconds to one second).

**The average model (discrete)** such as the one presented in this example. In this type of model the IGBT Voltage-sourced converters (VSC) are represented by equivalent voltage sources generating the

AC voltage averaged over one cycle of the switching frequency. This model does not represent harmonics, but the dynamics resulting from control system and power system interaction is preserved. This model allows using much larger time steps (typically 50 microseconds), thus allowing simulations of several seconds.

**The phasor model (continuous)** such as the one presented in the "power_wind_dfig" model in the Renewable Energy examples library. This model is better adapted to simulate the low frequency electromechanical oscillations over long periods of time (tens of seconds to minutes). In the phasor simulation method, the sinusoidal voltages and currents are replaced by phasor quantities (complex numbers) at the system nominal frequency (50 Hz or 60 Hz).This is the same technique which is used in transient stability software.

### 2. Description

A 9 MW wind farm consisting of six 1.5 MW wind turbines connected to a 25 kV distribution system exports power to a 120 kV grid through a 30 km, 25 kV feeder.

Wind turbines using a doubly-fed induction generator (DFIG) consist of a wound rotor induction generator and an AC/DC/AC IGBT-based PWM converter modeled by voltage sources. The stator winding is connected directly to the 60 Hz grid while the rotor is fed at variable frequency through the AC/DC/AC converter. The DFIG technology allows extracting maximum energy from the wind for low wind speeds by optimizing the turbine speed, while minimizing mechanical stresses on the turbine during gusts of wind.

In this example the wind speed is maintained constant at 15 m/s. The control system uses a torque controller in order to maintain the speed at 1.2 pu. The reactive power produced by the wind turbine is regulated at 0 Mvar.

Look under the mask of "DFIG Wind Turbine" block to see how the model is built. The sample time used to discretize the model (Ts= 50 microseconds) is specified in the Initialization function of the Model Properties.

Open the "DFIG Wind Turbine" block menu to see the data of the generator, the converter, the turbine, the drive train and the control systems. In the Display menu select "Turbine data for 1 wind turbine" check "Display wind turbine power characteristics" and then click Apply. The turbine Cp curves are displayed in Figure 1. The turbine power, the tip speed ratio lambda and the Cp values are displayed in Figure 2 as function of wind speed. For a wind speed of 15 m/s, the turbine output power is 1 pu of its rated power, the pitch angle is 8.7 deg and the generator speed is 1.2 pu.

### 3. Simulation

In this example you will observe the steady-state operation of the DFIG and its dynamic response to voltage sag resulting from a remote fault on the 120-kV system. Open the "120 kV" block modeling the voltage source and see how a six-cycle 0.5 pu voltage drop is programmed at t=0.03 s

Start simulation. Observe voltage and current waveforms on the Scope block. At simulation start the "xInitial" variable containing the initial state variables is automatically loaded (from the "power_wind_dfig_avg.mat" file specified in the Model Properties) so that the simulation starts in steady state.

Initially the DFIG wind farm produces 9 MW. The corresponding turbine speed is 1.2 pu of generator synchronous speed. The DC voltage is regulated at 1150 V and reactive power is kept at 0 Mvar. At t=0.03 s the positive-sequence voltage suddenly drops to 0.5 p.u. causing an oscillation on the DC bus voltage and on the DFIG output power. During the voltage sag the control system tries to

regulate DC voltage and reactive power at their set points (1150 V, 0 Mvar). The system recovers in approximately 4 cycles.

Double click the blue block entitled "Show Detailed and Average Simulation Results". A figure opens showing comparison of the phase A voltage at DFIG terminals, DC link voltage, active and reactive powers and speed for the detailed model and the average model. Notice that the two models are in good agreement. The average model represents correctly the low frequency control and power system oscillations produced by the voltage sag, but voltage waveforms do not show the high frequency harmonics produced by the PWM switching of the two converters.

**4. Regenerate Initial Conditions**

This example is set-up with all states initialized so that the simulation starts in steady-state. Otherwise, due to the long time constants of the electromechanical part of the wind turbine model and to its relatively slow regulators you would have to wait for tens of seconds before reaching steady-state. The initial conditions have been saved in the "power_wind_dfig_avg.mat" file. When you start simulation, the InitFcn callback (in the Model Properties/Callbacks) automatically loads into your workspace the contents of this .mat file ("xInitial" variable specified in the "Initial state" parameter in the Configuration Parameters pane).

If you modify this model, or change parameter values of power components, the initial conditions stored in the "xInitial" variable will no longer be valid and Simulink® will issue an error message. To regenerate the initial conditions for your modified model, follow the steps listed below:

1. In the Configuration Parameters pane, uncheck the "Initial state" parameter and check "Final States" parameter.

2. In the 120 kV Three-phase Voltage Source menu, disable the source voltage step by setting the "Time variation of " parameter to "none".

3. In order to shorten the time required to reach steady-state, temporarily decrease the inertia of the turbine-generator group. Open the DFIG Wind Turbine menu and in the Drive train data and Generator data, divide the H inertia constants by 10.

4. Change the Simulation Stop Time to 5 seconds. Note that in order to generate initial conditions coherent with the 60 Hz voltage source phase angles, the Stop Time must be an integer number of 60 Hz cycles.

5. Change the Simulation Mode from "Normal" to "Accelerator".

6. Start simulation. When Simulation is completed, verify that steady state has been reached by looking at waveforms displayed on the Scope block. The final states which have been saved in the "xFinal" structure with time can be used as initial states for future simulations. Executing the next two commands copies these final conditions in "xInitial" and saves this variable in a new file (myModel_init.mat).

```
>> xInitial=xFinal;

>> save myModel_init xInitial
```

7. In the InitFcn window of Model Properties pane, replace the first line of initialization commands with "load myModel_init". Next time you start a simulation with this model, the variable xInitial saved in the myModel_init.mat file will be loaded in your workspace.

8. In the Configuration Parameters pane, check "Initial state".

9. In the Wind Turbine Generator and Drive train data, reset the inertia constants H back to their original values.

10. Start simulation and verify that your model starts in steady-state.

11. In the 120 kV Three-phase voltage source menu, set the "Time variation of" parameter back to "Amplitude".

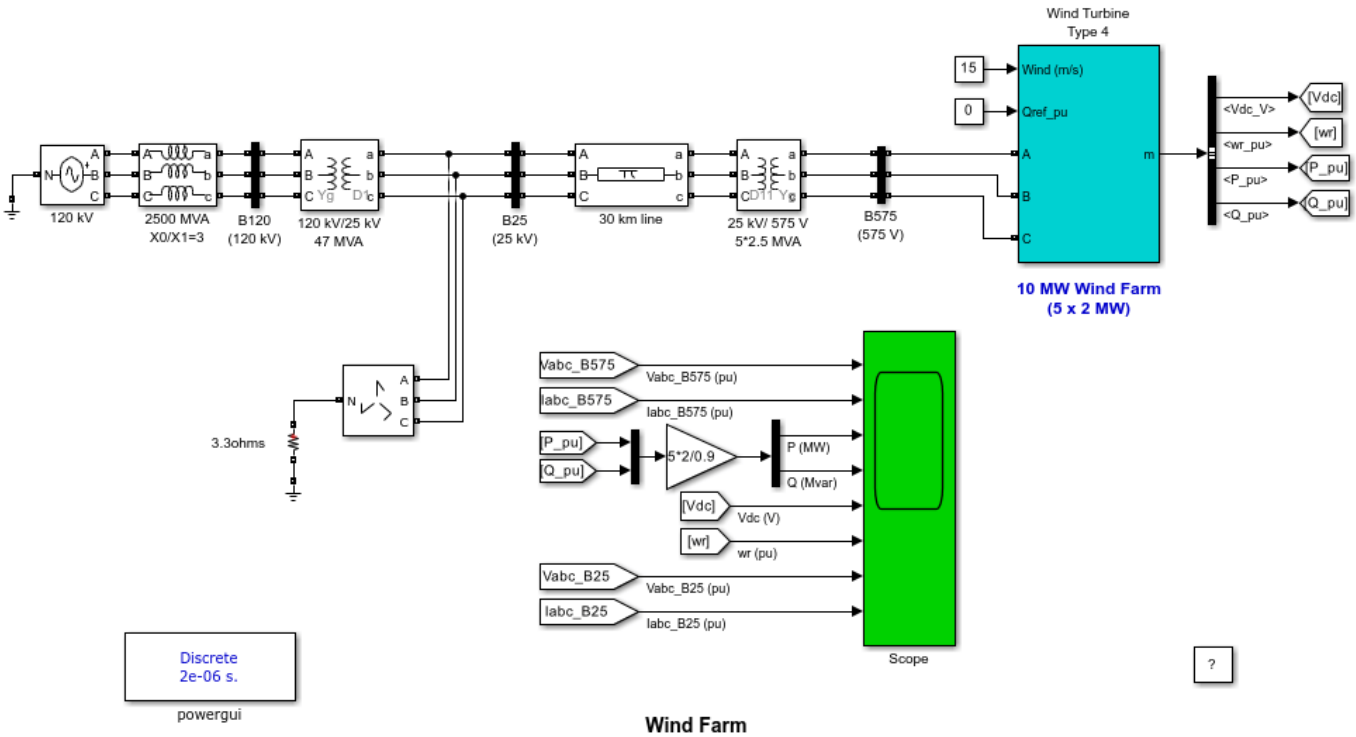
12. Change the Simulation Stop Time and Simulation Mode back to their original values (0.2 seconds, Normal).

13. Save your model.

# Wind Farm - Synchronous Generator and Full Scale Converter (Type 4) Detailed Model

This example shows a 10 MW wind farm using a detailed model of a Type 4 wind turbine.

Richard Gagnon and Jacques Brochu (Hydro-Quebec)




**Wind Farm**
**Synchronous Generator and Full Scale Converter (Type 4) Detailed Model**

This example uses an initial state vector to start the simulation from steady-state. When you make changes to the model (add, delete, rename blocks, etc), the initial state vector needs to be regenerated or disabled, otherwise Simulink signals an error when the simulation is started.
To disable the use of initial state vector for this model, go to the Data Import/Export section of the Model Configuration Parameters tool and uncheck the ***Initial state*** option under the 'Load from workspace' section.
To regenerate the initial state vector after a change to the model, follow the instructions given in the initialization file for this model.
Type open('init_power_wind_type_4_det') at the MATLAB prompt to view this file.

### 1. Simulation Methods of the Type 4 Wind Turbine

Depending on the range of frequencies to be represented, three simulation methods are currently available in Specialized Power Systems to model VSC based energy conversion systems connected on power grids.

**The detailed model (discrete)** such as the one presented in this example. The detailed model includes detailed representation of power electronic IGBT converters. In order to achieve an acceptable accuracy with the 2000 Hz and 3000 Hz switching frequencies used in this example, the model must be discretized at a relatively small time step (2 microseconds). This model is well suited for observing harmonics and control system dynamic performance over relatively short periods of times (typically hundreds of milliseconds to one second).

**The average model (discrete)** such as the one presented in the power_wind_type_4_avg model in the Renewable Energy examples library. In this type of model the IGBT Voltage-sourced converters (VSC) are represented by equivalent voltage sources generating the AC voltage averaged over one cycle of the switching frequency. A similar method is used for DC-DC converter. The average model does not represent harmonics, but the dynamics resulting from control system and power system interaction is preserved. This model allows using much larger time steps (typically 50 microseconds), thus allowing simulations of several seconds.

**The phasor model (continuous)** such as the one presented in the "power_wind_dfig" model in the Renewable Energy example library. This model is better adapted to simulate the low frequency electromechanical oscillations over long periods of time (tens of seconds to minutes). In the phasor simulation method, the sinusoidal voltages and currents are replaced by phasor quantities (complex numbers) at the system nominal frequency (50 Hz or 60 Hz).This is the same technique which is used in transient stability software.

### 2. Description

A 10 MW wind farm consisting of five 2 MW wind turbines connected to a 25 kV distribution system exports power to a 120 kV grid through a 30 km, 25 kV feeder.

The Type 4 wind turbine presented in this example consists of a synchronous generator connected to a diode rectifier, a DC-DC IGBT-based PWM boost converter and a DC/AC IGBT-based PWM converter. The Type 4 technology allows extracting maximum energy from the wind for low wind speeds by optimizing the turbine speed, while minimizing mechanical stresses on the turbine during gusts of wind.

In this example the wind speed is maintained constant at 15 m/s. The control system of the DC-DC converter is used to maintain the speed at 1 pu. The reactive power produced by the wind turbine is regulated at 0 Mvar.

Right-click on the "Wind Turbine Type 4" block and select "Look Under Mask" to see how the model is built. The sample time used to discretize the model (Ts= 2 microseconds) is specified in the Initialization function of the Model Properties.

Open the "Wind Turbine Type 4" block menu to see the data of the generator, the converter, the turbine, the drive train and the control systems. In the Display menu select "Turbine data for 1 wind turbine", check "Display wind turbine power characteristics" and then click Apply. The turbine Cp curves are displayed in Figure 1. The turbine power, the tip speed ratio lambda and the Cp values are displayed in Figure 2 as function of wind speed. For a wind speed of 15 m/s, the turbine output power is 1 pu of its rated power, the pitch angle is 8.8 deg and the generator speed is 1 pu.

### 3. Simulation

In this example you will observe the steady-state operation of the Type 4 wind turbine and its dynamic response to voltage sag resulting from a remote fault on the 120-kV system. Open the "120 kV" block modeling the voltage source and see how a six-cycle 0.25 pu voltage drop is programmed at t=0.03 s

Start simulation. Observe voltage and current waveforms on the Scope block. At simulation start the "xInitial" variable containing the initial state variables is automatically loaded (from the "power_wind_type_4_det.mat" file specified in the Model Properties) so that the simulation starts in steady state.

Initially the Type 4 wind farm produces 10 MW. The corresponding turbine speed is 1 pu of generator synchronous speed. The DC voltage is regulated at 1100 V and reactive power is kept at 0 Mvar. At

t=0.03 s the positive-sequence voltage suddenly drops to 0.75 p.u. causing an increase on the DC bus voltage and a drop on the Type 4 wind turbine output power. During the voltage sag the control systems try to regulate DC voltage and reactive power at their set points (1100 V, 0 Mvar). The system recovers after fault elimination.

**4. Regenerate Initial Conditions**

This example is set-up with all states initialized so that the simulation starts in steady-state. Otherwise, due to the long time constants of the electromechanical part of the wind turbine model and to its relatively slow regulators you would have to wait for tens of seconds before reaching steady-state. The initial conditions have been saved in the "power_wind_type_4_det.mat" file. When you start simulation, the InitFcn callback (in the Model Properties/Callbacks) automatically loads into your workspace the contents of this .mat file ("xInitial" variable specified in the "Initial state" parameter in the Simulation/Configuration Parameters menu).

If you modify this model, or change parameter values of power components, the initial conditions stored in the "xInitial" variable will no longer be valid and Simulink® will issue an error message. To regenerate the initial conditions for your modified model, follow the steps listed below:

1. In the Configuration Parameters pane, uncheck the "Initial state" parameter and check "Final States" parameter.

2. In the 120 kV Three-phase Voltage Source menu, disable the source voltage step by setting the "Time variation of " parameter to "none".

3. In order to shorten the time required to reach steady-state, temporarily decrease the inertia of the turbine-generator group. Open the Wind Turbine Type 4 menu and in the Drive train data and Generator data, divide the H inertia constants by 10.

4. Change the Simulation Stop Time to 5 seconds. Note that in order to generate initial conditions coherent with the 60 Hz voltage source phase angles, the Stop Time must be an integer number of 60 Hz cycles.

5. Change the Simulation Mode from "Normal" to "Accelerator".

6. Start simulation. When Simulation is completed, verify that steady state has been reached by looking at waveforms displayed on the Scope block. The final states which have been saved in the "xFinal" structure with time can be used as initial states for future simulations. Executing the next two commands copies these final conditions in "xInitial" and saves this variable in a new file (myModel_init.mat).

```
>> xInitial=xFinal;

>> save myModel_init xInitial
```

7. In the InitFcn window of Model Properties pane, replace the first line of initialization commands with "load myModel_init". Next time you start a simulation with this model, the variable xInitial saved in the myModel_init.mat file will be loaded in your workspace.

8. In the Configuration Parameters pane, check "Initial state".

9. In the Wind Turbine Generator and Drive train data, reset the inertia constants H back to their original values.

10. Start simulation and verify that your model starts in steady-state.

11. In the 120 kV Three-phase voltage source menu, set the "Time variation of" parameter back to "Amplitude".

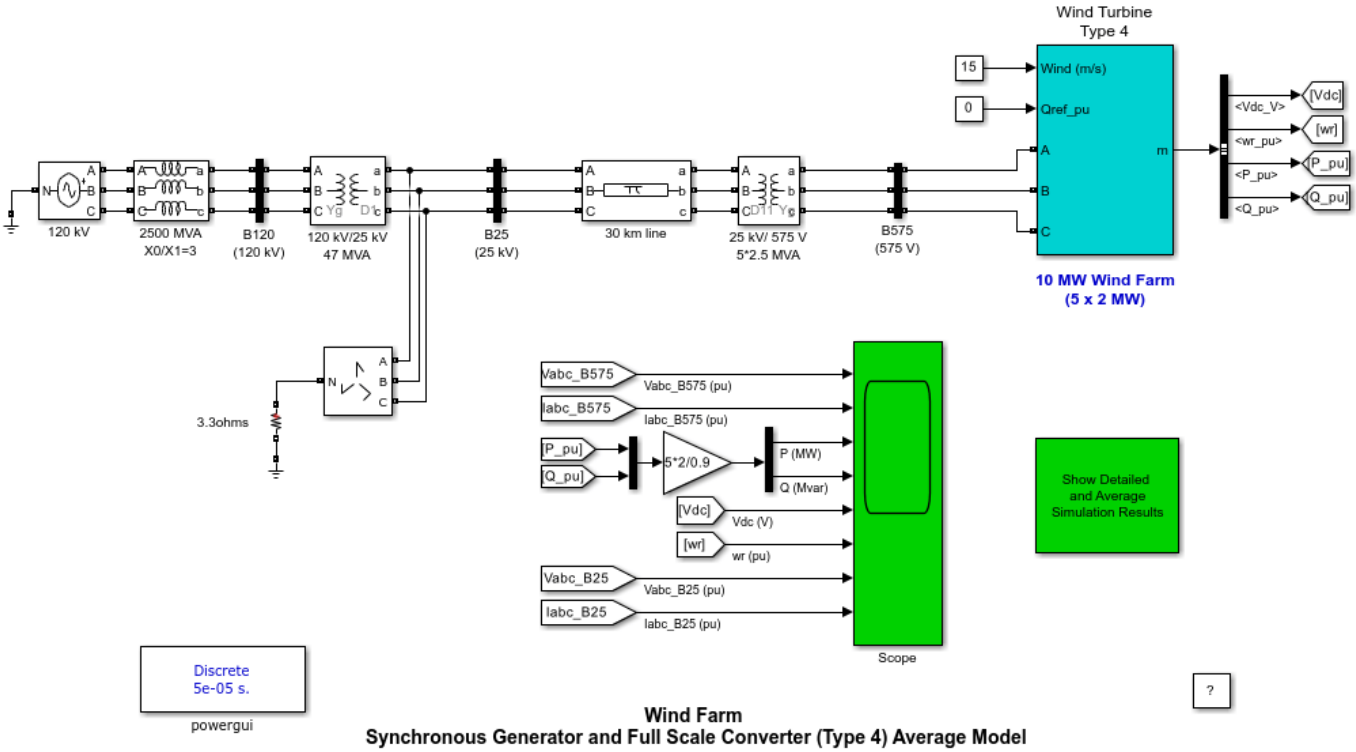
12. Change the Simulation Stop Time and Simulation Mode back to their original values (0.2 seconds, Normal).

13. Save your model.

# Wind Farm - Synchronous Generator and Full Scale Converter (Type 4) Average Model

This example shows a 10 MW wind farm using an average model of a Type 4 wind turbine.

Richard Gagnon and Jacques Brochu (Hydro-Quebec)




This example uses an initial state vector to start the simulation from steady-state. When you make changes to the model (add, delete, rename blocks, etc), the initial state vector needs to be regenerated or disabled, otherwise Simulink signals an error when the simulation is started.
To disable the use of initial state vector for this model, go to the Data Import/Export section of the Model Configuration Parameters tool and uncheck the ***Initial state*** option under the 'Load from workspace' section.
To regenerate the initial state vector after a change to the model, follow the instructions given in the initialization file for this model.
Type open('init_power_wind_type_4_avg') at the MATLAB prompt to view this file.

### 1. Simulation Methods of the Type 4 Wind Turbine

Depending on the range of frequencies to be represented, three simulation methods are currently available in Specialized Power Systems to model VSC based energy conversion systems connected on power grids.

**The detailed model (discrete)** such as the one presented in the power_wind_type_4_det model. The detailed model includes detailed representation of power electronic IGBT converters. In order to achieve an acceptable accuracy with the 2000 Hz and 3000 Hz switching frequencies used in this example, the model must be discretized at a relatively small time step (2 microseconds). This model is well suited for observing harmonics and control system dynamic performance over relatively short periods of times (typically hundreds of milliseconds to one second).

**The average model (discrete)** such as the one presented in this example. In this type of model the IGBT Voltage-sourced converters (VSC) are represented by equivalent voltage sources generating the

AC voltage averaged over one cycle of the switching frequency. A similar method is used for DC-DC converter. The average model does not represent harmonics, but the dynamics resulting from control system and power system interaction is preserved. This model allows using much larger time steps (typically 50 microseconds), thus allowing simulations of several seconds.

**The phasor model (continuous)** such as the one presented in the "power_wind_dfig" model in the Renewable Energy examples library. This model is better adapted to simulate the low frequency electromechanical oscillations over long periods of time (tens of seconds to minutes). In the phasor simulation method, the sinusoidal voltages and currents are replaced by phasor quantities (complex numbers) at the system nominal frequency (50 Hz or 60 Hz).This is the same technique which is used in transient stability software.

### 2. Description

A 10 MW wind farm consisting of five 2 MW wind turbines connected to a 25 kV distribution system exports power to a 120 kV grid through a 30 km, 25 kV feeder.

The Type 4 wind turbine presented in this example consists of a synchronous generator connected to a diode rectifier, a DC-DC IGBT-based PWM boost converter and a DC/AC IGBT-based PWM converter modeled by voltage sources. The Type 4 technology allows extracting maximum energy from the wind for low wind speeds by optimizing the turbine speed, while minimizing mechanical stresses on the turbine during gusts of wind.

In this example the wind speed is maintained constant at 15 m/s. The control system of the DC-DC converter is used to maintain the speed at 1 pu. The reactive power produced by the wind turbine is regulated at 0 Mvar.

Right-click on the "Wind Turbine Type 4" block and select "Look Under Mask" to see how the model is built. The sample time used to discretize the model (Ts= 50 microseconds) is specified in the Initialization function of the Model Properties.

Open the "Wind Turbine Type 4" block menu to see the data of the generator, the converter, the turbine, the drive train and the control systems. In the Display menu select "Turbine data for 1 wind turbine", check "Display wind turbine power characteristics" and then click Apply. The turbine Cp curves are displayed in Figure 1. The turbine power, the tip speed ratio lambda and the Cp values are displayed in Figure 2 as function of wind speed. For a wind speed of 15 m/s, the turbine output power is 1 pu of its rated power, the pitch angle is 8.9 deg and the generator speed is 1 pu.

### 3. Simulation

In this example you will observe the steady-state operation of the Type 4 wind turbine and its dynamic response to voltage sag resulting from a remote fault on the 120-kV system. Open the "120 kV" block modeling the voltage source and see how a six-cycle 0.25 pu voltage drop is programmed at t=0.03 s

Start simulation. Observe voltage and current waveforms on the Scope block. At simulation start the "xInitial" variable containing the initial state variables is automatically loaded (from the "power_wind_type_4_avg.mat" file specified in the Model Properties) so that the simulation starts in steady state.

Initially the Type 4 wind farm produces 10 MW. The corresponding turbine speed is 1 pu of generator synchronous speed. The DC voltage is regulated at 1100 V and reactive power is kept at 0 Mvar. At t=0.03 s the positive-sequence voltage suddenly drops to 0.75 p.u. causing an increase on the DC bus voltage and a drop on the Type 4 wind turbine output power. During the voltage sag the control

systems try to regulate DC voltage and reactive power at their set points (1100 V, 0 Mvar). The system recovers after fault elimination.

**4. Regenerate Initial Conditions**

This example is set-up with all states initialized so that the simulation starts in steady-state. Otherwise, due to the long time constants of the electromechanical part of the wind turbine model and to its relatively slow regulators you would have to wait for tens of seconds before reaching steady-state. The initial conditions have been saved in the "power_wind_type_4_avg.mat" file. When you start simulation, the InitFcn callback (in the Model Properties/Callbacks) automatically loads into your workspace the contents of this .mat file ("xInitial" variable specified in the "Initial state" parameter in the Simulation/Configuration Parameters menu).

If you modify this model, or change parameter values of power components, the initial conditions stored in the "xInitial" variable will no longer be valid and Simulink® will issue an error message. To regenerate the initial conditions for your modified model, follow the steps listed below:

1. In the Configuration Parameters pane, uncheck the "Initial state" parameter and check "Final States" parameter.

2. In the 120 kV Three-phase Voltage Source menu, disable the source voltage step by setting the "Time variation of " parameter to "none".

3. In order to shorten the time required to reach steady-state, temporarily decrease the inertia of the turbine-generator group. Open the Wind Turbine Type 4 menu and in the Drive train data and Generator data, divide the H inertia constants by 10.

4. Change the Simulation Stop Time to 5 seconds. Note that in order to generate initial conditions coherent with the 60 Hz voltage source phase angles, the Stop Time must be an integer number of 60 Hz cycles.

5. Change the Simulation Mode from "Normal" to "Accelerator".

6. Start simulation. When Simulation is completed, verify that steady state has been reached by looking at waveforms displayed on the Scope block. The final states which have been saved in the "xFinal" structure with time can be used as initial states for future simulations. Executing the next two commands copies these final conditions in "xInitial" and saves this variable in a new file (myModel_init.mat).

```
>> xInitial=xFinal;

>> save myModel_init xInitial
```

7. In the InitFcn window of Model Properties pane, replace the first line of initialization commands with "load myModel_init". Next time you start a simulation with this model, the variable xInitial saved in the myModel_init.mat file will be loaded in your workspace.

8. In the Configuration Parameters pane, check "Initial state".

9. In the Wind Turbine Generator and Drive train data, reset the inertia constants H back to their original values.

10. Start simulation and verify that your model starts in steady-state.

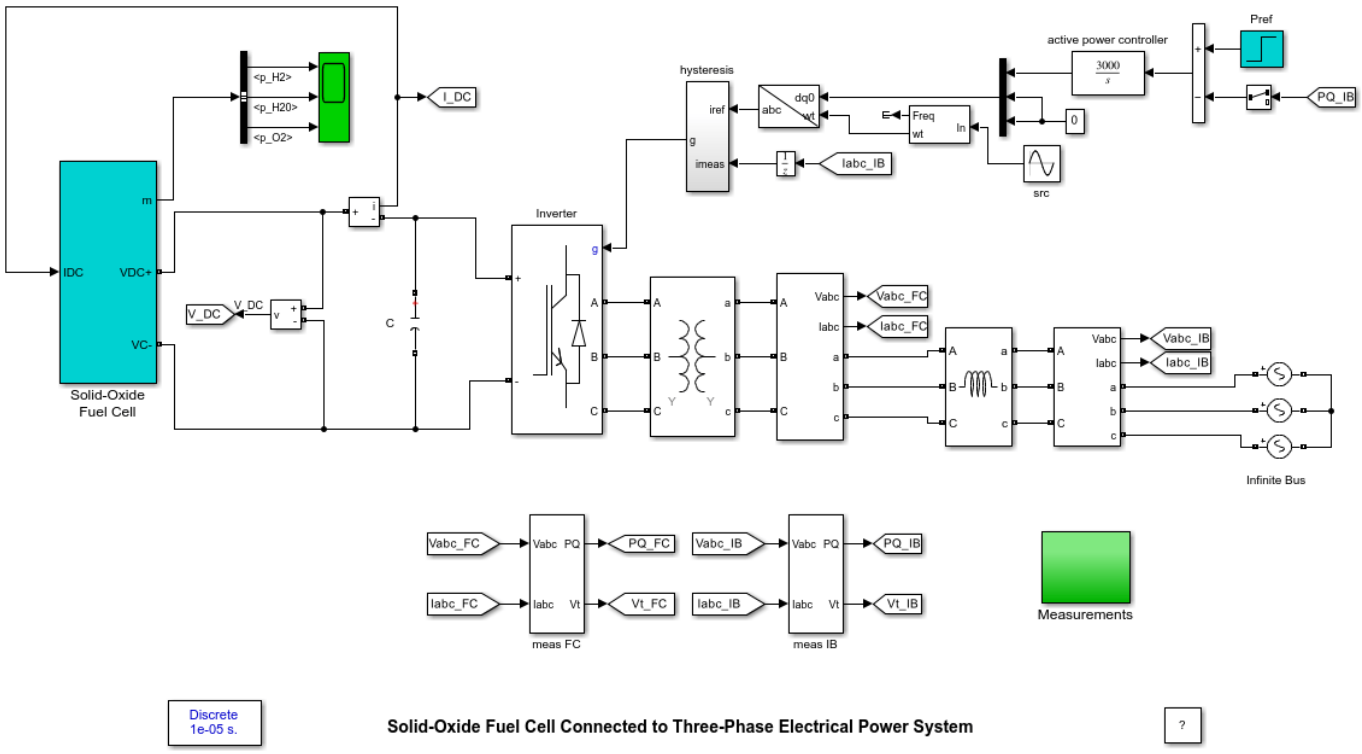
11. In the 120 kV Three-phase voltage source menu, set the "Time variation of" parameter back to "Amplitude".

12. Change the Simulation Stop Time and Simulation Mode back to their original values (0.2 seconds, Normal).

13. Save your model.

# Solid-Oxide Fuel Cell Connected to Three-Phase Electrical Power System

This example shows a model of a solid oxide fuel cell (SOFC) which can be utilized in Specialized Power Systems.




## Description

The system consists of a SOFC which is connected to a three-phase infinite bus through an IGBT inverter. The inverter uses hysteresis switching and controls active power by manipulation of direct-axis current while holding reactive power at 0VAr.

The measurement blocks are rated at 50kW. Therefore, an active power reference of 1pu = 50kW.

## Simulation

Start the simulation.

At t = 0s, an active power reference (Pref) of 0.3pu is commanded. Observe that the reference is captured within 0.2s.

At t = 0.4s, Pref = 1pu is commanded. Again, the reference is captured within 0.2s.

Observation of the H2, H20 and O2 pressures shows that the fuel cell does not reach a new equilibrium for the simulation of duration 1 second. Extended simulation periods are required to observe the dynamics of the chemical reaction.

### Notes

This model assumes the following:

1. The fuel cell gases are ideal
2. Only one pressure is defined in the interior of the electrodes
3. The fuel cell temperature is invariant
4. Nernst's equation applies.

The model is taken from:

Y. Zhu and K. Tomsovic, 'Development of models for analyzing the load-following performance of microturbines and fuel cells', Electric Power Systems Research, 62, 2002, 1-11.