MapleSim User's Guide

MapleSim User's Guide

This document was produced using a special version of Maple and DocBook.

Printed in Canada

ISBN 978-1-926902-09-8

Contents

Introduction

MapleSim Overview

MapleSim^{TM} is a modeling environment for creating and simulating complex multi-domain physical systems. It allows you to build component diagrams that represent physical systems in a graphical form. Using both symbolic and numeric approaches, MapleSim automatically generates model equations from a component diagram and runs high-fidelity simulations.

Build Complex Multi-domain Models

You can use MapleSim to build models that integrate components from various engineering fields into a complete system. MapleSim features a library of over 300 modeling components, including electrical, hydraulic, mechanical, and thermal devices; sensors and sources; and signal blocks. You can also create custom components to suit your modeling and simulation needs.

Advanced Symbolic and Numeric Capabilities

MapleSim uses the advanced symbolic and numeric capabilities of Maple^{TM} to generate the mathematical models that simulate the behavior of a physical system. You can, therefore, apply simplification techniques to equations to create concise and numerically efficient models.

Pre-built Analysis Tools and Templates

MapleSim provides various pre-built templates in the form of Maple worksheets for viewing model equations and performing advanced analysis tasks, such as parameter optimization. To analyze your model and present your simulation results in an interactive format, you can use Maple features such as embedded components, plotting tools, and document creation tools. You can also translate your models into C code and work with them in other applications and tools, including applications that allow you to perform real-time simulation.

Interactive 3-D Visualization Tools

The MapleSim 3-D visualization environment allows you to build and animate 3-D graphical representations of your multibody mechanical system models. You can use this environment to validate the 3-D configuration of your model and visually analyze the behavior of your system under different conditions.

Related Products

MapleSim 5 requires Maple 15.

Maplesoft^{TM} also offers a suite of toolboxes, add-ons, and other applications that extend the capabilities of Maple and MapleSim for engineering design projects. For a complete list of products, visit **http://www.maplesoft.com/products**.

Related Resources

Resource	**Description**
MapleSim Installation Guide	System requirements and installation instructions for MapleSim. The **MapleSim Installation Guide** is available in the **Install.html** file on your MapleSim installation DVD.
MapleSim Help System	Provides the following information: • **MapleSim User's Guide**: conceptual information about MapleSim, an overview of MapleSim features, and tutorials to help you get started. • **Using MapleSim**: help topics for model building, simulation, and analysis tasks. • **MapleSim Component Library**: descriptions of the modeling components available in MapleSim.
MapleSim Examples	Sample models from various engineering domains. These models are available in the **Examples** palette in the **Libraries** tab on the left side of the MapleSim window.
MapleSim Online Resources	Training webinars, product demonstrations, videos, sample applications, and more. For more information, visit **http://www.maplesoft.com/products/maplesim**.
MapleSim Application Center	A collection of sample models, custom components, and analysis templates that you can download and use in your MapleSim projects. For more information, visit **http://www.maplesoft.com/applications/maplesim**.

For additional resources, visit **http://www.maplesoft.com/site_resources**.

Getting Help

To request customer support or technical support, visit **http://www.maplesoft.com/support**.

Customer Feedback

Maplesoft welcomes your feedback. For comments related to the MapleSim product documentation, contact **doc@maplesoft.com**.

1 Getting Started with MapleSim

In this chapter:

1.1 Physical Modeling in MapleSim

Physical modeling, or physics-based modeling, incorporates mathematics and physical laws to describe the behavior of an engineering component or a system of interconnected components. Since most engineering systems have associated dynamics, the behavior is typically defined with ordinary differential equations (ODEs).

To help you develop models quickly and easily, MapleSim provides the following features:

Topological or “Acausal” System Representation

The signal-flow approach used by traditional modeling tools requires system inputs and outputs to be defined explicitly. In contrast, MapleSim allows you to use a topological representation to connect interrelated components without having to consider how signals flow between them.

Mathematical Model Formulation and Simplification

A topological representation maps readily to its mathematical representation and the symbolic capability of MapleSim automates the generation of system equations.

When MapleSim formulates the system equations, several mathematical simplification tools are applied to remove any redundant equations and multiplication by zero or one. The simplification tools then combine and reduce the expressions to get a minimal set of equations required to represent a system without losing fidelity.

Advanced Differential Algebraic Equation Solvers

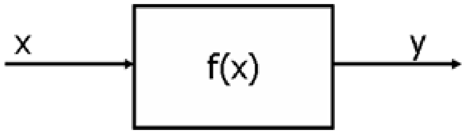
Algebraic constraints are introduced in the topological approach to model definition. Problems that combine ODEs with these algebraic constraints are called Differential Algebraic Equations (DAEs). Depending on the nature of these constraints, the complexity of the DAE problem can vary. An index of the DAEs provides a measure of the complexity of the problem. Complexity increases with the index of the DAEs.

The development of generalized solvers for complex DAEs is the subject of much research in the symbolic computation field. With Maple as its computation engine, MapleSim uses advanced DAE solvers that incorporate leading-edge symbolic and numeric techniques for solving high-index DAEs.

Acausal and Causal Modeling

Real engineered assemblies, such as motors and powertrains, consist of a network of interacting physical components. They are commonly modeled in software by block diagrams. The lines connecting two blocks indicate that they are coupled by physical laws.

When simulated by software, block diagrams can either be causal or acausal. Many simulation tools are restricted to causal (or signal-flow) modeling. In these tools, a unidirectional signal, which is essentially a time-varying number, flows into a block. The block then performs a well-defined mathematical operation on the signal and the result flows out of the other side. This approach is useful for modeling systems that are defined purely by signals that flow in a single direction, such as control systems and digital filters.



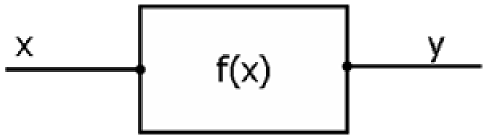
This approach is analogous to an assignment, where a calculation is performed on a known variable or set of variables on the right hand side and the result is assigned to another variable on the left:

$y := f(x)$

Modeling how real physical components interact requires a different approach. In acausal modeling, a signal from two connected blocks travels in both directions. The programming analogy would be a simple equality statement:

$y = f(x)$

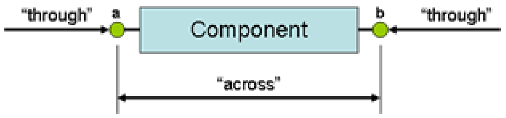
The signal includes information about which physical quantities (for example, energy, current, torque, heat and mass flows) must be conserved. The blocks contain information about which physical laws (represented by equations) they must obey and, hence, which physical quantities must be conserved.



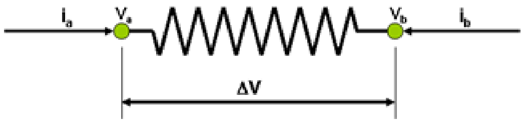
MapleSim allows you to use both approaches. You can simulate a physical system (with acausal modeling) together with the associated logic or control loop (with causal modeling) in a manner that suits either task best.

Through and Across Variables

When using the acausal modeling approach, it is useful to identify the *through* and *across* variables of the component you are modeling. In general terms, an across variable represents the driving force in a system and a through variable represents the flow of a conserved quantity:



For example, in an electrical circuit, the through variable, *i*, is the current and the across variable, *V*, is the voltage drop:



The following table lists some examples of through and across variables for other domains:

Domain	**Through**	**Across**
Electrical	Current (*A*)	Voltage (*V*)
Magnetic	Magnetic Flux (Wb)	MMF (A)
Mechanical (translational)	Force (*N*)	Velocity (*m/s*)
Mechanical (rotational)	Torque (*N.m*)	Angular Velocity (*rad/s*)
Hydraulic	Flow (m^3/s)	Pressure (N/m^2)
Heat flow	Heat flow (*W*)	Temperature (*K*)

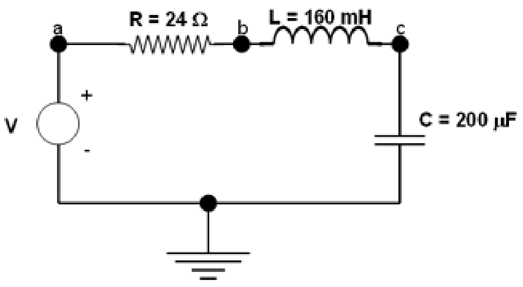
As a simple example, the form of the governing equation for a resistor is

$$V = R \cdot i$$

This equation, in conjunction with Kirchhoff's conservation of current law, allows a complete representation of a circuit.

$$R \cdot i_b = V_b - V_a \text{ and } i_b + i_a = 0$$

To extend this example, the following schematic diagram describes an RLC circuit, an electrical circuit consisting of a resistor, inductor, and a capacitor connected in series:



If you wanted to model this circuit manually, it can be represented with the following characteristic equations for the resistor, inductor, and capacitor respectively:

$$R \cdot i_R = V_a - V_b$$

$$L \frac{d}{dt} i_L = (V_b - V_c)$$

$$i_c = C \cdot \frac{d}{dt} V_c$$

By applying Kirchhoff's current law, the following conservation equations are at points *a*, *b*, and *c*:

$$i_V - i_R = 0$$

$$i_R - i_L = 0$$

$$i_L - i_C = 0$$

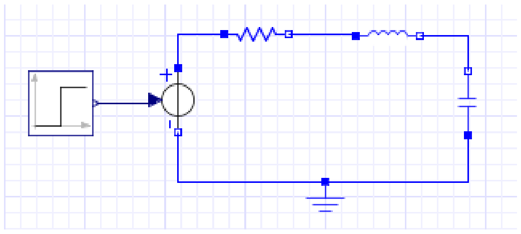
These equations, along with a definition of the input voltage (defined as a transient going from 0 to 1 volt, 1 second after the simulation starts)

$$V_a = \begin{cases} 0.0 & 0.0 \le t < 1.0 \\ 1.0 & \mathrm{t} \ge 1.0 \end{cases}$$

provide enough information to define the model and solve for the voltages and currents through the circuit.

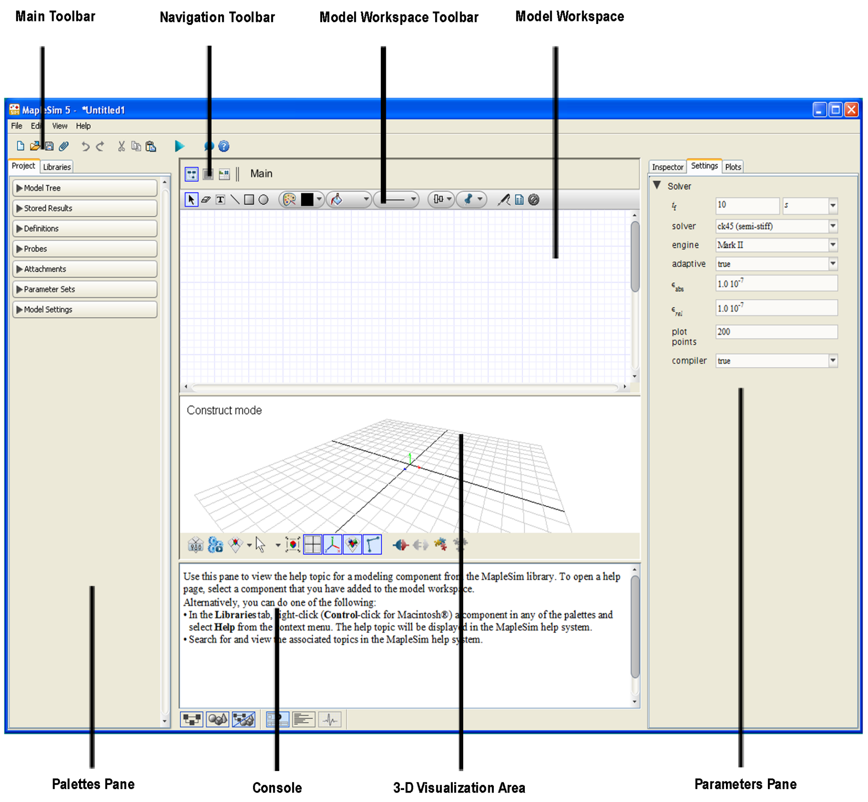
In MapleSim, all of these calculations are performed automatically; you only need to draw the circuit and provide the component parameters. These principles can be applied equally to all engineering domains in MapleSim and allow you to connect components in one domain with components in others easily.

In the *Basic Tutorial: Modeling an RLC Circuit and DC Motor (page 7)* section of this chapter, you will model the RLC circuit described above. The following image shows how the RLC circuit diagram appears when it is built in MapleSim.



1.2 The MapleSim Window

The MapleSim window contains the following panes and components:



t_f	10 s
solver	ck45 (semi-stiff)
engine	Mark II
adaptive	true
ϵ_{abs}	1.0 10^{-7}
ϵ_{rel}	1.0 10^{-7}
plot points	200
compiler	true

Use this pane to view the help topic for a modeling component from the MapleSim library. To open a help page, select a component that you have added to the model workspace.
Alternatively, you can do one of the following:
- In the **Libraries** tab, right-click (**Control**-click for Macintosh®) a component in any of the palettes and select **Help** from the context menu. The help topic will be displayed in the MapleSim help system.
- Search for and view the associated topics in the MapleSim help system.

Component	Description
Main Toolbar	Contains tools for running a simulation, attaching MapleSim analysis templates to your model, and performing other common tasks.
Navigation Toolbar	Contains tools for browsing your model and subsystems hierarchically, and changing the model view.
Model Workspace Toolbar	Contains tools for laying out and selecting objects, and adding elements such as annotations and probes.
Model Workspace	The area in which you build and edit a model in a block diagram view.
Palettes Pane	Contains expandable menus with tools that you can use to build a model and manage your MapleSim project. This pane contains two tabs:

Component	Description
	• **Libraries** tab: contains palettes with sample models and domain-specific components that you can add to models. • **Project** tab: contains palettes with tools to help you browse and build a model, and manage simulation results, probes, and documents that you attach to a model.
Console	Contains the following panes: • **Help** pane: displays the Help topic associated with a modeling component. • **Message Console** pane: displays progress messages indicating the status of the MapleSim engine during a simulation. • **Debugging** pane: displays diagnostic messages as you build your model identifying the subsystem in which the errors are located. You can use the buttons below the console ( ) to display each pane.
3-D Workspace	This visualization area allows you to build, analyze, and playback 3-D graphical representations of multibody systems. Changes to the model in either the 3-D Workspace or the Model Workspace are automatically reflected in both workspaces as you edit and manipulate your model.
Parameters Pane	Contains the following tabs: • **Inspector** tab: allows you to view and edit modeling component properties, such as names and parameter values, and specify simulation options and probe values. • **Settings** tab: allows you to specify simulation options, such as, the duration of the simulation and optional parameter values for the solver, simulation engine, and 3-D workspace. • **Plots** tab: allows you to define custom layouts for simulation graphs and plot windows. The contents of this pane change depending on your selection in the model workspace.

1.3 Basic Tutorial: Modeling an RLC Circuit and DC Motor

This tutorial introduces you to the modeling components and basic tools in MapleSim. It illustrates the ability to mix causal models with acausal models.

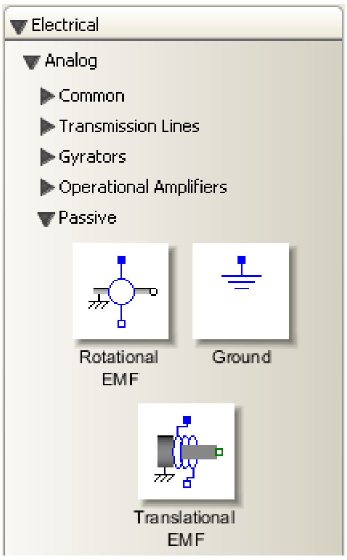
In this tutorial, you will perform the following tasks:

1. Build an RLC circuit model.
2. Set parameter values to specify component properties.
3. Add probes to identify values of interest for the simulation.
4. Simulate the RLC circuit model.
5. Modify the RLC circuit diagram to create a simple DC motor model.
6. Simulate the DC motor model using different parameters.

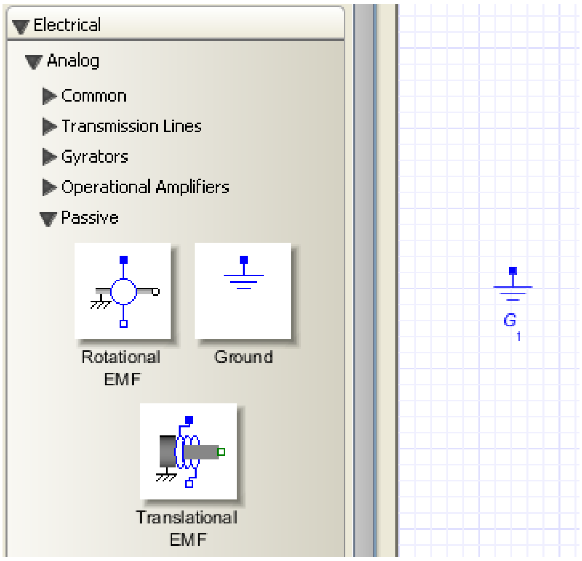
Building an RLC Circuit Model

To build the RLC circuit diagram, you add components in the model workspace and connect them in a system. In this example, the RLC circuit model contains ground, resistor, inductor, capacitor, and signal voltage source components from the Electrical component library. It also contains a step input source, which is a signal generator that drives the input voltage level in the circuit.

1. In the **Libraries** tab at the left of the model workspace, click the triangle beside **Electrical** to expand the palette. In the same way, expand the **Analog** menu, and then expand the **Passive** submenu.



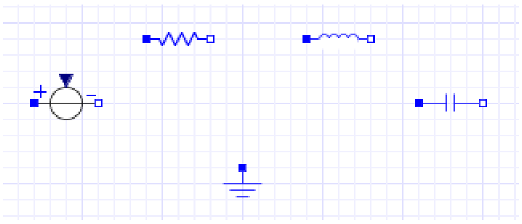
2. From the **Electrical** → **Analog** → **Passive** menu, drag the **Ground** component to the model workspace.



3. Add the remaining electrical components to the model workspace.

- From the **Electrical** → **Analog** → **Passive**→ **Resistors** menu, add the **Resistor** component.
- From the **Electrical** → **Analog** → **Passive** → **Inductors** menu, add the **Inductor** component.
- From the **Electrical** → **Analog** → **Passive** → **Capacitors** menu, add the **Capacitor** component.
- From the **Electrical** → **Analog** → **Sources** → **Voltage** menu, add the **Signal Voltage** component.

4. Drag the components in the arrangement shown below.

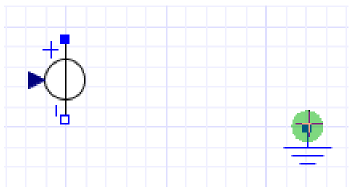


5. To rotate the **Signal Voltage** component clockwise, right-click (**Control**-click for Macintosh®) the **Signal Voltage** component in the model workspace and select **Rotate Clockwise**.

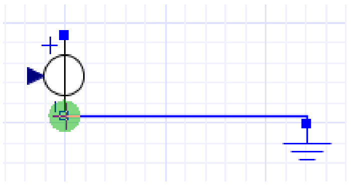
6. To flip the component horizontally, right-click (**Control**-click for Macintosh) the component again and select **Flip Horizontal**. Make sure that the positive (blue) port is at the top.
7. To rotate the **Capacitor** component clockwise, right-click (**Control**-click for Macintosh) the **Capacitor** icon in the model workspace and select **Rotate Clockwise**.

You can now connect the modeling components to define interactions in your system.

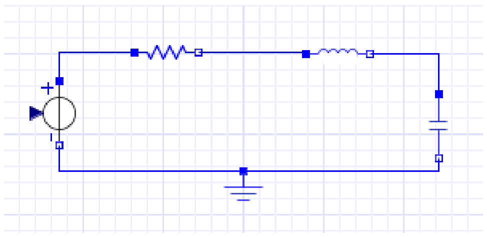
8. Hover your mouse pointer over the **Ground** component port. The port is highlighted in green.



9. Click the **Ground** input port to start the connection line.
10. Hover your mouse pointer over the negative port of the **Signal Voltage** component.



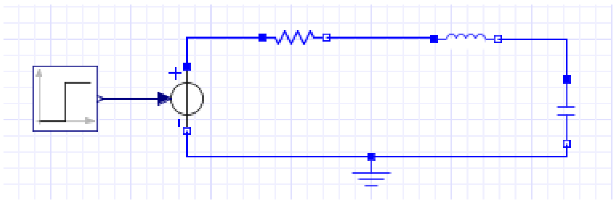
11. Click the port once. The **Ground** component is connected to the **Signal Voltage** component.
12. Connect the remaining components in the arrangement shown below.



You can now add a source to your model.

13. Expand the **Signal Blocks** palette. Expand the **Sources** menu and then expand the **Real** submenu.

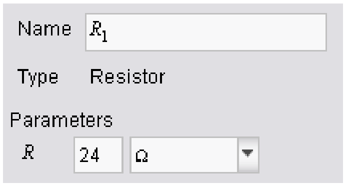
14. From the palette, drag the **Step** source and place it to the left of the **Signal Voltage** component in the model workspace. The step source has a specific signal flow, which is represented by the arrows on the connections. This flow causes the circuit to respond to the input signal.
15. Connect the **Step** source to the **Signal Voltage** component. The complete RLC circuit model is displayed below.



Specifying Component Properties

To specify component properties, you can set parameter values for components in your model.

1. In the model workspace, click the **Resistor** component. The **Inspector** tab at the right of the model workspace displays the name and parameter values of the resistor.
2. In the **R** field, enter **24**, and press **Enter**. The resistance is changed to **24***Ω*.



Name R_1

Type Resistor

Parameters

R 24 Ω

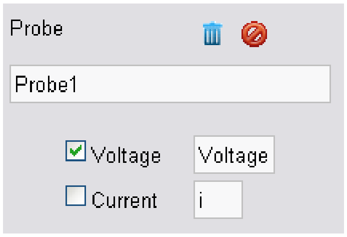
3. Specify the following parameter values for the other components. You can specify units for a parameter by selecting a value from the drop-down menu found beside the parameter value field.

- For the **Inductor**, specify an inductance of **160** *mH*.
- For the **Capacitor**, specify a capacitance of **200** *μF*.
- For the **Step** source, specify a $\mathbf{T_0}$ value of **0.1** s.

Adding a Probe

To specify data values for a simulation, you must attach probes to lines or ports to the model. In this example, you will measure the voltage of the RLC circuit.

1. In the model workspace toolbar, click the probe button ( ).
2. Hover your mouse pointer over the line that connects the **Inductor** and **Capacitor** components. The line is highlighted.
3. Click the line once. The probe is displayed in the model workspace.
4. Drag the probe to position it and then click the probe once to place it on the line.
5. Select the probe. The probe properties appear in the **Inspector** tab at the right of the model workspace.
6. Select the **Voltage** check box to include the voltage quantity in the simulation graph.
7. To display a custom name for this quantity in the model workspace, enter **Voltage** as shown below and press **Enter**.

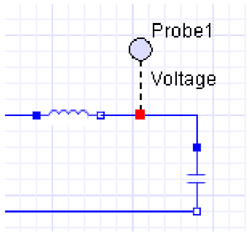


Probe

Probe1

✓ Voltage	Voltage
☐ Current	i

The probe is added to the connection line.



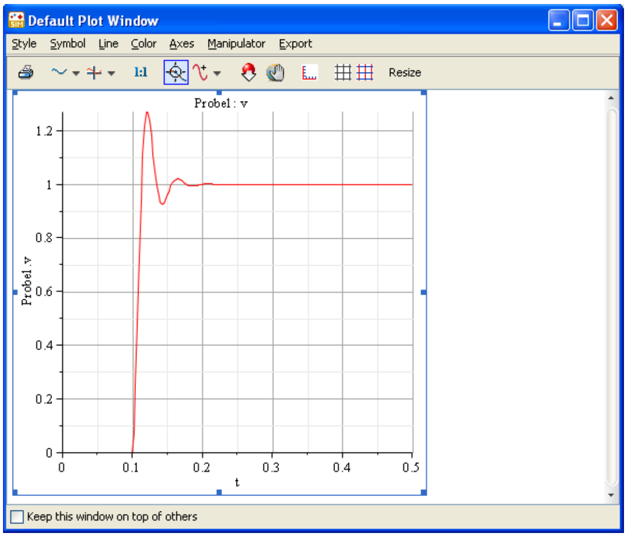
Simulating the RLC Circuit Model

Before simulating your model, you can specify the duration for which to run the simulation.

1. Click the **Settings** tab at the top of the Parameters Pane and in the **Solver** section, set the simulation end time (**t_f**) to **0.5** seconds.
2. Set the compiler to false and press **Enter**.

3. Click the simulation button (▶) in the main toolbar. MapleSim generates the system equations and simulates the response to the step input.

When the simulation is complete, the voltage response is plotted in a graph.

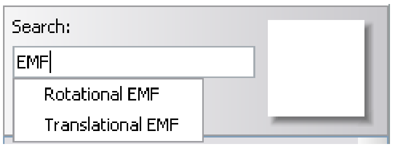


4. Save the model as **RLC_Circuit1.msim**. The probes and modified parameter values are saved as part of the model.

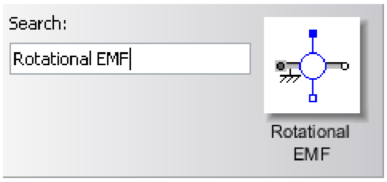
Building a Simple DC Motor Model

You will now add an electromotive force (EMF) component and a mechanical inertia component to the RLC circuit model to create a DC motor model. In this example, you will add components to the RLC circuit model using the search feature.

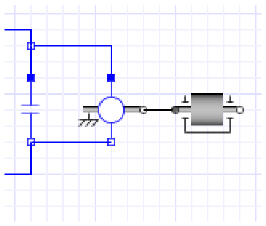
1. In the **Libraries** tab, in the **Search** field located above the palettes, type **EMF**. A drop-down list displays your search results.



2. Select **Rotational EMF** from the drop-down list. The **Rotational EMF** component appears in the square beside the search field.



3. Drag the **Rotational EMF** component to the modeling workspace and place it to the right of the **Capacitor** component.
4. In the **Search** field, search for **Inertia**.
5. Add the **Inertia** component to the model workspace and place it to the right of the **Rotational EMF** component.
6. Connect the components as shown below.



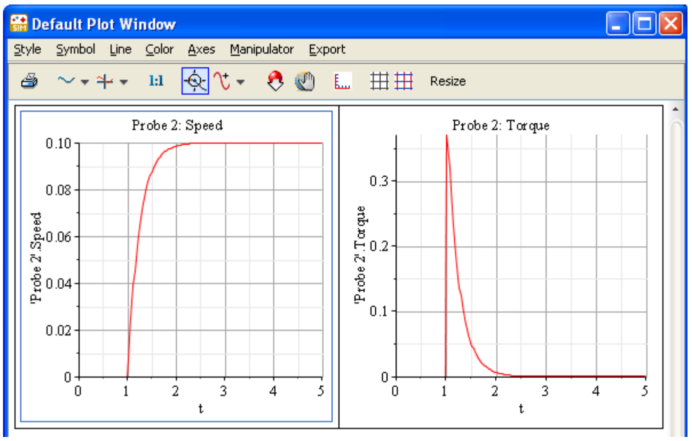
Note: To connect the positive blue port of the **Rotational EMF** component, click the port once, drag your mouse pointer to the line connecting the capacitor and inductor, and click the line.

7. In the model workspace, click the **Rotational EMF** component.
8. Click the **Inspector** tab and in the **Parameters** section, set the value of the transformation coefficient (**k**) to **10** $\frac{N \cdot m}{A}$.
9. Click the **Step** component and change the value of the parameter, $\mathbf{T_0}$, to **1**.

Simulating the DC Motor Model

1. In the model workspace, delete **Probe1**.
2. In the model workspace toolbar, click the probe button ( ).
3. Hover your mouse pointer over the line that connects the **Rotational EMF** and **Inertia** components.
4. Click the line and then click the probe once to position it.
5. Select the probe.

6. In the **Inspector** tab, select the **Speed** and **Torque** check boxes. The probe, with an arrow indicating the direction of the conserved quantity flow, is added to the model.
7. Click a blank area in the model workspace.
8. In the **Settings** tab, set the simulation end time (**t_f**) to **5** seconds.
9. Click the simulation button ( ) in the main toolbar. The following graphs appear.



10. Save the model as **DC_Motor1.msim**.

2 Building a Model

In this chapter:

2.1 The MapleSim Component Library

The MapleSim component library contains over 400 components that you can use to build models. All of these components are organized in palettes according to their respective domains: electrical, hydraulic, 1-D and multibody mechanical, thermal, and signal blocks. Most of these components are based on the Modelica® Standard Library 3.1.

Library	**Description**
Electrical	Components to model electrical analog circuits, single-phase and multiphase systems, and electric machines.
Magnetic	Components to model magnetic circuits.
Hydraulic	Components to model hydraulic systems such as fluid power systems, cylinders, and actuators.
1-D Mechanical	Components to model 1-D translational and rotational systems.
Multibody Mechanical	Components to model multibody mechanical systems, including force, motion, and joint components.
Signal Blocks	Components to manipulate or generate input and output signals.
Thermal	Components to model heat flow and heat transfer.

The library also contains sample models that you can view and simulate, for example, complete electrical circuits and filters. For more information about the MapleSim library structure and modeling components, see the **MapleSim Component Library** in the MapleSim Help system.

To extend the default library, you can create a custom modeling component from a mathematical model and add it to a custom library. For more information, see *Creating Custom Modeling Components (page 61)*.

Viewing Help Topics for Components

In the Help pane below the model workspace, you can view the Help topic for each component from the MapleSim Component Library. To display the Help pane, click the Help pane button (?) at the bottom of the MapleSim window. You can then select a component that you have added to the model workspace to view its Help topic. Alternatively, to view Help topics, you can perform one of the following tasks:

- Right-click (**Control-**click for Macintosh) a modeling component in any of the palettes and select **Help** from the context menu.
- Search for the Help pages for components in the MapleSim Help system.

Updating Models Created in MapleSim 4 or Earlier

In MapleSim 4 or earlier, components from the Modelica Standard Library 2.2.1 were included in the MapleSim Component Library. In MapleSim 4.5, the Modelica Standard Library 2.2.1 was replaced by the Modelica Standard Library 3.1.

If you created a model in MapleSim 4 or earlier, you can open it in MapleSim 4.5 or later; it will then be updated automatically to use equivalent components from the Modelica Standard Library 3.1. For more information, see **Using MapleSim → Updating Models Created in MapleSim 4.5 or Earlier** topic in the MapleSim Help system.

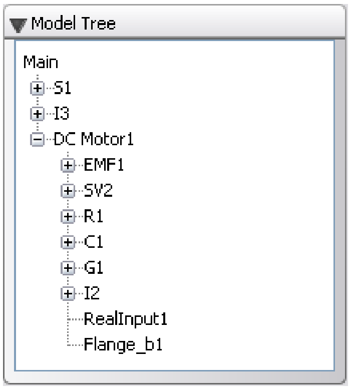
Note: Models that contain components from the Modelica Standard Library 3.1 cannot be used in MapleSim 4 or earlier.

2.2 Browsing a Model

Using the **Model Tree** palette or model navigation controls, you can browse your model to view hierarchical levels of components in the model workspace. You can browse to the top level for an overall view of your system. The top level is the highest level of your model: it represents the complete system, which can include individual modeling components and subsystem blocks that represent groups of components. You can also browse to sublevels in your model to view the contents of individual subsystems or components.

Model Tree

To browse your model, you can use the **Model Tree** palette in the **Project** tab located on the left side of the MapleSim window. Each node in the model tree represents a modeling component, subsystem, or connection port in your model. For example, the model tree of a DC motor is shown below.

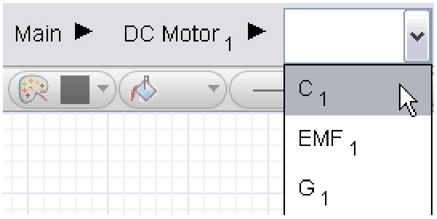


```
Model Tree
Main
⊞ S1
⊞ I3
⊟ DC Motor1
    ⊞ EMF1
    ⊞ SV2
    ⊞ R1
    ⊞ C1
    ⊞ G1
    ⊞ I2
      RealInput1
      Flange_b1
```

To browse your model and view the parameters associated with a component or subsystem, expand and double-click the nodes in the model tree. You can click the **Main** node to view the top level of your model and the child nodes to view the contents of a component or subsystem.

Model Navigation Controls

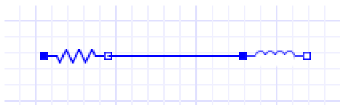
Alternatively, you can use the model navigation controls located above the model workspace toolbar to browse between modeling components, subsystems, and hierarchical levels in a diagram displayed in the model workspace.



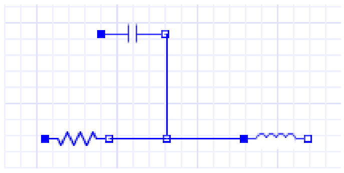
From the drop-down menu, select the name of the subsystem or modeling component that you want to view in the model workspace. You can click the **Main** button to browse to the top level of your model. You can also browse directly to subsystems in your model. For example, by clicking the **DC Motor$_1$** button in the example shown above, you can view the DC motor subsystem contents in the model workspace.

2.3 Defining How Components Interact in a System

To define interactions between modeling components, you connect them in a system. In the model workspace, you can draw a connection line between two connection ports.



You can also draw a connection line between a port and another connection line.



MapleSim permits connections between compatible domains only. By default, each line type appears in a domain-specific color.

Domain	**Line Color**
Mechanical 1-D rotational	Black
Mechanical 1-D translational	Green
Mechanical multibody	Black
Electrical analog	Blue
Electrical multiphase	Blue
Magnetic	Orange
Digital logic	Purple
Boolean signal	Pink
Causal signal	Navy blue
Integer signal	Orange
Thermal	Red

The connection ports for each domain are also displayed in specific colors and shapes. For more information about connection ports, see the **MapleSim Component Library → Connectors Overview** topic in the MapleSim Help system.

2.4 Specifying Component Properties

To specify component properties, you can set parameter values for components in your model. When you select a component in the model workspace, the configurable parameter values for that component appear in the **Inspector** tab located on the right side of the MapleSim window.

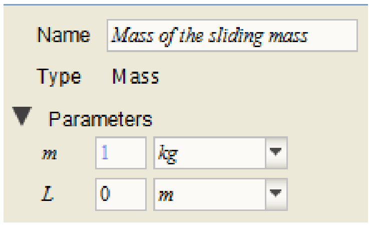
Note: Not all components provide editable parameter values.

You enter parameter values in 2-D math notation, which is a formatting option that allows you to add mathematical text such as superscripts, subscripts, and Greek characters. For more information, see *Entering Text in 2-D Math Notation (page 53)*.

Note: Most parameters in the MapleSim Component Library have default values. However, for some parameters, these default values are simply placeholders that may not represent realistic values for use in a simulation. These placeholder values use a blue font to distinguish them from other parameter values. You should replace these values with values that are more suitable for your simulation. For more information, see **Using MapleSim→Building a Model→Specifying Component Properties** in the MapleSim Help system.

Specifying Parameter Units

You can use the drop-down menus beside parameter fields with dimensions to specify units for parameter values. For example, the image below displays the configurable parameter fields for a **Mass** component. You can optionally specify the mass in *kg*, lb_m, *g*, or *slug*, and the length in *m*, *cm*, *mm*, *ft*, or *in*.



Name *Mass of the sliding mass*

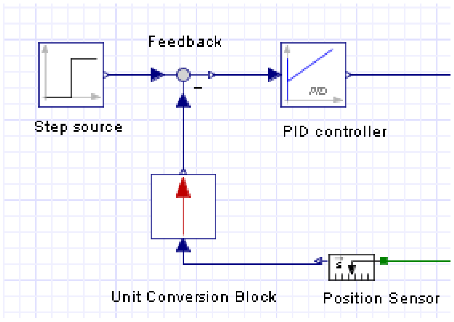
Type Mass

Parameters

m	1	*kg*
L	0	*m*

When you simulate a model, MapleSim automatically converts all parameter units to the International System of Units (SI). You can, therefore, select more than one system of units for parameter values throughout a model.

If you want to convert the units of a signal, use the **Unit Conversion Block** component from the **Signal Converters** menu in the **Signal Blocks** palette. This component allows you to perform conversions in dimensions such as time, temperature, velocity, pressure, and volume. In the following example, a **Unit Conversion Block** component is connected between a translational **Position Sensor** and **Feedback** component to convert the units of an output signal.

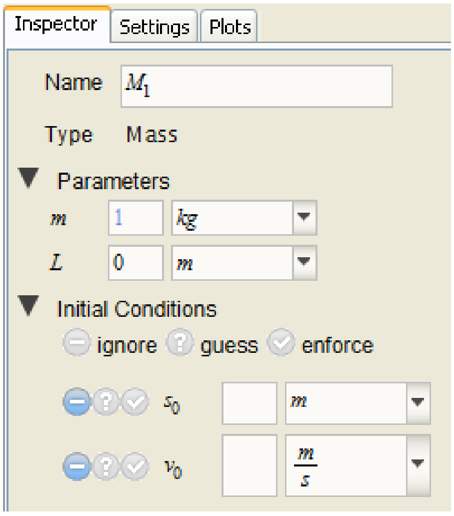


If you include an electrical, 1-D mechanical, hydraulic, or thermal sensor in your model, you can also select the units in which to generate an output signal.

Specifying Initial Conditions

You can set parameter values to specify initial conditions for components from all domains in MapleSim. When you select a component that contains state variables in the model workspace, the available initial condition fields appear in the **Inspector** tab, along with the other configurable parameter values for that component.

For example, the image below displays the initial velocity, position, and acceleration fields that you can set for a **Mass** component.



Name M_1

Type Mass

m	1	kg
L	0	m
s_0		m
v_0		$\frac{m}{s}$

Specifying How Initial Conditions are Enforced

You can determine how the initial conditions that you specified for a particular component are enforced during a simulation. The options are **ignore** (⊖), **guess** (?), and **enforce** (✓). You can select these options for initial condition parameters individually by clicking the buttons beside the applicable initial condition fields.

If you select the **ignore** option, the parameter value that you enter in the initial condition field is ignored during a simulation and the solver uses a default value for the initial condition, typically zero. This option is the default setting for all of the initial condition fields.

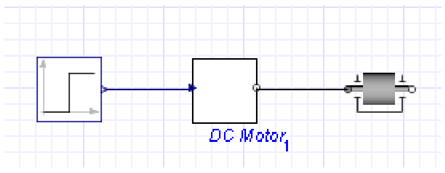
If you select the **guess** option, the solver treats the parameter value you entered in the initial condition field as a best guess value during a simulation. In other words, the best guess value is a starting point for determining the initial configuration of the system for which there is a solution to the set of equations that describe the system. The solver initially computes a solution to the system of equations using this best guess value; however, if no solution is found, the solver computes a solution to the system of equations using an initial condition value that is close to the best guess value.

If you select the **enforce** option, the solver uses the parameter value that you enter in the initial condition field as a start value for the simulation. Similar to the **guess** option, the solver searches for a solution to the system of equations using the parameter value you entered in the initial condition field. However, unlike the **guess** option, if there is no solution, no other value is substituted, and an error message appears.

For more information about selecting these options, see *Best Practices: Enforcing Initial Conditions (page 60)*.

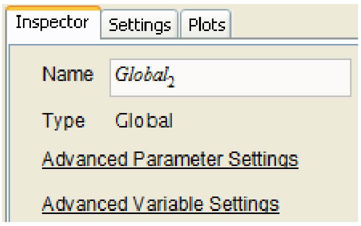
2.5 Creating and Managing Subsystems

A subsystem (or compound component) is a set of modeling components that are grouped in a single block component. A sample DC motor subsystem is shown below.



You can create a subsystem to group components that form a complete system, for example, a tire or DC motor. You can also create a subsystem to improve the layout of a diagram in the model workspace, add multiple copies of a system to a model, analyze a component group in Maple or to quickly assign parameters and variables. You can organize your model hierarchically by creating subsystems within other subsystems.

Once you create a subsystem you will be able to assign parameters and variables to all components in that subsystem using the **Advanced Parameter Settings** and **Advanced Variable Settings** tool in the **Inspector** tab.



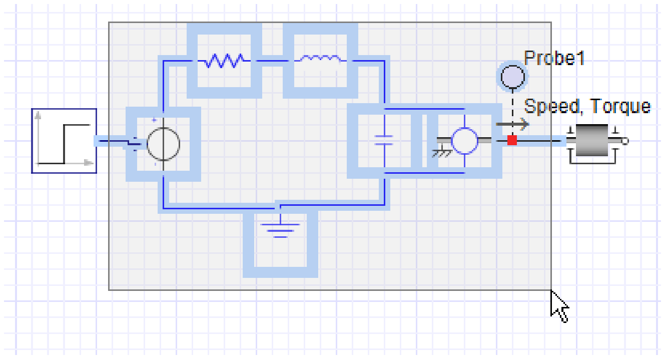
Name $Global_2$
Type Global

For best practices on creating subsystems in MapleSim, see *Best Practices: Laying Out and Creating Subsystems (page 55)*.

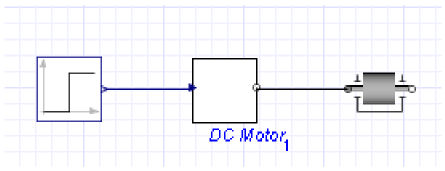
Example: Creating a Subsystem

In the following example, you will group the electrical components of a DC motor model into a subsystem.

1. In the **Libraries** tab on the left side of the MapleSim window, expand the **Examples** palette, expand the **Tutorial** menu, and then open the **Simple DC Motor** example.
2. Using the selection tool ( ) located above the model workspace, draw a box around the electrical components.



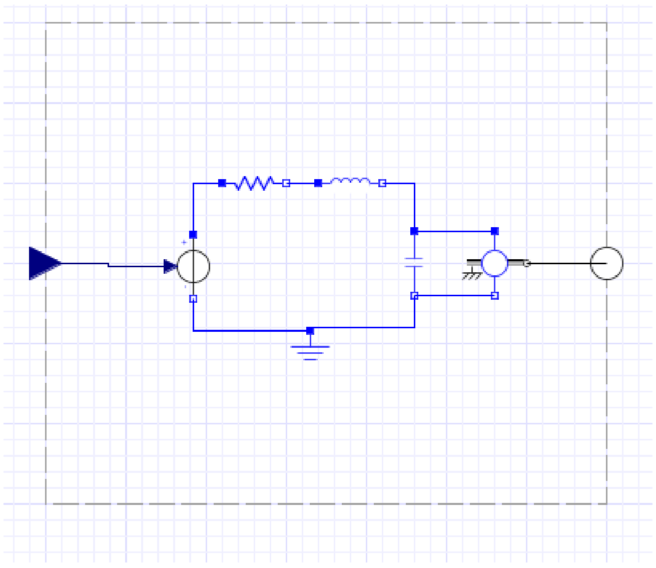
3. From the **Edit** menu, select **Create Subsystem** (or right-click the boxed area and select **Create Subsystem**).
4. In the dialog box, enter **DC Motor**.
5. Click **OK**. A white block, which represents the DC motor, appears in the model workspace.



In this example, you created a *stand-alone subsystem*, which can be edited and manipulated independently of other subsystems in your model. If you want to add multiple copies of the same subsystem to your model and edit those subsystems as a group, you can create a *subsystem definition*. For more information, see *Adding Multiple Copies of a Subsystem to a Model (page 26)*.

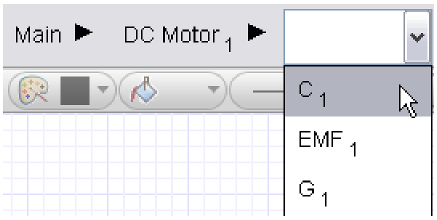
Viewing the Contents of a Subsystem

To view the contents of a subsystem, double-click the subsystem icon in the model workspace. The detailed view of a subsystem appears.



In this view, a broken line indicates the subsystem boundary. You can edit the connection lines and components within the boundary, add and connect components outside of the boundary, and add subsystem ports to connect the subsystem to other components. If you want to resize the boundary, click the broken line and drag one of the sizing handles displayed around the box.

To browse to the top level of the model or to other subsystems, use the controls in the navigation toolbar.



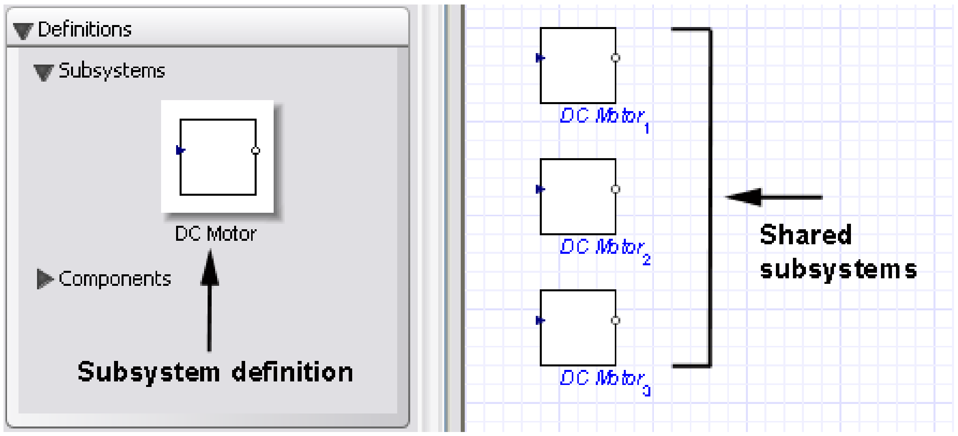
Adding Multiple Copies of a Subsystem to a Model

If you plan to add multiple copies of a subsystem to a model and want all of the copies to have the same configuration, you can create a *subsystem definition*. A subsystem definition is the base subsystem that defines the attributes and configuration that you want a series of subsystems to share.

For example, if you want to add three DC motor subsystems that all have identical components and resistance values in your model, you would perform the following tasks:

1. Build a DC motor subsystem with the desired configuration in the model workspace
2. Use that subsystem configuration to create a subsystem definition and add it to the **Definitions** palette, and
3. Add copies of the DC motor subsystem to your model using the subsystem definition as a source.

To add copies of the DC motor subsystem to your model, you can drag the DC Motor subsystem definition icon from the **Definitions** palette and place it in the model workspace. The copies that you add to the model workspace will then share a configuration that is identical to the subsystem definition in the **Definitions** palette; the copies in the model workspace are called *shared subsystems* because they share and refer to the configuration specified in their corresponding subsystem definition.



Shared subsystems that are copied from the same subsystem definition are *linked*, which means that changes that you make to one shared subsystem will be reflected in all of the other shared subsystems that were created from the same subsystem definition. The changes are also reflected in the subsystem definition entry in the **Definitions** palette.

Using the example shown above, if you change the resistance parameter of the **Resistor** component in the **DC Motor**$_2$ shared subsystem from **24** Ω to **10** Ω, the resistance value of the **Resistor** component in the **DC Motor**$_1$ and **DC Motor**$_3$ shared subsystems and the **DC Motor** subsystem definition in the **Definitions** palette will also be changed to **10** Ω.

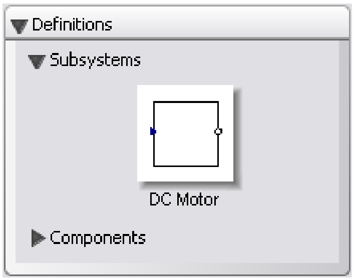
For more information, see *Editing Subsystem Definitions and Shared Subsystems (page 29)*.

Example: Adding Subsystem Definitions and Shared Subsystems to a Model

In the following example, you will create a **DC Motor** subsystem definition and add multiple shared subsystems to your model.

Adding a Subsystem Definition to the Definitions Palette

1. In the model workspace, right-click (**Control**-click for Macintosh) the stand-alone DC motor subsystem that you created in *Example: Creating a Subsystem (page 24)*.
2. From the context menu, select **Convert to Shared Subsystem**.
3. Enter **DC Motor** as the name for the subsystem definition and click **OK**.
4. In the **Project** tab on the left side of the model workspace, expand the **Definitions** palette and then expand the **Subsystems** menu.



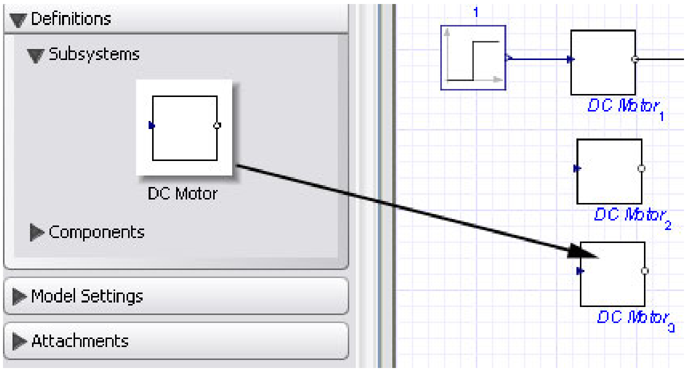
The subsystem definition is added to the **Definitions** palette and the subsystem in the model workspace is converted into a shared subsystem called **DC Motor$_1$**. This shared subsystem is linked to the **DC Motor** subsystem definition.

You can now use this subsystem definition to add multiple DC motor shared subsystems to your MapleSim model.

Tip: If you want to use a subsystem definition in another model, add the subsystem definition to a custom library. For more information, see *Creating and Managing Custom Libraries (page 48)*.

Adding Multiple DC Motor Shared Subsystems to a Model

To add multiple **DC Motor** shared subsystems to a model, drag the **DC Motor** subsystem definition icon from the **Definitions** palette and place it in the model workspace.



When you create a new stand-alone subsystem or add shared subsystems to a model, a unique subscript number is appended to the subsystem name displayed in the model workspace. As shown in the image above, subscript numbers are appended to the names of each DC Motor shared subsystem. These numbers can help you to identify multiple subsystem copies in your model.

Editing Subsystem Definitions and Shared Subsystems

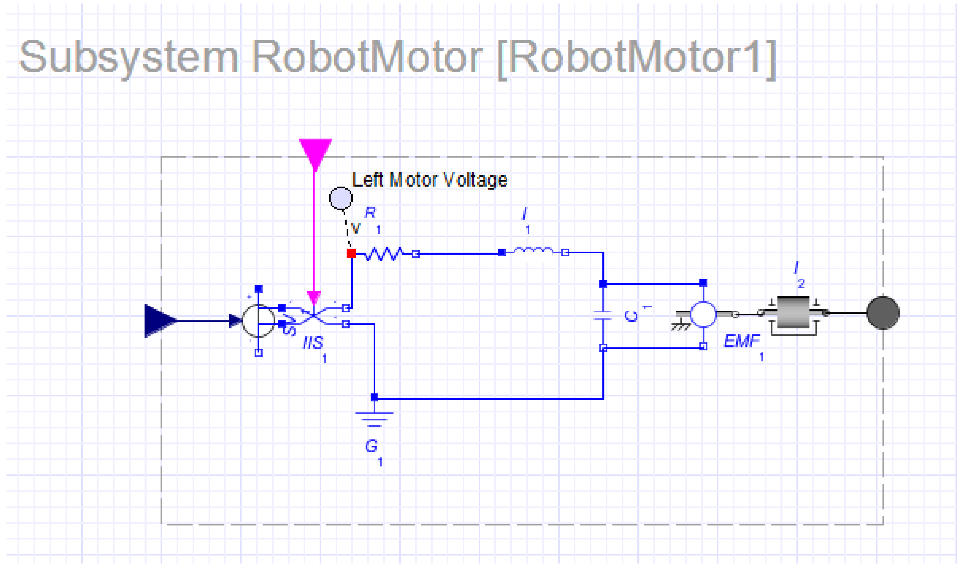
If you edit a shared subsystem in the model workspace, your changes will be reflected in the subsystem definition that is linked to the shared subsystem, as well as other shared subsystems that were copied from the same subsystem definition.

Example: Editing Shared Subsystems that are Linked to the Same Subsystem Definition

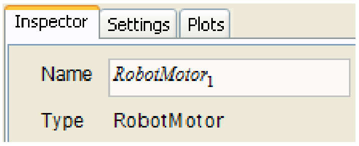
In this example, you will edit the resistance values and subsystem icons in a model that contains two DC motor shared subsystems called **RobotMotor**$_1$ and **RobotMotor**$_2$. These shared subsystems are linked to a subsystem definition called **RobotMotor**. When you change the resistance value in one **RobotMotor** shared subsystem, the other **RobotMotor** shared subsystem and **RobotMotor** shared subsystems that you add in the future will contain the changes.

To start, both **RobotMotor** shared subsystems in this model have a resistance value of **30Ω**.

1. In the **Libraries** tab on the left side of the MapleSim window, expand the **Examples** palette, expand the **Multidomain** menu, and then open the **Sumobot** example.
2. In the model workspace, double-click the **RobotMotor**$_1$ shared subsystem. The detailed view of the shared subsystem appears.



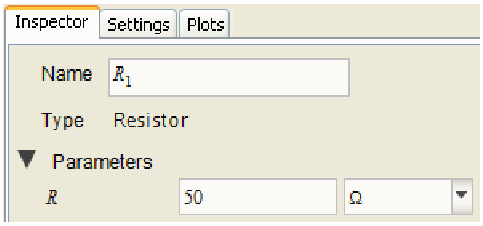
Note that a heading with the subsystem definition name (**RobotMotor**) followed by the shared subsystem name (**RobotMotor1**) appears at the top of the model workspace. In the detailed view of all shared subsystems, this heading also appears to help you identify multiple subsystem copies in your model. Also, when you select a shared subsystem, its subsystem definition name appears in the **Type** field in the **Inspector** tab.



Name $RobotMotor_1$

Type RobotMotor

3. Select the **Resistor** component (**R$_1$**) and, in the **Inspector** tab, click **Parameters**. Change the resistance value to **50 Ω**.



Name R_1

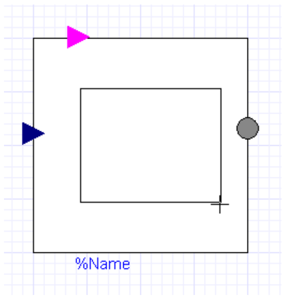
Type Resistor

Parameters

R 50 Ω

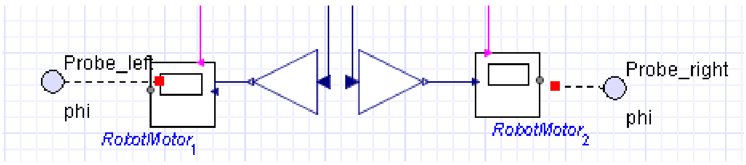
4. In the navigation toolbar, click the icon view button ( ).

5. Using the rectangle tool ( ) in the model workspace toolbar, click and drag your mouse pointer to draw a shape in the box.

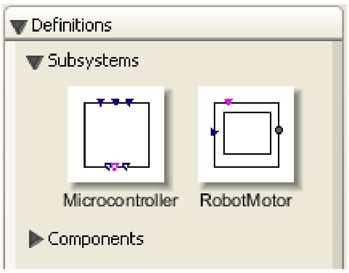


6. In the navigation toolbar, click the diagram view button ( ).

7. Click **Main** in the navigation toolbar to browse to the top level of the model. Both of the **RobotMotor** shared subsystems now display the square that you drew.



8. In the **Project** tab on the left side of the MapleSim window, expand the **Definitions** palette, and then expand the **Subsystems** menu. As shown in the image below, your changes are also reflected in the **RobotMotor** entry in this palette.



If you double-click the **RobotMotor** subsystems in the model workspace and select their **Resistor** components, you will also see that both of the shared subsystems now have a resistance value of **50** Ω .

9. From the **Definitions** palette, drag a new copy of the **RobotMotor** subsystem and place it anywhere in the model workspace. The new copy displays the square that you drew and its resistance value is also **50** Ω .

Example: Removing the Link Between a Shared Subsystem and its Subsystem Definition

If your model contains multiple shared subsystems that are linked and you want to edit one copy only, you can remove the link between a shared subsystem and its subsystem definition, and edit that subsystem without affecting others in the model workspace.

1. In the **Libraries** tab on the left side of the MapleSim window, expand the **Examples** palette, expand the **Multidomain** menu, and then open the **Sumobot** example.
2. In the model workspace, right-click (**Control**-click for Macintosh) the **RobotMotor**$_2$ shared subsystem.
3. Select **Convert to Stand-alone Subsystem**. The **RobotMotor**$_2$ subsystem is no longer linked to the **RobotMotor** subsystem definition in the **Definitions** palette; it is now called **copy of RobotMotor**$_1$.
4. Double-click the **RobotMotor**$_1$ shared subsystem.
5. Click the icon view button ( ).
6. Using the rectangle tool ( ), click and drag your mouse pointer to draw a shape in the box in the model workspace.

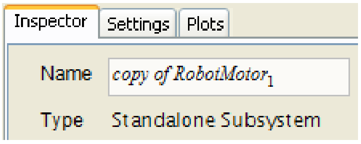
7. Click the diagram view button ( ) and click **Main** to browse to the top level of the model. Your change is shown in the **RobotMotor$_1$** shared subsystem in the model workspace and the **RobotMotor** subsystem definition in the **Definitions** palette. Note that your change is not shown in the **copy of RobotMotor$_1$** subsystem that is no longer linked to the **RobotMotor** subsystem definition.

Tip: When you convert a shared subsystem to a stand-alone subsystem, it is a good practice to assign the stand-alone subsystem a meaningful name that clearly distinguishes it from existing shared subsystems and subsystem definitions.

Working with Stand-alone Subsystems

Stand-alone subsystems are subsystems that are not linked to a subsystem definition. You can create a stand-alone subsystem in two ways: by creating a new subsystem as shown in *Example: Creating a Subsystem (page 24)* or by converting a shared subsystem to a stand-alone subsystem as shown in *Example: Removing the Link Between a Shared Subsystem and its Subsystem Definition (page 31)*. Stand-alone subsystems can be edited independently without affecting other subsystems in the model workspace.

To identify a subsystem as a stand-alone subsystem, select a subsystem in the model workspace and examine the **Inspector** tab. If that subsystem is a stand-alone subsystem, the **Type** field reads **Standalone Subsystem**.



Name *copy of RobotMotor$_1$*
Type Standalone Subsystem

Also, if you double-click a stand-alone subsystem to browse to its detailed view, no heading is shown for the subsystem in the model workspace.

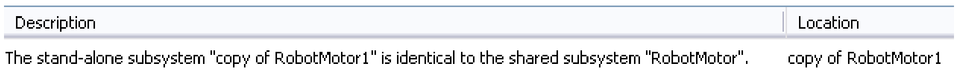
When you copy and paste a stand-alone subsystem in the model workspace, you can optionally convert that subsystem into a shared subsystem and create a new subsystem definition. For more information, see *Example: Copying and Pasting a Stand-alone Subsystem (page 34)*.

Example: Resolving Warning Messages in the Debugging Console

When you convert a shared subsystem into a stand-alone subsystem, the subsystem is highlighted in the model workspace and a warning message appears, informing you that the link to the subsystem definition has been removed.

Note: This example is an extension of *Example: Removing the Link Between a Shared Subsystem and its Subsystem Definition (page 31)*.

1. Click the debugging button ( ) at the bottom of the MapleSim window to display the debugging console. The following warning message appears in the console.



Description	Location
The stand-alone subsystem "copy of RobotMotor1" is identical to the shared subsystem "RobotMotor".	copy of RobotMotor1

2. To work with the **copy of RobotMotor$_1$** subsystem as a stand-alone subsystem, right-click (**Control**-click for Macintosh) the warning message and select **Ignore duplication warnings for "copy for RobotMotor1"** to hide the warning message from the debugging console.

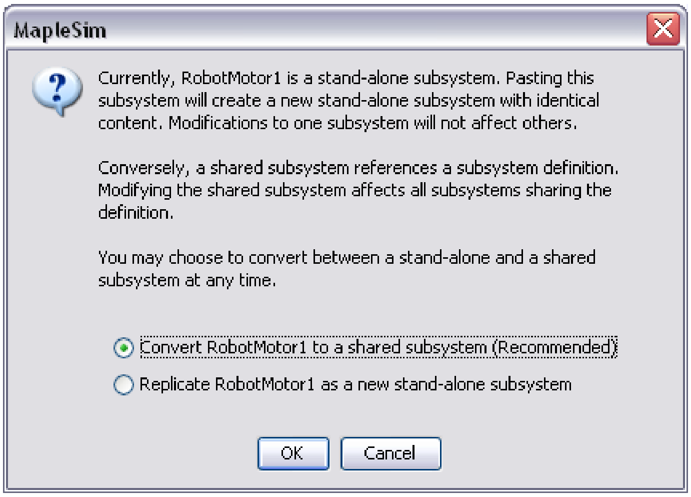
Tip: If you want to view warning messages that you hid from the debugging console, click the reset ignored warnings button ( ) below the console. All of the warning messages that you previously hid will be displayed in the debugging console again.

Alternatively, if you want to link the **copy of RobotMotor$_1$** stand-alone subsystem to the **RobotMotor** subsystem definition again, you can right-click (**Control**-click for Macintosh) the warning message and select **Update "copy of RobotMotor1" to use the shared subsystem "RobotMotor"**.

Example: Copying and Pasting a Stand-alone Subsystem

Note: This example is an extension of *Example: Removing the Link Between a Shared Subsystem and its Subsystem Definition (page 31)*.

1. In the model workspace, copy and paste the **copy of RobotMotor**$_1$ stand-alone subsystem. The following dialog box appears:


MapleSim

Currently, RobotMotor1 is a stand-alone subsystem. Pasting this subsystem will create a new stand-alone subsystem with identical content. Modifications to one subsystem will not affect others.

Conversely, a shared subsystem references a subsystem definition. Modifying the shared subsystem affects all subsystems sharing the definition.

You may choose to convert between a stand-alone and a shared subsystem at any time.

[✓ Convert RobotMotor1 to a shared subsystem (Recommended)]
Replicate RobotMotor1 as a new stand-alone subsystem

2. Select **Convert RobotMotor 1 to shared subsystem (Recommended)**. A new subsystem definition called **RobotMotor 1** is added to the **Definitions** palette.

In the model workspace, the **copy of RobotMotor**$_1$ stand-alone subsystem has been converted to a shared subsystem called **copy of RobotMotor**$_1$ and another copy of that shared subsystem called **copy of RobotMotor**$_2$ has been added to the model workspace. Both the **copy of RobotMotor**$_1$ and **copy of RobotMotor**$_2$ shared subsystems are linked to the new **RobotMotor 1** subsystem definition. Therefore, if you edit either **copy of RobotMotor**$_1$ or **copy of RobotMotor**$_2$ in the model workspace, your changes will not be reflected in subsystems that are linked to the original **RobotMotor** subsystem definition.

Note: Alternatively, you can select **Replicate RobotMotor 1 as a new stand-alone subsystem** to add another stand-alone subsystem that can be edited independently without affecting the other subsystems in the model workspace.

2.6 Global and Subsystem Parameters

Global Parameters

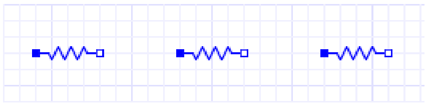
If your model contains multiple components that share a common parameter value, you can create a global parameter. A global parameter allows you to define a common parameter value in one location and then assign that common value to multiple components in your model.

The following example describes how to define and assign a global parameter. To view a more detailed example, see *Tutorial 1: Modeling a DC Motor with a Gearbox (page 107)* in Chapter 6 of this guide.

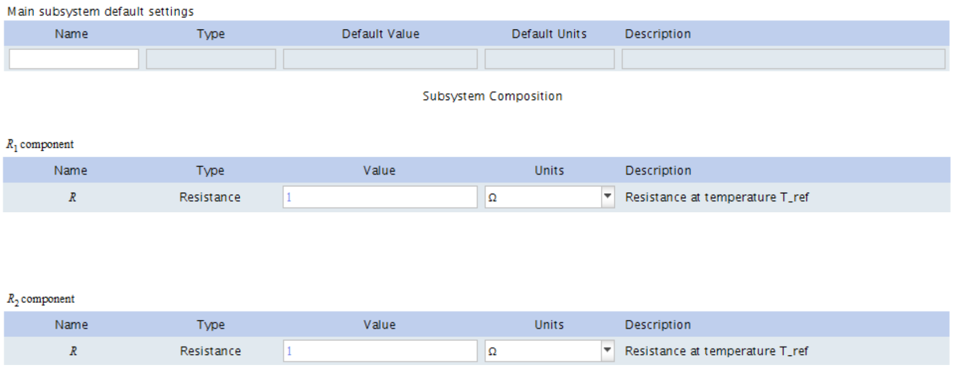
Example: Defining and Assigning a Global Parameter

If your model contains multiple **Resistor** components that have a common resistance value, you can define a global parameter for the resistance value in the parameter editor view.

1. In the **Libraries** tab, expand the **Electrical** palette, expand the **Analog** menu, expand the **Passive** menu, and then expand the **Resistors** menu.
2. From the palette, drag three copies of the **Resistor** component into the model workspace.



3. In the navigation toolbar, click the parameter editor button (), or click the workspace and from the **Inspector** tab, click **Add or Change Parameter** . The Main subsystem default settings screen appears. You will use this screen to define the global parameter and assign it to the **Resistor** components in your model.



Main subsystem default settings

Name	Type	Default Value	Default Units	Description

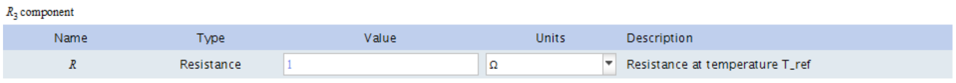
Subsystem Composition

R_1 component

Name	Type	Value	Units	Description
R	Resistance	1	Ω	Resistance at temperature T_ref

R_2 component

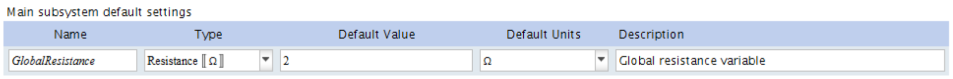
Name	Type	Value	Units	Description
R	Resistance	1	Ω	Resistance at temperature T_ref



R_3 component

Name	Type	Value	Units	Description
R	Resistance	1	Ω	Resistance at temperature T_ref

4. Click the first field in the **Main subsystem default settings** table.
5. Enter **GlobalResistance** as the global parameter name and press **Enter**.
6. Select Resistance[[Ω]] and specify a default value of 2.
7. Enter **Global resistance variable** as the description and press **Enter**.

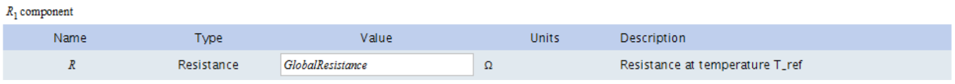


Main subsystem default settings

Name	Type	Default Value	Default Units	Description
GlobalResistance	Resistance ⟦ Ω ⟧	2	Ω	Global resistance variable

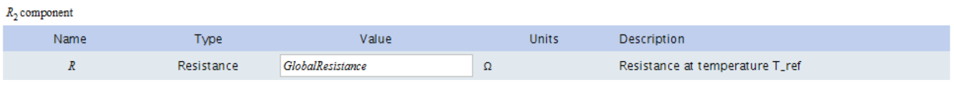
The global parameter for the resistance value is now defined. You can now assign the common **GlobalResistance** parameter value to the individual **Resistor** components that you added to the model workspace.

8. In the **R_1 component** table and **R_2 component** table, enter **GlobalResistance** as the resistance value.



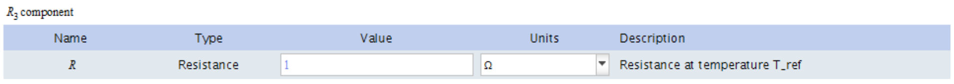
R_1 component

Name	Type	Value	Units	Description
R	Resistance	*GlobalResistance*	Ω	Resistance at temperature T_ref



R_2 component

Name	Type	Value	Units	Description
R	Resistance	*GlobalResistance*	Ω	Resistance at temperature T_ref



R_3 component

Name	Type	Value	Units	Description
R	Resistance	1	Ω	Resistance at temperature T_ref

The resistance value of the parameter **GlobalResistance** (**2**, as defined in the **Main subsystem default settings** table) has now been assigned to the resistance parameters of the **R_1** and **R_2** components.

The **R_1** and **R_2** components will now inherit any changes made to the **GlobalResistance** parameter value in the **Main subsystem default settings** table. For example, if you change the default value of the **GlobalResistance** parameter to **5** in the **Main subsystem default settings** table, the resistance parameters of the **R_1** and **R_2** components will also be changed to **5**. Any change to the **GlobalResistance** parameter value will not apply to the **R_3** component because it has not been assigned **GlobalResistance** as a parameter value.

Subsystem Parameters

You can create a subsystem parameter if you want to create a common parameter value to share with multiple components in a subsystem. Similar to global parameters, a subsystem parameter is a common value that you define in the parameter editor view and assign to components.

There are two ways to assign subsystem parameters; one is by using the parameter editor button ( ) and the other is by using the **Advanced Parameter Settings** tool in the **Inspector** tab. Parameters can only be assigned to components in the subsystem in which they are defined. If you click a subsystem in the model workspace, click the parameter editor button ( ) or **Advanced Parameter Settings**, and define a parameter in the parameter editor view, the parameter that you define is assigned to components in the subsystem that you selected and any nested subsystems.

To view an example, see *Tutorial 3: Modeling a Nonlinear Damper (page 116)* in Chapter 6 of this guide.

Note: If you create a parameter within a subsystem and assign its value to a component at the top level, the component at the top level will not inherit the parameter value.

Example: Assigning a Subsystem Parameter to a Shared Subsystem

If you assign a subsystem parameter to a shared subsystem in your model, the default subsystem parameter will also be assigned to other shared subsystems that are linked to it. However, after the default subsystem parameter is assigned, you can edit the subsystem parameter value for each shared subsystem separately without affecting other parameter values in the model.

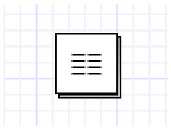
1. In the **Examples** palette, expand the **Multibody** submenu, and open the **Double Pendulum** model. This model contains two shared subsystems, $\mathbf{L_1}$ and $\mathbf{L_2}$, which are linked to a subsystem definition called **L**.
2. Click the $\mathbf{L_1}$ shared subsystem.
3. Click the parameter editor button ( ).
4. In the **L subsystem default settings table**, click the empty field at the bottom of the table.
5. Type **c** as the parameter name, keep the default value as **1**, and press **Enter**.
6. Click the diagram view button ( ). The new subsystem parameter, c, appears in the **Inspector** tab for the $\mathbf{L_1}$ shared subsystem.
7. In the top view of the model, select the $\mathbf{L_2}$ subsystem and examine the **Inspector** tab. The new subsystem parameter is also displayed for the $\mathbf{L_2}$ shared subsystem.

8. In the **Inspector** tab, change the value of **c** to **50**.
9. Click the $\mathbf{L_1}$ shared subsystem in the model workspace and examine the **Inspector** tab. Note that the value of its parameter, **c**, remains the same.

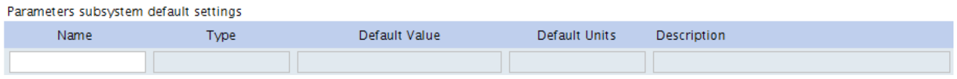
Creating Parameter Blocks

As an alternative to defining subsystem parameters using the methods described above, you can create a parameter block to define a set of subsystem parameters and assign them to components in your model. Parameter blocks allow you to apply parameters in multiple models at the top level of the model workspace.

The following image shows a parameter block that has been added to the model workspace.



When you double-click this block, the parameter editor view appears. This view allows you to define parameter values for the block.



Parameters subsystem default settings

Name	Type	Default Value	Default Units	Description

After defining parameter values, you can assign those values to the component parameters in your model.

To use parameter values in another model, you can add a parameter block to a custom library. For more information about custom libraries, see *Creating and Managing Custom Librar-ies (page 48)*.

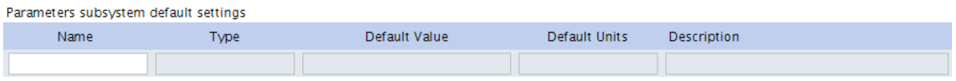
Notes:

- Parameter blocks must be placed in the same subsystem as the components to which you want to assign the parameter value.
- Parameter blocks at the same hierarchical level in a model cannot have the same parameter names. For example, two separate parameter blocks in the same subsystem cannot each contain a parameter called **mass**.

Example: Creating and Using a Parameter Block

In this example, you will create a set of parameters that can be shared by multiple components in your model. By creating a parameter block, you only need to edit parameter values in one location to compare results when you run multiple simulations.

1. In the **Libraries** tab on the left side of the MapleSim window, expand the **Examples** palette, expand the **Mechanical** menu, and then open the **PreLoad** example.
2. From the model workspace toolbar, click the parameter block button ( ) .
3. Click a blank area in the model workspace. The **Create Parameter Block** dialog box appears.
4. Click the **Inspector** tab and enter the name **SlidingMassParams** for the parameter block.
5. Double-click the **SlidingMassParams** parameter block in the model workspace. The parameter editor view appears.

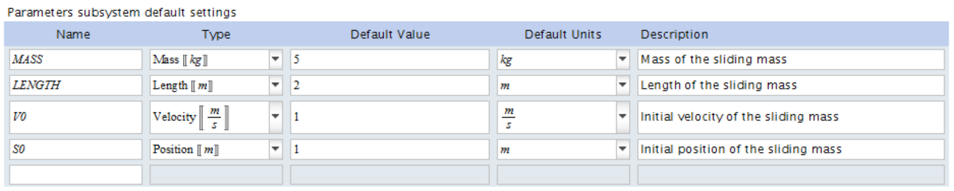

Parameters subsystem default settings

Name	Type	Default Value	Default Units	Description

6. Click the first field in the table and define a new symbolic parameter called **MASS**.
7. Press **Enter**. The remaining text fields are activated.
8. Enter a default value of **5** and the description **Mass of the sliding mass**.
9. In the same way, define the following parameters and values in the **SlidingMassParams subsystem default settings** table.

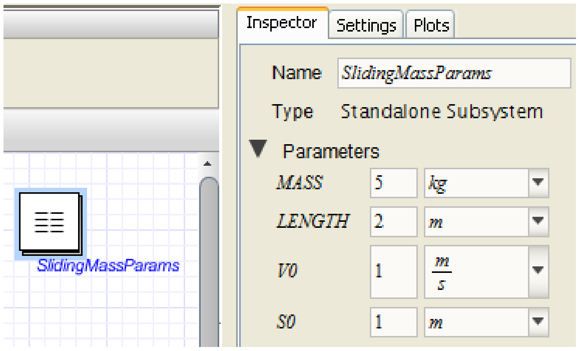
Name	**Default Value**	**Description**
LENGTH	2	Length of the sliding mass.
V0	1	Initial velocity of the sliding mass.
S0	1	Initial position of the sliding mass.

The parameter editor view appears as follows when the values are defined.


Parameters subsystem default settings

Name	Type	Default Value	Default Units	Description
MASS	Mass [[*kg*]]	5	*kg*	Mass of the sliding mass
LENGTH	Length [[*m*]]	2	*m*	Length of the sliding mass
V0	Velocity $[[\frac{m}{s}]]$	1	$\frac{m}{s}$	Initial velocity of the sliding mass
S0	Position [[*m*]]	1	*m*	Initial position of the sliding mass

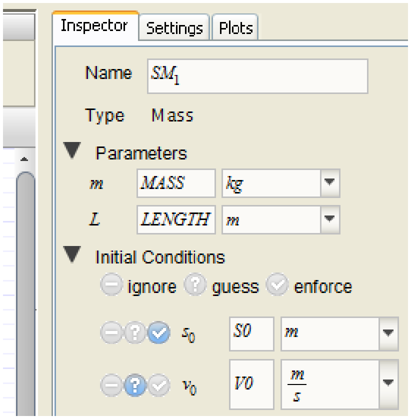
10. Click the diagram view button ( ) and then click **Main** in the navigation toolbar. When you select the parameter block in the model workspace, the defined parameters appear in the **Inspector** tab on the right side of the MapleSim window.



Name *SlidingMassParams*
Type Standalone Subsystem
Parameters

MASS	5	*kg*
LENGTH	2	*m*
V0	1	$\frac{m}{s}$
S0	1	*m*

11. In the model workspace, select one of the **Mass** components in the diagram.
12. In the **Inspector** tab, assign the following values and press **Enter**.



Name SM_1
Type Mass
Parameters

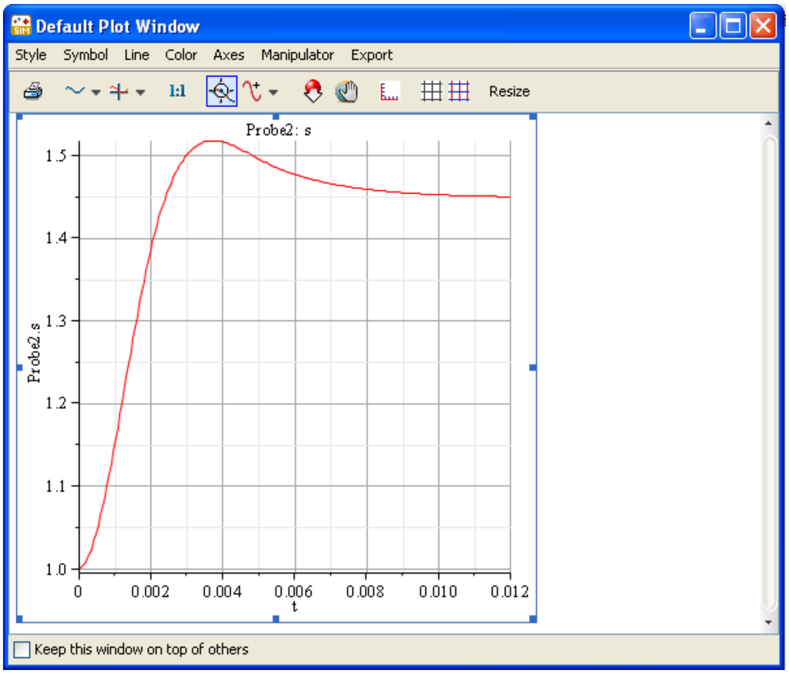
m	*MASS*	*kg*
L	*LENGTH*	*m*

Initial Conditions
ignore guess enforce

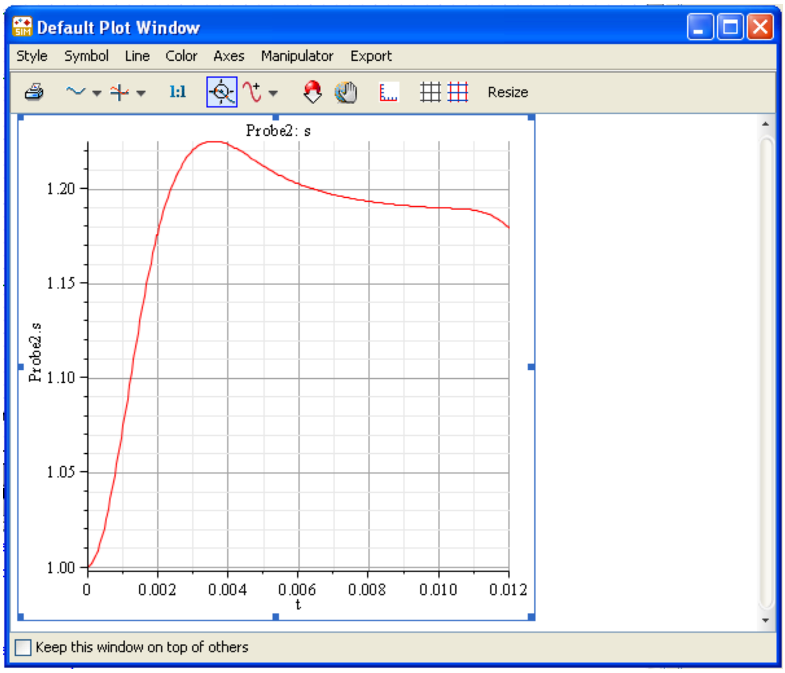
enforce	s_0	*S0*	*m*
guess	v_0	*V0*	$\frac{m}{s}$

The parameters of this **Mass** component now inherit the numeric values that you defined in the parameter block.

13. In the same way, assign the same values to the parameters of the other **Mass** components in the model.
14. In the model workspace, delete **Probe1**.
15. Select **Probe2**.
16. In the **Inspector** tab, clear the check box beside **Velocity**.
17. To simulate the model, click the simulation button (▶) in the main toolbar. The following graph appears.



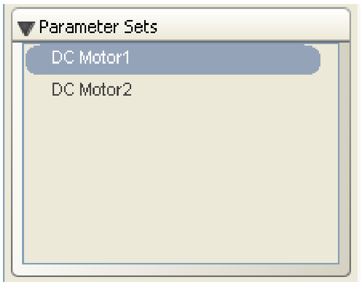
18. In the model workspace, click the parameter block.
19. In the **Inspector** tab, change the mass to **25**, the length to **10**, and the initial velocity to **5**. Press **Enter**. These changes apply to all of the **Mass** components to which you assigned the symbolic parameter values.
20. Simulate the model again. Another simulation graph appears, which you can compare to your first graph.



Creating Parameter Sets

The parameters you create for your model can be stored as reusable Parameter Sets. Parameter Sets let you save, reuse, and compare different sets of parameters for the same model displayed in the workspace. At any time you may easily apply and run different simulations, saving new values for each model. A Parameter Set provides a snapshot of all the parameters in the model workspace.

Parameter Sets for your model are listed in the **Project** tab, under Parameter Sets as shown in the following figure.



You can use, save, reuse, and compare different sets of parameters for the same model by right-clicking (**Control**-click for Macintosh) on a Parameter Set. For more information, see the **Using MapleSim → Building a Model → Storing and Applying Parameter Sets** section in the MapleSim help system.

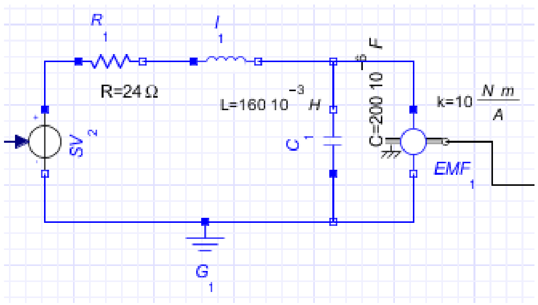
Using Advanced Parameter and Variable Settings

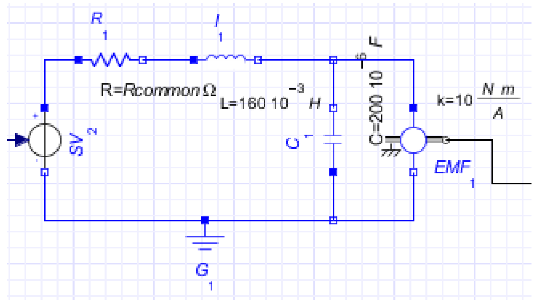
At the top level of your model, in the **Main subsystem default settings** window, you define the subsystem by adding parameters and setting their default values. An alternative is to directly assign subsystem parameters, variables and initial conditions to components in your subsystem by using the **Advanced Parameter Settings** and **Advanced Variable Settings** tool in the **Inspector** tab. Advanced Settings lets you override one or more default values.

Advanced Parameter Settings

Advanced Parameter Settings lets you override the default values for selected subsystem components. If desired, you can parametrize the override using the parametrization feature (). A component override in one subsystem can be converted to a parameter visible in all the other subsystems.

In the following model an override was applied to the initial value of R and changed to the parameter **Rcommon**.





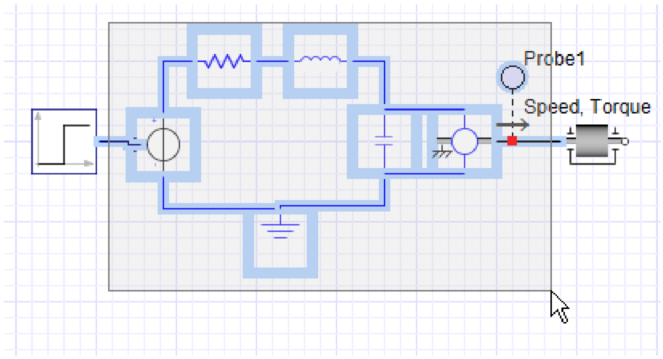
Advanced Variable Settings

Advanced Variable Settings lets you specify initial conditions for subsystem components. When you select **Advanced Variable Settings** the initial condition fields appear for all configurable components for that subsystem.

Example: Creating a Parameter Override

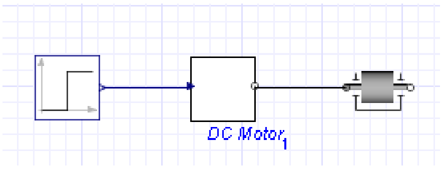
1. In the **Libraries** tab on the left side of the MapleSim window, expand the **Examples** palette, expand the **Tutorial** menu, and then open the **Simple DC Motor** example.

2. Using the selection tool (↖) located above the model workspace, draw a box around the electrical components.



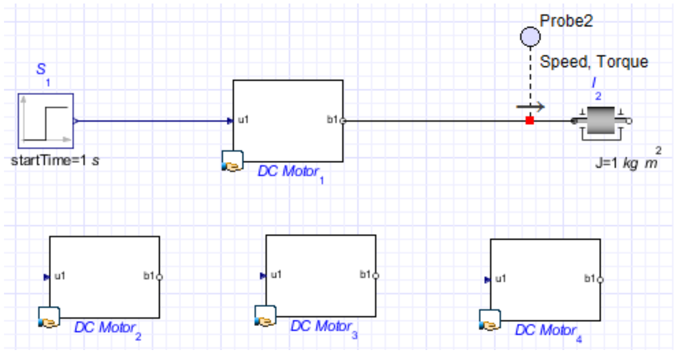
3. From the **Edit** menu, select **Create Subsystem** or right-click (**Control**-click for Macintosh) the boxed area and select **Create Subsystem**.

4. In the dialog box, enter **DC Motor and** click **OK**. The DC Motor subsystem appears.

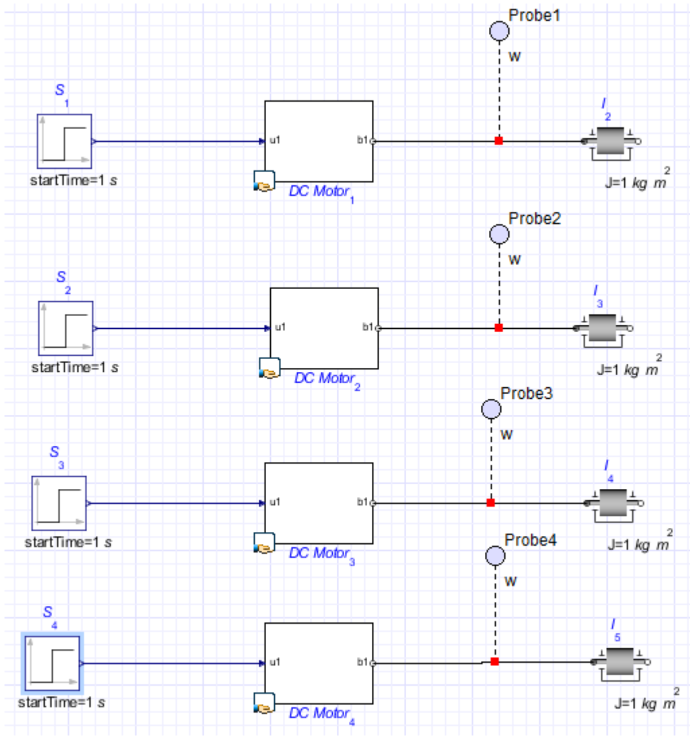


5. Use the DC Motor subsystem to create the shared subsystem definition and add it to the **Definitions** palette.

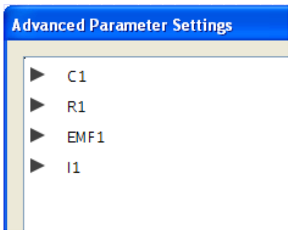
6. Add three copies of the DC motor subsystem to your model workspace by dragging the DC Motor subsystem definition icon from the **Definitions** palette and placing it in the model workspace.



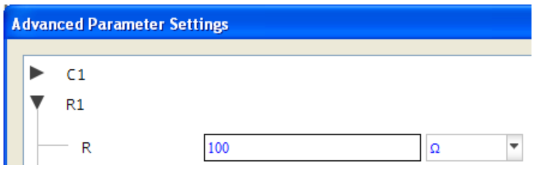
7. Create three additional DC Motor subsystems as shown below.



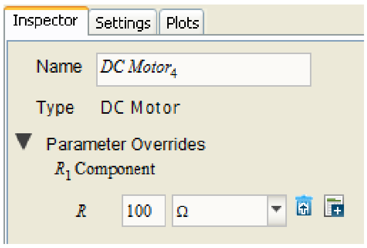
8. Click the DC Motor4 subsystem and then in the **Inspector** tab, click **Advanced Parameter Settings**. The Advanced Parameter Settings window appears showing all of the subsystem components.



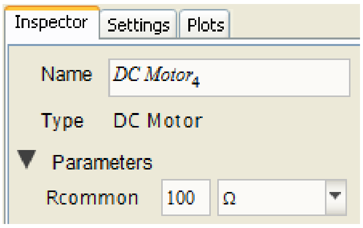
9. Expand R1 and enter a value of 100 for the Resistance parameter (R).



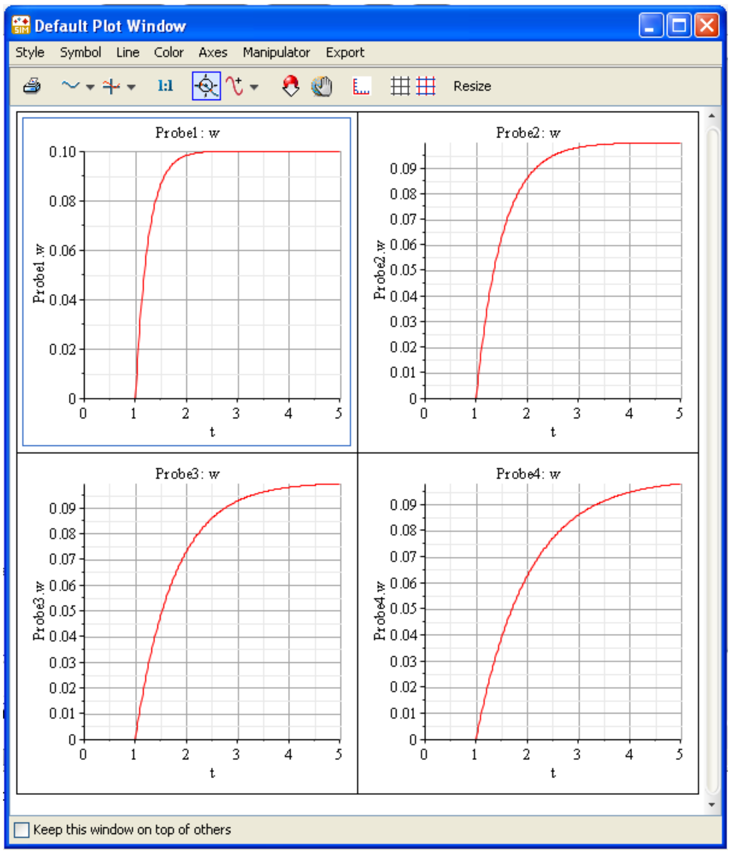
10. Click **OK**. The new parameter appears in the **Inspector** tab as an override.



11. To change this override to make it a reusable parameter click **Parametrize** (▦) and provide **Rcommon** as the new parameter name. **Rcommon** appears in the **Inspector** tab as a parameter that can be can now be reused in the other subsystems. Note that it is no longer an override.



12. For each of the other subsystems, click the subsystem and enter values of 75, 50 and 25 ohms for **Rcommon** in DC Motor subsystems 3, 2,and 1 respectively.
13. Click the simulation button (▶) in the main toolbar. The following graphs appear for each of the subsystems.



Specifying Initial Condition Overrides

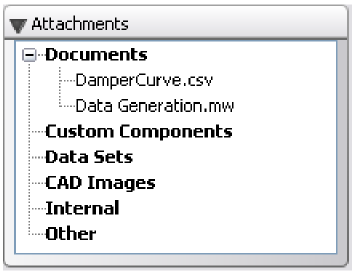
You can set initial condition values to override existing initial conditions for specific subsystem components. When you select a component, the available initial condition fields and any existing overrides appear in the **Inspector** tab, along with the other configurable parameter values for that component.

When you select a subsystem and then click **Advanced Variable Settings**, all subsystem components appear. You can select a component and specify the initial conditions for that component. This feature is especially useful for models that contain multiple shared subsystems.

2.7 Attaching Files to a Model

You can use the **Attachments** palette in the **Project** tab to attach files of any format to a model (for example, spreadsheets or design documents created in external applications). You can save files attached in the **Attachments** palette as part of the current model and refer to them when you work with that model in a future MapleSim session. To save a file, right-click (**Control**-click for Macintosh) the category in which you want to save the attachment in the palette and select **Attach File**.

The following image shows an **Attachments** palette that contains files called **DamperCurve.csv** and **Data Generation.mw**.



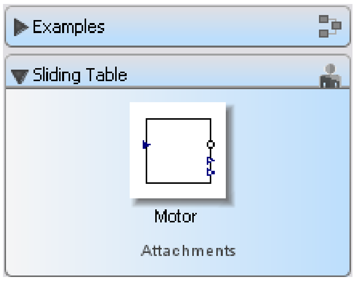
You can also use the **Attachments** palette to open MapleSim templates to perform analysis tasks in Maple, create custom modeling components, and generate data sets for a model. For more information about performing analysis tasks, see *Analyzing and Manipulating a Model (page 101)* in this guide.

2.8 Creating and Managing Custom Libraries

You can create a custom library to save a collection of subsystems, custom modeling components, or attachments that you plan to reuse in multiple files or MapleSim sessions. Custom libraries that you create appear in custom palettes below the **Examples** palette, in the **Libraries** tab, on the left side of the MapleSim window and saved as .msimlib files on your computer. These custom palettes will be displayed in the MapleSim window in future MapleSim sessions.

You can add any subsystems or custom modeling components saved in your custom library to models created in future MapleSim sessions; you can also save attachments that you want to keep with the custom library (for example, design documents for the subsystems or custom components in the custom library or other documents that you want to refer to in future MapleSim sessions).

A sample custom palette with a subsystem is shown below.



If you used a third-party tool to create models or model libraries based on the Modelica 3.1 programming language, you can import the .mo files for the models or model libraries into MapleSim as custom libraries. You can then use the imported models and libraries in your MapleSim models as you would use any other modeling components. For more information, see **Using MapleSim → Building a Model → Creating and Managing Custom Libraries → Importing Modelica Models and Libraries** in the MapleSim Help system.

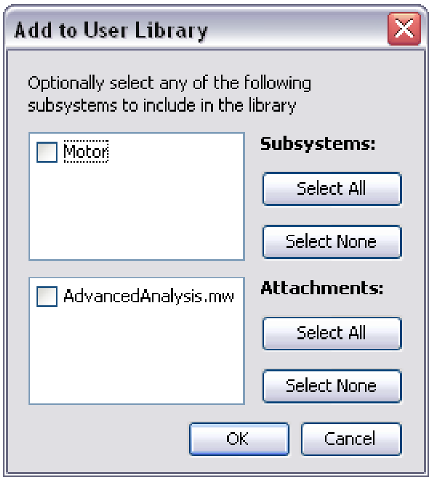
Example: Adding Subsystems and Attachments to a Custom Library

In this example, you will add a subsystem and an .mw attachment to a custom library to make them available in a future MapleSim session.

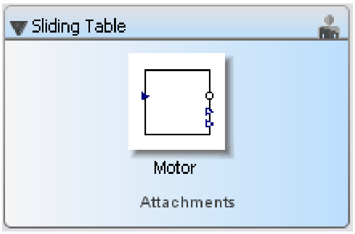
1. In the **Libraries** tab, expand the **Examples** palette, expand the **Multidomain** menu, and then open the **Sliding Table** example.
2. From the **File** menu, select **Create Library...**
3. Select a path and specify the file name **Sliding Table.msimlib**.

Note: This file will store the custom library and the file name that you specify will appear as the custom palette name in the MapleSim interface.

4. Click **Save**. The **Add to User Library** dialog box displays all of the subsystems in your model and files attached in the **Attachments** palette.

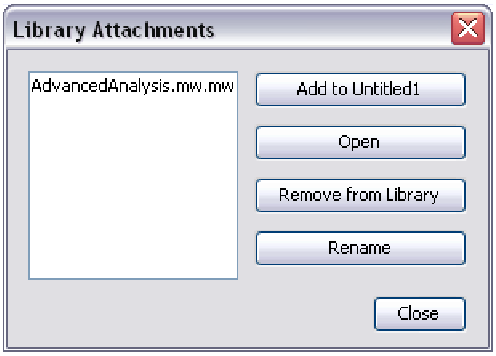


5. Select the check box beside **Motor** to add the subsystem to the custom library.
6. Select the check box beside **AdvancedAnalysis.mw** to add the attachment to the custom library.
7. Click **OK**. A new custom library palette is added in the **Libraries** tab on the left side of the MapleSim window.



This palette and its contents appear in the **Libraries** tab. They can be used in a model the next time you start MapleSim.

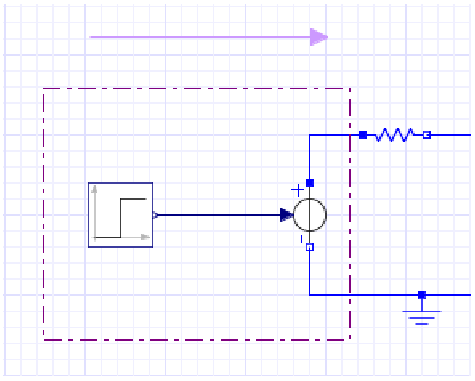
8. In the **Sliding Table** palette, click **Attachments**. The **Library Attachments** dialog box appears. This dialog box lists all of the attachments that you have added to the custom library.



You can also use this dialog box to add attachments to the **Attachments** palette of another model and open attachments in their associated applications.

2.9 Annotating a Model

You can use the tools in the model workspace toolbar to draw lines, arrows, and shapes. MapleSim also provides many tools for customizing the colors, line styles, and shape fills.

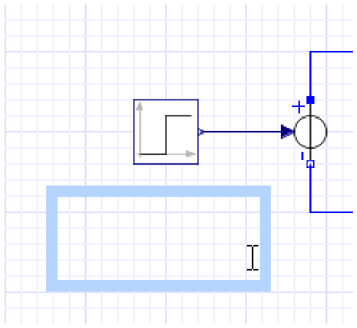


You can use the text tool (T) in the model workspace toolbar to add text annotations to your model. In text annotations, you can enter mathematical text in 2-D math notation and modify the style, color, and font of the text. For more information about 2-D math notation, see *Entering Text in 2-D Math Notation (page 53)*.

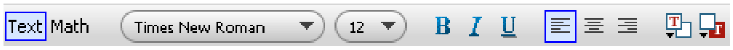
Example: Adding Text Annotation to a Model

1. In the **Libraries** tab, expand the **Examples** palette, expand the **Tutorial** menu and then open the **Simple DC Motor** example.
2. From the model workspace toolbar, click the text tool button (T).

3. In the model workspace, draw a text box for an annotation below the **Step** component.



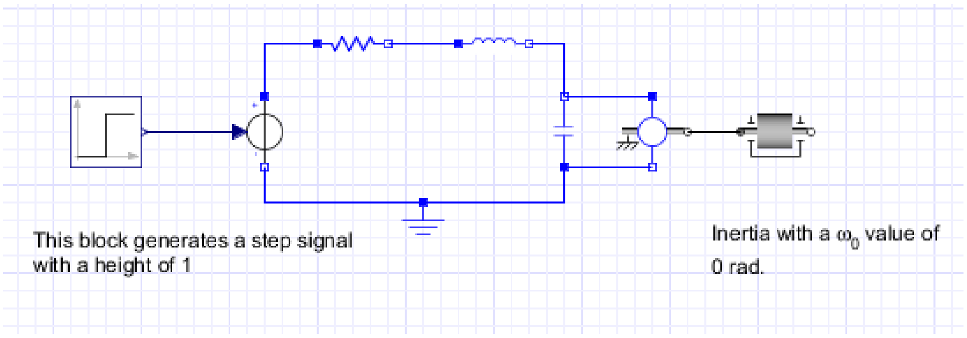
When you release your left mouse button, the toolbar above the model workspace switches to the text formatting toolbar.



4. Enter the following text: **This block generates a step signal with a height of 1.**
5. Select the text that you entered and change the font to **Arial**.
6. Click anywhere outside of the text box.
7. Draw another text box below the **Inertia** component.
8. Enter the following text: **Inertia with a ω_0 value of 0 rad.**

Tip: To enter the omega character (ω), press **F5** to switch to the 2-D math mode, type **omega**, and then press **Ctrl** + **Space** (Windows®), **Ctrl** + **Shift** + **Space** (Linux®), or **Esc** (Macintosh). To enter the subscript, press **Ctrl** + **Shift** + the **underscore** key (Windows and Linux) or **Command** + **Shift** + the **underscore** key (Macintosh) followed by **0**. Press the right arrow key to move the cursor from the subscript position.

9. Click anywhere outside of the text box.
10. Select the text that you entered and change the font to **Arial**.
11. Click anywhere outside of the text box to complete the annotation.


This block generates a step signal with a height of 1
Inertia with a ω_0 value of 0 rad.

2.10 Entering Text in 2-D Math Notation

In parameter values and annotations, you can enter text in 2-D math notation, which is a formatting option for adding mathematical elements such as subscripts, superscripts, and Greek characters. As you enter text in 2-D math notation, you can use the command and symbol completion feature to display a list of possible Maple commands or mathematical symbols that you can insert.

The following table lists common key combinations for 2-D math notation:

Task	Key Combination	Example
Switch between text and 2-D math mode (annotations only)	**F5**	-
Command and symbol completion (parameter values and annotations only)	1. Enter the first few characters of a symbol name, Greek character, or Maple command. 2. Enter the key combination for your platform: • **Ctrl** + **Space** (Windows) • **Ctrl** + **Shift** + **Space** (Linux) • **Esc** (Macintosh) 3. From the menu, select the symbol or command that you want to insert.	-
Enter a subscript for a variable	**Ctrl** (or **Command**) + **Shift** + **underscore** (_)	x_a
Enter a superscript	**caret** (^)	x^2
Enter a square root (annotations only)	Enter **sqrt** and press **Ctrl** (or **Command** for Macintosh) + **Space**.	$\sqrt{x}$

Task	Key Combination	Example
Enter a root (annotations only)	Enter **nthroot** and press **Ctrl** (or **Command**) + **Space**.	$\sqrt[n]{x}$
Enter a fraction	**forward slash** (/)	$\frac{1}{8}$
Enter a piecewise, matrix, or vector row (annotations only)	**Ctrl** (or **Command**) + **Shift** + **R**	$\begin{bmatrix} 3 & 8 \end{bmatrix}$
Enter a table column (annotations only)	**Ctrl** (or **Command**) + **Shift** + **C**	$\begin{bmatrix} 6 \end{bmatrix}$

For more information, see the **Using MapleSim → Building a Model → Annotating a Model → Key Combinations for 2-D Math Notation** topic in the MapleSim Help system.

2.11 Creating a Data Set for an Interpolation Table Component

You can create a data set to provide values for an interpolation table component in your model. For example, you can provide custom values for input signals, and electrical **Current Table** and **Voltage Table** sources. To create a data set, you can either attach a Microsoft® Excel® spreadsheet (.xls or .xlsx) or comma-separated values (.csv) file that contains the custom values, or you can create a data set in Maple using the Data Generation Template or Random Data Template provided in the MapleSim templates dialog box.

For more information about interpolation table components, see the **MapleSim Component Library →Signal Blocks → Interpolation Tables → Overview** topic in the MapleSim Help system.

Example: Creating a Data Set in Maple

In this example, you will use the Data Generation Template to create a data set for a MapleSim **1D Lookup Table** component. In this template, you can use any Maple commands to create a data set; however, for demonstration purposes, you will create a data set using a computation that has already been defined.

1. Open a new MapleSim document.
2. In the **Libraries** tab, expand the **Signal Blocks** palette, and then expand the **Interpolation Tables** menu.
3. Add a **1D Lookup Table** component to the model workspace.
4. In the main toolbar, click the templates button ( ).
5. From the templates list, select **Data Generation**.

6. In the **Attachment** field, enter **My First Data Set** and click **Create Attachment**. The Data Generation Template is opened in Maple.
7. To execute the entire worksheet, click !!! at the top of the Maple window.
8. At the bottom of the template, in the **Data set name** field, enter **TestDataSet**.
9. To make the data set available in MapleSim, click the **Attach Data in MapleSim** button.
10. In MapleSim, in the **Project** tab, expand the **Attachments** palette, and then expand the **Data Sets** category. The data set file appears in the list.

You can now assign this data set to the interpolation table component in the model workspace.

11. In the model workspace, select the **1D Lookup Table** component.
12. In the **Inspector** tab, from the **data** drop-down menu, select the **TestDataSet.mpld** file. The data set is now assigned to the **1D Lookup Table** component.
13. Save the Data Generation Template in Maple and then save your model in MapleSim.

2.12 Best Practices: Building a Model

This section describes best practices to consider when laying out and building a MapleSim model.

Best Practices: Laying Out and Creating Subsystems

To start building your model, drag components from the palettes to the center of the model workspace. Drag the components into the arrangement that you want in the model workspace and then, if necessary, change their orientation so that the components are facing in the direction that you want. When you have established the position and orientation of the components, connect them in the model workspace.

When grouping components into subsystems, make sure that you include logical component groups that fit on one screen at a time. This will allow you to see all of the subsystem components at a certain level without scrolling.

Create Subsystems for Component Groups That You Plan to Reuse

Create subsystems for component groups that you plan to reuse throughout a diagram or in multiple files. For example, if you plan to include multiple planar link models in a pendulum system, you can create a link subsystem so that multiple copies of that component group could be added. If you wanted to add the link subsystem to another pendulum model, you can create a custom library to use the subsystem in another file.

Create Subsystems for Component Groups That You Plan to Analyze

Make sure that you create subsystems for component groups that you plan to analyze in more depth, test, or translate into source code. Several MapleSim templates allow you to analyze and retrieve equations from particular subsystems. The Code Generation Template allows you to generate source code from subsystems only.

For more information about performing analysis tasks, see *Analyzing and Manipulating a Model (page 101)* in this guide.

Use the Debugging Console to Identify Subsystem Copies and Unconnected Lines

You can display the debugging pane by clicking the debugging button ( ) at the bottom of the MapleSim window.

The debugging pane displays diagnostic messages that can help you troubleshoot potential errors as you build a model. When you click the diagnostic tests button (✓) below the debugging pane (or from the **File** menu, select **Check Model**), MapleSim verifies whether your model contains unconnected lines or subsystems that have identical content but are not linked to a subsystem definition. When either of these issues are detected, a message that identifies the subsystem in which the issue is located appears in the debugging console. You can right-click (**Control**-click for Macintosh) the message in the debugging pane to display options that can help you to resolve the issue.

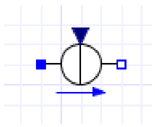
Best Practices: Building Electrical Models

Include a Ground Component in Electrical Circuits

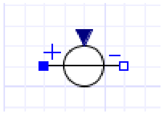
In each electrical circuit model, you must add and connect a **Ground** component to provide a reference for the voltage signals.

Verify the Connections of Current and Voltage Sources

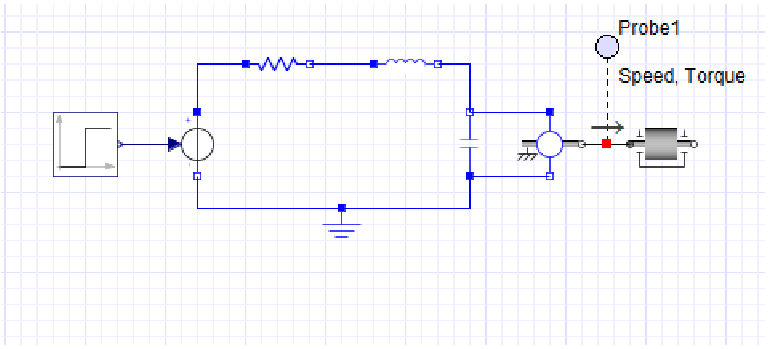
Simulation results can be affected by the way in which a current or voltage source is connected in your model. If you receive unexpected simulation results, verify the connections between electrical sources and other components in your model. All of the current sources in the MapleSim Component Library display an arrow that indicates the direction of the positive current.



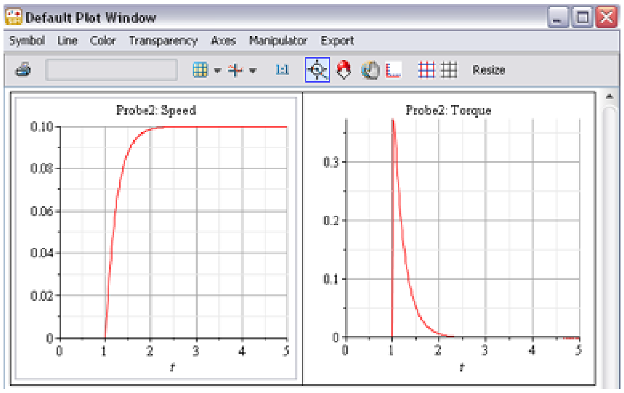
Also, all of the voltage sources display a plus sign indicating the location of the positive voltage and a minus sign indicating the location of the negative voltage.



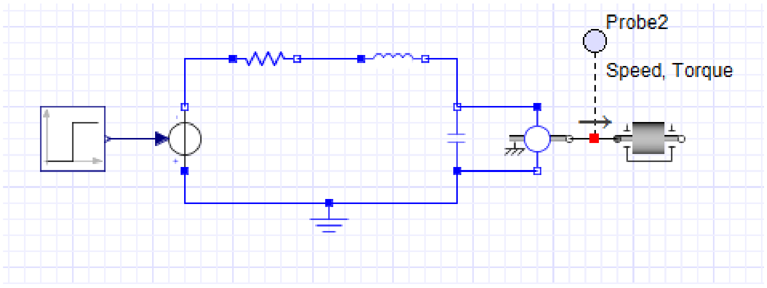
Consider the following DC motor model. Note that the positive port of the **Signal Voltage** source at the left of the diagram is connected to the positive port of the **Resistor** component.



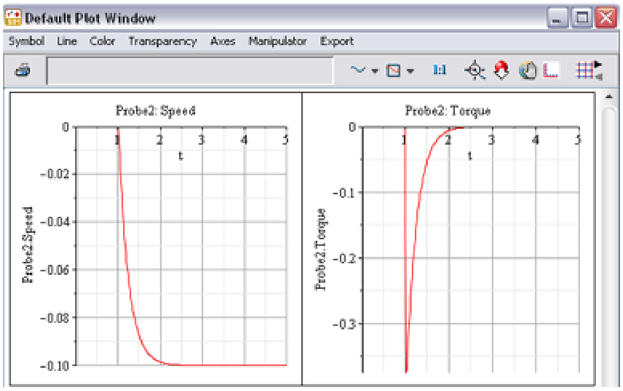
When this model is simulated, MapleSim returns the following results for the speed and torque quantities.



On the other hand, if the negative port of the **Signal Voltage** source is connected to the positive port of the **Resistor** component, as shown in the following model.



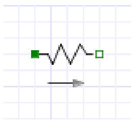
MapleSim returns different results for the speed and torque quantities.



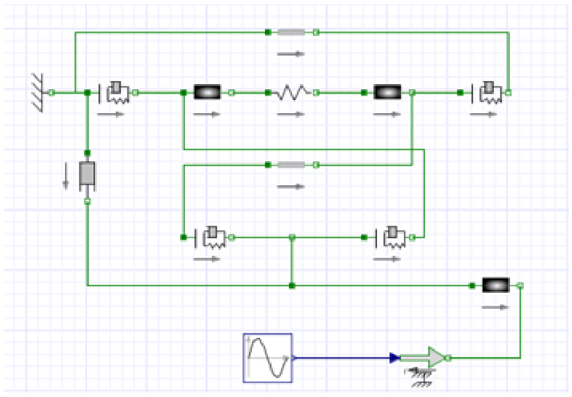
Best Practices: Building 1-D Translational Models

Verify That All Force Arrows Are Pointed in the Same Direction

In MapleSim, all of the 1-D translational mechanical components are defined in a 1-D co-ordinate system with the positive direction defined as the direction of the gray arrow displayed by the component icon.



Any positive forces acting on the model cause the component to move in the direction of the arrow, so make sure that all of the arrows displayed by the 1-D translational mechanical components in your model point in the same direction. As an example, note that all of the force arrows are pointed to the right in the following model.

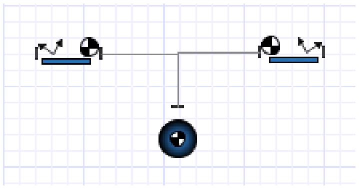


Best Practices: Building Multibody Models

Connect the Inboard Port of a Rigid Body Frame to a Center-of-mass Frame

Make sure that you connect the inboard port of any **Rigid Body Frame** components in your model to the center-of-mass frame of a **Rigid Body** component. This ensures that the local reference frame used to describe displacements and rotations for the **Rigid Body Frame** component match with the center-of-mass reference frame defined on the **Rigid Body** component.

In the following planar link example, the **Rigid Body Frame** inboard ports (that is, the ports with the cross-hatched circles) are both connected to a **Rigid Body** component.

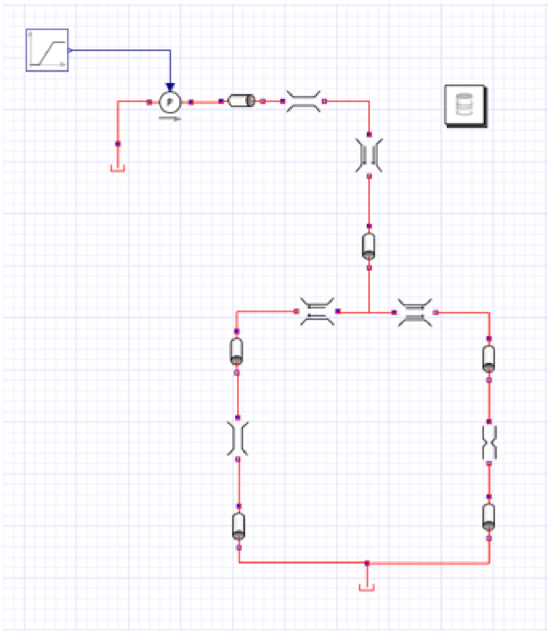


Best Practices: Building Hydraulic Models

Define Fluid Properties

When building hydraulic models, you must define the properties of the fluid that will be used by placing the **Hydraulic Fluid Properties** component at the top level of your model or at same level as a hydraulic subsystem. If you place this component at the top level of your model, all hydraulic components and subsystems in your model will inherit the fluid properties defined by that component instance; if you place the **Hydraulic Fluid Properties** component at the same level as a subsystem, all hydraulic components in that subsystem and all nested subsystems will inherit the properties defined by that component instance.

In the following example, all of the hydraulic components in the model inherit the fluid properties defined by the **Hydraulic Fluid Properties** component at the top-right of the diagram.



Best Practices: Enforcing Initial Conditions

In complex models, all of the initial conditions might not be independent of each other. In general, use the **enforce** (✓) option to strictly enforce as many initial conditions as you have degrees of freedom in your model. However, you can use the **guess** option (?) for a specified initial condition parameter value to help the solver determine the desired starting configuration for your system faster.

3 Creating Custom Modeling Components

In this chapter:

3.1 Overview

To extend the MapleSim component library, you can create custom modeling components based on mathematical models that you define. For example, you can create a custom component to contain a particular subsystem and to provide specialized functionality.

By using the Custom Component Template, which is a Maple worksheet available through the MapleSim templates dialog box, you perform the following tasks in Maple to create a custom component:

- Define the component equations and properties that determine the behavior of the component (for example, parameters and port variables)
- Test and analyze your mathematical model
- Add ports to the component and define the associated port variable mappings
- Generate the component and make it available in MapleSim

The Custom Component Template contains pre-built controls that allow you to perform these tasks. Each generated custom component is associated with a particular template.

3.2 Opening Custom Component Examples

The following custom component examples are available with your MapleSim installation:

- Custom component defined with an algebraic equation
- A sample DC motor component defined with a differential equation
- A sample nonlinear spring-damper component
- Custom component defined with a transfer function

To open an example:

1. In MapleSim, click the templates button ( ) in the main toolbar.

2. Click **Browse...**

3. In the dialog box, open the **Component Templates** folder.

4. Select the example that you want to open, and click **Use Template**

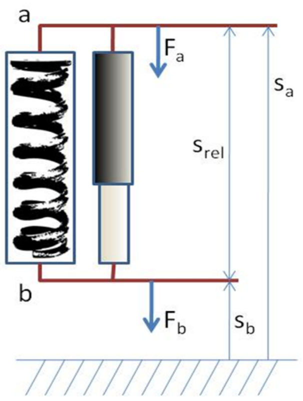
5. (Optional) In the **Attachment** field, enter a name for the template.

6. Click **Create Attachment**. The sample Custom Component Template is opened in Maple.

3.3 Example: Nonlinear Spring-Damper Component

In this example, you will use the Custom Component Template to create a nonlinear spring-damper custom component. The equations defined in this example are based on the **Translational Spring Damper** component in MapleSim. In this case, the stiffness and damping coefficients are replaced with input functions to the component.

To obtain the governing relationships, you can start with a free-body diagram. The diagram for the spring-damper system is shown below.



The end points, *a* and *b*, can be defined as the ports for the component; the equations are derived relative to these ports. Therefore, the general equation of motion,

$$d \cdot \frac{\mathrm{d}}{\mathrm{d}t} s_{rel}(t) + c \cdot s_{rel}(t) = F(t)$$

where *d* is the damping coefficient, *c* is the stiffness of the spring, and s_{rel} is the relative displacement between the two ports s_a and s_b, can be written as

$$s_{rel}(t) = s_b(t) - s_a(t)$$

Also, an examination of the net force on the system shows that $F(t) = F_b(t)$, where

$$F_a(t) + F_b(t) = 0$$

All of the above relationships are required to define the system behavior.

Opening the Custom Component Template

To start, open the Custom Component Template from the MapleSim templates dialog box.

1. In MapleSim, open the model to which you want to add the custom component.
2. Click the templates button ( ) in the main toolbar.
3. In the **Select Template** list, select **Custom Component**.
4. In the **Attachment** field, enter **Nonlinear Spring-Damper** as the name for the template and click **Create Attachment**. The Custom Component Template is opened in Maple.

Defining the Component Name and Equations

You can now specify the name that will be displayed for the component in the MapleSim interface, a variable to store the equations, and the equations. To define the component equations, you create a system model by using commands from the **DynamicSystems** package. For more information, see the **DynamicSystems** topic in the Maple help system.

1. In the **Component Description** section of the template, specify a component name called **NonLinearSpringDamper**.
2. In the **Component Equations** section, delete the default equations below the table that defines the variables.
3. To define the nonlinear system, enter the following equations.

> $eq := [d(t) * (diff(s[rel](t), t)) + c(t) * s[rel](t) = F(t),$
> $\quad s[rel](t) = s[b](t) - s[a](t), v[rel](t) = diff(s[rel](t), t),$
> $\quad F(t) = F[b](t), F[a](t) + F[b](t) = 0];$

> $params := [\,]:$

> $initialconditions := [\,]:$

Note that the equations are entered in a Maple list. The constants, *d* (damping) and *c* (stiffness) are replaced by the functions *d*(*t*) and *c*(*t*) to define them as input states to the system.

4. To assign the equations and the input and output definitions to a system object variable called *sys*, enter the following text.

> $sys := DynamicSystems[DiffEquation](\,eq, inputvariable$
> $= [F[a](t), F[b](t), c(t), d(t)], outputvariable$
> $= [s[a](t), s[b](t)])\ :$

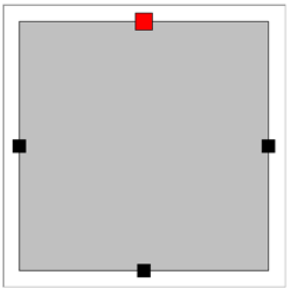
5. Click !!! at the top of the window to execute the entire worksheet.

You can now assign these input and output variables to ports that you will include in your generated custom component.

Defining Component Ports

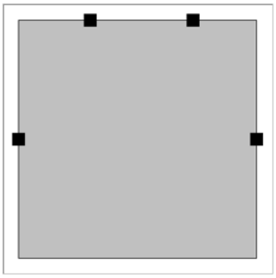
In the **Component Ports** section of the template, you assign input and output variables to ports that will appear in the generated component, and specify the layout of these ports.

1. To remove the sample ports from the diagram, click **Clear All Ports**.
2. Click the **Add Port** button four times. Four squares, which represent the ports that you will lay out and define, appear in the diagram.



3. Select the port on the left side of the diagram.
4. From the **Port Type** drop-down menu below the diagram, select **Translational Flange**.
5. In the **Port Components** table, in the **Position** row, select **s[b](t)** from the drop-down menu and, in the **Force** row, select **F[b](t)** from the drop-down menu. The left port is now defined as a translational flange and associated with the position variable **s[b](t)** and force variable **F[b](t)**.
6. Select the port on the right side of the diagram.
7. From the **Port Type** drop-down menu, select **Translational Flange**.
8. In the **Position** row, select **s[a](t)** from the drop-down menu and, in the **Force** row, select **F[a](t)** from the drop-down menu. The right port is now defined as a translational flange and associated with the position variable **s[a](t)** and force variable **F[a](t)**.

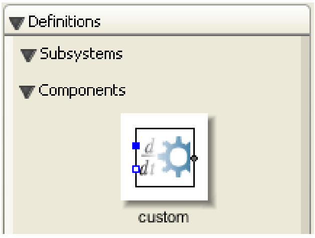
9. Select the port at the top of the diagram.
10. From the **Port Type** drop-down menu, select **Signal Input**.
11. In the **Value** row, select **c(t)** from the drop-down menu. This port is now defined as a signal input and associated with the stiffness variable **c(t)**.
12. Select the port at the bottom of the diagram.
13. From the **Port Type** drop-down menu, select **Signal Input**.
14. In the **Value** row, select **d(t)** from the drop-down menu. This port is now defined as a signal input and associated with the damping variable **d(t)**.
15. Drag the port that you defined in step 14 and place it at the top right of the diagram. You can also drag the other port to position it.



The ports will be displayed in this arrangement when you generate the custom component in MapleSim.

Generating the Custom Component

To generate the custom component, click the **Generate MapleSim Component** button at the bottom of the template. To find the generated custom component in MapleSim, select the **Project** tab, expand the **Definitions** palette, and then expand the **Components** submenu.



You can now change the name to **NonLinearSpringDamper** and add the custom component to a model by dragging it into the model workspace.

3.4 Working with Custom Components in MapleSim

In MapleSim, you can work with a custom component in the same ways as you would work with a subsystem. You can perform the following tasks:

Add Text and Illustrations to a Custom Component

To customize the appearance of a custom component, you can change the default images in the component icon. Select the custom component in the model workspace, click the icon view button ( ) in the navigation toolbar, and use the drawing and annotation tools to add text and illustrations.

Save a Custom Component as Part of the Current Model

To save a custom component as a part of the current model, add the component by dragging it into the model workspace and then save the model. The next time you open the file, the custom component will be displayed in the model workspace and **Definitions** palette.

Add a Custom Component to a Custom Library

If you want to use a custom component in a file other than the current model, add the component to a custom library. For more information, see *Creating and Managing Custom Libraries (page 48)*.

3.5 Editing a Custom Component

If you want to edit a custom component that you have generated, make your changes in the corresponding Maple worksheet and regenerate the component.

1. In the MapleSim model workspace, double-click the custom component that you want to edit. The corresponding Custom Component Template is opened in Maple.
2. In the Maple worksheet, edit the equations, properties, or port values.
3. At the bottom of the worksheet, click **Generate MapleSim Component**. Your changes are generated in the custom component displayed in MapleSim.
4. Save your changes in the .mw file and the .msim file to which you added the custom component.

4 Simulating and Visualizing a Model

In this chapter:

4.1 How MapleSim Simulates a Model

Modelica Description

The equations for many components in the MapleSim library are described using the Modelica physical modeling language. On the other hand, the equations for multibody components are generated by a special-purpose engine, which uses advanced mathematical techniques to ensure that the equations are as concise and efficient as possible, and then converted to Modelica.

For more information about Modelica, visit **http://www.modelica.org**.

Model Description

Each component in your model contains a system of equations that describes its behavior; these systems of equations can consist of purely algebraic equations or differential equations. Also, a component may define any number of events, which can change the component behavior during a simulation by enabling or disabling part of the equations in the system or changing state values. Connections between two or more components generate additional equations that describe how these components interact.

System Equations

All of these equations are then collected in one large system and parameter values are also substituted in. Now, the MapleSim simulation engine has a potentially large system of hybrid differential algebraic equations. This means that the system has differential equations with algebraic constraints, as well as discrete events.

Simplified Equations

A process called *index reduction* is applied to reduce the algebraic constraints as much as possible. Other symbolic simplification techniques also reduce the number of equations and variables. Note that algebraic constraints may still be present in the equations after this step. No information is lost during the simplification process and the full accuracy is preserved. At this point, initial values for all of the variables remaining in the system of equations must be computed. This is a non-trivial step because typically only a small number of the initial conditions is fixed in the system model. The remainder of the initial conditions must be computed in such a way that the entire equation set is consistent.

You can set initial values for some of the variables by specifying parameter values for certain components in the **Inspector** tab on the right side of the MapleSim window. If the specified initial conditions are not consistent, an error will be detected during the simulation.

Initialization

When all of these preprocessing steps are complete, the numeric solving process can begin. A sophisticated differential algebraic equation (DAE) solver based on the Rosenbrock integrator (for stiff systems), the ck45 integrator (for semi-stiff systems), and rkf45 integrator (for non-stiff systems) is used to numerically integrate the system of equations. Algebraic constraints are constantly monitored to avoid constraint drift, which would otherwise affect the solution accuracy. The **rosenbrock (stiff)** solver is a good choice for typical systems. In some cases, the non-stiff solvers will offer better performance; they are a good option for models where all quantities vary at approximately the same rate.

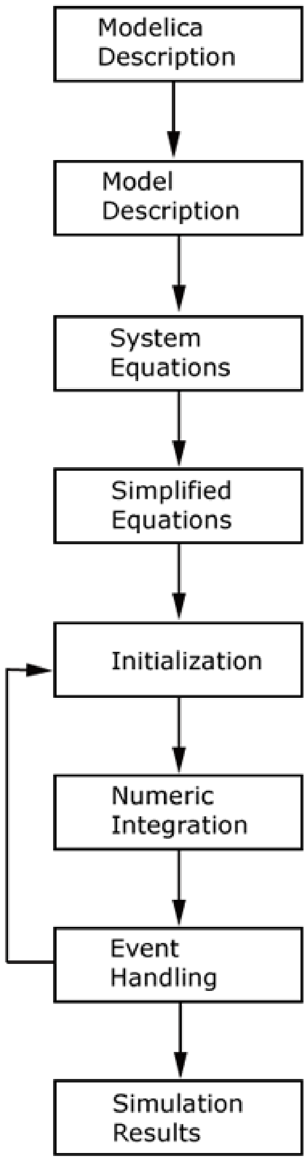
Numeric Integration and Event Handling

During numeric solving (or "integration"), inequality conditions that are part of the model are monitored and an event is triggered when one or more of these conditions change. Whenever such an event is encountered, the numeric solver is stopped and the simulation engine computes a new configuration of the system of equations based on the event conditions. This step also involves recomputing initial conditions for the new system configuration. The solver is then restarted and continues to numerically solve the system until another event is triggered or the simulation end time is reached.

Simulation Results

In the last step of the simulation process, the results are generated and displayed using graphs showing the quantities of interest and, optionally for multibody mechanical systems, a 3-D animation.

The simulation process is summarized in the following chart:

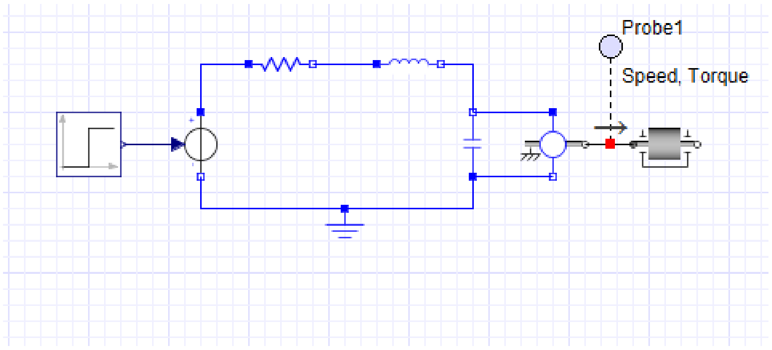


Note that the information in this section is a simplified description of the simulation process. For more information on the DAE solvers used by the simulation engine, see the **dsolve,numeric** topic in the Maple help system.

4.2 Simulating a Model

To view the behavior or response of physical properties (for example, current or voltage), add probes to connection lines, ports, or components in your 2-D or 3-D model. In MapleSim, probes allow you to identify the variables of interest that are associated with connection ports.

If you add a probe to measure a through variable, an arrow appears to indicate the direction of the positive flow in the model workspace.



You can specify the simulation duration, the type of solver to use, and other parameter values for the solver, simulation engine, and 3-D workspace. After running a simulation, a graph appears for each specified quantity.

You can change the original probe or parameter values and run another simulation to compare the results.

Simulation Settings

At the top level of your model, in the **Settings** tab, you must first specify the simulation duration, simulation engine type and any other specific model settings for the Solver.

For a description of the Multibody and 3-D Visualization settings see *3-D Visualization Settings (page 80)*.

Solver

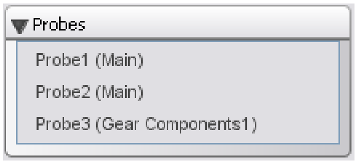
You can specify the type of integration and computational methods, and the initialization settings for your model.

Parameter	Default	Description
t_f	10	End time of the simulation. You can specify any positive value, including floating-point values. **Note:** For all simulations, the initial start time is 0.
solver	ck45 (semi-stiff)	DAE solver used during the simulation. • **ck45 (semi-stiff):** use a semi-stiff DAE solver (ck45 method). • **rkf45 (non-stiff):** use a non-stiff DAE solver (rkf45 method). • **rosenbrock (stiff):** use a stiff DAE solver (Rosenbrock method). If your model is complex, you may want to use a stiff DAE solver to reduce the time required to simulate a model.
adaptive	true	Specifies whether an adaptive solver or a fixed-step solver is used to determine sampling periods for the simulation. • **true**: use an adaptive solver. The sampling periods, as determined by the solver, vary throughout the simulation. • **false**: use a fixed-step solver. The sampling periods are a uniform step size throughout the simulation. You can specify the size in the **step size** field. If the state of your model changes rapidly, you may want to use a fixed-step solver to reduce the time required to run the simulation. **Note:** When a fixed-step solver is used, fewer sampling periods may be represented in the simulation results. For the most accurate results, use an adaptive solver to run the simulation.
step size	0.0010	Uniform size of the sampling periods if you are using a fixed-step solver to run the simulation. You can specify a floating-point value for this option when the **adaptive** field is set to **false**.
ϵ_{abs}	$1 \cdot 10^{-7}$	The limit on the absolute error tolerance for a successful integration step if you are using an adaptive solver to run the simulation. You can specify a floating-point value for this option when the **adaptive** field is set to **true**.
ϵ_{rel}	$1 \cdot 10^{-7}$	The limit on the relative error tolerance for a successful integration step if you are using an adaptive solver to run the simulation. You can specify a floating-point value for this option when the **adaptive** field is set to **true**.

Parameter	Default	Description
plot points	200	Minimum number of points to be plotted in a simulation graph. The data points are distributed evenly in the graph according to the simulation duration value. You can specify a positive integer. **Note:** This option allows you to specify the number of points for display purposes only. The actual number of points used during the simulation may differ from the number of points displayed in the simulation graph.
compiler	false	Specifies whether a native C compiler is used during the simulation. When this option is set to **true**, Maple procedures generated by the simulation engine are translated to C code, which is compiled by an external C compiler. If your model is complex, you may want to set this option to **true** to reduce the time required to run a simulation.
engine	Mark II	Specifies the formulation strategy (that is, the engine) used for system simplification, initialization, and simulation. • **Mark I:** the original engine used in MapleSim 4 and earlier. • **Mark II:** a new engine that generally reduces formulation and simulation times. This option is recommended by Maplesoft as of MapleSim 4.5

Editing Probe Values

In the **Project** tab, the **Probes** palette lists all of the probes that you have added to the current MapleSim model.



If a probe is attached at the top level of your model, **Main** appears in parentheses beside the probe name; otherwise, the subsystem for the attached probe appears beside the probe name. In the image shown above, three probes have been attached to a model: **Probe1** and **Probe2** at the top level of the model and **Probe3** in a subsystem called **Gear Components1**.

You can click the entries in this palette to browse to a probe in the model workspace, and view and edit the probe values in the **Inspector** tab. You can also right-click (**Control**-click for Macintosh) entries in this palette and manipulate probes using context menus.

For more information, see **Using MapleSim → Simulating a Model →Editing Probe Values** in the MapleSim help system.

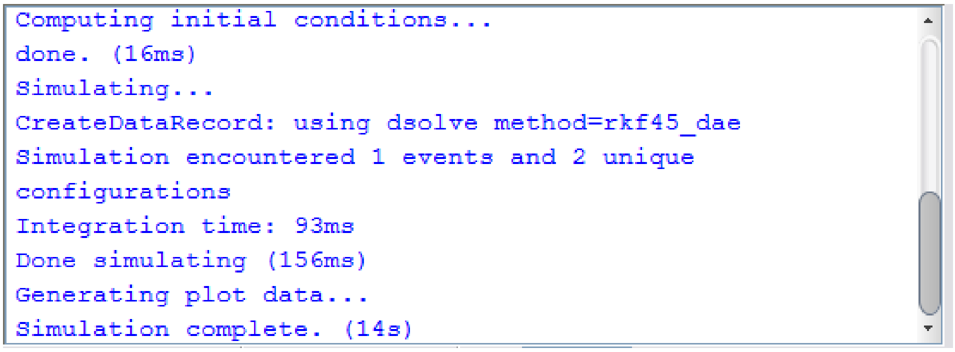
Storing Parameter Sets to Compare Simulation Results

You can store a group of parameter values that are assigned to a model in a parameter set. You can then run a simulation using one parameter set, replace those parameter values with another parameter set, and run another simulation to compare the results.

For more information, see the **Using MapleSim → Building a Model → Storing and Applying Parameter Sets** section in the MapleSim help system.

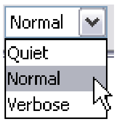
4.3 Simulation Progress Messages

During a simulation, you can view progress messages in the **Console** pane located below the model workspace. These messages indicate the status of the MapleSim engine as it generates a mathematical model; these messages can help you to debug simulation errors.



```
Computing initial conditions...
done. (16ms)
Simulating...
CreateDataRecord: using dsolve method=rkf45_dae
Simulation encountered 1 events and 2 unique
configurations
Integration time: 93ms
Done simulating (156ms)
Generating plot data...
Simulation complete. (14s)
```

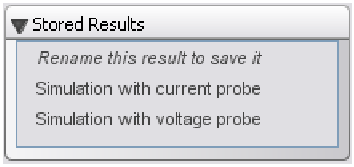
Optionally, before running a simulation, you can specify the amount of detail displayed in progress messages by clicking the message console button ( ) at the bottom of the MapleSim window and selecting a level from the drop-down menu.



4.4 Managing Simulation Results

The **Stored Results** palette in the **Project** tab allows you to view, save, and export results generated from multiple simulations. Whenever you simulate a model, a **Rename this result to save it** entry is added to the **Stored Results** palette. You can click this entry to view the graphs, progress messages, and (if applicable) 3-D animation generated from the most recent simulation. Whenever you simulate a model, the **Rename this result to save it** entry is overwritten with results generated from the most recent simulation.

You can save simulation results to compare and refer to multiple graphs generated during the current MapleSim session.


Stored Results
Rename this result to save it
Simulation with current probe
Simulation with voltage probe

If you want to refer to a set of simulation results in a future MapleSim session, you can also save the results as part of a model. When you open the model in a future MapleSim session, the graphs, progress messages, and 3-D animation that you saved will be available in the **Stored Results** palette.

If you want to work with your simulation data in another application, you can also export your results to a Microsoft Excel (.xls or .xlsx) or comma-separated value (.csv) file.

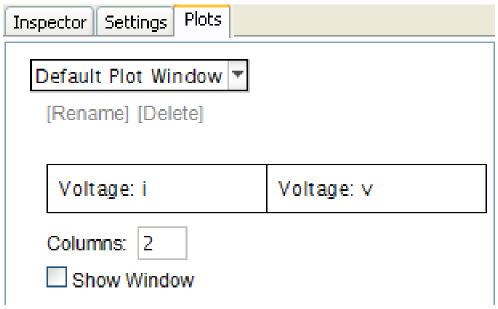
For more information, see the **Using MapleSim → Simulating a Model →Managing Simulation Results** section in the MapleSim help system.

4.5 Customizing Plot Windows

By default, quantities are plotted in separate simulation graphs and are alphabetically listed according to probe and quantity names. In each graph, the quantity values are plotted along the y axis versus the simulation time values along the x axis.

You can optionally create a custom plot window layout to specify the graph and customized plot titles, and specify the number of columns to appear in the plot window. You may want to create a custom plot window layout if, for example, you want to compare multiple quantities in the same graph, plot one quantity versus another, or view a simulation graph for a specific quantity without editing other probe values.

To create a custom plot window layout, you specify attributes in the **Plots** tab on the right side of the MapleSim window.



Voltage: i	Voltage: v

Columns: 2

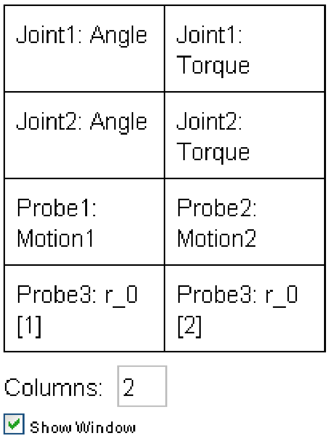
☐ Show Window

You can then use the new layout in a simulation. When you select the custom plot window layout from the drop-down menu in the **Plots** tab, select the **Show Window** check box, and then simulate your model, a custom plot window with the specified attributes appears in addition to the default plot window. You can store multiple layouts and select the one you want to use when you run a simulation.

Example: Plotting Multiple Quantities in Individual Graphs

In this example, you will create a custom plot window layout to plot and compare multiple quantities in the generated simulation graphs.

1. In the **Libraries** tab, expand the **Examples** palette, expand the **Multidomain** menu, and then open the **Controlled 2 Link Robot** example.
2. Click the **Plots** tab located on the right side of the MapleSim window. The following table, which displays all of the selected probe quantities, appears in the pane.



Joint1: Angle	Joint1: Torque
Joint2: Angle	Joint2: Torque
Probe1: Motion1	Probe2: Motion2
Probe3: r_0 [1]	Probe3: r_0 [2]

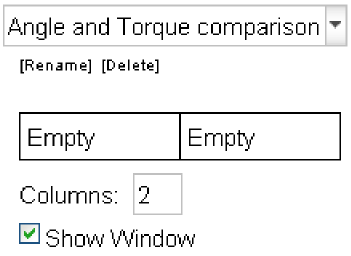
Columns: 2

✓ Show Window

This table displays the default layout of the generated graphs in the plot window. For example, the table shown in the image above indicates that the graph for the **Joint1:Angle** quantity will be displayed in the top-left corner of the plot window, the graph for the **Joint1:Torque** quantity will be displayed in the top-right corner of the plot window, and so on after you run a simulation.

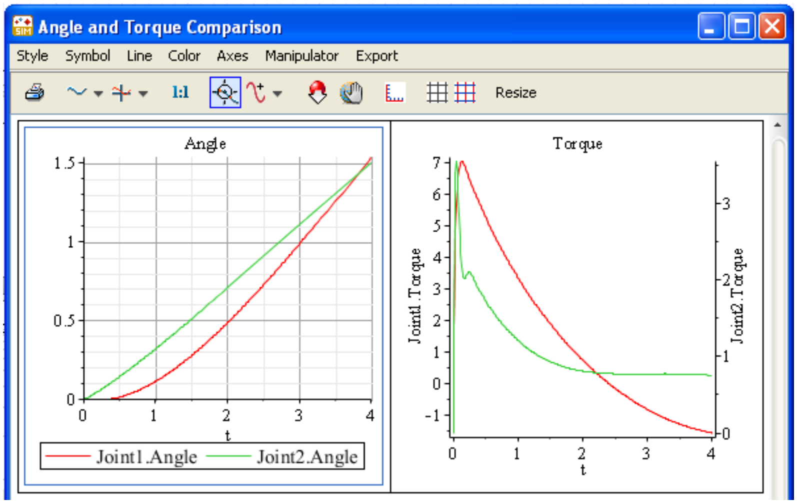
Create a custom plot window that displays both of the angle quantities in one graph and both of the torque quantities in another graph.

1. From the drop-down list at the top of the pane, select **Add Window**.
2. In the **Create Plot Window** dialog box, specify a plot window layout name **Angle and Torque Comparison**.
3. In the **Columns** field, type **2** and press **Enter**. The table in the pane now contains two cells; each cell represents a plot window area for which you can specify layout attributes.



4. Click **Empty** in the left cell.
5. In the **Title** field, enter **Angle**.
6. From the **Primary Y-axis** drop-down menu, select **Joint1: Angle**.
7. Click **[Add Variable]** below the drop-down menu.
8. From the second **Primary Y-axis** drop-down menu, select **Joint2: Angle**.
9. In the table at the top of the pane, click **Empty** in the top-right cell.
10. In the **Title** field, enter **Torque**.
11. From the **Primary Y-axis** drop-down menu, select **Joint1: Torque**.
12. Click **[Add Variable]** below the drop-down menu.
13. From the second **Primary Y-axis** drop-down menu, select **Joint2: Torque**. You can now simulate the model using the new plot window layout.
14. Make sure that the **Show Window** check box in the **Plots** tab is selected.

15. Click the simulation button (▶) in the main toolbar. The following custom plot window, which compares the angle values in one graph and the torque values in another, appears in addition to the default plot window.


Angle and Torque Comparison
Style Symbol Line Color Axes Manipulator Export
1:1
Resize
Angle
t
Joint1.Angle
Joint2.Angle
Torque
Joint1.Torque
Joint2.Torque
t

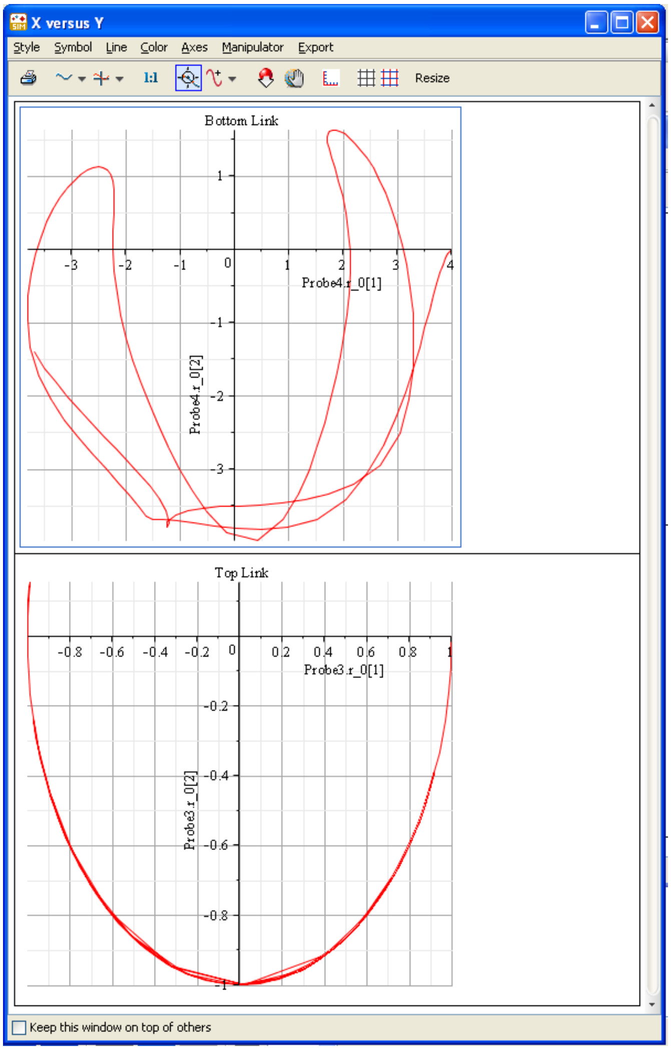
If you want to display the default plot window only, clear the **Show Window** check box in the **Plots** tab and simulate your model again.

Example: Plotting One Quantity Versus Another

In this example, you will create a custom plot window layout to plot the X and Y position of each of the links of a double pendulum.

1. In the **Libraries** tab, expand the **Examples** palette, expand the **Multibody** menu, and then open the **Double Pendulum** example.
2. In the model workspace toolbar, click the probe button ( ).
3. Click the right port of the **L_1** shared subsystem
4. Click the probe once to position it in the model workspace.
5. In the **Inspector** tab, select **Length[1]** and **Length[2]**.
6. Add another probe that measures the **Length[1]** and **Length[2]** quantities to the right port of the **L_2** shared subsystem.
7. Click the **Plots** tab on the right side of the MapleSim window.
8. From the drop-down list at the top of the pane, select **Add Window**.
9. In the **Create Plot Window** dialog box, assign the plot window layout the name **X versus Y**. Click **OK**.

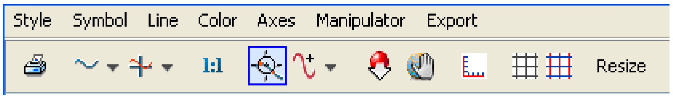
10. Click **Empty** in the table at the top of the pane.
11. In the **Title** field, enter **Bottom Link** and press **Enter**.
12. From the **X-axis** drop-down menu, select **Probe4: r_0[1]**
13. From the **Primary Y-axis** drop-down menu, select **Probe4:r_0[2]**.
14. In the table at the top of the pane, click **Empty** below the **Bottom Link** cell.
15. In the **Title** field, enter **Top Link** and press **Enter**.
16. From the **X-axis** drop-down menu, select **Probe3: r_0[1]**.
17. From the **Primary Y-axis** drop-down menu, select **Probe3:r_0[2]**.
18. Make sure that the **Show Window** check box is selected.
19. Click the simulation button (▶) in the main toolbar. The following custom plot window appears, in addition to the default plot window.



The plots above show the motion of the end point of each link in the pendulum. The bottom link follows a more disorderly path because of the interaction with the top link.

4.6 Plot Window Toolbar and Menus

After simulating a model, you can use the tools and menus in the plot window to customize curves, axes, and gridlines; browse simulation graphs; and export simulation graphs to several image formats. The following menus and toolbar appear at the top of each generated plot window.



You can hover your mouse pointer over any of the toolbar buttons to view their descriptions.

For more information about these tools, see the **Using MapleSim → Simulating a Model → Working with Simulation Graphs** section of the MapleSim help system.

4.7 Visualizing a Multibody Model

In MapleSim, the 3-D visualization environment allows you to build and analyze 3-D graphical representations of multibody systems. As you build a model and change its parameters, you can validate the 3-D configuration of the model and visually analyze your simulation results. You can build 3-D models by dragging and connecting objects in the 3-D workspace, and you can visualize your simulation results by playing animations that depict the movement of the objects.

As you build a block diagram in the model workspace, the corresponding changes are automatically reflected in the 3-D representation displayed in the 3-D workspace. Similarly, when you build a model in the 3-D workspace, the corresponding changes are automatically reflected in the block diagram displayed in the model workspace. Changes that you make in either of the workspaces are shown in both the model workspace and 3-D workspace as you edit your model.

In the 3-D workspace, you can view your model from any direction and control playback options to focus on specific components and their motions. Also, you can attach 3-D shapes to parts of your model to create a realistic-looking system representation. These shapes can either be imported from an external CAD file or selected from the **Multibody →Visualization** palette in the **Libraries** tab. You can also attach trace lines to show where components move during an animation.

For more information about adding 3-D shapes and using the 3-D workspace, see the **Using MapleSim → Visualizing a Model** section of the MapleSim help system.

3-D Visualization Settings

At the top level of your model, in the **Settings** tab, you must first specify the 3-D visualization settings for your model. for the Multibody and 3-D visualization area.

Multibody

You can specify the following parameter values for models containing multibody mechanical components:

Parameter	Default	Description
$\widehat{e}_g$	$[0, -1, 0]$	Direction of gravity.
g	9.81	The acceleration due to gravity of Earth at the surface. The default units are in $\frac{m}{s^2}$.
3-D animation	true	Specifies whether a 3-D animation is generated after running a simulation. When this option is set to **false**, no 3-D animation is generated. If your model is complex and if you do not plan on animating your model, you may want to set this option to **false** to reduce the time required to run a simulation.
3-D playback time	-	Specifies the playback duration of the 3-D animation, at a 1x speed rate, in seconds. This value differs from the $\mathbf{t_f}$ value, which in comparison specifies the simulation duration represented in your simulation graphs. You can specify a floating-point value for this option when the **3-D animation** field is set to **true**. You can specify a value in this field to increase or decrease the speed at which an animation is played. For example, if the $\mathbf{t_f}$ value is set to 0.5 seconds, you can set the **3-D playback time** value to 10 seconds to slow down the animation; an animation of the 0.5 second simulation would then be played back over a span of 10 seconds in real time. If no value is specified in this field, the **3-D playback time** value is the same as the value entered in the $\mathbf{t_f}$ field: an animation of a 10 second simulation, for example, would be played back over a span of 10 seconds in real time. The number of frames represented in an animation is determined by the value specified in the **plot points** field, and the **3-D playback time** value multiplied by the **3-D sampling rate** value.

Parameter	Default	Description
3-D sampling rate	30	Number of frames per second to include in the 3-D animation playback. You can increase this value to create a smoother transition between frames in your animation. You can specify a positive integer for this option when the **3-D animation** field is set to **true**. The number of frames represented in an animation is determined by the value specified in the **plot points** field, and the **3-D playback time** value multiplied by the **3-D sampling rate** value.

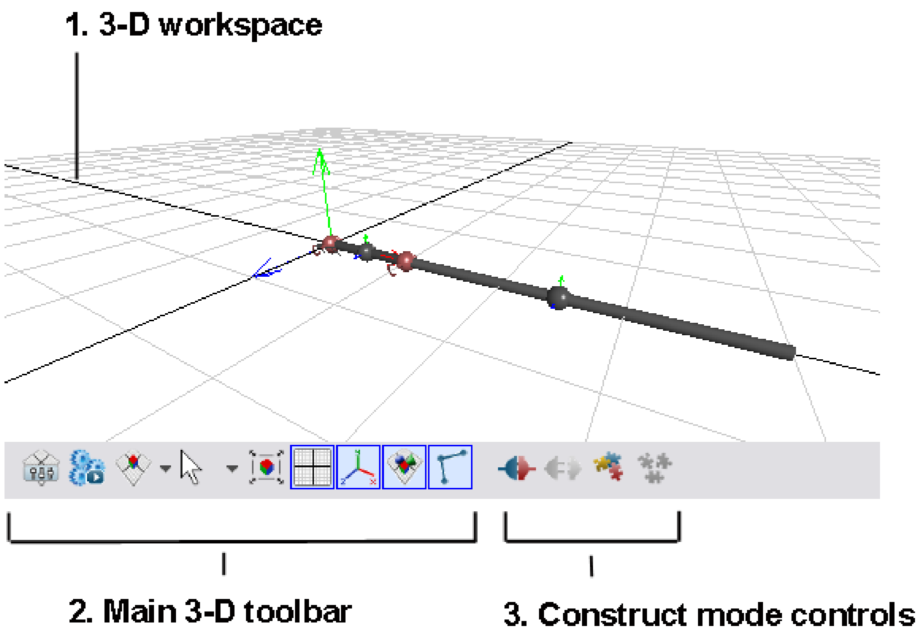
3-D Visualization

You can specify the following 3-D Visualization values for models containing multibody mechanical components:

Parameter	Default	Description
Enable translational snapping	off	When enabled, components are positioned at the closest location in 3-D space based on the translation snap delta values.
Translation snap delta	1.0	Specifies the translation snap delta spacing.
Enable rotational snapping	off	When enabled, components are positioned at the closest location in 3-D space based on the rotation snap delta values.
Rotation snap delta	$\frac{\mathrm{Pi}}{8}$	Specifies the rotational snap delta spacing.
Grid Extent	10	Specifies the extent of the grid spacing for the perspective view.
Grid Spacing	1.0	Specifies the grid spacing.
Base Radius	0.05	Specifies the radius of all the spheres as implicit geometry in the 3-D workspace for 3-D animation viewing purposes.
Enable view change animations	on	When enabled, a smooth transition occurs from the 3-D orthographic and perspective views in the 3-D model workspace.

The 3-D Workspace

The 3-D workspace is the area in which you build and animate 3-D models in the MapleSim window.



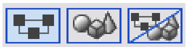
Component	Description
1. 3-D workspace	The area in which you build, view, and animate a 3-D model. The arrows at the origin indicate the directions of the world axes and are designated by color: • **X** - red • **Y** - green • **Z** - blue You can use the grid as a reference to determine the relative sizes and positions of elements in your 3-D model.
2. Main 3-D toolbar	Contains tools for hiding and displaying components in the 3-D workspace, toggling between different modes, selecting camera navigation tools, and changing the 3-D model view.

Component	Description
3. Construct mode controls	Controls for building and assembling a 3-D model, and connecting 3-D objects. **Note:** When you switch to playback mode, the construct mode controls are hidden and the playback controls for animating a 3-D model and specifying camera tracking options appear in this toolbar.

You can hover your mouse pointer over any of the buttons to view their descriptions.

Displaying the 3-D Workspace

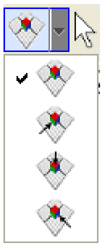
To display and hide the 3-D workspace and model workspace, use the buttons located at the bottom of the MapleSim window. By default, the 3-D workspace is not displayed.



Click the 3-D view button ( ) if you want to work with your model in the 3-D workspace only, the block diagram view button ( ) if you want to work in the model workspace only, or the combined view button ( ) if you want to work in both the model workspace and 3-D workspace at the same time.

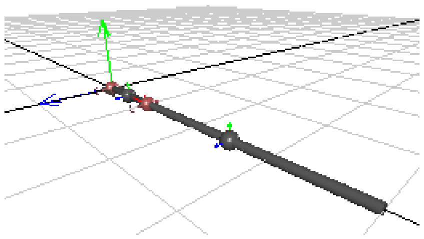
Viewing and Browsing 3-D Models

In the 3-D workspace, you can view and browse a 3-D model from the *perspective* view or one of the *orthographic* views using the **3-D View Controls** tool.



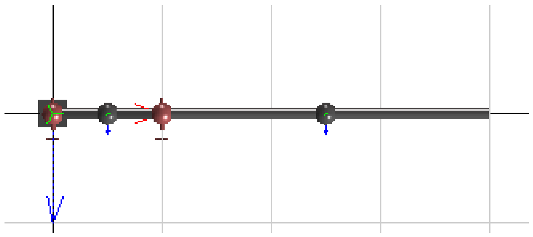
The perspective view allows you to examine and browse a model from all angles in 3-D space. It allows you to see 3-D spatial relationships between elements in your model. In the perspective view, objects that are closer to the camera appear larger than those that are further away from the camera.

In the following image, a double pendulum model is shown from the perspective view.



You can also view your 3-D model from front, top, and side orthographic views. Orthographic views use parallel projection as opposed to perspective projection, so your 3-D model appears as a flattened object because no depth information is shown. Orthographic views are sometimes referred to as "true length" views because they display undistorted lines and distances in the view plane that is orthogonal to the camera direction; these views are useful for analyzing spatial relationships or clearances between objects.

In the following image, the double pendulum model is shown from the top orthographic view.

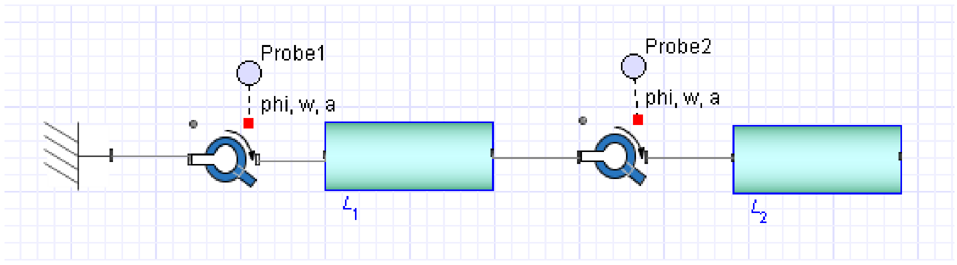


You can browse a model and change the model view while an animation is static or playing. In all of the views, you can pan and zoom into or out from your model. In the perspective view, you can also move the camera to view your model from above or below, and from any direction around your model.

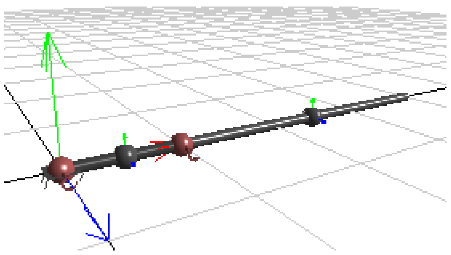
Tip: Before panning, zooming, or moving the camera around a large 3-D model, hover your mouse pointer over the object that you want to focus on. MapleSim adjusts the navigation controls according to the object on which you place the mouse pointer.

Adding Shapes to a 3-D Model

By default, basic spheres and cylinders called *implicit geometry* appear in the 3-D workspace to represent physical components in your model. For example, consider the following double pendulum model, which contains two revolute joints and two subsystems that represent planar links.



In the 3-D workspace, the implicit geometry of the fully assembled pendulum model appears as follows.



In this example, the spheres represent the revolute joints and rigid bodies, and the cylinders represent the planar links.

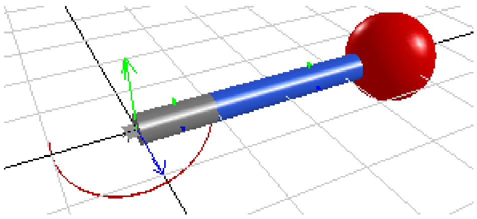
Implicit geometry that is not connected to other implicit geometry is drawn in a light gray color; implicit geometry that is assembled is drawn in a dark gray color, with the exception of joint objects, which are drawn in red.

Note: Components that you exclude from a simulation in the model workspace are not displayed in the 3-D workspace.

If you want to create a more realistic representation of your model, you can add shapes and lines called *attached shapes* to your model. To do so, you first add and connect attached shape components from the **Multibody →Visualization** palette to your block diagram in the model workspace.

When you simulate your model, the attached shapes appear in the 3-D workspace, in addition to the implicit geometry. In the following image, attached shapes have been added to represent the pendulum stem and bob pictorially. Also, a trace line - the curved line in the image

- is used to depict the locus of points that will be traced by a particular part of the model during a simulation.



You can customize the color, size, scale, and other visual aspects of the attached shapes by setting parameter values for individual components in the **Settings** tab before simulating the model.

If you want to view only the implicit geometry in the 3-D workspace, you can hide the attached shapes by clicking the attached shapes button ( ) in the main 3-D toolbar. If you want to view the attached shapes only, you can hide the implicit geometry by clicking the implicit geometry button ( ).

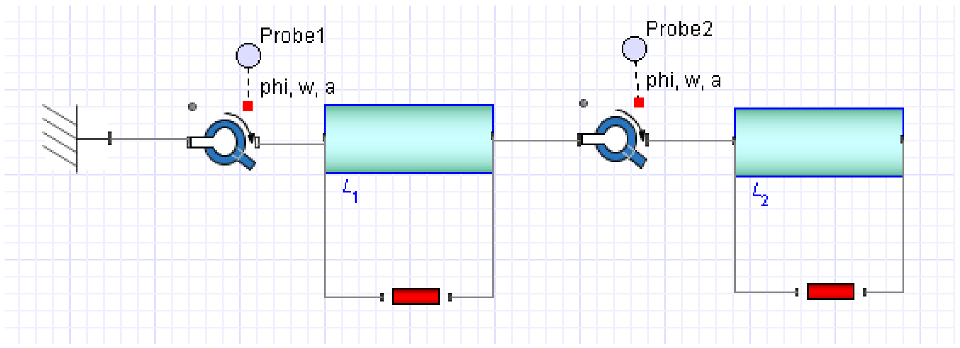
For more information about attached shape components, see the **MapleSim Component Library** → **Multibody** → **Visualization** → **Overview** topic in the MapleSim help system.

Note: If your model contains **Flexible Beam** components, deflection of the beam will not be depicted in the implicit geometry of your 3-D model.

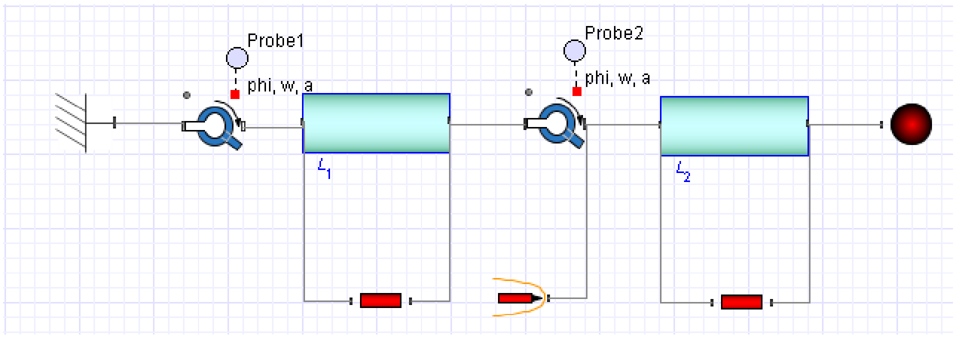
Example: Adding Attached Shapes to a Double Pendulum Model

In the following example, you will add cylinder shapes to represent the pendulum stem and a sphere component to represent the pendulum bob. You will also add a **Path Trace** component to display the path on which the revolute joint will move during an animation.

1. In the **Libraries** tab, expand the **Examples** palette, expand the **Multibody** menu, and then open the **Double Pendulum** example.
2. Expand the **Multibody** palette and then expand the **Visualization** menu.
3. Add two **Cylindrical Geometry** components below the planar link subsystems in the model workspace.
4. Connect the components as shown below.

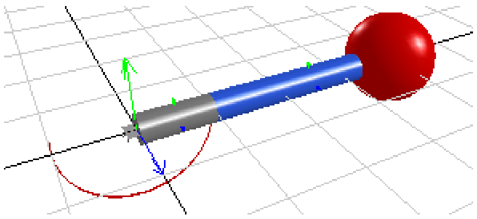


5. From the same menu, add a **Spherical Geometry** component and place it to the right of the **L_2** shared subsystem.
6. Right-click (**Control**-click for Macintosh) the **Spherical Geometry** component and select **Flip Horizontal**.
7. Add a **Path Trace** component and place it between the two **Cylindrical Geometry** components.
8. Connect the components as shown below.



9. Select the first **Cylindrical Geometry** component (**C_1**) in the model workspace.
10. In the **Inspector** tab on the right side of the MapleSim window, change the radius of the cylinder to **0.3**.
11. To select a color for the cylinder, click the box beside the **color** field and click one of the color swatches.
12. Select the second **Cylindrical Geometry** component (**C_2**) in the model workspace.
13. Change the radius of this cylinder to **0.3** and change the color.
14. To simulate the model, click the simulation button (▶) in the main toolbar.

When the simulation is complete, the 3-D workspace appears. The 3-D workspace is set to playback mode automatically and displays your model with the attached shapes.



15. To animate the model, click the play button (▶) below the 3-D workspace.

Building a Model in the 3-D Workspace

You can build MapleSim models by adding and connecting objects in the 3-D workspace. To add components to a 3-D model, you can drag multibody components from the **Multibody** palette, the **Favorites** palette, a custom library that you created, or from the search pane in the **Libraries** tab.

You can toggle between *construct mode* and *playback mode* to perform specific tasks in the 3-D workspace. In construct mode, you can add, connect, and lay out 3-D objects, and set initial conditions for joints and other multibody components by using graphical controls in the 3-D workspace. In playback mode, you can animate your 3-D model and specify camera tracking options to center an object in the 3-D workspace during an animation.

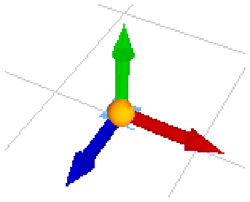
Any changes that you make to your 3-D model are automatically shown in the block diagram representation displayed in the model workspace and vice versa. For example, if you add and connect a **Flexible Beam** component in the 3-D workspace, the block diagram representation of the **Flexible Beam** with the added connection lines will be displayed in the model workspace at the same time.

Notes:

- Subsystems cannot be created in the 3-D workspace. They must be created in the model workspace.
- Components from the multibody **Forces and Moments**, **Sensors**, and **Visualization** component libraries cannot be dragged into the 3-D workspace. You must add these components in the model workspace.

Moving Objects in the 3-D Workspace

In construct mode, you can position individual objects or groups of objects by clicking and dragging the 3-D manipulators in the 3-D workspace.



To display the 3-D manipulator for a single unconnected object, click the object once in the 3-D workspace. You can then click and drag the blue arrow of the 3-D manipulator to move the object along the Z axis, the green arrow to move the object along the Y axis, and the red arrow to move the object along the X axis. You can also click and drag the sphere at the center of the 3-D manipulator to move the object in all directions.

For a group of connected objects, the configuration of your model determines where the 3-D manipulators are located.

- If your 3-D model contains a **Fixed Frame** component, click the square that represents the **Fixed Frame** component to display the 3-D manipulator.
- If your model does not contain a **Fixed Frame** component, click the object that defines the initial conditions for your system to display the 3-D manipulator. For example, if your model contains a **Rigid Body** component with its initial condition parameters set to **Strictly Enforce**, that **Rigid Body** component displays the 3-D manipulator. When a model is moved in the 3-D workspace, the initial conditions are updated for all of the other **Rigid Body** components that depend on the **Rigid Body** component that has its initial conditions set to **Strictly Enforce**.
- If your model does not contain a **Fixed Frame** component or a **Rigid Body** component with its initial conditions set to **Strictly Enforce**, click any of the objects in your 3-D model to display the 3-D manipulator. When you move the group of objects, the initial conditions of all of the multibody components in your model are set to **Treat as Guess**.

Note: To display 3-D manipulators, the multibody components in your model must contain numeric parameter values. If custom parameter values defined in a parameter block, global parameter, or subsystem parameter have been assigned to a multibody component, no 3-D manipulator will be displayed when you click that component in the 3-D workspace.

Assembling a 3-D Model

A 3-D model must be *assembled* before you can animate it in the 3-D workspace. Assembling a 3-D model refers to synchronizing the model displayed in the 3-D workspace with the initial configuration of your model defined by the assigned parameter values and initial condition guess values. The synchronization process occurs automatically when you simulate your model or click the 3-D view update button ( ) in the 3-D toolbar. In the 3-D workspace, assembled implicit geometry is drawn in a dark gray color, with the exception of joint objects, which are drawn in red.

Note: You can only assemble 3-D models with a valid configuration and valid connection lines. For example, if you attempt to assemble a 3-D model with missing connection lines, an error message will be displayed in the console pane and no animation will be generated.

For more information, see **Assembling a 3-D Model** in the MapleSim help system.

Using the Unenforce Constraints Button to Manipulate Joints in the 3-D Workspace

In construct mode, you can select a joint object in the 3-D workspace and click the unenforce constraints button ( ) to specify that the kinematic constraints of the joint are not enforced in the 3-D workspace as you build your model. Joints with unenforced kinematic constraints appear in pink in the 3-D workspace and its initial conditions are not shown in the 3-D workspace as you build your model.

You may want to use the unenforce constraints button if, for example, you are creating a closed-loop model in the 3-D workspace and you need a joint to remain in a specific position as you are building and laying out your 3-D model.

Notes:

- The unenforce constraints button does not affect the actual initial conditions specified for your joint components in the **Inspector** tab; it affects the initial conditions depicted in the 3-D workspace for display purposes only.
- Initial conditions for other joints with enforced kinematic constraints will be shown in the 3-D workspace, but will not affect related joints with unenforced kinematic constraints.

For example, consider a double pendulum 3-D model that contains a revolute joint with unenforced kinematic constraints and a second revolute joint with enforced kinematic constraints. If you change the initial angle of the joint with enforced kinematic constraints, the joint with the unenforced kinematic constraints will remain in its original position while the joint with enforced kinematic constraints will be shown at the new initial angle. To display all of the new initial conditions in the 3-D workspace, you must assemble your model by running a simulation or clicking the 3-D view update button ( ).

Displaying Attached Shapes as Your Build a 3-D Model

When you connect **Cylindrical Geometry**, **Tapered Cylinder Geometry**, **Box Geometry**, or **Spherical Geometry** components to your block diagram in the model workspace, the corresponding attached shapes appear in the 3-D workspace in both construct mode and playback mode. The attached shape appears in the 3-D workspace after you connect all of its ports to compatible ports of multibody components in the model workspace.

Working with CAD Geometry

CAD geometry can also be displayed in the 3-D workspace in both construct mode and playback mode. When you add a **CAD Geometry** component anywhere in the model workspace, the corresponding CAD image appears in the 3-D workspace regardless of whether the **CAD Geometry** component is connected to other components in your model. If a **CAD Geometry** component is not connected to other components, it will be drawn at the origin of the 3-D grid; if a **CAD Geometry** component is connected to another component, it will be drawn at the origin of the coordinate frame of the modeling component to which it is attached.

You can define the translational and rotational offset for CAD images either before or after connecting the corresponding **CAD Geometry** component to your model. To define these offsets, you can select the **CAD Geometry** component in the model workspace and specify parameter values in the **Inspector** tab.

Example: Building a Double Pendulum Model in the 3-D Workspace

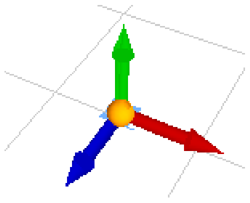
In this example, you will build and animate a double pendulum model in the 3-D workspace. You will perform the following tasks:

1. Add and move objects in the 3-D workspace.
2. Connect the 3-D objects.
3. Set initial conditions for the joints in your model.
4. Animate the 3-D model.

In a new MapleSim document, the 3-D workspace is set to construct mode by default.

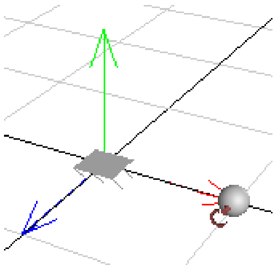
Adding and Moving Objects in the 3-D Workspace

1. Open a new MapleSim document.
2. At the bottom of the MapleSim window, click the 3-D view button () to display the 3-D workspace.
3. In the **Libraries** tab, expand the **Multibody** palette and then expand the **Bodies and Frames** menu.
4. From the palette, drag a **Fixed Frame** component into the 3-D workspace. A gray square, which represents the **Fixed Frame** component, is added to the 3-D workspace and its 3-D manipulator appears.

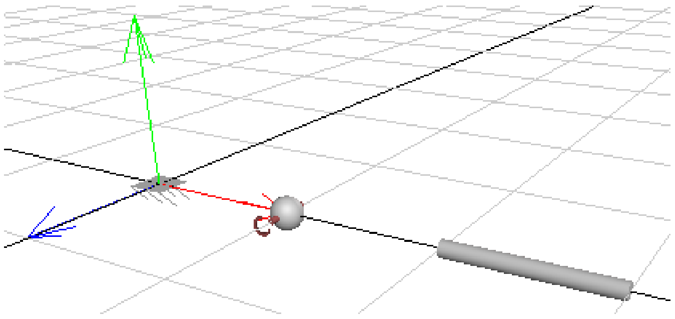


You can use this manipulator to position objects in the 3-D workspace.

5. Position the **Fixed Frame** object at the origin of the grid by clicking and dragging the 3-D manipulator arrow controls.
6. From the **Multibody → Joints and Motions** menu, drag a **Revolute** component into the 3-D workspace and place it to the right of the **Fixed Frame**.



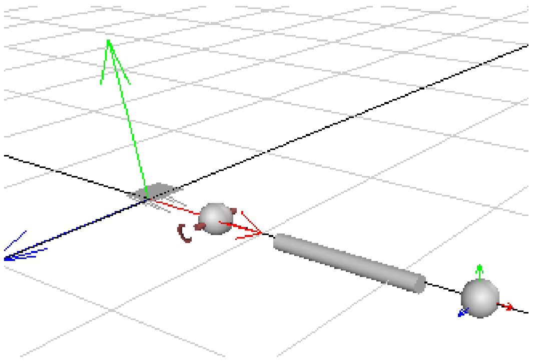
7. From the **Multibody → Bodies and Frames** menu, drag a **Rigid Body Frame** component into the 3-D workspace and place it to the right of the **Fixed Frame**.



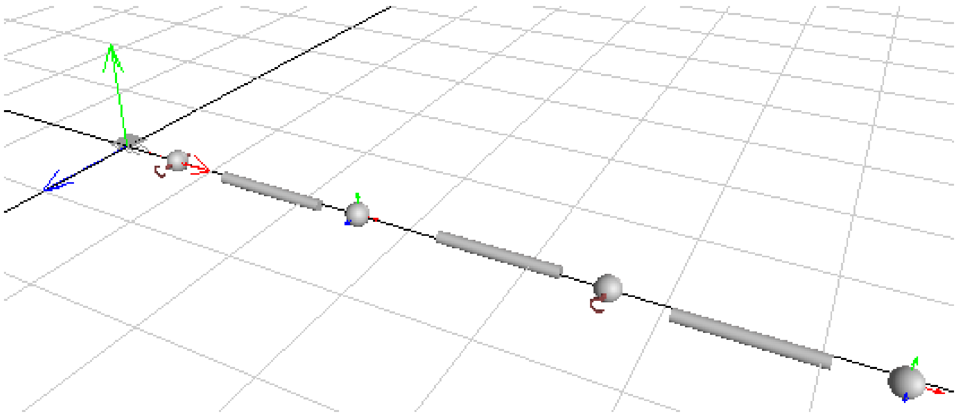
8. From the same menu, drag a **Rigid Body** component into the 3-D workspace and place it to the right of the **Rigid Body Frame**.

Tip: To zoom into and out from the 3-D workspace, hover your mouse pointer over the object that you want to focus on and rotate your mouse wheel. When zooming with the mouse wheel, the location under the pointer remains in place, allowing you to zoom in on that location. To pan your model, hold the **Shift** key and drag your mouse pointer in the 3-D workspace.

9. From the same menu, drag another **Rigid Body Frame** component into the 3-D workspace and place it to the right of the **Rigid Body**. The components for the first pendulum link have been added.



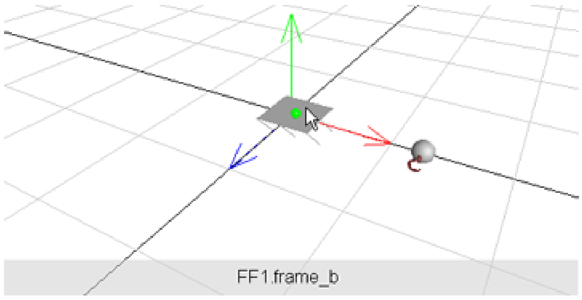
10. Repeat steps 7 to 9 to add components for a second pendulum link at the right of the last **Rigid Body Frame** component that you added.



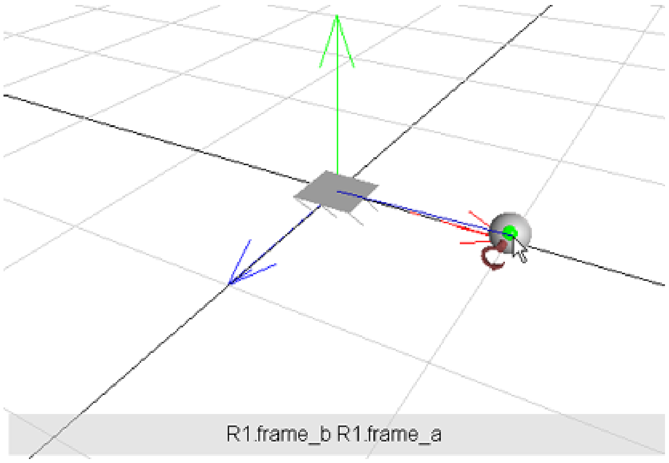
Connecting 3-D Objects

You will now connect the objects that you added in the previous task.

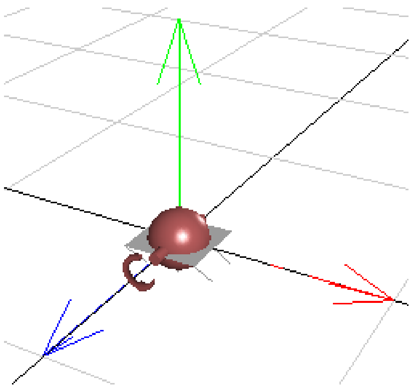
1. Click the connect button (🔗).
2. Hover your mouse pointer over the **Fixed Frame** object. A green dot appears.



3. Click the green dot once to start the connection line.
4. Hover your mouse pointer over the **Revolute** joint object. The gray panel at the bottom of the 3-D workspace displays the names of the **Revolute** joint frames.

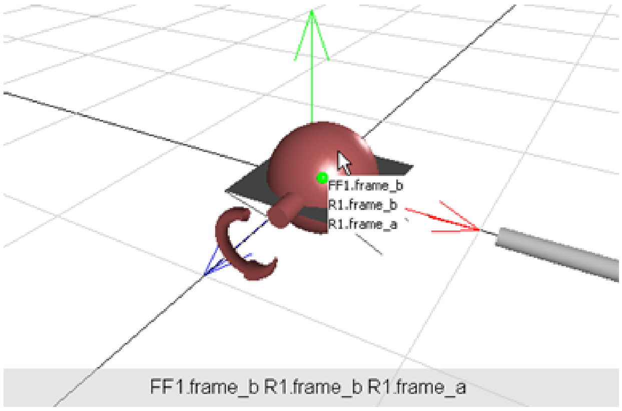


5. Click the **Revolute** joint object once. A context menu displays the names of the frames to which you can connect the line.
6. Select **R1.Frame_a**. The objects are connected in the 3-D workspace.

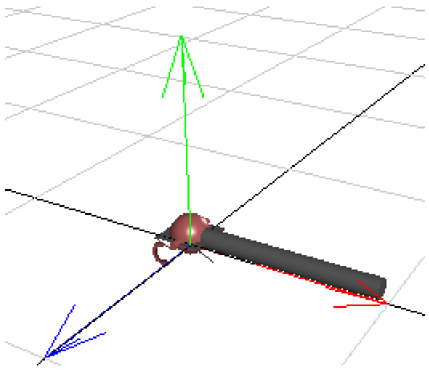


Note that the joint component is drawn in red when it is connected.

7. Click the connect button ( ) to start the next connection line.
8. Click the sphere that represents the **Revolute** joint. A context menu displays the frames of the **Revolute** joint, as well as the **Fixed Frame** to which it is connected.



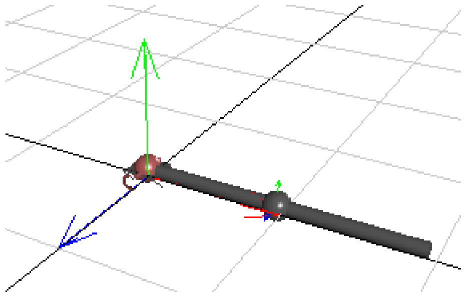
9. From the context menu, select **R1.frame_b**.
10. Drag your mouse pointer to the end of the cylinder that represents the **Rigid Body Frame** and click the green dot.



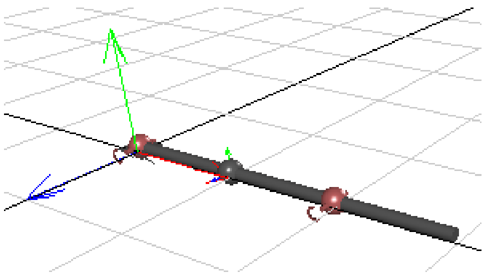
frame_b of the first revolute joint, $\mathbf{R_1}$, is now connected to **frame_a** of the first rigid body frame, $\mathbf{RBF_1}$.

11. Click the connect button to start a new connection line.
12. Hover your mouse pointer over the other end of the cylinder that represents the $\mathbf{RBF_1}$ component and click the cylinder once.
13. Drag your mouse pointer to the sphere that represents the first **Rigid Body** component, $\mathbf{RB_1}$, and click it once.
14. In the same way, connect **frame_b** of $\mathbf{RBF_1}$ to **frame_a** of the first rigid body, $\mathbf{RB_1}$, and then connect $\mathbf{RB_1}$ to the second **Rigid Body Frame, $\mathbf{RBF_2}$.**

Note: Click the connect button to start each connection line.



15. Connect **frame_b** of the second **Rigid Body Frame** to **frame_a** of the second **Revolute** joint.
16. Connect **frame_b** of the second **Revolute** joint to **frame_a** of the third **Rigid Body Frame**.
17. Connect the **Rigid Body Frame** to the second **Rigid Body**, and the connect the second **Rigid Body** to the fourth **Rigid Body Frame**. The complete 3-D model appears below.



If you click the block diagram view button (▣), you will see that all of the components are added and connected accordingly.

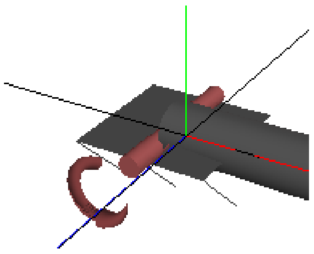
Tip: As you are building a 3-D model, it is a good practice to switch to the block diagram view periodically to check whether the block diagram is laid out the way you want.

Setting Initial Conditions for the Joint Components

In construct mode, you can set initial conditions for joint components by using graphical controls in the 3-D workspace.

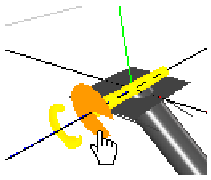
Note: Joint components that have been assigned custom parameter values defined in a parameter block, global parameter, or subsystem parameter will not allow the use of graphical controls for setting initial conditions. In these cases, use the fields in the **Inspector** tab to set the initial conditions.

1. If you are in the block diagram view, click the 3-D view button (▣) at the bottom of the MapleSim window to display the 3-D workspace.
2. In the 3-D workspace, to set the initial angle of the first revolute joint, click the sphere that represents the first revolute joint in the 3-D workspace. The red sphere, which represents the joint component, is removed temporarily from the 3-D workspace and the manipulator for the joint appears.



3. Hover your mouse pointer over the manipulator. The manipulator appears in yellow.

4. Click and drag your mouse pointer around the manipulator to display the meter that represents the initial angle value that you want to set for the revolute joint. A pie graph-shaped meter appears in orange.



When you drag your mouse pointer, you can adjust the initial angle value for the degree of freedom represented by the graphic. The angle value increases if you drag the mouse pointer up or to the right, and decreases if you drag the mouse pointer down or to the left.

5. Release your mouse button when the meter is at the approximate initial condition value that you want. In the **Inspector** tab, the $\boldsymbol{\theta_0}$ parameter displays the value that you selected and the implicit geometry is set to that value in the 3-D workspace.

Tips:

- Alternatively, you can set initial conditions for your model by entering a value for the $\boldsymbol{\theta_0}$ parameter in the **Inspector** tab. Initial conditions that you specify in the **Inspector** tab will be shown in the 3-D model.
- To specify precise initial angle conditions, turn on snapping by clicking the visualization settings button ( ) in the main 3-D toolbar and selecting **Enable rotational snapping** in the **Settings** tab.

Animating the 3-D Model

You will now simulate your 3-D model to generate the animation that can be viewed in playback mode.

To animate the 3-D model

1. Simulate your model by clicking the simulation button ( ) in the main toolbar. When the simulation is complete, the 3-D workspace is set to playback mode automatically.
2. To play the animation, click the play button ( ) below the 3-D workspace.

Exporting a Movie of the 3-D Model

The Export Movie feature allows you to export a recorded simulation as an .mpeg file to users who may not have MapleSim. For more information, see the Using MapleSim → Visualizing a 3-D Model → Exporting a Simulation as a Movie topic in the MapleSim help system.

4.8 Best Practices: Simulating and Visualizing a Model

This section describes best practices to consider when simulating and visualizing a model.

Use an External C Compiler to Run Simulations With Longer Durations

When you set the **compiler** parameter to **true** in the **Settings** tab/Solver, Maple procedures generated by the simulation engine are translated to C code and then compiled by an external C compiler. As a result, the time required to run a simulation can be reduced. In general, when you use a C compiler to simulate a model, the compilation process will be faster in simulations with longer durations.

Compare Results Generated By Sections of Your Model

For debugging purposes, you may want to view simulation results for a specific section or subsystem in your model. By selecting a section in your model and clicking the disable button (⊘) above the model workspace, you can exclude part of your model from the next simulation that you run. When you simulate your model, results will be displayed only for the model sections that you did not exclude. This feature allows you to view simulation results generated by specific sections in your model and compare results without having to delete components from the model workspace or build multiple models.

For more information, see the **Using MapleSim → Simulating a Model → Excluding Objects From a Simulation** topic in the MapleSim help system.

5 Analyzing and Manipulating a Model

In this chapter:

5.1 Overview

MapleSim is fully integrated with the Maple environment, so you can use Maple commands, embedded components, plotting tools, and many other technical document features to analyze and manipulate the dynamic behavior of a MapleSim model or subsystem. For example, you can use Maple to retrieve and work with model equations, test input and output values, translate your model into C code, and perform many other advanced analysis tasks.

To start working with your model in Maple, you can use the templates available in the MapleSim templates dialog box. These templates are Maple worksheets with pre-built tools for model building and analysis tasks: you first create a MapleSim model and open it in one of the available templates to perform an analysis task in Maple.

The following templates are available:

Template Name	**UI Display Name**	**Task**
C Code Generation Template	Code Generation	Translate your model into C code.
Custom Component Template	Custom Component	Create a custom modeling component based on a mathematical model. For more information, see *Creating Custom Modeling Components (page 61)*.
Data Generation Template	Data Generation	Define and generate a data set to be used in MapleSim, for example, a data set for an interpolation table component. For more information, see *Creating a Data Set for an Interpolation Table Component (page 54)*.
Equation Extraction Template	Equations	Retrieve equations from linear or nonlinear models.
Discrete State Space Custom Component	Custom Discrete State Space	Define and generate custom components for a MapleSim model from a discrete state space description.

Template Name	UI Display Name	Task
Discrete Transfer Functions Custom Component	Custom Discrete Transfer Function	Define and generate custom components for a MapleSim model from a discrete transfer function.
Excel Connectivity Template	Excel Connectivity	Import MapleSim parameter sets from, or export MapleSim parameter sets to, an Excel spreadsheet.
Linear System Analysis Template	Analysis	View and analyze the equations of a linear system.
Linearization	Linearization	Create a linear system object from a MapleSim continuous subsystem.
Custom Component Template: Modelica Code Definition	Modelica Custom Component	Define and generate a MapleSim custom component from Modelica code
Monte-Carlo Simulation Template	Monte Carlo Simulation	Define a random distribution for a parameter and run a simulation using the distribution.
Multibody Analysis Template	MultibodyAnalysis	Retrieve multibody equations in a form that is suitable for manipulation and analysis.
Parameter Optimization Template	Optimization	Analyze and edit the parameters of a model and view possible simulation results in a graph.
Random Data Template	Random Data	Define and generate a set of random data points to be used in MapleSim, for example, a data set for an interpolation table component.
Sensitivity Analysis	Sensitivity Analysis	Perform parameter sensitivity analysis.

Alternatively, to edit and analyze a model, you can insert a MapleSim Model embedded component into a Maple worksheet and open an existing MapleSim model in that component. For more information about the MapleSim Model component, see *Working with Maple Embedded Components (page 106)*.

Both methods of performing analysis tasks allow you to use commands from any Maple packages, including **MapleSim** and **DynamicSystems**, to work with your model programmatically.

Note: After using a MapleSim template, save the .mw file and then save the .msim file to which the .mw file is attached.

Tip: The pre-built analysis tools available in each template are Maple embedded components, which allow you to interact with Maple code through graphical interactive components. The code associated with each embedded component uses commands from Maple packages, including **MapleSim** and **DynamicSystems**.

To view the code associated with an embedded component, right-click (**Control**-click for Macintosh) any of the tools in the Maple worksheet, select **Component Properties**, and click **Edit**. For more information about embedded components, see the **EmbeddedComponents** topic in the Maple help system.

Working with Equations and Properties in a Maple Worksheet

When viewing and working with equations or properties in a template, note the following:

- The programmatic names of certain parameters, variables, and connectors displayed in the Maple worksheet differ from the names displayed for the corresponding elements in the MapleSim interface. For example, if an **Inertia** component is included in a model, the parameter for the initial value of the angular velocity is displayed as ω_0 in the MapleSim interface and *w_start* in a Maple worksheet. For more information about the mappings of parameter, variable, and connector names, see the **MapleSim Component Library** in the MapleSim help system.
- Subscripts and superscripts in the MapleSim interface are represented differently in a Maple worksheet. Subscripts in the MapleSim interface appear with an underscore character in a Maple worksheet. For example, a connector called $flange_a$ in the MapleSim interface would be displayed as *flange_a* in a Maple worksheet. Also, superscripts are formatted as regular characters in a Maple worksheet. For example, a variable called a^2 in the MapleSim interface would be displayed as *a2* in a Maple worksheet.

5.2 Retrieving Equations and Properties from a Model

You can use the Equation Analysis Template to retrieve and analyze equations and properties such as parameters, initial equations, and variables in your model.

1. In MapleSim, open the model for which you want to retrieve equations or properties.
2. Click the templates button ( ) in the main toolbar.
3. From the list, select **Equations**.
4. In the **Attachment** field, enter a name for the template and click **Create Attachment**. Your model is opened in the Equation Extraction Template in Maple.
5. Using the navigation tools above the model diagram, select the subsystem for which you want to view equations. If you want to retrieve equations from the complete system, click **Main**.
6. In the **Model Input from MapleSim** section, click **Retrieve System**.
7. To retrieve model equations, in the **Model Equations** section of the template, select the **System Equations** radio button. Alternatively, to retrieve other types of equations and properties, click the radio button beside the equation type or property that you want to view.
8. To work with the equations further in Maple, assign them to a variable by clicking the **Assign to variable** button.

You can now manipulate the equations using any Maple packages, for example, **DynamicSystems** and **MapleSim**. For more information about these packages, see the **DynamicSystems** and **MapleSim** topics in the Maple help system.

5.3 Analyzing Linear Systems

You can use the Linear System Analysis Template to view and analyze the equations of a linear system, test system input and output values, and view possible simulation results in a Bode or root locus plot.

1. In MapleSim, open the linear system model that you want to analyze.
2. Click the templates button ( ) in the main toolbar.
3. From the list, select **Analysis**.
4. In the **Attachment** field, enter a name for the template and click **Create Attachment**. Your model is opened in the Linear System Analysis Template in Maple.
5. Using the navigation tools above the model diagram, select the subsystem for which you want to view equations.

Note: To perform linear analysis using the tools in the **Analysis and Simulation** section of the template, you must select a subsystem; linear analysis cannot be performed on the entire system.

6. In the **Model Input from MapleSim** section, click **Retrieve System**.
7. Define the system input and output values. To add a value as an input or output, select an entry from the **System IO and Probes** menu, and click >. To add all of the values from the list, click >>.
8. Select the equation type that you want to analyze and click **Build Linear System Object**.

You can now select a parameter from the list, change the value of that parameter, and use the plotting tools in the **Linear System Analysis Tools** section to view possible simulation results.

5.4 Optimizing Parameters

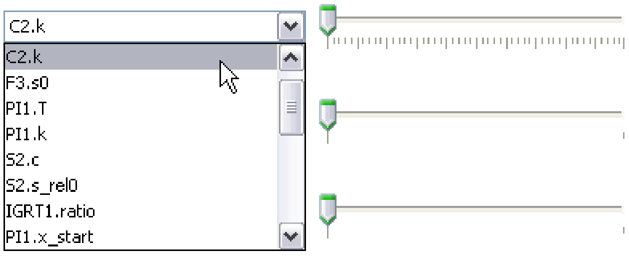
You can use the Parameter Optimization Template to test various conditions using the embedded components provided in the worksheet and view possible simulation results in a plot.

You can also use commands from the Global Optimization Toolbox to perform parameter optimization tasks. This product is not included with MapleSim. For more information, visit the Maplesoft Global Optimization Toolbox web site at

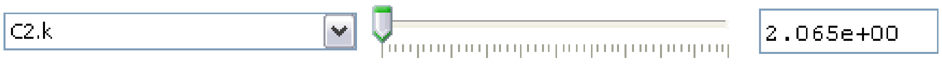
http://www.maplesoft.com/products/toolboxes/globaloptimization/.

To optimize parameters proceed as follows:

1. In MapleSim, open the linear system model that you want to analyze.
2. Click the templates button ( ) in the main toolbar.
3. From the list, select **Optimization**.
4. In the **Attachment** field, enter a name for the template and click **Create Attachment**. Your model is opened in the Parameter Optimization template in Maple.
5. In the **Parameter Investigation** section of the template, click **Retrieve System Parameters**.
6. In the **Simulation Settings** section, specify the simulation options to be used by the plotting analysis tools.
7. In the **Parameter Values** section, from the first drop-down menu, select the parameter that you want to test.



Note: When a parameter is selected, its assigned value is displayed in the field next to the slider.



8. In the **Range** fields beside the slider and parameter value field, specify the range of the slider. By default, the range is 0 to 10 unless the selected parameter value is outside of this range.
9. Using the process described above, select other parameters that you want to test.

When you have defined all of the parameters, you can move the sliders to test different values and view possible simulation results in the plot. You can also assign the parameters you defined to a Maple procedure to perform further analysis tasks in the **Parameter Optimization** section of the template.

5.5 Generating C Code from a Model

If you want to use or test your model in an application that supports the C programming language, you can use the Code Generation Template to translate a subsystem in your model into source code.

1. In MapleSim, open the model for which you want to generate code.
2. In the model workspace, make sure that the components for which you want to generate code are grouped in a subsystem.

Tip: If you want to generate code for your complete model, group all of the components at the top level of your model into a single subsystem.

3. Click the templates button ( ) in the main toolbar.
4. From the list, select **Code Generation**.
5. In the **Attachment** field, enter a name for the template and click **Create Attachment**. Your model opens in the C Code Generation template in Maple.
6. Using the navigation tools above the model diagram, select the subsystem for which you want to generate code.
7. In the **Model Input from MapleSim** section of the template, click **Retrieve System**.
8. Select the input and output values that will be included in the generated code.
9. In the **Generating Code** section, click **Code Generation**. The code is generated.
10. Click **Save C Function Library**. At the prompt save your code as a C file.

5.6 Working with Maple Embedded Components

In Maple, you can use the MapleSim Model embedded component to view, edit, and analyze the properties of MapleSim models programmatically. For example, you can view and change parameter values using commands in the **DocumentTools** package. Model or subsystem equations can be retrieved using commands from the **MapleSim** package and you can manipulate your model as a **DynamicSystems** object to analyze the model or subsystem behavior using any input functions.

You can also associate model properties with other Maple embedded components, including sliders and plots to create custom analysis tools.

For more information about advanced analysis tasks, open the **Sliding Table** example from the **Examples → Multidomain** palette in the **Libraries** tab, and open the **Advanced Analysis Worksheet** from the MapleSim **Attachments** palette.

For more information about the MapleSim Model component, see the **MapleSimModel** topic in the Maple help system.

6 MapleSim Tutorials

In this chapter:

6.1 Tutorial 1: Modeling a DC Motor with a Gearbox

In this tutorial, you will extend a DC motor model and perform the following tasks:

1. Add a gearbox to the DC motor model.
2. Simulate the DC motor with gearbox model.
3. Group the DC motor components into a subsystem.
4. Assign global parameters to the model.
5. Add signal block components and a PI controller to the model.
6. Simulate the modified DC motor model using different conditions.

Adding a Gearbox to a DC Motor Model

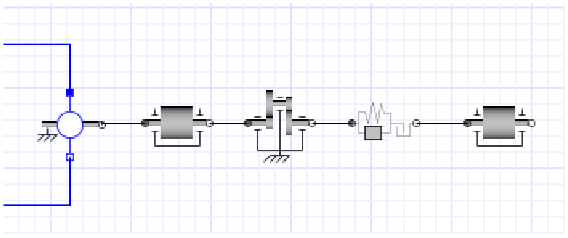
In this example, you will build the gearbox by adding and connecting an ideal gearbox component, a backlash component with a linear spring and damper, and an inertia component from the 1-D Mechanical library.

1. In the **Libraries** tab, expand the **Examples** palette, expand the **Tutorial** menu, and then open the **Simple DC Motor** example.
2. Perform the following tasks:

- From the **1-D Mechanical → Rotational → Bearings and Gears** menu, add an **Ideal Gear** component to the model workspace and place it to the right of the **Inertia** component.
- From the **1-D Mechanical →Rotational → Springs and Dampers** menu, add an **Elasto-Backlash** component to the model workspace and place it to the right of the **Ideal Gear** component.
- From the **1-D Mechanical → Rotational → Common** menu, add another **Inertia** component to the model workspace and place it to the right of the **Elasto-Backlash** component.

You can use the selection tool to drag and position components in the model workspace.

3. Connect the components as shown below.

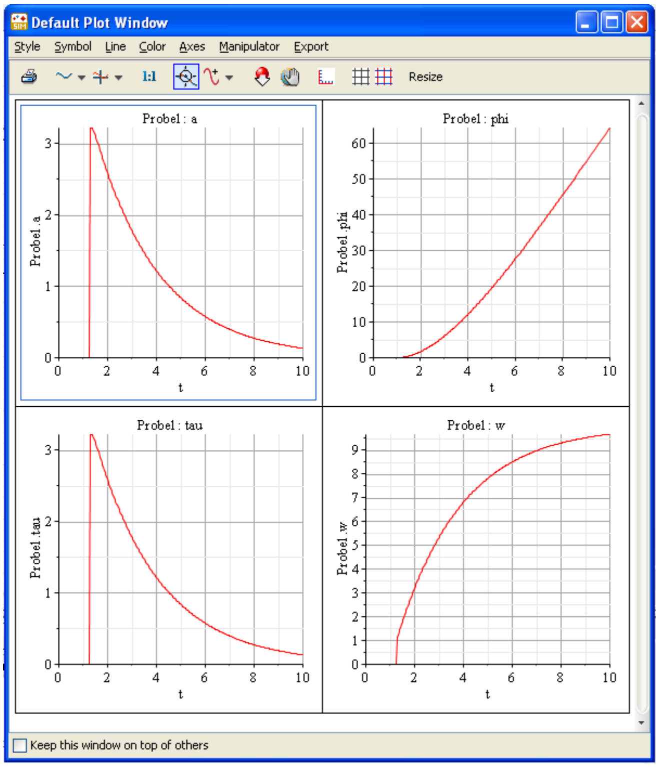


4. In the model workspace, click the **Ideal Gear** component.
5. In the **Inspector** tab, change the transmission ratio to **10** and press **Enter**.
6. Specify the following parameter values for the other components:

- For the **Elasto-Backlash** component, in the **b** field, change the total backlash value to **0.3** *rad*. In the **d** field, change the damping constant to $\mathbf{10^4}$ $\frac{N.m.s}{rad}$.
- For the first **Inertia** component ($\mathbf{I_2}$), in the **J** field, change the moment of inertia value to **10** $kg{\cdot}m^2$.
- For the **Step** source, in the **height** field, change the value to **100**.

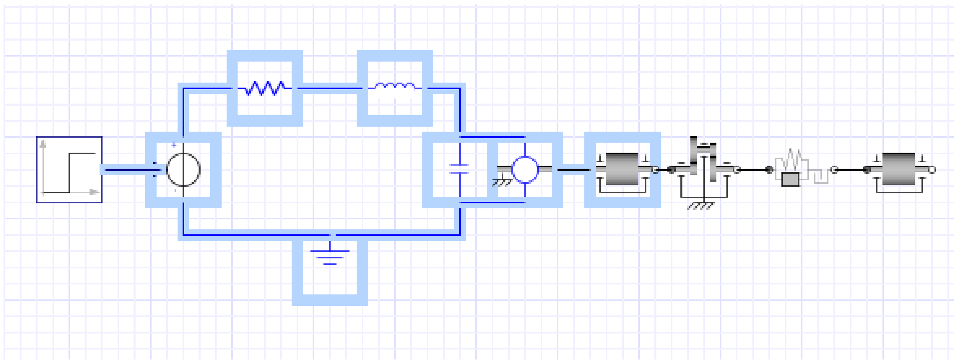
Simulating the DC Motor with Gearbox Model

1. Delete the existing probe from the model workspace.
2. From the model workspace toolbar, click the probe button ( ).
3. Hover your mouse pointer over the line that connects the **Elasto-Backlash** component and the second **Inertia** component ($\mathbf{I_3}$). The line is highlighted.
4. Click the line once and then click the probe to position it.
5. Select the probe in the model workspace.
6. To include the angle (φ), speed (*w*), acceleration (*a*), and torque (τ) values in the simulation graphs, in the **Inspector** tab, select **Angle**, **Speed**, **Acceleration**, and **Torque**.
7. Click a blank area in the model workspace.
8. In the **Settings** tab, set the $\mathbf{t_f}$ parameter to **10** seconds and press **Enter**.
9. Click the simulation button ( ) in the main toolbar. When the simulation is complete, the following graphs appear.



Grouping the DC Motor Components into a Subsystem

1. Using the selection tool ( ), draw a box around the electrical components and the first inertia component.



2. From the **Edit** menu, select **Create Subsystem**.

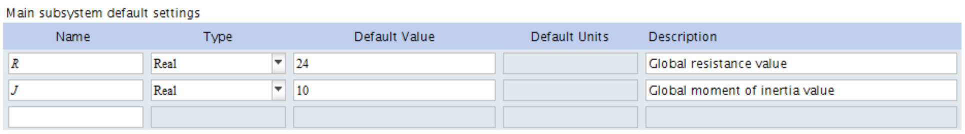
3. In the **Create Subsystem** dialog box, enter **DC motor**.
4. Click **OK**. A white block, which represents the DC motor, appears in the model workspace.

Tip: To view the components in the subsystem, double-click the **DC motor** subsystem in the model workspace. To browse to the top level of the model, click the **Main** button in the navigation toolbar.

Assigning Global Parameters to a Model

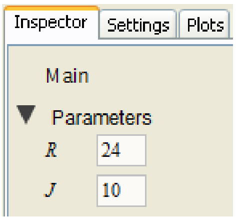
You can define a global parameter and assign its value to multiple components in your model.

1. Click **Main** in the navigation toolbar to browse to the top level of the model.
2. From the navigation toolbar, click the parameter editor button ( ) to switch to the parameter editor view.
3. In the first row of the **Main subsystem default settings** table, define a parameter called **R** and press **Enter**.
4. Specify a default value of **24** and enter **Global resistance value** as the description.
5. In the second row of the table, define a parameter called **J** and press **Enter**.
6. Specify a default value of **10** and enter **Global moment of inertia value** as the description.


Main subsystem default settings

Name	Type	Default Value	Default Units	Description
R	Real	24		Global resistance value
J	Real	10		Global moment of inertia value

7. To switch to the diagram view, click the diagram view button ( ) in the navigation toolbar. The new **R** and **J** parameters appear in the **Inspector** tab. You can now assign these parameter values to other components in your model.



Main

Parameters

R	24
J	10

8. In the navigation toolbar, click the parameter editor view button ( ).

9. In the I_3 **component** table, in the value field for the moment of inertia parameter, enter **J** and press **Enter**. The moment of inertia parameter now inherits the numeric value of the global parameter, **J** (in this example, **10**).
10. Switch to the diagram view and double-click the **DC Motor** subsystem.
11. In the navigation toolbar, click the parameter editor button ( ).
12. In the EMF_1 **component** table, in the value field for the transformation coefficient, enter **R·J** and press **Enter**.

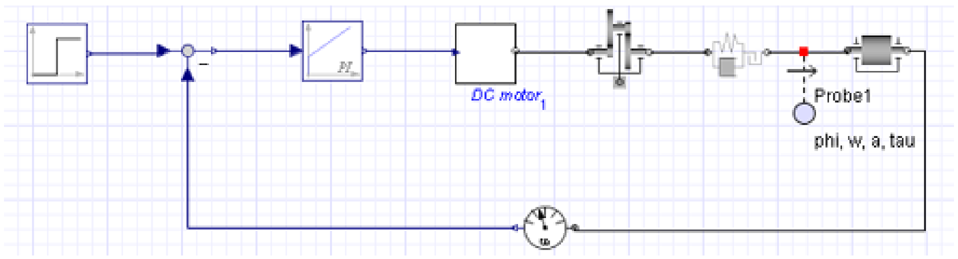
Note: This value is an approximation of the transformation coefficient.

13. In the R_1 **component** table, in the value field for the resistance parameter, enter **R** and press **Enter**.
14. Switch to the diagram view and browse to the top level of your model.
15. Save the model as **DC_Motor2.msim**.

Changing Input and Output Values

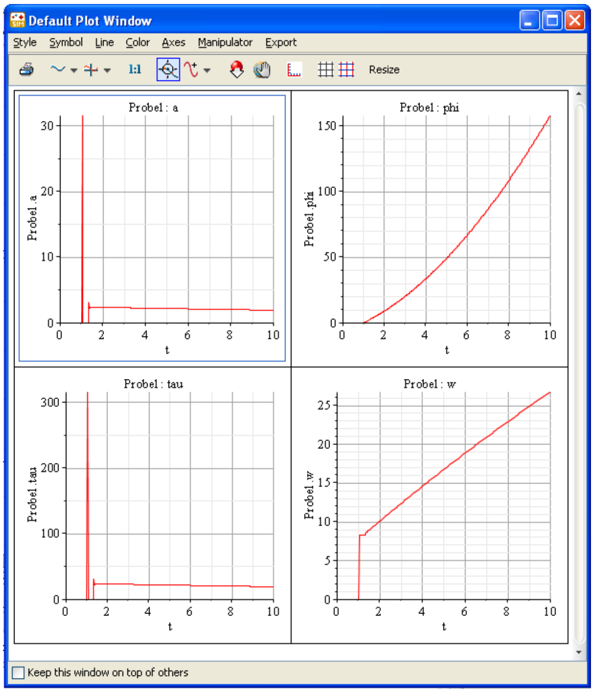
In this example, you will change the input and output values of the model to simulate different conditions.

1. From the **1-D Mechanical → Rotational → Sensors** menu in the **Libraries** tab, add the **Angular Velocity Sensor** component to the model workspace and place it below the gearbox components.
2. Right-click (**Control-**click for Macintosh) the **Angular Velocity Sensor** component and select **Flip Horizontal**.
3. Delete the connection line between the **Step** source and the **DC Motor** subsystem.
4. From the **Signal Blocks → Controllers** menu, add the **PI** component to the model workspace and place it to the left of the **DC Motor** subsystem.
5. From the **Signal Blocks → Mathematical → Operators** menu, add the **Feedback** component to the model workspace and place it to the left of the **PI** component.
6. Connect the components as shown below.



To draw a perpendicular line, click a point in the model workspace to anchor the line and then move your mouse cursor in a different direction to draw the second line segment.

7. Click the **PI** component in the model workspace.
8. In the **Inspector** tab, specify a gain of **20** in the **k** field, and a time constant of **3** seconds in the **T** field.
9. Simulate the model again. When the simulation is complete, the following graphs appear.



10. Save the model as **DC_Motor3.msim**.

6.2 Tutorial 2: Modeling a Cable Tension Controller

In this tutorial, you will extend the DC motor example to model a cable that is stretched with a pre-defined tension. The tension is defined by a **Constant** source and the **PI** controller provides the voltage to drive the motor. You will perform the following tasks:

1. Build a cable tension controller model.
2. Specify component properties.
3. Simulate the cable tension controller model.

Building a Cable Tension Controller Model

In this example, you will build the cable tension controller model using a combination of 1-D mechanical rotational and translational components. You will also group components into a **Gear** subsystem and add subsystem ports.

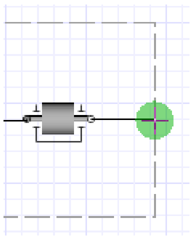
1. Open the **DC_Motor3.msim** file that you created in the previous tutorial and save the file as **Cable_Tension.msim**. Alternatively, open the **DC Motor with PI Control** example in the **Examples → Tutorial** palette.
2. Delete **Probe3** attached to the line that connects the **Elasto-Backlash** and **Inertia** components.
3. Delete **Angular Velocity Sensor** component and its connection lines.
4. Select the **Elasto-Backlash**, **Ideal Gear**, and **Inertia** components and group them into a subsystem called **Gear Components**.
5. Add the following components to the model workspace:

- From the **1-D Mechanical → Rotational → Bearings and Gears** menu, add the **Ideal Rotation to Translation Gear** component and place it to the right of the **Gear Components** subsystem.
- From the **1-D Mechanical → Translational → Sensors** menu, add the **Force Sensor** component and place it to the right of the **Ideal Rotation to Translation Gear** component.
- From the **1-D Mechanical → Translational → Springs and Dampers** menu, add the **Translational Spring** component and place it to the right of the **Force Sensor** component.
- From the **1-D Mechanical → Translational → Common** menu, add the **Translational Fixed** component and place it to the right of the **Translational Spring** component.

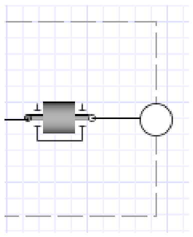
6. Right-click (**Control**-click for Macintosh) the **Translational Fixed** component in the model workspace and select **Rotate Counterclockwise**.
7. Delete the **Step** source and replace it with a **Constant** source from the **Signal Blocks → Sources → Real** menu.

Tip: You can connect the **Constant** source by dragging it onto the unconnected line end.

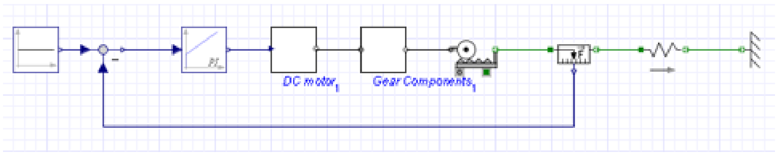
8. Double-click the **Gear Components** subsystem. You will now add a port to connect this subsystem with other components.
9. Click the negative (white) flange of the **Inertia** component and drag your mouse cursor to the boundary that surrounds the subsystem components.



10. Click the line once. The subsystem port is added to the line.



11. Click **Main** in the navigation toolbar to browse to the top level of your model.
12. Connect the components as shown below.



Specifying Component Properties

1. In the model workspace, double-click the **Gear Components** subsystem.
2. Specify the following parameter values for the subsystem components:

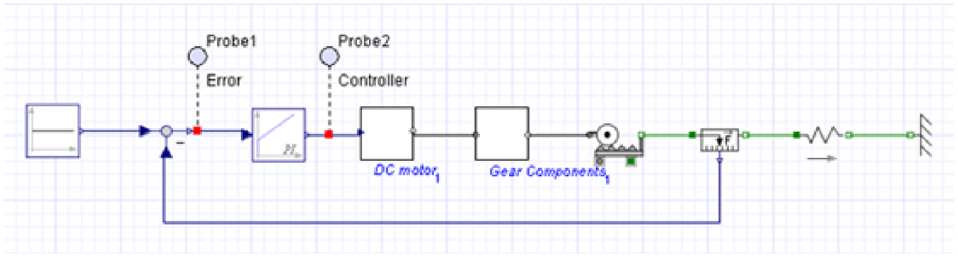
- For the **Ideal Gear** component, change the transmission ratio value to **0.01**.
- For the **Inertia** component, change the moment of inertia value to **0.1** $kg \cdot m^2$.

3. Click **Main** in the navigation toolbar to browse to the top level of the model.
4. Specify the following parameter values for the other components:

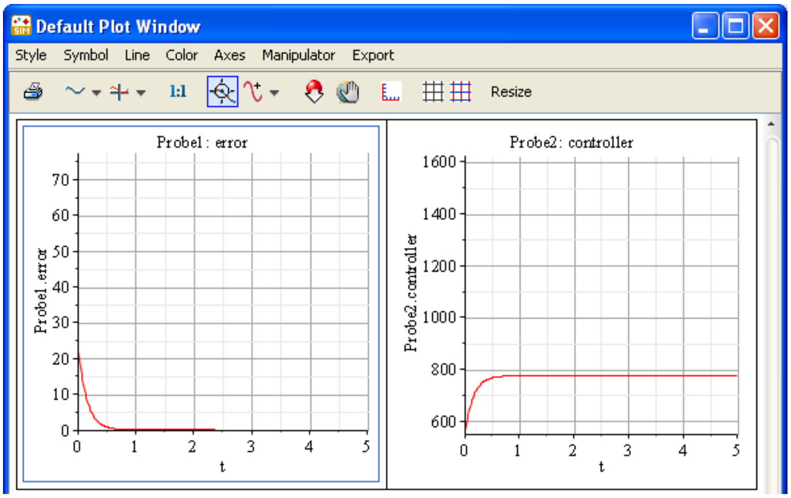
- For the **Translational Spring** component, in the **c** field, change the spring constant value to $\mathbf{210 \cdot 10^9}\ \frac{N}{m}$.
- For the **PI** controller, change the **T** value to **0.1** *s*.
- For the **Constant** source, in the **k** field, change the constant output value to **77.448**.

Simulating the Cable Tension Controller

1. Click the probe button ( ).
2. Click the line that connects the **Feedback** and **PI** components and then click the probe to position it.
3. Select the probe in the model workspace.
4. In the **Inspector** tab, select the **Real** quantity and change its name to **Error**.
5. Add another probe that measures the **Real** quantity to the line connecting the **PI** component and **DC** motor subsystem. Change the quantity name to **Controller**.



6. Click a blank area in the model workspace.
7. In the **Settings** tab, specify a $\mathbf{t}_f$ value of **5** seconds and set **adaptive** to **true**.
8. Click the simulation button ( ) in the main toolbar. When the simulation is complete, the following graphs appear.



9. Save the file as **CableTension.msim**.

6.3 Tutorial 3: Modeling a Nonlinear Damper

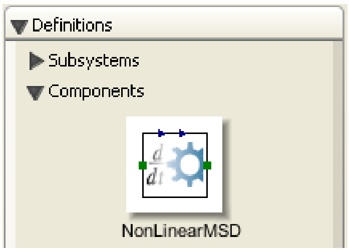
In this tutorial, you will model a nonlinear damper with a linear spring. This tutorial builds upon the concepts demonstrated in the previous tutorials. You will perform the following tasks:

1. Generate a custom spring damper defined by differential equations.
2. Provide custom damping coefficient values as input signals.
3. Build the nonlinear damper with linear spring model.
4. Assign a variable to a subsystem.
5. Simulate the nonlinear damper with linear spring model.

Generating a Custom Spring Damper

In MapleSim, you can create a custom component that is based on a mathematical model. Typically, you would define the component equations and ports before making the component available in MapleSim. For the the purpose of this tutorial, you will generate a sample custom component with pre-defined differential equations.

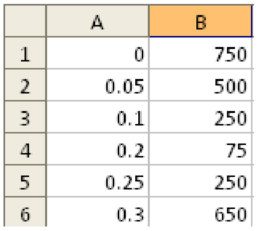
1. Open a new MapleSim document.
2. In the main toolbar, click the templates button ( ) .
3. Click **Browse...**
4. In the **Browse Templates** dialog box, open the **Component Templates** folder.
5. Select the **NonLinearMSD.mw** file and click **Use Template**.
6. In the **Attachment** field, enter **NonLinearMSD**.
7. Click **Create Attachment**. The Custom Component Template is opened in Maple.
8. To generate the custom component, click **Generate MapleSim Component** at the bottom of the worksheet. In MapleSim, the **NonLinearMSD** component appears in the **Definitions** palette, in the **Project** tab, on the left side of the MapleSim window. You will use this component later in this tutorial.



Providing Damping Coefficient Values

You can provide custom values for interpolation table components that you add to your model. In this example, you will provide damping coefficient values in an external file.

1. Create either a Microsoft Excel spreadsheet (.xls) or comma-separated values (.csv) file that contains the following values:



	A	B
1	0	750
2	0.05	500
3	0.1	250
4	0.2	75
5	0.25	250
6	0.3	650

The first column contains values for the relative displacement of the damper and the second column contains values for the damping coefficients.

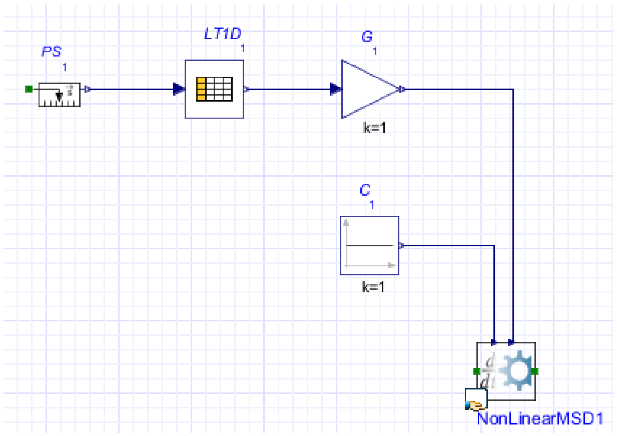
2. Save the file as **DamperCurve.xls** or **DamperCurve.csv**.
3. In MapleSim, expand the **Attachments** palette located in the **Project** tab.
4. Right-click (**Control**-click for Macintosh) **Data Sets** and select **Attach File**.
5. Browse to and select the Excel spreadsheet or .csv file that you created, and click **Attach...** The file containing the data set is attached to your model. You will use this file in the next task.

Building the Nonlinear Damper Model

1. From the **Definitions** palette, drag the **NonLinearMSD** component into the model workspace.
2. Add the following components to the model workspace:

- From the **Signal Blocks** → **Mathematical** → **Operators** menu, add a **Gain** component and place it above the **NonLinearMSD** component.
- From the **Signal Blocks** → **Sources** → **Real** menu, add a **Constant** component and place it between the **NonLinearMSD** and **Gain** components.
- From the **Signal Blocks** → **Interpolation Tables** menu, add a **1D Lookup Table** component and place it to the left of the **Gain** component.
- From the **1-D Mechanical** → **Translational** → **Sensors** menu, add a **Position Sensor** component and place it to the left of the **1D Lookup Table** component.

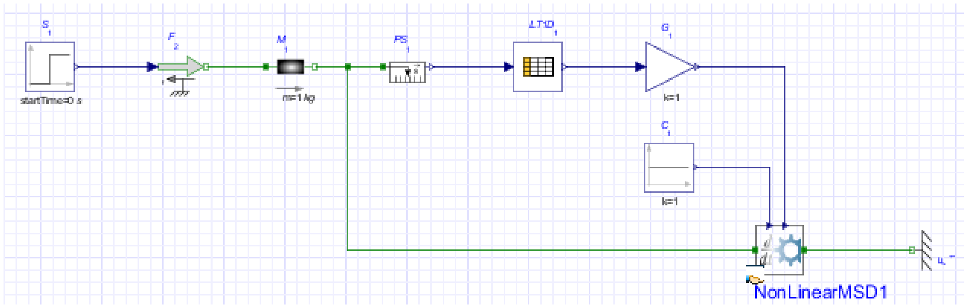
3. Connect the components as shown below.



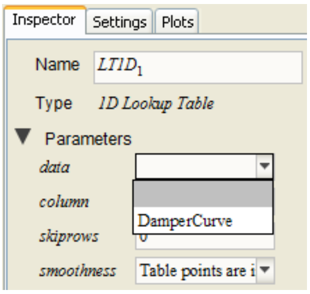
4. Add the following components to the model workspace:

- From the **1-D Mechanical** → **Translational** → **Common** menu, add a **Translational Fixed** component, rotate it counterclockwise, and place it to the right of the **NonLinearMSD** component.
- From the same menu, add **Mass** and **Force** components and place them to the left of the **Position Sensor** component.
- From the **Signal Blocks** →**Sources** → **Real** menu, add a **Step** source.

5. Connect the components as shown below.

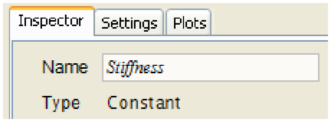


6. In the model workspace, select the **1D Lookup Table** component. In the **Inspector** tab, the **data** drop-down menu lists all of the documents that you have attached to the model.



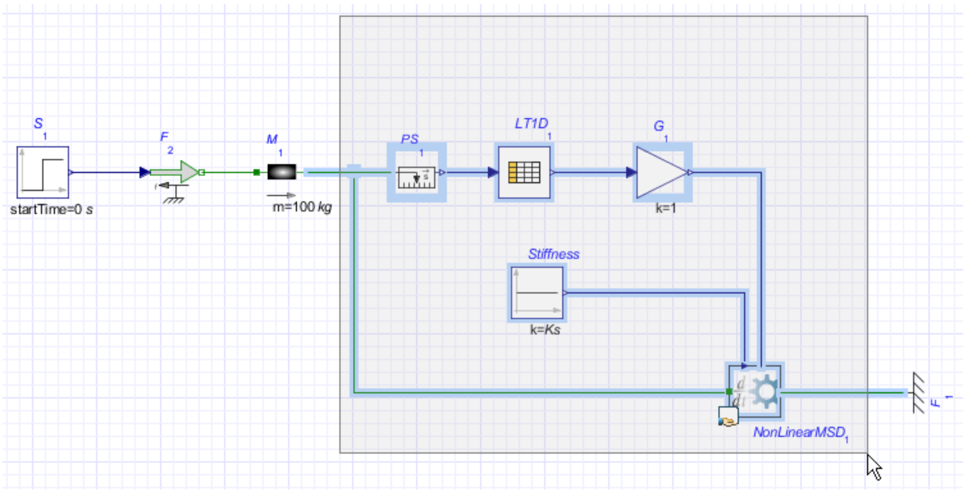
Name *LT1D₁*
Type *1D Lookup Table*
Parameters
data
column
skiprows 0
smoothness Table points are i
DamperCurve

7. Select the **DamperCurve.xls** or **DamperCurve.csv** file that you created in the previous task. You will now define the stiffness of the spring.
8. In the model workspace, select the **Constant** component.
9. In the **Inspector** tab, in the **Name** field, change the component name to **Stiffness**.

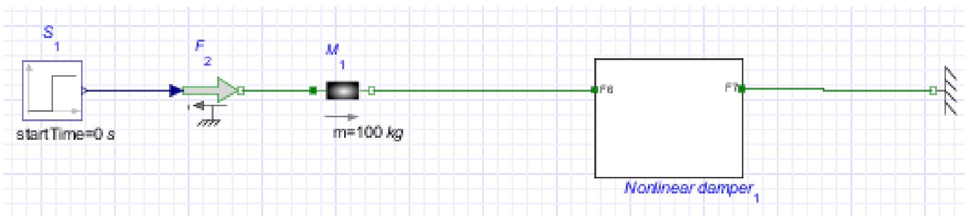


Name *Stiffness*
Type Constant

10. Select the **Step** component and change the step height to **100**.
11. Select the **Mass** component and change the mass to **100** *kg*.
12. Using the selection tool ( ), draw a box around all of the components in the nonlinear damper model.

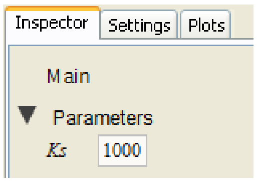


13. Group the selected components into a subsystem called **Nonlinear damper**. The complete model is shown below.



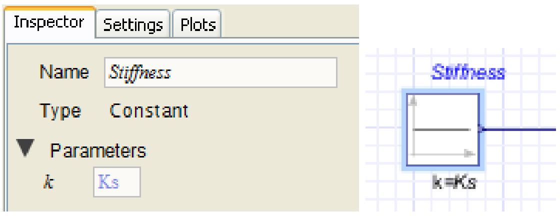
Assigning a Parameter to a Subsystem

1. In the model workspace, double-click the **Nonlinear damper** subsystem.
2. In the navigation toolbar, click the parameter view button ().
3. In the first row of the **Nonlinear damper subsystem default settings** table, define a spring constant parameter called **Ks** and press **Enter**.
4. In the same row, specify a default value of **1000** and enter **Spring constant** as the description. You can now assign the parameter value to other components in the **Nonlinear damper** subsystem.
5. In the navigation toolbar, click the diagram view button (). The **Ks** parameter appears as a field in the **Inspector** tab with the defined default value.



Ks 1000

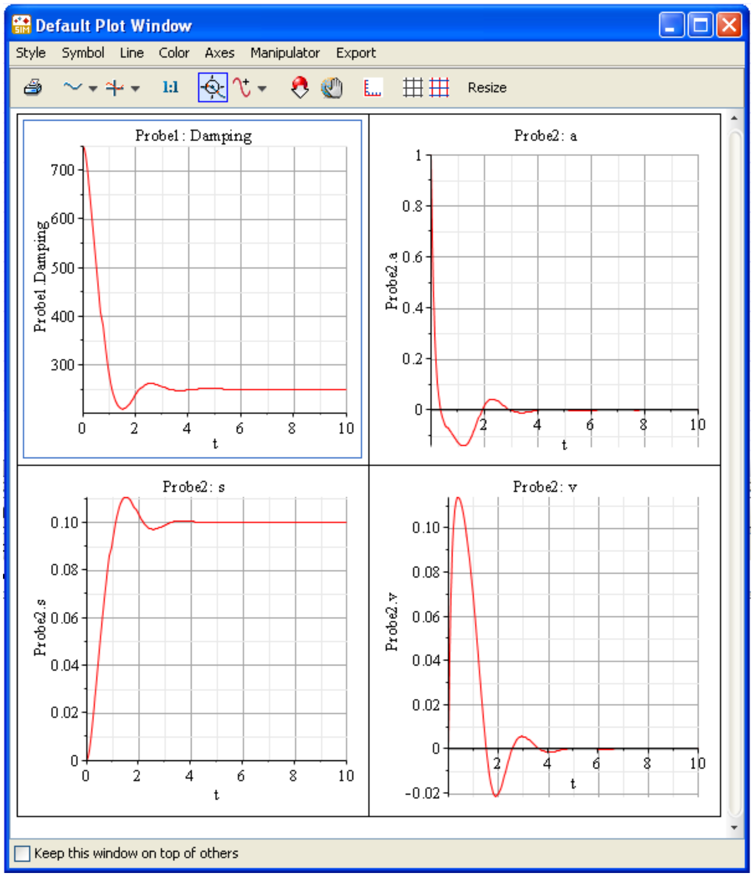
6. In the model workspace, select the **Stiffness** component and change the constant output parameter, **k**, to **Ks**. This component now inherits the numeric value of **Ks** (in this example, **1000**). Therefore, if you edit the numeric value of **Ks** at the subsystem level, the **k** parameter that has been assigned the variable, **Ks**, also inherits that change.



Name Stiffness
Type Constant
k Ks

Simulating the Nonlinear Damper with Linear Spring Model

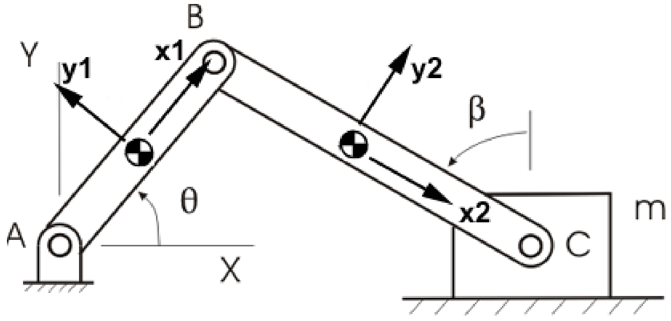
1. From the model workspace toolbar, click the probe button ( ).
2. Click the line that connects the **Gain** and **NonLinearMSD** component and click the probe to position it.
3. In the model workspace, select the probe.
4. In the **Inspector** tab, select the **Real** quantity and change its name to **Damping**.
5. Browse to the top level of the model.
6. To the line that connects the **Mass** component and the **Nonlinear damper** subsystem, add a probe that measures the length, speed, and acceleration quantities.
7. Click a blank area in the model workspace.
8. In the **Settings** tab, set the $\mathbf{t_f}$ parameter to **10** seconds.
9. Click the simulation button ( ) in the main toolbar. When the simulation is complete, the following graphs appear.



10. Save the file as **NonLinearMSD.msim**.

6.4 Tutorial 4: Modeling a Planar Slider-Crank Mechanism

Using components from the Multibody mechanical library, you will model the planar slider-crank mechanism shown in the following schematic diagram.



This model consists of a revolute joint, *A*, which is attached to a planar link. This planar link is attached to a connecting rod by a second revolute joint, *B.* The connecting rod connects to a sliding mass by a third revolute joint, *C*, and the sliding mass connects to ground by a prismatic joint. In practice, this mechanism converts rotational motion at the crank to translational motion at the sliding mass or vice versa. For the system shown in the diagram, gravity is assumed to be the only external force, acting along the negative Y axis (the y axis for the inertial frame).

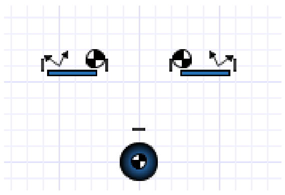
In this tutorial, you will perform the following tasks:

1. Create a planar link subsystem
2. Define and assign subsystem parameters.
3. Create the crank and connecting rod elements.
4. Add the fixed frame, sliding mass, and joint elements to the model.
5. Specify initial conditions.
6. Simulate the planar slider-crank mechanism.

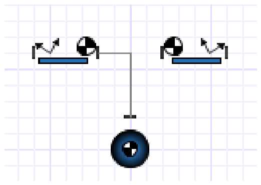
Creating a Planar Link Subsystem

The diagram above shows that the slider-crank has two associated planar links: the crank (the link from point A to B) and the connecting rod (the link from B to C). In both cases, these links have their longitudinal axis along their local x axis (**x1** and **x2**, respectively). Therefore, you will first create a generic planar link with two ports. The inboard port (base) will be located $-\frac{L}{2}$ units along the x axis of the link, and the outboard port (tip) will be located $\frac{L}{2}$ units along the x axis of the link. In this example, L refers to the length of the link and the center-of-mass is assumed to be in the middle of the link.

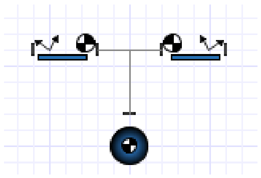
1. Open a new MapleSim document.
2. From the **Multibody → Bodies and Frames** menu in the **Libraries** tab, add two **Rigid Body Frame** components and a **Rigid Body** component.
3. In the model workspace, right-click (**Control**-click for Macintosh) the **RBF$_1$** component and select **Flip Horizontal**.
4. Right-click (**Control**-click for Macintosh) the **Rigid Body** component and select **Rotate Counter Clockwise**.
5. Drag the components in the arrangement shown below. You can now connect the components.



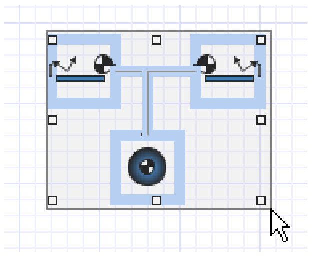
6. Draw a connection line between the **Rigid Body** component and the right frame of the **RBF$_1$** component.



7. Draw another connection line between the **Rigid Body** component and the left frame of the **RBF$_2$** component.



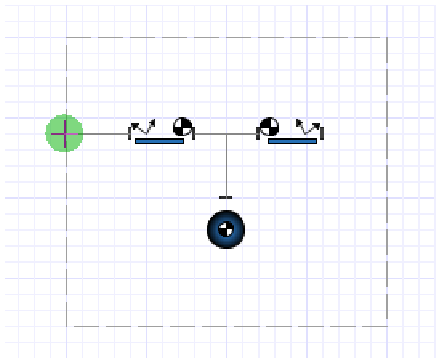
8. Using the selection tool ( ), draw a box around the components.



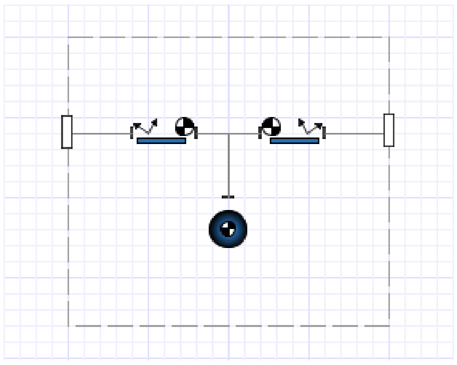
9. From the **Edit** menu (or right-click the highlighted components) select **Create Subsystem**.
10. In the **Create Subsystem** dialog box, enter **Link** and click **OK**.

You will now add ports to connect this subsystem to other components.

11. Double-click the **Link** subsystem.
12. Click the left frame of the **RBF$_1$** component and drag your mouse pointer to the left of the subsystem boundary.



13. Click the line once. A subsystem port is added.
14. In the same way, using the right frame of the **RBF$_2$** component, create another port on the right side of the subsystem boundary.



Defining and Assigning Parameters

In this task, you will define a subsystem parameter, *L*, to represent the length of the link and assign the parameter value as a variable to the parameters of the **Rigid Body Frame** components. The **Rigid Body Frame** components will then inherit the numeric value of *L*.

1. Double-click the **Link** subsystem and in the navigation toolbar, click the parameter editor button (), or from the **Inspector** tab click **Add or Change Parameters**. The **Link subsystem default settings** window appears.
2. In the first row of the **Link subsystem default settings** table, define a parameter called **L**, and press **Enter**.
3. Specify a default value of **1m** and enter **Length** as the description.
4. Specify the following parameter values (to enter a fraction, use the forward slash key(/)):

- For the **RBF$_1$** component, in the $\overline{r}_{XYZ}$ field, specify a position offset of $\left[-\frac{L}{2},0,0\right]$.
- For the **RBF$_2$** component, in the $\overline{r}_{XYZ}$ field, specify a position offset of $\left[\frac{L}{2},0,0\right]$.

Creating the Crank and Connecting Rod Elements

In this task, to create the crank and connecting rod elements, you will add a **Link** subsystem definition to your model and create **Crank** and **ConnectingRod** shared subsystems. You will also assign a different length value to the connecting rod element.

1. Click **Main** in the navigation toolbar to browse to the top level of your model. The **Link** subsystem appears in the model workspace.
2. Right-click (**Control**-click for Macintosh) the **Link** subsystem and select **Convert to Shared Subsystem**. The **Create Shared Subsystem** window appears. Click **OK**. A

Link subsystem definition is added to the **Definitions** palette in the **Project** tab and the **Link** subsystem in the model workspace is converted to a shared subsystem.

3. Select the **Link** shared subsystem in the model workspace and in the **Inspector** tab, in the **Name** field, change the name of the shared subsystem to **Crank**.
4. From the **Definitions, Subsystems** palette, drag the **Link** icon to the model workspace and place it to the right of the **Crank** shared subsystem.
5. In the model workspace, select the second copy of the **Link** shared subsystem.
6. In the **Inspector** tab, change the shared subsystem name to **ConnectingRod** and change the length value to **2**.

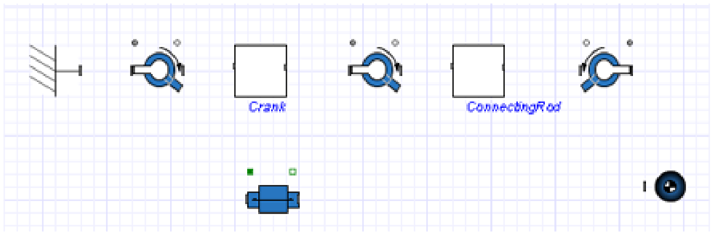
Adding the Fixed Frame, Sliding Mass, and Joint Elements

In this task, you will add a **Fixed Frame** component, a **Rigid Body** component to represent the sliding mass, and the joint components.

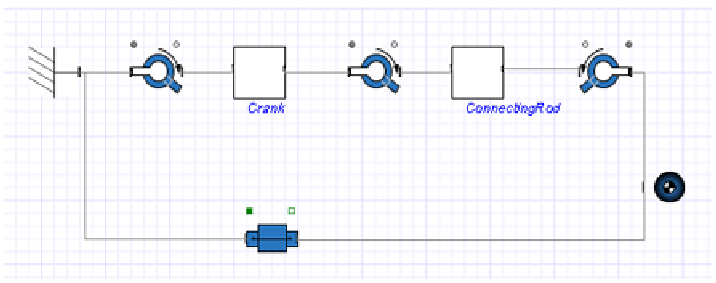
1. From the **Multibody** → **Bodies and Frames** Component Library menu, select the **Fixed Frame** component and place it to the left of the **Crank** shared subsystem.
2. From the same menu, select the **Rigid Body** component and place it slightly below and to the right of the **Connecting Rod** shared subsystem.
3. Add the following joints:

- From the **Multibody** → **Joints and Motions** menu, add a **Revolute** joint between the **Fixed Frame** component and the crank, a second **Revolute** joint between the crank and the connecting rod, and a third **Revolute** joint between the connection rod and the sliding mass.
- From the same menu, add a **Prismatic** joint and place it below the **Crank** subsystem.

4. Select the **RB$_2$** component in the model workspace and rename it **Sliding Mass**.
5. Right-click (**Control**-click for Macintosh) the **Sliding Mass** component and select **Flip Horizontal.** In the same way, flip the the **R$_3$** revolute joint horizontally.



6. Connect the components as shown below.



Tip: In this example, the default axes of motion for the revolute and prismatic joints line up with the desired axes of motion. For example, the revolute joints initially assume that they rotate about the z axis of the inboard frame, which always coincides with the inertial Z axis for XY-planar systems. If you create nonplanar models, you may need to change these axes to make sure that they allow motion along or about the correct directions.

Specifying Initial Conditions

To specify initial conditions, you can set parameter values for certain components in your model.

1. For the first revolute joint, in the $\boldsymbol{\theta_0}$ field, change the initial angle to $\frac{\pi}{4}$ *rad*.

Tip: To enter π, type **Pi** press **Ctrl** + **Space** (or **Command** + **Shift** + **Space** for Macintosh), and select the π symbol from the menu.

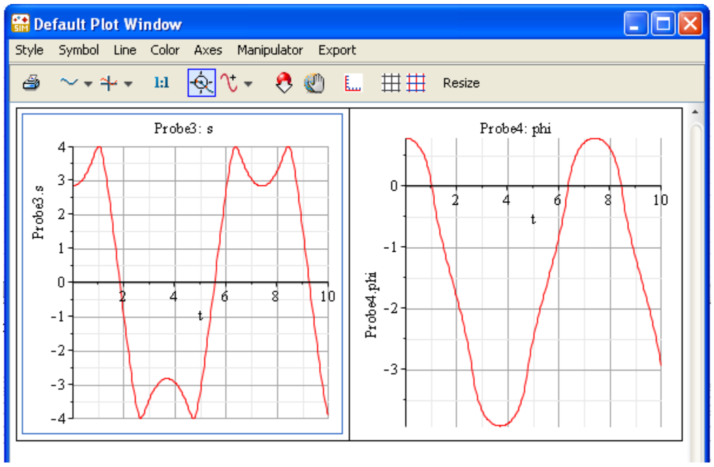
2. From the $\mathbf{IC_{\theta,\omega}}$ drop-down menu, select **Strictly Enforce**.

When MapleSim solves for the initial conditions, the first angle will be set to $\frac{\pi}{4}$ *rad* before the angles are set for the other joints.

Simulating the Planar Slider-Crank Mechanism

1. From the model workspace toolbar, click the probe button ( ).
2. In the model workspace, click the white 1-D translational flange (flange_b) at the top right of the **Prismatic** component icon and position the probe .
3. Click the probe in the model workspace.
4. In the **Inspector** tab, select the **Length** quantity to measure the displacement.
5. In the same way, add a probe that measures the **Angle** quantity to the white 1-D rotational flange (flange_b) at the top right of the $\mathbf{R_1}$ component icon.
6. Click a blank area in the model workspace.

7. In the **Settings** tab, expand **Solver** and set the $\mathbf{t_f}$ parameter to **10** seconds.
8. Click the simulation button (▶) in the main toolbar. When the simulation is complete, the following graphs appear.



9. Save the file as **SliderCrank.msim**.

7 Reference: MapleSim Keyboard Shortcuts

Opening, Closing, and Saving a Model

Task	Windows and Linux	Macintosh
Create a new model	**Ctrl** + **N**	**Command** + **N**
Open an existing model	**Ctrl** + **O**	**Command** + **O**
Close the active document	**Ctrl** + **F4**	**Command** + **W**
Save a model as an .msim file	**Ctrl** + **S**	**Command** + **S**

Building a Model in the Block Diagram View

Task	Windows and Linux	Macintosh
Rotate the selected modeling component 90 degrees clockwise	**Ctrl** + **R**	**Command** + **R**
Rotate the selected modeling component 90 degrees counter-clockwise	**Ctrl** + **L**	**Command** + **L**
Flip the selected modeling component vertically	**Ctrl** + **F**	**Command** + **F**
Flip the selected modeling component horizontally	**Ctrl** + **H**	**Command** + **K**
Group the selected modeling components into a subsystem	**Ctrl** + **G**	**Command** + **G**
Display or hide probes in the model workspace	**Ctrl** + **D**	**Command** + **D**

Browsing a Model in the Block Diagram View

Task	Windows and Linux	Macintosh
View the selected modeling component or subsystem in detail	**Ctrl** + **M**	**Command** + **M**
Zoom into the model workspace	**Ctrl** + numeric keypad **plus** key	**Command** + numeric keypad **plus** key
Zoom out from the model workspace	**Ctrl** + numeric keypad **minus** key	**Command** + numeric keypad **minus** key

Browsing a Model in the 3-D View

Task	Windows and Linux	Macintosh
Move the camera around a 3-D model in the perspective view	**Ctrl** + left mouse button click and drag	**Command** + mouse click and drag
Panning a 3-D model	**Shift** + left mouse button click and drag	**Shift** + mouse click and drag

Task	**Windows and Linux**	**Macintosh**
Zoom into or out from the 3-D workspace	**Alt** + left mouse button click and drag, or move the mouse wheel forward (zoom in), or backward (zoom out).	**Alt** + mouse click and drag, or mouse wheel

Glossary

Term	Description
2-D math notation	Formatting option that allows you to enter mathematical text, such as superscripts, subscripts, and Greek characters.
3-D workspace	The area of the MapleSim window in which you can build and edit a 3-D model.
Attached shapes	Shapes that you can display in a 3-D model to create a realistic representation of a system model. Attached shapes include cylinders, trace lines, and CAD geometry that you import from another file.
Camera	The point of view from which a 3-D scene is viewed.
Camera tracking	The process by which a camera follows the movement of a target 3-D component that you select. The target component is centered in the 3-D workspace during an animation.
Custom component	A user-defined component that you can create and add to a MapleSim model using the Custom Component Template.
Custom library	A collection of modeling components and subsystems that can be saved in a user-defined palette and used in a future MapleSim session.
Embedded component	Configurable graphical controls, buttons, meters, and other interactive components that you can add to a Maple standard worksheet to analyze, manipulate, and visualize equations and Maple commands.
Implicit geometry	Default cylinders and spheres that are displayed in a 3-D model to represent modeling components.
Maple package	A collection of routines or commands that can be used in Maple. Most Maple packages provide a set of commands for a particular mathematical or scientific domain, or field of study.
MapleSim component library	The default collection of domain-specific modeling components included in MapleSim. These modeling components can be found in the gray palettes in the **Libraries** tab.
Model workspace	The area of the MapleSim window in which you can build and edit a model in a block diagram view.
Orthographic view	A type of 3-D view that uses parallel projection and displays lines in the view plane at their "true length." In MapleSim, you can view a model from front, top, and side orthographic views.
Perspective view	A 3-D view that allows you to examine and browse a model from any direction in 3-D space.

Term	**Description**
Probe	The tool used to identify quantities of interest in order to simulate a MapleSim model.
Shared subsystem	A subsystem copy that shares the same configuration as other subsystems. All shared subsystems are linked to a particular *subsystem definition*, which defines the configuration.
Stand-alone subsystem	A subsystem that is not linked to a *subsystem definition* and can be edited and manipulated independent of other subsystems in a model.
Subsystem	A collection of modeling components grouped in a single block.
Subsystem definition	A subsystem block that defines the configuration for a series of *shared subsystems*.

Index