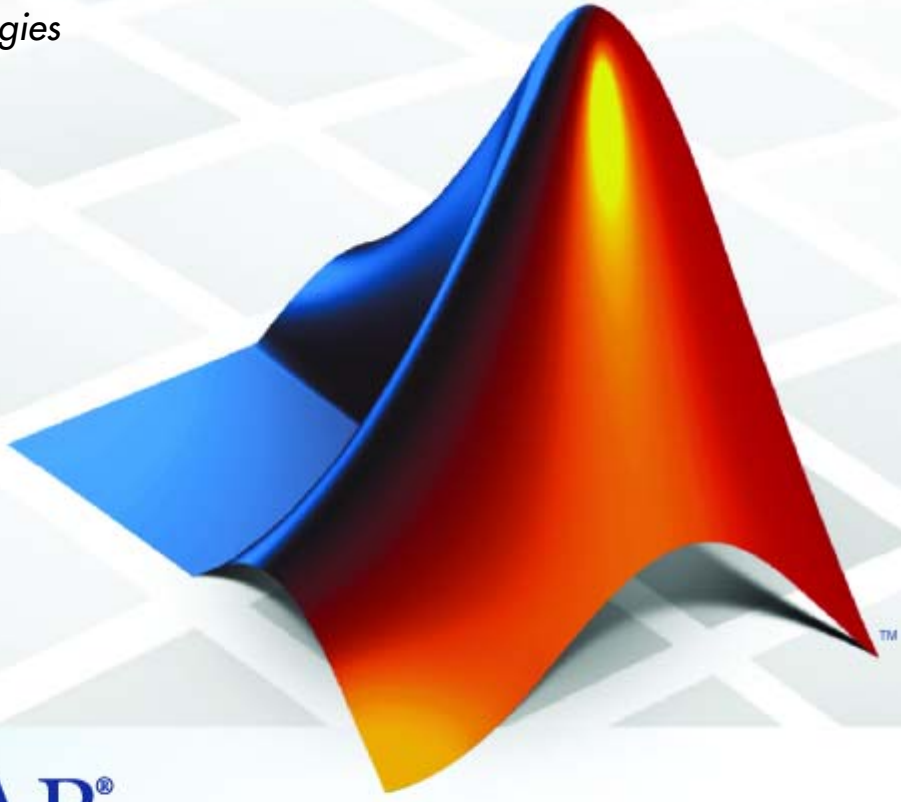
SimPowerSystems™ 4
User's Guide

Hydro-Québec
TransÉnergie Technologies



MATLAB® & SIMULINK®

The MathWorks™
Accelerating the pace of engineering and science

How to Contact The MathWorks

`www.mathworks.com`	Web
`comp.soft-sys.matlab`	Newsgroup
`www.mathworks.com/contact_TS.html`	Technical Support

`suggest@mathworks.com`	Product enhancement suggestions
`bugs@mathworks.com`	Bug reports
`doc@mathworks.com`	Documentation error reports
`service@mathworks.com`	Order status, license renewals, passcodes
`info@mathworks.com`	Sales, pricing, and general information

508-647-7000 (Phone)

508-647-7001 (Fax)

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

For contact information about worldwide offices, see the MathWorks Web site.

SimPowerSystems™ User's Guide

© COPYRIGHT 1998–2008 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

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

Revision History

January 1998	First printing	Version 1.0 (Release 10)
September 2000	Second printing	Revised for Version 2.1 (Release 12)
June 2001	Online only	Revised for Version 2.2 (Release 12.1)
July 2002	Online only	Revised for Version 2.3 (Release 13) (Renamed from *Power System Blockset User's Guide*)
February 2003	Third printing	Revised for Version 3.0 (Release 13SP1)
June 2004	Online only	Revised for Version 3.1 (Release 14)
October 2004	Fourth printing	Revised for Version 4.0 (Release 14SP1)
March 2005	Online only	Revised for Version 4.0.1 (Release 14SP2)
May 2005	Online only	Revised for Version 4.1 (Release 14SP2+)
September 2005	Online only	Revised for Version 4.1.1 (Release 14SP3)
March 2006	Online only	Revised for Version 4.2 (Release 2006a)
September 2006	Online only	Revised for Version 4.3 (Release 2006b)
March 2007	Online only	Revised for Version 4.4 (Release 2007a)
September 2007	Online only	Revised for Version 4.5 (Release 2007b)
March 2008	Online only	Revised for Version 4.6 (Release 2008a)

Acknowledgments

SimPowerSystems™ software Version 4 was developed by the following people and organizations.

Gilbert Sybille
Hydro-Québec Research Institute (IREQ), Varennes, Québec. Original author of SimPowerSystems software, technical coordinator, author of phasor simulation, discretization techniques, and documentation. Technical supervision and design of the FACTS and Distributed Resources libraries, and documentation.

Louis-A. Dessaint
École de Technologie Supérieure (ETS), Montréal, Québec. Author of machine models. Technical supervision and design of the electric drive library contents, and documentation.

Pierre Giroux, Richard Gagnon, Silvano Casoria
Hydro-Québec Research Institute (IREQ), Varennes, Québec. Development of the FACTS and Distributed Resources libraries. Key beta testers and developers of several SimPowerSystems blocks, demos, and documentation.

Patrice Brunelle
TransÉnergie Technologies Inc., Montréal, Québec. Main software engineer. Author of graphical user interfaces, model integration into Simulink® and Physical Modeling, and documentation.

Roger Champagne
École de Technologie Supérieure (ETS), Montréal, Québec. Author of machine models, of revised state space formulation. Design of the graphical user interface of the electric drive library.

Hoang Lehuy
Université Laval, Québec City. Validation tests and author of several models, functions, and documentation. Validation of the electric drives library.

Bruno DeKelper
École de Technologie Supérieure (ETS), Montréal, Québec. Revision and validation of C files and author of TLC functions associated with the simulation of the state space equations.

Handy Fortin-Blanchette, Olivier Tremblay, Christophe Semaille
École de Technologie Supérieure (ETS), Montréal, Québec. Development of the AC and DC drives models.

Hassan Ouquelle, Jean-Nicolas Paquin
École de Technologie Supérieure (ETS), Montréal, Québec. Development of the Single-Phase Asynchronous Machine model and Saturation in Asynchronous Machine model.

Pierre Mercier
iOMEGAt. Project manager for the Power System Blockset™ software versions 1 and 2 and for the Simulink electric drives library.

The authors acknowledge the contributions of the following people:

Innocent Kamwa, Raymond Roussel, Kamal Al-Haddad, Mohamed Tou, Christian Dufour, Momcilo Gavrilovic, Christian Larose, David McCallum, Bahram Khodabakhchian, Manuel Alvarado Sandoval, and Stéphane Desjardins

Contents

Acknowledgments

1 Getting Started

Product Overview **1-2**
Introduction **1-2**
The Role of Simulation in Design **1-2**
SimPowerSystems™ Libraries **1-3**
Required and Related Products **1-4**

Using This Guide **1-5**
If You Are a New User **1-5**
If You Are an Experienced Blockset User **1-5**
All Users **1-6**
Units **1-6**

Building and Simulating a Simple Circuit **1-7**
Introduction **1-7**
Building the Electrical Circuit with powerlib Library **1-8**
Interfacing the Electrical Circuit with Other Simulink® Blocks **1-13**
Measuring Voltages and Currents **1-14**
Basic Principles of Connecting Capacitors and Inductors . . **1-15**
Using the Powergui Block to Simulate SimPowerSystems™ Models **1-16**

Analyzing a Simple Circuit **1-18**
Introduction **1-18**
Electrical State Variables **1-18**
State-Space Representation Using power_analyze **1-19**
Steady-State Analysis **1-19**
Frequency Analysis **1-21**

Specifying Initial Conditions **1-27**
Introduction **1-27**
State Variables **1-27**
Initial States **1-28**
Specify Initial Electrical States with Powergui **1-29**

Simulating Transients **1-33**
Introduction **1-33**
Simulating Transients with a Circuit Breaker **1-33**
Continuous, Variable Time Step Integration Algorithms . . **1-35**
Discretizing the Electrical System **1-37**

Introducing the Phasor Simulation Method **1-40**
Introduction **1-40**
When to Use the Phasor Solution **1-40**
Phasor Simulation of a Circuit Transient **1-41**

2 Advanced Components and Techniques

Introducing Power Electronics **2-2**
Introduction **2-2**
Simulation of the TCR Branch **2-4**
Simulation of the TSC Branch **2-7**

Simulating Variable Speed Motor Control **2-10**
Introduction **2-10**
Building and Simulating the PWM Motor Drive **2-12**
Using the Multimeter Block **2-18**
Discretizing the PWM Motor Drive **2-20**
Performing Harmonic Analysis Using the FFT Tool **2-20**

Three-Phase Systems and Machines **2-24**
Introduction **2-24**
Three-Phase Network with Electrical Machines **2-24**
Load Flow and Machine Initialization **2-27**
Using the Phasor Solution Method for Stability Studies . . **2-35**

Building and Customizing Nonlinear Models **2-39**
Introduction **2-39**
Modeling a Nonlinear Inductance **2-39**
Customizing Your Nonlinear Model **2-44**
Modeling a Nonlinear Resistance **2-47**
Creating Your Own Library **2-52**
Connecting Your Model with Other Nonlinear Blocks **2-52**

Building a Model Using Model Construction Commands **2-56**

3 Improving Simulation Performance

How SimPowerSystems™ Software Works **3-3**

Choosing an Integration Method **3-5**
Introduction **3-5**
Continuous Versus Discrete Solution **3-5**
Phasor Solution Method **3-6**

Simulating with Continuous Integration Algorithms . . **3-7**
Choosing an Integration Algorithm **3-7**
Simulating Switches and Power Electronic Devices **3-8**

Simulating Discretized Electrical Systems **3-9**
Introduction **3-9**
Limitations of Discretization with Nonlinear Models **3-9**

Simulating Power Electronic Models **3-11**
Introduction **3-11**
Circuit Using Forced-Commutated Power Electronics **3-11**
Circuit Using Naturally Commutated Power Electronics . . **3-11**

Increasing Simulation Speed **3-13**
Ways to Increase Simulation Speed **3-13**
Using Accelerator Mode and Real-Time Workshop® Software **3-13**

The Nonlinear Model Library **3-16**
How to Access the Nonlinear Model Library **3-16**
The Continuous Library **3-16**
The Discrete Library **3-17**
The Phasors Library **3-17**
The Switch Current Source Library **3-17**
Limitations of the Nonlinear Models **3-17**
Modifying the Nonlinear Models of the powerlib_models Library **3-18**

Creating Your Own Library of Models **3-19**

Changing Your Circuit Parameters **3-20**
Introduction **3-20**
Example of MATLAB® Script Performing a Parametric Study **3-20**

4 Systems with Electric Drives

About the Electric Drives Library **4-3**

Getting Started with Electric Drives Library **4-5**
What Is an Electric Drive? **4-5**
Three Main Components of an Electric Drive **4-5**
Multiquadrant Operation **4-8**
Average-Value Models **4-9**
User Interface **4-10**
General Layout of the Library's GUIs **4-10**
Features of the New GUIs **4-11**
Advanced Usage **4-13**

Simulating a DC Motor Drive **4-14**
Introduction **4-14**
Regenerative Braking **4-15**
Example: Thyristor Converter-Based DC Motor Drive . . . **4-15**

Simulating an AC Motor Drive **4-39**

Introduction 4-39
Dynamic Braking 4-39
Modulation Techniques 4-40
Open-Loop Volts/Hertz Control 4-45
Closed-Loop Speed Control with Slip Compensation 4-46
Flux-Oriented Control 4-46
Direct Torque Control 4-49
Example: AC Motor Drive 4-50

Mechanical Models 4-70
Mechanical Shaft Block 4-70
Speed Reducer Block 4-71

Mechanical Coupling of Two Motor Drives 4-72
Introduction 4-72
System Description 4-73
Speed Regulated AC4 with Torque Regulated DC2 4-75
Torque Regulated AC4 with Speed Regulated DC2 4-76

Winding Machine 4-79
Introduction 4-79
Description of the Winder 4-79
Block Description 4-81
Simulation Results 4-84

Robot Axis Control Using Brushless DC Motor Drive .. 4-87
Introduction 4-87
Description of the Robot Manipulator 4-87
Position Control Systems for Joints 1 and 2 4-88
Modeling the Robot Position Control Systems 4-89
Tracking Performance of the Motor Drives 4-93

Building Your Own Drive 4-98
Introduction 4-98
Description of the Drive 4-99
Modeling the Induction Motor Drive 4-101
Starting the Drive 4-105
Steady-State Voltage and Current Waveforms 4-106
Speed Regulation Dynamic Performance 4-106

Transients and Power Electronics in Power Systems

5

Series-Compensated Transmission System **5-3**
Description of the Transmission System **5-3**
Setting the Initial Load Flow and Obtaining Steady State **5-9**
Transient Performance for a Line Fault **5-10**
Frequency Analysis **5-14**
Transient Performance for a Fault at Bus B2 . . . . . **5-17**

Thyristor-Based Static Var Compensator **5-21**
Introduction **5-21**
Description of the SVC **5-22**
Steady-State and Dynamic Performance of the SVC **5-25**
Misfiring of TSC1 . . . . . **5-27**

GTO-Based STATCOM **5-30**
Introduction **5-30**
Description of the STATCOM **5-31**
Steady-State and Dynamic Performance of the STATCOM **5-37**

Thyristor-Based HVDC Link **5-40**
Description of the HVDC Transmission System **5-40**
Frequency Response of the AC and DC Systems **5-42**
Description of the Control and Protection Systems **5-44**
System Startup/Stop — Steady-State and Step Response **5-49**
DC Line Fault **5-56**
AC Line-to-Ground Fault at the Inverter **5-59**

VSC-Based HVDC Link **5-63**
Introduction **5-63**
Description of the HVDC Link **5-63**
VSC Control System **5-67**
Dynamic Performance **5-73**

Transient Stability of Power Systems Using Phasor Simulation

6

Transient Stability of a Power System with SVC and PSS **6-3**
Introduction **6-3**
Description of the Transmission System **6-3**
Single-Phase Fault — Impact of PSS — No SVC **6-5**
Three-Phase Fault — Impact of SVC — PSS in Service . . . **6-7**

Control of Power Flow Using a UPFC and a PST **6-10**
Introduction **6-10**
Description of the Power System **6-10**
Power Flow Control with the UPFC **6-13**
UPFC P-Q Controllable Region **6-14**
Power Flow Control Using a PST **6-15**

Wind Farm Using Doubly-Fed Induction Generators . . **6-20**
Description of the Wind Farm **6-20**
Turbine Response to a Change in Wind Speed **6-24**
Simulation of a Voltage Sag on the 120 kV System **6-26**
Simulation of a Fault on the 25 kV System **6-28**

Index

Getting Started

Product Overview (p. 1-2)	Use SimPowerSystems™ simulation with other MathWorks™ products to design electrical power systems
Using This Guide (p. 1-5)	Understand the expected background of users and how to most effectively use this guide
Building and Simulating a Simple Circuit (p. 1-7)	Build a simple circuit with SimPowerSystems blocks and connect it to other Simulink® blocks
Analyzing a Simple Circuit (p. 1-18)	Use the Powergui block and analyze static and frequency-domain response
Specifying Initial Conditions (p. 1-27)	Learn about the state variables of a Simulink diagram and specifying initial conditions for the electrical state variables
Simulating Transients (p. 1-33)	Create an electrical subsystem, simulate transients, and discretize simple circuits
Introducing the Phasor Simulation Method (p. 1-40)	Use the phasor method to analyze magnitudes and phases in linear circuits

Product Overview

In this section...
"Introduction" on page 1-2
"The Role of Simulation in Design" on page 1-2
"SimPowerSystems™ Libraries" on page 1-3
"Required and Related Products" on page 1-4

Introduction

SimPowerSystems™ software and other products of the Physical Modeling product family work together with Simulink® software to model electrical, mechanical, and control systems.

SimPowerSystems software operates in the Simulink environment. Therefore, before starting this user's guide, make yourself familiar with Simulink documentation. Or, if you perform signal processing and communications tasks (as opposed to control system design tasks), see the Signal Processing Blockset™ documentation.

The Role of Simulation in Design

Electrical power systems are combinations of electrical circuits and electromechanical devices like motors and generators. Engineers working in this discipline are constantly improving the performance of the systems. Requirements for drastically increased efficiency have forced power system designers to use power electronic devices and sophisticated control system concepts that tax traditional analysis tools and techniques. Further complicating the analyst's role is the fact that the system is often so nonlinear that the only way to understand it is through simulation.

Land-based power generation from hydroelectric, steam, or other devices is not the only use of power systems. A common attribute of these systems is their use of power electronics and control systems to achieve their performance objectives.

SimPowerSystems software is a modern design tool that allows scientists and engineers to rapidly and easily build models that simulate power systems. It uses the Simulink environment, allowing you to build a model using simple *click and drag* procedures. Not only can you draw the circuit topology rapidly, but your analysis of the circuit can include its interactions with mechanical, thermal, control, and other disciplines. This is possible because all the electrical parts of the simulation interact with the extensive Simulink modeling library. Since Simulink uses the MATLAB® computational engine, designers can also use MATLAB toolboxes and Simulink blocksets. SimPowerSystems software belongs to the Physical Modeling product family and uses similar block and connection line interface.

SimPowerSystems™ Libraries

SimPowerSystems libraries contain models of typical power equipment such as transformers, lines, machines, and power electronics. These models are proven ones coming from textbooks, and their validity is based on the experience of the Power Systems Testing and Simulation Laboratory of Hydro-Québec, a large North American utility located in Canada, and also on the experience of École de Technologie Supérieure and Université Laval. The capabilities of SimPowerSystems software for modeling a typical electrical system are illustrated in demonstration files. And for users who want to refresh their knowledge of power system theory, there are also self-learning case studies.

The SimPowerSystems main library, **powerlib**, organizes its blocks into libraries according to their behavior. The **powerlib** library window displays the block library icons and names. Double-click a library icon to open the library and access the blocks. The main **powerlib** library window also contains the Powergui block that opens a graphical user interface for the steady-state analysis of electrical circuits.

Nonlinear Simulink Blocks for SimPowerSystems Models

The nonlinear Simulink blocks of the **powerlib** library are stored in a special block library named **powerlib_models**. These masked Simulink models are used by SimPowerSystems software to build the equivalent Simulink model of your circuit. See Chapter 3, “Improving Simulation Performance” for a description of the **powerlib_models** library.

Required and Related Products

SimPowerSystems software requires the following products:

- MATLAB
- Simulink

In addition to SimPowerSystems software, the Physical Modeling product family includes other products for modeling and simulating mechanical and electrical systems. Use these products together to model physical systems in Simulink environment. There are also a number of closely related toolboxes and other products from The MathWorks™ that you can use with SimPowerSystems software. For more information about any of these products, see the MathWorks Web site at `http://www.mathworks.com`; see the “Products” section.

Using This Guide

In this section...
"If You Are a New User" on page 1-5
"If You Are an Experienced Blockset User" on page 1-5
"All Users" on page 1-6
"Units" on page 1-6

If You Are a New User

Begin with this chapter and also the next chapter to learn how to

- Build and simulate electrical circuits using the **powerlib** library
- Interface an electrical circuit with Simulink® blocks
- Analyze the steady-state and frequency response of an electrical circuit
- Discretize your model to increase simulation speed, especially for power electronic circuits and large power systems
- Use the phasor simulation method
- Build your own nonlinear models

If You Are an Experienced Blockset User

See the Release Notes for details on the latest release.

Also, see these chapters:

- Chapter 1, "Getting Started" to learn how to simulate discretized electrical circuits
- Chapter 2, "Advanced Components and Techniques" to learn how to apply the phasor simulation to transient stability study of multimachine systems
- Chapter 3, "Improving Simulation Performance" to learn how to increase simulation speed

All Users

For reference information on blocks, simple demos, and GUI-based tools, use the SimPowerSystems™ Reference.

For commands, refer to “Command Reference” for a synopsis of the command syntax, as well as a complete explanation of options and operation.

Units

This guide uses the International System of Units (SI) and the per unit (pu) system. See Technical Conventions for details.

Building and Simulating a Simple Circuit

In this section...

"Introduction" on page 1-7

"Building the Electrical Circuit with powerlib Library" on page 1-8

"Interfacing the Electrical Circuit with Other Simulink® Blocks" on page 1-13

"Measuring Voltages and Currents" on page 1-14

"Basic Principles of Connecting Capacitors and Inductors" on page 1-15

"Using the Powergui Block to Simulate SimPowerSystems™ Models" on page 1-16

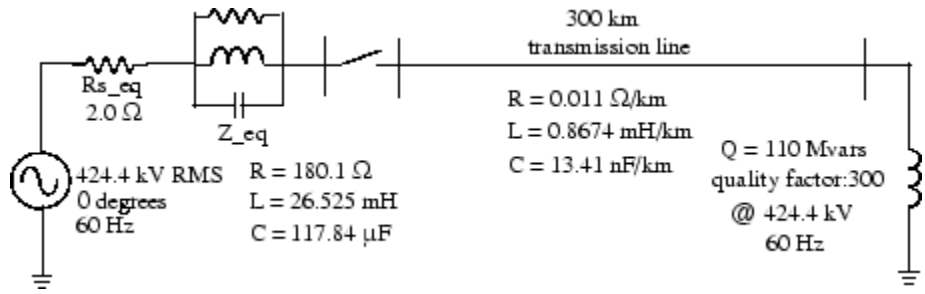
Introduction

SimPowerSystems™ software allows you to build and simulate electrical circuits containing linear and nonlinear elements.

In this section you

- Explore the **powerlib** library
- Learn how to build a simple circuit from the **powerlib** library
- Interconnect Simulink® blocks with your circuit

The circuit below represents an equivalent power system feeding a 300 km transmission line. The line is compensated by a shunt inductor at its receiving end. A circuit breaker allows energizing and de-energizing of the line. To simplify matters, only one of the three phases is represented. The parameters shown in the figure are typical of a 735 kV power system.



Circuit to Be Modeled

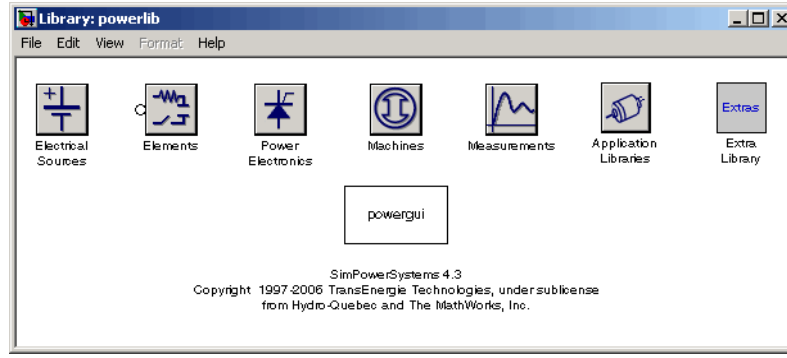
Building the Electrical Circuit with powerlib Library

The graphical user interface makes use of the Simulink functionality to interconnect various electrical components. The electrical components are grouped in a library called **powerlib**.

1 Open the SimPowerSystems main library by entering the following command at the MATLAB® prompt.

```
powerlib
```

This command displays a Simulink window showing icons of different block libraries.



You can open these libraries to produce the windows containing the blocks to be copied into your circuit. Each component is represented by a special

icon having one or several inputs and outputs corresponding to the different terminals of the component:

2 From the **File** menu of the **powerlib** window, open a new window to contain your first circuit and save it as `circuit1`.

3 Open the Electrical Sources library and copy the AC Voltage Source block into the **circuit1** window.

4 Open the **AC Voltage Source** dialog box by double-clicking the icon and enter the Amplitude, Phase, and Frequency parameters according to the values shown in Circuit to Be Modeled on page 1-8.

   Note that the amplitude to be specified for a sinusoidal source is its peak value (424.4e3*sqrt(2) volts in this case).

5 Change the name of this block from AC Voltage Source to Vs.

6 Copy the Parallel RLC Branch block, which can be found in the Elements library of **powerlib**, set its parameters as shown in Circuit to Be Modeled on page 1-8, and name it Z_eq.

7 The resistance Rs_eq of the circuit can be obtained from the Parallel RLC Branch block. Duplicate the Parallel RLC Branch block, which is already in your **circuit1** window. Select R for the Branch Type parameter and set the R parameter according to Circuit to Be Modeled on page 1-8.

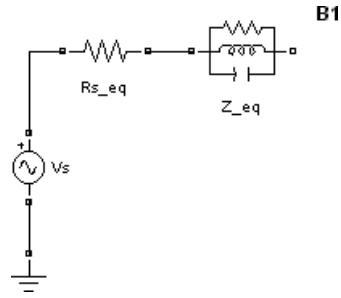
   Once the dialog box is closed, notice that the L and C components have disappeared so that the icon now shows a single resistor.

Note With the Branch Type parameter set to RLC, setting L and C respectively to `inf` and zero in a parallel branch changes automatically the Branch Type to R and produces the same result. Similarly, with the Series RLC Branch block, setting R, L, and C respectively to zero, zero, and `inf` eliminates the corresponding element.

---

8 Name this block Rs_eq.

9 Resize the various components and interconnect blocks by dragging lines from outputs to inputs of appropriate blocks.



10 To complete the circuit of Circuit to Be Modeled on page 1-8, you need to add a transmission line and a shunt reactor. You add the circuit breaker later in "Simulating Transients" on page 1-33.

The model of a line with uniformly distributed R, L, and C parameters normally consists of a delay equal to the wave propagation time along the line. This model cannot be simulated as a linear system because a delay corresponds to an infinite number of states. However, a good approximation of the line with a finite number of states can be obtained by cascading several PI circuits, each representing a small section of the line.

A PI section consists of a series R-L branch and two shunt C branches. The model accuracy depends on the number of PI sections used for the model. Copy the PI Section Line block from the Elements library into the **circuit1** window, set its parameters as shown in Circuit to Be Modeled on page 1-8, and specify one line section.

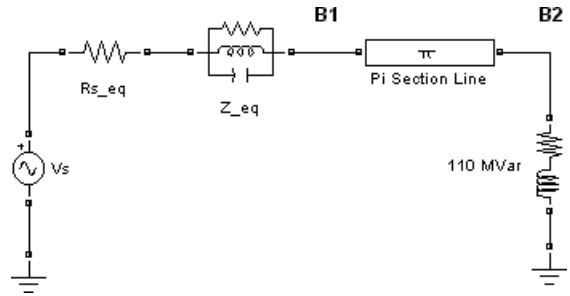
11 The shunt reactor is modeled by a resistor in series with an inductor. You could use a Series RLC Branch block to model the shunt reactor, but then you would have to manually calculate and set the R and L values from the quality factor and reactive power specified in Circuit to Be Modeled on page 1-8.

Therefore, you might find it more convenient to use a Series RLC Load block that allows you to specify directly the active and reactive powers absorbed by the shunt reactor.

Copy the Series RLC Load block, which can be found in the Elements library of **powerlib**. Name this block 110 Mvar. Set its parameters as follows:

Vn	`424.4e3 V`
fn	`60 Hz`
P	`110e6/300 W (quality factor = 300)`
QL	`110e6` vars
Qc	`0`

Note that, as no reactive capacitive power is specified, the capacitor disappears on the block icon when the dialog box is closed. Interconnect the new blocks as shown.



12 You need a Voltage Measurement block to measure the voltage at node B1. This block is found in the Measurements library of **powerlib**. Copy it and name it U1. Connect its positive input to the node B1 and its negative input to a new Ground block.

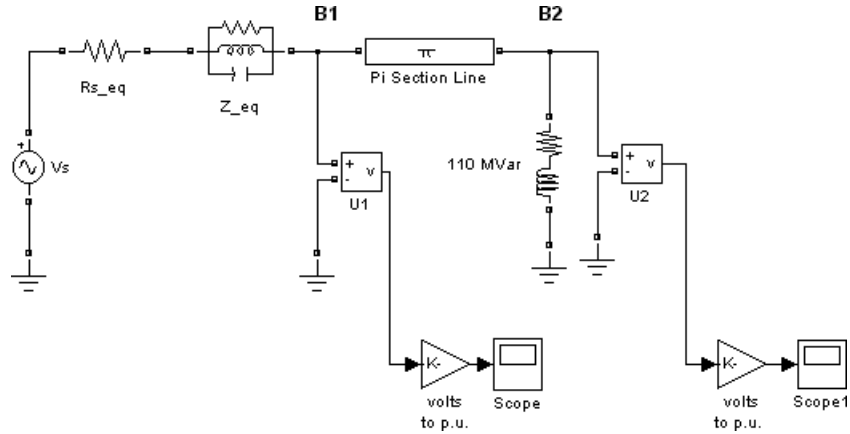
13 To observe the voltage measured by the Voltage Measurement block named U1, a display system is needed. This can be any device found in the Simulink Sinks library.

Open the Sinks library and copy the Scope block into your **circuit1** window. If the scope were connected directly at the output of the voltage measurement, it would display the voltage in volts. However, electrical engineers in power systems are used to working with normalized quantities (per unit system). The voltage is normalized by dividing the value in volts

by a base voltage corresponding to the peak value of the system nominal voltage. In this case the scaling factor K is

$$K = \frac{1}{424.4 \times 10^3 \times \sqrt{2}}$$

14 Copy a Gain block from the Simulink library and set its gain as above. Connect its output to the Scope block and connect the output of the Voltage Measurement block to the Gain block. Duplicate this voltage measurement system at the node B2, as shown below.



15 From the **Simulation** menu, select **Start**. A Powergui block is automatically added to your model. The purpose of this block is discussed in "Using the Powergui Block to Simulate SimPowerSystems™ Models" on page 1-16.

16 Open the Scope blocks and observe the voltages at nodes B1 and B2.

17 While the simulation is running, open the Vs block dialog box and modify the amplitude. Observe the effect on the two scopes. You can also modify the frequency and the phase. You can zoom in on the waveforms in the scope windows by drawing a box around the region of interest with the left mouse button.

To simulate this circuit, the default integration algorithm (`ode45`) was used. However, for most SimPowerSystems applications, your circuits contain

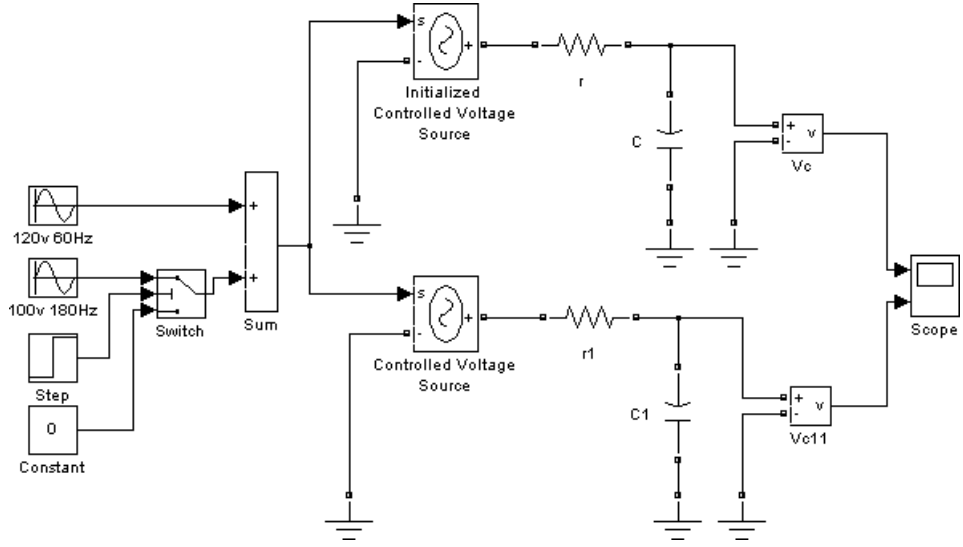
switches and other nonlinear models. In such a case, you must specify a different integration algorithm. This is discussed in “Simulating Transients” on page 1-33, where a circuit breaker is added to your circuit.

Interfacing the Electrical Circuit with Other Simulink® Blocks

The Voltage Measurement block acts as an interface between the SimPowerSystems blocks and the Simulink blocks. For the system shown above, you implemented such an interface from the electrical system to the Simulink system. The Voltage Measurement block converts the measured voltages into Simulink signals.

Similarly, the Current Measurement block from the Measurements library of **powerlib** can be used to convert any measured current into a Simulink signal.

You can also interface from Simulink blocks to the electrical system. For example, you can use the Controlled Voltage Source block to inject a voltage in an electrical circuit, as shown in the following figure.



Electrical Terminal Ports and Connection Lines SimPowerSystems modeling environment is similar to that of other products in the Physical Modeling family. Its blocks often feature both normal Simulink input and output ports > and special electrical terminal ports □:

- Lines that connect normal Simulink ports > are directional signal lines.
- Lines that connect terminal ports □ are special electrical connection lines. These lines are nondirectional and can be branched, but you cannot connect them to Simulink ports > or to normal Simulink signal lines.
- You can connect Simulink ports > only to other Simulink ports and electrical terminal ports □ only to other electrical terminal ports.
- Converting Simulink signals to electrical connections or vice versa requires using a SimPowerSystems block that features both Simulink ports and electrical terminal ports.

  Some SimPowerSystems blocks feature only one type of port.

Measuring Voltages and Currents

When you measure a current using a Current Measurement block, the positive direction of current is indicated on the block icon (positive current flowing from + terminal to – terminal). Similarly, when you measure a voltage using a Voltage Measurement block, the measured voltage is the voltage of the + terminal with respect to the – terminal. However, when voltages and currents of blocks from the Elements library are measured using the Multimeter block, the voltage and current polarities are not immediately obvious because blocks might have been rotated and there are no signs indicating polarities on the block icons.

Unlike Simulink signal lines and input and output ports, the SimPowerSystems connection lines and terminal ports □ lack intrinsic directionality. The voltage and current polarities are determined, not by line direction, but instead by block orientation. To find out a block orientation, first click the block to select it. Then enter the following command.

```
get_param(gcb,'Orientation')
```

The following table indicates the polarities of the currents and voltages measured with the Multimeter block for single-phase and three-phase RLC branch and loads (and of the polarity of the capacitor voltage and the inductor current), surge arresters, and single-phase and three-phase breakers.

Block Orientation	Positive Current Direction	Measured Voltage
`right`	left —> right	Vleft – Vright
`left`	right —> left	Vright – Vleft
`down`	top —> bottom	Vtop – Vbottom
`up`	bottom —> top	Vbottom – Vtop

The natural orientation of the blocks (that is, their orientation in the Element library) is *right* for horizontal blocks and *down* for vertical blocks.

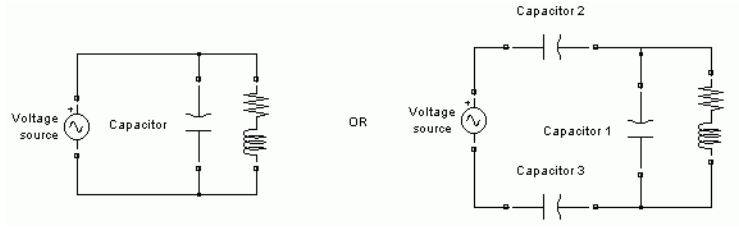
For single-phase transformers (linear or saturable), with the winding connectors appearing on the left and right sides, the winding voltages are the voltages of the top connector with respect to the bottom connector, irrespective of the block orientation (*right* or *left*). The winding currents are the currents entering the top connector.

For three-phase transformers, the voltage polarities and positive current directions are indicated by the signal labels used in the Multimeter block. For example, Uan_w2 means phase A-to-neutral voltage of the Y connected winding #2, Iab_w1 means winding current flowing from A to B in the delta-connected winding #1.

Basic Principles of Connecting Capacitors and Inductors

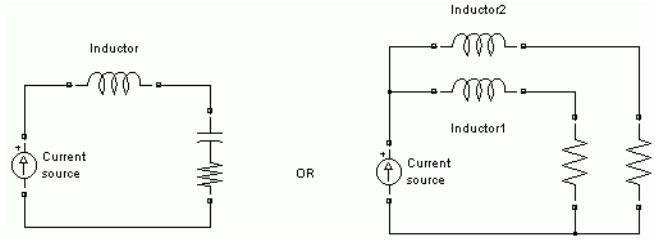
You have to pay particular attention when you connect capacitor elements together with voltage sources, or inductor elements in series with current sources. When you start the simulation, the software displays an error message if one of the following two connection errors are present in your diagram:

1 You have connected a voltage source in parallel with a capacitor, or a series of capacitor elements in series, like in the two examples below.



To fix this problem, you can add a small resistance in series between the voltage source and the capacitors.

2 You have connected a current source in series with an inductor, or a series of inductors connected in parallel, like in the example below.



To fix this problem, you can add a large resistance in parallel with the inductor and the capacitors.

Using the Powergui Block to Simulate SimPowerSystems™ Models

The Powergui block is necessary for simulation of any Simulink model containing SimPowerSystems blocks. It is used to store the equivalent Simulink circuit that represents the state-space equations of the SimPowerSystems blocks.

You must follow these rules when using this block in a model:

- Place the Powergui block at the top level of diagram for optimal performance. However, you can place it anywhere inside subsystems for your convenience; its functionality will not be affected.
- You can have a maximum of one Powergui block per model
- You must name the block `powergui`

Note When you start the simulation, the software will automatically add a Powergui block at the top level of a diagram if no such block is found in your model.

Analyzing a Simple Circuit

In this section...

"Introduction" on page 1-18

"Electrical State Variables" on page 1-18

"State-Space Representation Using power_analyze" on page 1-19

"Steady-State Analysis" on page 1-19

"Frequency Analysis" on page 1-21

Introduction

In this section you

- Obtain the state-space representation of your model with the `power_analyze` command
- Compute the steady-state voltages and currents using the graphical user interface of the Powergui block
- Analyze an electrical circuit in the frequency domain

Electrical State Variables

The electrical state variables are the Simulink® states of your diagram associated to the capacitor and inductor devices of the SimPowerSystems™ blocks. Inductors and capacitors elements are found in the RLC-branch type blocks such as the Series RLC Branch block, Three-Phase Parallel RLC Load block, in the transformer models, in the PI Section Line block, in the snubber devices of the power electronic devices, etc.

The electrical state variables consist of the inductor currents and the capacitor voltages. Variable names forSimPowerSystems electrical states contain the name of the block where the inductor or capacitor is found, preceded by the `Il_` prefix for inductor currents or the `Uc_` prefix for capacitor voltages.

State-Space Representation Using power_analyze

You compute the state-space representation of the model `circuit1` with the `power_analyze` command. Enter the following command at the MATLAB® prompt.

```
[A,B,C,D,x0,electrical_states,inputs,outputs]=power_analyze('circuit1')
```

The `power_analyze` command returns the state-space model of your circuit in the four matrices A, B, C, and D. x0 is the vector of initial conditions of the electrical states of your circuit. The names of the electrical state variables, inputs, and outputs are returned in three string matrices.

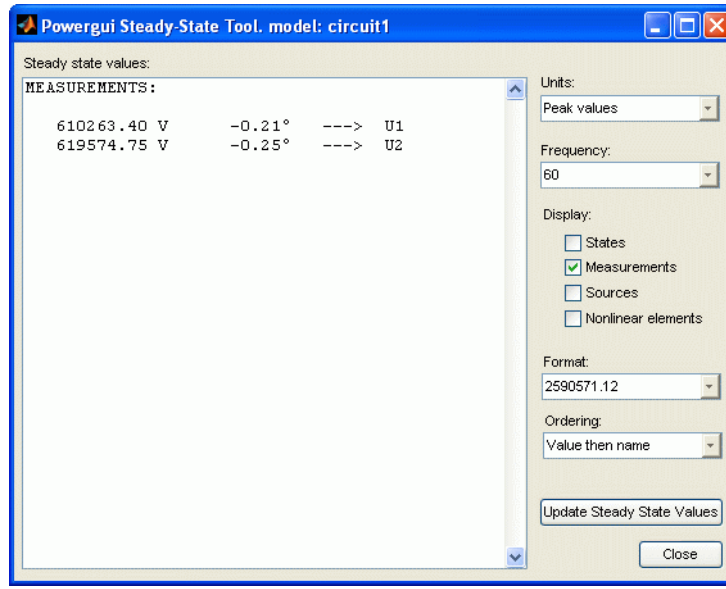
```
electrical_states =

Il_110 Mvars
Uc_input PI Section Line
Il_ sect1 PI Section Line
Uc_output PI Section Line
Il_Z_eq
Uc_Z_eq

inputs =

U_Vs

outputs =

U_U1
U_U2
```

Note that you could have obtained the names and ordering of the electrical states, inputs, and outputs directly from the Powergui block. See the `power_analyze` reference page for more details on how to use this function.

Steady-State Analysis

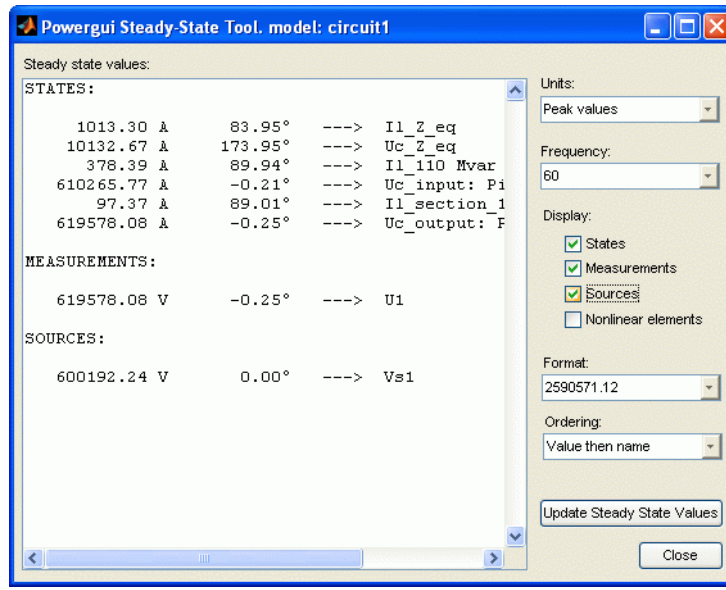
To facilitate the steady-state analysis of your circuit, the **powerlib** library includes a graphical user interface tool. If the Powergui block is not already present in your model, copy the block from the library into your **circuit1** model and double-click the block icon to open it.

From the **Analysis tools** menu of the Powergui block, select **Steady-State Voltages and Currents**. This opens the **Steady-State Tool** window where the steady-state phasors voltages measured by the two voltage measurement blocks of your model are displayed in polar form.



Each measurement output is identified by a string corresponding to the measurement block name. The magnitudes of the phasors U1 and U2 correspond to the peak value of the sinusoidal voltages.

From the **Steady-State Tool** window, you can also display the steady-state values of the source voltage or the steady-state values of the inductor currents and capacitor voltages by selecting either the **Sources** or the **States** check box.



Note Depending on the order you added the blocks in your `circuit1` diagram, the electrical state variables might not be ordered in the same way as in the preceding figure.

Refer to the section “Measuring Voltages and Currents” on page 1-14 for more details on the sign conventions used for the voltages and currents of sources and electrical state variables listed in the **Steady-State Tool** window.

Frequency Analysis

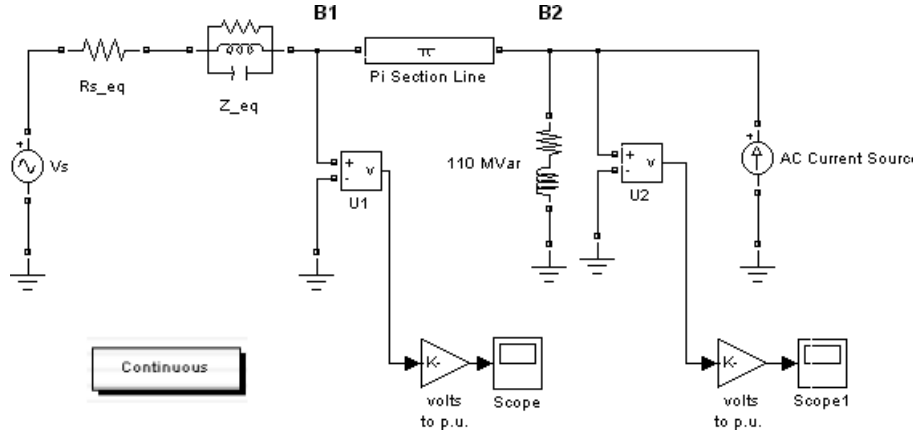
The Measurements library of **powerlib** contains an Impedance Measurement block that measures the impedance between any two nodes of a circuit. In the following two sections, you measure the impedance of your circuit between node B2 and ground by using two methods:

- Calculation from the state-space model
- Automatic measurement using the Impedance Measurement block and the Powergui block

Obtaining the Impedance vs. Frequency Relation from the State-Space Model

Note The following section assumes you have Control System Toolbox™ software installed.

To measure the impedance versus frequency at node B2, you need a current source at node B2 providing a second input to the state-space model. Open the Electrical Sources library and copy the AC Current Source block into your model. Connect this source at node B2, as shown below. Set the current source magnitude to zero and keep its frequency at `60 Hz`. Rearrange the blocks as follows.



AC Current Source at the B2 Node

Now compute the state-space representation of the model `circuitl` with the `power_analyze` command. Enter the following command at the MATLAB prompt.

```
sys1 = power_analyze('circuit1','ss')
```

This command returns a state-space model representing the continuous-time state-space model of your electrical circuit.

In the Laplace domain, the impedance Z2 at node B2 is defined as the transfer function between the current injected by the AC current Source block and the voltage measured by the U2 Voltage Measurement block.

$$Z2(s) = \frac{U2(s)}{I2(s)}$$

You obtain the names of the inputs and outputs of this state-space model as follows.

```
sys1.InputName
ans =
    'U_Vs'
    'I_AC Current Source'
sys1.OutputName
ans =
    'U_U2'
    'U_U1'
```

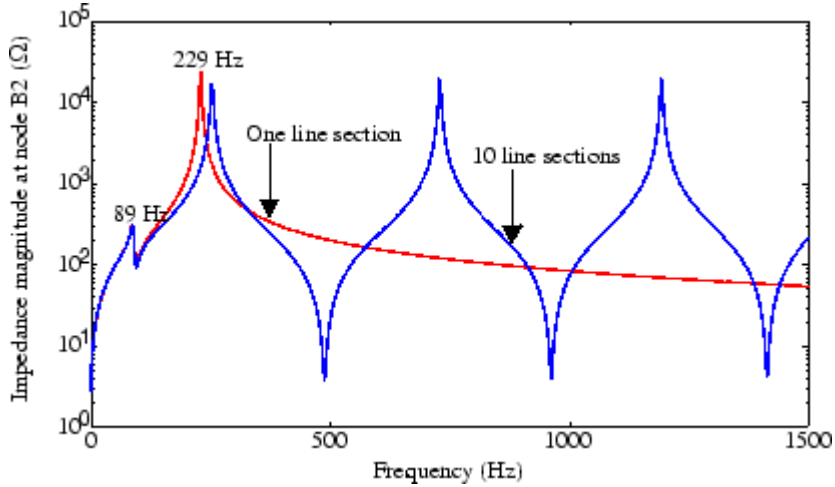
The impedance at node B2 then corresponds to the transfer function between output 2 and input 1 of this state-space model. For the 0 to 1500 Hz range, it can be calculated and displayed as follows.

```
freq=0:1500;
w=2*pi*freq;
bode(sys1(1,2),w);
```

Repeat the same process to get the frequency response with a 10 section line model. Open the **PI Section Line** dialog box and change the number of sections from 1 to 10. To calculate the new frequency response and superimpose it upon the one obtained with a single line section, enter the following commands.

```
sys10 = power_analyze('circuit1','ss');
bode(sys1(1,2),sys10(1,2),w);
```

Open the property editor of the Bode plot and select units for Frequency in Hz using linear scale and Magnitude in absolute using log scale. The resulting plot is shown below.



Impedance at Node B2 as Function of Frequency

This graph indicates that the frequency range represented by the single line section model is limited to approximately 150 Hz. For higher frequencies, the 10 line section model is a better approximation.

The system with a single PI section has two oscillatory modes at 89 Hz and 229 Hz. The 89 Hz mode is due to the equivalent source, which is modeled by a single pole equivalent. The 229 Hz mode is the first mode of the line modeled by a single PI section.

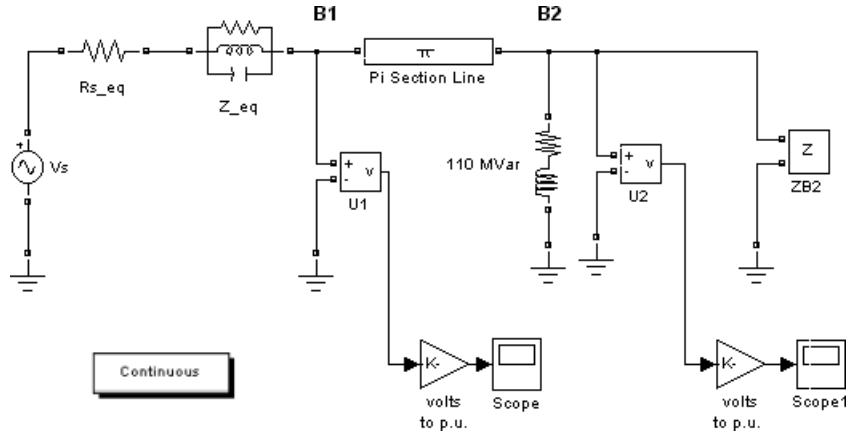
For a distributed parameter line model the propagation speed is

$$v = \frac{1}{\sqrt{L \cdot C}} = 293{,}208 \text{ km/s}$$

The propagation time for 300 km is therefore T = 300/293,208 = 1.023 ms and the frequency of the first line mode is f1 = 1/4T = 244 Hz. A distributed parameter line would have an infinite number of modes every 244 + n*488 Hz (n = 1, 2, 3...). The 10 section line model simulates the first 10 modes. The first three line modes can be seen in Impedance at Node B2 as Function of Frequency on page 1-24 (244 Hz, 732 Hz, and 1220 Hz).

Obtaining the Impedance vs. Frequency Relation from the Impedance Measurement and Powergui Blocks

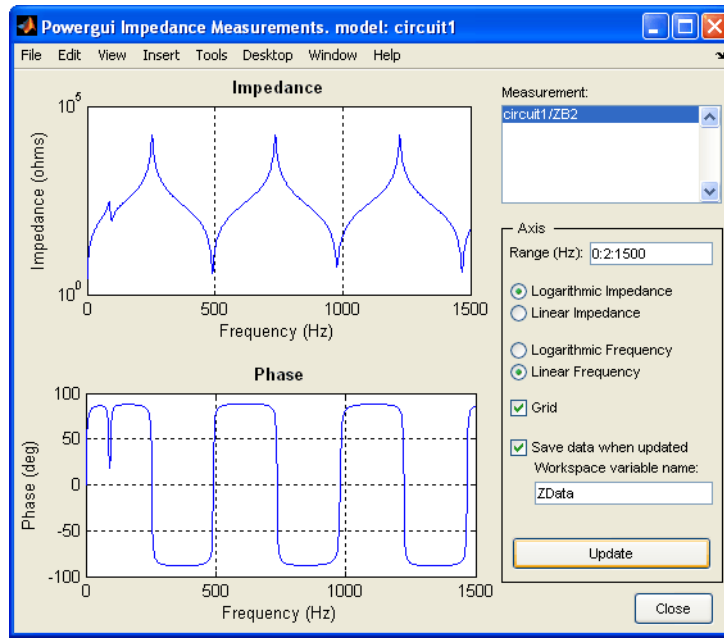
The process described above to measure a circuit impedance has been automated in a SimPowerSystems block. Open the Measurements library of **powerlib**, copy the Impedance Measurement block into your model, and rename it ZB2. Connect the two inputs of this block between node B2 and ground as shown.



Measuring Impedance vs. Frequency with the Impedance Measurement Block

Now open the Powergui dialog. In the **Tools** menu, select **Impedance vs Frequency Measurement**. A new window opens, showing the list of Impedance Measurement blocks available in your circuit.

In your case, only one impedance is measured, and it is identified by ZB2 (the name of the ZB2 block) in the window. Fill in the frequency range by entering `0:2:1500` (zero to 1500 Hz by steps of 2 Hz). Select the logarithmic scale to display Z magnitude. Select the **Save data when updated** check box and enter `ZData` as the variable name to contain the impedance vs. frequency. Click the **Update** button.



When the calculation is finished, the window displays the magnitude and phase as functions of frequency. The magnitude should be identical to the plot (for one line section) shown in Impedance at Node B2 as Function of Frequency on page 1-24. If you look in your workspace, you should have a variable named `ZData`. It is a two-column matrix containing frequency in column 1 and complex impedance in column 2.

Specifying Initial Conditions

In this section...
“Introduction” on page 1-27
“State Variables” on page 1-27
“Initial States” on page 1-28
“Specify Initial Electrical States with Powergui” on page 1-29

Introduction

In this section you

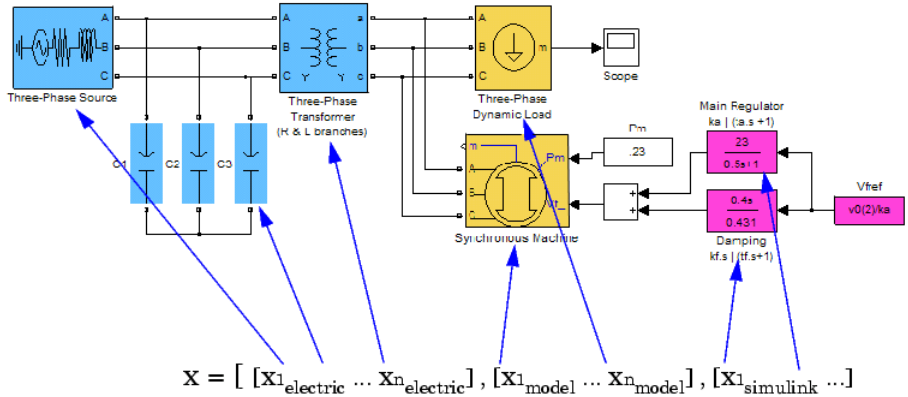
- Learn what are the state variables of a Simulink® diagram containing SimPowerSystems™ blocks
- Specify initial conditions for the electrical state variables

State Variables

The state variables of a Simulink diagram containing SimPowerSystems blocks consist of

- The electrical states associated to RLC branch-type SimPowerSystems blocks. They are defined by the state-space representation of your model. See “Electrical State Variables” on page 1-18 for more details about the electrical states.
- The Simulink states of the SimPowerSystems electrical models such as the Synchronous Machine block, the Saturable Transformer block, or the Three-Phase Dynamic Load block.
- The Simulink states of the other Simulink blocks of your model (controls, user-defined blocks, and other blocksets).

The following picture provides an example that contains the three types of state variables:



Initial States

Initial conditions, which are applied to the entire system at the start of the simulation, are generally set in the blocks. Most of the Simulink blocks allow you to specify initial conditions. For the case of the electrical states, the SimPowerSystems software automatically sets the initial values of the electrical states to start the simulation in steady state.

However, you can specify the initial conditions for the capacitor voltage and inductor currents in the mask of these blocks:

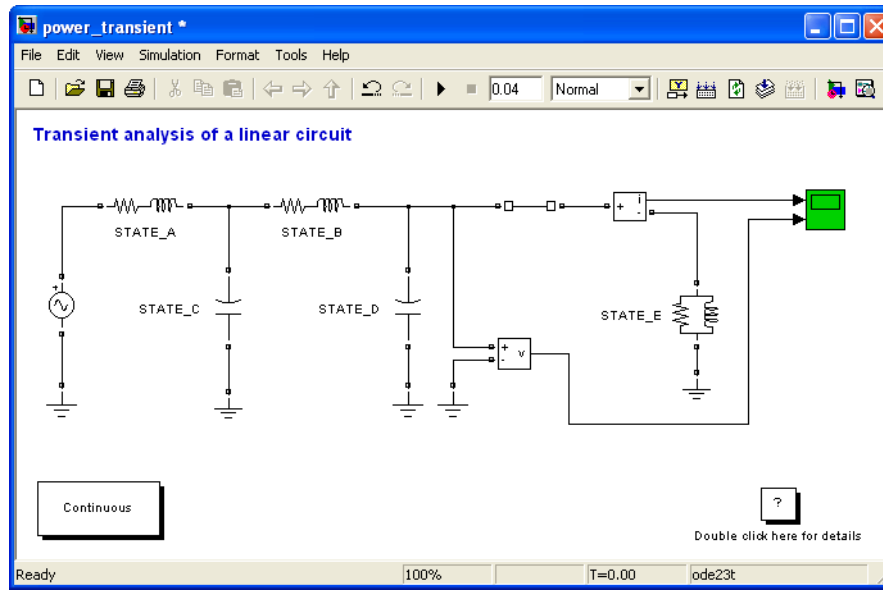
- the Series and Parallel RLC Branch blocks
- the Series and Parallel RLC Load blocks

The initial values entered in the mask of these block will overwrite the default steady-state parameters calculated by the SimPowerSystems software. In the same sense, you can overwrite initial conditions of the overall blocks by specifying them in the **States** area of the **Simulation Parameters** pane.

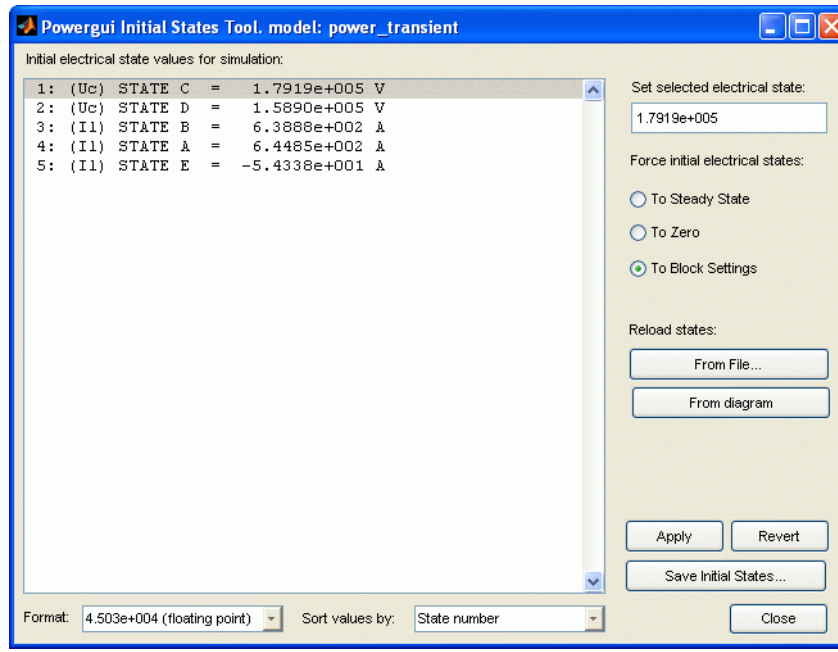
See the `power_init` function reference page for more details on how you can specify initial states for a Simulink diagram with SimPowerSystems blocks.

Specify Initial Electrical States with Powergui

1 Open the SimPowerSystems demo entitled Transient Analysis of a Linear Circuit by typing `power_transient` at the command line. Rename the RLC Branch blocks as shown in the next figure.



2 From the **Analysis tools** menu of the Powergui block, select **Initial State Settings**. The initial values of the five electrical state variables (three inductor currents and two capacitor voltages) are displayed. These initial values corresponds to the values that the software automatically sets to start the simulation in steady state.



3 Open the Scope block and start the simulation. As the electrical state variables are automatically initialized, the system starts in steady state and sinusoidal waveforms are observed.

4 The initial value for `STATE_D` state is set to 1.589e5 V. It corresponds to the initial capacitor voltage found in the `STATE_D` block. Open this block, select the **Set the initial capacitor voltage** parameter, then specify a capacitor initial voltage of -2e5 V. Click the **OK** button.

5 Click the **From diagram** button of the Powergui Initial States Tool to refresh the display of initial states. The initial state of `STATE_D` block is now set to -2e5 V.

6 Start the simulation. In the second trace of the Scope block, zoom around the transient at the beginning of the simulation. As expected, the model does not start in steady state, but the initial value for the capacitor voltage measured by the Voltage Measurement block is -2e5 V.

7 Select the STATE_A state variable in the **Initial States Tool** list. In the **Set selected electrical state** field, set the initial inductor current to 50 A. Open the mask of the STATE_A block, and note that the **Set the initial inductor current** parameter is selected and the initial inductor current is set to 50 A.

Run the simulation and observe the new transient caused by this new setting.

Using the Force Initial State to Zero Check Box

Now suppose that you want to reset all the initial electrical states to zero without loosing the settings you have done in the previous steps.

1 From the Initial State Tool window, select the **To zero** check box under **Force initial electrical state**, then click **Apply**. Restart the simulation and observe the transient when all the initial conditions starts from zero.

2 Open the masks of the STATE_C and STATE_A blocks and note that even if initial conditions are still specified in these blocks, the setting for the initial states is forced to zero by the Powergui block.

A message is displayed at the command line to remind you every time you start the simulation that the electrical initial states of your model are forced to zero by the Powergui block, which overwrites the block settings in your model.

Using the Force Initial State to Steady State Check Box

Similarly, you can set all the initial states to steady without loosing the settings you have done in the previous steps.

1 From the Initial State Tool window, select the **To steady state** check box under **Force initial electrical state**, then click **Apply**.

2 Restart the simulation and observe that the simulation starts in steady state.

A message is displayed at the command line to remind you every time you start the simulation that the electrical initial states of your model are forced to steady state by the Powergui block.

Using the Force Initial State to Block Settings Check Box

To return to the block settings, select the **To block settings** check box under **Force initial electrical state**, then click **Apply**.

Simulating Transients

In this section...

"Introduction" on page 1-33

"Simulating Transients with a Circuit Breaker" on page 1-33

"Continuous, Variable Time Step Integration Algorithms" on page 1-35

"Discretizing the Electrical System" on page 1-37

Introduction

In this section you

- Learn how to create an electrical subsystem
- Simulate transients with a circuit breaker
- Compare time domain simulation results with different line models
- Learn how to discretize a circuit and compare results thus obtained with results from a continuous, variable time step algorithm

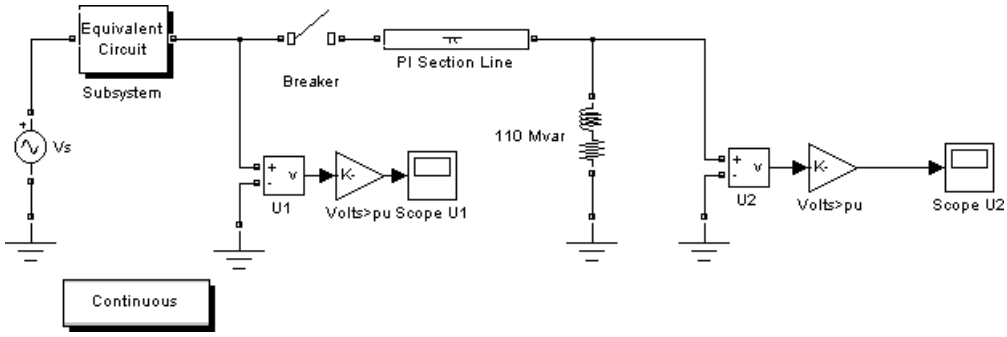
Simulating Transients with a Circuit Breaker

One of the main uses of SimPowerSystems™ software is to simulate transients in electrical circuits. This can be done with either mechanical switches (circuit breakers) or switches using power electronic devices.

First open your `circuit1` system and delete the current source connected at node B2. Save this new system as `circuit2`. Before connecting a circuit breaker, you need to modify the schematic diagram of `circuit2`. You can group several components into a subsystem. This feature is useful to simplify complex schematic diagrams.

Use this feature to transform the source impedance into a subsystem:

1 Select the two blocks identified as Rs_eq and Z_eq by surrounding them by a box with the left mouse button and use the **Edit > Create subsystem** menu item. The two blocks now form a new block called Subsystem.



2 Using the **Edit > Mask subsystem** menu item, change the icon of that subsystem. In the **Icon** section of the mask editor, enter the following drawing command:

```
disp('Equivalent\nCircuit')
```

The icon now reads Equivalent Circuit, as shown in the figure above.

3 Use the **Format > Show drop shadow** menu item to add a drop shadow to the Subsystem block.

4 You can double-click the Subsystem block and look at its content.

5 Insert a circuit breaker into your circuit to simulate a line energization by opening the Elements library of **powerlib**. Copy the Breaker block into your **circuit2** window.

The circuit breaker is a nonlinear element modeled by an ideal switch in series with a resistance. Because of modeling constraints, this resistance cannot be set to zero. However, it can be set to a very small value, say 0.001 Ω, that does not affect the performance of the circuit:

1 Open the Breaker block dialog box and set its parameters as follows:

Ron	`0.001 Ω`
Initial state	`0 (open)`
Rs	`inf`

Cs	`0`
Switching times	`[(1/60)/4]`

2. Insert the circuit breaker in series with the sending end of the line, then rearrange the circuit as shown in the previous figure.

3. Open the scope U2 and click the **Parameters** icon and select the **Data history** tab. Click the **Save data to workspace** button and specify a variable name `U2` to save the simulation results; then change the **Format** option for U2 to be **Array**. Also, clear **Limit rows to last** to display the entire waveform for long simulation times.

You are now ready to simulate your system.

Continuous, Variable Time Step Integration Algorithms

Open the **PI section Line** dialog box and make sure the number of sections is set to 1. Open the **Simulation > Simulation parameters** dialog box. As you now have a system containing switches, you need a stiff integration algorithm to simulate the circuit. In the **Solver** pane, select the variable-step stiff integration algorithm `ode23t`.

Keep the default parameters (relative tolerance set at `1e-3`) and set the stop time to `0.02` seconds. Open the scopes and start the simulation. Look at the waveforms of the sending and receiving end voltages on ScopeU1 and ScopeU2.

Once the simulation is complete, copy the variable `U2` into `U2_1` by entering the following command in the MATLAB® Command Window.

```
U2_1 = U2;
```

These two variables now contain the waveform obtained with a single PI section line model.

Open the **PI section Line** dialog box and change the number of sections from `1` to `10`. Start the simulation. Once the simulation is complete, copy the variable `U2` into `U2_10`.

Before modifying your circuit to use a distributed parameter line model, save your system as `circuit2_10pi`, which you can reuse later.

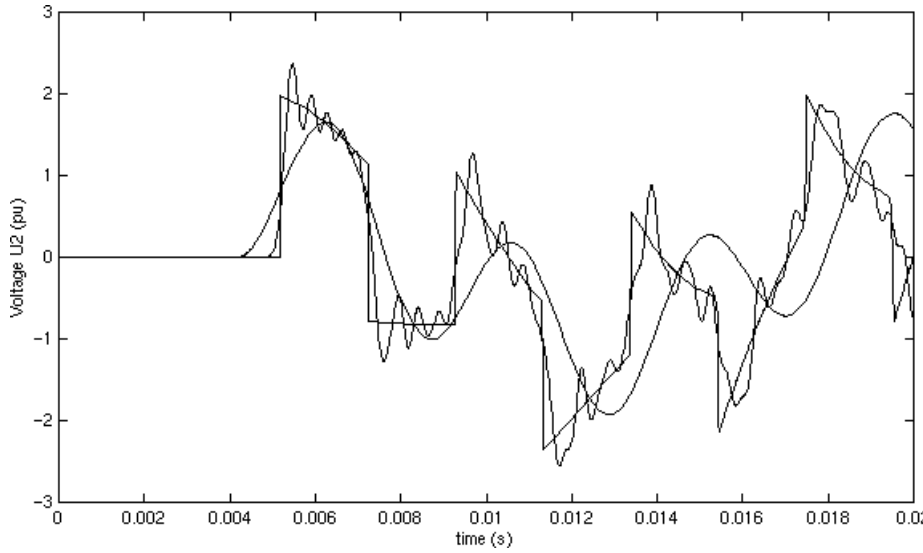
Delete the PI section line model and replace it with a single-phase Distributed Parameter Line block. Set the number of phases to `1` and use the same R, L, C, and length parameters as for the PI section line (see Circuit to Be Modeled on page 1-8). Save this system as `circuit2_dist`.

Restart the simulation and save the `U2` voltage in the `U2_d` variable.

You can now compare the three waveforms obtained with the three line models. Each variable `U2_1`, `U2_10`, and `U2_d` is a two-column matrix where the time is in column 1 and the voltage is in column 2. Plot the three waveforms on the same graph by entering the following command.

```
plot(U2_1(:,1), U2_1(:,2), U2_10(:,1),U2_10(:,2),
U2_d(:,1),U2_d(:,2));
```

These waveforms are shown in the next figure. As expected from the frequency analysis performed during “Analyzing a Simple Circuit” on page 1-18, the single PI model does not respond to frequencies higher than 229 Hz. The 10 PI section model gives a better accuracy, although high-frequency oscillations are introduced by the discretization of the line. You can clearly see in the figure the propagation time delay of 1.03 ms associated with the distributed parameter line.



Receiving End Voltage Obtained with Three Different Line Models

Discretizing the Electrical System

An important product feature is its ability to simulate either with continuous, variable step integration algorithms or with discrete solvers. For small systems, variable time step algorithms are usually faster than fixed step methods, because the number of integration steps is lower. For large systems that contain many states or many nonlinear blocks such as power electronic switches, however, it is advantageous to discretize the electrical system.

When you discretize your system, the precision of the simulation is controlled by the time step. If you use too large a time step, the precision might not be sufficient. The only way to know if it is acceptable is to repeat the simulation with different time steps and find a compromise for the largest acceptable time step. Usually time steps of 20 µs to 50 µs give good results for simulation of switching transients on 50 Hz or 60 Hz power systems or on systems using line-commutated power electronic devices such as diodes and thyristors. You must reduce the time step for systems using forced-commutated power electronic switches. These devices, the insulated-gate bipolar transistor (IGBT), the field-effect transistor (FET), and the gate-turnoff thyristor (GTO) are operating at high switching frequencies.

For example, simulating a pulse-width-modulated (PWM) inverter operating at 8 kHz would require a time step of at most 1 µs.

You now learn how to discretize your system and compare simulation results obtained with continuous and discrete systems. Open the `circuit2_10pi` system that you saved from a previous simulation. This system contains 24 electrical states and one switch. Open the Powergui and select **Discretize electrical model**. Set the sample time to `25e-6 s`. When you restart the simulation, the power system is discretized using the Tustin method (corresponding to trapezoidal integration) using a 25 µs sample time.

Open the **Simulation > Simulation parameters > Solver** dialog box and set the simulation time to `0.2 s`. Start the simulation.

Note Once the system is discretized, there are no more continuous states in the electrical system. So you do not need a variable-step integration method to simulate. In the **Simulation > Simulation parameters > Solver** dialog box, you could have selected the `Fixed-step` and `discrete (no continuous states)` options and specified a fixed step of 25 µs.

To measure the simulation time, you can restart the simulation by entering the following commands.

```
tic; sim(gcs); toc
```

When the simulation is finished the elapsed time in seconds is displayed in the MATLAB Command Window.

To return to the continuous simulation, open the Powergui block and select **Continuous**. If you compare the simulation times, you find that the discrete system simulates approximately 3.5 times faster than the continuous system.

To compare the precision of the two methods, perform the following three simulations:

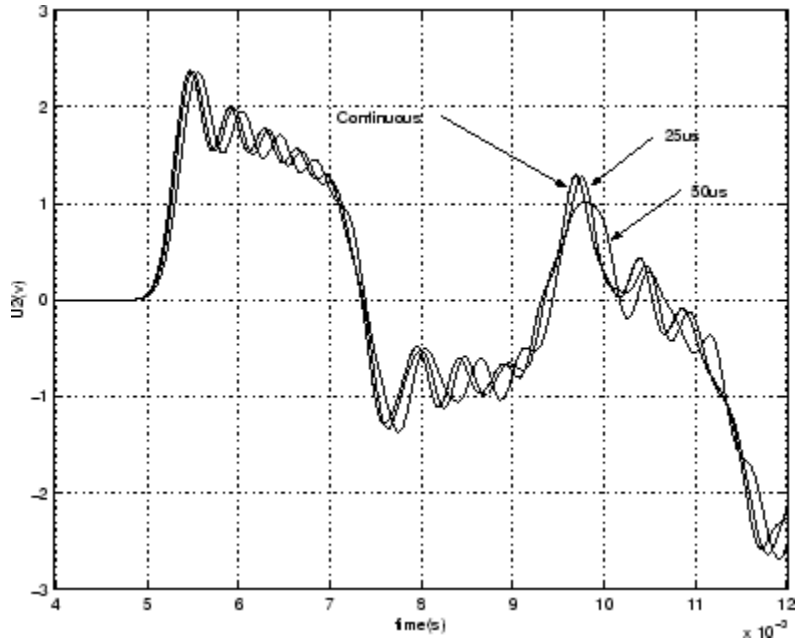
1. Simulate a continuous system, with Ts = 0.
2. Simulate a discrete system, with Ts = 25 µs.

3 Simulate a discrete system, with Ts = 50 µs.

For each simulation, save the voltage U2 in a different variable. Use respectively `U2c`, `U2d25`, and `U2d50`. Plot the U2 waveforms on the same graph by entering the following command.

```
plot(U2c(:,1), U2c(:,2), U2d25(:,1),U2d25(:,2),
U2d50(:,1),U2d50(:,2))
```

Zoom in on the 4 to 12 ms region of the plot window to compare the differences on the high-frequency transients. The 25 µs compares reasonably well with the continuous simulation. However, increasing the time step to 50 µs produces appreciable errors. The 25 µs time step would therefore be acceptable for this circuit, while obtaining a gain of 3.5 on simulation speed.



Comparison of Simulation Results for Continuous and Discrete Systems

Introducing the Phasor Simulation Method

In this section...

"Introduction" on page 1-40

"When to Use the Phasor Solution" on page 1-40

"Phasor Simulation of a Circuit Transient" on page 1-41

Introduction

In this section, you

- Apply the phasor simulation method to a simple linear circuit
- Learn advantages and limitations of this method

Up to now you have used two methods to simulate electrical circuits:

- Simulation with variable time steps using the continuous Simulink® solvers
- Simulation with fixed time steps using a discretized system

This section explains how to use a third simulation method, the phasor solution method.

When to Use the Phasor Solution

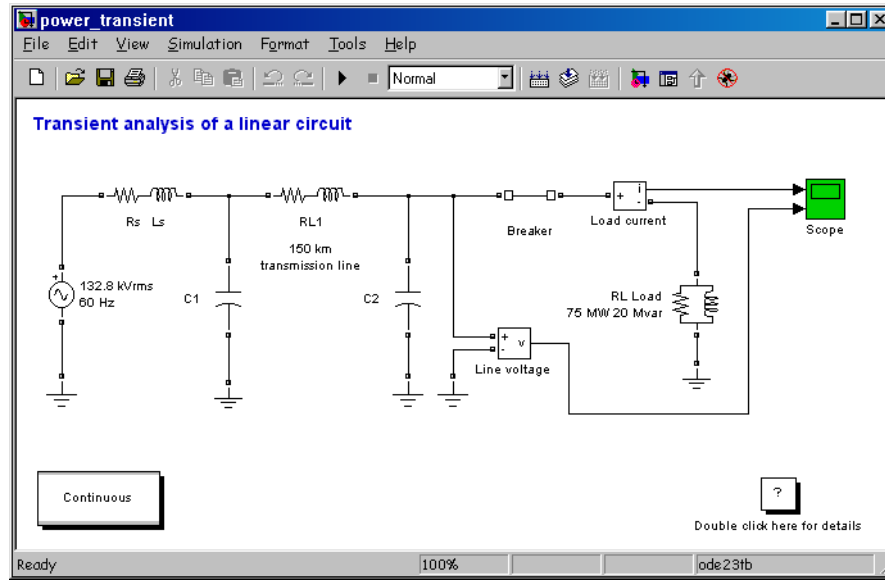
The phasor solution method is mainly used to study electromechanical oscillations of power systems consisting of large generators and motors. An example of this method is the simulation of a multimachine system in "Three-Phase Systems and Machines" on page 2-24. However, this technique is not restricted to the study of transient stability of machines. It can be applied to any linear system.

If, in a linear circuit, you are interested only in the changes in magnitude and phase of all voltages and currents when switches are closed or opened, you do not need to solve all differential equations (state-space model) resulting from the interaction of R, L, and C elements. You can instead solve a much simpler set of algebraic equations relating the voltage and current phasors. This is what the phasor solution method does. As its name implies, this method

computes voltages and currents as phasors. Phasors are complex numbers representing sinusoidal voltages and currents at a particular frequency. They can be expressed either in Cartesian coordinates (real and imaginary) or in polar coordinates (amplitude and phase). As the electrical states are ignored, the phasor solution method does not require a particular solver to solve the electrical part of your system. The simulation is therefore much faster to execute. You must keep in mind, however, that this faster solution technique gives the solution only at one particular frequency.

Phasor Simulation of a Circuit Transient

You now apply the phasor solution method to a simple linear circuit. Open the Demos library of **powerlib.** Open the General Demos library and select the demo named "Transient Analysis." A system named `power_transient` opens.



Simple Linear Circuit

This circuit is a simplified model of a 60 Hz, 230 kV three-phase power system where only one phase is represented. The equivalent source is modeled by a voltage source (230 kV RMS / sqrt(3) or 132.8 kV RMS, 60 Hz) in series with its internal impedance (Rs Ls). The source feeds an RL load through a 150 km transmission line modeled by a single PI section (RL1 branch and two shunt

capacitances, C1 and C2). A circuit breaker is used to switch the load (75 MW, 20 Mvar) at the receiving end of the transmission line. Two measurement blocks are used to monitor the load voltage and current.

The Powergui block at the lower-left corner indicates that the model is continuous. Start the simulation and observe transients in voltage and current waveforms when the load is first switched off at t = 0.0333 s (2 cycles) and switched on again at t = 0.1167 s (7 cycles).

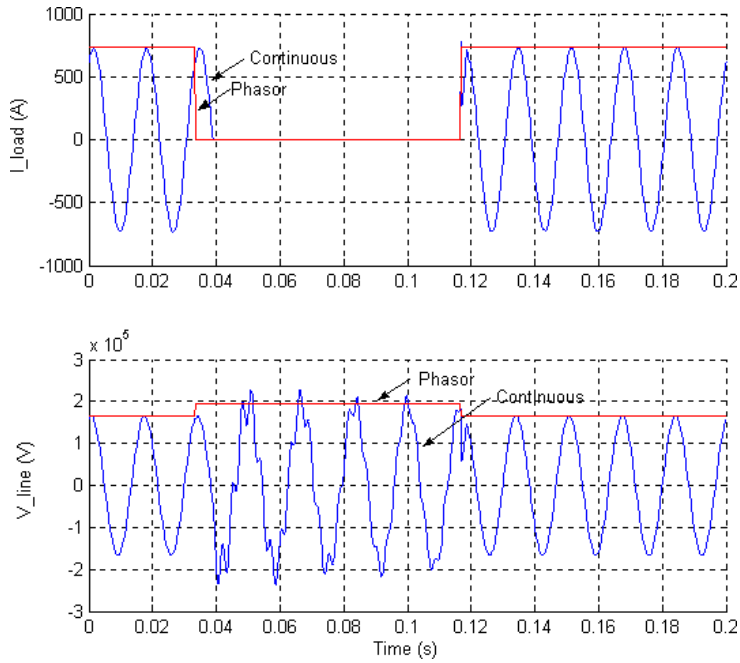
Invoking the Phasor Solution in the Powergui Block

You now simulate the same circuit using the phasor simulation method. This option is accessible through the Powergui block. Open this block and select **Phasor simulation.** You must also specify the frequency used to solve the algebraic network equations. A default value of 60 Hz should already be entered in the **Frequency** menu. Close the Powergui and notice that the word Phasors now appears on the Powergui icon, indicating that the Powergui now applies this method to simulate your circuit. Before restarting the simulation, you need to specify the appropriate format for the two signals sent to the Scope block.

Selecting Phasor Signal Measurement Formats

If you now double-click the Voltage Measurement block or the Current Measurement block, you see that a menu allows you to output phasor signals in four different formats: `Complex` (default choice), `Real-Imag`, `Magnitude-Angle`, or just `Magnitude`. The `Complex` format is useful when you want to process complex signals. Note that the oscilloscope does not accept complex signals. Select `Magnitude` format for both the Line Voltage and the Load Current Measurement blocks. This will allow you to observe the magnitude of the voltage and current phasors.

Restart the simulation. The magnitudes of the 60 Hz voltage and current are now displayed on the scope. Waveforms obtained from the continuous simulation and the phasor simulation are superimposed in this plot.



Waveforms Obtained with the Continuous and Phasor Simulation Methods

Note that with continuous simulation, the opening of the circuit breaker occurs at the next zero crossing of current following the opening order; whereas for the phasor simulation, this opening is instantaneous. This is because there is no concept of zero crossing in the phasor simulation.

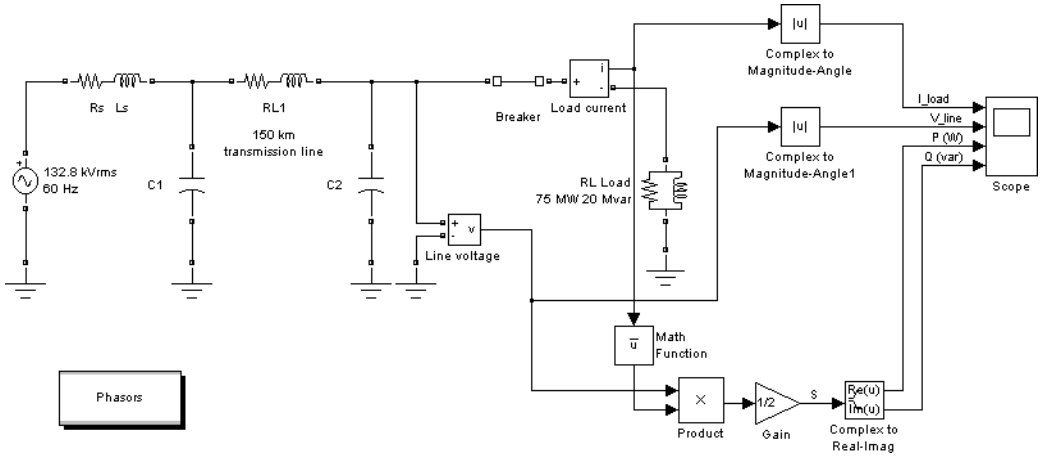
Processing Voltage and Current Phasors

The `Complex` format allows the use of complex operations and processing of phasors without separating real and imaginary parts. Suppose, for example, that you need to compute the power consumption of the load (active power *P* and reactive power *Q*). The complex power *S* is obtained from the voltage and current phasors as

$$\overline{S} = P + jQ = \frac{1}{2} \cdot \overline{V} \cdot \overline{I}^{*}$$

where I^* is the conjugate of the current phasor. The 1/2 factor is required to convert magnitudes of voltage and current from peak values to RMS values.

Select the `Complex` format for both current and voltage and, using blocks from the Simulink Math library, implement the power measurement as shown.



Power Computation Using Complex Voltage and Current

The Complex to Magnitude-Angle blocks are now required to convert complex phasors to magnitudes before sending them to the scope.

The power computation system you just implemented is already built into the SimPowerSystems™ software. The Active & Reactive Power (Phasor Type) block is available in the Extras/Phasor library.

2

Advanced Components and Techniques

This chapter introduces methods and devices that enhance your power system simulations and make them more realistic.

The first two tutorials illustrate power electronics, simple motors, and Fourier analysis. The third tutorial demonstrates three-phase power systems, electrical machinery, load flow, and use of the phasor solution method for transient stability studies of electromechanical systems. The fourth explains how you can create and customize your own nonlinear blocks.

Introducing Power Electronics (p. 2-2)	Use power electronics and transformers and vary circuit initial conditions
Simulating Variable Speed Motor Control (p. 2-10)	Model and discretize simple motors with specialized blocks. Use the FFT Analysis tool of the Powergui block to perform harmonic analysis
Three-Phase Systems and Machines (p. 2-24)	Use electrical machines and three-phase components. Apply the phasor solution method to study of electromechanical oscillations of power systems
Building and Customizing Nonlinear Models (p. 2-39)	Model nonlinear systems and create your own blocks to represent them
Building a Model Using Model Construction Commands (p. 2-56)	Use model construction commands to add blocks to your models and connect them.

Introducing Power Electronics

In this section...

"Introduction" on page 2-2

"Simulation of the TCR Branch" on page 2-4

"Simulation of the TSC Branch" on page 2-7

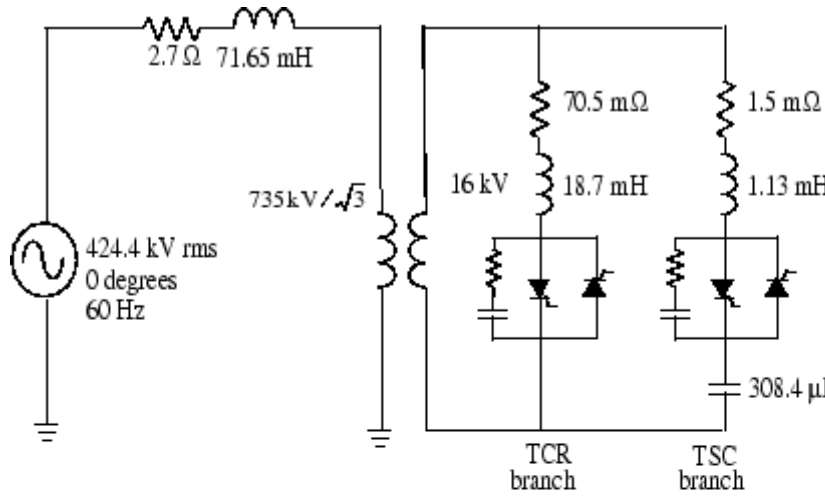
Introduction

In this section you

- Learn how to use power electronics components
- Learn how to use transformers
- Change initial conditions of a circuit

SimPowerSystems™ software is designed to simulate power electronic devices. This section uses a simple circuit based on thyristors as the main example.

Consider the circuit shown below. It represents one phase of a static var compensator (SVC) used on a 735 kV transmission network. On the secondary of the 735 kV/16 kV transformer, two variable susceptance branches are connected in parallel: one thyristor-controlled reactor (TCR) branch and one thyristor-switched capacitor (TSC) branch.



Transformer parameters:

Nominal power 110 MVA

Primary: Rated voltage 424.4 kV
RMS leakage reactance = 0.15 pu
resistance = 0.002 pu

Secondary: Rated voltage 16 kV
RMS leakage reactance = 0 pu
resistance = 0.002 pu

Magnetizing current at 1 pu
voltage: Inductive: 0.2%
Resistive: 0.2%

Thyristor parameters:

Ron = 1 mΩ; Vf = 14*0.8 V

(14 thyristors in series)

Snubber: Rs = 500Ω; Cs = 0.15 µF

One Phase of a TCR/TSC Static Var Compensator

The TCR and TSC branches are both controlled by a valve consisting of two thyristor strings connected in antiparallel. An RC snubber circuit is connected across each valve. The TSC branch is switched on/off, thus providing discrete step variation of the SVC capacitive current. The TCR branch is phase controlled to obtain a continuous variation of the net SVC reactive current.

Now build two circuits illustrating the operation of the TCR and the TSC branches.

Simulation of the TCR Branch

1 Open a new window and save it as `circuit3`.

2 Open the Power Electronics library and copy the Thyristor block into your `circuit3` model.

3 Open the **Thyristor** menu and set the parameters as follows:

Ron	`1e-3`
Lon	`0`
Vf	`14*0.8`
Rs	`500`
Cs	`0.15e-6`

Notice that the snubber circuit is integral to the **Thyristor** dialog box.

4 Rename this block Th1 and duplicate it.

5 Connect this new thyristor Th2 in antiparallel with Th1, as shown in Simulation of the TCR Branch on page 2-5.

As the snubber circuit has already been specified with *Th1*, the snubber of *Th2* must be eliminated.

6 Open the **Th2** dialog box and set the snubber parameters to

Rs	`Inf`
Cs	`0`

Notice that the snubber disappears on the *Th2* icon.

The linear transformer is located in the Elements library. Copy it, rename it to TrA, and open its dialog box. Set its nominal power, frequency, and winding parameters (**winding 1** = `primary`; **winding 2** = `secondary`) as shown in One Phase of a TCR/TSC Static Var Compensator on page 2-3.

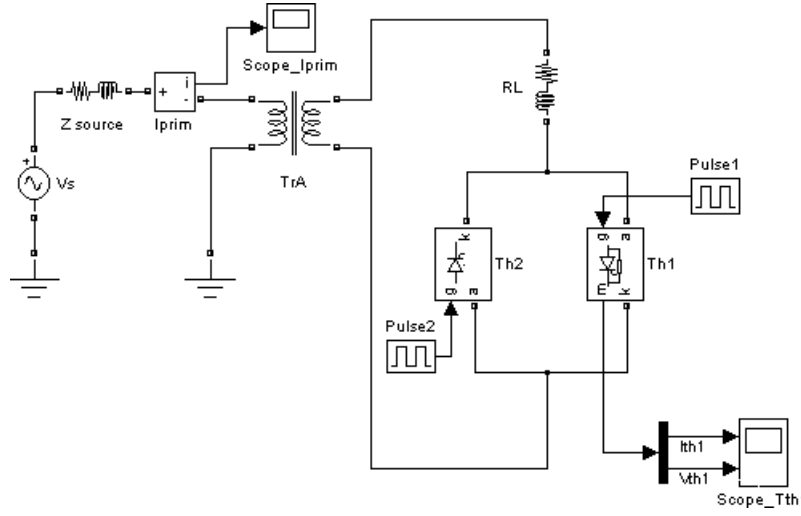
The Units parameter allows you to specify the resistance R and leakage inductance L of each winding as well as the magnetizing branch Rm/Lm,

either in SI units (ohms, henries) or in per units (pu). Keep the default pu setting to specify directly R and L in per unit quantities. As there is no tertiary winding, deselect **Three windings transformer**. Winding 3 disappears on the *TrA* block.

Finally, set the magnetizing branch parameters **Rm** and **Xm** at `[500, 500]`. These values correspond to 0.2% resistive and inductive currents.

Add a voltage source, series RL elements, and a Ground block. Set the parameters as shown in One Phase of a TCR/TSC Static Var Compensator on page 2-3. Add a current measurement to measure the primary current. Interconnect the circuit as shown in Simulation of the TCR Branch on page 2-5.

Notice that the Thyristor blocks have an output identified by the letter `m`. This output returns a Simulink® vectorized signal containing the thyristor current (Iak) and voltage (Vak). Connect a Demux block with two outputs at the m output of *Th1*. Then connect the two demultiplexer outputs to a dual trace scope that you rename Scope_Th1. (To create a second input to your scope, in the **Scope properties > General** menu item, set the number of axes to `2`.) Label the two connection lines `Ith1` and `Vth1`. These labels are automatically displayed on the top of each trace.



Simulation of the TCR Branch

You can now model the synchronized pulse generators firing thyristors *Th1* and *Th2*. Copy two Simulink pulse generators into your system, name them Pulse1 and Pulse2, and connect them to the gates of *Th1 and Th2*.

Now you have to define the timing of the Th1 and Th2 pulses. At every cycle a pulse has to be sent to each thyristor α degrees after the zero crossing of the thyristor commutation voltage. Set the *Pulse1* and *Pulse2* parameters as follows:

Amplitude	`1`
Period	`1/60 s`
Pulse width (% of period)	`1% (3.6 degrees pulses)`
Phase Delay	`1/60+T for Pulse1` `1/60+1/120+T for Pulse2`

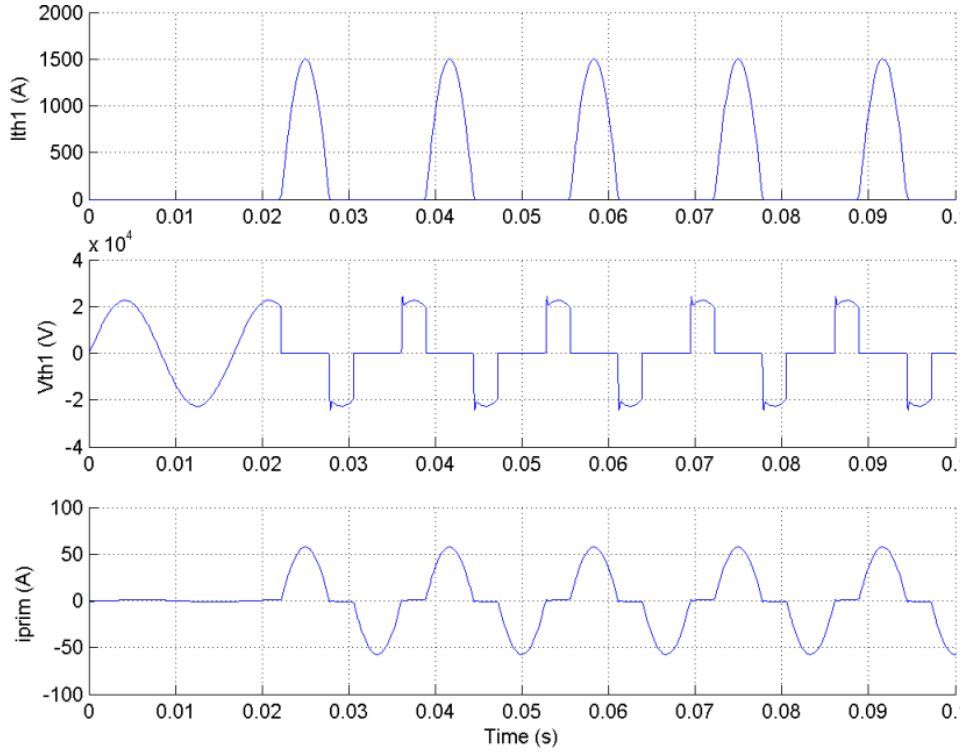
The pulses sent to Th2 are delayed by 180 degrees with respect to pulses sent to Th1. The delay T is used to specify the firing angle α. To get a 120 degree firing angle, specify `T` in the workspace by entering

```
T = 1/60/3;
```

Now open the **Simulation > Simulation parameters** dialog box. Select the `ode23t` integration algorithm. Keep the default parameters but set the relative tolerance to `1e-4` and the stop time to `0.1`.

Add a Powergui block at the top level of your model, then start the simulation. The results are shown in TCR Simulation Results on page 2-7.

Note You could also choose to discretize your system. Try, for example, a sample time of 50 µs. The simulation results should compare well with the continuous system.



TCR Simulation Results

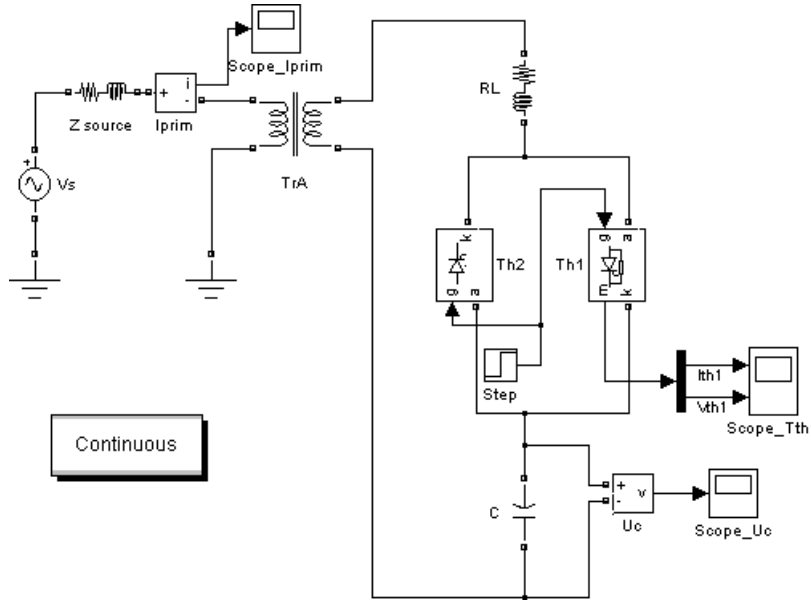
Simulation of the TSC Branch

You can now modify your `circuit3` system and change the TCR branch to a TSC branch. Save `circuit3` as a new system and name it `circuit4`.

Connect a capacitor in series with the RL inductor and Th1/Th2 valve as shown in the figure below. Change the **R**, **L**, and **C** parameters as shown in One Phase of a TCR/TSC Static Var Compensator on page 2-3. Connect a voltmeter and scope to monitor the voltage across the capacitor.

Contrary to the TCR branch, which was fired by a synchronous pulse generator, a continuous firing signal is now applied to the two thyristors. Delete the two pulse generators. Copy a Step block from the Simulink library and connect its output at both gates of Th1 and Th2. Set its step time at

1/60/4 (energizing at the first positive peak of the source voltage). Your circuit should now be similar to the one shown here.



Simulation of the TSC Branch

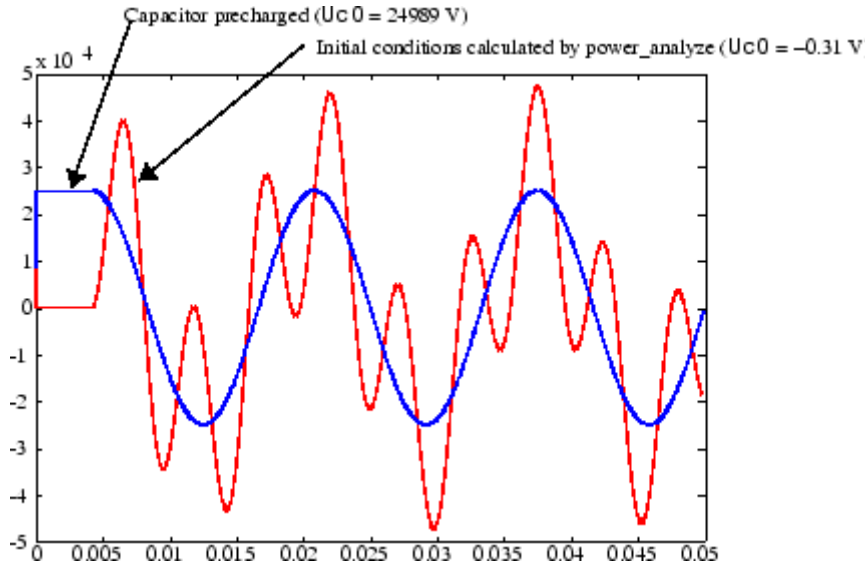
Open the three scopes and start the simulation.

As the capacitor is energized from zero, you can observe a low damping transient at 200 Hz, superimposed with the 60 Hz component in the capacitor voltage and primary current. During normal TSC operation, the capacitor has an initial voltage left since the last valve opening. To minimize the closing transient with a charged capacitor, the thyristors of the TSC branch must be fired when the source voltage is at maximum value and with the correct polarity. The initial capacitor voltage corresponds to the steady-state voltage obtained when the thyristor switch is closed. The capacitor voltage is 17.67 kVrms when the valve is conducting. At the closing time, the capacitor must be charged at the peak voltage.

$$Uc = 17670 \times \sqrt{2} = 24989\ V$$

You can now use the Powergui block to change the capacitor initial voltage. Open the Powergui and select **Initial States Setting**. A list of all the state variables with their default initial values appears. The value of the initial voltage across the capacitor C (variable `Uc_C`) should be -0.3141 V. This voltage is not exactly zero because the snubber allows circulation of a small current when both thyristors are blocked. Now select the `Uc_C` state variable and enter `24989` in the upper right field. Then click the **Apply** button to make this change effective.

Start the simulation. As expected the transient component of capacitor voltage and current has disappeared. The voltages obtained with and without initial voltage are compared in this plot.



Transient Capacitor Voltage With and Without Initial Charge

Simulating Variable Speed Motor Control

In this section...

"Introduction" on page 2-10

"Building and Simulating the PWM Motor Drive" on page 2-12

"Using the Multimeter Block" on page 2-18

"Discretizing the PWM Motor Drive" on page 2-20

"Performing Harmonic Analysis Using the FFT Tool" on page 2-20

Introduction

In this section you

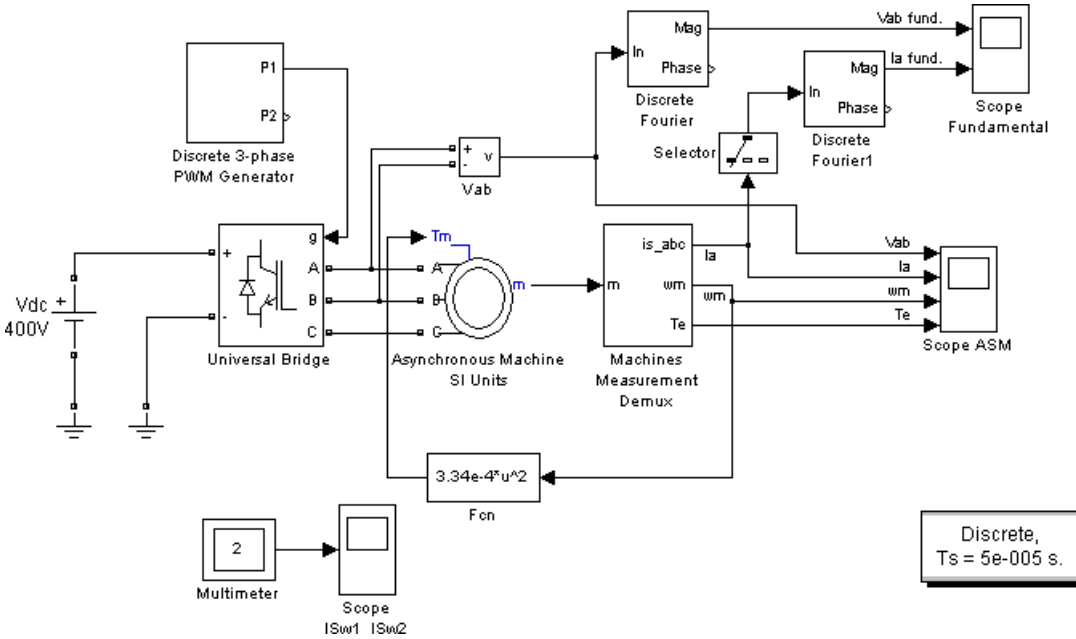
- Use electrical machines and power electronics to simulate a simple AC motor drive with variable speed control
- Learn how to use the Universal Bridge block
- Discretize your model and compare variable-step and fixed-step simulation methods
- Learn how to use the Multimeter block
- Learn how to use the FFT tool

Variable speed control of AC electrical machines makes use of forced-commutated electronic switches such as IGBTs, MOSFETs, and GTOs. Asynchronous machines fed by *pulse width modulation* (PWM) voltage sourced converters (VSC) are nowadays gradually replacing the DC motors and thyristor bridges. With PWM, combined with modern control techniques such as field-oriented control or direct torque control, you can obtain the same flexibility in speed and torque control as with DC machines. This section shows how to build a simple open loop AC drive controlling an asynchronous machine. Chapter 4 will introduce you to a specialized library containing 13 models of DC and AC drives. These "ready to use" models will enable you to simulate electric drive systems without the need to build those complex systems yourself.

The Machines library contains four of the most commonly used three-phase machines: simplified and complete synchronous machines, asynchronous machine, and permanent magnet synchronous machine. Each machine can be used either in generator or motor mode. Combined with linear and nonlinear elements such as transformers, lines, loads, breakers, etc., they can be used to simulate electromechanical transients in an electrical network. They can also be combined with power electronic devices to simulate drives.

The Power Electronics library contains blocks allowing you to simulate diodes, thyristors, GTO thyristors, MOSFETs, and IGBT devices. You could interconnect several blocks together to build a three-phase bridge. For example, an IGBT inverter bridge would require six IGBTs and six antiparallel diodes.

To facilitate implementation of bridges, the Universal Bridge block automatically performs these interconnections for you.



Circuit 5: PWM Control of an Induction Motor

Building and Simulating the PWM Motor Drive

Follow these steps to build a PWM-controlled motor.

Assembling and Configuring the Motor Blocks

In the first steps, you copy and set up the motor blocks:

1. Open a new window and save it as `circuit5`.
2. Open the Power Electronics library and copy the Universal Bridge block into your `circuit5` model.
3. Open the Universal Bridge dialog box and set its parameters as follows:

Power electronic device		`IGBT/Diodes`
Snubber		
	Rs	`1e5 Ω`
	Cs	`inf`
	Ron	`1e-3 Ω`
Forward voltages		
	Vf	`0 V`
	Vfd	`0 V`
Tail		
	Tf	`1e-6 s`
	Tt	`1e-6 s`

Notice that the snubber circuit is integral to the Universal Bridge dialog box. As the Cs capacitor value of the snubber is set to `Inf` (short-circuit), we are using a purely resistive snubber. Generally, IGBT bridges do not use snubbers; however, because each nonlinear element in SimPowerSystems™ software is modeled as a current source, you have to provide a parallel path across each IGBT to allow connection to an inductive circuit (stator of the asynchronous machine). The high resistance value of the snubber does not affect the circuit performance.

4 Open the Machines library. Copy the Asynchronous Machine SI Units block as well as the Machine Measurement Demux block into your `circuit5` model.

5 Open the Asynchronous Machine menu and look at its parameters. Set the nominal power Pn parameter to 3*746 VA and the nominal line-to-line voltage Vn to 220 Vrms to implement a 3 HP, 60 Hz machine with two pairs of poles. Its nominal speed is therefore slightly lower than the synchronous speed of 1800 rpm, or w_s= 188.5 rad/s.

6 Notice that the three rotor terminals `a`, `b`, and `c` are made accessible. During normal motor operation these terminals should be short-circuited together. In the Asynchronous Machine menu change the rotor type to **Squirrel cage**. Notice that after this change the rotor connections are no longer accessible.

7 Open the Machine Measurement Demux block menu. When this block is connected to a machine measurement output, it allows you to access specific internal signals of the machine. First select the **Asynchronous** machine type. Deselect all signals except the following three signals: `is_abc` (three stator currents), `wm` (rotor speed), and `Te` (electromagnetic torque).

Loading and Driving the Motor

You now implement the torque-speed characteristic of the motor load. Assume a quadratic torque-speed characteristic (fan or pump type load). The torque T is then proportional to the square of the speed ω

$$T = k \times \omega^2$$

The nominal torque of the motor is

$$T_n = \frac{3 \times 746}{188.5} = 11.87 Nm$$

Therefore, the constant k should be

$$k = \frac{T_n}{\omega_s^2} = \frac{11.87}{188.5^2} = 3.34 \times 10^{-4}$$

1 Open the User-Defined Functions library of Simulink® and copy the Fcn block into your `circuit5` model. Open the block menu and enter the expression of torque as a function of speed: `3.34e-4*u^2`.

2 Connect the input of the Fcn block to the speed output of the Machines Measurement Demux block, labeled `wm`, and its output to the torque input of the motor, labeled `Tm`.

3 Open the Electrical Sources library and copy the DC Voltage Source block into your `circuit5` model. Open the block menu and set the voltage to 400 V.

4 Open the Measurements library and copy a Voltage Measurement block into your `circuit5` model. Change the block name to Vab.

5 Using Ground blocks from the Elements library, complete the power elements and voltage sensor interconnections as shown in Circuit 5: PWM Control of an Induction Motor on page 2-11.

Controlling the Inverter Bridge with a Pulse Generator

To control your inverter bridge, you need a pulse generator. Such a generator is available in the Extras library of **powerlib**:

1 Open the Extras/Discrete Control blocks library and copy the Discrete 3-Phase PWM Generator block into your `circuit5` model. This block can be used to generate pulses for a two-level or a three-level bridge. In addition the block generates two sets of pulses (outputs P1 and P2) that can be sent to two different three-arm bridges when the converter uses a twin bridge configuration. In this case, use it as a two-level single-bridge PWM generator. The converter operates in an open loop, and the three PWM modulating signals are generated internally. Connect the P1 output to the pulses input of the Universal Bridge block

2 Open the Discrete Three-Phase PWM Generator block dialog box and set the parameters as follows.

Type	`2 level`
Mode of operation	`Un-synchronized`
Carrier frequency	`18*60Hz (1080 Hz)`

Internal generation of modulating signals	`selected`
Modulation index m	`0.9`
Output voltage frequency	`60 Hz`
Output voltage phase	`0 degrees`
Sample time	`10e-6 s`

3 Use the **Edit > Look Under Mask** menu item of your model window to see how the PWM is implemented. This control system is made entirely with Simulink blocks. The block has been discretized so that the pulses change at multiples of the specified time step. A time step of 10 µs corresponds to +/- 0.54% of the switching period at 1080 Hz.

One common method of generating the PWM pulses uses comparison of the output voltage to synthesize (60 Hz in this case) with a triangular wave at the switching frequency (1080 Hz in this case). This is the method that is implemented in the Discrete 3-Phase PWM Generator block. The line-to-line RMS output voltage is a function of the DC input voltage and of the modulation index *m* as given by the following equation:

$$V_{LLrms} = \frac{m}{2} \times \frac{\sqrt{3}}{\sqrt{2}} Vdc = m \times 0.612 \times VDC$$

Therefore, a DC voltage of 400 V and a modulation factor of 0.90 yield the 220 Vrms output line-to-line voltage, which is the nominal voltage of the asynchronous motor.

Displaying Signals and Measuring Fundamental Voltage and Current

1 You now add blocks measuring the fundamental component (60 Hz) embedded in the chopped Vab voltage and in the phase A current. Open the Extras/Discrete Measurements library of **powerlib** and copy the discrete Fourier block into your `circuit5` model.

Open the discrete Fourier block dialog box and check that the parameters are set as follows:

Fundamental frequency f1	`60 Hz`
Harmonic number	`1`
Initial input	`[0 0]`
Sample time	`10e-6 s`

Connect this block to the output of the Vab voltage sensor.

2 Duplicate the Discrete Fourier block. To measure the phase A current, you need to select the first element of the `is_abc` output of the ASM Measurement Demux block.

Copy a Selector block from the Signals & Systems Simulink library.

Open its menu and set **Element** to `1`. Connect the Selector output to the second Discrete Fourier block and its input to the `is_abc` output of the Machines Measurement Demux block as shown in Circuit 5: PWM Control of an Induction Motor on page 2-11.

3 Finally, add scopes to your model. Copy one Scope block into your circuit. This scope is used to display the instantaneous motor voltage, currents, speed, and electromagnetic torque. In the **Scope properties > General** menu of the scope, set the following parameters:

Number of axes	`4`
Time range	`0.05 s`
Tick labels	`bottom axis only`

Connect the four inputs and label the four connection lines as shown in TCR Simulation Results on page 2-7. When you start the simulation, these labels are displayed on top of each trace.

To allow further processing of the signals displayed on the oscilloscope, you have to store them in a variable. In the **Scope properties > Data history** menu of the scope, set the following parameters:

Limit data point to last	`deselected`
Save data to workspace	`selected`
variable name	`ASM`
Format	`Structure with time`

After simulation, the four signals displayed on the scope are available in a structure array named ASM.

4 Duplicate the four-input Scope and change its number of inputs to 2. This scope is used to display the fundamental component of Vab voltage and Ia current. Connect the two inputs to the outputs of the Fourier blocks. Label the two connection lines as shown in TCR Simulation Results on page 2-7.

You are now ready to simulate the motor starting.

Simulating the PWM Motor Drive with Continuous Integration Algorithm

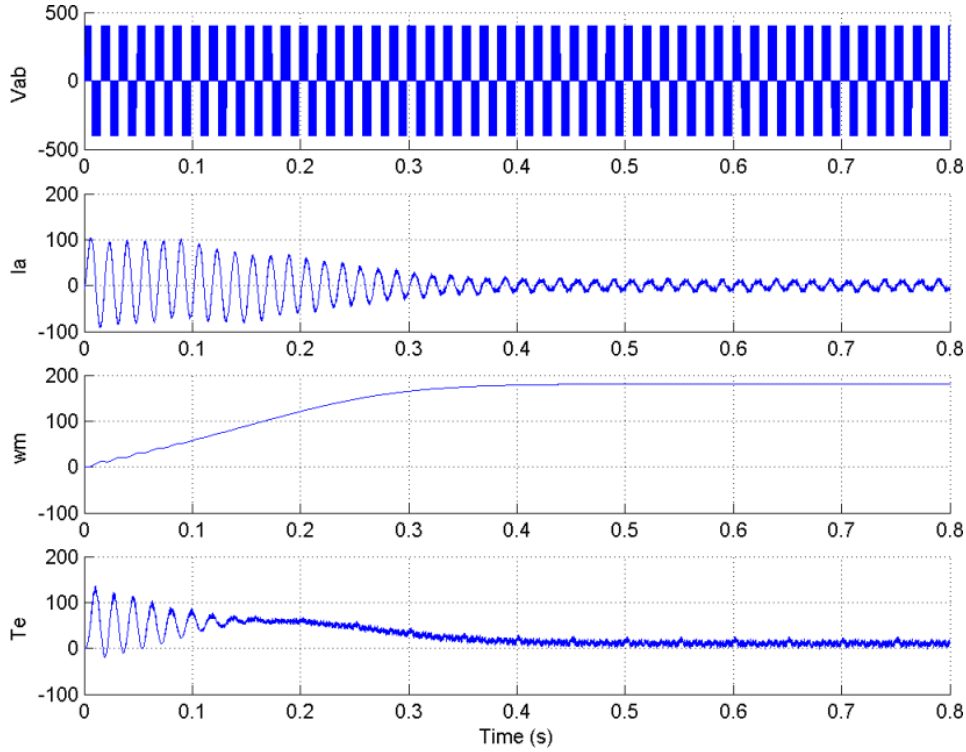
Open the **Simulation —> Simulation parameters** menu. Select the `ode23t` integration algorithm. Set the relative tolerance to `1e-4`, the absolute tolerance and the Max step size to `auto`, and the stop time to `1 s`. Start the simulation. The simulation results are shown in PWM Motor Drive; Simulation Results for Motor Starting at Full Voltage on page 2-18.

The motor starts and reaches its steady-state speed of 181 rad/s (1728 rpm) after 0.5 s. At starting, the magnitude of the 60 Hz current reaches 90 A peak (64 A RMS) whereas its steady-state value is 10.5 A (7.4 A RMS). As expected, the magnitude of the 60 Hz voltage contained in the chopped wave stays at

$$220 \times \sqrt{2} = 311V$$

Also notice strong oscillations of the electromagnetic torque at starting. If you zoom in on the torque in steady state, you should observe a noisy signal with a mean value of 11.9 N.m, corresponding to the load torque at nominal speed.

If you zoom in on the three motor currents, you can see that all the harmonics (multiples of the 1080 Hz switching frequency) are filtered by the stator inductance, so that the 60 Hz component is dominant.



PWM Motor Drive; Simulation Results for Motor Starting at Full Voltage

Using the Multimeter Block

The Universal Bridge block is not a conventional subsystem where all the six individual switches are accessible. If you want to measure the switch voltages and currents, you must use the Multimeter block, which gives access to the bridge internal signals:

1. Open the **Universal Bridge** dialog box and set the **Measurement** parameter to `Device currents`.

2. Copy the Multimeter block from the Measurements library into your `circuit5` circuit. Double-click the Multimeter block. A window showing the six switch currents appears.

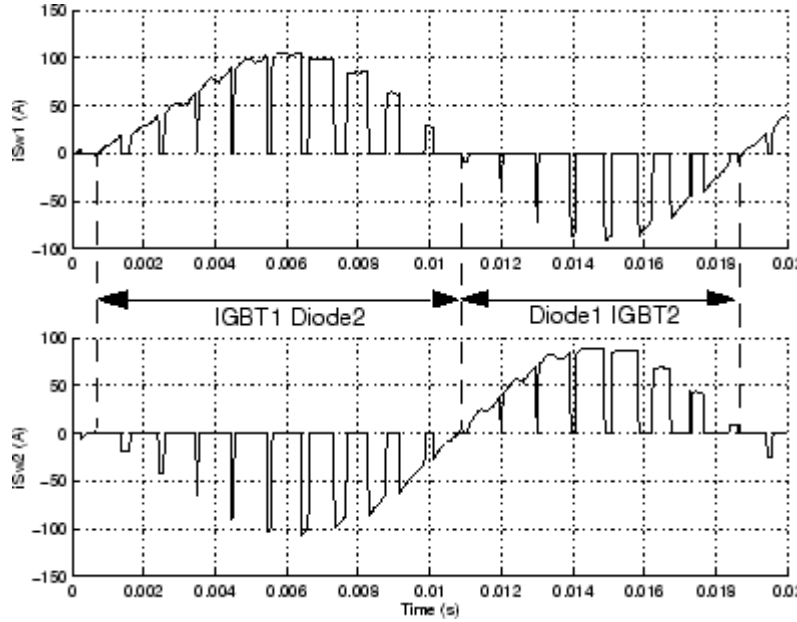
3 Select the two currents of the bridge arm connected to phase A. They are identified as

iSw1	`Universal Bridge`
iSw2	`Universal Bridge`

4 Click **OK**. The number of signals (2) is displayed in the multimeter icon.

5 Using a Demux block, send the two multimeter output signals to a two-trace scope and label the two connection lines (Trace 1: `iSw1` Trace 2: `iSw2`).

6 Restart the simulation. The waveforms obtained for the first 20 ms are shown in this plot.



Currents in IGBT/Diode Switches 1 and 2

As expected, the currents in switches 1 and 2 are complementary. A positive current indicates a current flowing in the IGBT, whereas a negative current indicates a current in the antiparallel diode.

Note Multimeter block use is not limited to the Universal Bridge block. Many blocks of the Electrical Sources and Elements libraries have a Measurement parameter where you can select voltages, currents, or saturable transformer fluxes. A judicious use of the Multimeter block reduces the number of current and voltage sensors in your circuit, making it easier to follow.

Discretizing the PWM Motor Drive

You might have noticed that the simulation using a variable-step integration algorithm is relatively long. Depending on your computer, it might take tens of seconds to simulate one second. To shorten the simulation time, you can discretize your circuit and simulate at fixed simulation time steps.

Open the Powergui and select **Discretize electrical model**. Set the **Sample Time** to `10e-6 s`. When you restart the simulation, the power system, including the asynchronous machine, is discretized at a 10 µs sample time.

As there are no more continuous states in the electrical system, you do not need a variable-step integration method to solve this system. In the **Simulation —> Simulation parameters —> Solver** dialog box pane, select the `Fixed-step` and `discrete (no continuous states)` options.

Start the simulation. Observe that the simulation is now approximately three times faster than with the continuous system. Results compare well with the continuous system.

Performing Harmonic Analysis Using the FFT Tool

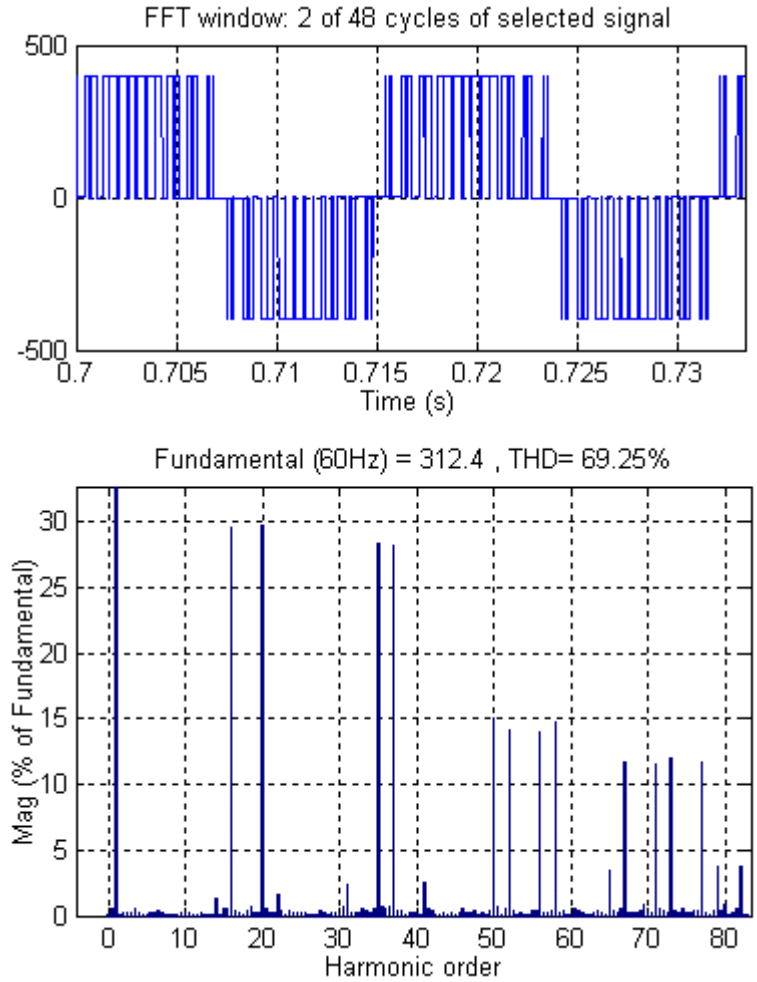
The two Discrete Fourier blocks allow computation of the fundamental component of voltage and current while simulation is running. If you would like to observe harmonic components also you would need a Discrete Fourier block for each harmonic. This approach is not convenient.

Now use the FFT tool of Powergui to display the frequency spectrum of voltage and current waveforms. These signals are stored in your workspace in the ASM structure with time variable generated by the Scope block. Because your model is discretized, the signal saved in this structure is sampled at a fixed step and consequently satisfies the FFT tool requirements.

Open the Powergui and select **FFT Analysis**. A new window opens. Set the parameters specifying the analyzed signal, the time window, and the frequency range as follows:

Structure	`ASM`
Input	`Vab`
Signal number	`1`
Start time	`0.7 s`
Number of cycles	`2`
(pull-down menu)	Display FFT window
Fundamental frequency	`60 Hz`
Max Frequency	`5000 Hz`
Frequency axis	`Harmonic order`
Display style	`Bar (relative to Fund or DC)`

The analyzed signal is displayed in the upper window. Click **Display**. The frequency spectrum is displayed in the bottom window, as shown in the next figure.



FFT Analysis of the Motor Line-to-Line Voltage

The fundamental component and *total harmonic distortion* (THD) of the Vab voltage are displayed above the spectrum window. The magnitude of the fundamental of the inverter voltage (312 V) compares well with the theoretical value (311 V for m=0.9).

Harmonics are displayed in percent of the fundamental component. As expected, harmonics occur around multiples of carrier frequency (n*18 +- k).

Highest harmonics (30%) appear at 16th harmonic (18 - 2) and 20th harmonic (18 + 2). Note that the THD value (69%) has been computed for the specified 0 to 5000 Hz frequency range. If you recompute the FFT with a maximum frequency range of 10000 Hz, you should see the THD increasing to 74% (5% contribution in THD for the 5000 to 10000 Hz frequencies).

Finally, select input Ia instead of Vab and display its current spectrum.

Three-Phase Systems and Machines

In this section...

"Introduction" on page 2-24

"Three-Phase Network with Electrical Machines" on page 2-24

"Load Flow and Machine Initialization" on page 2-27

"Using the Phasor Solution Method for Stability Studies" on page 2-35

Introduction

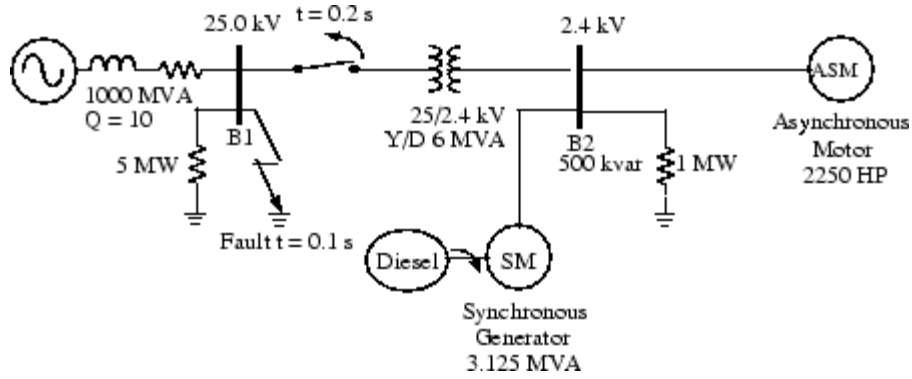
In this section you

- Learn how to simulate a three-phase power system containing electrical machines and other three-phase models
- Perform a load flow study and initialize machines to start simulation in steady state by using the **Load Flow and Machine Initialization** option of the Powergui
- Simulate the power system and observe its dynamic performance by using both the standard solution technique using a continuous solver and the **Phasor Solution** method

You now use three types of machines of the Electrical Machines library: simplified synchronous machine, detailed synchronous machine, and asynchronous machine. You interconnect these machines with linear and nonlinear elements such as transformers, loads, and breakers to study the transient stability of an uninterruptible power supply using a diesel generator.

Three-Phase Network with Electrical Machines

The two-machine system shown in this single line diagram is this section's main example:



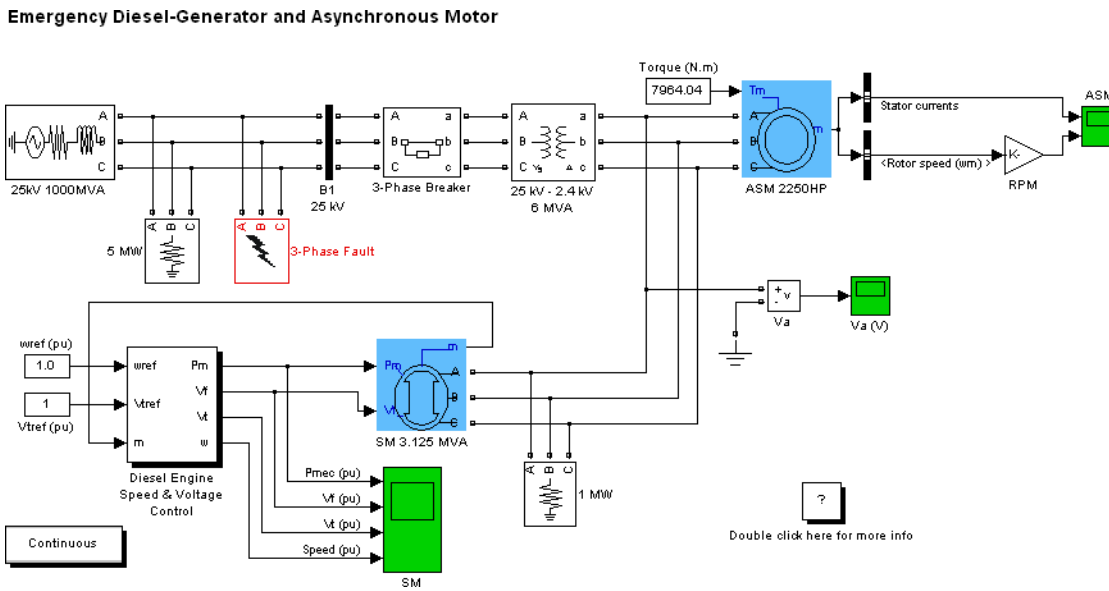
Diesel Generator and Asynchronous Motor on Distribution Network

This system consists of a plant (bus B2), simulated by a 1 MW resistive load and a motor load (ASM) fed at 2400 V from a distribution 25 kV network through a 6 MVA, 25/2.4 kV transformer, and from an emergency synchronous generator/diesel engine unit (SM).

The 25 kV network is modeled by a simple R-L equivalent source (short-circuit level 1000 MVA, quality factor X/R = 10) and a 5 MW load. The asynchronous motor is rated 2250 HP, 2.4 kV, and the synchronous machine is rated 3.125 MVA, 2.4 kV.

Initially, the motor develops a mechanical power of 2000 HP and the diesel generator is in standby, delivering no active power. The synchronous machine therefore operates as a synchronous condenser generating only the reactive power required to regulate the 2400 V bus B2 voltage at 1.0 pu. At t = 0.1 s, a three-phase to ground fault occurs on the 25 kV system, causing the opening of the 25 kV circuit breaker at t = 0.2 s, and a sudden increase of the generator loading. During the transient period following the fault and islanding of the motor-generator system, the synchronous machine excitation system and the diesel speed governor react to maintain the voltage and speed at a constant value.

This system is modeled in a SimPowerSystems™ demo. Open the Demos library of **powerlib** and double-click the demo called Three-Phase Machines and Load Flow. A system named `power_machines` opens.

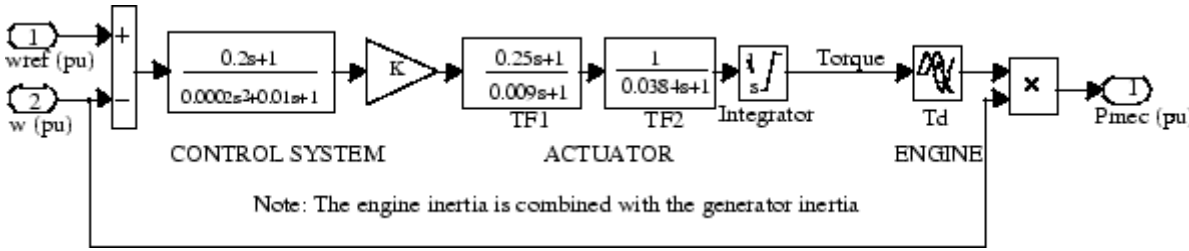


Power System of Diesel Generator and Asynchronous Motor on Distribution Network

The Synchronous Machine (SM) block uses standard parameters, whereas the Asynchronous Machine (ASM) block uses SI parameters.

The other three-phase elements such as the inductive voltage source, the Y grounded/Delta transformer, and the loads are standard blocks from the Electrical Source and Elements libraries of **powerlib**. If you open the dialog box of the Three-Phase Fault and Three-Phase Breaker blocks, you see how the switching times are specified. The Machine Measurement Demux block provided in the Machines library is used to demux the output signals of the SM and ASM machines.

The SM voltage and speed outputs are used as feedback inputs to a Simulink® control system that contains the diesel engine and governor block as well as an excitation block. The excitation system is the standard block provided in the Machines library. The SM parameters as well as the diesel engine and governor models were taken from reference [1].



Diesel Engine and Governor System

If you simulate this system for the first time, you normally do not know what the initial conditions are for the SM and ASM to start in steady state.

These initial conditions are

- SM block: Initial values of speed deviation (usually 0%), rotor angle, magnitudes and phases of currents in stator windings, and initial field voltage required to obtain the desired terminal voltage under the specified load flow
- ASM block: Initial values of slip, rotor angle, magnitudes and phases of currents in stator windings

Open the dialog box of the Synchronous Machine and Asynchronous Machine blocks. All initial conditions should be set at `0`, except for the initial SM field voltage and ASM slip, which are set at `1 pu`. Open the three scopes monitoring the SM and ASM signals as well as the bus B2 voltage. Start the simulation and observe the first 100 ms before fault is applied.

As the simulation starts, note that the three ASM currents start from zero and contain a slowly decaying DC component. The machine speeds take a much longer time to stabilize because of the inertia of the motor/load and diesel/generator systems. In our example, the ASM even starts to rotate in the wrong direction because the motor starting torque is lower than the applied load torque. Stop the simulation.

Load Flow and Machine Initialization

To start the simulation in steady state with sinusoidal currents and constant speeds, all the machine states must be initialized properly. This is a difficult task to perform manually, even for a simple system. In the next section you

learn how to use the **Load Flow and Machine Initialization** option of the Powergui block to perform a load flow and initialize the machines.

Double-click the Powergui block**.** In the **Tools** menu, click the **Load Flow and Machine Initialization** button. A new window appears. In the upper right window you have a list of the machines appearing in your system. Select the SM 3.125 MVA machine. Note that for the **Bus Type**, you have a menu allowing you to choose either PV Generator, PQ Generator, or Swing Generator.

For synchronous machines you normally specify the desired terminal voltage and the active power that you want to generate (positive power for generator mode) or absorb (negative power for motor mode). This is possible as long as you have a *swing (*or *slack)* bus that generates or absorbs the excess power required to balance the active powers throughout the network.

The swing bus can be either a voltage source or any other synchronous machine. If you do not have any voltage source in your system, you must declare one of the machines as a swing machine. In the next section, you perform a load flow with the 25 kV voltage source connected to bus B1, which is used as a swing bus.

Load Flow Without a Swing Machine

In the **Load Flow** window, your SM **Bus Type** should already be initialized as `P & V generator`, indicating that the load flow is performed with the machine controlling its active power and terminal voltage. By default, the desired **Terminal Voltage UAB** is initialized at the nominal machine voltage (2400 Vrms). Keep it unchanged and set the **Active Power** to zero. The synchronous machine therefore absorbs or generates reactive power only to keep terminal voltage at 1 pu. Now select the ASM 2250 HP machine in the upper right window. The only parameter that is needed is the **Mechanical power** developed by the motor. Enter 2000*746 (2000 HP). You now perform the load flow with the following parameters.

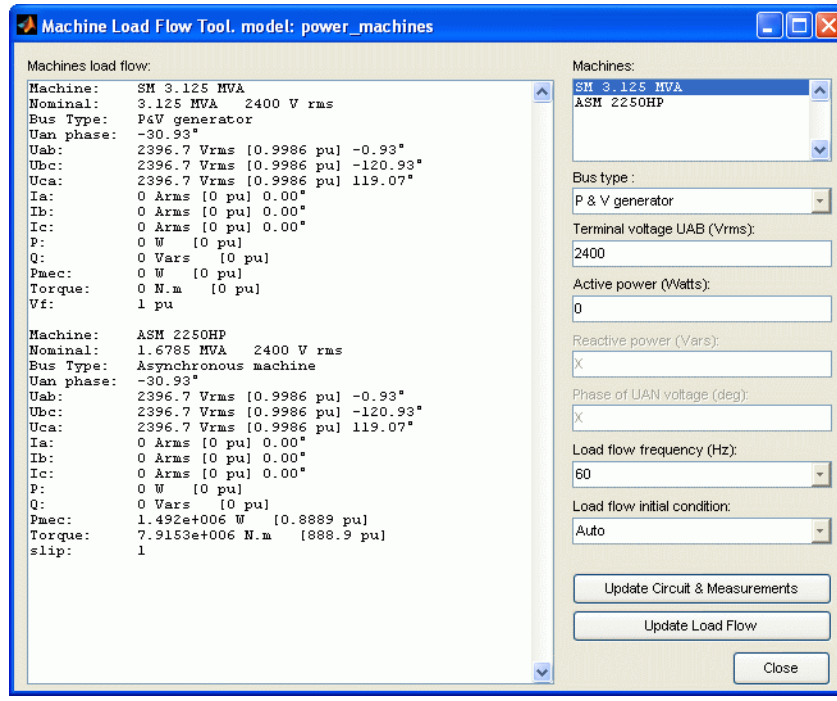
SM

Terminal Voltage	`2400 Vrms`
Active Power	`0 kW`

ASM

Mechanical Power 2000*746 W (2000 HP)

Click the **Update Load Flow** button. Once the load flow is solved, the three phasors of line-to-line machine voltages as well as currents are updated as shown on the next figure. Values are displayed both in SI units (volts RMS or amperes RMS) and in pu.



The SM active and reactive powers, mechanical power, and field voltage are displayed.

P	0 W
Q	856 kvar or 856/3125 = 0.2739 pu

Pmec	844.2 W or 0.00027 pu, representing internal machine losses in stator windings
Ef (field voltage)	1.427 pu

The ASM active and reactive powers absorbed by the motor, slip, and torque are also displayed.

P	1.515 MW (0.9024 pu)
Q	615 kvar (0.3662 pu)
Pmec	1.492 MW (2000 HP)
Slip	0.006119
Torque	7964 N.m (0.8944 pu)

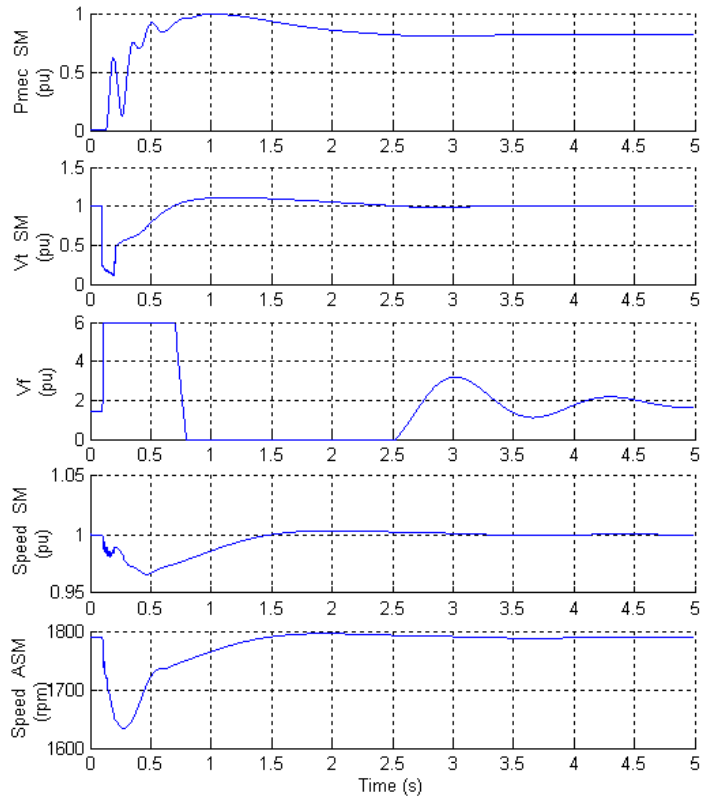
Close the **Load Flow** window.

The ASM torque value (7964 N.m) should already be entered in the Constant block connected at the ASM torque input. If you now open the **SM** and **ASM** dialog boxes you can see the updated initial conditions. If you open the Powergui, you can see updated values of the measurement outputs. You can also click the **Nonlinear** button to obtain voltages and currents of the nonlinear blocks. For example, you should find that the magnitude of the Phase A voltage across the fault breaker (named `Uc_3-Phase Fault/Breaker1`) is 14.42 kV RMS, corresponding to a 24.98 kV RMS phase-to-phase voltage.

To start the simulation in steady state, the states of the Governor & Diesel Engine and the Excitation blocks should also be initialized according to the values calculated by the load flow. Open the Governor & Diesel Engine subsystem, which is inside the Diesel Engine Speed and Voltage Control subsystem. Notice that the initial mechanical power has been automatically set to `0.0002701 pu`. Open the Excitation block and notice that the initial terminal voltage and field voltage have been set respectively to `1.0` and `1.427 pu`.

Note that the load flow also initializes the Constant blocks connected at the reference inputs (wref and vref) of the Governor and Excitation blocks as well as the Constant block connected at the load torque input (Tm) of the Asynchronous Machine block.

Open the three scopes displaying the internal signals of synchronous and asynchronous machines and phase A voltage. Start the simulation. The simulation results are shown in the following figure.



Simulation Results

Observe that during the fault, the terminal voltage drops to about 0.2 pu, and the excitation voltage hits the limit of 6 pu. After fault clearing and islanding, the SM mechanical power quickly increases from its initial value of 0 pu

to 1 pu and stabilizes at the final value of 0.82 pu required by the resistive and motor load (1.0 MW resistive load + 1.51 MW motor load = 2.51 MW = 2.51/3.125 = 0.80 pu). After 3 seconds the terminal voltage stabilizes close to its reference value of 1.0 pu. The motor speed temporarily decreases from 1789 rpm to 1635 rpm, then recovers close to its normal value after 2 seconds.

If you increase the fault duration to 12 cycles by changing the breaker opening time to 0.3 s, notice that the system collapses. The ASM speed slows down to zero after 2 seconds.

Load Flow with a Swing Machine

In this section you make a load flow with two synchronous machine types: a *PV generator* and a *swing generator*. In your **power_machines** window, delete the inductive source and replace it with the Simplified Synchronous Machine block in pu from the Machines library. Rename this machine SSM 1000MVA. Add two constant blocks at the Pm and E inputs of the Simplified Synchronous Machine. These two blocks, which are used to specify the mechanical power and the machine internal voltage, will be automatically initialized when you perform a new load flow. Save this new system in your working directory as `power_machines2.mdl`.Open the SSM 1000 MVA dialog box and enter the following parameters:

Connection type	`3-wire Y`
Pn(VA), Vn(Vrms), fn(Hz)	`[1000e6 25e3 60]`
H(s), Kd(), p (),	`[inf 0 2]`
R(pu), X(pu)	[0.1 1.0]
Init. cond.	`Leave all initial conditions at zero.`

As you specify an infinite inertia, the speed and therefore the frequency of the machine are kept constant. Notice how easily you can specify an inductive short-circuit level of 1000 MVA and a quality factor of 10 with the per unit system.

Also, connect at inputs 1 and 2 of the SSM block two Constant blocks specifying respectively the required mechanical power (Pmec) and its internal

voltage (E). These two constants are updated automatically according to the load flow solution.

When there is no voltage source imposing a reference angle for voltages, you must choose one of the synchronous machines as a reference. In a load flow program, this reference is called the *swing bus*. The swing bus absorbs or generates the power needed to balance the active power generated by the other machines and the power dissipated in loads as well as losses in all elements.

Open the Powergui. In the **Tools** menu, select **Load Flow and Machine Initialization**. Change the SSM **Bus Type** to `Swing Generator`. Specify the load flow by entering the following parameters for the SM and ASM machines:

`SM 1000 MVA:`

Terminal voltage UAB	`2400 Vrms`
Active power	`0 W`

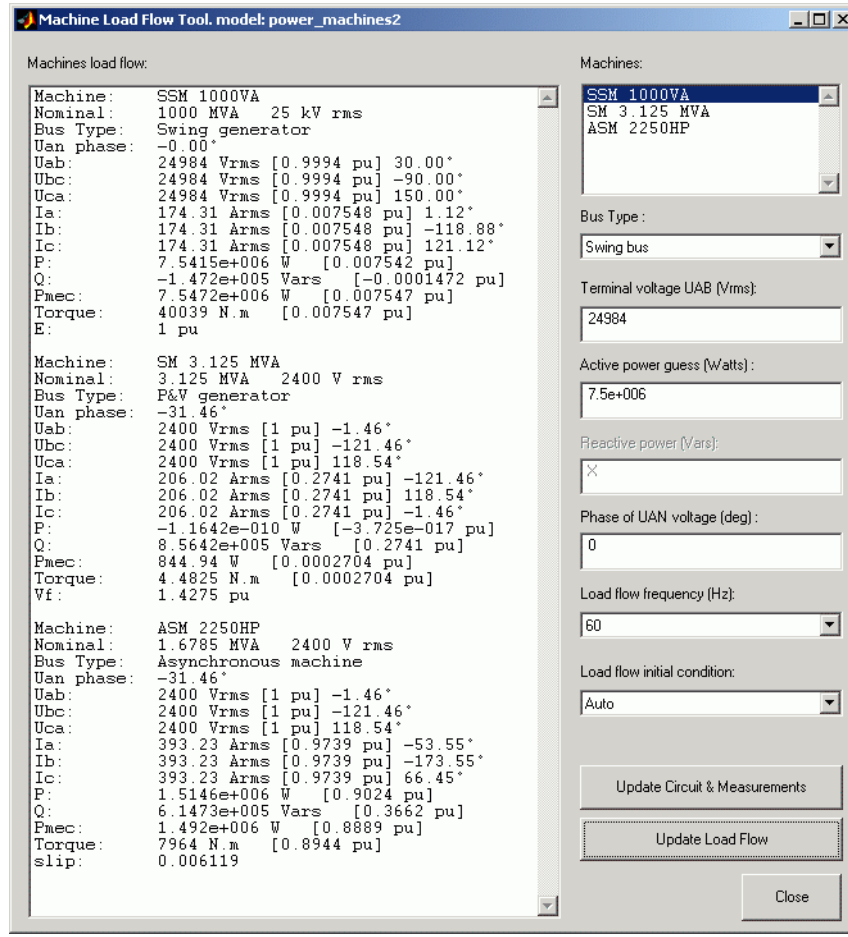
`ASM 2250 HP:`

Mechanical power	`1.492e+06 W (2000 HP)`

For the SSM swing machine you only have to specify the requested terminal voltage (magnitude and phase). The active power is unknown. However, you can specify an active power that is used as an initial guess and help load flow convergence. Respecify the following SSM parameters:

Terminal voltage	`24984 Vrms` (this voltage obtained at bus B1 from the previous load flow)
Phase of UAN voltage	`0 degrees`
Active power guess	`7.5e6 W` (estimated power = 6 MW (resistive load) + 1.5 MW motor load)

Click the **Update Load Flow** button. Once the load flow is solved the following solution is displayed. Use the scroll bar of the left window to look at the solution for each of the three machines.



The active and reactive electrical powers, mechanical power, and internal voltage are displayed for the SSM block.

```
P=7.542 MW; Q=-147 kvar
Pmec=7.547 MW (or 7.547/1000=0.007547 pu)
Internal voltage E=1.0 pu
```

The active and reactive electrical powers, mechanical power, and field voltage of the SM block are

```
P=0 W; Q=856 kvar
Pmec=844 W
Vf=1.428 pu
```

The active and reactive powers absorbed by the motor, slip, and torque of the ASM block are also displayed.

```
P=1.515MW Q=615 kvar Pmec=1.492 MW (2000 HP)
Slip=0.006119 Torque=7964 N.m
```

As expected, the solution obtained is exactly the same as the one obtained with the R-L voltage source. The active power delivered by the swing bus is 7.54 MW (6.0 MW resistive load + 1.51 MW motor load = 7.51 MW, the difference (0.03 MW) corresponding to losses in the transformer).

Restart the simulation. You should get the same waveforms as those shown in the figure called Simulation Results on page 2-31.

Reference

[1] Yeager, K.E., and J.R.Willis, "Modeling of Emergency Diesel Generators in an 800 Megawatt Nuclear Power Plant," *IEEE® Transactions on Energy Conversion*, Vol. 8, No. 3, September, 1993.

Using the Phasor Solution Method for Stability Studies

Up to now, you have simulated a relatively simple power system consisting of a maximum of three machines. If you increase complexity of your network by adding extra lines, loads, transformers, and machines, the required simulation time becomes longer and longer. Moreover, if you are interested in slow electromechanical oscillation modes (typically between 0.02 Hz and 2 Hz on large systems) you might have to simulate for several tens of seconds, implying simulation times of minutes and even hours. The conventional continuous or discrete solution method is therefore not practical for stability studies involving low-frequency oscillation modes. To allow such studies, you

have to use the phasor technique (see “Introducing the Phasor Simulation Method” on page 1-40).

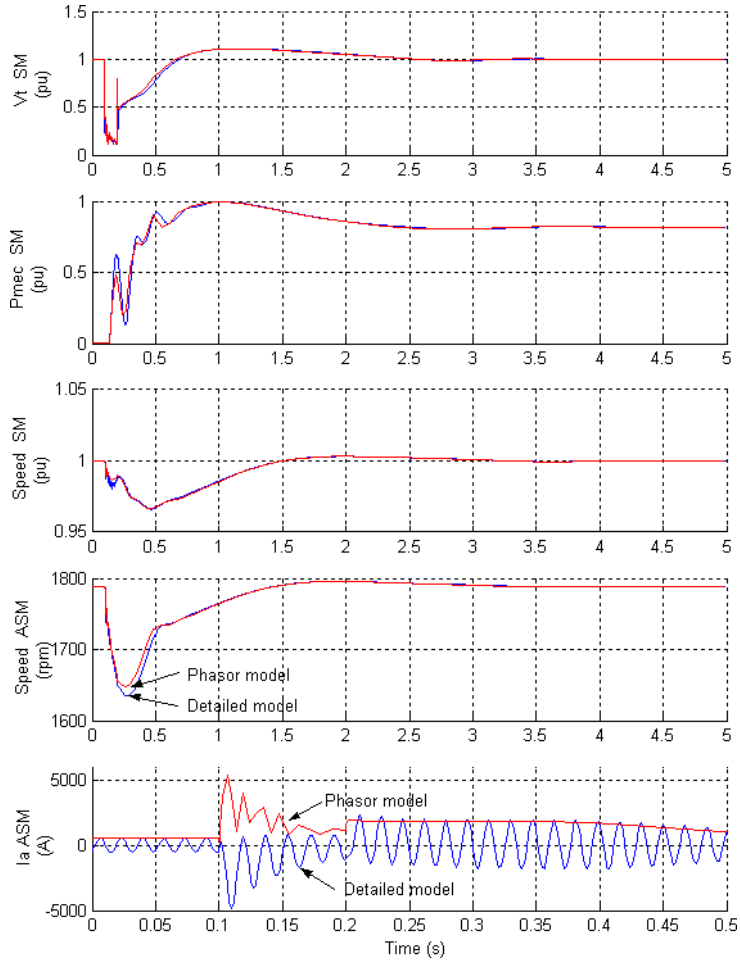
For a stability study, we are not interested in the fast oscillation modes resulting from the interaction of linear R, L, C elements and distributed parameter lines. These oscillation modes, which are usually located above the fundamental frequency of 50 Hz or 60 Hz, do not interfere with the slow machine modes and regulator time constants. In the phasor solution method, these fast modes are ignored by replacing the network's differential equations by a set of algebraic equations. The state-space model of the network is therefore replaced by a transfer function evaluated at the fundamental frequency and relating inputs (current injected by machines into the network) and outputs (voltages at machine terminals). The phasor solution method uses a reduced state-space model consisting of slow states of machines, turbines, and regulators, thus dramatically reducing the required simulation time. Continuous variable-step solvers are very efficient in solving this type of problem. Recommended solvers are `ode23t` or `ode23tb` with a maximum time step of one cycle of the fundamental frequency (1/60 s or 1/50 s).

Now apply the phasor solution method to the two-machine system you have just simulated with the conventional method. Open the `power_machines` demo.

Double-click the Powergui. Select the **Phasor simulation** option. You must also specify the fundamental frequency used to solve the algebraic network equations. A default value of 60 Hz should already be entered in the **Frequency** menu. Close the Powergui and notice that **Phasors** appears on thee Powergui icon, indicating that this new method can be used to simulate your circuit. To start the simulation in steady state, you must first repeat the load flow and machine initialization procedure explained in the previous section, “Load Flow and Machine Initialization” on page 2-27.

In the **Simulation Parameters** dialog box, specify a **Max step size** of `1/60 s` (one cycle) and start the simulation.

Observe that simulation is now much faster. The results compare well with those obtained in the previous simulation. A comparison of synchronous machine and asynchronous machine signals is shown below.



Comparison of Results for Continuous and Phasor Simulation Methods

The phasor solution method is illustrated on more complex networks presented in the Demos library. These demos are identified as

- Transient stability of two machines with power system stabilizers (PSS) and a static var compensator (SVC) (power_svc_pss model)
- Performance of three power system stabilizers for interarea oscillations (power_PSS model)

The first demo illustrates the impact of PSS and use of a SVC to stabilize a two-machine system. The second demo compares the performance of three different types of power system stabilizers on a four-machine, two-area system.

The phasor solution method is also used for FACTS models available in the factslib library. Three case studies demonstrating phasor simulation are presented in Chapter 6, “Transient Stability of Power Systems Using Phasor Simulation”.

Building and Customizing Nonlinear Models

In this section...

"Introduction" on page 2-39

"Modeling a Nonlinear Inductance" on page 2-39

"Customizing Your Nonlinear Model" on page 2-44

"Modeling a Nonlinear Resistance" on page 2-47

"Creating Your Own Library" on page 2-52

"Connecting Your Model with Other Nonlinear Blocks" on page 2-52

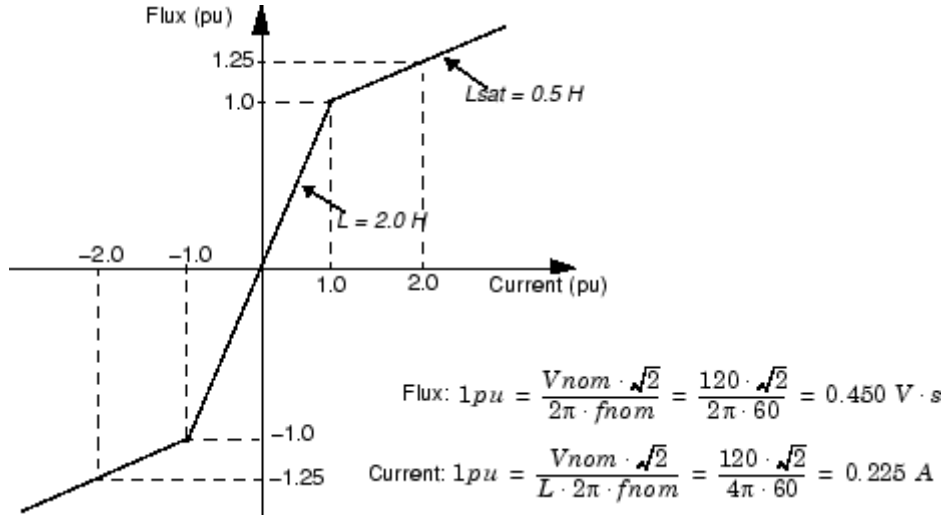
Introduction

SimPowerSystems™ software provides a wide collection of nonlinear models. It can happen, however, that you need to interface your own nonlinear model with the standard models provided in the **powerlib** library. This model could be a simple nonlinear resistance simulating an arc or a varistor, a saturable inductor, a new type of motor, etc.

In the following section you learn how to build such a nonlinear model. A simple saturable inductance and a nonlinear resistance serve as examples.

Modeling a Nonlinear Inductance

Consider an inductor of 2 henries designed to operate at a nominal voltage, Vnom = 120 V RMS, and a nominal frequency, fnom = 60 Hz. From zero to 120 V RMS the inductor has a constant inductance, L = 2 H. When voltage exceeds its nominal voltage, the inductor saturates and its inductance is reduced to Lsat = 0.5 H. The nonlinear flux-current characteristic is plotted in the next figure. Flux and current scales are in per units. The nominal voltage and nominal current are chosen as base values for the per-unit system.



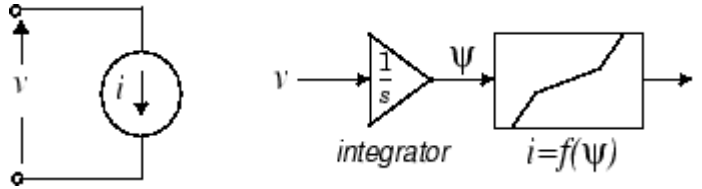
Flux-Current Characteristic of the Nonlinear Inductance

The current i flowing in the inductor is a nonlinear function of flux linkage ψ that, in turn, is a function of v appearing across its terminals. These relations are given by the following equations:

$$v = L \cdot \frac{di}{dt} = \frac{d\psi}{dt} \qquad \text{or} \qquad \psi = \int v \cdot dt$$

$$i = \frac{\psi}{L(\psi)}$$

The model of the nonlinear inductance can therefore be implemented as a controlled current source, where current i is a nonlinear function of voltage v, as shown.



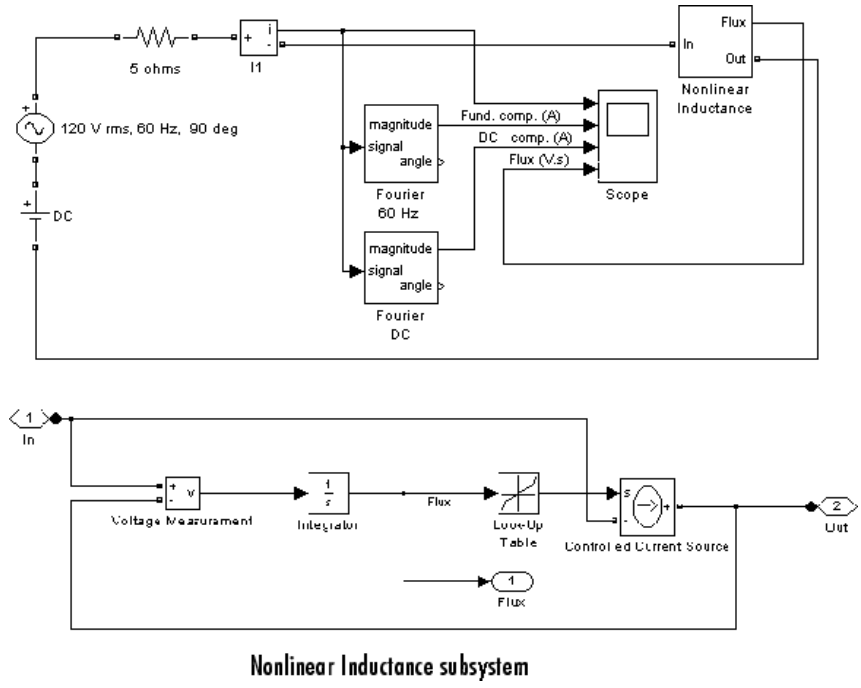
Model of a Nonlinear Inductance

Implementation of a Nonlinear Inductance on page 2-42 shows a circuit using a 2 H nonlinear inductance. The nonlinear inductance is connected in series with two voltage sources (an AC Voltage Source block of 120 volts RMS, 60 Hz, and a DC Voltage Source block) and a 5 ohm resistor.

All the elements used to build the nonlinear model have been grouped in a subsystem named Nonlinear Inductance. The inductor terminals are labeled `In` and `Out`. Notice that a second output returning the flux has been added to the subsystem. You can use this output to observe the flux by connecting it to a Simulink® Scope block.

The nonlinear model uses two **powerlib** blocks and two Simulink blocks. The two **powerlib** blocks are a Voltage Measurement block to read the voltage at the inductance terminals and a Controlled Current Source block. The direction of the arrow of the current source is oriented from input to output according to the model shown above.

The two Simulink blocks are an Integrator block computing the flux from the voltage input and a Look-Up Table block implementing the saturation characteristic $i = f(\psi)$ described by Flux-Current Characteristic of the Nonlinear Inductance on page 2-40.



Nonlinear Inductance subsystem

Implementation of a Nonlinear Inductance

Two Fourier blocks from the Measurements library of **powerlib_extras** are used to analyze the fundamental component and the DC component of the current.

Using blocks of the **powerlib** and Simulink libraries, build the circuit shown above. To implement the $i = f(\psi)$ relation, specify the following vectors in the Look-Up Table block:

Vector of input values (flux)	[-1.25 -1 1 1.25] *(120*sqrt(2)/(2π*60))
Vector of output values (current)	[-2 -1 1 2]*(120*sqrt(2)/(4π*60))

Save your circuit as `circuit7`.

Set the following parameters for the two sources:

AC source		
	Peak amplitude	`120*sqrt(2)`
	Phase	`90 degrees`
	Frequency	`60 Hz`
DC source		
	Amplitude	`0 V`

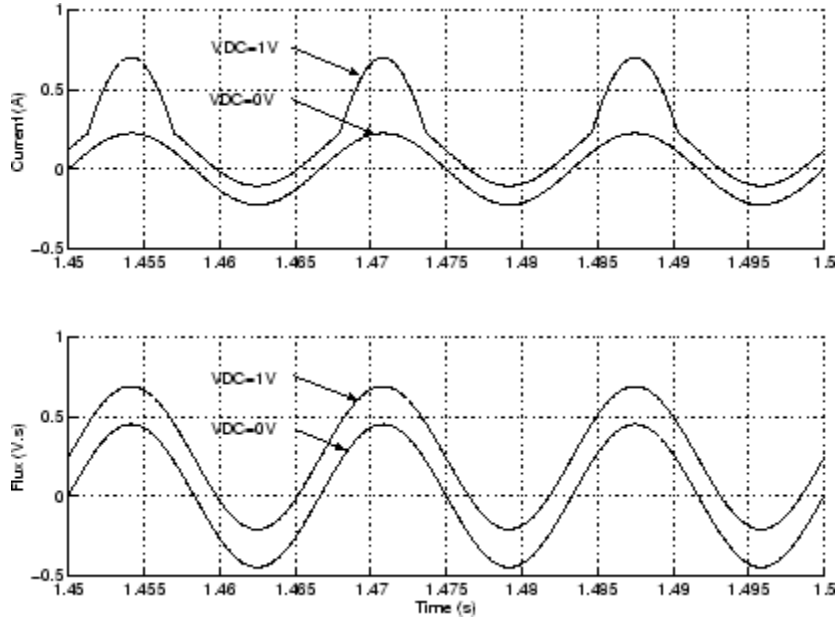
Adjust the simulation time to `1.5 s` and select the `ode33tb` integration algorithm with default parameters. Start the simulation.

As expected, the current and the flux are sinusoidal. Their peak values correspond to the nominal values.

$$Peak \cdot Current = \frac{120 \cdot \sqrt{2}}{2 \cdot 2\pi \cdot 60} = 0.225\ A$$

$$Peak \cdot Flux = \frac{120 \cdot \sqrt{2}}{2\pi \cdot 60} = 0.450\ V \cdot s$$

Current and flux waveforms are shown.



Current and Flux Waveforms Obtained with VDC = 0 V and VDC = 1 V

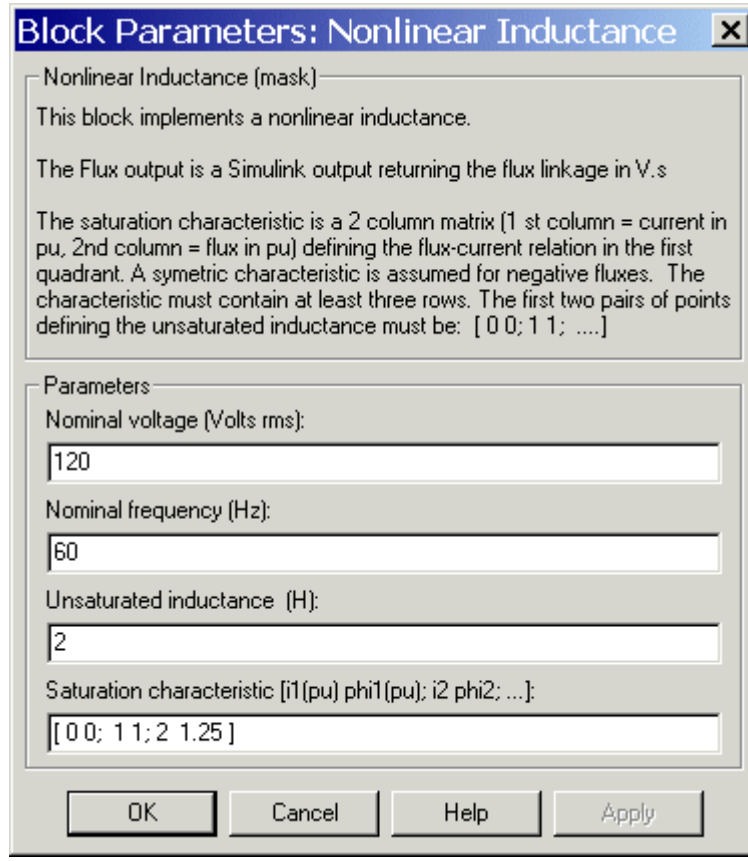
Now change the DC voltage to 1 V and restart the simulation. Observe that the current is distorted. The 1 V DC voltage is now integrated, causing a flux offset, which makes the flux enter into the nonlinear region of the flux-current characteristic ($\psi > 0.450$ V.s). As a result of this flux saturation, the current contains harmonics. Zoom in on the last three cycles of the simulation. The peak value of the current now reaches 0.70 A and the fundamental component has increased to 0.368 A. As expected, the DC component of the current is 1 V/ 0.5 Ω = 0.2. The current and flux waveforms obtained with and without saturation are superimposed in the figure above.

Customizing Your Nonlinear Model

Simulink software provides the Masking facilities to create a dialog box for your models. You can create a mask that specifies the following prompts and variables:

Nominal voltage (Volts rms):	`Vnom`
Nominal frequency (Hz):	`Fnom`
Unsaturated inductance (H):	`L`
Saturation characteristic [i1(pu) phi1(pu); i2 phi2; ...]:	`sat`

The resulting mask for your nonlinear inductance block is shown in the next figure.



Dialog Box of the Nonlinear Inductance Block

The following code in the mask initializations of the block prepares the two vectors `Current_vect` and `Flux_vect` to be used in the Look-Up Table block of the model.

```
% Define base current and Flux for pu system
I_base = Vnom*sqrt(2)/(L*2*pi*fnom);
Phi_base = Vnom*sqrt(2)/(2*pi*fnom);

% Check first two points of the saturation characteristic
if ~all(all(sat(1:2,:)==[0 0; 1 1])),
    h=errordlg('The first two points of the characteristic must
be [0 0; 1 1]','Error');
    uiwait(h);
end

% Complete negative part of saturation characteristic
[npoints,ncol]=size(sat);
sat1=[sat ; -sat(2:npoints,:)];
sat1=sort(sat1);

% Current vector (A)  and flux vector (V.s)
Current_vect=sat1(:,1)*I_base;
Flux_vect=sat1(:,2)*Phi_base;
```

As the saturation characteristic is specified only in the first quadrant, three lines of code are added to complete the negative part of the saturation characteristic. Notice also how the validity of the first segment of the saturation characteristic is verified. This segment must be defined by two points `[0 0; 1 1]` specifying a 1 pu inductance (nominal value) for the first segment.

Before you can use the masked block, you must apply the two internal variables defined in the initialization section of the block. Open the Look-Up Table block dialog box and enter the following variable names in the two fields:

Vector of input values (flux)	`Flux_vect`
Vector of output values (current)	`Current_vect`

Close the Nonlinear Inductance subsystem and start the simulation. You should get the same waveforms as shown in Current and Flux Waveforms When Energizing the Nonlinear Inductance with Maximum Flux Offset on page 2-55.

Modeling a Nonlinear Resistance

The technique for modeling a nonlinear resistance is similar to the one used for the nonlinear inductance.

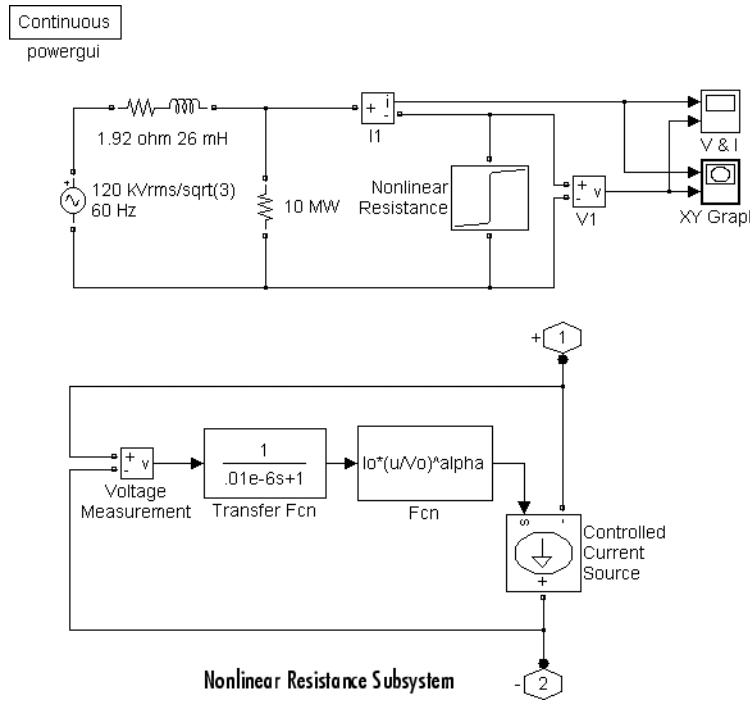
A good example is a metal-oxide varistor (MOV) having the following V-I characteristic:

$$i = I_o \cdot \left(\frac{v}{V_o}\right)^{\alpha}$$

where

v, i =	Instantaneous voltage and current
Vo =	Protection voltage
Io =	Reference current used to specify the protection voltage
α =	Exponent defining the nonlinear characteristic (typically between 10 and 50)

The following figure shows an application of such a nonlinear resistance to simulate a MOV used to protect equipment on a 120 kV network. To keep the circuit simple, only one phase of the circuit is represented.

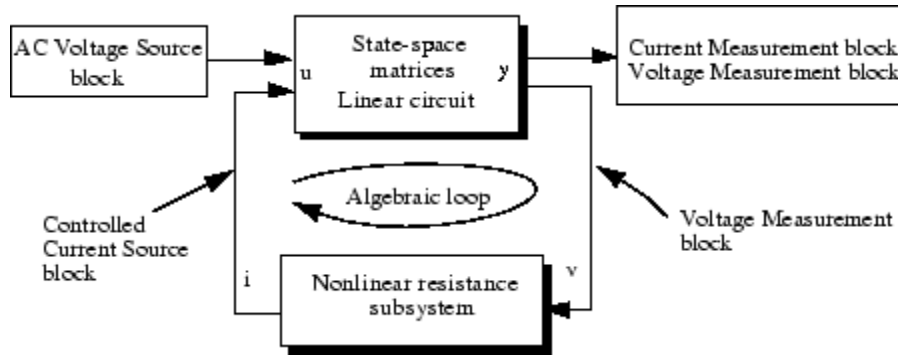


Nonlinear Resistance Applied on a 120 kV Network

Using blocks of the **powerlib** and Simulink libraries, build this circuit. Group all components used to model the nonlinear model in a subsystem named Nonlinear Resistance. Use an X-Y Graph block to plot the V-I characteristic of the Nonlinear Resistance subsystem.

The model does not use a Look-Up Table block as in the case of the nonlinear inductance model. As the analytical expression of current as a function of voltage is known, the nonlinear *I(V)* characteristic is implemented directly with a Fcn block from the User-Defined Functions Simulink library.

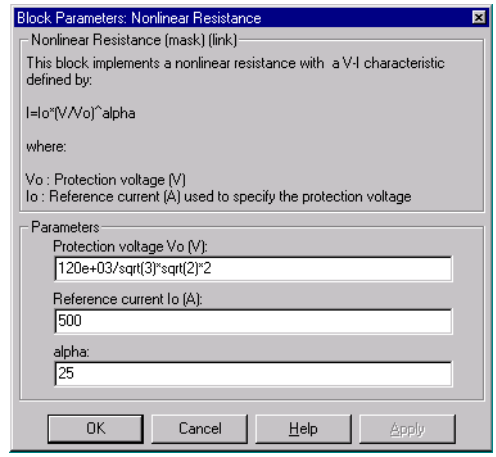
This purely resistive model contains no states. It produces an algebraic loop in the state-space representation of the circuit, as shown in the next figure.



Algebraic Loop Introduced by the Nonlinear Resistance Model

Algebraic loops often lead to slow simulation times. You should break the loop with a block that does not change the nonlinear characteristic. Here a first-order transfer function *H(s) = 1/(1+Ts)* is introduced into the system, using a fast time constant (T = 0.01 µs).

Use the technique explained for the nonlinear inductance block to mask and customize your nonlinear resistance block as shown.



Dialog Box of the Nonlinear Resistance Block

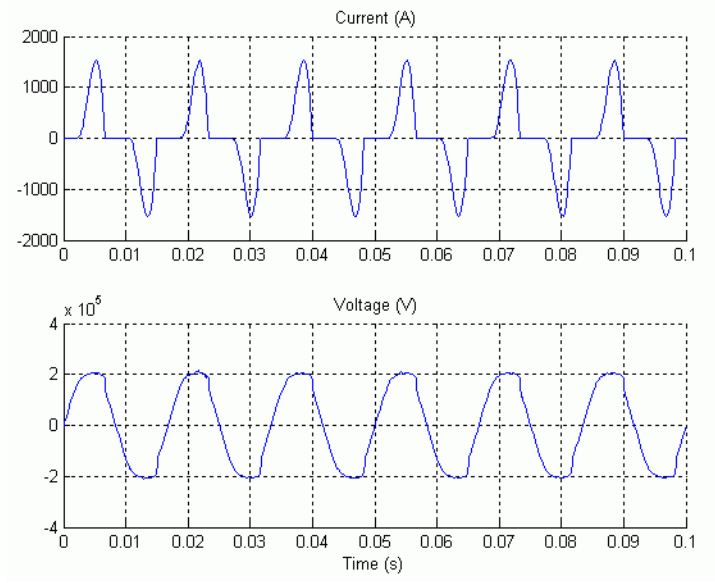
Open the dialog box of your new masked block and enter the parameters shown in the figure above. Notice that the protection voltage Vo is set at 2

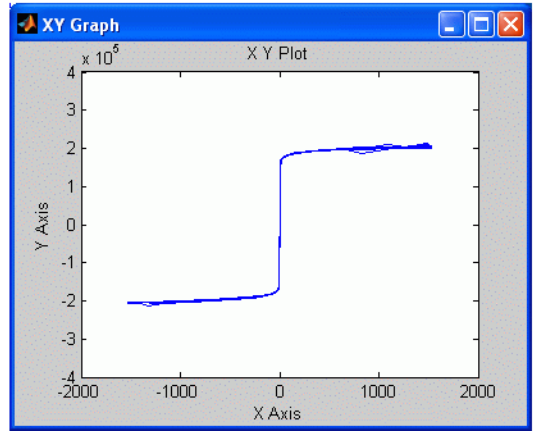
pu of the nominal system voltage. Adjust the source voltage at 2.3 pu by entering the following peak amplitude:

```
120e3/sqrt(3)*sqrt(2)*2.3
```

Save your circuit as `circuit8`.

Using the ode23t integration algorithm, simulate your `circuit8` system for 0.1 s. The results are shown below.

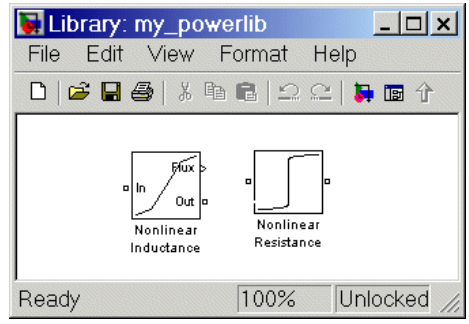




Current and Voltage Waveforms and V-I Characteristic Plotted by the X-Y Graph Block

Creating Your Own Library

You can create your own block libraries. To create a library, in the **File** menu choose **New Library**. A new Simulink window named **Library: untitled** opens. Now copy the Nonlinear Inductance block of your `circuit7` system and the Nonlinear Resistance block of your `circuit8` system into that library. Save this library as **my_powerlib**. Next time you develop a new model, you can add it to your personal library. You can also organize your library in different sublibraries according to their functions, as is done in the **powerlib** library.



Nonlinear Inductance and Resistance Blocks in my_powerlib

One advantage of using a library is that all blocks that you copy from that library are referenced to the library. In other words, if you make a correction in your library block, the correction is automatically applied to all circuits using that block.

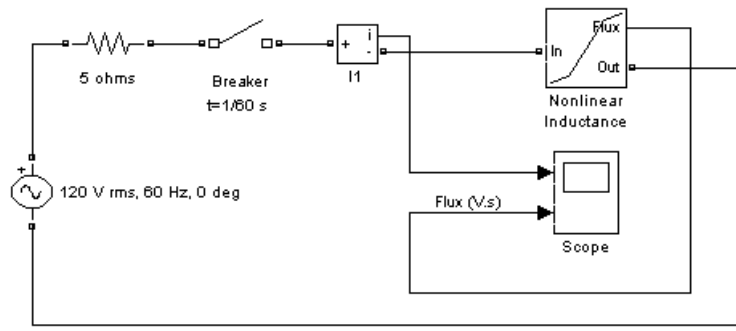
Connecting Your Model with Other Nonlinear Blocks

You now learn how to avoid error messages that can appear with nonlinear blocks when they are simulated by a current source. Obviously, a current source cannot be connected in series with an inductor, another current source, or an open circuit. Such circuit topologies are forbidden in SimPowerSystems models.

Similarly, if your nonlinear model uses a Controlled Voltage Source block, this model could not be short-circuited or connected across a capacitor.

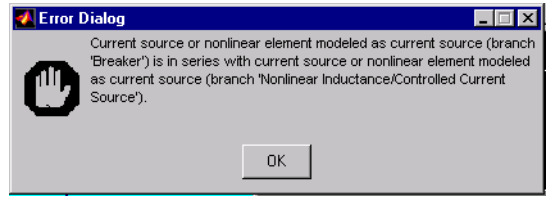
Suppose, for example, that you want to study the inrush current in a nonlinear inductance when it is energized on a voltage source. Using blocks from **powerlib** library and **my_powerlibrary**, you can build the circuit shown here. Change the Breaker block parameters as follows:

Snubber resistance Rs	`inf (no snubber)`
Snubber capacitance Cs	`0`
External control	`Not selected`
Switching times	`[1/60]`



Circuit Topology Causing an Error

If you try to simulate this circuit, you get the following error message.



This topology is forbidden because two nonlinear elements simulated by current sources are connected in series: the Breaker block and the Nonlinear Inductance block. To be able to simulate this circuit, you must provide a current path around one of the two nonlinear blocks. You could, for

example, connect a large resistance, say 1 MΩ, across the Breaker block or the Inductance block.

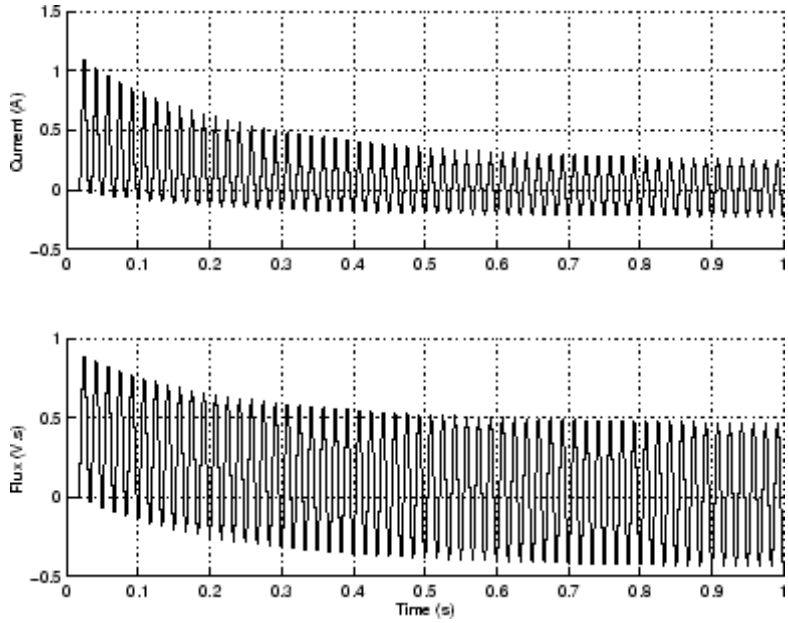
In this case, it is more convenient to choose the Breaker block because a series RC snubber circuit is provided with the model. Open the Breaker block dialog box and specify the following snubber parameters:

Snubber resistance Rs (ohms)	`1e6`
Snubber capacitance Cs (F)	`inf`

Notice that to get a purely resistive snubber you have to use an infinite capacitance.

Note Using an inductive source impedance (R-L series) instead of a purely resistive impedance would have produced another error message, because the current source modeling the nonlinear inductance would have been in series with an inductance, even with a resistive snubber connected across the breaker. In such a case, you could add either a parallel resistance across the source impedance or a large shunt resistance connected between one breaker terminal and the source neutral terminal.

Make sure that the phase angle of the voltage source is zero. Use the `ode23t` integration algorithm and simulate the circuit for 1 second. Voltage and current waveforms are shown here.



Current and Flux Waveforms When Energizing the Nonlinear Inductance with Maximum Flux Offset

The figure above shows that energizing the inductor at a zero crossing of voltage results in a maximum flux offset and saturation.

Building a Model Using Model Construction Commands

This section shows you how to use model construction commands to add blocks to your models and connect them.

Suppose you want to add a PI Section Line block and a Voltage Measurement block to your model, connect the + terminal of the Voltage Measurement block to the left end of the PI Section Line block, and connect the - terminal of the Voltage Measurement block to the right end of the PI Section Line block.

The following code shows you how to add and position the two blocks in your model.

```
add_block('powerlib/Elements/Pi Section Line','Mymodel/Block1');
add_block('powerlib/Measurements/Voltage Measurement',
'Mymodel/Block2');
set_param('Mymodel/Block1','position',[340,84,420,106]);
set_param('Mymodel/Block2','position',[520,183,545,207]);
```

For each block you want to connect, you need to know the handles of the terminal ports.

```
Block1PortHandles = get_param('Mymodel/Block1','PortHandles');
Block2PortHandles = get_param('Mymodel/Block2','PortHandles');
```

The `add_line` command uses the `RConn` and `Lconn` fields of the `Block1PortHandles` and `Block2PortHandles` structure variables to connect the blocks. The `RConn` field represents the right connectors of the blocks and the `Lconn` field represents the left connectors. You then need to specify to the `add_line` command the indices of the connectors you want to connect.

```
add_line('Mymodel',Block1PortHandles.LConn(1),
Block2PortHandles.LConn(1));
add_line('Mymodel',Block1PortHandles.RConn(1),
Block2PortHandles.LConn(2));
```

3

Improving Simulation Performance

How SimPowerSystems™ Software Works (p. 3-3)	Overview of what the software does when it analyzes and runs your models
Choosing an Integration Method (p. 3-5)	Advantages and disadvantages of continuous, discrete, and phasor simulation of power system models
Simulating with Continuous Integration Algorithms (p. 3-7)	How to integrate continuous time power models
Simulating Discretized Electrical Systems (p. 3-9)	How to solve discretized power models
Simulating Power Electronic Models (p. 3-11)	How to simulate power electronic models
Increasing Simulation Speed (p. 3-13)	Ways to optimize simulation speed and efficiency, including the Simulink® Accelerator mode and Real-Time Workshop® software
The Nonlinear Model Library (p. 3-16)	Using and modifying the powerlib_models library to model nonlinear power components

Creating Your Own Library of Models (p. 3-19)	Creating your own custom power system blocks with the Simulink block masking feature
Changing Your Circuit Parameters (p. 3-20)	Modifying SimPowerSystems™ block parameters during simulation and automating with MATLAB® scripts

How SimPowerSystems™ Software Works

Every time you start the simulation, a special initialization mechanism is called. This initialization process computes the state-space model of your electric circuit and builds the equivalent system that can be simulated by Simulink® software. This process performs the following steps, as shown in :

1. Sorts all SimPowerSystems™ blocks, gets the block parameters and evaluates the network topology. The blocks are separated into linear and nonlinear blocks, and each electrical node is automatically given a node number.

2. Once the network topology has been obtained, the state-space model (A, B, C, D matrices) of the linear part of the circuit is computed. All steady-state calculations and initializations are performed at this stage.

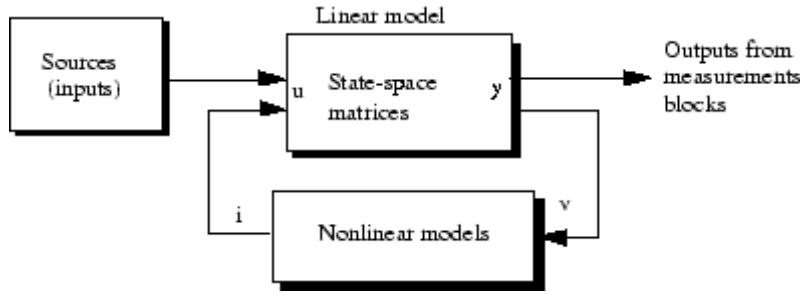
   If you have chosen to discretize your circuit, the discrete state-space model is computed from the continuous state-space model, using the Tustin method.

   If you are using the phasor solution method, the state-space model is replaced with the complex transfer matrix $H(j\omega)$ relating inputs and outputs (voltage and current phasors) at the specified frequency. This matrix defines the network algebraic equations.

3. Builds the Simulink model of your circuit and stores it inside the Powergui block located at the top level of your model.

The Simulink model uses an S-Function block to model the linear part of the circuit. Predefined Simulink models are used to simulate nonlinear elements. These models can be found in the SimPowerSystems **powerlib_models** library. Simulink Source blocks connected at the input of the State-Space block are used to simulate the electrical source blocks.

The next figure represents the interconnections between the different parts of the complete Simulink model. The nonlinear models are connected in feedback between voltage outputs and current inputs of the linear model.



Interconnection of Linear Circuit and Nonlinear Models

Once SimPowerSystems software has completed the initialization process, the simulation starts. You can observe waveforms on scopes connected at the outputs of your measurement blocks. Through the Powergui, you can access the LTI viewer and obtain transfer functions of your system between any pair of input and output. The Powergui also allows you to perform a FFT analysis of recorded signals to obtain their frequency spectrum.

If you stop the simulation and double-click the Powergui block, you have access to the steady-state values of inputs, outputs, and state variables displayed as phasors. You can also use the Powergui to modify the initial conditions. The Powergui block interface allows you to perform a load flow with circuits involving three-phase machinery and initialize the machine models so that the simulation starts in steady state. This feature avoids long transients due to mechanical time constants of machines. The Powergui block allows you to specify the desired frequency range, visualize impedance curves, and store results in your workspace for Impedance Measurement blocks connected in your circuit.

Choosing an Integration Method

In this section...

"Introduction" on page 3-5

"Continuous Versus Discrete Solution" on page 3-5

"Phasor Solution Method" on page 3-6

Introduction

Three solution methods are available through the Powergui block. These are:

- Continuous solution method using Simulink® variable-step solvers
- Discretization for solution at fixed time steps
- Phasor solution method using Simulink variable-step solvers

Continuous Versus Discrete Solution

One important feature of SimPowerSystems™ software is its ability to simulate electrical systems either with continuous variable-step integration algorithms or with a fixed-step using a discretized system. For small size systems, the continuous method is usually more accurate. Variable-step algorithms are also faster because the number of steps is fewer than with a fixed-step method giving comparable accuracy. When using line-commutated power electronics, the variable-step, event-sensitive algorithms detect the zero crossings of currents in diodes and thyristors with a high accuracy so that you do not observe any current chopping. However, for large systems (containing either a large number of states or nonlinear blocks), the drawback of the continuous method is that its extreme accuracy slows down the simulation. In such cases, it is advantageous to discretize your system. In the following two sections, we explain these two methods, their advantages, and their limitations.

What do we mean by "small size" and "large size"? Although the distinction is not sharp, you can consider small size a system that contains fewer than 30 electrical states and fewer than 6 electronic switches. Circuit breakers do not affect the speed much, because unlike power electronic switches, which

are commutated at every cycle, these devices are operated only a couple of times during a test.

Phasor Solution Method

If you are interested only in the changes in magnitude and phase of all voltages and currents when switches are closed or opened, you don't need to solve all differential equations (state-space model) resulting from the interaction of R, L, C elements. You can instead solve a much simpler set of algebraic equations relating the voltage and current phasors. This is what the phasor solution method does. As its name implies, this method computes voltages and currents as phasors. The phasor solution method is particularly useful for studying transient stability of networks containing large generators and motors. In this type of problem, we are interested in electromechanical oscillations resulting from interactions of machine inertias and regulators. These oscillations produce a modulation of the magnitude and phase of fundamental voltages and currents at low frequencies (typically between 0.02 Hz and 2 Hz). Long simulation times are therefore required (several tens of seconds). The continuous or discrete solution methods are not appropriate for this type of problem.

In the phasor solution method, the fast modes are ignored by replacing the network differential equations by a set of algebraic equations. The state-space model of the network is replaced by a complex matrix evaluated at the fundamental frequency and relating inputs (currents injected by machines into the network) and outputs (voltages at machine terminals). As the phasor solution method uses a reduced state-space model consisting of slow states of machines, turbines and regulators, it dramatically reduces the required simulation time.

Continuous variable-step solvers are very efficient in solving this type of problem. Recommended solvers are `ode23t` or `ode23tb` with a maximum time step of one cycle of the fundamental frequency (1/60 s or 1/50 s).You must keep in mind however that this faster solution technique gives the solution only in the vicinity of the fundamental frequency.

Simulating with Continuous Integration Algorithms

In this section...

"Choosing an Integration Algorithm" on page 3-7

"Simulating Switches and Power Electronic Devices" on page 3-8

Choosing an Integration Algorithm

Simulink® software provides a variety of solvers. Most of the variable-step solvers work well with linear circuits. However circuits containing nonlinear models, especially circuits with circuit breakers and power electronics, require stiff solvers.

Best accuracy and fastest simulation speed is usually achieved with `ode23t`.

Solver	`ode23t`
Relative tolerance	`1e-4`
Absolute tolerance	`auto`
Maximum step size	`auto`
Initial step size	`auto`
Solver reset method	`fast`

Normally, you can choose `auto` for the absolute tolerance and the maximum step size. In some occasions you might have to limit the maximum step size and the absolute tolerance. Selecting too small a tolerance can slow down the simulation considerably. The choice of the absolute tolerance depends on the maximum expected magnitudes of the state variables (inductor currents, capacitor voltages, and control variables).

For example, if you work with high-power circuit where expected voltage and currents are thousands of volts and amperes, an absolute tolerance of 0.1 or even 1.0 would be sufficient for the electric states. However, if your electrical circuit is associated with a control system using normalized control signals (varying around 1), the absolute tolerance is imposed by the control states. In this case, choosing an absolute tolerance of 1e-3 (1% of control signal) would

be appropriate. If, on the other side, you are working with a very low power circuit with expected currents of milliamperes, you should probably set the absolute tolerance to 1e-6.

Note Usually, keeping the **Solver reset method** parameter of the ode23t solver to its default value (`Fast`) will give best simulation performance. However, for some highly nonlinear circuits it may be necessary to set this parameter to `Robust`. When you build a new model, we recommend that you try both the `Robust` and the `Fast` reset methods. If you do not notice a difference in simulation results, then keep the `Fast` method, which provides fastest simulation speed.

Simulating Switches and Power Electronic Devices

Two methods are used for simulation of switches and power electronic devices:

- If the switch is purely resistive the switch model is considered as part of the linear circuit. The state-space model of the circuit, including open and closed switches, is therefore recalculated at each switch opening or closing, producing a change in the circuit topology. This method is always used with the Breaker block and the Ideal Switch block because these elements do not have internal inductance. It is also applied for the power electronic blocks when Ron > 0 and Lon = 0, and for the Universal Bridge with forced commutated devices.
- If the switch contains a series inductance (Diode and Thyristor with Lon > 0, IGBT, MOSFET, or GTO), the switch is simulated as a current source driven by voltage across its terminals. The nonlinear element (with a voltage input and a current output) is then connected in feedback on the linear circuit, as shown in the Interconnection of Linear Circuit and Nonlinear Models on page 3-4.

Note You have therefore the choice to simulate diodes and thyristors with or without Lon internal inductance. In most applications, it is not necessary to specify an inductance Lon. However, for certain circuit topologies, you may have to specify a switch inductance Lon to help commutation.

Simulating Discretized Electrical Systems

In this section...

"Introduction" on page 3-9

"Limitations of Discretization with Nonlinear Models" on page 3-9

Introduction

You implement discretization by selecting **Discretize electrical model** in the Powergui block dialog box. The sample time is specified in the block dialog box. The electrical system is discretized using the Tustin method, which is equivalent to a fixed-step trapezoidal integration. To avoid algebraic loops, the electrical machines are discretized using the Forward Euler method.

The precision of the simulation is controlled by the time step you choose for the discretization. If you use too large a sample time, the precision might not be sufficient. The only way to know if it is acceptable is to repeat the simulation with different sample times or to compare with a continuous method and to find a compromise for the largest acceptable sample time. Usually sample times of 20 µs to 50 µs give good results for simulation of switching transients on 50 Hz or 60 Hz power systems or on systems using line-commutated power electronic devices such as diodes and thyristors. However, for systems using forced-commutated power electronic switches, you must reduce the time step. These devices, the insulated-gate-bipolar transistor (IGBT), the field-effect transistor (FET), and the gate-turnoff thyristor (GTO) are usually operating at high switching frequencies. For example, simulating a pulse-width-modulated (PWM) inverter operating at 8 kHz requires a maximum time step of 1 µs or less.

Note that even if you discretize your electric circuit, you can still use a continuous control system. However, the simulation speed is improved by use of a discrete control system.

Limitations of Discretization with Nonlinear Models

There are a few limitations to discretizing nonlinear models.

Minimal Load Is Required at Machine Terminals

When using electrical machines in discrete systems, you might have to use a small parasitic resistive load, connected at the machine terminals, to avoid numerical oscillations. Large sample times require larger loads. The minimum resistive load is proportional to the sample time. As a rule of thumb, remember that with a 25 µs time step on a 60 Hz system, the minimum load is approximately 2.5% of the machine nominal power. For example, a 200 MVA synchronous machine in a power system discretized with a 50 µs sample time requires approximately 5% of resistive load or 10 MW. If the sample time is reduced to 20 µs, a resistive load of 4 MW should be sufficient.

Lon = 0 Is Used for Power Electronic Devices

Naturally commutated devices (diodes and thyristors), as well as forced-commutated devices (IGTO, IGBT, FET), used in a discretized circuit must have a zero internal inductance. If you discretize a circuit containing power electronic devices with Lon > 0, SimPowerSystems™ software prompts you with a warning indicating that Lon will be reset to zero.

Simulating Power Electronic Models

In this section...

"Introduction" on page 3-11

"Circuit Using Forced-Commutated Power Electronics" on page 3-11

"Circuit Using Naturally Commutated Power Electronics" on page 3-11

Introduction

There are two types of circuits using power electronics:

- Circuit using forced-commutated power electronics
- Circuit using naturally commutated power electronics

Circuit Using Forced-Commutated Power Electronics

In this type of circuit, the semiconductor such as IGBT GTO FET turns off instantaneously when the gate signal is removed. This switching off may in turn cause instantaneous switching on of another device (usually a diode).

This problem of instantaneous switching of two or more devices in the same time step has been solved in the SimPowerSystems™ software, both for continuous and discrete fixed time step solvers. It is now possible to discretize circuits with any type of topology, like multilevel converters. This was possible before only with the use of the Universal Bridge.

The power electronic models are implemented directly in the S-functions that compute the state-space equations of the model.

Circuit Using Naturally Commutated Power Electronics

In this type of circuit, the diode or thyristor is turned off at zero crossing of current. It means that if the model is discretized at fixed time step, the natural zero crossing will most probably occur between two time steps. As current inversion is detected at a time step following the real zero crossing, a small negative current will be chopped. This current chopping in inductive

circuits will cause numerical oscillations that can be observed in simple rectifier circuits.

This problem is not yet solved in the SimPowerSystems software because we still use a fixed-step solver. (The system is discretized using a trapezoidal solver.) The way to avoid this problem is to increase the RC snubbers connected across diodes or thyristors. But if large time steps are used, this may lead to unacceptable leakage current in the snubbers.

Fortunately, the trend of power electronics is to use forced-commutated power electronics and PWM rather than naturally commutated power electronics.

Increasing Simulation Speed

In this section...

"Ways to Increase Simulation Speed" on page 3-13

"Using Accelerator Mode and Real-Time Workshop® Software" on page 3-13

Ways to Increase Simulation Speed

Once the proper method (continuous, discrete, or phasor), solver type, and parameters have been selected, there are additional steps you can take to optimize your simulation speed:

- Discretizing your electric circuit and your control system. You can even use a larger sample time for the control system, provided that it is a multiple of the smallest sample time.
- Simulating large systems or complex power electronic converters can be time consuming. If you have to repeat several simulations from a particular operating point, you can save time by specifying a vector of initial states in the **Simulation > Simulation parameters > Workspace IO** dialog box pane. This vector of initial conditions must have been saved from a previous simulation run.
- Reducing the number of open scopes and the number of points saved in the scope also helps in reducing the simulation time.
- Using the Simulink® Accelerator mode. The performance gain obtained with the Accelerator varies with the size and complexity of your model. Typically you can expect performance improvements by factors of two to 10.

Using Accelerator Mode and Real-Time Workshop® Software

The Simulink Accelerator mode is explained in the "Accelerating Models" documentation.

The Accelerator mode speeds up the execution of Simulink models by replacing the interpreted M code running beneath the Simulink blocks with compiled code as your model executes. The Accelerator mode uses portions of Real-Time Workshop® software to generate this code on the fly. Although

the Accelerator mode uses this technology, Real-Time Workshop license is not required to run it. Also, if you do not have your own C compiler installed, you can use the lcc C compiler provided with your MATLAB® installation.

To activate the Accelerator mode, select **Accelerator** from the **Simulation** menu of your model window. Alternatively, you can select `Accelerator` from the pull-down menu in the model window toolbar.

The following table shows typical performance gains obtained with discretization and Accelerator mode applied on the following two demos: a DC drive using a chopper and the AC-DC converter using a three-phase, three-level voltage-sourced converter. Two versions of the DC drive model are provided in the Demos library: a continuous version, `power_dcdrive`, and a discrete version, `power_dcdrive_disc`. The AC-DC converter is available as the `power_3levelVSC` demo.

	Simulation Time in Seconds*	
Simulation Method	DC drive	AC-DC converter
	(Stop time = 2 s)	(Stop time = 0.2 s)
Continuous: `ode23t` default parameters	12	—
Discrete	9.0 (`Ts` = 10 µs)	14.5 (`Ts` = 5 µs)
Discrete + Accelerator	5.2 (`Ts` = 10 µs)	3.3 (`Ts` = 5 µs)

* Simulation times obtained on a Pentium IV 2.6 GHz processor, with 512 MB of RAM

The table shows how discretizing your circuit speeds up the simulation by a factor of 1.33 for the DC drive. Using the Accelerator mode, an additional factor of 1.7 performance gain is obtained. For the AC-DC converter the Accelerator mode provides a gain of 4.4 times. For complex power electronic converter models, the Accelerator mode provides performance gains up to factors of 15.

To take full advantage of the performance enhancements made possible by converting your models to code, you must use Real-Time Workshop software

to generate stand-alone C code. You can then compile and run this code and, with xPC Target™ software, also run it on a target PC operating the xPC Target real-time kernel.

The Nonlinear Model Library

In this section...

"How to Access the Nonlinear Model Library" on page 3-16

"The Continuous Library" on page 3-16

"The Discrete Library" on page 3-17

"The Phasors Library" on page 3-17

"The Switch Current Source Library" on page 3-17

"Limitations of the Nonlinear Models" on page 3-17

"Modifying the Nonlinear Models of the powerlib_models Library" on page 3-18

How to Access the Nonlinear Model Library

The building blocks used to assemble the Simulink® model of the nonlinear circuit are stored in a library named **powerlib_models**. You do not normally need to work with the **powerlib_models** library. You can access that library by entering `powerlib_models` in the MATLAB® Command Window.

The Continuous Library

The Continuous library contains blocks that use voltage inputs (output of the state-space model of the linear circuit) and their current output is fed into the state-space model. For complex models, such as electrical machines requiring several inputs and outputs, vectorized signals are used. Useful internal signals are also returned by most of the models in a measurement output vector `m`.

For example, the Asynchronous Machine model is stored in two blocks named asynchronous_machine (electrical model) and ASM_mechanics (mechanical model). The electrical model uses as inputs a vector of four voltages, two rotor voltages and two stator voltages, respectively: (VabR, VbcR, VabS, VbcS). It returns a vector of four currents, two rotor currents and two stator currents, respectively: (IaR, IbR, IaS, IbS). The model also returns a measurement output vector of 20 signals. When the Asynchronous Machine block is used from the **powerlib** library, this measurement output vector

is accessible through the `m` output of the machine icon. You can get details on the model inputs and outputs from the documentation of **powerlib** and **powerlib_models** block icons.

The Discrete Library

The Discrete library contains the discrete versions of the continuous models described above.

The Phasors Library

The Phasors library contains the phasor versions of some of the continuous models described above. See Chapter 1, “Getting Started” for more details on the phasor simulation.

The Switch Current Source Library

The Switch Current Source library contains models of power electronic devices with an internal inductance Lon > 0. Al these continuous models are simulated by a current source external to the linear circuit. These devices are the diode, thyristor, gate-turnoff thyristor (GTO), metal-oxide-semiconductor field-effect transistor (MOSFET), and the insulated-gate-bipolar transistor (IGBT). As for electrical machines, these models use a voltage input (output of the state-space model of the linear circuit) and their current output is fed into the state-space model. All these models are vectorized.

Limitations of the Nonlinear Models

Because nonlinear models are simulated as current sources, they cannot be connected in series with inductors and their terminals cannot be left open.

If you feed a machine through an inductive source, power_analyze prompts you with an error message. You can avoid this by connecting large resistances in parallel with the source inductances or across the machine terminals.

A series RC snubber circuit is included in the model of the Breaker block and power electronics blocks. You should not have any problems if you keep these snubber circuits in service. The snubber can be changed to a single resistance by setting **Cs** to `Inf`, or to a single capacitor by setting **Rs** = `0`. To eliminate the snubber, specify **Rs** = `Inf` or **Cs** = `0`.

Modifying the Nonlinear Models of the powerlib_models Library

We do not recommend to modify the powerlib_models library. However you can use the models as a starting point to create your own version of the models. Use the technique described in “Building and Customizing Nonlinear Models” on page 2-39.

Creating Your Own Library of Models

SimPowerSystems™ software provides a variety of basic building blocks to build more complex electric blocks. Using the masking feature of the Simulink® software, you can assemble several elementary blocks from the **powerlib** library into a subsystem, build your own parameter dialog box, create the desired block icon, and place this new block in your personal library.

Chapter 1, “Getting Started” explains how to build a nonlinear model using a Voltage Measurement block and a Controlled Current Source block. The proposed examples (a nonlinear inductance and a nonlinear resistance) were relatively simple. Using the same principle, you can develop much more complex models using several controlled current sources, or even controlled voltage sources. Refer to the tutorial “Building and Customizing Nonlinear Models” on page 2-39.

Changing Your Circuit Parameters

In this section...

"Introduction" on page 3-20

"Example of MATLAB® Script Performing a Parametric Study" on page 3-20

Introduction

Each time that you change a parameter of the **powerlib** library blocks, you have to restart the simulation to evaluate the state-space model and update the parameters of the nonlinear models. However, you can change any source parameter (Magnitude, Frequency, or Phase) during the simulation. The modification takes place as soon as you apply the modification or close the source block menu.

As for the Simulink® blocks, all the **powerlib** library block parameters that you specify in the dialog box can contain MATLAB® expressions using symbolic variable names. Before running the simulation, you must assign a value to each of these variables in your MATLAB workspace. This allows you to perform parametric studies by changing the parameter values in a MATLAB script.

Example of MATLAB® Script Performing a Parametric Study

Suppose that you want to perform a parametric study in a circuit named `my_circuit` to find the impact of varying an inductance on switching transients. You want to find the highest overvoltage and the inductance value for which it occurred.

The inductance value of one of the blocks contains variable `L1`, which should be defined in your workspace. `L1` is varied in 10 steps from 10 mH to 100 mH and the values to be tested are saved in a vector, `L1_vec`. The voltage waveform to be analyzed is stored in a ToWorkspace block in array format with `V1` variable name.

You can write a MATLAB M-file that loops on the 10 inductance values and displays the worst case.

```
L1_vec= (10:10:100)*1e-3; % 10 inductances values 10/100 mH
V1_max=0;
for i=1:10
 L1=L1_vec(i);
 fprintf('Test No %d L1= %g H\n', i, L1);
 sim('my_circuit'); % performs simulation
 % memorize worst case
 if max(abs(V1))>V1_max,
  imax=i;
  V1_max=max(abs(V1));
 end
end

fprintf('Maximum overvoltage= %g V occured for L1=%g H\n',
V1_max, L1_vec(imax));
```

4

Systems with Electric Drives

About the Electric Drives Library (p. 4-3)	Presentation of the Electric Drives library: its contents and the advantages for the user
Getting Started with Electric Drives Library (p. 4-5)	Information on electric drive fundamentals, including the general layout and features of the library's graphical user interface (GUI)
Simulating a DC Motor Drive (p. 4-14)	Step-by-step example showing how to simulate a DC drive model
Simulating an AC Motor Drive (p. 4-39)	Step-by-step example showing how to simulate an AC drive model
Mechanical Models (p. 4-70)	Descriptions of the Mechanical Shaft block and the Speed Reducer block
Mechanical Coupling of Two Motor Drives (p. 4-72)	Study of the mechanical coupling of the AC4 (DTC three-phase induction motor-based drive) and DC2 (single-phase dual-converter DC motor drive) blocks
Winding Machine (p. 4-79)	Study of a winding machine driven by the DC3 (two-quadrant three-phase rectifier DC motor drive) block

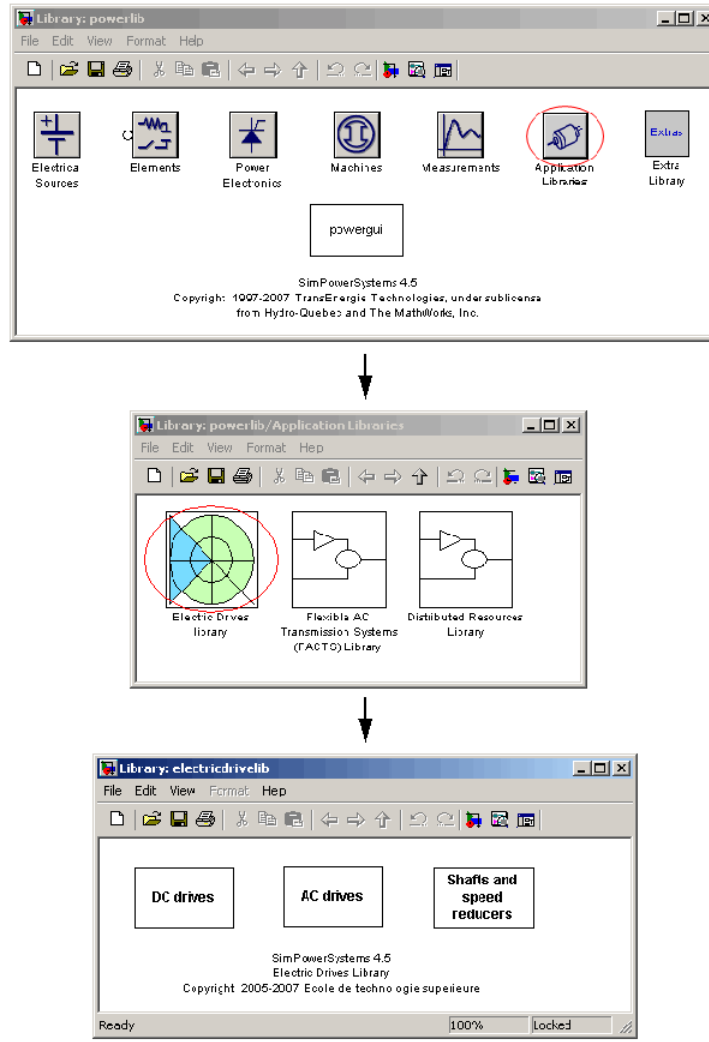
Robot Axis Control Using Brushless DC Motor Drive (p. 4-87)	Study of a six-degrees-of-freedom robot driven by the AC6 (brushless DC motor drive) blocks
Building Your Own Drive (p. 4-98)	Study of how to build your own motor drive model according to your specific requirements

About the Electric Drives Library

The Electric Drives library is designed for engineers from many disciplines who want to incorporate easily and accurately electric drives in the simulation of their systems. A specialized interface presents the parameters of the selected drive in a system-look topology, thereby simplifying the adjustments users may want to bring to the default values. Then they can seamlessly use any other toolboxes or blocksets to analyze the time or frequency behavior of the electric drive interacting with its system. The library is most helpful when a powerful drive has to be carefully maneuvered without ignoring the operating limits of the load on one side and of the power source on the other side. A good example is the electric drive system of a hybrid car that can switch in milliseconds from driving the wheels to recharging the batteries when the brakes are engaged.

Engineers and scientists can work readily with the library. The library has seven typical direct current (DC) drives used in industries and transportation systems, six alternating current (AC) drives providing more efficient and versatile motors from traction to positioning devices, and shaft and speed reducer models useful for connecting to the motor a model of load made of Simulink® blocks. An added value of the library is parameters that assure the validity of the motor, the power converters, and the control system. When designing the library, particular attention was devoted to the motor models by comparing the models' behavior to the published data of the major manufacturers. Numerous examples, demos or case studies of typical drives are supplied with the library. Hopefully, typical user systems are similar to these analyzed systems, thereby saving time in building the practical system and supplying a known reference point in the analysis.

To access the Electric Drives library, open the SimPowerSystems™ main library, **powerlib**, then double-click the **Applications Libraries** icon. A new window opens containing the icons for the Electric Drives library, FACTS library, and DR library, as shown in the following illustration.



Accessing the Electric Drives Library

Getting Started with Electric Drives Library

In this section...

"What Is an Electric Drive?" on page 4-5
"Three Main Components of an Electric Drive" on page 4-5
"Multiquadrant Operation" on page 4-8
"Average-Value Models" on page 4-9
"User Interface" on page 4-10
"General Layout of the Library's GUIs" on page 4-10
"Features of the New GUIs" on page 4-11
"Advanced Usage" on page 4-13

What Is an Electric Drive?

An electric drive is a system that performs the conversion of electric energy into mechanical energy at adjustable speeds. This is the reason why an electric drive is also called adjustable speed drive (ASD). Moreover, the electric drive, as we will see later, always contains a current (or torque) regulation in order to provide safe current control for the motor. Therefore, the electric drive torque/speed is able to match in steady state the torque/speed characteristics of any mechanical load. This motor to mechanical load match means better energy efficiency and leads to lower energy costs. In addition, during the transient period of acceleration and deceleration, the electric drive provides fast dynamics and allows soft starts and stops, for instance.

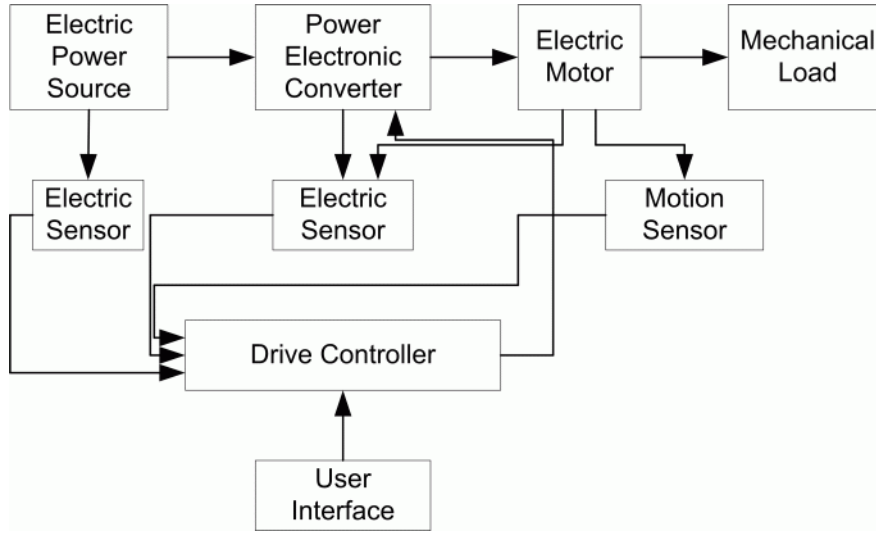
A growing number of applications require that the torque and speed must vary to match the mechanical load. Electric transportation means, elevators, computer disk drives, machine tools, and robots are examples of high-performance applications where the desired motion versus time profile must be tracked very precisely. Pumps, fans, conveyers, and HVAC (heat, ventilation, air conditioners) are examples of moderate performance applications where variable-speed operation means energy savings.

Three Main Components of an Electric Drive

An electric drive has three main components:

- The electric motor
- The power electronic converter
- The drive controller

The following figure shows the basic topology of an electric drive. Beside the three main components, the figure shows an electric power source, a mechanical load, electric and motion sensors, and a user interface.



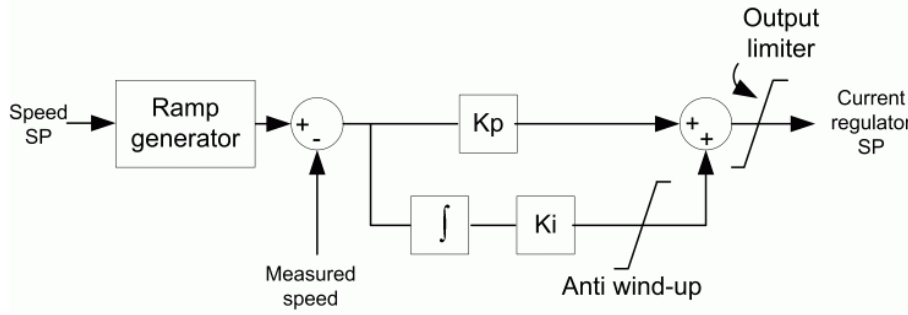
Electric Drive Basic Topology

The motor used in an electric drive is either a direct current (DC) motor or an alternating current (AC) motor. The type of motor used defines the electric drive's classification into DC motor drives and AC motor drives. The ease of producing a variable DC voltage source for a wide range of speed control made the DC motor drive the favorite electric drive up to the 1960s. Then the advances of power electronics combined with the remarkable evolution of microprocessor-based controls paved the way to the AC motor drive's expansion. In the 1990s, the AC motor drives took over the high-performance variable-speed applications.

The power electronic converter produces variable AC voltage and frequency from the electric power source. There are many types of converters

depending on the type of electric drive. The DC motor drives are based on phase-controlled rectifiers (AC-DC converters) or on choppers (DC-DC converters), while the AC motor drives use inverters (DC-AC converters) or cyclo converters (AC-AC converters). The basic component of all the power electronic converters is the electronic switch, which is either semicontrolled (controllable on-state), as in the case of the thyristor, or fully controllable (controllable on-state and off-state), as in the cases of the IGBT (insulated gate bipolar transistor) and the GTO (gate turn off thyristor) blocks. The controllable feature of the electronic switch is what allows the converter to produce the variable AC voltage and frequency.

The purpose of the drive controller is essentially to convert the desired drive torque/speed profile into triggering pulses for the electronic power converter, taking into account various drive variables (currents, speed, etc.) fed back by the sensors. To accomplish this, the controller is based first on a current (or torque) regulator. The current regulator is mandatory because, as mentioned previously, it protects the motor by precisely controlling the motor currents. The set point (SP) of this regulator can be supplied externally if the drive is in torque regulation mode, or internally by a speed regulator if the drive is in speed regulation mode. In the Electric Drives library, the speed regulator is in series with the current regulator and is based on a PI controller that has three important features. First, the SP rate of change is limited so that the desired speed ramps gradually to the SP, in order to avoid sudden step changes. Second, the speed regulator output that is the SP for the current regulator is limited by maximum and minimum ceilings. Finally, the integral term is also limited in order to avoid wind-up. The following figure shows a block diagram of a PI controller-based speed controller.

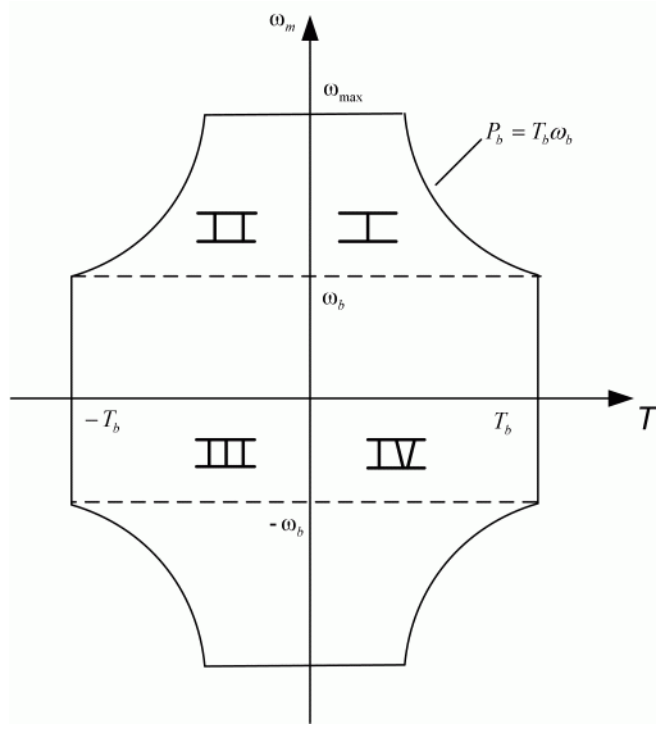


Block Diagram of the PI Controller-Based Speed Regulator

Multiquadrant Operation

For each electric drive application, the mechanical load to be driven has a specific set of requirements. The torque/speed possibilities of the electric drive can be represented as a speed versus torque graph consisting of four quadrants. In the first quadrant, the electric torque and the speed signs are both positive, indicating forward motoring since the electric torque is in the direction of motion. In the second quadrant, the electric torque sign is negative and the speed sign is positive, indicating forward braking since the electric torque is opposite to the direction of motion. In the third quadrant, the electric torque and speed signs are both negative, indicating reverse motoring. In the fourth quadrant, the electric torque sign is positive and the speed is negative, indicating reverse braking. The drive braking is handled either by a braking chopper (dynamic braking) or by bidirectional power flow (regenerative braking).

The following figure illustrates the four-quadrants operating region of an electric drive. Each quadrant has a constant torque region from 0 to +/- nominal speed ω_b and a region where the torque decreases inversely with the speed from ω_b to the maximum speed ω_{max}. This second region is a constant power region and is obtained by decreasing the motor magnetic flux.



Four-Quadrant Operation of an Electric Drive

Average-Value Models

The AC and DC library allows two levels of simulation, detailed simulations or average-value simulations. The detailed simulations use the Universal Bridge block to represent the detailed behavior of rectifier and inverter controlled drives. This simulation level requires small simulation time steps to achieve correct representation of the high frequency electrical signal components of the drives.

The average-value simulations use average-value models of the power converters. When simulating in average-value mode, the electrical input and output currents and voltages of the power converters driving the electrical motors represent the average values of the real-life currents and voltages. By doing so, the high frequency components are not represented and the simulations can use much bigger time steps. Each power converter

average-value model is described in the detailed documentation associated with each DC or AC model type. The time step used in a drive at average-value level can usually be increased up to the smallest controller sampling time used in a model. For example, if a drive uses a 20 µs time step for the current loop and a 100 µs time step for the speed loop, then the simulation time step in average-value mode can be increased up to 20 µs. Simulation time step guidelines are given in the detailed documentation of each model.

Switching between the detailed simulation level and the average-value simulation level can easily be done via the new GUI associated with each model, as explained in "Selecting the Detailed or the Average-Value Model Detail Level" on page 4-12.

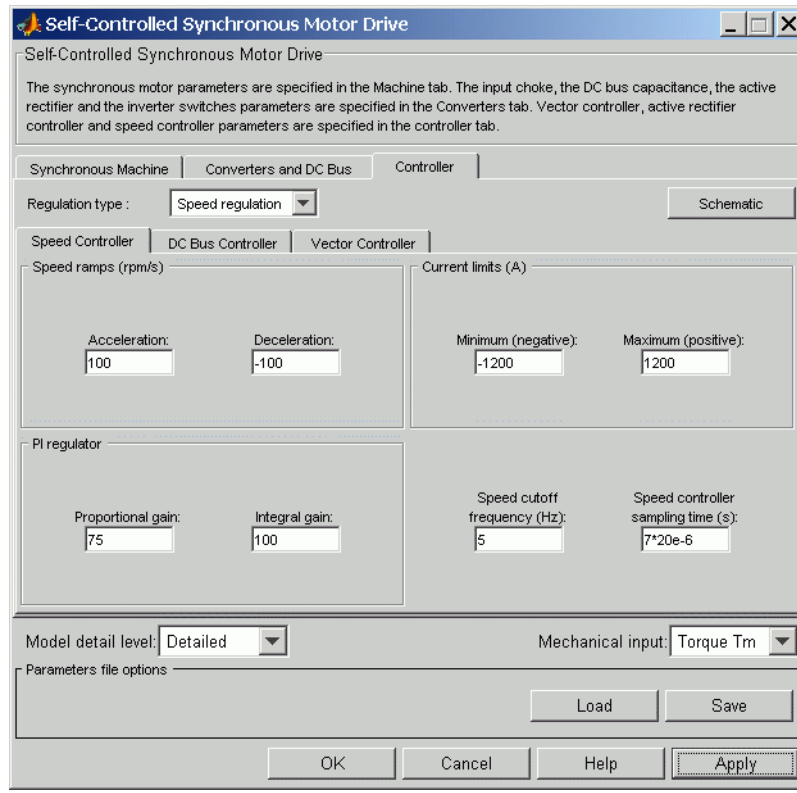
User Interface

The drive models supplied in the library are relatively complex and involve a large number of parameters. The Electric Drives library provides new GUIs for all models. The new GUIs offer all the functionality you would expect from existing Simulink® masks, plus some new features, as outlined below.

General Layout of the Library's GUIs

The general layout of the GUIs is identical to the layout of Simulink masks. A short description of the model appears at the top, parameters are entered in the middle portion, and buttons are placed at the bottom.

The parameters section is divided in three tabs at the top level, for all drive models supplied in the library. You enter parameters related to the electric machine, converters and DC bus, and controller in the first, second and third tabs, respectively. The following figure illustrates the **Self-Controlled Synchronous Motor Drive** GUI with the **Controller** tab active.



Features of the New GUIs

The new GUIs offer the same functionality as Simulink masks. You can enter as parameters numerical values, valid MATLAB® expressions, and MATLAB variables. An important difference between these GUIs and Simulink masks is that you can only enter a single value in each input field (e.g., vectors and arrays are not allowed).

The new features (with respect to Simulink masks) are outlined below.

Parameter Validation

The new GUIs are designed to signal invalid parameters as early as possible. Hence, if you enter an invalid constant (for example 1.2.3 or –2) in a drive

model's GUI, an error will be flagged as soon as you move away from the invalid parameter (for instance if you try to change another parameter in the GUI). Variables are treated slightly differently. If you enter a variable name that has yet not been defined in the MATLAB workspace, parameter validation is deferred until you start the simulation of the diagram that contains the model.

Saving Parameters in a File

You can see in the previous figure that the GUIs include the usual buttons found at the bottom of Simulink masks, plus two new ones, **Load** and **Save**. The **Save** button enables you to save in a file the complete set of parameters if the GUI. The format of the file is the standard MATLAB binary (`.MAT`) format. The **Load** button enables you to recover a previously saved set of parameters for a given drive type (e.g., AC1, DC2, etc.). When you load a set of parameters, the drive type of the saved parameters is compared to the drive type of the model you are loading the parameters into, to ensure that you are loading parameters compatible with the model.

When you use the **Load** button, the dialog that appears will point to the directory in your MATLAB installation that contains the standard sets of parameters supplied for all the drives in the library.

However, when you use the **Save** button, the dialog that appears will point to the current working directory in the MATLAB workspace.

Displaying the Controller's Schematic

There is a **Schematic** button in the top right corner of the controller tab in all drive models. When you click this button, the control schematic of the drive model will appear in a new window.

Selecting the Detailed or the Average-Value Model Detail Level

You can select the detailed or the average-value simulation level by using the **Model detail level** menu located in the lower part of the GUI. Remember to modify the simulation time step with respect to the model detail level used.

Selecting the Mechanical Input

You can select either the load torque or the motor speed as mechanical input. Use the **Mechanical input** menu located in the lower part of the GUI. Note that if you select the motor speed as mechanical input, the internal mechanical system is not used and the inertia and viscous friction parameters are not displayed. You have to include these parameters in the external mechanical system.

Advanced Usage

It is important to note that if you decide to disable the link between a drive model and its library, the new GUI will no longer be available for that particular instance of the model. Double-clicking on the model in such conditions will simply open the subsystem, as in the case of an unmasked Simulink subsystem. You can then enter parameters in the individual masks of subsystems that the drive model is composed from.

Note that in the standard (e.g., linked) situation, these masks are disabled to ensure that the top-level GUI is the only place where parameters can be changed. This is required to ensure proper synchronization of the two levels of user interfaces (e.g., the new GUI and the underlying subsystems' masks).

Simulating a DC Motor Drive

In this section...

"Introduction" on page 4-14

"Regenerative Braking" on page 4-15

"Example: Thyristor Converter-Based DC Motor Drive" on page 4-15

Introduction

In this section, you will learn how to use the DC drive models of the Electric Drives library. First, we will specify the types of motor, converters, and controllers used in the seven DC drive models of the library, designated DC1 to DC7. These seven models are based on the DC brush motor in the Electric Drives library. As in any electric motor, the DC brush motor has two main parts, the stator (fixed) part and the rotor (movable) part. The DC brush motor also has two types of windings, the excitation or field winding and the armature winding. As its name implies, the field winding is used to produce a magnetic excitation field in the motor whereas the armature coils carry the induced motor current. Since the time constant (L/R) of the armature circuit is much smaller than that of the field winding, controlling speed by changing armature voltage is quicker than changing the field voltage. Therefore the excitation field is fed from a constant DC voltage source while the armature windings are fed by a variable DC source. The latter source is produced by a phase-controlled thyristor converter for the DC1 to DC4 models and by a transistor chopper for the DC5, DC6, and DC7 models. The thyristor converter is fed by a single-phase AC source in the cases of DC1 and DC2 and by a three-phase AC source in the cases of DC3 and DC4. Finally, the seven DC models can work in various sets of quadrants. All these possibilities are summarized in the following table.

DC Models

Model	Type of Converter	Operation Quadrants
DC1	Single-phase thyristor converter	I-II

DC Models (Continued)

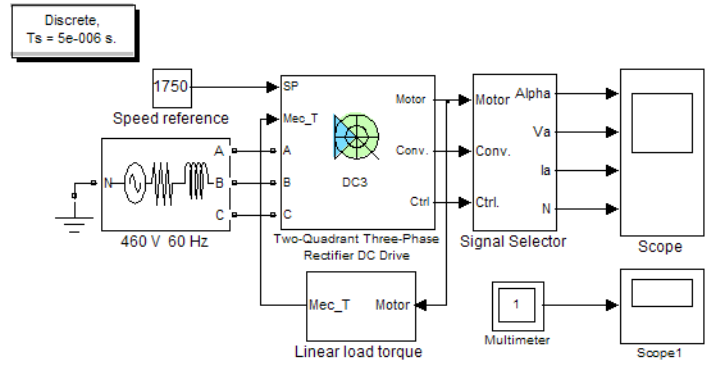
Model	Type of Converter	Operation Quadrants
DC2	Single-phase thyristor converter	I-II-III-IV
DC3	Three-phase thyristor converter	I-II
DC4	Three-phase thyristor converter	I-II-III-IV
DC5	Chopper	I
DC6	Chopper	I-II
DC7	Chopper	I-II-III-IV

Regenerative Braking

Operation in quadrants II and IV corresponds to forward and reverse braking, respectively. For the DC models of the Electric Drives library, this braking is regenerative, meaning that the kinetic energy of the motor-load system is converted to electric energy and returned to the power source. This bidirectional power flow is obtained by inverting the motor's connections when the current becomes null (DC1 and DC3) or by the use of a second converter (DC2 and DC4). Both methods allow inverting the motor current in order to create an electric torque opposite to the direction of motion. The chopper-fed DC drive models (DC5, DC6, DC7) produce regenerative braking in similar fashions.

Example: Thyristor Converter-Based DC Motor Drive

In this example, you will build and simulate the simple thyristor converter-based DC motor drive shown in Thyristor Converter-Based DC Motor Drive Example Circuit on page 4-16.



Thyristor Converter-Based DC Motor Drive Example Circuit

This step-by-step example illustrates the use of the DC3 model with a 200 hp DC motor parameter set during speed regulation. The DC3 block models a two-quadrant three-phase thyristor converter drive. During this example, the motor will be connected to a load and driven to its 1750 rpm nominal speed.

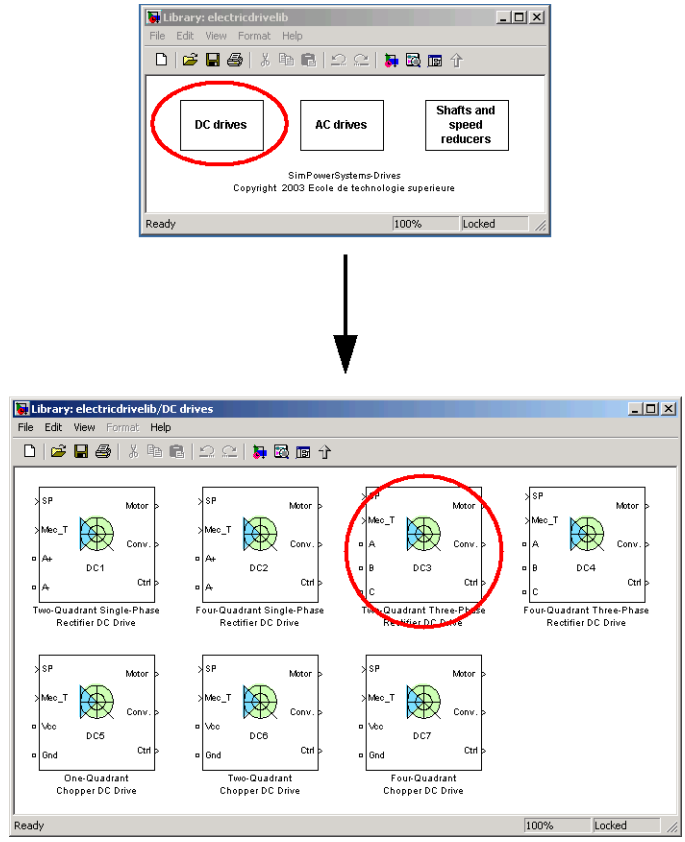
In this tutorial, you learn about

- "Getting the DC3 Model from the Drives Library" on page 4-17
- "Connecting the DC3 Model to a Voltage Source" on page 4-18
- "Connecting the DC3 Model to a Mechanical Load" on page 4-20
- "Defining the Set Point" on page 4-23
- "Visualizing Internal Signals" on page 4-23
- "Setting the Fixed-Step Simulation Environment" on page 4-26
- "Setting the High Power Drive Parameter Set" on page 4-27
- "Setting the Motor Inertia Value" on page 4-29
- "Setting the DC3 Controller Parameters and Simulation Results" on page 4-30

Getting the DC3 Model from the Drives Library

1 Open a new window and save it as `DC_example`.

2 Open the SimPowerSystems™ Electric Drives library. You can open the library by typing `electricdrivelib` in the MATLAB® Command Window or by using the Simulink® library browser. The DC3 model is located inside the DC Drives library. Copy the DC3 block and drop it in the `DC_example` window.



DC3 Model Inside the Electric Drives Library

Connecting the DC3 Model to a Voltage Source

All models of the library have three types of inputs: the electrical power inputs, the speed or torque set point input (SP), and the mechanical torque input (Mec_T). Because the DC3 model is a three-phase drive, it presents three electrical inputs: A, B, and C. In order for the DC3 model to work, you must now connect those inputs to a proper voltage source:

1. Open the Electrical Sources library and copy the 3-Phase Source block into your circuit. Connect the voltage source outputs A, B, and C to the DC3 A, B, and C inputs, respectively.

2. Open the Connectors library and copy the Ground (output) block into `DC_example`. Connect the ground output to the 3-Phase Source neutral point N.

   In this example, you are driving a 200 hp DC motor of 500 V nominal armature voltage. The mean output voltage $\hat{V}_{out}$ of a three-phase thyristor rectifier bridge is given by

   $$\hat{V}_{out} = \frac{3\sqrt{2} \cdot V_{l,rms}}{\pi} \cdot \cos\alpha$$

   where $V_{l,rms}$ is the phase-to-phase rms voltage value of the three-phase voltage source and α is the firing angle value of the thyristors. For better voltage control, a lower firing angle limit is usually imposed, and the maximum mean output voltage available from the rectifier bridge is thus given by

   $$\hat{V}_{out,max} = \frac{3\sqrt{2} \cdot V_{l,rms}}{\pi} \cdot \cos\alpha_{min}$$

   where α_{min} is the lower firing angle limit. In our case, the lower firing angle limit used in the DC3 model is 20 degrees. With such an angle value and in order to have a maximum mean output voltage value of 500 V to drive the 200 hp motor to its nominal speed, the needed phase-to-phase rms voltage value given by the preceding equation is 370 V. Assuming the drive is connected to an American electrical network, the closest standard voltage value is 460 V.

3 Set the AC source phase-to-phase rms voltage value to 460 V and the frequency to 60 Hz. Name the AC source `460 V 60 Hz`.

Note that the voltage source amplitude and frequency values needed for each drive model of the Electric Drives library can be found in the reference notes. The nominal values of the corresponding motors are also included. DC3, 200 HP Drive Specifications on page 4-19 contains the values corresponding to the DC3 200 hp model.

DC3, 200 HP Drive Specifications

Drive Input Voltage		
	Amplitude	460 V
	Frequency	60 Hz
Motor Nominal Values		
	Power	200 hp
	Speed	1750 rpm
	Voltage	500 V

In order to represent a real-life three-phase source, you must specify correct source resistance R and inductance L values. To determine these, one usually uses the short-circuit power value *Psc* and a given X/R ratio (where $X = L \cdot \omega$, ω being the angular frequency of the voltage source). As a rule of thumb, the short-circuit power absorbed by the source impedance is supposed to be at least 20 times bigger than the nominal power of the drive, and the X/R ratio is usually close to 10 for industrial plants.

The value of the source impedance Z is obtained by

$$Z = \frac{V^2}{P_{sc}}$$

where V is the phase-to-phase rms voltage value of the voltage source. For a high X/R ratio r, the source resistance R is approximately equal to

$$R = \frac{Z}{r} \tag{4-1}$$

and the source inductance L to

$$L = \frac{Z}{\omega} \tag{4-2}$$

In this example, the phase-to-phase rms voltage is worth 460 V and the source frequency is 60 Hz. If we assume a short-circuit power of 25 times the nominal drive power, we find a source impedance of 0.056 Ω. For an X/R ratio of 10, using Equation 4-1 and Equation 4-2, we find a resistance value of 0.0056 Ω and an inductance value of 0.15 mH.

4 Set the AC source resistance value to 0.0056 Ωand the inductance to 0.15 mH.

Connecting the DC3 Model to a Mechanical Load

The Mec_T input represents the load torque applied to the shaft of the DC motor. If the values of the load torque and the speed have opposite signs, the acceleration torque will be the sum of the electromagnetic and load torques. Many load torques are proportional to the speed of the driven load such as represented by the equation

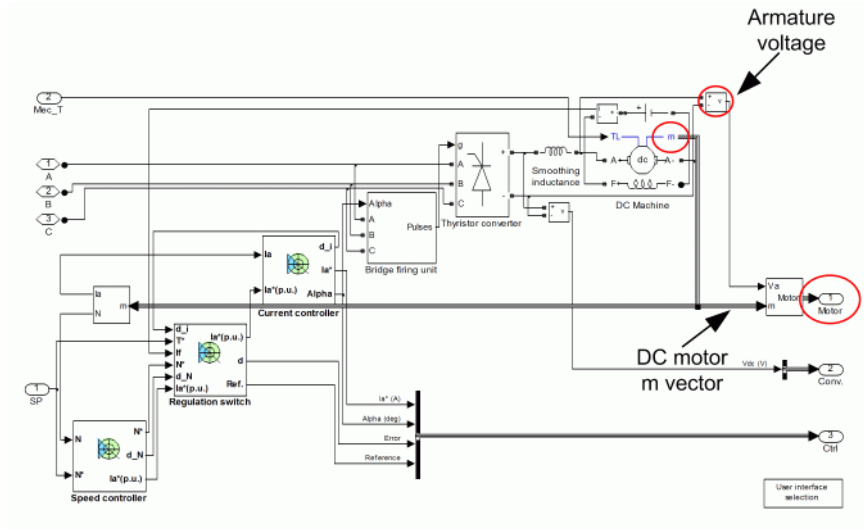
$$T_{mec} = K \cdot \omega_m = K' \cdot N_m \tag{4-3}$$

where $\omega_m$is the speed in rad/s and N the speed in rpm. You will now build such a load.

To compute this type of mechanical load torque, the speed of the DC motor is needed. This one can be obtained by using the outputs of the DC3 model. All drive models of the Electric Drives library have three output vectors: Motor, Conv., and Ctrl. The Motor vector contains all motor-related variables, the Conv. vector contains all converter voltage and current values, and the Ctrl vector contains all the regulation important values, such as the speed or torque reference signals, the speed or torque regulation error, the firing angle value, etc. All input-output descriptions are available in the reference notes of every model.

For the mechanical load torque, you can obtain the speed by using the Motor output. When you are using a DC motor, this vector is composed of the

armature voltage and of the DC motor m vector, as shown in The Motor Vector on page 4-21.



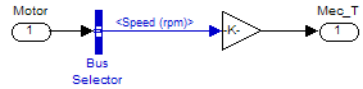
The Motor Vector

The Motor vector contains the following:

- The armature voltage
- The motor speed in rpm (the speed is converted from rad/s to rpm)
- The armature current
- The field current
- The electromagnetic torque

The speed is thus obtained by extracting the second element of the Motor vector. The speed is then multiplied by the constant K' of Equation 4-3 to obtain the load torque signal to be connected to the Mec_T input of the DC3 model:

1 Build the subsystem following and name it `Linear load torque`.



Linear Load Torque Subsystem

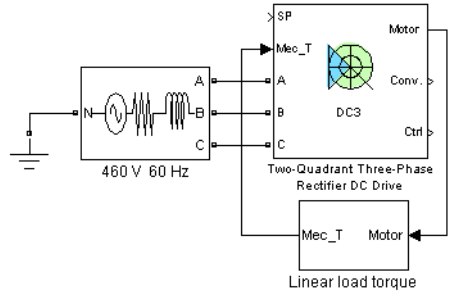
The constant K' can be computed knowing that at nominal speed, the motor should develop nominal torque. As shown in DC3, 200 HP Drive Specifications on page 4-19, the DC motor used in this simulation has a nominal speed $N_{m,n}$ of 1750 rpm. Since the nominal mechanical output power $P_{m,n}$ of the motor is 200 hp, the nominal mechanical load torque $T_{mec,n}$ can be computed following Equation 4-4 (where viscous friction is neglected)

$$P_{m,n} = T_{mec,n} \cdot \omega_{m,n} = T_n \cdot \frac{\pi \cdot N_{m,n}}{30} \qquad \textbf{(4-4)}$$

where $\omega_{m,n}$ is the nominal speed in rad/s. Using this equation, we find a nominal mechanical torque of 814 N.m. Finally Equation 4-3 gives us a K' value of 0.47.

2 Set the constant value of the Linear load torque block to `0.47`.

3 Connect the input and output of the Linear load torque block to the Motor output vector and Mec_T input of the DC3 block, respectively. Your schematic should now look like the following.



Building the Example Circuit

Defining the Set Point

The set point input of the DC3 model can either be a speed value (in rpm) or a torque value (in N.m) depending on the regulation mode (speed or torque regulation). In this example, we will set the DC3 block in speed regulation mode and drive the 200 hp DC motor to its nominal speed of 1750 rpm.

1. Open the Simulink Sources library and copy a Constant block into `DC_example`.
2. Connect the Constant block to the set point input of the DC3 model and name it `Speed reference`.
3. Set the set point to 1750 rpm.

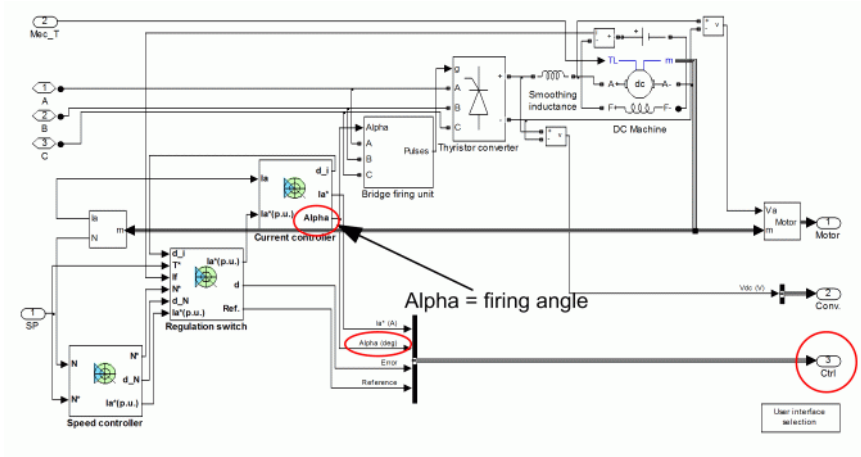
Visualizing Internal Signals

You must now use the DC3 model outputs to visualize interesting signals with a scope. Suppose you need to visualize the following signals:

- The thyristor bridge firing angle
- The motor armature voltage
- The motor armature current and reference
- The speed reference and the motor speed

Note that all model input-output descriptions can be found in the corresponding reference notes. To see which signals are connected to the DC3 outputs, select the DC3 model and use the **Edit/Look Under Mask** menu.

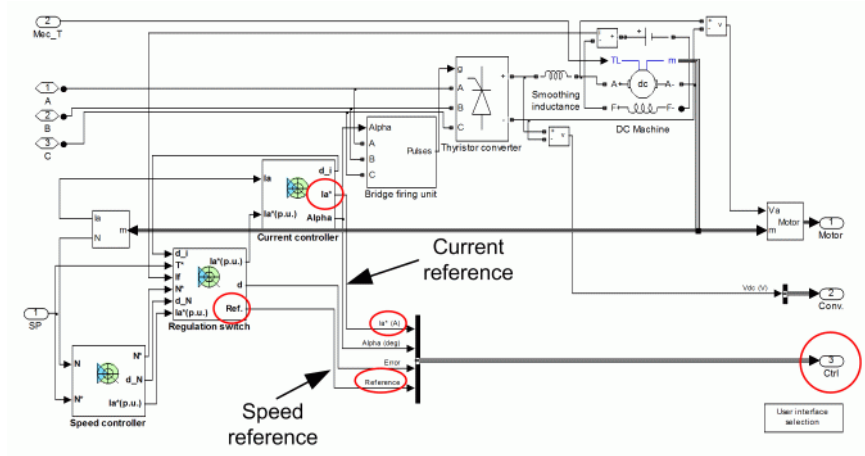
As you can see below, the firing angle is contained inside the Ctrl output vector. The firing angle Alpha (see the DC3 block reference notes) is the second element of this vector.



Location of the Firing Angle Signal Inside the Ctrl Output Vector

The Motor vector contains three of the needed signals: the armature voltage and current signals are the first and third elements, respectively (The Motor Vector on page 4-21). The speed is the second element of the Motor vector.

Finally, the current and speed reference signals are the first and fourth elements of the Ctrl vector, respectively (see the following figure). Note that the Ref. signal of the Regulation switch block would be a torque reference in torque regulation mode.

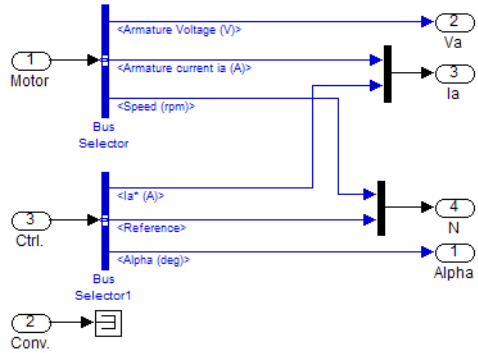


Location of the Speed Reference Signal Inside the Ctrl Output Vector

Internal bridge current and voltage signals can be extracted via the Conv. output, which is connected to a multimeter output. By clicking the Multimeter block, you can select the converter signals you want to output. Refer to the Multimeter block reference page for more information on how to use the Multimeter block.

By using Selector blocks of the Signal Routing library, you can now extract the needed signals from the three output vectors in order to visualize them:

1 Build the subsystem following in order to extract all the needed visualization signals. Name it `Signal Selector`.



Signal Selector Subsystem

2 Connect the Motor, Conv., and Ctrl outputs of the DC3 block to the Motor, Conv., and Ctrl inputs of your Signal Selector block.

3 Copy a scope to your model. It will be used to display the output signals of the Signal Selector block. Open the **Scope Parameters** dialog box. On the **General** tab, set the number of axes to `4`, the simulation time range to `auto`, and use a decimation of `20`. Clear the **Limit Data Points to last** check box on the **Data history** tab. Connect the four outputs of the Signal Selector block to the inputs of the scope.

Setting the Fixed-Step Simulation Environment

All drive models of the library are discrete models. In order to simulate your system, you must now specify the correct simulation time step and set the fixed-step solver option. Recommended sample time values for DC drives, AC drives, and mechanical models can be found in the Remarks sections of the corresponding block reference pages. The recommended sample time for the DC3 model is 5 µs. Follow these steps:

1 Open the SimPowerSystems library and copy a Powergui block into `DC_example`. Set the Sample Time to 5 µs.

Your circuit should now look like Thyristor Converter-Based DC Motor Drive Example Circuit on page 4-16.

2 Open the **Simulation/Configuration Parameters** dialog box. Select the `fixed-step`, `discrete (no continuous states)` solver option. Set the stop time to `12` seconds.

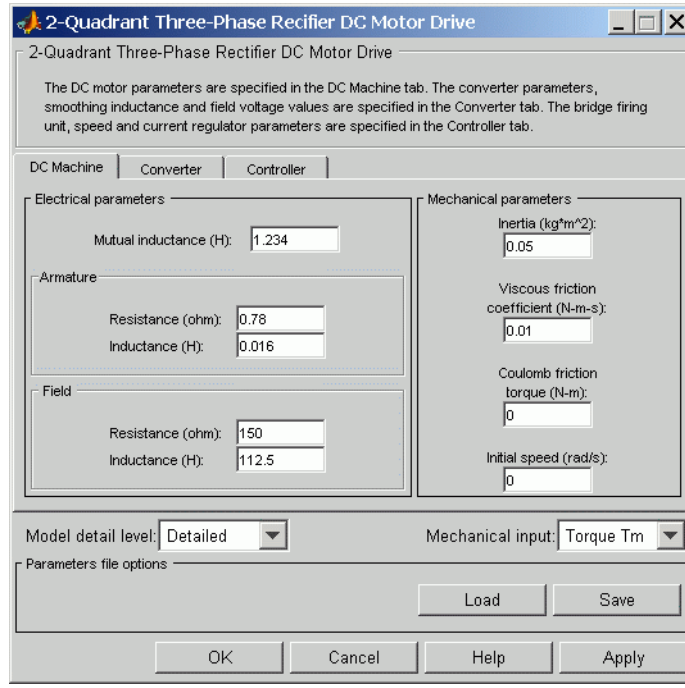
Before simulating your circuit, you must first set the correct DC3 internal parameters.

Setting the High Power Drive Parameter Set

Many models of the Electric Drives library have two sets of parameters: a low-power set and a high-power set. By default, all models are initially loaded with the low-power set. The DC3 model parameters currently loaded in `DC_example` are those of a 5 hp drive.

You will now set the high-power drive parameters, which are those of a 200 hp drive. To do this, you will use the graphical user interface:

1 Open the user interface by double-clicking the DC3 block. The interface is shown.

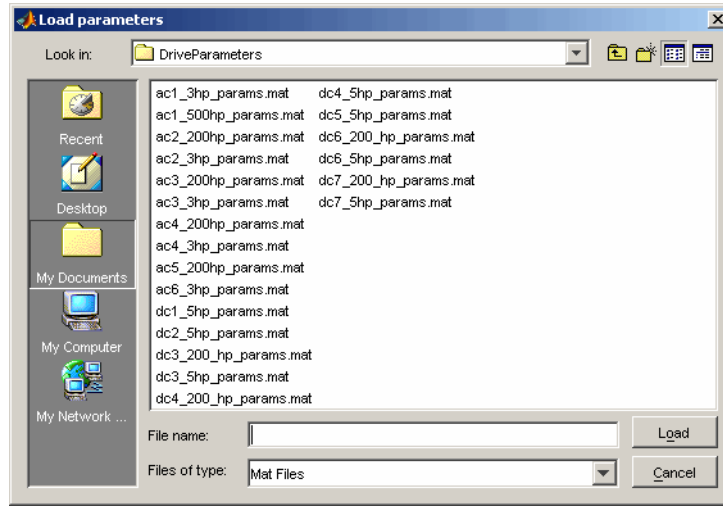


DC3 User Interface

The interface is divided following the three main parts of a drive system: the motor parameters (**DC Machine** tab), the converter parameters (**Converter** tab), and the regulation parameters of the drive controller (**Controller** tab).

2 To load the 200 hp parameters, click the **Load** button.

When you click the **Load** button, a window containing the low-power and high-power parameter files of every AC and DC model appears. These files contain all the parameters used by the graphical user interface. The name of each file is composed of the model name followed by the power value. The 200 hp version of DC3 is thus named `DC3_200hp`.



Parameters Selection Window

3 In the parameters selection window, select the `DC3_200hp.mat` file and click **Load**.

The 200 hp parameters are now loaded. Note that you can also save custom drive parameters by using the **Save** button. When you do so, your custom parameters are saved in a MAT-file format and can be reloaded at any time.

Setting the Motor Inertia Value

All default inertias of the library drives are “no-load” inertias that only represent rotor inertias. When the motor is coupled to a load, the inertia field of the **DC Machine** tab represents the combined inertias of the rotor and of the driven load. In this example, the no-load inertia of the DC3 200 hp motor is 2.5 kg*m^2. Since the drive is directly coupled to a load, you must increase this value by the inertia of the load. Suppose that the new combined inertia amounts to 15 kg*m^2.

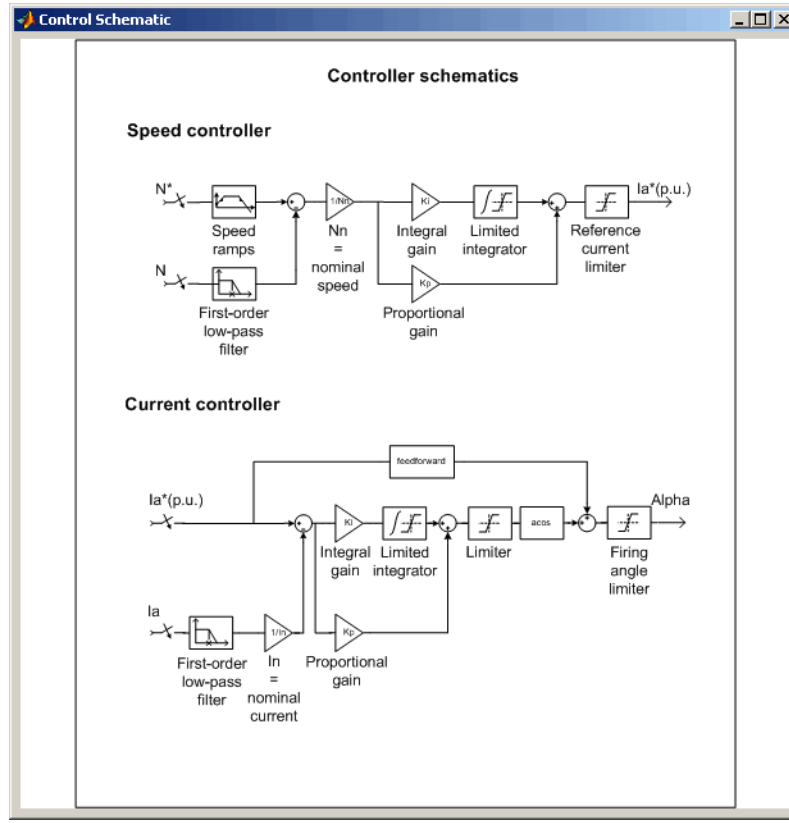
1 In the **DC Machine** section of the dialog box, change the inertia value to 15 kg*m^2.

2 Click **OK** to apply the changes and close the dialog box.

Setting the DC3 Controller Parameters and Simulation Results

The speed and current controllers of the DC3 block are both composed of a proportional-integral regulator. More details on the regulators of each drive model of the library can be found in the corresponding reference notes. To have a quick idea of the internal structure of a drive controller, a schematic is available inside the user interface of each model. Let's open the schematics related to our DC3 model:

1 Open the user interface. Click the **Controller** section and then the **Schematic button**. You should see the controller schematics of Controller Schematics of the User Interface on page 4-30.

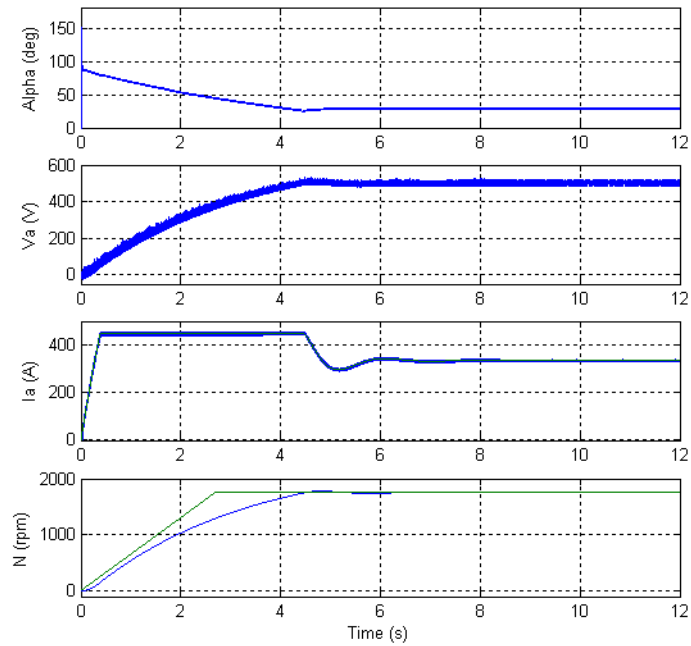


Controller Schematics of the User Interface

All default regulation parameters (speed and current controller parameters) have been trimmed for “no-load” inertias. Because the inertia has been modified, some changes are needed regarding the speed controller. The current controller should not be modified, the change of inertia having little influence on the current control.

In order to visualize the changes that need to be made, run a simulation of the present circuit.

2 Start the simulation. The simulation results visualized on the scope are shown below.



Simulation Results

Notice that the armature current follows its reference very well, but saturates at 450 A during the accelerating phase. This saturation is a result of the current controller reference limit of 1.5 pu. This results in insufficient acceleration torque, and the motor is unable to follow the

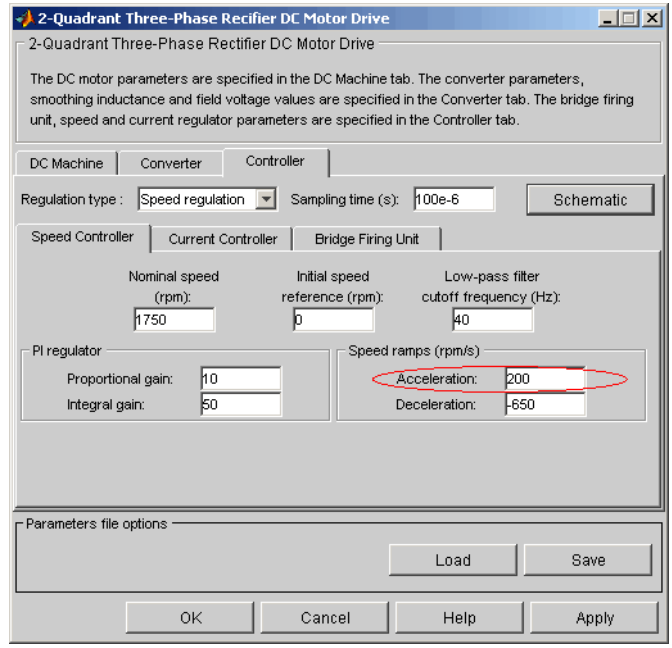
650 rpm/s default speed ramp. Since the acceleration torque cannot be increased (this would result in a burnout of the armature circuit), the speed ramp must be lowered. A guideline is to lower the speed ramp by the same amount that the inertia was increased. Indeed, following the equation below, the same torque vs. speed curve (or current vs. speed) as the default one obtained with a 2.5 kg*m^2 inertia can be obtained with the new inertia I, if the speed ramp $\dot{\omega}$ is reduced by an amount equal to the inertia increase.

$$T_{em}(\omega) = I \cdot \dot{\omega} + T_{mec} + B \cdot w = I \cdot \dot{\omega} + K' \cdot \omega + B \cdot \omega$$

The $B \cdot \omega$ term represents the viscous friction in the drive where B is the viscous friction coefficient.

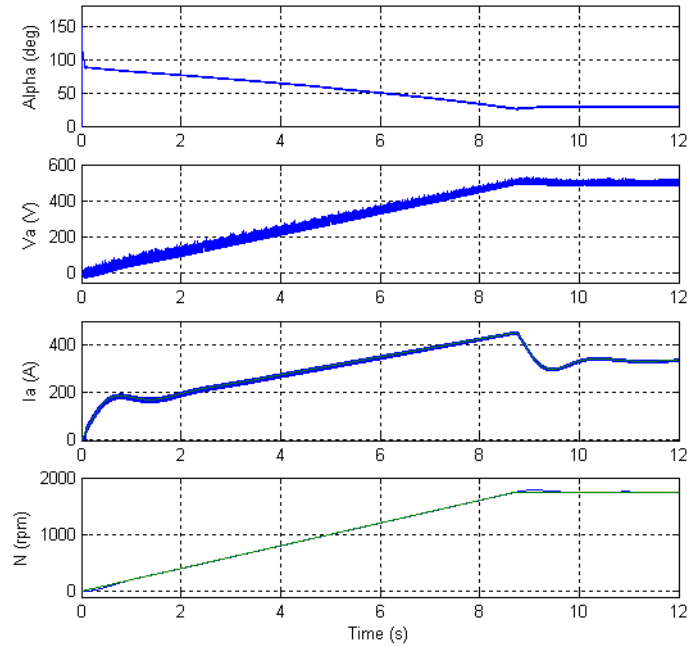
In this case, we decrease the speed ramp slightly less than the inertia increase in order to have a high enough acceleration, and set it to 200 rpm/s.

3 Open the user interface. In the **Controller** section, set the acceleration speed ramp parameter of the speed controller menu to `200 rpm/s`.



Change of the Acceleration Speed Ramp Parameter

4 Start the simulation and observe the new results on the scope.



Simulation Results with a New Acceleration Speed Ramp Value

The current regulation is very good, and no current regulator changes will be undertaken. The speed regulation is satisfactory, but some improvements could be made: the initial tracking of the speed reference could be faster, and the speed overshoot and the small speed ramping error encountered during the accelerating phase could be reduced. A modification of the proportional and integral gains of the PI speed regulator will allow you to achieve these goals:

- By increasing the proportional gain of the speed controller, you increase the controller's sensitivity by making it react a lot faster to small speed regulation errors. This allows a better initial tracking of the speed reference by a faster reaction of the current reference issued by the speed controller. This increased sensitivity also reduces the speed

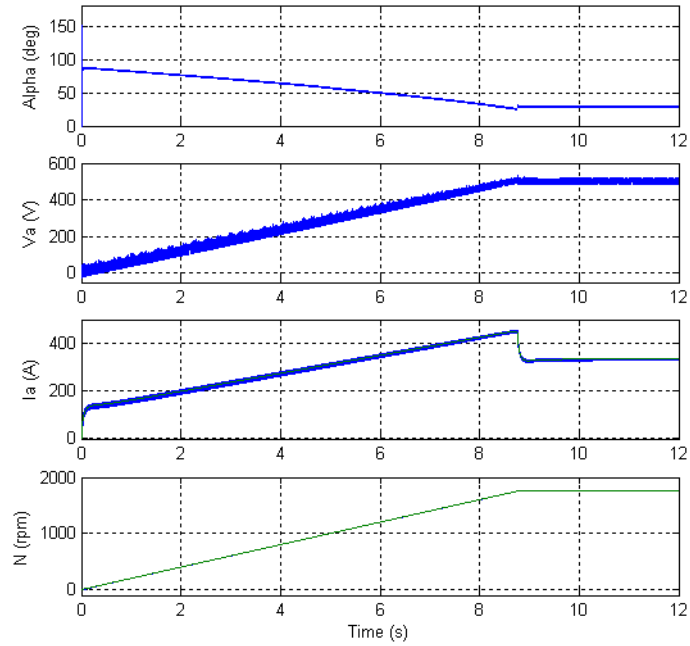
overshooting, the armature current being reduced a lot faster once the desired speed is reached.

- An increase of the integral gain will allow the motor speed to catch up with the speed reference ramp a lot faster during ramping periods. This will indeed allow a faster reaction to small speed error integral terms that occur when a signal is regulated following a ramp. The controller will react in order to diminish the speed error integral a lot faster by producing a slightly higher acceleration torque when following an acceleration ramp.

Be aware that too high an increase of the proportional and integral gains can cause instability, the controller becoming oversensitive. Too high gains can also cause current saturation. An easy way to adjust the speed controller gains is to increase them step by step and to simulate the new configuration after each change until the desired system performances are obtained (trial/error method).

Note that when the current controller has to be trimmed, a good way to achieve this is to keep the rotor still by setting a very high combined inertia value. This allows a decoupling of the electrical and mechanical parameters. You then adjust the current controller parameters until the current follows given current references perfectly. The same remarks can be made for the current regulator as those made above for speed regulation. Once the current regulator is trimmed, you can then trim the speed regulator by resetting the combined inertia to its initial value.

5 Try different speed regulator values and observe the resulting changes in the system dynamics. A proportional gain of 80 and an integral gain of 200 give very good results, as shown.



Simulation Results with Trimmed Speed Regulator Parameters

Observe that the firing angle value lowers with the speed increase in order to generate a growing converter output voltage. The converter is here working in rectifier mode, the power transiting from the AC source to the DC motor. The voltage increase allows the converter to keep feeding current to the DC motor during the acceleration phase, the armature voltage increasing proportionally with the speed. The current increase observed during this phase is due to the increasing torque opposed by the load. Around t = 8.5 s, the speed reaches its set point, and the armature current lowers to about 335 A since no more acceleration torque is needed.

Before concluding this example, notice the two first-order filters used in the speed and current controllers of Controller Schematics of the User Interface on page 4-30. These filters remove unwanted current and speed harmonics in the current and speed measurement signals. These harmonics are caused by the rectified output voltages of the three-phase full converters. The

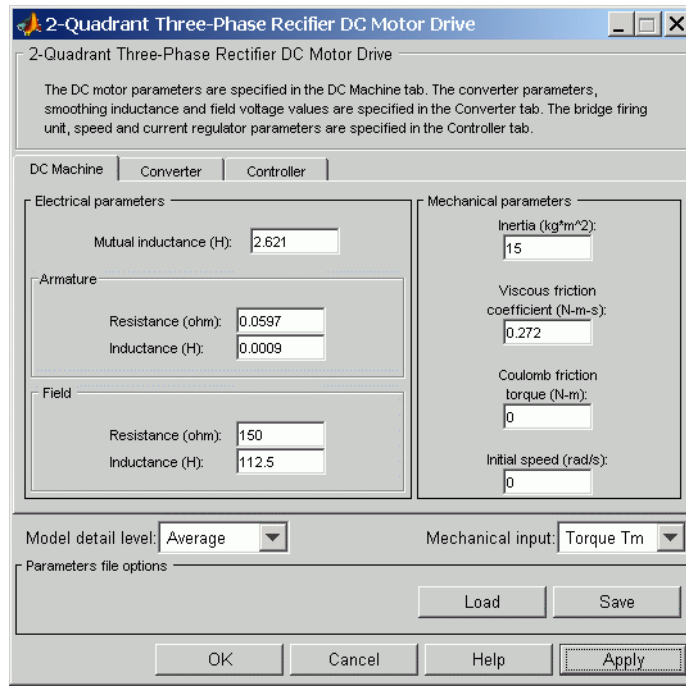
main ripple frequency introduced by a three-phase full converter is equal to six times the voltage source frequency (6th harmonic). In the case of this example, the first harmonic frequency is thus equal to 360 Hz. The cutoff frequency of the first-order filters must at least be lower than 360 Hz. Since the filters are first-order filters, the cutoff frequency must be a lot lower to have a reasonably good harmonic rejection. Keep in mind that too low a cutoff frequency can cause system instability. In the case of chopper drives like DC5, DC6, and DC7, the fundamental frequency is equal to the PWM frequency.

Simulating in Average-Value Mode

Every model can be simulated in average-value mode. In such mode, the Universal Bridge blocks used to simulate the power converters driving the motors are replaced by average-value converters. The average-value converter models used are described in the reference pages of each drive model. This lets you increase the simulation time step and thus increase simulation speed.

Use the following procedure to simulate a model in average-value mode.

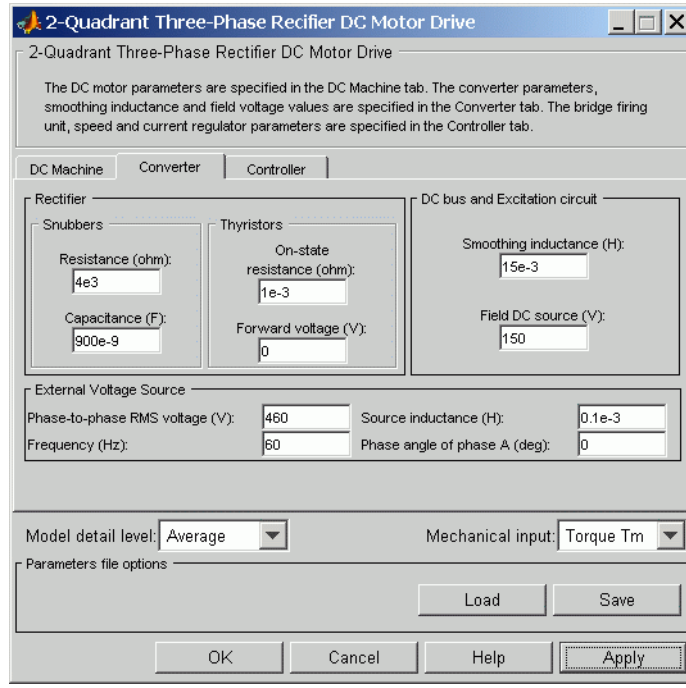
1. Open the user interface. Select the `Average` option in the **Model detail level** drop-down list located in the lower part of the user interface, as shown in the following illustration.



Selecting the Average-Value Simulation Mode

2 Select the **Converter** section.

Notice that it contains some extra parameters specific to average-value mode. These parameters affect the external voltage source and are used by the average-value rectifier. All parameters are described in the reference pages.



Extra Parameters Used in Average-Value Mode

When simulating in average-value mode, the time step can be increased in order to run faster simulations. A guideline is to increase the time step up to the smallest controller sampling time used in the model. In this case the sampling time is the same for the speed and current controllers and is equal to 100 µs.

3 Close the user interface and open the Powergui block. Set the **Sample Time** to 100 µs. Run the simulation.

Notice that the simulation time is reduced. Observe the simulation results: the rectifier output voltage and current ripples are not represented, you can see only the average value of these signals. If you later try to visualize the input current, you will only see the 60 Hz fundamental component of the detailed current.

Simulating an AC Motor Drive

In this section...

"Introduction" on page 4-39
"Dynamic Braking" on page 4-39
"Modulation Techniques" on page 4-40
"Open-Loop Volts/Hertz Control" on page 4-45
"Closed-Loop Speed Control with Slip Compensation" on page 4-46
"Flux-Oriented Control" on page 4-46
"Direct Torque Control" on page 4-49
"Example: AC Motor Drive" on page 4-50

Introduction

In this section, you will learn how to use the AC drive models of the Electric Drives library. First, we will specify the types of motors, converters, and controllers used in the six AC drive models of the library designated AC1 to AC6. The AC1, AC2, AC3, and AC4 models are based on the three-phase induction motor. This motor has a three-phase winding at the stator and a wound rotor or a squirrel-cage rotor. The squirrel-cage rotor consists of slots of conducting bars embedded in the rotor iron. The conducting bars are short-circuited together at each end of the rotor by conducting rings. The AC5 model is based on a wound rotor synchronous motor, and the AC6 model uses a permanent magnet synchronous motor. The models of these three types of motors are available in the Machines library. These AC motors are fed by a variable AC voltage and frequency produced by an inverter. The type of inverter used in the six AC drive models is a voltage source inverter (VSI) in the sense that this inverter is fed by a constant DC voltage. This constant voltage is provided by an uncontrolled diode rectifier and a capacitor (capacitive DC bus voltage).

Dynamic Braking

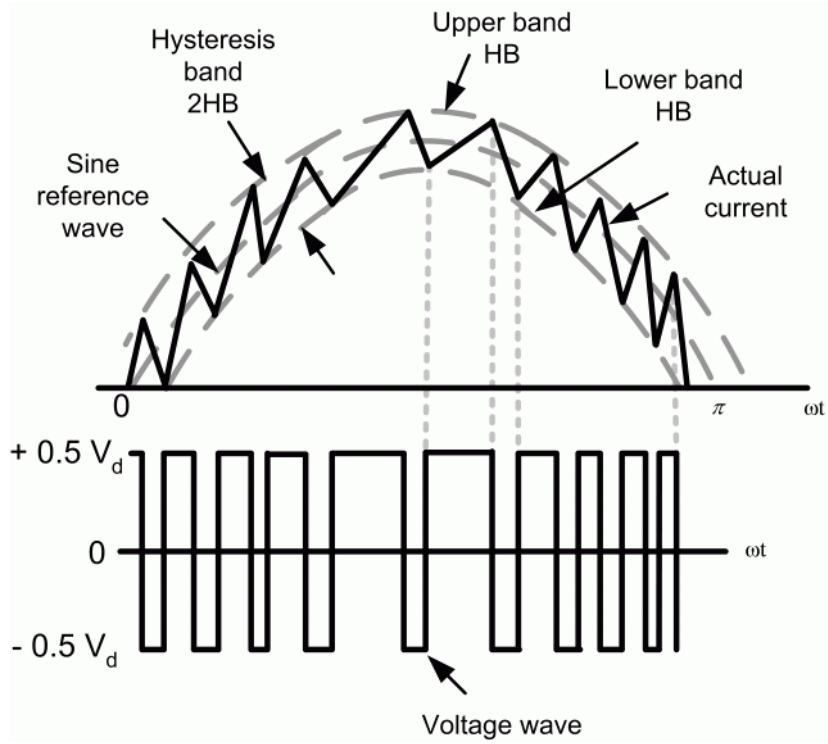
When the DC bus is provided by a diode rectifier, the drive doesn't have a bidirectional power flow capability and therefore cannot perform regenerative braking. In the AC1, AC2, AC3, AC4, and AC6 models, a braking resistor in

series with a chopper ensures the braking of the motor-load system. This braking scheme is called dynamic braking. It is placed in parallel with the DC bus in order to prevent its voltage from increasing when the motor decelerates. With dynamic braking, the kinetic energy of the motor-load system is converted into heat dissipated in the braking resistor.

Modulation Techniques

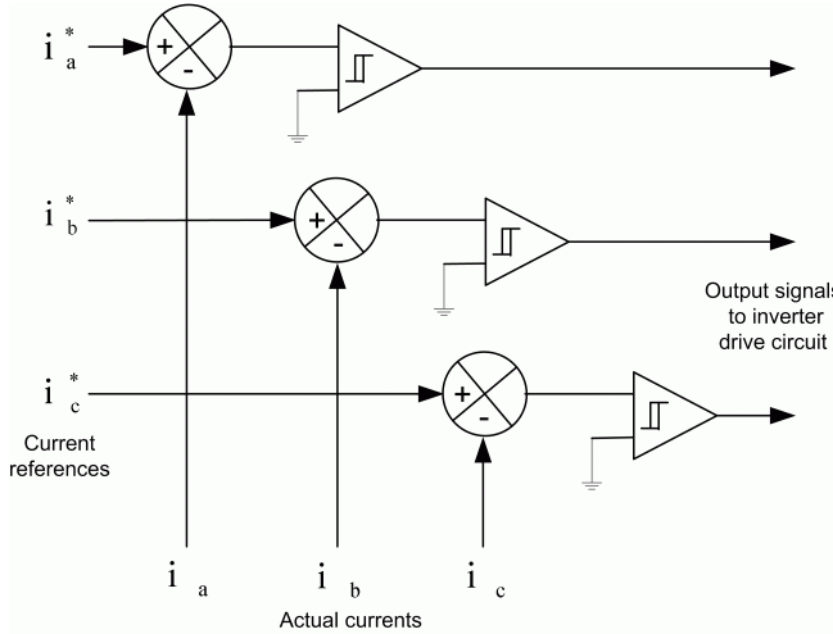
The VSI inverters used in the AC drive models of the library are based on two types of modulation, hysteresis modulation and space vector pulse width modulation (PWM).

The hysteresis modulation is a feedback current control method where the motor current tracks the reference current within a hysteresis band. The following figure shows the operation principle of the hysteresis modulation. The controller generates the sinusoidal reference current of desired magnitude and frequency that is compared with the actual motor line current. If the current exceeds the upper limit of the hysteresis band, the upper switch of the inverter arm is turned off and the lower switch is turned on. As a result, the current starts to decay. If the current crosses the lower limit of the hysteresis band, the lower switch of the inverter arm is turned off and the upper switch is turned on. As a result, the current gets back into the hysteresis band. Hence, the actual current is forced to track the reference current within the hysteresis band.



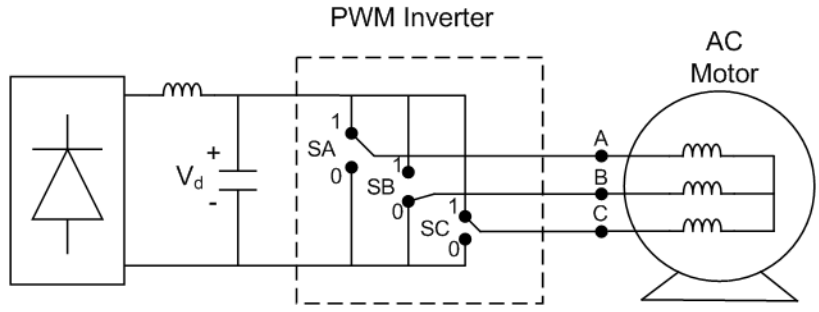
Operation Principle of Hysteresis Modulation

The following figure shows the hysteresis current control modulation scheme, consisting of three hysteresis comparators, one for each phase. This type of closed-loop PWM is used in AC3 and AC5 models.



Typical Hysteresis Current Controller

The space vector modulation technique differs from the hysteresis modulation in that there are not separate comparators used for each of the three phases. Instead, a reference voltage space vector V_s is produced as a whole, sampled at a fixed frequency, and then constructed through adequate timing of adjacent nonzero inverter voltage space vectors V_1 to V_6 and the zero voltage space vectors V_0, V_7. A simplified diagram of a VSI inverter is shown below. In this diagram, the conduction state of the three legs of the inverter is represented by three logic variables, SA, SB, and SC. A logical 1 means that the upper switch is conducting and logical 0 means that the lower switch is conducting.



Simplified Diagram of a VSI PWM Inverter

The switching of SA, SB, SC results in eight states for the inverter. The switching states and the corresponding phase to neutral voltages are summarized in Inverter Space Voltage Vectors on page 4-44. The six active vectors are an angle of 60 degrees apart and describe a hexagon boundary. The two zero vectors are at the origin.

For the location of the V_s vector shown in Inverter Space Vector Voltages on page 4-44, as an example, the way to generate the inverter output is to use the adjacent vectors V_1 and V_2 on a part-time basis to satisfy the average output demand. The voltage V_s can be resolved as

$$V_b = \frac{2}{\sqrt{3}} V_s \sin\delta$$

$$V_a = V_s \cos\delta - \frac{1}{2} V_b$$

where V_a and V_b are the components of V_s along V_1 and V_2, respectively. Considering the period T_c during which the average output should match the command, we can write the time durations of the two states 1 and 2 and the zero voltage state as

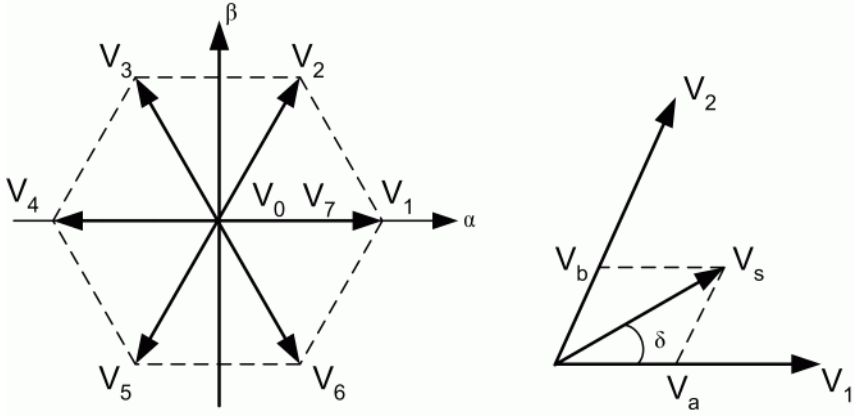
$$t_a = \frac{3}{2} \cdot \frac{V_a}{V_d} \cdot T_c$$

$$t_b = \frac{2}{3} \cdot \frac{V_b}{V_d} \cdot T_c$$

$$t_z = T_c - (t_a + t_b)$$

Inverter Space Voltage Vectors

State	SA	SB	SC	Inverter Operation	Space Voltage Vector
0	1	1	1	Freewheeling	V_0
1	1	0	0	Active	V_1
2	1	1	0	Active	V_2
3	0	1	0	Active	V_3
4	0	1	1	Active	V_4
5	0	0	1	Active	V_5
6	1	0	1	Active	V_6
7	0	0	0	Freewheeling	V_7



Inverter Space Vector Voltages

Open-Loop Volts/Hertz Control

The AC machine stator flux is equal to the stator voltage to frequency ratio since

$$\varphi(t) = \int v(t)dt$$

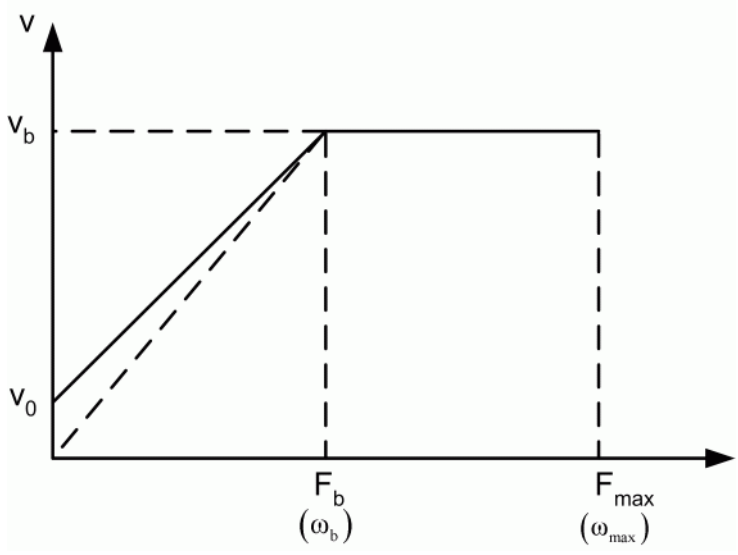
where

$$v(t) = \sqrt{2}V\sin(\omega t)$$

therefore

$$\varphi(t) = \frac{\sqrt{2}V}{\omega}\cos(\omega t)$$

Since the motor is fed with a variable AC source voltage and frequency, it is important to maintain the volts/Hz constant in the constant torque region if magnetic saturation is to be avoided. A typical volts/Hz characteristic is shown below. Notice that the straight line has a small voltage boost in order to compensate for resistance drop at low frequency. Open-loop volts/Hz control is used with low-dynamics applications such as pumps or fans where a small variation of motor speed with load is tolerable. The AC1 model is based on an open-loop volts/Hz controller.



Volts/Hz Characteristics with Compensation at Low Frequency

Closed-Loop Speed Control with Slip Compensation

In this type of control, a slip speed command is added to the measured rotor speed to produce the desired inverter frequency. A PI-based speed regulator produces the slip command. The desired inverter frequency generates the voltage command through a volts/Hz characteristic such as the one shown above. The AC2 model is based on a closed-loop speed control that uses volts/Hz and slip regulation.

Flux-Oriented Control

The construction of a DC machine is such that the field flux is perpendicular to the armature flux. Being orthogonal, these two fluxes produce no net interaction on one another. Adjusting the field current can therefore control the DC machine flux, and the torque can be controlled independently of flux by adjusting the armature current. An AC machine is not so simple because of the interactions between the stator and the rotor fields, whose orientations are not held at 90 degrees but vary with the operating conditions. You can obtain DC machine-like performance in holding a fixed and orthogonal orientation between the field and armature fields in an AC machine by orienting the

stator current with respect to the rotor flux so as to attain independently controlled flux and torque. Such a control scheme is called flux-oriented control or vector control. Vector control is applicable to both induction and synchronous motors. We will see now how it applies to induction motors.

Considering the d-q model of the induction machine in the reference frame rotating at synchronous speed ω_e,

$$V_{qs} = R_s i_{qs} + \frac{d}{dt}\varphi_{qs} + \omega_e \varphi_{ds}$$

$$V_{ds} = R_s i_{ds} + \frac{d}{dt}\varphi_{ds} - \omega_e \varphi_{qs}$$

$$0 = R_s i_{qr} + \frac{d}{dt}\varphi_{qr} + (\omega_e - \omega_r)\varphi_{dr}$$

$$0 = R_r i_{dr} + \frac{d}{dt}\varphi_{dr} - (\omega_e - \omega_r)\varphi_{qr}$$

$$T_e = 1.5\, p \frac{L_m}{L_r}(\varphi_{dr} i_{qs} - \varphi_{qr} i_{ds})$$

where

$$\varphi_{qs} = L_s i_{qs} + L_m i_{qr}$$

$$\varphi_{ds} = L_s i_{ds} + L_m i_{dr}$$

$$\varphi_{qr} = L_r i_{qr} + L_m i_{qs}$$

$$\varphi_{dr} = L_r i_{dr} + L_m i_{ds}$$

The field-oriented control implies that the i_{ds} component of the stator current would be aligned with the rotor field and the i_{qs} component would be perpendicular to i_{ds}. This can be accomplished by choosing ω_e to be the speed of the rotor flux and locking the phase of the reference frame system such that the rotor flux is aligned precisely with the d axis, resulting in

$$\varphi_{qr} = 0 \Rightarrow \frac{d}{dt}\varphi_{qr} = 0$$

and

$$\varphi_{dr} = \varphi_r$$

which implies that

$$\omega_{sl} = (\omega_e - \omega_r) = \left(\frac{L_m R_r}{\varphi_r L_r}\right) i_{qs}$$

and that

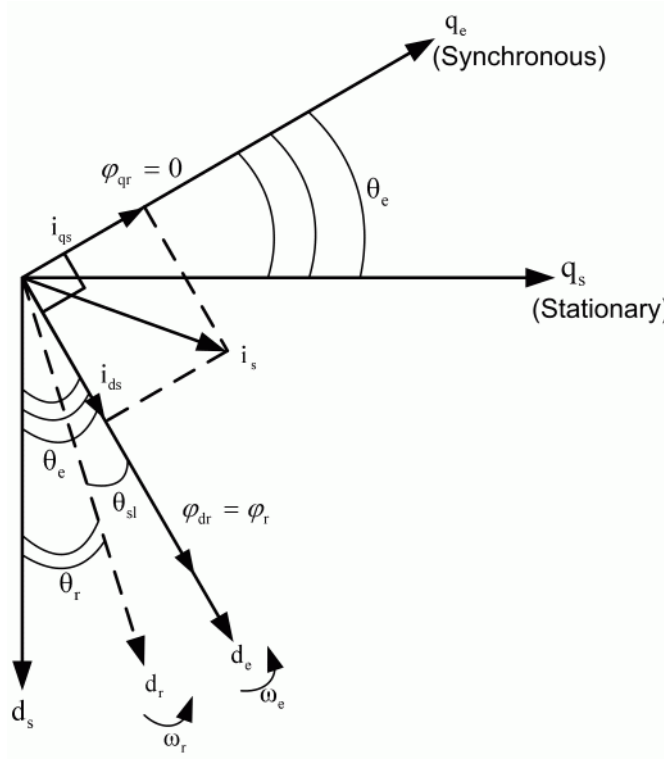
$$T_e = 1.5 p \frac{L_m}{L_r} (\varphi_r i_{qs})$$

It also follows that

$$\frac{d}{dt}\varphi_r = -\left(\frac{R_r}{L_r}\right)\varphi_r + \left(\frac{L_m R_r}{L_r}\right) i_{ds}$$

The analogy with DC machine performance is now clear. The electric torque is proportional to the i_{qs} component, whereas the relation between the flux φ_r and the i_{ds} component is given by a first-order linear transfer function with a time constant L_r/R_r.

You cannot directly measure the rotor flux orientation in a squirrel-cage rotor induction machine. It can only be estimated from terminal measurements. An alternative way is to use the slip relation derived above to estimate the flux position relative to the rotor, as shown. The latter control scheme is called indirect field-oriented control and is used in the AC3 model.



Rotor Flux Position Obtained from the Slip and Rotor Positions

Direct Torque Control

The field-oriented control is an attractive control method but it has a serious drawback: it relies heavily on precise knowledge of the motor parameters. The rotor time constant is particularly difficult to measure precisely, and to make matters worse it varies with temperature.

A more robust control method consists first in estimating the machine stator flux and electric torque in the stationary reference frame from terminal measurements. The following relations are used

$$\varphi_{ds} = \int (V_{ds} - R_s i_{ds}) dt$$

$$\varphi_{qs} = \int (V_{qs} - R_s i_{qs}) dt$$

$$\hat{\varphi}_s = \sqrt{\varphi_{ds}^2 + \varphi_{qs}^2} \angle \text{atan}\left(\frac{\varphi_{qs}}{\varphi_{ds}}\right)$$

$$T_e = 1.5 p (\varphi_{ds} i_{qs} - \varphi_{qs} i_{ds})$$

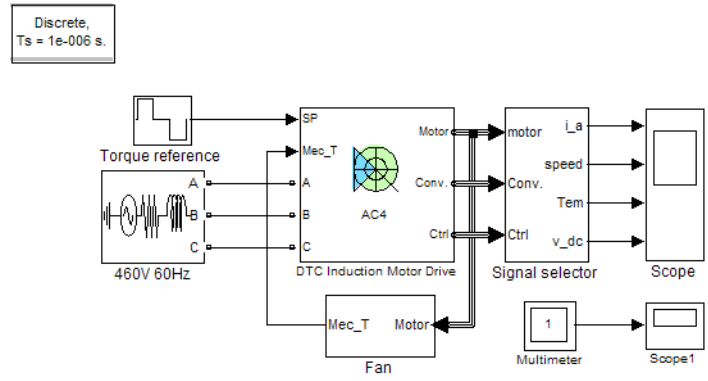
The estimated stator flux and electric torque are then controlled directly by comparing them with their respective demanded values using hysteresis comparators. The outputs of the two comparators are then used as input signals of an optimal switching table. The following table outputs the appropriate switching state for the inverter.

Switching Table of Inverter Space Vectors

H_ψ	H_{Te}	S(1)	S(2)	S(3)	S(4)	S(5)	S(6)
	1	V_2	V_3	V_4	V_5	V_6	V_1
1	0	V_0	V_7	V_0	V_7	V_0	V_7
	-1	V_6	V_1	V_2	V_3	V_4	V_5
	1	V_3	V_4	V_5	V_6	V_1	V_2
-1	0	V_7	V_0	V_7	V_0	V_7	V_0
	-1	V_5	V_6	V_1	V_2	V_3	V_4

Example: AC Motor Drive

In this example, you will build and simulate the simple induction motor drive system in Induction Motor Drive Example Circuit on page 4-51.



Induction Motor Drive Example Circuit

This step-by-step example illustrates the use of the AC4 model with a 200 hp induction motor parameter set during torque regulation. The AC4 block models a DTC drive. During this example, the motor is connected to a fan and its reaction to torque steps is simulated.

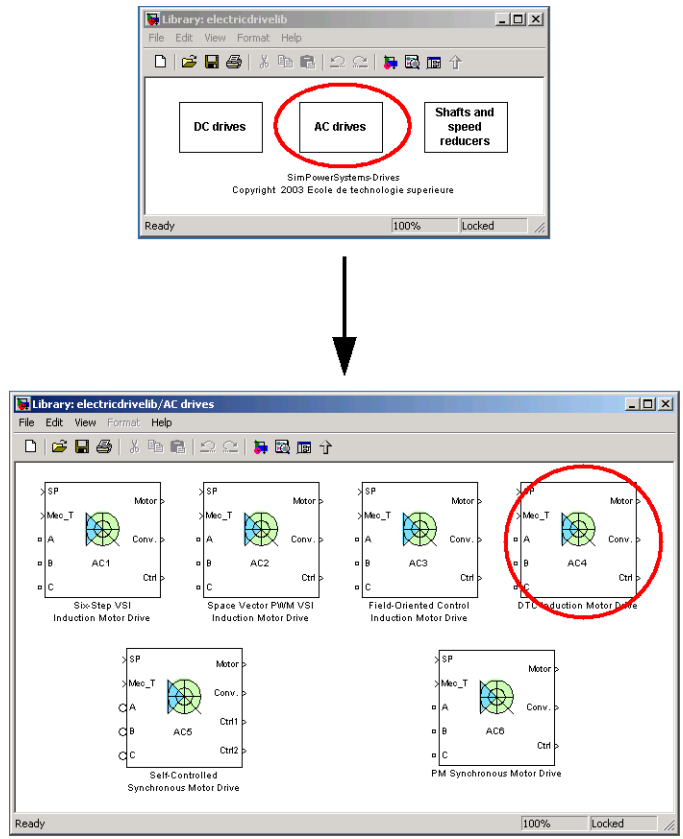
In this tutorial, you learn about

- "Getting the AC4 Model from the Electric Drives Library" on page 4-52
- "Connecting the AC4 Model to a Voltage Source" on page 4-53
- "Connecting the AC4 Model to a Mechanical Load" on page 4-54
- "Defining the Set Point" on page 4-56
- "Setting the Fixed-Step Simulation Environment" on page 4-59
- "Setting the Fixed-Step Simulation Environment" on page 4-59
- "Setting the High Power Drive Parameter Set" on page 4-60
- "Setting the Motor Inertia Value" on page 4-61
- "Setting the Braking Chopper Resistance Value" on page 4-62
- "Setting the DC Bus Initial Voltage Value" on page 4-63
- "Setting the AC4 Controller Parameters" on page 4-64

Getting the AC4 Model from the Electric Drives Library

1 Open a new window and save it as `ac_example`.

2 Open the Electric Drives library. You can open the library by typing `electricdrivelib` in the MATLAB® Command Window or by using the Simulink® library browser. The AC4 model is located inside the AC drives library. Copy the AC4 block and drop it in the `ac_example` window.



AC4 Model Inside the Drives Library

Connecting the AC4 Model to a Voltage Source

As with the DC example, you must now connect the AC4 block to a proper voltage source:

1 Open the Electrical Sources library and copy the Three-Phase Source block into your circuit. Connect the voltage source outputs A, B, and C to the AC4 A, B, and C inputs, respectively.

   In this example, we will be driving a 200 hp induction motor of 460 V nominal armature voltage and 60 Hz nominal frequency. As specified in the DC example, the voltage source amplitude and frequency values needed for each drive model of the Electric Drives library can be found in the reference notes. The nominal values of the corresponding motors are also included. The following table contains the values corresponding to the AC4 200 hp model.

AC4, 200 HP Drive Specifications

Drive Input Voltage		
	Amplitude	460 V
	Frequency	60 Hz
Motor Nominal Values		
	Power	200 hp
	Speed	1800 rpm
	Voltage	460 V

   We will thus set the AC source voltage amplitude and frequency values to 460 V and 60 Hz, respectively.

2 Set the AC source phase-to-phase rms voltage value to 460 V, and the frequency to 60 Hz. Name the AC source `460 V 60 Hz`.

   In order to represent a real-life three-phase source, you must specify correct source resistance *R* and inductance *L* values. The procedure to determine these values has been discussed above in the step-by-step example “Connecting the DC3 Model to a Voltage Source” on page 4-18.

Following this procedure, you determine a resistance value of 0.0056 Ω and an inductance value of 0.15 mH.

3 Set the AC source resistance value to 0.0056 Ω and the inductance to 0.15 mH.

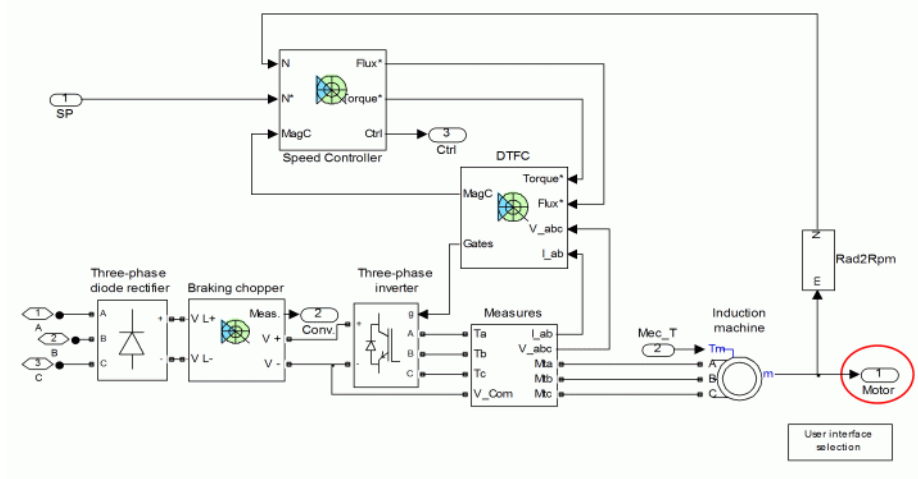
Connecting the AC4 Model to a Mechanical Load

The Mec_T input of the AC4 block represents the load torque applied to the shaft of the induction motor. In this case, the load torque is opposed by a fan. This type of torque is typically a quadratic function of the speed, as shown in Equation 4-5:

$$T_{mec} = K \cdot \omega_m^2 = K' \cdot N_m^2 \qquad \textbf{(4-5)}$$

where ω_m is the speed in rad/s and N_m is the speed in rpm. You will now build such a load.

To compute the mechanical load torque, the speed of the induction motor is needed. As discussed in the DC example, the speed value can be obtained from the Motor output vector of the AC4 model. As shown, the Motor vector is composed of the m output vector of the induction motor.



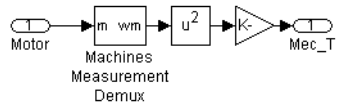
Motor Vector

In order to extract motor variable values from this vector, the Machine Measurement Demux block is needed. This block is located in the Machines library.

1 Open the Machines library. Copy the Measurement Demux block into `ac_example`. Select `asynchronous machine` in the **Machine Type** field and extract the rotor speed.

   To obtain the load torque, the speed value must be multiplied by the constant K of Equation 4-5 (the speed is here in rad/s).

2 Build the subsystem of the following figure and name it `Fan`.

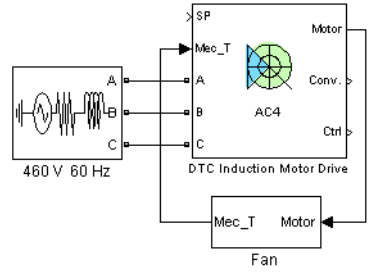


Fan Block

The constant K should be imposed so that at nominal speed, the motor develops nominal torque. This torque can be determined using Equation 4-4. Using this equation we find a nominal value of 790 N.m. Finally, Equation 4-5 gives us a K value of 0.022.

3 Set the constant value K to 0.022.

4 Connect the input and output of the Fan block to the Motor output vector and Mec_T input of the AC4 block, respectively. Your schematic should now look like the following:



Building the Example Circuit

Defining the Set Point

We must now define the set point (SP) input of AC4. During this example, we will control the induction motor's torque and impose a series of torque set points. A series of set points can be defined with the help of the Timer block.

1 Open the Control Blocks section of the Extra library and copy the Timer into `ac_example`. Connect the block to the set point input of the AC4 model and name it `Torque reference`.

The Timer block generates a signal changing at specified times. During this example, we will generate the following torque series:

Torque Set Point Series

t (s)	Torque Set Point (N.m)
0	0
0.02	600
0.25	0
0.5	-600
0.75	0

2 Set the **Time** field of the Timer block to `[0.02 0.25 0.5 0.75]`. Set the **Amplitude** field of the timer block to `[600 0 -600 0]`.

Visualizing Internal Signals

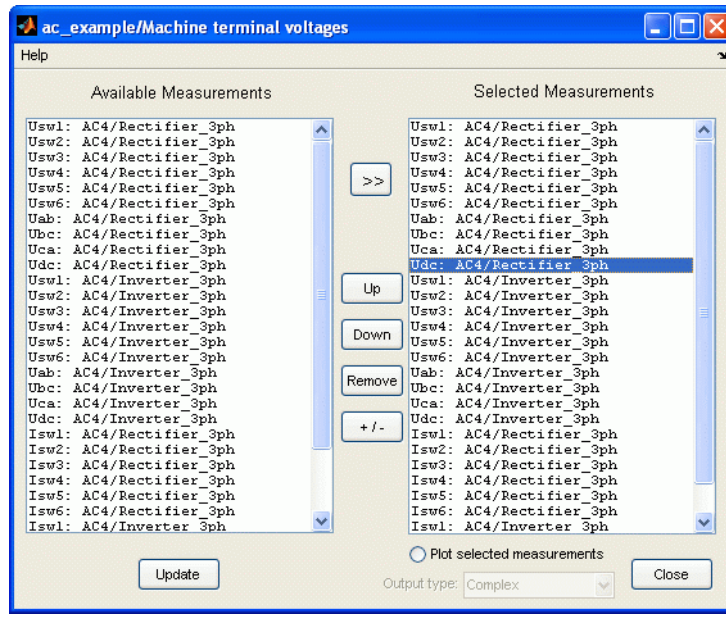
You must now use the AC4 model outputs to visualize interesting signals with a scope. Suppose you need to visualize the following signals:

- The motor torque value and set point
- The motor speed
- The motor flux modulus
- The motor statoric currents
- The DC bus voltage

Note that all model input-output descriptions can be found in the corresponding reference notes. All motor variable values can be read via the Motor vector. The Conv. vector contains all converter related data. The Ctrl vector includes all reference signals and other control values.

As we did for the fan block, we will use the Machine Measurement Demux block to read the motor variables. For the Conv. and Ctrl vectors, we will use Selector blocks of the Simulink Signal Routing library.

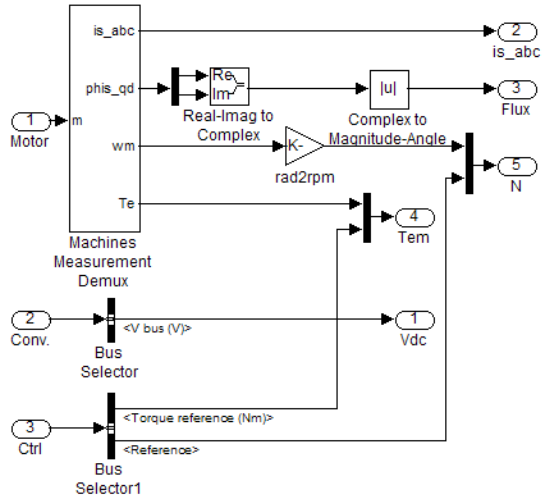
The contents of the Conv. vector can be easily determined by adding a Multimeter block to the model. The DC bus voltage, named `UDC: AC4/Rectifier_3ph`, is the 10th signal of output vector Conv.



Multimeter Window

Following the input-output description of the reference notes, the torque reference signal is the third signal of output vector Ctrl.

1 Build the subsystem below in order to extract all the needed visualization signals. Name the subsystem `Signal Selector`.



Signal Selector Subsystem

In the Machine Measurement Demux block, select the following signals: stator currents (ia, ib, ic), stator fluxes, rotor speed, and electromagnetic torque. The rad2rpm block shown above contains the constant $30/\pi$ in order to convert the rotor speed issued by the Machine Measurement Demux block from rad/s to rpm. A Real-Imag to Complex block and a Complex to Magnitude-Angle block are used to compute the magnitude of the flux vector.

2 Copy a scope to your model. It will be used to display the output signals of the Signal Selector block. Open the **Scope Parameters** dialog box. On the **General** tab, set the number of axes to `5`, set the simulation time range to `auto`, and use a decimation of `25`. Clear the **Limit Data Points to last** check box on the **Data history** tab. Connect the five outputs of the Signal Selector block to the inputs of the scope.

Setting the Fixed-Step Simulation Environment

All drive models of the library are discrete models. In order to simulate your system, you must now specify the correct simulation time step and set the fixed-step solver option. Recommended sample time values for DC drives, AC drives, and mechanical models can be found in the Remarks sections of

the corresponding block reference pages. The recommended sample time for the AC4 model is 1 µs.

1 Open the SimPowerSystems library and copy a Powergui block into `ac_example`. Set the sample time to 1 µs.

   Your circuit should now look like Induction Motor Drive Example Circuit on page 4-51.

2 Open the **Simulation/Configuration Parameters** dialog box. Select the `fixed-step`, `discrete (no continuous states)` solver option. Set the stop time to `1` s and the fixed-step size to `Ts`.

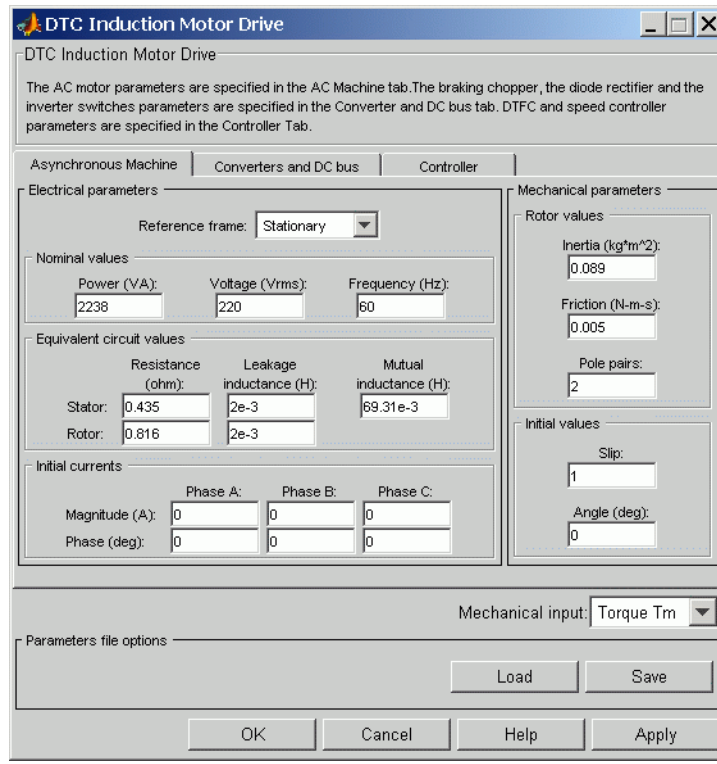
Before simulating your circuit, you must first set the correct AC4 internal parameters.

Setting the High Power Drive Parameter Set

As explained in the DC example, many drive models of the Electric Drives library have two sets of parameters: a low power set and a high power set. By default, all models are initially loaded with the low power set. The AC4 model parameters currently loaded in `ac_example` are those of a 3 hp drive.

You now set the high power drive parameters, which are those of a 200 hp drive. To do this, you use the **Load** button of the user interface as specified in the DC example:

1 Open the user interface by double-clicking the AC4 block. The interface is shown below:



AC4 User Interface

2 To load the 200 hp parameters, click the **Load** button.

3 Select the `ac4_200hp.mat` file and click **Load**.

The 200 hp parameters are now loaded.

Setting the Motor Inertia Value

You must now set the motor inertia value. Note that the inertia values currently specified in each AC and DC model are “no-load” inertias that only represent the inertia of the rotor. If the motor is coupled to a load, these values must be increased by the load inertias. In this case, the current value of the inertia amounts to 3.1 kg*m^2. Assume that the combined inertia of the motor and the fan amounts to 10 kg*m^2. Note that the use of a flexible shaft connected between the motor and the fan would allow decoupling of the

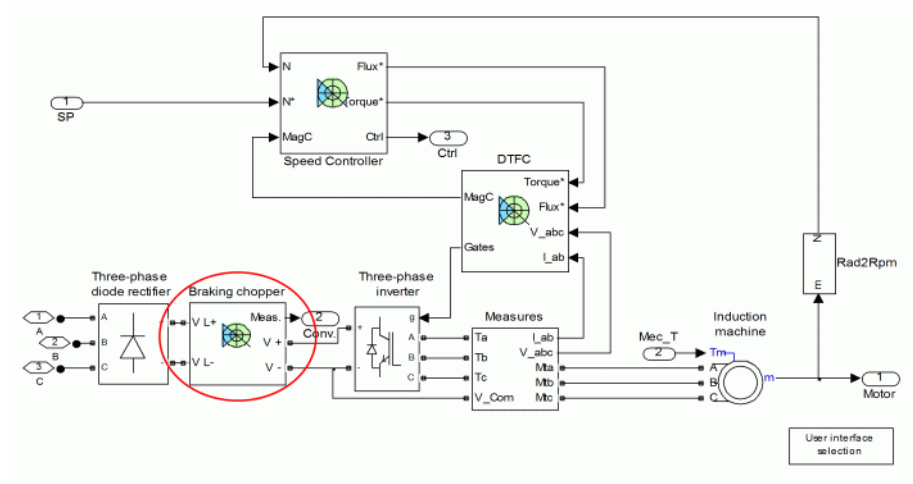
motor and load inertias. In that case, the inertia value of the AC4 block would only be the sum of the rotor and shaft inertias.

1 In the **Asynchronous Machine** section of the dialog box, change the inertia value to 10 kg*m^2.

2 Click **OK** to apply the changes and close the dialog box.

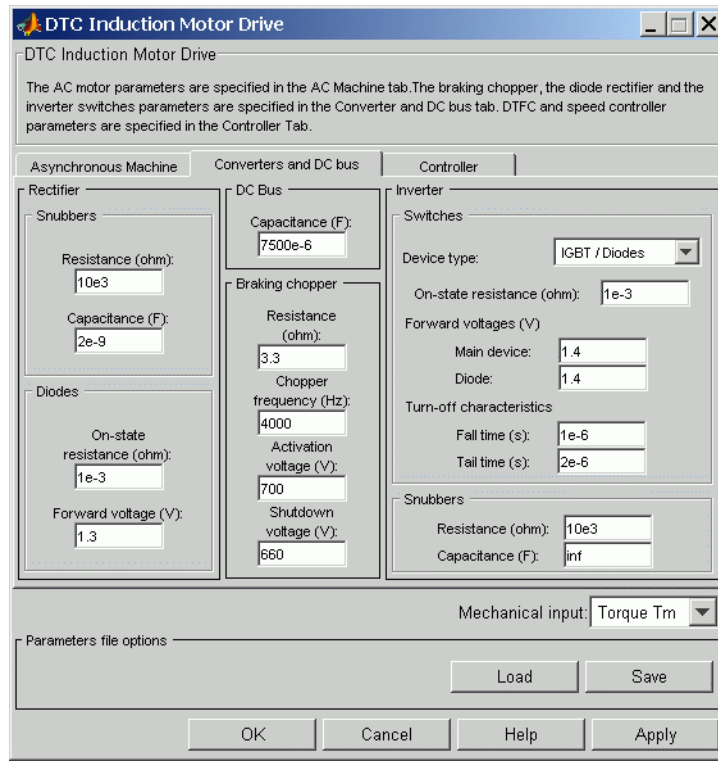
Setting the Braking Chopper Resistance Value

The three-phase inverter of the DTC system is fed by a DC voltage produced by a three-phase diode rectifier. A capacitor located at the output of the rectifier reduces the DC bus voltage ripples. A braking chopper block has also been added between the rectifier block and the inverter block, in order to limit the DC bus voltage when the motor feeds back energy to the drive (shown below). This energy is "burned" through a resistance when the DC bus voltage is too high.



Braking Chopper

The parameters of the braking chopper are available in the **Converters and DC bus** section of the dialog box, as shown below:



Converters and DC Bus Section of the User Interface

The braking chopper parameters are currently set to limit the DC bus voltage to about 700 V. Regarding the power P to be dissipated and the DC bus voltage limit V_{lim}, you can use the following equation to set the chopper resistance value:

$$R_{chop} = \frac{V_{lim}^2}{P}$$

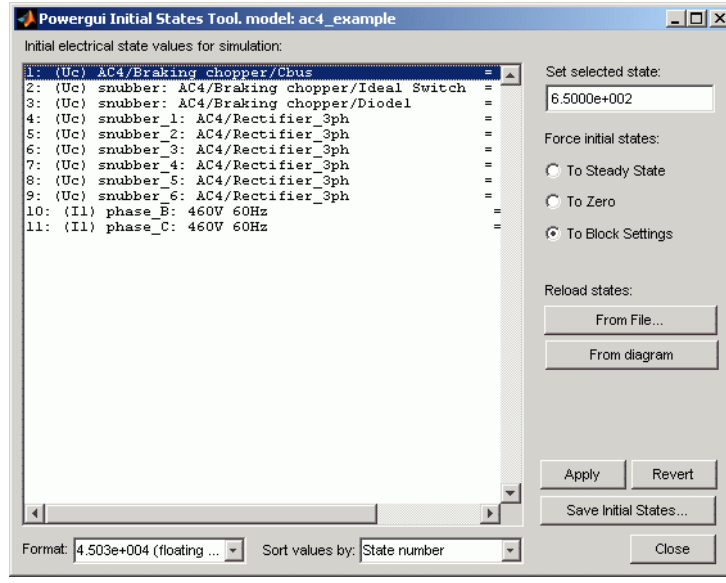
A resistance of 3.3 Ω will dissipate 200 hp at 700 V.

Setting the DC Bus Initial Voltage Value

Notice that the DC bus capacitance has a large value to reduce DC voltage ripples to small values. The AC4 model does not include a DC bus capacitor

preload system. If you start the simulation with too small an initial bus voltage, too high initial currents will be drawn from the rectifier to charge the capacitor. These high current values could damage a real-life system. You must thus set an initial DC bus voltage value to avoid such currents. This initial bus voltage should be equal to the rectified peak value of the AC source. If the AC voltage source amplitude is equal to 460 V, the rectified DC bus voltage obtained with a capacitor would be about $460 \times \sqrt{2}$ V.

1 Double-click the Powergui block located at the top level of `ac_example`. Click the **Initial States Setting** button. Set the `Uc_AC4/Braking chopper/Cbus` value to `650` V. Click **Apply** and then **Close**.

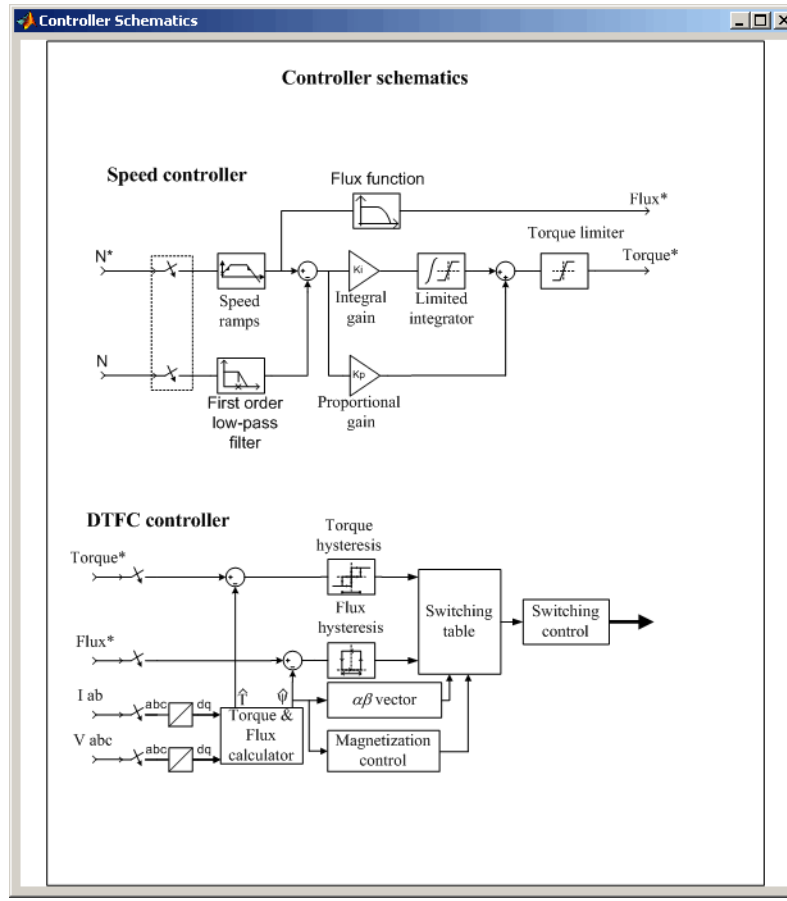


Setting the DC Bus Initial Voltage Value

Setting the AC4 Controller Parameters

The control system of AC4 has two main parts, a speed controller and a torque and flux controller (DTC). Information on these two parts can be found in the corresponding reference notes. To have a quick idea of the internal structure of the drive control system, a schematic is available inside the user interface of the model. Let's open the schematics related to the AC4 model.

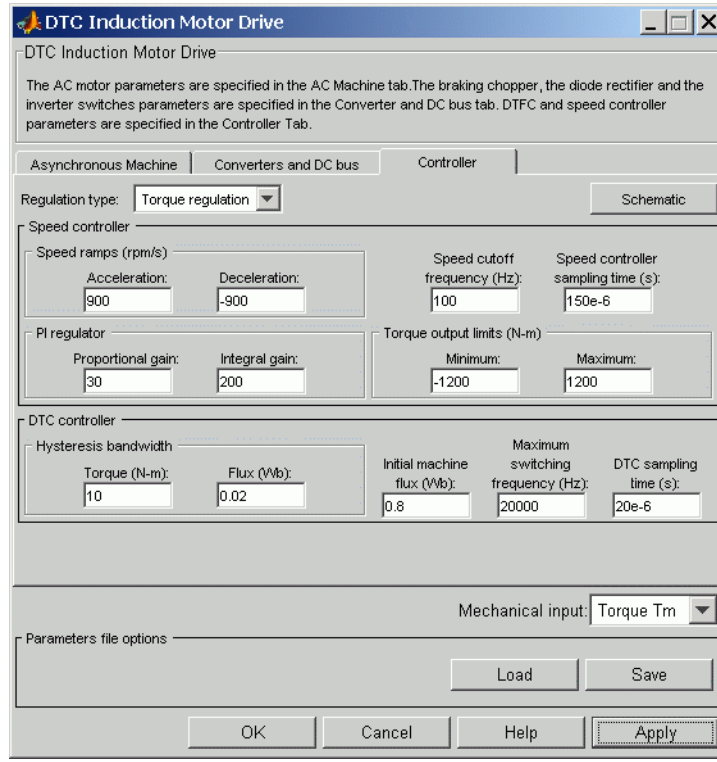
1 Open the user interface. Click the **Controller** section and then the **Schematic** button. You should see the controller schematics shown.



Controller Schematics of the User Interface

The speed controller consists of a simple proportional-integral regulator. The parameters of this controller are the proportional and integral gains, the speed ramp values, the low-pass filter cutoff frequency, the torque reference limits, and the sampling time. In this example, we will only control the motor torque; the speed controller is not used. Refer to “Setting the DC3 Controller Parameters and Simulation Results” on page 4-30 for more details on how to trim a PI controller.

Regarding the DTC controller, there is not much to trim. As you can see below, the parameters are the torque and flux bandwidths, the initial machine flux, the maximum switching frequency, and the DTC controller sampling time. All these parameters are already trimmed and should normally not be modified.



Controller Section of the User Interface

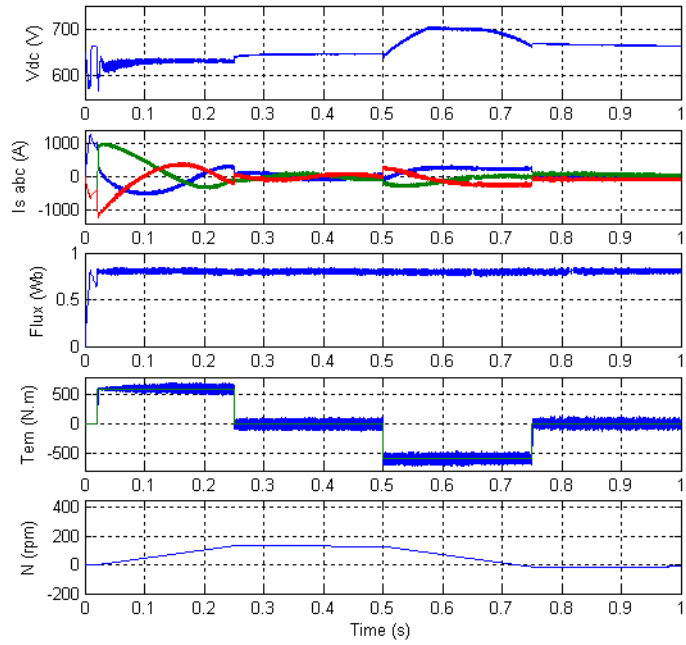
The default regulation mode is speed regulation. In order to have torque regulation, you must change the regulation mode in the **Controller** section of the user interface.

2 In the **Controller** section of the user interface, select `Torque regulation` for the **Regulation type** field. Click **OK** to apply the changes and close the dialog box.

The circuit is now ready for simulation.

Simulation Results

The simulation results are shown below.



Simulation Results

Observe the motor's fast torque response to the torque set point changes. From 0.02 s to 0.25 s, the fan speed increases because of the 600 N.m acceleration torque produced by the induction motor. At t = 0.25 s, the electromagnetic torque jumps down to 0 N.m and the speed decreases because of the load torque opposed by the fan. At t = 0.5 s, the motor torque develops a -600 N.m torque and allows braking of the fan. During braking mode, power is sent back to the DC bus and the bus voltage increases. As planned, the braking chopper limits the DC bus voltage to 700 V. At t = 0.75 s, the electromagnetic torque jumps back to 0 N.m and the speed settles around -10 rpm and

decreases toward 0 rpm. Notice that the flux stays around 0.8 Wb throughout the simulation. The flux and torque oscillation amplitudes are slightly higher than 0.02 Wb and 10 N.m respectively as specified in the user interface. This is due to the combined effects of the 15 µs DTC controller sampling time, the hysteresis control, and the switching frequency limitation.

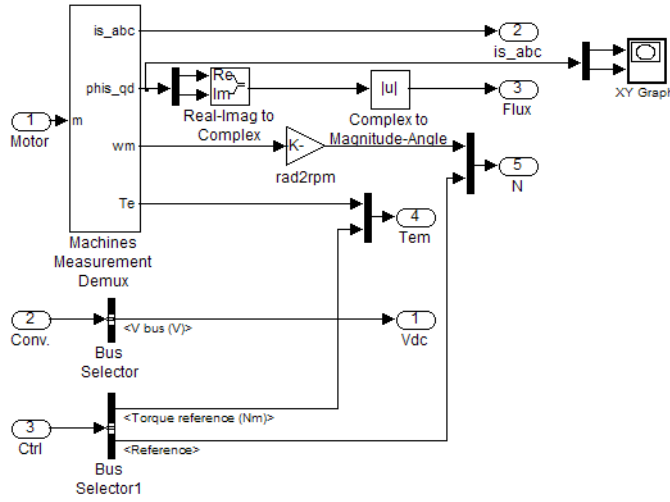
It is interesting to visualize the rotating flux produced by the stator. To do so, use a XY scope from the Sinks library.

1 Open the Sinks library.

2 Copy an XY scope inside the Signal Selector block of `ac_example`.

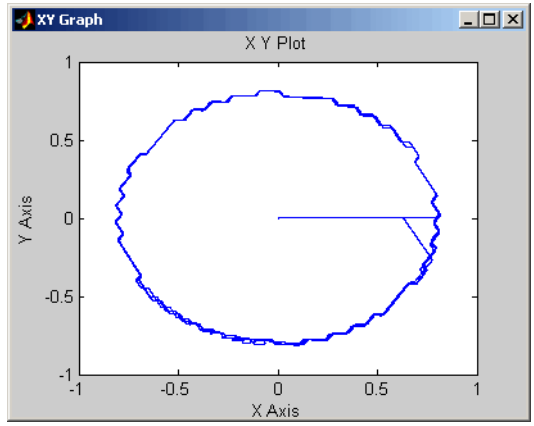
3 Connect the scope as shown.

4 Run a new simulation.



Adding a XY Graph to Visualize the Rotating Statoric Flux

The following figure shows the simulation results of the XY scope. The rotating field is clearly visible. Its modulus is about 0.8 Wb and its bandwidth is slightly bigger than 0.2 Wb.



Rotating Statoric Flux

Mechanical Models

In this section...
“Mechanical Shaft Block” on page 4-70
“Speed Reducer Block” on page 4-71

Mechanical Shaft Block

The Mechanical Shaft block is used to simulate a shaft interconnecting mechanically a motor drive block to a mechanical load block. Hence the Mechanical Shaft block allows decoupling of the mechanical parameters of the load from the ones of the motor. The mechanical shaft is represented by its stiffness coefficient K_{sh} and damping coefficient D_{sh}. The shaft transmitted torque T_{sh} is computed with

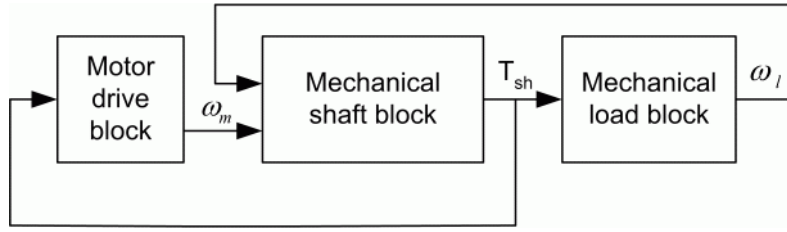
$$T_{sh} = K_{sh}\int(\omega_m - \omega_l)dt + D_{sh}(\omega_m - \omega_l)$$

where ω_m and ω_l are the speeds of the motor and the load, respectively.

The following figure shows the interconnections between the Motor Drive block, the Mechanical Shaft block, and the Mechanical Load block. The Mechanical Shaft block has two inputs, the load and motor speeds, and one output, the shaft transmitted torque. Note that the transmitted torque is applied at the load torque input of the motor. The transmitted torque is also applied at the input of the Mechanical Load block, which can be modeled by

$$T_{sh} = J_l\frac{d\omega_l}{dt} + B_l\omega_l$$

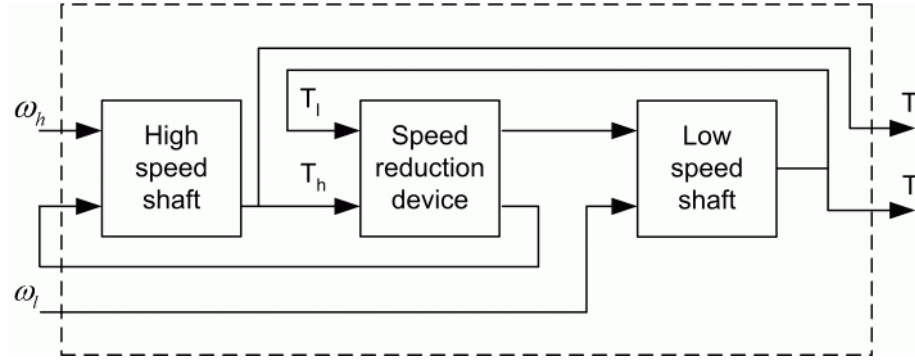
where J_l and B_l are the inertia coefficient and the viscous friction coefficient, respectively.



Interconnection Diagram of the Transmission Shaft

Speed Reducer Block

In many applications, the mechanical load requires high torque at low speed rather than low torque at high speed. This can be obtained by interconnecting the motor to the mechanical load by a speed reducer. The Speed Reducer block of the Electric Drives library is composed of a high-speed shaft and a low-speed shaft connected by a speed reduction device, as shown in the following figure. The speed reducer block has seven parameters: the stiffness and damping coefficients of the high-speed and the low-speed shafts, the reduction ratio, and the speed reduction device inertia and efficiency. The two inputs of the speed reducer block are the motor speed (high speed) and the load speed (low speed) while the outputs are the high-speed shaft torque T_h and the low-speed shaft torque T_l. The high-speed shaft torque must be applied to the load torque input of the motor. The low-speed shaft torque must be applied directly to the mechanical load block.



Interconnection Diagram of the Three Internal Blocks of the Speed Reducer Block

Mechanical Coupling of Two Motor Drives

In this section...

"Introduction" on page 4-72

"System Description" on page 4-73

"Speed Regulated AC4 with Torque Regulated DC2" on page 4-75

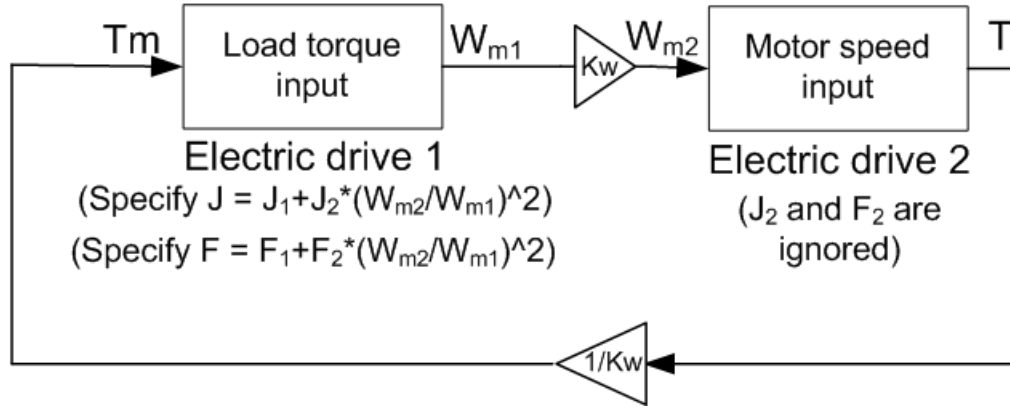
"Torque Regulated AC4 with Speed Regulated DC2" on page 4-76

Introduction

In order to test a motor drive under various load conditions, you must provide a variable and bidirectional load at the motor shaft. Moreover, an ideal load should also allow returning the absorbed energy from the motor back to the power grid as electric energy. Such a load can be implemented using a four-quadrant motor drive such as the DC2 or DC4 models. Either of these two motor drives can be conveniently coupled to the motor drive model being tested by the use of the mechanical shaft model.

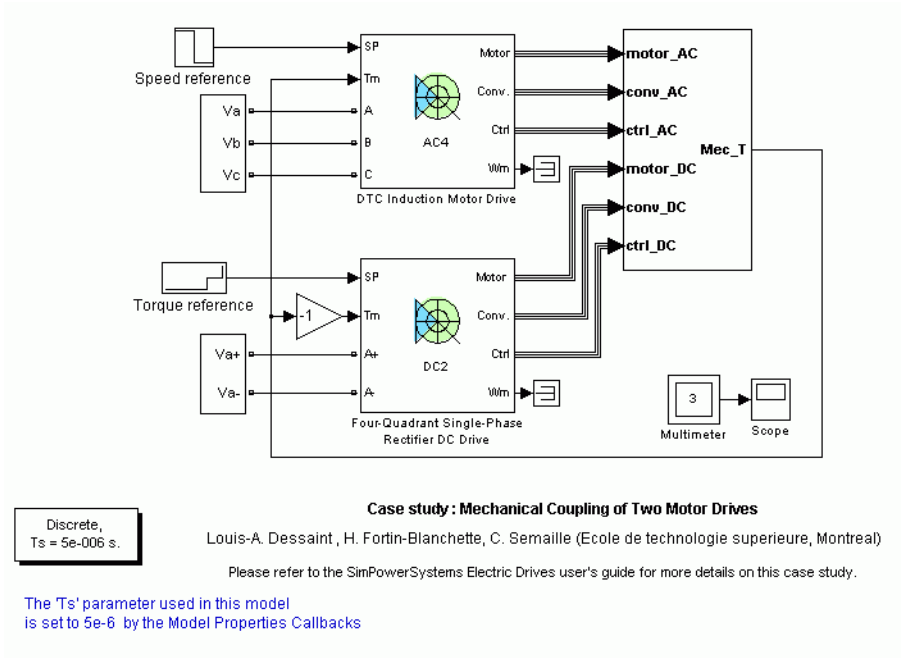
Therefore this case study will consist of coupling the AC4 motor drive model to the DC2 motor drive. The AC4 motor drive is a DTC three-phase induction motor-based drive. The DC2 motor drive is a single-phase dual-converter DC motor drive. In such a system, one drive is speed regulated while the other is torque regulated, but each drive can operate either as a motor or as a generator, as will be seen later. The DC2 motor drive is rated 3 hp, 240 V, 1800 rpm, and the AC4 motor drive is rated 3 hp, 380 V, 60 Hz, 4 poles.

Note It is also possible to couple two motor drives using the **Mechanical input** menu located in the lower part of the GUI. The next figure indicates how to model a stiff shaft interconnection in a motor-generator configuration. The speed output of drive 1 (mechanical input is load torque) is connected to the speed input of drive 2 (mechanical input is motor speed), while drive 2 electromagnetic torque output Te is applied to the mechanical torque input Tm of drive 1. The Kw factor represents the speed reduction ratio. Also, because inertia J2 and viscous friction F2 are ignored in the machine of drive 2, they have to be added in the machine tab of drive 1.

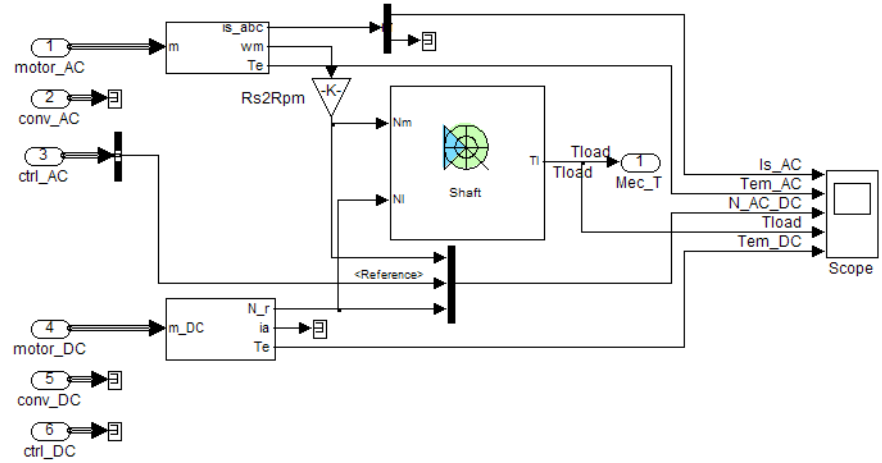


System Description

The complete system consisting of two motor drives mechanically coupled together is shown in SPS Diagram of the Two Interconnected Drives on page 4-74. The mechanical shaft model is contained in the third block of the diagram. If you open this block, you will see, as in Interconnections of the Mechanical Shaft Model on page 4-74, that the AC4 and DC2 motor speed signals are connected respectively to the Nm and Nl inputs of the mechanical shaft model. The output Tl of the mechanical shaft model represents the mechanical torque transmitted from the AC4 motor to the DC2 generator. Therefore, this output is connected directly to the mechanical torque input of AC4, and is also sign inverted and then connected to the mechanical torque input of DC2, as can be seen in SPS Diagram of the Two Interconnected Drives on page 4-74.



SPS Diagram of the Two Interconnected Drives

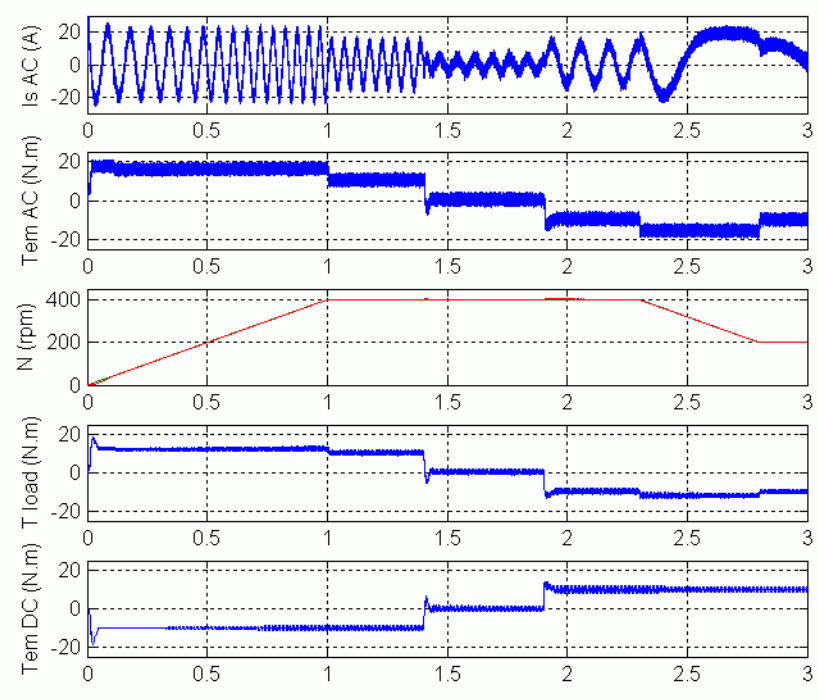


Interconnections of the Mechanical Shaft Model

Speed Regulated AC4 with Torque Regulated DC2

To begin with, the AC4 model operates as a speed regulated motor loaded by the DC2 model operating as a torque regulated generator. This setup, contained in the `cs_coupling_1.mdl` file, allows the testing of the AC4 model speed ramps and load torque disturbance responses. Note that in steady state, the signs of the AC4 electric torque and speed should be the same, confirming that AC4 operates as a motor. The DC2 electric torque and speed should be of opposite signs, confirming that DC2 operates as a generator. This is in line with the sign of the reference torque applied to the DC2 motor drive that is opposite to the speed sign.

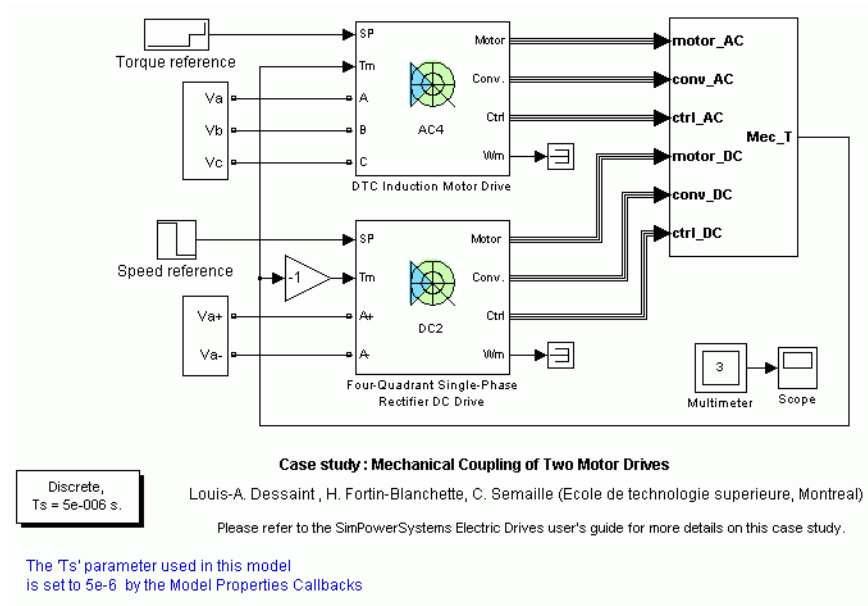
Speed Ramp and Load Disturbance Torque Responses of the AC4 Motor Drive on page 4-76 shows the results of an AC4 motor drive startup at nearly full load followed by the application of load disturbance torques. You can see that the AC4 motor speed is exactly superposed to the reference ramp of +400 rpm/s since the AC4 electric torque maximum limit is high enough. The AC4 motor speed reaches the demanded value of 400 rpm at t = 1.0 s. At that moment, the AC4 electric torque drops down to 10 N.m. Then at t = 1.4 s, a reference torque of 0 N.m is applied to DC2; the AC4 electric torque immediately drops down to zero in order to maintain the regulated speed. At t = 1.9 s, a reference torque of +10 N.m is applied to the DC2 drive, forcing AC4 to operate as a generator and DC2 as a motor (look at the speed and torque signs of the two drives). Finally, a negative reference speed ramp of -400 rpm/s is applied to AC4 at t = 2.3 s. Note that, again, AC4 precisely follows the demanded ramp. A new steady state is reached at t = 2.8 s, and the AC4 electric torque stabilizes at -10 N.m. Speed Ramp and Load Disturbance Torque Responses of the AC4 Motor Drive on page 4-76 also shows the mechanical torque transmitted by the shaft, which is similar to the AC4 electric torque but contains less ripple.



Speed Ramp and Load Disturbance Torque Responses of the AC4 Motor Drive

Torque Regulated AC4 with Speed Regulated DC2

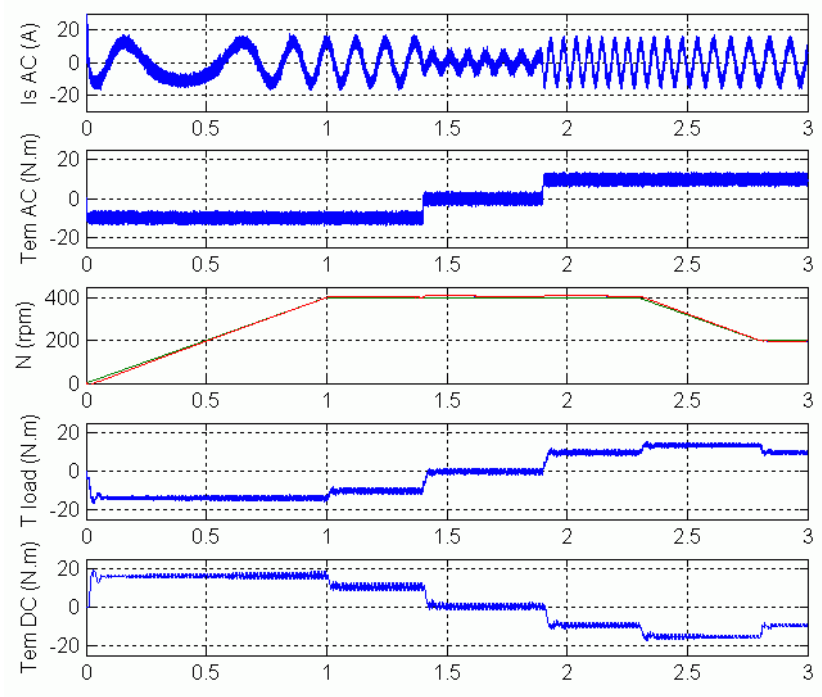
This time, AC4 operates as a torque regulated motor loaded by the DC2 drive that is speed regulated. The complete system is shown in SPS Diagram of the Two Interconnected Drives on page 4-77 and is contained in the `cs_coupling_2.mdl` file. The interconnection of the mechanical shaft model with the two drives remains unchanged with respect to Interconnections of the Mechanical Shaft Model on page 4-74. All the regulator gains of both drives are the same as in the previous case. The setup is tested in the same conditions as before.



SPS Diagram of the Two Interconnected Drives

Speed Ramp and Load Disturbance Torque Responses of the DC2 Motor Drive on page 4-78 shows the results of a DC2 motor drive startup at nearly full load followed by the application of load disturbance torques. Note that the DC2 motor speed follows the reference ramp of 400 rpm/s with some overshoot and undershoot. The DC2 motor speed reaches the demanded value of 400 rpm at t = 1.0 s and stabilizes completely at t = 1.2 s. Then at t = 1.4 s, a reference torque of 0 N.m is applied to AC4; observe how fast the AC4 torque responds. At t = 1.9 s, a reference torque of +10 N.m is applied to the AC4 drive, forcing DC2 to operate as a generator and AC4 as a motor (look at the speed and torque signs of the two drives). Observe that the DC2 speed overshoots each time the load torque changes. Finally, a negative reference speed ramp of -400 rpm/s is applied to DC2 at t = 2.3 s. The DC2 speed follows well but presents a small overshoot and a small undershoot. A new steady state is reached at t = 2.8 s, and the DC2 electric torque stabilizes at -10 N.m. Speed Ramp and Load Disturbance Torque Responses of the DC2 Motor Drive on page 4-78 also shows the mechanical torque transmitted by the shaft, which is very similar to the negative of the DC2 electric torque but with more ripple.

You can see from the results shown in Speed Ramp and Load Disturbance Torque Responses of the AC4 Motor Drive on page 4-76 and Speed Ramp and Load Disturbance Torque Responses of the DC2 Motor Drive on page 4-78 that the speed ramp responses are more precise and the load torque disturbance more efficiently rejected with the AC4 drive than with the DC2 drive. This is essentially due to the fast dynamics of the AC4 electric torque. Recall that the AC4 drive consists of a direct torque controller based on hysteresis comparators and high-frequency switching, while the DC2 drive relies entirely on naturally commutated thyristor converters. However, the torque ripple magnitude of the AC4 drive is higher than for the DC2 drive.



Speed Ramp and Load Disturbance Torque Responses of the DC2 Motor Drive

Winding Machine

In this section...

"Introduction" on page 4-79

"Description of the Winder" on page 4-79

"Block Description" on page 4-81

"Simulation Results" on page 4-84

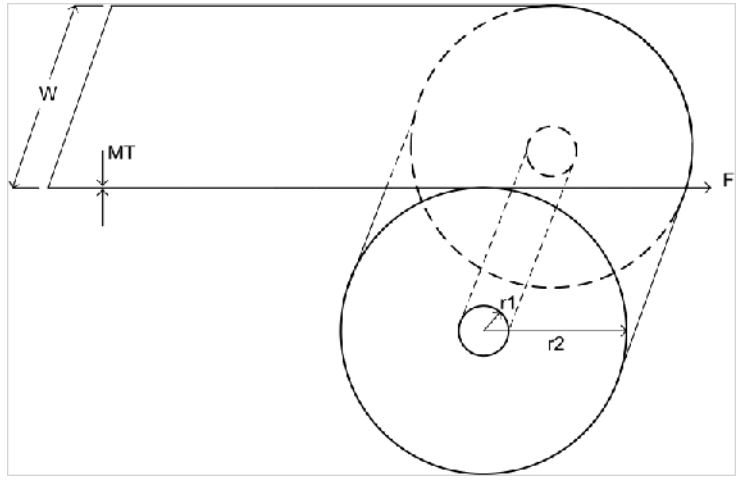
Introduction

Winding machines, also called winders, are used in the pulp and paper industry as well as in the textile, steel, and plastic industries.

An important characteristic of most winders is that the force acting on the winding material must remain constant. This is realized by controlling the winder torque proportionally to the roll variable radius. Note that it is assumed here that the material is fed to the winder at constant speed. The latter implies that the winder angular speed is forced to decrease proportionally to the roll radius. Hence the winding machine is a constant power application, because the product of the winder mechanical torque and its angular speed is constant.

Description of the Winder

Physical Representation of a Winder on page 4-80 shows a physical representation of a winder where W is the roll width, r_1 the core radius, r_2 the roll radius, and MT the material thickness.



Physical Representation of a Winder

Beside the variables described above, the simulation also requires the following parameters and variables:

MV	Material mass per unit volume
L	Material length
M	Material mass
J_r	Material inertia
J_c	Winder core inertia
B_w	Winder viscous friction coefficient

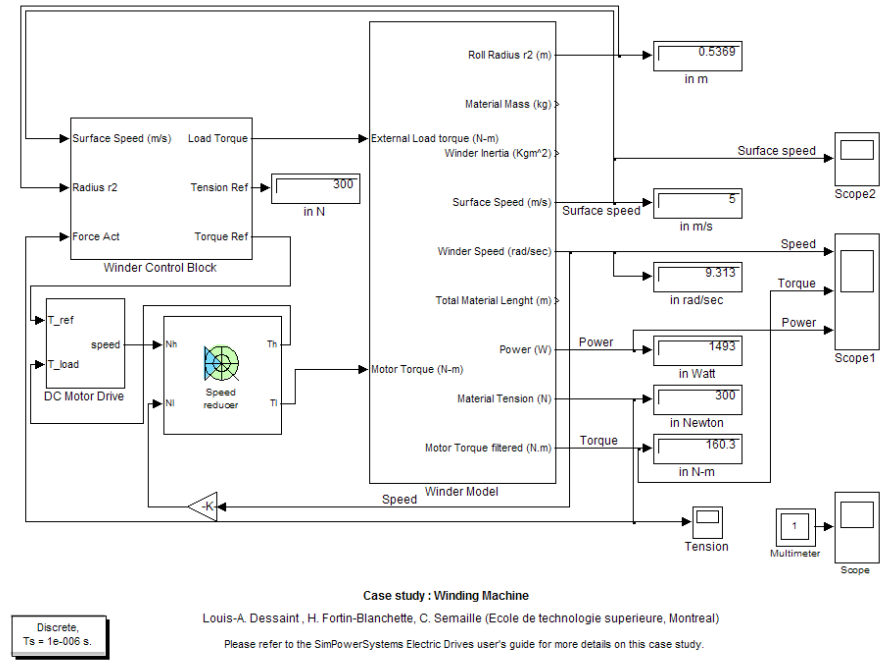


Diagram of the Complete Winding System

Diagram of the Complete Winding System on page 4-81 shows a Simulink® diagram of the complete winding system. This system consists of four blocks: the Winder Control block, the DC Motor Drive block, the Speed Reducer block, and the Winder Model block.

Block Description

Winder Model Block

This block computes various winder variables using the following equations.

Surface speed

$$\text{Surface speed} = \text{winder angular speed} \times r_2$$

Material length L

$$L = \int \text{surface speed} dt$$

Roll radius r_2

$$r_2 = \sqrt{\frac{L \times MT}{\pi} + r_1^2}$$

Material mass M

$$M = MV \times \pi \times W \times (r_2^2 - r_1^2)$$

Total winder inertia Jt and material inertia J_w

$$J_t = J_w + J_c$$

where

$$J_w = \frac{1}{2} \cdot M \cdot (r_2^2 + r_1^2)$$

The winder angular speed is calculated using the following differential equation

$$T_e = J_t \cdot \frac{d\omega}{dt} + B \cdot \omega + T_l$$

where T_l is the winder load torque and T_e is the motor drive electric torque. The calculation of the tension or force F applied on the winding material is based on the same differential equation as above where the load torque is expressed as $T_l = F \times r_2$. Rearranging the equations in term of F yields

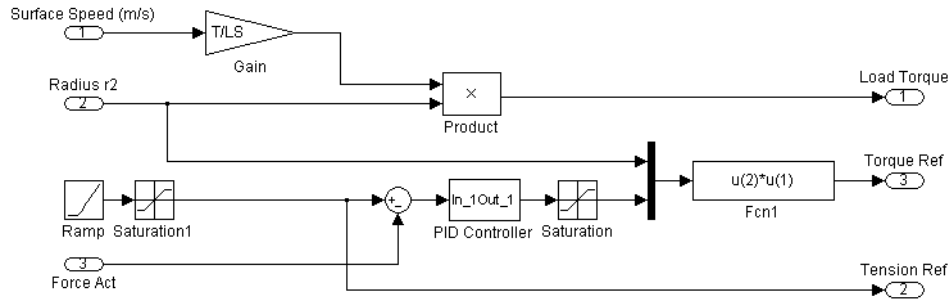
$$F = \frac{T_e - (J_t \cdot \dot{\omega}) - (B \cdot \omega)}{r_2}$$

This estimated force is fed back to the winder control block in order to be regulated.

Note that in the above two equations, the term $\omega \cdot \dot{J}_t$ is omitted because it has been found to be negligible for the case considered here.

Winder Control Block

This block contains a PID controller that regulates the tension applied on the winding material. The output of this force controller is a torque reference set point for the winder motor drive. The Winder Control block shown in Winder Control Block on page 4-83 also contains the tension versus speed characteristic of the external process supplying the material to the winder at constant speed. This characteristic consists in a straight line of slope equal to the ratio of the reference material tension on the constant surface speed.



Winder Control Block

DC Motor Drive Block

This block contains a complete two-quadrant three-phase rectifier DC drive with its three-phase voltage source. The DC drive is rated 5 hp, 220 V, 50 Hz and is torque regulated.

Speed Reducer Block

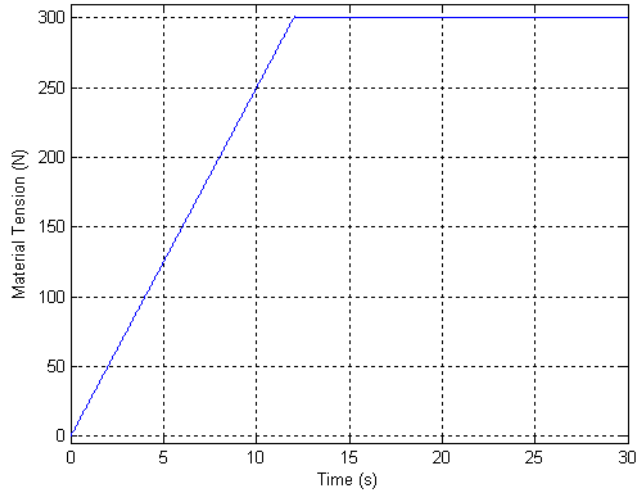
The DC motor is connected to the winder by a Speed Reducer block. The speed reduction ratio is 10, allowing the winder to turn 10 times slower than the motor, while the shaft-transmitted torque is almost 10 times higher on the low-speed side. The torque required by the winder in this case study is approximately 200 N.m.

Simulation Results

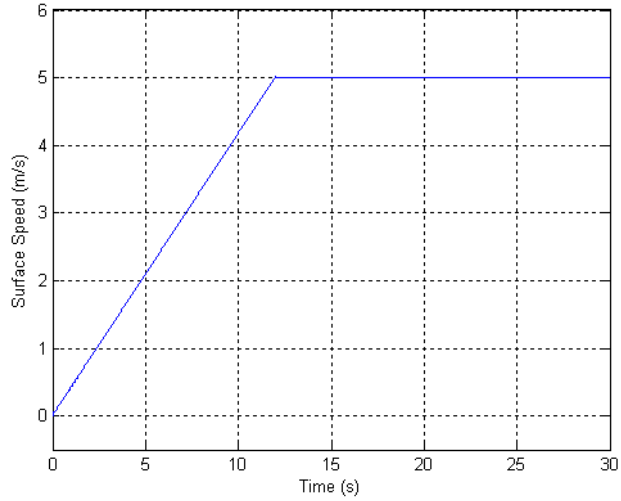
The winding machine simulation model is contained in the file `cs_winder.mdl`. The simulation parameters are those of a paper winding application where the roll width is 10 m. Open the file and look at the parameters in the Simulink masks of the Winder Model block, the Winder Control block, the DC Motor Drive block, and the Speed Reducer block. In the Winder Control block, you will see that the tension set point is 300 N and the surface speed set point is 5 m/s.

The rate of change of the tension set point is limited internally to 25 N/s so that the tension set point requires 12 s to reach its final value. Note that the simulation time step of the complete model is 1 µs in order to comply with the speed reducer, which is the block that requires the smallest simulation time step.

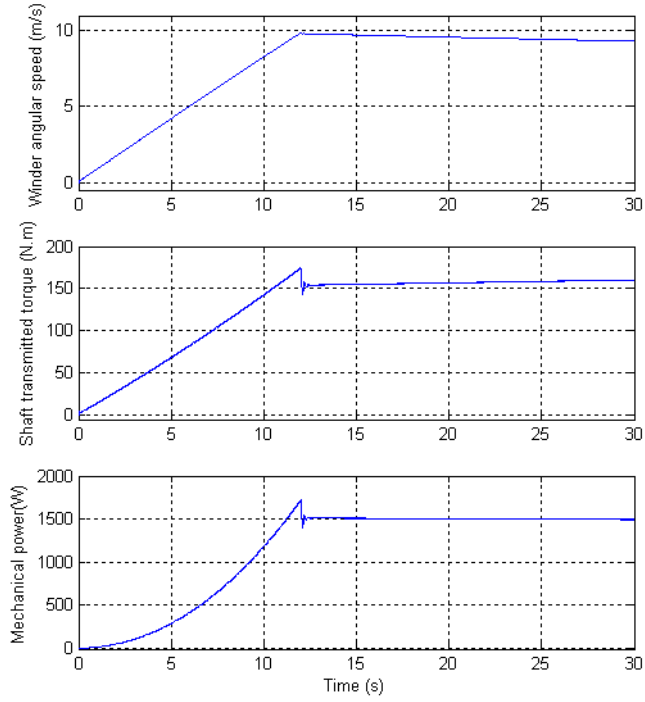
Start the simulation and observe how well the material tension and the surface speed ramp to their prescribed values in Material Tension on page 4-85 and Surface Speed on page 4-85 respectively. Winder Angular Speed, Mechanical Torque, and Power on page 4-86 shows the winder angular speed, mechanical torque, and power. Note that once the operating point is reached (300 N, 5 m/s), the angular speed decreases and the torque increases, both linearly, so that the power is approximately constant. The reason why the mechanical power is not precisely constant but decreases slightly is that the decreasing speed winder own inertia supplies a small part of the constant power required by the winder.



Material Tension



Surface Speed



Winder Angular Speed, Mechanical Torque, and Power

Robot Axis Control Using Brushless DC Motor Drive

In this section...

"Introduction" on page 4-87

"Description of the Robot Manipulator" on page 4-87

"Position Control Systems for Joints 1 and 2" on page 4-88

"Modeling the Robot Position Control Systems" on page 4-89

"Tracking Performance of the Motor Drives" on page 4-93

Introduction

Robots are complex electromechanical systems where several electric drives are used to control the movement of articulated structures. The design of axis control systems for robots can be greatly facilitated by the Electric Drives library, which can model complete axes including motor drives, speed reducers, mechanical model of the arm, and controllers in the same diagram.

This case study presents the modeling and simulation of a six-degrees-of-freedom robot manipulator using Electric Drives library blocks in combination with Simulink® blocks. The two main joints models are built using brushless DC motor drives that are connected to the rest of the manipulator through speed reducers (a model included in the Electric Drives library). The control system, which consists essentially of two position control loops, is built with Simulink blocks. The inner speed and torque control loops are already included in the drive model. The rest of the manipulator and its load are represented by two Simulink nonlinear models, one for each motor drive.

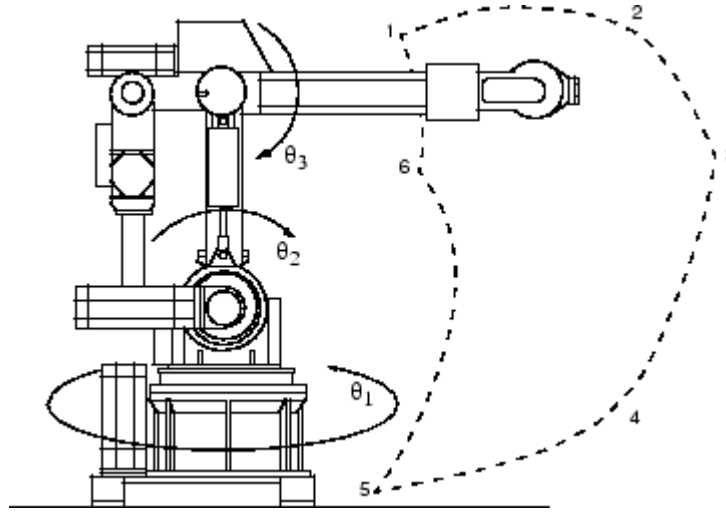
Detailed modeling is presented to demonstrate the versatility of the Electric Drives library. The operation of the joints using typical trajectories is simulated and results are presented.

Description of the Robot Manipulator

The robot considered in this example is a general-purpose six-degrees-of-freedom robot manipulator (GMF S-360) of parallelogram linkage type. Six-Degrees-of-Freedom Robot Manipulator on page 4-88 shows

the structure of the robot and its workspace. The robot has six axes. The three axes (Θ_1, Θ_2, Θ_3) shown in the figure are for arm positioning and the others (α, β, γ) are for orientation of the end effector. In the horizontal plane, the robot can cover a 300 degree arc (Θ_1 = -150° to Θ_1 = 150°).

The robot's axes are driven by brushless DC motors that are modeled by permanent-magnet synchronous motors fed by PWM inverters (AC6 drive model). Speed reducers of belt type and gearbox are used to transmit torque from the motors to the joints.



Six-Degrees-of-Freedom Robot Manipulator

Position Control Systems for Joints 1 and 2

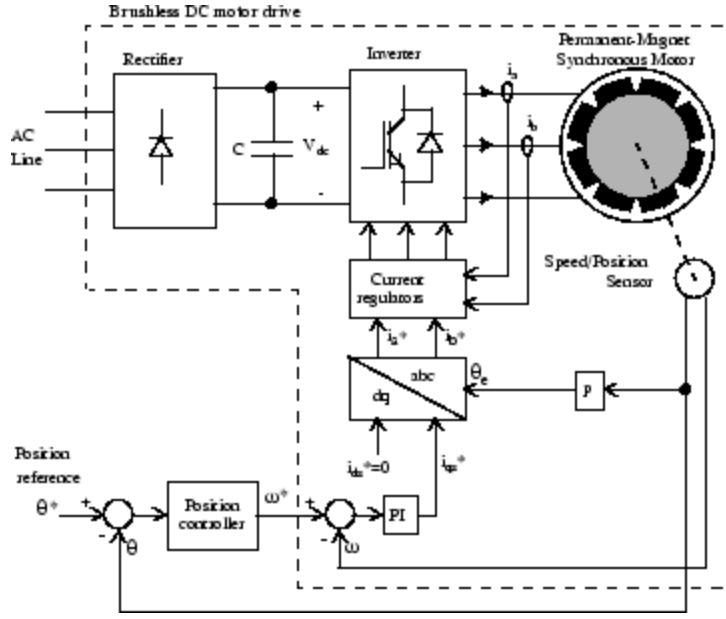
We will consider in particular the two first joints, which drive the entire robot and its load. The first axis uses a 2 kW brushless DC motor and a 1:130 speed reducer. The second axis uses a 1 kW brushless DC motor and a 1:100 speed reducer. Brushless DC Motor Drive for Position Control of Robot Joint on page 4-89 shows a simplified diagram of the position control system for one robot link.

The control system consists of three control loops connected in a cascade configuration: an outer position loop includes an inner speed control loop and an innermost current control loop. The PM synchronous motor is

fed by a three-phase PWM inverter operating in current-controlled mode. Field-orientation scheme is used to decouple the variables so that flux and torque can be separately controlled by stator direct-axis current i_{ds} and quadrature-axis current $i_{qs,}$ respectively. The quadrature-axis current reference i_{qs}* (which represents the torque command) is provided by the speed control loop. The direct-axis current reference i_{ds}* is kept equal to 0.

A speed/position sensor is used to provide the information required by the speed and position control loops. The rotor position is also required for coordinates conversion (dq to abc).

Each motor drives the rest of the robot structure, including the other links and the load, through a speed reducer.



Brushless DC Motor Drive for Position Control of Robot Joint

Modeling the Robot Position Control Systems

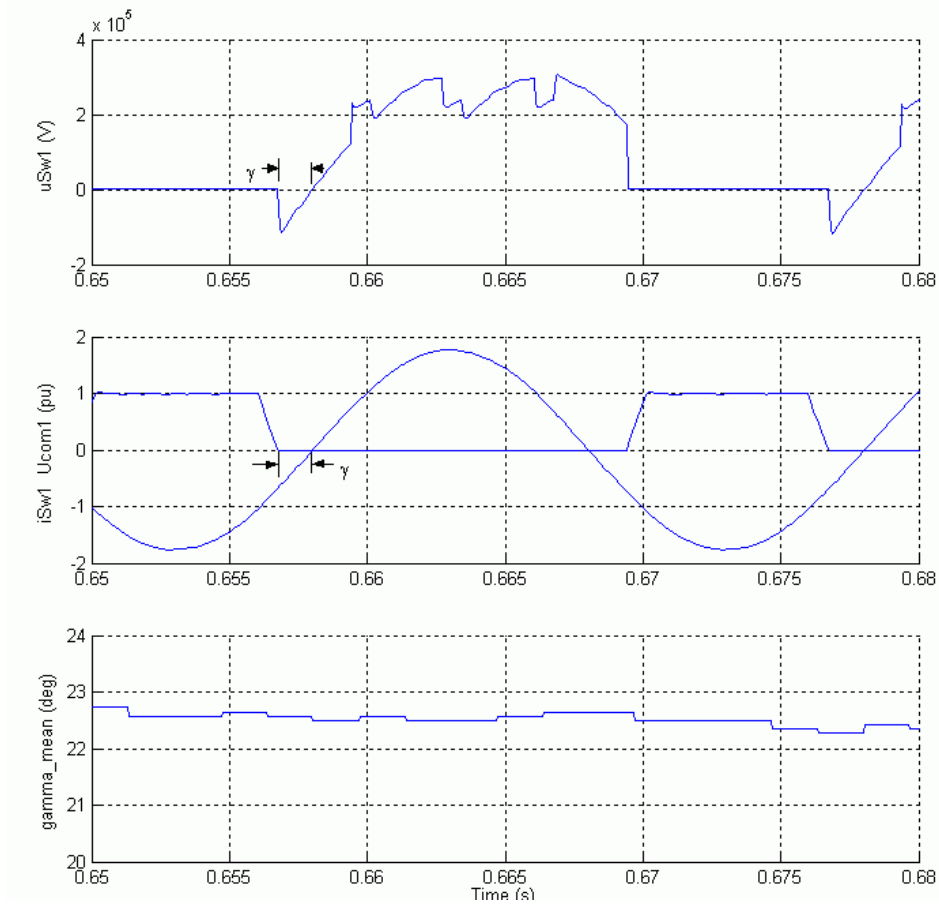
The entire drive system for the robot's two first joints, including motor drives, speed reducers, equivalent loads, and controllers can be modeled in the same diagram using blocks from the Electric Drives library and Simulink libraries,

as shown in Diagram Representing the Robot's Main Axes Drive Systems on page 4-90.

Diagram Representing the Robot's Main Axes Drive Systems

The brushless DC motor drives are represented by two AC6 (PM Synchronous Motor Drive) blocks from the Electric Drives library. This block models a complete brushless DC motor drive including a permanent-magnet synchronous motor (PMSM), an IGBT inverter, speed controller, and current controller. The AC6 inputs are the speed commands and the outputs are the motor speed, which are fed to the inputs of the speed reducers.

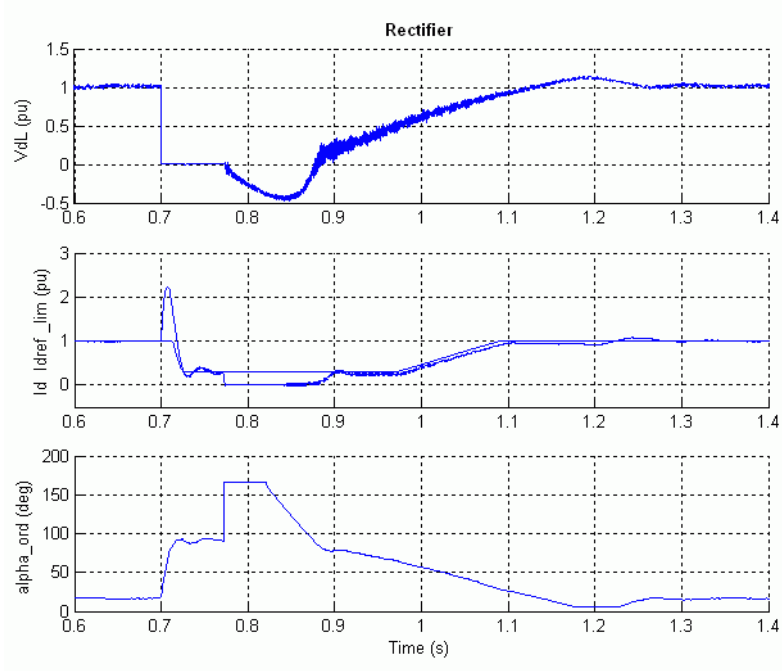
The speed reducers are modeled by two Speed Reducer blocks from the Electric Drives library. The inputs for these blocks are the motors' speeds, and the outputs are the torques from the low-speed sides, which are applied to the robot structure model. The speed reducers are characterized by their ratio and inertia and the stiffness and damping of input and output shafts.

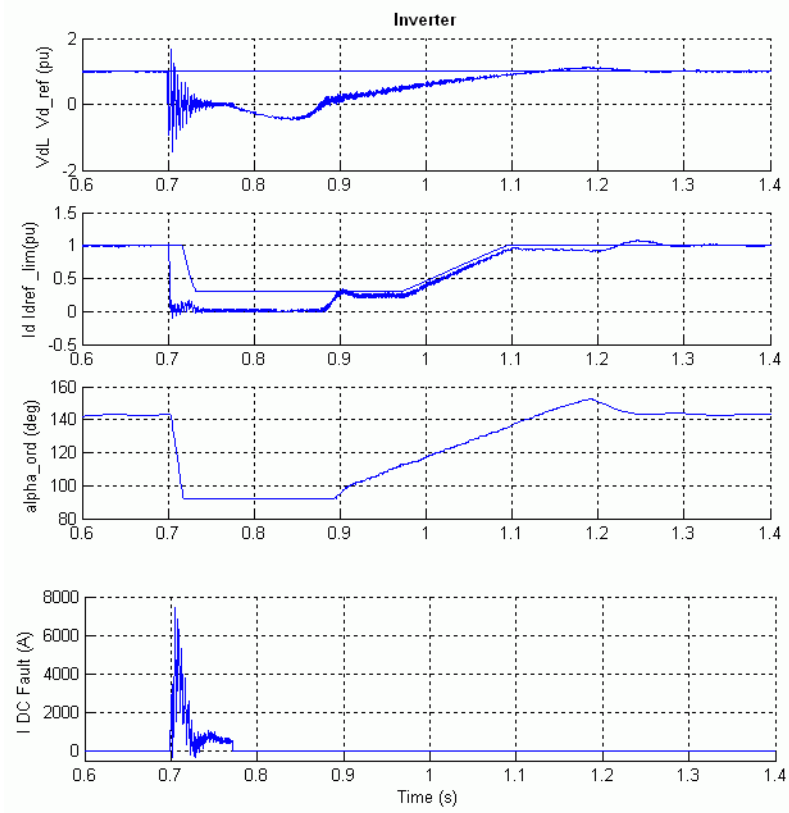


Current and Commutation Voltage of Valve 1 Showing γ

DC Line Fault

Deactivate the steps applied on the current reference and on the voltage reference in the Master Control and in the Inverter Control and Protection respectively by setting the switches in lower position. In the DC Fault block, change the multiplication factor of 100 to 1, so that a fault is now applied at t = 0.7 s. Reduce the Simulation Stop time to 1.4 s. Open the Rectifier scope as well as the Fault scope to observe the fault current and the Protection Rectifier scope to observe the DC Fault protection action. Restart the simulation.


Rectifier
VdL (pu)
Id Idref_lim (pu)
alpha_ord (deg)
Time (s)

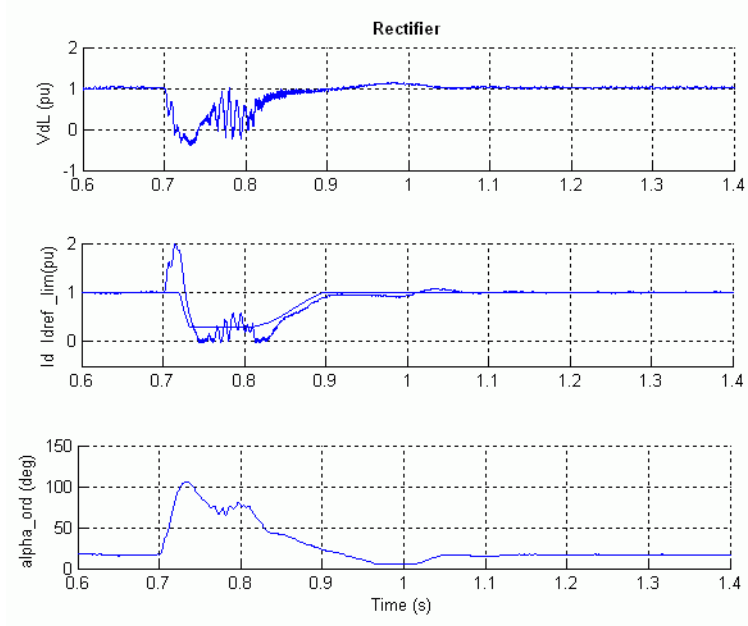


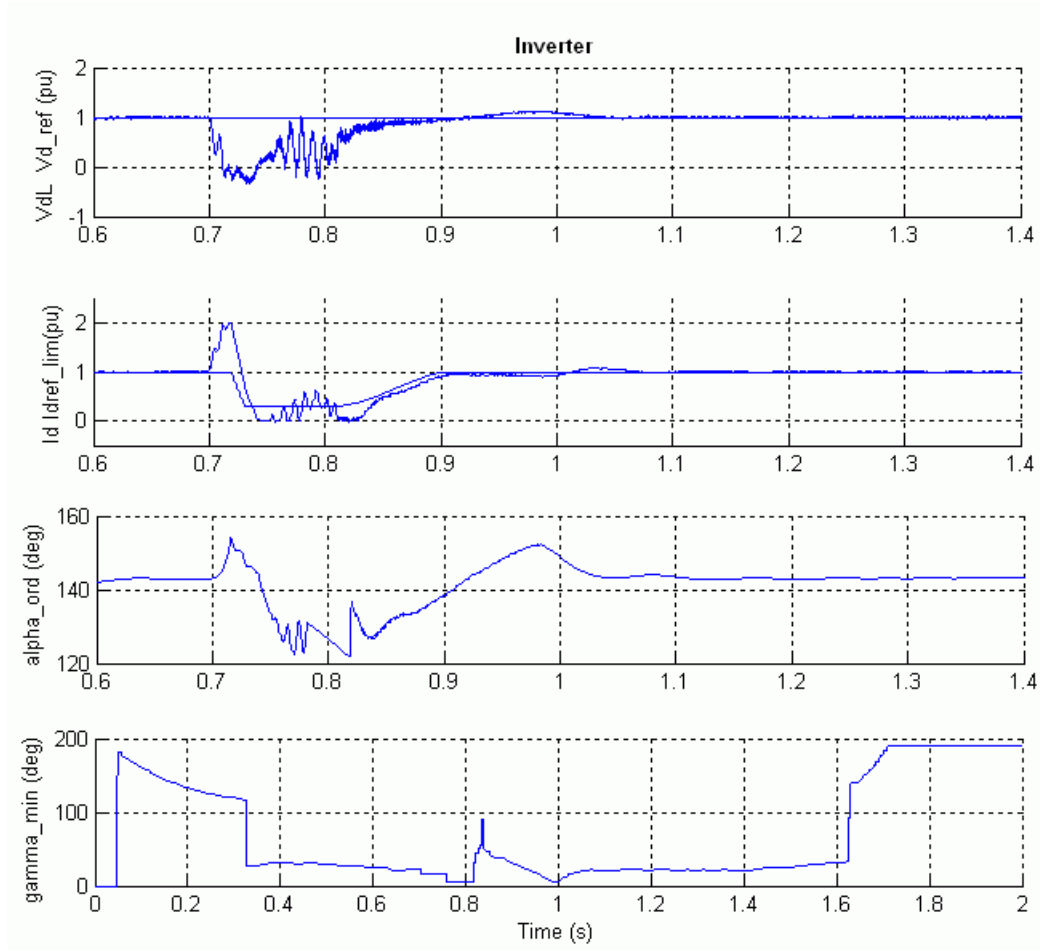
DC Line Fault on the Rectifier Side

At fault application (t = 0.7 s), the DC current increases to 2.2 pu and the DC voltage falls to zero at the rectifier. This DC voltage drop is seen by the Voltage Dependent Current Order Limiter (VDCOL) and the DC Fault protection. The VDCOL 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 α firing angle is forced to 166 degrees by the DC Fault protection after detecting a low DC voltage. 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 system, causing rapid extinction of the fault current at its next zero crossing. Then α is released at t = 0.82 s and the normal DC voltage and current recover in approximately 0.5 s. Notice, the temporary mode change in the Rectifier controls between 1.18 s and 1.25 s.

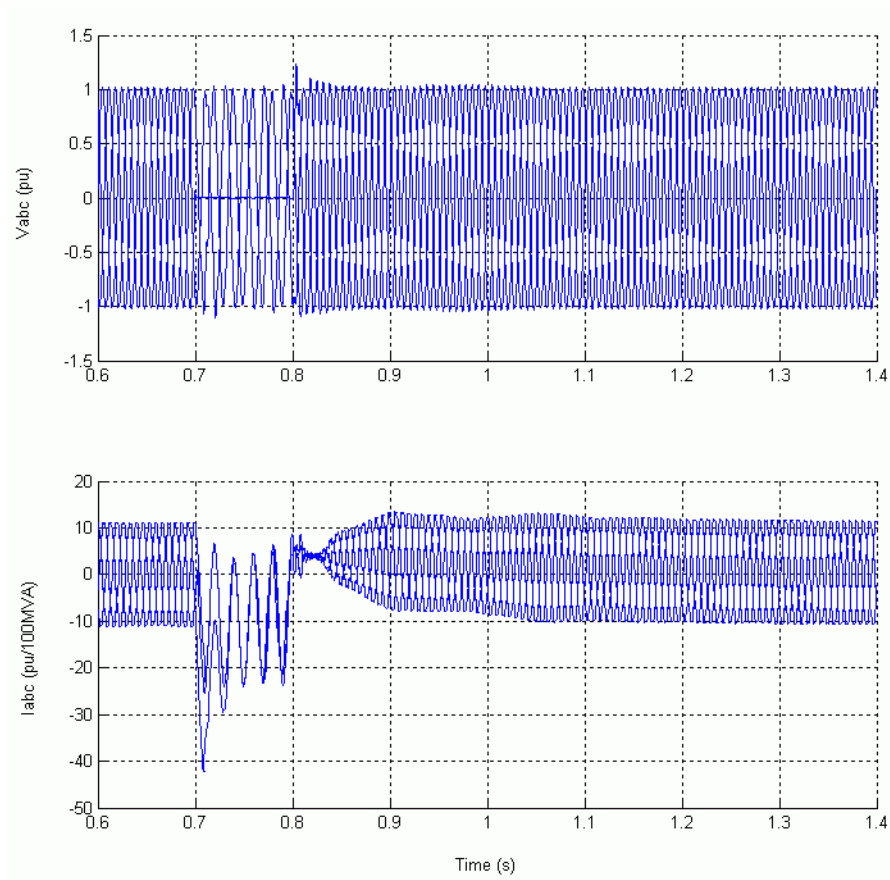
AC Line-to-Ground Fault at the Inverter

Now modify the fault timings to apply a line-to-ground fault. In the DC Fault block, change the multiplication factor of 1 to 100, so that the DC fault is now eliminated. In the A-G Fault block, change the multiplication factor in the switching times to 1, so that a six-cycle line-to-ground fault is now applied at t = 0.7 s at the inverter. Restart the simulation.





Rectifier, Inverter Signals for an AC Line Fault on Inverter Side



Voltages and Currents on the 50 Hz Side for an AC Line Fault on the Inverter Side

Notice the 120 Hz oscillations in the DC voltage and currents during the fault. An unavoidable commutation failure occurs at the inverter at the very beginning of the fault and the DC current increases to 2 pu. 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.

Look at the waveforms displayed on the PROTECTION INVERTER scope. The Low AC Voltage block detects the fault and locks the DC Fault protection that in this case should not detect a DC fault even if the DC line voltage

dips. Look at the Commutation Failure Prevention Control (CFPREV) output (A_min_I) which decreases the maximum delay angle limit to increase the commutation margin during and after the fault.

Now open the dialog box of the CFPREV block located inside the Inverter Protections subsystem and deactivate the CFPREV protection by deselecting the "ON State." Restart the simulation. Note that a second commutation failure now occurs during the fault (at t around 0.775 s). 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.

References

[1] Arrilaga, J., *High Voltage Direct Current Transmission,* IEEE® Power Engineering Series 6, Peter Peregrinus, Ltd., 1983.

[2] Lidong Zhang, Lars Dofnas, "A Novel Method to Mitigate Commutation Failures in HVDC Systems," *Proceedings PowerCon 2002. International Conference on,* Volume: 1, 13–17 Oct. 2002, pp. 51–56.

VSC-Based HVDC Link

In this section...

"Introduction" on page 5-63

"Description of the HVDC Link" on page 5-63

"VSC Control System" on page 5-67

"Dynamic Performance" on page 5-73

Introduction

The increasing rating and improved performance of self-commutated semiconductor devices have made possible High Voltage DC (HVDC) transmission based on Voltage-Sourced Converter (VSC). Two technologies offered by the manufacturers are the HVDC Light [1] and the HVDC^{plus} [2].

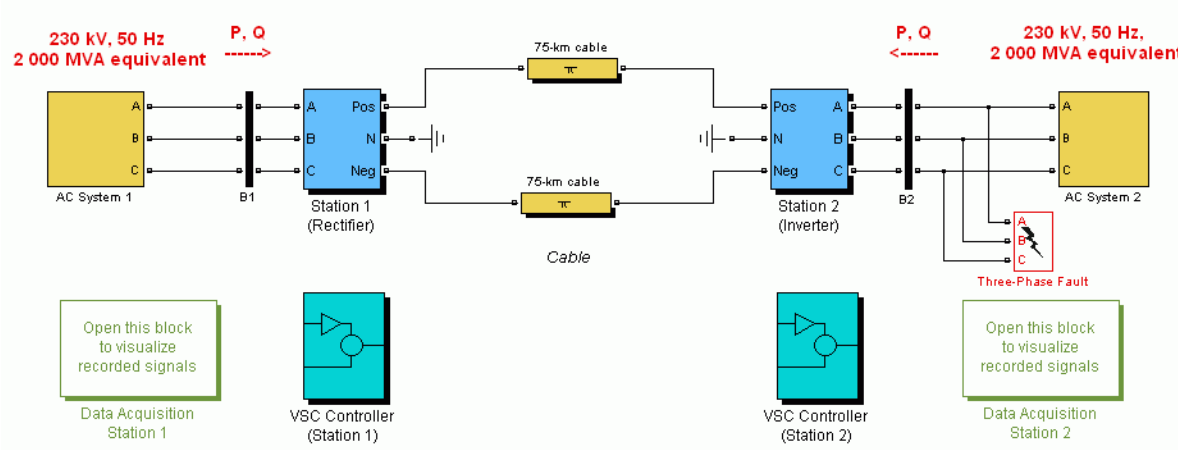
The example described in this section illustrates modeling of a forced-commutated Voltage-Sourced Converter high-voltage direct current (VSC-HVDC) transmission link. The objectives of this example are to demonstrate the use of SimPowerSystems™ blocks in the simulation of a HVDC transmission link based on three-level Neutral Point Clamped (NPC) VSC converters with single-phase carrier based Sinusoidal Pulse Width Modulation (SPWM) switching. Perturbations are applied to examine the system dynamic performance.

Description of the HVDC Link

The principal characteristic of VSC-HVDC transmission is its ability to independently control the reactive and real power flow at each of the AC systems to which it is connected, at the Point of Common Coupling (PCC). In contrast to line-commutated HVDC transmission, the polarity of the DC link voltage remains the same with the DC current being reversed to change the direction of power flow.

The HVDC link described in this example is available in the `power_hvdc_vsc` model. You can run the command by entering the following in the MATLAB® Command Window: `power_hvdc_vsc`. Load this model and save it in your working directory as `case5.mdl` to allow further modifications to the original

system. This model shown on VSC-HVDC Transmission System Model on page 5-64 represents a 200 MVA, +/- 100 kV VSC-HVDC transmission link.



VSC-HVDC Transmission System Model

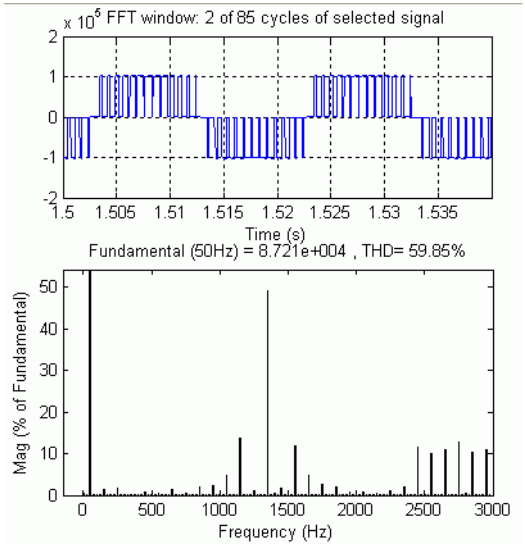
The 230 kV, 2000 MVA AC systems (AC system1 and AC system2 subsystems) are modeled by damped L-R equivalents with an angle of 80 degrees at fundamental frequency (50 Hz) and at the third harmonic. The VSC converters are three-level bridge blocks using close to ideal switching device model of IGBT/diodes. The relative ease with which the IGBT can be controlled and its suitability for high-frequency switching, has made this device the better choice over GTO and thyristors. Open the Station 1 and Station 2 subsystems to see how they are built.

A converter transformer (Wye grounded /Delta) is used to permit the optimal voltage transformation. The present winding arrangement blocks tripplen harmonics produced by the converter. The transformer tap changer or saturation are not simulated. The tap position is rather at a fixed position determined by a multiplication factor applied to the primary nominal voltage of the converter transformers The multiplication factors are chosen to have a modulation index around 0.85 (transformer ratios of 0.915 on the rectifier side and 1.015 on the inverter side). The converter reactor and the transformer leakage reactance permit the VSC output voltage to shift in phase and amplitude with respect to the AC system, and allows control of converter active and reactive power output.

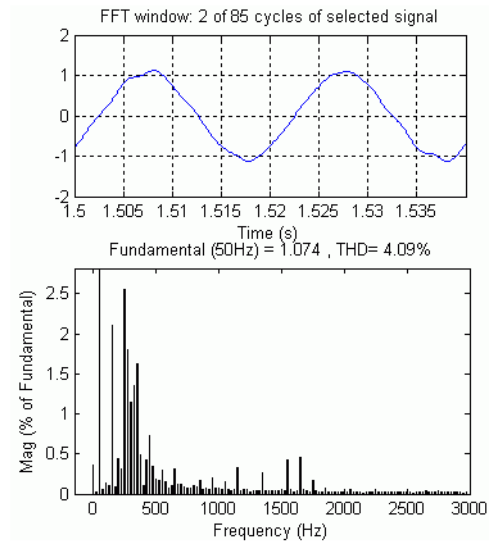
To meet AC system harmonic specifications, AC filters form an essential part of the scheme. They can be connected as shunt elements on the AC system side or the converter side of the converter transformer. Since there are only high frequency harmonics, shunt filtering is therefore relatively small compared to the converter rating. It is sufficient with a high pass-filter and no tuned filters are needed. The later arrangement is used in our model and a converter reactor, an air cored device, separates the fundamental frequency (filter bus) from the raw PWM waveform (converter bus). The AC harmonics generation [4] mainly depends on the:

- Type of modulation (e.g. single-phase or three-phase carrier based, space vector, etc.)
- Frequency index p = carrier frequency / modulator frequency (e.g. p = 1350/50 = 27)
- Modulation index m = fundamental output voltage of the converter / pole to pole DC voltage

The principal harmonic voltages are generated at and around multiples of p. The shunt AC filters are 27th and 54th high pass totaling 40 Mvar. To illustrate the AC filter action, we did an FFT analysis in steady state of the converter phase A voltage and the filter bus phase A voltage, using the Powergui block. The results are shown in Phase A Voltage and FFT Analysis: (a) Converter Bus (b) Filter Bus on page 5-66.



(a)



(b)

Phase A Voltage and FFT Analysis: (a) Converter Bus (b) Filter Bus

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 size of the capacitor is defined by the time constant τ corresponding to the time it takes to charge the capacitor to the base voltage (100 kV) if it is charged with the base current (1 kA). This yields

$$\tau = C \cdot Z_{base} = 70e\text{-}6 \cdot 100 = 7 \text{ ms}$$

with Z_{base} = 100kV/1 kA

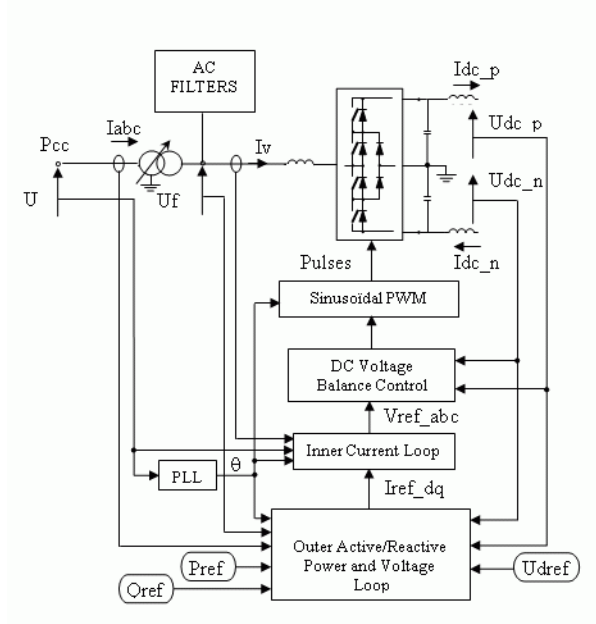
The DC side filters blocking high-frequency are tuned to the 3rd harmonic, i.e., the main harmonic present in the positive and negative pole voltages. It is shown that a reactive converter current generate a relatively large third harmonic in both the positive and negative pole voltages [3] but not in the total DC voltage. The DC harmonics can also be zero-sequence harmonics (odd multiples of 3) transferred to the DC side (e.g., through the grounded AC filters). A smoothing reactor is connected in series at each pole terminal.

To keep the DC side balanced, the level of the difference between the pole voltages has to be controlled and kept to zero (see the DC Voltage Balance Control block in the VSC Controller block).

The rectifier and the inverter are interconnected through a 75 km cable (2 pi sections). The use of underground cable is typical for VSC-HVDC links. 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.

VSC Control System

Overview of the Control System of a VSC Converter and Interface to the Main Circuit on page 5-67 shows an overview diagram of the VSC control system and its interface with the main circuit [3].



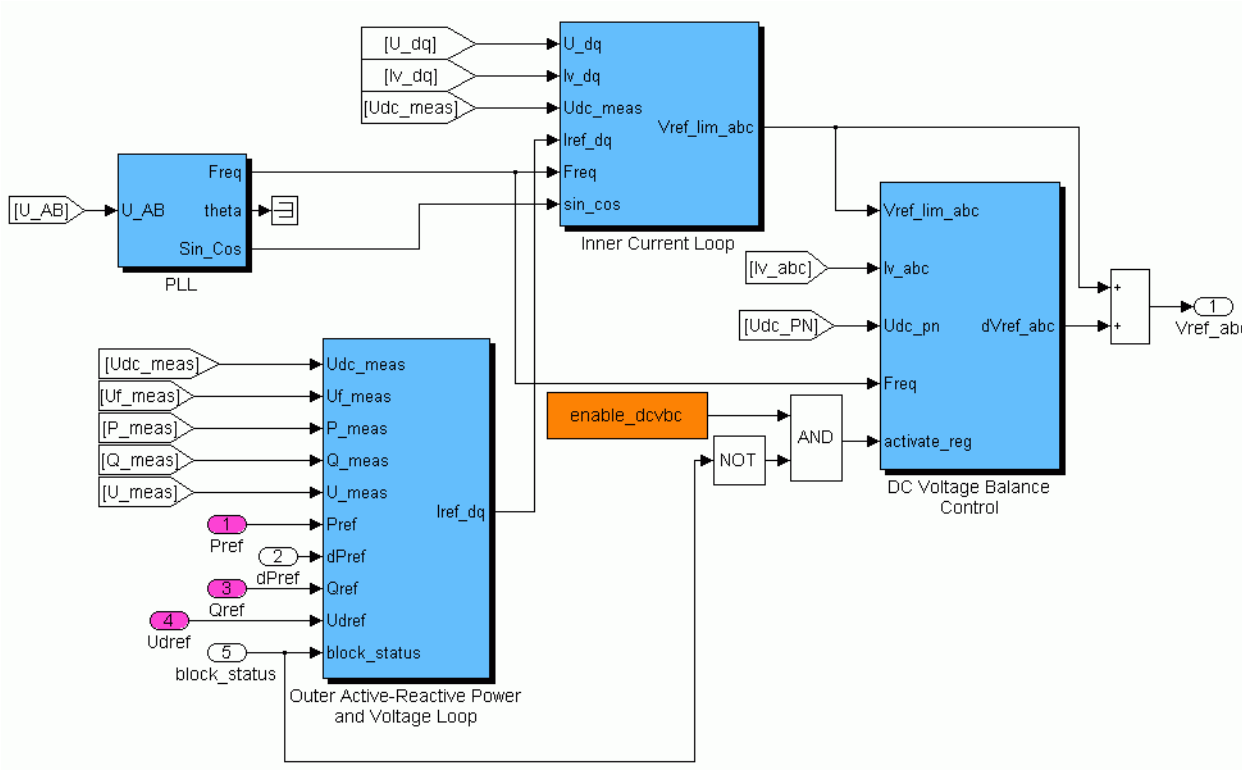
Overview of the Control System of a VSC Converter and Interface to the Main Circuit

The converter 1 and converter 2 controller designs are identical. The two controllers are independent with no communication between them. Each converter has two degrees of freedom. In our case, these are used to control:

- P and Q in station 1 (rectifier)
- Udc and Q in station 2 (inverter).

The control of the AC voltage would be also possible as an alternative to Q. This requires an extra regulator which is not implemented in our model.

A high level block diagram of the Simulink® discrete VSC controller model is shown in High Level Block Diagram of the Discrete VSC Controller on page 5-68.



High Level Block Diagram of the Discrete VSC Controller

Open the VSC Controller subsystem to see the details.

The sample time of the controller model (Ts_Control) is 74.06 µs, which is ten times the simulation sample time. The later is chosen to be one hundredth of the PWM carrier period (i.e., 0.01/1350 s) giving an acceptable simulation precision. The power elements, the anti-aliasing filters and the PWM Generator block use the fundamental sample time (Ts_Power) of 7.406 µs. The unsynchronized PWM mode of operation is chosen for our model.

The normalized sampled voltages and currents (in pu) are provided to the controller.

The Clark Transformations block transforms the three-phase quantities to space vector components α and β (real and imaginary part). The signal measurements (U and I) on the primary side are rotated by ±pi/6 according to the transformer connection (YD11 or YD1) to have the same reference frame with the signal measured on the secondary side of the transformer (see block CLARK YD).

The dq transformations block computes the direct axis "d" and the quadratic axis "q" quantities (two axis rotating reference frame) from the α and β quantities.

The Signal Calculations block calculates and filters quantities used by the controller (e.g., active and reactive power, modulation index, DC current and voltage, etc.).

Phase Locked Loop (PLL)

The Phase Locked Loop block measures the system frequency and provides the phase synchronous angle Θ (more precisely [sin(Θ), cos(Θ)]) for the dq Transformations block. In steady state, sin(Θ) is in phase with the fundamental (positive sequence) of the α component and phase A of the PCC voltage (Uabc).

Outer Active and Reactive Power and Voltage Loop

The active and reactive power and voltage loop contains the outer loop regulators that calculates the reference value of the converter current vector (Iref_dq) which is the input to the inner current loop. The control modes are:

in the “d” axis, either the active power flow at the PCC or the pole-to-pole DC voltage; in the “q” axis, the reactive power flow at the PCC. Note that, it would be also possible to add an AC voltage control mode at the PCC in the “q” axis. The main functions of the Active and reactive power and voltage loop are described below.

The Reactive Power Control regulator block combines a PI control with a feedforward control to increase the speed response. To avoid integrator wind-up the following actions are taken: the error is reset to zero, when the measured PCC voltage is less then a constant value (i.e., during an AC perturbation); when the regulator output is limited, the limitation error is fed back with the right sign, to the integrator input. The AC Voltage control override block, based on two PI regulators, will override the reactive power regulator to maintain the PCC AC voltage within a secure range, especially in steady-state.

The Active Power Control block is similar to the Reactive Power Control block. The extra Ramping block ramps the power order towards the desired value with an adjusted rate when the control is de-blocked. The ramped value is reset to zero when the converter is blocked. The DC Voltage control override block, based on two PI regulators, will override the active power regulator to maintain the DC voltage within a secure range, especially during a perturbation in the AC system of the station controlling the DC voltage.

The DC Voltage Control regulator block uses a PI regulator. The block is enabled when the Active Power Control block is disabled. The block output is a reference value, for the “d” component of converter current vector, for the Current Reference Limitation block.

The Current Reference Calculation block transforms the active and reactive power references, calculated by the P and Q controllers, to current references according to the measured (space vector) voltage at the filter bus. The current reference is estimated by dividing the power reference by the voltage (up to a minimum preset voltage value).

The current reference vector is limited to a maximum acceptable value (i.e., equipment dependent) by the Current Reference Limitation block. In power control mode, equal scaling is applied to the active and reactive power reference when a limit is imposed. In DC voltage control mode, higher priority

is given to the active power when a limit is imposed for an efficient control of the voltage.

Inner Current Loop

The main functions of Inner Current Loop block are described below.

The AC Current Control block tracks the current reference vector ("d" and "q" components) with a feed forward scheme to achieve a fast control of the current at load changes and disturbances (e.g., so short-circuit faults do not exceed the references) [3] [5] [6]. In essence, it consist of knowing the U_dq vector voltages and computing what the converter voltages have to be, by adding the voltage drops due to the currents across the impedance between the U and the PWM-VSC voltages. The state equations representing the dynamics of the VSC currents are used (an approximation is made by neglecting the AC filters). The "d" and "q" components are decoupled to obtain two independent first-order plant models. A proportional integral (PI) feedback of the converter current is used to reduce the error to zero in steady state. The output of the AC Current Control block is the unlimited reference voltage vector Vref_dq_tmp.

The Reference Voltage Conditioning block takes into account the actual DC voltage and the theoretical maximum peak value of the fundamental bridge phase voltage in relation to the DC voltage to generate the new optimized reference voltage vector. In our model (i.e., a three-level NPC with carrier based PWM), the ratio between the maximum fundamental peak phase voltage and the DC total voltage (i.e., for a modulation index of 1) is $(\sqrt{2})/(\sqrt{3})$ = 0.816. By choosing a nominal line voltage of 100 kV at the transformer secondary bus and a nominal total DC voltage of 200 kV the nominal modulation index would be 0.816. In theory, the converter should be able to generate up to 1/0.816 or 1.23 pu when the modulation index is equal to 1. This voltage margin is important for generating significant capacitive converter current (i.e., a reactive power flow to the AC system).

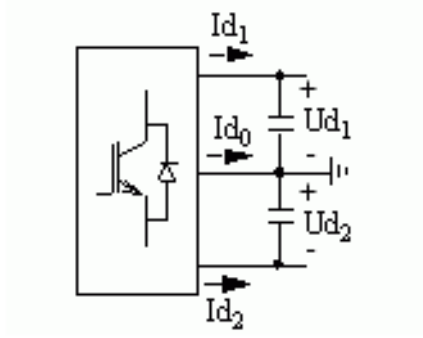
The Reference Voltage Limitation block limits the reference voltage vector amplitude to 1.0, since over modulation is not desired.

The Inverse dq and Inverse Clark transformation blocks are required to generate the three-phase voltage references to the PWM.

DC Voltage Balance Control

The DC Voltage Balance Control can be enabled or disabled. The difference between the DC side voltages (positive and negative) are controlled to keep the DC side of the three level bridge balanced (i.e., equal pole voltages) in steady-state. Small deviations between the pole voltages may occur at changes of active/reactive converter current or due to nonlinearity on lack of precision in the execution of the pulse width modulated bridge voltage. Furthermore, deviations between the pole voltages may be due to inherent unbalance in the circuit components impedance.

The DC midpoint current Id_0 determines the difference Ud_0 between the upper and lower DC voltages (DC Voltages and Currents of the Three-Level Bridge on page 5-72) .



$$Id_0 = -(Id_1 + Id_2) = -C \cdot \frac{d}{dt}(Ud_1 - Ud_2) = -C \cdot \frac{d}{dt}(Ud_0)$$

DC Voltages and Currents of the Three-Level Bridge

By changing the conduction time of the switches in a pole it is possible to change the average of the DC midpoint current Id_0 and thereby control the difference voltage Ud_0. For example, a positive difference ($Ud_0 \geq 0$) can be decreased to zero if the amplitude of the reference voltage which generates a positive midpoint current is increased at the same time as the amplitude of the reference voltage which generates a negative DC midpoint current is decreased. This is done by the addition of an offset component to the sinusoidal reference voltage. Consequently, the bridge voltage becomes distorted, and to limit the distortion effect, the control has to be slow. Finally, for better performance this function should be activated in the station controlling the DC voltage.

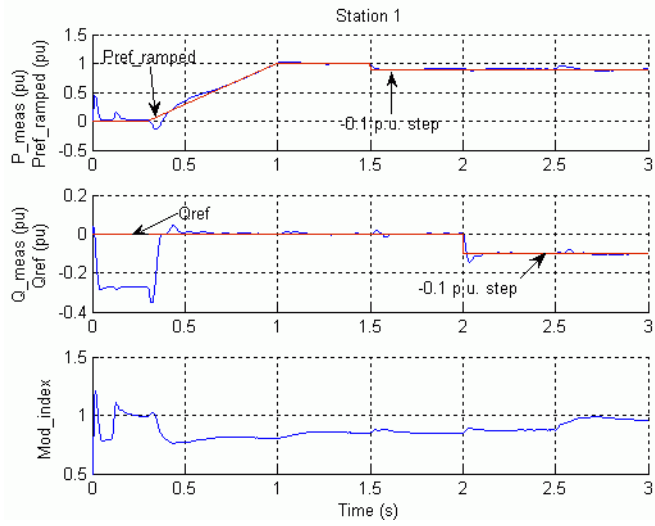
Dynamic Performance

In the next sections, the dynamic performance of the transmission system is verified by simulating and observing the

- Dynamic response to step changes applied to the principal regulator references, like active/reactive power and DC voltage
- Recovery from minor and severe perturbations in the AC system

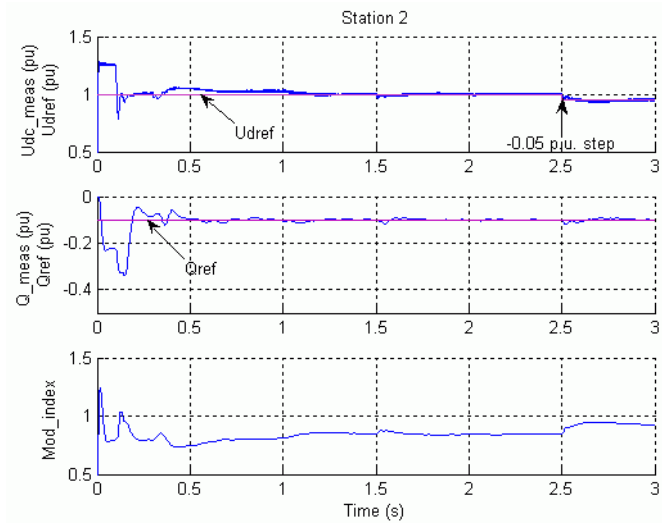
For a comprehensive explanation of the procedure followed obtaining these results and more, refer to the Model Information block.

System Startup - Steady-State and Step Response



Startup and P & Q Step Responses in Station 1

The main waveforms from the scopes are reproduced below.



Startup and Udc Step Response in Station 2

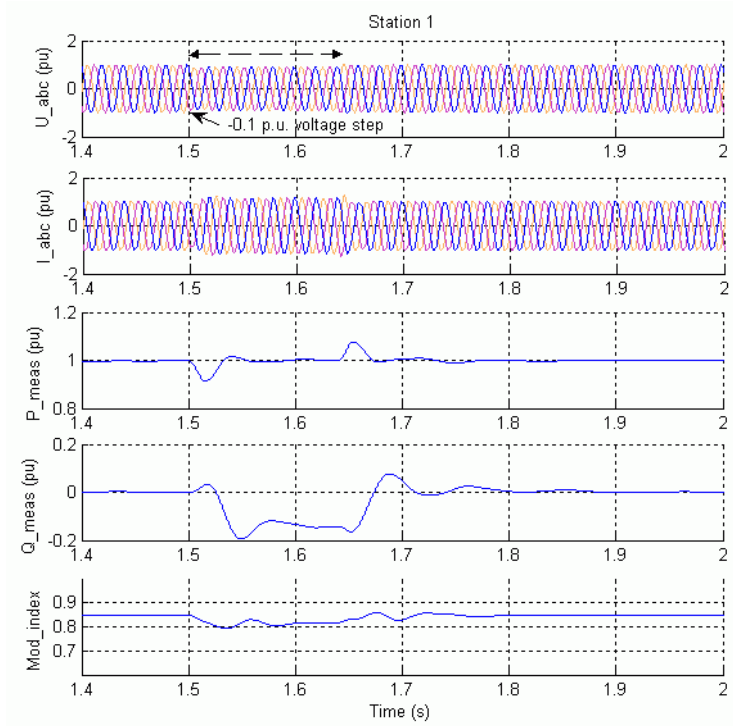
Station 2 converter controlling DC voltage is first deblocked at t=0.1 s. Then, station 1 controlling active power converter is deblocked at t=0.3 s and power is ramped up slowly to 1 pu. Steady state is reached at approximately t=1.3 s with DC voltage and power at 1.0 pu (200 kV, 200 MW). Both converters control the reactive power flow to a null value in station 1 and to 20 Mvar (-0.1 pu) into station 2 system.

After steady state has been reached, a -0.1 pu step is applied to the reference active power in converter 1 (t=1.5 s) and later a -0.1 pu step is applied to the reference reactive power (t=2.0 s). In station 2, a -0.05 pu step is applied to the DC voltage reference. The dynamic response of the regulators are observed. Stabilizing time is approximately 0.3 s.The control design attempts to decouple the active and reactive power responses. Note how the regulators are more or less mutually affected.

AC Side Perturbations

From the steady-state condition, a minor and a severe perturbation are executed at station 1 and 2 systems respectively. A three-phase voltage sag is first applied at station 1 bus. Then, following the system recovery, a three-phase to ground fault is applied at station 2 bus. The system recovery

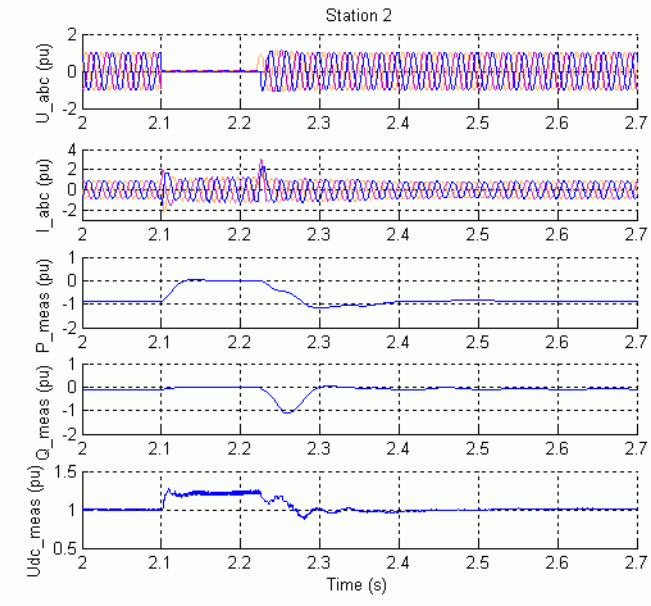
from the perturbations should be prompt and stable. The main waveforms from the scopes are reproduced in the two figures below.



Voltage Step on AC System 1

The AC voltage step (-0.1 pu) is applied at t=1.5 s during 0.14 s (7 cycles) at station 1. The results show that the active and reactive power deviation from the pre-disturbance is less than 0.09 pu and 0.2 pu respectively. The recovery time is less than 0.3 s and the steady state is reached before next perturbation initiation.

The fault is applied at t=2.1 s during 0.12 s (6 cycles) at station 2.



Three-Phase to Ground Fault at Station 2 Bus

Note that during the three-phase fault the transmitted DC power is almost halted and the DC voltage tends to increase (1.2 pu) 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. The system recovers well after the fault, within 0.5 s. Note the damped oscillations (around 10 Hz) in the reactive power.

References

[1] Weimers, L. "A New Technology for a Better Environment," *Power Engineering Review, IEEE®*, vol. 18, issue 8, Aug. 1998.

[2] Schettler F., Huang H., and Christl N. "HVDC transmission systems using voltage source converters – design and applications," *IEEE Power Engineering Society Summer Meeting*, July 2000.

[3] Lindberg, Anders "PWM and control of two and three level high power voltage source converters," *Licentiate thesis, ISSN-1100-1615, TRITA-EHE 9501,* The Royal Institute of Technology, Sweden, 1995.

[4] Sadaba, Alonso, O., P. Sanchis Gurpide, J. Lopez Tanerna, I. Munoz Morales, L. Marroyo Palomo, "Voltage Harmonics Generated by 3-Level Converters Using PWM Natural Sampling," *Power Electronics Specialist Conference, 2001, IEEE 32nd Annual,* 17–21 June 2001, vol. 3, pp. 1561–1565.

[5] Lu, Weixing, Boon-Teck Ooi, "Optimal Acquisition and Aggregation of Offshore wind Power by Multiterminal Voltage-Source HVDC," *IEEE Trans. Power Delivery,* vol. 18, pp. 201–206, Jan. 2003.

[6] Sao, K., P.W. Lehn, M.R. Iravani, J.A. Martinez, "A benchmark system for digital time-domain simulation of a pulse-width-modulated D-STATCOM," *IEEE Trans. Power Delivery,* vol. 17, pp. 1113–1120, Oct. 2002.

6

Transient Stability of Power Systems Using Phasor Simulation

These case studies provide detailed, realistic examples of how to use the phasor simulation method of SimPowerSystems™ software in typical power utility applications.

As explained in the section "Using the Phasor Solution Method for Stability Studies" on page 2-35, phasor simulation is the preferred method for simulating power grids when you are interested in the magnitude and phase of voltages and currents at fundamental frequency (50 Hz or 60 Hz). The phasor simulation is activated by means of the Powergui block. It supports all the elements of the `powerlib` library, including machines. In addition, SimPowerSystems software contains two libraries of phasor models of power equipments found in utility grids (some of them including power electronics): the Flexible AC Transmission Systems (FACTS) library (`factslib`) and the Distributed Resources (DR) library (`drlib`). The case studies listed below show application examples of some of these phasor models.

Transient Stability of a Power System with SVC and PSS (p. 6-3)	Use of a static var compensator (SVC) and power system stabilizers (PSS) for improving stability of a two-machine transmission system
Control of Power Flow Using a UPFC and a PST (p. 6-10)	Control of power flow using a unified power flow controller (UPFC) and a phase shifting transformer (PST)
Wind Farm Using Doubly-Fed Induction Generators (p. 6-20)	Study of a wind farm using doubly-fed induction generators driven by wind turbines

Transient Stability of a Power System with SVC and PSS

In this section...

"Introduction" on page 6-3

"Description of the Transmission System" on page 6-3

"Single-Phase Fault — Impact of PSS — No SVC" on page 6-5

"Three-Phase Fault — Impact of SVC — PSS in Service" on page 6-7

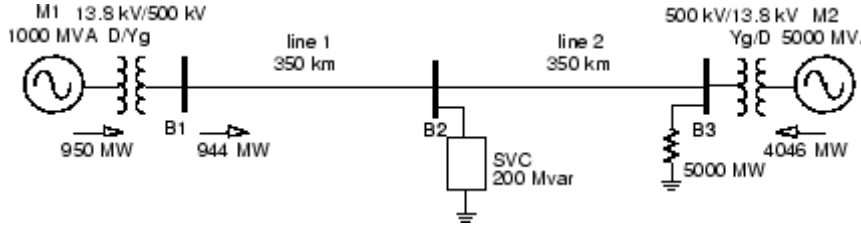
Introduction

The example described in this section illustrates modeling of a simple transmission system containing two hydraulic power plants. A static var compensator (SVC) and power system stabilizers (PSS) are used to improve transient stability and power oscillation damping of the system. The power system illustrated in this example is quite simple. However, the phasor simulation method allows you to simulate more complex power grids.

If you are not familiar with the SVC and PSS, please see the reference pages for the following blocks: Static Var Compensator (Phasor Type), Generic Power System Stabilizer, and Multiband Power System Stabilizer.

Description of the Transmission System

The single line diagram shown below represents a simple 500 kV transmission system.



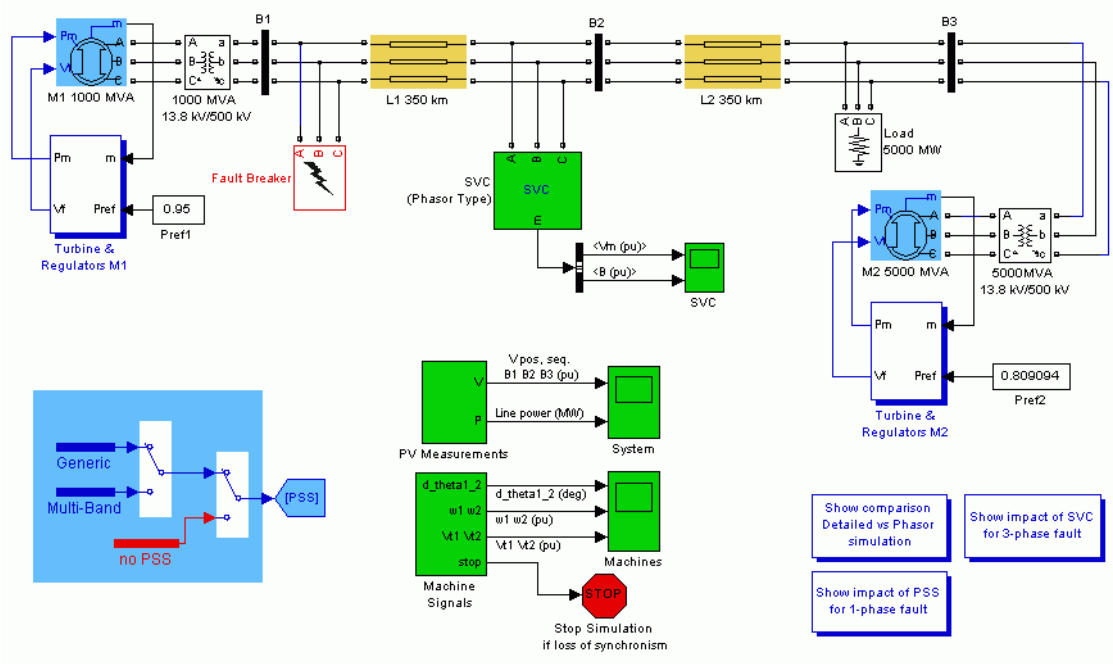
500 kV Transmission System

A 1000 MW hydraulic generation plant (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 MVA plant and a local generation of 5000 MVA (plant M2).

A load flow has been performed on this system with plant M1 generating 950 MW so that plant M2 produces 4046 MW. The line carries 944 MW which is close to its surge impedance loading (SIL = 977 MW). To maintain system stability after faults, the transmission line is shunt compensated at its center by a 200 Mvar static var compensator (SVC). 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).

This system is available in the `power_svc_pss` model. Load this model and save it in your working directory as `case1.mdl` to allow further modifications to the original system. This model is shown in Model of the Transmission System (power_svc_pss) on page 6-4



Model of the Transmission System (power_svc_pss)

.

First, look inside the two Turbine and Regulators subsystems to see how the HTG and the excitation system are implemented. Two types of stabilizers can be connected on the excitation system: a generic model using the acceleration power (Pa= difference between mechanical power Pm and output electrical power Peo) and a Multiband stabilizer using the speed deviation (dw). These two stabilizers are standard models of the **powerlib**/Machines library. Manual Switch blocks surrounded by a blue zone allow you to select the type of stabilizer used for both machines or put the PSS out of service.

The SVC is the phasor model from the FACTS library. Open its dialog box and check in the Power data parameters that the SVC rating is +/- 200 Mvar. In the Control parameters, you can select either Voltage regulation or Var control (Fixed susceptance Bref) mode. Initially the SVC is set in Var control mode with a susceptance Bref=0, which is equivalent to having the SVC out of service.

A Fault Breaker block is connected at bus B1. You will use it to program different types of faults on the 500 kV system and observe the impact of the PSS and SVC on system stability.

To start the simulation in steady-state, the machines and the regulators have been previously initialized by means of the Load Flow and Machine Initialization utility of the Powergui block. Load flow has been performed with machine M1 defined as a PV generation bus (V=13800 V, P=950 MW) and machine M2 defined as a swing bus (V=13800 V, 0 degrees). After the load flow has been solved, the reference mechanical powers and reference voltages for the two machines have been automatically updated in the two constant blocks connected at the HTG and excitation system inputs: Pref1=0.95 pu (950 MW), Vref1=1.0 pu; Pref2=0.8091 pu (4046 MW), Vref2=1.0 pu.

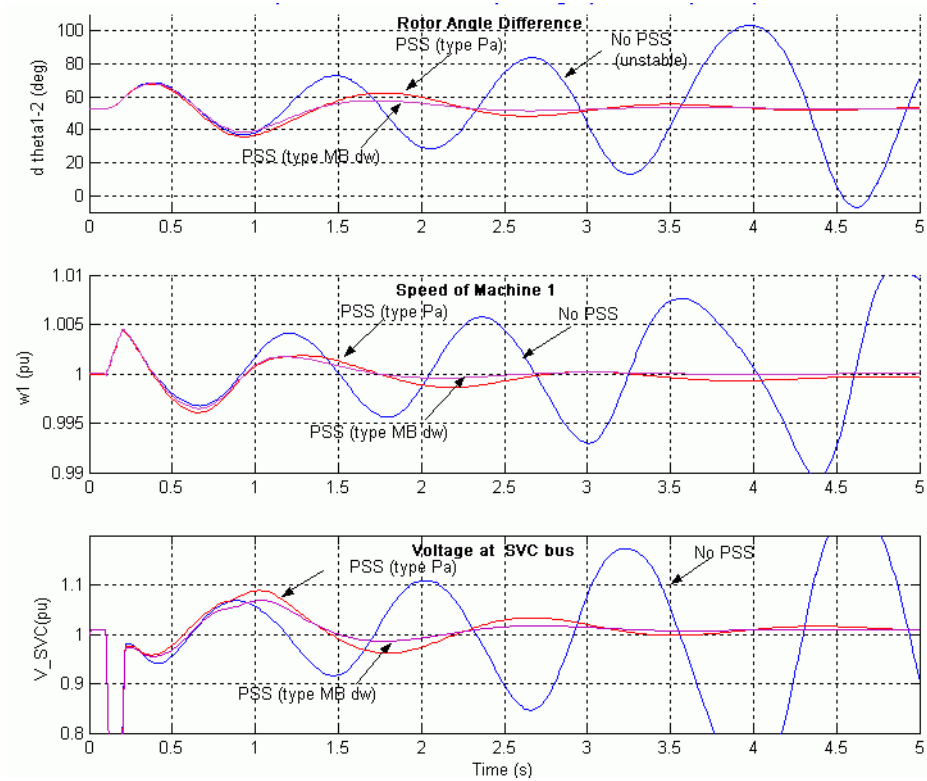
Single-Phase Fault — Impact of PSS — No SVC

Verify that the PSSs (Generic Pa type) are in service and that a 6-cycle single-phase fault is programmed in the Fault Breaker block (Phase A checked, fault applied at t=0.1 s and cleared at t=0.2 s).

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 0.6 Hz oscillation is quickly damped. This oscillation mode is typical of interarea oscillations in a large power system. First trace on the Machines 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 too long a 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 PSSs (Pa type) succeed to damp the 0.6 Hz mode but they are not efficient for damping the 0.025 Hz mode. If you select instead the Multi-Band PSS, you will notice that this stabilizer type succeeds to damp both the 0.6 Hz mode and the 0.025 Hz mode.

You will now repeat the test with the two PSSs out of service. Restart simulation. Notice that the system is unstable without PSS. You can compare results with and without PSS by double-clicking on the blue block on the right side labeled “Show impact of PSS for 1-phase fault.” The displayed waveforms are reproduced below.



Impact of PSS for a Single-Phase Fault

Note This system is naturally unstable without PSS. If you remove the fault (by deselecting phase A in the Fault Breaker), you will see the instability slowly building up at approximately 1 Hz after a few seconds.

Three-Phase Fault — Impact of SVC — PSS in Service

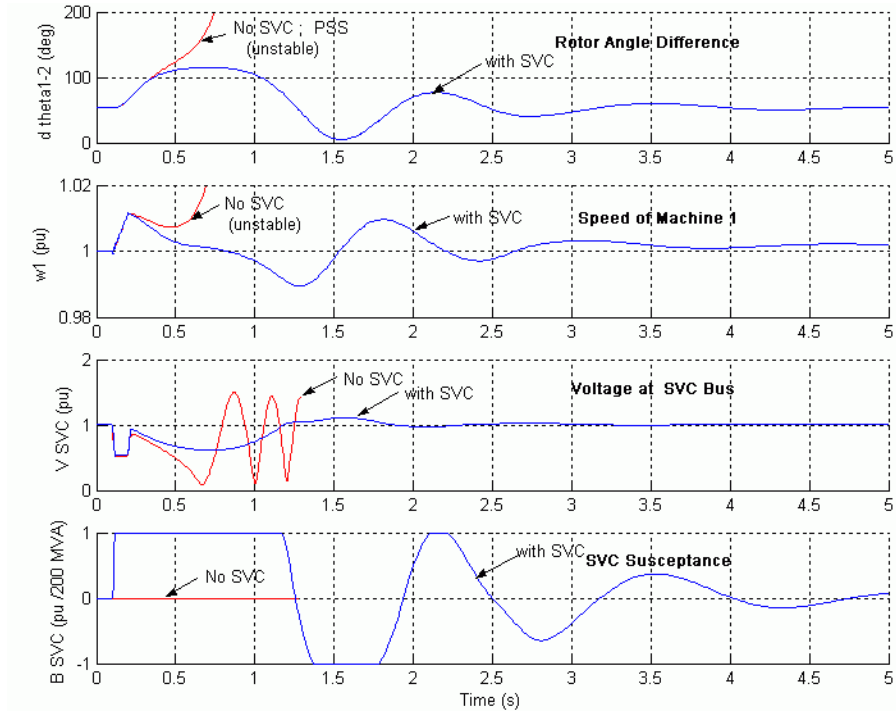
You will now apply a 3-phase fault and observe the impact of the SVC for stabilizing the network during a severe contingency.

First put the two PSS (Generic Pa type) in service. Reprogram the Fault Breaker block 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*360 degrees.

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 blue block labeled “Show impact of SCV for 3-phase fault.” The displayed waveforms are reproduced below.



Impact of the SVC for a Three-Phase Fault

Control of Power Flow Using a UPFC and a PST

In this section...

"Introduction" on page 6-10

"Description of the Power System" on page 6-10

"Power Flow Control with the UPFC" on page 6-13

"UPFC P-Q Controllable Region" on page 6-14

"Power Flow Control Using a PST" on page 6-15

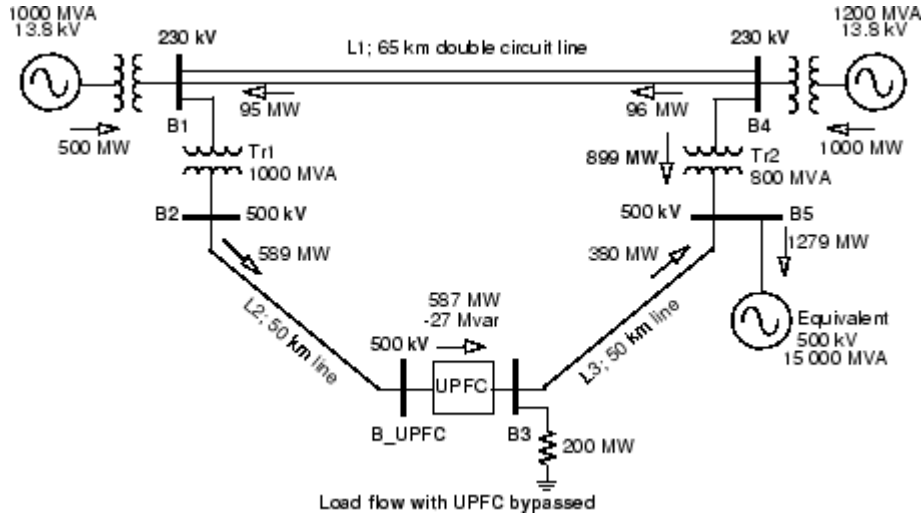
Introduction

The example described in this section illustrates application of SimPowerSystems™ software to study the steady-state and dynamic performance of a unified power flow controller (UPFC) used to relief power congestion in a transmission system.

If you are not familiar with the UPFC, please see the reference page for the Unified Power Flow Controller (Phasor Type) block.

Description of the Power System

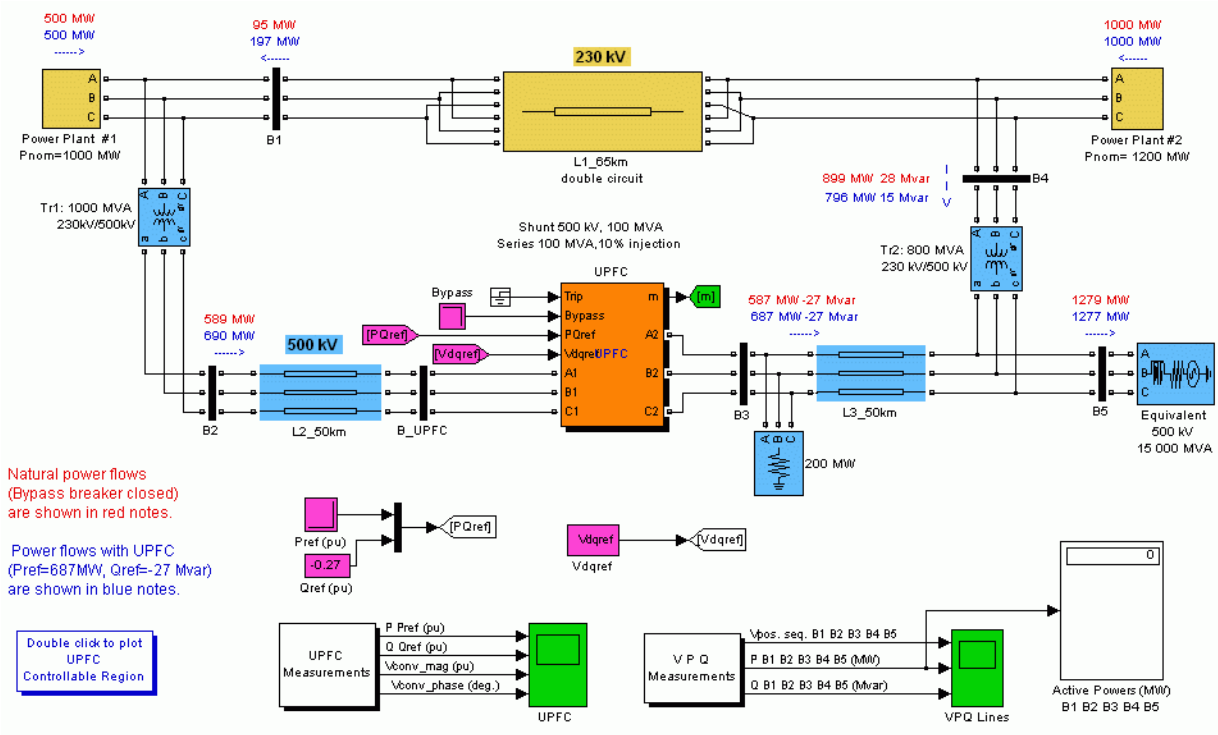
The single-line diagram of the modeled power system is shown in 500 kV / 230 kV Transmission System on page 6-11.



500 kV / 230 kV Transmission System

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 three 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. Each plant model includes 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 two 400 MVA transformers connected between buses B4 and B5. For this demo we are considering a contingency case where only two transformers out of three are available (Tr2= 2*400 MVA = 800 MVA). 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) and that 96 MW is circulating in the loop. Transformer Tr2 is therefore overloaded by 99 MVA. The example illustrates how a UPFC can relief 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. The UPFC consists of two 100 MVA, IGBT-based, converters (one shunt converter and one series converter interconnected 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 example is available in the `power_upfc` model. Load this model and save it in your working directory as `case2.mdl` to allow further modifications to the original system. This model is shown in Model of the UPFC Controlling Power on a 500 kV/230 kV Power System (power_upfc) on page 6-12.



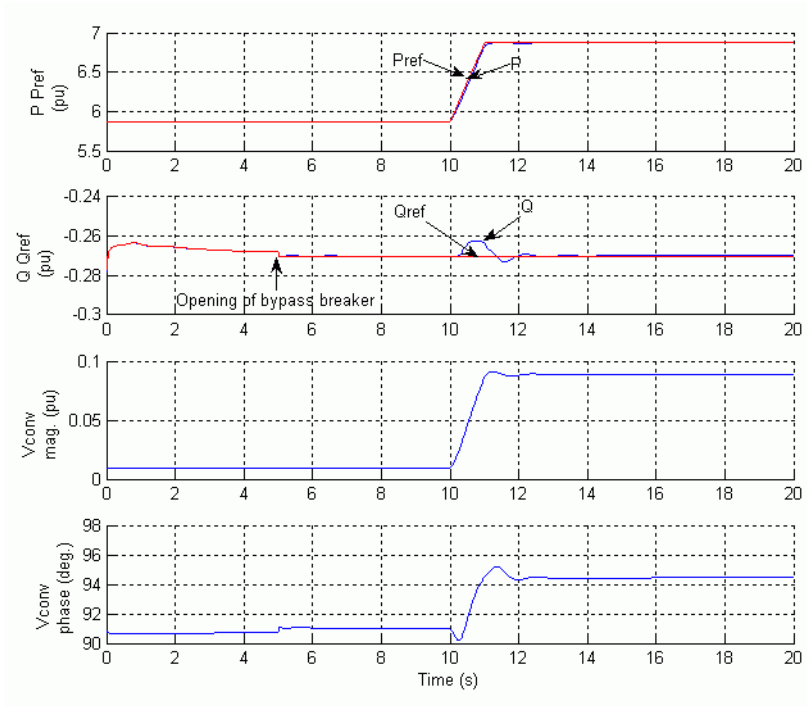
Model of the UPFC Controlling Power on a 500 kV/230 kV Power System (power_upfc)

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 with the UPFC out of service (Bypass breaker closed). The resulting power flow obtained at buses B1 to B5 is indicated on the model by red numbers. This load flow corresponds to load flow shown in the single-line diagram, in 500 kV / 230 kV Transmission System on page 6-11.

Power Flow Control with the UPFC

Parameters of the UPFC are given in the dialog box. Verify, in the Power data parameters, that the series converter is rated 100 MVA with a maximum voltage injection of 0.1 pu. The shunt converter is also rated 100 MVA. Also verify, in the control parameters, that the shunt converter is in Voltage regulation mode and that the series converter is in Power flow control mode. The UPFC reference active and reactive powers are set in the magenta 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 power 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. Waveforms are reproduced below.



UPFC Dynamic Response to a Change in Reference Power from 587 MW to 687 MW

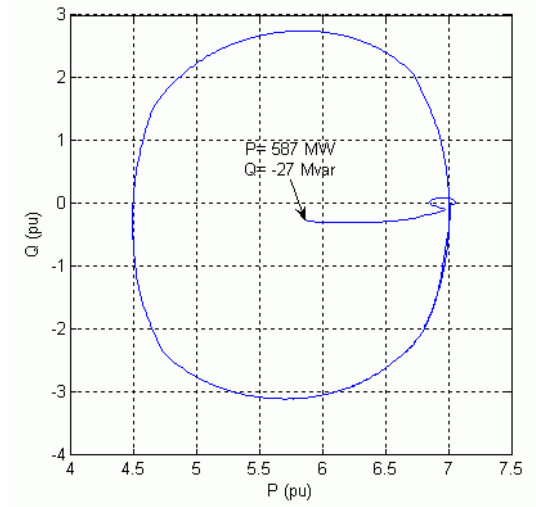
At t=5 s, when the Bypass breaker is opened, the natural power is diverted from the Bypass breaker to the UPFC series branch without noticeable transient. At t=10 s, the power increases at a rate of 1 pu/s. It takes one second for the power to increase to 687 MW. This 100 MW increase of active power 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 P-Q Controllable Region

Now, open the UPFC dialog box and select Show Control parameters (series converter). Select Mode of operation = Manual Voltage injection. In this

control mode the voltage generated by the series inverter is controlled by two external signals Vd, Vq multiplexed at the Vdqref input and generated in the Vdqref magenta block. For the first five seconds the Bypass breaker stays closed, so that the PQ trajectory stays at the (-27Mvar, 587 MW) point. Then when the breaker opens, the magnitude of the injected series voltage is ramped, from 0.0094 to 0.1 pu. At 10 s, the angle of the injected voltage starts varying at a rate of 45 deg/s.

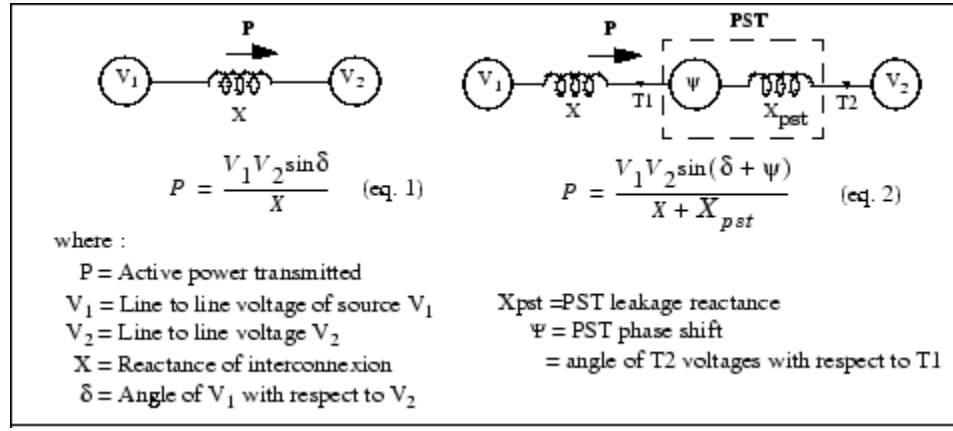
Run the simulation and look on the UPFC scope the P and Q signals who vary according to the changing phase of the injected voltage. At the end of the simulation, double-click on the blue block labeled “Double click to plot UPFC Controllable Region.” The trajectory of the UPFC reactive power as function of its active power, measured at bus B3, is reproduced below. The area located inside the ellipse represents the UPFC controllable region.



UPFC Controllable Region

Power Flow Control Using a PST

Although not as flexible as the UPFC, the phase shifting transformer (PST) is nevertheless a very efficient means to control power flow because it acts directly on the phase angle δ, as shown in Power Transfer Between Two Voltage Sources Without and With PST on page 6-16. The PST is the most commonly used device to control power flow on power grids.



Power Transfer Between Two Voltage Sources Without and With PST

You will now use a PST with an on load tap changer (OLTC) to control the power flow on your power system. A phasor model of PST using the delta hexagonal connection is available in the FACTS/Transformers library. For details on this PST connection, please refer to the Three-Phase OLTC Phase Shifting Transformer Delta-Hexagonal (Phasor Type) block reference page.

Delete the UPFC block in your model as well as the magenta blocks controlling the UPFC. Also delete the UPFC Measurements subsystem and the UPFC scope. Open the Transformer subsystem of the FACTS library and copy the Three-Phase OLTC Phase Shifting Transformer Delta-Hexagonal (Phasor Type) block in your model. Connect the ABC terminals to the B_UPFC bus and connect the abc terminals to the B3 bus. Now, open the PST block dialog box and modify the following parameters:

Nominal parameters [Vnom(Vrms Ph Ph) Pnom(VA) Fnom (Hz)] `[500e3 800e6 60]`

Number of taps per half tapped winding `20`

The nominal power is set to 800 MVA (maximum expected power transfer through the PST). The number of taps is set to 20, so that the phase shift resolution is approximately 60/20 = 3 degrees per step.

In the power system, the natural power flow (without PST) from B_UPFC to B3 is P=+587 MW. If V1and V2 in Power Transfer Between Two Voltage Sources Without and With PST on page 6-16 represent the internal voltages of systems connected respectively to B_UPFC and B3, it means that the angle δ of equation 1 is positive. Therefore, according to equation 2, to increase power flow from B_UPFC to B3, the PST phase shift Ψof abc terminals with respect to ABC terminals must be also positive. For this type of PST the taps must be moved in the negative direction. This is achieved by sending pulses to the Down input of the PST tap changer.

The tap position is controlled by sending pulses to either the Up input or the Down input. In our case, as we need to increase phase shift from zero toward positive values, we have to send pulses to the Down input. Copy a Pulse Generator block from the Simulink® Sources library and connect it to the Down input of the PST. Open the block dialog box and modify the following parameters:

Period (secs)	5
Pulse Width (% of period)	10

Therefore, every 5 seconds the taps will be moved by one step in the negative direction and the phase shift will increase by approximately 3 degrees.

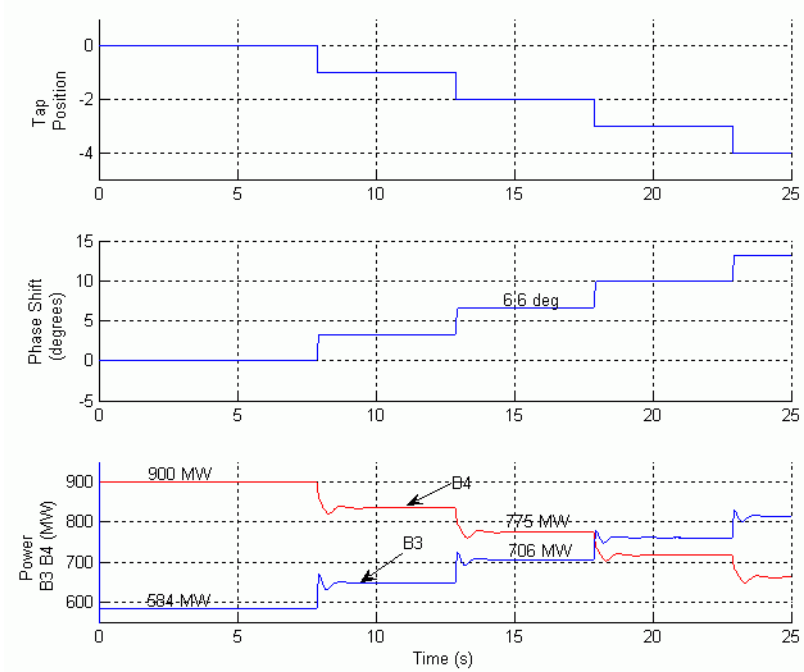
Finally, connect a Bus Selector block (from the Simulink Signal Routing library) to the measurement output m of the PST. Open its dialog box and select the following two signals:

- Tap
- Psi (degrees)

Connect these two signals to a two input scope to observe the tap position and the phase shift during simulation. Set the simulation time to 25 s and start simulation.

On the VPQ lines scope, observe voltages at buses B1 to B5 and active and reactive power transfer through these buses. The variation of tap position, PST phase shift Ψand active power transfer through bus B3 (power through

PST) and B4 (power through transformer Tr2) are reproduced on the figure below.

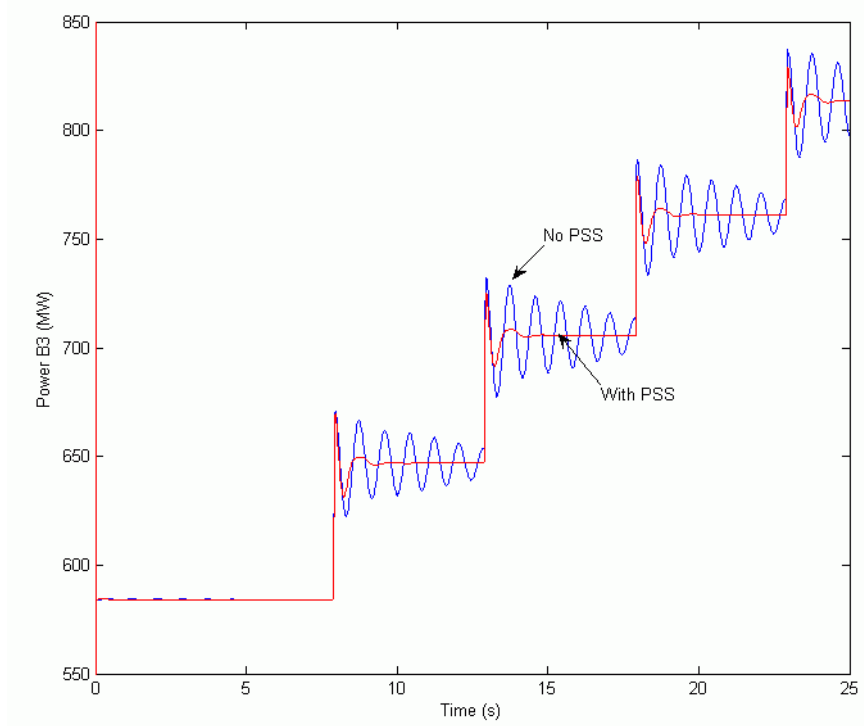


Control of Active Power Through B3 and B4 by Changing Tap Position of PST

Each tap change produces a phase angle variation of approximately 3 degrees, resulting in a 60 MW power increase through B3. At tap position -2, the power through transformer Tr2 as decreased from 900 MW to 775 MW, thus achieving the same goal as the UPFC for steady state control. You could get a better resolution in phase angle and power steps by increasing the number of taps in the OLTC.

You can notice that the discrete variation of phase angle produces overshoots and slight oscillations in active power. These power oscillations which are typical interarea electromechanical oscillations of machines in power plants 1 and 2 are quickly damped by the power system stabilizers (PSSs) connected on the excitation systems.

If you disconnect the PSS from the vstab input of the excitation system (located in the Reg_M1 and Reg_M2 subsystems of the power plants) you will realize the impact of PSS on interarea oscillation damping. The active power through B3 with and without PSS is reproduced below. Without PSS, the 1.2 Hz under damped power oscillations are clearly unacceptable.



Damping of Power Oscillations by PSS

Wind Farm Using Doubly-Fed Induction Generators

In this section...

"Description of the Wind Farm" on page 6-20

"Turbine Response to a Change in Wind Speed" on page 6-24

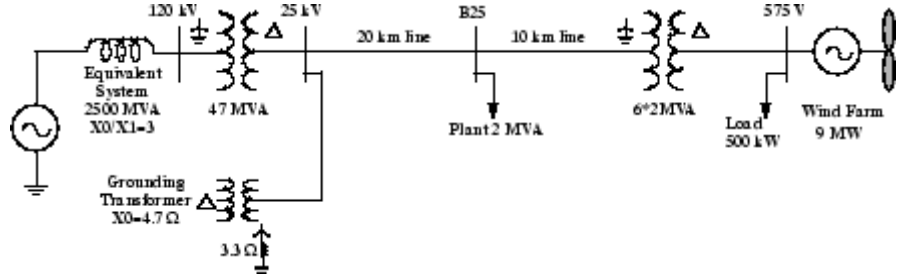
"Simulation of a Voltage Sag on the 120 kV System" on page 6-26

"Simulation of a Fault on the 25 kV System" on page 6-28

Description of the Wind Farm

The example described in this section illustrates application of SimPowerSystems™ software to study the steady-state and dynamic performance of a 9 MW wind farm connected to a distribution system.

The wind farm consists of six 1.5 MW wind turbines connected to a 25 kV distribution system exporting 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. A 500 kW load is also connected on the 575 V bus of the wind farm. The single-line diagram of this system is illustrated in Single-Line Diagram of the Wind Farm Connected to a Distribution System on page 6-20.

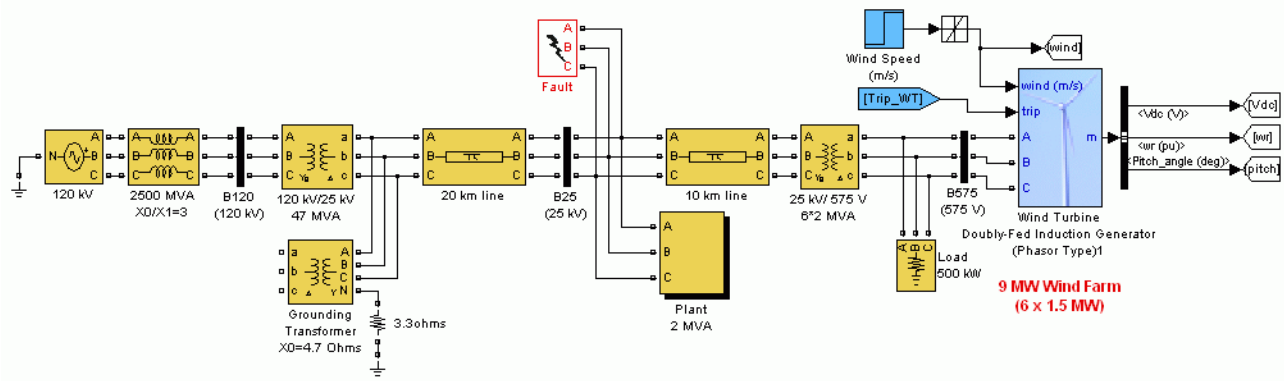


Single-Line Diagram of the Wind Farm Connected to a Distribution System

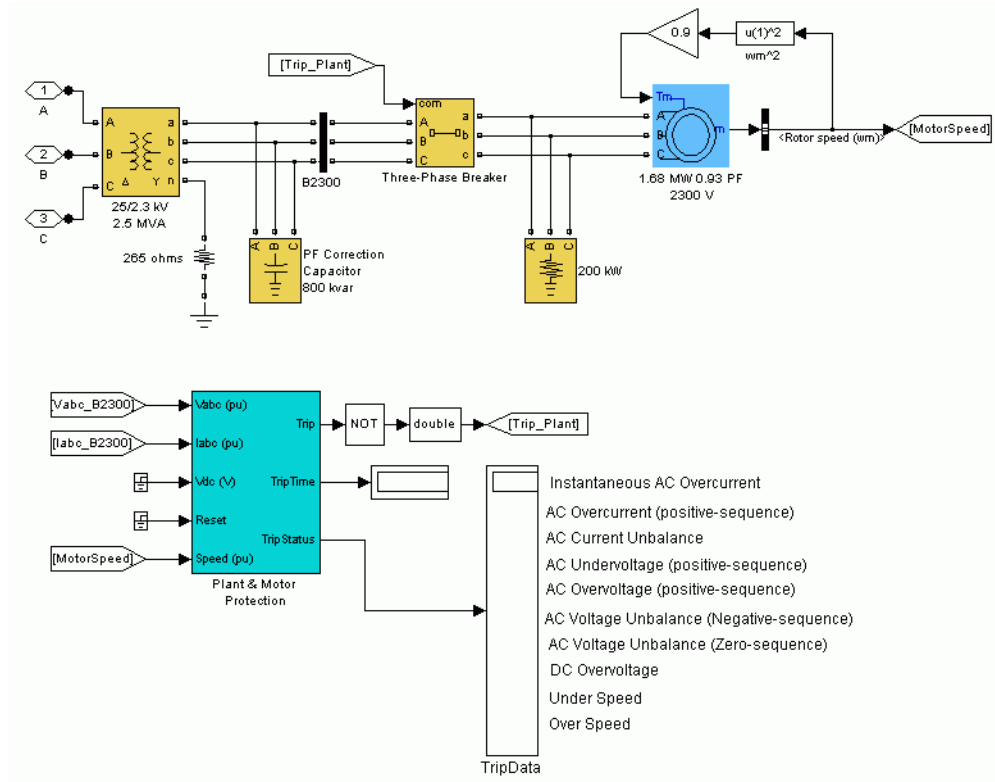
Both the wind turbine and the motor load have a protection system monitoring voltage, current and machine speed. The DC link voltage of the DFIG is also monitored. Wind turbines use a doubly-fed induction generator (DFIG) consisting of a wound rotor induction generator and an AC/DC/AC IGBT-based PWM 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 (see Wind Turbine Doubly-Fed Induction Generator (Phasor Type) block of the DRlib/Wind Generation library for more details). Another advantage of the DFIG technology is the ability for power electronic converters to generate or Turbine Data Menu and the Turbine Power Characteristics on page 6-23 absorb reactive power, thus eliminating the need for installing capacitor banks as in the case of squirrel-cage induction generators.

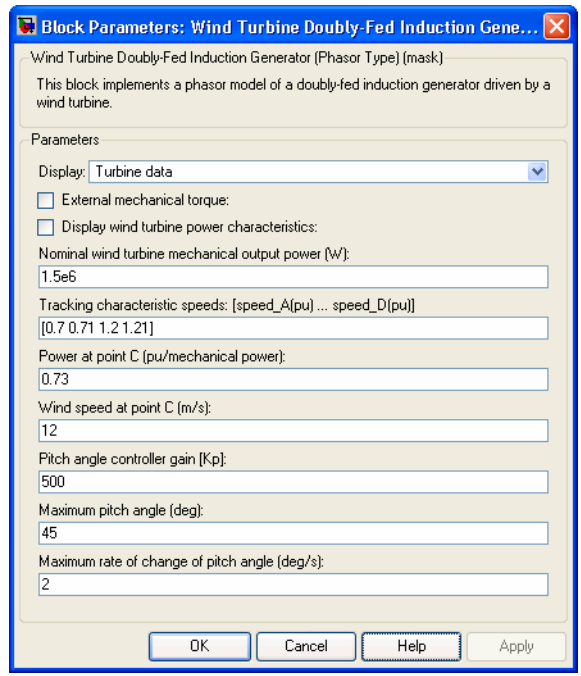
This system is available in the `power_wind_dfig` model. Load this model and save it in your working directory as `case3.mdl` to allow further modifications to the original system. The SimPowerSystems diagram is shown in Single-Line Diagram of the Wind Farm Connected to a Distribution System on page 6-20 and SimPowerSystems™ Diagram of the 2 MVA Plant with Its Protection System on page 6-22. In this case study, the rotor is running at subsynchronous speed for wind speeds lower than 10 m/s and it is running at a super-synchronous speed for higher wind speeds. The turbine mechanical power as function of turbine speed is displayed in for wind speeds ranging from 5 m/s to 16.2 m/s. These characteristics are obtained with the specified parameters of the `Turbine data` (Turbine Data Menu and the Turbine Power Characteristics on page 6-23).

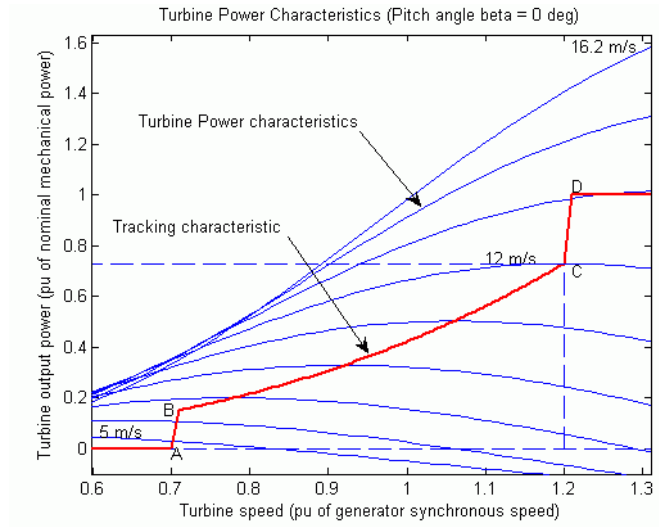


SimPowerSystems™ Diagram of the Wind Farm Connected to the Distribution System (power_wind_dfig)



SimPowerSystems™ Diagram of the 2 MVA Plant with Its Protection System





Turbine Data Menu and the Turbine Power Characteristics

The DFIG is controlled to follow the ABCD curve in Turbine Data Menu and the Turbine Power Characteristics on page 6-23. Turbine speed optimization is obtained between point B and point C on this curve.

The wind turbine model is a phasor model that allows transient stability type studies with long simulation times. In this case study, the system is observed during 50 s. The 6-wind-turbine farm is simulated by a single wind-turbine block by multiplying the following three parameters by six, as follows:

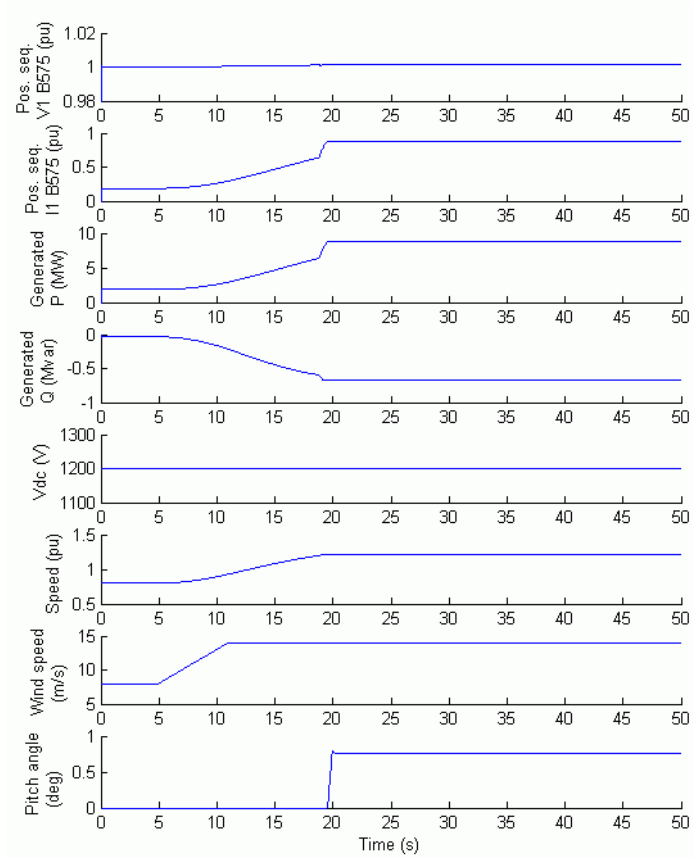
- The nominal wind turbine mechanical output power: 6*1.5e6 watts, specified in the `Turbine data` menu
- The generator rated power: 6*1.5/0.9 MVA (6*1.5 MW at 0.9 PF), specified in the `Generator data` menu
- The nominal DC bus capacitor: 6*10000 microfarads, specified in the `Converters data` menu

The mode of operation is set to `Voltage regulation` in the **Control Parameters** dialog box. 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).

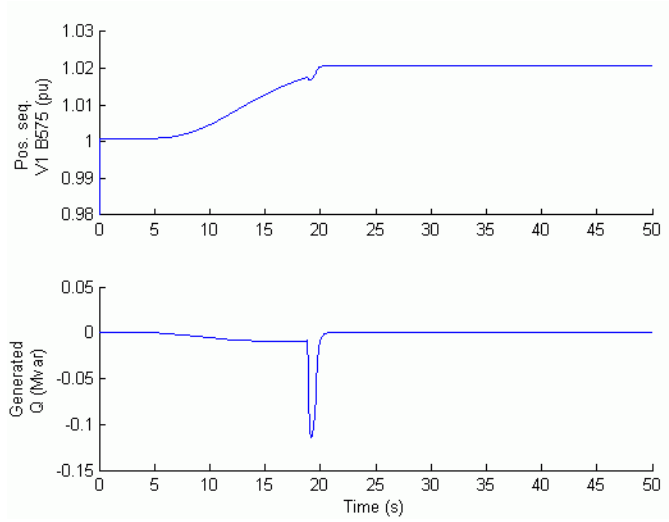
Turbine Response to a Change in Wind Speed

Observe the turbine response to a change in wind speed. Initially, wind speed is set at 8 m/s, and then at t=5s, wind speed increases suddenly at 14 m/s. Waveforms for a Gust of Wind (Wind Farm in Voltage Regulation Mode) on page 6-25 illustrates the waveforms associated with this simulation. At t=5 s, the generated active power starts increasing smoothly (together with the turbine speed) to reach its rated value of 9MW in approximately 15s. Over that time frame the turbine speed increases 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 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 the voltage increases to 1.021 pu when the wind turbine generates its nominal

power at unity power factor (Waveforms for a Gust of Wind (Wind Farm in Var Regulation Mode) on page 6-26).



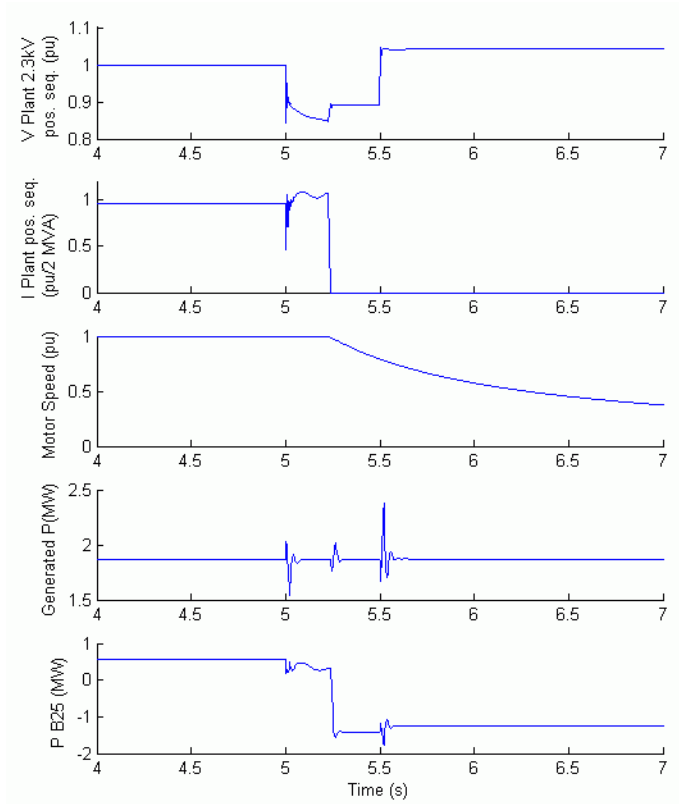
Waveforms for a Gust of Wind (Wind Farm in Voltage Regulation Mode)



Waveforms for a Gust of Wind (Wind Farm in Var Regulation Mode)

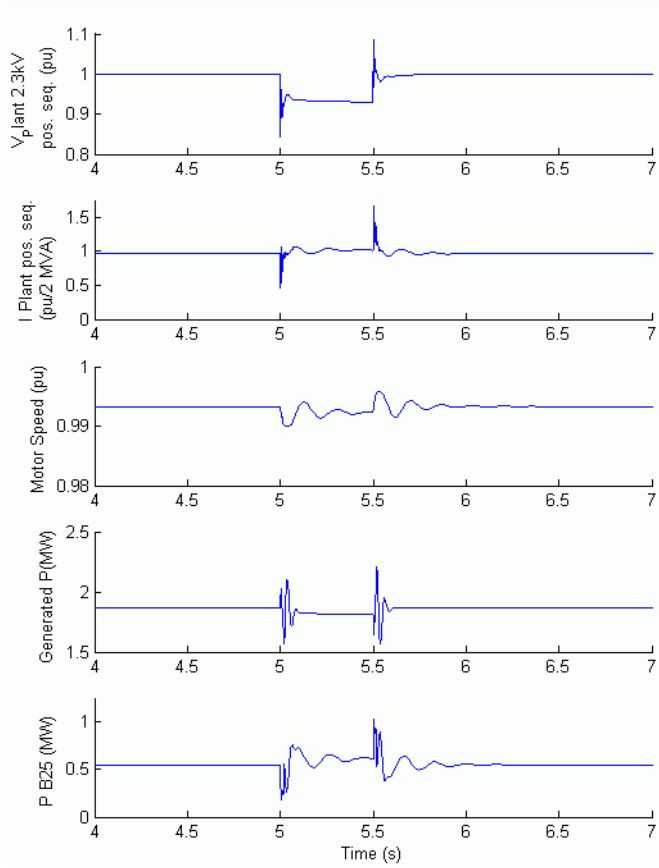
Simulation of a Voltage Sag on the 120 kV System

Now observe the impact of a voltage sag resulting from a remote fault on the 120 kV system. In this simulation the mode of operation is initially `Var regulation` with Qref=0 and the wind speed is constant at 8 m/s. A 0.15 pu voltage drop lasting 0.5 s is programmed, in the 120 kV voltage source menu, to occur at t=5 s. The simulation results are illustrated in Voltage Sag on the 120 kV System (Wind Farm in Var Regulation Mode) on page 6-27. 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 (exceeding protection settings for 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.



Voltage Sag on the 120 kV System (Wind Farm in Var Regulation Mode)

Now, the wind turbine control mode is changed to `Voltage regulation` and the simulation is repeated. 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 (Voltage Sag on the 120 kV System (Wind Farm in Voltage Regulation Mode) on page 6-28).

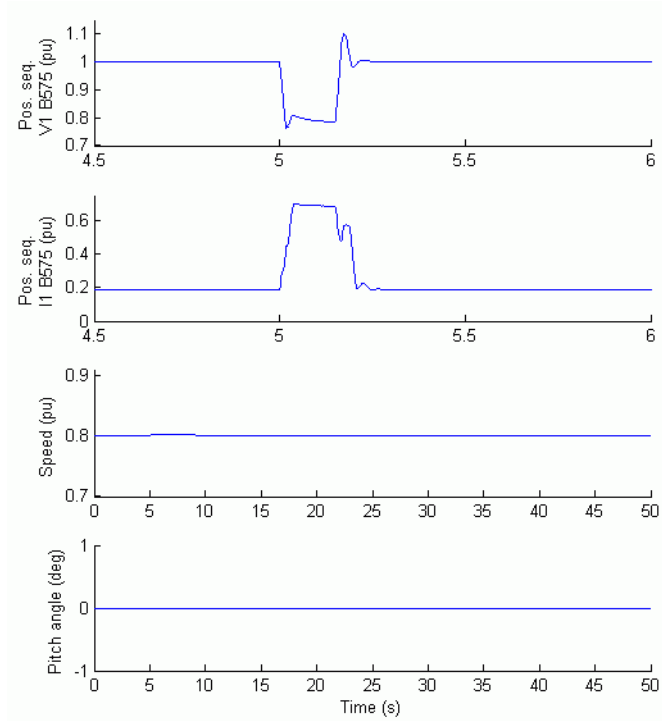


Voltage Sag on the 120 kV System (Wind Farm in Voltage Regulation Mode)

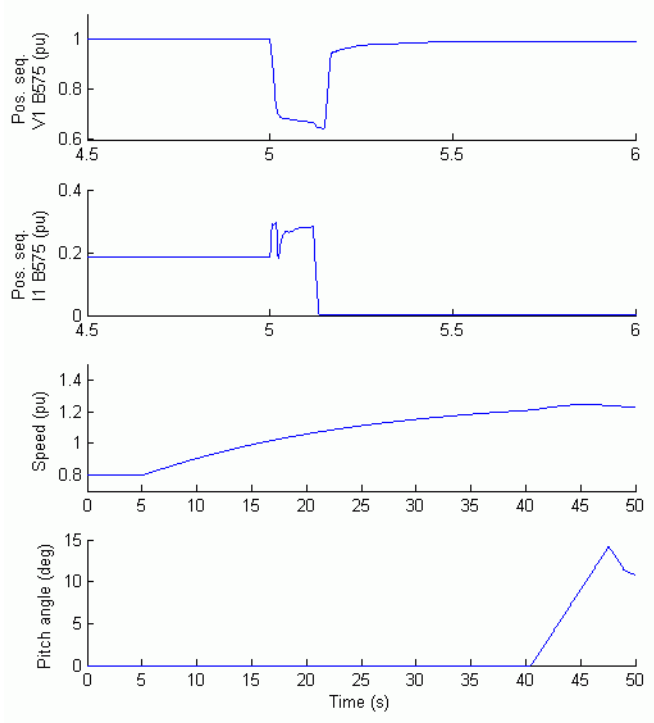
Simulation of a Fault on the 25 kV System

Finally, now observe the impact of a single phase-to-ground fault occurring on the 25 kV line. At t=5 s a 9 cycle (0.15 s) phase-to-ground fault is applied on phase A at B25 bus. 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 (Wind Farm Waveforms During Fault at Bus B25 (Wind Farm in Voltage Regulation Mode) on page 6-29). 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 to limit the speed (Wind Farm Waveforms During Fault at Bus B25 (Wind Farm in Var Regulation Mode) on page 6-30).



Wind Farm Waveforms During Fault at Bus B25 (Wind Farm in Voltage Regulation Mode)



Wind Farm Waveforms During Fault at Bus B25 (Wind Farm in Var Regulation Mode)

Index

A

AC motor drives 4-39
    See also electric drive 4-39
AC transmission network 6-3
adjustable speed drive (ASD) 4-5

B

bidirectional power flow capability 4-39
block diagrams
    creating 1-7
blocks
    nonlinear 3-16
    powerlib block library 1-9
build and simulate a system with
    AC motor drive 4-50
    DC motor drive 4-15
    mechanical shaft block 4-70
    speed reducer block 4-71

C

case studies
    mechanical Coupling of Two Motor Drives 4-72
    robot manipulator 4-87
    winding machine 4-79
circuit
    building a simple 1-7
connecting blocks 1-13
control
    close loop with slip speed compensation 4-46
    direct torque (DTC) 4-49
    flux oriented 4-46
    hysteresis modulation 4-40
    open loop Volts/Hertz 4-45
    pulse width modulation (PWM) 4-40
    schematic in user interface 4-30
    set point 4-23
    slip compensation 4-46
    space vector modulation technique 4-42

D

direct torque control (DTC) 4-49
display signals 1-11
distributed parameter line
    propagation speed 1-24
dynamic braking 4-39

E

electric drive
    applications 4-5
    components 4-5
    controller 4-7
    converter 4-6
    definition 4-5
    motor 4-6
    topology 4-5

F

FACTS (flexible AC transmission systems) 5-1
fixed-step simulation environment 4-59

H

HTG (hydraulic turbine and governor) 6-4
HVDC (high voltage direct current) 5-40
hysteresis modulation 4-40

I

interconnections
    between electric and Simulink blocks 1-7
interface
    between Simulink and SimPowerSystems 1-13

L

library
opening 4-17
linear and nonlinear elements 1-7
lines
connection lines 1-14
signal lines 1-14
load
bidirectional load at the motor shaft 4-72

M

mechanical Coupling of Two Motor Drives
speed regulated motor 4-75
system description 4-72
torque regulated motor 4-76
mechanical load
connecting to a DC model 4-20
connecting to an AC model 4-54
models
limitations with nonlinear 3-16
nonlinear model library 3-16
multiquadrant operation 4-8

P

parameters
default regulation 4-31
how to change a value 4-32
initially loaded 4-27
load procedure 4-27
low and high power sets 4-27
PCC (point of common coupling) 5-63
per unit system 1-11
PI section line
frequency response 1-24
PLL (phase locked loop) 5-69
ports
Simulink ports 1-14
terminal ports 1-14
power system 1-7
powerlib library 1-8
PSS (power system stabilizer) 6-3
PST (phase shifting transformer) 6-15
pulse width modulation (PWM) technique 4-40

Q

quadrants
operating region 4-8
representation 4-8

R

regenerative braking 4-15
robot
case study 4-87
position control systems 4-88
tracking Performance of drive 4-93

S

saturable transformer model 5-7
series-compensated transmission network 5-3
set point input to a drive 4-56
simulation
speed 3-20
sinusoidal source 1-9
STATCOM (static synchronous
compensator) 5-30
state variable
names 1-29
SVC (static var compensator) 5-21
synchronous machine 4-98
with regulators 4-98
systems
six-degrees-of-freedom robot
manipulator 4-87
two motor drives mechanically coupled
together 4-73
winding machine 4-79

T

TCR (thyristor-controlled reactor) 5-21
TCS (thyristor-switched capacitor) 5-21
transformers
    three-phase 5-7
transmission lines
    propagation time 1-24

U

UPFC (unified power flow controller) 6-10
user interface
    display of parameters 4-27
    drive library 4-27
    load button 4-28
    schematic button 4-30
    tabs 4-28

V

VDCOL (voltage dependent current order limiter) 5-47
visualizing internal signals of
    AC drive block 4-57
    DC drive block 4-23
voltage source
    connection to a DC drive block 4-18
    connection to an AC drive block 4-53
VSC (voltage sourced converter) 5-63
VSI inverters in drive 4-40

W

winders
    control 4-83
    description 4-79
    simulation results 4-84