

Power System Blockset

For Use with Simulink®

Hydro-Québec

TEQSIM International

Modeling
|

Simulation
|

Implementation
|



User's Guide

Version 2

How to Contact The MathWorks:



508-647-7000

Phone



508-647-7001

Fax



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

Mail



<http://www.mathworks.com>
<ftp.mathworks.com>
<comp.soft-sys.matlab>

Web
Anonymous FTP server
Newsgroup



support@mathworks.com
suggest@mathworks.com
bugs@mathworks.com
doc@mathworks.com
subscribe@mathworks.com
service@mathworks.com
info@mathworks.com

Technical support
Product enhancement suggestions
Bug reports
Documentation error reports
Subscribing user registration
Order status, license renewals, passcodes
Sales, pricing, and general information

Power System Blockset User's Guide

© COPYRIGHT 1998 - 2000 by TEQSIM International Inc., a sublicense of Hydro-Quebec, and 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 or for the federal government of the United States. By accepting delivery of the Program, the government hereby agrees that this software qualifies as "commercial" computer software within the meaning of FAR Part 12.212, DFARS Part 227.7202-1, DFARS Part 227.7202-3, DFARS Part 252.227-7013, and DFARS Part 252.227-7014. The terms and conditions of The MathWorks, Inc. Software License Agreement shall pertain to the government's use and disclosure of the Program and Documentation, and shall supersede any conflicting contractual terms or conditions. If this license fails to meet the government's minimum needs or is inconsistent in any respect with federal procurement law, the government agrees to return the Program and Documentation, unused, to MathWorks.

MATLAB, Simulink, Stateflow, Handle Graphics, and Real-Time Workshop are registered trademarks, and Target Language Compiler is a trademark of The MathWorks, Inc.

Other product or brand names are trademarks or registered trademarks of their respective holders.

Printing History:	January 1998	First printing
	January 1999	Revised for Version 1.1 (Release 11) (Online only)
	February 2000	Revised for Version 2 (Release 11.1) (Online only)
	September 2000	Revised for Version 2.1 (Release 12)

Getting Started

Preface

What Is the Power System Blockset	xii
Using This Guide	xiii
Units	xiv
About the Authors	xx
Related Products	xxi

Tutorial

1

Introduction	1-2
Session 1: Simulating a Simple Circuit	1-3
Building the Electrical Circuit with Powerlib Library	1-3
Interfacing the Electrical Circuit with Simulink	1-7
Simulating Your Circuit	1-8
Session 2: Analyzing a Simple Circuit	1-9
Steady-State Analysis	1-9
Frequency Analysis	1-12
Session 3: Simulating Transients	1-19
Continuous Variable Time-Step Integration Algorithms	1-20

Discretizing the Electrical System	1-22
Session 4: Introducing Power Electronics	1-25
Session 5: Simulating Motor Drives	1-32
Building and Simulating the PWM Motor Drive	1-33
Using the Multimeter Block	1-39
Discretizing the PWM Motor Drive	1-41
Session 6: Three-Phase Systems and Machines	1-42
Three-Phase Network with Electrical Machines	1-42
Load Flow and Machine Initialization	1-45
Session 7: Building and Customizing Your Own Nonlinear Models	1-52
Modeling a Nonlinear Inductance	1-52
Customizing Your Nonlinear Model	1-57
Modeling a Nonlinear Resistance	1-63
Creating Your Own Library	1-68
Connecting Your Model with Other Nonlinear Blocks	1-68

Using the Power System Blockset

Case Studies

2

Series Compensated Transmission Network	2-3
Description of the Transmission Network	2-3
Setting the Initial Load Flow and Obtaining Steady-State	2-9
Transient Performance for a Line Fault	2-10
Frequency Analysis	2-14
Transient Performance for a Fault at Bus B2	2-16
Chopper-Fed DC Motor Drive	2-21
Description of the Drive System	2-21
Modeling the DC Drive	2-24

Simulation of the DC Drive	2-27
Drive Starting	2-28
Steady-State Voltage and Current Waveforms	2-29
Speed Regulation Dynamic Performance	2-30
Synchronous Machine and Regulators	2-32
Introduction	2-32
Mathematical Model	2-32
Feedback Linearization Design	2-35
Simulation Results	2-37
Variable-Frequency Induction Motor Drive	2-40
Description of the Induction Motor Drive	2-40
A Field-Oriented Variable-Speed Induction Motor Drive	2-42
Modeling the Induction Motor Drive	2-45
Simulating the Induction Motor Drive	2-48
Drive Starting	2-49
Steady-State Voltage and Current Waveforms	2-50
Speed Regulation Dynamic Performance	2-50
HVDC System	2-52
Description of the HVDC Transmission System	2-52
Frequency Response of the AC and DC Systems	2-54
Description of the Control System	2-56
System Start-Up and Steady-State	2-61
Response to a Step of Reference Current	2-65
DC line Fault	2-66
AC Line-to-Ground Fault at the Rectifier	2-68

3

Advanced Topics

How the Power System Blockset Works	3-2
Which Integration Method Should Be Used - Continuous or Discrete?	3-5
Simulating with Continuous Integration Algorithms	3-6

Choosing the Right Integration Algorithm	3-6
Simulation of Switches and Power Electronic Devices	3-6
Simulating Discretized Electrical Systems	3-10
How to Increase Simulation Speed	3-13
The Nonlinear Model Library	3-14
The Continuous Library	3-15
The Discrete Library	3-16
The Switch Current Source Library	3-16
Limitations with the Nonlinear Models	3-16
Modifying the Nonlinear Models of the Powerlib_models Library	3-16
Creating Your Own Library of Models	3-18
Changing Your Circuit Parameters	3-19
Example of MATLAB Script Performing a Parametric Study	3-19

Reference

Block Reference

4

Block Reference Page Description	4-2
The Power System Block Libraries	4-3
AC Current Source	4-9
AC Voltage Source	4-11
Asynchronous Machine	4-13
Breaker	4-25
Bus Bar	4-30
Controlled Current Source	4-31
Controlled Voltage Source	4-34
Current Measurement	4-37

DC Machine	4-38
DC Voltage Source	4-43
Diode	4-45
Discrete System	4-50
Distributed Parameter Line	4-51
Excitation System	4-58
Ground	4-61
GTO	4-62
Hydraulic Turbine and Governor	4-69
Ideal Switch	4-75
IGBT	4-80
Impedance Measurement	4-87
Linear Transformer	4-89
MOSFET	4-94
Multimeter	4-100
Mutual Inductance	4-104
Neutral	4-108
Parallel RLC Branch	4-110
Parallel RLC Load	4-114
Permanent Magnet Synchronous Machine	4-116
PI Section Line	4-122
Powergui	4-127
Saturable Transformer	4-137
Series RLC Branch	4-144
Series RLC Load	4-148
Simplified Synchronous Machine	4-151
Steam Turbine and Governor	4-157
Surge Arrester	4-166
Synchronous Machine	4-171
Three-Phase Transformer (Two Windings)	4-185
Three-Phase Transformer (Three Windings)	4-191
Thyristor	4-197
Universal Bridge	4-204
Voltage Measurement	4-213

Introduction	5-2
circ2ss	5-3
power2sys	5-21
powerinit	5-27

Preface

What Is the Power System Blockset	x
Using This Guide	xi
Units	xii
About the Authors	xviii
Related Products	xix

What Is the Power System Blockset

Electrical power systems are combinations of electrical circuits, and electro-mechanical devices, like motors and generators. Engineers working in this discipline are constantly asked to improve 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, 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.

The Power System Blockset was designed to provide a modern design tool that will allow scientists and engineers to rapidly and easily build models that simulate power systems. The blockset uses the Simulink[®] environment, allowing a model to be built using simple *click and drag* procedures. Not only can the circuit topology be drawn rapidly, but the analysis of the circuit can include its interactions with mechanical, thermal, control, and other disciplines. This is possible because all of the electrical parts of the simulation interact with Simulink's extensive modeling library. Since Simulink uses MATLAB[®] as the computational engine, MATLAB's toolboxes can also be used by the designer.

Users will find that the blockset can be rapidly put to work. The 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-Quebec, a large North American utility located in Canada. The capabilities of the blockset for modeling a typical electrical grid are illustrated in demonstration files. And for the users who want to refresh their knowledge of power system theory, there are also self-learning case studies.

Using This Guide

If you are a new user, begin with Chapter 1 and 2 to learn:

- How to build and simulate electrical circuits using the **powerlib** library
- How to interfacing an electrical circuit with Simulink blocks
- How to analyze the steady-state and frequency response of an electrical circuit.
- How to build your own nonlinear models

If you are an experienced blockset user, see:

- The *Release Notes* for details on the latest release
- Chapter 1, “Tutorial” to learn how to simulate discretized electrical circuits
- Chapter 2, “Case Studies” for an overview of some applications of the Power System Blockset and the revised case studies
- Chapter 3, “Advanced Topics” to learn how to increase simulation speed

All blockset users should use Chapter 4, “Block Reference” for reference information on blocks, simple demos, and GUI-based tools. For functions, see Chapter 5, “Power System Commands” for a synopsis of the function’s syntax, as well as complete explanation of options and operation.

Units

In this manual, we use the International System of Units (SI).

Quantity	Unit	Symbol
Time	second	s
Length	meter	m
Mass	kilogram	kg
Energy	joule	J
Current	ampere	A
Voltage	volt	V
Active power	watt	W
Apparent power	volt ampere	VA
Reactive power	var	var
Impedance	ohm	Ω
Resistance	ohm	Ω
Inductance	henry	H
Capacitance	farad	F
Flux linkage	volt second	V. s
Rotation speed	radians per second revolutions per minute	rad/s rpm
Torque	newton meter	N.m
Inertia	kilogram (meter) ²	kg.m ²
Friction factor	newton meter second	N.m.s

We also use the per unit (p.u.) system on occasion to define the model parameters.

What Is the Per Unit System?

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

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

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

Generally the following two base values are chosen:

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

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

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

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

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

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

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

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

Instead of specifying the rotor inertia in $\text{kg}\cdot\text{m}^2$, you would generally give the inertia constant H defined as:

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

$$H = \frac{\frac{1}{2} \times J \cdot \omega^2}{P_{nom}}$$

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

Example 1: Three-Phase Transformer

Let us consider, for example, a three-phase two winding transformer, The following typical parameters could be provided by the manufacturer:

- Nominal power = 300 kVA total for three phases
- Nominal frequency = 60 Hz
- Winding 1: connected in wye, nominal voltage = 25 kV rms line-to-line resistance 0.01 p.u., leakage reactance = 0.02 p.u.
- Winding 2: connected in delta, nominal voltage = 600 V rms line-to-line resistance 0.01 p.u., leakage reactance = 0.02 p.u.
- Magnetizing losses at nominal voltage in % of nominal current:

Resistive 1%, Inductive 1%

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

- For winding 1:
 - base power : $300 \text{ kVA}/3 = 100\text{e}3 \text{ VA/phase}$
 - base voltage: $25 \text{ kV}/\sqrt{3} = 14434 \text{ V rms}$
 - base current : $100\text{e}3/14434 = 6.928 \text{ A rms}$
 - base impedance : $14434/6.928 = 2083 \Omega$
 - base resistance : $14434/6.928 = 2083 \Omega$
 - base inductance : $2083/(2\pi \cdot 60) = 5.525 \text{ H}$

- For winding 2:
 base power : $300 \text{ kVA}/3 = 100\text{e}3 \text{ VA}$
 base voltage: 600 V rms
 base current : $100\text{e}3/600 = 166.7 \text{ A rms}$
 base impedance : $600/166.7 = 3.60 \Omega$
 base resistance : $600/166.7 = 3.60 \Omega$
 base inductance : $3.60/(2\pi*60) = 0.009549 \text{ H}$

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

- For winding 1: $R1 = 0.01 * 2083 = 20.83 \Omega$; $L1 = 0.02 * 5.525 = 0.1105 \text{ H}$
- For winding 2: $R2 = 0.01 * 3.60 = 0.0360 \Omega$; $L2 = 0.02 * 0.009549 = 0.191 \text{ mH}$

For the magnetizing branch, magnetizing losses of 1% resistive and 1% inductive mean a magnetizing resistance R_m of 100 p.u. and a magnetizing inductance L_m of 100 p.u. Therefore, the values expressed in SI units referred to winding 1 are:

$$R_m = 100 * 2083 = 208.3 \text{ k}\Omega$$

$$L_m = 100 * 5.525 = 552.5 \text{ H}$$

Example 2: Asynchronous Machine

Let us now consider the three-phase four-pole Asynchronous Machine in SI units provided in the Machines library of **powerlib**. It is rated 3 HP, 220 V rms line-to-line, 60 Hz.

The stator and rotor resistance and inductance referred to stator are:

- $R_s = 0.435 \Omega$; $L_s = 2 \text{ mH}$
- $R_r = 0.816 \Omega$; $L_r = 2 \text{ mH}$

The mutual inductance is $L_m = 69.31 \text{ mH}$. The rotor inertia is: $J = 0.089 \text{ kg.m}^2$.

The base quantities for one phase are calculated as follows:

$$\begin{aligned} \text{base power} &: 3 \text{ HP} * 746 \text{ VA}/3 = 746 \text{ VA/phase} \\ \text{base voltage} &: 220 \text{ V}/\sqrt{3} = 127.0 \text{ V rms} \\ \text{base current} &: 746/127.0 = 5.874 \text{ A rms} \end{aligned}$$

base impedance : $127.0/5.874 = 21.62 \Omega$
 base resistance : $127.0/5.874 = 21.62 \Omega$
 base inductance : $21.62/(2\pi \cdot 60) = 0.05735 \text{ H} = 57.35 \text{ mH}$
 base speed : $1800 \text{ rpm} = 1800 \cdot (2\pi)/60 = 188.5 \text{ radians/second}$
 base torque (3phase): $746 \cdot 3/188.5 = 11.87 \text{ newton.meters}$

Using the above base values, you can compute the values in per units:

$R_s = 0.435 / 21.62 = 0.0201 \text{ p.u.}$ $L_s = 2 / 57.35 = 0.0349 \text{ p.u.}$
 $R_r = 0.816 / 21.62 = 0.0377 \text{ p.u.}$ $L_r = 2 / 57.35 = 0.0349 \text{ p.u.}$
 $L_m = 69.31/57.35 = 1.208 \text{ p.u.}$

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

$$H = \frac{\frac{1}{2} \times J \cdot \omega^2}{P_{nom}} = \frac{\frac{1}{2} \times 0.089 \times 188.5^2}{3 \times 746} = 0.7065 \text{ seconds}$$

If you open the dialog box of the Asynchronous Machine block in p.u. units provided in the Machines library of **powerlib**, you will find that the parameters in p.u. are the ones calculated above.

Base Values for Instantaneous Voltage and Current Waveforms

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

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

Here are main reasons for justifying the use of the per unit system:

- 1 When values are expressed in p.u., the comparison of electrical quantities with their “normal” value is straightforward.

For example, a transient voltage reaching a maximum of 1.42 p.u. indicates immediately that this voltage exceeds the nominal value by 42%.

- 2 The values of impedances expressed in p.u. stay fairly constant whatever the power and voltage ratings.

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

Similarly, for a salient pole synchronous machines, the synchronous reactance X_d is generally between 0.60 and 1.50 p.u whereas the sub transient reactance X'_d is generally between 0.20 and 0.50 p.u.

It means that if you don't know the parameters for a 10 kVA transformer, you won't make a big mistake by assuming an average value of 0.02 p.u. for leakage reactances and 0.0075 p.u for winding resistances.

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

About the Authors

The Power System Blockset version 2.0 was developed by the following people and organizations.

Gilbert Sybille

Hydro-Quebec Research Institute (IREQ) - Varennes, Québec. Technical coordinator, author of discretization techniques, revised power electronics, and documentation.

Patrice Brunelle

Teqsim International Inc. - St-Bruno, Québec. Author of graphical user interfaces, model integration into Simulink, and documentation.

Roger Champagne, Louis Dessaint

École de Technologie Supérieure (ETS) - Montréal, Québec. Authors of machine models, state space formulation, and documentation.

Hoang Lehuu

Université Laval - Québec City. Validation tests and author of documentation and some functions.

Pierre Mercier

Hydro-Quebec Research Institute (IREQ) - Varennes, Québec. Project manager for versions 1.0 and 2.0

Acknowledgements

The authors acknowledge the contributions of following people involved in versions 1 and 2.

Silvano Casoria, Momcilo Gavrilovic, Christian Larose, David McCallum, Kamal Al-Haddad, Christian Dufour, and Bahram Khodabakhchian.

Related Products

The MathWorks provides several products that are especially relevant to the kinds of tasks you can perform with the Power System Blockset. They are listed in the table below. In particular, the Power System Blockset *requires* these products:

- MATLAB®
- Simulink®

For more information about any of these products, see either:

- The online documentation for that product, if it is installed or if you are reading the documentation from the CD
- The MathWorks Web site, at <http://www.mathworks.com>; see the “products” section

Note The products listed below complement the functionality of the Power System Blockset.

Product	Description
Stateflow®	Tool for graphical modeling and simulation of complex control logic
DSP Blockset	Simulink block libraries for the design, simulation, and prototyping of digital signal processing systems
Nonlinear Control Design (NCD) Blockset	Simulink block libraries that provide a time-domain-based optimization approach to system design; automatically tunes parameters based on user-defined time-domain performance constraints

Product	Description
Fuzzy Logic Toolbox	Tool to help master fuzzy logic techniques and their application to practical control problems
-Analysis and Synthesis Toolbox	Tool for robust control design using optimal control and the structured singular value
Neural Network Toolbox	Comprehensive environment for neural network research, design, and simulation within MATLAB
Optimization Toolbox	Tool for general and large-scale optimization of nonlinear problems, as well as for linear programming, quadratic programming, nonlinear least squares, and solving nonlinear equations
Robust Control Toolbox	Tool for advanced robust multivariable feedback control
System Identification Toolbox	Tool for building accurate, simplified models of complex systems from noisy time-series data

Tutorial

Introduction	1-2
Session 1: Simulating a Simple Circuit	1-3
Building the Electrical Circuit with Powerlib Library	1-3
Interfacing the Electrical Circuit with Simulink	1-7
Simulating Your Circuit	1-8
Session 2: Analyzing a Simple Circuit	1-10
Steady-State Analysis	1-10
Frequency Analysis	1-13
Session 3: Simulating Transients	1-20
Continuous Variable Time-Step Integration Algorithms	1-21
Discretizing the Electrical System	1-23
Session 4: Introducing Power Electronics	1-26
Session 5: Simulating Motor Drives	1-33
Building and Simulating the PWM Motor Drive	1-34
Using the Multimeter Block	1-40
Discretizing the PWM Motor Drive	1-42
Session 6: Three-Phase Systems and Machines	1-43
Three-Phase Network with Electrical Machines	1-43
Load Flow and Machine Initialization	1-46
Session 7: Building and Customizing Your Own Nonlinear Models	1-53
Modeling a Nonlinear Inductance	1-53
Customizing Your Nonlinear Model	1-58
Modeling a Nonlinear Resistance	1-64
Creating Your Own Library	1-69
Connecting Your Model with Other Nonlinear Blocks	1-69

Introduction

To master the Power System Blockset, you must learn how to build and simulate electrical circuits. The Power System Blockset operates in the Simulink environment. Therefore, before starting this training you should be familiar with Simulink. For information on using Simulink, see the [Using Simulink](#) guide.

The tutorial is organized in seven different sessions. Sessions 1 through 3 are based on a simple power system. Sessions 4 and 5 illustrate power electronics, and Session 6 demonstrates three-phase power systems, electrical machinery, and load flow. Session 7 explains how you can create and customize your own nonlinear blocks.

Session 1: Simulating a Simple Circuit

The Power System Blockset (PSB) allows you to build and simulate electrical circuits containing linear and nonlinear elements. During Sessions 1 through 3 you will build, analyze, and simulate the circuit of Figure 1-1.

In this session you will:

- Explore the **powerlib** library of the Power System Blockset
- Learn how to build a simple circuit from the **powerlib** library
- Interconnect Simulink blocks with your circuit

The circuit of Figure 1-1 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. In order to simplify matters, only one of the three phases is represented. The parameters shown on the figure are typical of a 735 kV power system.

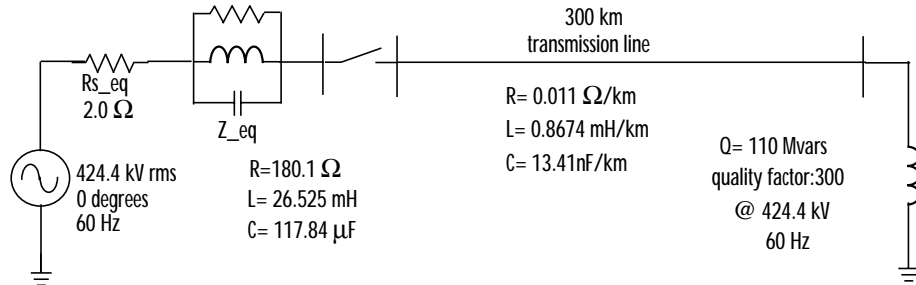


Figure 1-1: Circuit to be Modeled with the Power System Blockset

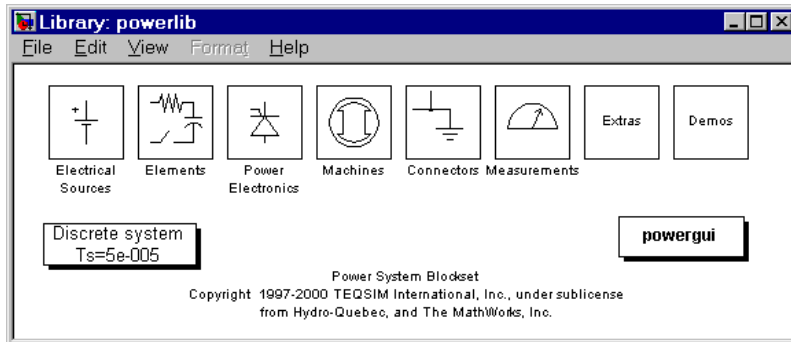
Building the Electrical Circuit with Powerlib Library

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

Open the Power System Blockset library by entering the following command at the MATLAB prompt:

```
powerlib
```

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



These libraries can be opened 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:

- 1 From the **File** menu of the **powerlib** window, open a new window that will contain your first circuit and save it as **circuit1**.
- 2 Open the **Electrical Sources** library and copy the **AC Voltage Source** block into the **circuit1** window.
- 3 Open the **AC Voltage Source** dialog box by double-clicking on the icon and enter the **Amplitude**, **Phase**, and **Frequency** parameters according to the values shown on Figure 1-1.

Note that the amplitude to be specified for a sinusoidal source is its peak value ($424.4e3 \cdot \sqrt{2}$ volts in our case).

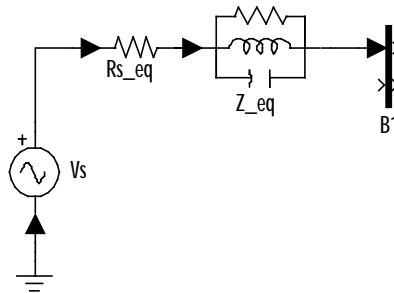
- 4 Change the name of this block from **Voltage Source** to **Vs**.
- 5 Copy the **Parallel RLC Branch** block, which can be found in the **Elements** library of **powerlib**, set its parameters as shown on Figure 1-1, and name it **Z_eq**.
- 6 The resistance **Rs_eq** of the circuit can be obtained from the **Parallel RLC Branch** block. Duplicate the **Parallel RLC Branch** block, which is already in

your **circuit1** window, set the R parameter according to Figure 1-1, and set the L and C parameters respectively to infinite (inf) and zero.

When the dialog box is closed, you will notice that the L and C components have disappeared so that the icon now shows a single resistor. The same result would have been obtained with the Series RLC Branch block by setting L and C respectively at zero and inf.

- 7 Name this block Rs_eq.
- 8 Open the Connectors library of **powerlib** and copy a bus bar.
- 9 Open the **Bus Bar** dialog box and set its parameters at two inputs and two outputs and name it B1. Also copy the Ground block (select the block with an output connection).

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



In order to complete the circuit of Figure 1-1, you need to add a transmission line, and a shunt reactor. You will add the circuit breaker later in Session 3.

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 **circuit1** window, set its parameters as shown on Figure 1-1, and specify one line section.

The shunt reactor will be modeled by a resistor in series with an inductor. You could use a Series RLC Branch block to model the shunt reactor. Set the R and L values corresponding to the active and reactive power specified on Figure 1-1 ($Q=110$ Mvar; $P=110/300=0.37$ MW at $V=424.4$ kV rms and $f=60$ Hz).

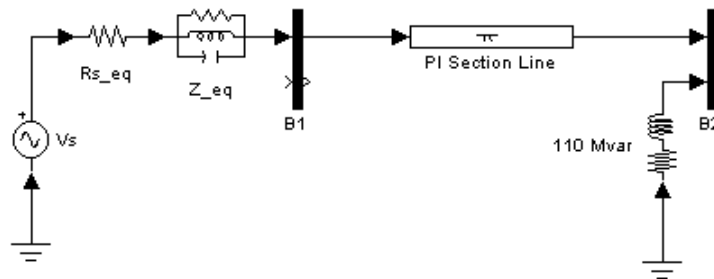
You may find it more convenient to use a Series RLC Load block that allows you to specify directly the active and reactive powers absorbed by the shunt reactor.

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:

$V_n=424.4\text{e}3\text{V}$; $f_n=60$ Hz; $P=110\text{e}6/300$ W (quality factor=300);
 $Q_L=110\text{e}6$ vars and $Q_C=0$.

Note that, as no reactive capacitive power has been specified, the capacitor disappears on the block icon when the dialog box is closed.

Add a receiving end bus bar B2 by duplicating B1 and interconnect all these new blocks as shown below.



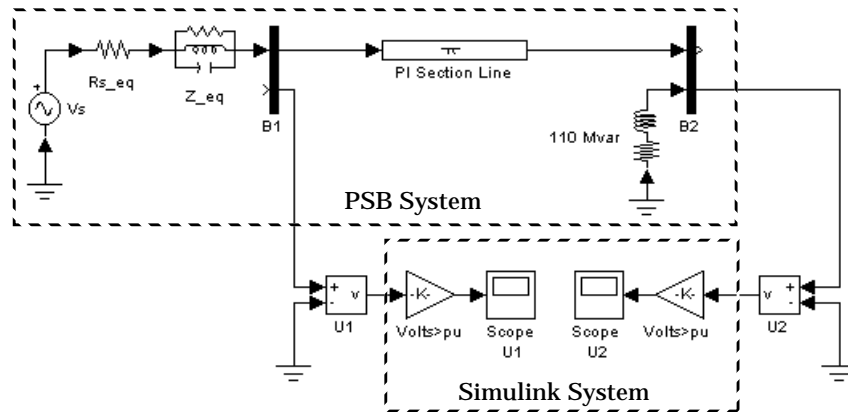
You need a Voltage Measurement block to measure the voltage at bus bar B1. This block can be found in the Measurements library of **powerlib**. Copy it and name it U1. Connect its positive input to the second output of bus bar B1 and its negative input to a new Ground block.

In order to observe the voltage at the bus bar B1 measured by the Voltage Measurement block named U1, a display system is needed. This could be any device found in the Sinks library of Simulink.

Open the Sinks library of Simulink and copy the Scope block in 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 our case the scaling factor K is:

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

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 bus bar B2, as shown below.

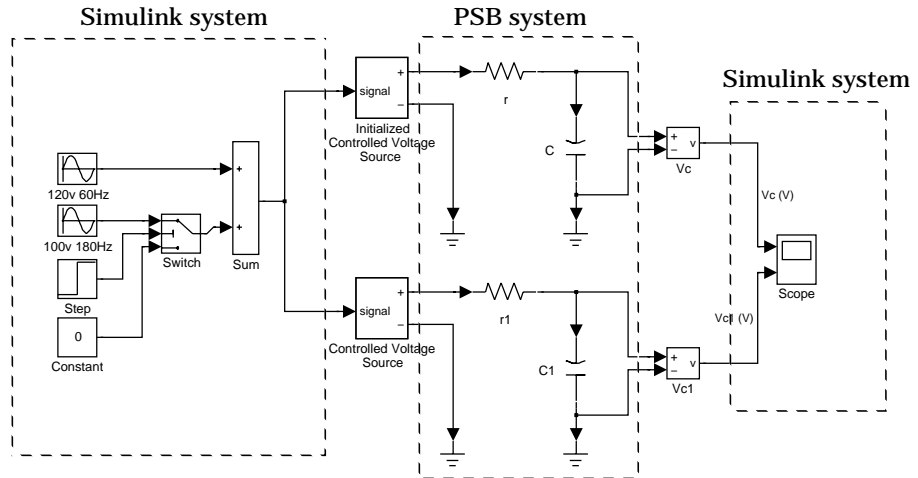


Interfacing the Electrical Circuit with Simulink

The Voltage Measurement block acts as an interface between the Power System Blockset blocks and the Simulink blocks. For the system shown above, the link is done from the electrical system to the Simulink system. The Voltage Measurement blocks convert the measured voltages into Simulink signals.

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

The link from Simulink blocks to the electrical system is also possible. For example you can use the Controlled Voltage Source block to inject a voltage in an electrical circuit. The voltage is then controlled by a Simulink signal.



Simulating Your Circuit

Now you can start the simulation from the **Simulation** menu. As expected, voltage is sinusoidal with peak value of 1 p.u.

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. Remember that you may zoom in on the waveforms in the scope windows by rubber banding (using the left mouse button) around the region of interest.

Note To simulate this circuit, the default integration algorithm (ode45) has been used. However, for most applications of the Power System Blockset your circuits will contain switches and other nonlinear models. In such a case, you will have to specify a different integration algorithm. This is discussed in Session 3 where a circuit breaker will be added to your circuit.

Session 2: Analyzing a Simple Circuit

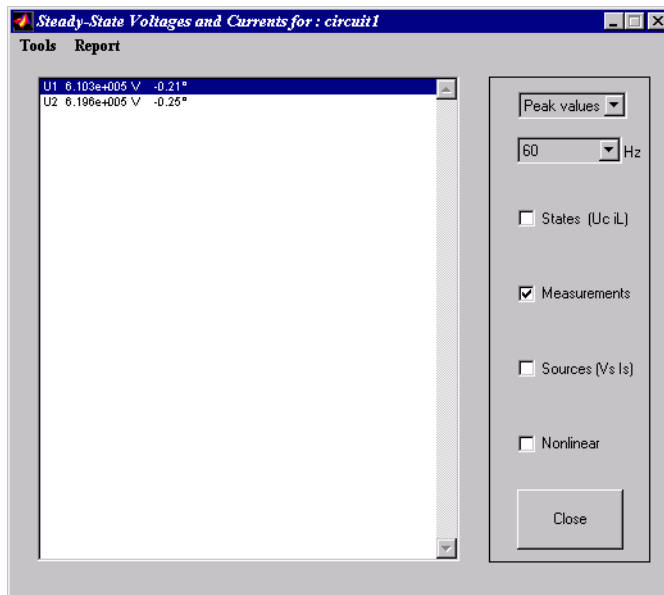
In this session you will:

- Use the Powergui (graphical user interface) block
- Obtain the steady-state outputs of the system
- Analyze your circuit with the `power2sys` function
- Analyze an electrical circuit in the frequency domain

Steady-State Analysis

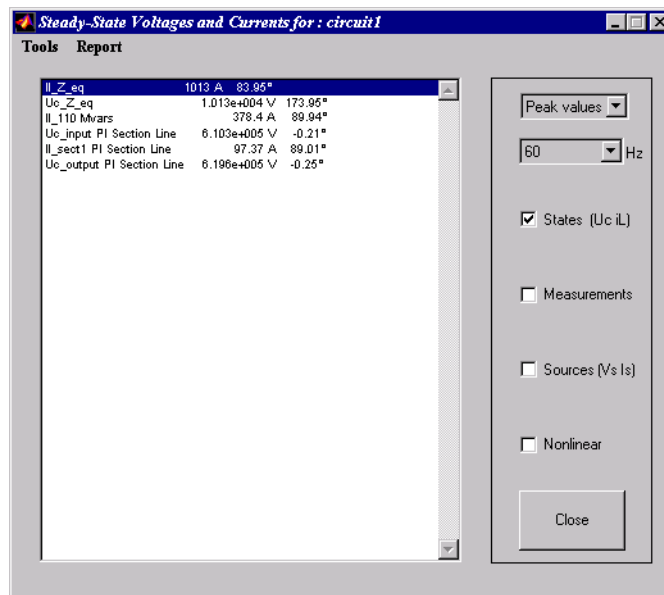
In order to facilitate the steady-state analysis of your circuit, a graphical user interface (GUI) is provided in the **powerlib** library. Copy the Powergui block into your **circuit1** window and double-click on the block icon to open it.

This opens the **Steady State** window where the steady-state phasors measured by the two measurement blocks are displayed in polar form.



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

From the **Steady State** window you can also choose to display the steady-state values of the source voltage or the steady-state values of states by checking either the **Sources** or the **States** check box.



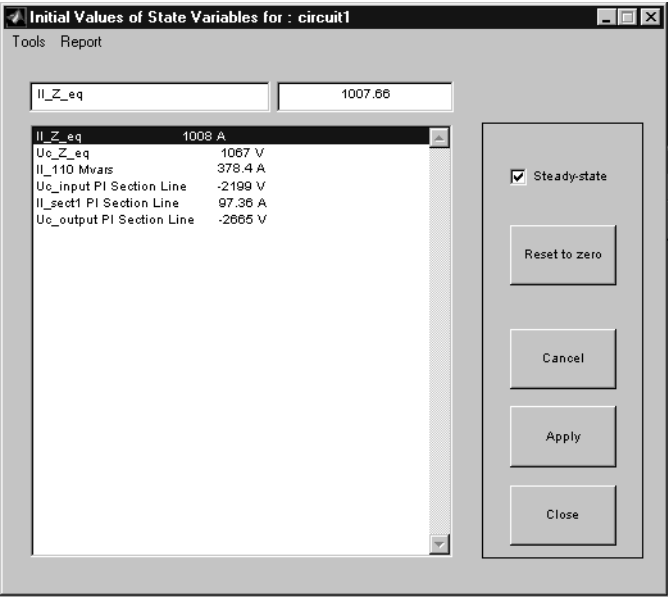
The state variable names contain the name of the block where the inductor or capacitor is found, preceded by the I1_ prefix for inductor currents or the Uc_ prefix for capacitor voltages.

The sign conventions used for the voltages and currents of sources and state variables are determined by the orientation of the blocks:

- Inductor currents flowing in the arrow direction are considered positive.
- Capacitor voltages are $V_{block\ output} - V_{block\ input}$

Note Depending on the exact position of the various blocks in your circuit diagram, the state variables may not be ordered the same way as in the figure above.

Now, from the **Tools** menu of the **powergui**, select **Initial Values of State Variables / Display or Modify Initial State Values**. The initial values of the six state variables (three inductor currents and three capacitor voltages) are displayed. These initial values are set in order to start the simulation in steady-state.



Frequency Analysis

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

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

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

To measure the impedance versus frequency at bus bar B2, you will need a current source at bus bar B2 providing a second input to the state-space model. Open the Electrical Sources library and copy the AC Current Source block in your model. Connect this source at bus bar B2 as shown on Figure 1-2. Set the current source magnitude to 0 and keep its frequency at 60 Hz. Rearrange the blocks as follows.

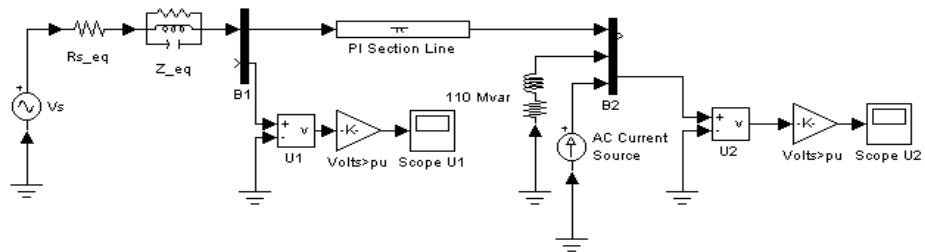


Figure 1-2: AC Current Source at the B2 Bus Bar

Now compute the state-space representation of the model circuit1 with the power2sys function. Enter the following command at the MATLAB prompt:

```
[A, B, C, D, x0, states, inputs, outputs]=power2sys('circuit1');
```

The power2sys function returns the state space model of your circuit in the four matrices A, B, C, and D. x0 is the vector of initial conditions that you have just

displayed with the Powergui block. The names of the state variables, inputs, and outputs are returned in three string matrices.

```
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
I_AC Current Source

outputs =

U_U1
U_U2
```

Note that you could have obtained the names and ordering of the states, inputs and outputs directly from the Powergui block.

Once the state-space model of the system is known, it can be analyzed in the frequency domain. For example, the modes of this circuit can be found from the eigenvalues of matrix A (use the MATLAB `ei g` command):

```
ei g(A)
ans =

1.0e+05 *
-2.4972
-0.0001 + 0.0144i <---229 Hz
-0.0001 - 0.0144i
-0.0002 + 0.0056i <---89 Hz
-0.0002 - 0.0056i
-0.0000
```

This system 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.

If you own the Control System Toolbox, you can compute the impedance of the network as a function of frequency by using the bode function.

In the Laplace domain, the impedance Z_2 at bus B2 is defined as the transfer function between the current injected at bus B2 (input 2 of the system) and the voltage measured at bus B2 (output 2 of the system).

$$Z_2(s) = \frac{U_2(s)}{I_2(s)}$$

The impedance at bus B2 for the 0-1500Hz range can be calculated and visualized as follows:

```
freq=0: 1500;  
w=2*pi *freq;  
[mag1, phase1]=bode(A, B, C, D, 2, w);  
semilogy(freq, mag1(:, 2));
```

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 with the one obtained with a single line section, enter the following commands:

```
[A, B, C, D]=power2sys('ci rcui t1');  
[mag10, phase10]=bode(A, B, C, D, 2, w);  
semilogy(freq, mag1(:, 2), freq, mag10(:, 2));
```

The resulting plot is shown in Figure 1-3.

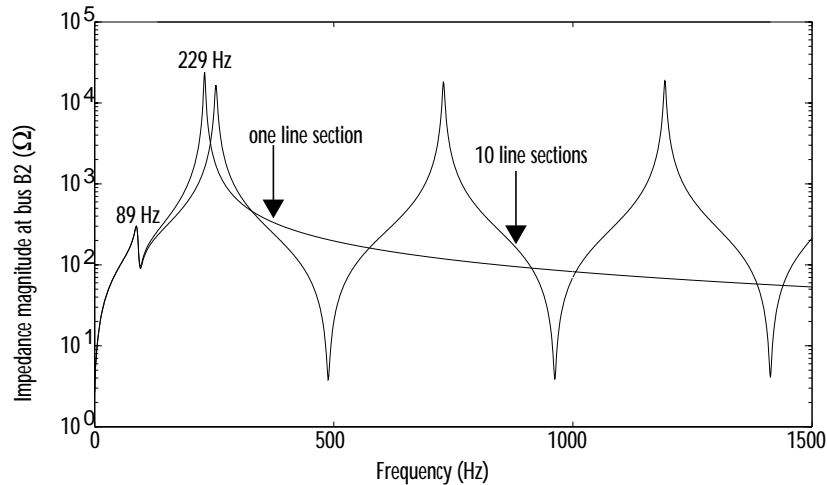


Figure 1-3: Impedance at Bus B2 as Function of Frequency

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.

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 $f_1 = 1/4T = 244$ Hz. A distributed parameter line would have an infinite number of modes every $244 + n \cdot 488$ Hz ($n = 1, 2, 3, \dots$). The 10 section line model simulates the first 10 modes. The first three line modes can be seen on Figure 1-3. (244 Hz, 732 Hz and 1220 Hz).

Obtaining the Impedance vs Frequency from the Impedance Measurement Block and the Powergui Block

The process we have described above to measure a circuit impedance has been automated in the Power System Blockset. Open the Measurements library of **powerlib** and copy the Impedance Measurement block in your model and rename it ZB2. The block uses a current source and a voltage measurement to

perform the impedance measurement. Connect the two inputs of this block between bus bar B2 and ground as shown on Figure 1-4.

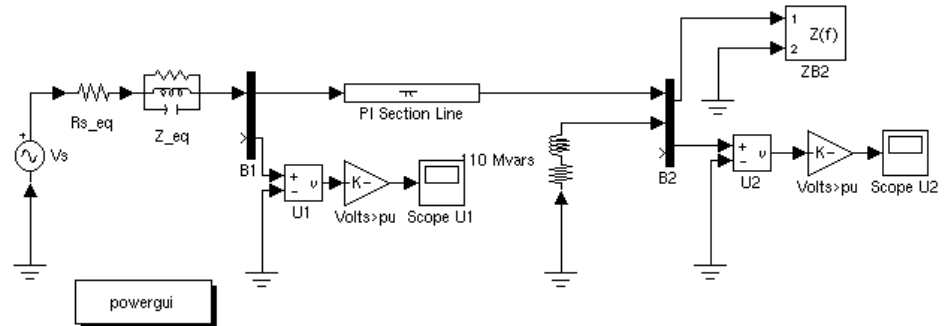
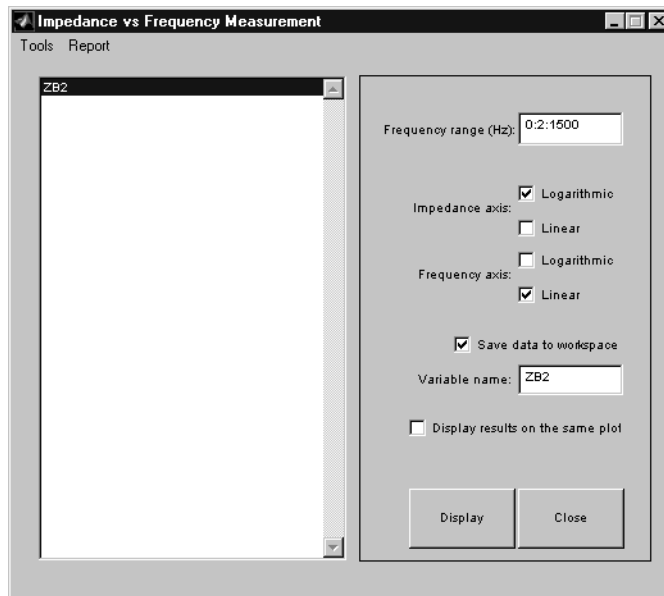


Figure 1-4: Measuring Impedance vs Frequency with the Impedance Measurement block

Now open the **powergui**. 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 typing 0: 2: 1500 (zero to 1500 Hz by steps of 2 Hz). Select the logarithmic scale to display Z magnitude. Check the **Save data to workspace** button and enter ZB2 as the variable name that will contain the impedance vs frequency. Click on the **Display** button.

When the calculation is finished, a graphic window appears with the magnitude and phase as function of frequency. The magnitude should be identical to the plot (for one line section) shown on Figure 1-3. If you look in your workspace, you should have a variable named ZB2. It is a two-column matrix containing frequency in column 1 and complex impedance in column 2.

Now open the **Simulation/Parameters** menu of your circuit1 model. In the **Solver** section, select the **ode23tb** integration algorithm. Set the relative tolerance to 1e-4 and keep auto for the other parameters. Set the stop time to 0.05. Open the scopes and start the simulation.

Look at the waveforms of the sending and receiving end voltages on ScopeU1 and ScopeU2. As the state variables have been automatically initialized, the system starts in steady state and sinusoidal waveforms are observed.

Finally open the **powergui**. In the **Tools** menu select **Initial Values of State Variables / Display or Modify Initial State Value** and reset all the states to 0 by selecting the **Reset to zero** button and then **Apply** button. Restart the simulation and observe the transient when the line is energized from zero.

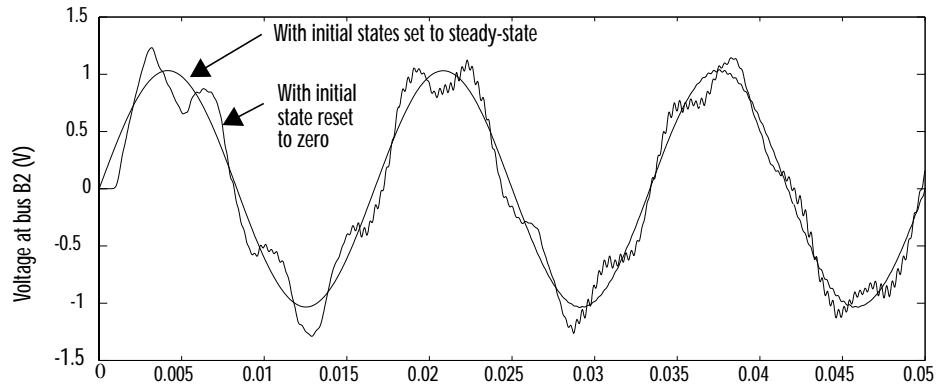


Figure 1-5: Receiving End Voltage U2 with 10 pi Section Line

Session 3: Simulating Transients

In this session you will:

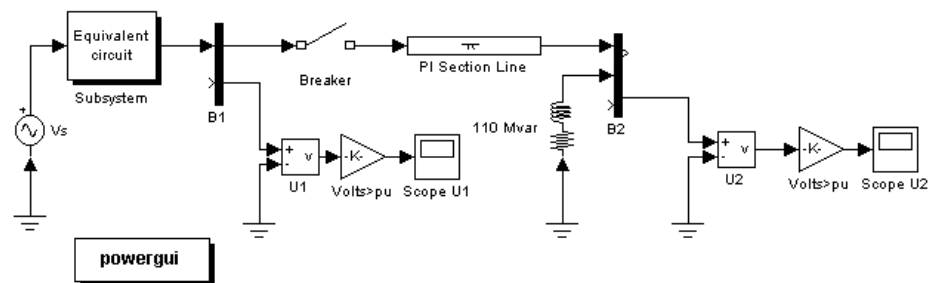
- Learn how to create an electrical subsystem
- Simulate transients with a circuit breaker
- Compare time domain simulation results with different line models
- Learn how to discretize a circuit and compare results obtained with continuous variable time-step algorithm and discrete system

One of the main uses of the Power System Blockset 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 `ci rcui t 1` system and delete the current source connected at bus B2. Save this new system as `ci rcui t 2`. Before connecting a circuit breaker, you need to modify the schematic diagram of `ci rcui t 2`. As with Simulink, the Power System Blockset allows you to 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 by:

- 1 Selecting the two blocks identified `Rs_eq` and `Z_eq` by surrounding them with a bounding box (left mouse button) and use the **Edit/Create Subsystem** menu. The two blocks now form a new block called Subsystem.



- 2 Using the **Edit/Mask Subsystem** menu, change the icon of that subsystem. In the **Icon** section of the mask editor, type the following drawing command:
`disp('Equivalent \nCircuit')`
- 3 Use the **Format/Show Drop shadow** menu to get the appearance shown on the figure. You can now double-click on the Subsystem block and look at its content.
- 4 Insert a circuit breaker in your circuit in order 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. Due to modeling constraints, this resistance cannot be set to 0. However, it can be set to a very small value, say $0.001\ \Omega$, 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 W; 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 on the previous figure.
- 3 Finally connect a Scope block from the Sinks library of Simulink, at the output of the Gain block measuring U2. Click on the Scope properties and select the **Data history** tab. Check 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, deselect the **Limit rows to last** button. This will allow you 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/Parameters** menu. As you now have a system

containing switches, you will need a stiff integration algorithm to simulate the circuit. In the Solver section select the variable step, stiff integration algorithm **ode23tb**.

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 variables U2 into U2_1 by entering the following command in the MATLAB 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 variables U2 into U2_10.

Before modifying your circuit to use a distributed parameter line model, save your system as `circuit2_10pi`. You will have to reuse this circuit 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 Figure 1-1). 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 on Figure 1-6. As expected from the frequency analysis performed during Session 2, 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 on the figure the propagation time delay of 1.03 ms associated with the distributed parameter line.

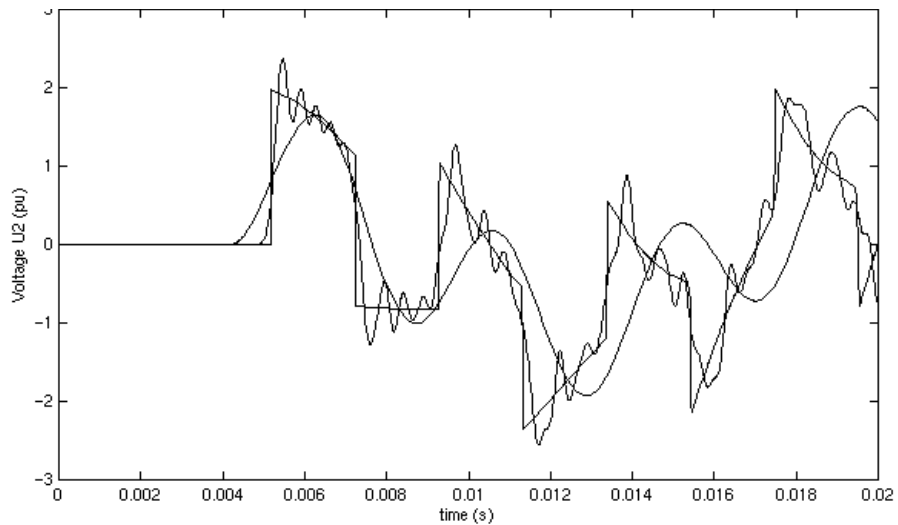


Figure 1-6: Receiving End Voltage Obtained with Three Different Line Models

Discretizing the Electrical System

One important feature of the Power System Blockset, which has been introduced with version 2.0, is its ability to simulate either with continuous variable time-step integration algorithms or with discrete solvers. For small systems, variable time steps algorithms are usually faster than fixed-time step methods because the number of integration steps is lower. However, for large systems that contain many states or many nonlinear blocks such as power electronic switches, it is advantageous to discretize the electrical system.

When you discretize your system, the precision of the simulation will be controlled by the time step. If you use too large time step, the precision may 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 $20\ \mu\text{s}$ to $50\ \mu\text{s}$ will 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 will have to 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-turn-off 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 least 1 μs .

You will now learn how to discretize your system and compare simulation results obtained with continuous and discrete systems. Open the `ci rcui t2_10pi` system that you saved from a previous simulation. This system contains 24 states and one switch. Copy the Discrete System block of the **powerlib** library in your `ci rcui t5` system. Open it and set the sample time to 25e-6 s. When you restart the simulation, the power system will be discretized using the tustin method (corresponding to trapezoidal integration) using a 25 μs sample time.

Open the **Simulation/Parameters/Solver** menu 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 don't need a variable step integration method to simulate. In the **Simulation/Parameters/Solver** menu, you could have selected the **Fixed-step /discrete (no continuous states)** option and specified a Fixed-step of 50 μs .

In order 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 window.

To return to the continuous simulation open the Discrete System block and set the Sample time to zero. If you compare the simulation times, you will find that the discrete system simulates approximately 3.5 times faster than the continuous system.

In order to compare the precision of the two methods, perform the following three simulations:

- 1 Simulate a continuous system, with $T_s=0$
- 2 Simulate a discrete system, with $T_s=25 \mu\text{s}$

3 Simulate a discrete system, with $T_s=50\text{ }\mu\text{s}$

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), U2d(50: , 2))
```

Using the zoom button of the graphic window, zoom in on the 4 - 12 ms region. You will see differences on the high frequency transients. The $25\text{ }\mu\text{s}$ compares reasonably well with the continuous simulation. However, increasing the time step to $50\text{ }\mu\text{s}$ produces appreciable errors. The $25\text{ }\mu\text{s}$ time step would therefore be acceptable for this circuit, while obtaining a gain of 3.5 on simulation speed.

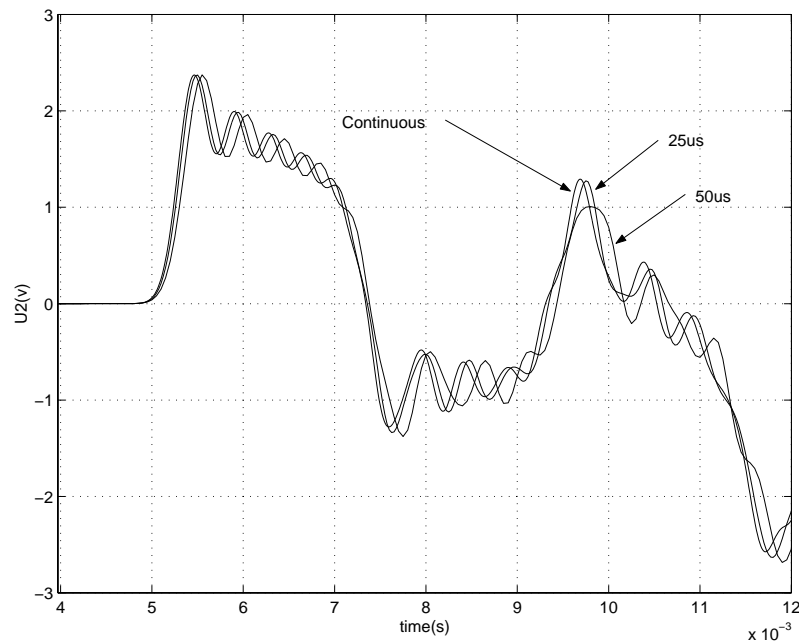


Figure 1-7: Comparison of Simulation Results for Continuous and Discrete Systems

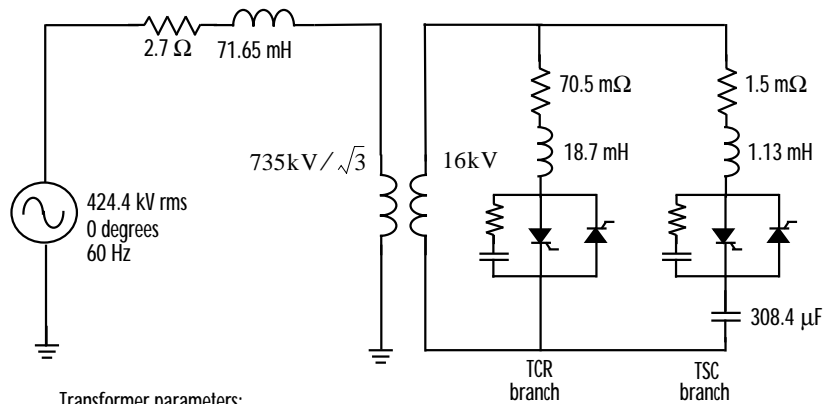
Session 4: Introducing Power Electronics

In this session you will:

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

The Power System Blockset has been designed to simulate power electronic devices. In this session, you will build a simple circuit using thyristors.

Consider the circuit given in Figure 1-8. It represents one phase of a static var compensator (SVC) used on a 735 kV transmission network. On the secondary of the 735kV /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 p.u.; resistance=0.002 p.u.

Secondary: Rated voltage 16 kV rms; leakage reactance=0 p.u.; resistance=0.002 p.u.

Magnetizing current at 1 p.u. voltage: Inductive: 0.2%
Resistive: 0.2%

Thyristors parameters:

$R_{on} = 1\text{ m}\Omega$; $V_f = 14 \times 0.8\text{ V}$ (14 thyristors in series)

Snubber: $R_s = 500\Omega$ $C_s = 0.15\mu\text{F}$

Figure 1-8: 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 in order to obtain a continuous variation of the net SVC reactive current.

You will 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 `ci rcui t3`.
- 2 Open the Power Electronics library and copy the Thyristor block into your `ci rcui t3` model.

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

($R_{on}=1e-3$; $L_{on}=0$; $V_f=14*0.8$; $R_s=500$; $C_s=0.15e-6$).

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 on Figure 1-9.

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 $R_s=Inf$; $C_s=0$.

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 on Figure 1-8.

Note that the leakage reactance and resistance of each winding have to be specified directly in per unit quantities. As there is no tertiary winding, type 0

in the field corresponding to winding 3. Notice that the winding 3 disappears on the TrA block.

Finally, set the magnetizing branch parameters R_m and X_m at [500, 500]. These values correspond to 0.2% resistive and inductive currents as specified in Figure 1-8.

Add a voltage source, series RL elements and a ground block. Set the parameters as shown in Figure 1-8. Add a current measurement to measure the primary current. By using appropriate connectors, you should be able to interconnect the circuit as shown in Figure 1-9.

Notice that the Thyristor blocks have an output identified by the letter *m*. This output returns a Simulink vectorized signal containing the thyristor current (I_{ak}) and voltage (V_{ak}). Connect a Demultiplexer block with two outputs at the *m* output of Th1. Then connect the two multiplexer 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, set the number of axes to 2). Identify the two connection lines *I th1* and *V th1*. These identifications will be automatically displayed on the top of each trace.

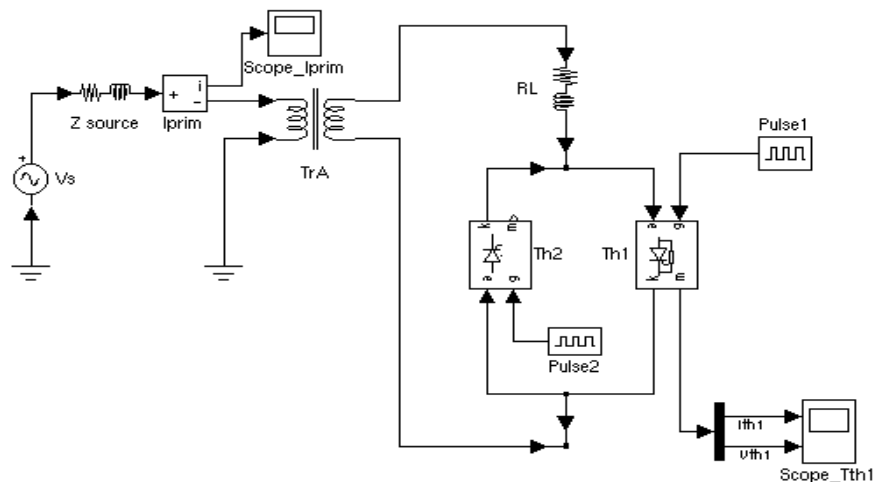


Figure 1-9: Simulation of the TCR Branch

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:

```
Period : 1/60 s
Duty cycle: 1% (3.6 degrees pulses)
Amplitude : 1
Start time : 1/60+T for Pulse1; 1/60+1/120+T for Pulse2
```

The pulses sent to Th1 are delayed by 180 degrees with respect to pulses sent to Th2. The delay T is used to specify the α firing angle. In order to get a 120 degrees firing angle, specify T in the workspace by typing:

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

Now open the **Simulation/Parameters** menu. Select the **ode23tb** integration algorithm. Keep the default parameters but set the relative tolerance to 1e-4 and the stop time to 0. 1. Start the simulation. The results are shown on Figure 1-10.

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

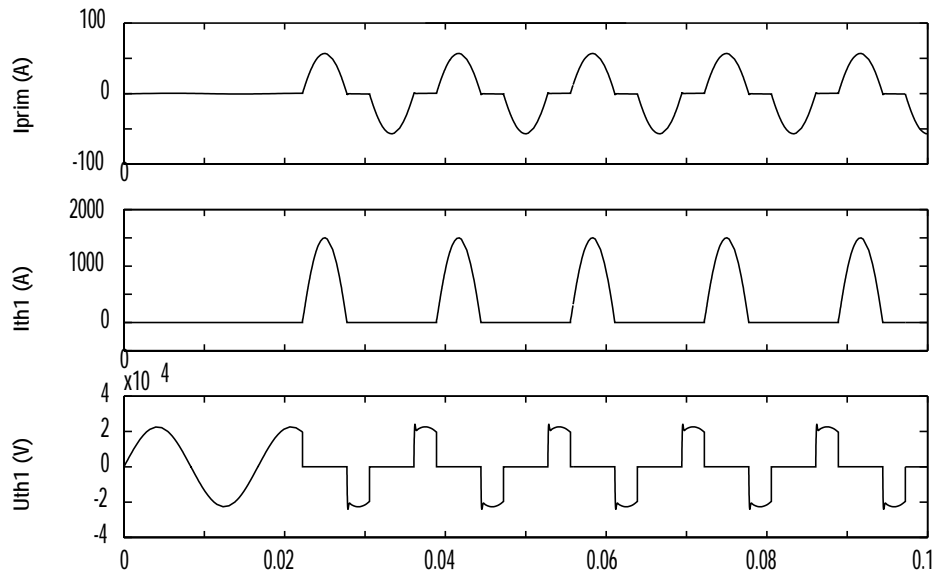


Figure 1-10: TCR Simulation Results

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 Figure 1-11. Change the R, L, and C parameters as shown in Figure 1-8. 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 will be 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 in Figure 1-11.

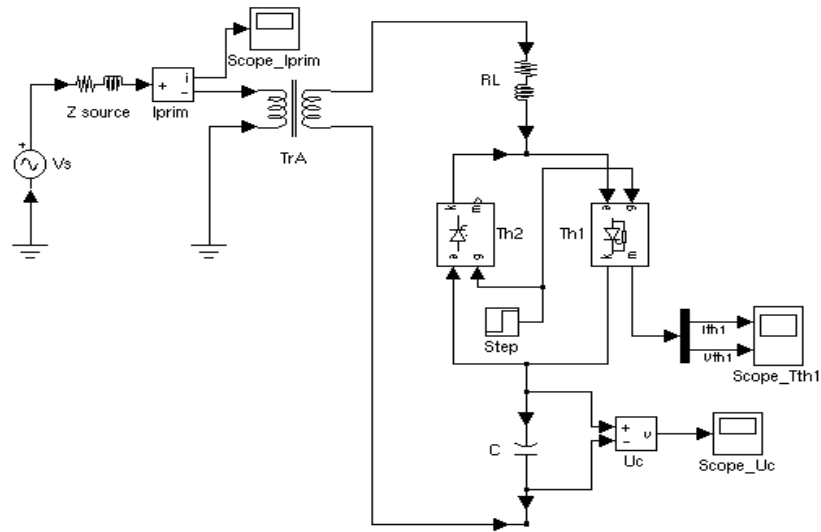


Figure 1-11: Simulation of the TSC Branch

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 will have an initial voltage left since the last valve opening. In order 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:

$$U_c = 17670 \times \sqrt{2} = 24989 \text{ Volts}$$

You can now use the Powergui block to change the capacitor initial voltage. Open the **powergui**. In the **Tools** menu, select **Initial Values of State Variables / Display or Set Initial Conditions**. A list of all the state variables with their default initial values appears. The value of the initial voltage across

the capacitor C (variable U_{c_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 U_{c_C} state variable and enter 24989 in the upper right field. Then click on the **Apply** button in order 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 on Figure 1-12.

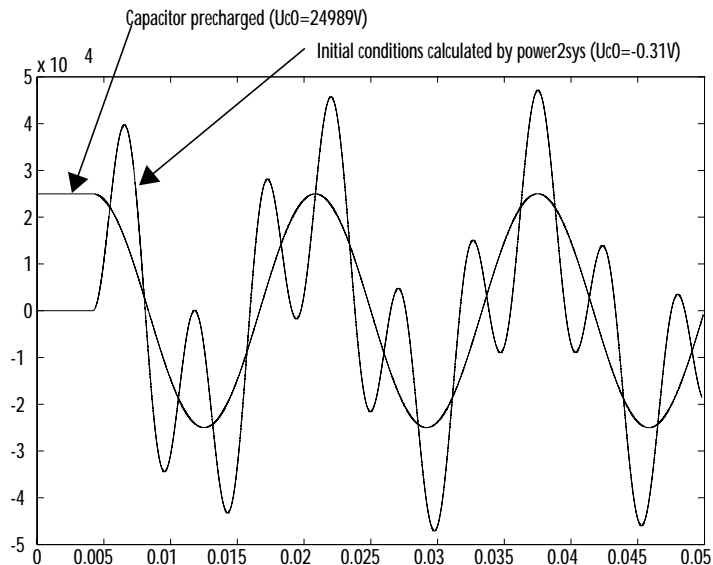


Figure 1-12: Transient Capacitor Voltage with and Without Initial Charge

Session 5: Simulating Motor Drives

In this session you will:

- Use electrical machines and power electronics to simulate a simple motor drive
- 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

Variable speed control of AC electrical machines make use of forced-commutated electronic switches such as IGBTs, MOSFETs and GTOs. Asynchronous machines fed by pulse width modulation (PWM) inverters are now gradually replacing the DC motors and thyristor bridges. With PWM associated with modern control techniques such as field oriented control or direct torque control, it is now possible to obtain the same flexibility in speed and torque control as with DC machines. In this session you will build a simple open loop DC drive controlling an asynchronous machine. A more elaborate example of PWM drive is presented in the Case Studies chapter. The Power System Blockset circuit to simulate is shown on Figure 1-13. It uses blocks of the Machines and Power Electronics libraries.

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 to simulate diodes, thyristors, GTO thyristors, MOSFETs, and IGBTs 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.

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

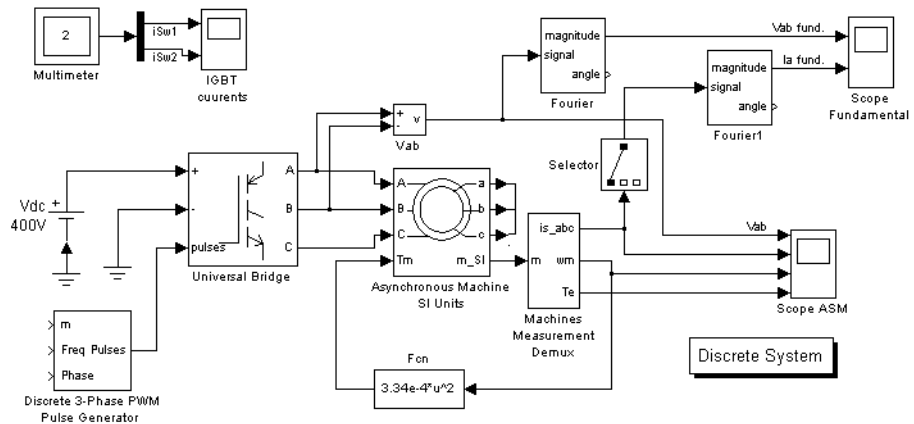


Figure 1-13: Circuit 5: PWM Control of an Induction Motor

Building and Simulating the PWM Motor Drive

- 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** menu and set its parameters as follows:

Power Electronic device =IGBT/Diodes; Port configuration= ABC as output terminals; Snubber $R_s=1e5$ W $C_s=inf$; $R_{on}=1e-3$ W; Tail: $T_f=1e-6$ s; $T_t=1e-6$ s).

Notice that the snubber circuit is integral to the **Universal Bridge** dialog box. As the C_s capacitor value of the snubber has been 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 the Power System Blockset is modeled as a current source, you have to provide a parallel path across each IGBT in order to allow connection to an

inductive circuit (stator of the asynchronous machine). The high resistance value of the snubber will not affect the circuit performance.

- 4 Open the Machines library. Copy the Asynchronous Machine SI Units block as well as the Machines Measurement Demux block into your circuit model.
- 5 Open the **Asynchronous Machine** menu and look at its parameters. The parameters are set for a 3 HP, 220 V, 60 Hz, two pairs of poles machine. Its nominal speed is therefore slightly lower than the synchronous speed of 1800 rpm or $\omega_s = 188.5$ 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. Open the Connectors library. Copy the vertical Bus Bar block with two inputs and one output into your circuit model.
- 7 Open the **Bus Bar** menu and change the number of inputs to three and the number of outputs to zero. Resize the block vertically and connect its three inputs to the three rotor terminals as shown on Figure 1-13.
- 8 Open the **Machines Measurement Demux** block menu. When this block is connected at the ASM measurement output, it allows you to access specific internal signals of the ASM. Deselect all signals except the following three signals: i_{s_abc} (three stator currents), ω_m (rotor speed) and T_e (electromagnetic torque).
- 9 You will now implement the torque-speed characteristic of the motor load. We will 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:

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

Therefore, the constant k should be:

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

Open the Functions & Tables library of Simulink and copy the Fcn block into your circuit5 model. Open the block menu and enter the expression of torque as function of speed:

$$3.34 \times 10^{-4} \cdot u^2$$

- 10 Connect the input of the Simulink Fcn block to the speed output of the Machines Measurement Demux block labeled ω_m and its output to the torque input of the motor labeled T_m .
- 11 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.
- 12 Open the Measurement library and copy a Voltage Measurement block into your circuit5 model. Change the block name to V_{ab} .
- 13 Using ground blocks of the Connectors library, complete the power elements and voltage sensor interconnections as shown in Figure 1-13.
- 14 In order to control your inverter bridge, you need a pulse generator. Such a generator is available in the Extras library of **powerlib**. Open the Extras/Control Blocks library and copy the Discrete 3-Phase PWM Pulse Generator block into your circuit5 model. Connect its Pulses output to the Pulses input of the Universal Bridge block.
- 15 Open the **Discrete 3-Phase PWM Pulse Generator** block menu and set the parameters as follows:
Modulation index $m = 0.90$; Frequency of output voltage = 60 Hz;
Phase of output voltage = 0 degrees; Switching frequency $F_s = 1080$ Hz; Time step = 10×10^{-6} s
- 16 Use the **Edit/Look Under Mask** menu of your **circuit5** window to see how the PWM is implemented. This control system is entirely made with Simulink blocks. The block has been discretized so that the pulses will

change at multiples of the specified time step. A time step of 10 μ s corresponds to +/- 0.54% of the switching period at 1080 Hz.

- 17 One common method of generating the PWM pulses uses comparison of the output voltage to synthesize (60 Hz in our case) with a triangular wave at the switching frequency (1080 Hz in our case). This is the method that is implemented in the Discrete 3-Phase PWM Pulse Generator. 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:

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

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.

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

Open the **Fourier** block menus and check that the parameters are set as follows:

Fundamental frequency f1= 60 Hz; Harmonic number= 1;

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

- 19 Duplicate the Fourier block. In order to measure the phase A current, you need to select the first element of the i_{s_abc} output of the ASM Measurement Demux.

Copy a Selector block from the Signal & Systems library of Simulink.

Open its menu and set Element to 1. Connect the Selector output to the second Fourier block and its input to the i_{s_abc} output of the Machines Measurement Demux block as shown on Figure 1-13.

- 20 Finally, add scopes to your model. Copy one Scope block in your circuit. This scope will be 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 on Figure 1-13. When you start the simulation, these labels will be displayed on top of each trace.

- 21 Duplicate the four- input Scope and change its number of inputs to 2. This scope will be 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 on Figure 1-13.

You are now ready to simulate the motor starting.

Simulating the PWM Motor Drive with Continuous Integration Algorithm

Open the **Simulation/Parameters** menu. Select the **ode23tb** 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 on.

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:

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

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

If you zoom on the three motor currents, you can notice 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.

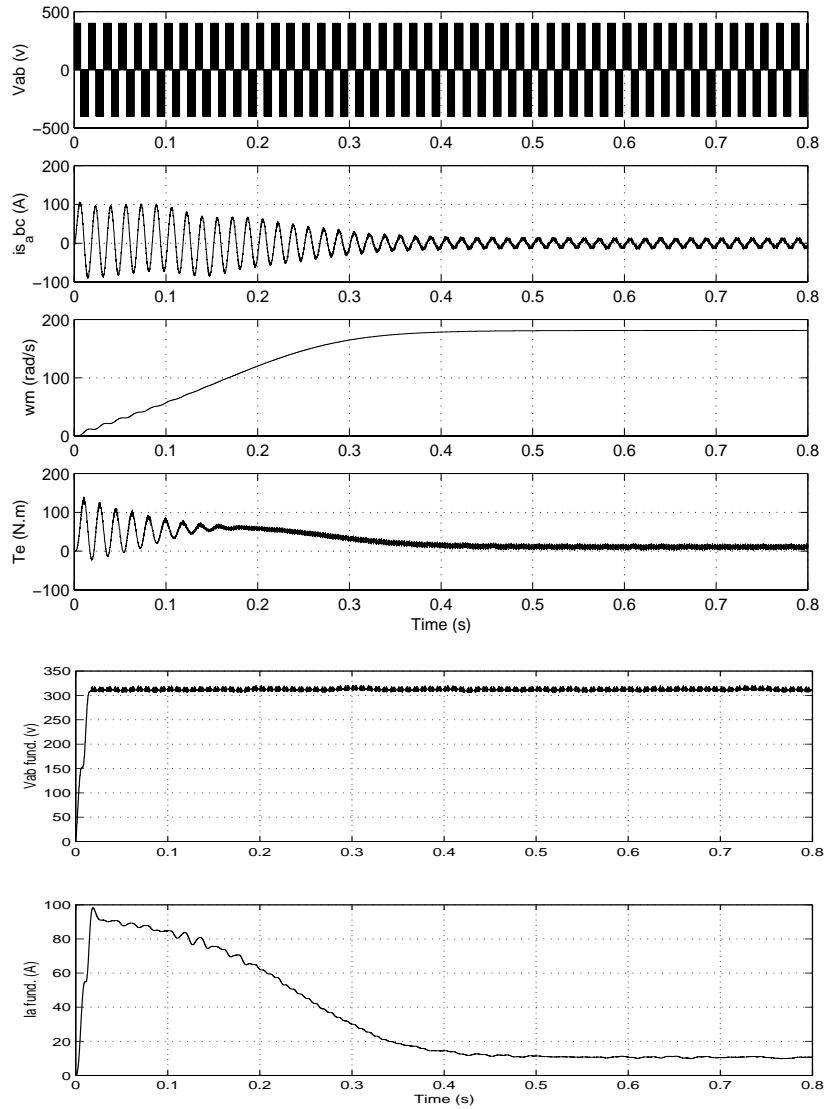


Figure 1-14: PWM Motor Drive; Simulation Results for Motor Starting at Full Voltage

Using the Multimeter Block

You probably have noticed that 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 will have to 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 on 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 on **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 as shown on Figure 1-13 (Trace 1: iSw1 Trace 2: iSw2).
- 6 Restart the simulation. The waveforms obtained for the first 20 ms are shown on Figure 1-15.

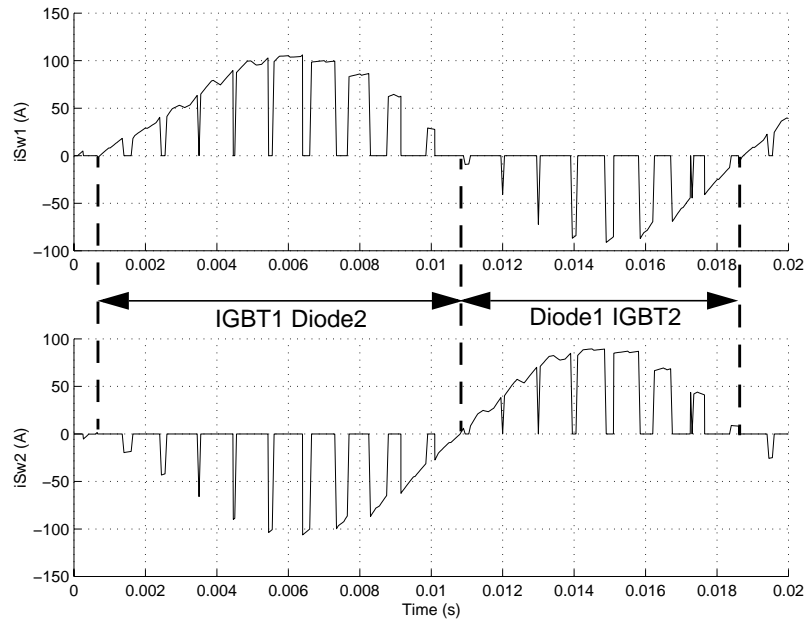


Figure 1-15: Currents in IGBT/Diode Switches 1 and 2

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 The Multimeter block use is not restrained to the Universal Bridge block. All the elements of the Electrical Sources and Elements libraries have a Measurement parameter where you can select voltages, currents, and 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 probably have noticed that the simulation using variable step integration algorithm is relatively long. Depending on your computer, it may take some minutes to simulate 1 second. In order to shorten the simulation time, you can discretize your circuit and simulate at fixed simulation time steps.

Copy the Discrete System block of the **powerlib** library in your circuit5 system. Open it and set the sample time to 10e-6 s. When you will restart the simulation, the power system, including the asynchronous machine, will be discretized at a 10 μ s sample time.

As there are no more continuous states in the electrical system, you don't need a variable step integration method to solve this system. In the **Simulation/Parameters/Solver** menu, you could select the **Fixed-step /discrete (no continuous states)** option. However, as your system contains two blocks that contain continuous states (Fourier blocks), you still need continuous integration. In the **Simulation/Parameters/Solver** menu, select the **Fixed-step /ode1 (Euler)** option and specify a Fixed-step of 10 μ s.

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.

Session 6: Three-Phase Systems and Machines

In this session you will:

- Learn how to use electrical machines
- Use the three-phase library and three-phase transformers
- Initialize machines to start simulation in steady state and use the **Machine Load Flow** option of the **powergui**

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

Three-Phase Network with Electrical Machines

During this session you will simulate the three-machines system shown on the single line diagram of Figure 1-16.

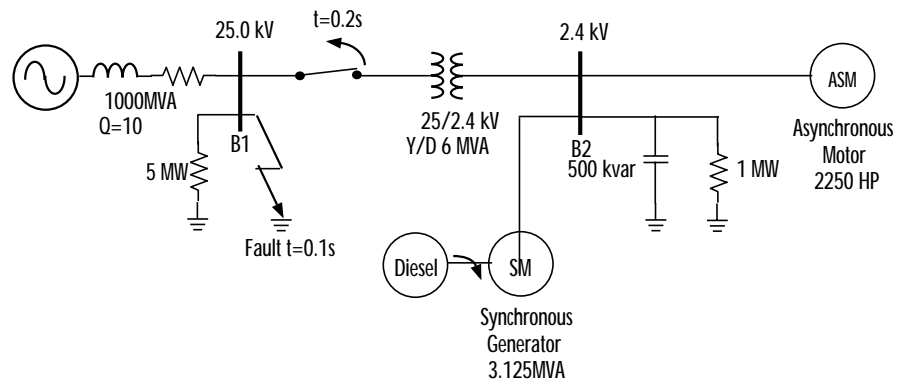


Figure 1-16: Diesel Generator and Asynchronous Motor on Distribution Network

This system consists of a plant (bus B2), simulated by a resistive and 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).

A 500 kvar capacitor bank is used for power factor correction at the 2.4 kV bus. 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 delivers 500 kW of active power. The synchronous machine controls the 2400V bus B2 voltage at 1.0 p.u. and generates 500 kW of active power. 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 will react to maintain the voltage and speed at a constant value.

This system has already been built in the Power System Blockset. Open the Demos library of **powerlib** and double-click on the demo entitled Three-Phase Machines and Load Flow. A system named **psbmachines** opens.

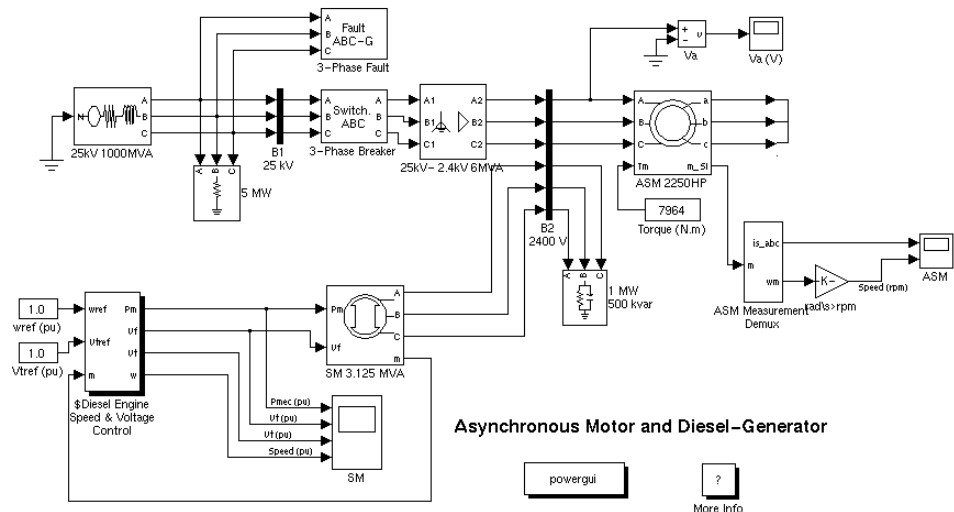


Figure 1-17: Power System of Figure 1-16 Built with the Power System Blockset

The Synchronous Machine (SM) block is using standard parameters whereas the Asynchronous machine (ASM) block is using S.I. parameters.

The other three-phase elements such as the inductive voltage source, the Y grounded/Delta transformer, and the loads are masked blocks built with standard single-phase Power System Blockset blocks. They are available in the Extra/Three-Phase library of **powerlib**. The 3-Phase Fault and the 3-Phase Breaker blocks are also available in the same library. If you open their dialog box, you will see how the switching times are specified. Special measurement blocks provided in the Machine library are used to demultiplex the SM and ASM machine outputs.

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].

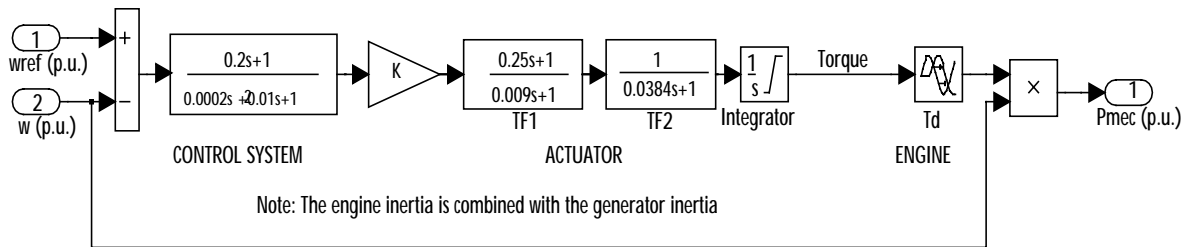


Figure 1-18: Diesel Engine and Governor System

If you simulate this system for the first time, you normally don't 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 p.u. Open the three scopes monitoring the SM and ASM speeds as well as the ASM stator currents. Start the simulation and observe the first 100 ms before fault is applied.

As the simulation starts, you will notice that the three ASM currents start from 0 and contain a slowly decaying DC component. The machine speeds will 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

In order 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 will learn how to use the **Load Flow** option of the **powergui** to perform a load flow and initialize the machines.

Double-click on the **powergui**. In the **Tools** menu, select **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 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 will generate or absorb 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 don't have any voltage source in your system, you must declare one of the machines as a swing machine. In the next section you will make a load flow with the 25 kV voltage source connected to bus B1 used as a swing bus.

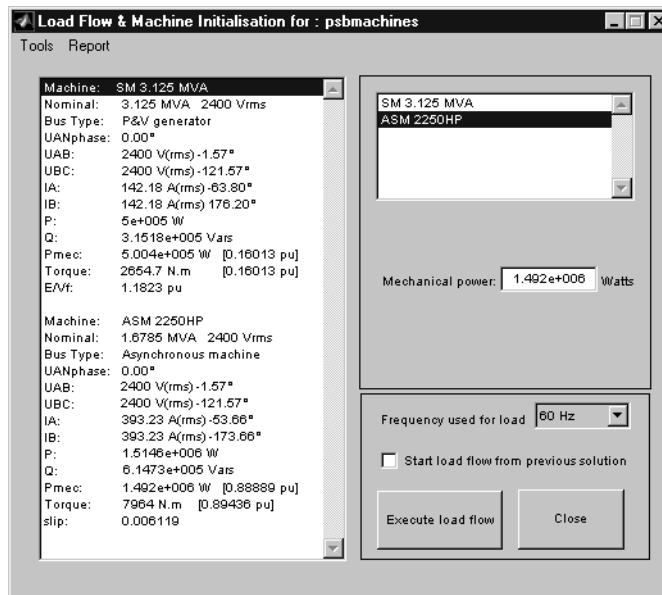
Load Flow Without a Swing Machine

In the **Load Flow** window, your SM Bus Type should be already initialized as PV generator indicating that the load flow will be performed with the machine controlling its active power and terminal voltage. By default, the desired Terminal Voltage is initialized at the nominal machine voltage (2400 Vrms). Keep it unchanged and set the Active Power to 500e3 (500 kW). 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 will now perform the load flow with the following parameters:

SM: Terminal voltage = 2400 Vrms; Active Power = 500kW.

ASM: Mechanical Power = 2000*746 W (2000 HP)

Select the **Execute load flow** button. Once the load flow is solved, the phasors of AB and BC machine voltages as well as currents flowing in phases A and B are updated as shown on the next figure.



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

SM: $P=500$ kW; $Q=315$ kvar;
 $P_{mec}=500.4$ kW (or $500/3125=0.1601$ pu)
 Field voltage $E_f=1.182$ pu

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

ASM: $P=1.515$ MW; $Q=615$ kvar; $P_{mec}=1.492$ MW (2000 HP)
 Slip $=0.006119$; Torque $=7964$ N.m

Close the **Load Flow** window.

The ASM torque value (7964 N.m) should be already 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 will see updated values of the measurement outputs. You can also click on 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_3phase_fault/Breaker1`) is 20.40 kV corresponding to a 24.985 kV, rms phase-phase voltage.

In order to start the simulation in steady state, the states of the Governor & Diesel Engine and the Excitation blocks should be also 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. The initial mechanical power has been already set to 0.1601 p. u. Open the Excitation block and notice that the initial terminal voltage and field voltage have been set respectively to 1.0 and 1.182 p. u.

Note that the load flow automatically initializes the machine blocks but not the associated control blocks. Therefore, if you perform a new load flow, you should change the initial values in the control blocks.

Open the four scopes displaying the terminal voltage, field voltage, mechanical power, and speed of the synchronous machine as well as the scope displaying the asynchronous motor speed. Start the simulation. The simulation results are shown in Figure 1-19.

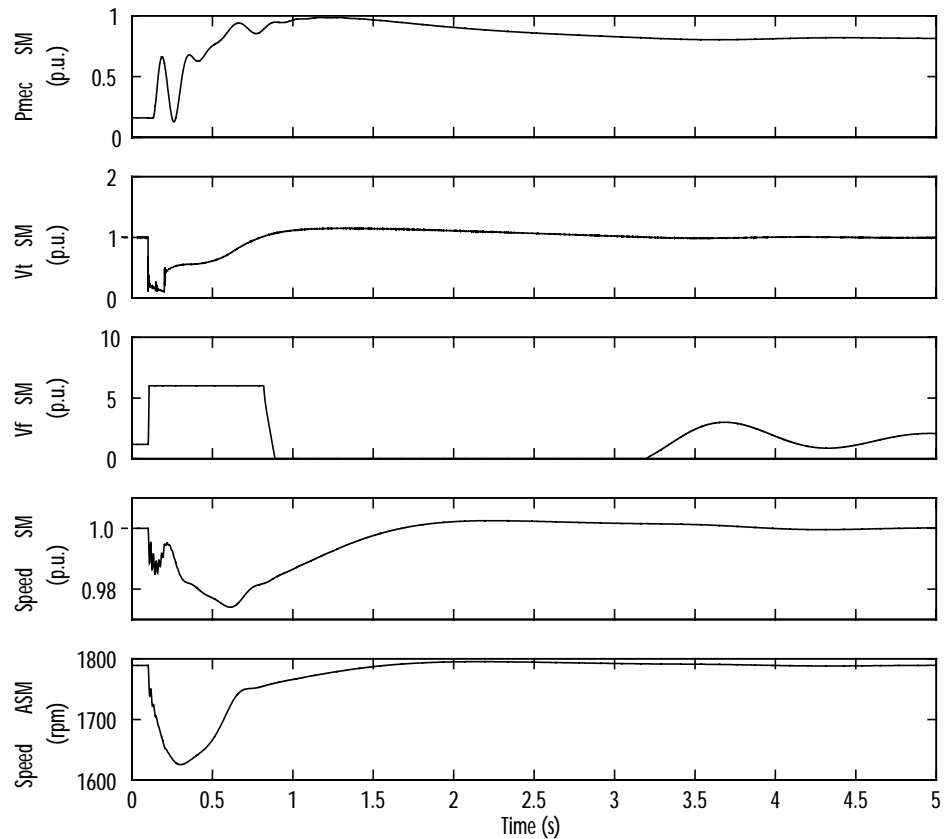


Figure 1-19: Simulation Results

Observe that during the fault the terminal voltage drops to about 0.2 p.u. and the excitation voltage hits the limit of 6 p.u. . After fault clearing and islanding, the SM mechanical power quickly increases from its initial value of 0.16 p.u. to 1 p.u. and stabilizes at the final value of 0.80 p.u. required by the resistive and motor load (1.0 MW resistive load + 1.51 MW motor load = 2.51 MW = $2.51/3.125 = 0.80$ p.u.). After 3 seconds the terminal voltage stabilizes close to its reference value of 1.0 p.u. The motor speed temporarily decreases from 1789 rpm down to 1625 rpm, then it 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 0.3s, you will 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 will make a load flow with two machine types: a *PV generator* and a *Swing generator*. In your **psbmachines** window, delete the inductive source and replace it with the Simplified Synchronous Machine block in p.u. that you will find in the Machines library. Rename it SSM 1000MVA and save this new system in your working directory as **psbmachine2**. Open the **SSM 1000MVA** dialog box and enter the following parameters:

1st line: 3 wires Y

2nd line: $V_n(V)$, $P_n(VA)$ $f_n(Hz)$: [1000e6 25e3 60]

3rd line: $H(s)$ $K_d()$ $p()$ [inf 0 2]

As you specify an infinite inertia, the speed and therefore the frequency of the machine will be kept constant.

4nd line: $R(p.u.)$ $X(p.u.)$: [0.1 1.0]

(Notice how easily you can specify an inductive short circuit level of 1000MVA and a quality factor 10 with the per unit system)

5th line: Leave all initial conditions at 0.

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 will absorb or generate 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**. Leave the SM bus Type as PV Generator and change the SSM bus Type to Swing Generator. Specify the load flow by entering the following parameters:

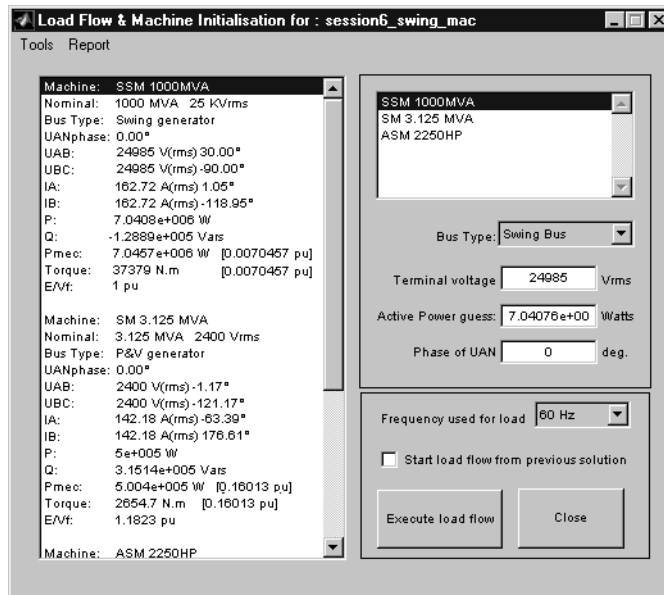
SM: Terminal Voltage =2400 Vrms; Active Power =500e3 W;

ASM: Mechanical power= 2000*746 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 may specify an active power which will be used as an initial guess and help load flow convergence. Specify the following parameters:

SSM: Terminal Voltage =24985 Vrms; (Voltage obtained at bus B1 from the previous load flow) Phase of UAN voltage =0 degrees;
Active Power = 0 W;

Click on the **Execute 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 field voltage are displayed for the SSM block:

P=500 kW; Q=315 kvar;
Pmec=500.4 MW (or $500/3125=0.1601$ pu);
Field voltage Ef=1.182 pu

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

$P=7.041\text{MW}$; $Q=-129\text{ kvar}$;
 $P_{\text{mec}}=7.046\text{MW}$ (or $7.046/1000=0.007046\text{ pu}$); $E=1.0\text{ pu}$

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

$P=1.515\text{MW}$; $Q=615\text{ kvar}$; $P_{\text{mec}}=1.492\text{ MW}$ (2000 HP)
 $\text{Slip}=0.006119$; $\text{Torque}=7964\text{ N.m}$

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 7.04 MW (6.0 MW resistive load + 1.51 MW load - 0.5 MW generated by SM = 7.01 MW, the difference (0.03 MW) corresponding to losses in the transformer).

Connect at inputs 1 and 2 of the SSM block two Constant blocks specifying respectively the required mechanical power (0.007046 p.u.) and its internal voltage (1.0 p.u.). Restart the simulation. You should get the same waveforms as those of Figure 1-19.

Reference

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

Session 7: Building and Customizing Your Own Nonlinear Models

The Power System Blockset provides a wide collection of nonlinear models. It may 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 will learn how to build such a nonlinear model. We will use as examples a simple saturable inductance and a nonlinear resistance.

Modeling a Nonlinear Inductance

Let us consider an inductor of 2 henries designed to operate at a nominal voltage $V_{nom} = 120$ V rms, and a nominal frequency $f_{nom} = 60$ 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 $L_{sat} = 0.5$ H. The nonlinear flux-current characteristic is plotted on Figure 1-20. Flux and current scales are in per units. The nominal voltage and nominal current are chosen as base values for the per-unit system.

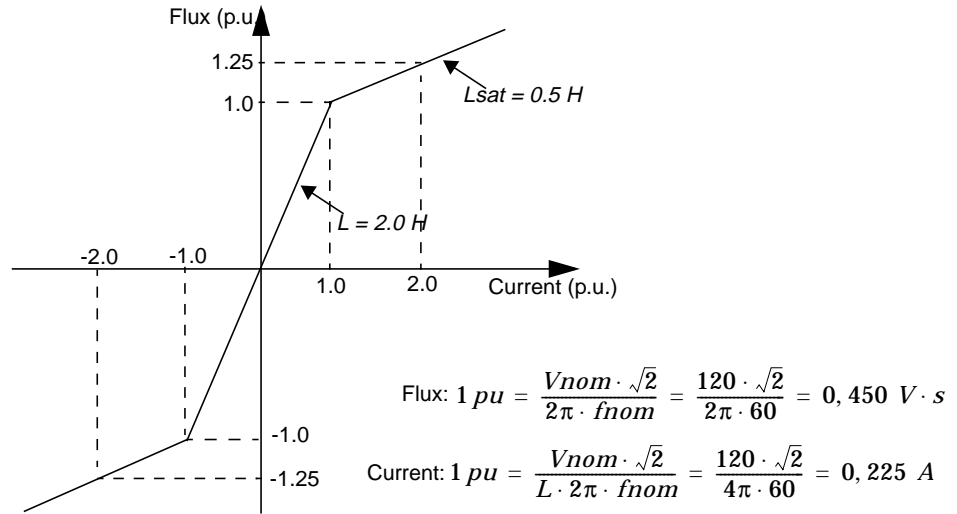


Figure 1-20: Flux-Current Characteristic of the Nonlinear Inductance

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

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

where:

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

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 on Figure 1-21.

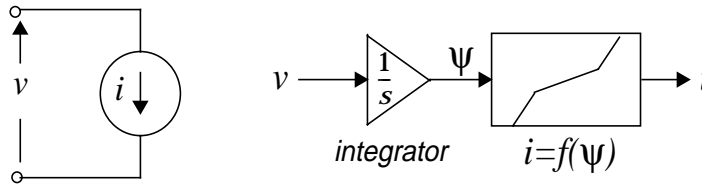


Figure 1-21: Model of a Nonlinear Inductance

Figure 1-22 shows a PSB 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. This Simulink output can be used 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 on Figure 1-21.

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 Figure 1-20.

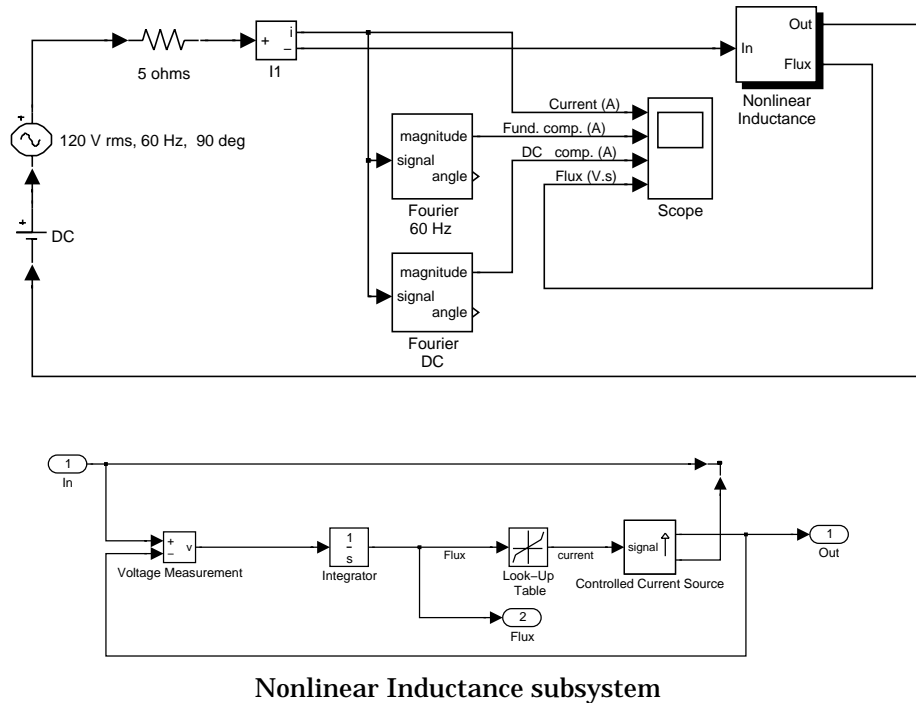


Figure 1-22: PSB Implementation of a Nonlinear Inductance

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 of Figure 1-22. To implement the $i = f(\psi)$ relation, specify the following vectors in the Look-Up Table block:

Vector of input values (flux): $[-1.25 \quad -1 \quad 1 \quad 1.25] * (120 * \sqrt{2}) / (2 * \pi * 60)$
 Vector of output values (current): $[-2 \quad -1 \quad 1 \quad 2] * (120 * \sqrt{2}) / (4 * \pi * 60)$

Save your circuit as circuit7.

Set the following parameters for the two sources:

AC source: Peak amplitude = $120 \cdot \sqrt{2}$; 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:

$$Peak \cdot Current = \frac{120 \cdot \sqrt{2}}{2 \cdot 2\pi \cdot 60} = 0,225 \text{ A}$$

$$Peak \cdot Flux = \frac{120 \cdot \sqrt{2}}{2\pi \cdot 60} = 0,450 \text{ V} \cdot s$$

Current and flux waveforms are shown on Figure 1-23.

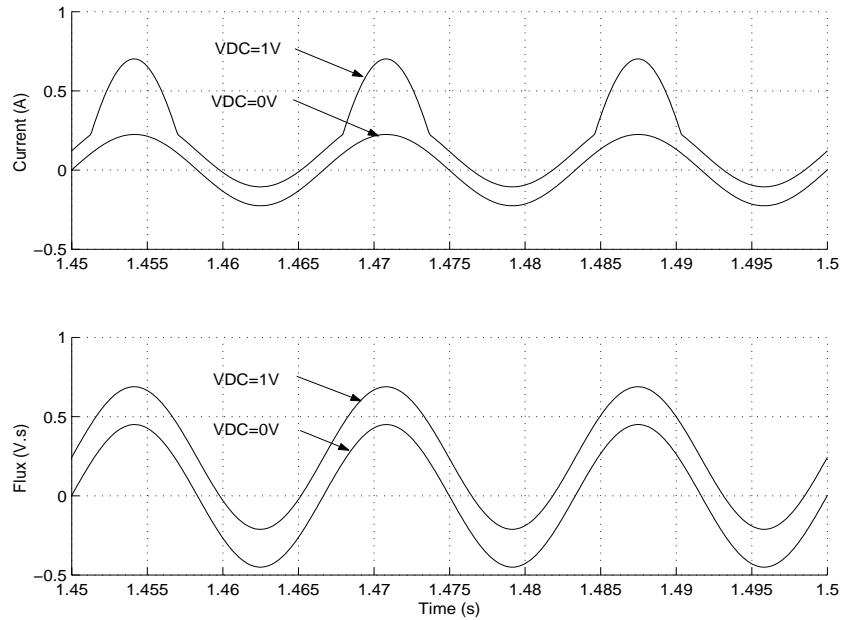


Figure 1-23: Current and Flux Waveforms Obtained with VDC=0V and VDC= 1 V

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 to 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 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 $1 \text{ V} / 0.5 \Omega = 0.2$. The current and flux waveforms obtained with and without saturation are superimposed on Figure 1-23.

Customizing Your Nonlinear Model

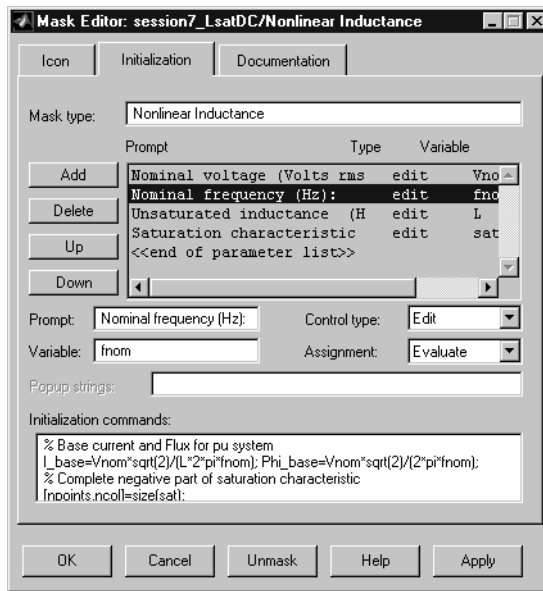
Up to now, you have used a nonlinear model with fixed parameters. If you plan to use this block in other circuits with different parameters (for example an inductance with a different voltage rating or a saturation characteristic

defined with more than two segments), you will find more convenient to enter the block parameters in a dialog box, rather than modifying individual blocks of your subsystem.

In the following section, you will learn how to use the Simulink masking facility to create a dialog box, an icon and a documentation for your model. For more details, refer to the chapter entitled “Using Masks to Customize Blocks” in the *Using Simulink* manual.

Block Initialization

Select the Nonlinear Inductance subsystem and in the **Edit** menu, select **Mask Subsystem**. The Mask Editor window appears.



Select the **Initialization** tab. In the **Mask type** field, enter:

Nonl i near Inductance

The parameters that you have to specify are: the nominal voltage, the inductance in the linear region and the flux-current characteristic (flux and current vectors in p.u.).

Click on **Add**. In the **Prompt** field, enter:

Nominal voltage (Volts rms) :

In the **Variable** field, enter the variable name associated with that field:

Vnom

Repeat the above steps to define the dialogs and associated variables listed below:

Nominal frequency (Hz):

fnom

Unsaturated inductance (H):

L

Saturation characteristic [i1(pu) phi1(pu); i2 phi2; ...]:

sat

In the **Initialization commands** section, type the following MATLAB commands. This code 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 p.u. 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 p.u. inductance (nominal value) for the first segment.

Press on the **OK** button to close **Mask Editor** window. Double-click on the icon of your masked block. Its dialog opens with all fields empty. Enter the values as shown on Figure 1-24.

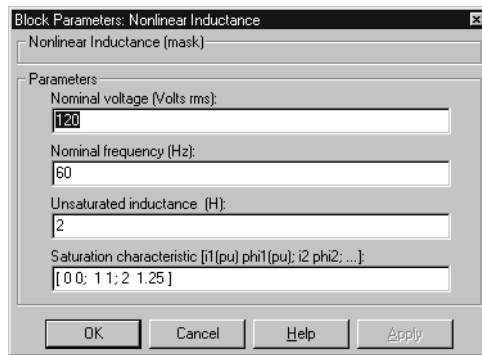


Figure 1-24: Dialog Box of Your Nonlinear Inductance

Before you can use the masked block, you must apply the two internal variables defined in the initialization section to the Look-Up Table block. Select your block and, In the **Edit** menu, select **Look Under Mask**.

The Nonlinear Inductance subsystem opens. Open the Look-Up Table block dialog box and enter the following variable names in the two fields:

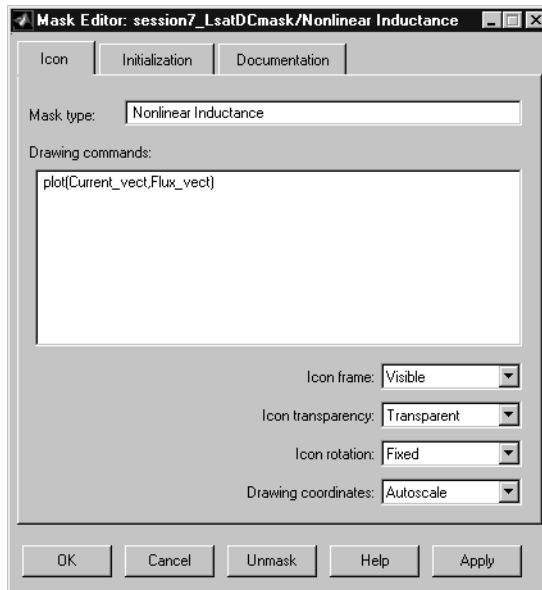
Vector of input values (flux): `Flux_vect`
 Vector of output values (current): `Current_vect`

Close the Nonlinear Inductance subsystem and start the simulation. You should get the same waveforms as shown on Figure 1-23.

Block Icon

In this section you will learn how to customize your block's icon and make it more attractive.

Select your block and, in the **Edit** menu, select **Edit Mask**. The Mask Editor window opens. Select the **Icon** tab.



In the **Drawing commands** section, you can specify any drawing that will appear in your block icon by using the plot function. You can, for example, plot the flux-current characteristic of your inductance.

Remember that the currents and fluxes of the nonlinear characteristic are stored respectively in the `Current_vect` and `Flux_vect` internal variables of the masked block. Type the following command in the **Drawing commands** section:

```
plot(Current_vect, Flux_vect)
```

Press **Apply** and notice that the saturation characteristic is displayed on the icon. Notice also that the input and output names have disappeared.

To make them visible, in the **Icon transparency** pop-up menu, select **Transparent**. Click on **OK** to close the **Mask Editor** window.

Block Documentation

In this section, you will add documentation appearing in your block dialog box. Select your block and, in the **Edit** menu, select **Edit Mask**. The **Mask Editor** window opens.

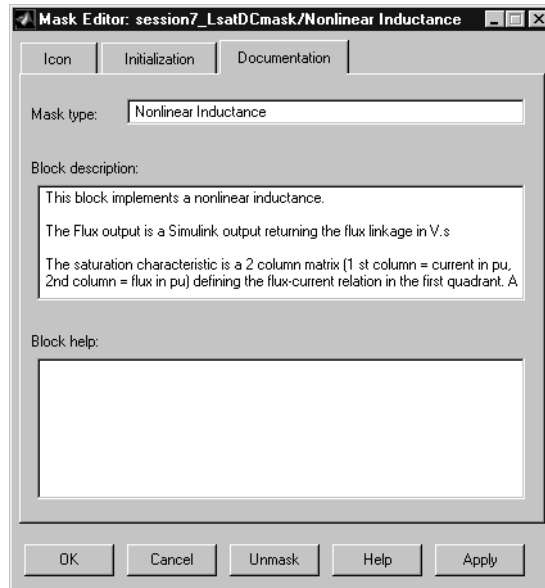


Figure 1-25: Icon and Documentation Windows of the Block Editor

Select the **Documentation** tab and enter in the **Block description** the text shown in the dialog box of Figure 1-26. Then, click on **OK** to close the **Mask Editor** window. The next time you will double-click on your block, this description will appear on the dialog box of the block.

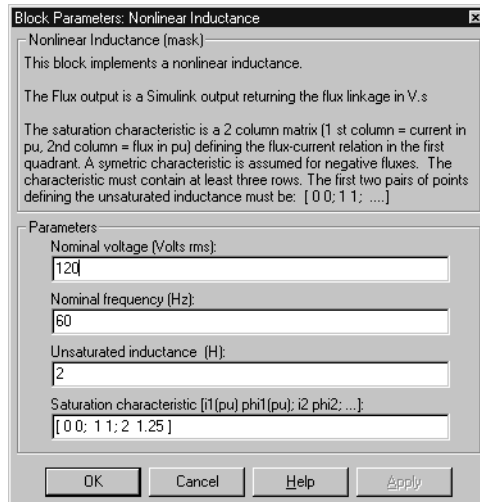


Figure 1-26: Block Dialog Box with Its Documentation

Modeling a Nonlinear Resistance

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

We use as an example a metal-oxide varistor (MOV) having the following V-I characteristic defined by the equation

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

where:

v, i = Instantaneous voltage and current

V_o = Protection voltage

I_o = Reference current used to specify the protection voltage

α = Exponent defining the nonlinear characteristic (typically between 10 and 50)

Figure 1-27 shows an application of such a nonlinear resistance to simulate a MOV used to protect equipment on a 120 kV network. In order to keep the circuit simple, only one phase of the circuit is represented.

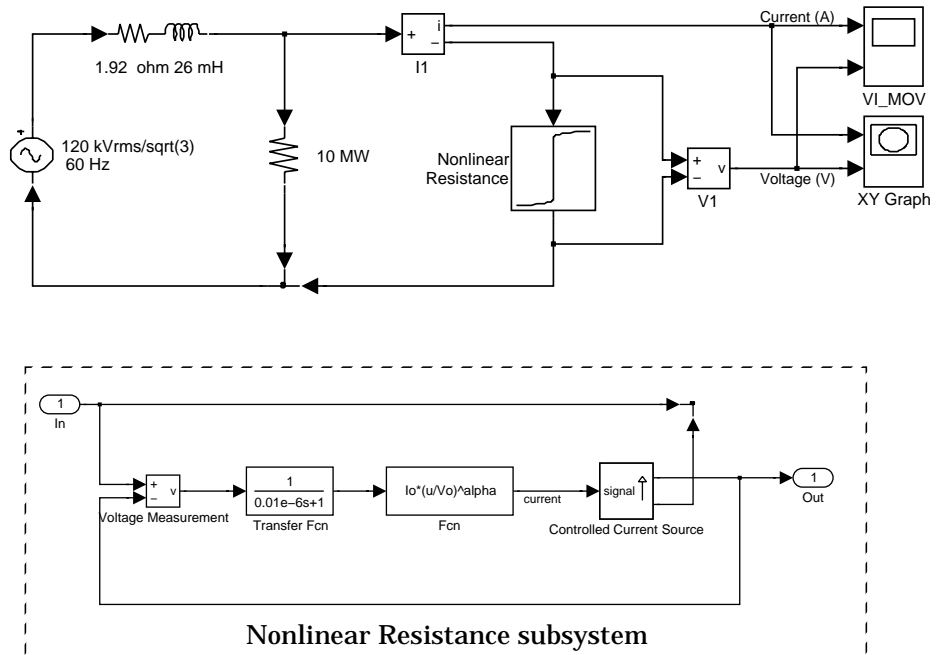


Figure 1-27: Nonlinear Resistance Applied on a 120 kV Network

Using blocks of the **powerlib** and Simulink libraries, build the circuit of Figure 1-27. 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.

Notice, that 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 function of voltage is known, the nonlinear $I(V)$ characteristic is implemented directly with a Fcn block from the Fcn & Tables library of Simulink.

This purely resistive model contains no states. It produces an algebraic loop in the state-space representation of the circuit, as shown in Figure 1-28. See Chapter 4, “Block Reference” for more details on how the Power System Blockset works.

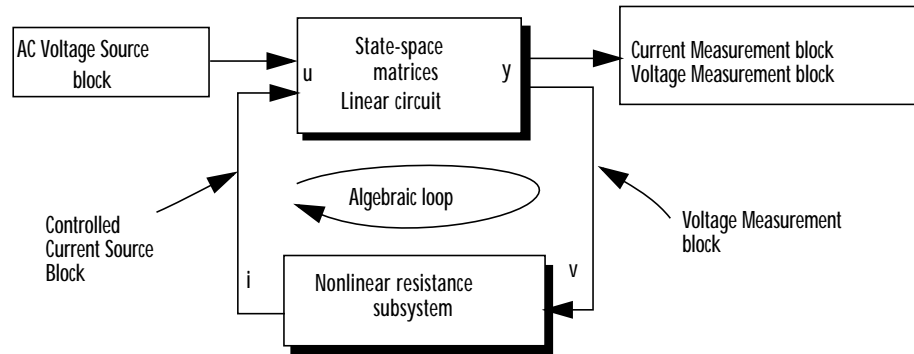


Figure 1-28: Algebraic Loop Introduced by the Nonlinear Resistance Model

Although Simulink is able to solve algebraic loops, this could result in slow simulation times. It is therefore recommended to break the loop with a block that will not change the nonlinear characteristic. We have introduced a first order transfer function $H(s)=1/(1+Ts)$ in the system, using a fast time constant ($T=0.01\mu s$).

Use the technique explained for the nonlinear inductance block to mask and customize your nonlinear resistance block, as shown on Figure 1-29.

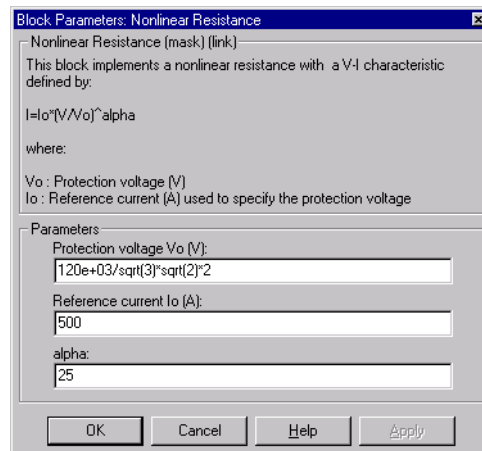


Figure 1-29: Dialog Box of the Nonlinear Resistance Block

Open the dialog box of your new masked block and enter the parameters shown on Figure 1-29. Notice that the protection voltage V_o has been set at 2 p.u. of the nominal system voltage. Adjust the source voltage at 2.3 p.u. by entering the following peak amplitude:

$$120e3/\sqrt{3} * \sqrt{2} * 2.3$$

Save your circuit as `circuit8`.

Using the `ode23tb` integration algorithm, simulate your `circuit8` system for 0.1 s. Results are shown on Figure 1-30.

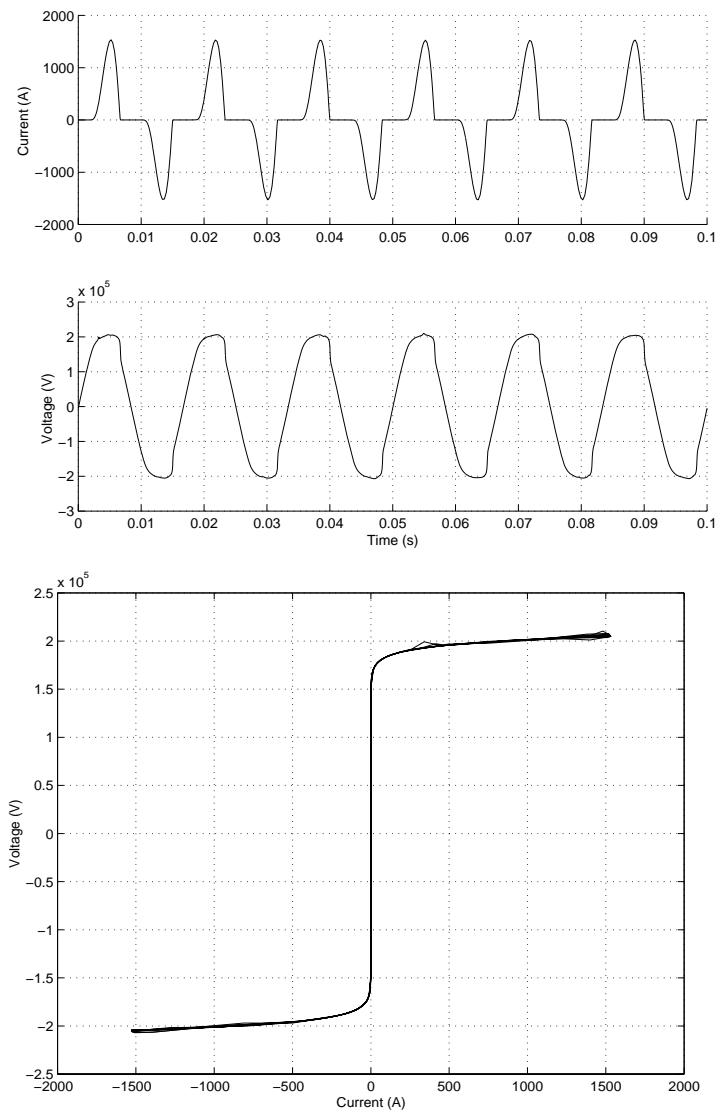


Figure 1-30: Current and Voltage Waveforms and V-I Characteristic Plotted by the X-Y Graph Block

Creating Your Own Library

Simulink gives you the possibility to create your own library of Power System Blockset blocks. 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_PSlibrary**. Next time that you develop a new model, you will add it to your personal library. You can also organize your library in different sub-libraries according to their functions as it is done in the **powerlib** library.

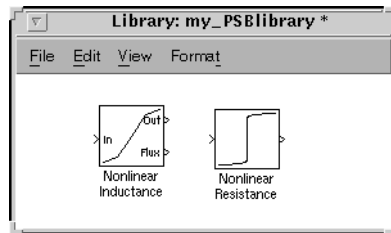


Figure 1-31: The Nonlinear Inductance and Resistance Blocks in my_PSlibrary

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

Connecting Your Model with Other Nonlinear Blocks

You will now learn how to avoid error messages that may 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 the Power System Blockset.

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

Let us 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_PSBlibrary**, you can build the circuit shown on Figure 1-32. Change the Breaker block parameters as follows:

External control: not checked

Switching times: [1/60]

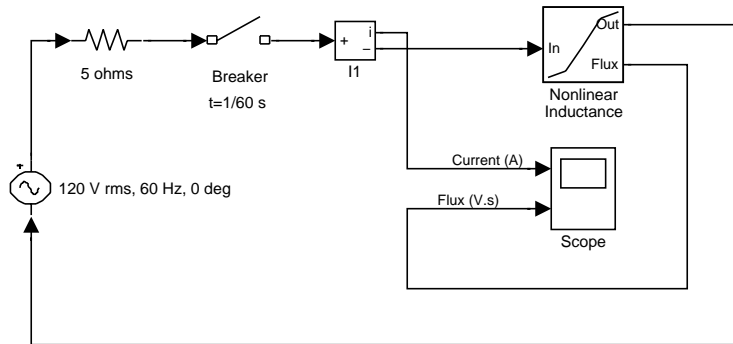
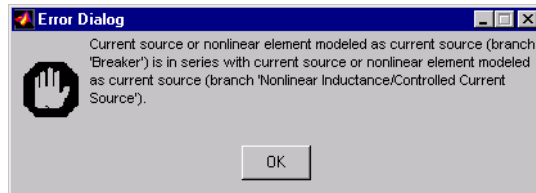


Figure 1-32: Circuit Topology Causing an Error

If you try to simulate this circuit, you will 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 $1\text{ M}\Omega$, across the Breaker block or the Inductance block.

In our 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 in order 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 ode23tb integration algorithm and simulate the circuit for 1 second. Voltage and current waveforms are shown on Figure 1-33.

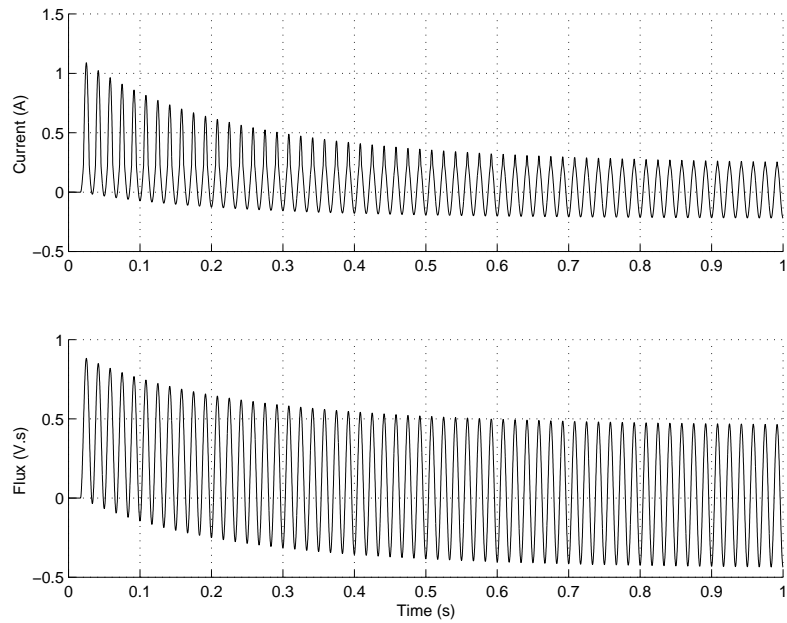


Figure 1-33: Current and Flux Waveforms When Energizing the Nonlinear Inductance with Maximum Flux Offset

Figure 1-33 shows that energizing the inductor at a zero crossing of voltage results in a maximum flux offset and saturation.

Case Studies

Series Compensated Transmission Network	2-3
Chopper-Fed DC Motor Drive	2-21
Synchronous Machine and Regulators	2-32
Variable-Frequency Induction Motor Drive	2-40
HVDC System	2-52

The case studies in this chapter were built to provide examples of uses for the Power System Blockset. They are:

- Series Compensated Transmission Network
- Chopper-Fed DC Motor Drive
- Synchronous Machine and Regulators
- Variable-Frequency Induction Motor Drive
- HVDC System

Cases 1 and 5 are studies of AC and DC transmission on power systems. Cases 2 and 4 illustrate typical applications of the Power System Blockset to motor drives. Case 3 demonstrates the performance of a nonlinear voltage regulator on a synchronous alternator.

Series Compensated Transmission Network

The example described in this section illustrates phenomena related to subsynchronous resonance in a series-compensated AC transmission network.

Description of the Transmission Network

The single diagram shown in Figure 2-1 represents a three-phase, 60 Hz, 735 kV power system transmitting power from a power plant consisting of six 350 MVA generators to an equivalent network through a 600 km transmission line. The transmission line is split in two 300 km lines connected between buses B1, B2, and B3.

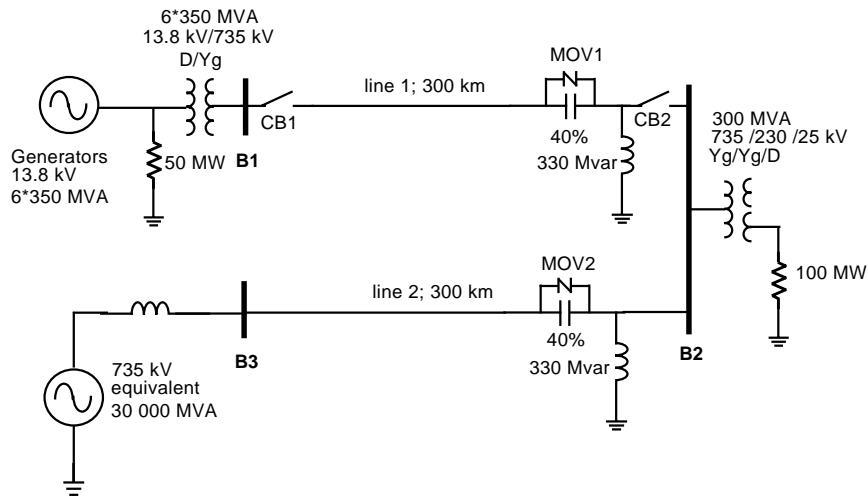


Figure 2-1: Series and Shunt Compensated Network

In order to increase the transmission capacity, each line is series compensated by capacitors representing 40% of the line reactance. Both lines are also shunt compensated by a 330 Mvar shunt reactance. The shunt and series compensation equipment is located at the B2 substation where a 300 MVA-735/230 kV transformer feeds a 230 kV-250 MW load through a 25 kV tertiary winding.

Each series compensation bank is protected by metal oxide varistors (MOV1 and MOV2). The two circuit breakers of line 1 are shown as CB1 and CB2.

We want to study the transient behavior of this circuit when faults are applied on line 1 and at bus B2.

This network is available in the `psb3phseri escomp.mdl` demonstration file. Load the `psb3phseri escomp` system and save it in your working directory as `case1` in order to allow further modifications to the original system.

Compare the circuit modeled in the Power System Blockset (Figure 2-2) with the schematic diagram of Figure 2-1. The generators are simulated with a Simplified Synchronous Machine block. A Three-Phase Transformer (Two-Windings) block and a Three-Phase Transformer (Three-Windings) block are used to model the two transformers. Saturation is implemented on the transformer connected at bus B2.

B1 and B3 blocks are 3-phase V-I Measurement blocks taken from the Measurements library of **powerlib_extras**. B2 is a similar block that has been modified to accommodate two 3-phase inputs and one 3-phase output. These blocks have been reformatted and given a black background color to give them the appearance of bus bars. They output the three line-to-ground voltages multiplexed on output 4 and the three line currents multiplexed on output 5. Open the dialog boxes of B1 and B2. See how the blocks are programmed to output voltages in p.u. and current in pu/100 MVA.

The fault is applied on line 1, on the line side of the capacitor bank. Note that the fault and the two line circuit breakers are simulated with blocks from the three-phase library. Open the dialog boxes of the 3-Phase Fault block and of the 3-Phase breakers CB1 and CB2. See how the initial breaker status and switching times are specified. A line-to-ground fault is applied on phase A at $t=1$ cycle. The two circuit breakers that are initially closed are then open at $t=5$ cycles, simulating a fault detection and opening time of 4 cycles. The fault is eliminated at $t=6$ cycles, one cycle after the line opening.

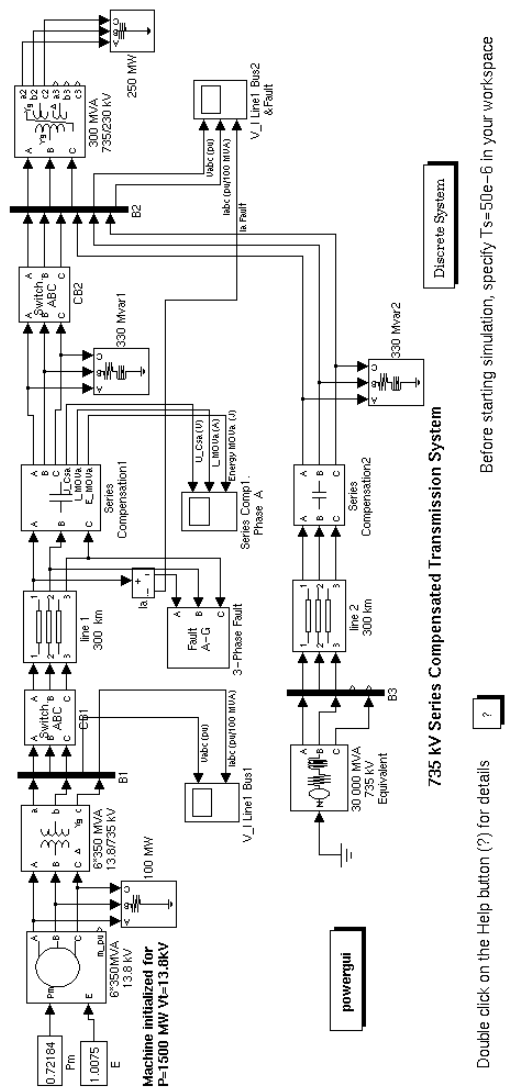


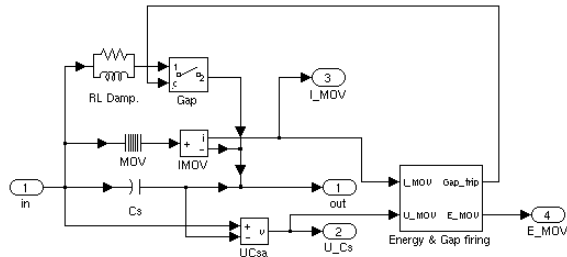
Figure 2-2: Series Compensated Network (psb3phseriescomp.mdl)

Series Compensation1 Subsystem

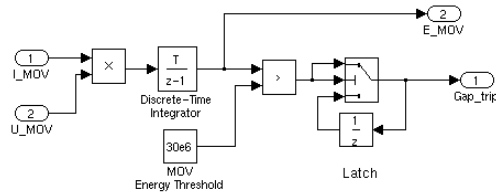
Now, open the Series Compensation1 subsystem of the `psb3phseriescomp` system. The three-phase module consists of three identical subsystems, one for each phase. A note indicates how the capacitance value and the MOV protection level have been calculated. Open the Series Compensation1/ Phase A subsystem. You can see the details of the connections of the series capacitor and the Surge Arrester block (MOV block). The transmission line is 40% series compensated by a 62.8 μF capacitor. The capacitor is protected by the MOV block. If you open the dialog box of the MOV block, you will notice that it consists of 60 columns and that its protection level (specified at a reference current of 500 A/column or 30 kA total) is set at 298.7 kV. This voltage corresponds to 2.5 times the nominal capacitor voltage obtained at a nominal current of 2 kA rms.

A gap is also connected in parallel with the MOV block. The gap is fired when the energy absorbed by the surge arrester exceeds a critical value of 30 MJ. In order to limit the rate of rise of capacitor current when the gap is fired, a damping RL circuit is connected in series. Open the Energy & Gap firing subsystem. It shows how the energy dissipated in the MOV is calculated by integrating the power (product of the MOV voltage and current). When the energy exceeds the 30 MJ threshold, a closing order is sent to the breaker block simulating the gap.

SeriesCompensation1/PhaseA Subsystem



SeriesCompensation1/PhaseA Subsystem/Energy & Gap firing



SeriesCompensation1 Subsystem

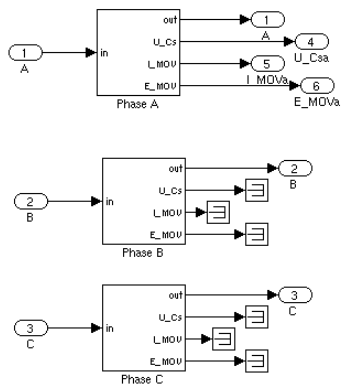


Figure 2-3: Series Compensation Module

Three-Phase Saturable Transformer Model

Open the **300 MVA 735/230 kV Transformer** dialog box and notice that the current-flux saturation characteristic has been set at:

[0 0 ; 0.0012 1.2; 1 1.45] in pu

This data is the current and flux values at points 1, 2, and 3 of the piece-wise linear approximation to the flux linkage curve shown in Figure 2-4.

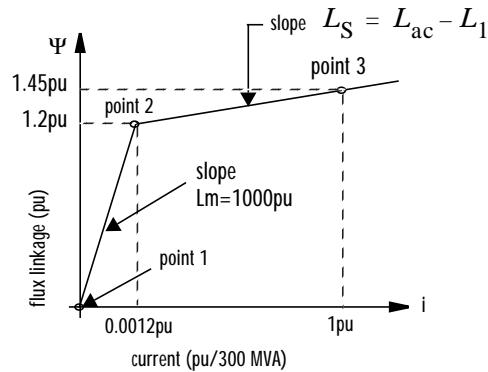
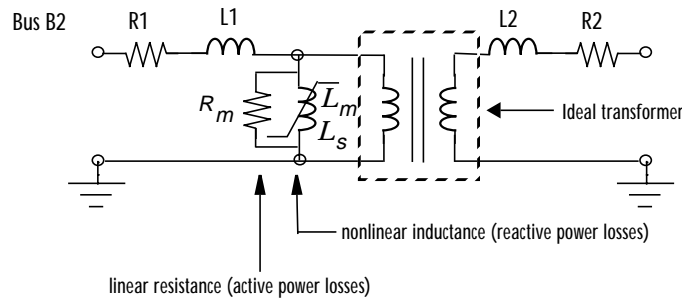


Figure 2-4: Saturable Transformer Model

The flux-current characteristic is approximated by two segments (see Figure 2-4). The saturation knee point is 1.2 p.u. The first segment corresponds to the magnetizing characteristic in the linear region (for fluxes below 1.2 pu). At 1 p.u. voltage, the inductive magnetizing current is $0.0010/1.0 = 0.001$ pu, corresponding to 0.1% reactive power losses.

The iron core losses (active power losses) are specified by the magnetization resistance $R_m=1000$ pu, corresponding to 0.1% losses at nominal voltage.

The slope of the saturation characteristic in the saturated region is 0.25 p.u. Therefore, taking into account the primary leakage reactance ($L_1=0.15$ pu), the air core reactance of the transformer seen from the primary winding is 0.4pu/300 MVA).

Setting the Initial Load Flow and Obtaining Steady-State

Before performing transient tests on your case1 system you must initialize your system for the desired load flow. Use the load flow utility of the **powergui** to obtain an active power flow of 1500 MW out of the machine with a terminal voltage of 1 p.u. (13.8 kV).

Open the **powergui**. In the **Tools** menu, select **Load Flow and Machine Initialization**. A new window appears. In the upper right window you have a the name of the only machine appearing in your system: 6*350*MVA 13.8 kV. Its Bus Type should be PV Generator and the desired Terminal Voltage should be already set to the default value (nominal voltage of 13800 V). In the Active Power field, enter 1500e6 (1500 MW) as the desired output power. Click on the **Execute Load Flow** button. Once the load flow is solved, the phasors of AB and BC machine voltages as well as currents flowing in phases A and B are updated in the left window. The required mechanical power to drive the machine is displayed in watts and in p.u. and the required excitation voltage E is displayed in pu:

```
Pmec : 1.5159e9 W [0.72184 pu]
E/Vf : 1.0075 pu
```

Notice that constant blocks containing these two values are already connected to the Pm and E inputs of the machine block. If you open the machine dialog box, you will see that the machine initial conditions (initial speed deviation $dw=0$; internal angle theta, current magnitudes and phase angles) have been automatically transferred in the last line.

Once the load flow is performed, you can obtain the corresponding voltage and current measurements at the different buses. In the **Tools** menu, select **Steady State Voltages and Currents**. You can observe, for example, the phasors for phase A voltages at buses B1, B2 and B3 and the current entering line 1 at bus B1.

B1/Va : 6.088e5 V ; 18.22 degrees
 B2/Va : 6.223e5 V ; 9.26 degrees
 B3/Va : 6.064e5 V ; 2.04 degrees
 B1/Ia : 1560 A ; 30.50 degrees

The active power flow for phase A entering line 1 is therefore

$$P_a = V_a \cdot I_a \cdot \cos(\varphi_a) = \frac{608.8 \text{ kV}}{\sqrt{2}} \cdot \frac{1.56 \text{ kA}}{\sqrt{2}} \cos(30.50 - 18.22) = 464 \text{ MW}$$

corresponding to a total of $464 \times 3 = 1392$ MW for the three phases.

Transient Performance for a Line Fault

In order to speed up the simulation, you need to discretize the network. Notice that your case1 contains the Discrete System block. The sample time is specified in the block dialog as a variable Ts. This sample time Ts is also used in the integrator block of the MOV energy calculator controlling the gap.

In the MATLAB window define the variable

Ts=50e-6;

Check that the simulation parameters are set as follows:

Stop time : 0.2

Solver options Type: Fixed-step; discrete(no continuous state)

Fixed step size: Ts

Line-to-Ground Fault Applied on Line 1

Check that the fault breaker is programmed for a line-to-ground fault on phase A. Start the simulation and observe the waveforms on the three scopes. These waveforms are reproduced on Figure 2-5.

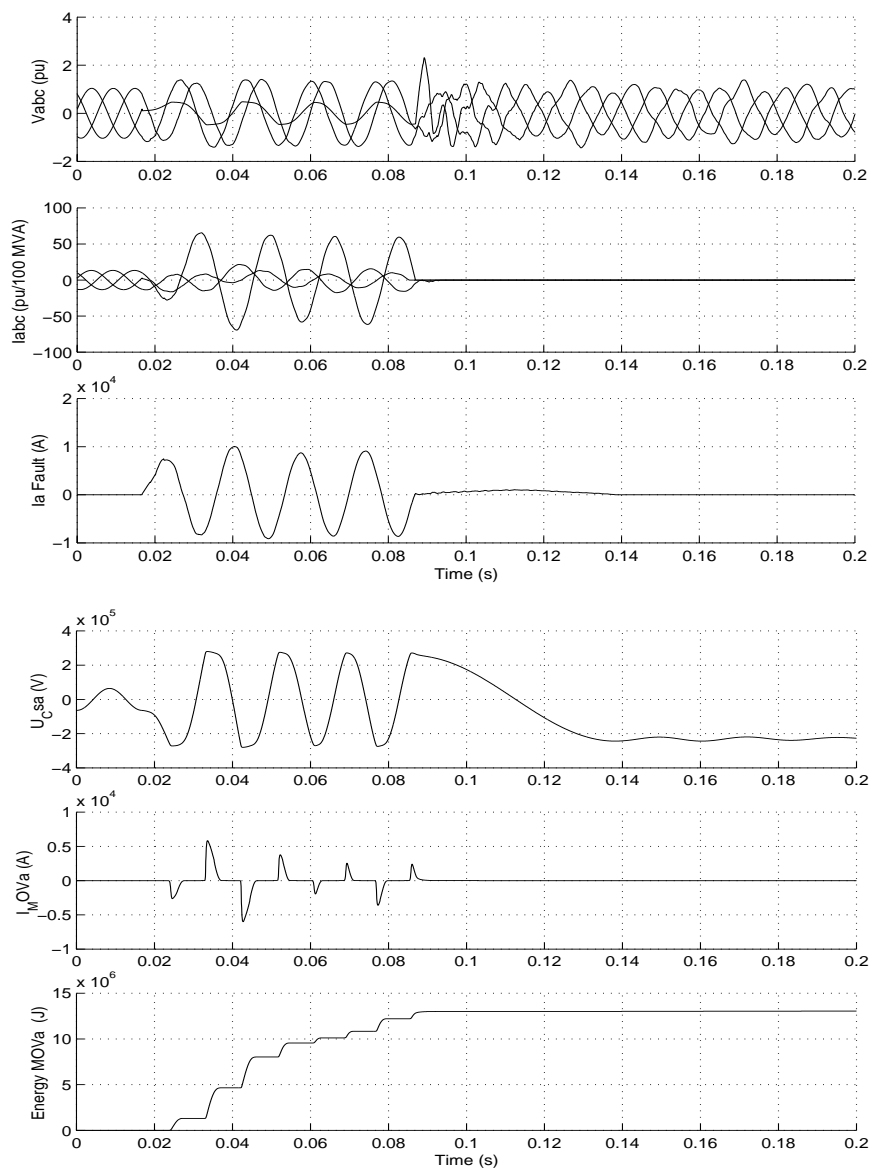


Figure 2-5: Simulation Results for a 4 Cycle Line-to-Ground Fault at the End of Line 1

The simulation starts in steady state. At the $t=1$ cycle, a line-to-ground fault is applied and the fault current reaches 10 kA (trace 3). During the fault, the MOV conducts at every half cycle (trace 5) and the energy dissipated in the MOV (trace 6) builds up to 13 MJ. At $t=5$ cycles the line protection relays open breakers CB1 and CB2 (see three line currents on trace 2) and the energy stays constant at 13 MJ. As the maximum energy does not exceed the 30 MJ threshold level, the gap is not fired. At the breaker opening, the fault current drops to a small value and the line and series capacitance starts to discharge through the fault and the shunt reactance. The fault current extinguishes at the first zero crossing after the opening order given to the fault breaker ($t=6$ cycles). Then the series capacitor stops discharging and its voltage oscillates around 220 kV (trace 4).

Three-Phase-to-Ground Fault Applied on Line 1

Open the **3-Phase Fault** block dialog box. Check the Phase B Fault and Phase C Fault, so that you now have a three-phase-to-ground fault.

Restart the simulation. The waveforms are reproduced on Figure 2-6.

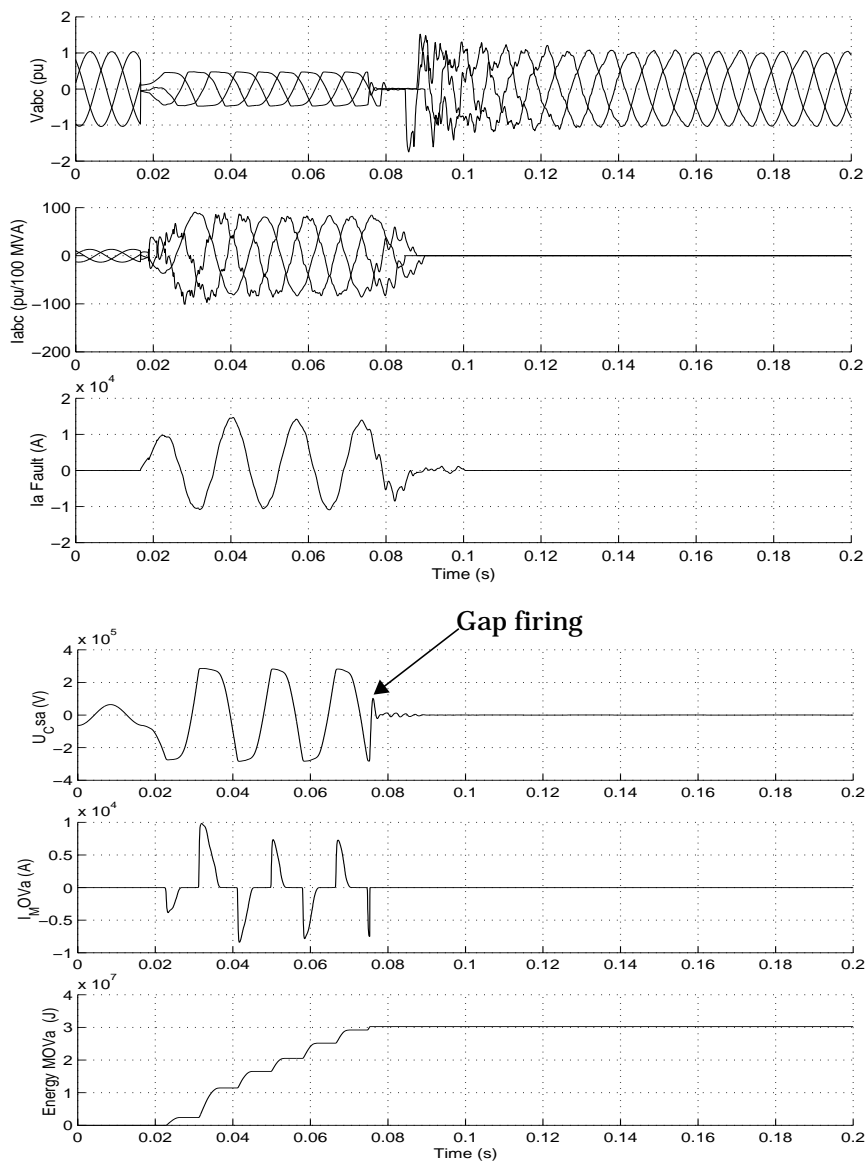


Figure 2-6: Simulation Results for a 4-Cycle Three-Phase-to-Ground Fault at the End of Line 1

Notice that during the fault the energy dissipated in the MOV (trace 6) builds up faster than in the case of a line-to-ground fault. The energy reaches the 30 MJ threshold level after three cycles, one cycle before opening of the line breakers. As a result, the gap is fired and the capacitor voltage (trace 4) quickly discharges to zero through the damping circuit.

Frequency Analysis

One particular characteristic of series compensated systems is the existence of subsynchronous modes (poles and zeros of the system impedance below the fundamental frequency). Dangerous resonances can occur if the mechanical torsion modes of turbine/generator shafts are in the vicinity of the zeros of the system impedance. Also, high subsynchronous voltages due to impedances poles at subsynchronous frequencies will drive transformers into saturation. The transformer saturation due to subsynchronous voltages is illustrated at the end of this case study. The torque amplification on a thermal machine is illustrated in another demonstration (see `psbthermal.mdl` file).

Now measure the positive-sequence impedance versus frequency seen from bus B2.

We have already explained in Session 2 of the Tutorial chapter how the Impedance Measurement block allows you to compute the impedance of a linear system from its state-space model. However, your `case1` system contains several nonlinear blocks (machine and saturation of transformers). If you connect the Impedance Measurement block to your system, all nonlinear blocks will be ignored. This is correct for the transformer, but you would get the impedance of the system with the machine disconnected. Before measuring the impedance, you must therefore replace the machine block with an equivalent linear block having the same impedance.

Delete the Simplified Synchronous Machine block from your `case1` circuit and replace it with the Inductive source with neutral block from the Three-Phase library of **powerlib_extras**. Open the block dialog box and set the parameters as follows in order to get the same impedance value ($L=0.22$ pu/ (6*350 MVA) Quality factor=15):

Voltage magnitude (V) : $13.8e3/\sqrt{3}*\sqrt{2}$
 Phase angle : 0
 Frequency (Hz) : 60
 Source resistance (Ohms) : $13.8^2/(6*350)*0.22/15$
 Source inductance (H) : $13.8^2/(6*350)*0.22/(2*\pi*60)$

Save your modified circuit as case1Zf.

Open the Measurements library of **powerlib** and copy the Impedance Measurement block in your model. This block contains a current source and a voltage measurement that will be used to perform the impedance measurement. Connect the two inputs of this block between phase A and phase B of B2 bus. Measuring the impedance between two phases will give two times the positive-sequence impedance. Therefore you must apply a factor of 1/2 on the impedance in order to obtain the correct impedance value. Open the **Impedance Measurement** dialog box and set the Multiplication factor to 0.5.

Now open the **powergui**. In the **Tools** menu select **Impedance vs Frequency Measurement**. A new window opens, showing your Impedance Measurement block name. Fill in the frequency range by typing 0: 500. Select the linear scales to display Z magnitude vs frequency plot. Check the **Save data to workspace** button and enter Zcase1 as the variable name that will contain the impedance vs frequency. Click on the **Display** button.

When the calculation is finished a graphic window appears with the magnitude and phase as a function of frequency. If you check in your workspace, you should have a variable named Zcase1. It is a two-column matrix containing frequency in column 1 and complex impedance in column 2.

The impedance as function of frequency (magnitude and phase) is shown in Figure 2-7.

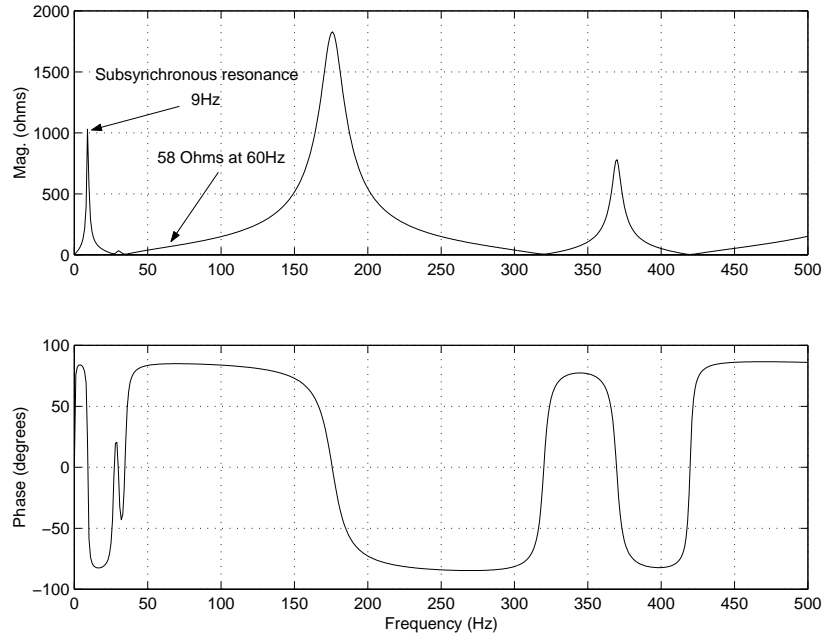


Figure 2-7: Impedance vs. Frequency Seen From Bus B2

You can observe three main modes: 9 Hz, 175 Hz, and 370 Hz. The 9 Hz mode is mainly due to a parallel resonance of the series capacitor with the shunt inductors. The 175 Hz and 370 Hz modes are due to the 600 km distributed parameter line. These three modes are likely to be excited at fault clearing.

If you zoom on the impedance in the 60 Hz region, you can find the system's short circuit level at bus B2. You should find a value of 58Ω at 60 Hz, corresponding to a three-phase short circuit power of $(735\text{kV})^2/58 = 9314 \text{ MVA}$.

Transient Performance for a Fault at Bus B2

The configuration of the substation circuit breakers normally allows to clear a bus fault without losing lines or transformers. You will now modify your case1

circuit in order to perform a three-cycle, three-phase-to-ground fault at bus B2 as shown on Figure 2-9:

- 1 Disconnect the 3-Phase Fault block and reconnect it as shown on Figure 2-9 so that the fault is now applied on bus B2.
- 2 Open the **3-Phase Fault** dialog box and make the following modifications:
Phase A, Phase B, Phase C, Ground Faults : all checked
Transition times : [2/60 5/60]
Transition status [1, 0, 1...]: (0/1)

You have now programmed a three-phase-to-ground fault applied at $t=1$ cycle.

- 3 Open the dialog boxes of circuit breakers CB1 and CB2 and make the following modifications:
Switching of Phase A: not checked
Switching of Phase B: not checked
Switching of Phase C: not checked
- The circuit breakers will not be switched anymore. They will stay at their initial state (closed).
- 4 Insert a Selector block (from the Simulink/Signals & Systems library) in the Vabc output of bus B2 connected to the Scope. Set its Elements parameter to 1. This will allow you to see clearly the phase A voltage on the scope.
- 5 You will now add blocks to read the flux and the magnetization current of the saturable transformer connected at bus B2.

Copy the Multimeter block from the Measurement library into your case1 circuit. Open the **Transformer** dialog box. In the **Measurements** pop-up menu, select **Flux and magnetization Current**. Open the Multimeter block. Verify that you have six signals available. Select flux and magnetization current on phase A, and click on **OK**.

- 6 You have now two signals available at the output of the Multimeter block. Use a Demux block to send these two signals on a two trace scope (See Figure 2-9).

- 7 In the **Simulation/Parameters** menu, change the stop time to 0.5. This longer simulation time will allow you to observe the expected low frequency modes (9 Hz). Start the simulation.

Waveforms of interest are plotted on Figure 2-8.

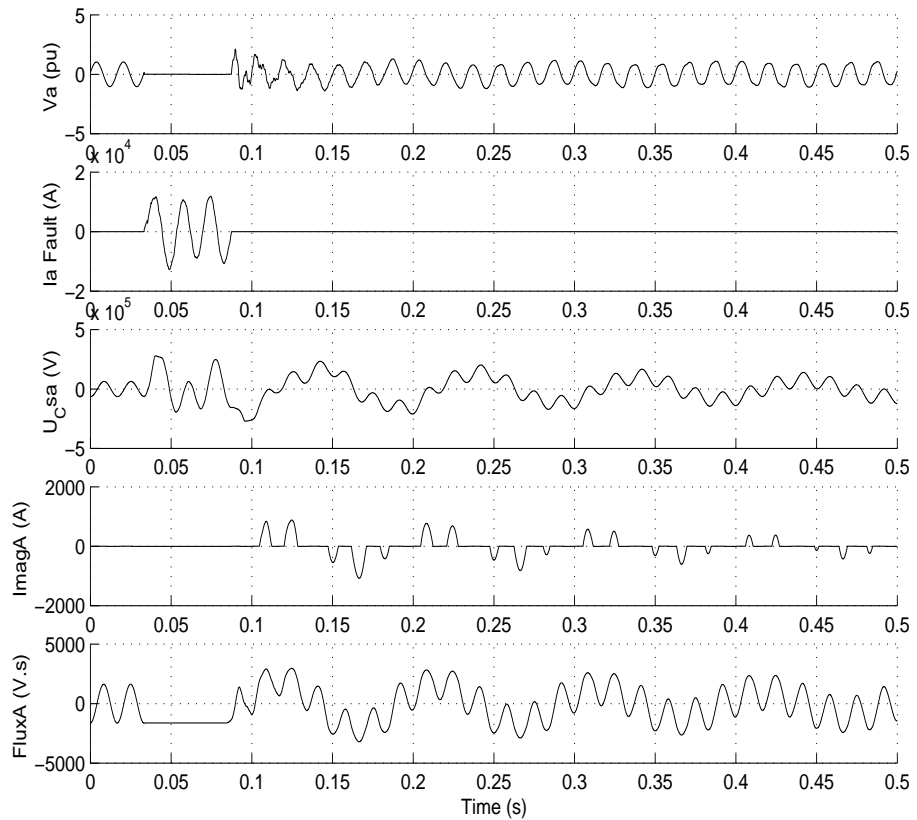


Figure 2-8: Simulation Results for a 3-Cycle 3-Phase-to-Ground Fault at Bus B2

The 9 Hz subsynchronous mode excited at fault clearing is clearly seen on the phase A voltage at bus B2 (trace 1) and capacitor voltage (trace 3). The 9 Hz voltage component appearing at bus B2 drives the transformer into saturation

as shown on the transformer magnetizing current (trace 4). The flux in phase A of the transformer is plotted on trace 5. At fault application the voltage at transformer terminals drops to zero and the flux stays constant during the fault. At fault clearing, when the voltage recovers, the transformer is driven into saturation as a result of the flux offset created by the 60 Hz and 9Hz voltage components. The pulses of the transformer magnetizing current appear when the flux exceeds its saturation level. This current contains a 60 Hz reactive component modulated at 9 Hz.

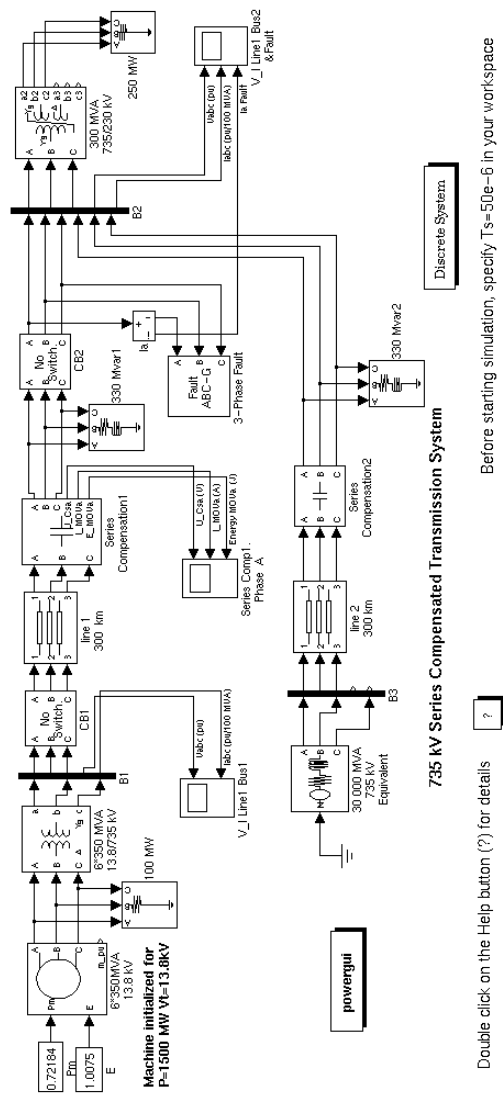


Figure 2-9: S Series Compensated Network Used for a Fault at Bus B2

Chopper-Fed DC Motor Drive

The example described in this section illustrates application of the Power System Blockset to the operation of a DC motor drive in which the armature voltage is controlled by a GTO thyristor chopper.

The objective of this example is to demonstrate the use of electrical blocks, in combination with Simulink blocks, in the simulation of an electromechanical system with a control system. The electrical part of the DC motor drive including the DC source, the DC motor, and the chopper is built using blocks from the Elements, Machines, and Power Electronics libraries. The DC Machine block of **powerlib** models both electrical and mechanical dynamics. The load torque-speed characteristic and the control system are built using Simulink blocks.

Description of the Drive System

A simplified diagram of the drive system is shown in Figure 2-10. The DC motor is fed by the DC source through a chopper that consists of the GTO thyristor, Th1, and the free-wheeling diode D1. The DC motor drives a mechanical load that is characterized by the inertia J , friction coefficient B , and load torque T_L (which can be a function of the motor speed).

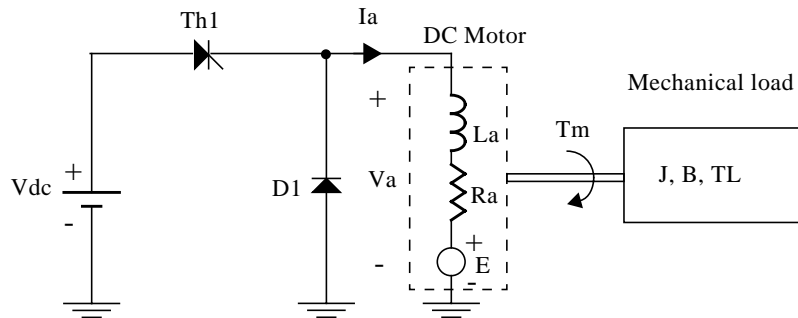


Figure 2-10: Chopper-Fed DC Motor Drive

In this diagram, the DC motor is represented by its equivalent circuit consisting of inductor L_a and resistor R_a in series with the counter electromotive force (emf) E .

The back EMF is proportional to the motor speed

$$E = K_E \omega$$

where K_E is the motor voltage constant and ω is the motor speed.

In a separately excited DC machine, the motor voltage constant K_E is proportional to the field current if

$$K_E = L_{af} I_f$$

where L_{af} is the field-armature mutual inductance.

The torque developed by the DC motor is proportional to the armature current I_a

$$T_m = K_T I_a$$

where K_T is the motor torque constant.

The DC motor torque constant is equal to the voltage constant

$$K_T = K_E$$

Thyristor Th1 is triggered by a pulse width modulated (PWM) signal to control the average motor voltage. Theoretical waveforms illustrating the chopper operation are shown in Figure 2-11.

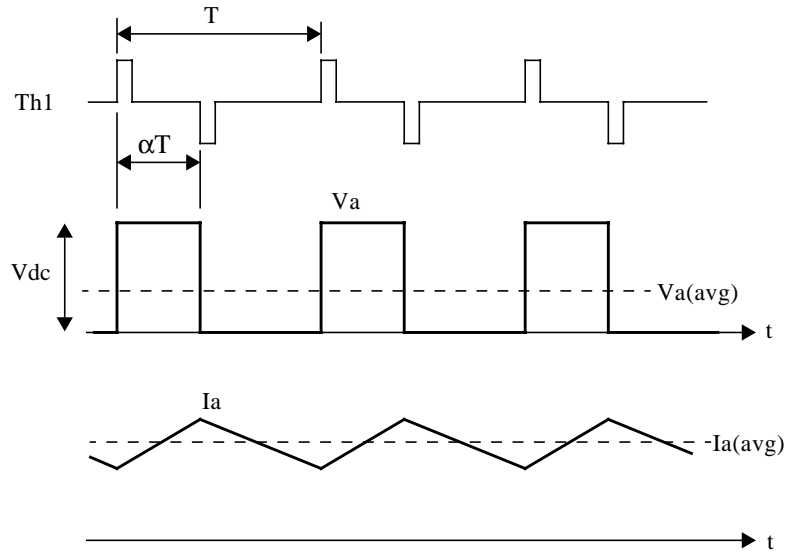


Figure 2-11: Waveforms Illustrating the Chopper Operation

The average armature voltage is a direct function of the chopper duty cycle α :

$$V_a(avg) = \alpha V_{dc}$$

Note that this relation is valid only when the armature current is continuous. In steady-state, the armature average current is equal to:

$$I_a(avg) = \frac{V_a(avg) - E}{R_a}$$

The peak-to-peak current ripple is

$$\Delta i = \frac{V_{dc}(1 - e^{-\alpha r} + e^{-r} - e^{-(1-\alpha)r})}{R_a(1 - e^{-r})}$$

where α is the duty cycle and r is the ratio between the chopper period and the DC motor electrical time constant:

$$r = \frac{T}{(L_a/R_a)}$$

In this case study, we consider a variable-speed DC motor drive using a cascade control configuration. A block diagram of this drive is shown in Figure 2-12.

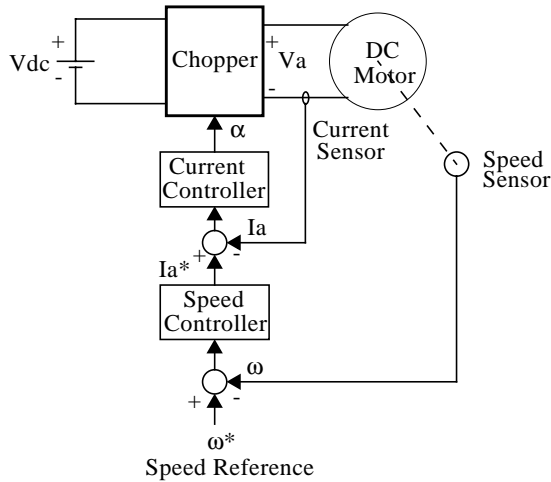


Figure 2-12: Variable-Speed DC Motor Drive

The motor torque is controlled by the armature current I_a , which is regulated by a current control loop. The motor speed is controlled by an external loop, which provides the current reference I_a^* for the current control loop.

Modeling the DC Drive

Open the `psbdcdrive.mdl` file of the **powerlib** library by typing `psbdcdrive` in MATLAB command window. A circuit diagram titled `psbdcdrive` will appear. Before running the example, save this circuit as `case2.mdl` in your working directory so that you can make further modifications without altering the original file.

The drive system diagram is built using electrical blocks contained in the **powerlib** library combined with Simulink blocks. Voltage Measurement and Current Measurement blocks are used as the interface between the two block types. The system diagram is shown in Figure 2-13.

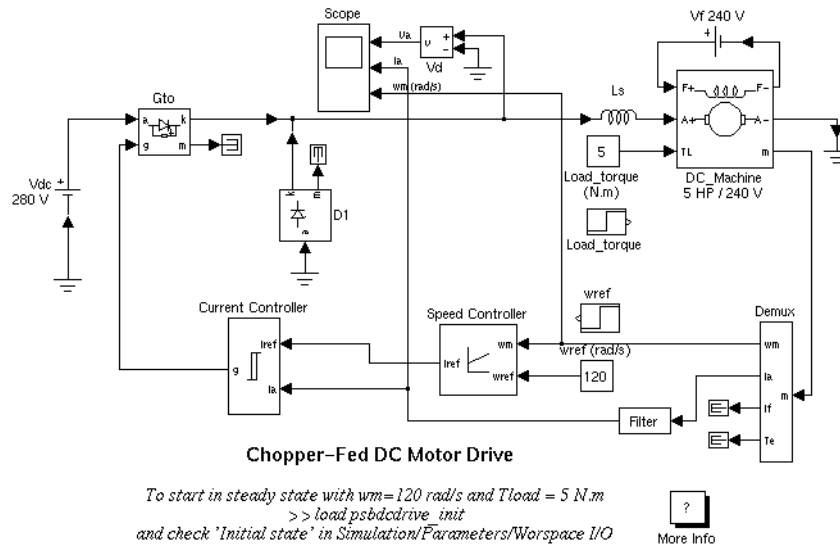
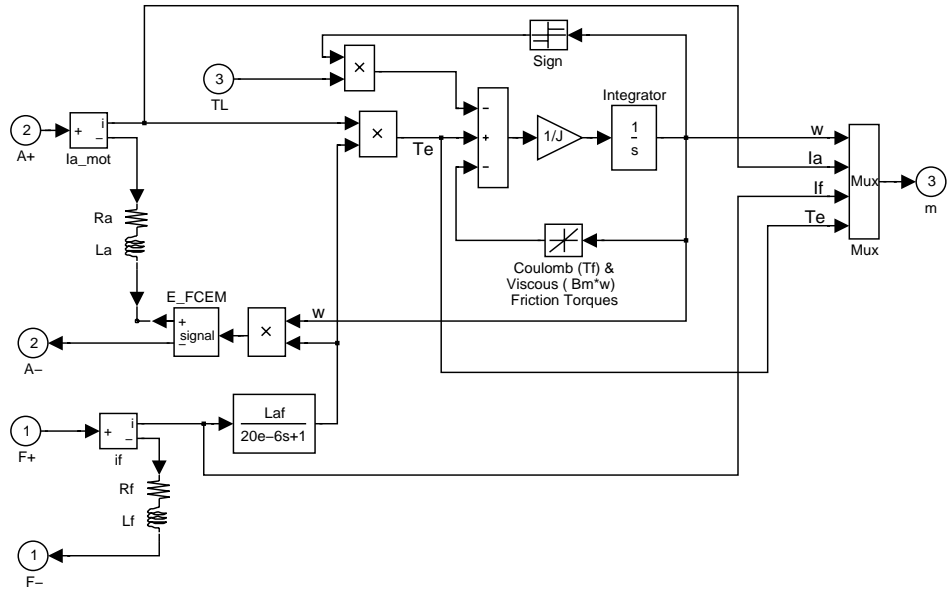


Figure 2-13: DC Motor Drive Using Power System Blockset (psbdcdrive.mdl)

The DC motor represented by the DC Machine block is modeled in two separate parts: electrical and mechanical. To view the Simulink model of the DC motor, click on the DC Machine block and use the LookUnderMask command in the **Edit** menu.



The armature circuit is represented by an RL circuit in series with a controlled voltage source, the value of which is $K_E\omega$.

The field circuit is represented by an RL circuit.

The mechanical part is represented by Simulink blocks, which implement the following equation:

$$T_m = J \frac{d\omega}{dt} + B\omega + \text{sgn } T_L$$

The DC machine parameters are set to the desired values by using the dialog mask of the DC Machine block.

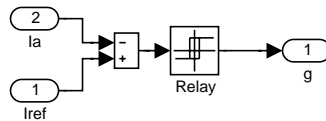
The load torque-speed characteristic can be implemented by a Simulink Fcn block.

The motor used in this case study is a separately excited 5 HP/240 V DC motor having the following parameters: $R_a = 0.5 \, \Omega$, $L_a = 10 \, \text{mH}$, $K_E = 1.23 \, \text{V}/(\text{rad/s})$, $K_T = 1.23 \, \text{N.m/A}$.

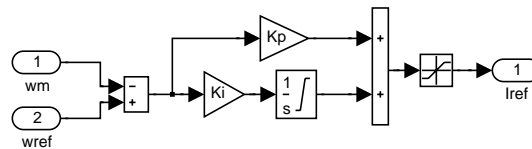
A 10mH inductor (L_s) is connected in series with the DC motor to smooth out the armature current. The constant excitation is implemented by connecting a DC Voltage Source block to the field winding.

The required trigger signal for the GTO thyristor is generated by a hysteresis current controller, which forces the motor current to follow the reference within $\pm h/2$ and $-h/2$ limits (h is the hysteresis band).

The current controller is a masked block that contains:



The speed control loop uses a proportional-integral controller, which is implemented by Simulink blocks:



Simulation of the DC Drive

Run the simulation by selecting **Start** from the **Simulation** menu in Simulink. Set the simulation parameters in the **Simulation Parameters** menu as follows:

Simulation time: Start Time: 0, Stop time: 1.2
 Solver Type: Variable-step ode23tb (stiff/TR-BDF2)
 Max Step Size: auto
 Initial Step Size: auto
 Relative Tolerance: 1e-3
 Absolute Tolerance: 1e-3

The motor voltage, current waveforms and motor speed are displayed on three axes of the scope connected to the variables V_d , I_a and ω .

Once the simulation is completed, you can return to the MATLAB command window to examine the results with more details by using the `plot` function.

Drive Starting

In this test, we simulate the starting transient of the DC drive. The inertia of the mechanical load is small in order to bring out the details of the chopper commutation details. The speed reference is stepped from 0 to 120 rad/s at $t=0.0$ s and we observe the DC motor speed and current.

The transient responses for the starting of the DC motor drive are shown in Figure 2-14.

Note that the final system state vector `xFinal` can be saved by selecting **Workspace I/O/Save to workspace/Final state** in the **Simulation Parameters** window. It can be used as initial state in subsequent simulation so that the simulation can start under steady-state conditions.

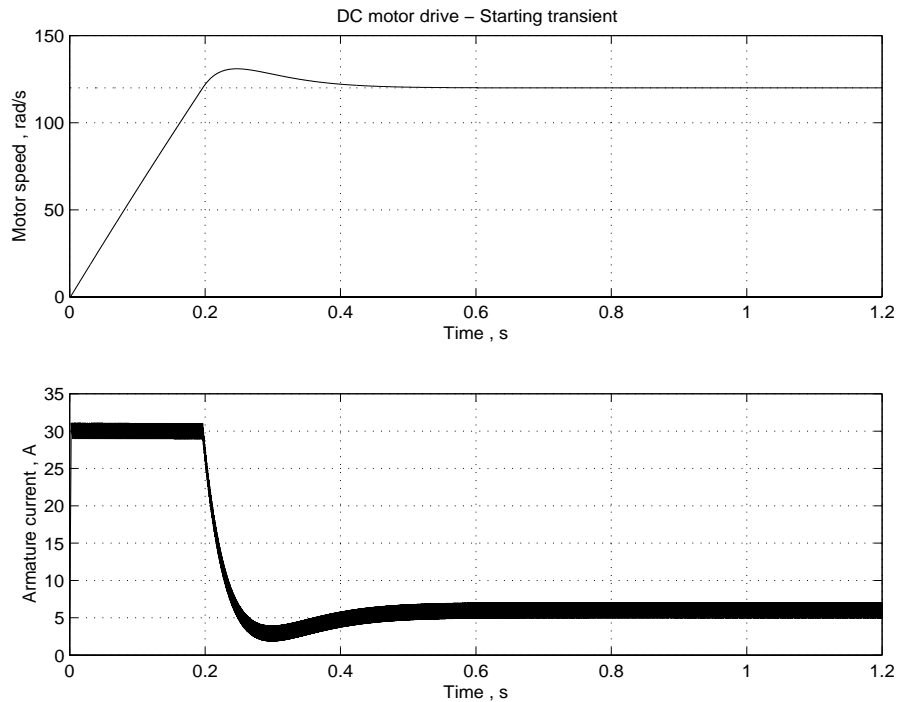


Figure 2-14: Starting of the DC Motor Drive

Steady-State Voltage and Current Waveforms

When the steady-state is attained, you can stop the simulation and plot the current and voltage waveforms using the variables V_a and I_a sent back in MATLAB workspace by the scope.

The DC motor current and voltage waveforms obtained at the end of the starting test are shown in Figure 2-15.

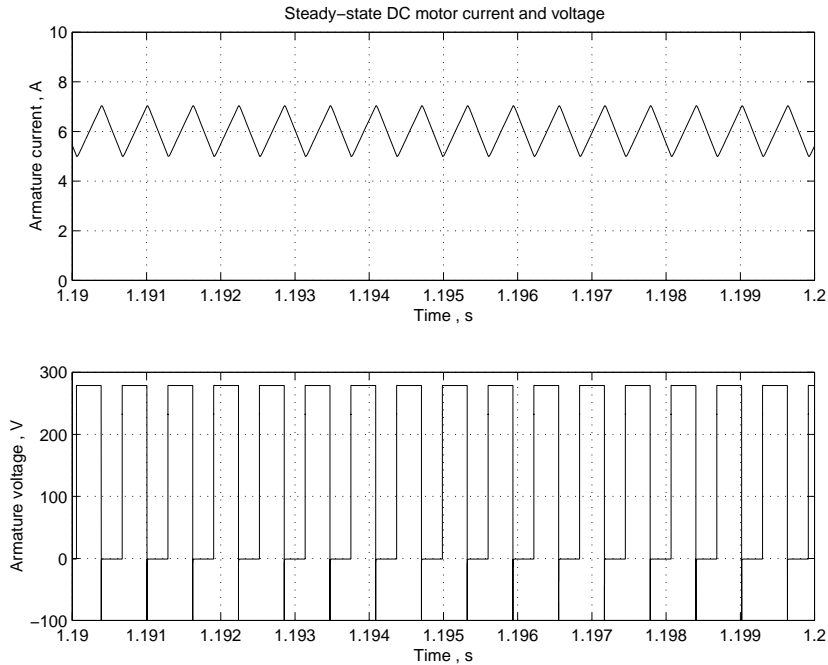


Figure 2-15: Steady-State Motor Current and Voltage Waveforms

Speed Regulation Dynamic Performance

We can study the drive dynamic performance, (speed regulation performance versus reference and load torque changes), by applying two successive changing operating conditions to the DC drive: a step change in speed reference and a step change in load torque.

Replace the block named `wref(rad/s)` and the block named `Load_torque(N.m)` in the diagram by two Simulink step blocks with different starting times. The speed reference steps from 120 rad/s to 160 rad/s at $t = 0.4$ s and the load torque steps from 5 N.m to 25 N.m at $t = 1.2$ s. The final state vector obtained with the previous simulation can be used as initial condition so that the simulation will start from steady-state. Load the `psbdcdrive_init.mat` file, which will create the `xInitial` variable. Select **Workspace I/O/Load from workspace/Initial state** in the **Simulation Parameters** window and restart the simulation.

The obtained response of the DC motor drive to successive changes in speed reference and load torque is shown in Figure 2-16.

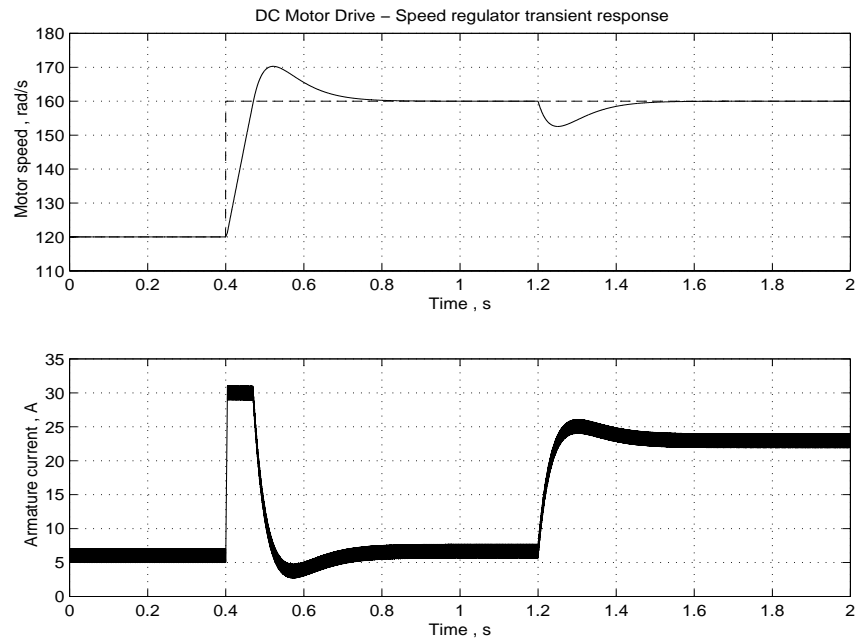


Figure 2-16: Dynamic Transient of the DC Motor Drive

References

- [1] Leonhard, W. *Control of Electrical Drives*. Springer-Verlag, Berlin 1996.

Synchronous Machine and Regulators

This case study investigates the application of a multi-input multi-output nonlinear controller to a system consisting of a hydraulic turbine and a synchronous generator connected to an infinite bus. The complete system is modeled using the Power System Blockset and Simulink blocks.

The objective of this case study is to demonstrate the use of the Synchronous Machine block connected to a complex control system implemented with Simulink blocks. The controller is based on a feedback linearization scheme. Its main goal is to control the rotor angle as well as the terminal voltage, to improve the stability properties, and to obtain good dynamic response. Simulation results will show that the nonlinear controller is able to replace the standard linear controllers and give better performance.

Introduction

Traditionally, stabilization of power systems was ensured by linear regulators such as the automatic voltage regulator (AVR), the speed governor, or the power system stabilizer (PSS). These compensators assume a linearized model of the power system around an operating point.

The demand for improved performance has created the need to operate power systems closer to the limits and therefore well outside the linear domain. Nonlinearities begin to have a significant effect then, especially after important disturbances that lead to a large variation of the operating point.

This case study will allow you to step through the design of a nonlinear controller that takes into account all the nonlinearities of the model. The objectives of the controller are to regulate both the terminal voltage and the internal power angle. The control inputs are the field excitation voltage and the gate opening of the turbine.

Mathematical Model

The model considered is a single machine infinite bus (SMIB) system, as shown in Figure 2-17. The machine is a synchronous generator driven by a hydraulic turbine.

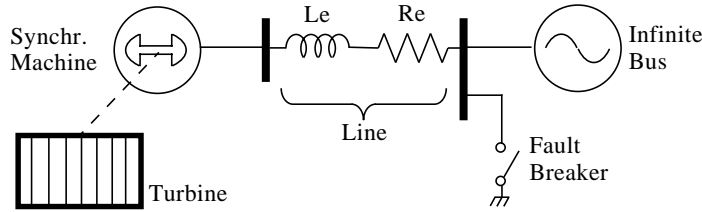


Figure 2-17: Diagram of Case Study

The dynamic equations of the machine that are used to derive the linear feedback controller are for the three-phase Synchronous Machine block, and the Hydraulic Turbine and Governor block (see Chapter 4, “Block Reference”). Since the synchronous machine is connected to an infinite bus, the dq terminal voltages v_d and v_q are constrained by the load equations. In the Park-transformed coordinates (rotor reference frame), we can write:

$$\begin{bmatrix} v_d \\ v_q \end{bmatrix} = R_e \begin{bmatrix} i_d \\ i_q \end{bmatrix} + L_e \begin{bmatrix} \dot{i}_d \\ \dot{i}_q \end{bmatrix} - \omega L_e \begin{bmatrix} i_q \\ -i_d \end{bmatrix} + V^\infty \begin{bmatrix} \cos(\delta - a) \\ -\sin(\delta - a) \end{bmatrix}$$

This equation can be combined with the complete model of the SMIB system in the nonlinear state-space form:

$$\dot{x} = F(x) + G(x)u$$

$F(x)$ and $G(x)$ are given by:

$$\begin{aligned} \tilde{\tau}(x) = & \begin{bmatrix} A_{11}x_1 + A_{12}x_3 + A_{13}x_2x_7 + A_{14}x_4 + A_{15}x_5x_7 + A_{16}\cos(x_6 - a) \\ A_{21}x_1x_7 + A_{22}x_2 + A_{23}x_3x_7 + A_{24}x_4x_7 + A_{25}x_5 + A_{26}\sin(x_6 - a) \\ A_{31}x_1 + A_{32}x_3 + A_{33}x_2x_7 + A_{34}x_4 + A_{35}x_5x_7 + A_{36}\cos(x_6 - a) \\ A_{41}x_1 + A_{42}x_3 + A_{43}x_2x_7 + A_{44}x_4 + A_{45}x_5x_7 + A_{46}\cos(x_6 - a) \\ A_{51}x_1x_7 + A_{52}x_2 + A_{53}x_3x_7 + A_{54}x_4x_7 + A_{55}x_5 + A_{56}\sin(x_6 - a) \\ (x_7 - 1)\omega_r \\ A_{71}x_1x_2 + A_{72}x_2x_3 + A_{73}x_2x_4 + A_{74}x_1x_5 + A_{75}x_7 + A_{76}\frac{x_8^3}{x_7x_9^2} \\ A_{81} - A_{82}\frac{x_8^2}{x_9^2} \\ A_{91}x_9 \end{bmatrix} \\ \tilde{x}(x) = & \begin{bmatrix} g_{11} & 0 & g_{31} & g_{41} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & g_{92} \end{bmatrix}^T \end{aligned}$$

The explicit expressions of the coefficients A and g can be found in the Chapter 4, “Block Reference” and are omitted here for simplicity. The other terms of the state-space equation are x , the vector of state variables, and u , the vector of control inputs. They are defined as follows:

$$\begin{aligned} x &= \begin{bmatrix} i_d & i_q & i_{fd} & i_{kd} & i_{kq} & \delta & \omega & q & G \end{bmatrix}^T \\ u &= \begin{bmatrix} v_{fd} & u_G \end{bmatrix}^T \end{aligned}$$

The currents i_d , i_q and voltages v_d , v_q are the projection of the actual line currents and terminal voltages on the direct and quadrature axes (d-q frame). i_{fd} and v_{fd} represent the field current and voltage. i_{kq} and i_{kd} represent the damper windings currents, and ω the angular speed of the machine. δ is the electrical angle measured from a synchronously rotating frame. G and q are respectively the opening of the gate and the flow rate of the turbine. Finally, u_G is the voltage applied to the gate servo-motor.

Feedback Linearization Design

The input-output feedback linearization technique consists in the exact cancellation of the nonlinearities of the system in order to obtain a linear relationship between inputs and outputs in closed-loop. The nonlinear control law is deduced by successively differentiating each of the outputs until at least one input appears. Consider the first output as the terminal voltage V_t :

$$y_1 = V_t = \sqrt{V_d^2 + V_q^2}$$

The terminal voltage is a complicated function of the state variables in which the control input V_{fd} appears explicitly with a multiplying factor of a very small order of magnitude. We therefore ignore this direct dependence between V_t and V_{fd} and compute the time derivate of output y_1 :

$$\begin{aligned} \frac{dy_1}{dt} &= \alpha_1(x) + \beta_{11}(x)u_1 + \beta_{12}(x)u_2, \quad \text{where} \\ \alpha_1(x) &= \frac{\partial V_t}{\partial x} \cdot F(x) \\ &= \frac{1}{2V_t} \left(2V_d \frac{\partial V_t}{\partial x} + 2V_q \frac{\partial V_t}{\partial x} \right) F(x) \\ \beta_{11}(x) &= \frac{\partial V_t}{\partial x} \cdot G_1(x) \\ &= \frac{1}{2V_t} \left(2V_d \frac{\partial V_t}{\partial x} + 2V_q \frac{\partial V_t}{\partial x} \right) G_1(x) \\ \beta_{12}(x) &= \frac{\partial V_t}{\partial x} \cdot G_2(x) \\ &= 0 \end{aligned}$$

The second output y_2 is the angle δ that has to be differentiated three times before the inputs appear. This yields:

$$\begin{aligned} \frac{d^3 y_2}{dt^3} &= \omega_{r\overline{\partial x}} \cdot F(x) + \omega_{r\overline{\partial x}} G_1(x) u_1 + \omega_{r\overline{\partial x}} G_2(x) u_2 \\ &= \alpha_2(x) + \beta_{21}(x) u_1 + \beta_{22}(x) u_2 \end{aligned}$$

Combining the equations of the outputs, the following input-output nonlinear system is obtained

$$\begin{bmatrix} y_1^{(1)} \\ y_2^{(3)} \end{bmatrix} = \begin{bmatrix} \alpha_1(x) \\ \alpha_2(x) \end{bmatrix} + \begin{bmatrix} \beta_{11}(x) & 0 \\ \beta_{21}(x) & \beta_{22}(x) \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

and the nonlinear control law is easily deduced.

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} \beta_{11}(x) & 0 \\ \beta_{21}(x) & \beta_{22}(x) \end{bmatrix}^{-1} \left[- \begin{bmatrix} \alpha_1(x) \\ \alpha_2(x) \end{bmatrix} + \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \right]$$

This will yield, in closed-loop, the exactly linearized input-output system

$$\begin{bmatrix} y_1^{(1)} \\ y_2^{(3)} \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$$

Once the system has been linearized, any linear control design can be applied to regulate the outputs. Here, the pole placement method was chosen and the following linear control law is proposed:

$$\begin{aligned} v_1 &= k_{11}(V_t - V_{tref}) \\ v_2 &= k_{21}(\delta - \delta_{ref}) + k_{22}(\dot{\delta} - \dot{\delta}_{ref}) + k_{23}(\ddot{\delta} - \ddot{\delta}_{ref}) + k_{24}(x_9 - x_{9ref}) \end{aligned}$$

The last term in the equation of v_2 is introduced in order to stabilize the internal dynamics. These dynamics come about because the original nonlinear system is a ninth order while the linearized system is a fourth order. This is called *partial linearization* and we must ensure that the remaining dynamics

are asymptotically stable. A complete treatment of this question can be found in reference [1].

Simulation Results

The performance of the nonlinear controller is tested on the nonlinear turbine-generator system. The controller and turbine are simulated using Simulink blocks while the generator is represented by the Synchronous Machine block from the **powerlib** library. A three-phase short-circuit is simulated on the load busbar and the fault is cleared after 100 ms. The performance of the nonlinear controller is analyzed.

The case study is included in the `psbregulator.mdl` file. The system is illustrated in Figure 2-18 below. Before running the simulation, make sure that the simulation parameters are set as follows:

- Solver: ode23tb; Maximum order: 5
- Stop time: 1. 0
- Max step size: auto; Initial step size: auto
- Relative tolerance: 1e-3; Absolute tolerance: auto
- Workspace I/O: Load initial states: `psbregul i n i t`

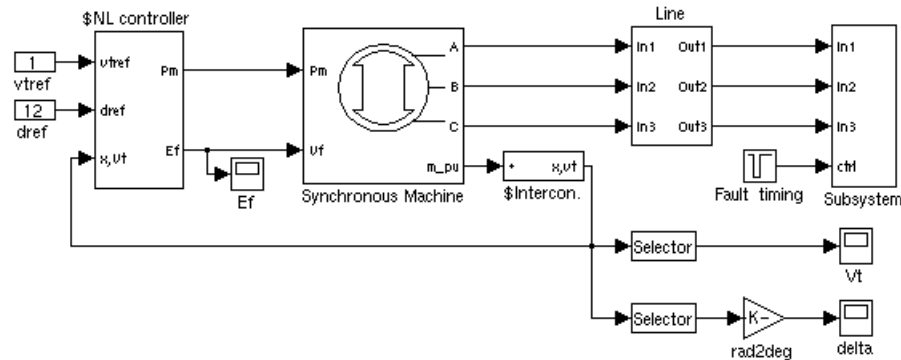


Figure 2-18: Simulink Diagram of Case Study (psbregulator.mdl)

Due to nonlinearities present in this system, computation of initial conditions was not carried out. Instead, a long simulation (10 s) was executed and the final states saved in file `psbregul dat a. mat`. These final states are used as initial

states in this case study. The simulation consequently starts in steady-state. At $t=0.1$ s, the fault is suddenly applied and removed after 100 ms (6 cycles). The post-fault transient is then observed.

The nonlinear controller calls a MATLAB initialization function to compute the gains before the simulation. Although this process has been automated to take into account the parameters in the dialog boxes of the various blocks, it is not recommended that you change any value in any block.

If you decide to change some values, a long simulation must be run and the final states must be in a file called `psbregul i n i t . m a t`. Figure 2-19 shows the response of the generator's terminal voltage, load angle, and the control effort of the regulator. You can observe how the stabilization of V_t is obtained in less than 0.25 seconds with this controller. The load angle takes longer to stabilize, because the time constant of the mechanical part of the system is much larger than the electrical time constants. If you want to compare results with classical regulators, replace the nonlinear controller with the same excitation system and Hydraulic Turbine and Governor block used in the `psbturbi ne` demo. You will notice that the system takes longer to stabilize than in this case study.

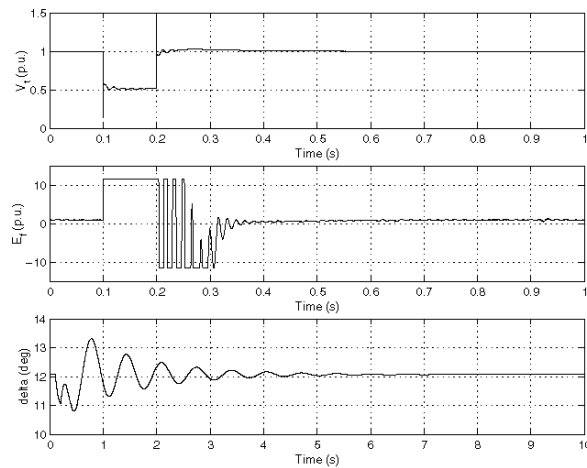


Figure 2-19: Simulation Results Obtained With Case Study

References

- [1] Akhrif O., F.A. Okou, L.A. Dessaint, and R. Champagne, "Application of a Multivariable Feedback Linearization Scheme for Rotor Angle Stability and Voltage Regulation of Power Systems." *IEEE Transactions on Power Systems*, Vol. 14, No. 2, May 1999, pp. 620-628.

Variable-Frequency Induction Motor Drive

This case study presents a variable-frequency AC motor drive in which a pulse width modulated (PWM) inverter is used as a variable-voltage variable-frequency source to drive an induction motor in variable-speed operation.

The drive, including the motor, the power converter, and the speed control system, is modeled by using the Power System Blockset and Simulink blocks. The drive operation is studied for different operating conditions: starting, steady-state, and transients.

The objective of this example is to demonstrate the use of Machine library, and Power Electronics library blocks in combination with Simulink blocks in the simulation of a complex electromechanical system operating at high frequency. The electrical part of the AC motor drive including the PWM inverter is built using the Universal Bridge block. The induction motor is represented by the Asynchronous Machine block, which models both electric and mechanical dynamics. The control system including current and speed regulators is built using Simulink blocks. The interface between electrical and control systems is ensured by blocks of the Measurement library of **powersys**.

Description of the Induction Motor Drive

The induction motor requires a variable-frequency three-phase source for variable-speed operation. This source can be realized by using a power converter system consisting of a rectifier connected to an inverter through a DC link.

Figure 2-20 shows a block diagram of the power circuit of a typical variable-frequency induction motor drive.

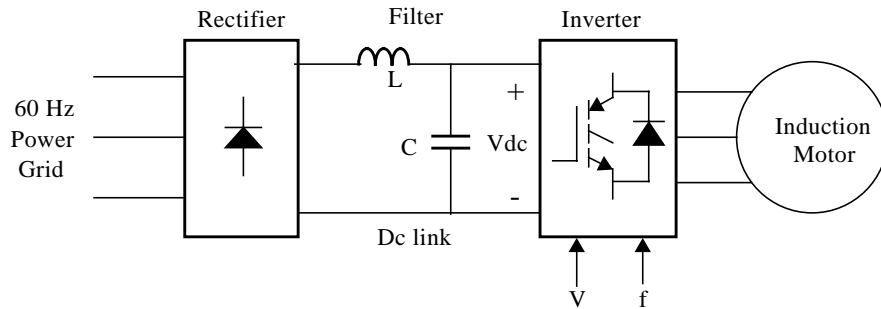


Figure 2-20: Variable-Frequency Induction Motor Drive

The power grid AC voltage is converted into a fixed DC voltage by the rectifier. The harmonics are filtered out by an LC filter to provide a smooth DC voltage, which is then applied to the inverter input.

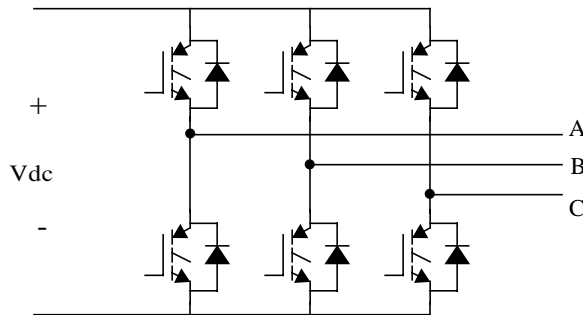


Figure 2-21: Three-Phase IGBT Inverter

The inverter consists essentially of six power switches that can be metal-oxide-semiconductor-field-effect transistors (MOSFET), gate-turn-off thyristors (GTO), or insulated-gate-bipolar transistors (IGBT), depending on the drive power capacity and the inverter switching frequency (Hz). Figure 2-21 shows a simplified diagram of a three-phase IGBT inverter.

The inverter converts the DC link voltage into an adjustable three-phase AC voltage. Different control schemes can be used to control the inverter output voltage and frequency. One of the most utilized schemes is pulse width modulation (PWM) in which three-phase variable sinusoidal voltage

waveforms are obtained by modulating the on and off times of the power switches.

In industrial drive applications, the PWM inverter operates as a three-phase variable-frequency, variable-voltage source with fundamental frequency varying from zero to three times the motor nominal frequency.

In some control schemes where a three-phase, variable-frequency current source is required, current control loops are added to force the motor currents to follow an input reference (usually sinusoidal).

The inverter-fed induction motor drive can be controlled by using various schemes depending on the application, desired performance, and controller design complexity. The most utilized schemes are:

- Stator V/Hz control
- Stator currents and open loop flux control
- Vector control (field-oriented control)
- Direct torque control (DTC)

A Field-Oriented Variable-Speed Induction Motor Drive

In this case study, we consider a variable-speed induction motor drive using field-oriented control. In this control scheme, a d-q coordinates reference frame locked to the rotor flux space vector is used to achieve decoupling between the motor flux and torque. They can be thus separately controlled by stator direct-axis current and quadrature-axis current respectively, like in a DC motor. Figure 2-22 shows a block diagram of a field-oriented induction motor drive.

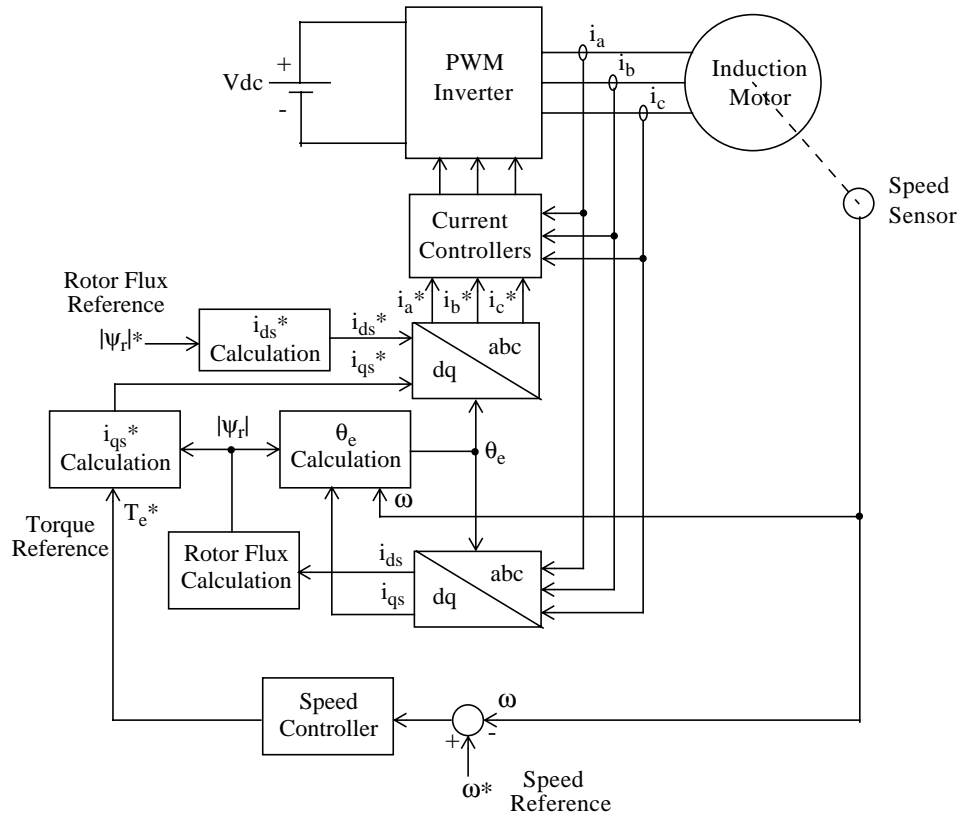


Figure 2-22: Field-Oriented Variable-Frequency Induction Motor Drive

The induction motor is fed by a current-controlled PWM inverter, which operates as a three-phase sinusoidal current source. The motor speed ω is compared to the reference ω^* and the error is processed by the speed controller to produce a torque command T_e^* .

As shown in Figure 2-23, the rotor flux and torque can be separately controlled by the stator direct-axis current i_{ds} and quadrature-axis current i_{qs} , respectively.

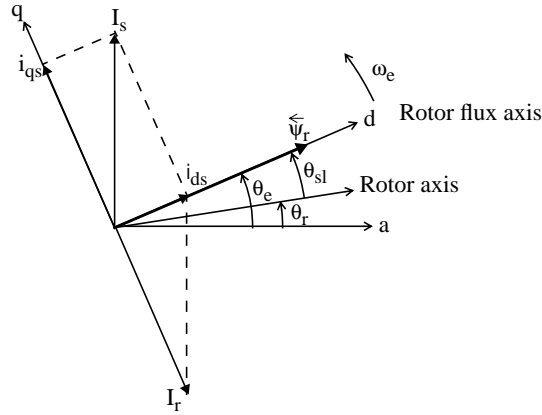


Figure 2-23: Field-Oriented Control Principle

The stator quadrature-axis current reference i_{qs}^* is calculated from torque reference T_e^* as

$$i_{qs}^* = \frac{2}{3} \cdot \frac{2}{p} \cdot \frac{L_r}{L_m} \cdot \frac{T_e^*}{|\psi_r|_{est}}$$

where L_r is the rotor inductance, L_m is the mutual inductance, and $|\psi_r|_{est}$ is the estimated rotor flux linkage given by

$$|\psi_r|_{est} = \frac{L_m i_{ds}}{1 + \tau_r s}$$

where $\tau_r = L_r/R_r$ is the rotor time constant.

The stator direct-axis current reference i_{ds}^* is obtained from rotor flux reference input $|\psi_r|^*$.

$$i_{ds}^* = \frac{|\psi_r|^*}{L_m}$$

The rotor flux position θ_e required for coordinates transformation is generated from the rotor speed ω_m and slip frequency ω_{sl} .

$$\theta_e = \int (\omega_m + \omega_{sl}) dt$$

The slip frequency is calculated from the stator reference current i_{qs}^* and the motor parameters.

$$\omega_{sl} = \frac{L_m}{|\Psi_r|_{est}} \cdot \frac{R_r}{L_r} \cdot i_{qs}^*$$

The i_{qs}^* and i_{ds}^* current references are converted into phase current references i_a^* , i_b^* , i_c^* for the current regulators. The regulators process the measured and reference currents to produce the inverter gating signals.

The role of the speed controller is to keep the motor speed equal to the speed reference input in steady-state and to provide a good dynamic during transients. It can be of proportional-integral (PI) type.

Modeling the Induction Motor Drive

Open the `psbacdrive.mdl` file of the **powerlib** library by typing `psbacdrive` in the MATLAB command window. A circuit diagram titled **psbacdrive** will appear. Before running the example, save this circuit as `case4.mdl` in your working directory so that you can make further modifications without altering the original file.

Figure 2-24 shows the **psbacdrive** diagram in which blocks from the Power System Blockset and Simulink are used to model the induction motor drive.

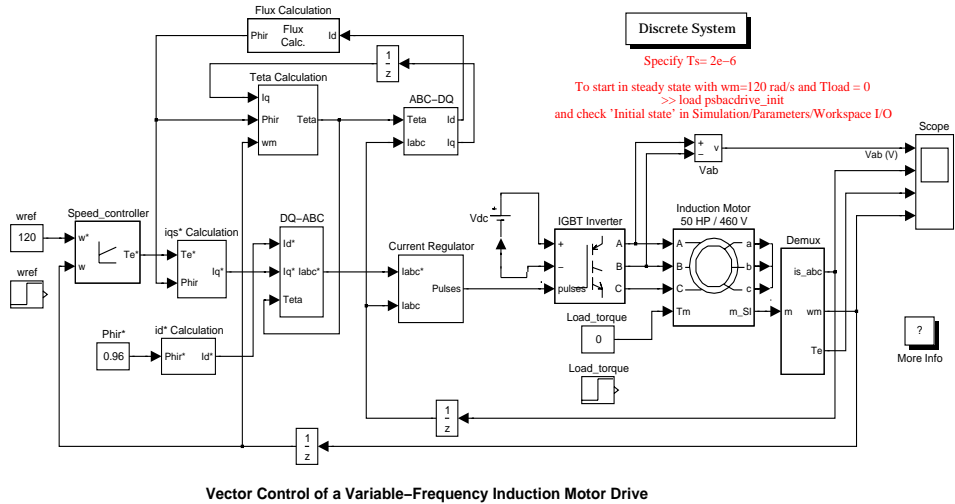
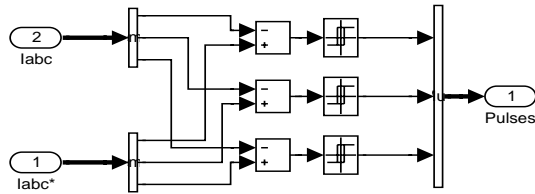


Figure 2-24: Variable-Speed Field-Oriented Induction Motor Drive (psbacdrive.mdl)

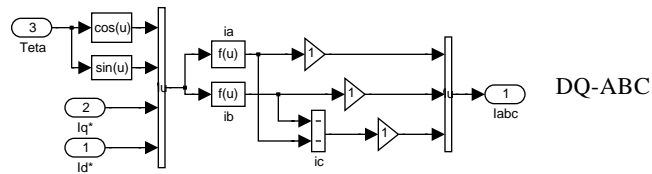
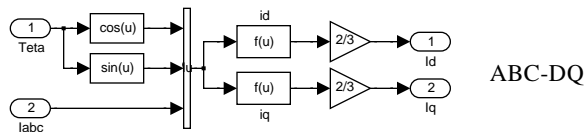
The induction motor is modeled by an Asynchronous Machine block. The motor used in this case study is a 50HP, 460V, 4 poles, 60 Hz motor having the following parameters: $R_s = 0.087 \, \Omega$, $L_{ls} = 0.8 \, \text{mH}$, $L_m = 34.7 \, \text{mH}$, $R_r = 0.228 \, \Omega$, $L_{lr} = 0.8 \, \text{mH}$.

The current-controlled PWM inverter circuit is shown in Figure 2-24. The IGBT inverter is modeled by a Universal Bridge block in which the Power Electronic device and Port configuration options are selected as IGBT/Diode and ABC as output terminals respectively. The DC link input voltage is represented by a 780 V DC voltage source.

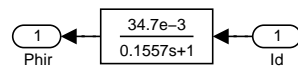
The current regulator, which consists of three hysteresis controllers, is built with Simulink blocks. The motor currents are provided by the measurement output of the Asynchronous Machine block.



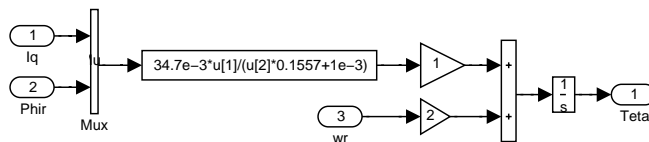
The conversions between abc and dq reference frames are executed by the ABC-DQ and DQ-ABC blocks of Figure 2-24:



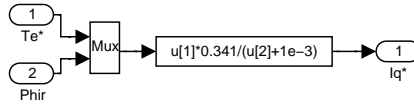
The rotor flux is calculated by the Flux_Calculation block of Figure 2-24:



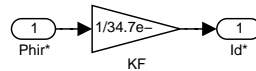
The rotor flux position (θ_e) is calculated by the Teta Calculation block of Figure 2-24:



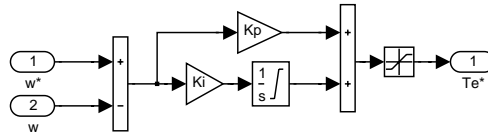
The stator quadrature-axis current reference (i_{qs}^*) is calculated by the i_{qs}^* _Calculation block of Figure 2-24:



The stator direct-axis current reference (i_{ds}^*) is calculated by the i_{ds}^* _Calculation block of Figure 2-24:



The speed controller is of proportional-integral (PI) type and is implemented using Simulink blocks:



Simulating the Induction Motor Drive

Run the simulation by selecting **Start** from the **Simulation** menu in Simulink. The simulation parameters in the **Simulation Parameters** menu can be set as follows:

Simulation time: Start Time: 0, Stop time: 1.5
 Solver option: Type: Variable-step ode23tb (stiff/TR-BDF2)
 Max Step Size: auto
 Initial Step Size: auto
 Relative Tolerance: 1e-3
 Absolute Tolerance: 1e-3

The motor voltage and current waveforms as well as the motor speed and torque are displayed on three axes of the scope connected to the variables V_{ab} , I_s , T_e , and ω .

Once the simulation is complete, return to the MATLAB command window to examine the results with more details by using the plot instruction.

Drive Starting

You can start the drive from a standstill by specifying 0 initial conditions for all state variables in the **powergui** interface and specifying [1, 0, 0, 0, 0, 0, 0, 0] as the initial conditions for the Asynchronous Machine block. In this example, the speed reference is stepped from 0 to 120 rad/s at $t = 0$ and we observe the motor speed, torque, and current.

The transient responses for the starting of the induction motor drive are shown in Figure 2-25.

Note that you can save the final system state vector by previously selecting **Workspace I/O/Save to work space/Final state** in the **Simulation Parameters** window.

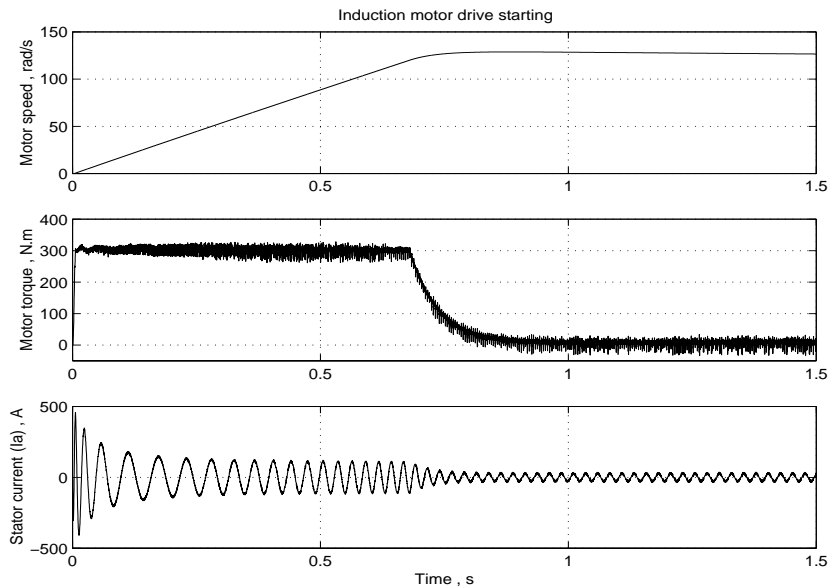


Figure 2-25: Starting of the Induction Motor Drive

Steady-State Voltage and Current Waveforms

When the steady state is attained, you can stop the simulation and plot the voltage and current waveforms using the variables V_{ab} and I_a sent back in the MATLAB workspace by the scope.

Figure 2-26 shows the motor current and voltage waveforms obtained when the load torque is 50 Nm.

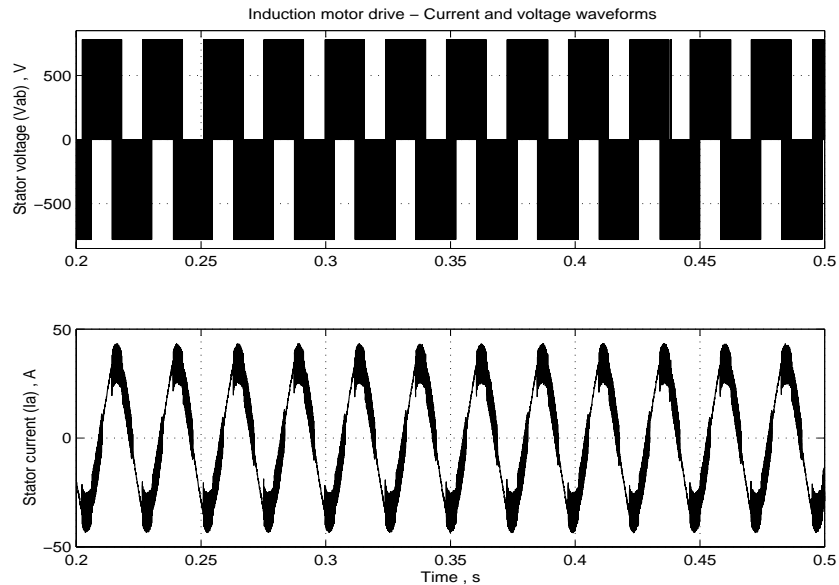


Figure 2-26: Steady-State Motor Current and Voltage Waveforms

Speed Regulation Dynamic Performance

We can study the drive dynamic performance (speed regulation performance versus reference and load torque changes) by applying two changing operating conditions to the drive: a step change in speed reference and a step change in load torque.

Replace the Constant w_{ref} and Load_torque blocks in the `psbacdrive.mdl` diagram by two Simulink step functions with different starting times.

The speed reference steps from 120 rad/s to 160 rad/s at $t = 0.2$ s and the load torque steps from 0 N.m to 200 N.m at $t = 1.4$ s. The final state vector obtained with the previous simulation can be used as the initial condition so that the simulation will start from steady-state. Load the `psbacdrive_init.mat` file, which will create the `xInitial` variable. Select **Workspace I/O/Load from workspace/Initial state** in the **Simulation Parameters** window and restart the simulation.

The obtained response of the induction motor drive to successive changes in speed reference and load torque is shown in Figure 2-27.

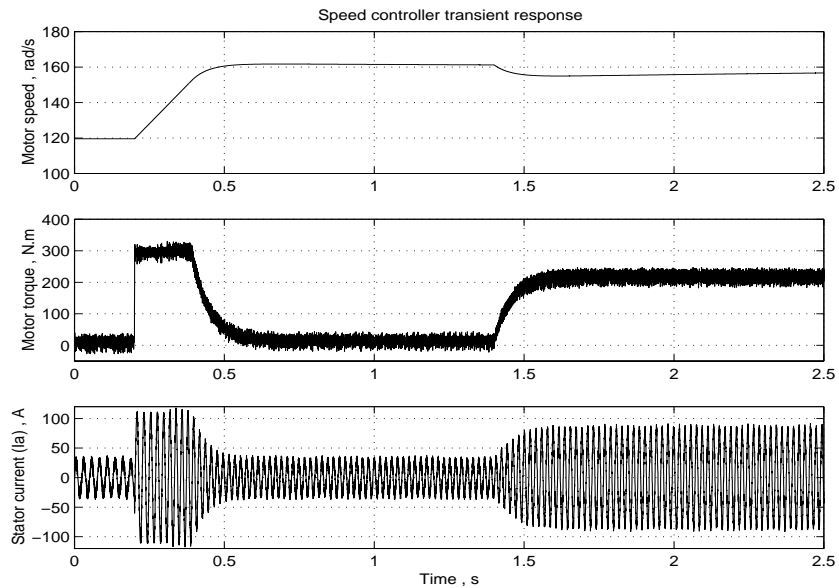


Figure 2-27: Dynamic Performance of the Induction Motor Drive

References

- [1] Leonhard W., *Control of Electrical Drives*. Springer-Verlag, Berlin 1996.
- [2] Murphy J. M. D. and Turnbull F. G., *Power Electronic Control of AC Motors*. Pergamon Press, Oxford 1985.
- [3] Bose B. K., *Power Electronics and AC Drives*. Prentice-Hall. Englewood Cliffs 1986.

HVDC System

The final example described in this section illustrates modeling of a high voltage direct current (HVDC) transmission link [1]. Perturbations are applied in order to examine the system performance [2]. The objectives of this example are to demonstrate the use of the Universal Bridge block, and the Three-Phase Transformer (Three-Windings) block in combination with Simulink blocks in the simulation of a complete pole of a 12-pulse HVDC transmission system. The electrical part representing the AC network is built using three-phase blocks. The Discrete 12-Pulse HVDC control system is a generic control available in the Controls library of **powerlib_extras**.

Description of the HVDC Transmission System

This network is available as a demo entitled Complete 12-Pulse HVDC Transmission System. Open this demo and save the `psbhvdc12pul se.mdl` file in your working directory as `case5` in order to allow further modifications to the original system. This system is shown on Figure 2-28.

A 1000 MW (500 kV, 2kA) DC interconnection is used to transmit power from a 500 kV, 5000 MVA, 60Hz network to a 345 kV, 10 000MVA, 50Hz network. The AC networks are represented by damped L-R equivalents with an angle of 80 degrees at fundamental frequency (60 Hz or 50 Hz) and at the third harmonic.

The rectifier and the inverter are 12-pulse converters using two Universal Bridge blocks connected in series. Open the two converter subsystems to see how they are built. The converters are interconnected through a 300 km distributed parameter line and 0.5 H smoothing reactors. The converter transformers (Wye grounded /Wye/Delta) are modeled with Three-Phase Transformer (Three-Windings). The transformer tap changers are not simulated. The tap position is rather at a fixed position determined by a multiplication factor applied on the primary nominal voltage of the converter transformers (0.90 on rectifier side; 0.96 on inverter side).

From the AC point of view, an HVDC converter acts as a source of harmonic currents. From the DC point of view, it is a source of harmonic voltages. The order n of these characteristic harmonics are related to the pulse number p of the converter configuration: $n = kp \pm 1$ for the AC current, and $n = kp$ for the direct voltage, k being any integer. In our example, $p = 12$, so that injected harmonics on the AC side are: 11, 13, 23, 25, and on the DC side are: 12, 24.

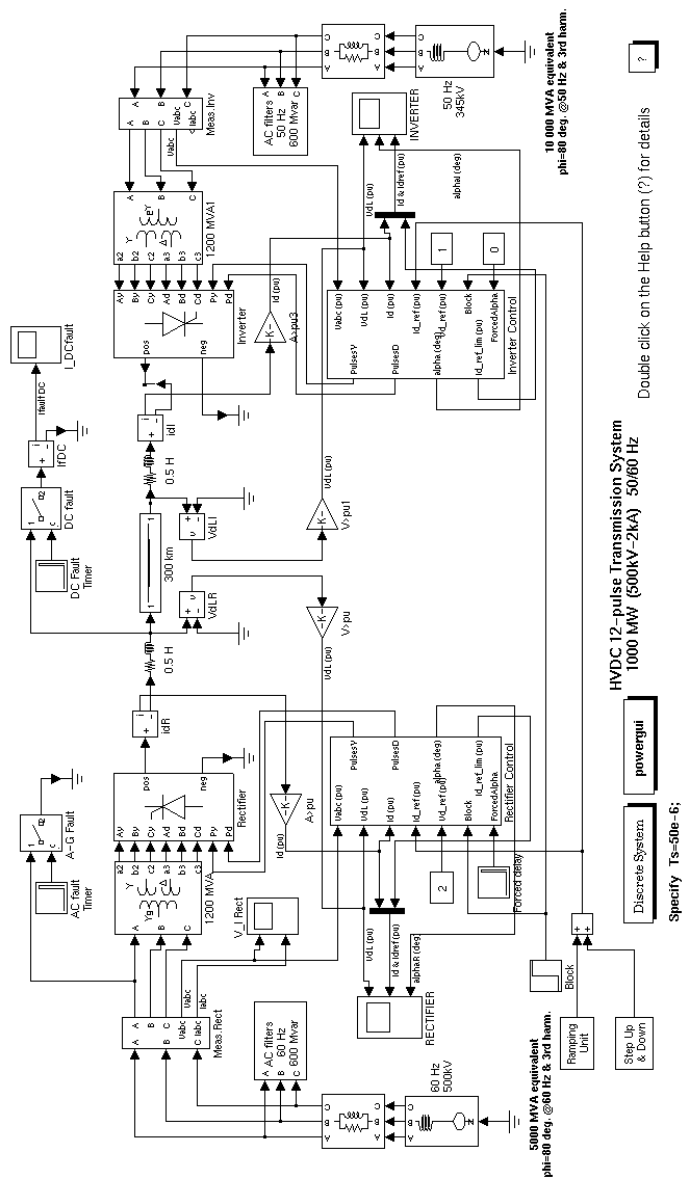


Figure 2-28: HVDC System

AC filters are used to prevent the odd harmonic currents from spreading out on the network. The filters are grouped in two subsystems. These filters also appear as large capacitors at fundamental frequency, thus providing reactive power compensation for the rectifier consumption due to the firing angle α . For $\alpha=30$ degrees, the converter reactive power demand is approximately 60% of the power transmitted at full load. Look under the AC filters subsystem mask to see the high Q (100) tuned filters at the 11th and 13th harmonic and the low Q (3), or damped filter, used to eliminate the higher order harmonics, e.g., 23rd and up. Extra reactive power is also provided by capacitor banks.

Two circuit breakers are used to apply faults on the rectifier AC and DC sides.

The rectifier and inverter control systems use the Discrete 12-pulse HVDC Control block of the Control library of **powerlib_extras**.

The power system and the control system are both discretized with the same sample time T_s .

Define parameter $T_s=50e-6$ in your workspace prior to start simulations.

Frequency Response of the AC and DC Systems

You will now measure the frequency response of the AC systems (rectifier and inverter sides) and of the DC line.

We have already explained in Session 2 in the Tutorial chapter how the Impedance Measurement block allows you to compute the impedance of a linear system from its state-space model. As the thyristor valves of the converters are nonlinear blocks, they will be ignored in the impedance calculation and you will get the impedances with the valves open.

Open the Measurements library and copy three Impedance Measurement blocks in your model and rename them Zrec, Zinv, and ZDC. Connect the two inputs of Zrec and Zinv between phase A and phase B of the AC system on the rectifier and inverter sides. Measuring the impedance between two phases will give two times the positive-sequence impedance. Therefore you must apply a factor of 1/2 on the impedance in order to obtain the correct impedance value. Open the two Impedance Measurement blocks and set the Multiplication factor to 0.5. Finally, connect input 1 of the ZDC block between the DC line terminal and the rectifier smoothing reactor and input 2 to ground. Save your modified circuit as case5Zf.

Now open the **powergui**. In the **Tools** menu select **Impedance vs Frequency Measurement**. A new window opens showing the three Zmeter block names. Fill in the Frequency range by typing 10:2:1500. Select the lin scale to display the Z magnitude and lin scale for the frequency axis. Check the **Save data to workspace** button and enter Zcase5 as the variable name that will contain the impedance vs frequency. Click on the **Display** button.

When the calculation is finished, three graphic windows appear with the magnitude and phase as function of frequency measured by the three Impedance Measurement blocks. If you check in your workspace, you should have a variable named Zcase5. It is a four-column matrix containing frequency in column 1 and the three complex impedances in columns 2, 3, and 4 with the same the same order as in the window displaying the block names.

The magnitudes of the three impedances as function of frequency are shown in Figure 2-29.

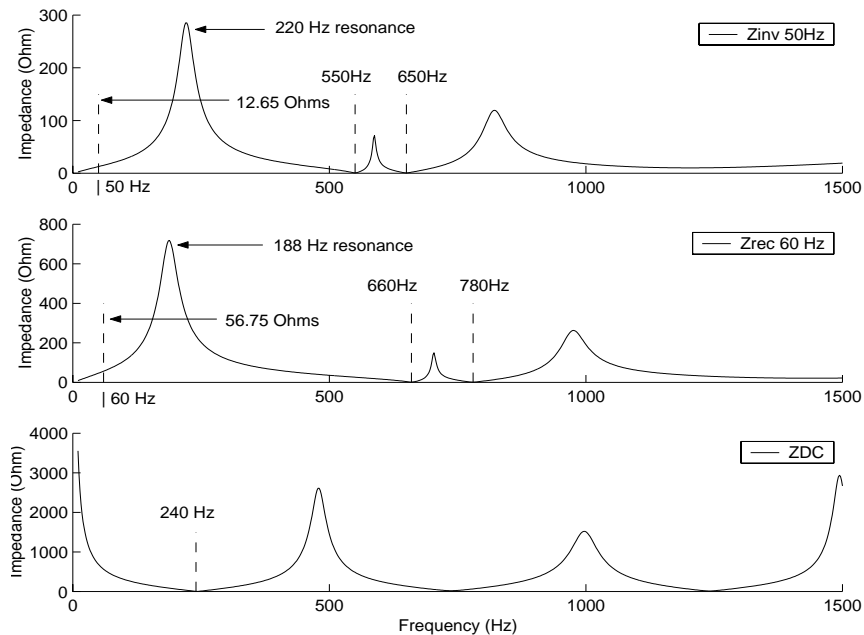


Figure 2-29: Positive-Sequence Impedances of the Two AC Networks and of the DC Line

Notice the two minimum impedances on the Z magnitudes of the AC systems. These series resonances are created by the 11th and 13th harmonic filters. They occur at 660 Hz and 780 Hz on the 60 Hz system. Notice also that the addition of 600 Mvar capacitive filters on the inductive systems creates resonances (around 188 Hz on the rectifier side and 220 Hz on the inverter side). Zoom in on the impedance magnitude in the 60 Hz region. You should find a magnitude of 56.75 Ω for the 60 Hz system, corresponding to an effective short circuit level of $500^2/56.75 = 4405$ MVA on the rectifier side (5000 MVA - 600 Mvar of filters).

For the DC line, note the series resonance at 240 Hz, which corresponds to the main mode likely to be excited on the DC side, under large disturbances.

Description of the Control System

The control systems of the rectifier and of the inverter use the same 12-pulse HVDC Control block from the Controls library of **powerlib_extras**. The block can operate either in rectifier or inverter mode. Use the Look Under Mask to see how this block is built.

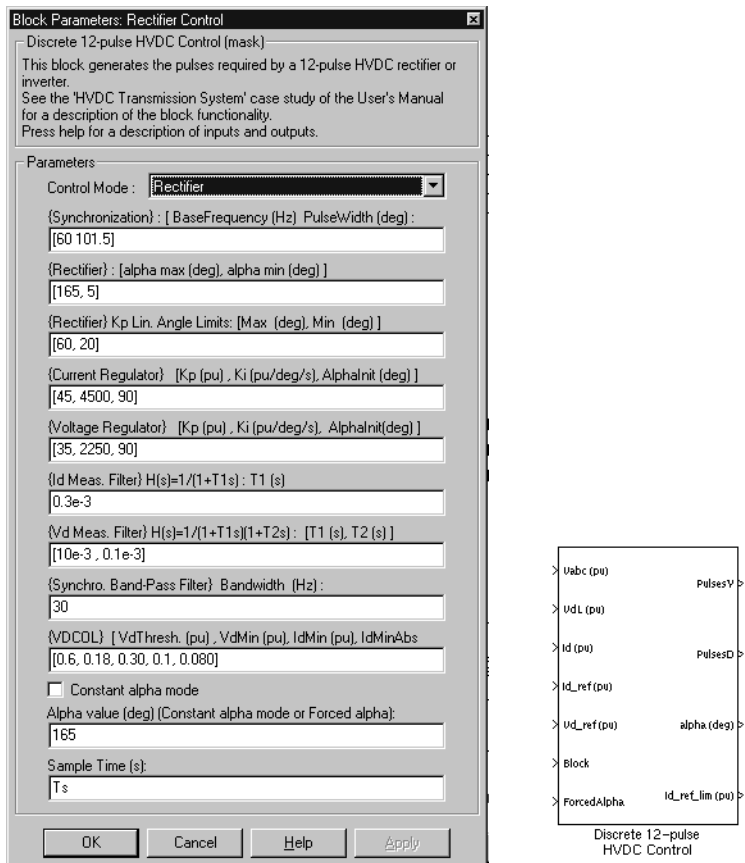


Figure 2-30: 12-Pulse HVDC Control Block and Its Dialog Box for Inverter Mode

Inputs and Outputs

Input 1 (Vabc) is a vectorized signal of the three line-to-ground voltages measured at the primary of the converter transformer. These three voltages are used to synchronize the pulse generation on the line voltages. Inputs 2 and 3 are the DC line voltage (VdL) and current (Id). Note that the measured DC currents (IdR and IdI in A) and DC voltages (VdLR and VdLI in V) are scaled to p.u (1pu current = 2 kA; 1pu voltage = 500 kV) before they are used in the controllers.

Inputs 4 and 5 (V_{d_ref} and I_{d_ref}) are the V_d and I_d reference values in p.u. The V_{dL} and I_d inputs are filtered before being processed by the regulators. A first order filter is used on I_d input and a second order filter is used on V_{dL} input. The filter parameters are shown in the dialog box of Figure 2-30.

Input 6 (Block) accepts a logical signal (0-1) used to block the converter when Block=1. Input 7 is also a logical signal that can be used for protection purposes. If this signal is high (1), the firing angle will be forced at the value defined in the block dialog box.

The first two block outputs (PulseY and PulseD) contain the vectorized signals of the six pulses to be sent to each of the six-pulse converter connected to the wye and delta windings of the converter transformer. The third output (alpha) is the firing delay angle in degrees ordered by the regulator. The fourth output ($I_{d_ref_lim}$) is the actual reference current value (value of I_{d_ref} limited by the VDCOL function as explained below).

Synchronization System

The Discrete 12-Pulse HVDC Control block uses the primary voltages (input 1) to synchronize and generate the pulses according to V_{d_ref} and I_{d_ref} set points (inputs 4 and 5). The synchronizing voltages are measured at the primary side of the converter transformer because the waveforms are less distorted. The firing command pulse generator is synchronized to the fundamental frequency of the AC source. At the zero-crossings of the commutating voltages (AB, BC, CA), a ramp is reset. A firing pulse, is generated whenever the ramp value becomes equal to the desired delay angle provided by the regulator. In order to improve the commutating voltages used by the pulse generator, the primary voltages (V_{abc}) are filtered by a low Q second-order band-pass filter centered at the fundamental system frequency. The base system frequency and the filter bandwidth are defined in the block dialog box.

Steady-State V-I Characteristic

The Discrete 12-Pulse HVDC Control block implements the steady-state characteristic shown on Figure 2-31.

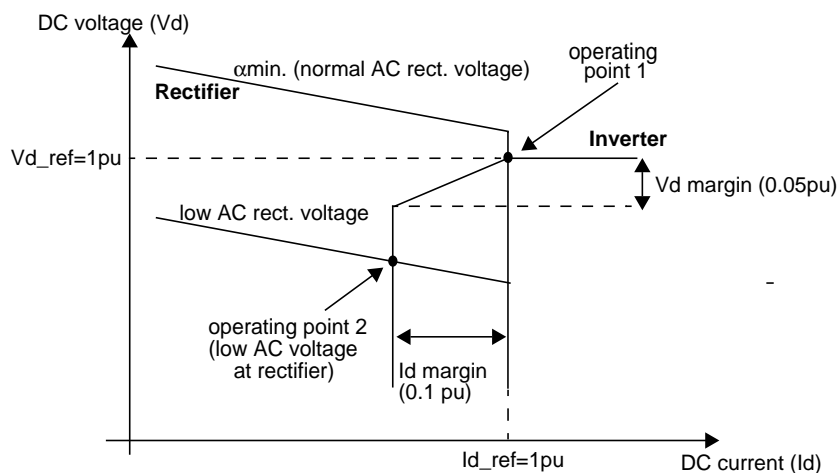


Figure 2-31: Rectifier and Inverter Steady-State Characteristics and VDCOL Function

In normal operation, the rectifier controls the current at the I_{d_ref} reference value whereas the inverter controls the voltage at the V_{d_ref} reference value. The I_{d_margin} and V_{d_margin} parameters are defined in the inverter dialog box. They are set respectively at 0.1 p.u. and 0.05 p.u. The system normally operates at point 1 as shown on the figure. However, during a severe contingency producing a voltage drop on the AC network 1 feeding the rectifier, the operating point will move to point 2. The rectifier will therefore be forced to α minimum mode and the inverter will be in current control mode.

Note In industrial controllers, the α angle at the inverter is normally limited in order to keep a minimum γ angle, where:

γ = extinction angle = $180^\circ - \alpha - \mu$

μ = commutation or overlap angle

The γ control required to avoid commutation failures is not implemented in this version of the HVDC control.

VDCOL Function

Another important control function is implemented to change the reference current according to the value of the DC voltage. This control named Voltage Dependent Current Order Limiter (VDCOL) automatically reduces the reference current (I_{d_ref}) set point when V_{dL} decreases (as for example, during a DC line fault or a severe AC fault). Reducing the I_{d_ref} reference currents also reduces the reactive power demand on the AC network, helping to recover from fault. The VDCOL parameters of the Discrete 12-Pulse HVDC Control block dialog box are explained in Figure 2-32.

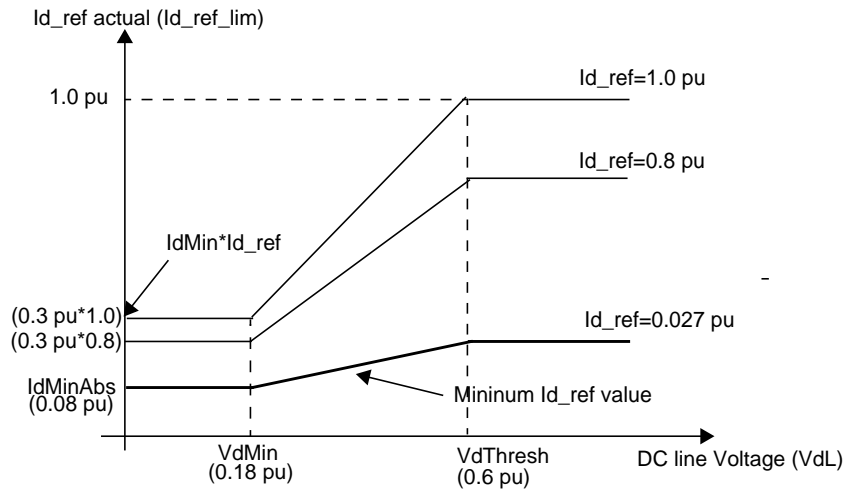


Figure 2-32: VDCOL Characteristic; $I_{d_ref} = f(V_{dL})$

The I_{d_ref} value starts to decrease when the V_d line voltage falls below a threshold value $V_{dThresh}$ (0.6 pu). The actual reference current used by the controllers is available at the fourth controller output named $I_{d_ref_lim}$. $I_{dMinAbs}$ is the absolute minimum I_{d_ref} value set at 0.08 p.u. When the DC line voltage falls below the $V_{dThresh}$ value, the VDCOL reduces instantaneously I_{d_ref} . However, when the DC voltage recovers, VDCOL limits the I_{d_ref} rise time with a time constant defined by parameter T_{up} (80 ms in our example).

Current and Voltage Regulators

The rectifier and the inverter controls both have a voltage and a current regulator operating in parallel calculating firing angles α_v and α_i . The effective α angle is the minimum value of α_v and α_i . This angle is available at the third block output named `alpha (deg)`. Both regulators are of the proportional and integral type (PI). They should have high enough gains for low frequencies (<10 Hz) to maintain the current or voltage response equal to the reference current (`Id_ref_lim`) or reference voltage (`Vd_ref`), as long as α is within the minimum and maximum limits ($5^\circ < \alpha < 165^\circ$ for rectifier, $92^\circ < \alpha < 165^\circ$ for inverter). The regulator gains K_p and K_i are adjusted during small perturbations in the current reference. The following gains are used:

- Current regulator: $K_p=92^\circ/\text{pu}$ $K_i=4500^\circ/\text{pu/s}$
- Voltage regulator: $K_p=35^\circ/\text{pu}$ $K_i=2250^\circ/\text{pu/s}$

Another particularity of the regulator is the linearization of the proportional gain. As the V_d voltage generated by the rectifier and the inverter is proportional to $\cos(\alpha)$, the ΔV_d variation due to a $\Delta \alpha$ change is proportional to $\sin(\alpha)$. With a constant K_p value, the effective gain would therefore be proportional to $\sin(\alpha)$. In order to keep a constant proportional gain, independent of the α value, the gain is linearized by multiplying K_p constant with $1/\sin(\alpha)$. This linearization is applied for a range of α defined by two limits specified in the dialog box (third line).

System Start-Up and Steady-State

Notice that the system is discretized, using sample time T_s (you should already have $T_s=50\text{e-}6$ defined in your workspace).

The system is programmed to start and reach a steady state. Then, a step is applied on the reference current to observe the dynamic response of the regulators.

Start the simulation and observe the signals on the rectifier and inverter scopes. The waveforms are reproduced on Figure 2-33.

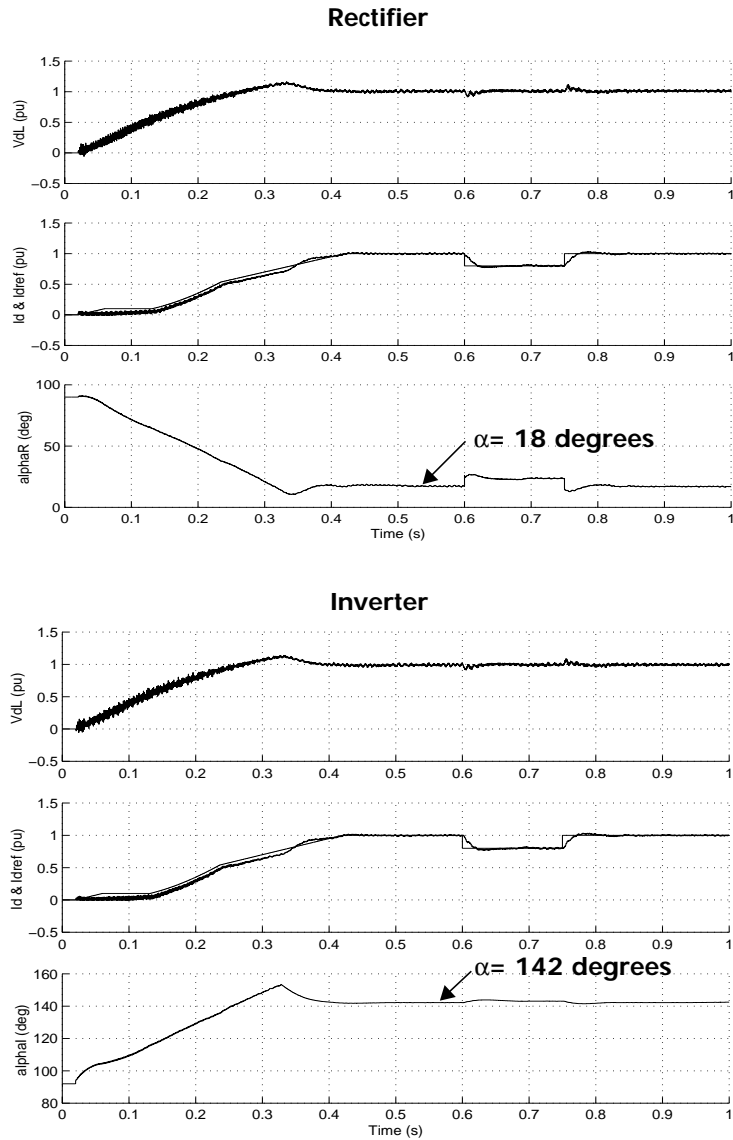


Figure 2-33: Start Up of the DC System and Step Applied on the Reference Current

The reference current follows a ramp from zero to 1 p.u. (2kA) in 0.4 s. Observe that the DC current starts to build up at $t=20$ ms, time at which the controller and the pulse generators are deblocked. The DC current and voltages start from zero and reach steady-state in approximately 0.5 s. The rectifier controls the current and the inverter controls the voltage. Trace 1 of both rectifier and inverter scopes shows the DC line voltage (1pu=500 kV). Trace 2 shows the reference current and the measured I_d current (1pu=2 kA). Once steady-state is attained, the α firing angles are 18 degrees and 142 degrees respectively on the rectifier and inverter side.

Then, at $t=0.6$ s a step is applied on the reference current to observe the dynamic response of the regulators.

The main equations governing the steady-state operation of the DC system are given here so that you can check the theoretical values against the simulation results.

The following expression relates the mean direct voltage V_d of a 12-pulse bridge to the direct current I_d and firing angle α

$$V_d = 2 \times (V_{do} \times \cos(\alpha) - R_c \times I_d)$$

where V_{do} is the ideal no-load direct voltage for a 6-pulse bridge

$$V_{do} = (3\sqrt{2}/\pi) \times V_c$$

V_c is the line to line rms commutating voltage that is dependent on the AC system voltage and the transformer ratio.

R_c is the equivalent commutating resistance

$$R_c = (3/\pi) \times X_c$$

X_c is the commutating reactance or transformer reactance referred to the valve side.

The following rectifier parameters were used in the simulation.

The V_c voltage must take into account the effective value of the voltage on the 500kV bus and the transformer ratio. If you look at the waveforms displayed on the V_{I_Rect} scope, you will find 0.96 p.u. If you open the rectifier transformer dialog box, you will find a multiplication factor of 0.91 applied on the primary nominal voltage. The voltage applied to the inverter is therefore boosted by a factor 1/0.91.

$$V_c = 0.96 \cdot 200 \text{ kV} / 0.90 = 213.3 \text{ kV};$$

$$I_d = 2 \text{ kA}$$

$$\alpha = 18^\circ;$$

$$X_c = 0.24 \text{ p.u. based on 1200 MVA and 222.2 kV} = 9.874 \Omega$$

Therefore:

$$V_{do} = (3\sqrt{2}/\pi) \times 213.2 = 287.9 \text{ kV}$$

$$R_c = (3/\pi) \times 9.874 = 9.429 \Omega$$

$$V_d = 2 \times (288, 1 \text{ kV} \times \cos(18^\circ) - 9.429 \times 2) = 510 \text{ kV}$$

This theoretical voltage corresponds well with the expected rectifier voltage calculated from the inverter voltage and the voltage drop in the DC line:

$$V_d = V_{dL_{inverter}} + (R_{DCline} + R_{inductance}) \times I_d$$

$$V_d = 500 \text{ kV} + (4.5 \Omega + 1 \Omega) \times 2 = 511 \text{ kV}$$

The μ commutation or overlap angle can be also calculated. Its theoretical value depends on α , the DC current I_d and the commutation reactance X_c .

$$\mu = \arccos \left[\cos(\alpha) - \frac{X_c \cdot I_d \cdot \sqrt{2}}{V_c} \right] - \alpha$$

$$\mu = \arccos \left[\cos(18^\circ) - \frac{9.874 \cdot 2 \cdot \sqrt{2}}{213.3} \right] - 18^\circ = 16.9^\circ$$

Now verify the commutation angle by plotting the currents in two valves, showing for example current extinction in valve 1 and current build up in valve 3 of one six-pulse bridge of the rectifier.

Open the rectifier subsystem. Then, open the upper bridge dialog box and the select **All voltages and currents** for the Measurement parameter. Now, copy the Multimeter block from the Measurements library into your case5 circuit. Double-click on the Multimeter block. A window showing all the bridge voltages and currents appears. Select the following signals

- uSw1: Rectifier/Universal Bridge
- iSw1 : Rectifier/Universal Bridge
- iSw3 : Rectifier/Universal Bridge

The number of signals (3) is displayed in the multimeter icon. Using a Demux block, send the three multimeter output signals to a two-trace scope. (Trace 1: uSw1 Trace 2: i Sw1 and i Sw3). Restart the simulation. The waveforms illustrating two cycles are shown on Figure 2-34. The measured commutation angle is 14 steps of $50\mu\text{s}$ or 15.1° of a 60 Hz period. Knowing that the resolution with a $50\mu\text{s}$ time step is 1.1° , this angle compares reasonably well with the theoretical value.

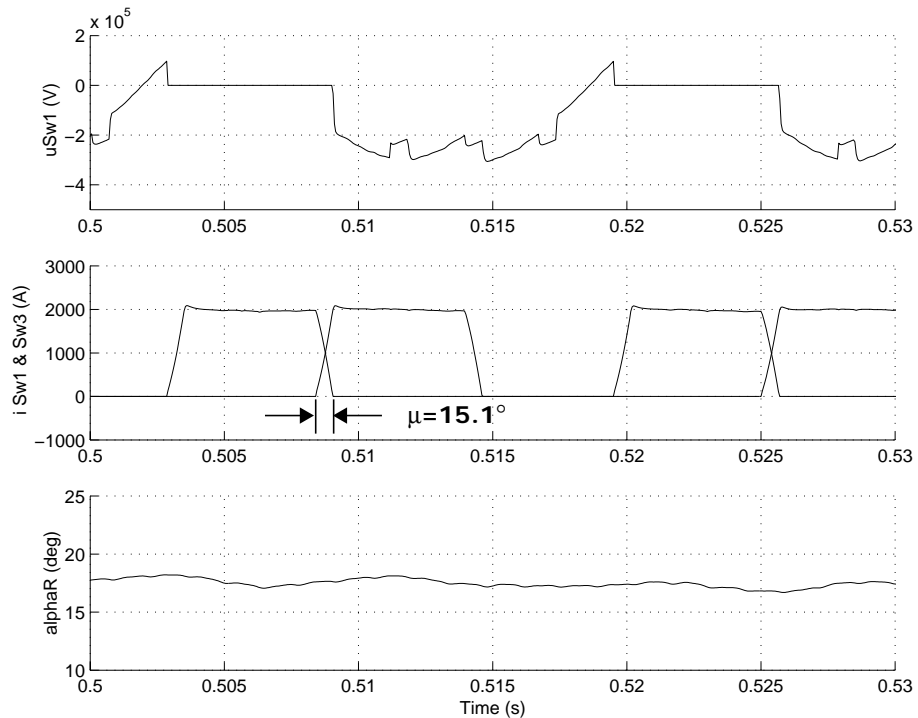


Figure 2-34: Valve Voltage and Currents (Commutation from Valve 1 to Valve 3)

Response to a Step of Reference Current

At $t=0.6$ s, a 0.2 p.u. step is applied on the reference current (decrease from 1 p.u. to 0.8 pu). At $t=0.75$ s, another step is applied to set the reference back to 1

p.u. Observe the response of the current regulator. It stabilizes in approximately 0.1 s.

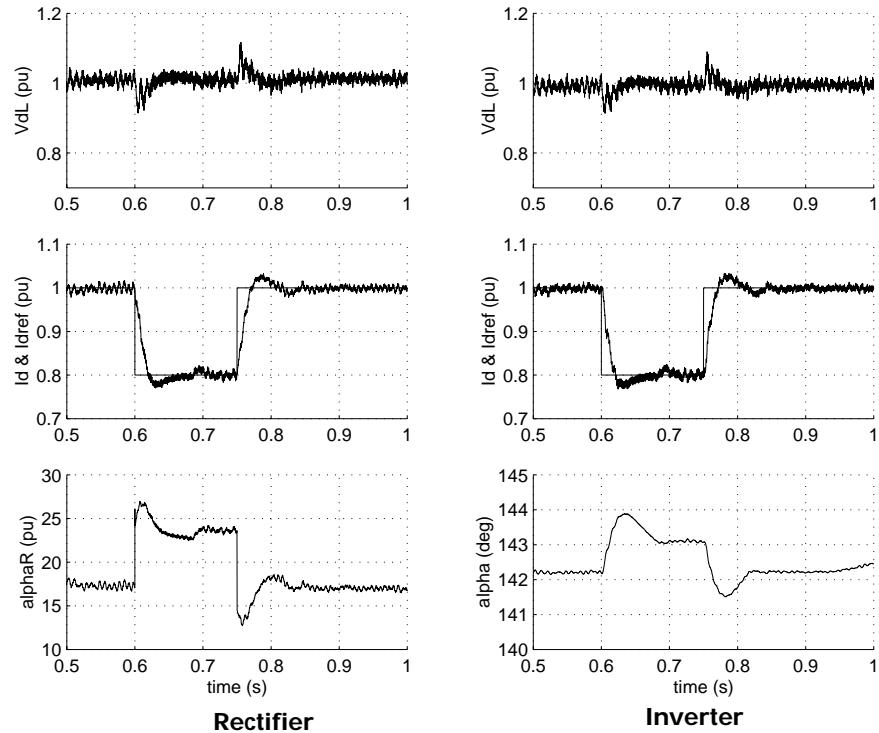


Figure 2-35: Response to a 0.2 p.u. Step of the Reference Current

DC line Fault

Disconnect the Step Up & Down block in order to eliminate the step disturbance applied on the reference current. In the DC Fault Timer and Forced Delay blocks of the psbhvdc12pul se circuit, change the multiplication factor of 100 in the Transition Times to 1, so that a fault is now applied at t=0.6s. Open the I_DCfault scope to observe the fault current. Restart the simulation.

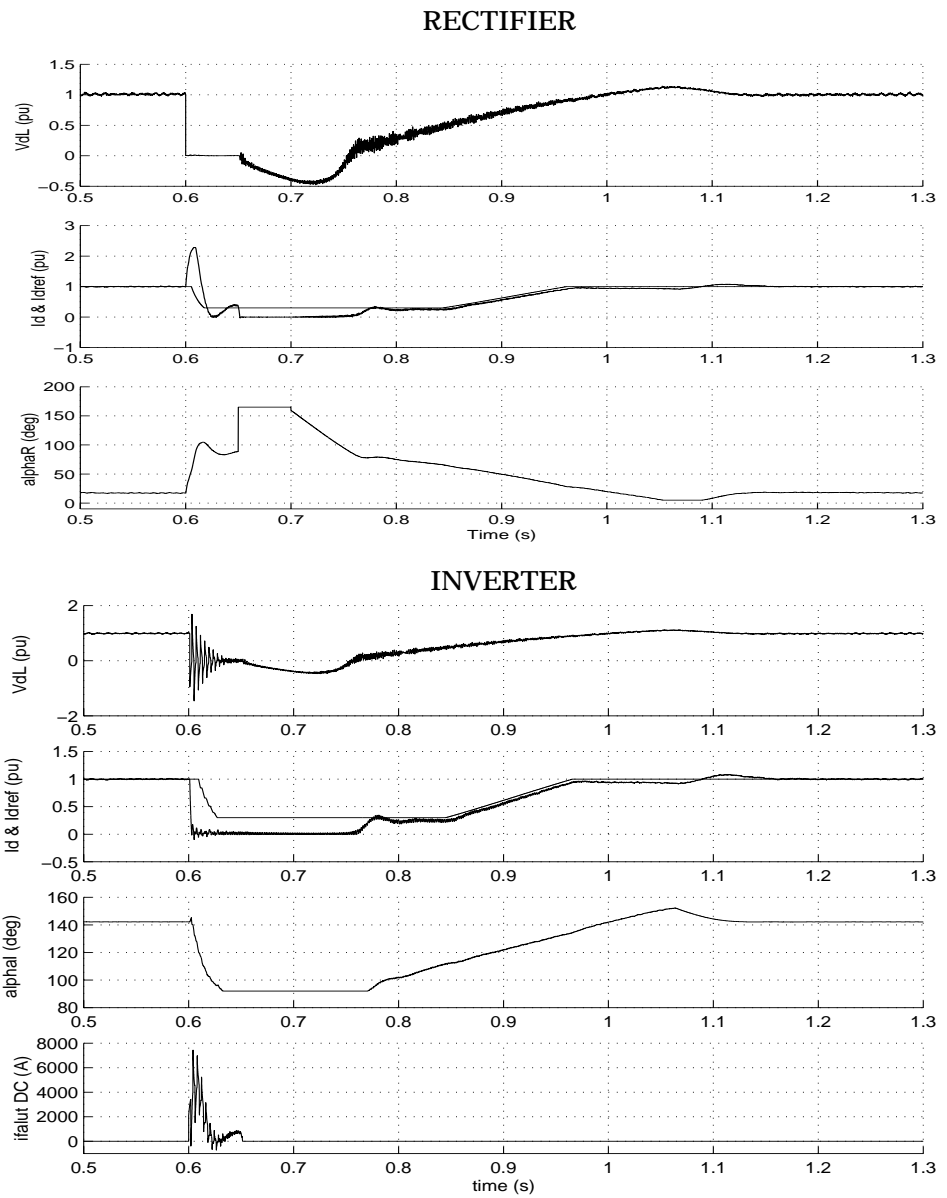


Figure 2-36: DC Line Fault on the Rectifier Side

At fault application ($t=0.6$ s), the DC current quickly increases to 2.3 p.u. and the DC voltage falls to zero at the rectifier. This DC voltages drop is seen by the Voltage Dependant Current Order Limiter (VDCOL), which reduces the reference current to 0.3 p.u. at the rectifier. A DC current still continues to circulate in the fault. Then, at $t=0.65$ s, the rectifier α firing angle is forced to 165 degrees when the signal applied to the ForcedAlpha input goes high. This signal would normally be provided by the protection system not simulated here. The rectifier now operates in inverter mode. The DC line voltage becomes negative and the energy stored in the line is returned to the AC network, causing rapid extinction of the fault current at its next zero-crossing. Then, α is released at $t=0.7$ s and the normal DC voltage and current recover in approximately 0.5 s

AC Line-to-Ground Fault at the Rectifier

You will now modify the timer blocks in order to apply a line-to-ground fault. In the DC Fault Timer and Forced Delay blocks of psbhvdc12pulse, change the multiplication factor of 1 in the Transition Times to 100, so that the DC fault is now eliminated. In the AC Fault Timer block, change the multiplication factor in the transition times to 1, so that a 6 cycle line-to-ground fault is now applied at the rectifier. Restart the simulation.

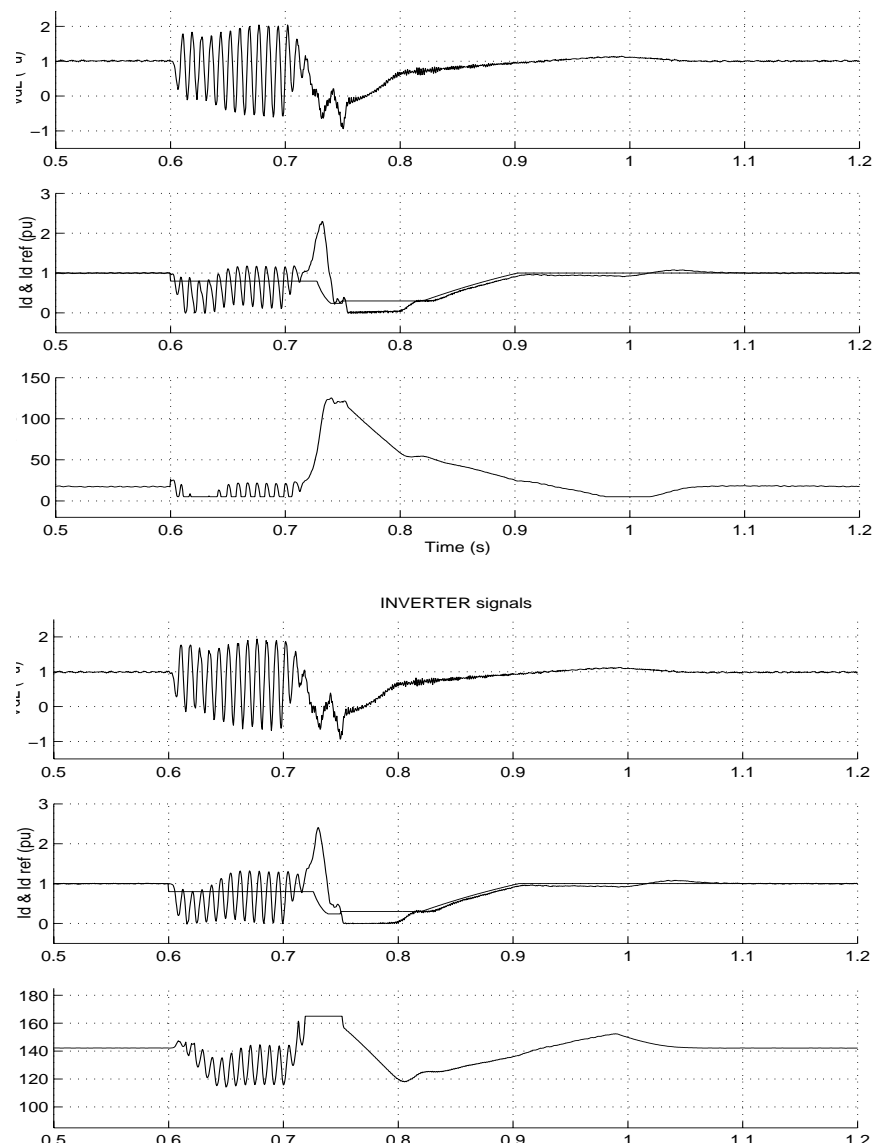


Figure 2-37: Rectifier and Inverter Signals for an AC Line Fault on the Rectifier Side

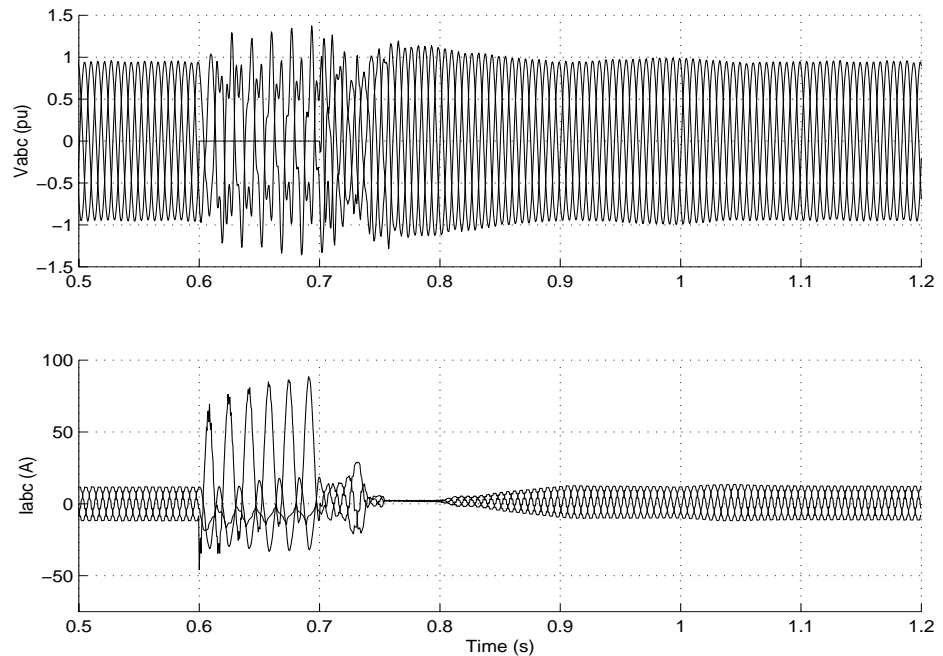


Figure 2-38: Voltages and Currents on the 60Hz Side for an AC Line Fault on the Rectifier Side

Notice the 120 Hz oscillations in the DC voltage and currents during the fault. When the fault is cleared at $t=0.7$ s, the VDCOL operates and reduces the reference current to 0.3 p.u. The system recovers in approximately 0.4 s after fault clearing.

References

- [1] Arrilaga J., *High Voltage Direct Current Transmission*, IEEE Power Engineering Series 6, Peter Peregrinus Ltd. 1983.
- [2] *Electromagnetic Transients Program (EMTP), Workbook IV (TACS)*, EL-4651, Volume 4, Electric Power Research Institute, 1989.

Advanced Topics

How the Power System Blockset Works	3-2
Which Integration Method Should Be Used - Continuous or Discrete?	3-5
Simulating with Continuous Integration Algorithms	3-6
Choosing the Right Integration Algorithm	3-6
Simulation of Switches and Power Electronic Devices	3-6
Simulating Discretized Electrical Systems	3-10
How to Increase Simulation Speed	3-13
The Nonlinear Model Library	3-14
The Continuous Library	3-15
The Discrete Library	3-16
The Switch Current Source Library	3-16
Limitations with the Nonlinear Models	3-16
Modifying the Nonlinear Models of the Powerlib_models Library	3-16
Creating Your Own Library of Models	3-18
Changing Your Circuit Parameters	3-19
Example of MATLAB Script Performing a Parametric Study	3-19

How the Power System Blockset Works

Once you have built your circuit with the electrical blocks of **powerlib**, you can start the simulation just like any other Simulink model. Each 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.

The `power2sys` function is part of the process. It gets the state-space model and builds the Simulink model of your circuit. `power2sys` can also be called from the command line to obtain the state-space model of the linear part of the circuit. When called by the initialization process, `power2sys` performs the following four steps as shown on Figure 3-1:

- Sorts all the blocks contained in the system into two categories: the Simulink blocks and the Power System Blockset blocks. Then it gets the block parameters and evaluates the network topology. The Power System Blockset blocks are separated into linear and nonlinear blocks, and each electrical node is automatically given a node number.
- Once the network topology has been obtained, the state-space model of the linear part of the circuit is computed by the `ci rc2ss` function. 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.
- Builds the Simulink model of your circuit and stores it inside one of the measurement blocks. This means that you need at least one measurement block (Current Measurement block, Voltage Measurement block, or Multimeter block) in your model. The connections between the equivalent circuit and measurements blocks are performed by invisible links using the Goto and From blocks.

The Simulink model uses a Simulink State-Space block or a special S-function block to model the linear part of the circuit. Pre-defined Simulink models are used to simulate nonlinear elements. These models can be found in the **Powerlib_models** library available with the Power System Blockset. Simulink source blocks connected at the input of the state-space block are used to simulate the electrical sources blocks.

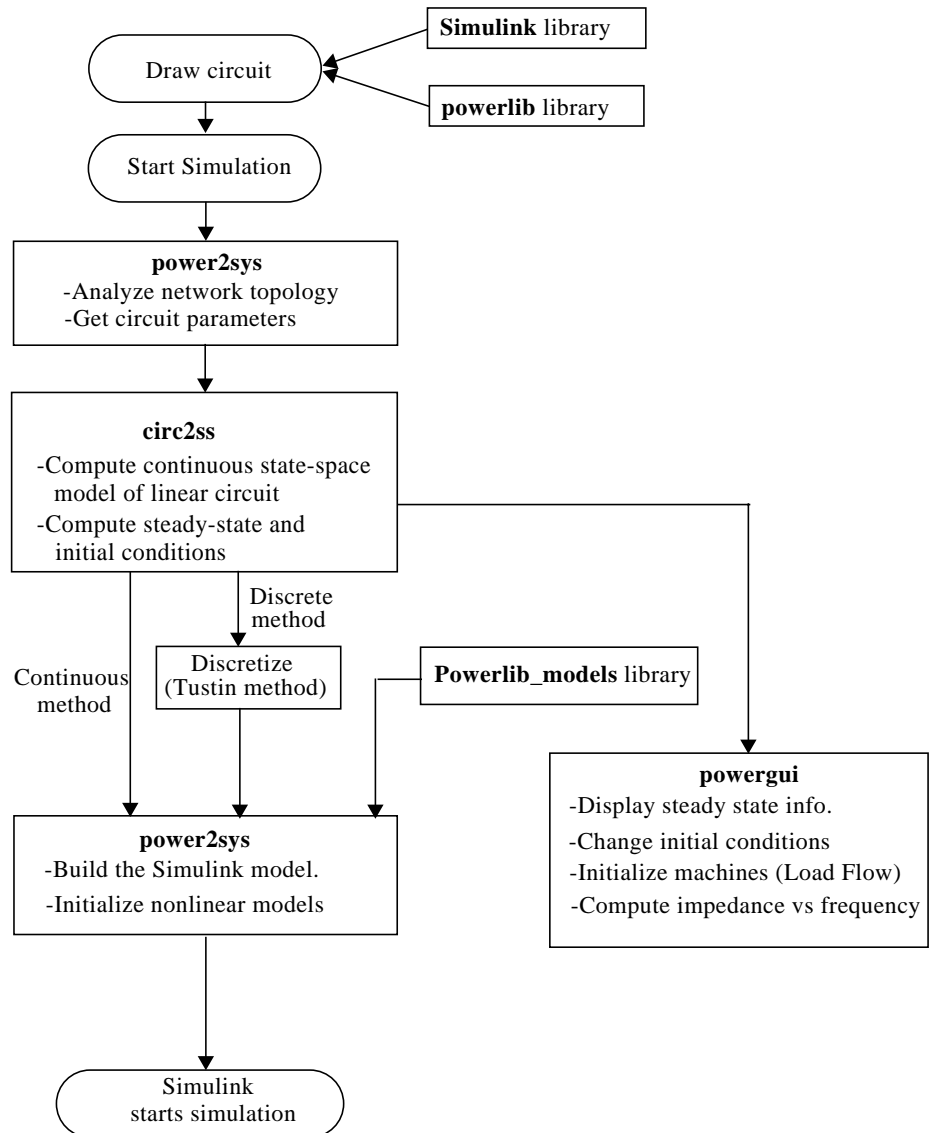
**Figure 3-1: Power System Blockset Flowchart**

Figure 3-2 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.

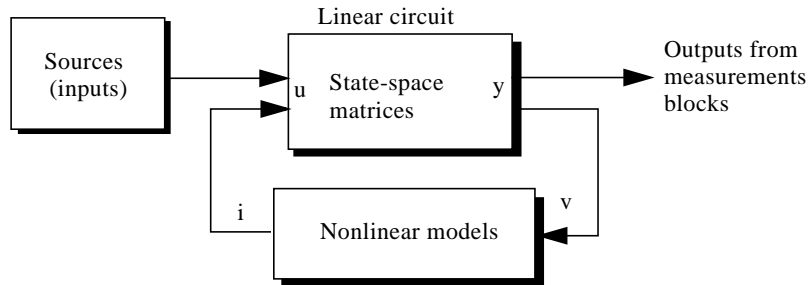


Figure 3-2: Interconnection of Linear Circuit and Nonlinear Models

Once **power2sys** has completed the initialization process, Simulink starts the simulation and you can observe waveforms on scopes connected at the outputs of your measurement blocks.

If you stop the simulation and drag a copy of the Powergui block into your circuit window, you will have access to the steady-state values of inputs, outputs, and state variables displayed as phasors. You can also use the interface 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. Finally, if Impedance Measurement blocks are connected in your circuit, the Powergui block allows you to specify the desired frequency range, visualize impedance curves, and store results into your workspace.

Which Integration Method Should Be Used - Continuous or Discrete?

One important feature of the Power System Blockset that has been introduced with version 2, is its ability to simulate electrical systems either with continuous variable time-step integration algorithms or with a fixed time step using a discretized system. For small size systems, the continuous method is usually more accurate. Variable time step algorithms are also faster because the number of steps will be less than with a fixed time step method giving comparable accuracy. When using line-commutated power electronics, the variable-step, event-sensitive algorithms will detect the zero crossings of currents in diodes and thyristors with a high accuracy so that you won't 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” systems? Although the distinction is not clear, you can consider small size a system that contains less than 30 electrical states and less than 6 electronic switches. Circuit breakers do not affect the speed too much, because unlike power electronic switches, which are commutated at every cycle, these devices are operated only a couple of times during a test.

Simulating with Continuous Integration Algorithms

Choosing the Right Integration Algorithm

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

Fastest simulation speed is usually achieved with ode23tb or ode15s with default parameters:

```
Solver : ode23tb or ode15s
Relative tolerance = 1e-3
Absolute tolerance = auto
Maximum step size = auto; Initial step size = auto.
Initial step size = auto
Maximum order (for ode15s) = 5
```

Normally, you can choose auto for the absolute tolerance and the maximum step size. In some occasions you may have to limit the maximum step size and the absolute tolerance. Selecting a too small 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 and capacitor voltages). For example, if you work with high power converters where expected voltage and currents are thousands of Volts and Amperes, an absolute tolerance of 1e-1 or even 1.0 should be sufficient. If you are working with low power circuits involving maximum values of 100 V and 10 A, you should use a smaller absolute tolerance like 1e-3 or 1e-2.

Simulation of 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 Circuit Breaker and the Ideal Switch because these elements don't have

internal inductance. It is also applied for Diode and Thyristor with $R_{on} > 0$ and $L_{on} = 0$, and for the Universal Bridge with forced commutated devices.

- If the switch contains a series inductance (Diode and Thyristor with $L_{on} > 0$, 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 on Figure 3-2.

You have therefore the choice to simulate diodes and thyristors with or without L_{on} internal inductance. In most applications, it is not necessary to specify an inductance L_{on} . However, for circuit topologies resulting in zero commutation or overlap angle you will have to specify a switch inductance L_{on} in order to help commutation.

Let us consider for example the circuit shown on Figure 3-3. This circuit is available in the `psbrectifierideal.mdl` file. The thyristor bridge is fed from an infinite source (zero impedance) so that the commutation between thyristors is quasi instantaneous.

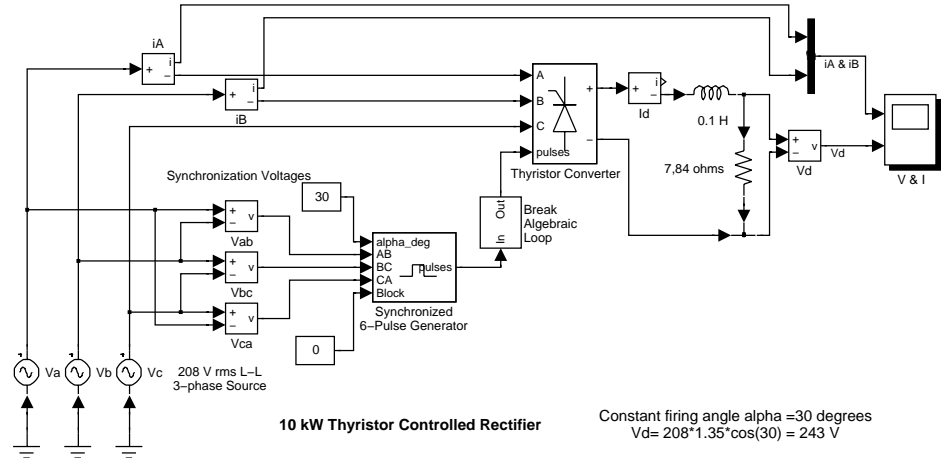


Figure 3-3: Three-Phase Thyristor Rectifier on Infinite Source

If you simulate this circuit without internal thyristor inductances ($L_{on} = 0$), you will observe high current spikes flowing in the three lines. This happens because during commutation two thyristors connected to the same positive or

negative terminal of the bridge are in conduction for a short period of time, applying a line-to-line short circuit on the source (see Figure 3-4). During commutation, the current is limited only by the internal resistance of thyristors (with $R_{on} = 0.01 \text{ ohm}$, the current reaches $208 \cdot \sqrt{2} \cdot \sin(30) / (2 \cdot 0.01) = 7.35 \text{ kA}$ or 245 times the normal DC current of 30 A). These short circuits can be avoided by using a small $L_{on} = 1 \text{ } \mu\text{H}$ in the thyristor model. If you repeat the simulation, you will get “ideal” square current waveforms with a peak value of 30 A.

If you zoom on the line current during a commutation, you will discover that the commutation is not instantaneous. The commutation time depends on the L_{on} value and the DC current.

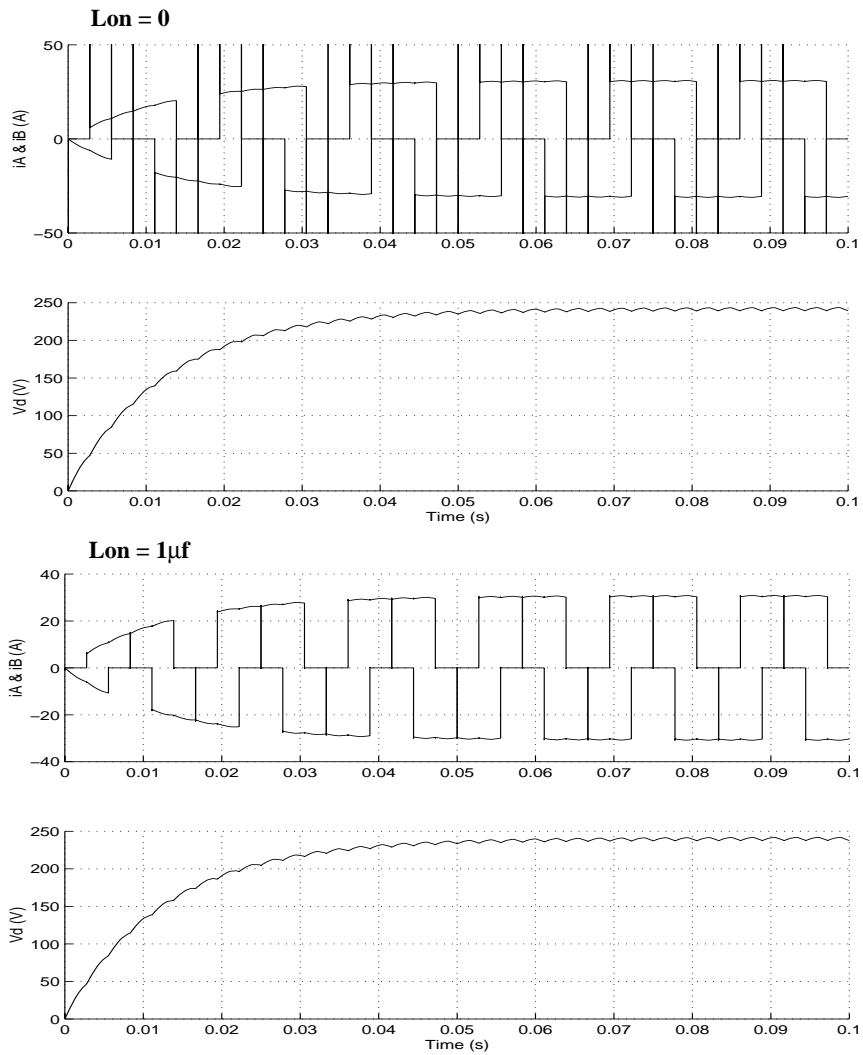


Figure 3-4: Source Currents and DC Load Voltage with $L_{on} = 0$ and $L_{on} = 1\mu H$

Simulating Discretized Electrical Systems

Discretization is performed by dragging the Discrete System block in your system. The sample time is specified in the block dialog box. Next time that you start the simulation, the electrical system will be discretized using the Tustin method, which is equivalent to a fixed-step trapezoidal integration. In order to avoid algebraic loops, the electrical machines are discretized using the Forward Euler method.

When you discretize your system, the precision of the simulation will be controlled by the time step. If you use too large a sample time, the precision may 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 will 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 will have to reduce the time step. These devices, the insulated-gate-bipolar-transistor (IGBT), the field-effect-transistor (FET), and the gate-turn-off thyristor (GTO) are usually operating at high switching frequencies. For example, simulating a pulse-width modulated (PWM) inverter operating at 8 kHz would require a 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 will be improved by using a discrete control system.

Limitations with Nonlinear Models

- 1 Discretization of individual forced-commutated electronic devices is not permitted.

Discretization of circuits containing forced-commutated power electronic devices (IGBT GTO or FET) is permitted only with the Universal Bridge block. Discretization of circuits containing individual forced commutated devices is not allowed. For example, an attempt to discretize the buck DC chopper circuit saved in the `psbbuckconv.mdl` demonstration file will produce a warning message as shown on Figure 3-5.

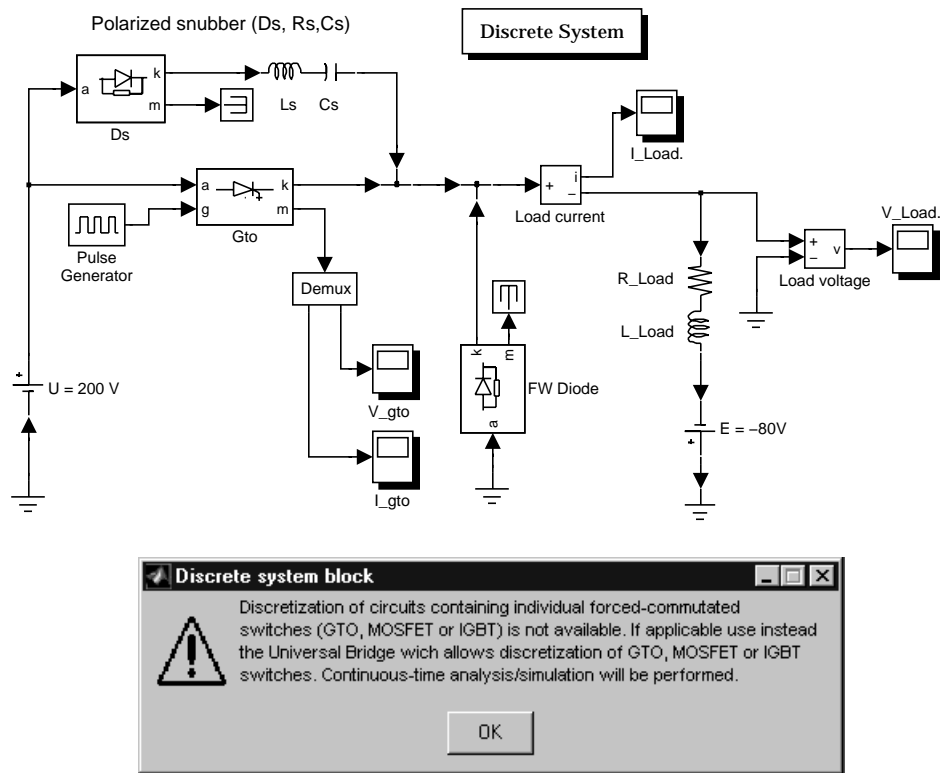


Figure 3-5: A Circuit Containing Individual Forced Commutated Electronic Switches Cannot be Discretized

In this circuit, opening of the GTO will force quasi instantaneous conduction of the free wheeling diode. If the circuit was discretized, the diode would be fired with one step delay, and the inductive current chopping would produce large overvoltages. However, for conventional converter topologies as in the case of the Universal Bridge, the switch interactions are known in advance. For example, in a six-switch IGBT/Diode inverter (Figure 3-6), opening of IGBT1 causes instantaneous conduction of diode D2 in the same arm. As the circuit topology is predetermined, it is possible to force firing of the diode in the same step that the IGBT opens. If you prefer to use individual IGBT and Diode blocks to simulate a complete inverter, you should use a continuous method.

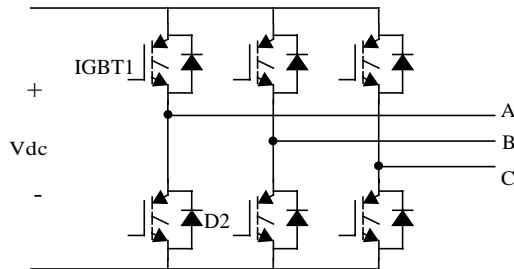


Figure 3-6: IGBT Inverter Simulated by the Universal Bridge

2 Minimal load is required at machine terminals.

When using electrical machines in discrete systems, you may have to use a small “parasitic” resistive load, connected at the machine terminals in order to avoid numerical oscillations. Large sample times request 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.

3 Lon=0 is used for diodes and thyristors in discrete circuits.

Diodes and thyristors used in a discretized circuit must have a zero internal inductance. If you discretize a circuit containing diodes or thyristors with $\text{Lon} > 0$, the Power System Blockset will prompt you a warning, indicating that Lon will be set to zero.

How to Increase Simulation Speed

Once the proper method (continuous or discrete), solver type, and parameters have been selected, there are still some ways of optimizing the simulation speed:

- Discretize 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 may 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/Parameters/Workspace IO** menu. 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 will also help in reducing the simulation time.

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 normally don't need to work with the **Powerlib_models** library. However, you may have to look inside the models or modify them for particular applications. You can access that library by typing `powerlib_models` in the MATLAB command window.

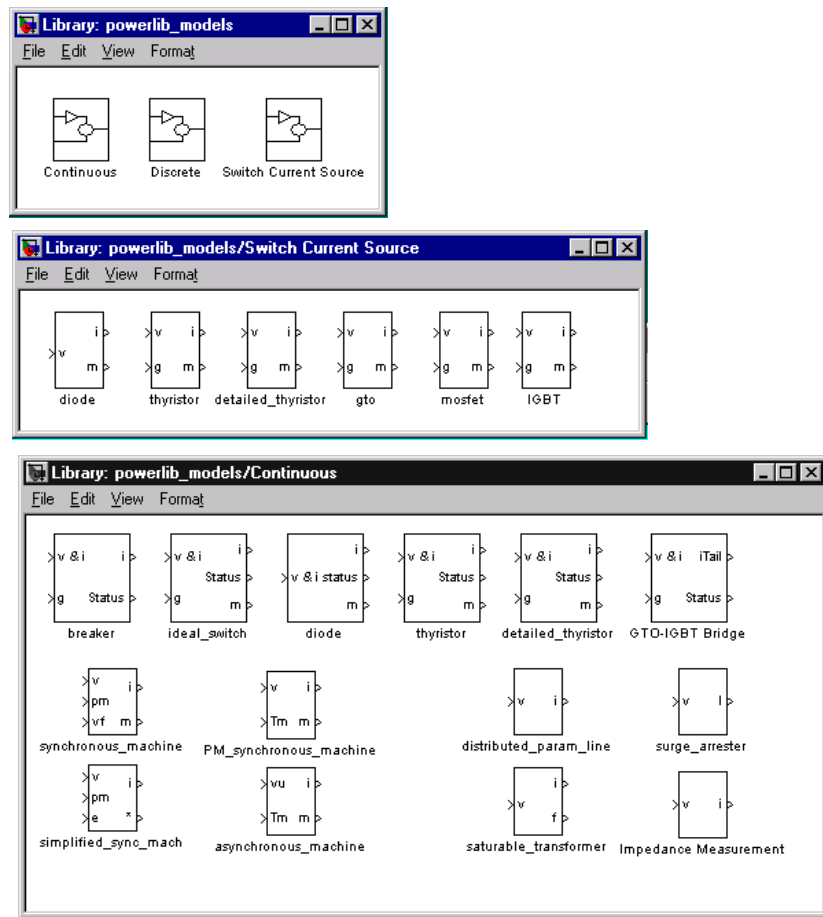


Figure 3-7: The Nonlinear Model Library (powerlib_models)

The Continuous Library

The Continuous library contains two types of blocks:

- 1 Current sources simulating continuous machine models, surge arrester, saturable transformer, and distributed parameter lines
- 2 Switching logics used for purely resistive power electronic devices (Breaker, Diode, Thyristor, and Universal Bridge of forced-commutated devices)

Nonlinear Blocks Simulated by Current Sources

These blocks 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. 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 *m* measurement output vector.

For example, the Asynchronous Machine model is stored in the block named `asynchronous_machine`. The model uses as inputs a vector of four voltages: two rotor voltages (V_{abR} and V_{bcR}) and two stator voltages (V_{abS} and V_{bcS}). It returns a vector of four currents: two rotor currents (I_{aR} and I_{bR}) and two stator currents (I_{aS} and I_{bS}). The model also returns a measurement output vector of 20 signals. When the Asynchronous Machine block is used from `powerlib` 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.

Logics for Switches and Power Electronic Devices

For switches and power electronic devices, the blocks contain only the logic returning the status (open or closed) of the switch. The switch status is passed to an S-function, which recomputes the state-space model of the linear circuit each time that a switch status is changed. The *m* output is a vector returning the switch current and voltage. The *i* output returns the tail current of forced commutated devices such as IGBTs and GTOs. All the switch logics are vectorized. It means that a single model is used by **power2sys** to simulate all the devices having the same type.

The Discrete Library

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

The Switch Current Source Library

This library contains models of power electronic devices, which are simulated by a current source external to the linear circuit. These devices are the diode and the thyristor with $L_{on} > 0$, and the three forced commutated devices: Gate-turn-off thyristor (GTO), metal-oxide-semiconductor-field effect transistor (MOSFET), and the insulated-gate-bipolar transistor (IGBT). All these models are continuous and contain an internal inductance, allowing to handle fast transitions of forced commutated converters. 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 with 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, for example, you feed a machine through an inductive source, **power2sys** will prompt you with an error message. This can be avoided by connecting large resistances in parallel with the source inductances or across the machine terminals.

For the Breaker block and power electronic switches, a series RC snubber circuit is included in the model. You won't have any problem if you keep these snubber circuits in service. The snubber can be changed to a single resistance by setting $C_s = \text{Inf}$, or to a single capacitor by setting $R_s = 0$. To eliminate the snubber, specify $R_s = \text{Inf}$ or $C_s = 0$.

Modifying the Nonlinear Models of the Powerlib_models Library

This version of the Power System Blockset does not allow you to add a block icon in the **powerlib** library and its corresponding model in the **Powerlib_models** library. However, you can temporarily modify the nonlinear models provided with the Power System Blockset. To use your own **Powerlib_models** library, you must first copy the `powerlib_model s.mdl` file

into your working directory or in any other directory. If you are using a directory different from the current directory, you must specify this new directory in the MATLAB search path in front of the standard blockset directory.

Then you can customize this new **Powerlib_models** library as long as you don't change the names of the blocks, the number of inputs and outputs, and the number of parameters in their dialog box. The next time that you run the simulation, the modifications will take place in your circuit.

Creating Your Own Library of Models

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

In Chapter 2, “Tutorial,” we explained 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 Session 7 of the Tutorial chapter.

The **powerlib_extras** library provided with the blockset gives examples of masked blocks that have been used to create a three-phase library. Open this library and note how the masked blocks have been created.

You can even make your block icons change dynamically according to the parameter values. For example, open the dialog box of the 3-phase RLC Parallel Load block provided in the Three-Phase library of **powerlib_extras** and notice how its icon changes when the active and reactive powers are successively set to zero.

Changing Your Circuit Parameters

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

As for the Simulink blocks, all the **powerlib** 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

Let us 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 would like to find the highest overvoltage and for which inductance value 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 matrix format with `V1` variable name.

You may write a MATLAB M-file that will loop on the 10 inductance values and display 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,
        i_max=i;
        V1_max=max(abs(V1));
```

```
        end
    end

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

Block Reference

Block Reference Page Headings	4-2
The Power System Block Libraries	4-3

Block Reference Page Description

Blocks appear in alphabetical order and contain the following information:

- The block name, icon, and block library that contains the block
- The purpose of the block
- A description of the block's use
- The block dialog box and parameters
- Additional information, as it applies to the block:
 - Inputs and outputs - A description of the inputs and outputs of the block
 - Assumptions and limitations to the block's use
- An example using the block
- A See Also of related blocks

The Power System Block Libraries

The Power System Blockset's main library **powerlib** organizes its blocks into libraries according to their behavior. The **powerlib** window displays the block library icons and names:

- The Electrical Sources library contains blocks that generate electric signals.
- The Elements library contains linear and nonlinear network elements.
- The Power Electronics library contains power electronics devices.
- The Machines library contains machinery models.
- The Connectors library contains blocks that can be used to interconnect blocks in various situations.
- The Measurements library contains blocks for the current and voltage measurements.
- The Extras library contains three-phase blocks and specialized measurements and control blocks. This library can also be opened by typing `powerlib_extras` in the MATLAB window. These blocks are not documented in the block reference section.
- The Demos library contains useful demos and case studies.

The **powerlib** window also contains the following blocks:

- The Powergui block that opens a graphical user interface for the steady-state analysis of electrical circuits.
- The Discrete System block that is used to discretize your electrical models.

The Simulink models of the nonlinear blocks of **powerlib** are stored in a library named **Powerlib_models**. These Simulink models are used by the Power System Blockset to build the equivalent Simulink model of your circuit. See the Advanced Topics chapter for a description of the **Powerlib_models** library.

Table 4-1: Electrical Sources Library Blocks

Block Name	Purpose
AC Current Source	Implement a sinusoidal current source
AC Voltage Source	Implement a sinusoidal voltage source
Controlled Current Source	Implement a controlled current source
Controlled Voltage Source	Implement a controlled voltage source
DC Voltage Source	Implement a DC voltage source

Table 4-2: Elements Library Blocks

Block Name	Purpose
Breaker	Implement a circuit breaker opening at current zero crossing
Distributed Parameter Line	Implement a N-phases distributed parameter line model with lumped losses
Linear Transformer	Implement a two- or three-windings linear transformer
Mutual Inductance	Implement a magnetic coupling between two or three windings
Parallel RLC Branch	Implement a parallel RLC branch
Parallel RLC Load	Implement a linear parallel RLC load
PI Section Line	Implement a single phase transmission line with lumped parameters
Saturable Transformer	Implement a two- or three-windings Saturable Transformer
Series RLC Branch	Implement a series RLC branch
Series RLC Load	Implement a linear series RLC load
Surge Arrester	Implement a metal-oxide surge arrester

Table 4-2: Elements Library Blocks (Continued)

Block Name	Purpose
Three-Phase Transformer (Two Windings)	Implement a three-phase transformer with two windings
Three-Phase Transformer (Three Windings)	Implement a three-phase transformer with three windings

Table 4-3: Power Electronics Library Blocks

Block Name	Purpose
Diode	Implement a diode model
GTO	Implement a gate-turn-off (GTO) thyristor model
Ideal Switch	Implement an ideal switch model
IGBT	Implement an insulated-gate-bipolar-transformer (IGBT) model
MOSFET	Implement a metal-oxide-semiconductor-field-effect-transistor (MOSFET) model
Thyristor	Implement a thyristor model

Table 4-4: Machines Library Blocks

Block Name	Purpose
Asynchronous Machine	Model the dynamics of a three-phase asynchronous machine
DC Machine	Model a separately excited DC machine.
Excitation System	Provide an excitation system for the synchronous machine and regulate its terminal voltage in generating mode
Hydraulic Turbine and Governor	Model a hydraulic turbine and a PID governor system
Permanent Magnet Synchronous Machine	Model the dynamics of a three-phase permanent magnet synchronous machine with sinusoidal flux distribution

Table 4-4: Machines Library Blocks (Continued)

Block Name	Purpose
Simplified Synchronous Machine	Model the dynamics of a simplified three-phase synchronous machine
Steam Turbine and Governor	Implements a steam turbine and governor system
Synchronous Machine	Model the dynamics of a three-phase salient-pole synchronous machine

Table 4-5: Connectors Library Blocks

Block Name	Purpose
Bus Bar	Implement a labeled network node
Ground	Provide a connection to the ground
Neutral	Implement a local common node in the circuit

Table 4-6: Measurements Library Blocks

Block Name	Purpose
Current Measurement	Measure a current in a circuit
Impedance Measurement	Measure the impedance in a circuit as a function of the frequency
Multimeter	Measure voltage and current in Power System Blockset blocks
Voltage Measurement	Measure a voltage in a circuit

Purpose Implement a sinusoidal current source.

Library Electrical Sources

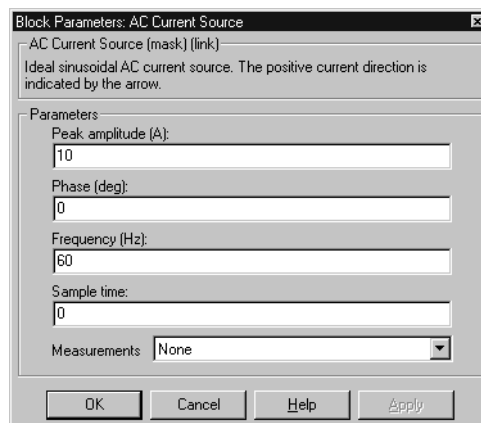
Description The AC Current Source block implements an ideal AC current source. The positive current direction is indicated by the arrow in the block icon. The generated current I is described by the following relationship:



$$I = \text{Amplitude} \times \sin(2\pi \times \text{Frequency} \times t + \text{Phase} \times \pi/180)$$

Negative values are allowed for Amplitude and Phase. A zero frequency specifies a DC current source. Negative Frequency is not allowed, otherwise Simulink signals an error, and the block will display a question mark in the block icon. You can modify the three source parameters at any time during the simulation.

Dialog Box and Parameters



Peak amplitude

The peak amplitude of the generated current source, in amperes (A).

Phase

The phase in degrees.

Frequency

The source frequency in hertz (Hz).

AC Current Source

Sample time

The sample period in seconds (s). The default is 0, corresponding to a continuous source.

Measurements

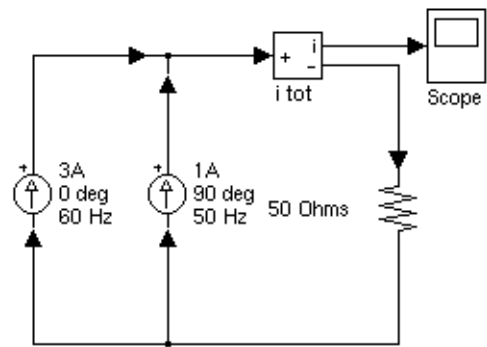
Select **Current** to measure the current flowing through the AC Current Source block.

Place a Multimeter block in your model to display the selected measurements during the simulation. In the **Available Measurement** listbox of the Multimeter block, the measurement will be identified by a label followed by the block name.

Measurement	Label
Current	i Src:

Example

Parallel connection of two AC Current Source blocks is used to sum two sinusoidal currents in a resistor.



This circuit is available in the `psbaccurrent.mdl` file.

See Also

Controlled Current Source, Multimeter

Purpose Implement a sinusoidal voltage source.

Library Electrical Sources

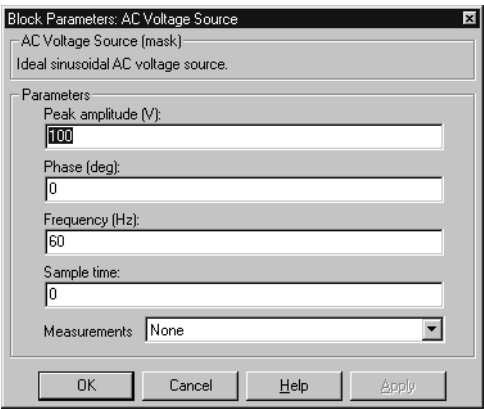
Description The AC Voltage Source block implements an ideal AC voltage source. The output and input of the block correspond respectively to the positive and negative terminals of the source. The generated voltage U is described by the following relationship:



$$U = \text{Amplitude} \times \sin(2 \pi \times \text{Frequency} \times t + \text{Phase} \times \pi / 180)$$

Negative values are allowed for Amplitude and Phase. A zero frequency specifies a DC voltage source. Negative Frequency is not allowed, otherwise Simulink signals an error, and the block will display a question mark in the block icon. You can modify the three source parameters at any time during the simulation.

Dialog Box and Parameters



Peak amplitude

The peak amplitude of the generated voltage source, in volts (V).

Phase

The phase in degrees.

Frequency

The source frequency in hertz (Hz).

AC Voltage Source

Sample time

The sample period in seconds (s). The default is 0, corresponding to a continuous source.

Measurements

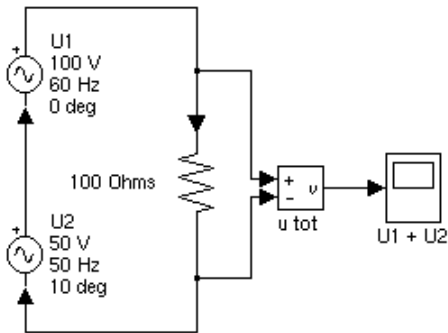
Select **Voltage** to measure the voltage across the terminals of the AC Voltage Source block.

Place a Multimeter block in your model to display the selected measurements during the simulation. In the **Available Measurement** listbox of the Multimeter block, the measurement will be identified by a label followed by the block name.

Measurement	Label
Voltage	u Src:

Example

Two AC Voltage Source blocks at different frequencies are connected in series across a resistor. The sum of the two voltages is read by a Voltage Measurement block. This circuit is available in the `psbacvol tage.mdl` file.



See Also

Controlled Voltage Source, DC Voltage Source, Multimeter

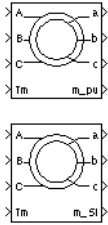
Purpose

Model the dynamics of a three-phase asynchronous machine

Library

Machines

Description

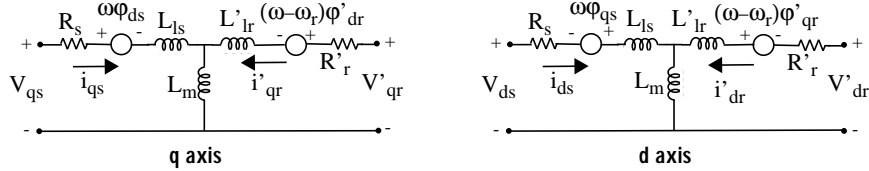


The Asynchronous Machine block operates in either generating or motoring mode. The mode of operation is dictated by the sign of the mechanical torque (positive for motoring, negative for generating). The electrical part of the machine is represented by a fourth-order state-space model and the mechanical part by a second-order system. All electrical variables and parameters are referred to the stator. This is indicated by the prime signs (') in the machine equations given below. All stator and rotor quantities are in the arbitrary two-axis reference frame (qd frame). The subscripts used are defined as follows:

- d : d axis quantity
- q : q axis quantity
- r : rotor quantity
- s : stator quantity
- l : leakage inductance
- m : magnetizing inductance

Asynchronous Machine

Electrical System



$$V_{qs} = R_s i_{qs} + \frac{d}{dt} \phi_{qs} + \omega \phi_{ds}$$

$$V_{ds} = R_s i_{ds} + \frac{d}{dt} \phi_{ds} - \omega \phi_{qs}$$

$$V_{qr} = R'_r i'_{qr} + \frac{d}{dt} \phi'_{qr} + (\omega - \omega_r) \phi'_{dr}$$

$$V_{dr} = R'_r i'_{dr} + \frac{d}{dt} \phi'_{dr} - (\omega - \omega_r) \phi'_{qr}$$

$$T_e = 1.5 p (\phi_{ds} i_{qs} - \phi_{qs} i_{ds})$$

$$\phi_{qs} = L_s i_{qs} + L_m i'_{qr}$$

$$\phi_{ds} = L_s i_{ds} + L_m i'_{dr}$$

$$\phi'_{qr} = L'_r i'_{qr} + L_m i_{qs}$$

$$\phi'_{dr} = L'_r i'_{dr} + L_m i_{ds}$$

$$L_s = L_{ls} + L_m$$

$$L'_r = L'_{lr} + L_m$$

where

Mechanical System

$$\frac{d}{dt} \omega_m = \frac{1}{2H} (T_e - F \omega_m - T_m)$$

$$\frac{d}{dt} \theta_m = \omega_m$$

The Asynchronous Machine block parameters are defined as follows (all quantities referred to the stator):

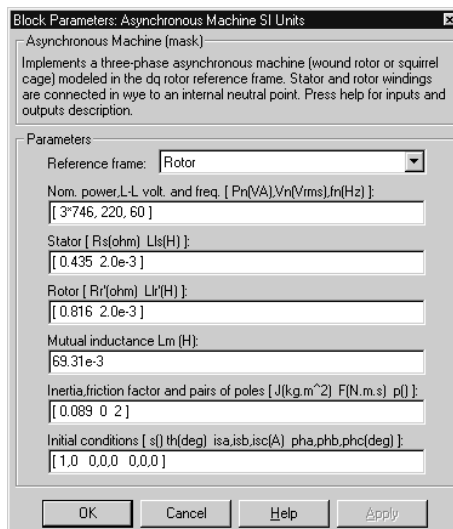
- R_s, L_{ls} : stator resistance and leakage inductance
- R'_r, L'_{lr} : rotor resistance and leakage inductance
- L_m : magnetizing inductance
- L_s, L'_r : total stator and rotor inductances
- V_{qs}, i_{qs} : q axis stator voltage and current
- V_{qr}, i'_{qr} : q axis rotor voltage and current
- V_{ds}, i_{ds} : d axis stator voltage and current
- V_{dr}, i'_{dr} : d axis rotor voltage and current

- ϕ_{qs} , ϕ_{ds} : stator q and d axis fluxes
- ϕ'_{qr} , ϕ'_{dr} : rotor q and d axis fluxes
- ω_m : angular velocity of the rotor
- θ_m : rotor angular position
- p: number of pole pairs
- ω_r : electrical angular velocity ($\omega_m \times p$)
- θ_r : electrical rotor angular position ($\theta_m \times p$)
- T_e : electromagnetic torque
- T_m : shaft mechanical torque
- J: combined rotor and load inertia coefficient (set to infinite to simulate locked rotor)
- H: combined rotor and load inertia constant (set to infinite to simulate locked rotor)
- F: combined rotor and load viscous friction coefficient

Parameters and Dialog Boxes

In the **powerlib** library you can choose between two Asynchronous Machine blocks to specify the electrical and mechanical parameters of the model.

S.I. Units Dialog Box



Block Parameters: Asynchronous Machine SI Units

Asynchronous Machine (mask)
Implements a three-phase asynchronous machine (wound rotor or squirrel cage) modeled in the dq rotor reference frame. Stator and rotor windings are connected in wye to an internal neutral point. Press help for inputs and outputs description.

Parameters

Reference frame:

Nom. power, L-L volt. and freq. [Pn(VA), Vn(Vrms), fn(Hz)]:

Stator [Rs(ohm) Lls(H)]:

Rotor [Rr(ohm) Llr(H)]:

Mutual inductance Lm (H):

Inertia, friction factor and pairs of poles [J(kg.m^2) F(N.m.s) p()]:

Initial conditions [s() th(deg) isa, isb, isc(A) pha, phb, phc(deg)]:

OK Cancel Help Apply

Reference Frame

Specifies the reference frame that is used to convert input voltages (abc reference frame) to the dq reference frame and output currents (dq reference frame) to the abc reference frame. You can choose between the following reference frame transformations:

- Rotor (Park transformation)
- Stationary (Clarke or $\alpha\beta$ transformation)
- Synchronous

The following relationships describe the abc to qd reference frame transformations applied to the Asynchronous Machine block's input voltages.

$$\begin{bmatrix} V_{qs} \\ V_{ds} \end{bmatrix} = \frac{1}{3} \begin{bmatrix} 2 \cos \theta & \cos \theta + \sqrt{3} \sin \theta \\ 2 \sin \theta & \sin \theta - \sqrt{3} \cos \theta \end{bmatrix} \begin{bmatrix} V_{abs} \\ V_{bcs} \end{bmatrix}$$
$$\begin{bmatrix} V'_{qr} \\ V'_{dr} \end{bmatrix} = \frac{1}{3} \begin{bmatrix} 2 \cos \beta & \cos \beta + \sqrt{3} \sin \beta \\ 2 \sin \beta & \sin \beta - \sqrt{3} \cos \beta \end{bmatrix} \begin{bmatrix} V'_{abr} \\ V'_{bcr} \end{bmatrix}$$

In the above equations, θ is the angular position of the reference frame, while $\beta = \theta - \theta_r$ is the difference between the position of the reference frame and the position (electrical) of the rotor. Since the machine windings are connected in a three-wire wye configuration, there is no homopolar (zero) component. This also justifies the fact that two line-to-line input voltages are used instead of three line-to-neutral voltages. The following relationships describe the qd to abc reference frame transformations applied to the Asynchronous Machine block's output currents.

$$\begin{bmatrix} i_{as} \\ i_{bs} \end{bmatrix} = \begin{bmatrix} \cos \theta & \sin \theta \\ -\frac{\cos \theta + \sqrt{3} \sin \theta}{2} & \frac{-\sqrt{3} \cos \theta - \sin \theta}{2} \end{bmatrix} \begin{bmatrix} i_{qs} \\ i_{ds} \end{bmatrix}$$

$$\begin{bmatrix} i'_{ar} \\ i'_{br} \end{bmatrix} = \begin{bmatrix} \cos \beta & \sin \beta \\ -\frac{\cos \beta + \sqrt{3} \sin \beta}{2} & \frac{-\sqrt{3} \cos \beta - \sin \beta}{2} \end{bmatrix} \begin{bmatrix} i'_{qr} \\ i'_{dr} \end{bmatrix}$$

$$i_{cs} = -i_{as} - i_{bs}$$

$$i'_{cr} = -i'_{ar} - i'_{br}$$

The following table shows the values taken by θ and β in each reference frame (θ_e is the position of the synchronously rotating reference frame).

Reference Frame	θ	β
Rotor	θ_r	0
Stationary	0	$-\theta_r$
Synchronous	θ_e	$\theta_e - \theta_r$

The choice of reference frame will affect the waveforms of all dq variables. It will also affect the simulation speed and in certain cases the accuracy of the results. The following guidelines are suggested in [1]:

- Use the stationary reference frame if the stator voltages are either unbalanced or discontinuous and the rotor voltages are balanced (or zero)
- Use the rotor reference frame if the rotor voltages are either unbalanced or discontinuous and the stator voltages are balanced
- Use either the stationary or synchronous reference frames if all voltages are balanced and continuous

Asynchronous Machine

Nominal

The nominal apparent power P_n (VA), rms line-to-line voltage V_n (V), and frequency f_n (Hz).

Stator

The stator resistance R_s (Ω) and leakage inductance L_{ls} (H).

Rotor

The rotor resistance R_r' (Ω) and leakage inductance L_{lr}' (H), both referred to the stator.

Magnetizing inductance

The magnetizing inductance L_m (H).

Mechanical

The combined machine and load inertia coefficient J (kg.m^2), combined viscous friction coefficient F (N.m.s), and pole pairs p .

Initial conditions

The initial slip s , electrical angle θ_e (deg), stator currents magnitudes (A) and phase angles (deg). Can all be computed by the load flow utility in the Powergui block.

Per Unit (p.u.) Dialog Box

Block Parameters: Asynchronous Machine pu Units

Asynchronous Machine (mask)

Implements a three-phase asynchronous machine (wound rotor or squirrel cage) modelled in the dq rotor reference frame. Stator and rotor windings are connected in wye to an internal neutral point. Press help for inputs and outputs description.

Parameters

Reference frame: Rotor

Nom. power, L-L volt. and freq. [Pn(VA), Vn(Vrms), fn(Hz)]:
[3746, 220, 60]

Stator [Rs, Lls] (pu):
[0.0201, 0.0349]

Rotor [Rr', Llr'] (pu):
[0.0377, 0.0349]

Mutual inductance Lm (pu):
1.2082

Inertia constant, friction factor and pairs of poles [H(s) F(pu) p(l)]:
[0.7065, 0.2]

Initial conditions [s() th(deg) isa, isb, isc(p.u.) pha, phb, phc(deg)]:
[1.0 0.0, 0 0.0, 0]

OK

Cancel

Help

Apply

Reference frame

Specifies the reference frame that is used to convert input voltages (abc reference frame) to the dq reference frame and output currents (dq reference frame) to the abc reference frame. See the previous dialog box description for more detail.

Nominal

The nominal apparent power P_n (VA), rms line-to-line voltage V_n (V), and frequency f_n (Hz).

Stator, Rotor, Mutual Inductance, Mechanical parameters

The same electrical and mechanical parameters as in the S.I. units dialog box, expressed in p.u., except inertia constant H , which is expressed in s.

Initial conditions

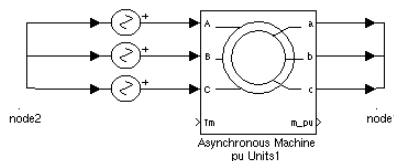
The same initial conditions as in the **S.I. Units** dialog box, except that current magnitudes are in p.u.

Note This block simulates exactly the same way as the asynchronous machine model; the only difference is the way of entering the parameter units.

Inputs and Outputs

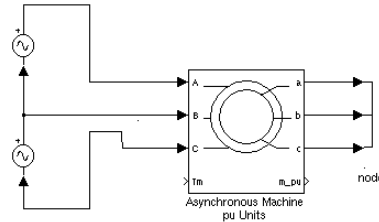
The electrical inputs of the block are the three electrical connections of the stator while the electrical outputs are the three electrical connections of the rotor. Note that the neutral connections of the stator and rotor windings are not available; three-wire Y connections are assumed. The rotor's connections should normally be short-circuited or connected to an external circuit, for example, external resistors or a power converter.

You must be careful when you connect ideal sources to the machine's stator. If you choose to supply the stator via a three-phase Y-connected infinite voltage source, you must use three sources connected in Y.



Asynchronous Machine

However, if you choose to simulate a delta source connection, you must only use two sources connected in series.

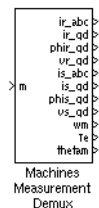


The Simulink input of the block is the mechanical torque at the machine's shaft. This input must be positive in the motoring mode and negative in the generating mode.

The Simulink output of the block is a vector containing 21 variables. They are, in order (refer to Description section above, all currents flowing into machine).

- 1-3: rotor currents i'_{ra} , i'_{rb} and i'_{rc}
- 4-9: i'_{qr} , i'_{dr} , ϕ'_{qr} , ϕ'_{dr} , v'_{qr} and v'_{dr}
- 10-12: stator currents i_{sa} , i_{sb} and i_{sc}
- 13-18: i_{qs} , i_{ds} , ϕ_{qs} , ϕ_{ds} , v_{qs} and v_{ds}
- 19-21: ω_m , T_e and θ_m

These variables can be demultiplexed by using the special Machines Measurement Demux block provided in the Machines library.

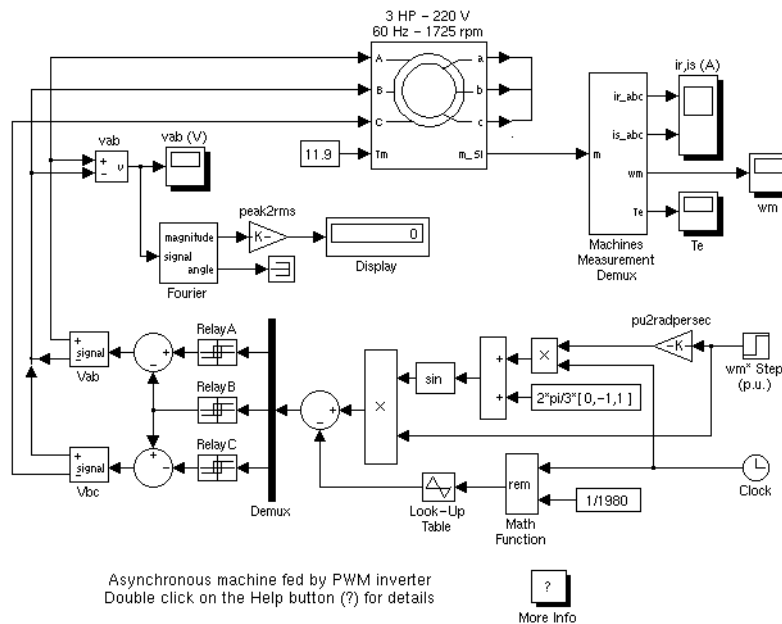


Limitations

The Asynchronous Machine block does not include a representation of the effects of stator and rotor iron saturation.

Example

This example illustrates the use of the Asynchronous Machine block in motoring mode. It consists of an asynchronous machine in an open-loop speed control system. The machine's rotor is short-circuited and the stator is fed by a PWM inverter, which is built with Simulink blocks and interfaced to the Asynchronous Machine block through the Controlled Voltage Source block. The inverter uses sinusoidal pulse-width modulation, which is described in [2]. The base frequency of the sinusoidal reference wave is set at 60 Hz and the triangular carrier wave's frequency is set at 1980 Hz. This corresponds to a frequency modulation factor m_f of 33 ($60 \text{ Hz} \times 33 = 1980$). It is recommended in [2] that m_f be an odd multiple of three and that the value be as high as possible. The 3 HP machine is connected to a constant load of nominal value (11.9 N.m). It is started and reaches the setpoint speed of 1.0 p.u. at $t=0.9$ second. The parameters of the machine are those found in the **SI Units** dialog box above, except for the stator leakage inductance, which is set to twice its normal value. This is done to simulate a smoothing inductor placed between the inverter and the machine. Also, the stationary reference frame was used to obtain the results shown below.

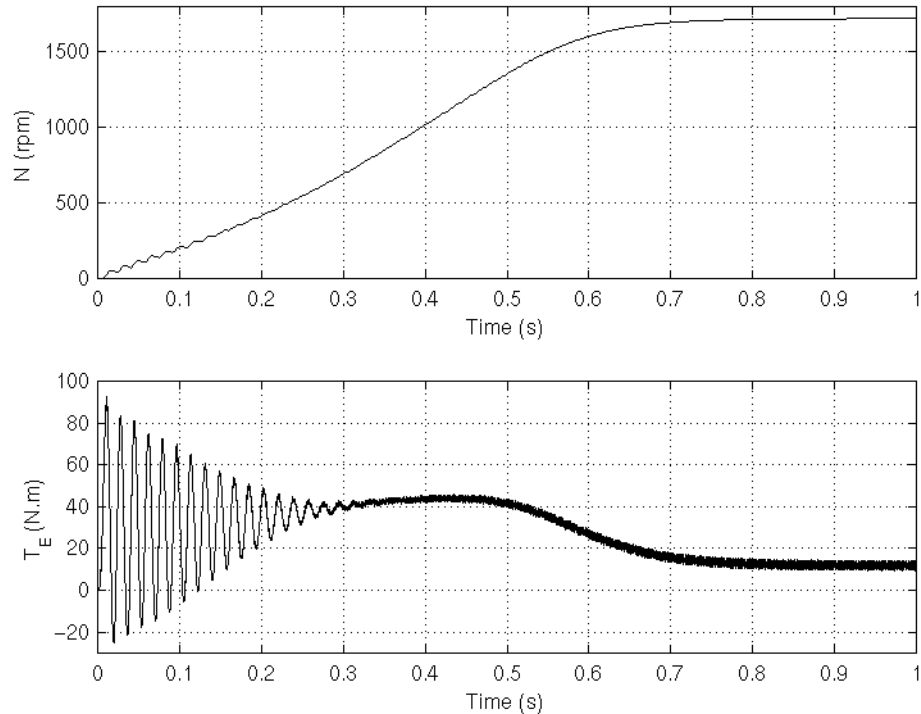


Asynchronous Machine

Open the Simulink diagram by typing `psbpwm` or by choosing **Asynchronous Machine** from the Demos group in the **powerlib** library. Set the simulation parameters as follows:

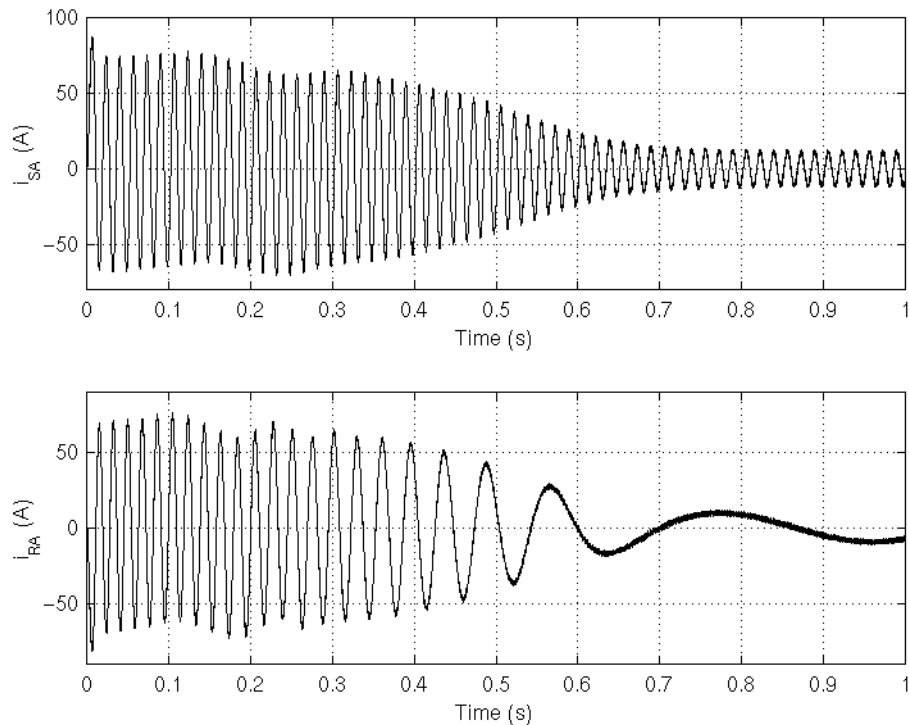
- Integrator type: `Stiff, ode15s`
- Stop time: `1.0 s`
- Integration options: Use default options, except for Relative tolerance, which must be set to `1e-9`. This relatively small tolerance is required because of the high switching rate of the inverter.

Run the simulation by choosing **Start** from the **Simulation** menu. Once the simulation is completed, observe the machine's speed and torque.



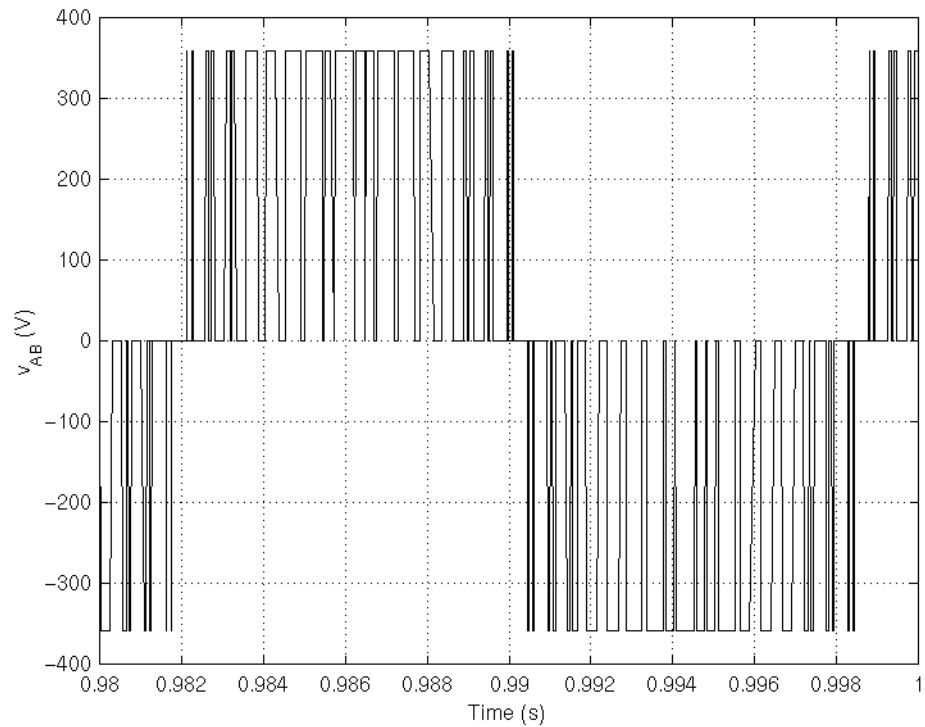
The top graph shows the machine's speed going from 0 to 1725 rpm (1.0 p.u.). The bottom graph shows the electromagnetic torque developed by the machine. Since the stator is fed by a PWM inverter, a noisy torque is observed.

However, this noise is not visible in the speed since it is filtered out by the machine's inertia, but it can also be seen in the stator and rotor currents, which are observed next.



Finally, look at the output of the PWM inverter. Since nothing of interest can be seen at the simulation time scale, the graph concentrates on the last moments of the simulation.

Asynchronous Machine



References

- [1] Krause P.C., O. Wasynczuk, S.D. Sudhoff, *Analysis of Electric Machinery*, IEEE Press, 1995.
- [2] Mohan N., T.M. Undeland, W.P. Robbins, *Power Electronics: Converters, Applications, and Design*, John Wiley & Sons, Inc., 1995, section 8.4.1.

Purpose Implement a circuit breaker opening at current zero crossing.

Library Elements

Description



The Breaker block implements a circuit breaker where the opening and closing times can be controlled either from an external Simulink signal (External control mode), or from an internal control timer (Internal control mode).

The arc extinction process is simulated by opening the breaker device when the current passes through zero (first current zero-crossing following the transition of the Simulink control input from 1 to 0).

When the breaker is closed it behaves as a resistive circuit. It is represented by a resistance (R_{on}). The R_{on} value can be set as small as necessary in order to be negligible compared with external components (typical value $R_{on}=10m\Omega$). When the breaker is open it has an infinite resistance.

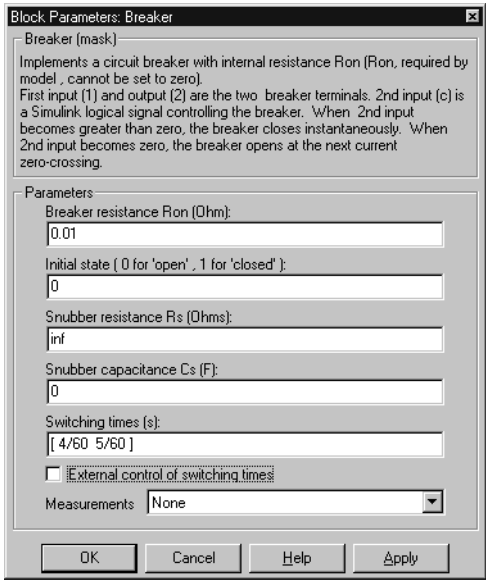
If the Breaker block is set in the external control mode, a control input appears in the block icon. The control signal connected to this second input must be either 0 or 1, 0 to open the breaker, 1 to close it. If the Breaker block is set in the internal control mode, the switching times are specified in the dialog box of the block.

If the breaker initial state is set to 1 (closed) the Power System Blockset automatically initializes all the states of the linear circuit and the Breaker block initial current so that the simulation starts in steady state.

A series Rs-Cs snubber circuit is included in the model. It can be connected or not to the circuit breaker.

Breaker

Dialog Box and Parameters



Breaker resistance Ron

The internal breaker resistance, in ohms (Ω). The **Breaker resistance Ron** parameter cannot be set to zero.

Initial state

The initial state of the breaker. A closed contact is displayed in the block icon when the **Initial state** parameter is set to 1, and an open contact is displayed when set to 0.

Snubber resistance Rs

The snubber resistance, in ohms (Ω). Set the **Snubber resistance Rs** parameter to `inf` to eliminate the snubber from the model.

Snubber capacitance Cs

The snubber capacitance, in farads (F). Set the **Snubber capacitance Cs** parameter to 0 to eliminate the snubber, or to `inf` to get a purely resistive snubber.

Switching times

Specify the vector of switching times when using the Breaker block in the internal control mode. At each switching time the Breaker block will open or close depending on its initial state. For example, if the **Initial state** parameter is 0 (open), the breaker will close at first switching time, open at second switching time, and so on. The **Switching times** parameter is not visible in the dialog box if the **External control of switching times** parameter is checked.

External control of switching times

If checked, adds a second input port to the Breaker block for an external control of the switching times of the breaker. The switching times are defined by a Simulink signal (0 or 1) connected to the second input port of the Breaker block.

Measurements

Select **Branch voltage** to measure the voltage across the Breaker block terminals.

Select **Branch current** to measure the current flowing through the Breaker block. If the snubber device is connected to the breaker model, the measured current is the one flowing through the breaker contacts only.

Select **Branch voltage and current** to measure the breaker voltage and the breaker current.

Place a Multimeter block in your model to display the selected measurements during the simulation. In the **Available Measurement** listbox of the Multimeter block, the measurement will be identified by a label followed by the block name:

Measurement	Label
Branch voltage	ub:
Branch current	i b:

Limitations

The Breaker block is modeled as a current source driven by the voltage appearing across its terminals. When the block is connected in series with an inductor or another current source, you must add the snubber circuit across its

Breaker

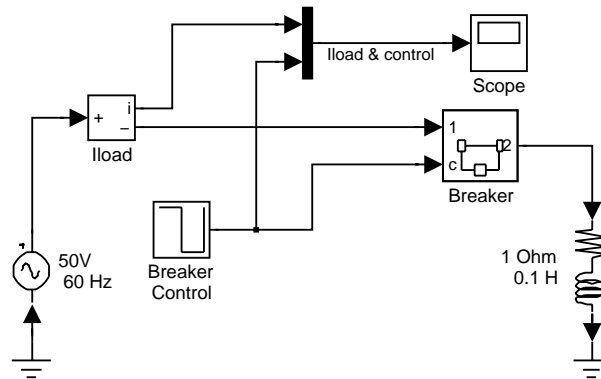
terminals. In most applications you can use a resistive snubber (**Snubber capacitance** parameter set to ∞) with a large resistor value (**Snubber resistance** parameter set to $1e6$, or so).

Due to modeling constraint the internal breaker inductance R_{on} cannot be set to 0.

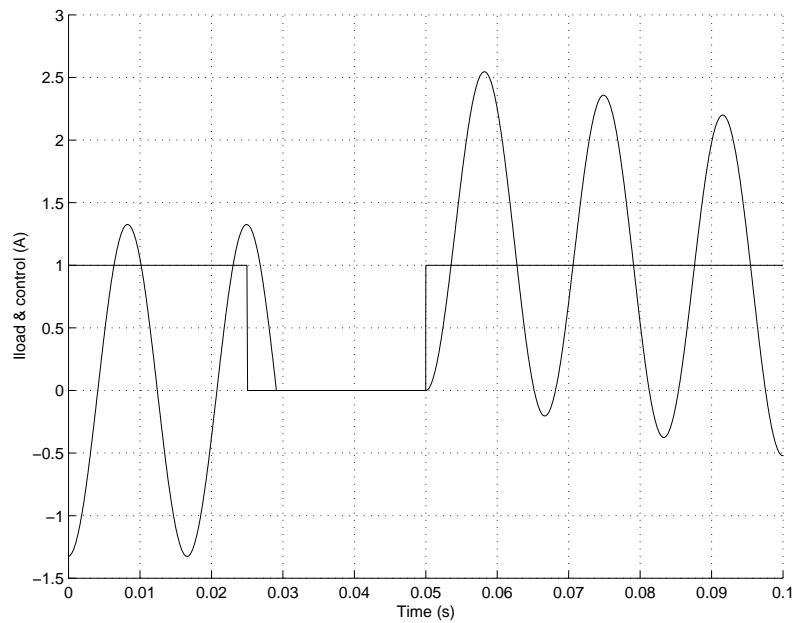
You must use a stiff integration algorithm to simulate circuits with Breaker block. `ode23tb` or `ode15s` with default parameters usually give best simulation speed.

Example

A circuit breaker is connected in series with a series RL circuit on a 60Hz voltage source. The switching times of the Breaker block are controlled by a Simulink signal. The breaker device is initially closed and an opening order is given at $t = 1.5$ cycles, when current reaches a maximum. The current stops at the next zero crossing then the breaker is reclosed at a zero crossing of voltage at $t = 3$ cycles.



This circuit is available in the `psbbreaker.mdl` file. Simulation produces the following results.



Note that the breaker device opens only when the load current has reached zero, after the opening order.

See Also

Ideal Switch, Multimeter

Bus Bar

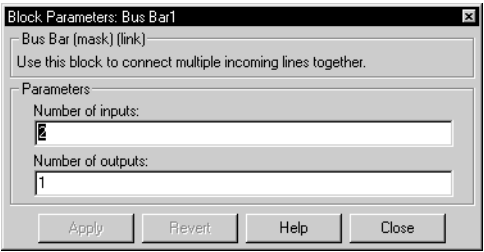
Purpose Implement a labeled network node.

Library Connectors

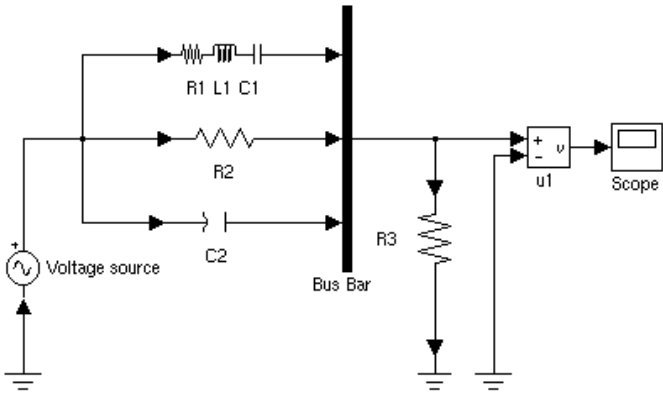
Description The Bus Bar block is used to interconnect Power System Blockset blocks. It allows multiple electrical block outputs and inputs to be connected together.



Dialog Box



Example A three input, one output bus bar is used to connect three elements in parallel.

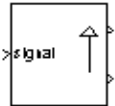


See Also Ground, Neutral

Purpose Implement a controlled current source.

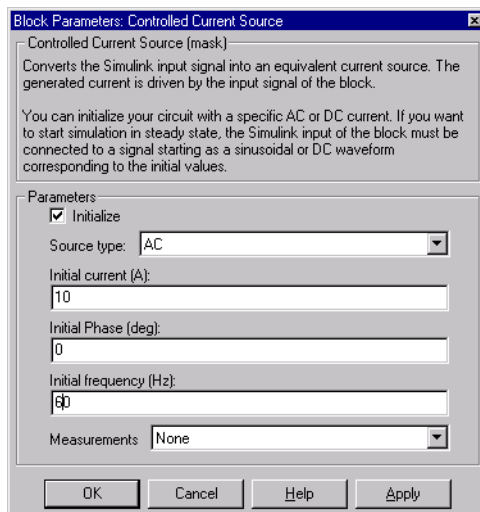
Library Electrical Sources

Description The Controlled Current Source block provides a current source controlled by a Simulink signal. The positive current direction is as shown by the arrow in the block icon.



You can initialize the Controlled Current Source block with a specific AC or DC current. If you want to start the simulation in steady-state, the block input must be connected to a signal starting as a sinusoidal or DC waveform corresponding to the initial values.

Dialog Box and Parameters



Initialize

If checked, initialize the Controlled Current Source block with the specified **Initial current**, **Initial phase**, and **Initial frequency** parameters.

Source type

The type of current source. Select **AC** to initialize the Controlled Current Source Block as an AC current source. Select **DC** to initialize the Controlled Current Source block as a DC current.

Controlled Current Source

The **Source type** parameter is not visible in the dialog box if the **Initialize** parameter is not checked.

Initial current

The initial peak current for the initialization of the source, in amperes (A). The **Initial current** parameter is not visible in the dialog box if the **Initialize** parameter is not checked.

Initial phase

The initial phase for the initialization of the source, in degrees. The **Initial phase** parameter is not visible in the dialog box if the **Source type** parameter is set to **DC**.

Initial frequency

The initial frequency for the initialization of the source, in hertz (Hz). The **Initial frequency** parameter is not visible in the dialog box if the **Source type** parameter is set to **DC**.

Measurements

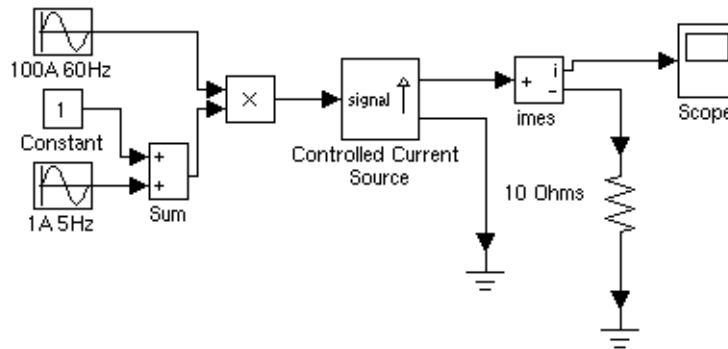
Select **Current** to measure the current flowing through the Controlled Current Source block.

Place a Multimeter block in your model to display the selected measurements during the simulation. In the **Available Measurement** listbox of the Multimeter block, the measurement will be identified by a label followed by the block name:

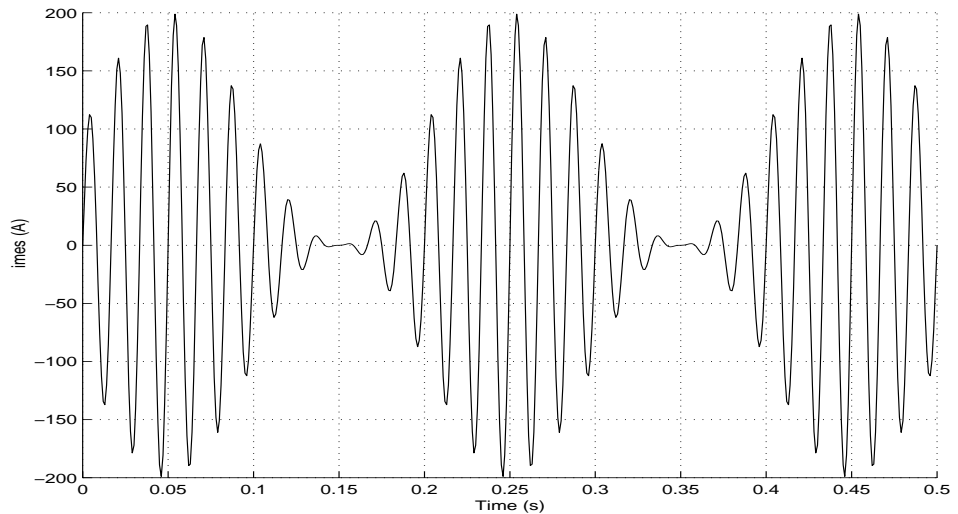
Measurement	Label
Current	i Src:

Example

Generate a 60 Hz current modulated at 5 Hz with the Controlled Current Source block.



This circuit is available in the `psbcontrol curr.mdl` file. Simulation produces the following waveforms:



See Also

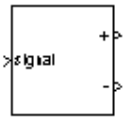
AC Current Source, Controlled Voltage Source, Multimeter

Controlled Voltage Source

Purpose Implement a controlled voltage source.

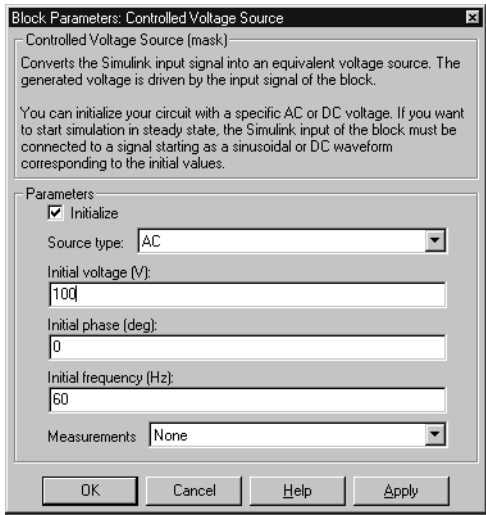
Library Electrical Sources

Description The Controlled Voltage Source block provides a voltage source controlled by a Simulink signal. The first and second outputs of the block are respectively the positive and the negative terminals of the voltage source.



You can initialize the Controlled Voltage Source block with a specific AC or DC voltage. If you want to start the simulation in steady-state, the block input must be connected to a signal starting as a sinusoidal or DC waveform corresponding to the initial values.

Dialog Box and Parameters



Initialize

If checked, initialize the Controlled Voltage Source block with the specified **Initial voltage**, **Initial phase**, and **Initial frequency** parameters.

Source type

The type of voltage source. Select **AC** to initialize the Controlled Voltage Source block with an AC voltage source. Select **DC** to initialize the Controlled Voltage Source Block with a DC voltage.

The **Source type** parameter is not available in the dialog box if the **Initialize** parameter is not checked.

Initial voltage

The initial voltage for the initialization of the source, in amperes (A). The **Initial voltage** parameter is not available in the dialog box if the **Initialize** parameter is not checked.

Initial phase

The initial phase for the initialization of the source, in degrees. The **Initial phase** parameter is not available in the dialog box if the **Source type** parameter is set to **DC**.

Initial frequency

The initial frequency for the initialization of the source, in hertz (Hz). The **Initial frequency** parameter is not available in the dialog box if the **Source type** parameter is set to **DC**.

Measurements

Select **Voltage** to measure the voltage across the terminals of the Controlled Voltage Source block.

Place a Multimeter block in your model to display the selected measurements during the simulation. In the **Available Measurement** listbox of the Multimeter block, the measurement will be identified by a label followed by the block name:

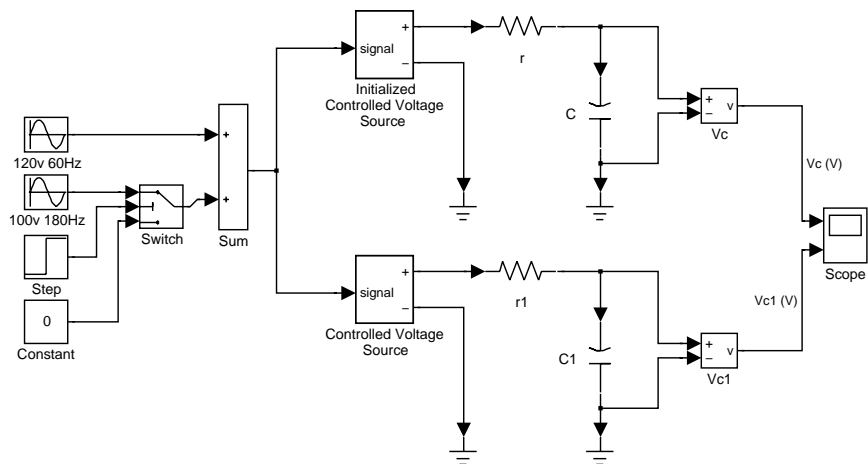
Measurement	Label
Voltage	u Src:

Example

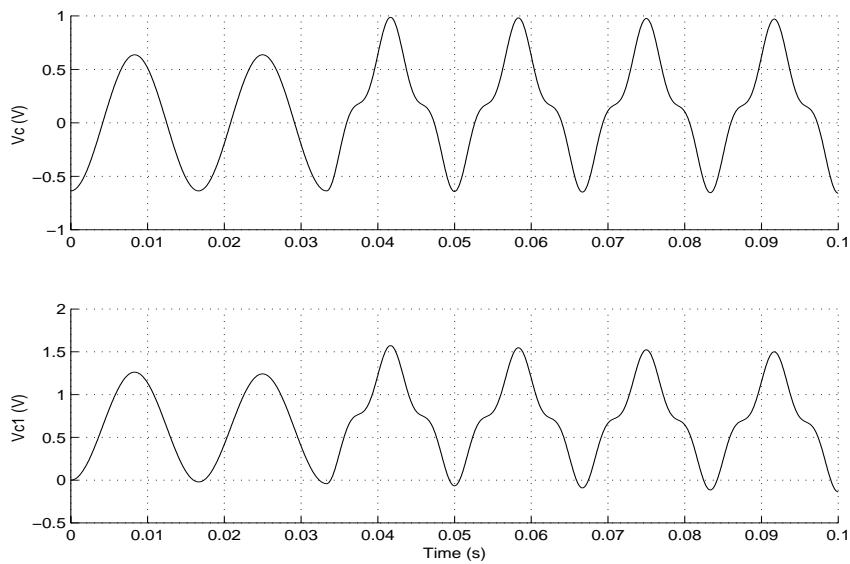
Generate a 60 Hz sinusoidal voltage containing a third harmonic. One Controlled Voltage Source block is initialized as a 120 V AC voltage source with an initial frequency of 60 Hz and initial phase set to zero. The second Controlled Voltage Block is not initialized.

At $t=0.0333$ s, a 100 V-180 Hz sinusoidal signal is added to the 120 V Simulink signal. The resulting capacitor voltages are compared on a scope block.

Controlled Voltage Source



This circuit is stored in the file `psbcontrol vol t.mdl`. The V_c voltage starts in steady-state whereas the V_{c1} voltage contains a DC offset.



See Also

AC Current Source, Controlled Current Source, Multimeter

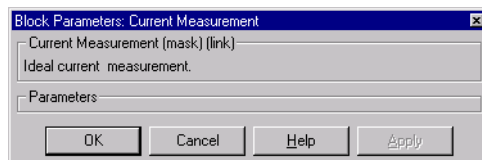
Purpose Measure a current in a circuit.

Library Measurements

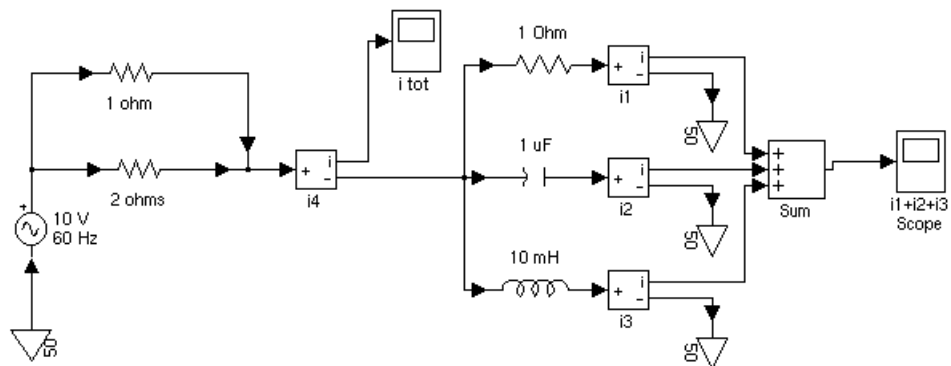
Description The Current Measurement block is used to measure the instantaneous current flowing in any electrical block or connection line. The first output provides a Simulink signal that can be used by other Simulink blocks.



Dialog Box



Example Four Current Measurement blocks are used to read currents in different branches of a circuit. The two scopes display the same current. This circuit is available in the `psbccurrmeasure.mdl` file.



See Also Voltage Measurement, Multimeter

DC Machine

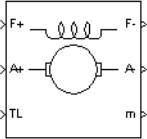
Purpose

Implement a separately excited DC machine.

Library

Machines

Description



This block implements a separately excited DC machine. An access is provided to the field connections (F+, F-) so that the machine model can be used as a shunt-connected or a series-connected DC machine.

The armature circuit (A+, A-) consist of an inductor L_a and resistor R_a in series with a counter electromotive force (CEMF) E .

The CEMF is proportional to the machine speed:

$$E = K_E \omega$$

K_E is the voltage constant and ω is the machine speed.

In a separately excited DC machine model, the voltage constant K_E is proportional to the field current if

$$K_E = L_{af} I_f$$

where L_{af} is the field-armature mutual inductance.

The electromechanical torque developed by the DC machine is proportional to the armature current I_a

$$T_e = K_T I_a$$

where K_T is the torque constant. The sign convention for T_e is:

$$T_e > 0 : \text{generator mode}$$

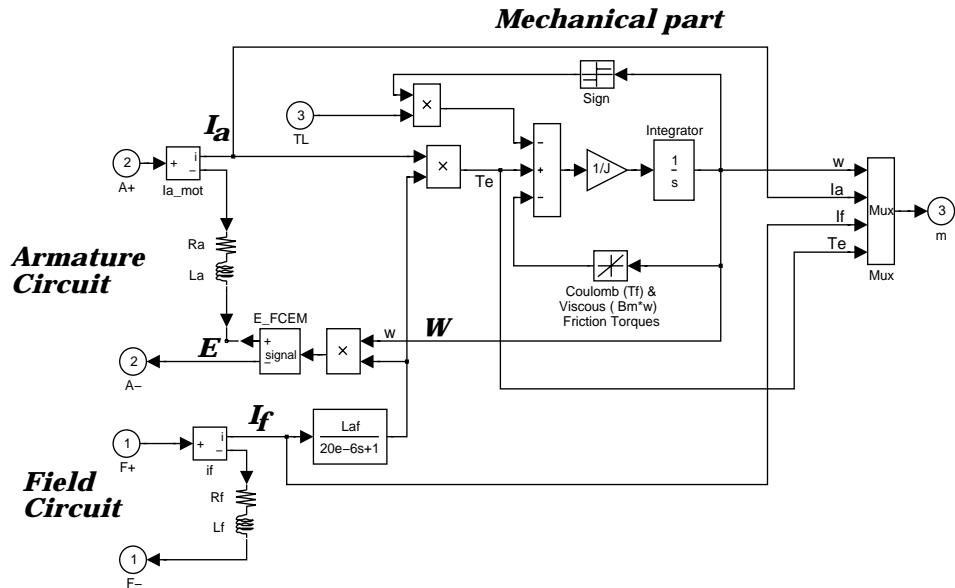
$$T_e < 0 : \text{motor mode}$$

The torque constant is equal to the voltage constant:

$$K_T = K_E$$

The armature circuit and the field circuit of the DC machine model are built with blocks from **powerlib** library. The armature circuit is connected between the input port A+ and the output port A- of the DC Machine block. It is

represented by a Series RLC Branch block in series with a Controlled Voltage Source and a Current Measurement block.



The field circuit is represented by an RL circuit. It is connected between the input port F+ and the output ports F- of the DC Machine block.

The mechanical part computes the speed of the DC machine from the net torque applied to the rotor. The speed is used to implement the CEMF voltage E of the armature circuit.

The mechanical part is represented by Simulink blocks that implement the equation:

$$T_e = J \frac{d\omega}{dt} + B\omega + \text{sgn } T_L$$

Measurements

Four internal signals are multiplexed on the Simulink measurement output vector (third block output) returning:

DC Machine

- Rotor speed in rad/s
- Armature current in A
- Field current in A
- Electromechanical torque in N.m.

Dialog Box

Block Parameters: DC Machine

DC machine (mask) (link)

This block implements a separately excited DC machine. Access is provided to the field connections so that the machine can be used as a shunt-connected or a series-connected DC machine.

Input 1 and output 1 : positive and negative armature terminals
Input 2 and output 2 : positive and negative field terminals
Input 3 : Load torque
Output 3 : Simulink measurement output [w Ia If Te]

Parameters

Armature resistance and inductance [Ra (ohms) La (H)]

[0.6 0.012]

Field resistance and inductance [Rf (ohms) Lf (H)]

[240 120]

Field-armature mutual inductance Laf (H) :

1.8

Total inertia J (kg.m²)

1

Viscous friction coefficient Bm (N.m.s)

0

Coulomb friction torque Tf (N.m)

0

Initial speed (rad/s) :

1

OK

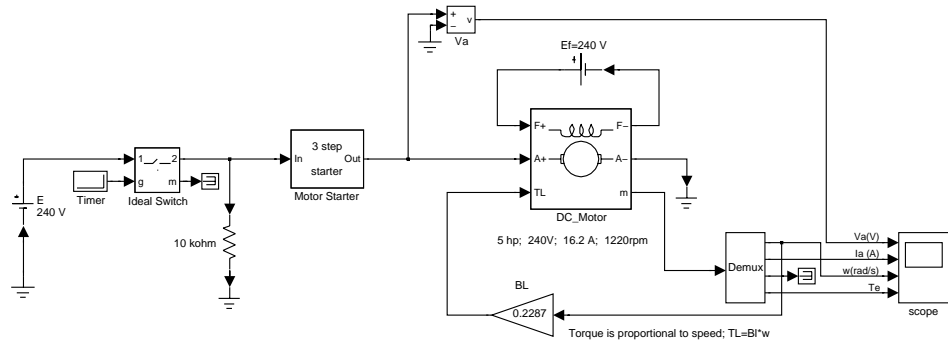
Cancel

Help

Apply

Example

The following circuit illustrates the starting of a 5 HP 240 V DC Machine with a three-step resistance starter. The circuit is available in the psbdcmotor.mdl demo



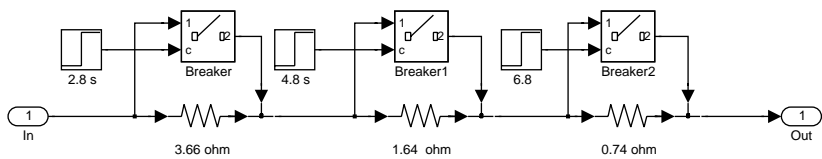
Starting of a 5 HP 240V DC motor with a 3 step resistance starter

?

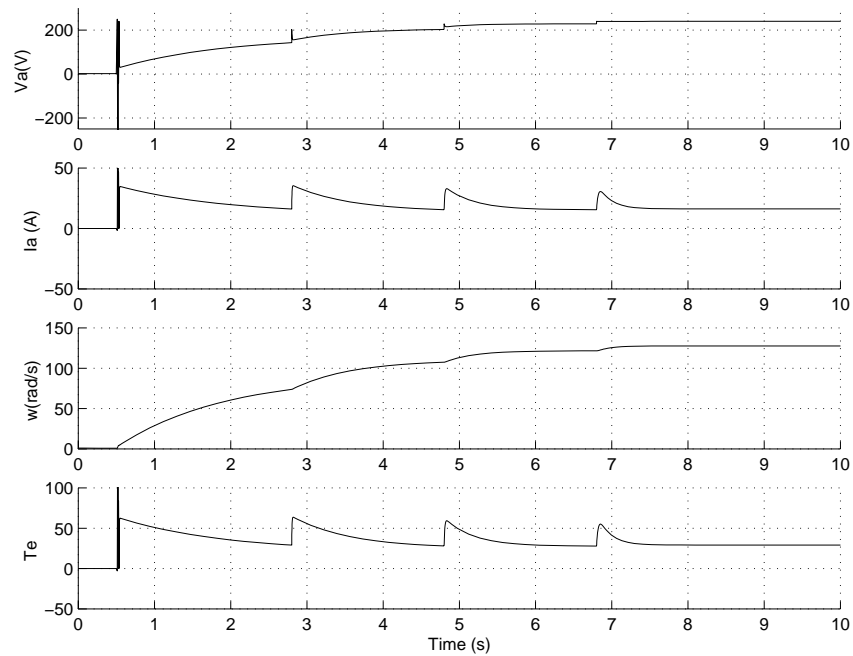
More Info

powergui

The Motor Starter subsystem is:



DC Machine



Reference

Reference Analysis of Electric Machinery, Krause and al., pp. 89-92.

See Also

Asynchronous Machine, Synchronous Machine

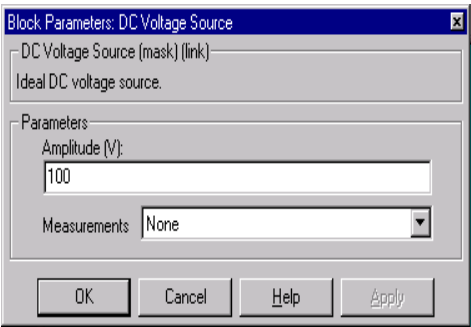
Purpose Implement a DC voltage source.

Library Electrical Sources

Description The DC Voltage Source block implements an ideal DC voltage source. The output and input are respectively the positive and negative source terminals. The voltage level is set in the dialog box. You can modify the voltage at any time during the simulation.



Dialog Box and Parameters



Amplitude

The amplitude of the source, in volts (V).

Measurements

Select **Voltage** to measure the voltage across the terminals of the DC Voltage Source block.

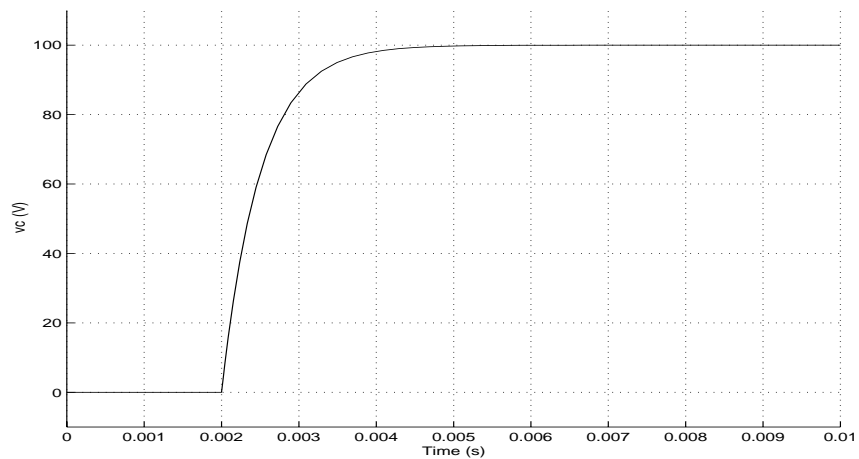
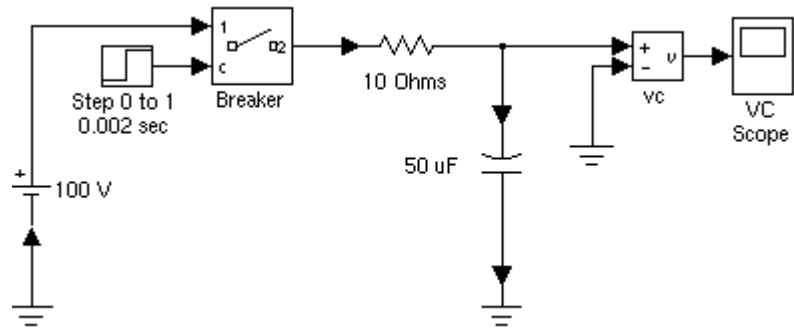
Place a Multimeter block in your model to display the selected measurements during the simulation. In the **Available Measurement** listbox of the Multimeter block, the measurement will be identified by a label followed by the block name:

Measurement	Label
Voltage	u Src:

DC Voltage Source

Example

Simulation of the transient response of a first order RC circuit. This circuit is available in the psbdcvol t age. mdl file.



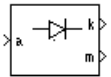
See Also

AC Voltage Source, Controlled Voltage Source

Purpose Implement a diode model.

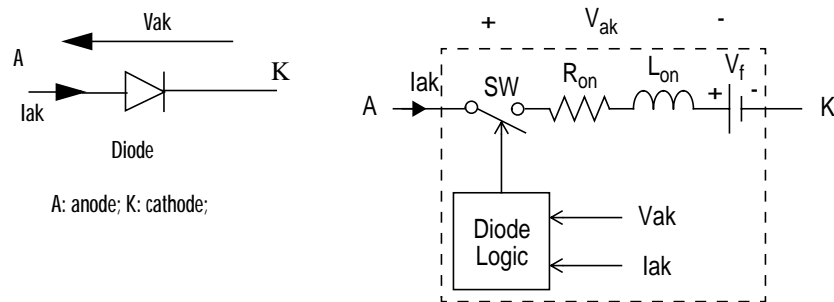
Library Power Electronics

Description



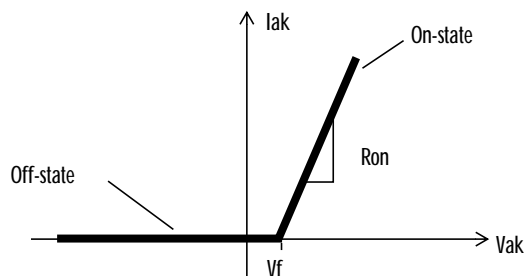
A diode is a semiconductor device that is controlled by its own voltage and current. When a diode is forward biased ($V_{ak} > 0$), it starts to conduct with a small forward voltage across it. It turns off when the current flow into the device becomes zero. When the diode is reverse biased ($V_{ak} < 0$), it stays in the off-state.

The Diode block is simulated as a resistor, an inductor, and a DC voltage source (V_f) connected in series with a switch. The switch is controlled by the voltage V_{ak} and current I_{ak} .



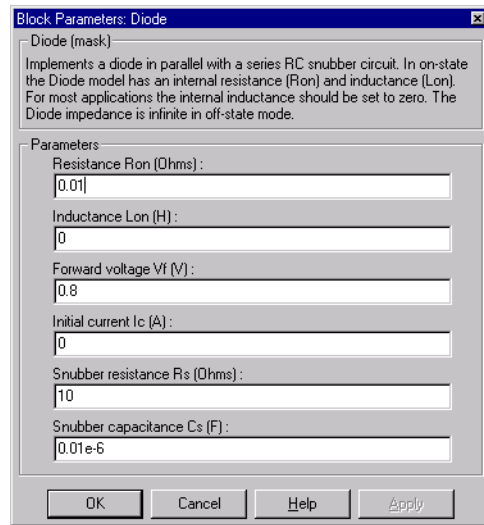
The Diode block also contains a series Rs-Cs snubber circuit that can be connected in parallel with the diode device.

The static VI characteristic of this model is shown in the figure below.



Diode

Dialog Box and Parameters



Resistance Ron

The diode internal resistance Ron, in ohms (Ω). The **Resistance Ron** parameter cannot be set to 0 when the **Inductance Lon** parameter is set to 0.

Inductance Lon

The diode internal inductance Lon, in henries (H). The **Inductance Lon** parameter cannot be set to 0 when the **Resistance Ron** parameter is set to 0.

Forward voltage Vf

The forward voltage of the diode device, in volts (V).

Initial current Ic

When the **Inductance Lon** parameter is greater than zero, you can specify an initial current flowing in the diode device. It is usually set to zero in order to start the simulation with the diode device blocked.

You can specify an initial current Ic value corresponding to a particular state of the circuit. In such a case all states of the linear circuit must be set accordingly. Initializing all states of a power-electronic converters is a complex task. Therefore, this option is useful only with simple circuits.

Snubber resistance R_s

The snubber resistance, in ohms (Ω). Set the **Snubber resistance R_s** parameter to `inf` to eliminate the snubber from the model.

Snubber capacitance C_s

The snubber capacitance in farads (F). Set the **Snubber capacitance C_s** parameter to 0 to eliminate the snubber, or to `inf` to get a purely resistive snubber.

Inputs and Outputs

The input of the block is the anode (a) of the diode and the first output is the cathode (k). The second output (m) is a Simulink measurement output vector [Iak, Vak] returning the diode current and voltage.

Assumptions and Limitations

The Diode block implements a macro-model of a diode device. It does not take into account either the geometry of the device or the complex physical processes underlying the state change [1]. The leakage current in the blocking state and the reverse-recovery (negative) current are not considered. In most circuits, the reverse current does not affect converter or other device characteristics.

Depending on the value of the inductance L_{on} , the diode is modeled either as a current source ($L_{on} > 0$) or as a variable topology circuit ($L_{on} = 0$). See Advanced Topics chapter for more details on this topic.

As the diode device is modeled as a current source, the Diode block cannot be connected in series with an inductor, a current source, or an open circuit, unless its snubber circuit is in use.

You must use a stiff integrator algorithm to simulate circuits containing diodes. `ode23tb` or `ode15s` with default parameters usually give best simulation speed.

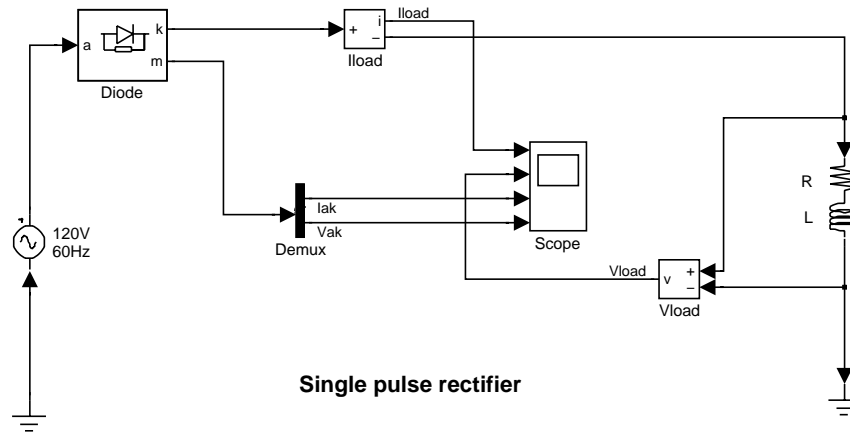
The inductance L_{on} will be forced to zero if you choose to discretize your circuit.

Example

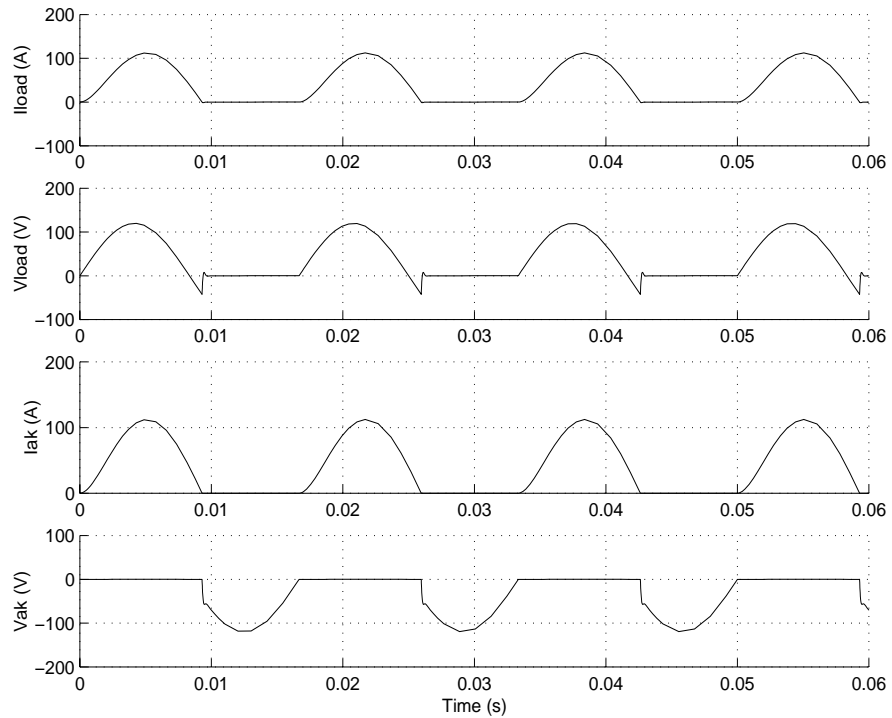
Single pulse rectifier consisting of a Diode block, an RL load, and an AC Voltage source block, with the following parameters:

Diode block: $R_{on} = 0.001 \Omega$, $L_{on} = 0 \text{ H}$, $V_f = 0.8 \text{ V}$, $R_s = 20 \Omega$, $C_s = 4 \times 10^{-6} \text{ F}$; $R = 1 \Omega$ $L = 1 \text{ mH}$.

Diode



This circuit is available in the psbdi ode. mdl file. Simulation produces the following results:



References

- [1] Rajagopalan, V., *Computer-Aided Analysis of Power Electronic Systems*, Marcel Dekker, Inc., New York, 1987.
- [2] Mohan, N., *Power Electronic, Converters, Applications and Design*, John Wiley & Sons, Inc., New York, 1995.

See Also

Thyristor, Universal Bridge

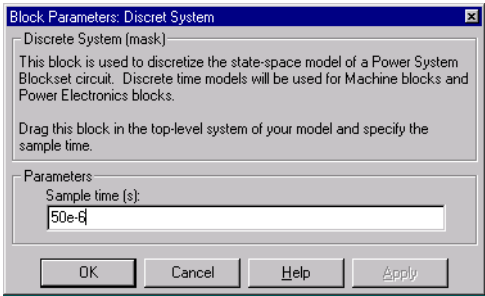
Discrete System

Purpose	Discretize the state-space model of a Power System Blockset circuit.
Library	powerlib
Description	<p>The Discrete System block is used to discretize the state-space model of a Power System Blockset model. Discrete time models will be used for the linear elements as well as for the nonlinear blocks of the Elements, Machines, and Power Electronics libraries of powerlib.</p> <p>Drag this block in the top-level system of your model and specify the sample-time for the discretization. The state-space matrices of the linear part of the circuit will be discretized using the Tustin integration method, that is equivalent to fixed step trapezoidal integration.</p>

Discrete system
 $T_s=5e-005$

Continuous
system

Dialog Box and Parameter



Sample time

The sample time used to discretize the state-space matrices of the linear part of the circuit. Set the **Sample time** parameter to a value greater than zero to discretize the circuit. The icon displays the value of the sample time. If you set the **Sample time** parameter to 0, the model will not be discretized and "Continuous system" will be displayed in the block icon.

Limitations	Discretization of circuits containing individual force-commutated switches (GTO, IGBT, or MOSFET) is not possible. If applicable, use instead the Universal Bridge block that allows discretization of GTO, IGBT, and MOSFET blocks.
Examples	See Sessions 3 and 5 in the Tutorial chapter for examples using the Discrete System block.

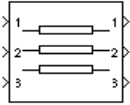
Purpose

Implement an N-phase distributed parameter transmission line model with lumped losses.

Library

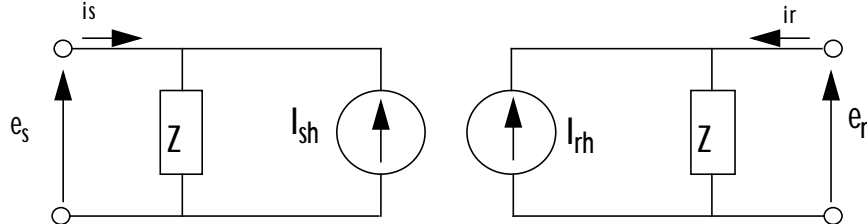
Elements

Description



The Distributed Parameter Line block implements an N-phase distributed parameter line model with lumped losses. The model is based on the Bergeron's travelling wave method used by the Electromagnetic Transient Program (EMTP)[1]. In this model, the lossless distributed LC line is characterized by two values (for a single phase line): the surge impedance $Z_c = \sqrt{L/C}$ and the phase velocity $v = 1/\sqrt{LC}$.

The model uses the fact that the quantity $e+Zi$, where e is line voltage and i is line current, entering one end of the line must arrive unchanged at the other end after a transport delay of $\tau = d/v$, where d is the line length. By lumping $R/4$ at both ends of the line and $R/2$ in the middle and using the current injection method of the Power System Blockset, the following two-port model is derived:



$$I_{sh}(t) = \left(\frac{1+h}{2}\right) \left[\frac{1}{Z} e_r(t-\tau) + h i_r(t-\tau) \right] + \left(\frac{1-h}{2}\right) \left[\frac{1}{Z} e_s(t-\tau) + h i_s(t-\tau) \right]$$

$$I_{rh}(t) = \left(\frac{1+h}{2}\right) \left[\frac{1}{Z} e_s(t-\tau) + h i_s(t-\tau) \right] + \left(\frac{1-h}{2}\right) \left[\frac{1}{Z} e_r(t-\tau) + h i_r(t-\tau) \right]$$

$$\text{where } Z = Z_C + \frac{R}{4} \quad h = \frac{Z_C - \frac{R}{4}}{Z_C + \frac{R}{4}} \quad Z_C = \sqrt{\frac{L}{C}} \quad \tau = d\sqrt{LC}$$

Distributed Parameter Line

For multiphase line models, modal transformation is used to convert line quantities from phase values (line currents and voltages) into modal values independent of each other. The previous calculations are made in the modal domain before being converted back to phase values.

In comparison to the pi sections line model, the distributed line represents wave propagation phenomena and line end reflections with much better accuracy. See comparison between the two models in the Example section.

Dialog Box and Parameters

Block Parameters: Distributed Parameters Line

Distributed Parameters Line (mask)

Implements N-phases, distributed parameter line model. The R, L, and C line parameters are specified by [N*N] matrices.

To model a two-, three-, or a six-phase symmetrical line, you can either specify complete [N*N] matrices or simply enter sequence parameters vectors: the positive and zero sequence parameters for a two-phase or three-phase transposed line, plus the mutual zero-sequence for a six-phase transposed line (2 coupled 3-phase lines).

Parameters

Number of phases N

8

Frequency used for R L C specification (Hz)

60

Resistance per unit length (Ohms/km) [N*N matrix] or [R1 R0]

[0.01273 0.3864]

Inductance per unit length (H/km) [N*N matrix] or [L1 L0 L0m]

[0.9337e-3 4.1264e-3]

Capacitance per unit length (F/km) [N*N matrix] or [C1 C0 C0m]

[12.74e-9 7.751e-9]

Line length (km)

300

Measurements

None

OK

Cancel

Help

Apply

Number of phases N

Specifies the number of phases N of the model. The block icon dynamically changes according to the number of phases that you specify. When you apply the parameters or close the dialog box, the number of inputs and outputs is updated. The icon displays the individual conductors for $N \leq 3$. For $N > 3$, only one conductor is displayed.

Frequency used for RLC specifications

Specifies the frequency used to compute the modal resistance R , inductance L , and capacitance C matrices of the line model.

Resistance per unit length

The resistance R per unit length, N -by- N matrix in ohms/km (Ω/km).

For a symmetrical line, you can either specify the N -by- N matrix or the sequence parameters: For a two-phase or three-phase continuously transposed line, you can enter the positive and zero-sequence resistances $[R1 \ R0]$. For a symmetrical six-phase line you can enter the sequence parameters plus the zero-sequence mutual resistance $[R1 \ R0 \ R0m]$.

For unsymmetrical lines, you must specify the complete N -by- N resistance matrix.

Inductance per unit length

The inductance L per unit length, N -by- N matrix in henries/km (H/km).

For a symmetrical line, you can either specify the N -by- N matrix or the sequence parameters: For a two-phase or three-phase continuously transposed line, you can enter the positive and zero-sequence inductances $[L1 \ L0]$. For a symmetrical six-phase line you can enter the sequence parameters plus the zero-sequence mutual inductance $[L1 \ L0 \ L0m]$.

For unsymmetrical lines, you must specify the complete N -by- N inductance matrix.

Capacitance per unit length

The Capacitance C per unit length, N -by- N matrix in farad/km (F/km).

For a symmetrical line, you can either specify the N -by- N matrix or the sequence parameters: For a two-phase or three-phase continuously transposed line, you can enter the positive and zero-sequence capacitances $[C1 \ C0]$. For a symmetrical six-phase line you can enter the sequence parameters plus the zero-sequence mutual capacitance $[C1 \ C0 \ C0m]$.

For unsymmetrical lines, you must specify the complete N -by- N capacitance matrix.

Line length

The line length, in km.

Distributed Parameter Line

Measurements

Select **Phase-to-ground voltages** to measure the sending end and receiving end voltages for each phase of the line model.

Place a Multimeter block in your model to display the selected measurements during the simulation. In the **Available Measurement** listbox of the Multimeter block, the measurement will be identified by a label followed by the block name:

Measurement	Label
Phase-to-ground voltages, sending end	Us_ph1_gnd: , Us_ph2_gnd: , Us_ph3_gnd: , etc.
Phase-to-ground voltages, receiving end	Ur_ph1_gnd: , Ur_ph2_gnd: , Ur_ph3_gnd: , etc.

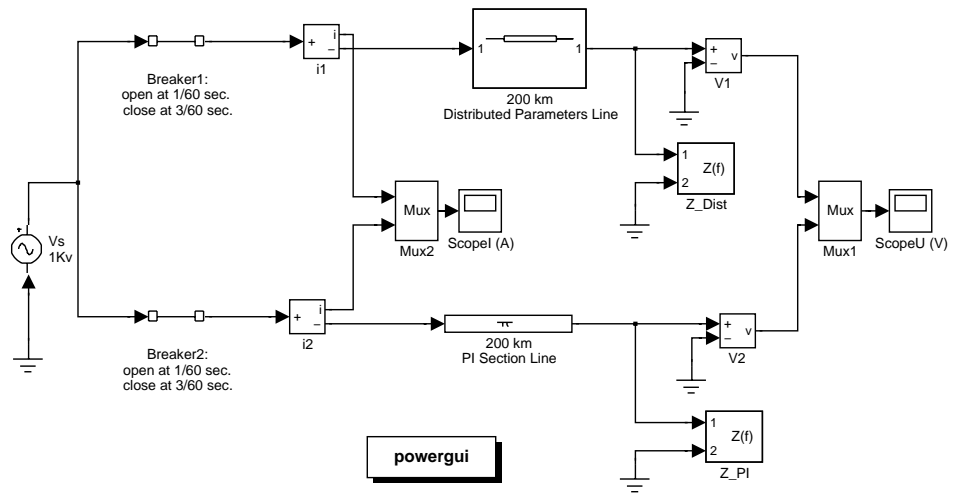
Limitations

This model does not represent accurately the frequency dependence of R L C parameters of real power lines. Indeed, because of the skin effects in the conductors and ground, the R and L matrices exhibit strong frequency dependence, causing an attenuation of the high frequencies.

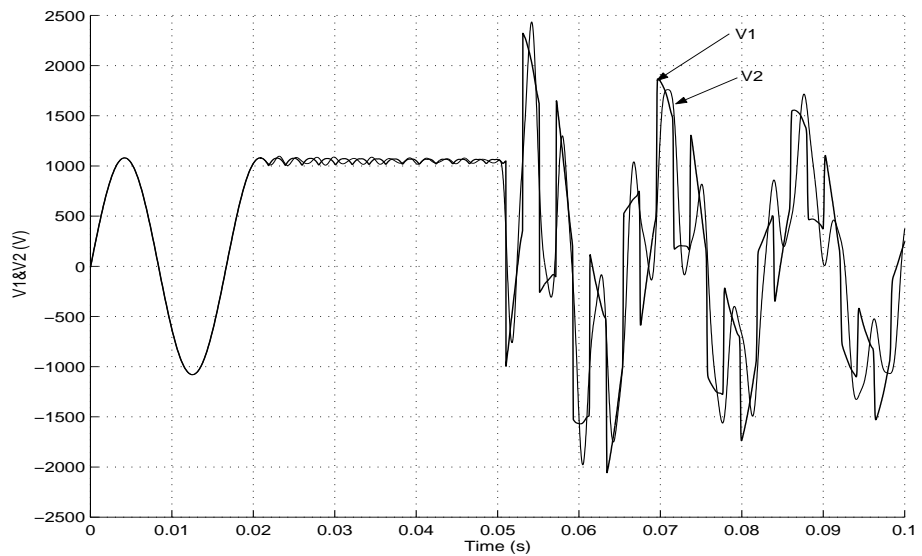
Example

A 200 km line is connected on a 1 kv, 60 Hz infinite source. The line is de-energized and then re-energized after 2 cycles. The simulation is performed simultaneously with the Distributed Parameter Line block and with the PI Section Line block. This circuit is available in the psbmonophase1 ne. mdl file.

Distributed Parameter Line

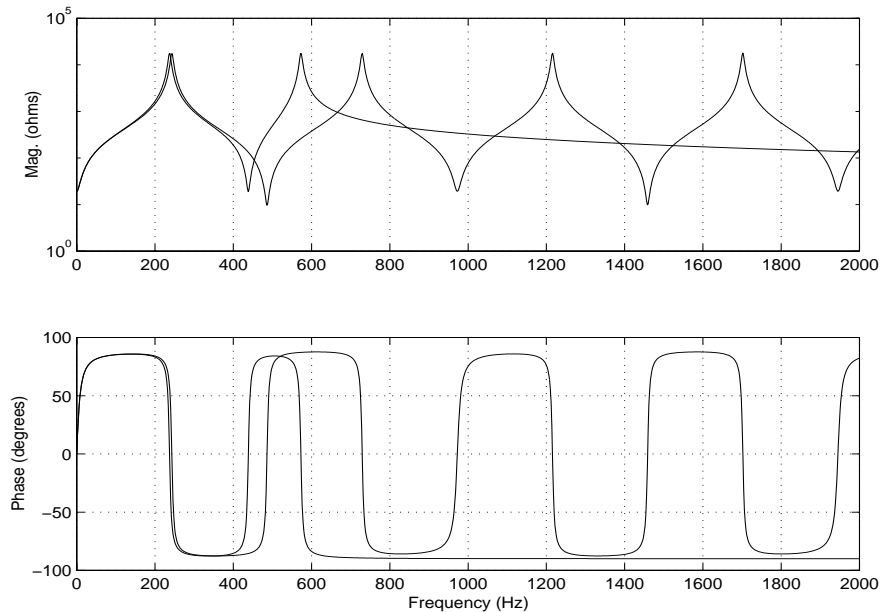


The receiving end voltage obtained with the Distributed Parameter Line block is compared with the one obtained with the PI Section Line block (2 sections).



Distributed Parameter Line

Open the **powergui**. In the **Tools** menu select **Impedance vs Frequency Measurement**. A new window appears, listing the two Impedance Measurement blocks connected to your circuit. Set the parameters of **powergui** to compute impedance in the 0:2000 Hz frequency range. Click on the **Display** button. The two impedances are displayed on the same graph.



Note that the distributed parameter line shows a succession of poles and zeros equally spaced, every 486 Hz. The first pole occurs at 243 Hz, corresponding to frequency $f=1/(4*T)$ where:

$$T = \text{travelling time} = l * \sqrt{L * C} = 200 * \sqrt{2.137e-3 * 12.37e-9} = 1.028 \text{ ms}$$

The pi section line only shows two poles because it consists of two pi sections. Impedance comparison shows that a two-section PI line gives a good approximation of the distributed line for the 0-350 Hz frequency range.

References

[1] Dommel, H, "Digital Computer Solution of Electromagnetic Transients in Single and Multiple Networks," *IEEE Transactions on Power Apparatus and Systems*, Vol PAS-88, No. 4, April 1969

See Also

PI Section Line, Multimeter

Excitation System

Purpose

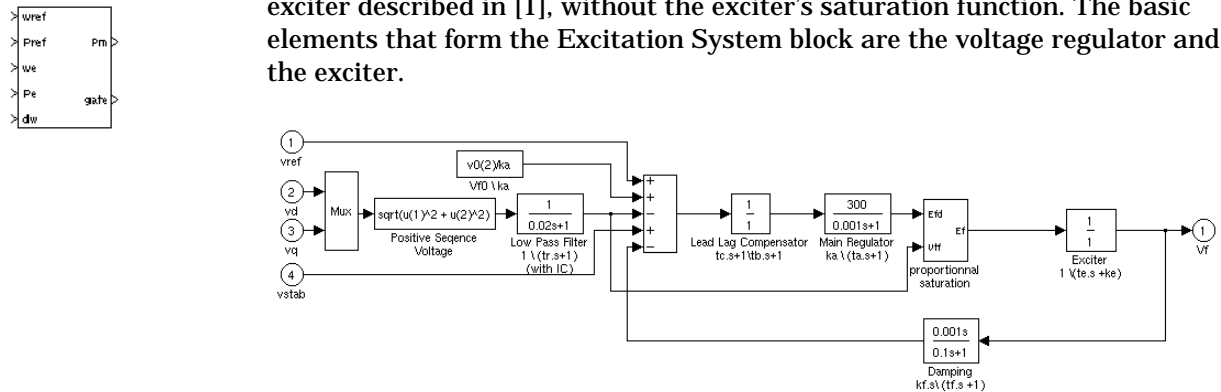
Provide an excitation system for the synchronous machine and regulate its terminal voltage in generating mode.

Library

Machines

Description

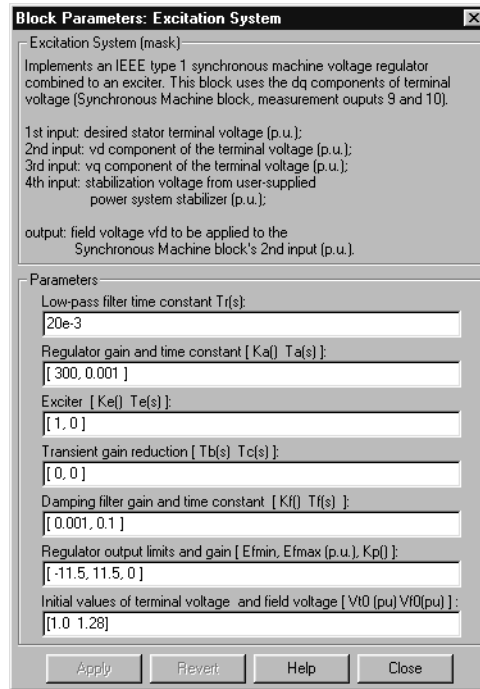
The Excitation System block is a purely Simulink system implementing a DC exciter described in [1], without the exciter’s saturation function. The basic elements that form the Excitation System block are the voltage regulator and the exciter.



The exciter is represented by the following transfer function between the exciter voltage V_{fd} and the regulator’s output ef :

$$\frac{V_{fd}}{ef} = \frac{1}{Ke + sTe}$$

Dialog Box and Parameters



Low-pass filter time constant

The time constant T_r , in seconds (s), of the first-order system which represents the stator terminal voltage transducer.

Regulator

The gain K_a and time constant T_a , in seconds(s), of the first-order system representing the main regulator.

Exciter

The gain K_e and time constant T_e , in seconds (s) of the first-order system representing the exciter.

Transient gain reduction time constants

The time constants T_b , in seconds (s), and T_c , in seconds (s), of the first-order system representing a lead-lag compensator.

Excitation System

Damping filter

The gain K_f and time constant T_f , in seconds (s), of the first-order system representing a derivate feedback.

Regulator output limits

Limits E_{fmin} and E_{fmax} are imposed to the output of the voltage regulator. The upper limit can be constant and equal to E_{fmax} or variable and equal to the rectified stator terminal voltage V_{tf} times a proportional gain K_p . If K_p is set to zero, the former will apply. If K_p is set to a positive value, the latter will apply.

Initial conditions

The initial values of terminal voltage V_{t0} (p.u.) and field voltage V_{f0} (p.u.). When set correctly, they allow you to start the simulation in steady-state. Initial terminal voltage should normally be set to 1 p.u. Initial field voltage can be computed by the load flow utility of the Powergui block.

Example

See the Hydraulic Turbine and Governor block.

Inputs and Outputs

The first input of the block is the desired value of the stator terminal voltage. The following two inputs are the v_q and v_d components of the terminal voltage. The fourth input can be used to provide additional stabilization of power system oscillations. All inputs are in p.u. The output of the block is the field voltage V_f for the Synchronous Machine block (p.u).

References

[1] "Recommended Practice for Excitation System Models for Power System Stability Studies." *IEEE Standard 421.5-1992*, August 1992

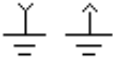
See Also

Hydraulic Turbine and Governor, Powergui, Steam Turbine and Governor, Synchronous Machine

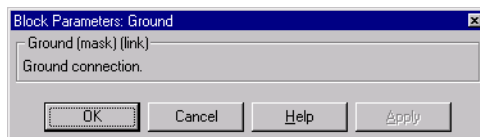
Purpose Provide a connection to the ground.

Library Connectors

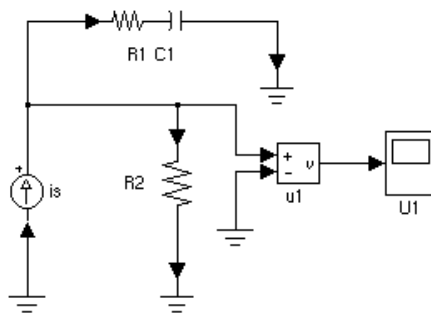
Description The Ground block implements a connection to the ground. For drawing ease, two types of Ground blocks are provided: one block with an input and one block with an output.



Dialog Box



Example The following circuit shows an application of both types of Ground blocks. This circuit is available in the psbground.mdl file.



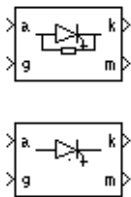
See Also Neutral, Bus Bar

GTO

Purpose Implement a gate turn-off (GTO) thyristor model.

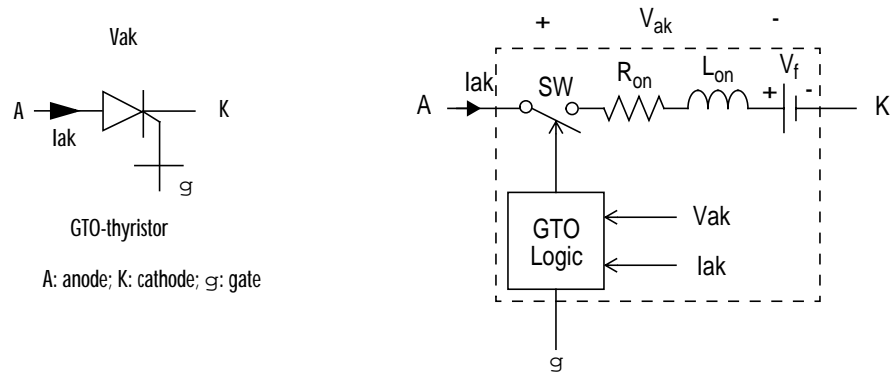
Library Power Electronics

Description

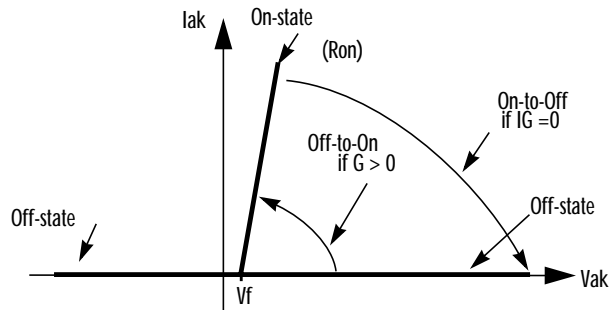


The Gate Turn-Off (GTO) thyristor is a semiconductor device that can be turned on and off via a gate signal. Like a conventional thyristor, the GTO thyristor can be turned on by a positive gate signal ($g > 0$). However, unlike the thyristor which can be turned off only at a zero crossing of current, the GTO can be turned off at any time by applying a gate signal equal to zero.

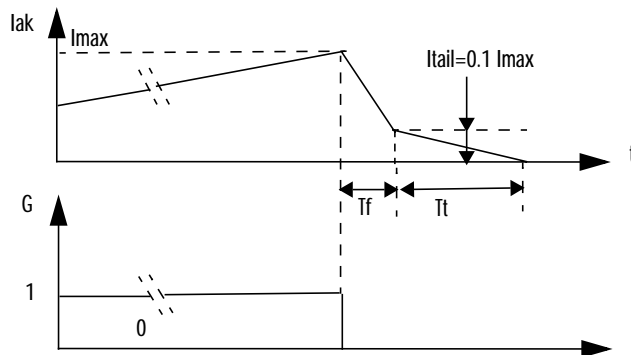
The GTO thyristor is simulated as a resistor (R_{on}), an inductor (L_{on}) and a DC voltage source (V_f) connected in series with a switch (SW). The switch is controlled by a logical signal depending on the voltage V_{ak} , current I_{ak} and the gate signal (g).



The GTO thyristor turns on when the anode-cathode voltage is greater than V_f and a positive pulse signal is present at the gate input ($g > 0$). When the gate signal is set to zero, the GTO thyristor starts to block but its current does not stop instantaneously.



Since the current extinction process of a GTO thyristor contributes significantly to the turn-off losses, the turn-off characteristic is built into the model. The current decrease is approximated by two segments. When the gate signal becomes zero, the current I_{ak} first decreases from the value I_{max} (value of I_{ak} when the GTO thyristor starts to open) to $I_{max}/10$, during the fall time (T_f), and then from $I_{max}/10$ to zero during the tail time (T_t). The GTO thyristor turns off when the current I_{ak} becomes zero. The latching and holding currents are not considered.



V_f , R_{on} , and L_{on} are the forward voltage drop while in conduction, the forward conducting resistance and the inductance of the device, respectively.

The GTO block contains a series R_s - C_s snubber circuit that can be connected in parallel with the GTO thyristor.

Dialog Box and Parameters

Block Parameters: GTO

Gto (mask)

Implements a GTO thyristor in parallel with a series RC snubber circuit. In on-state the GTO thyristor model has internal resistance (Ron) and inductance (Lon). In off-state the GTO thyristor model has infinite impedance. The internal inductance cannot be set to zero.

Discretization of the GTO thyristor model is available only through the Universal Bridge block.

Parameters

Resistance Ron (Ohms):

Inductance Lon (H):

Forward voltage Vf (V):

Current 10% fall time Tf (s):

Current tail time Tt (s):

Initial current Ic (A):

Snubber resistance Rs (Ohms):

Snubber capacitance Cs (F):

OK Cancel Help Apply

Resistance Ron

The internal resistance Ron, in ohms (Ω).

Inductance Lon

The internal inductance Lon, in henries (H). The **Inductance Lon** parameter cannot be set to 0.

Forward voltage Vf

The forward voltage of the GTO thyristor device, in volts (V).

Current 10% fall time

The current fall time Tf, in seconds (s).

Current tail time

The current tail time Tt, in seconds (s).

Initial current I_c

You can specify an initial current flowing in the GTO thyristor. It is usually set to zero in order to start the simulation with the device blocked.

You can specify an initial current I_c value corresponding to a particular state of the circuit. In such a case all states of the linear circuit must be set accordingly. Initializing all states of a power-electronic converters is a complex task. Therefore, this option is useful only with simple circuits.

Snubber resistance R_s

The snubber resistance, in ohms (Ω). Set the **Snubber resistance R_s** parameter to ∞ to eliminate the snubber from the model.

Snubber capacitance C_s

The snubber capacitance, in farads (F). Set the **Snubber capacitance C_s** parameter to 0 to eliminate the snubber, or to ∞ to get a purely resistive snubber.

Inputs and Outputs

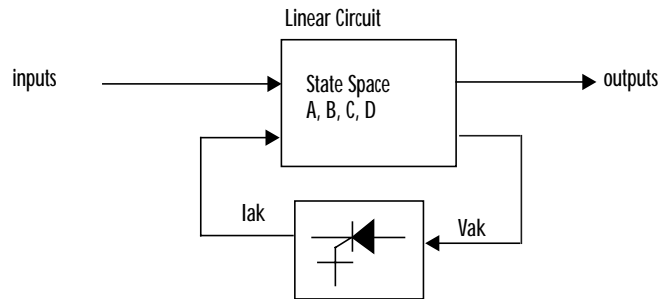
The first input and output are the GTO thyristor device terminals connected respectively to anode (a) and cathode (k). The second input (g) is a Simulink signal applied to the gate. The second output (m) is a Simulink measurement output vector [I_{ak} , V_{ak}] returning the GTO thyristor current and voltage.

Assumptions and Limitations

The GTO block implements a macro-model of a real GTO thyristor. It does not take into account either the geometry of the device or the underlying physical processes of the device [1].

The GTO block requires a continuous application of the gate signal ($g > 0$) in order to be in the on-state (with $I_{ak} > 0$). The latching current and the holding current are not considered. The critical value of the derivative of the re-applied anode-cathode voltage is not considered.

The GTO block is modeled as a nonlinear element interfaced with the linear circuit, as shown on the next figure.



Therefore, in order to avoid an algebraic loop, the GTO block inductance L_{on} cannot be set to zero. Each GTO block adds an extra state to the electrical circuit model. The GTO block is modeled as a current source. It cannot be connected in series with an inductor, a current source, or an open circuit, unless its snubber circuit is used.

Circuits containing individual GTO blocks cannot be discretized. However discretization is permitted for GTO/Diodes bridges simulated with the Universal Bridge block.

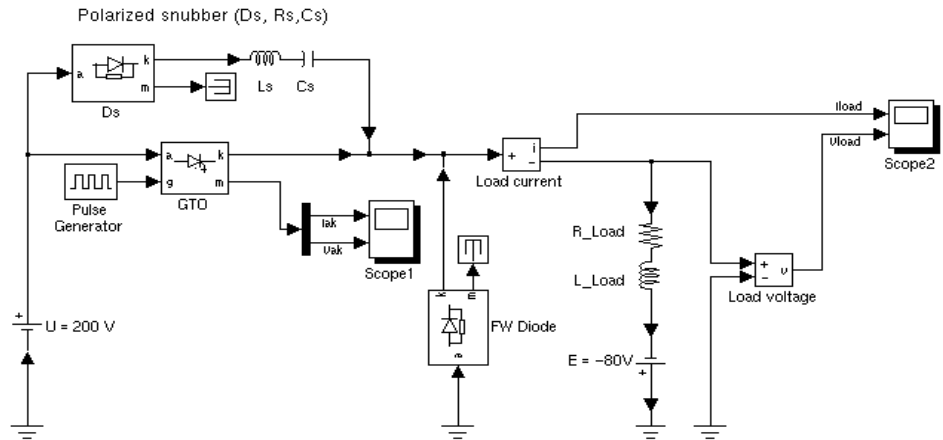
You must use a stiff integrator algorithm to simulate circuits containing GTO blocks. `ode23tb` or `ode15s` with default parameters usually give best simulation speed.


Example

The following example illustrates the use of the GTO block in a buck converter topology. The basic polarized snubber circuit is connected across the GTO block. The snubber circuit consists of a capacitor C_s , a resistor R_s and a diode D_s . The parasitic inductance L_s of the snubber circuit is also taken into consideration.

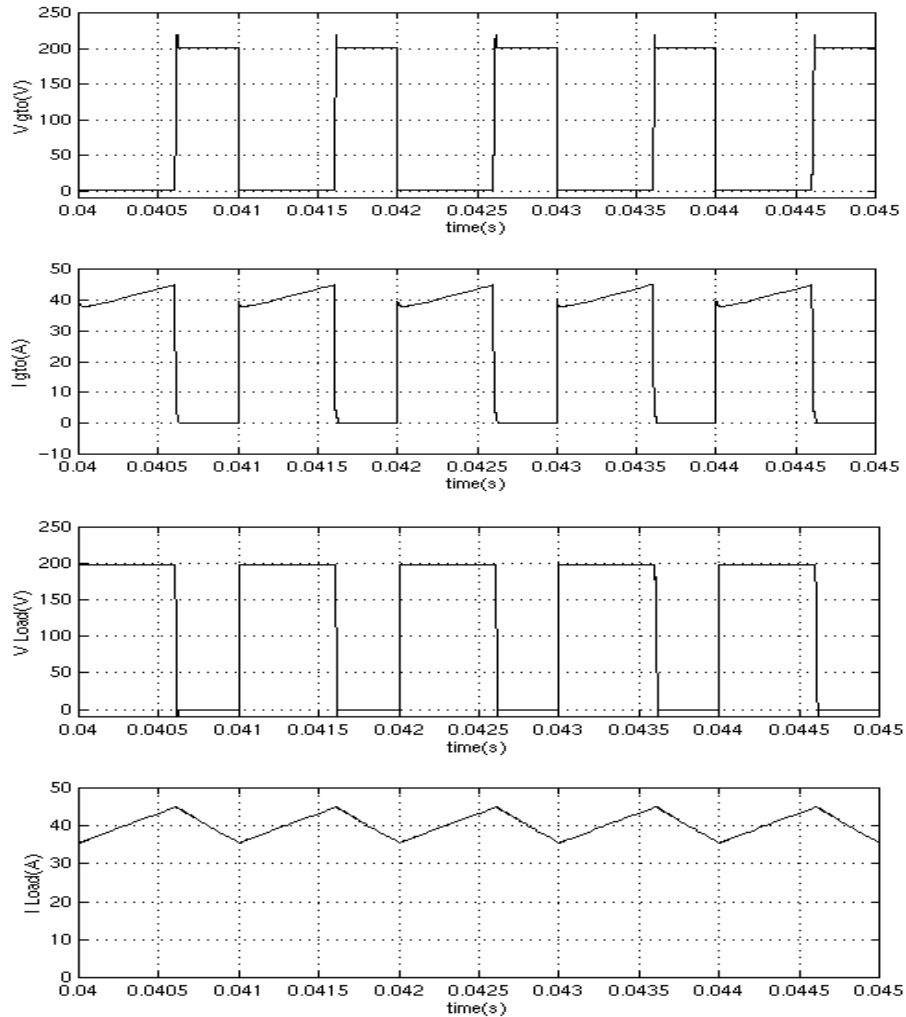
The parameters of the GTO block are those found in the Dialog Box section, except for the internal snubber which is out of service ($R_s = \infty$ $C_s = 0$).

The switching frequency is 1000 Hz and the pulse width is 216 degrees (duty cycle 60%). This example is available in the `psbbuckconv.mdl` file.



GTO Thyristor in the buck converter topology
 Double click on the More Info button (?) button for details  More Info

Run the simulation. Observe the GTO block voltage and current as well as, the load voltage and current.



References

[1] Mohan N., *Power Electronic, Converters, Applications and Design*, John Wiley & Sons, Inc., New York, 1995

See Also

Diode, Thyristor, Mosfet, Ideal Switch

Purpose

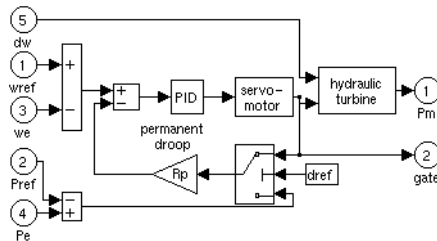
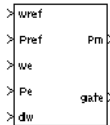
Model a hydraulic turbine and a PID governor system.

Library

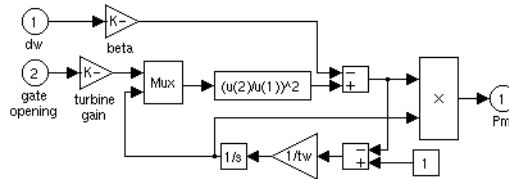
Machines

Description

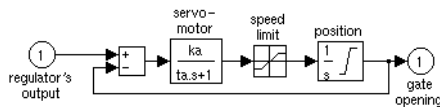
The Hydraulic Turbine and Governor block implements a nonlinear hydraulic turbine model, a PID governor system, and a servo-motor [1].



The hydraulic turbine is modeled by the following nonlinear system.



The gate servo-motor is modeled by a second-order system.



Hydraulic Turbine and Governor

Dialog Box and Parameters

Block Parameters: HTG

Hydraulic Turbine and Governor (mask) (link)

Implements a hydraulic turbine combined to a PID governor system.

1st input: desired speed (p.u.);
2nd input: desired electrical power (p.u.);
3rd input: synchronous machine's actual speed (p.u., measurement output 13 of SM block);
4th input: synchronous machine's actual electrical power (p.u., measurement output 14 of SM block);
5th input: synchronous machine's actual speed deviation with respect to nominal (% , measurement output 15 of SM block);

1st output: mechanical power to be applied to the Synchronous Machine block's 1st input (p.u.);
2nd output: gate opening (p.u.).

Parameters

Servo-motor [Ka() Ta(sec)]:
[10/3 0.07]

Gate opening limits [gmin,gmax(pu) vgmin,vgmax(pu/s)]:
[0.01 0.97518 -0.1 0.1]

Permanent droop and regulator [Rp() Kp() Ki() Kd() Td(s)]:
[0.05 1.163 0.105 0 0.01]

Hydraulic turbine [beta() Tw(sec)]:
[0 2.67]

Droop reference [0=power error, 1=gate opening]:
0

Initial mechanical power (pu):
0.7516

OK

Cancel

Help

Apply

Servo-motor

The gain K_a and time constant T_a , in seconds (s), of the first-order system representing the servo-motor.

Gate opening limits

The limits g_{min} and g_{max} (p.u.) imposed on the gate opening and vg_{min} and vg_{max} (p.u./s) imposed on gate speed.

Permanent droop and regulator

The static gain of the governor is equal to the inverse of the permanent droop R_p in the feedback loop. The PID regulator has a proportional gain K_p , an integral gain K_i and a derivative gain K_d . The high frequency gain of the PID is limited by a first-order low-pass filter with time constant T_d (s).

4-70

Hydraulic turbine

The speed deviation damping coefficient β and water starting time T_w (s).

Droop reference

Specifies the input of the feedback loop: gate position (set to one) or electrical power deviation (set to zero).

Initial mechanical power

The initial mechanical power P_{m0} (p.u.) at machine's shaft. This value can be computed by the load flow utility of the Powergui block.

Inputs and Outputs

The first two inputs are the desired speed and mechanical power. The third and fourth inputs are the machine's actual speed and electrical power. The fifth input is the speed deviation. Inputs 2 and 4 can be left unconnected if one wishes to use the gate position as input of the feedback loop instead of the power deviation. All inputs are in p.u. The outputs of the block are mechanical power P_m for the Synchronous Machine block and gate opening (both in p.u.).

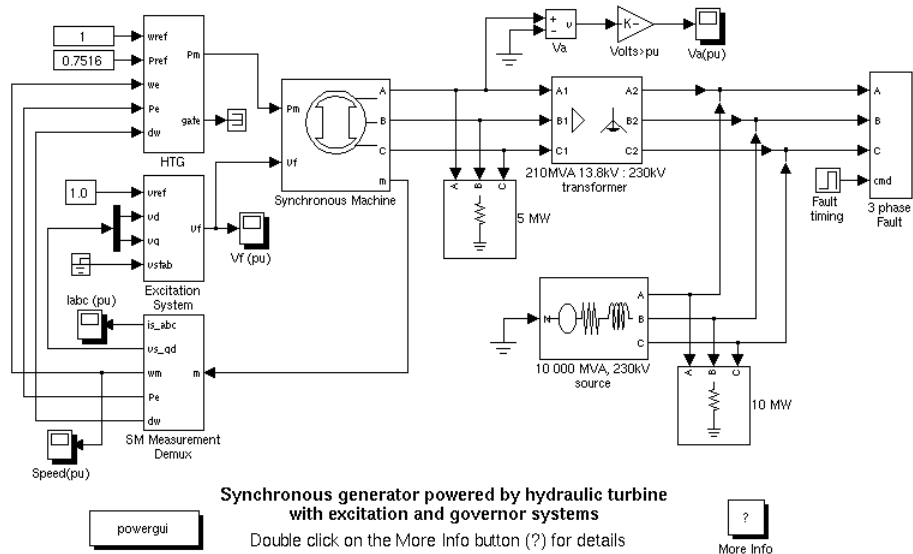
Example

This demonstration illustrates the use of the Synchronous Machine associated with the Hydraulic Turbine and Governor (HTG) and Excitation System blocks. It also demonstrates the use of the Load Flow to initialize machine currents. A three-phase generator rated 200 MVA, 13.8 kV, 112.5 rpm is connected to a 230 kV network through a Delta-Wye 210 MVA transformer. The network short-circuit level is 10000 VA and the transformer has a 0.16 p.u. leakage reactance. The system starts in steady-state with the generator supplying 150 MW of active power. At $t=0.1$ s, a three-phase to ground fault occurs on the 230 kV bus of the transformer. The fault is cleared after six cycles ($t=0.2$ s). Open the Simulink diagram by typing `psbturbi ne`.

In order to start the simulation in steady-state, you must initialize the Synchronous Machine block for the desired load flow. Open the **powergui** and select **Load flow and machine initialization** from the **Tools** menu. The machine Bus Type should be already initialized as "PV generator", indicating that the load flow will be performed with the machine controlling the active power and its terminal voltage. Specify the desired values by entering the following parameters:

- U_{AB} (Vrms) = 13800
- P (watts) = 150e6

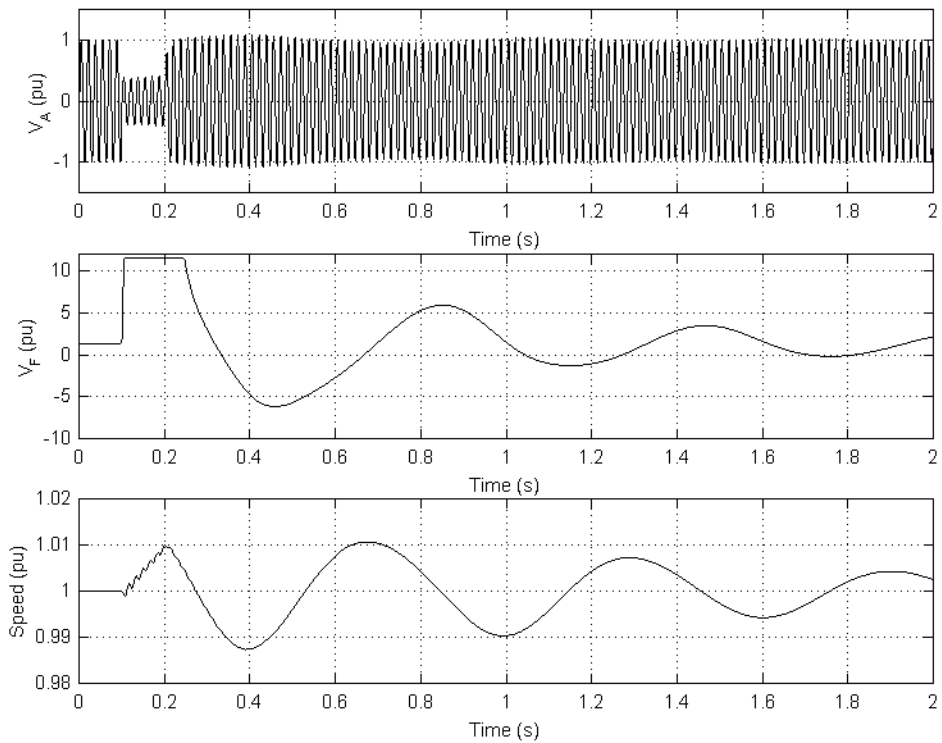
Hydraulic Turbine and Governor



Then press the **Do Load Flow** button. Once the load flow has been solved, the phasors of AB and BC machine voltages as well as the currents flowing out of phases A and B are updated. The machine reactive power, mechanical power, and field voltage requested to supply the electrical power should also be displayed:

- $Q = 3.4 \text{ Mvar}$
- $P_{mec} = 150.32 \text{ MW (0.7516 p.u.)}$
- Field voltage $E_f = 1.291 \text{ p.u.}$

In order to start the simulation in steady state with the HTG and Excitation System connected, these two Simulink blocks must also be initialized according to the values calculated by the load flow. Open the HTG block menu and notice that the initial mechanical power has been set to 0.5007 p.u. (100.14 MW). Then, open the Excitation System block menu and note that the initial terminal voltage and field voltage have been set respectively to 1.0 and 1.126 pu. Open the 4 scopes and start the simulation. The simulation starts in steady state.



Observe that the terminal voltage V_A is 1.0 p.u. at the beginning of the simulation. It falls to about 0.4 p.u. during the fault and returns to nominal quickly after the fault is cleared. This quick response in terminal voltage is due to the fact that the Excitation System output V_F can go as high as 11.5 p.u. which it does during the fault. The speed of the machine increases to 1.01 p.u. during the fault then it oscillates around 1 p.u. as the governor system regulates it. The speed takes much longer than the terminal voltage to stabilize mainly because the rate of valve opening/closing in the governor system is limited to 0.1 p.u./s.

References

[1] IEEE Working Group on Prime Mover and Energy Supply Models for System Dynamic Performance Studies, "Hydraulic Turbine and Turbine Control Models for Dynamic Studies", *IEEE Transactions on Power Systems*, vol. 7, no. 1, February 1992, pp. 167-179.

Hydraulic Turbine and Governor

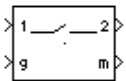
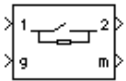
See Also

Excitation System, Powergui, Synchronous Machine, Steam Turbine and Governor

Purpose Implement an ideal switch device.

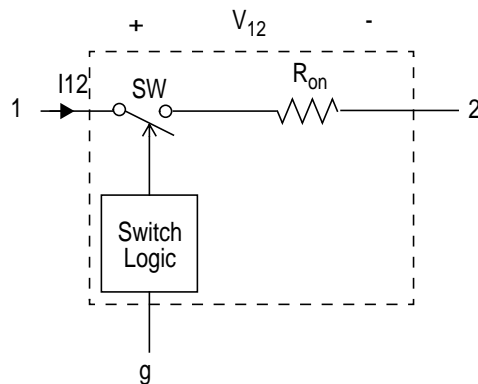
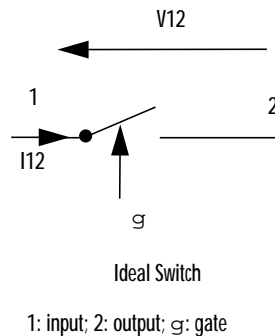
Library Power Electronics

Description



The Ideal Switch block does not correspond to a particular physical device. When used with appropriate switching logic, it can be used to model simplified semiconductor devices such as a GTO or a MOSFET, or even a power circuit breaker with current chopping. The switch is simulated as a resistor (R_{on}) in series with a switch controlled by a logical g .

The Ideal Switch block also contains a series Rs-Cs snubber circuit that can be connected in parallel with the ideal switch.

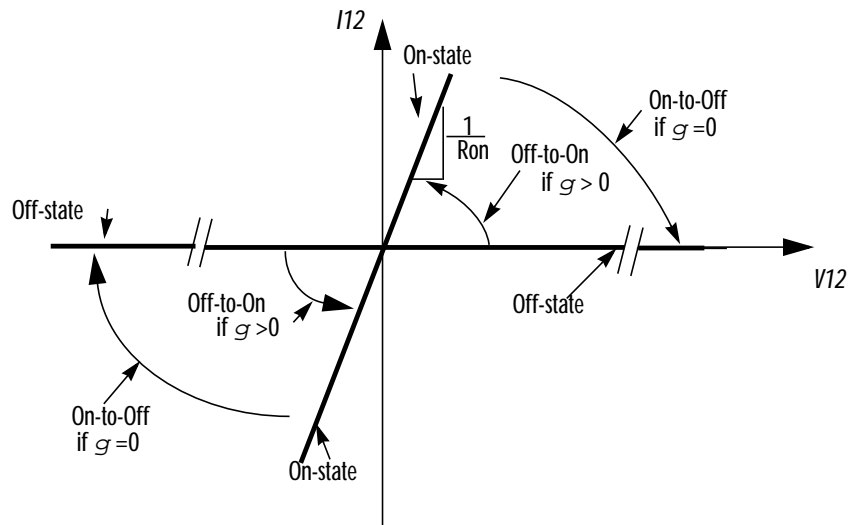


The Ideal Switch block is fully controlled by the gate signal ($g>0$ or $g=0$). It has the following characteristics:

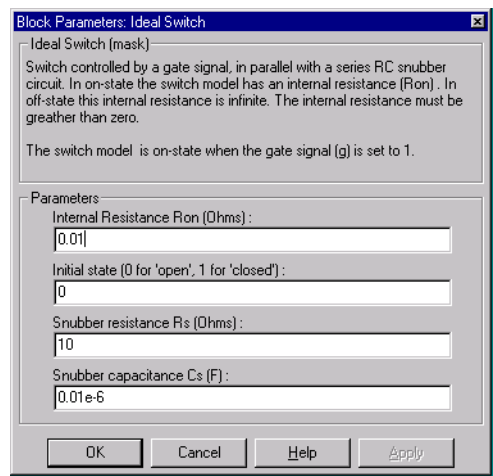
- Blocks any forward or reverse applied voltage with zero current flow when $g=0$
- Conducts any bidirectional current with quasi zero voltage drop when $g>0$
- Switches instantaneously between on and off state when triggered

The Ideal Switch block turns on when a positive signal is present at the gate input ($g>0$). It turns off when the gate signal equals zero ($g=0$).

Ideal Switch



Dialog Box and Parameters



Internal Resistance R_{on}

The internal resistance of the switch device, in ohms (Ω). The **Internal resistance R_{on}** parameter cannot be set to zero.

Initial state

The initial state of the Ideal Switch block. If the **Initial state** parameter is set to 1 (closed) the states of the linear circuit are automatically initialized so that the simulation starts in steady state.

Snubber resistance R_s

The snubber resistance, in ohms (Ω). Set the **Snubber resistance R_s** parameter to `inf` to eliminate the snubber from the model.

Snubber capacitance C_s

The snubber capacitance in farads (F). Set the **Snubber capacitance C_s** parameter to 0 to eliminate the snubber, or to `inf` to get a purely resistive snubber.

Inputs and Outputs

The first input (1) and output (2) of the block are the ideal switch electrical connections. The second input (g) is a Simulink signal applied to the gate. The second output (m) is a Simulink measurement output vector $[I_{ak}, V_{ak}]$ returning the Ideal Switch block current and voltage.

Assumptions and Limitations

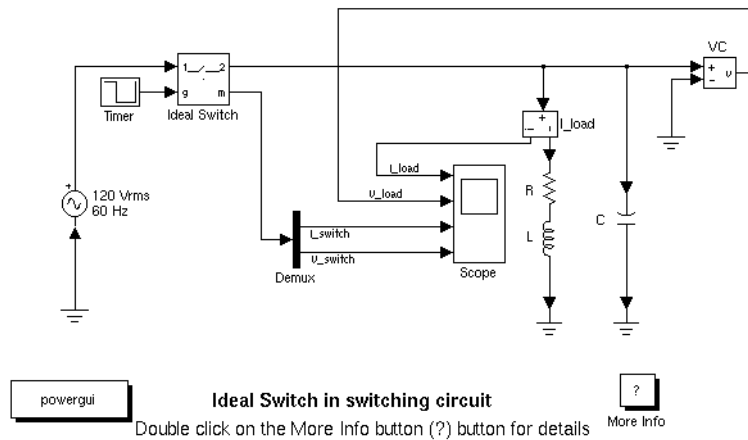
The Ideal Switch block is modeled as a current source, therefore it cannot be connected in series with an inductor, a current source or an open circuit, unless a snubber circuit is used.

You must use a stiff integrator algorithm to simulate circuits containing Ideal Switch blocks. `Ode23tb` or `Ode15s` with default parameters usually give best simulation speed.

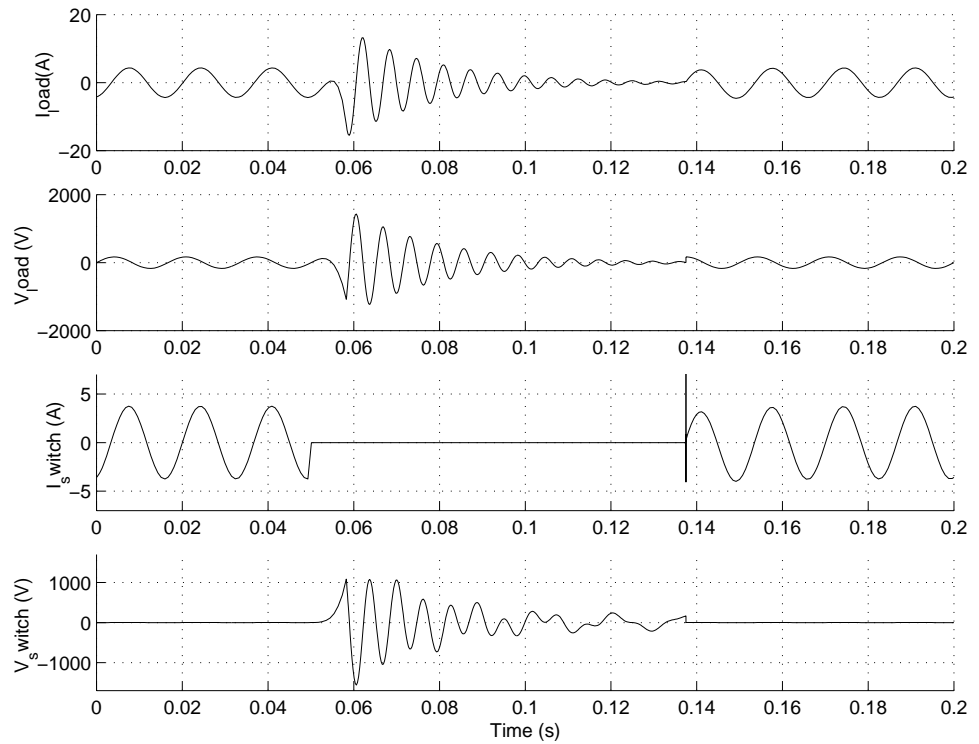
Example

An Ideal Switch block is used to switch an RLC circuit on an AC source (60 Hz). The switch, which is initially closed, is first open at $t=50$ ms (3 cycles) and then reclosed at $t=138$ ms (8.25 cycles). The Ideal Switch block has a 0.01 ohms resistance and no snubber is used. This example is available in the `psbswitch.mdl` file.

Ideal Switch



Run the simulation and observe the inductor current, the switch current and the capacitor voltage. Notice the high frequency overvoltage produced by inductive current chopping. Note also the high switch current spike when the switch is reclosed on the capacitor at maximum source voltage.



See Also

Breaker

References

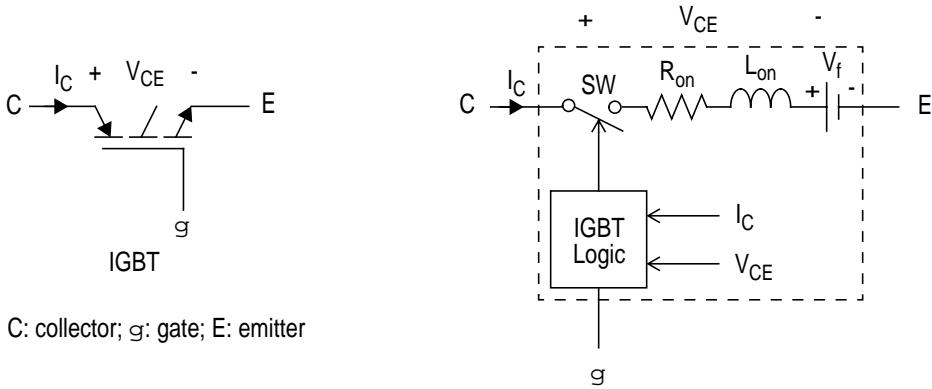
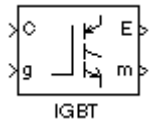
Mohan, N., *Power Electronic, Converters, Applications and Design*, John Wiley & Sons, Inc., New York, 1995.

IGBT

Purpose Implement an Insulated-Gate-Bipolar-Transistor (IGBT).

Library Power Electronics

Description The IGBT block implements a semiconductor device controllable by the gate signal. The IGBT is simulated as a series combination of a resistor (R_{on}), and inductor (L_{on}), and a DC voltage source (V_f) in series with a switch controlled by a logical signal ($g>0$ or $g=0$).

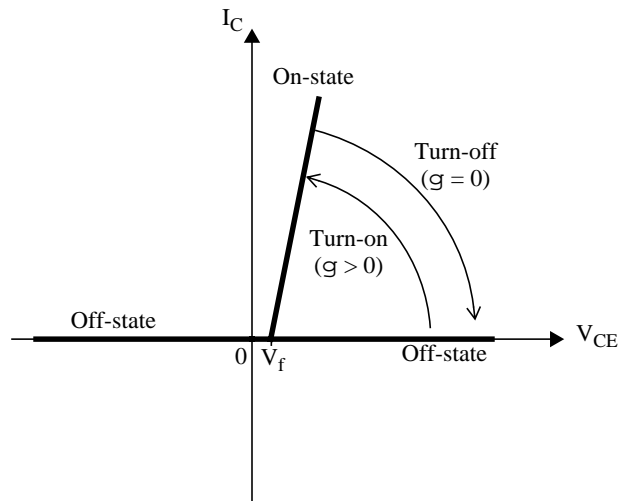


C: collector; g: gate; E: emitter

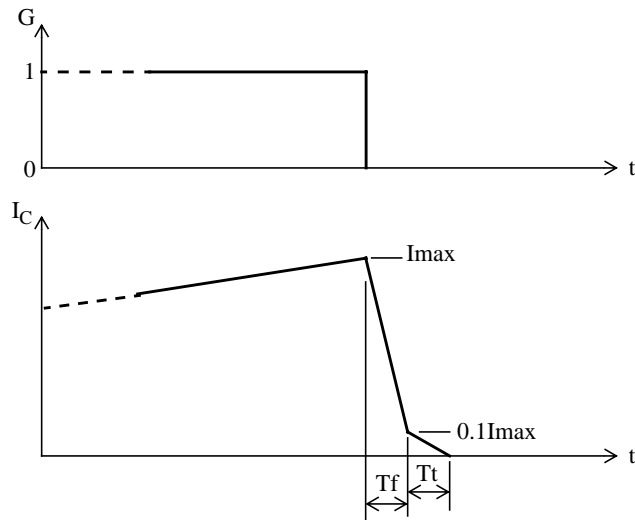
The IGBT turns on when the collector-emitter voltage is positive and greater than V_f and a positive signal is applied at the gate input ($g > 0$). It turns off when the collector-emitter voltage is positive and a zero signal is applied at the gate input ($g = 0$).

The IGBT device is in off-state when the collector-emitter voltage is negative. Note that many commercial IGBTs do not have the reverse blocking capability. Therefore, they are usually used with an anti-parallel diode.

The IGBT block contains a series Rs-Cs snubber circuit, which is connected in parallel with the IGBT device.



The turn-off characteristic of the IGBT model is approximated by two segments. When the gate signal falls to zero, the collector current will decrease from I_{max} to $0.1I_{max}$ during the fall time (T_f), and then from $0.1I_{max}$ to zero during the tail time (T_t).



IGBT

Dialog Box and Parameters

Block Parameters: IGBT

IGBT (mask) (link)

IGBT in parallel with a series RC snubber circuit. In on-state the IGBT model has internal resistance (Ron) and inductance (Lon). In off-state the IGBT model has infinite impedance. The internal inductance cannot be set to zero.
Discretization of the IGBT is available only through the Universal Bridge block.

Parameters

Resistance Ron (Ohms) :

0.01

Inductance Lon (H) :

1e-6

Forward voltage Vf (V) :

1

Current 10% fall time Tf (s) :

1e-6

Current tail time Tt (s) :

1e-6

Initial current Ic (A) :

0

Snubber resistance Rs (Ohms) :

100

Snubber capacitance Cs (F) :

0.1e-6

OK

Cancel

Help

Apply

Resistance Ron

The internal resistance Ron, in ohms (Ω).

Inductance Lon

The internal inductance Lon, in henries (H). The **Inductance Lon** parameter cannot be set to 0

Forward voltage Vf

The forward voltage of the IGBT device, in volts (v).

Current 10% fall time

The current fall time Tf, in seconds (s).

Current tail time

The current tail time Tt, in seconds (s).

Initial current I_c

You can specify an initial current flowing in the IGBT. It is usually set to zero in order to start the simulation with the device blocked.

You can specify an initial current I_c value corresponding to a particular state of the circuit. In such a case all states of the linear circuit must be set accordingly. Initializing all states of a power-electronic converters is a complex task. Therefore, this option is useful only with simple circuits.

Snubber resistance R_s

The snubber resistance, in ohms (Ω). Set the **Snubber resistance R_s** parameter to ∞ to eliminate the snubber from the model.

Snubber capacitance C_s

The snubber capacitance in farads (F). Set the **Snubber capacitance C_s** parameter to 0 to eliminate the snubber, or to ∞ to get a purely resistive snubber.

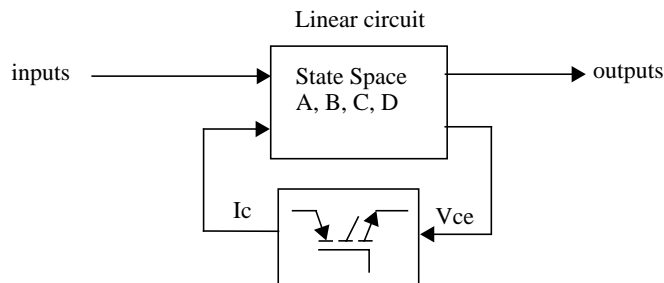
Inputs and Outputs

The first input and output are the IGBT terminals connected respectively to collector (c) and emitter (e). The second input (g) is a logical Simulink signal applied to the gate. The second output is a Simulink measurement vector [I_c , V_{ce}] returning the IGBT current and voltage.

Assumptions and Limitations

The IGBT block implements a macro-model of the real IGBT device. It does not take into account either the geometry of the device or the complex physical processes [1].

The IGBT is modeled as a nonlinear element interfaced with the linear circuit as shown below.



Therefore, in order to avoid an algebraic loop, the IGBT inductance L_{on} cannot be set to zero. Each IGBT adds an extra state to the electrical circuit model. As the IGBT is modeled as a current source, it cannot be connected in series with an inductor, a current source, or an open circuit, unless a snubber circuit is used.

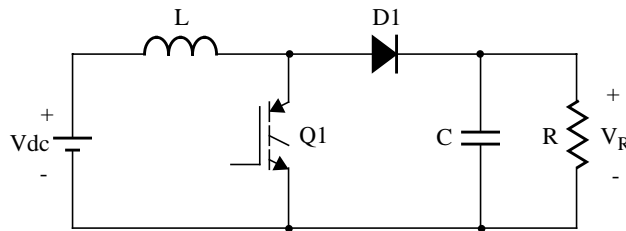
Circuits containing individual IGBT blocks cannot be discretized. However discretization is permitted for IGBT/Diodes bridges simulated with the Universal Bridge block.

You must use a stiff integrator algorithm to simulate circuits containing IGBTs. Ode23tb or Ode15s with default parameters usually give best simulation speed.

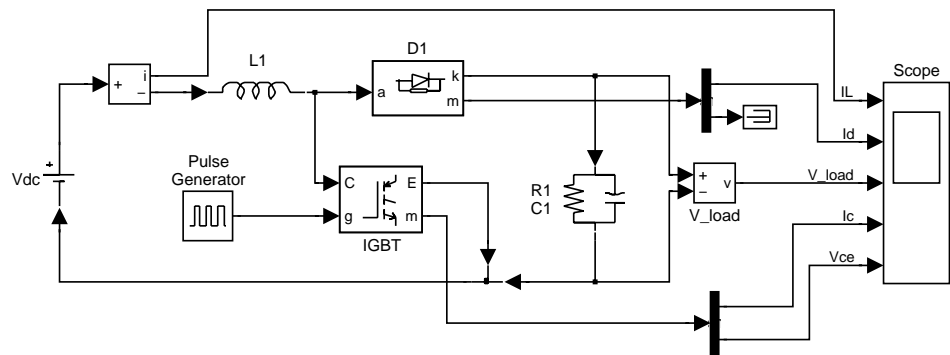
Example

The following example illustrates the use of the IGBT block in a boost dc-dc converter. The IGBT is switched on and off at a frequency of 10 kHz to transfer energy from the DC source to the load (RC). The average output voltage (V_R) is a function of the duty cycle (α) of the IGBT switch:

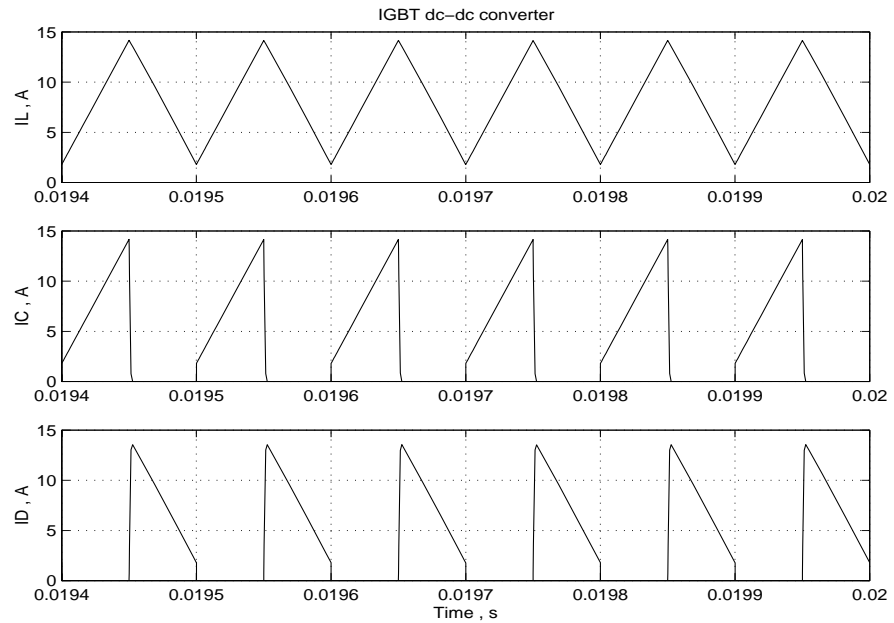
$$V_R = \frac{1}{1 - \alpha} V_{dc}$$

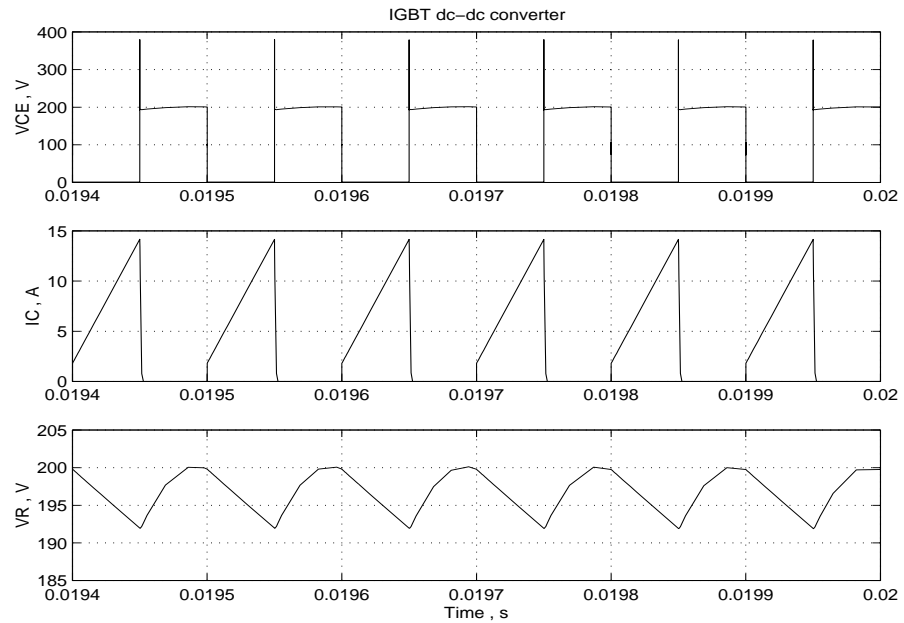


This example is available in the `psbi gbt conv. mdl` file.



Run the simulation and observe the inductor current (I_L), the IGBT collector current (I_C), the diode current (I_D), the IGBT device collector-emitter voltage (V_{CE}), and the load voltage (V_R).





References

[1] Mohan, N., *Power Electronic, Converters, Applications and Design*. John Wiley & Sons, Inc., New York, 1995.

See Also

GTO, Thyristor, MOSFET, Universal Bridge

Purpose

Measure the impedance of a circuit as a function of frequency.

Description

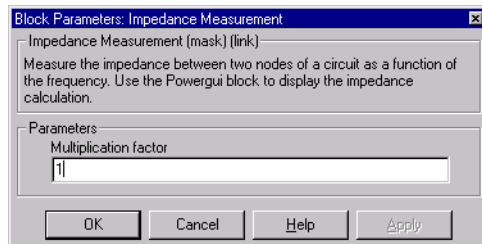


The Impedance Measurement block measures the impedance between two nodes of a circuit, as a function of the frequency. It consists of a current source I_z connected between inputs one and two of the Impedance Measurement block and a voltage measurement V_z connected across the terminals of the current source. The network impedance is calculated as the transfer function $H(s)$ from the current input to the voltage output of the state-space model.

$$H(s) = \frac{V_z(s)}{I_z(s)}$$

The measurement takes into account the initial status of the Breaker and Ideal Switch blocks. It also allows impedance measurements with Distributed Parameter Line blocks in your circuit.

Dialog Box and Parameter



Multiplication factor

If you plan to use the Impedance Measurement block in a three-phase circuit you can use the **Multiplication factor** parameter to rescale the measured impedance. For example, measuring the impedance between two phases of a three-phase circuit will give two times the positive-sequence impedance. Therefore you must apply a multiplication factor of 1/2 on the impedance in order to obtain the correct positive-sequence impedance value.

Similarly, to measure the zero-sequence impedance of a balanced three-phase circuit, you can connect the Impedance Measurement block between ground or neutral and the three phases connected together.

Impedance Measurement

In that case, you are measuring one third of the zero-sequence impedance and you must apply a multiplication factor of 3 to obtain the correct zero-sequence value.

Limitations

The only nonlinear blocks that are taken into account during the impedance measurement are the Breaker block, the Ideal Switch block, and the Distributed Parameter Line block. All other nonlinear blocks such as machines and power electronic devices are not considered and they will be disconnected during the measurement.

If you plan to connect the Impedance Measurement block in series with an inductance, a current source, or any nonlinear element, you must add a large resistor across the terminals of the block. The reason is that the Impedance Measurement block is simulated as a current source block.

Example

See the Powergui block reference page for an example using the Impedance Measurement block.

See Also

Powergui

Purpose

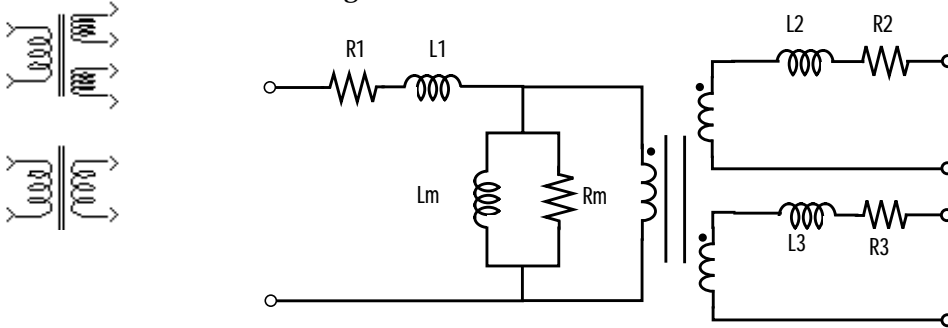
Implement a two winding or three winding linear transformer.

Library

Elements

Description

The Linear Transformer block model shown below consists of three coupled windings wound on the same core.



The model takes into account the winding resistances (R_1 R_2 R_3) the leakage inductances (L_1 L_2 L_3) as well as the magnetizing characteristics of the core which is modeled by a linear (R_m L_m) branch.

The Per Unit Conversion

In order to comply with the industry practice, you must specify the resistance and inductance of the windings in per unit (pu). The values are based on the transformer rated power P_n , in VA, nominal frequency f_n in Hz, and nominal voltage V_n , in Vrms, of the corresponding winding. For each winding the per unit resistance and inductance are defined as:

$$R(\text{p.u.}) = \frac{R(\Omega)}{R_{base}}$$

$$L(\text{p.u.}) = \frac{L(H)}{L_{base}}$$

The base resistance and base inductance used for each winding are:

$$R_{base} = \frac{(V_n)^2}{P_n}$$

$$L_{base} = \frac{R_{base}}{2\pi f_n}$$

Linear Transformer

For the magnetization resistance R_m and inductance L_m , the p.u. values are based on the transformer rated power and on nominal voltage of the winding one.

For example, the default parameters of winding one specified in the dialog box section give the following bases:

$$R_{base} = \frac{(735e3/\sqrt{3})^2}{250e6} = 720.3\Omega \quad L_{base} = \frac{720.3}{2\pi 60} = 1.91H$$

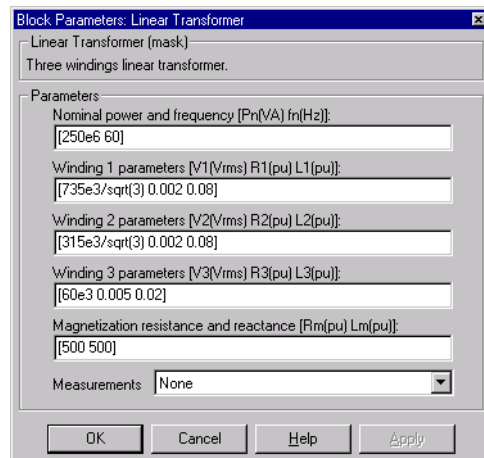
Suppose that the winding 1 parameters are $R_1=1.44\Omega$ and $L_1=0.1528H$, the corresponding values to be entered in the dialog box are:

$$R_1 = \frac{1.44\Omega}{720.3\Omega} = 0.002\text{p.u.}$$

$$L_1 = \frac{0.1528H}{1.91H} = 0.08\text{p.u.}$$

To specify a magnetizing current of 0.2% (resistive and inductive) based on nominal current you must enter per unit values of $1/0.002 = 500$ p.u. for the resistance and the inductance of the magnetizing branch. Using the base values calculated above these per unit values correspond to $R_m = 8.6e5$ ohms and $L_m = 995$ henries.

Dialog Box and Parameters



Nominal power and frequency

The nominal power rating P_n in volt amperes (V.A) and frequency f_n , in hertz (Hz), of the transformer.

Winding 1 parameters

The nominal voltage V_1 in volts rms, resistance, and leakage inductance in p.u. The p.u. values are based on the nominal power P_n and on V_1 .

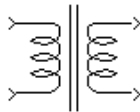
Winding 2 parameters

The nominal voltage V_2 in volts rms, resistance, and leakage inductance in p.u. The p.u. values are based on the nominal power P_n and on V_2 .

Winding 3 parameters

The nominal voltage in volts rms (V_{rms}), resistance, and leakage inductance in p.u. The p.u. values are based on the nominal power P_n and on V_3 .

Setting the **Winding 3 parameters** parameter to 0 implements a Linear Transformer block with two windings and a new icon will be displayed:



Magnetization resistance and reactance

The resistance and inductance simulating the core active and reactive losses, both in p.u. The p.u. values are based on the nominal power P_n and on V_1 . For example, to specify 0.2% of active and reactive core losses, at nominal voltage, use $R_m = 500$ p.u. and $L_m = 500$ p.u.

Measurements

Select **Winding voltages** to measure the voltage across the winding terminals of the Linear Transformer block.

Select **Winding currents** to measure the current flowing through the windings of the Linear Transformer block.

Select **Magnetization current** to measure the magnetization current of the Linear Transformer block.

Linear Transformer

Select **All voltages and currents** to measure the winding voltages and currents plus the magnetization current.

Place a Multimeter block in your model to display the selected measurements during the simulation.

In the **Available Measurement** listbox of the Multimeter block, the measurements will identified by a label followed by the block name.

Measurement	Label
Winding voltages	u w1: , u w2: , u w3:
Winding currents	i w1: , i w2: , i w3:
Magnetization current	i mag:

Note To implement an ideal transformer model, set the winding resistances and inductances to zero, and the magnetization resistance and inductance to inf.

Inputs and Outputs

The winding 1 is connected between input one and input two of the Linear Transformer block. The winding 2 is connected between output one and output two, and the winding 3 (if defined) is connected between output three and output four.

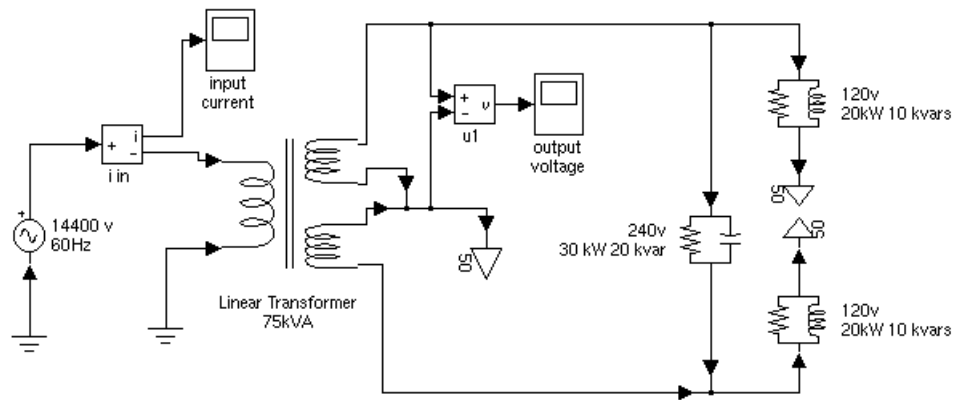
Input one, output one and output three (if exist) are at the same instantaneous polarity.

Limitations

Windings can be left floating (i.e., not connected to the rest of the circuit). However the floating winding will be connected internally to the main circuit through a resistor. This internal connection does not affect voltage and current measurements.

Example

Typical residential distribution transformer network feeding line to neutral and line to line loads. This circuit is available in the psbtransformer.mdl file.



See Also

Saturable Transformer, Mutual Inductance

MOSFET

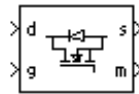
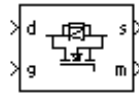
Purpose

Implement a MOSFET model.

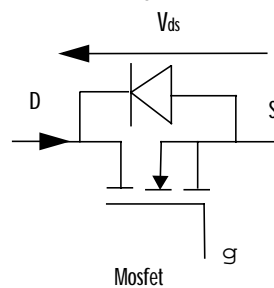
Library

Power Electronics

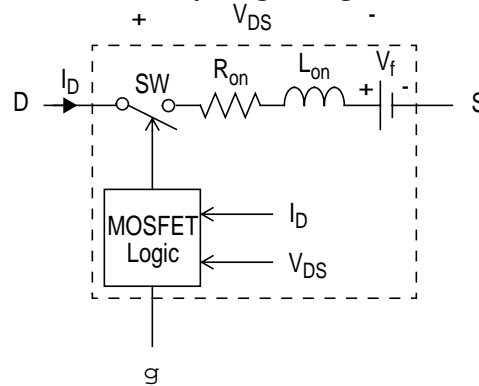
Description



The Metal-Oxide-Semiconductor-Field-Effect-Transistor (MOSFET) is a semiconductor device controllable by the gate signal ($g > 0$) if its current I_d is positive ($I_d > 0$). The MOSFET device is connected in parallel with an internal diode which turns on when the MOSFET device is reverse biased ($V_{ds} < 0$). The model is simulated as a series combination of a variable resistor (R_t) and inductor (L_{on}) in series with a switch controlled by a logical signal ($G > 0$ or $g = 0$).



D: drain; S: source; g: gate



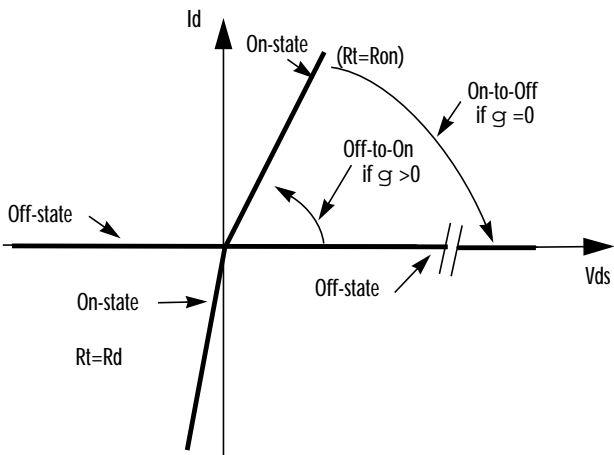
The MOSFET device turns on when the drain-source voltage is positive and a positive signal is applied at the gate input ($g > 0$).

With a positive current flowing through the device, the MOSFET turns off when the gate input becomes zero. If the current I_d is negative (I_d flowing in the internal diode) and without a gate signal ($g = 0$), the MOSFET turns off when the current I_d becomes zero ($I_d = 0$).

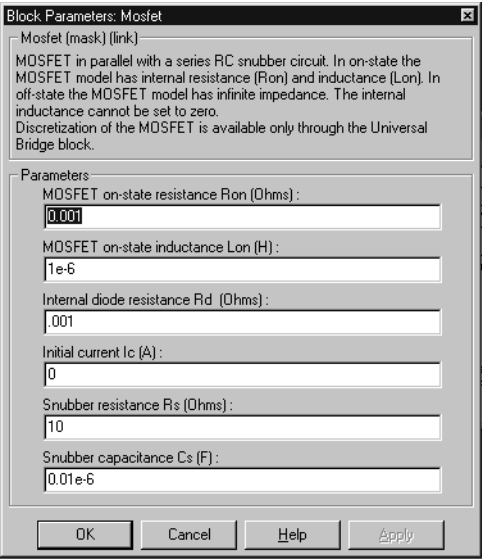
Note that the on-state resistance R_t depends on the drain current direction:

- $R_t = R_{on}$ if $I_d > 0$, where R_{on} represents the typical value of the forward conducting resistance of the MOSFET device.
- $R_t = R_d$ if $I_d < 0$, where R_d represents the internal diode resistance

The MOSFET block also contains a series R_s - C_s snubber circuit that can be connected in parallel with the MOSFET.



Dialog Box and Parameters



Resistance R_{on}

The internal resistance R_{on} , in ohms (Ω).

Inductance Lon

The internal inductance Lon, in henries (H). The **Inductance Lon** parameter cannot be set to 0.

Internal diode resistance Rd

The internal resistance of the internal diode, in ohms (Ω).

Initial current Ic

You can specify an initial current flowing in the MOSFET device. It is usually set to zero in order to start the simulation with the device blocked.

You can specify an initial current Ic value corresponding to a particular state of the circuit. In such a case all states of the linear circuit must be set accordingly. Initializing all states of a power-electronic converters is a complex task. Therefore, this option is useful only with simple circuits.

Snubber resistance Rs

The snubber resistance, in ohms (Ω). Set the **Snubber resistance Rs** parameter to `inf` to eliminate the snubber from the model.

Snubber capacitance Cs

The snubber capacitance, in amperes (A). Set the **Snubber capacitance Cs** parameter to 0 to eliminate the snubber, or to `inf` to get a purely resistive snubber.

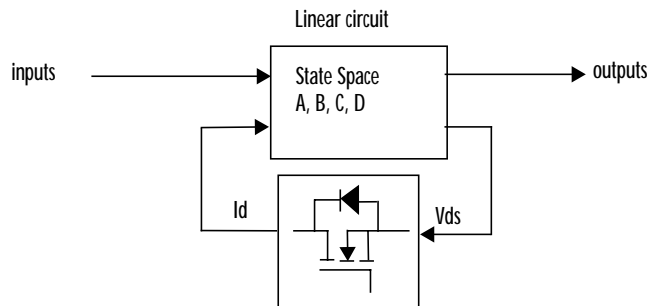
Inputs and Outputs

The first input and output are the MOSFET connections to drain (d) and source (s). The second input is a logical Simulink signal applied to the gate. The second output is a Simulink measurement vector [Id, Vds] returning the MOSFET device current and voltage.

Assumptions and Limitations

The MOSFET block implements a macro-model of the real MOSFET device. It does not take into account either the geometry of the device or the complex physical processes [1].

The MOSFET block is modeled as a nonlinear element interfaced with the linear circuit as shown on the next figure.



Therefore, in order to avoid an algebraic loop, the inductance L_{on} cannot be set to zero. The MOSFET block adds an extra state to the electrical circuit model. As the Mosfet is modeled as a current source, it cannot be connected in series with an inductor, a current source, or an open circuit, unless its snubber circuit is used.

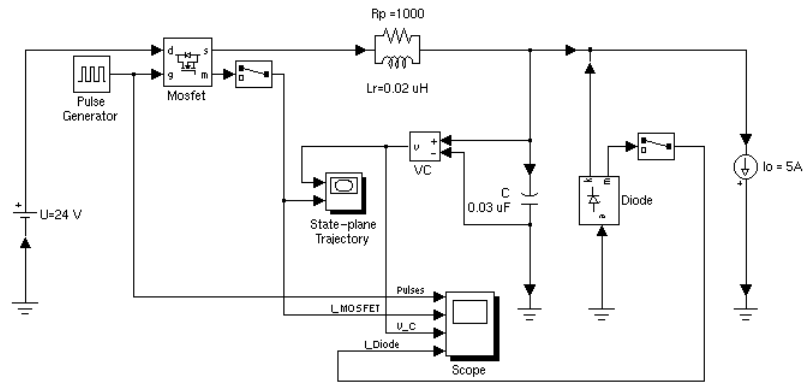
Circuits containing individual MOSFET blocks cannot be discretized. However discretization is permitted for MOSFET/Diodes bridges simulated with the Universal Bridge block.

You must use a stiff integrator algorithm to simulate circuits containing MOSFETs. Ode23tb or Ode15s with default parameters usually gives best simulation speed.

Example

The following example illustrates the use of the MOSFET block in a zero-current-quasi-resonant switch converter. In such a converter, the current produced by the Lr-Cr resonant circuit flows through the MOSFET and internal diode. The negative current flows through the internal diode that turns off at zero current [1]. The switching frequency is 2 MHz and the pulse width is 72 degrees (duty cycle: 20%). This example is available in the `psbmosconv.mdl` file.

MOSFET

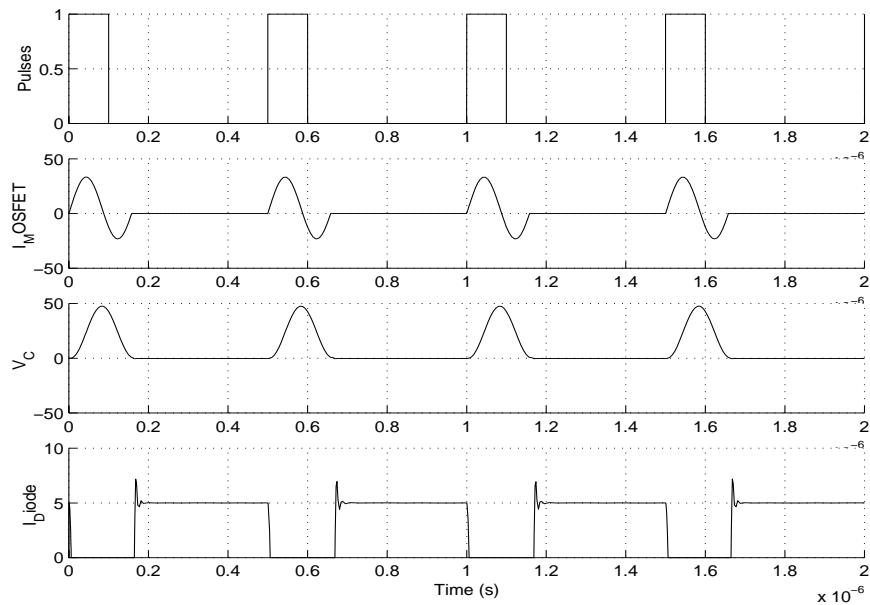


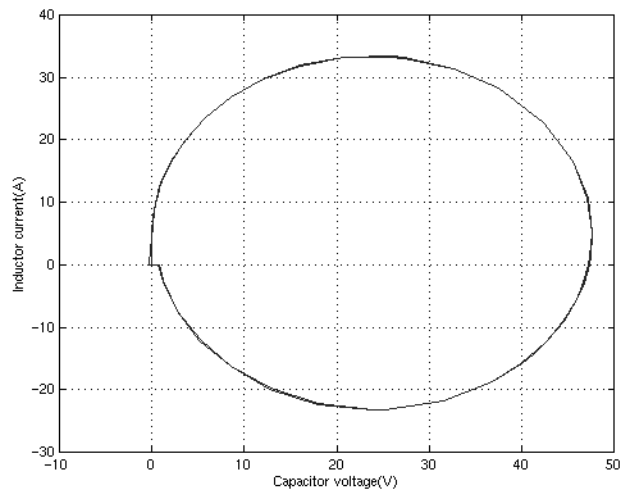
MOSFET in Zero-Current-Quasi-Resonant Converter

Double click on the More Info button (?) for details



Run the simulation and observe the gate pulse signal, the MOSFET current, the capacitor voltage and the diode current on the four trace scope block. Also observe the state-plane trajectory (inductor current versus capacitor voltage).





References

[1] Mohan N., *Power Electronic, Converters, Applications and Design*, John Wiley & Sons, Inc., New York, 1995.

See Also

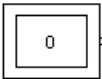
Diode, GTO, Ideal Switch, Thyristor

Multimeter

Purpose Measure the voltages and currents specified in dialog box of Power System Blockset blocks.

Library Measurements

Description The Multimeter block is used to measure voltages and currents of the measurements described by the dialog box of Power System Blockset blocks.



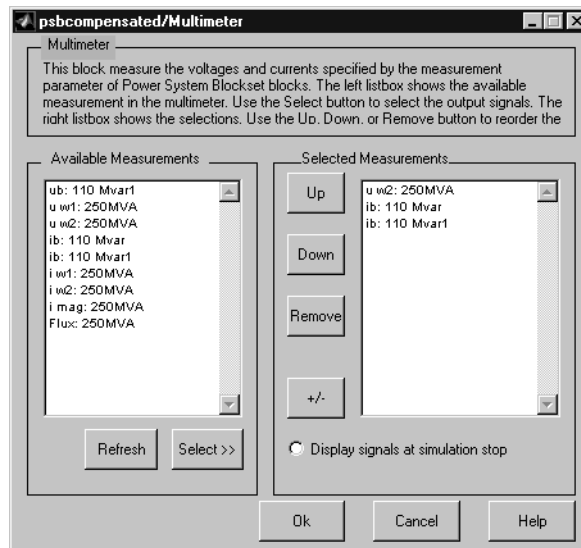
The powerlib blocks listed in Table 4-7 have a special parameter (Measurements) that allow you to measure voltages or currents related to the block. Choosing voltages or currents through this measurement parameter is equivalent to connecting an internal voltage or current measurement block inside your blocks. The measured signals can be observed through a Multimeter block placed in your circuit.

Drag the Multimeter block in the top-level system of your circuit and double-click on the icon to open the graphical user interface (GUI).

Table 4-7: Power System Blockset Blocks with Internal Measurements

Block Name	Block Name
AC Current Source	Parallel RLC Branch
AC Voltage Source	Parallel RLC Load
Controlled Current Source	PI Section Line
Controlled Voltage Source	Saturable Transformer
DC Voltage Source	Series RLC Branch
Breaker	Series RLC Load
Distributed Parameter Line	Surge Arrester
Linear Transformer	Three-Phase Transformer (Two and Three Windings)
Mutual Inductance	Universal Bridge

Dialog Box and Parameters



Available Measurements

The **Available Measurement** listbox shows the measurements in the Multimeter block. Use the **Select** button to select measurements from the **Available Measurements** listbox. Use the **Refresh** button to update the list of available measurements in the Multimeter block.

The measurements in the list box are identified by the name of the block where the measurement is done. The type of measurement (voltage measurement, current measurement, or flux) is defined by a label preceding the block name. See the reference section of blocks listed in Table 4-7 for a description of these measurements.

Selected Measurement

The **Selected Measurements** listbox shows the measurements sent to the output of the block. You can order the measurements by using the **Up**, **Down**, and **Remove** buttons. The **+/-** button allow you to reverse the polarity of any selected measurement.

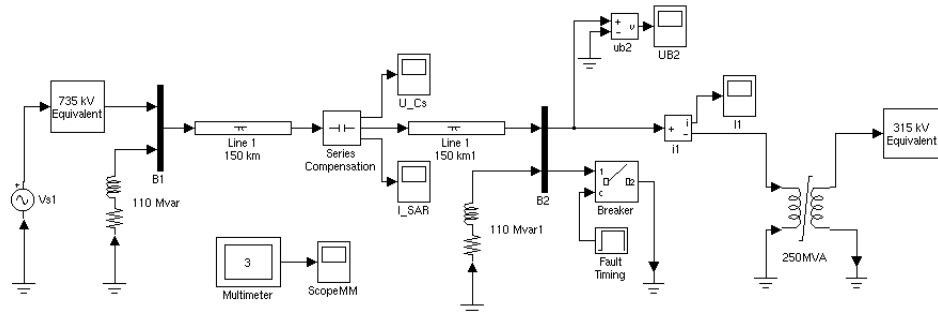
Display signals at simulation stop

If checked, displays a plot of selected measurements using a MATLAB figure window. The plot is generated when the simulation stops.

Multimeter

Example

The following circuit uses a Multimeter block to measure the voltage at the secondary winding of a Saturable Transformer block and the currents flowing through two Series RLC Load block.

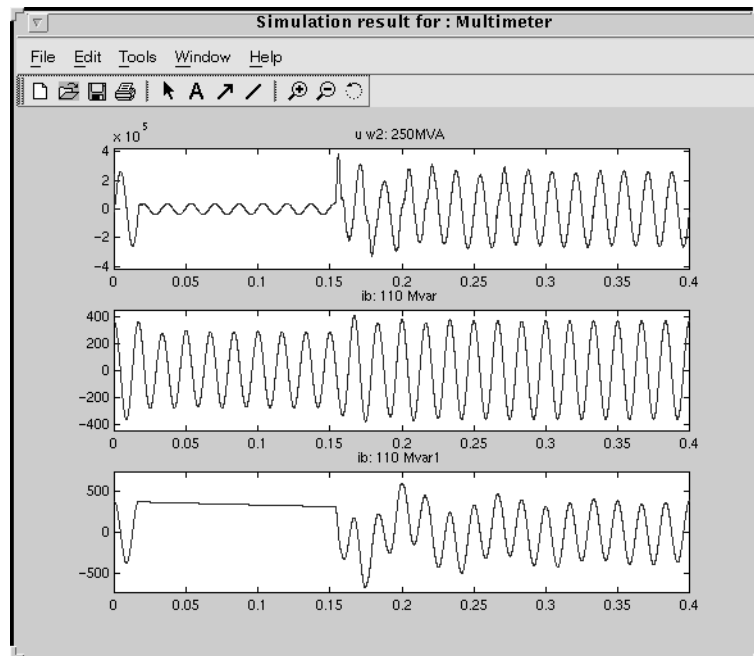


The circuit is available in the `psbcompensated.mdl` demo. A Multimeter block is dragged into the model. In the dialog box of the 250 MVA block, you set the **Measurements** parameter to All measurements (V, I, flux). In the 110 Mvar block, you set it to Branch voltage and in the 110 Mvar1 block, you set it to Branch voltage and current.

The Output of the Multimeter Block is connected to a Scope block in order to display the measurements during the simulation. In addition, you can also check the **Display signals at simulation stop** parameter to display a plot of selected measurements when simulation stops.

Open the Multimeter block dialog box, select the signals you want to observe, as shown on the Parameters and Dialog Box section of the Multimeter block. Notice the labels used to defined the available measurements in the Multimeter block. See the reference section of the Saturable Transformer block and Series RLC Load block for a description of these labels.

Start the simulation. After 0.4 sec., the simulation stops and a MATLAB figure window opens to display the selected measurements in the Multimeter block.



See Also

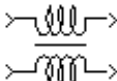
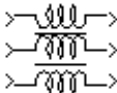
Voltage Measurement, Current Measurement

Mutual Inductance

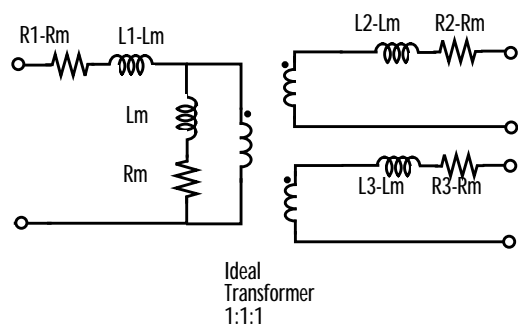
Purpose Implement a magnetic coupling between two or three windings.

Library Elements

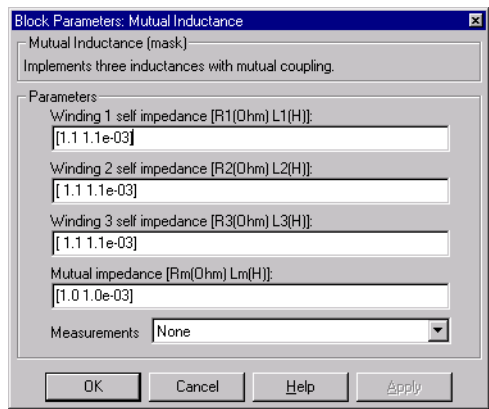
Description The Mutual Inductance block implements a magnetic coupling between three separate windings. Specify the self resistance and inductance of each winding on the first three entries of the dialog box and the mutual resistance and inductance in the last entry.



The electrical model for this block is given below:



Dialog Box and Parameters



Winding 1 self impedance

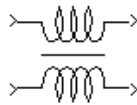
The self resistance and inductance for the winding 1, in ohms (Ω) and henries (H).

Winding 2 self impedance

The self resistance and inductance for the winding 2, in ohms (Ω) and henries (H).

Winding 3 self impedance

The self resistance and inductance in ohms (Ω) and henries (H) for the winding 3. Set the **Winding 3 self impedance** parameter to 0 to implement a Mutual Inductance block with two windings; a new icon will be displayed:



Mutual impedance

The mutual resistance and inductance between windings, in ohms (Ω) and henries (H). If the mutual resistance and reactance are set to $[0, 0]$, the block will implement three separate inductances with no mutual coupling.

Measurements

Select **Winding voltages** to measure the voltage across the winding terminals.

Select **Winding currents** to measure the current flowing through the windings.

Select **Winding voltages and currents** to measure the winding voltages and currents.

Place a Multimeter block in your model to display the selected measurements during the simulation.

Mutual Inductance

In the **Available Measurement** listbox of the Multimeter block, the measurements will identified by a label followed by the block name.

Measurement	Label
Winding voltages	u w1: , u w2: , u w3:
Winding currents	i w1: , i w2: , i w3:

Inputs and Outputs

The winding 1 is connected between input and output one of the Mutual Inductance block. The winding 2 is connected between input and output two and the winding 3, if defined, is connected between input and output three.

The input ports of the Mutual impedance block are at the same instantaneous polarity.

Limitations

Due to modeling constraints, the following restrictions apply:

$$R_1, R_2, R_3 \neq R_m$$

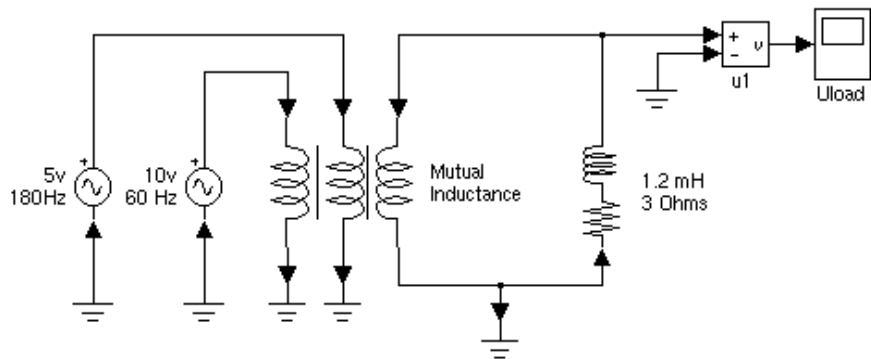
$$L_1, L_2, L_3 \neq L_m$$

Negative values are allowed for the self and mutual inductances as long as the self inductances are different from the mutual inductance.

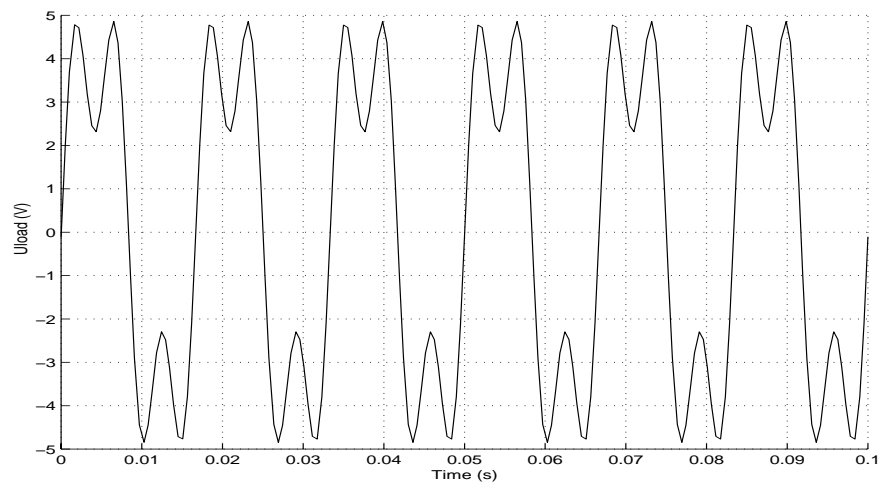
Windings can be left floating (i.e not connected by an impedance to the rest of the circuit). The floating winding will be internally connected to the main circuit through a resistor. This invisible connection does not affect voltage and current measurements.

Example

Three coupled windings are used to inject a third harmonic voltage into a circuit fed at 60 Hz. This example is available in the `psbmutual.mdl` file.



Simulation produces the following load voltage waveform:



See Also

Linear Transformer, Saturable Transformer

Neutral

Purpose Implement a common node in the circuit.

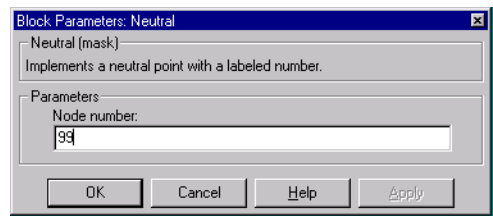
Library Elements

Description The Neutral block implements a common node with a specific node number. If the node number is set to zero, the neutral block automatically makes a connection to ground. The node number is displayed on the icon. You can use this block to create a floating neutral or to interconnect two points without drawing a connection line.

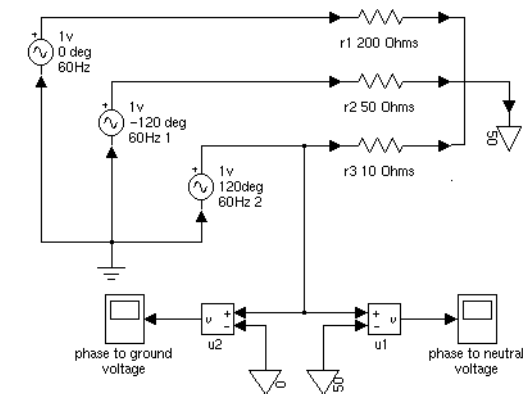


For the drawing ease, two types of Neutral blocks are available in the library: one block with an input and one block with an output.

Dialog Box



Example The following circuit uses three Neutral blocks. Three resistors are connected to a floating neutral (node 50). This example is available in the `psbneutral.mdl` file.



See Also

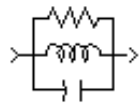
Ground

Parallel RLC Branch

Purpose Implement a parallel RLC branch.

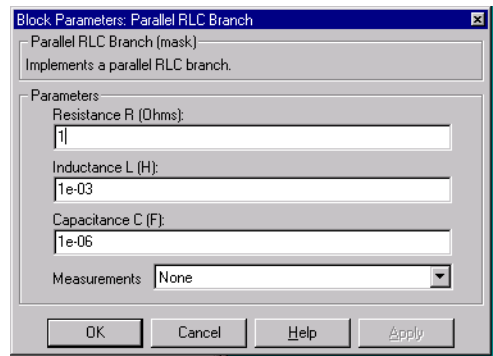
Library Elements

Description The Parallel RLC Branch block implements a single resistor, inductor, and capacitor or a parallel combination of these. To eliminate either the resistance, inductance, or capacitance of the branch, the R, L and C values must be set respectively to infinity, infinity, and zero. Only existing elements will be displayed in the block icon.



Negative values are allowed for resistance, inductance, and capacitance.

Dialog Box and Parameters



Resistance R

The branch resistance, in ohms (Ω).

Inductance L

The branch inductance, in henries (H).

Capacitance C

The branch capacitance, in farads (F).

Measurements

Select **Branch voltage** to measure the voltage across the Parallel RLC Branch block terminals.

Select **Branch current** to measure the total current (sum of R, L, C currents) flowing through the Parallel RLC Branch block.

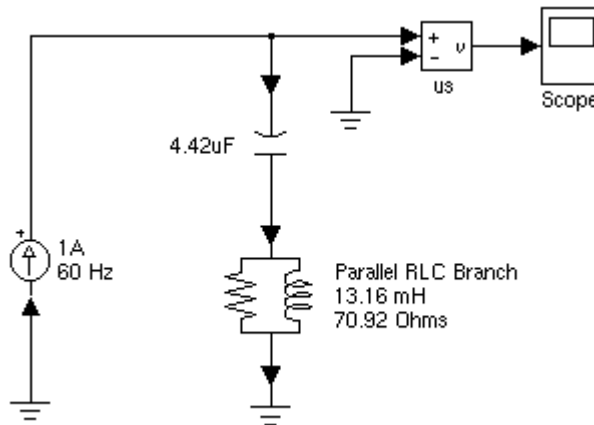
Select **Branch voltage** and **current** to measure the voltage and the current of the Parallel RLC Branch block.

Place a Multimeter block in your model to display the selected measurements during the simulation. In the **Available Measurement** listbox of the Multimeter block, the measurement will be identified by a label followed by the block name.

Measurement	Label
Branch voltage	ub:
Branch current	i b:

Example

Obtain the frequency response of an eleventh-harmonic filter turned at 660 Hz connected on the 60 Hz power system shown in the following figure. This example is available in the psbparal branch.mdl file.



The network impedance in Laplace domain is:

$$Z(s) = \frac{V(s)}{I(s)} = \frac{RLCs^2 + Ls + R}{LCs^2 + RCs}$$

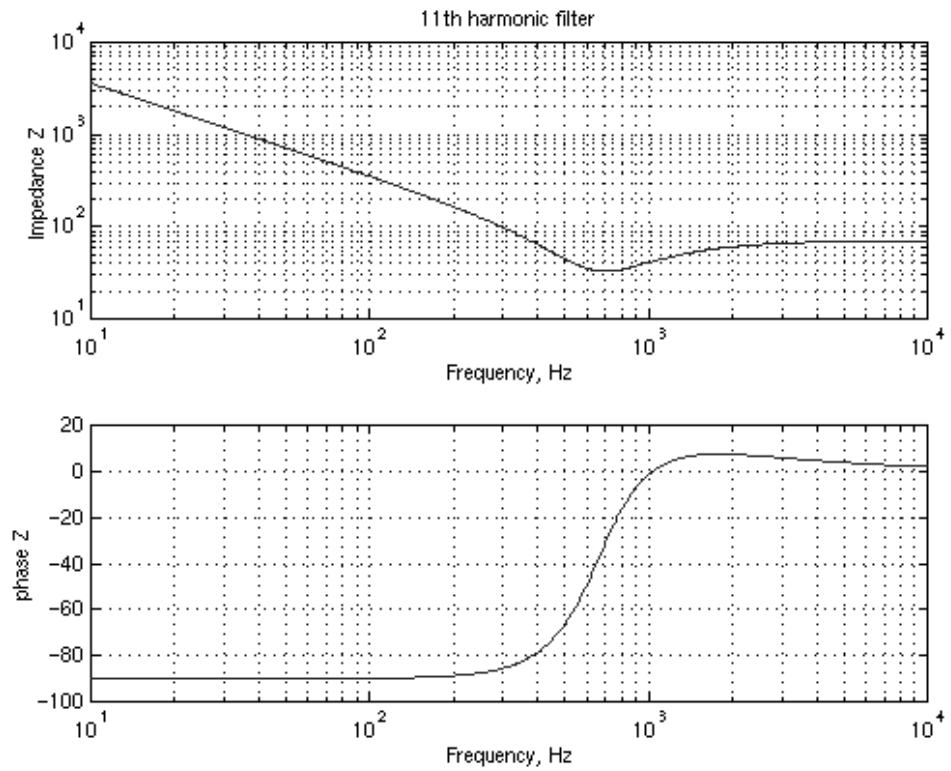
Parallel RLC Branch

To obtain the frequency response of the impedance you have to get the state-space model (A B C D matrices) of the system.

This system is a one input (Is) and one output (Vs) system. If you own the Control System Toolbox, you can get the transfer function Z(s) from the state-space matrices and the bode function.

```
[A, B, C, D] = power2sys('psbparal branch');
freq = logspace(1, 4, 500);
w = 2*pi*freq;
[Zmag, Zphase] = bode(A, B, C, D, 1, w);
subplot(2, 1, 1)
loglog(freq, Zmag)
grid
title('11th harmonic filter')
xlabel('Frequency, Hz')
ylabel('Impedance Z')
subplot(2, 1, 2)
semilogx(freq, Zphase)
xlabel('Frequency, Hz')
ylabel('phase Z')
grid
```

You can also use the Impedance Measurement block and the **powergui** to plot the impedance as function of frequency.



See Also

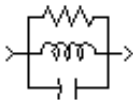
Parallel RLC Load, Series RLC Branch, Series RLC Load

Parallel RLC Load

Purpose Implement a linear parallel RLC load.

Library Elements

Description The Parallel RLC Load block implements a linear load as a parallel combination of RLC elements. At the specified frequency, the load will exhibit a constant impedance and its power will be proportional to the square of the applied voltage.



Only elements associated with nonzero powers will be displayed in the block icon.

Dialog Box and Parameters

Block Parameters: Parallel RLC Load

Parallel RLC Load (mask)
Electrical load branch
(parallel combination of RLC elements).

Parameters

Nominal voltage V_n (Vrms):
1000

Nominal frequency f_n (Hz):
60

Active power P (W):
10e3

Inductive reactive power Q_L (positive var):
100

Capacitive reactive power Q_c (negative var):
100

Measurements: None

OK Cancel Help Apply

Nominal voltage V_n

The nominal voltage of the load, in volts rms (Vrms).

Nominal frequency f_n

The nominal frequency, in hertz (Hz).

Active power P

The active power of the load, in watts.

Inductive reactive power Q_L

The inductive reactive power Q_L , in vars. Specify a positive value, or zero.

Capacitive reactive power QC

The capacitive reactive power QC, in vars. Specify a positive value, or zero.

Measurements

Select **Branch voltage** to measure the voltage across the Parallel RLC Load block terminals.

Select **Branch current** to measure the current flowing through the Parallel RLC Load block.

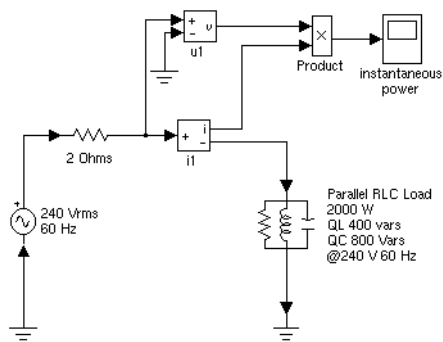
Select **Branch voltage and current** to measure the voltage and the current of the Parallel RLC Load block.

Place a Multimeter block in your model to display the selected measurements during the simulation. In the **Available Measurement** listbox of the Multimeter block, the measurement will be identified by a label followed by the block name.

Measurement	Label
Branch voltage	ub:
Branch current	i b:

Example

The following circuit is using a parallel RLC load block to implement a load. This example is available in the psbparal load. mdl file.



See Also

Parallel RLC Branch, Series RLC Load, Series RLC Branch

Permanent Magnet Synchronous Machine

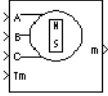
Purpose

Model the dynamics of a three-phase permanent magnet synchronous machine with sinusoidal flux distribution.

Library

Machines

Description



The Permanent Magnet Synchronous Machine block operates in either generating or motoring mode. The mode of operation is dictated by the sign of the mechanical torque (positive for motoring, negative for generating). The electrical and mechanical parts of the machine are each represented by a second-order state-space model. The model assumes that the flux established by the permanent magnets in the stator is sinusoidal, which implies that the electromotive forces are sinusoidal.

The block implements the following equations expressed in the rotor reference frame (qd frame).

Electrical System

$$\begin{aligned}\frac{d}{dt}i_d &= \frac{1}{L_d}v_d - \frac{R}{L_d}i_d + \frac{L_q}{L_d}p\omega_r i_q \\ \frac{d}{dt}i_q &= \frac{1}{L_q}v_q - \frac{R}{L_q}i_q - \frac{L_d}{L_q}p\omega_r i_d - \frac{\lambda p\omega_r}{L_q} \\ T_e &= 1.5 p[\lambda i_q + (L_d - L_q)i_d i_q]\end{aligned}$$

where (all quantities in the rotor reference frame, referred to the stator):

- L_q, L_d : q and d axis inductances
- R : resistance of the stator windings
- i_q, i_d : q and d axis currents
- v_q, v_d : q and d axis voltages
- ω_r : angular velocity of the rotor
- λ : amplitude of the flux induced by the permanent magnets of the rotor in the stator phases
- p : number of pole pairs
- T_e : electromagnetic torque

Permanent Magnet Synchronous Machine

Mechanical System

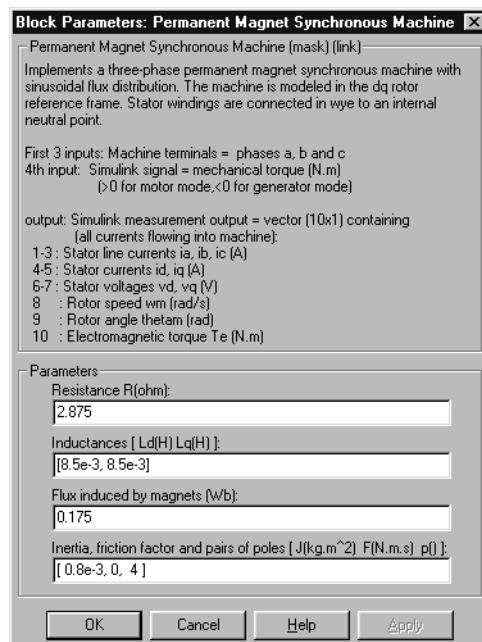
$$\frac{d}{dt}\omega_r = \frac{1}{J}(T_e - F\omega_r - T_m)$$

$$\frac{d\theta}{dt} = \omega_r$$

where:

- J: Combined inertia of rotor and load
- F: Combined viscous friction of rotor and load
- θ : Rotor angular position
- T_m : Shaft mechanical torque

Dialog Box and Parameters



Block Parameters: Permanent Magnet Synchronous Machine

Permanent Magnet Synchronous Machine (mask) (link)

Implements a three-phase permanent magnet synchronous machine with sinusoidal flux distribution. The machine is modeled in the dq rotor reference frame. Stator windings are connected in wye to an internal neutral point.

First 3 inputs: Machine terminals = phases a, b and c
4th input: Simulink signal = mechanical torque (N.m)
>0 for motor mode, <0 for generator mode

output: Simulink measurement output = vector (10x1) containing
(all currents flowing into machine):
1-3 : Stator line currents ia, ib, ic (A)
4-5 : Stator currents id, iq (A)
6-7 : Stator voltages vd, vq (V)
8 : Rotor speed wm (rad/s)
9 : Rotor angle thetam (rad)
10 : Electromagnetic torque Te (N.m)

Parameters

Resistance R(ohm):
2.875

Inductances [Ld(H) Lq(H)]:
[8.5e-3, 8.5e-3]

Flux induced by magnets (Wb):
0.175

Inertia, friction factor and pairs of poles [J(kg.m^2) F(N.m.s) p()]:
[0.8e-3, 0, 4]

OK Cancel Help Apply

Resistance

The stator resistance R (Ω).

Permanent Magnet Synchronous Machine

Inductances

The d-axis and q-axis stator inductances L_d (H) and L_q (H).

Flux induced by magnets

The constant flux λ (Wb) induced in the stator windings by the magnets.

Mechanical

The combined machine and load inertia coefficient J (kg.m^2), combined viscous friction coefficient F (N.m.s), and pole pairs p .

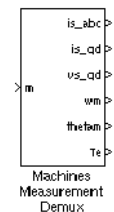
Inputs and Outputs

The first three inputs are the electrical connections of the machine's stator. The fourth input is the mechanical torque at the machine's shaft (Simulink signal). This input should normally be positive because the Permanent Magnet Synchronous Machine block is usually used as a motor. Nevertheless, you can apply a negative torque input if you choose to use the block in generating mode.

The block outputs a vector containing the following 10 variables (all currents flowing into machine):

- 1-3: Line currents i_a , i_b and i_c , in A
- 4-5: q and d axis currents i_q and i_d , in A
- 6-7: q and d axis voltages v_q and v_d , in V
- 8: Rotor mechanical speed ω_r , in rad/sec
- 9: Rotor mechanical angle θ , in rad
- 10: Electromagnetic torque T_e , in N.m.

These variables can be demultiplexed by using the special Machines Measurement Demux block provided in the Machines library.



Assumption

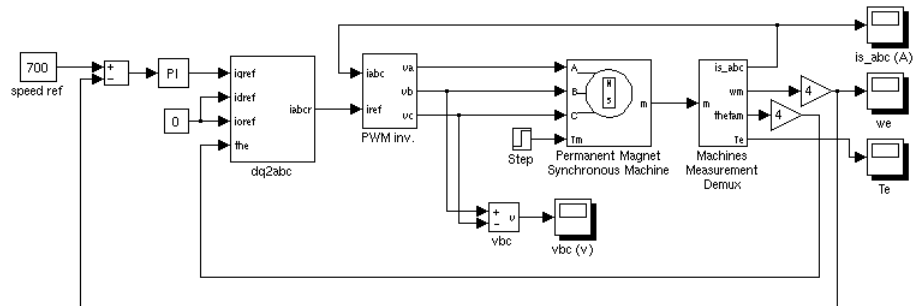
The Permanent Magnet Synchronous Machine block assumes a linear magnetic circuit with no saturation of the stator and rotor iron. This

Permanent Magnet Synchronous Machine

assumption can be made because of the large air gap usually found in permanent magnet synchronous machines.

Example

This example illustrates the use of the Permanent Magnet Synchronous Machine block in the motoring mode with a closed-loop control system built entirely in Simulink. The interfacing is done using Controlled Voltage Source blocks from the Electrical Sources library. The complete system consists of a PWM inverter built with ideal switches (Simulink Relay blocks). Two control loops are used; the inner loop is used to regulate the motor line currents and the outer loop regulates the motor's speed. More elaborate and performant control schemes for Permanent Magnet Synchronous Machine block can be found for instance in [1]. The mechanical torque applied at the motor's shaft is originally 3 N.m (nominal) and steps to 1 N.m at $t=0.04$ sec. The parameters of the machine are those found in the dialog box section.



Permanent magnet synchronous motor fed by PWM inverter

Double click on the Help button (?) for details

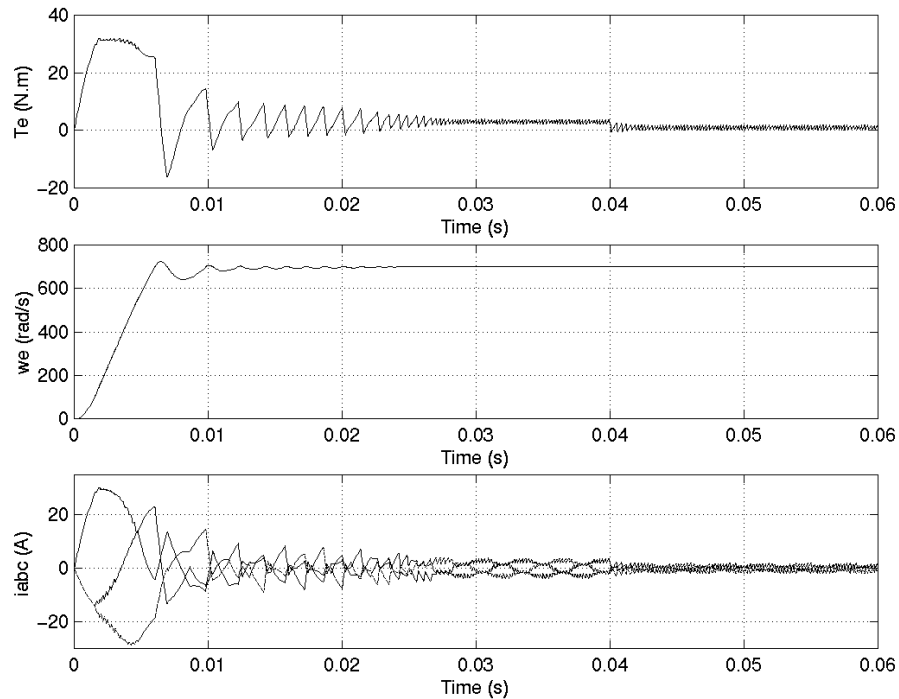


Open the Simulink diagram by typing `psbpmmotor` or by choosing **Permanent Magnet Sync. Mach** from the demos group in the **powerlib** library. Set the simulation parameters as follows:

- Integrator type: Stiff, ode15s
- Stop time: 0.06
- Integration options: Use default options, except for Absolute tolerance which you can set to $1e-3$

Permanent Magnet Synchronous Machine

Run the simulation. Once the simulation is completed, observe the motor's torque, speed, and currents.



The torque climbs to nearly 32 N.m when the motor starts but stabilizes rapidly to its nominal value (3 N.m), until the step is applied, at which point the torque oscillates slightly before stabilizing to its new value (1 N.m).

As for the speed, you can see that it stabilizes quite fast at start-up and is not affected by the load step.

The currents are initially high when the machine starts, like the torque, but stabilize quickly to their nominal value, until the step is applied, at which point they oscillate before stabilizing to a lower value, corresponding to the load torque decrease.

References

[1] Grenier D., L.-A. Dessaint, O. Akhrif, Y. Bonnassieux, and B. LePioufle, "Experimental Nonlinear Torque Control of a Permanent Magnet Synchronous

Motor Using Saliency", *IEEE Transactions on Industrial Electronics*, Vol. 44, No. 5, October 1997, pp. 680-687.

PI Section Line

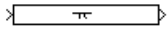
Purpose

Implement a single phase transmission line with lumped parameters.

Library

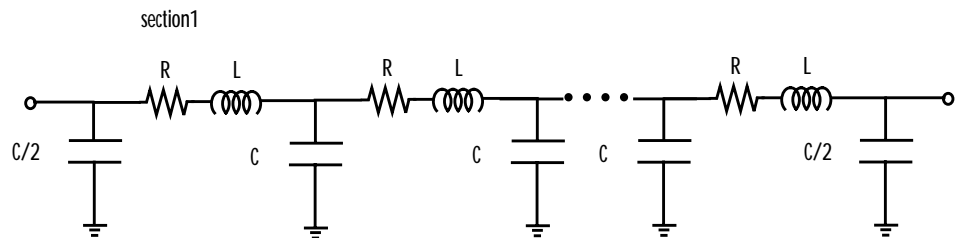
Elements

Description



The PI Section Line block implements a single phase transmission line with parameters lumped in pi sections.

For a transmission line, the resistance, inductance and capacitance are uniformly distributed along the line. An approximate model of the distributed parameter line is obtained by cascading several identical pi sections as shown in the figure below.



Unlike the Distributed Parameter Line block, which has an infinite number of states, the pi section linear model has a finite number of states that permits to compute a linear state-space model. The number of sections to be used depends on the frequency range to be represented.

A good approximation of the maximum frequency range represented by the pi line model is given by the following equation

$$f_{max} = \frac{Nv}{8l}$$

where:

- N = Number of pi sections
- v = Propagation speed in km/s = $1/\sqrt{LC}$ L in H/km, C in F/km
- l = Line length in km

For example, for a 100 km aerial line having a propagation speed of 300 000 km/s, the maximum frequency range represented with a single pi section is approximately 375 Hz. For studying interactions between a power system and

a control system, this simple model could be sufficient. However for switching surge studies involving high frequency transients in the kHz range, much shorter pi sections should be used. In fact, accurate results would probably only be obtained by using a distributed parameters line model.

Dialog Box and Parameters

Frequency used for RLC specifications

Frequency that has been used to compute the line parameters, in hertz (Hz)

Resistance per unit length

The resistance per unit length of the line, in ohms/km (Ω).

Inductance per unit length

The inductance per unit length of the line, in henries/km (H/km).

Capacitance per unit length

The capacitance per unit length of the line, in farad/km (F/km).

Length

The line length in km.

PI Section Line

Number of pi sections

The number of pi sections. The minimum value is 1.

Measurements

Select **Input and output voltages** to measure the sending end (input port) and receiving end (output port) voltages of the line model.

Select **Input and output voltages** to measure the sending end and receiving end currents of the line model.

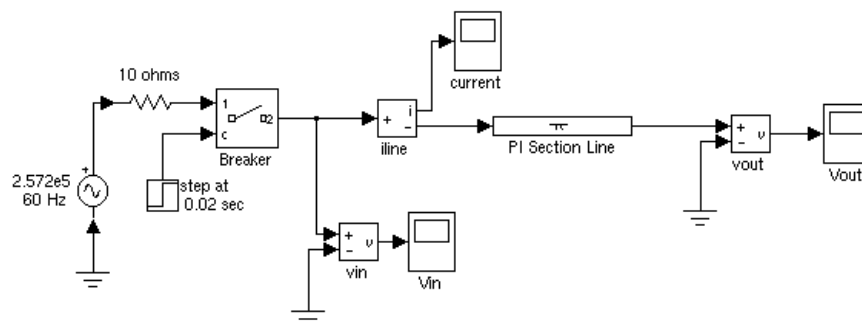
Select **All voltages and currents** to measure the sending end and receiving end voltages and currents of the line model.

Place a Multimeter block in your model to display the selected measurements during the simulation. In the **Available Measurement** listbox of the Multimeter block, the measurement will be identified by a label followed by the block name.

Measurement	Label
Sending end voltage (block input)	Us:
Receiving end voltage (block output)	Ur:
Sending end current (input current)	Is:
Receiving end current (output current)	Ir:

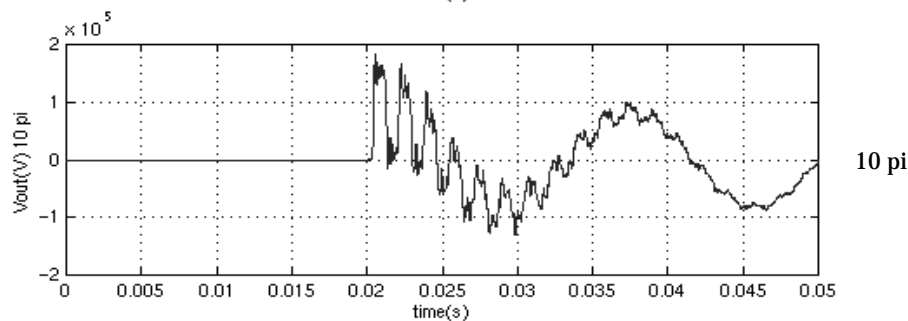
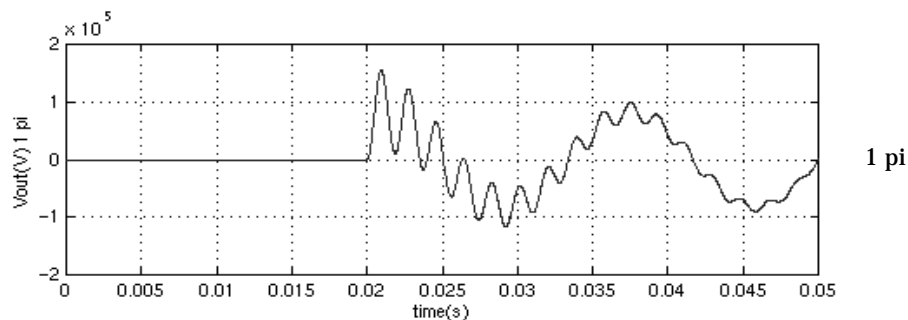
Example

Obtain the line energization voltages and current in the following circuit. This circuit is available in the `psbpi line.mdl` file.



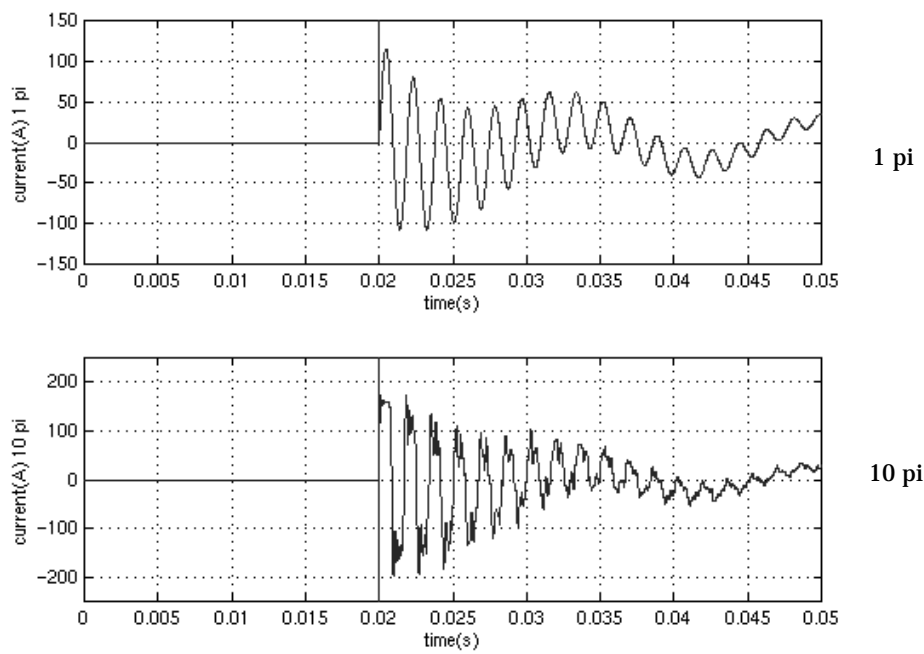
The results obtained with the line modeled by one pi section of 100 km and 10 pi sections of 10 km are shown below.

Voltages:



PI Section Line

Currents:



See Also

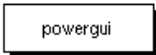
Distributed Parameter Line

Purpose

Graphical user interface for the analysis of circuits and systems.

Library

powerlib

Description

The Powergui block opens a graphical user interface (GUI) that displays steady-state values of measured current and voltages as well as all state variables (inductor currents and capacitor voltages).

The Powergui block allows you to modify the initial states in order to start the simulation from any initial conditions.

It allows load flow computation and initialization of three phase networks containing machines.

The Powergui block also displays impedance versus frequency plots when Impedance Measurement blocks are present in your model.

If you own the Control System Toolbox, the Powergui block can generate the state-space model (SS) of your system and automatically opens the LTI Viewer interface for time and frequency domain responses.

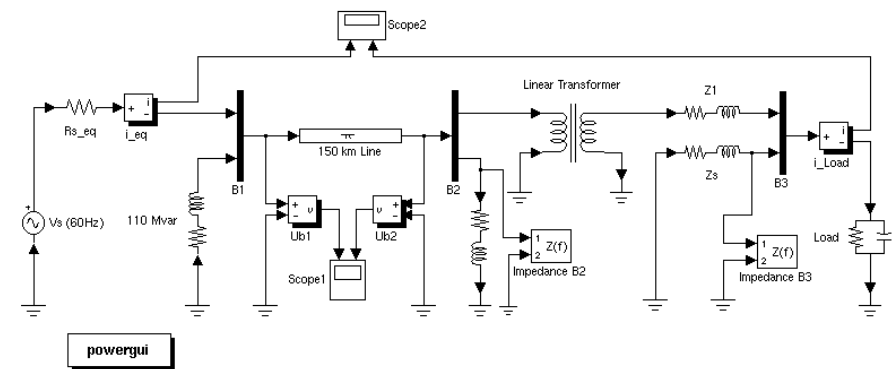
Copy the Powergui block in the top level of your model and double-click on the block to open the interface.

The main menu of the Powergui block provides tools to:

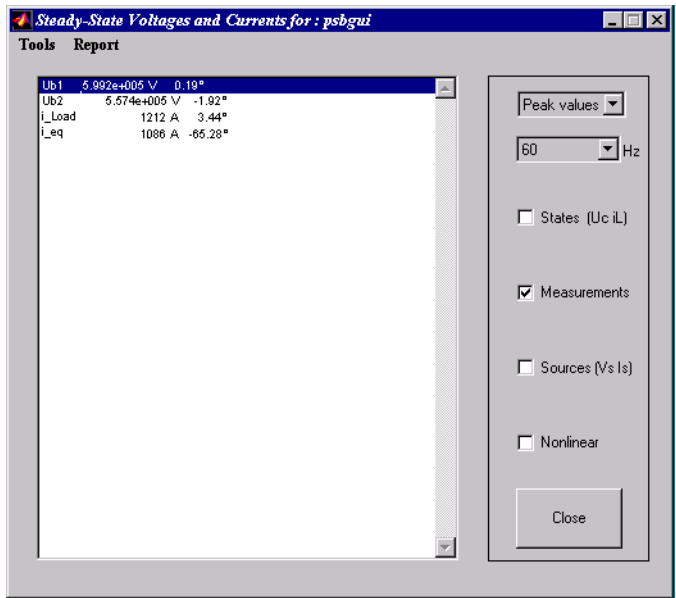
- Display steady-state voltage and currents
- Display and modify initial state values
- Perform load flows and machine initialization
- Display impedance vs frequency measurements
- Use the LTI Viewer of the Control System Toolbox
- Generate a report of the steady-state calculations

Example

The Powergui block is presented here for the psbgui.mdl circuit.



In the psbgui.mdl demo, double-click on the Powergui block to open the interface. A window named **Steady-State Voltages and Currents** opens.



The window displays the steady-state phasor voltages and currents of the four measurements blocks of the circuit. Voltages and currents are displayed at the frequency of the Vs (60 Hz) source.

Steady-State Voltages and Currents Window

The **Steady-State Voltages and Currents** window contains the following parameters:

Units

Set the **Units** parameter to **Peak values** to display the peak values of the selected values. Set the **Units** parameter to **RMS** to display the root-mean-square (RMS) values of the selected values.

Hz

Allows to choose the frequency, in hertz (Hz), you are interested in to display phasors. If your circuit contains sources with different frequencies, this menu allows you to select any frequency in the list.

States (Uc iL)

If checked, the window displays the steady-state phasors of the capacitor voltages and inductor currents of the circuit.

Measurements

If checked, the window displays the steady-state phasor voltages and currents of the measurements blocks of the circuit.

Sources (Vs Is)

If checked, the window displays the steady-state phasor voltages of the electrical sources of the circuit.

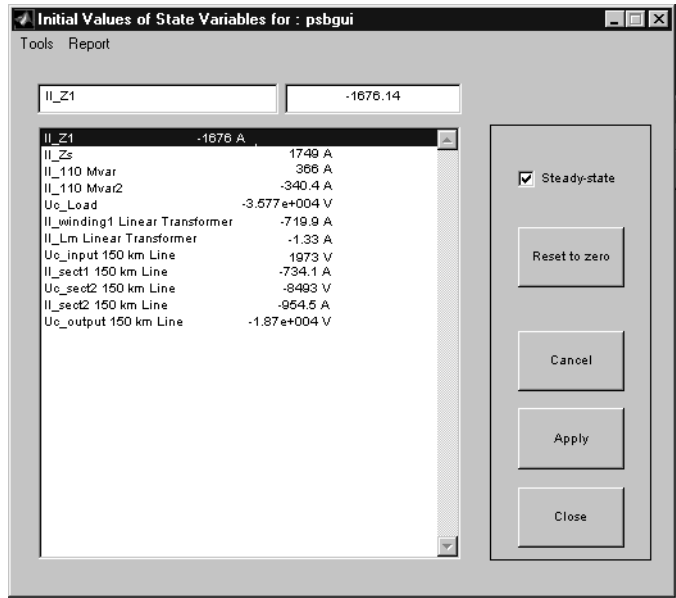
Nonlinear

If checked, the window displays the steady-state voltages and currents of the nonlinear blocks of the circuit. Note that there is no nonlinear element in the model given in the example.

To display the initial state values of the state variables of the circuit, select the **Initial Values of State Variables** menu item in the **Tools** menu, then choose the **Display or modify initial state values** menu item. Note that from the menu you can also load initial values from a previous session or save the current initial values into a MAT-file.

Initial Values of State Variables Window

The **Initial Values of State Variables** that appears on your window displays the initial values of state variables.



The name of the state variables consist of the name of the block where the capacitor or the inductor is found. The name of the block is preceded by the Uc_ label for capacitor voltages and by the IL_ label for the inductor currents.

Steady-state

If checked, specify to the Power System Blockset to use the initial conditions for a steady-state simulation. If you have previously changed initial conditions, you will have to check the **Steady State** parameter in order to return to steady state initial conditions.

Reset to zero

Sets the initial states to zero.

Apply

Apply the settings to the simulation.

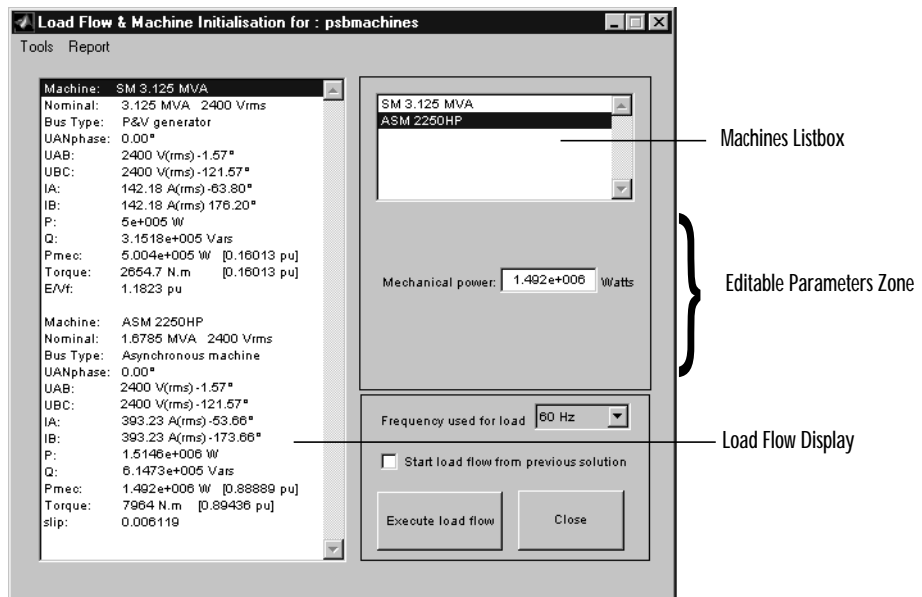
Close

Close the window.

Load Flow & Machine Initialization Window

The **Load Flow & Machines Initialization** window allows you to perform load flows and initialize three-phase circuits containing three-phase machines so that the simulation starts in steady-state. This option is available with network containing the following three types of machines: Simplified Synchronous Machine, Synchronous Machine, and Asynchronous Machine blocks.

Since there is no machine block in our example, we reproduce here the window displayed for the two-machine case illustrated in Session 5 of the Tutorial chapter (available in the `psbmachines.mdl` demo).



Machines Listbox

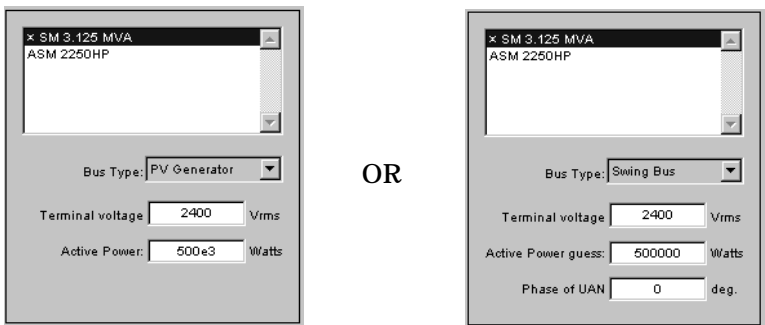
The Machines Listbox displays the names of the Simplified Synchronous Machines, the Synchronous Machines, and the Asynchronous Machine blocks of your circuit. Select a machine in the listbox in order to set its

parameters for the desired load flow (power and voltage, or voltage and phase).

Editable Parameters Zone

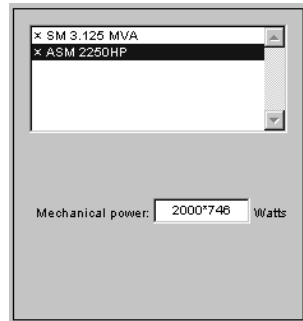
Depending on the type of machine you select in the **Machines Listbox** section, the Editable Parameters Zone displays different parameters to be set for the load flow.

If you select a machine that is a Simplified Synchronous Machine block or a Synchronous Machine block, you will have the following editable parameters:



If the **Bus Type** parameter is set to **PV Generator**, you can set the desired terminal voltage and active power of the machine. If the **Bus Type** parameter is set to **Swing Bus**, you can set the desired terminal voltage, enter an active power guess, and specify the phase of the UAN terminal voltage of the machine.

If you select a machine that is an Asynchronous Machine block, you will only have to enter the desired mechanical power delivered by the machine.



Frequency used for load flow

Specifies the frequency to be used in the load flow calculations (normally 60 Hz or 50 Hz).

Start load flow from previous solution

If checked, the load flow will start with initial conditions corresponding to the previous load flow solution. Try this option if the load flow fails to converge.

Execute load flow

Executes the load flow calculations for the given load flow parameters

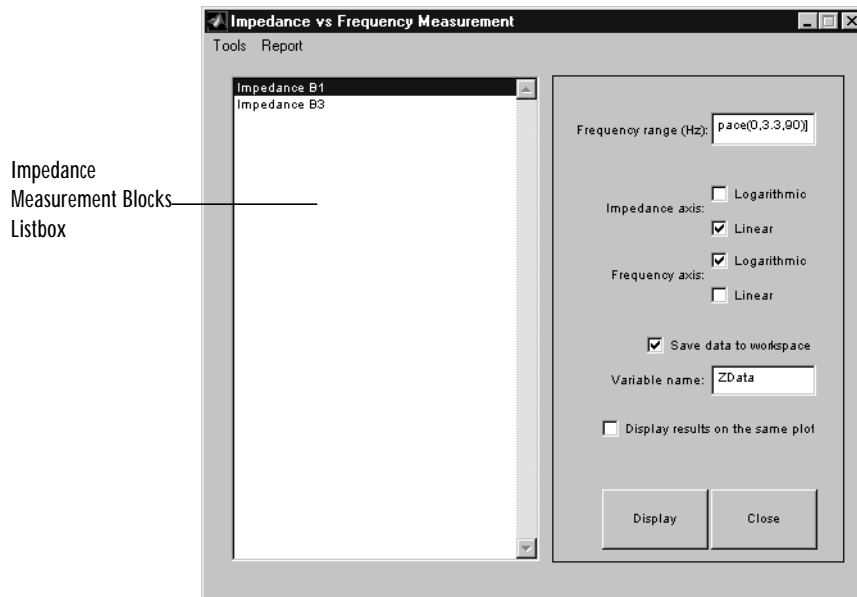
Load Flow Display

Once the load flow has been solved for the given load flow parameters, the initial conditions are automatically set in the dialog boxes of the machines and you can start the simulation in steady state.

Impedance vs Frequency Measurement Window

To perform the impedance versus frequency measurements defined by the Impedance Measurement blocks of the circuit, select the **Impedance vs frequency Measurement** menu item in the **Tools** menu.

The **Impedance vs Frequency Measurement** that appears on your window is as follows.



Frequency range (Hz)

Specifies the frequency vector, in hertz (Hz). You can specify in that field any valid MATLAB expression defining a vector of frequencies, for example `0: 2: 1000` or `linspace(0, 1000, 500)`. The default is `logspace(0, 3, 50)`.

Impedance axis

Defines the units of the impedance axis. Check **Logarithmic** or **linear**.

Frequency axis

Defines the units of the frequency axis. Check **Logarithmic** or **linear**.

Save data to workspace

Data can be saved in a variable in the workspace. The name of the variable is defined by the **Variable name** parameter.

Display results on the same plot

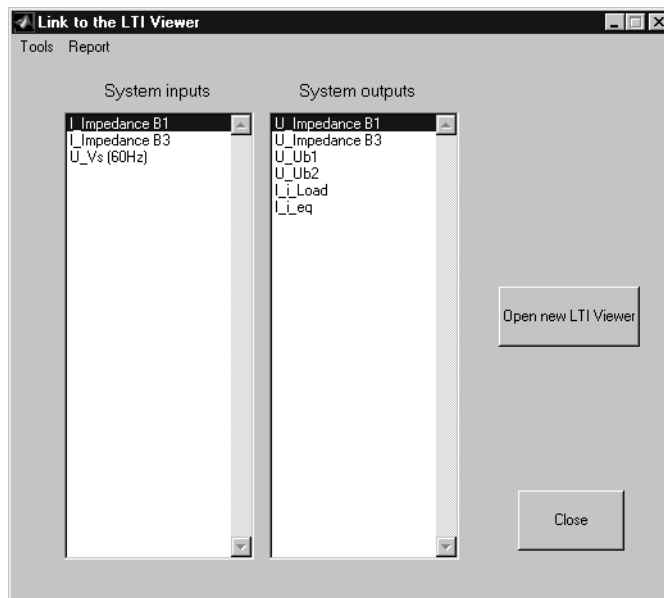
If checked, the impedance vs frequency plots will be displayed on the same plot when several Impedance Measurement blocks are used in the circuit.

Display

Click on the button to start the impedance vs frequency measurement and display results.

Link to the LTI Viewer Window

If you own the Control System Toolbox, you can open the LTI viewer for time and frequency responses. To do this, select the Control System Toolbox (LTI Viewer) menu item in the **Tools** menu of the **powergui**. Your **Link to LTIviewer** window looks like this.



System inputs

Lists the inputs of the state-space equivalent system of your circuit. Select the inputs to be used by the LTI Viewer.

System outputs

Lists the outputs of the state-space equivalent system of your circuit. Select the outputs to be used by the LTI Viewer.

Open new LTI Viewer

Generate the state-space model of the circuit and opens the LTI viewer for the selected system inputs and outputs.

Generate a Report

You can generate a report of the steady-state calculations done by the Powergui block by selecting the **Generate report** menu item in the **Report** menu.

The Powergui block will generate a report in a file containing steady-state values of the measurement blocks, the sources, the nonlinear models, and the states of your circuit. The report will be saved in a file, with the extension. rep. For example the file report for the psbgui.mdl will be named psbgui.rep.

See Also

powerinit, power2sys

Purpose

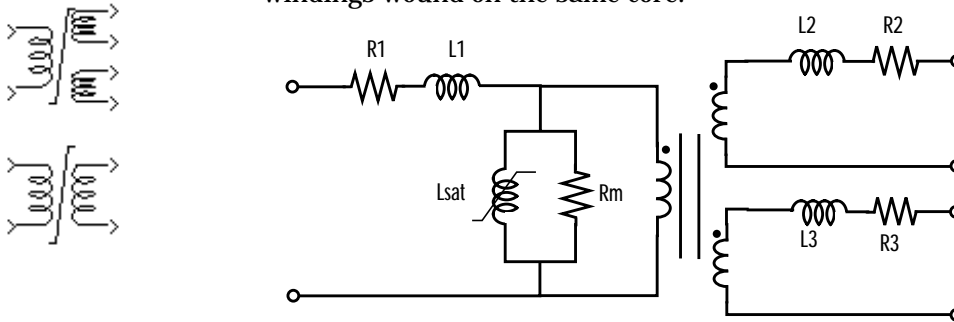
Implement a two- or three- winding saturable transformer.

Library

Elements

Description

The Saturable Transformer block model shown below consists of three coupled windings wound on the same core.



The model takes into account the winding resistances ($R1$ $R2$ $R3$), the leakage inductances ($L1$ $L2$ $L3$) as well as the magnetizing characteristics of the core which is modeled by a resistance Rm simulating the core active losses and a saturable inductance $Lsat$. The saturation characteristic is specified as a piecewise linear characteristic.

The Per Unit Conversion

In order to comply with the industry practice, you must specify the resistance and inductance of the windings in per unit (p.u.). the values are based on the transformer rated power P_n in VA, nominal frequency f_n in Hz, and nominal voltage V_n , in V_{rms} , of the corresponding winding. For each winding the per unit resistance and inductance are defined as:

$$R(\text{p.u.}) = \frac{R(\Omega)}{R_{base}}$$

$$L(\text{p.u.}) = \frac{L(H)}{L_{base}}$$

Saturable Transformer

The base resistance and base inductance used for each winding are:

$$R_{base} = \frac{(V_n)^2}{P_n}$$
$$L_{base} = \frac{R_{base}}{2\pi f_n}$$

For the magnetization resistance R_m , the p.u. values are base on the transformer rated power and on nominal voltage of the winding one.

The default parameters of winding one specified in the dialog box section give the following bases:

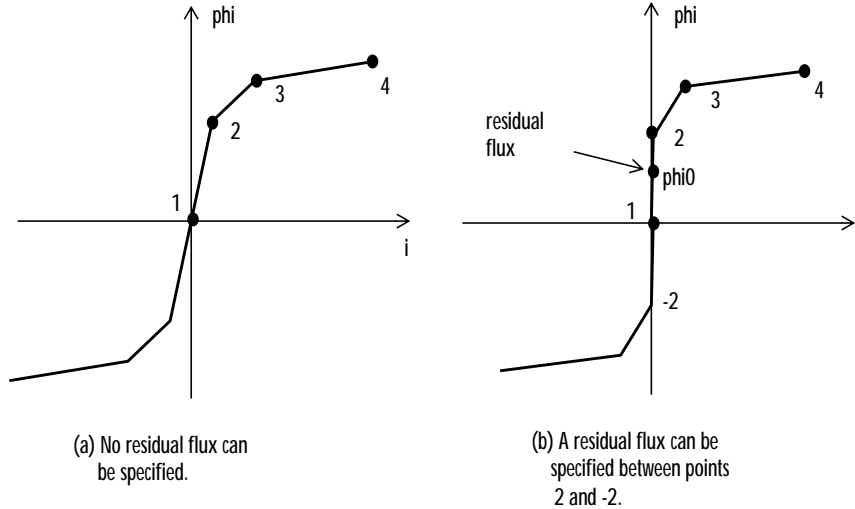
$$R_{base} = \frac{(735e3/\sqrt{3})^2}{250e6} = 720.3\Omega \quad L_{base} = \frac{720.3}{2\pi 60} = 1.91H$$

Suppose that the winding 1 parameters are $R_1=1.44\Omega$ and $L_1=0.1528H$, the corresponding values to be entered in the dialog box are:

$$R_1 = \frac{1.44\Omega}{720.3\Omega} = 0.002\text{p.u.}$$
$$L_1 = \frac{0.1528H}{1.91H} = 0.08\text{p.u.}$$

Saturation Characteristic

The saturation characteristic of the Saturable Transformer block is defined by a piece-wise linear relationship between the flux and the magnetization current.



Therefore, if you want to specify a residual flux ϕ_{i0} the second point of the saturation characteristic should correspond to a zero current as shown on the figure (b).

The saturation characteristic is entered as (i, phi) pair values in per unit, starting with pair (0,0). The Power System Blockset converts the vector of fluxes Φ_{pu} and the vector of currents I_{pu} into standard units to be used in saturation model of the Saturable Transformer block

$$\Phi = \Phi_{pu} \Phi_{base}$$

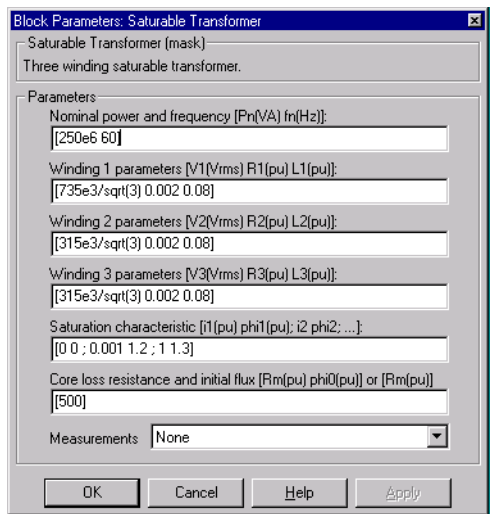
$$I = I_{pu} I_{base}$$

where the base flux (Φ_{base}) and base current (I_{base}) are the in peak values obtained at nominal voltage power and frequency:

$$I_{base} = \frac{P_n}{V_1} \sqrt{2} \quad \Phi_{base} = \frac{V_1}{2\pi f_n} \sqrt{2}$$

Saturable Transformer

Dialog Box and Parameters



Nominal power and frequency

The nominal power rating, P_n in volt amperes (VA), and frequency, in hertz (Hz), of the transformer.

Winding 1 parameters

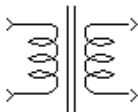
The nominal voltage in volts rms, resistance, and leakage inductance in p.u. for the winding 1.

Winding 2 parameters

The nominal voltage in volts rms, resistance, and leakage inductance in p.u. for the winding 2.

Winding 3 parameters

The nominal voltage in volts rms, resistance, and leakage inductance in p.u. for the winding 3. Setting the **Winding 3 parameters** parameter to 0 implements a Saturable Transformer block with only two windings and a new icon will be displayed:



Saturation characteristic

Specify a series of current (p.u.) - flux (p.u.) pairs starting with (0,0).

Core loss resistance and initial flux

Specify the active power dissipated in the core by entering the equivalent resistance R_m in p.u. For example, to specify a 0.2% of active power core loss at nominal voltage, use $R_m = 500$ p.u. You can also specify the initial flux ϕ_0 (p.u). This initial flux becomes particularly important when the transformer is energized. If ϕ_0 is not specified, the initial flux will be automatically adjusted so that the simulation starts in steady state.

Measurements

Select **Winding voltages** to measure the voltage across the winding terminals of the Linear Transformer block.

Select **Winding currents** to measure the current flowing through the windings of the Linear Transformer block.

Select **Flux and magnetization current** to measure the magnetization current of the Linear Transformer block.

Select **All Measurements (V, I, Flux)** to measure the winding voltages and currents plus the magnetization current.

Place a Multimeter block in your model to display the selected measurements during the simulation.

In the **Available Measurement** listbox of the Multimeter block, the measurements will be identified by a label followed by the block name.

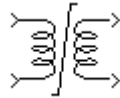
Measurement	Label
Winding voltages	u w1: , u w2: , u w3:
Winding currents	i w1: , i w2: , i w3:
Magnetization current	i mag:
Flux	Flux:

Inputs and Outputs

Input one, output one and output three (if it exists) are at the same instantaneous polarity.

Saturable Transformer

If you set the entry for the third winding to zero, the blockset will consider a transformer with two windings and a new icon will be displayed:



Limitations

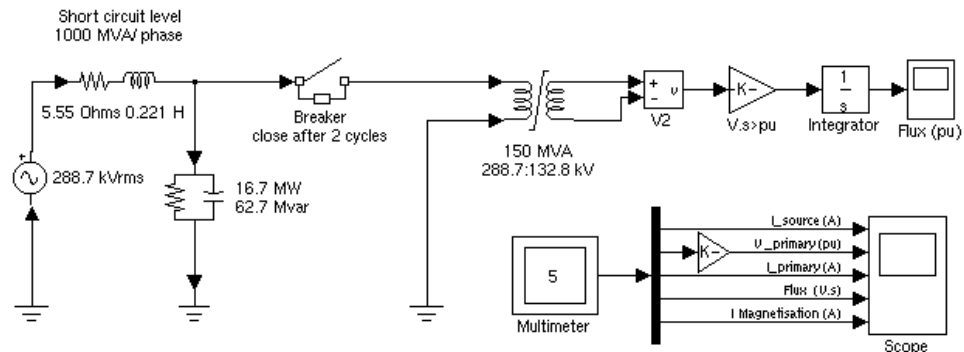
Windings can be left floating (i.e not connected by an impedance to the rest of the circuit). However the floating winding will be connected internally to the main circuit through a resistor. This invisible connection does not affect voltage and current measurements.

The flux saturation model does not include hysteresis.

Example

Energization of one phase of a three-phase 450 MVA, 500/230 kV transformer on a 3000 MVA source. The transformer parameters are:

Nominal power: 150e6, 60Hz, Winding 1 parameters (primary): 500e3Vrms/sqrt(3), $R=0.002$ p.u. $X=0.08$ p.u., winding 2 parameters (secondary): 230e3Vrms/sqrt(3), $R=0.002$ p.u. $X=0.08$ p.u., Core loss resistance: 1000p.u., Saturation characteristic: [0 0; 0 1.2; 1.0 1.52], residual flux=0.8p.u.

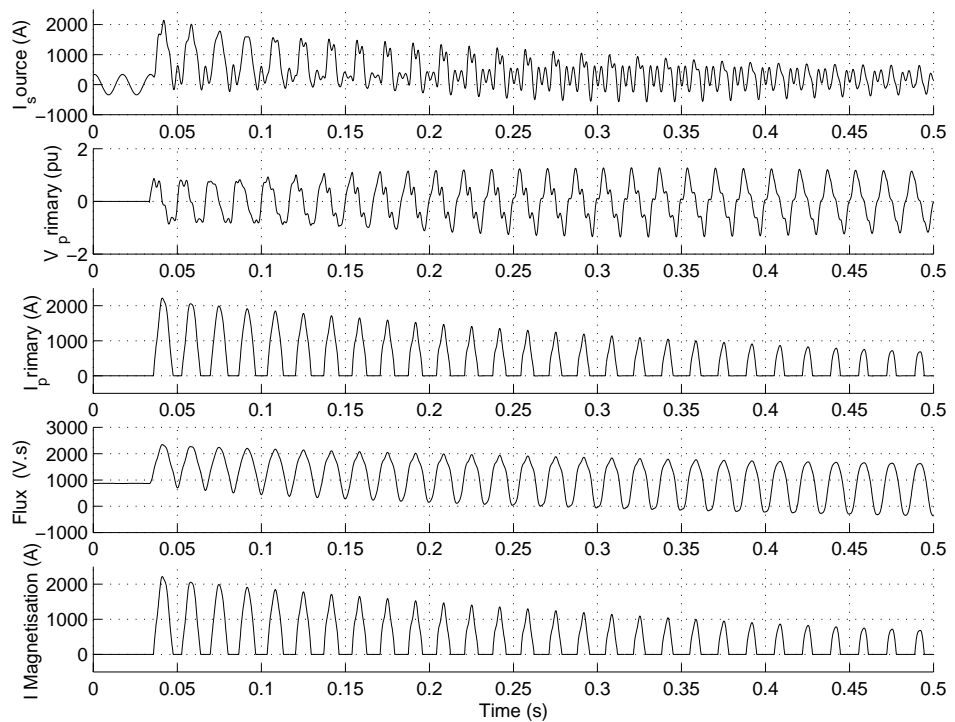


This circuit is available in the psbxfosaturable.mdl file.

As the source is resonant at the 4th harmonic, you can observe a high fourth harmonic content in the secondary voltage. In this circuit the flux is calculated in two ways:

- 1 By integrating the secondary voltage
- 2 By using the Multimeter block

Simulation of this circuit illustrates the saturation effect on the transformer current and voltage:



See Also

Linear Transformer, Mutual Inductance

Series RLC Branch

Purpose Implement a series RLC branch.

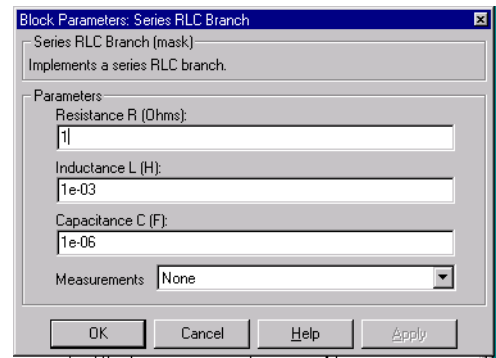
Library Elements

Description The Series RLC Branch block implements a single resistor, inductor or capacitor, or a series combination of these. To eliminate either the resistance, inductance, or capacitance of the branch, the R, L, and C values must be set respectively to zero, zero, and infinity. Only existing elements will be displayed in the block icon.



Negative values are allowed for resistance inductance and capacitance.

Dialog Box and Parameters



Resistance R
The branch resistance, in ohms (Ω).

Inductance L
The branch inductance, in henries (H).

Capacitance C
The branch capacitance, in farads (F).

Measurements
Select **Branch voltage** to measure the voltage across the Series RLC Branch block terminals.

Select **Branch current** to measure the current flowing through the Series RLC Branch block.

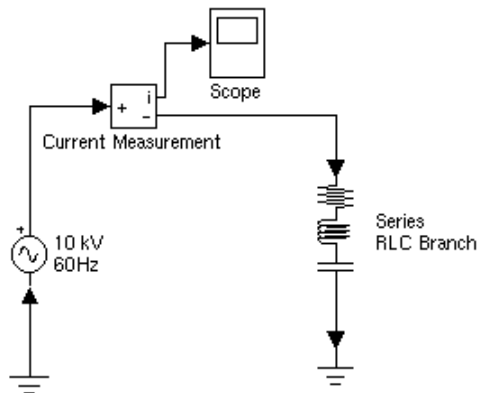
Select **Branch voltage and current** to measure the voltage and the current of the Parallel RLC Branch block.

Place a Multimeter block in your model to display the selected measurements during the simulation. In the **Available Measurement** listbox of the Multimeter block, the measurement will be identified by a label followed by the block name.

Measurement	Label
Branch voltage	ub:
Branch current	i b:

Example

Obtain the frequency response of a fifth-harmonic filter (tuned frequency = 300 Hz) connected on a 60 Hz power system. This example is available in the `psbseriesbranch.mdl` file.



The network impedance in Laplace domain is:

$$Z(s) = \frac{V(s)}{I(s)} = \frac{LCs^2 + RCs + 1}{Cs}$$

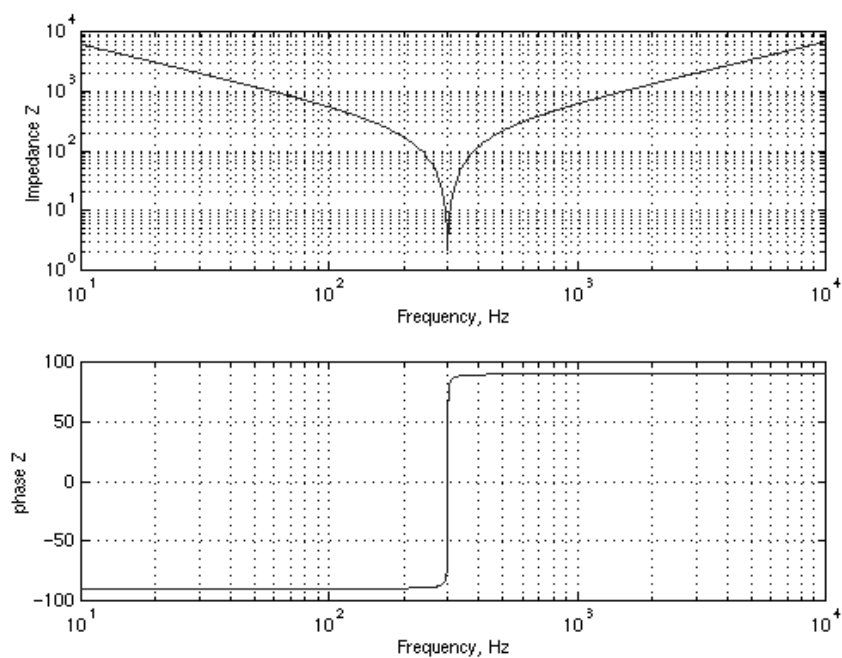
To obtain the frequency response of the impedance you have to get the state-space model (A B C D matrices) of the system.

Series RLC Branch

This system is a one input (Vsource) and one output (Current Measurement block) system. If you own the Control System Toolbox, you can get the transfer function $Z(s)$ from the state-space matrices and the bode function as follows:

```
[A, B, C, D] = power2sys('psbseriesbranch');
freq = logspace(1, 4, 500);
w = 2*pi*freq;
[Ymag, Yphase] = bode(A, B, C, D, 1, w);
% invert Y(s) to get Z(s)
Zmag = 1./Ymag;
Zphase = -Yphase;
subplot(2, 1, 1)
loglog(freq, Zmag)
grid
title('5th harmonic filter')
xlabel('Frequency, Hz')
ylabel('Impedance Zmag')
subplot(2, 1, 2)
semilogx(freq, Zphase)
xlabel('Frequency, Hz')
ylabel('phase Z')
grid
```

You can also use the Impedance Measurement block and the Powergui block to plot the impedance as function of frequency. In order to measure the impedance you must disconnect the voltage source.



See Also

Series RLC Load, Parallel RLC Branch, Parallel RLC Load

Series RLC Load

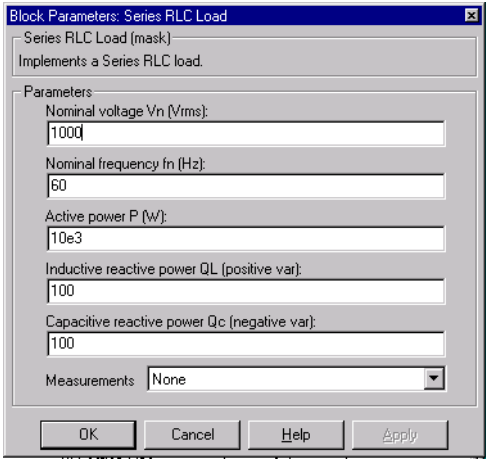
Purpose Implement a linear series RLC load.

Library Elements

Description The Series RLC Load block implements a linear load as a series combination of R L C elements. At the specified frequency, the load will exhibit constant impedance and its power will be proportional to the square of the applied voltage. Only elements associated with nonzero powers will be displayed in the block icon.



Dialog Box and Parameters



Nominal voltage Vn

The nominal voltage of the load, in volts rms.

Nominal frequency fn

The nominal frequency, in hertz.

Active power P

The active power of the load, in watts.

Inductive reactive power QL

The inductive reactive power QL, in vars. Specify a positive value, or zero.

Capacitive reactive power QC

The capacitive reactive power QC, in vars. Specify a positive value, or zero.

Measurements

Select **Branch voltage** to measure the voltage across the Series RLC Load block terminals.

Select **Branch current** to measure the current flowing through the Series RLC Load block.

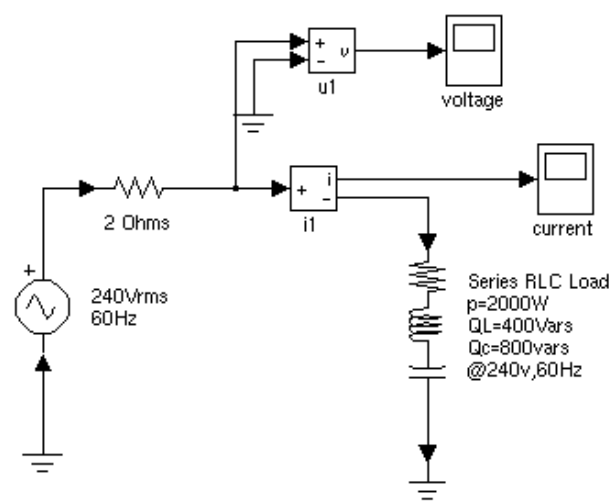
Select **Branch voltage and current** to measure the voltage and the current of the Series RLC Load block.

Place a Multimeter block in your model to display the selected measurements during the simulation. In the **Available Measurement** listbox of the Multimeter block, the measurement will be identified by a label followed by the block name:

Measurement	Label
Branch voltage	ub:
Branch current	i b:

Example

The following circuit uses a Series RLC Load block to implement a simple load. This example is available in the psbseri esl oad. mdl file.



Series RLC Load

See Also

Series RLC Branch, Parallel RLC Branch, Parallel RLC Load

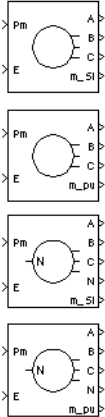
Purpose

Model the dynamics of a simplified three-phase synchronous machine.

Library

Machines

Description



The Simplified Synchronous Machine block models both the electrical and mechanical characteristics of a simple synchronous machine.

The electrical system for each phase consists of a voltage source in series with an RL impedance, which implements the internal impedance of the machine. The value of R can be zero but the value of L must be positive.

The Simplified Synchronous Machine block implements the mechanical system described by

$$\Delta\omega(t) = \frac{1}{2H} \int_0^t (Tm - Te) dt - Kd\Delta\omega(t)$$

$$\omega(t) = \Delta\omega(t) + \omega_0$$

where:

$\Delta\omega$ = Speed variation with respect to speed of operation

H = Constant of inertia

Tm = Mechanical torque

Te = Electromagnetic torque

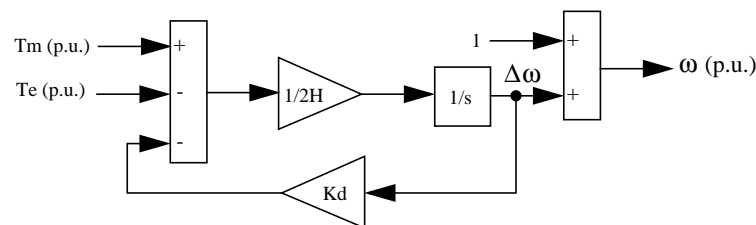
Kd = Damping factor

$\omega(t)$ = Mechanical speed of the rotor

ω_0 = Speed of operation (1 p.u.)

Although the parameters can be entered in either SI units or per unit in the dialog box, the internal calculations are done in per unit. The following block diagram illustrates how the mechanical part of the model is implemented. Notice that the model computes a deviation with respect to the speed of operation, and not the absolute speed itself.

Simplified Synchronous Machine



Dialog Box and Parameters

In the **powerlib** library you can choose between two Simplified Synchronous Machine blocks to specify the electrical and mechanical parameters of the model.

Block Parameters: Simplified Synchronous Machine pu Units

Simplified Synchronous Machine (mask) (link)

Implements a 3-phase simplified synchronous machine. Machine is modelled as an internal voltage behind a R-L impedance. Stator windings are connected in wye to an internal neutral point.

1st input: Simulink signal: mechanical power supplied to the machine (p.u.) >0 for generator mode, <0 for motor mode

2nd input: Simulink signal: internal voltage (p.u.)

First 3 outputs: Machine terminals = phases a, b and c

4th output: Simulink measurement output = vector [12x1] containing:

- 1-3: Line currents flowing out of machine ia, ib, ic (p.u.)
- 4-6: Terminal voltages va, vb, vc (p.u.)
- 7-9: Internal voltages ea, eb, ec (p.u.)
- 10: Rotor angle theta (rad)
- 11: Rotor speed wm (p.u.)
- 12: Electrical power Pe (p.u.)

Parameters

Connection type: 3-wire Y

Nom. power, L-L volt., and freq. [Pn(VA) Vn(Vrms) fn(Hz)]:

[1000e6, 315e3, 60]

Inertia, damping factor and pairs of poles [H(sec) Kd(p) p()]:

[inf, 0.2]

Internal impedance [R(pu) X(pu)]:

[0.02, 1]

Init. cond. [dw(%) th(deg) ia,ib,ic(pu) pha,phb,phc(deg)]:

[0 0 0.0, 0 0, 0]

OK Cancel Help Apply

Per Unit (p.u.) Dialog Box

Block Parameters: Simplified Synchronous Machine SI Units

Simplified Synchronous Machine (mask) (link)

Implements a 3-phase simplified synchronous machine. Machine is modelled as an internal voltage behind a R-L impedance. Stator windings are connected in wye to an internal neutral point.

1st input: Simulink signal: mechanical power supplied to the machine (W) >0 for generator mode, <0 for motor mode

2nd input: Simulink signal: RMS value of phase-to-phase internal voltage (V)

First 3 outputs: Machine terminals = phases a, b and c

4th output: Simulink measurement output = vector [12x1] containing:

- 1-3: Line currents flowing out of machine ia, ib, ic (A)
- 4-6: Terminal voltages va, vb, vc (V)
- 7-9: Internal voltages ea, eb, ec (V)
- 10: Rotor angle theta (rad)
- 11: Rotor speed wm (rad/s)
- 12: Electrical power Pe (W)

Parameters

Connection type: 3-wire Y

Nom. power, L-L volt., and freq. [Pn(VA) Vn(Vrms) fn(Hz)]:

[1000e6, 315e3, 60]

Inertia, damping factor and pairs of poles [J(kg.m^2) Kd(p) p()]:

[inf, 0.2]

Internal impedance [R(ohm) L(H)]:

[1.9845, 263.15e-3]

Init. cond. [dw(%) th(deg) ia,ib,ic(A) pha,phb,phc(deg)]:

[0 0 0.0, 0 0, 0]

OK Cancel Help Apply

SI Units Dialog Box

Connection type

Specifies number of wires used in three-phase wye connection: either three-wire (neutral not accessible) or four-wire (neutral is accessible).

Nominal

The nominal apparent power P_n (VA), frequency f_n (Hz), and rms line-to-line voltage V_n (V). Used to compute nominal torque and convert SI units to p.u.

Mechanical

The moment of inertia (N.m or p.u.) and damping factor. The damping factor has been scaled to act like the damping factor of a second order system. This means, for example, that for no overshoot and minimum settling time, a damping factor of 0.9 would be used.

Internal impedance

The resistance R (Ω or p.u.) and reactance L (H or p.u.) for each phase.

Initial conditions

The initial speed deviation (% of nominal), rotor angle (deg), line currents magnitudes (A or p.u.) and phase angles (deg). These values can be computed by the load flow utility of the Powergui block.

Note These two blocks simulate exactly the same simplified synchronous machine model; the only difference is the way of entering the parameter units

Inputs and Outputs

The first input of the Simplified Synchronous Machine block is the mechanical power supplied to the machine. This input can be a constant or the output of the Hydraulic Turbine and Governor block. The frequency of the voltage sources depends on the mechanical speed of the machine. The amplitude of these voltages is given by the second input of the block, which can be a constant or the output of a voltage regulator. If you use SI units these two inputs should be in watts and volts phase-to-phase rms. If you use p.u. both inputs should be in p.u.

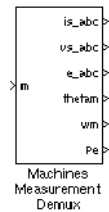
The first three outputs are the electrical terminals of the stator. The last output of the block is a vector containing the following 12 variables:

- 1-3: Line currents (flowing out of the machine) i_a , i_b , i_c
- 4-6: Terminal voltages v_a , v_b , v_c
- 7-9: Internal voltages e_a , e_b , e_c

Simplified Synchronous Machine

- 10: Mechanical angle θ
- 11: Rotor speed ω
- 12: Electrical power P_e

These variables can be demultiplexed by using the special Machines Measurement Demux block provided in the Machines library.



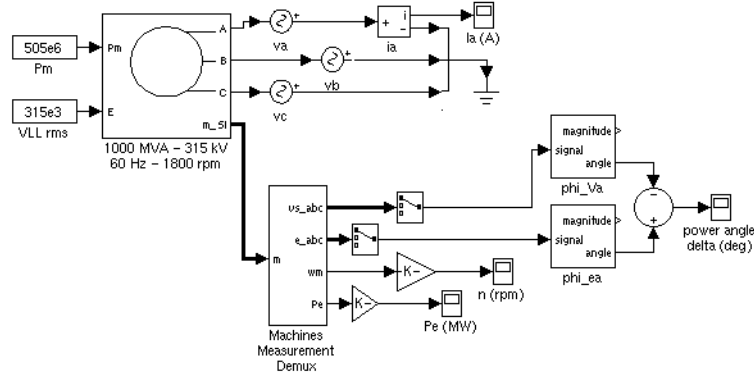
Assumptions

The electrical system of the Simplified Synchronous Machine block consists solely of a voltage source behind a synchronous reactance and resistance. All the other self and magnetizing inductances of the armature, field and damping windings are neglected. The three voltage sources and RL impedance branches are Y-connected (3 wires). The load may or may not be balanced.

Example

The following example uses the Simplified Synchronous Machine block. In this example, the Simplified Synchronous Machine block, which represents a 1000 MVA 315 kV equivalent source, is connected to an infinite bus (three AC Voltage Source blocks) and is used as a synchronous generator. The internal resistance of the machine is set to 0.02 p.u., or 1.9845 ohms. Its inductance is set in such a way that the total impedance is 1 p.u. ($L=263.15$ mH). The inertia of the machine is 56290 kg.m^2 . This example is included in the `psbsimplealt.mdl` file.

Simplified Synchronous Machine



In this example, the machine has an initial speed deviation of 0.5%. The initial mechanical angle and phase currents i_a , i_b and i_c are set to zero. The power transfer between the machine and the bus is given by the following relation:

$$P_T = \frac{V_1 \times V_2}{X} \times \sin \delta$$

P_T = power transfer (500 MW)

V_1 = machine voltage (315 kV)

V_2 = bus voltage (315 kV)

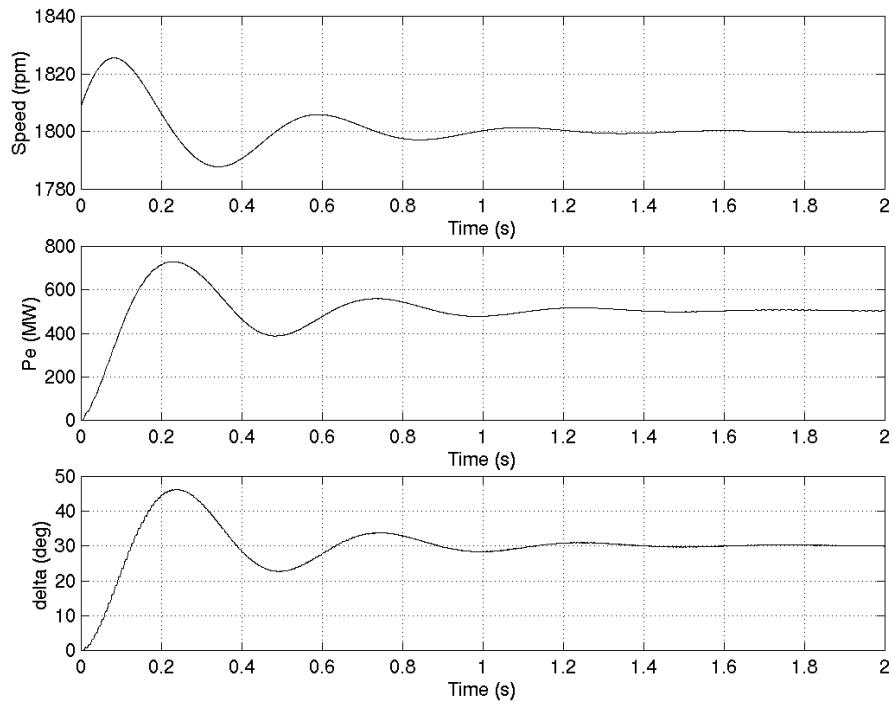
X = total reactance (263.15 mH x 120 x π)

δ = electrical angle difference (power angle δ) between machine internal voltage and terminal voltage

With the above parameters, the steady-state internal voltage is 30° ahead of the terminal voltage ($\delta = +30^\circ$). The machine is supplied with 505 MW of mechanical power in order to compensate for its resistive losses. The electrical angle δ is displayed as the phase difference between the internal and terminal voltages of phase A. With simulation parameters set as follows, the results shown below are obtained:

- Integrator type: Stiff, ode15s
- Stop time: 2.0
- Integration options: Use default settings

Simplified Synchronous Machine



The speed vs. time graph clearly shows that the machine is initially running at a speed of 1.005 p.u. (1809 rpm) and that speed stabilizes itself at its nominal value of 1800 rpm. As expected, the electrical power supplied by the machine stabilizes at 500 MW. The power angle δ also settles at its expected value of 30° . The mechanical system is clearly under-damped, the damping factor being set to 0.3.

See Also

Excitation System, Hydraulic Turbine and Governor, Powergui,
Steam Turbine and Governor, Synchronous Machine

Purpose

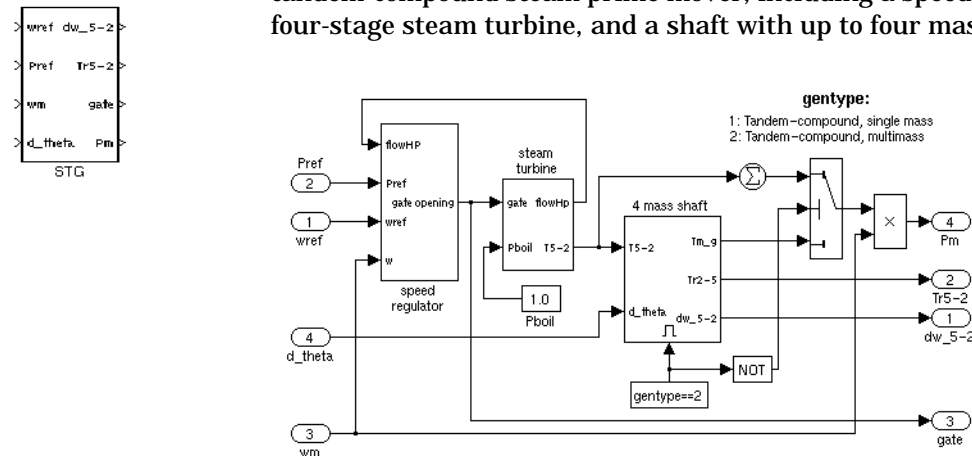
Model the dynamics of a speed governing system, steam turbine, and multimass shaft.

Library

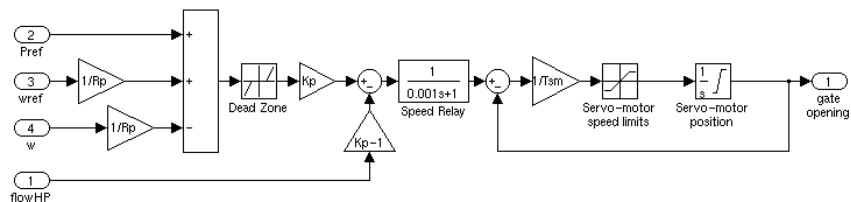
Machines

Description

The Steam Turbine and Governor block implements a complete tandem-compound steam prime mover, including a speed governing system, a four-stage steam turbine, and a shaft with up to four masses.



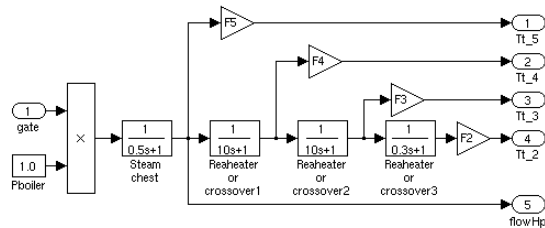
The speed governing system consists of a proportional regulator, a speed relay and a servo-motor controlling the gate opening. It is similar to one of the models proposed in [1].



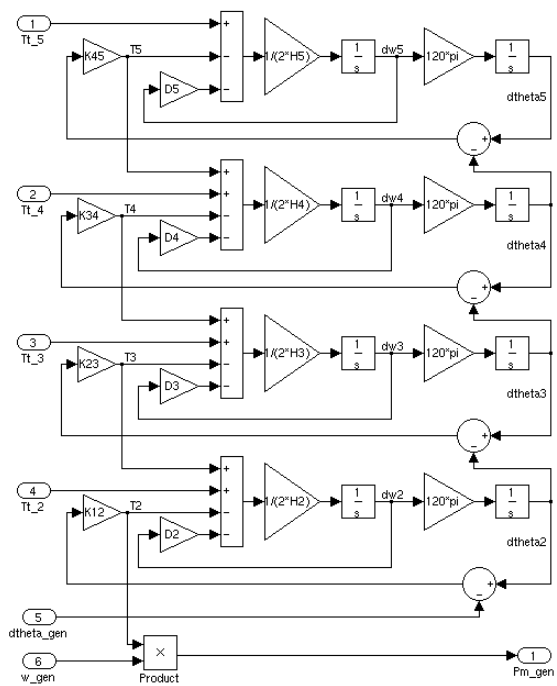
The steam turbine has four stages, each modeled by a first-order transfer function. The first stage represents the steam chest while the three other stages represent either reheaters or crossover piping. The boiler is not modeled

Steam Turbine and Governor

and boiler pressure is constant at 1.0 p.u. Fractions F2 to F5 are used to distribute the turbine power to the various shaft stages.



The shaft models a four mass system, which is coupled to the mass in the Synchronous Machine model for a total of five masses. The machine's mass is labeled mass #1. The mass in the Steam Turbine and Governor block, which is closest to the machine's mass is mass #2, while the mass farthest from the machine is mass #5. The shaft is characterized by mass inertias H , damping factors D and rigidity coefficients K . If you choose to simulate a single mass shaft, the entire four mass shaft subsystem in the Steam Turbine and Governor block will be disabled and all the torque from the turbine will be added together and applied to the machine's mass.



Steam Turbine and Governor

Dialog Box and Parameters

Block Parameters: STG

Steam Turbine and Governor (mask) (link)

Implements a complete tandem-compound steam prime mover system, including speed regulator, steam turbine, and a shaft with up to four masses. The generator's mass is labeled mass #1 and is not included here. The shaft mass closest to the generator is #2; the farthest is #5. If a mass is not to be included, set its inertia H to zero. The damping factor and rigidity coefficients corresponding to omitted masses are not considered and can be left as is. When masses are omitted, the remaining system is "compressed" towards the generator, i.e., if only two masses are used, they will be masses #2 and #3. The input data for the masses considered is shifted accordingly.

Press Help for inputs and outputs description.

Parameters

Generator type Tandem-compound (multi-mass)

Regulator gain, perm. droop, dead zone [Kp Rp(p.u.) Dz(p.u.)]:
[1 0.05 0]

Speed relay and servo-motor time constants [Tsr Tsm] (s):
[0.001 0.15]

Gate opening limits [vgmin,vgmax (p.u./s) gmin,gmax (p.u.)]:
[-0.1 0.1 0 4.496]

Steam turbine time constants [T2 T3 T4 T5] (s):
[0 10 3.3 0.5]

Turbine torque fractions [F2 F3 F4 F5]:
[0 0.36 0.36 0.28]

Coelf. of inertia [H2 H3 H4 H5] (s):
[0.046 1.430 1.465 0.192]

Stiffness coeff. [K12 K23 K34 K45] (pu/rad):
[90.44 53.88 64.59 62.94]

Damping factors [D2 D3 D4 D5] (p.u. T/p.u. dw):
[0.0115 0.3575 0.3663 0.0137]

Initial power and generator rotor angle [Pm0 (p.u.) th0(deg)]:
[250.35/555 -51.2424]

OK

Cancel

Help

Apply

- ← Select either a single mass or multimass shaft.
- } These are the speed governing system parameters.
- } These are the steam turbine parameters. Torque fractions must total one.
- } These are the multimass shaft parameters. They will not be visible if you choose a single mass shaft. Enter zero for inertia (H) if a mass is not to be simulated.
- ← If you select a single mass shaft, only initial power is required.

Generator type

Specifies rotor type: single mass or multimass tandem-compound. If you choose a single mass system, the multimass shaft subsystem in the Steam Turbine and Governor block is disabled and the turbine's output torques are summed together and applied to the single mass in the Synchronous Machine block.

Regulator

The gain K_p , permanent droop R_p (p.u.), and dead-zone width D_z (p.u.). Set gain to three if you want to use the steam flow feedback loop. Otherwise, set gain to one.

Time constants

The speed relay and gate servo-motor time constants T_{sr} (s) and T_{sm} (s).

Gate Limits

The minimum and maximum gate opening speed vg_{min} and vg_{max} (both in p.u./s), and minimum and maximum gate opening g_{min} and g_{max} (both in p.u.).

Turbine time constants

The turbine time constants T_2 to T_5 (s). Numbered consistently with turbine torque fractions and mass numbers, i.e., T_5 is the time constant of the first turbine stage, which models the steam chest.

Turbine torque fractions

The turbine torque fractions F_2 to F_5 . Must total one, otherwise an error message will appear. Fraction numbers correspond to mass numbers, i.e., F_2 is the fraction of torque to be applied to mass #2 of the multimass shaft.

Multi-mass shaft

Only visible if generator type is multimass. Coefficients of inertia H_2 to H_5 (s), stiffness coefficients K_{12} to K_{45} (p.u./rad), and damping factors D_2 to D_5 (p.u. torque /p.u. speed deviation) associated to the masses of the multimass shaft. K_{12} corresponds to the rigidity coefficient between masses 1 and 2, and so on.

Note If you do not want to simulate all four masses in the multimass shaft, simply set the inertia of unwanted masses to zero. The rigidity coefficient and damping factor corresponding to omitted masses are not considered. When masses are not simulated, the remaining system is “compressed” toward the generator, i.e., if only 2 masses are used (excluding the generator), they will be masses #2 and #3. The input data for the masses considered is shifted accordingly. In any case, inertias must be consistent with torque fractions. You can't set an inertia to zero and set the corresponding torque fraction to a nonzero value. However, you can set a torque fraction to zero and set the corresponding mass inertia to a nonzero value

Steam Turbine and Governor

Initial conditions

If multimass shaft, the initial mechanical power P_{m0} (p.u.) and initial generator angle θ_{e0} (deg). If single mass shaft, only initial mechanical power.

Initial mechanical power can be computed by the load flow utility of the Powergui block. Initial angle is also computed by the load flow utility and is written in the associated Synchronous Machine block's dialog box.

Inputs and Outputs

The first input is the speed reference, in p.u. This will normally be connected to a Constant block and the value set to 1.0 p.u.

The second input is the electrical power reference, in p.u. This will normally be set to a constant value corresponding to the initial active power drawn from the Synchronous Machine block connected to the Steam Turbine and Governor block.

The third input is the generator's speed, in p.u. This is one of the signals in the last output of the Synchronous Machine model (internal variables).

The fourth input is the generator's power angle deviation. It is also one of the signals in the last output of the Synchronous Machine model (internal variables).

The first output is a vector containing the speed deviations, in p.u., of masses five to two, in that order.

The second output is also a vector containing the torques, in p.u., transmitted by masses five to two.

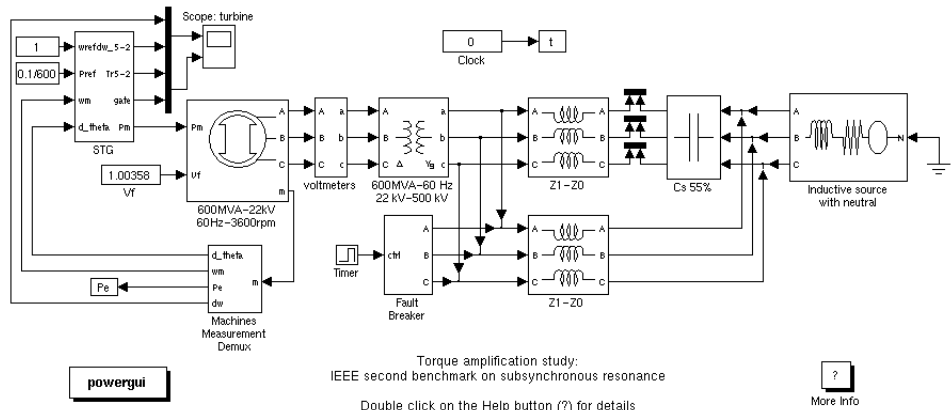
The third output is the mechanical power, in p.u., which you must connect to the first input of a Synchronous Machine block.

Example

This example, available in the `psbthermal.mdl` file, illustrates the use of the Steam Turbine and Governor block. This system is an IEEE benchmark used to study subsynchronous resonance and particularly torque amplification after a fault on a series-compensated power system [2]. It consists in a single generator connected to an infinite bus via two transmission lines, one of which is series-compensated. The subsynchronous mode introduced by the compensation capacitor after a fault has been applied and cleared excites the oscillatory torsional modes of the multi-mass shaft and the torque

amplification phenomenon can be observed. Open the simulink diagram by typing psbthermal.

This system is slightly different from the one presented in [2]. Since we are using the Synchronous Machine's mass as the first mass, we can't model the exciter's mass as is done in [2]. Therefore, our system has only three masses, representing the generator's rotor (mass #1), and the turbine's low and high pressure stages (masses #2 and #3, respectively).



In order to start the simulation in steady-state, a load flow was performed on the system, setting the generator as a **PV generator** with initial power of 100 kW ($1e5$ W). This is done to simulate an initially unloaded generator. The load flow returns initial mechanical power of 100 010 W. This value was converted in p.u. by dividing it by the generator's nominal VA rating ($600e6$ VA) and the result was entered as the first initial condition in the Steam Turbine and Governor block. The second initial condition is the generator's initial angle. This value is computed by the load flow and is written in the initial conditions vector of the generator. After the load flow is finished, you can open the Synchronous Machine block's dialog box and copy the initial angle in the Steam Turbine and Governor block's dialog box. The Steam Turbine and Governor block is now correctly initialized. The electrical power (load) reference, the second input of the Steam Turbine and Governor block, is set to the desired electrical power supplied by the generator, in p.u. ($1e5/600e6$, or $0.1/600$).

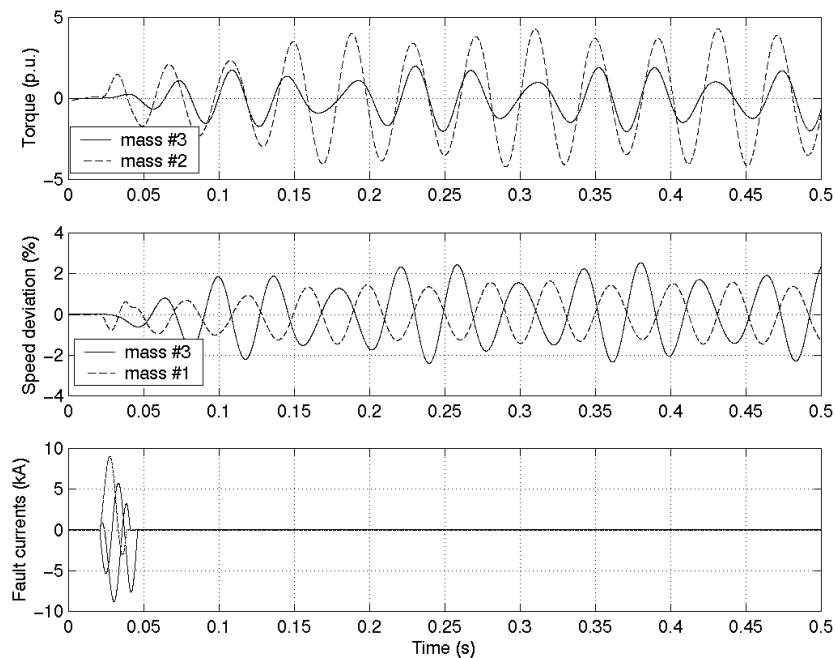
Steam Turbine and Governor

This test is performed without regulators. The speed governing system is forced to output a constant value by setting the gate opening limits very close to each other, around the initial gate opening which is also the initial mechanical power in p.u. (100 010/600e6, or 0.00016668 p.u.). The machine's excitation voltage is also set to a constant value (1.00358 p.u.), which is computed by the load flow.

Set the simulation parameters as follows:

- Integrator type: Variable-step, ode23tb
- Stop time: 0.5 s
- Integration options: use default options, except for Max step size, which you set to 50e-6. This is not absolutely required, but the simulation runs faster if this value is set as prescribed.

Run the simulation by choosing **Start** from the **Simulation** menu. Once the simulation is completed, observe the mass speed deviations and torques and the fault current.



The peak values of all these signals correspond within 3% to those given in Table 5, case 1A, of [2]. The torque amplification is clearly observed on all masses of the shaft system. The high-pressure mass (#3) transmits a peak torque of 1.91 p.u. to the low-pressure mass (#2), while the low-pressure mass transmits a peak torque of 4.05 p.u. to the generator's rotor (mass #1).

References

[1] IEEE committee report, "Dynamic models for steam and hydro turbines in power system studies", *IEEE Transactions on Power Apparatus and Systems*, vol. PAS-92, no. 6, 1973, pp. 1904-1915.

[2] IEEE Subsynchronous resonance working group, "Second benchmark model for computer simulation of subsynchronous resonance", *IEEE Transactions on Power Apparatus and Systems*, vol. PAS-104, no. 5, 1985, pp. 1057-1066.

See Also

Excitation System, Hydraulic Turbine and Governor, Powergui, Synchronous Machine

Surge Arrester

Purpose Implement a metal-oxide surge arrester.

Library Elements

Description

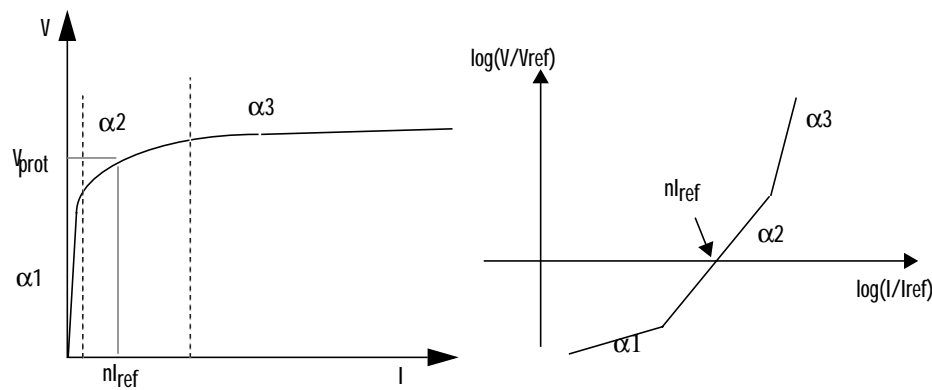


The Surge Arrester block implements a highly nonlinear resistor used to protect power equipments against overvoltages. For applications requiring high power dissipation, several columns of metal-oxide discs are connected in parallel inside the same porcelain housing. The nonlinear V-I characteristic of each column of the surge arrester is modeled by a combination of three exponential functions of the form:

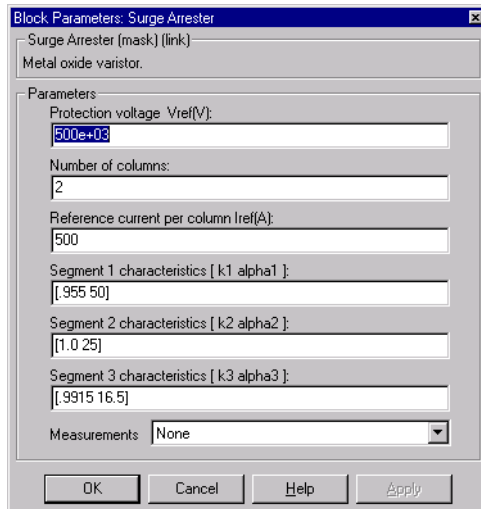
$$\frac{V}{V_{ref}} = K_i \left(\frac{I}{I_{ref}} \right)^{1/\alpha_i}$$

The protection voltage obtained with a single column is specified at a reference current (usually 500A or 1kA). Default parameters k and α given in the dialog box fit the average V-I characteristic provided by the main metal oxide arrester manufacturers and they do not change with the protection voltage. The required protection voltage is obtained by adding discs of zinc oxide in series in each column.

This V-I characteristic is graphically represented as follows (on a linear scale and on a logarithmic scale).



Dialog Box and Parameters



Protection voltage V_{ref}

The protection voltage of the Surge Arrester block, in volts (V).

Number of columns

The number of metal-oxide disc columns, the minimum is one.

Reference current per column I_{ref}

The reference current of one column used to specify the protection voltage, in amperes (A).

Segment 1 characteristic

The K and α parameters of segment one.

Segment 2 characteristic

The K and α parameters of segment two.

Segment 3 characteristic

The K and α characteristics of segment three.

Measurements

Select **Branch voltage** to measure the voltage across the Surge Arrester block terminals.

Surge Arrester

Select **Branch current** to measure the current flowing through the Surge Arrester block.

Select **Branch voltage and current** to measure the surge arrester voltage and current.

Place a Multimeter block in your model to display the selected measurements during the simulation. In the **Available Measurement** listbox of the Multimeter block, the measurement will be identified by a label followed by the block name.

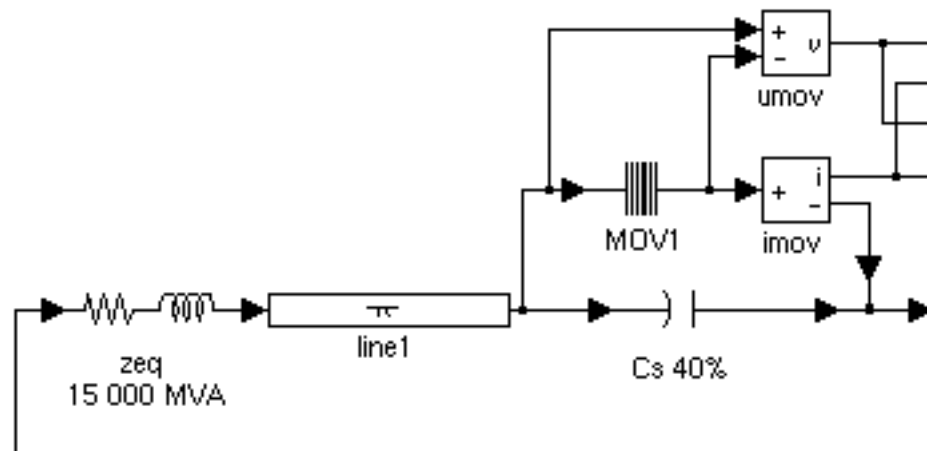
Measurement	Label
Branch voltage	ub:
Branch current	i b:

Limitations

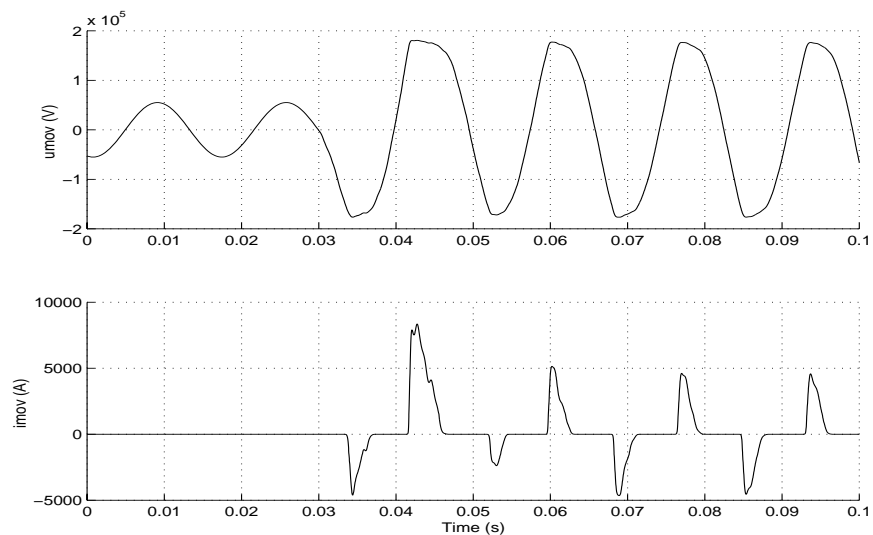
The Surge Arrester block is modeled as a current source driven by the voltage appearing across its terminals. Therefore, it cannot be connected in series with an inductor or another current source. As the Surge Arrester block is highly nonlinear, a stiff integrator algorithm must be used to simulate the circuit. 0de15s or 0de23tb with default parameters usually give the best simulation speed. For continuous simulation, in order to avoid an algebraic loop, the voltage applied to the nonlinear resistance is filtered by a first order filter with a time constant of 0.01 micro seconds. This very fast time constant does not affect significantly the result accuracy. When the Surge Arrester block is used in a discrete system, a time delay of one simulation step is used. This delay may cause numerical oscillations if the sample time is too large.

Example

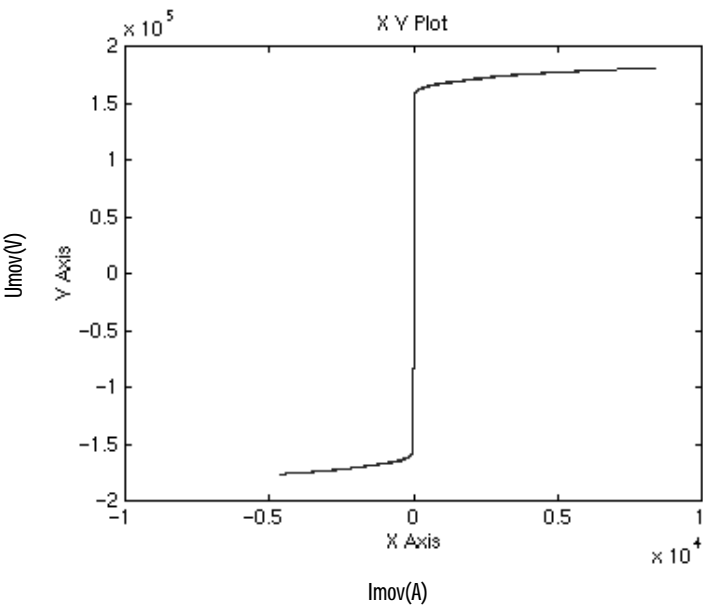
The example provided in the psbarrester.mdl demonstration file illustrates the use of metal oxide varistors (MOV) on a 735 kV series compensated network. Only one phase of the network is represented. The capacitor connected in series with the line is protected by a 30 column arrester. At t=0.03 seconds, a fault is applied at the load terminals. The current increases in the series capacitor and produces an overvoltage which is limited by the Surge Arrester block. Then the fault is cleared at t=0.3 second.



At fault application, the resulting overvoltage makes the MOV to conduct. The waveforms displayed by U_{mov} and I_{mov} measurements as well as the V-I characteristic plotted by the X-Y scope are shown below:



Surge Arrester



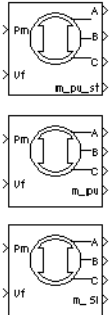
Purpose

Model the dynamics of a three-phase round rotor or salient-pole synchronous machine.

Library

Machines

Description

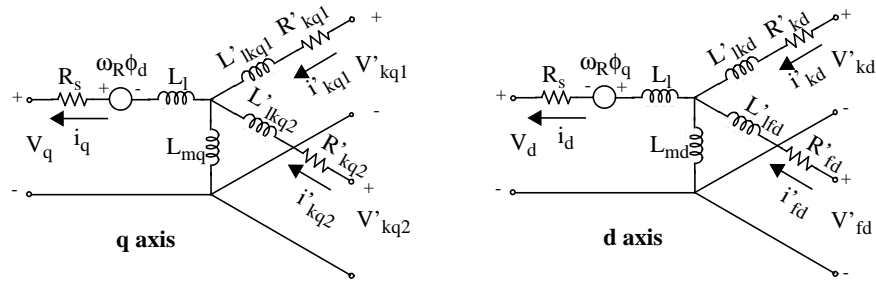


The Synchronous Machine block operates in generating or motoring modes. The operating mode is dictated by the sign of the mechanical power (positive for generating, negative for motoring). The electrical part of the machine is represented by a sixth-order state-space model and the mechanical part is the same as in the Simplified Synchronous Machine block.

The model takes into account the dynamics of the stator, field and damper windings. The equivalent circuit of the model is represented in the rotor reference frame (qd frame). All rotor parameters and electrical quantities are viewed from the stator. They are identified by primed variables. The subscripts used are defined as follows:

- d, q : d and q axis quantity
- R, s : Rotor and stator quantity
- l, m : Leakage and magnetizing inductance
- f, k : Field and damper winding quantity

The electrical model of the machine is



with the following equations:

Synchronous Machine

$$V_d = R_s i_d + \frac{d}{dt} \Phi_d - \omega_R \Phi_q$$

$$V_q = R_s i_q + \frac{d}{dt} \Phi_q + \omega_R \Phi_d$$

$$V'_{fd} = R'_{fd} i'_{fd} + \frac{d}{dt} \Phi'_{fd}$$

$$V'_{kd} = R'_{kd} i'_{kd} + \frac{d}{dt} \Phi'_{kd}$$

$$V'_{kq1} = R'_{kq1} i'_{kq1} + \frac{d}{dt} \Phi'_{kq1}$$

$$V'_{kq2} = R'_{kq2} i'_{kq2} + \frac{d}{dt} \Phi'_{kq2}$$

$$\Phi_d = L_d i_d + L_{md} (i'_{fd} + i'_{kd})$$

$$\Phi_q = L_q i_q + L_{mq} i'_{kq}$$

$$\Phi'_{fd} = L'_{fd} i'_{fd} + L_{md} (i_d + i'_{kd})$$

$$\Phi'_{kd} = L'_{kd} i'_{kd} + L_{md} (i_d + i'_{fd})$$

$$\Phi'_{kq1} = L'_{kq1} i'_{kq1} + L_{mq} i_q$$

$$\Phi'_{kq2} = L'_{kq2} i'_{kq2} + L_{mq} i_q$$

Dialog Box and Parameters

In the **powerlib** library you can choose between three Synchronous Machine blocks to specify the parameters of the model.

Fundamental Parameters in SI Units

Block Parameters: Synchronous Machine SI Fundamental

Synchronous Machine (mask)
Implements a 3-phase synchronous machine modelled in the dq rotor reference frame. Stator windings are connected in wye to an internal neutral point. Press Help for inputs and outputs description.

Parameters

Rotor type: Salient-pole

Nom. power, volt., freq. and field cur. [Pn(VA) Vn(Vrms) fn(Hz)]
[187E6 13800 60 1087]

Stator [R_s(ohm) L_lL_{md}L_{mq}(H)]:
[2.9069E-03 3.0892E-04 3.2164E-03 9.7153E-04]

Field [R_f(ohm) L_{fd}(H)]:
[5.9013E-04 3.0712E-04]

Dampers [R_kd₁L_kd₁ R_kq₁L_kq₁] (R=ohm,L=H):
[1.1900E-02 4.9076E-04 2.0081E-02 1.0365E-03]

Inertia, friction factor and pole pairs [J(kg.m²) F(N.m.s) p()]:
[inf 0 20]

Init. cond. [dw(%) th(deg) ia,ib,ic(A) pha,phb,phc(deg) Vf(V)]:
[0 0 0 0 0 0 0 70.3192]

☒ Simulate saturation
Saturation parameters [ifd1,ifd2,... (A) ; vt1,vt2,... (VLL rms)]:
[695.64,774.7,917.5,1001.6,1082.2,1175.9,1293.6,1430.2,1583]

☐ Display Vf_d which produces nominal V_t

OK Cancel Help Apply

Rotor type

Specifies rotor type: salient-pole or round (cylindrical). This choice affects the number of rotor circuits in the q-axis (damper windings).

Nominal

The total three-phase apparent power P_n (VA), rms line-to-line voltage V_n (V), frequency f_n (Hz), and field current i_{fn} (A).

The nominal field current is the current that produces nominal terminal voltage under no-load conditions. This model was developed with all quantities viewed from the stator. The nominal field current makes it possible to compute the transformation ratio of the machine, which allows you to apply the field voltage viewed from the rotor, as in true life. This also allows the field current, which is a variable in the output vector of the model, to be viewed from the rotor. If the value of the nominal field current is not known, you must enter zero. Since the transformation ratio can't be determined in this case, you will have to apply the field voltage as viewed

Synchronous Machine

from the stator and the field current in the output vector will also be viewed from the stator.

Stator

The resistance R_s (Ω), leakage inductance L_{ls} (H), and d-axis and q-axis magnetizing inductances L_{md} (H) and L_{mq} (H).

Field

The field resistance R_f (Ω) and leakage inductance L_{lfd} (H), both referred to the stator.

Dampers

The d-axis resistance R_{kd} (Ω) and leakage inductance L_{lkd} (H), the q-axis resistance R_{kq1} (Ω) and leakage inductance L_{lkq1} (H), and (if round rotor only) the q-axis resistance R_{kq2} (Ω) and leakage inductance L_{lkq2} (H). All these values are referred to the stator.

Mechanical

The inertia coefficient J (kg.m^2), damping coefficient D (N.m.s./rad), and number of pole pairs p .

Initial conditions

The initial speed deviation $\Delta\omega$ (% of nominal speed), electrical angle of the rotor θ_e (deg), line currents magnitudes i_a , i_b , i_c (A) and phase angles ϕ_a , ϕ_b , ϕ_c (deg), and the initial field voltage V_f (V).

You can specify the initial field voltage in one of two ways. If you know the nominal field current (first line, last parameter) enter in the dialog box the initial field voltage in Volts DC referred to the rotor. Otherwise, enter a zero as nominal field current, as explained earlier, and specify the initial field voltage in Volts DC referred to the stator. The nominal field voltage viewed from the stator can be easily determined by checking the **Display Vfd which produces a nominal Vt** check box at the bottom of the dialog box.

Simulate saturation

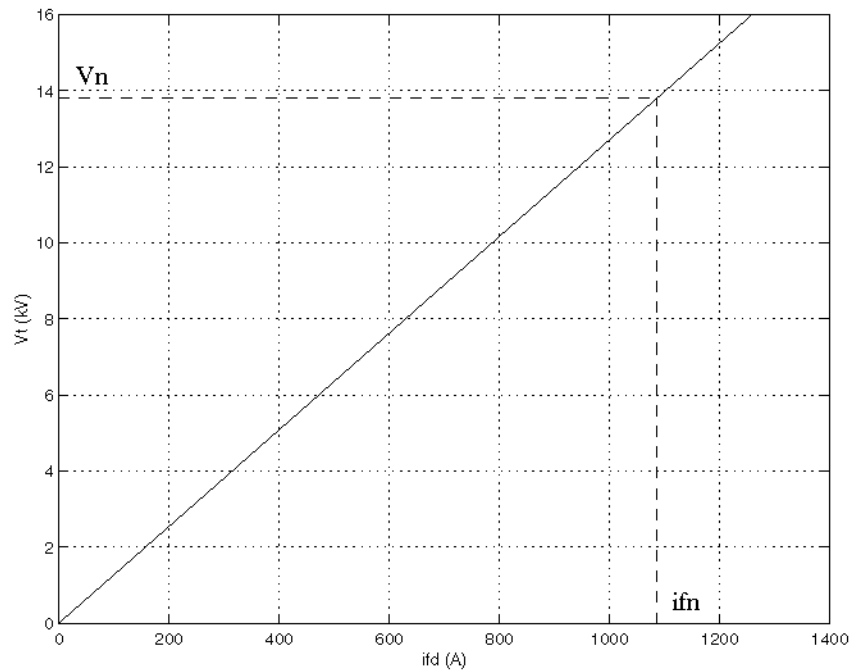
Specifies if magnetic saturation of rotor and stator iron is to be simulated or not.

Saturation

The no-load saturation curve parameters. Magnetic saturation of stator and rotor iron is modeled by a non-linear function, in this case a polynomial, using points on the no-load saturation curve. You must enter a 2 by n matrix, where n is the number of points taken from the saturation curve. The first row of this matrix contains the values of field currents while the second row contains values of corresponding terminal voltages. The first point (first column of the matrix) must correspond to the point where the effect of saturation begins. You must check the **Simulate saturation** check box to simulate saturation. This check box allows you to enter the matrix of parameters for simulating the saturation. If you do not want to model saturation in your simulation, simply don't check the **Simulate saturation** check box. In this case the relationship between I_{fd} and V_t obtained is linear (no saturation).

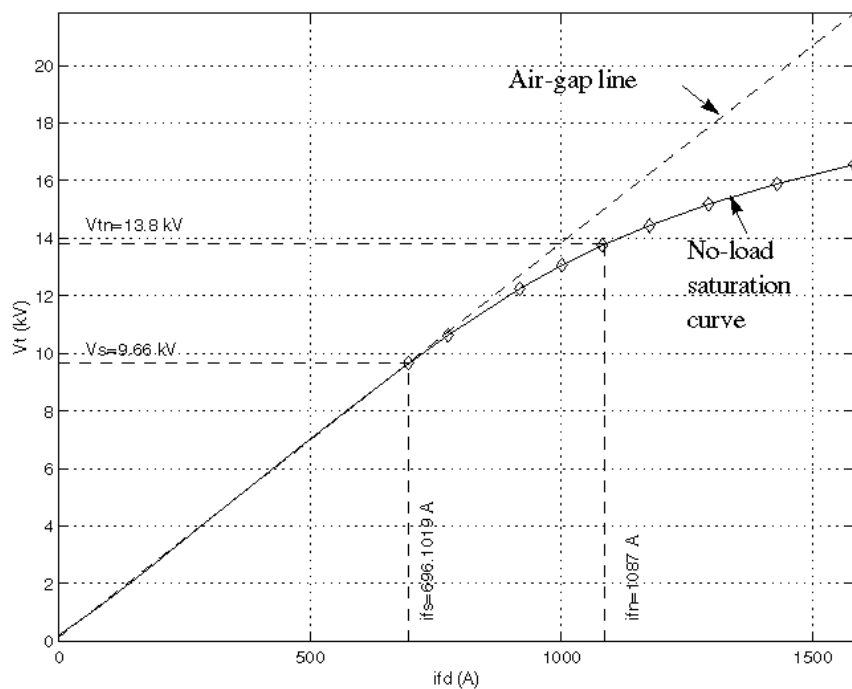
As an example, without saturation, a typical curve might be as shown below. Here I_{fn} is 1087A and V_n is 13800 V rms line-to-line, which is also 11268 V peak line-to-neutral.

Synchronous Machine



When saturation is modeled, a polynomial is fitted to the curve corresponding to the matrix of points you enter. The more points you enter, the better the fit to the original curve.

The next figure illustrates this graphically (the diamonds are the actual points entered in the dialog box).



In this particular case, the following values were used:

- $i_{fn} = 1087$ A
- $i_{fd} = [695.64, 774.7, 917.5, 1001.6, 1082.2, 1175.9, 1293.6, 1430.2, 1583.7]$ A
- $V_t = [9660, 10623, 12243, 13063, 13757, 14437, 15180, 15890, 16567]$ V

Synchronous Machine

Fundamental Parameters in p.u.

Block Parameters: Synchronous Machine pu Fundamental

Synchronous Machine (mask) (link)
Implements a 3-phase synchronous machine modelled in the dq rotor reference frame. Stator windings are connected in wye to an internal neutral point. Press Help for inputs and outputs description.

Parameters

Rotor type: Salient-pole

Nom. power, L-L volt. and freq. [Pn(VA) Vn(Vrms) fn(Hz)]:
[187E6 13800 60]

Stator [Rs Ld Lmd Lmq] (pu):
[2.8544E-03 0.11436 1.1906 0.35964]

Field [Rf Lfd] (pu):
[5.7947E-04 0.11369]

Dampers [Rkd,Lkd Rkq1,Lkq1] (pu):
[1.1685E-02 0.18167 1.9718E-02 0.38371]

Coeff. of inertia, friction factor and pole pairs [H(s) F(pu) p()]:
[inf 0 20]

Init. cond. [dw(%) th(deg) ia,ib,ic(pu) pha,phb,phc(deg) Vf(pu)]:
[0 0 0 0 0 0 0 0 1]

☒ Simulate saturation

Saturation parameters [ifd1,ifd2,... (p.u.) ; vt1,vt2,... (p.u.)]:
[0.6404,0.7127,0.8441,0.9214,0.9956,1.082,1.19,1.316,1.457,0]

OK

Cancel

Help

Apply

Rotor type

Specifies rotor type: salient-pole or round (cylindrical).

Nominal

Total three-phase apparent power (VA), rms line-to-line voltage (V), frequency (Hz), and field current (A).

This line is identical to the first line of the fundamental parameters in SI dialog box, except that you don't specify a nominal field current. This value is not required here because we don't need the transformation ratio. Since rotor quantities are viewed from the stator, they are converted to p.u. using the stator base quantities derived from the three nominal parameters above.

Stator, field and dampers

Contain exactly the same parameters as in the previous dialog box, but they are expressed here in p.u. instead of SI units.

Mechanical

The inertia constant H (s), where H is the ratio of energy stored in the rotor at nominal speed over the nominal power of the machine, the damping coefficient D (p.u. torque/p.u. speed deviation), and the number of pole pairs p .

Initial conditions, Simulate saturation, Saturation

The same initial conditions and saturation parameters as in the S.I. units dialog box, but all values are expressed in p.u. instead of SI units. For saturation, the nominal field current multiplied by the d-axis magnetizing inductance and nominal rms line-to-line voltage are the base values for the field current and terminal voltage, respectively.

Standard Parameters in p.u.

Block Parameters: Synchronous Machine pu Standard

Synchronous Machine (mask) (link)
Implements a 3-phase synchronous machine modelled in the dq rotor reference frame. Stator windings are connected in wye to an internal neutral point. Press Help for inputs and outputs description.

Parameters

Rotor type: Salient-pole

Nom. power, L-L volt. and freq. [Pn(VA) Vn(Vrms) fn(Hz)]:
[187E6 13800 60]

Reactances [XdXd'Xd''XqXq'Xq''] (pu):
[1.305, 0.296, 0.252, 0.474, 0.243, 0.18]

d axis time constants: Short-circuit

q axis time constant(s): Open-circuit

Time constants [Td' Td'' Tqo''] (s):
[1.01, 0.053, 0.1]

Stator resistance Rs (p.u.):
2.8544e-3

Coeff. of inertia, friction factor and pole pairs [H(s) F(pu) p()]:
[inf 0 20]

Init. cond. [dw(%) th(deg) ia,ib,ic(pu) pha,phb,phc(deg) Vt(pu)]:
[0 0 0 0 0 0 0 0 1]

☒ Simulate saturation

Saturation parameters [ifd1,ifd2,... (p.u.) ; vt1,vt2,... (p.u.)]:
[0.6404,0.7127,0.8441,0.9214,0.9956,1.082,1.19,1.316,1.457,0]

OK Cancel Help Apply

Synchronous Machine

Rotor type, Nominal

The same parameters as the fundamental p.u. dialog box.

Reactances

The d-axis synchronous reactance X_d , transient reactance X_d' , and subtransient reactance X_d'' , the q axis synchronous reactance X_q , transient reactance X_q' (only if round rotor), and subtransient reactance X_q'' , and finally the leakage reactance X_l (all in p.u.)

d-axis time constants, q-axis time constant(s)

Specify which time constants you will supply for each axis: either open-circuit or short-circuit.

Time constants

The d-axis and q-axis time constants (all in s). These values must be consistent with choices made on the two previous lines: d axis transient open-circuit (T_{do}') or short-circuit (T_d') time constant, d axis subtransient open-circuit (T_{do}'') or short-circuit (T_d'') time constant, q axis transient open-circuit (T_{qo}') or short-circuit (T_q') time constant (only if round rotor), q axis subtransient open-circuit (T_{qo}'') or short-circuit (T_q'') time constant.

Stator resistance

The stator resistance R_s (p.u.).

Mechanical, Initial conditions, Simulate saturation, Saturation

The same parameters as the fundamental parameters in p.u. dialog box.

Note These three blocks simulate exactly the same Synchronous machine model, the only difference is the way of entering the parameter units.

Inputs and Outputs

The units of inputs and outputs will vary according to which dialog box was used to enter the block parameters. For the non-electrical connections, there are two possibilities. If the first dialog box (fundamental parameters in SI units) is used, the inputs and outputs are in SI units (except for dw in the vector of internal variables, which is always in p.u., and angle θ which is always in rad). If the second or third dialog boxes are used, the inputs and outputs are in p.u.

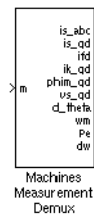
The first input is the mechanical power at the machine's shaft. In the generating mode, this input can be a positive constant or function or the output of a prime mover block (see the Hydraulic Turbine and Governor or Steam Turbine and Governor blocks). In the motoring mode, this input is usually a negative constant or function.

The second input of the block is the field voltage and can be supplied by a voltage regulator (see the Excitation System block) in the generating mode and is usually a constant in the motoring mode.

The first three outputs are the electrical terminals of the stator. The last output of the block is a vector containing 16 variables. They are, in order:

- 1-3: Stator currents (flowing out of machine) i_{sa} , i_{sb} and i_{sc}
- 4-5: q and daxis stator currents (flowing out of machine) i_q , i_d
- 6-8: Field and damper winding currents (flowing into machine) i_{fd} , i_{kq} and i_{kd}
- 9-10: q and d axis magnetizing fluxes ϕ_{mq} , ϕ_{md}
- 11-12: q and d axis stator voltages v_q , v_d
- 13: Rotor angle deviation $\Delta\theta$, also known as power angle δ
- 14: Rotor speed ω_r
- 15: Electrical power P_e
- 16: Rotor speed deviation $\Delta\omega$

These variables can be demultiplexed by using the special Machines Measurement Demux block provided in the Machines library.

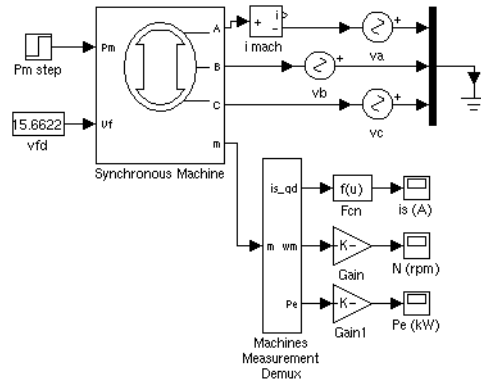


Example

This example, available in the psbsyncmachine.mdl file, illustrates the use of the Synchronous Machine block in motoring mode. The simulated system consists of an industrial grade synchronous motor (150 HP, 440V) connected to an infinite bus. After the machine reaches a stable speed, the load (mechanical

Synchronous Machine

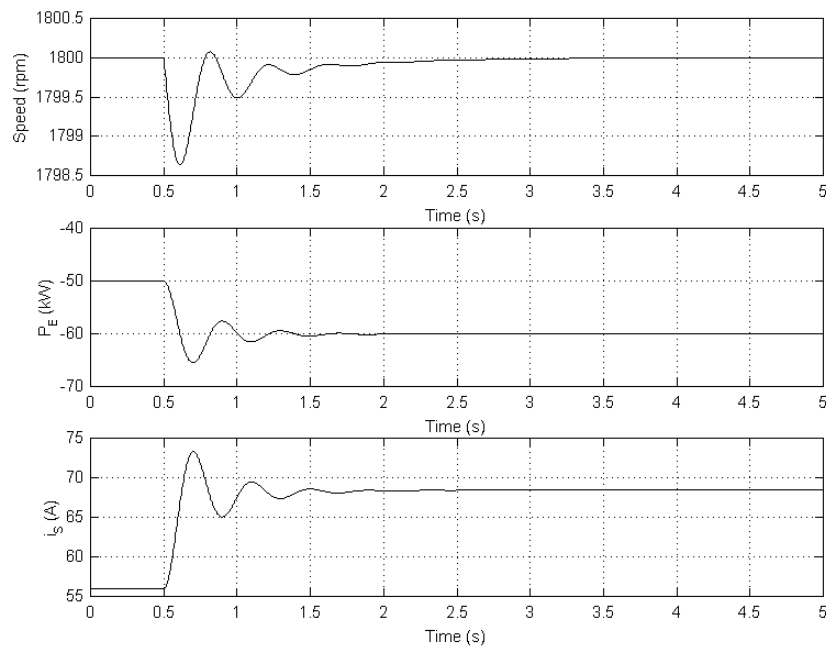
power) is changed from 50 kW to 60 kW. The initial conditions are set in such a way that the simulation starts in steady-state. Open the Simulink diagram by typing `psbsyncmachine`.



Set the simulation parameters as follows:

- Integrator type: `Stiff, ode15s`
- Stop time: `5 s`
- Integration options: Use default settings, except for Max. step size, which you must set to `0.005 s`

Run the simulation and observe the speed, power and current of the motor.



Since this is a four pole machine, the nominal speed is 1800 rpm. The initial speed is 1800 rpm as prescribed (top graph). The load passes from 50 kW to 60 kW at $t=0.5$ s. The machine then oscillates before stabilizing to 1800 rpm.

Now, look at the electrical power (middle graph). Since we are in motoring mode, the machine absorbs power and P_e is negative. As expected, the power starts at -50 kW until the load is changed at $t=0.5$ seconds, at which point the power oscillates before settling at -60 kW.

Finally, look at the stator current i_s . As expected, the current starts with the value corresponding to a three-phase power of 50 kW (56 A), before oscillating and settling to the value corresponding to a 60 kW load (68.5 A).

References

[1] Krause P.C., *Analysis of Electric Machinery*, McGraw-Hill, 1986, section 12.5.

Synchronous Machine

[2] Kamwa I. et al., “Experience with Computer-Aided Graphical Analysis of Sudden-Short-Circuit Oscillograms of Large Synchronous Machines”, *IEEE Transactions on Energy Conversion*, Vol.10, No.3, September 1995.

See Also

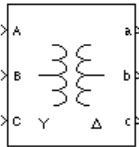
Excitation System, Hydraulic Turbine and Governor, Powergui,
Simplified Synchronous Machine, Steam Turbine and Governor

Three-Phase Transformer (Two Windings)

Purpose Implement a three-phase transformer with configurable winding connections.

Library Elements

Description This block implements a three-phase transformer using three single-phase transformers. The saturable core can be simulated or not, simply by setting the appropriate checkbox in the parameter menu of the block. See the Linear Transformer block and Saturable Transformer block sections for a detailed description of the electrical model of single-phase transformers.



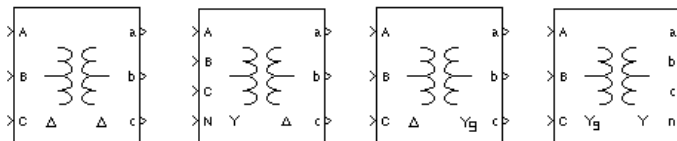
The two windings of the transformer can be connected in the following manner.

- Wye
- Wye with accessible neutral
- Grounded Wye
- Delta (D11), delta leading wye by 30 degrees
- Delta (D1), delta lagging wye by 30 degrees

Note The D11 and D1 convention assumes that the wye voltage phase angle is at noon (12), on a clock display. D1 and D11 refer respectively to 11 AM (-30 degrees) and 1 PM (+30 degrees)

The block takes into account the connection type you have selected and the icon of the block is automatically updated. An input port labeled N is added to the block if you select the wye connection with accessible neutral for the winding 1. If you ask for an accessible neutral on winding 2 an extra output port labeled n is generated.

The following icons are displayed for four arbitrary settings

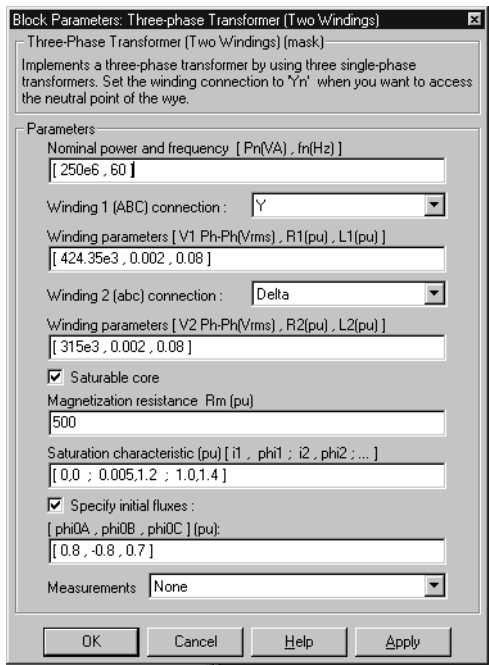


Three-Phase Transformer (Two Windings)

The saturation characteristic, when activated, is the same as the one described for the saturable transformer block and the icon of the block is automatically updated. If the fluxes are not specified, the initial values will be automatically adjusted so that the simulation starts in steady state.

The leakage inductance and resistance of each winding is given in p.u. based on the transformer nominal power P_n and on the nominal voltage of the winding (V_1 or V_2). Refer to the per unit explanations given in the Linear Transformer and Saturable Transformer blocks reference sections.

Dialog Box and Parameters



Nominal power and frequency

The nominal power rating, in volt amperes (VA), and nominal frequency, in hertz (Hz), of the transformer.

Winding 1 (ABC) connection

The phase connection for the winding 1.

Three-Phase Transformer (Two Windings)

Winding parameters

The phase-to-phase nominal voltage in volts rms, resistance, and leakage inductance in p.u. for the winding 1.

Winding 2 (abc) connection

The phase connection for the winding 2.

Winding parameters

The phase-to-phase nominal voltage in volts rms, resistance, and leakage inductance in p.u. for the winding 2.

Saturable core

If checked, implements a saturable three-phase transformer.

Magnetization resistance Rm

The magnetization resistance Rm, in p.u.

Magnetization reactance Lm

The magnetization inductance Lm, in p.u., for a non-saturable core. The **Magnetization reactance Lm** parameter is not visible in the dialog box if the **Saturable core** parameter is checked.

Saturation characteristic

The saturation characteristic for the saturable core. Specify a serie of current/ flux pairs (in p.u.) starting with the pair (0,0). This parameter is visible only if the **Saturable core** parameter is checked.

Specify initial fluxes

if checked, the initial fluxes are defined by the [phi 0A, phi 0B, phi 0C] parameter

[phi0A, phi0B, phi0C]

Specifies initial fluxes for each phase of the transformer. This parameter is visible only if the **Specify initial fluxes** and **Saturable core** parameters are checked.

Measurements

Select **Winding voltages** to measure the voltage across the winding terminals of the Three-Phase Transformer block.

Three-Phase Transformer (Two Windings)

Select **Winding currents** to measure the current flowing through the windings of the Three-Phase Transformer block.

Select **Fluxes and magnetization currents** to measure the flux linkage, in volt seconds (V.s), and the magnetization current (for saturable transformers only).

Select **All measurement (V, I, Flux)** to measure the winding voltages, currents, magnetization currents, and the fluxes.

Place a Multimeter block in your model to display the selected measurements during the simulation. In the **Available Measurement** listbox of the Multimeter block, the measurements will be identified by a label followed by the block name.

If the **Winding 1 ABC connection** parameter is set to **Y**, **Yn**, or **Yg**, the labels will be as follows.

Measurement	Label
Winding voltages	Uan_w1: , Ubn_w1: , Ucn_w1: or : Uag_w1: , Ubg_w1: , Ucg_w1:
Winding currents	Ian_w1: , Ibn_w1: , Icn_w1: or : Iag_w1: , Ibg_w1: , Icg_w1:
Fluxes	Flux A: , Flux B: , Flux C:
Magnetization currents	Imag A: , Imag B: , Imag C:

If the **Winding 1 ABC connection** parameter is set to **Delta (D11)** or **Delta (D1)**, the labels will be as follows.

Measurement	Label
Winding voltages	Uab_w1: , Ubc_w1: , Uca_w1:
Winding currents	Iab_w1: , Ibc_w1: , Ica_w1:

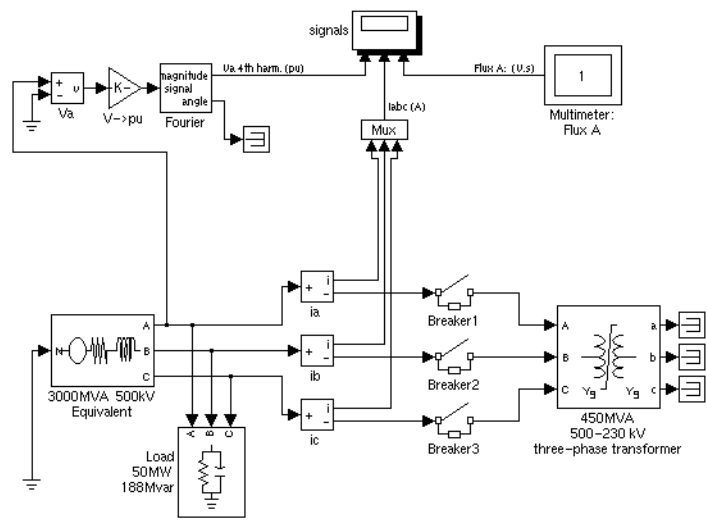
Three-Phase Transformer (Two Windings)

Measurement	Label
Fluxes	Flux A: , Flux B: , Flux C:
Magnetization currents	Imag A: , Imag B: , Imag C:

The same labels apply for the **Winding 2 (abc) connection** parameter, except that 1 is replaced by 2 in the labels.

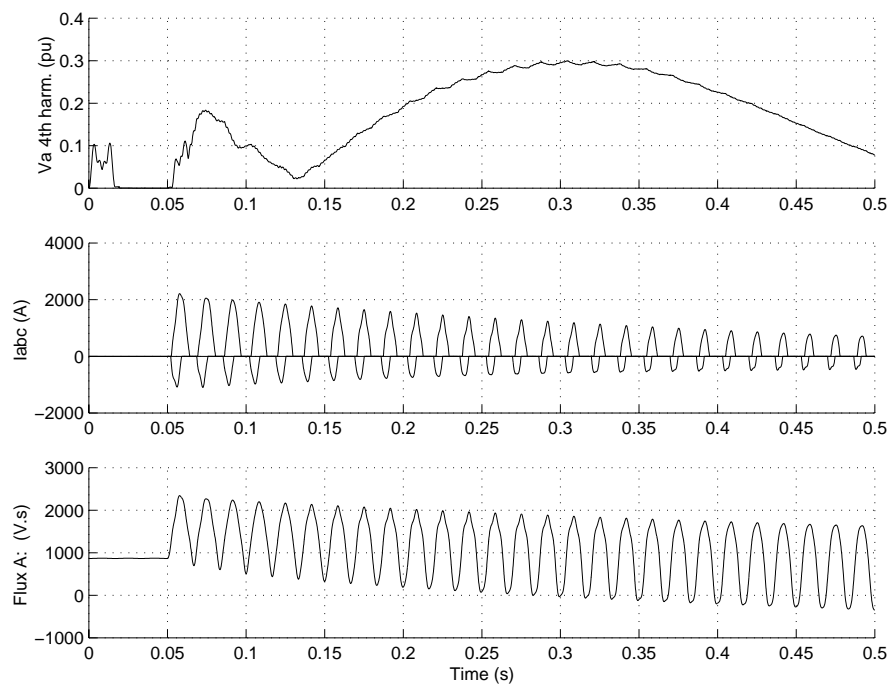
Example

The psbtransfo3ph.mdl circuit is using the Three-Phase Transformer block where the saturable core is simulated. Both windings are connected in a wye grounded configuration. Note that the neutral points of the two windings are internally connected to the ground.



Run the simulation and observe the simulation results.

Three-Phase Transformer (Two Windings)



See Also [Three-Phase Transformer \(Three Windings\)](#)

Three-Phase Transformer (Three Windings)

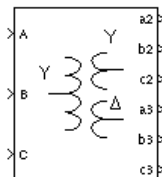
Purpose

Implement a three-phase transformer with configurable phase connection.

Library

Elements

Description



This block implements a three-phase transformer by using three single-phase transformers with three windings. The saturable core can be simulated or not, simply by setting the appropriate checkbox in the parameter menu of the block. See the Linear transformer and Saturable transformer sections for a detailed description of the electrical model of single-phase transformers.

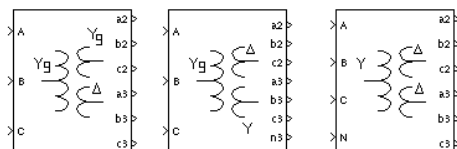
The three windings of the transformer can be connected in the following manner.

- Wye
- Wye with accessible neutral (for the windings 1 and 3 only)
- Grounded Wye
- Delta (D11), delta lagging wye by 30 degrees
- Delta (D1), delta leading wye by 30 degrees

Note The D11 and D1 convention assumes that the wye voltage phase angle is at noon (12), on a clock display. D1 and D11 refer respectively to 11 AM (-30 degrees) and 1 PM (+30 degrees)

The block take into count the connection type you have selected and the icon of the block is automatically updated. An input port labeled N is added to the block if you select the wye connection with accessible neutral for the winding 1. If you ask for an accessible neutral on winding 3, an extra output port labeled n3 is generated.

The following icons are displayed for three arbitrary settings



Three-Phase Transformer (Three Windings)

The saturation characteristic, when activated, is the same as the one described for the saturable transformer block and the icon of the block is automatically updated. If the fluxes are not specified, the initial values will be automatically adjusted so that the simulation starts in steady state.

The leakage inductances and resistance of each winding is given in p.u. based on the transformer nominal power P_n and on the nominal voltage of the winding (V_1 , V_2 or V_3). Refer to the per unit explanations given in the Linear Transformer and Saturable Transformer blocks reference sections.

Dialog Box and Parameters

Block Parameters: Three-Phase Transformer (Three Windings)

Three-Phase Transformer (Three Windings) (mask)

Implements a three-phase transformer by using three single-phase transformers. Set the winding connection to 'Yn' when you want to access the neutral point of the wye (for winding 1 and 3 only).

Parameters

Port configuration

ABC as input terminals

Nominal power and frequency [Pn(VA) , fn(Hz)]

[250e6 , 60]

Winding 1 (ABC) connection:

Y

Winding parameters [V1 Ph-Ph(Vrms) , R1(pu) , L1(pu)]

[735e3 , 0.002 , 0.08]

Winding 2 (abc-2) connection:

Y

Winding parameters [V2 Ph-Ph(Vrms) , R2(pu) , L2(pu)]

[315e3 , 0.002 , 0.08]

Winding 3 (abc-3) connection:

Delta

Winding parameters [V3 Ph-Ph(Vrms) , R3(pu) , L3(pu)]

[315e3 , 0.002 , 0.08]

☒ Saturable core

Magnetization reactance Lm(pu)

500

Saturation characteristic (pu) [i1 phi1 ; i2 phi2 ; ...]

[0.0 ; 0.005,1.2 ; 1.0,1.4]

☒ Specify initial fluxes:

[phi0A phi0B phi0C] (pu)

[0.8 -0.8 0.7]

Measurements

None

OK

Cancel

Help

Apply

Port configuration

Specify the ABC port of winding one as input terminals or as output terminals. The terminals of the winding two and three will then be reconfigured consequently.

Three-Phase Transformer (Three Windings)

Nominal power and frequency

The nominal power rating, in volt amperes (VA), and nominal frequency, in hertz (Hz), of the transformer.

Winding 1 (ABC) connection

The phase connection for the winding 1.

Winding parameters

The phase-to-phase nominal voltage in volts rms, resistance, and leakage inductance in p.u. for the winding 1.

Winding 2 (abc2) connection

The phase connection for the winding 2.

Winding parameters

The phase-to-phase nominal voltage in volts rms, resistance, and leakage inductance in p.u. for the winding 2.

Winding 3 (abc3) connection

The phase connection for the winding 3.

Winding parameters

The phase-to-phase nominal voltage in volts rms, resistance, and leakage inductance in p.u. for the winding 3.

Saturable core

if checked, implements a saturable three-phase transformer.

Magnetization resistance R_m

The magnetization resistance R_m , in p.u.

Magnetization reactance L_m

The magnetization inductance L_m , in p.u., for a nonsaturable core. The **Magnetization reactance L_m** parameter is not visible in the dialog box if the **Saturable core** parameter is checked.

Saturation characteristic

The saturation characteristic for the saturable core. Specify a series of current/ flux pairs (in p.u.) starting with the pair (0,0). This parameter is visible only if the **Saturable core** parameter is checked.

Three-Phase Transformer (Three Windings)

Specify initial fluxes

if checked, the initial fluxes are defined by the [phi 0A, phi 0B, phi 0C] parameter

[phi0A, phi0B, phi0C] (pu):

Specifies initial fluxes for each phase of the transformer. This parameter is visible only if the **Specify initial fluxes** and **Saturable core** parameters are checked.

Measurements

Select **Winding voltages** to measure the voltage across the winding terminals of the Three-Phase Transformer block.

Select **Winding currents** to measure the current flowing through the windings of the Three-Phase Transformer block.

Select **Fluxes and magnetization currents** to measure the winding voltages, currents, magnetization currents, and the fluxes.

Select **All measurements (V, I, Flux)** to measure the winding voltages, winding currents, magnetization currents, and fluxes.

Place a Multimeter block in your model to display the selected measurements during the simulation. In the **Available Measurement** listbox of the Multimeter block, the measurements will be identified by a label followed by the block name.

If the **Winding 1 ABC connection** parameter is set to **Y**, **Yn**, or **Yg**, the labels will be as follows.

Measurement	Label
Winding voltages	Uan_w1: , Ubn_w1: , Ucn_w1: or : Uag_w1: , Ubg_w1: , Ucg_w1:
Winding currents	Ian_w1: , Ibn_w1: , Icn_w1: or : Iag_w1: , Ibg_w1: , Icg_w1:

Three-Phase Transformer (Three Windings)

Measurement	Label
Fluxes	Flux A: , Flux B: , Flux C:
Magnetization currents	Imag A: , Imag B: , Imag C:

If the **Winding 1 ABC connection** parameter is set to **Delta (D11)** or **Delta (D1)**, the labels will be as follows.

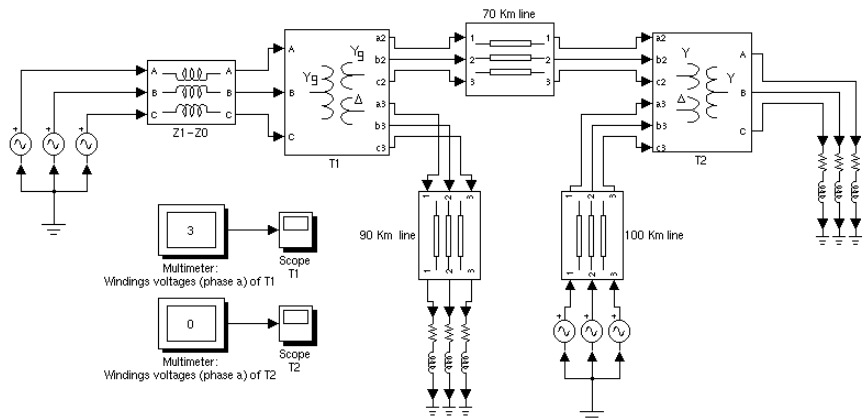
Measurement	Label
Winding voltages	Uab_w1: , Ubc_w1: , Uca_w1:
Winding currents	Iab_w1: , Ibc_w1: , Ica_w1:
Fluxes	Flux A: , Flux B: , Flux C:
Magnetization currents	Imag A: , Imag B: , Imag C:

The same labels apply for the **Winding 2 (abc2) connection** and **Winding 3 (abc3) connection** parameters, except that the 1 is replaced by 2 or by 3 in the labels.

Example

The psbtransfo3wdn.mdl circuit is using two Three-Phase Transformer blocks. In this example the ABC terminals of winding one of T1 transformer are configured as inputs and the ABC terminals of winding one of T2 transformer are set configured as outputs.

Three-Phase Transformer (Three Windings)

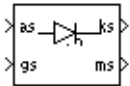
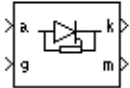


See Also Three-Phase Transformer (Two Windings)

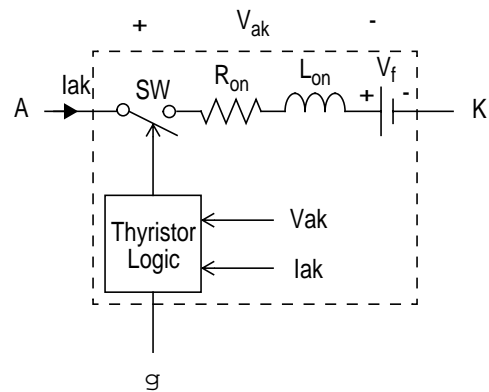
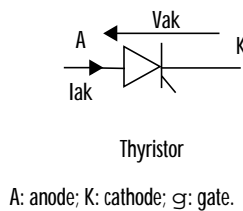
Purpose Implement a thyristor model.

Library Elements

Description

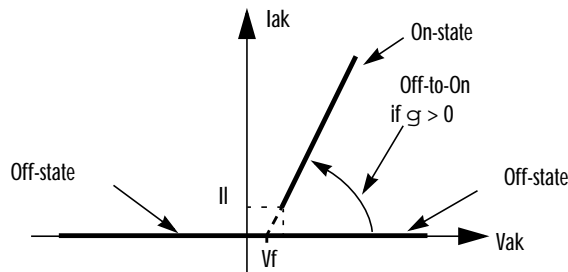


A Thyristor is a semiconductor device that can be turned on via a gate signal. The thyristor model is simulated as a resistor (R_{on}), an inductor (L_{on}), and a DC voltage source (V_f), connected in series with a switch. The switch is controlled by a logical signal depending on the voltage V_{ak} , the current I_{ak} and the gate signal (g).



The Thyristor block also contains a series R_s - C_s snubber circuit that can be connected in parallel with the thyristor device.

The static VI characteristic of this model is shown in figure below.



Thyristor

The thyristor device turns on when the anode-cathode voltage is greater than V_f and a positive pulse signal is applied at the gate input ($g > 0$). The pulse height must be greater than zero and last long enough to allow the thyristor anode current to become larger than the latching current I_l .

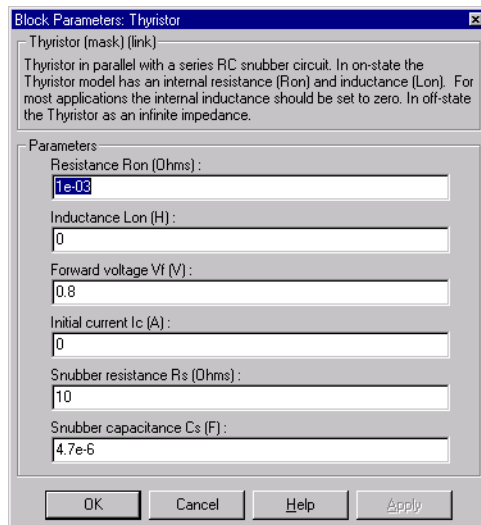
The thyristor device turns off when the current flowing in the device becomes zero ($I_{ak}=0$) and a negative voltage appears across the anode and cathode for at least a period of time equal to the turn-off time T_q . If the voltage across the device becomes positive within a period of time less than T_q , the device will turn on automatically even if the gate signal is low ($g = 0$) and the anode current is less than the latching current. Furthermore, if during turn on, the device current amplitude stays below the latching current level specified in the dialog box, the device turns off after the gate signal level becomes low ($g = 0$).

The turn-off time T_q represents the carrier recovery time: it is the time interval between the instant the anode current has decreased to zero and the instant when the thyristor is capable of withstanding positive voltage V_{ak} without turning on again.

Dialog Boxes and Parameters

Thyristor Model and Detailed Thyristor Model

In order to optimize simulation speed, two models of thyristors are available: the thyristor model and the detailed thyristor model. For the thyristor model, the latching current I_l and recovery time T_q are assumed to be zero.



Resistance R_{on}

The thyristor internal resistance R_{on} , in ohms (Ω). The **Resistance R_{on}** parameter cannot be set to 0 when the **Inductance L_{on}** parameter is set to 0.

Inductance L_{on}

The thyristor internal inductance L_{on} , in henries (H). The **Inductance L_{on}** parameter cannot be set to 0 when the **Resistance R_{on}** parameter is set to 0.

Forward voltage V_f

The forward voltage of the thyristor, in volts (V).

Initial current I_c

When the Inductance L_{on} parameter is greater than zero, you can specify an initial current flowing in the thyristor. It is usually set to zero in order to start the simulation with the thyristor blocked.

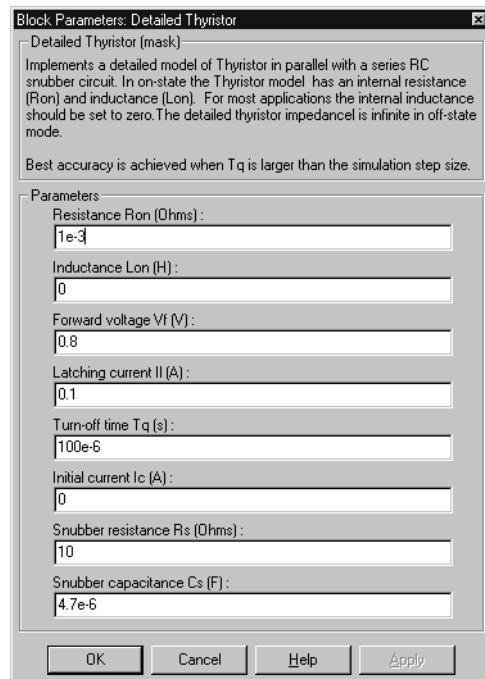
You can specify an initial current I_c value corresponding to a particular state of the circuit. In such a case all states of the linear circuit must be set accordingly. Initializing all states of a power-electronic converters is a complex task. Therefore, this option is useful only with simple circuits.

Snubber resistance R_s

The snubber resistance, in ohms (Ω). Set the **Snubber resistance R_s** parameter to `inf` to eliminate the snubber from the model.

Snubber capacitance C_s

The snubber capacitance in farads (F). Set the **Snubber capacitance C_s** parameter to 0 to eliminate the snubber, or to `inf` to get a purely resistive snubber.



Block Parameters: Detailed Thyristor

Detailed Thyristor (mask)

Implements a detailed model of Thyristor in parallel with a series RC snubber circuit. In on-state the Thyristor model has an internal resistance (R_{on}) and inductance (L_{on}). For most applications the internal inductance should be set to zero. The detailed thyristor impedance is infinite in off-state mode.

Best accuracy is achieved when T_q is larger than the simulation step size.

Parameters

Resistance R_{on} (Ohms):	<input type="text" value="1e-3"/>
Inductance L_{on} (H):	<input type="text" value="0"/>
Forward voltage V_f (V):	<input type="text" value="0.8"/>
Latching current I_l (A):	<input type="text" value="0.1"/>
Turn-off time T_q (s):	<input type="text" value="100e-6"/>
Initial current I_c (A):	<input type="text" value="0"/>
Snubber resistance R_s (Ohms):	<input type="text" value="10"/>
Snubber capacitance C_s (F):	<input type="text" value="4.7e-6"/>

OK Cancel Help Apply

Latching current I_l

The latching current of the detailed thyristor model, in amperes (A).

Turn-off time T_q

The turn-off time T_q of the detailed thyristor model, in amperes (A).

Inputs and Outputs

The Thyristor block consists of two inputs and two outputs. The first input and output are the thyristor terminals connected respectively to anode (a) and cathode (k). The second input (g) is a Simulink logical signal applied to the gate. The second output (m) is a Simulink measurement output vector [Iak, Vak] returning the thyristor current and voltage.

Assumptions and Limitations

The Thyristor block implements a macro-model of the real thyristor. It does not take into account either the geometry of the device or complex physical processes that model the behavior of the device [1-2]. The forward breakover voltage and the critical value of the derivative of the reapplied anode-cathode voltage are not considered by the model.

Depending on the value of inductance L_{on} , the Thyristor block is modeled either as a current source ($L_{on} > 0$) or as a variable topology circuit ($L_{on} = 0$). See Advanced Topics chapter for more details.

As the Thyristor block is modeled as a current source, it cannot be connected in series with an inductor, a current source or an open circuit, unless a snubber circuit is used.

You must use a stiff integrator algorithm to simulate circuits containing thyristors. `Ode23tb` or `Ode15s` with default parameters usually give best simulation speed.

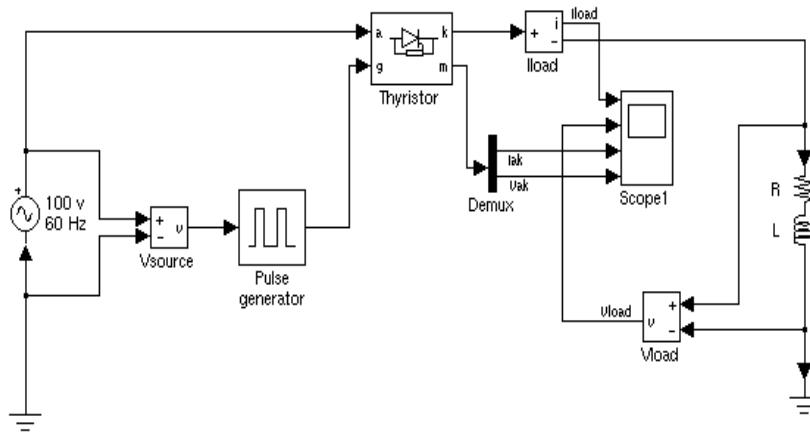
The inductance L_{on} will be forced to zero if you choose to discretize your circuit.

Example

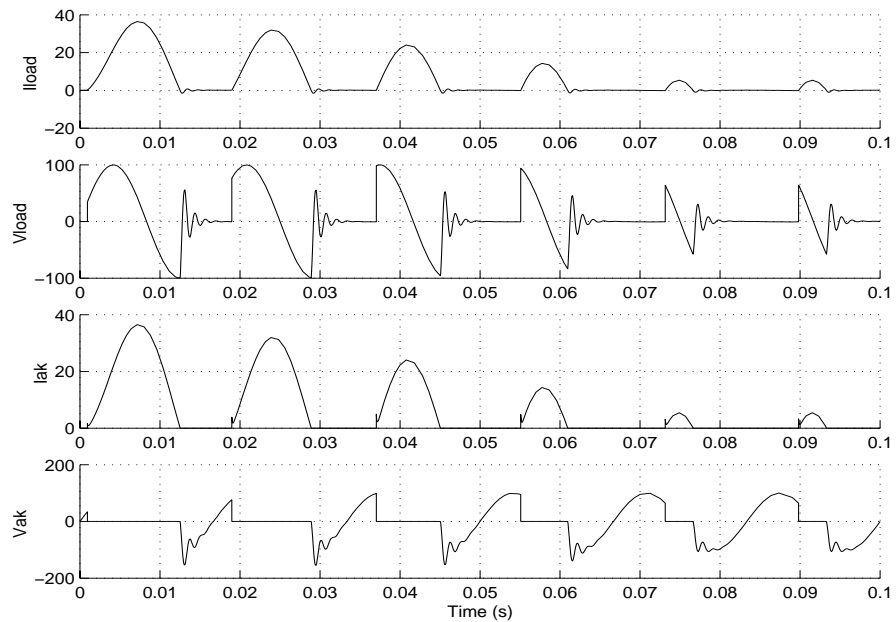
A single pulse thyristor rectifier is used to feed a RL load. The gate pulses are obtained from a pulse generator synchronized on the source voltage. The circuit is available in the `psbthyristor.mdl` file. The following parameters are used:

$R=1\Omega$; $L=10mH$; Thyristor block: $R_{on}=0.001\ \Omega$, $L_{on}=0\ H$, $V_f=0.8\ V$, $R_s=20\ \Omega$, $C_s=4e-6\ F$.

Thyristor



The firing angle is varied by a pulse generator synchronized on the voltage source. Run the simulation and observe the load current, load voltage, as well as the thyristor current and voltage.



References

[1] Rajagopalan, V., *Computer-Aided Analysis of Power Electronic Systems*, Marcel Dekker, Inc., New York, 1987.

[2] Mohan, N., *Power Electronic, Converters, Applications and Design*, John Wiley & Sons, Inc., New York, 1995.

See Also

Diode, Universal Bridge

Universal Bridge

Purpose

Implement a universal three-phase bridge converter with selectable configuration and power switch type.

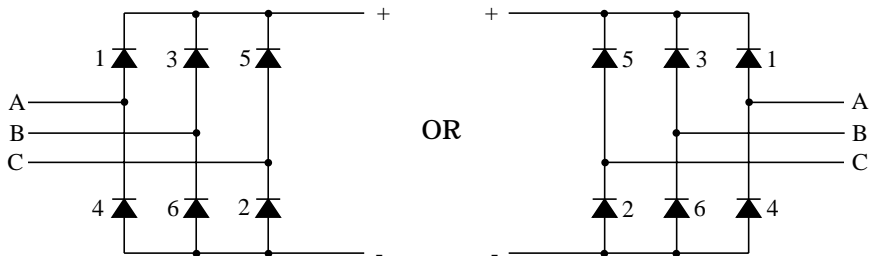
Library

Power Electronics

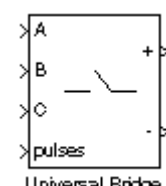
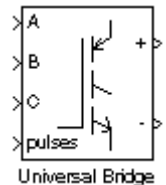
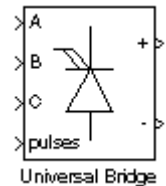
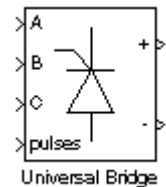
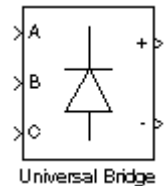
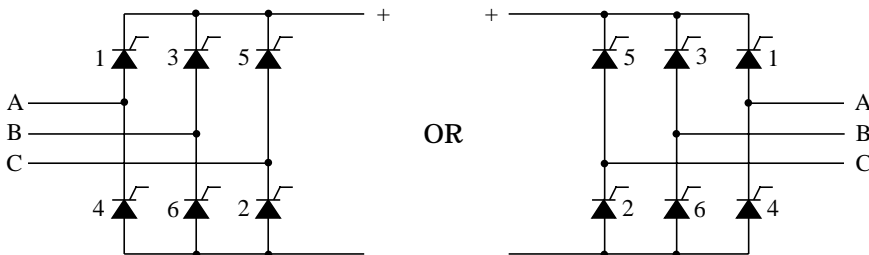
Description

The Universal Bridge block implements a universal three-phase power converter that consists of six power switches connected as a bridge. The type of power switch and converter configuration are selectable from the dialog box.

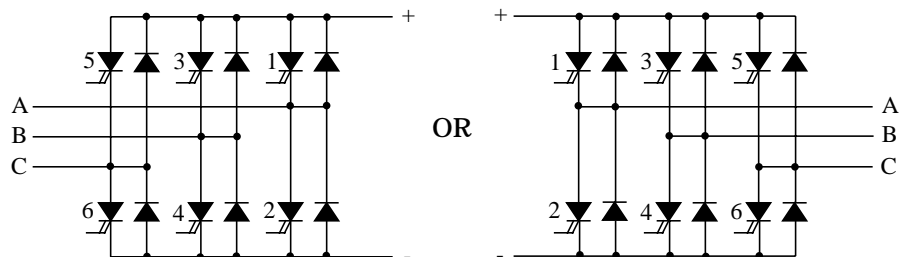
Diode bridges:



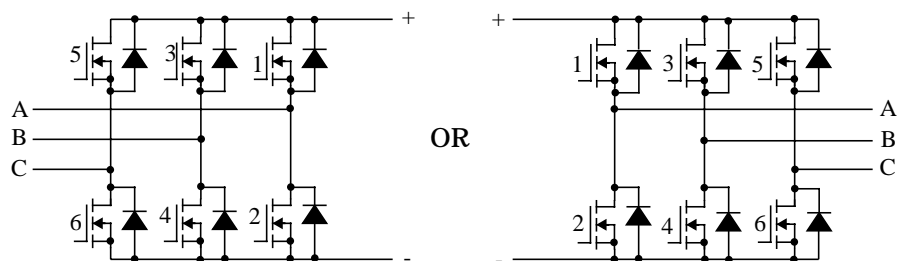
Thyristor bridges:



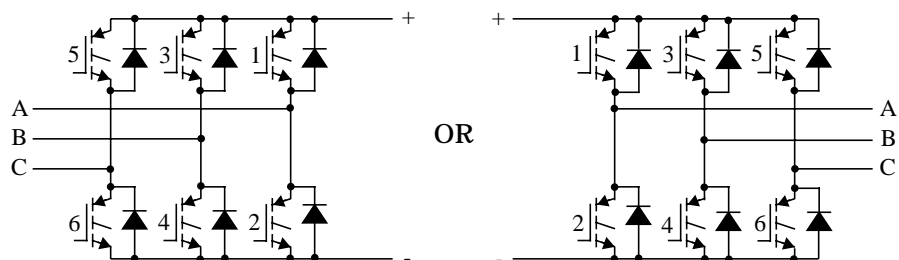
GTO-Diode bridges:



MOSFET-Diode bridges:

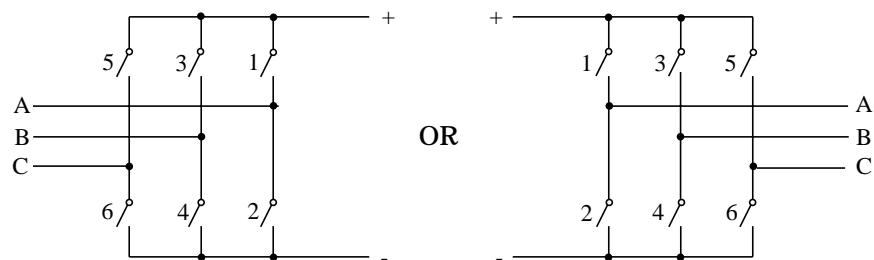


IGBT-Diode bridges:



Universal Bridge

Ideal switch bridge:



Dialog Box and Parameters

Block Parameters: Universal Bridge

Universal Bridge (mask) (link)

This block implement a six pulses bridge of selected power electronics devices. Series RC snubber circuits are connected in parallel with each switch device. For most applications, except for the MOSFET device, the internal inductance should be set to zero.

Parameters

Port configuration: ABC as input terminals

Snubber resistance R_s (Ohms): 10

Snubber capacitance C_s (F): .01e-6

Power Electronic device: Thyristors

R_{on} (Ohms): 1e-3

L_{on} (H): 0

V_f (V): .8

Measurements: None

OK Cancel Help Apply

Port configuration

Set to **ABC as input terminals** to connect the A,B, and C phases of the bridge to the input ports one, two, and three of the Universal Bridge block. The DC + and - terminals will be connected at outputs one and two.

Set to **ABC as output terminals** to connect the A,B, and C phases of the bridge to the output ports one, two, and three of the Universal Bridge block. The DC + and - terminals will be connected at outputs one and two.

Snubber resistance Rs

The snubber resistance, in ohms (Ω). Set the **Snubber resistance Rs** parameter to `inf` to eliminate the snubbers from the model.

Snubber capacitance Cs

The snubber capacitance, in farads (F). Set the **Snubber capacitance Cs** parameter to 0 to eliminate the snubbers, or to `inf` to get a purely resistive snubber.

Power electronic device

Select the type of power electronic device to use in the bridge.

Ron (Ohms)

Internal resistance of the selected device, in ohms (Ω).

Lon (H)

Internal inductance, in henries (H), for the Diode, the Thyristor, or the MOSFET device.

[Tf (s), Tt (s)]

Fall time Tf and tail time Tt, in seconds (s), for the GTO or the IGBT devices.

Measurements

Select **Device voltages** to measure the voltage across the six power electronic device terminals.

Select **Device currents** to measure the current flowing through the six power electronic devices. If the snubber devices are defined, the measured currents are the one flowing through the power electronic devices only.

Select **UAB UBC UCA UDC voltages** to measure the terminal voltages (AC and DC) of the Universal Bridge block.

Select **All voltages and currents** to measure all voltages and currents defined for the Universal Bridge block.

Universal Bridge

Place a Multimeter block in your model to display the selected measurements during the simulation. In the **Available Measurement** listbox of the Multimeter block, the measurement will be identified by a label followed by the block name.

Measurement	Label
Device voltages	uSw1: , uSw2: , uSw3: , uSw4: , uSw5: , uSw6:
Branch current	i Sw1: , i Sw2: , i Sw3: , i Sw4: , i Sw5: , i Sw6:
Terminal voltages	uAB: , uBC:, uCA: , uDC:

Inputs and Outputs

The bridge configuration is selectable so that the inputs and outputs depend on the configuration chosen:

- When A, B, C are selected as inputs, the dc terminals are the outputs.
- When A, B, C are selected as outputs, the dc terminals are the inputs.

Except for the case of diode bridge, the Pulses input accepts a Simulink-compatible vectorized gating signal containing six pulse trains. The gating signals are sent to the power switches according to the number shown in the diagrams above.

Note The pulses ordering in the vector of the gate signals corresponds to the switch number indicated in the six circuits shown in the Description section. For the diode and thyristor bridges the pulse ordering corresponds to the natural order of commutation. For all other forced commutated switches, pulses are sent to upper and lower switches of phases A, B, and C with the following order: [A upper, A lower, B upper, B lower, C upper, C lower]

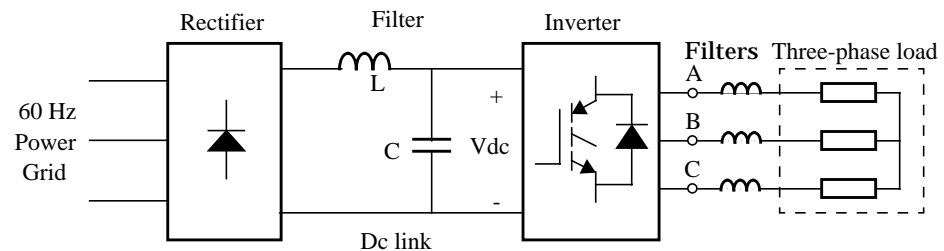
Assumptions and Limitations

Universal Bridge blocks can be discretized to be used in a discrete time step simulation specified by the Discrete System block. In this case, the internal commutation logic of the Universal Bridge takes care of the commutation between the power switches and the diodes in the converter legs.

Note In a converter built with individual forced-commutated power components (GTOs, MOSFETs, IGBTs), discretization of the model is not available. See Advanced Topics chapter for more details.

Example

This example illustrates the use of two Universal Bridge blocks in an ac-ac converter consisting of a rectifier feeding an IGBT inverter through a dc link. The inverter is pulse-width modulated (PWM) to produce a three-phase variable-voltage variable-frequency sinusoidal voltage to the load. In this example the inverter chopping frequency is 2000 Hz and the modulation frequency is 50 Hz.

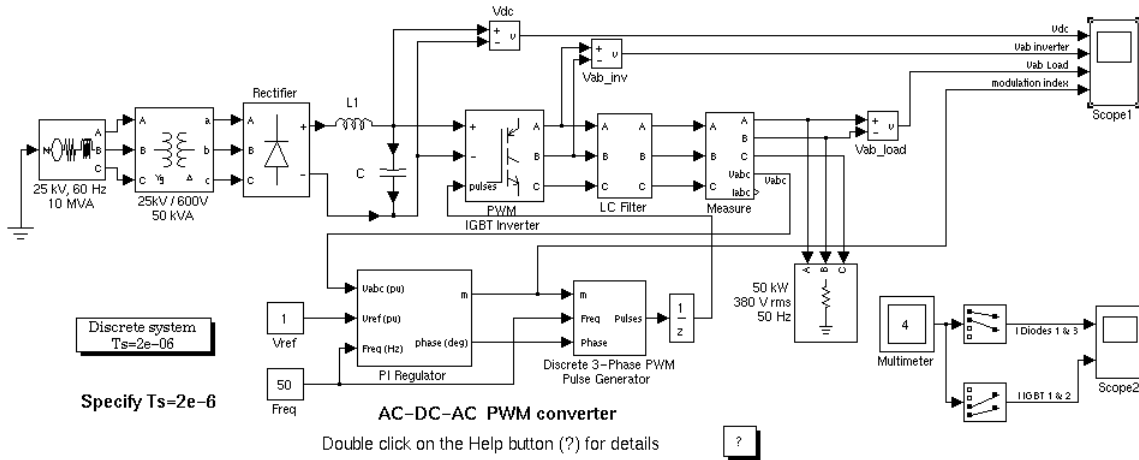


The IGBT inverter is controlled in closed loop with a PI regulator in order to maintain a 1 p.u. voltage (380 Vrms, 50 Hz) at the load terminals.

A multimeter block is used to observe commutation of currents between diodes 1 and 3 in the diode bridge and between IGBT/Diodes switches 1 and 2 in the IGBT bridge.

.The circuit is available in the psbbri dges. mdl demo.

Universal Bridge



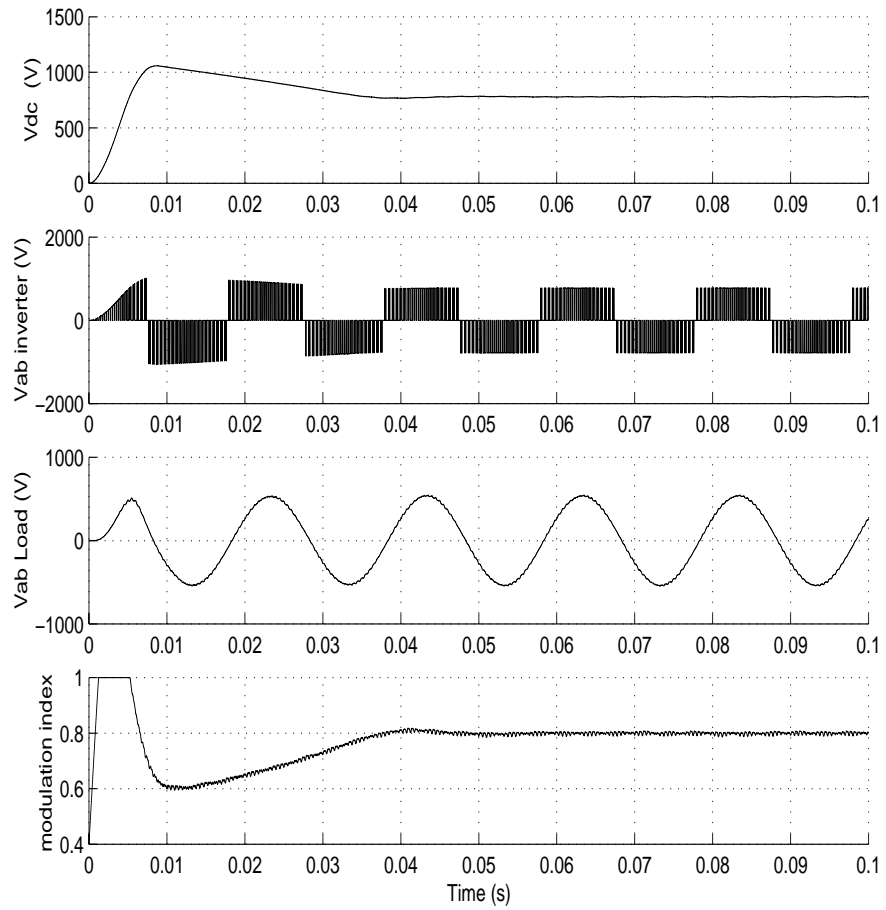
Start simulation. After a transient period of approximately 70 ms, the system reaches a steady state. Observe voltage waveforms at DC bus, inverter output and load on Scope1. The harmonics generated by the inverter around multiples of 2 kHz are filtered by the LC filter. As expected the peak value of the load voltage is 537 V (380 V rms).

In steady state the mean value of the modulation index is $m=0.77$ and the mean value of the DC voltage is 780 V. The fundamental component of 50 Hz voltage buried in the chopped inverter voltage is therefore:

$$V_{ab} = 780 \text{ V} * 0.612 * 0.80 = 382 \text{ V rms}$$

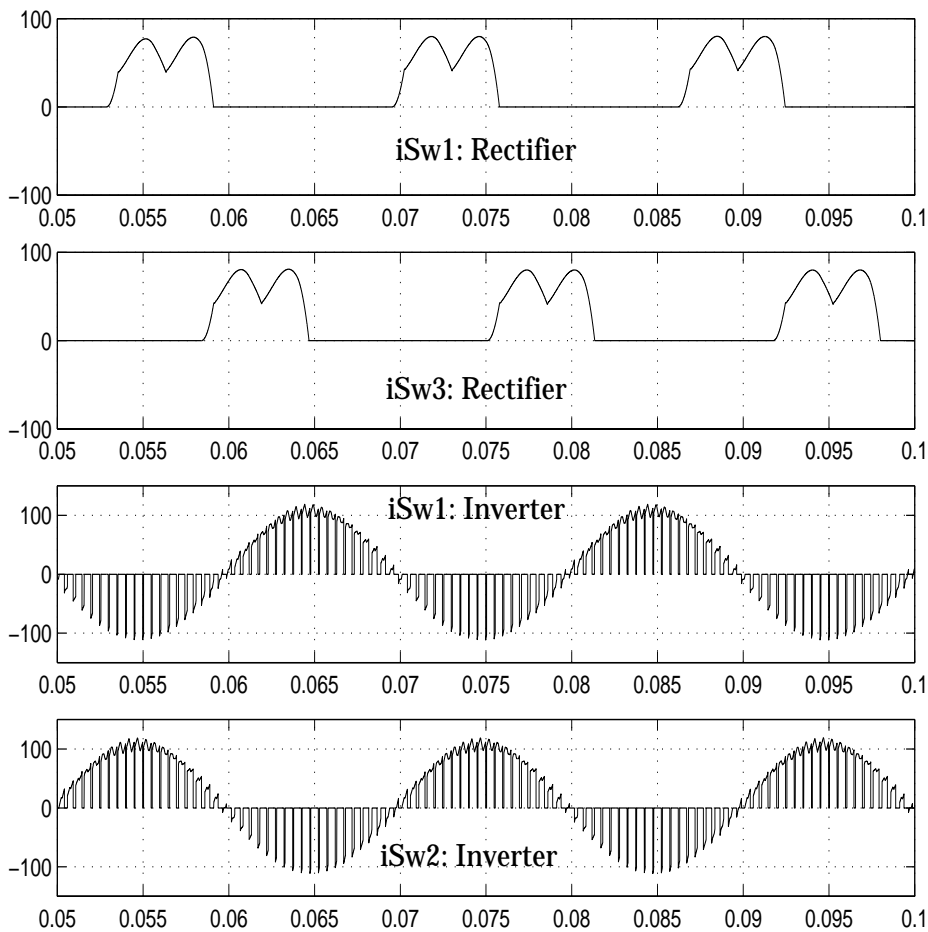
Observe diode currents on trace 1 of Scope2, showing commutation from diode 1 to diode 3. Also observe on trace 2 currents in switches 1 and 2 of the IGBT/ Diode bridge (upper and lower switches connected to phase A). These two currents are complementary. A positive current indicates a current flowing in the IGBT, whereas a negative current indicates a current flowing in the anti parallel diode.

Scope 1



Universal Bridge

Scope 2



See Also

Diode, GTO, Ideal Switch, IGBT, MOSFET, Thyristor

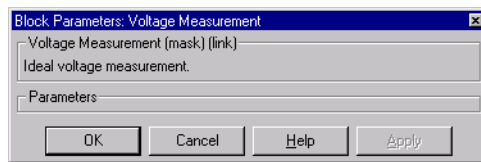
Purpose Measure a voltage in a circuit.

Library Measurements

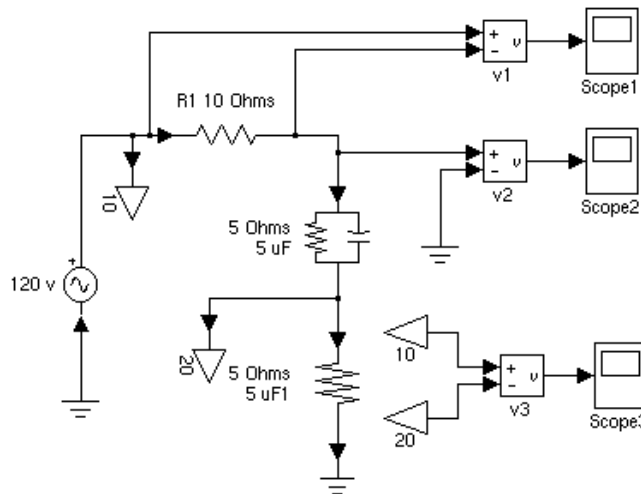
Description The Voltage Measurement block measures the instantaneous voltage between two electric nodes. The output provides a Simulink signal that can be used by other Simulink blocks.



Dialog Box



Example The following example uses three Voltage Measurement blocks to read voltages. This example is available in the `psbvoltmeasure.mdl` file.



See Also Current Measurement

Voltage Measurement

Power System Commands

This table indicates the tasks performed by the commands described in this chapter. The reference section of this chapter lists the commands in alphabetical order.

Task	Command
Compute the linear state-space model of an electrical circuit	<code>ci rc2ss</code>
Analyze an electric circuit built with the Power System Blockset	<code>power2sys</code>
Set the initial states values of an electrical circuit	<code>poweri ni t</code>

Purpose Compute the state-space model of a linear electrical circuit

Synopsis You must call `circ2ss` with a minimum of seven input arguments:

```
[A, B, C, D, states, x0, x0sw, r1sw, u, x, y, freq, Asw, Bsw, Csw, Dsw, Hlin] =
circ2ss(r1c, switches, source, line_dist, yout, y_type, unit)
```

You can also specify additional arguments. To use options, the number of input arguments must be 12, 13, 14 or 16:

```
[A, B, C, D, states, x0, x0sw, r1sw, u, x, y, freq, Asw, Bsw, Csw, Dsw, Hlin] =
circ2ss(r1c, switches, source, line_dist, yout, y_type, unit, net_arg1,
net_arg2, net_arg3, ...
netsim_flag, fid_outfile, freq_sys, ref_node, vary_name, vary_val)
```

Description Computes the state-space model of a linear electrical circuit expressed as

$$\begin{aligned}\dot{x} &= Ax + Bu \\ y &= Cx + Du\end{aligned}$$

where x is the vector of state-space variables (inductor currents and capacitor voltages), u is the vector of voltage and current inputs, and y is the vector of voltage and current outputs.

When you build a circuit from the Power System Blockset icons of the **powerlib** library, `circ2ss` is automatically called by `power2sys`. `circ2ss` has also been made available as a stand-alone function for expert users. This allows you to generate state-space models without using the Power System Blockset graphical user interface and to access options that are not available through **powerlib**. For example, you can specify transformers and mutual inductances with more than three windings.

The linear circuit may contain any combination of voltage and current sources, RLC branches, multiwinding transformers, mutually coupled inductances and switches. The state variables are inductor currents and capacitor voltages.

The state-space representation (matrices A,B,C,D and vector x0) computed by `circ2ss` can then be used in a Simulink system, via a State-Space block, to perform simulation of the electrical circuit (see Example section below).

Nonlinear elements (mechanical or electronic switches, transformer saturation, machines, distributed parameter lines, etc.) can be connected to the linear circuit.

These Simulink models are interfaced with the linear circuit through voltage outputs and current inputs of the state-space model. You can find the models of the nonlinear elements provided with the Power System Blockset in the **Powerlib_models** library (see the Advanced Topics chapter).

Input Arguments

The number of input arguments must be 7, 12, 13, 14 or 16. Arguments 8 to 16 are optional. The first seven arguments that must be specified are:

- `rlc`: Branch matrix specifying the network topology as well as the resistance R, inductance L, and capacitance C values. See format below.
- `switches`: Switch matrix. Specify an empty variable if no switches are used. See format below.
- `source`: Source matrix specifying the parameters of the electrical voltage and current sources. Specify an empty variable if no sources are used. See format below.
- `line_dist`: Distributed parameter line matrix. Specify an empty variable if no distributed lines are used. See format below.
- `yout`: String matrix of output expressions. See format below.
- `y_type`: Integer vector indicating output types (zero for voltage output, one for current output).
- `unit`: String specifying the units to be used for R, L, and C values in the `rlc` matrix. If `unit='OHM'`, R L C values are specified in ohms Ω at the fundamental frequency specified by `freq_sys` (default value is 60 Hz). If `unit='OMU'`, R L C values are specified in ohms (Ω), millihenries (mH), and microfarads (μF).

The last nine arguments are optional. The first three are used to pass arguments from the `power2sys` function. Hereafter, we describe only the arguments to be specified when `circ2ss` is used as a stand-alone function:

- `net_arg1`, `net_arg2`, `net_arg3`: Used to pass arguments from `power2sys`. Specify empty variables `[]` for each of these variables.

- **net sim_flag**: Integer controlling the messages displayed during the execution of **circ2ss**. Default value is zero.
If **net sim_flag** =0, the version number, number of states, inputs, outputs and modes are displayed. Output values are displayed in polar form for each source frequency.
If **net sim_flag**=1, only version number, number of states, inputs, and outputs are displayed.
If **net sim_flag**=2, no message is displayed during execution.
- **fid_outfile**: File identifier of the **circ2ss** output file containing parameter values, node numbers, steady-state outputs, and special messages. Default value is zero.
- **freq_sys**: Fundamental frequency (Hz) considered for specification of X_L and X_C reactances if **unit** is set to ' OHM' . Default value is 60 Hz.
- **ref_node**: Reference node number used for ground of pi transmission lines. If -1 is specified, the user will be prompted to specify a node number.
- **vary_name**: String matrix containing the symbolic variable names used in output expressions. These variables must be defined in your MATLAB workspace.
- **vary_val**: Vector containing the values of the variable names specified in **vary_name**.

Output Arguments

- **A, B, C, D**: State-space matrices of the linear circuit with all switches open. **A** (nstates, nstates) **B** (nstates, ni nput) **C** (nout put, nstates) **D** (nout put, ni nput) where nstates is the number of state variables, ni nput is the number of inputs, and nout put is the number of outputs.
- **states**: String matrix containing the names of the state variables. Each string has the following format:

Inductor currents: **I1_bxx_nzz1_zz2**

Capacitor voltages: **Uc_bxx_nzz1_zz2**

where:

xx = branch number

zz1= first node number of the branch

zz2 = second node number of the branch

The last lines of the states matrix, which are followed by an asterisk, indicate inductor currents and capacitor voltages that are not considered as state variables. This situation arises when inductor currents or capacitor voltages are not independent (inductors forming a cutset or capacitors forming a loop). The currents and voltages followed by an asterisk can be expressed as a linear combination of the other state variables:

- **x0**: Column vector of initial values of state variables considering the open or closed status of switches.
- **x0sw**: Vector of initial values of switch currents.
- **rlsw**: Matrix (nswitch,2) containing the R and L values of series switch impedances in ohms (Ω) and henries (H). nswitch is the number of switches in the circuit.
- **u, x, y**: Matrices u(ninput,nfreq), x(nstates,nfreq) and y(noutput,nfreq) containing the steady-state complex values of inputs, states and outputs. nfreq is the length of the freq vector. Each column corresponds to a different source frequency as specified by the next argument freq.
- **freq**: Column vector containing the source frequencies ordered by increasing frequencies.
- **Asw, Bsw, Csw, Dsw**: State-space matrices of the circuit including the closed switches. Each closed switch with an internal inductance adds one extra state to the circuit.
- **Hlin**: Three-dimensional array (nfreq, noutput, ninput) of the nfreq complex transfer impedance matrices of the linear system corresponding to each frequency of the freq vector.

Format of the RLC Input Matrix

Two formats are allowed:

- Six columns: Implicit branch numbering. Branch numbers correspond to the RLC line numbers.
- Seven columns: Explicit branch numbering. Branch number Nobr is assigned by the user.

Each line of the RLC matrix must be specified according to the following format:

[node1, node2, type, R, L, C, Nobr] for RLC branch or line branch

[node1, node2, type, R, L, C, Nobr] for transformer magnetizing branch

[node1, node2, type, R, L, U, Nobr] for Transformer winding

[node1, node2, type, R, L, U, Nobr] for mutual inductances

- node1: First node number of the branch. The node number must be positive or zero. Decimal node numbers are allowed.
- node2: Second node number of the branch. The node number must be positive or zero. Decimal node numbers are allowed.
- type: Integer indicating the type of connection of RLC elements, or the transmission line length (negative value).

type=0: Series RLC element

type=1: Parallel RLC element

type=2: Transformer winding

type=3: Coupled (mutual) winding

If type is negative value: transmission line modeled by a PI section. See details below.

For a mutual inductor or a transformer having N windings, N+1 consecutive lines must be specified in RLC matrix:

- 1 N lines with type=2 or type=3; (one line per winding). Each line specifies R/L/U or R/Xl /Xc where [R/L, R/Xl = winding resistance and leakage reactance for a transformer or winding resistance and self reactance for mutually coupled windings. U is the nominal voltage of transformer winding (specify 0 if type =3).
- 2 One extra line with type=1 for the magnetizing branch of a transformer (parallel Rm/Lm or Rm/Xm) or one line with type=0 for a mutual impedance (series Rm/Lm or Rm/Xm).

For a transformer magnetizing branch or a mutual impedance, the first node number is an internal node located behind the leakage reactance of the first winding. The second node number must be the same as the second node number of the first winding.

To model a saturable transformer, you must use a nonlinear inductance instead of the linear inductance simulating the reactive losses. Set the Lm/Xm value to zero (no linear inductance) and use the Transfosat block with proper flux-current characteristics.

This block can be found in the **Powerlib_models** library. This block must be connected to the linear part of the system (State-Space block or S-function) between a voltage output (voltage across the magnetizing branch) and a current input (current source injected into the transformer internal node). See the example given at the end of the `circ2ss` documentation.

If `type` is a negative value: length (km) of a transmission line simulated by a PI section. For a transmission line, the R/L/C or R/Xl/Xc values must be specified in (Ω/km) or (Ω , mH, or μF per km):

- R: Branch resistance (Ω).
- Xl: Branch inductive reactance (Ω at `freq_sys`) or transformer winding leakage reactance (Ω at `freq_sys`).
- L: Branch inductance (mH).
- Xc: Branch capacitive reactance (Ω at `freq_sys`) (The negative sign of Xc is optional).
- C: Capacitance (μF).
- U: Nominal voltage of transformer winding. Same units Volts or kV must be used for each winding. For a mutual inductance (`type=3`), this value must be set to zero.

Zero values for R, L or Xl, C or Xc in a series or parallel branch indicate that the corresponding element does not exist.

The following restrictions apply for transformer winding R-L values. Null values are not allowed for secondary impedances if some transformer secondaries form loops (such as in a three-phase delta connection). Specify a very low value for R or L or both (e.g., $1\text{e-}6$ p.u. based on rated voltage and power) to simulate a quasi-ideal transformer. The resistive and inductive parts of the magnetizing branch can be set to infinite (no losses; specify $X_m=R_m=\text{inf}$).

Format of the SOURCE Input Matrix

Three formats are allowed:

- Five columns: All sources are generating the same frequency specified by `freq_sys`.
- Six columns: The frequency of each source is specified in column 6.
- Seven columns: The seventh column is used to specify the type of nonlinear element modeled by the current source.

Each line of the source matrix must be specified according to the following format:

[node1, node2, type, amp, phase, freq, model]

- node1, node2: Node numbers corresponding to the source terminals
Polarity conventions: Voltage source: node1 is the positive terminal. Current source: Positive current flowing from node1 to node2 inside the source.
- type: Integer indicating the type of source: 0 for voltage source; 1 for current source.
- amp: Amplitude of the AC or DC voltage or current (V or A).
- phase: Phase of the AC voltage or current (degree).
- freq: Frequency (Hz) of the generated voltage or current. Default value is 60 Hz. For a DC voltage or current source specify phase=0 and freq=0. amp can be set to a negative value. The generated signals are:
 $\text{amp} * \sin(2 * \pi * \text{freq} * t + \text{phase})$ for AC, amp for DC.
- model: Integer specifying the type of nonlinear element modeled by the current source (saturable inductance, thyristor, switch...). Used by power2sys only.

Order in Which Sources Must Be Specified

The functions which compute the state-space representation of a system expect the sources to be in a certain order. This order must be respected in order to obtain correct results. You must be particularly careful if the system contains any switches. The following list gives the proper ordering of sources:

- 1 The currents from all switches that have a null inductance ($L_{on}=0$), if any
- 2 The currents from all nonlinear models that have a finite inductance (switches with $L_{on}>0$, the magnetizing inductance in saturable transformers, etc.), if any
- 3 All other voltage and current sources in any order, if any

Refer to the Example section below for an example illustrating proper ordering of sources for a system containing nonlinear elements.

Format of the SWITCHES Input Matrix

Switches are nonlinear elements simulating mechanical or electronic devices such as circuit breakers, diodes, or thyristors. As other nonlinear elements, they are simulated by current sources driven by the voltage appearing across their terminals. Therefore, they cannot have a null impedance. They are simulated as ideal switches in series with a series R-L circuit. Various models of switches (circuit breaker, ideal switch, and power electronic devices) are available in the **Powerlib_models** library. They must be interconnected to the linear part of the system through appropriate voltage outputs and current inputs.

The switch parameters must be specified in a line of the switches matrix in seven different columns according to the following format:

```
[ node1, node2, status, R, L/Xl, no_I, no_U ]
```

- node1, node2: Node numbers corresponding to the switch terminals.
- status: Code indicating the initial status of the switch at $t=0$.
- 0= open; 1= closed
- R: Resistance of the switch when closed (Ω).
- L/Xl: Inductance of the switch when closed (mH) or inductive reactance (Ω at freq_sys).

Note: For these last two fields, the same units as those specified for the r l c matrix must be used. Either can be set to zero, but not both.

The next two fields specify the current input number and the voltage output number to be used for interconnecting the switch model to the state-space block. The output number corresponding to the voltage across a particular switch must be the same as the input number corresponding to the current from the same switch (see Example section below):

- no_I: Current input number coming from the output of the switch model.
- no_U: Voltage output number driving the input of the switch model.

Format of the LINE_DIST Matrix

The distributed parameter line model contains two parts:

- 1 A linear part containing current sources and resistances that are connected at the line sending and receiving buses together with the linear circuit.

- 2 A nonlinear part available in the Dist_line block of the **Powerlib_models** library. This block performs the phase to mode transformations of voltage and currents and simulates the transmission delays for each mode. The Dist_line block must be connected to appropriate voltage outputs and current inputs of the linear part of the system. The line parameters have to be specified in the line_dist matrix and also in the Dist_line block.

Each row of the line_dist matrix is used to specify a distributed parameter transmission line. The number of columns of line_dist depends on the number of phases of the transmission line.

For an nphase line, the first $(4+3*nphase+nphase^2)$ columns are used. For example, for a three-phase line, 22 columns are used:

[nphase, no_I, no_U, length, L/Xl, Zc, Rm, speed, Ti]

- nphase: Number of phases of the transmission line.
- no_I: Input number in the source matrix corresponding to the first current source Is_1 of the line model. Each line model uses $2*nphase$ current sources specified in the source matrix as follows:
Is_1 Is_2...Is_nphase for the sending end followed by
Ir_1 Ir_2...Ir_nphase for the receiving end.
- no_U: Output number of the state-space corresponding to the first voltage output Vs_1 feeding the line model. Each line model uses $2*nphase$ voltage outputs in the source matrix as follows:
Vs_1 Vs_2...Vs_nphase for the sending end followed by Vr_1
Vr_2...Vr_nphase for the receiving end.
- length: Length of the line (km)
- Zc: Vector of the nphase modal characteristic impedances (Ω)
- Rm: Vector of the nphase modal series resistances (Ω/km)
- speed: Vector of the nphase modal propagation speeds (km/s)
- Ti : Transformation matrix from mode to phase currents such that $I_{\text{phase}} = T_i \cdot I_{\text{mod}}$. The $nphase*nphase$ matrix must be given in vector format [col_1, col_2, ... col_nphase].

Format of the YOUT Matrix

The desired outputs are specified by a string matrix yout. Each line of the yout matrix must be an algebraic expression containing a linear combination of states and state derivatives specified according to the following format:

- U_{c_bn} : Capacitor voltage of branch #n.
- I_{l_bn} : Inductor current of branch #n.
- dU_{c_bn} , dI_{l_bn} : Derivative of U_{c_bn} or I_{l_bn} .
- U_n , I_n : Source voltage or current specified by line #n of the source matrix.
- U_{nx1_x2} : Voltage between nodes x1 and x2.
- I_{bn} : Current in branch #n. For a parallel RLC branch, I_{bn} corresponds to the total current $I_R + I_L + I_C$.
- I_{bn_nx} : Current flowing into node x of a pi transmission line specified by line #n of the rlc matrix. This current includes the series inductive branch current and the capacitive shunt current.

Each output expression is built from voltage and current variable names defined above, their derivatives, constants, other variable names, parenthesis and operators (+- */^) in order to form a valid MATLAB expression. For example,

```
yout =
char([' R1*I_b1+Uc_b3- L2*dI_l_b2' , ' U_n10_20' , ' I 2+3*I_b5' ] );
```

If variable names are used (R1 and L2 in the above example), their names and values must be specified by the two input arguments vary_name and vary_val .

Sign Conventions for Voltages and Currents

I_{bn} I_{l_bn} , I_n : Branch current, inductor current of branch #n or current of source #n is oriented from node1 to node2.

I_{bn_nx} : Current at one end (node x) of a pi transmission line. If $x = \text{node1}$, the current is entering the line. If $x = \text{node2}$, the current is leaving the line.

U_{c_bn} , U_n : Voltage across capacitor or source voltage ($U_{\text{node1}} - U_{\text{node2}}$)

U_{nx1_x2} : Voltage between nodes x1 and x2 = $U_{x1} - U_{x2}$. Voltage of node x1 with respect to node x2.

Order in Which Outputs Must Be Specified

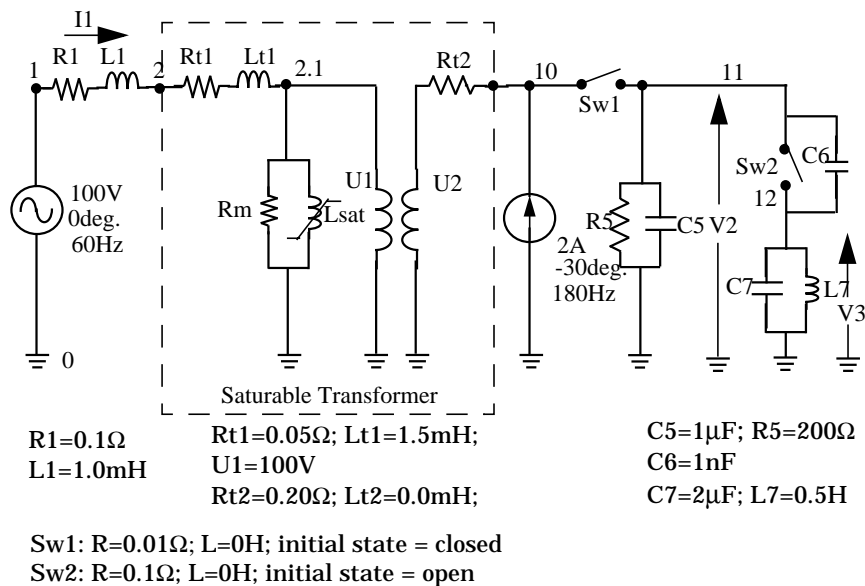
The functions that compute the state-space representation of a system expect the outputs to be in a certain order. This order must be respected in order to obtain correct results. You must be particularly careful if the system contains any switches. The following list gives the proper ordering of outputs:

- 1 The voltages across all switches that have a null inductance ($L_{on}=0$), if any;
- 2 The currents of all switches that have a null inductance ($L_{on}=0$), if any, in the same order as the voltages above;
- 3 The voltages across all nonlinear models that have a finite inductance (switches with $L_{on}>0$, the magnetizing inductance in saturable transformers, etc.);
- 4 All other voltage and current measurements that you request, in any order.

Refer to the Example section below for an example illustrating proper ordering of outputs for a system containing nonlinear elements.

Example

The following circuit consists of two sources (one voltage source and one current source), two series RLC branches ($R1-L1$ and $C6$), two parallel RLC branches ($R5-C5$ and $L7-C7$), one saturable transformer, and two switches ($Sw1$ and $Sw2$). $Sw1$ is initially closed whereas $Sw2$ is initially open. Three measurement outputs are specified ($I1$, $V2$, and $V3$). This circuit has seven nodes numbered 0, 1, 2, 2.1, 10, 11, and 12. Node 0 is used for the ground. Node 2.1 is the internal node of the transformer where the magnetization branch is connected.



You can use the `circ2ss` function to find the state-space model of the linear part of the circuit. The nonlinear elements `Sw1`, `Sw2`, and `Lsat` must be modeled separately by means of current sources driven by the voltage appearing across their terminals. Therefore you must foresee three additional currents sources and three additional voltage outputs for interfacing the nonlinear elements to the linear circuit.

You will obtain the state-space model of the circuit by entering the following commands in a MATLAB script file. The example is available in the `psbci rc2ss.m` file. Notice that an output text file containing information on the system is requested in the call to `circ2ss`.

```
unit='OMU'; % Units = Ohms, mH and uF

rlc=[
%N1N2 typeR L C(uF)/U(V)
1 2 0 0.1 1 0 %R1 L1
2 0 2 0.05 1.5 100 %transfo Wind. #1
10 0 2 0.200 200 %transfo Wind. #2
2.10 1 10000 0 %transfo mag. branch
11 0 1 200 0 1 %R5 C5
11 12 0 0 0 1e-3 %C6
12 0 1 0 500 2 %L7 C7
];

source=[
%N1N2 typeU/I phasefreq
10 11 1 0 0 0 %Sw1
11 12 1 0 0 0 %Sw2
2.10 1 0 0 0 %Saturation
1 0 0 100 0 60 %Voltage source
0 10 1 2 -30 180 %Current source
];

switches=[
%N1N2 statusR(ohm)L(mH)I#U#
10 11 1 0.010 1 1 %Sw1
11 12 0 0.1 0 2 2 %Sw2
];
```



```
%outputs
%
% Both switches have Lon=0, so their voltages must be the first
% outputs,
% immediately followed by their currents (in the same order as the
% voltages).
% The voltage across all nonlinear models which don't have L=0
% follow (in
% this case the saturable transformer's magnetizing inductor).
% The measure-
% ments which you request follow, in any order.
%
y_u1='U_n10_11'; %U_Sw1= Voltage across Sw1
y_u2='U_n11_12'; %U_Sw2= Voltage across Sw2
y_i3='I1'; %I1= Switch current Sw1
y_i4='I2'; %I2= Switch current Sw2
y_u5='U_n2.1_0'; %U_sat= Voltage across saturable reactor
y_i6='I_b1'; %I1 measurement
y_u7='U_n11_0'; %V2 measurement
y_u8='U_n12_0'; %V3 measurement

yout=char(y_u1,y_u2,y_i3,y_i4,y_u5,y_i6,y_u7,y_u8); % outputs
y_type=[0,0,1,1,0,1,0,0]; %output types; 0=voltage 1=current

% Open file that will contain circ2ss output information
fid=fopen('psbcirc2ss.net','w');

[A,B,C,D,states,x0,x0sw,r1sw,u,x,y,freq,Asw,Bsw,Csw,Dsw,Hlin]=.
..

circ2ss(r1c,switches,source,[],yout,y_type,unit,[],[],[],0,fid)
;
```

While circ2ss is executing the following messages are displayed:

```
Computing state-space representation of linear electrical circuit
(V2.0)...
(4 states ; 5 inputs ; 7 outputs)

Oscillatory modes and damping factors:
F=159.115Hz zeta=4.80381e-08
```

Steady state outputs @ F=0 Hz :

```
y_u1= 0Volts  
y_u2= 0Volts  
y_i3= 0Amperes  
y_i4= 0Amperes  
y_u5= 0Volts  
y_i6= 0Amperes  
y_u7= 0Volts  
y_u8= 0Volts
```

Steady state outputs @ F=60 Hz :

```
y_u1 = 0.009999 Volts < 3.168 deg.  
y_u2 = 199.4 Volts < -1.148 deg.  
y_i3 = 0.9999 Amperes < 3.168 deg.  
y_i4 = 0 Amperes < 0 deg.  
y_u5 = 99.81 Volts < -1.144 deg.  
y_i6 = 2.099 Amperes < 2.963 deg.  
y_u7 = 199.4 Volts < -1.148 deg.  
y_u8 = 0.01652 Volts < 178.9 deg.
```

Steady state outputs @ F=180 Hz :

```
y_u1 = 0.00117 Volts < 65.23 deg.  
y_u2 = 22.78 Volts < 52.47 deg.  
y_i3 = 0.117 Amperes < 65.23 deg.  
y_i4 = 0 Amperes < 0 deg.  
y_u5 = 11.4 Volts < 53.48 deg.  
y_i6 = 4.027 Amperes < 146.5 deg.  
y_u7 = 22.83 Volts < 52.47 deg.  
y_u8 = 0.0522 Volts < 52.47 deg.
```

The names of the state variables are returned in the states string matrix:

```
states  
states =  
I1_b2_n2_2.1  
Uc_b5_n11_0  
Uc_b6_n11_12  
I1_b7_n12_0  
I1_b1_n1_2*  
Uc_b7_n12_0*
```

Although this circuit contains a total of six inductors and capacitors, there are only four state variables. The names of the state variables are given by the first four lines of the `states` matrix. The last two lines are followed by an asterisk indicating that these two variables are a linear combination of the state variables. The dependencies can be viewed in the output file `psbci rc2ss. net`:

The following capacitor voltages are dependant:

$$Uc_b7_n12_0 = + Uc_b5_n11_0 - Uc_b6_n11_12$$

The following inductor currents are dependant:

$$I1_b1_n1_2 = + I1_b2_n2_0$$

The A,B,C,D matrices contain the state-space model of the circuit without nonlinear elements (all switches open). The `x0` vector contains the initial state values considering the switch `Sw1` closed. The `Asw`, `Bsw`, `Csw`, and `Dsw` matrices contain the state-space model of the circuit considering the closed switch `Sw1`. The `x0sw` vector contains the initial current in the closed switch.

$$A = \begin{bmatrix} -4.0006e+05 & 0 & 0 & 0 \\ 0 & -49950 & -499.25 & 0 \\ 0 & -4992.504 & 9925e+05 & 0 \\ 0 & 2 & -2 & 0 \end{bmatrix}$$

$$Asw = \begin{bmatrix} -80.999 & -199.9900 & 4.9947e+05 & -5244.70 & -499.25 \\ 4.9922e+05 & -5242.104 & 9925e+05 & 0 & 2 & -2 & 0 \end{bmatrix}$$

The system source frequencies are returned in the `freq` vector:

$$freq = \begin{bmatrix} 0 & 60 & 180 \end{bmatrix}$$

The corresponding steady-state complex outputs are returned in the (6-by-3) `y` matrix where each column corresponds to a different source frequency.

For example, you can obtain the magnitude of the six voltage and current outputs at 60 Hz as follows:

```
abs(y(:, 2))
ans =
    0.0099987
   199.42
    0.99987
    0
   99.808
    2.0993
   199.41
    0.016519
```

The initial values of the four state variables are returned in the `x0` vector. You must use this vector in the State-Space block to start the simulation in steady-state.

```
x0
x0 =
    2.3302
   14.111
   14.07
   3.1391e-05
```

The initial values of switch currents are returned in `x0sw`. To start the simulation in steady-state you must use these values as initial currents for the nonlinear model simulating the switches.

```
x0sw
x0sw =
    0.16155
    0
```

The Simulink model of the circuit is shown on the figure below and is available in the `psbci rc2ss_sl k.mdl` file. The linear part of the circuit is simulated by the `sfun_psbcont c` S-function. Appropriate inputs and outputs are used to connect the switch and saturable reactance models to the linear system. Notice that the status of each switch is fed back from the breaker block to the S-function, after the inputs mentioned earlier. You can find the Breaker and Transfosat blocks in the **Powerlib_models** library containing all the nonlinear

models used by the blockset. As the breaker model is vectorized, a single block is used to simulate the two switches Sw1 and Sw2.

If you had used the **powerlib** library to build your circuit, the same Simulink system would have been generated automatically by the power2sys function. The **powerlib** version of this system is also available in the psbci rc2ss_psb.mdl file and is shown below.

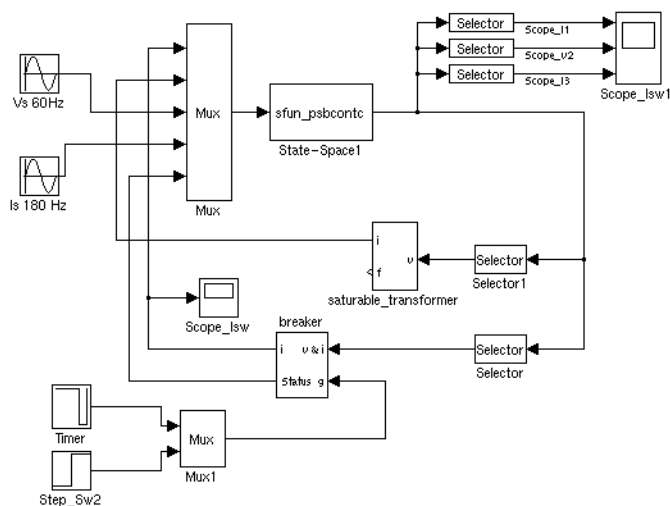


Figure 5-1: psbcirc2ss_slk.mdl Example diagram

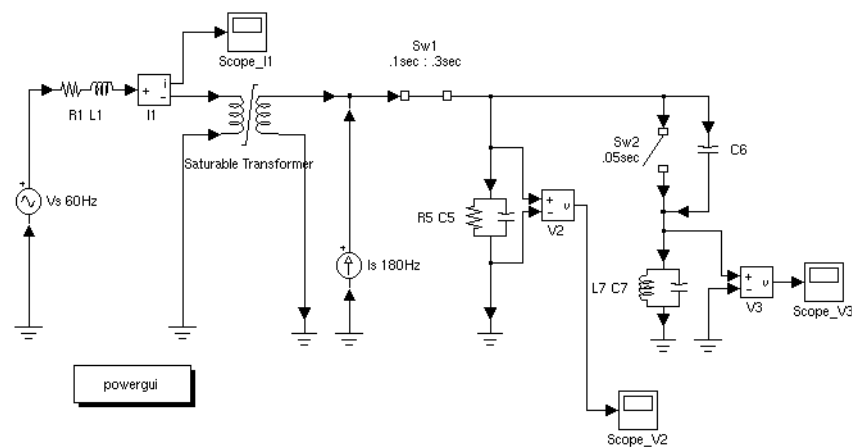


Figure 5-2: psbcirc2ss_psb.mdl Example Diagram

See Also

power2sys

Purpose	Analyze an electric circuit built with the Power System Blockset
Syntax	<pre>psb = power2sys(' sys' , ' structure') psb = power2sys(' sys' , ' sort') psb = power2sys(' sys' , ' ss') [A, B, C, D, x0, states, i nput s, output s, uss, xss, yss, freqyss, Hl i n]= power2sys(' si mmi n'); psb = power2sys(' sys' , ' net')</pre>
Description	<p>The power2sys function computes the equivalent state-space model of the specified electrical model built with the Power System Blockset. It evaluates the A, B, C, D standard matrices of the state-space system described by the equations</p> $\dot{x} = Ax + Bu$ $y = Cx + Du$ <p>where the state variables contained in the x vector are the inductor currents and capacitor voltages. Nonlinear elements are simulated by current sources driven by the voltages across the nonlinear elements.</p> <p>The inputs of the system contained in the u vector are the voltage and current sources plus the current sources simulating the nonlinear elements. The following conventions are used for inputs:</p> <ul style="list-style-type: none">• Source current flowing in the arrow direction is positive.• Positive source voltage is indicated by a + sign on the icon. <p>The outputs of the system contained in the y vector are the voltage and current measurement plus the voltages across the nonlinear elements.</p> <p>psb= power2sys(' sys' , ' structure') creates a structure array with fields and values describing the model 'sys'.</p>

The fields are defined in the following order.

Field	Description
circuit	The name of the model
states	char array of state variable names
inputs	char array of system input names
outputs	char array of system output names
A	nstates-by-nstates state-space A matrix
B	nstates-by-ninput state-space B matrix
C	noutput-by-nstates state-space C matrix
D	noutput-by-ninput state-space D matrix
x0	nstates-by-1 vector of initial conditions
Aswitch	A matrix including closed switches
Bswitch	B matrix including closed switches
Cswitch	C matrix including closed switches
Dswitch	D matrix including closed switches
x0switch	Vector of initial values of switch currents
uss	ninput -by-nfreq steady-state values of inputs
xss	nstates-by-nfreq steady-state values of states variables
yss	noutput-by-nfreq steady-state values of outputs
Hlin	nfreq-by-noutput-by-ninput transfer function of impedances
frequencies	1-by-nfreq vector of input source frequencies

Field	Description
LoadFlow	Load flow information for circuits with machines
OscillatoryModes	Oscillatory modes of linear part of the system

The table uses the following conventions:

- `nstates` is the number of states.
- `ni nput` is the number of inputs.
- `noutput` is the number of outputs.
- `nfreq` is the number of input source frequencies.

`states` is a string matrix containing names of the state variables. Each line of `states` begins with a prefix `Uc_` for capacitor voltages or `Il_` for inductor currents, followed by the name of the block in which the element (C or L) is found. Inductor current direction and capacitor voltages polarities are defined by the input and output of the block. The following conventions are used:

- Current flowing in the arrow direction is positive.
- Voltage equals $V_{input} - V_{output}$.

A suffix is added to the line for blocks containing more than two inductances or capacitors. For example, the Linear Transformer blocks will produce three lines in the `states` matrix, one for each leakage inductance, with the suffix `wi ndi ngx` where `x` is the winding number of the transformer.

`i nputs` is a string matrix containing names of the inputs of the system. Each line of `i nputs` begins with a prefix `U_` for voltage sources or `I_` for current sources, followed by the name of the source block.

A suffix may be added to the input for blocks containing more than one source. For example, the Simplified Synchronous Machine block produces two current inputs with suffixes `AB` and `BC`.

`outputs` is a string matrix containing names of the outputs of the state-space system (vector `y`). Each line of `outputs` begins with a prefix `U_` for voltage outputs or `I_` for current outputs, followed by the name of the block which produces the output. Sign conventions are indicated by the polarities of the voltage measurement and current measurement blocks.

A, B, C, D are the state-space matrices of the linear part of the model.

x0 is a vector containing the initial conditions of the state variables listed in the states variable.

uss, xss, and yss are complex matrices containing the steady state values of inputs, states and outputs. If voltage and current sources all generate the same frequency, these are column vectors. If sources with different frequencies are used, each column of the matrices corresponds to a frequency contained in the frequencies vector.

frequencies is a column vector containing the input source frequencies ordered by increasing values.

Hlin is the complex transfer impedance three-dimension array (nfreq-by-noutput-by ninput) of the linear system corresponding to the frequencies contained in the frequencies vector. For a particular frequency, Hlin is defined by:

$$y_{ss}(:,ifreq) = Hlin(ifreq,:,:) \times u_{ss}(:,ifreq)$$

psb = power2sys(' sys' , ' sort') returns a structure array with the following fields related to the interconnection of Power System Blockset blocks in a model. The fields are defined in the following order.

Field	Description
circuit	The name of the model
SampleTime	Sample time for discrete systems
RlcBranch	r l c matrix in the ci rc2ss format
RlcBranchNames	List of blocks containing state variable
SourceBranch	Source matrix in the ci rc2ss format
SourceBranchNames	The names of the blocks defined as sources
InputNames	Names of the inputs of the system
OutputNames	Names of the outputs of the system
OutputExpressions	Output expression in the ci rc2ss format

Field	Description
OutputMatrix	Output expression in matrix format (internal)
MeasurementBlocks	Names of the voltage and current measurement

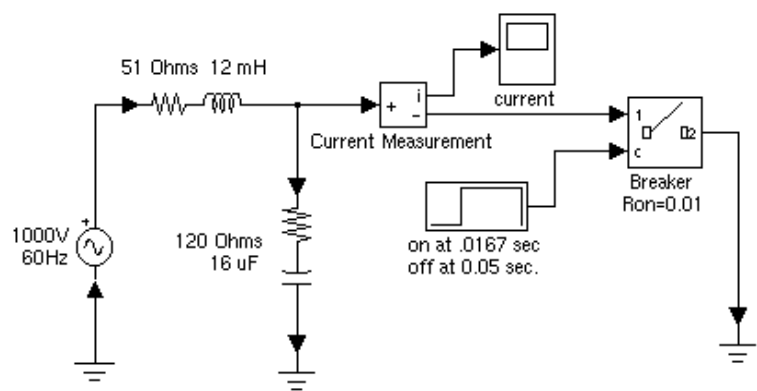
[A, B, C, D, x0, states, inputs, outputs, uss, xss, yss, frequencies, Hl i n] = power2sys(' sys') returns the state-space calculations into separate variables.

If you own the Control System Toolbox, psb = power2sys(' sys' , ' ss') creates a continuous state-space model of 'sys' model with matrices A, B, C, D. The output is a SS object.

Netlist. When called with an extra argument ' net ' , powers2sys generates a netlist stored in a file, sys. net. This file contains the node numbers automatically generated by power2sys, as well as parameter values of all linear elements. See formats in the ci rc2ss reference page.

Example

Obtain the state-space matrices and steady-state voltages and currents for the psbnetsi m2. mdl circuit.



The command

```
psb = power2sys(' psbnetsi m2' , ' structure' )
```

returns the state-space model in psb structure variable.

```
psb.A =  
    1.0e+04 *  
        0      6.2500  
    -0.0083 -1.4250  
  
psb.uss =  
    0  
    10000  
  
psb.xss =  
    1.0e+02 *  
    4.8392 - 5.1314i  
    0.0310 + 0.0292i  
  
psb.yss =  
    1.0e+02 *  
    8.5535 - 1.6287i  
    0  
    0  
  
psb.inputs =  
    I_Breaker  
    U_1000V 60Hz  
  
psb.outputs =  
    U_Breaker  
    I_Breaker  
    I_Current Measurement
```

The inductor of the 51 ohms 12 mH block and the capacitor of the 120 ohms 16 uF block are the two state variables in this circuit. The Breaker block is a nonlinear element that is represented by the first current source driven by the voltage across the breaker (the first output). Note that the current of the Breaker block is also an output of the system (third output).

See Also

`ci rc2ss`, `powerini t`, `powergui`

Purpose	Set the initial states values of a model built with the Power System Blockset
Syntax	<pre>powerinit('sys','look') powerinit('sys','reset') powerinit('sys','steady') powerinit('sys','set',X0) powerinit('sys','setb','StateVariableName',Value)</pre>
Description	<p><code>powerinit('sys','look')</code> displays the current initial states for the specified system.</p> <p><code>powerinit('sys','reset')</code> resets to zero the initial states of specified system.</p> <p><code>powerinit('sys','steady')</code> sets the initial states of specified system in order to start the simulation in steady-state.</p> <p><code>powerinit('sys','set',X0)</code> sets the initial states values of 'sys' to the specified vector X0. The ordering of the states variables is given by the <code>powerinit('sys','look')</code> command.</p> <p><code>powerinit('sys','setb','StateVariableName',Value)</code> sets the initial state variable specified in 'StateVariableName' to Value. The names of the states variables are given by the <code>powerinit('sys','look')</code> command.</p>
Example	<p>The following commands reset to zero the initial states values of the <code>psbfilter.mdl</code> demo.</p> <pre>psbfilter powerinit('psbfilter','reset')</pre> <p>This command returns the name of the states and their current values.</p> <pre>powerinit('psbfilter','look')</pre> <p>Initial states for a particular case:</p> <pre>Il_5th Harm. Filter = 0 Uc_5th Harm. Filter = 0 Il_Zsource = 0</pre>
See Also	<code>powergui</code> , <code>power2sys</code>

A

- AC Current Source 4-9
- AC transmission network 2-3
- AC Voltage Source 4-11
- analyze
 - power2sys command 5-21
 - Powergui block graphical interface 4-127
- Asynchronous Machine 4-13, 4-25
 - per unit system 4-18

B

- block diagrams
 - creating 1-3
- blocks
 - nonlinear 3-14
 - powerlib block library 1-4
- Breaker 4-25
- Bus Bar 4-30

C

- circ2ss command 5-3
- circuit
 - building a simple 1-3
- circuit breaker 4-25
- connecting Simulink blocks 1-7
- Connectors library 4-3, 4-6
- control
 - Speed Control System 4-21
 - using the Control System Toolbox 1-14
- Control System Toolbox 1-14
- Controlled Current Source 4-31
- Controlled Voltage Source 4-21, 4-34
- Current Measurement 4-37

D

- DC Voltage Source 4-43
- Demos library 4-3
- Diode 4-45
- display signals 1-7
- Distributed Parameter Line 4-51
- distributed parameter line
 - propagation speed 1-15
- drives
 - DC motor 2-21
 - variable-frequency induction motor 2-52

E

- electric blocks
 - connecting Simulink blocks 1-7
- electrical circuits 1-2
- Electrical Sources library 4-3
- Elements library 4-3, 4-4
- examples
 - buck converter 4-66
 - circuit breaker 4-28
 - distributed parameter line 4-54
 - line energization 4-124
 - modulated current source 4-32
 - permanent magnet synchronous machine 4-119
 - PWM inverter 4-21
 - single pulse rectifier 4-201
 - surge arresters in series compensated network 4-168
 - synchronous machine in motoring mode 4-181
 - zero-current-quasi-resonant switch converter 4-84, 4-97
- Excitation System 4-58

F

feedback linearization 2-35
 simulation 2-38
frequency analysis 4-112

G

Ground 4-61
GTO 2-21, 4-62

H

HVDC system 2-52
Hydraulic Turbine and Governor 4-69, 4-157

I

Ideal Switch 4-75
interconnections
 between electric and Simulink blocks 1-3
interface
 between Simulink and electric circuit 1-7

L

libraries
 Connectors 4-3
 Demos 4-3
 Electrical Sources 4-3, 4-4
 Elements 4-3
 Machines 4-3
 Measurements 4-3
 Power Electronics 4-3
linear and nonlinear elements 1-3
Linear Transformer 4-89

M

Machines library 4-3, 4-5
Measurements
 Current 4-37
measurements
 voltage 4-213
Measurements library 4-3, 4-6
models
 limitations with nonlinear 3-14
 nonlinear model library 3-14
Mosfet 4-94
 inverter 2-41
Mutual Inductance 4-104

N

Neutral 4-108
nonlinear models
 adding 3-16

P

Parallel RLC Branch 4-110
Parallel RLC Load 4-114
per unit system 1-7
Permanent Magnet Synchronous Machine 4-116
PI Section Line 4-122
PI section line
 frequency response 1-15
Power Electronics library 4-3, 4-5
power system 1-3
Power Systems Blockset 1-2
power2sys command 5-21
Powergui block 4-127
powerinit command 5-27
powerlib library 1-3
PWM 2-22, 2-40

PWM inverter 4-21

S

Saturable Transformer 4-137

saturable transformer model 2-8

Series RLC Branch 4-144

Series RLC Load 4-148

series-compensated transmission network 2-3

Simplified Synchronous Machine 4-151

simulation

- modifying block parameter 1-8

- speed 3-19

sinusoidal source 1-4

snubber circuits 4-45

state variable

- names 1-11

state-space model

- obtaining state-space matrices 5-25

Surge Arrester 4-166

Synchronous Machine 4-171

synchronous machine 2-32

- and regulators 2-32

- with hydraulic turbine 2-32

T

Thyristor 4-197

transformer 2-8

- linear 4-89

transmission line

- propagation time 1-15

V

Voltage Measurement 4-213

