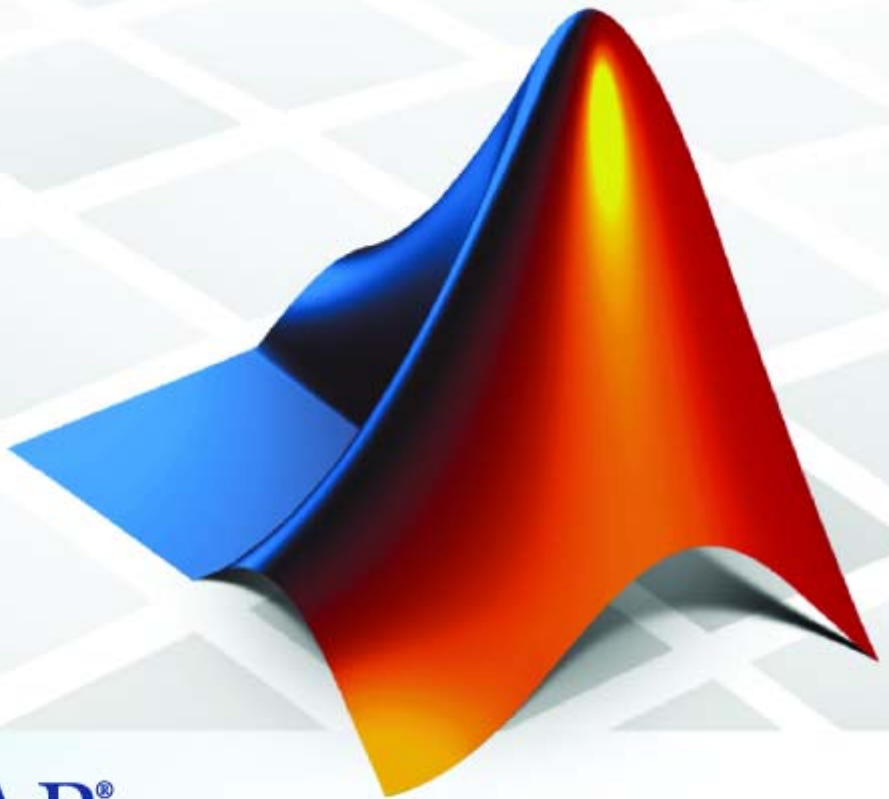


# Gauges Blockset 2

## User's Guide



**MATLAB<sup>®</sup>**  
& **SIMULINK<sup>®</sup>**

## How to Contact The MathWorks



[www.mathworks.com](http://www.mathworks.com)  
[comp.soft-sys.matlab](mailto:comp.soft-sys.matlab)  
[www.mathworks.com/contact\\_TS.html](http://www.mathworks.com/contact_TS.html)

Web  
Newsgroup  
Technical Support



[suggest@mathworks.com](mailto:suggest@mathworks.com)  
[bugs@mathworks.com](mailto:bugs@mathworks.com)  
[doc@mathworks.com](mailto:doc@mathworks.com)  
[service@mathworks.com](mailto:service@mathworks.com)  
[info@mathworks.com](mailto:info@mathworks.com)

Product enhancement suggestions  
Bug reports  
Documentation error reports  
Order status, license renewals, passcodes  
Sales, pricing, and general information



508-647-7000 (Phone)



508-647-7001 (Fax)



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

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

*Gauges Blockset User's Guide*

© COPYRIGHT 1999–2007 by The MathWorks, Inc.

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

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

### Trademarks

MATLAB, Simulink, Stateflow, Handle Graphics, Real-Time Workshop, SimBiology, SimHydraulics, SimEvents, and xPC TargetBox are registered trademarks and The MathWorks, the L-shaped membrane logo, Embedded MATLAB, and PolySpace are trademarks of The MathWorks, Inc.

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

### Patents

The MathWorks products are protected by one or more U.S. patents. Please see [www.mathworks.com/patents](http://www.mathworks.com/patents) for more information.

### **Revision History**

September 1999	Online only	New for Version 1.0
September 2000	First printing	Revised for Version 1.1 (Release 12)
May 2001	Online only	Revised for Version 1.1.1 (Release 12.1)
July 2002	Second printing	Revised for Version 1.1.2 (Release 13)
June 2004	Online only	Revised for Version 1.2 (Release 14)
October 2004	Third printing	Revised for Version 2.0 (Release 14SP1) (New title)
March 2005	Online only	Revised for Version 2.0.1 (Release 14SP2)
September 2005	Online only	Revised for Version 2.0.2 (Release 14SP3)
March 2006	Online only	Revised for Version 2.0.3 (Release 2006a)
September 2006	Online only	Revised for Version 2.0.4 (Release 2006b)
March 2007	Online only	Revised for Version 2.0.5 (Release 2007a)
September 2007	Online only	Revised for Version 2.0.5 (Release 2007b)



## Getting Started

**1**

<b>What Is Gauges Blockset?</b> .....	<b>1-2</b>
<b>Related Products</b> .....	<b>1-3</b>
External Mode Support .....	<b>1-3</b>
Real-Time Workshop Support .....	<b>1-3</b>
<b>Installing Gauges Blockset</b> .....	<b>1-4</b>
Installing and Confirming Installation .....	<b>1-4</b>
Troubleshooting the Gauges Blockset Installation .....	<b>1-4</b>
<b>Accessing the Preconfigured Blocks</b> .....	<b>1-5</b>
Using the gaugeslib Command in MATLAB .....	<b>1-5</b>
Using the Simulink Library Browser .....	<b>1-6</b>
<b>Moving and Selecting Blocks</b> .....	<b>1-9</b>
<b>Example: Building a Simple Model</b> .....	<b>1-10</b>
Overview .....	<b>1-10</b>
The Original Simulink Model .....	<b>1-10</b>
Replacing Simulink Blocks with Gauges .....	<b>1-11</b>
Building the Model .....	<b>1-11</b>
Running the Simulation .....	<b>1-12</b>
Saving the Model .....	<b>1-13</b>
Printing the Model .....	<b>1-13</b>
<b>Modifying Properties of Blocks</b> .....	<b>1-15</b>
Accessing the Properties .....	<b>1-15</b>
Example of Modifying Properties .....	<b>1-16</b>
Learning More About Properties .....	<b>1-17</b>

## Using Gauges in a Model

### 2

<b>Connecting Blocks in a Model</b> .....	<b>2-2</b>
<b>Modifying ActiveX Control Properties</b> .....	<b>2-3</b>
Overview .....	<b>2-3</b>
Using Multiple Styles Within One Block .....	<b>2-3</b>
Understanding ID Properties .....	<b>2-7</b>
Displaying Text on a Block .....	<b>2-9</b>
Modifying the Displayed Range .....	<b>2-11</b>
Modifying Multiple Tick Marks .....	<b>2-13</b>
<b>Controlling Multiple Graphical Elements</b> .....	<b>2-18</b>
Overview .....	<b>2-18</b>
Simulating a Multiple-Needle Stopwatch .....	<b>2-18</b>
Updating Multiple Portions of a Pie Chart .....	<b>2-23</b>
<b>Saving and Reusing a Customized Control</b> .....	<b>2-30</b>
Saving Customized Controls Automatically .....	<b>2-30</b>
Saving Customized Controls Using the Library Panel .....	<b>2-30</b>

## Categories of ActiveX Controls

### 3

<b>Angular Gauges</b> .....	<b>3-2</b>
Library Overview .....	<b>3-2</b>
Customizing Angular Gauges .....	<b>3-2</b>
<b>LEDs</b> .....	<b>3-5</b>
Library Overview .....	<b>3-5</b>
Customizing LEDs .....	<b>3-5</b>
<b>Linear Gauges</b> .....	<b>3-7</b>
Library Overview .....	<b>3-7</b>
Customizing Linear Gauges .....	<b>3-7</b>

<b>Numeric Displays</b> .....	<b>3-11</b>
Library Overview .....	<b>3-11</b>
Customizing Numeric Displays .....	<b>3-11</b>
Customizing the Odometer Block .....	<b>3-13</b>
<b>On Off Gauges</b> .....	<b>3-14</b>
Library Overview .....	<b>3-14</b>
Customizing On Off Gauges .....	<b>3-14</b>
<b>Percent Indicators</b> .....	<b>3-15</b>
Library Overview .....	<b>3-15</b>
Customizing Percent Indicators .....	<b>3-15</b>
<b>Strip Chart</b> .....	<b>3-18</b>
<b>Using Your Own ActiveX Control</b> .....	<b>3-20</b>
Adding the ActiveX Control Block to a Model .....	<b>3-20</b>
Notes on Third-Party ActiveX Control Blocks .....	<b>3-21</b>
<b>Block Parameters for the ActiveX Control Block</b> .....	<b>3-24</b>
Summary of Parameters .....	<b>3-24</b>
Program ID .....	<b>3-25</b>
Connections .....	<b>3-25</b>
Input Property .....	<b>3-25</b>
Initialization Command .....	<b>3-26</b>
Other Events and Handlers .....	<b>3-26</b>
Update Command .....	<b>3-27</b>
In-Block Control .....	<b>3-27</b>
Border .....	<b>3-28</b>

## **Placing ActiveX Controls in a Different Window**

### **4**

<b>Placing ActiveX Controls in a Different Model</b> .....	<b>4-2</b>
Example Overview .....	<b>4-2</b>
Creating a Model Window Containing Gauges .....	<b>4-2</b>
Associating the Main Model with the Gauges .....	<b>4-4</b>

<b>Placing ActiveX Controls in a Subsystem</b> .....	<b>4-7</b>
Example Overview .....	<b>4-7</b>
Creating a Subsystem Containing Gauges .....	<b>4-7</b>
Associating Top-Level Blocks with the Subsystem .....	<b>4-8</b>
<b>Placing ActiveX Controls in a Figure Window</b> .....	<b>4-10</b>
Example Overview .....	<b>4-10</b>
Creating Helper M-Files and Building the Model .....	<b>4-10</b>
Saving and Reopening the Model .....	<b>4-13</b>

## Blocks — Alphabetical List

---

**5**

### Examples

---

**A**

<b>Getting Started</b> .....	<b>A-2</b>
<b>Modifying ActiveX Control Properties</b> .....	<b>A-2</b>
<b>Controlling Multiple Graphical Elements</b> .....	<b>A-2</b>
<b>Placing ActiveX Controls in Different Windows</b> .....	<b>A-2</b>

### Index

---



# Getting Started

---

What Is Gauges Blockset? (p. 1-2)	The blockset and its typical applications
Related Products (p. 1-3)	MathWorks products related to this blockset
Installing Gauges Blockset (p. 1-4)	Installing the blockset and registering its ActiveX controls
Accessing the Preconfigured Blocks (p. 1-5)	How to access blocks from their library window or from the Simulink® Library Browser
Moving and Selecting Blocks (p. 1-9)	How to move and select blocks in Gauges Blockset
Example: Building a Simple Model (p. 1-10)	How to build and work with an example model using blocks from Gauges Blockset
Modifying Properties of Blocks (p. 1-15)	How to modify properties of a preconfigured Gauges Blockset block

## What Is Gauges Blockset?

Gauges Blockset is a collection of blocks that provides graphical displays for monitoring signals in Simulink models. Using Gauges Blockset, you can set up realistic-looking icons that are custom-designed for your Simulink model and visually representative of the environment that you are modeling.

Typical applications of Gauges Blockset include displaying signals as one would see in

- Automobile dashboards
- Airplane cockpits
- Control and process plants
- Communication and power systems
- Medical systems

Gauges Blockset requires MATLAB® and Simulink. It uses Component Object Model (COM) technology and runs only on Microsoft Windows platforms.

Gauges Blockset is extensible. An ActiveX block is provided in which you can place your custom-built controls.

## Related Products

In this section...
“External Mode Support” on page 1-3
“Real-Time Workshop Support” on page 1-3

For information about products related to Gauges Blockset, see <http://www.mathworks.com/products/gauges/related.html>.

### External Mode Support

Support for external mode in Gauges Blockset enables you to incorporate gauges into any target that you can connect to through external mode (such as the xPC Target and Real-Time Windows Target environments; see the documentation for those products for details).

For more information about external mode, see “External Mode” in the Real-Time Workshop® documentation.

### Real-Time Workshop Support

You can use Real-Time Workshop to generate code from models that include blocks from Gauges Blockset.

Gauges are ignored during code generation, except through the use of external mode (see above). If you want to view the gauges, you can do so through the external mode in Real-Time Workshop.

## Installing Gauges Blockset

In this section...
“Installing and Confirming Installation” on page 1-4
“Troubleshooting the Gauges Blockset Installation” on page 1-4

### Installing and Confirming Installation

To build and run the models in this manual, you must first install Simulink and Gauges Blockset. You can find instructions for installing these products in the MATLAB installation documentation for your platform.

To determine what products are installed on your system, enter `ver` in the MATLAB Command Window. This displays information about the version of MATLAB you are running, including a list of all toolboxes and blocksets installed on your system.

### Troubleshooting the Gauges Blockset Installation

Normally, the installation process automatically registers the ActiveX controls associated with Gauges Blockset. However, in exceptional cases you might see an error message referring to an `.ocx` component, similar to the following message:

```
Copying Gauges Blockset files
ads.ocx self registering file did not register
```

If you see such a message, or if the graphical icons do not appear on the blocks in this blockset, then enter `gauges_register_ocx` in the MATLAB Command Window.

## Accessing the Preconfigured Blocks

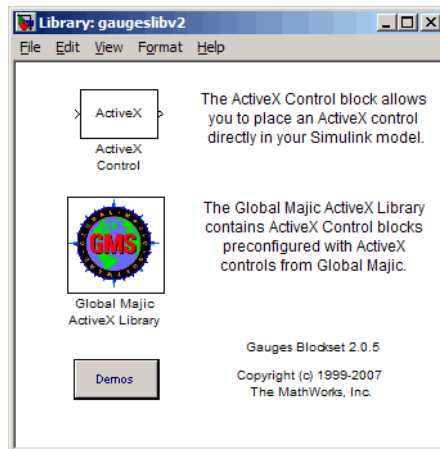
### In this section...

“Using the gaugeslib Command in MATLAB” on page 1-5

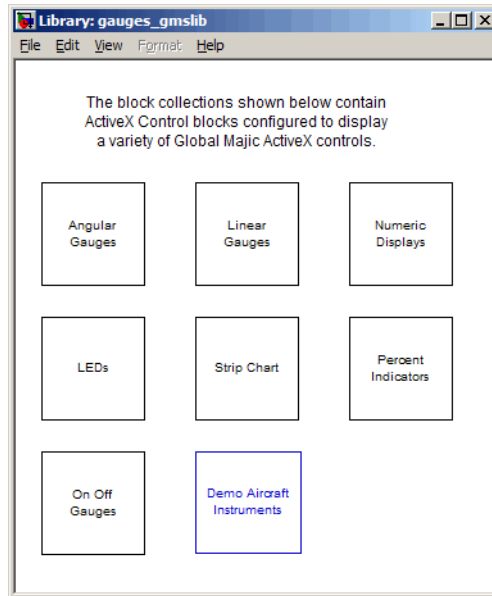
“Using the Simulink Library Browser” on page 1-6

## Using the gaugeslib Command in MATLAB

- 1 Enter the `gaugeslib` command in the MATLAB Command Window, which causes the following window to appear.



- 2 Double-click the Global Majic ActiveX Library icon to access the libraries it contains.



- 3 Double-click an icon to access the blocks in the library that the icon represents. If all the blocks say “ActiveX” and do not look like graphical displays, then follow the instructions in “Troubleshooting the Gauges Blockset Installation” on page 1-4. Each library also includes a question-mark block that provides access to online help for the ActiveX controls in that library.

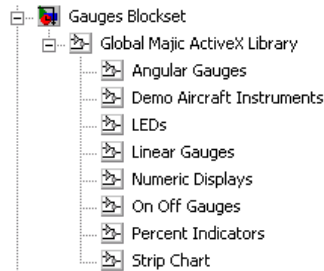
## Using the Simulink Library Browser

As an alternative to the `gaugeslib` command, you can use the Simulink Library Browser to access the preconfigured blocks:

- 1 Open Gauges Blockset by clicking the plus sign to the left of the blockset name. This displays the listing for the Global Majic ActiveX Library.



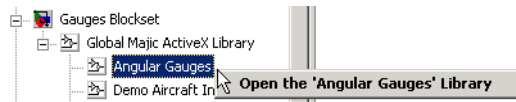
- 2 Open the Global Majic ActiveX Library to display its libraries of blocks.



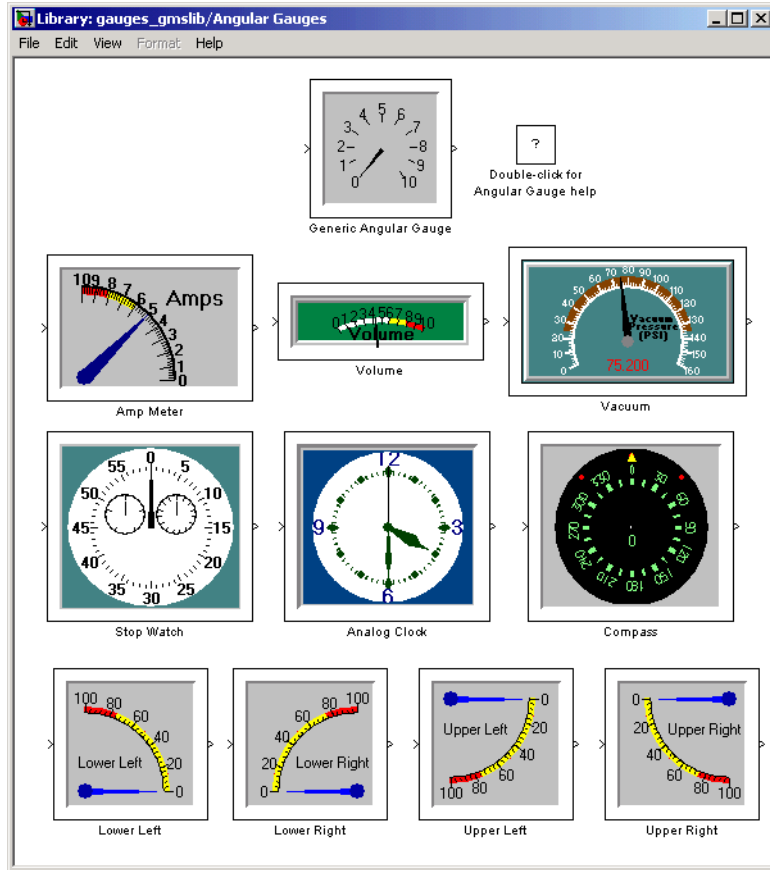
- 3 Select the name of a library to display the library's contents in the right pane of the Simulink Library Browser.

### Opening a Library Window from the Simulink Library Browser

You can also view the blocks as icons in a library window by right-clicking the library name, and then selecting the option that appears. For example, the figure below shows the context menu that appears when you right-click the Angular Gauges listing.



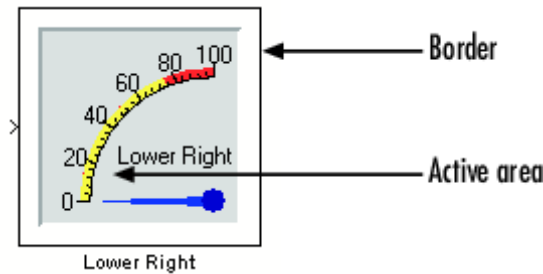
The next figure shows the Angular Gauges library contents as icons in a library window. If the window you see does not look like the figure, then follow the instructions in “Troubleshooting the Gauges Blockset Installation” on page 1-4.





## Moving and Selecting Blocks

The way you move and select blocks from Gauges Blockset differs from how you move and select a Simulink block. Gauges Blockset blocks consist of an “active” area containing the actual ActiveX control, and a border surrounding the active area. The border gives you a way to manipulate the block as a Simulink entity, without affecting the ActiveX control.



The table below describes how to manipulate a Gauges Blockset block.

Task	Mouse Action
Add block to model	From the Simulink Library Browser, drag the block by its icon in the right pane.
	From the library window (displaying blocks as icons), drag the block by its border.
Move block	Drag the block's border. You can do this only if the border is visible.
Select block	Click the block's border. Or “rubber-band select” the block.
Resize block	Select the block, and then drag one of the selection handles (as you would resize a Simulink block).

## Example: Building a Simple Model

### In this section...

“Overview” on page 1-10

“The Original Simulink Model” on page 1-10

“Replacing Simulink Blocks with Gauges” on page 1-11

“Building the Model” on page 1-11

“Running the Simulation” on page 1-12

“Saving the Model” on page 1-13

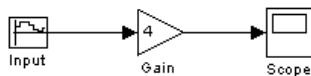
“Printing the Model” on page 1-13

### Overview

This section illustrates how to build and use a simple system, first using Simulink blocks alone, and then using a block from Gauges Blockset. By building the latter model, you can practice finding and using a block from Gauges Blockset. By comparing the two models, you can get a better sense of how graphical icons might enhance the look, feel, and usability of your own models.

### The Original Simulink Model

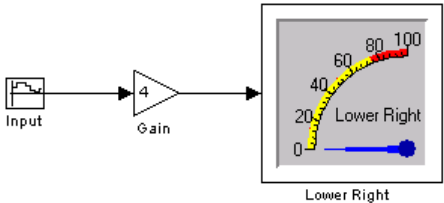
Consider a system in which a Sine Wave block feeds into a Gain block, while a Scope block displays the output from the Gain block. All three of these blocks are part of Simulink.



If you simulate this system and double-click the Scope block, then the Scope traces the value of its input signal over time.

## Replacing Simulink Blocks with Gauges

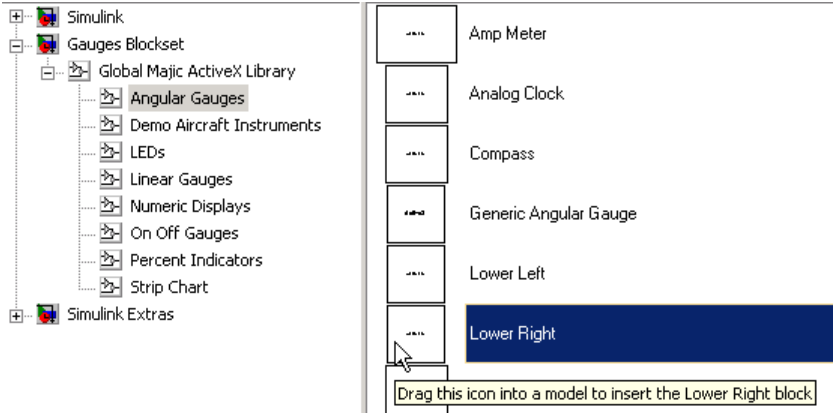
Using Gauges Blockset, you can replace the Scope block from Simulink with a realistic-looking display. For example, a Lower Right block can display the sine wave value at each instant during the simulation.



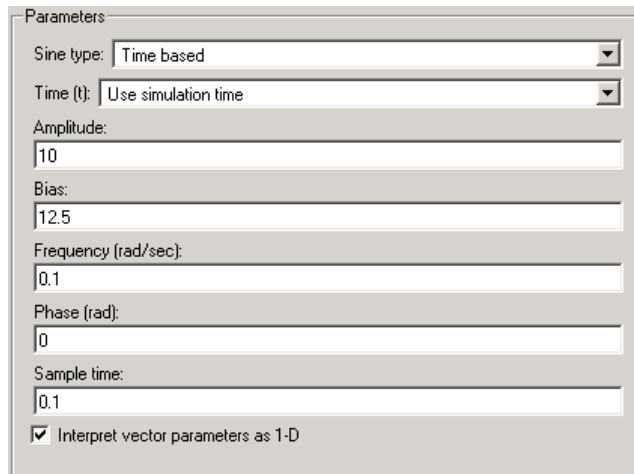
## Building the Model

To build the model described earlier, follow the steps below. Alternatively, enter `gauges_simple` in MATLAB to open a completed copy of the model.

- 1 Open the Simulink Library Browser and create a new model window.
- 2 From the Angular Gauges library, drag the Lower Right block into the model. To do this, select Angular Gauges in the left pane of the Simulink Library Browser, and then drag the Lower Right block from the right pane into the model.



- 3 From the Simulink Sources library, drag the Sine Wave block into the model window.
- 4 Double-click the Sine Wave block and set the block's parameters as indicated below.



The image shows the 'Parameters' dialog box for a Sine Wave block in Simulink. The parameters are as follows:

Parameter	Value
Sine type:	Time based
Time (t):	Use simulation time
Amplitude:	10
Bias:	12.5
Frequency (rad/sec):	0.1
Phase (rad):	0
Sample time:	0.1
Interpret vector parameters as 1-D	<input checked="" type="checkbox"/>

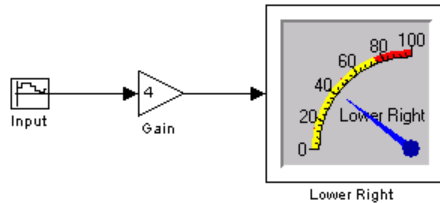
- 5 From the Simulink Math library, drag the Gain block into the model window.
- 6 Double-click the Gain block and change the **Gain** parameter to 4.
- 7 Draw connection lines from the Sine Wave block to the Gain block, and from the Gain block to the Lower Right block.
- 8 From the model window's **Simulation** menu, choose **Configuration parameters**. Set the **Stop time** parameter to Inf.

Now you can run the model and watch how the sine wave affects the needle on the Lower Right block.

## Running the Simulation

Run the simulation by choosing **Start** from the model window's **Simulation** menu. While the simulation is running, you can observe results on the Lower

Right block. This figure shows the model after the needle of the Lower Right block is displaced from its default position.



To stop the simulation, choose **Stop** from the model window's **Simulation** menu.

## Saving the Model

Save the model by choosing **Save** from the model window's **File** menu. When you save a model that contains blocks from Gauges Blockset, MATLAB automatically saves the current state of the ActiveX control that is embedded in the block.

## Printing the Model

You can print the structure of the model by choosing **Print** from the model window's **File** menu. Note that the printing functionality in Simulink does not print the active areas of Gauges Blockset blocks. Instead, it shows only the outline of those blocks.

To capture the exact appearance of a model that contains Gauges Blockset blocks, you can create a .bmp file that represents the model by entering either of these commands in the MATLAB Command Window.

```
print -smodelname -dbitmap filename
```

```
print(['-s', 'modelname'], '-dbitmap', 'filename')
```

Here, `modelname` and `filename` list the names of the Simulink model and the bitmap file, respectively. For example, if the open model is called `sample`, then this command saves it in a file called `samplepic.bmp`.

```
print -ssample -dbitmap samplepic
```

After MATLAB creates the bitmap file, you can insert it into an application that can print it.

## Modifying Properties of Blocks

### In this section...

“Accessing the Properties” on page 1-15

“Example of Modifying Properties” on page 1-16

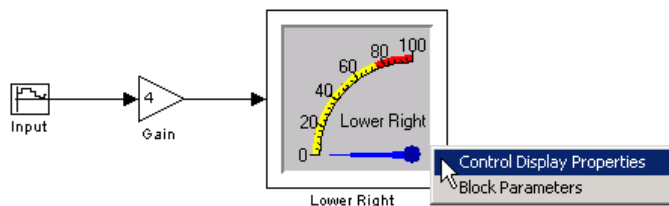
“Learning More About Properties” on page 1-17

### Accessing the Properties

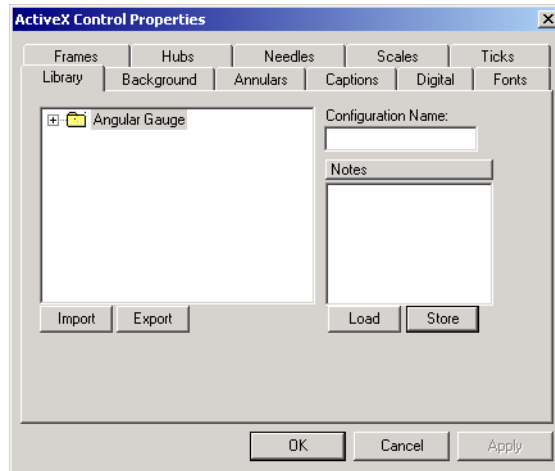
You can view ActiveX control properties by using one of these procedures:

- Double-click the active area of the block that contains the control.
- Right-click the active area of the block and select the **Control Display Properties** option.

This figure shows the context menu that appears when you right-click the Lower Right block.



After you select the **Control Display Properties** option, the ActiveX Control Properties dialog box appears. This dialog box enables you to modify ActiveX control properties. The next figure shows the dialog box for the Lower Right block.



If you modify any values in this dialog box, then the block is visually updated immediately. However, the changes are not permanent until you choose **OK** or **Apply**; if you choose **Cancel**, then the changes will be undone.

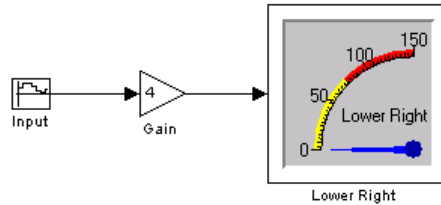
## Example of Modifying Properties

Returning to the model that you built in the section “Example: Building a Simple Model” on page 1-10, you can modify the range of output values by modifying the properties of the Lower Right block. For example, the instructions below change the maximum needle value from 100 to 150.

- 1 Open the ActiveX Control Properties dialog box by double-clicking the active area of the Lower Right block.
- 2 Display the panel that controls the scaling of values by clicking the **Scales** tab.
- 3 Set the **Max** parameter to 150.
- 4 Display the panel that controls tick marks by clicking the **Ticks** tab.
- 5 Set the **TickCount (Major Ticks)** parameter to 3. This prevents the block from looking too crowded.



The figure shows the resulting model.



## Learning More About Properties

Gauges Blockset blocks have many properties. Changing the appearance of a block might require changing several properties and can be quite complex. “Modifying ActiveX Control Properties” on page 2-3 discusses how to make some common changes, such as changing the range of values displayed on a block.

For information about specific properties, consult the ActiveX control’s help by double-clicking the question-mark block that appears in each library of Gauges Blockset. Some libraries provide more than one question-mark block when the blocks contained in the library are significantly different from each other. Once in the Help window, use the Properties link to display information about block properties.



# Using Gauges in a Model

---

Connecting Blocks in a Model (p. 2-2)	How to determine which types of connections a block can have
Modifying ActiveX Control Properties (p. 2-3)	How to change various properties of a block using the ActiveX Control Properties dialog box
Controlling Multiple Graphical Elements (p. 2-18)	How to make a multiple-component gauge display multiple input values simultaneously
Saving and Reusing a Customized Control (p. 2-30)	How to store your customized block properties for later use or to share with other users

# Connecting Blocks in a Model

Before you connect a Gauges Blockset block with other blocks, you should know whether it is meant to be an output device (with an input connection), or a pass-through device (with an input and output connection). Gauges Blockset blocks cannot serve as input devices (with only an output connection). Gauges Blockset blocks are initially drawn with both an inport and an outport, but Simulink removes unused ports when the simulation starts running or when you update the block diagram.

To determine whether a Gauges Blockset block is meant to be used as an output or pass-through device, right-click the block and select the **Block Parameters** option.

---

**Note** If you built your own ActiveX control by customizing the generic ActiveX Control block, then another way to display the custom block's Block Parameters dialog box is to double-click the border of the block.

---

In the Block Parameters dialog box, the **Connections** field determines the type of connection the block currently uses:

- **Input** indicates that the block is a sink; that is, it has an inport and receives a signal. The **Input property** parameter indicates the block's property whose value is changed by the input.
- **Both** indicates that the block has an inport and an outport. The values at both ports are the same. That is, the block becomes a unity function.
- **Neither** indicates that the block has neither an inport nor an outport.

To specify a connection different from the block's default setup, change the block's **Connection** setting and make sure that the **Input property** field is filled in with the appropriate property name. See "Block Parameters for the ActiveX Control Block" on page 3-24 for information about the other fields and check boxes. You can also use the **Help** button to find out about other parameters.

# Modifying ActiveX Control Properties

## In this section...

“Overview” on page 2-3

“Using Multiple Styles Within One Block” on page 2-3

“Understanding ID Properties” on page 2-7

“Displaying Text on a Block” on page 2-9

“Modifying the Displayed Range” on page 2-11

“Modifying Multiple Tick Marks” on page 2-13

## Overview

You can modify many properties of a preconfigured Gauges Blockset block using its ActiveX Control Properties dialog box, introduced in “Accessing the Properties” on page 1-15. This section discusses some of the more complicated tasks and concepts associated with the modification of properties.

For more information about individual properties of the preconfigured blocks, see the online help for the corresponding ActiveX controls. To access such help, open the library window and double-click the question-mark block. The online help summarizes the functionality and contains links to information about properties, events, and methods.

## Using Multiple Styles Within One Block

Some ActiveX control properties let you use more than one style for a given component or characteristic, in the same block. For example, you might use multiple styles to create

- Different font characteristics for text in different places
- Multiple colors within a graphical element such as an annular region or a divided pie chart
- Multiple sets of ticks, each with its own size or labeling characteristics
- Multiple components, such as LEDs or needles, each with its own characteristics

These sections discuss the use of multiple styles in preconfigured Gauges Blockset blocks:

- “Blocks That Use Multiple Styles by Default” on page 2-4
- “Determining When Multiple Styles Are Allowed” on page 2-4
- “Creating Styles” on page 2-5
- “Applying Styles” on page 2-6

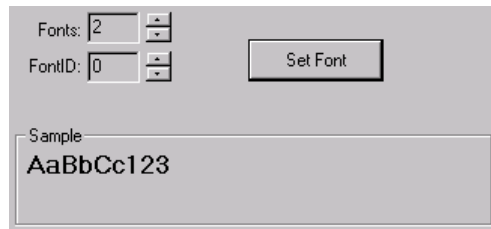
### **Blocks That Use Multiple Styles by Default**

Many Gauges Blockset blocks include multiple styles by default:

- The Vacuum block in the Angular Gauges library uses three text styles: one for the tick labels, one for the number at the bottom of the gauge, and one for the text near the center of the gauge.
- The Volume block in the Angular Gauges library uses three adjacent annular regions, each with a different color.
- The Thermometer block in the Linear Gauges library uses two styles for ticks: one for numbered ticks every 10 degrees and another for unnumbered ticks every 2 degrees.
- The Circle Meter block in the LEDs library applies one of three LED styles to each of 10 LEDs. The three styles differ in their colors.

### **Determining When Multiple Styles Are Allowed**

If a component supports multiple styles, then its property dialog box has properties that enable you to set the number of styles and refer to the styles by number. As an example, the figure below shows how a dialog box supports multiple font styles. The **Fonts** property indicates the number of font styles, while the **FontID** property refers to a given style by number.



To determine whether a component supports multiple styles, look in the block's property dialog box for a pair of properties whose names are

- A plural noun describing the component, such as **Fonts** or **Scales**
- A word that combines the noun and the letters "ID," such as **FontID** or **ScaleID**

If the dialog box has no such properties for the component, then you cannot create multiple styles for that component. For example, in the **Background** panel of a block's dialog box, you can define the color of an outline, but you cannot create multiple concentric outlines of different colors.

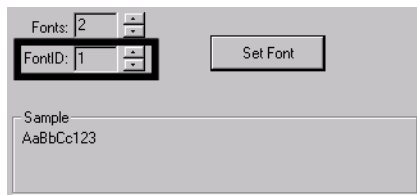
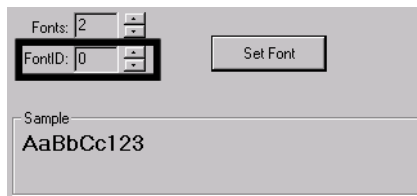
## Creating Styles

After locating the style-identifying pair of properties for the component you are interested in, follow these steps to create an additional style:

- 1 Click once on the up arrow next to the value of the first property in the pair (**Fonts** in the figure). This value is the number of defined styles. If  $N$  styles are defined, then each is associated with an integer between 0 and  $N-1$ . The corresponding ID property (**FontID** in the figure) can assume values between 0 and  $N-1$ .
- 2 Click repeatedly on the up arrow next to the ID property to set it to its maximum value. This causes the dialog box panel to reflect the attributes of that particular style instead of the other defined styles.
- 3 Configure other properties in the dialog box panel to match the attributes that you want that particular style to have. In the figure, the **Set Font** button enables you to set font attributes and the **Sample** box displays text using those attributes. In many cases, all properties in the panel except the original style-identifying pair are attributes of the style. In a few cases,

only part of the panel contains attributes of the style and others are global attributes that apply to all styles.

To view attributes of an existing style, set the ID property to the integer associated with that style. Then, properties on the dialog box panel other than the style-identifying pair reflect attributes of that style. For example, the following figures show how the **Sample** portion of the **Fonts** panel of a dialog box displays either a large bold font or a font of medium size and weight, depending on whether the **FontID** value is set to 0 or 1.

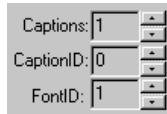


### Applying Styles

In some cases, creating a style implicitly causes the block to apply it. For example, creating an additional style for tick marks automatically creates an additional set of tick marks on the block. In other cases, creating a style does not implicitly cause the block to apply it. For example, even after you create an additional font style, you will not see its effect on the block until you indicate which text should use that style. This section describes how to apply styles that the block does not apply immediately after you create them.



To determine where you can apply a style you have created, look for the corresponding ID property on a panel of the dialog box *other than* the panel where you defined the style. For example, the figure below shows part of a **Captions** panel containing the **FontID** property. The fact that the **FontID** property is not preceded by a **Fonts** property indicates that this is a panel that enables you to apply font styles but not define them.



Once you have located a part of the dialog box where you can apply a style you previously created, set the ID property to match the ID property of that style. For example, the figure above shows that the block has exactly one caption, and that the caption’s font style is the one whose ID is 1. If you change the **FontID** value in the **Captions** panel to a different number, then you will probably notice a change in some text on the block.

## Understanding ID Properties

Many blocks have properties whose names end with **ID**, such as **FontID**, **ScaleID**, and **NeedleID**. Such properties enable you to use more than one style in the same block, as in the situations listed in “Using Multiple Styles Within One Block” on page 2-3. This section describes how to interpret ID property settings. For an example that examines ID property settings among a block’s default settings, see “Modifying Multiple Tick Marks” on page 2-13.

The value of an ID property refers to a style by number. To determine the purpose of the ID property, first see whether the property directly above it is a plural noun similar to the ID property’s name. (For example, see whether the property directly above **FontID** is **Fonts**.) Then,

- If the property directly above the ID property is a plural noun similar to the ID property’s name, then this panel of the dialog box *defines* a set of styles. The ID property associates a number with each style. Other properties in the dialog box panel reflect the definition of the style whose number is the current value of the ID property. By changing the value of the ID property, you can view the definition of a different style.

For example, in the **Fonts** panel of the Volume block, the **FontID** property occurs directly underneath a **Fonts** property. This panel of the dialog box defines font styles, and the **Sample** box displays text using the font style whose number is the current value of the **FontID** property.

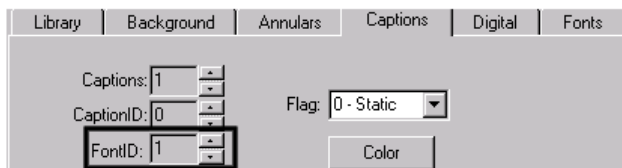
---

**Caution** If you decrease the value of the property named by the plural noun (for example, the **Fonts** property), then the style corresponding to the highest ID value is removed. To replace that style, you have to add a new style and recreate the settings of the deleted style from the default settings.

---

- If the property directly above the ID property is *not* a plural noun similar to the ID property's name, then the ID property *applies* a style that was previously defined in another panel of the dialog box. Other properties in the dialog box panel indicate the context in which the style is applied. By changing the value of the ID property, you can select a different style to apply.

For example, in the **Captions** panel of the Volume block, the **FontID** property does not occur directly underneath a **Fonts** property. The purpose of the **FontID** property in this case is to reference previously defined font styles and apply them to captions. (The font styles are defined on the **Fonts** panel of the dialog box.)



Sometimes, multiple styles are combined so seamlessly that it is not obvious why multiple styles are needed or which parts of the block correspond to which style definitions. You can often adjust the definition of the style to make the style usage more apparent. For example, if you change the colors of different annular regions and then look for the corresponding change in the block, then you should be able to determine how the design is split among multiple annular regions.

## Displaying Text on a Block

Many blocks enable you to include text on the block. Such text might describe the quantity being measured, the units of measurement, or other information. The table below lists some types of text that are associated with a specific part of the block, as well as the part of the ActiveX Control Properties dialog box panel that defines the text. Some types of text apply only to certain blocks.

Type of Text	Part of Dialog Box That Defines or Enables Text
Title appearing in block's outline	<b>Title</b> property on <b>Background</b> panel
Numerical labels near tick marks	<b>Labels</b> area on <b>Ticks</b> panel. On Strip Chart block, <b>Labels</b> properties on <b>Tracks</b> and <b>X Axis</b> panels.
Numerical labels near pointer or needle	<b>Digital</b> panel
Captions appearing anywhere on block	<b>Captions</b> panel

## Using the Captions Panel to Display Text

When it is present, the **Captions** panel of the ActiveX Control Properties dialog box enables you to place text anywhere on the block. Blocks that use text captions by default include Mixer Scale, Tank, Thermometer, Amp Meter, and Volume. This section describes how to add, remove, and change characteristics of text captions using the **Captions** panel.

**Adding Text Captions.** To create a new text caption, follow these steps:

- 1 Increase the value of the **Captions** property by one.
- 2 Set **CaptionID** to its maximum value. This is the index that corresponds to the newest text caption.
- 3 Type the desired text in the **Caption** edit field.

**Removing Text Captions.** To remove the most recently added text caption (that is, the one with the largest **CaptionID** value), decrease the value of the **Captions** property by one. Note that this removes all characteristics of that text caption.

**Changing Fonts and Other Characteristics of Text Captions.** To change the font of an existing text caption, you must create a numbered font style and then apply that style to the caption. Follow these steps:

- 1 Open the **Fonts** panel of the dialog box.
- 2 Allocate space for a new font style by increasing the value of the **Fonts** property by one.
- 3 Set **FontID** to its maximum value. This is the index that corresponds to the newest font style.
- 4 Use the **Set Font** button to select font characteristics.
- 5 Open the **Captions** panel of the dialog box.
- 6 Set **CaptionID** to the index that corresponds to the text caption whose font you want to change.
- 7 Apply the font style to the caption by setting **FontID** to the font style's index.

To change other characteristics of an existing text caption, first set the **CaptionID** property on the **Captions** panel to the value that corresponds to the text caption you want to change. Then use other properties on the dialog box panel, *except* the **Captions** counter, to configure the text caption accordingly.

---

**Note** For text captions, the color choice on the **Captions** dialog box panel overrides the color choice on the **Fonts** dialog box panel.

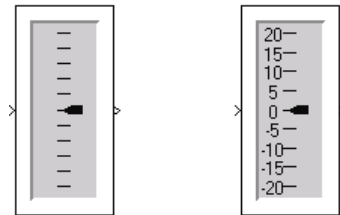
---

## Modifying the Displayed Range

Changing the range of values displayed on a block involves adjusting these properties:

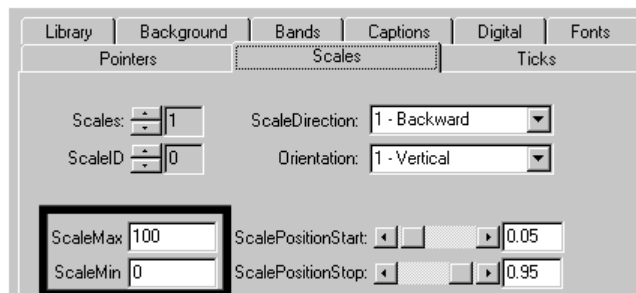
- Scale properties define the extent of the units displayed by the block, the location of the block's center, and the block's start and stop positions.
- Tick-mark properties define tick marks on the block, including start and stop values, the interval between tick marks, and label positions.
- Needle or pointer properties indicate the value.

To illustrate how to use these properties to adjust the range of values displayed on a block, this example changes the Generic Linear Gauge to display values from -20 to 20, sets the interval between tick marks to 5, and shows the tick-mark labels. This figure shows the Generic Linear Gauge with its default settings (left) and with modified settings (right).



## Changing the Scale

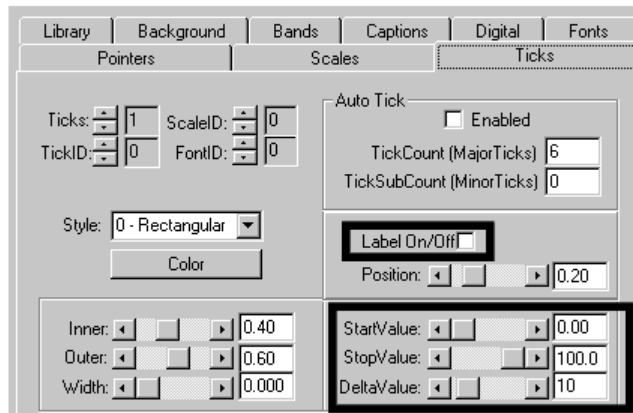
Click the **Scales** tab to display the scales properties page. This figure shows the default scale properties for the Generic Linear Gauge.



To modify the scale range, change **ScaleMax** to 20 and **ScaleMin** to -20.

### Displaying Labels Next to Tick Marks

Click the **Ticks** tab to display the tick-mark properties page. This figure shows the default tick-mark properties.

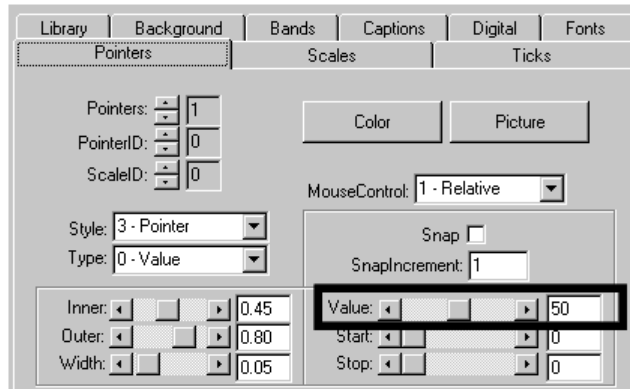


To show tick-mark labels, check the **Label On/Off** check box.

To set the starting and ending tick marks so they mark the minimum and maximum scale settings, set **StartValue** to -20 and **StopValue** to 20. Change the **DeltaValue** property, which sets the spacing between tick marks. A value of 5 is reasonable for default block size.

### Setting the Current Pointer Value

Click the **Pointers** tab to display the pointer properties page. This figure shows the default pointer properties.

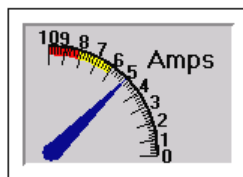


The **Value** property indicates the current pointer value. Set the initial value to 0, halfway between the maximum and minimum scale values. Click **OK** to accept the changes and close the dialog box.

## Modifying Multiple Tick Marks

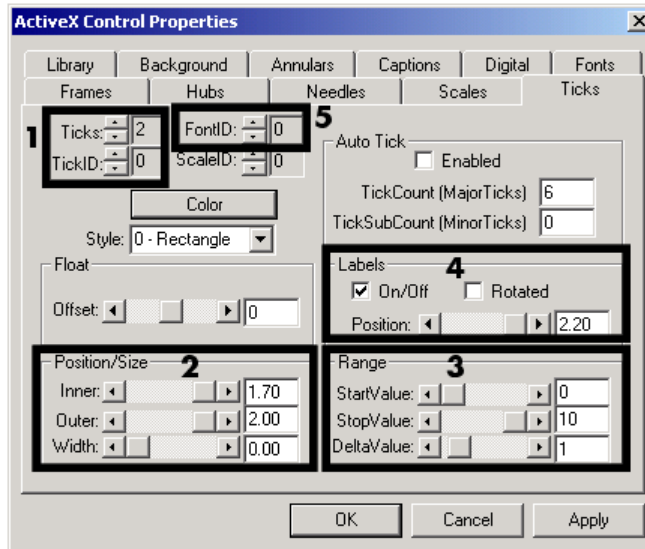
This example illustrates the use of multiple tick marks and the use of the ID property to manage them. Instead of modifying a block, this example examines the default settings for a particular block.

This figure shows the Amp Meter block. Notice that the tick marks have two different lengths. These are created by defining two sets of tick marks.



The first set consists of 11 longer tick marks, each positioned at one of the label values, positioned at increments of 1.0. The second set consists of five shorter tick marks for each integer change in the scale, positioned at increments of 0.2.

To examine how these tick marks have been created, double-click the Amp Meter block to display its properties dialog box. Select the **Ticks** tab.



The **Ticks** and **TickID** properties, in the box labeled 1, are defined as follows:

- The **Ticks** property specifies how many sets of tick marks are used by the block. For this block, this property is set to **2**.
- The **TickID** property indicates which set of tick marks is defined by the other properties on this page. When specifying the characteristics of a set of tick marks, you set the **TickID** property, and then define the property values for that set of tick marks. In the dialog box page above, the settings for all the properties on the page apply to the first set, identified as **TickID 0**.

---

**Note** When you define multiple components, the first instance is identified by an ID of 0. In this example, the two sets of tick marks have IDs of 0 and 1.

---



The **Position/Size** properties, in the box labeled 2, are defined as follows:

- The **Inner** property defines the edge of the tick mark closest to the needle center and the **Outer** property defines the edge of the tick mark farthest from the needle center. To see where the tick marks are located relative to the needle length, examine the needle length by selecting the **Needles** page. The needle length is 2.0. The **Inner** position is 1.70 and the **Outer** position is 2.00. These tick marks are 0.3 units long.
- The **Width** property of the tick marks is 0.00, the narrowest width.

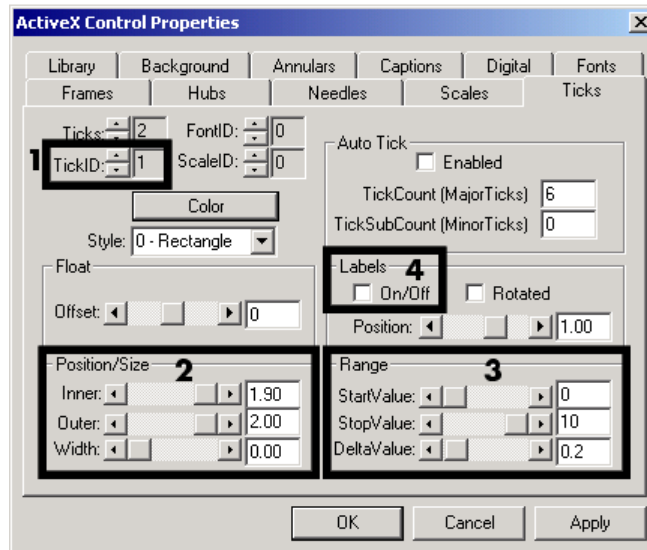
The **Range** properties, in the box labeled 3, are defined as follows:

- **StartValue** determines at which scale value the first tick mark is displayed. For these tick marks, the value is 0.
- **StopValue** determines at which scale value the last tick mark is displayed. For these tick marks, the value is 10.
- **DeltaValue** determines the interval between tick marks. For these tick marks, the value is 1.

The **Labels** properties **On/Off** check box, in the box labeled 4, determines whether the labels are displayed. For the first set of tick marks, the labels are displayed.

The **FontID** property, in the box labeled 5, determines which of multiple fonts defined for this block is used for the label. In this case, two font sets are defined. The first (**FontID** 0) is for the tick marks, while the second (**FontID** 1) is for the caption, "Amps."

To examine the second set of tick marks, change the **TickID** property value to 1 by clicking the up arrow to the left of the value. The **Ticks** page looks like this.



The **Position/Size** properties, in the box labeled 2, are defined as follows:

- The **Inner** position is 1.90 and the **Outer** position is 2.00. These tick marks are 0.10 units long, one-third the length of the longer tick marks.
- The **Width** property of the tick marks is 0.00, the same as the longer tick marks.

The **Range** properties, in the box labeled 3, are defined as follows.

- **StartValue** for these tick marks is 0. The first short tick mark and the first long tick mark appear in the same place.
- **StopValue** for these tick marks is 10. The last short tick mark and the last long tick mark appear in the same place.
- **DeltaValue** determines the interval between tick marks. For these tick marks, the value is 0.2.

The **Labels** properties **On/Off** check box determines whether labels appear next to the tick marks. No labels appear next to this set of tick marks.

If you decrease the **Ticks** property, then the tick-mark settings corresponding to the highest **TickID** value is removed. To replace that set of tick marks, you will have to recreate the settings from the defaults.

## Controlling Multiple Graphical Elements

In this section...
“Overview” on page 2-18
“Simulating a Multiple-Needle Stopwatch” on page 2-18
“Updating Multiple Portions of a Pie Chart” on page 2-23

### Overview

This section describes techniques for displaying multiple input values simultaneously and dynamically on a gauge block that includes multiple graphical elements. Examples of multiple-component gauge blocks include these preconfigured blocks:

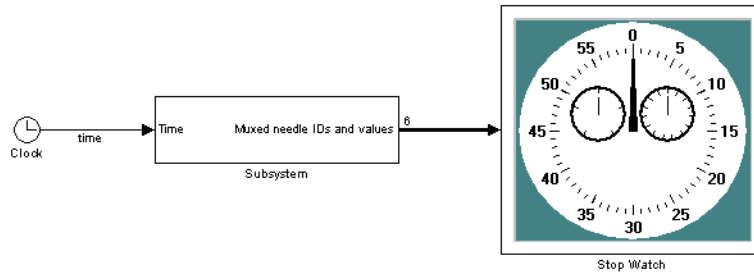
- Multiple Scales in the Linear Gauges library
- Pie Chart and Dynamic Pie in the Percent Indicators library
- Analog Clock and Stop Watch in the Angular Gauges library

### Simulating a Multiple-Needle Stopwatch

This example model simulates a typical sailing stopwatch. The model uses the Stop Watch block, which displays three needles on individual angular scales. The needles mark simulation time simultaneously, as follows:

- The large needle spans 1 minute.
- The small needle on the left spans 1 second.
- The small needle on the right spans 15 minutes.

To open this example, enter `gauges_stopwatch` in the MATLAB Command Window. Run the simulation and watch how the three needles move.



These sections describe how the model works:

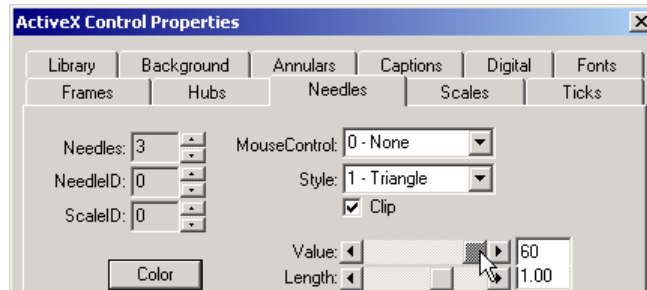
- “How NeedleID Values Correspond to Needles” on page 2-19
- “Configuration of the Stop Watch Block” on page 2-21
- “Preparing the Input Signal for the Stop Watch Block” on page 2-22

### How NeedleID Values Correspond to Needles

To explain how the model controls three needles, this section first explains how the block makes its needles accessible to you. Each needle on the block has an associated pair of **NeedleID** and **Value** parameters, where

- **NeedleID** is a number that distinguishes that needle from other needles on the block.
- **Value** is the numerical value for that needle along its angular scale.

You can view or control these parameters via the **Needles** panel of the block’s ActiveX Control Properties dialog box.



To find out which needle and range of values correspond to a given ID number, use this procedure:

- 1 Right-click the Stop Watch block and choose **Control Display Properties**.
- 2 Click the **Needles** tab.
- 3 Set **NeedleID** to the ID number you want to investigate: 0, 1, or 2 in this case.
- 4 Change the **Value** parameter while watching which needle on the block moves. By dragging the **Value** slider to its extremes, you can also find out the minimum and maximum values for the needle with that **NeedleID** number.

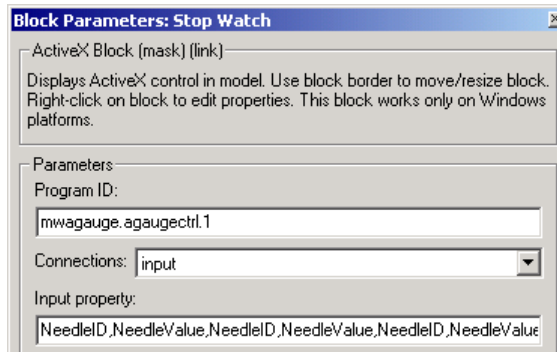
The table below summarizes what the procedure reveals.

NeedleID	Needle on Block	Range of Values
0	Large needle	[0, 60] seconds
1	Small needle on right	[0, 15] minutes
2	Small needle on left	[0, 5] fifths of a second

For more information about ID properties, see “Understanding ID Properties” on page 2-7.

## Configuration of the Stop Watch Block

After you run the model, the number 6 appears near the connector line that represents the input to the Stop Watch block. This indicates that the signal on that line is a vector of length 6. To understand why, first right-click the Stop Watch block and choose **Block Parameters**.



Then notice that the **Input property** field in the dialog box is set to

NeedleID,NeedleValue,NeedleID,NeedleValue,NeedleID,NeedleValue

By contrast, the default value for this property is `NeedleValue`, which you can see by examining the block in the Angular Gauges library.

The comma-separated list in this example model causes the Stop Watch block to use the six values in its vector input signal to assign these parameters to the block, in sequence:

- The ID number, 0, of the large needle
- The numerical value for the large needle
- The ID number, 2, of the small needle on the left
- The numerical value for the small needle on the left
- The ID number, 1, of the small needle on the right
- The numerical value for the small needle on the right

---

**Note** It is important that each needle's ID number precede its numerical value. Referring first to the ID number tells the block which needle to apply changes to.

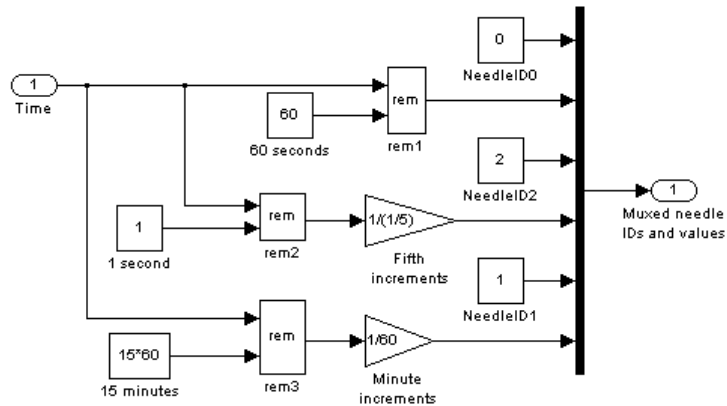
---

### Preparing the Input Signal for the Stop Watch Block

The model aims to reflect the time on the Stop Watch block. However, it is necessary to process the Clock block's output somewhat to prepare it for the Stop Watch block. Double-click the Subsystem block to open it. The subsystem accomplishes these key tasks:

- Processing the Clock block's output value, measured in seconds, to make it an appropriate numerical value for each needle. To do this, the block performs these computations:
  - Reduces modulo 60, to get the large needle's value.
  - Reduces modulo 1 and then multiplies by 5, to get the left needle's value. Recall that values for the left needle range between 0 and 5.
  - Reduces modulo  $15 \cdot 60$  and then divides by 60, to get the right needle's value. Recall that values for the right needle range between 0 and 15.
- Forming a six-element vector corresponding to the comma-separated list in "Configuration of the Stop Watch Block" on page 2-21. To form this vector, the subsystem uses three Constant blocks, the three processed Clock block signals, and a Mux block.

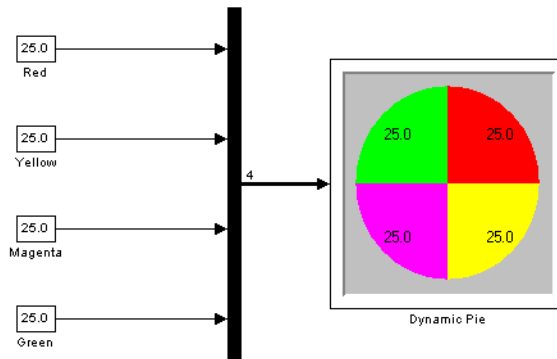




## Updating Multiple Portions of a Pie Chart

This example model enables you to control four portions of a pie chart independently using four sources. The model uses the Dynamic Pie block. Internally, the model uses an M-file S-function to drive the Dynamic Pie block. Compared to the technique in “Simulating a Multiple-Needle Stopwatch” on page 2-18, the S-function technique is more complicated but also more flexible and powerful.

To open this example, enter `gauges_pie` in the MATLAB Command Window. Change the values of the Constant blocks to alter the composition of the pie chart. Note that because Constant is a source block, the simulation will pause while the block dialog box is open. You must close the dialog by clicking **OK**, which will resume the simulation and allow you to see the result of your changes.



These sections describe how the model works:

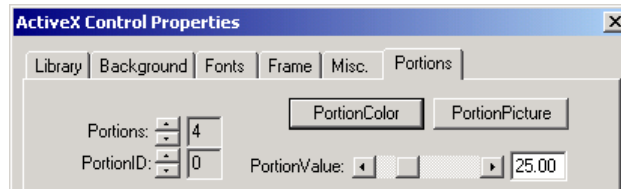
- “How PortionID Values Correspond to Portions of the Pie” on page 2-24
- “Configuration of the Dynamic Pie Block” on page 2-25
- “How the S-Function Updates the Pie” on page 2-27
- “Initial Portion Sizes in the Model” on page 2-28

### How PortionID Values Correspond to Portions of the Pie

To explain how the model controls four portions, this section first explains how the block makes its portions accessible to you. Each portion on the block has an associated pair of **PortionID** and **PortionValue** parameters, where

- **PortionID** is a number that distinguishes that portion from other portions on the block.
- **PortionValue** is the numerical value for that portion, as a percentage. The S-function that assigns the set of **PortionValue** parameters ensures that the percentages for all portions add up to 100.

You can view or control these parameters via the **Portions** panel of the block’s ActiveX Control Properties dialog box.



To find out which portion corresponds to a given ID number, use this procedure:

- 1** Right-click the Dynamic Pie block and choose **Control Display Properties**.
- 2** Click the **Partitions** tab.
- 3** Set **PortionID** to the ID number you want to investigate: 0, 1, 2, or 3 in this case.
- 4** Change the **PortionValue** parameter while watching which portion on the block changes in size.

The table below summarizes what the procedure reveals.

PortionID	Portion on Block
0	Red portion
1	Yellow portion
2	Magenta portion
3	Green portion

For more information about ID properties, see “Understanding ID Properties” on page 2-7.

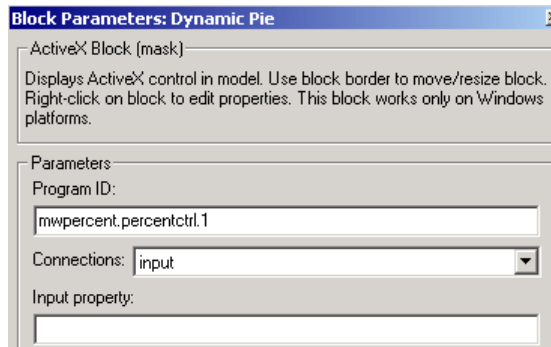
### Configuration of the Dynamic Pie Block

In the model, a vector containing the four constant values is the input to the Dynamic Pie block. While the simulation is running, an S-function called `gauges_pie_sfun.m` uses the vector to make the Dynamic Pie block reflect

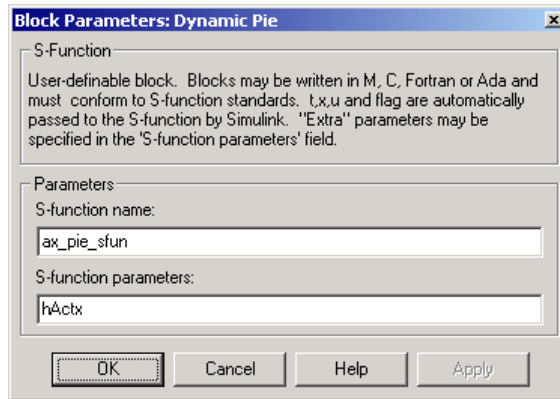
the constant values. While the behavior of the S-function is discussed below (“How the S-Function Updates the Pie” on page 2-27), this section describes how the Dynamic Pie block is linked with the S-function.

The Dynamic Pie block in this model is a customized copy of the original one in the Percent Indicators library. The customized copy differs from the original in these ways:

- The **Input property** field in the Block Parameters dialog box is blank. This is because the block is being updated by an S-function that ignores the data in the **Input property** field. To see the **Input property** field, right-click the block and choose **Block Parameters**.



- The block is driven by a customized S-function, `ax_pie_sfun.m`, rather than the S-function that drives the original library block. To see this, select the outer border of the block and choose **Edit -> Look under mask** from the model window's menu.



The S-function `ax_pie_sfun.m` is a customized version of `ax_strip_sfun.m`, which is an S-function designed to drive the Strip Chart block. Many parts of `ax_pie_sfun.m` are also similar to `sfuntmpl.m`, which is an M-file S-function template included in the Simulink distribution. Many features of that S-function template are not required for controlling blocks in Gauges Blockset, which simplifies the task of writing S-functions for use with the blockset. For more information about S-functions in general, see the Writing S-Functions documentation.

If you were building this model yourself starting from the original library block, then you would have to break the library link before changing the values in the S-Function dialog box shown previously. To break the library link for a library block, use this procedure:

- 1 Select the outer border of the block.
- 2 Choose **Edit -> Link options -> Disable link**.
- 3 Choose **Edit -> Link options -> Break link**.

### How the S-Function Updates the Pie

While the simulation is running, the S-function `ax_pie_sfun.m` drives the Dynamic Pie block. In particular, this S-function

- Receives the vector that is the input signal to the Dynamic Pie block

- Normalizes the vector so that the values add up to 100
- Updates each portion of the pie to reflect the corresponding number from the vector

**Receiving the Vector Input Signal.** During the simulation, Simulink invokes the S-function and passes it the Dynamic Pie block's input vector. Within the S-function, the vector is called *u*. Simulink also passes to the S-function a handle of the Dynamic Pie ActiveX control. The handle is called *hActx*.

**Normalizing the Input Vector.** The S-function uses the code below to normalize the vector *u* so that its elements add up to 100:

```
% First make sure there is no division by zero.
if sum(u)==0
    u=u+0.001;
end

% Now perform the normalization.
u = 100/sum(u).*u;
```

**Updating the Dynamic Pie Block.** After *sum(u)* is 100, the S-function updates the portions of the Dynamic Pie block by setting each one to the corresponding element of *u*. The code uses the handle *hActx* to access the **PortionID** and **PortionValue** parameters of the Dynamic Pie block.

```
% Loop through the portions and update their values.
%
if (length(u) ~= 0)
    for n=1:length(u)
        hActx.PortionID = n-1;
        hActx.PortionValue = u(n);
    end
end
```

### Initial Portion Sizes in the Model

When you first open the `gauges_pie` model, the portions of the pie have equal sizes, unlike the portions in the default instance of the Dynamic Pie block

in the Percent Indicators library. The equal-sized portions result from the configuration from which the model was saved.

If you were building this model yourself starting from the original library blocks, then you would first run the model to make the portion sizes reflect the values on the Constant blocks, and then save the model to make Simulink record the blocks' configurations.

## Saving and Reusing a Customized Control

<b>In this section...</b>
“Saving Customized Controls Automatically” on page 2-30
“Saving Customized Controls Using the Library Panel” on page 2-30

### Saving Customized Controls Automatically

Saving the model causes MATLAB to save all property settings for Gauges Blockset blocks in the model file. To share your customized controls with other users, give them the model file. To use your customized block in a new model, copy the block from the old model and paste it into the new model.

### Saving Customized Controls Using the Library Panel

Alternatively, you can use the ActiveX Control Properties dialog box to save property settings for later use on your own machine. However, this method does not enable you to share these customized controls with users of other machines. The steps are

- 1 Select the **Library** tab of the ActiveX Control Properties dialog box.
- 2 Assign a name to the collection of modified settings by entering a new name in the **Configuration Name** field.

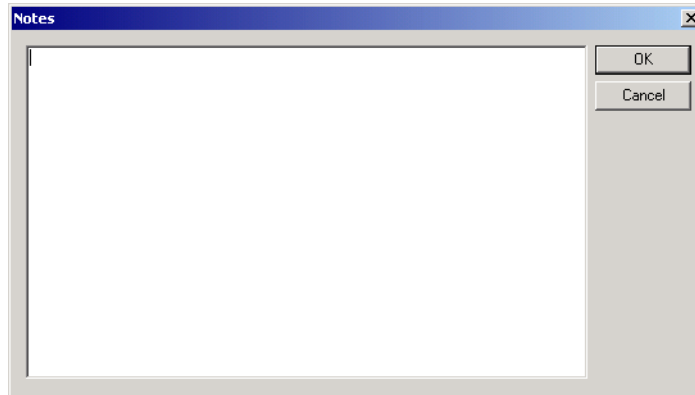
---

**Note** If you leave this field blank, the new property settings write over the previous settings, which means that you cannot access the original version except by reinstalling the blockset or by registering the ActiveX controls again. To learn how to register the ActiveX controls, see “Troubleshooting the Gauges Blockset Installation” on page 1-4.

---

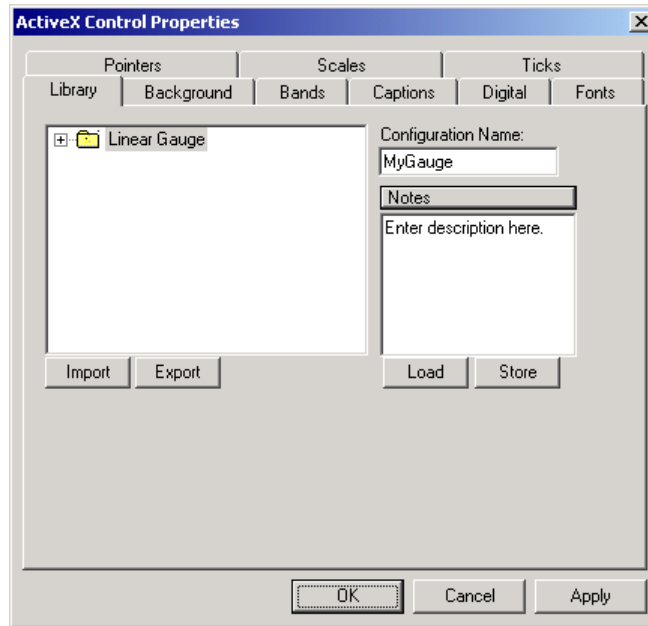


- 3 To provide textual information about the block, click **Notes**. This dialog box appears.



- 4 Enter a description in the text area and click **OK**.
- 5 Select the directory in which to store the modified control by expanding the library hierarchy at the left. The new set of property settings is stored in the directory you select. Click **Store**.
- 6 Click **OK** to accept all the changes and close the dialog box.

The next figure shows the dialog box with fields filled in. The customized control is stored in the **Linear Gauge** directory.



An alternative to this procedure is to export customized controls to .gms files. To do this, select a directory from the left side of the panel and click **Export**. You can later access these controls by using the **Import** button, or share the controls by sharing the .gms files.

# Categories of ActiveX Controls

---

Angular Gauges (p. 3-2)	Controls that show their input values graphically along an arc of a circle
LEDs (p. 3-5)	Controls that use graphical elements to imitate light-emitting diodes
Linear Gauges (p. 3-7)	Controls that show their input values graphically along a linear scale
Numeric Displays (p. 3-11)	Controls that display the numerical values of their input signals
On Off Gauges (p. 3-14)	Two-state gauges
Percent Indicators (p. 3-15)	Controls that are designed to display percentages and ratios
Strip Chart (p. 3-18)	A control that plots data on a strip chart
Using Your Own ActiveX Control (p. 3-20)	How to incorporate ActiveX controls into your model if they are not part of the standard Gauges Blockset package
Block Parameters for the ActiveX Control Block (p. 3-24)	Options in the Block Parameters dialog box for an ActiveX Control block

## Angular Gauges

### In this section...

“Library Overview” on page 3-2

“Customizing Angular Gauges” on page 3-2

### Library Overview

The Angular Gauges library contains controls that show their input value graphically along an arc of a circle. Blocks in the library differ from each other in their numerical ranges and in their use of needles, numerical labels, text captions, annular components, and tick marks.

### Customizing Angular Gauges

This section describes how to customize angular gauges by making changes that are specific to the Angular Gauges library. For changes that apply to multiple categories of blocks, see these sections:

- “Using Multiple Styles Within One Block” on page 2-3
- “Displaying Text on a Block” on page 2-9
- “Modifying the Displayed Range” on page 2-11
- “Modifying Multiple Tick Marks” on page 2-13

The table below lists some common customizations involving the ActiveX Control Properties dialog box that are specific to blocks in the Angular Gauges library.

Task	Description
Change the shape or size of a needle	On the <b>Needles</b> panel, set <b>NeedleID</b> to the ID of the needle you want to change (0 if there is exactly one needle). Then use the <b>Style</b> property to choose the shape, and the <b>Length</b> and <b>Width</b> properties to determine the length and thickness.

<b>Task</b>	<b>Description</b>
Label a needle by displaying the corresponding number	On the <b>Digital</b> panel, set <b>NeedleID</b> to the ID of the needle you want to label and check the <b>Enabled</b> check box.
Change the appearance of a needle label	On the <b>Digital</b> panel, first set <b>NeedleID</b> to the ID of the needle whose label you want to change. Then use <b>Decimals</b> to set the number of digits after the decimal point, <b>Color</b> to set the color of the number, and <b>FontID</b> to refer to a previously defined font (on the <b>Fonts</b> panel).
Move a needle label	On the <b>Digital</b> panel, first set <b>NeedleID</b> to the ID of the needle whose label you want to change. Then use <b>X Position</b> and <b>Y Position</b> to set the fixed position for the label.
Draw an annular region along the scale	On the <b>Annulars</b> panel, increase the value of the <b>Annulars</b> property. The ID of the new region is the <b>Annulars</b> property value minus one. To specify properties of the new region, see the next task.
Change the appearance of an annular region	On the <b>Annulars</b> panel, first set <b>AnnularID</b> to the ID of the annular region you want to change. Use the <b>Radius</b> properties to control the annular region's thickness and radial position. Use the <b>Value</b> properties to control the portion of the scale's range that the annular region includes. Use <b>Color</b> to control the annular region's color.
Delete the most recently added annular region	On the <b>Annulars</b> panel, decrease the <b>Annulars</b> property. This deletes all properties associated with the region, such as its color and thickness.

### Combining Multiple Needles in One Display

To display multiple needles on a single block, see the table below. To learn how to control multiple needles simultaneously, see “Controlling Multiple Graphical Elements” on page 2-18.

<b>Task</b>	<b>Description</b>
Add another needle to the display	On the <b>Needles</b> panel, increase the <b>Needles</b> property. The ID of the new region is the <b>Needles</b> property value minus one. To specify properties of the new needle, set <b>NeedleID</b> to that ID and then set the remaining properties on the dialog box panel accordingly.
Delete the most recently added needle from the display	On the <b>Needles</b> panel, decrease the <b>Needles</b> property. This deletes all properties associated with the needle, such as its color and shape.

## LEDs

### In this section...

“Library Overview” on page 3-5

“Customizing LEDs” on page 3-5

## Library Overview

The LEDs library contains controls that use graphical elements to imitate light-emitting diodes (LEDs). Each block reflects its input value by setting one or more graphical elements to an on or off state. By default, the number of LEDs in the on state is the rounded value of the block’s input.

Most blocks in this library contain a single LED. These blocks differ from each other in the appearance of their LEDs. The Vertical Meter, Horizontal Meter, and Circle Meter blocks contain multiple LEDs per block.

## Customizing LEDs

The table below lists some common ways to customize a block in the LEDs library, using its ActiveX Control Properties dialog box.

Task	Description
Add or remove LEDs	Change the <b>NumLEDs</b> property on the <b>LEDs/General</b> panel.
Change the shape or color of a particular LED	On the <b>LEDs/General</b> panel, set <b>LEDIndex</b> to the number corresponding to the LED you want to customize. To apply a previously defined style, set the <b>LEDStyleID</b> to the number corresponding to the style. To define a new style for this LED, increase the <b>StyleID</b> property on the <b>Styles</b> panel and then configure the color, picture, or shape properties accordingly.

<b>Task</b>	<b>Description</b>
Change the size or layout of a set of LEDs	On the <b>LEDs/General</b> panel, use <b>LEDWidth</b> and <b>LEDHeight</b> to control the size of each LED. Use <b>LEDSeparation</b> to control the spacing between successive LEDs. Use <b>Orientation</b> and/or <b>Direction</b> to control how multiple LEDs are arranged along a line.
Display a binary representation of the (rounded) input	Set the <b>Mode</b> property on the <b>LEDs/General</b> panel to Bitwise. The first LED corresponds to the least significant bit.
Display decaying maximum value of the input, in addition to the current input	Check the <b>MaxDecay</b> check box on the <b>LEDs/General</b> panel. The <b>DecayRate</b> value controls how quickly the displayed value decays from the maximum to the current input value. Larger positive values correspond to a slower decay. A value of zero causes the block to reflect its maximum value with no decay.



## Linear Gauges

### In this section...

“Library Overview” on page 3-7

“Customizing Linear Gauges” on page 3-7

### Library Overview

The Linear Gauges library contains controls that show their input values graphically along a linear scale. Blocks in the library differ from each other in their numerical ranges and in their use of pointers, numerical labels, text captions, and tick marks.

The blocks in this library fall into two categories, as listed below:

Pointer Linear Gauges	Bar Gauges
Generic Linear Gauge	Generic Bar Gauges
Min-Max Thermometer	Reverse Sliding Scale
Mixer	Scaled Bar Gauge
Mixer Scale	Tank
Multiple Scales	Thermometer

Blocks in the two categories have slightly different interfaces, capabilities, and appearances. Pointer linear gauges use one or more pointers to indicate values. Bar gauges use a level indicator or a two-color bar to reflect a single value or level.

### Customizing Linear Gauges

This section describes how to customize linear gauges by making changes that are specific to the Linear Gauges library. For changes that apply to multiple categories of blocks, see these sections:

- “Using Multiple Styles Within One Block” on page 2-3
- “Displaying Text on a Block” on page 2-9

- “Modifying the Displayed Range” on page 2-11
- “Modifying Multiple Tick Marks” on page 2-13

### Customizing Pointer Linear Gauges

The table below lists some common customizations involving the ActiveX Control Properties dialog box that are specific to pointer linear gauges in the Linear Gauges library.

Task	Description
Change the shape or size of a pointer	On the <b>Pointers</b> panel, set <b>PointerID</b> to the ID of the pointer you want to change (0 if there is exactly one pointer). Then use the <b>Style</b> property to choose the shape, the <b>Inner</b> and <b>Outer</b> properties to determine the length, and the <b>Width</b> property to determine the thickness.
Label a pointer by displaying the corresponding number	On the <b>Digital</b> panel, set <b>PointerID</b> to the ID of the pointer you want to label and check the <b>PointerDigital</b> check box.
Change the appearance of a pointer label	On the <b>Digital</b> panel, first set <b>PointerID</b> to the ID of the pointer whose label you want to change. Then use <b>Decimals</b> to set the number of digits after the decimal point, <b>PointerDigitalColor</b> to set the color of the number, and <b>FontID</b> to refer to a previously defined font (on the <b>Fonts</b> panel).
Move a pointer label to a fixed position	On the <b>Digital</b> panel, first set <b>PointerID</b> to the ID of the pointer whose label you want to change. Clear the <b>PointerDigitalAttach</b> check box and use <b>PointerDigitalX</b> and <b>PointerDigitalY</b> to set the fixed position for the label.
Move a pointer label to a position relative to the pointer	On the <b>Digital</b> panel, first set <b>PointerID</b> to the ID of the pointer whose label you want to change. Check the <b>PointerDigitalAttach</b> check box. For a vertical linear scale, use <b>PointerDigitalX</b> to set the independent coordinate for the label. For a horizontal linear scale, use <b>PointerDigitalY</b> to set the independent coordinate for the label.

## Customizing Bar Gauges

The table below lists some common customizations involving the ActiveX Control Properties dialog box that are specific to bar gauges in the Linear Gauges library.

Task	Description
Change the range of values along the bar	On the <b>General</b> panel, use the <b>Min Value</b> and <b>Max Value</b> properties to define the range.
Change the orientation or direction of the bar	On the <b>General</b> panel, use <b>Orientation</b> to determine whether the bar is horizontal or vertical. Use <b>Direction</b> to determine which end of the bar corresponds to the minimum value.
Change the size or position of the bar	On the <b>Bar</b> panel, use the <b>BarInner</b> and <b>BarOuter</b> properties to define the width and position of the bar in the direction perpendicular to the linear scale. Use the <b>BarStart</b> and <b>BarStop</b> properties to define the length and position of the bar in the direction of the linear scale. These properties do not affect the numerical values associated with the bar, only the graphical depiction of the bar.
Change the colors of the portions of the bar on either side of the indicated level	On the <b>Bar</b> panel, use the <b>OnColor</b> and <b>OffColor</b> properties to define the colors associated with values below and above, respectively, the indicated level along the bar.
Change the shape or size of the level indicator	On the <b>Knob</b> panel, use the <b>Style</b> property to choose the shape. Use the <b>Inner Value</b> and <b>Outer Value</b> properties to determine the thickness and position in the dimension perpendicular to the linear scale. Use the <b>Width</b> property to determine the width along the linear scale.
Label the indicated level using a number	On the <b>Digital</b> panel, check the <b>Enabled</b> check box.

<b>Task</b>	<b>Description</b>
Change the appearance of the label	On the <b>Digital</b> panel, check the <b>Enabled</b> check box. Then use <b>Decimals</b> to set the number of digits after the decimal point, <b>Color</b> to set the color of the number, and <b>FontID</b> to refer to a previously defined font (on the <b>Fonts</b> panel).
Move the label to a fixed position	On the <b>Digital</b> panel, clear the <b>Attach</b> check box. Then use <b>X Position</b> and <b>Y Position</b> to set the fixed position for the label.
Move the label to a position relative to the level indicator	On the <b>Digital</b> panel, check the <b>Attach</b> check box. For a vertical (respectively, horizontal) linear scale, use <b>X Position</b> (respectively, <b>Y Position</b> ) to set the independent coordinate for the label.

### Combining Multiple Pointers in One Display

To display multiple pointers on a pointer linear gauge, see the customizations in the table below. To learn how to control multiple pointers simultaneously, see “Controlling Multiple Graphical Elements” on page 2-18.

<b>Task</b>	<b>Description</b>
Add another pointer to the display	On the <b>Pointers</b> panel, increase the <b>Pointers</b> property. The ID of the new region is the <b>Pointers</b> property value minus one. To specify properties of the new pointer, set <b>PointerID</b> to that ID and then set the remaining properties on the dialog box panel accordingly.
Delete the most recently added pointer from the display	On the <b>Pointers</b> panel, decrease the <b>Pointers</b> property. This deletes all properties associated with the pointer, such as its color and shape.

# Numeric Displays

In this section...
“Library Overview” on page 3-11
“Customizing Numeric Displays” on page 3-11
“Customizing the Odometer Block” on page 3-13

## Library Overview

The Numeric Displays library contains controls that display the numerical values of their input signals. The Odometer block differs from the other blocks in this library in its appearance and dialog box. The topics in this section are

## Customizing Numeric Displays

The table below lists some common ways to customize any block in the Numeric Displays library, *except* the Odometer block, using the **General** panel of its ActiveX Control Properties dialog box.

Task	Description
Change the number of digits in the display	Set <b>Digits</b> to the total number of digits.
Specify the number of digits after the decimal point	Set <b>Decimals</b> to the number of digits you want after the decimal point, and check the <b>FixedDecimal</b> check box.
Pad the display with leading zeros	Check the <b>LeadingZeros</b> check box.
Display a plus or minus sign	Check the <b>LeadingPlusMinus</b> check box.

<b>Task</b>	<b>Description</b>
Change the appearance of all digits	Use the <b>ItalicsOffset</b> property to control the slanting angle of digits. Use the <b>Segment Width</b> and <b>Segment Separation</b> properties to control the width of the line segments that compose each digit and the spacing between the line segments, respectively. Use the two <b>Spacing</b> properties to control the padding around each digit.
Avoid using nonnumeric characters such as a colon in the display, making the block easier to use with Simulink signals	First set <b>DisplayMode</b> to 0 - Numeric. Then open the block's Block Parameters dialog box by double-clicking the block's border, and set <b>Input property</b> to Value.

## Customizing the Odometer Block

The table below lists some common ways to customize the Odometer block, using the **General** panel of its ActiveX Control Properties dialog box.

Task	Description
Change the number of digits in the display	Set <b>Digits</b> to the total number of digits. Set <b>Decimals</b> to the number of digits after the decimal point. The block does not display a decimal point character, but digits that represent proper fractions appear with inverted colors.
Make the display change gradually from old to new values, instead of registering the change instantaneously	Check the <b>Transition Enabled</b> check box. Set the <b>Steps</b> value to the number of steps in the gradual transition. Use the <b>Rate</b> value to control the speed of the transition, where larger values indicate a slower transition.

## On Off Gauges

In this section...
“Library Overview” on page 3-14
“Customizing On Off Gauges” on page 3-14

### Library Overview

The On Off Gauges library contains gauges that can display two states, on and off. A block input of 0 corresponds to an “off” state and a block input of 1 corresponds to an “on” state. The blocks in this library differ in cosmetic ways, such as the images shown on the block.

### Customizing On Off Gauges

The table below lists some common ways to customize a block in the On Off Gauges library, using its ActiveX Control Properties dialog box.

Task	Description
Associate an image with a state	Use the <b>Picture</b> button on the <b>On</b> or <b>Off</b> panel to select a graphics file. You cannot associate both an image and text with a state.
Associate text with a state	Use the <b>Caption</b> field on the <b>On</b> or <b>Off</b> panel. The <b>X</b> and <b>Y</b> values control the position of the text. The <b>BackColor</b> and <b>ForeColor</b> buttons control the colors of the background and text, respectively. You cannot associate both an image and text with a state.
Associate a sound with a state	On the <b>On</b> or <b>Off</b> panel, check the <b>Sound</b> check box and type the name of a .wav file in the <b>Wave file</b> field. You can either type the name of the sound file or browse for it using the ... button.
Use beveling to make the button appear three-dimensional	Use the <b>BevelInner</b> and <b>BevelOuter</b> properties on the <b>Background</b> panel.



## Percent Indicators

### In this section...

“Library Overview” on page 3-15

“Customizing Percent Indicators” on page 3-15

### Library Overview

The Percent Indicators library contains controls that are designed to display percentages and ratios. The Generic Percent and Simple Light Blue blocks are probably the most useful blocks in this library. By default, these blocks reflect scalar input values between 0 and 100 by coloring a corresponding segment of a linear scale. By customizing the blocks, you can also have them display an input value  $X$  between  $m$  and  $M$  as the percentage  $100 * ((X - m) / (M - m))$ .

### Customizing Percent Indicators

The table below lists some common ways to customize a block in the Percent Indicators library, using its ActiveX Control Properties dialog box.

Task	Description
Use a radial percentage scale that reflects the input as a sector of a circle	On the <b>Misc.</b> panel, set <b>DisplayMode</b> to Radial. Use the <b>StartAngle</b> value to indicate where the sector begins; a value of 0 corresponds to a vertical radius above the circle's center, while a value of 90 corresponds to a horizontal radius to the right of the circle's center.
Change the direction in which a radial percentage scale increases	On the <b>Misc.</b> panel, use the <b>Direction</b> property to reverse the scale's polarity. If <b>Direction</b> is set to Forward, then the scale increases clockwise.
Use a linear percentage scale that reflects the input as a portion of a rectangle	On the <b>Misc.</b> panel, set <b>DisplayMode</b> to Linear.

<b>Task</b>	<b>Description</b>
Change the direction in which a linear percentage scale increases	On the <b>Misc.</b> panel, use the <b>Orientation</b> property to indicate whether the linear scale is horizontal or vertical. Use the <b>Direction</b> property to reverse the scale's polarity. If <b>Direction</b> is set to Forward, then a horizontal scale increases to the right and a vertical scale increases downward.
Specify the range to use when converting the input to a percentage	On the <b>Misc.</b> panel, use the <b>Min</b> and <b>Max</b> properties. If the input value is X, then the block displays the percentage $100 * ((X - \text{Min}) / (\text{Max} - \text{Min}))$ .
Display a number near or inside the corresponding colored area	On the <b>Portions</b> panel, set <b>DigitalStyle</b> to Floating. You can use the <b>DigitalPosition</b> value to vary the position along one dimension (radius in the case of a radial scale, height in the case of a horizontal scale, and horizontal coordinate in the case of a vertical scale).
Display a number in a fixed position	On the <b>Portions</b> panel, set <b>DigitalStyle</b> to Fixed. To specify the position of the number, first set <b>PortionID</b> to the ID of the portion you want to configure (0 if you are displaying only the scalar input signal) and then use the <b>PortionDigitalX</b> and <b>PortionDigitalY</b> values to indicate the position.

## Combining Multiple Regions in One Display

To display multiple regions on a single block, see the customizations in the table below. To learn how to control multiple regions simultaneously, see “Controlling Multiple Graphical Elements” on page 2-18.

Task	Description
Add another region to the display	On the <b>Portions</b> panel, increase the <b>Portions</b> property. The ID of the new region is the <b>Portions</b> property value minus one. To specify properties of the new region, set <b>PortionID</b> to that ID and then set the remaining properties on the dialog box panel accordingly. Note that the <b>DigitalStyle</b> and <b>DigitalFormat</b> properties apply to all regions on the block.
Delete the most recently added region from the display	On the <b>Portions</b> panel, decrease the <b>Portions</b> property. This deletes all properties associated with the region, such as its color.

## Strip Chart

The interface to the Strip Chart block is different from the interface to the other preconfigured blocks in Gauges Blockset. You can configure the Strip Chart block using properties in its dialog box, just as you would for other preconfigured blocks. However, to plot data on the chart, you must invoke methods for the block. You can use the MATLAB command `invoke` to call methods of ActiveX control blocks and pass arguments to those methods.

An M-file S-function provided with Gauges Blockset plots data on the Strip Chart block by using the `invoke` method. More generally, this S-function illustrates how to communicate with any ActiveX control from the MATLAB language through an S-function.

The file is called `ax_strip_sfun.m` and is located in the main Gauges Blockset directory. You can use the following MATLAB command to find the location of this file on your computer.

```
which ax_strip_sfun
```

During initialization, the Simulink block attributes (sample time, input width, etc.) are configured and the Strip Chart configuration is set. The infrastructure of Gauges Blockset provides the handle of the ActiveX control (`hActX`) and is available in this S-function.

You can use this handle to set the properties of the Strip Chart through the standard dot notation. For example, the following line sets the `LastX` property of the Strip Chart to zero.

```
hActX.LastX = 0;
```

Any property of the Strip Chart can be set in this fashion.

In the outputs section of the S-function, each track of the Strip Chart is initialized to zero on the time axes and the actual plotting of the data is performed. A loop is included in this section to account for vector signals sent to the Strip Chart from Simulink.

Note that S-functions offer more options than those shown in this example. See the Writing S-Functions documentation for more details on writing your own S-functions.

## Using Your Own ActiveX Control

### In this section...

“Adding the ActiveX Control Block to a Model” on page 3-20

“Notes on Third-Party ActiveX Control Blocks” on page 3-21

See also “Block Parameters for the ActiveX Control Block” on page 3-24.

### Adding the ActiveX Control Block to a Model

To use your own ActiveX control in a Simulink model, you must associate it with the generic ActiveX Control block. To configure the ActiveX Control block to display your ActiveX control, you need to know some of the programming features of the ActiveX control:

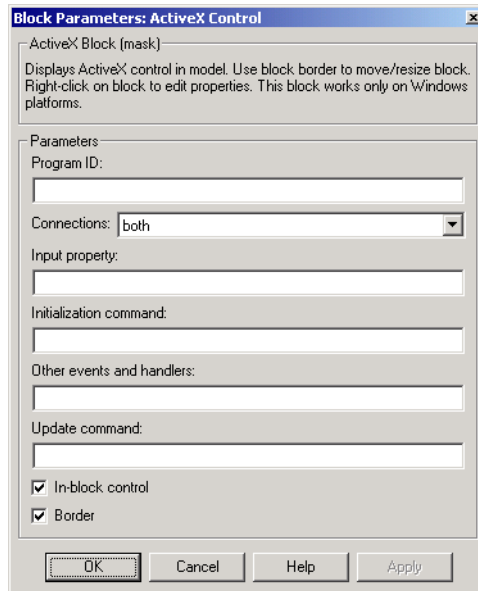
- The name under which the ActiveX control is registered on your system
- The events that cause the ActiveX control to perform an action
- The ActiveX control properties that are affected by events, by signals passed to the block, or by initialization commands

To use an ActiveX Control block in a Simulink model:

- 1 Drag the ActiveX Control block from the top level of Gauges Blockset to your model. Place the block where you want the control to appear.



- 2 Double-click the block to display its Block Parameters dialog box. Specify the appropriate values, described in subsequent sections.




---

**Note** Double-clicking the border of a preconfigured block (supplied with the blockset) displays its ActiveX Control Properties dialog box, which lists properties in multiple tabbed panels. Double-clicking a block that you created by customizing the generic ActiveX Control block displays its Block Parameters dialog box.

---

## Notes on Third-Party ActiveX Control Blocks

This section contains additional notes about third-party ActiveX controls. One note is about editing ActiveX controls that ignore mouse events, while another concerns the colors of ActiveX controls.

### Editing ActiveX Controls That Ignore Mouse Events

Certain ActiveX controls do not handle typical mouse events (double-click, right-click, etc.). These ActiveX controls appear uneditable when used with Gauges Blockset. Double-clicking or right-clicking blocks that use these controls has no effect. To edit this type of block, you must first select the

block so that it is current in the Simulink diagram. Then enter the following commands at the MATLAB prompt:

```
temp = get_param(gcf, 'userdata');  
propedit(temp.hActx);
```

This opens the properties dialog box for that control. See the MATLAB COM documentation for more information on the `propedit` command and assigning event callbacks to ActiveX controls.

Additionally, you can choose an event on your control through which you want to open the property editor. For example, write an M-file function to open the property editor (or whatever you want the event to do). The function must take multiple arguments, of which the first one is the handle of the ActiveX control. For example, a simple function to open the property editor of a control looks like this:

```
function axeventhandler(varargin)  
propedit(varargin{1})
```

Next enter an event with the handler you just wrote in the **Other Events and Handlers** parameter field. Assuming the `keypress` event is valid, the event and handler entry looks like this:

```
{ 'keypress', 'axeventhandler' }
```

To use the error-checking code already written for Gauges Blockset, you can enter `ax_block_dclk` for events that should open the property editor (note that the editor does not open when the simulation is running). For example, to make a keystroke open the property editor (assuming that the `keypress` event is valid), enter the event and handler pair as follows:

```
{ 'keypress', 'ax_block_dclk' }
```

### Colors of ActiveX Controls

ActiveX controls that try to determine their colors by inheriting from the window in which they reside do not work properly in Simulink. More specifically, ActiveX controls that send the `WM_CTLCOLOR` message to their parent have this problem. `WM_CTLCOLOR` is a Microsoft Windows message



sent by an ActiveX control to enable the parent container to determine the color used by the control.

---

**Caution** Placing one of these controls in the ActiveX Control block causes MATLAB and Simulink to crash.

---

## Block Parameters for the ActiveX Control Block

In this section...
“Summary of Parameters” on page 3-24
“Program ID” on page 3-25
“Connections” on page 3-25
“Input Property” on page 3-25
“Initialization Command” on page 3-26
“Other Events and Handlers” on page 3-26
“Update Command” on page 3-27
“In-Block Control” on page 3-27
“Border” on page 3-28

### Summary of Parameters

Name	Description
<b>Program ID</b>	Name of the ActiveX control
<b>Connections</b>	Whether the ActiveX Control block has ports
<b>Input property</b>	Name of the property that is set when the ActiveX Control block receives a signal
<b>Initialization command</b>	Command that sets the initial conditions for the ActiveX Control block
<b>Other events and handlers</b>	Events that trigger an action by the ActiveX Control block
<b>Update command</b>	Function that Simulink invokes when it updates the block during the simulation

Name	Description
<b>In-block control</b>	Whether the ActiveX Control block displays an ActiveX control or is connected to an ActiveX Control block somewhere else
<b>Border</b>	Whether a border appears around the control

## Program ID

The **Program ID** parameter is the name of the ActiveX control displayed on the block. To determine the **Program ID** of other ActiveX controls, consult their documentation.

## Connections

The **Connections** parameter determines whether the block has an inport only, both an inport and an outport, or no ports. If the block has both an inport and an outport, then the values at both ports are the same.

## Input Property

The **Input property** parameter indicates the name of the block property whose value is set by the input signal. Each preconfigured Gauges Blockset block stores the block's current value in a property, as listed in the table below.

### Names of Input Properties

Library	Property Name
Angular Gauges	NeedleValue
LEDs	Value
Linear Gauges	<ul style="list-style-type: none"> <li>• BandStop (Min-Max Thermometer)</li> <li>• PointerValue (other pointer linear gauges)</li> <li>• Value (bar gauges)</li> </ul>
Numeric Displays	<ul style="list-style-type: none"> <li>• Value (Generic Numeric LED, Odometer, PlusMinus XX.XXX)</li> <li>• AlphaNumeric (others)</li> </ul>
On Off Gauges	Value
Percent Indicators	PortionValue

### Initialization Command

The **Initialization command** parameter is a string that sets the initial conditions of the ActiveX Control block.

The handle of the ActiveX Control block is hActX.

### Other Events and Handlers

The **Other events and handlers** parameter specifies actions taken by the ActiveX Control block when you perform a defined action on the ActiveX Control block. You must enter an event as an n-by-2 cell array. The first entry in each row must be the name of the ActiveX event. The second entry in each row must be the MATLAB callback to be executed.

For a list and description of supported events for an ActiveX control, consult the ActiveX control's help.

## Update Command

The **Update command** parameter is the name of the function that Simulink invokes when it updates the block during a simulation. Simulink passes these arguments to the function:

- The handle of the ActiveX control
- The current input value

The function is not invoked when you update the diagram.

## In-Block Control

The **In-block control** check box determines where the ActiveX Control block value is displayed. The ActiveX control can be on the block icon, in the same model window, in a different model, in a subsystem, or in a MATLAB figure.

If checked, the control whose name is specified in the **Program ID** field appears on the ActiveX Control block.

If cleared, the block is connected to the ActiveX control whose handle is specified in the **Handle location** field (this field appears when you clear the box):

- If the window is a MATLAB figure window, specify the name of a function whose return value is the figure handle. You can also specify initialization commands in the function to set the initial conditions of the ActiveX Control block.
- If the window contains a Simulink subsystem, the ActiveX control must be displayed on an ActiveX Control block contained in that subsystem. Specify the path of the ActiveX Control block on which the control is to appear.

For example, if a model named `my_model` has a subsystem called `sub_disp_signals` that contains an ActiveX Control block named `signal1`, the path is `my_model/sub_disp_signals/signal1`.

Using this feature is useful in a complex model that displays signals in multiple subsystems on ActiveX Control blocks. If you feed the signals into ActiveX Control blocks but display the ActiveX controls themselves in a separate system or window, it is not necessary to have the subsystems open

to see the results. For more information, see Chapter 4, “Placing ActiveX Controls in a Different Window”.

### **Border**

The **Border** check box determines whether the block displays a border around the ActiveX control.

---

**Note** Be careful when clearing this box, because the only way to move a block is to drag it with the border. Clearing the **Border** box renders the ActiveX Control block immovable.

---

# Placing ActiveX Controls in a Different Window

---

Placing ActiveX Controls in a Different Model (p. 4-2)

How to use a control located in a different model

Placing ActiveX Controls in a Subsystem (p. 4-7)

How to use a control located in a subsystem of a model

Placing ActiveX Controls in a Figure Window (p. 4-10)

How to use a control embedded in a MATLAB figure window

## Placing ActiveX Controls in a Different Model

### In this section...

“Example Overview” on page 4-2

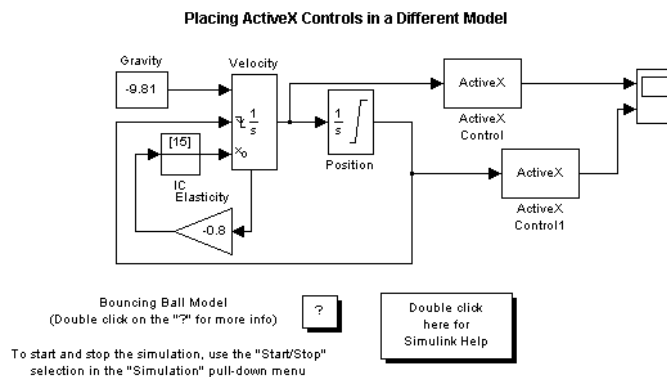
“Creating a Model Window Containing Gauges” on page 4-2

“Associating the Main Model with the Gauges” on page 4-4

### Example Overview

This example modifies the Simulink bounce demo by displaying the position and velocity signals on Gauges Blockset blocks contained in another model window.

To open the original demo model, enter `bounce` in MATLAB. To open the modified version, enter `gauges_bounce` in MATLAB. The modified version includes two ActiveX Control blocks on the signals that feed into the Scope block, as in the figure below.



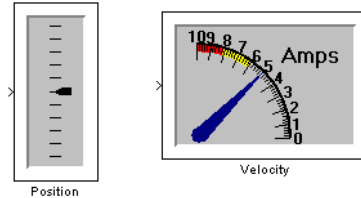
### Creating a Model Window Containing Gauges

Create a new model called `gauges_bounce_gui` and copy the following Gauges Blockset blocks into it:

- The Generic Linear Gauge block from the Linear Gauges library. Change the block's name to `Position`.



- The Amp Meter block from the Angular Gauges library. Change the block's name to Velocity.



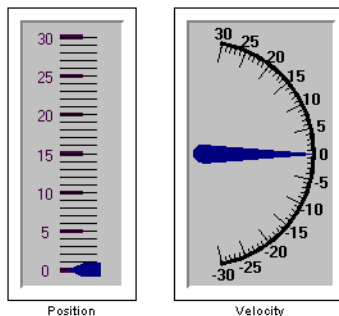
### Customizing the Gauges

If you want to customize the gauges, particularly the range of values that they can display, then use this optional procedure:

- 1 Open the ActiveX Control Properties dialog box for the Position (Generic Linear Gauge) block.
- 2 From the **Scales** panel, set **ScaleMax** to 30. This enables the gauge to display values between 0 and 30.
- 3 From the **Ticks** panel, set **StopValue** to 30, set **DeltaValue** to 5, check the **Label On/Off** check box, and set **Width** to 0.012. This creates labeled major ticks.
- 4 Still on the **Ticks** panel, set **Ticks** to 2, set **TickID** to 1, set **DeltaValue** to 1, set **Inner** to 0.4, and set **Outer** to 0.75. This creates a set of unlabeled minor ticks.
- 5 From the **Pointers** panel, click **Color**, choose the color that matches the pointer on the Velocity (Amp Meter) block, and click **OK**. Also, set **Value** to 0.
- 6 Click **OK**.
- 7 Open the ActiveX Control Properties dialog box for the Velocity (Amp Meter) block.
- 8 From the **Captions** panel, set **Captions** to 0. This removes the word Amps.

- 9 From the **Annulars** panel, set **Annulars** to 1. This removes the colored shading of the annular region.
- 10 From the **Needles** panel, set **Value** to 0. This moves the needle so that it points to zero.
- 11 From the **Scales** panel, set **Min** to -30, set **Max** to 30, select **Backward**, set **Start** to 10, and set **Stop** to 170. This causes the block to display values between -30 and 30 along the right half of a circle.
- 12 Still on the **Scales** panel, set **X** to -1.06 and set **Y** to 0.04. This helps center the control in the block.
- 13 From the **Ticks** panel, set **DeltaValue** to 5. This creates labeled major ticks.
- 14 Still on the **Ticks** panel, set **TickID** to 1 and set **DeltaValue** to 1. This creates unlabeled minor ticks.
- 15 Click **OK**.

You might also want to enlarge the blocks. They should now look like this.



### Associating the Main Model with the Gauges

Open the original bounce model and save it in your working directory as `gauges_bounce`. Insert two ActiveX Control blocks on the signals that feed into the Scope block. To connect the ActiveX Control blocks to the controls, make these changes in the Block Parameters dialog box in each of the ActiveX Control blocks:

- 1 Clear the **In-block control** check box, because the signal is being communicated between ActiveX Control blocks in one window and ActiveX Control blocks in another window. When you clear the **In-block control** check box, the number of fields on the dialog box changes.
- 2 In the **Input property** field, specify NeedleValue for the velocity display and PointerValue property for the position display. This property controls the current values of these ActiveX gauges. Doing this passes the value of the input signal to this property.

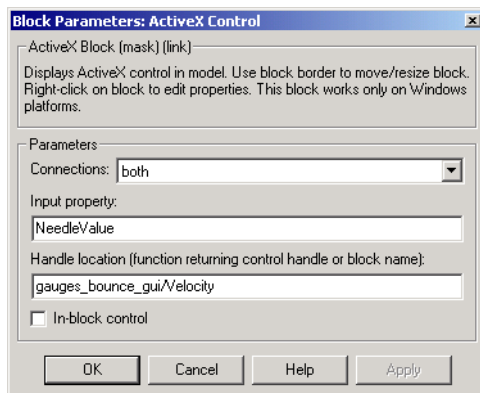
---

**Note** If you adapt this example to use the Strip Chart control instead, then set **Input property** to Y. For other controls in this blockset, set **Input property** to the value used in the corresponding parameter field in the library block.

---

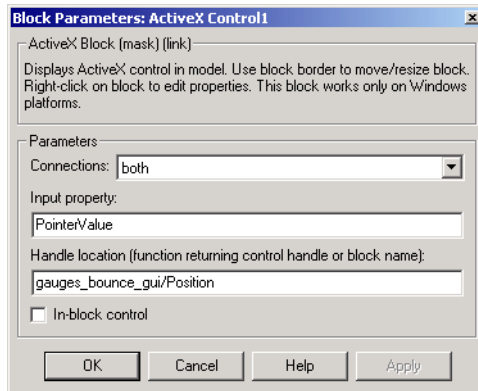
- 3 Specify the path of each gauge in the **Handle location** field. In this case, the new model is named gauges\_bounce\_gui.

The dialog boxes should look like those in the following figures. Now, when you simulate the main model window, the gauges in the auxiliary model window reflect the velocity and position of the bouncing ball.



### For Displaying the Velocity

## 4 Placing ActiveX Controls in a Different Window



### For Displaying the Position

## Placing ActiveX Controls in a Subsystem

### In this section...

“Example Overview” on page 4-7

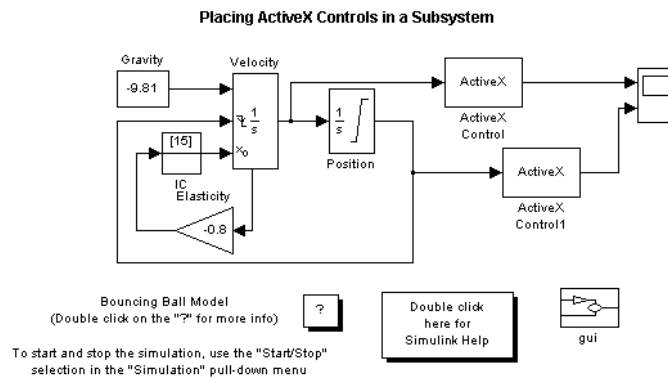
“Creating a Subsystem Containing Gauges” on page 4-7

“Associating Top-Level Blocks with the Subsystem” on page 4-8

### Example Overview

This example builds on the one described in “Placing ActiveX Controls in a Different Model” on page 4-2, but places the Gauges Blockset blocks in a subsystem of the main model rather than a different model. This approach simplifies operations such as saving and closing the system because the system involves only a single .mdl file.

To open a completed version of this example, enter `gauges_bounce_subsys` in the MATLAB Command Window. Notice that the model includes a subsystem called `gui` in the lower right.



### Creating a Subsystem Containing Gauges

To create the subsystem, follow these steps:

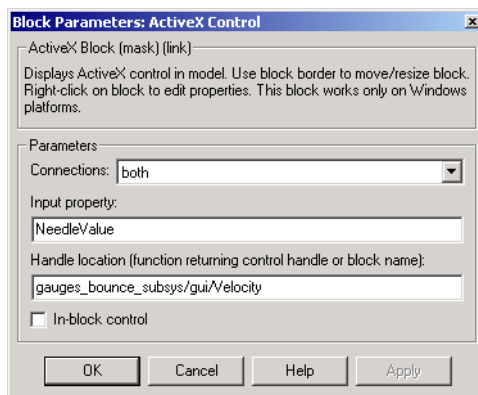
- 1 Open the bounce model and save it in your working directory as `gauges_bounce_subsys`.

- 2 Copy a Subsystem block from the Simulink Signals & Systems library into the model window. Change the block's name to gui.
- 3 Double-click the subsystem to open it.
- 4 Copy a Generic Linear Gauge block from the Linear Gauges library into the subsystem. Change the block's name to Position.
- 5 Copy an Amp Meter block from the Angular Gauges library into the subsystem. Change the block's name to Velocity.
- 6 In the Block Parameters dialog box for each of the two gauge blocks, set the **Connections** parameter to neither and clear the **Input property** edit field.

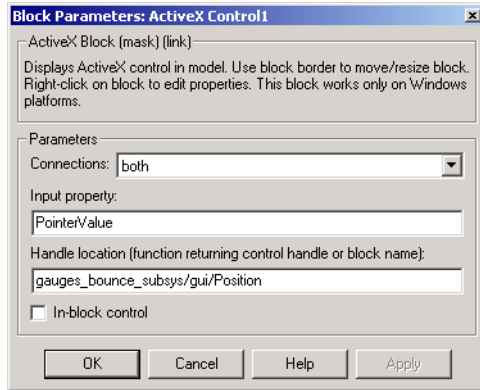
To customize the gauge blocks, see “Customizing the Gauges” on page 4-3.

### Associating Top-Level Blocks with the Subsystem

The procedure for associating the top-level ActiveX Control blocks with the gauge blocks that are inside the subsystem is similar to the procedure described in “Associating the Main Model with the Gauges” on page 4-4. The only difference is that the **Handle location** parameters have different values for a subsystem than for a separate model. The dialog boxes should look like those in the following figures.



#### For Displaying the Velocity



### For Displaying the Position

# Placing ActiveX Controls in a Figure Window

In this section...
“Example Overview” on page 4-10
“Creating Helper M-Files and Building the Model” on page 4-10
“Saving and Reopening the Model” on page 4-13

## Example Overview

In this example, a simple model displays the simulation time on an ActiveX control located in a figure window.

To open a completed copy of the model, enter `gauges_offblock` in MATLAB. Alternatively, follow the instructions below for building it yourself.

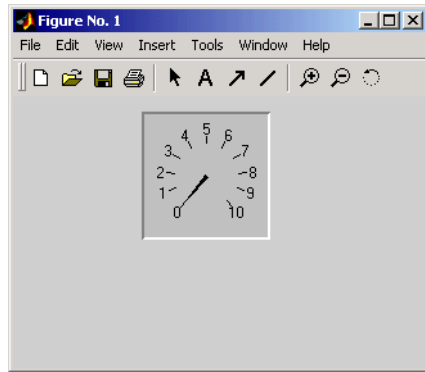
## Creating Helper M-Files and Building the Model

- 1 Create and execute an M-file called `gauges_gaugewindow` that consists of these statements.

```
f = figure;  
h = actxcontrol('mwagauge.agaugectrl.1',[100 100 100 100],f);
```

This M-file creates a figure window containing a Generic Angular Gauge, whose program ID is `mwagauge.agaugectrl.1`. The figure window looks like this (resized).





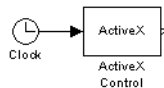
The M-file also specifies the position of the ActiveX control in the figure window. For more information about `actxcontrol`, see its reference documentation.

- 2 Create an M-file called `gauges_off_block` that consists of these statements.

```
function hactx = gauges_off_block
hactx = evalin('base','h');
```

The `gauges_off_block` function returns the handle of the ActiveX control that is to be connected to the ActiveX Control block.

- 3 In a new model window, connect a Clock block to an ActiveX Control block.



- 4 Open the ActiveX Control block to modify its parameters as follows:
  - a Clear the **In-block control** check box. The number of fields on the dialog box changes as a result.
  - b In the **Connections** field, select input. When you apply this change later, the output on the ActiveX Control block will disappear.
  - c In the **Input property** field, enter `NeedleValue`. During the simulation, the Generic Angular Gauges (that is, the ActiveX control referenced by

the block) sets the `NeedleValue` property to the value of the signal at the ActiveX Control block's input.

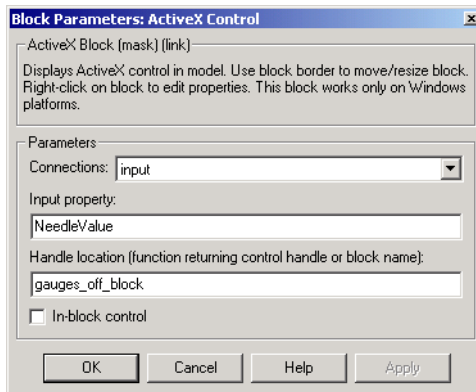
---

**Note** If you adapt this example to use the Strip Chart control instead, then set **Input property** to `Y`. For other controls in this blockset, set **Input property** to the value used in the corresponding parameter field in the library block.

---

**d** In the **Handle location** field, enter `gauges_off_block`.

As a result of these changes, the Block Parameters dialog box looks like this.



- 5** Click **OK**. MATLAB executes the `gauges_off_block` M-file, which returns the handle of the ActiveX control in the figure window.
- 6** Run the simulation. Notice that the Generic Angular Gauge reflects the clock time.

---

**Note** If you accidentally close the figure window before you are finished exploring the model, you can recreate it by executing `gauges_gaugewindow`.

---

## Saving and Reopening the Model

If you want to use this model in a different MATLAB session, then you must preserve both the model and the MATLAB commands that create the figure window and gauge. Here is an easy way to do this:

- 1 Save the model to give it a name.
- 2 If the model's name is `mymodel`, then use these commands in MATLAB to preserve the commands that create the figure window and gauge.

```
set_param('mymodel', 'PreLoadFcn', 'gauges_gaugewindow');  
save_system
```

Now, whenever you open `mymodel`, MATLAB automatically creates the figure that contains the gauge.



# Blocks — Alphabetical List

---

# Angular Gauges

---

**Purpose** Display input value on arc

**Description** Blocks in the Angular Gauges library show their input values graphically on a scale that lies along an arc of a circle. If the input value is greater than the scale's maximum or less than the scale's minimum, then the block displays the maximum or minimum value, respectively. To learn how to use and customize blocks in this library, see "Angular Gauges" on page 3-2.

---

**Note** Blocks in this library can display multiple needles. The Stop Watch and Analog Clock blocks display multiple needles by default. To learn how to control multiple needles simultaneously, see "Controlling Multiple Graphical Elements" on page 2-18.

---

## Blocks in the Library

The blocks in the Angular Gauges library are

- Amp Meter
- Analog Clock
- Compass
- Generic Angular Gauge
- Lower Left
- Lower Right
- Stop Watch
- Upper Left
- Upper Right
- Vacuum
- Volume

## Dialog Box

The ActiveX Control Properties dialog box governs the appearance and functionality of the ActiveX control that is embedded in the block. The table below lists the panels of the ActiveX Control Properties dialog box.

Panel	Purpose
<b>Annulars</b>	Display annular regions along the block's scale
<b>Background</b>	Configure the background and outline of the block
<b>Captions</b>	Display annotations on the block
<b>Digital</b>	Display the numerical value corresponding to the needle
<b>Frames</b>	Display a border on the block
<b>Fonts</b>	Define text styles (The <b>Captions</b> , <b>Digital</b> , and <b>Ticks</b> panels use the <b>FontID</b> property to reference the styles defined here.)
<b>Hubs</b>	Embellish a needle's axis of rotation
<b>Library</b>	Refer to property settings as a named collection
<b>Needles</b>	Display one or more needles on the block (The <b>Digital</b> panel uses the <b>NeedleID</b> property to reference the needles defined here.)
<b>Scales</b>	Define the ranges and locations of values displayed on the block (The <b>Annulars</b> , <b>Hubs</b> , <b>Needles</b> , and <b>Ticks</b> panels use the <b>ScaleID</b> property to reference the ranges defined here.)
<b>Ticks</b>	Display markers and/or numbers at intervals along the scale

# Angular Gauges

---

The Block Parameters dialog box governs the relationship between Simulink and the ActiveX control embedded in the block. See “Block Parameters for the ActiveX Control Block” on page 3-24 for details.



**Purpose** Display input value using two-state graphical elements

**Description** Blocks in the LEDs library use graphical elements to imitate light-emitting diodes (LEDs). Each block reflects its input value by setting one or more graphical elements to an on or off state. By default, the number of LEDs in the on state is the rounded value of the block's input. If the rounded value is nonpositive, then all LEDs are in the off state. If the rounded value exceeds the number of LEDs, then all LEDs are in the on state. To learn how to use and customize blocks in this library, see "LEDs" on page 3-5.

### **Blocks in the Library**

The blocks in the LEDs library are

- Circle Meter
- Generic LED
- Green Rect
- Horizontal Meter
- Rect Bitmap
- Red Rect
- Red Rectangle Plain
- Red Star
- Round Red
- Vertical Meter

### **Dialog Box**

The ActiveX Control Properties dialog box governs the appearance and functionality of the ActiveX control that is embedded in the block. The table below lists the panels of the ActiveX Control Properties dialog box.

# LEDs

---

<b>Panel</b>	<b>Purpose</b>
<b>Background</b>	Configure the background and outline of the block
<b>LEDs/General</b>	Define the number, arrangement, and behavior of LEDs on the block
<b>Library</b>	Refer to property settings as a named collection
<b>Style</b>	Define the appearance of LED graphical elements (The <b>LEDs/General</b> panel uses the <b>LEDStyleID</b> property to reference the styles defined here.)

The Block Parameters dialog box governs the relationship between Simulink and the ActiveX control embedded in the block. See “Block Parameters for the ActiveX Control Block” on page 3-24 for details.

**Purpose** Display input value on line

**Description** Blocks in the Linear Gauges library show their input value graphically on a scale that lies along a line. If the input value is greater than the scale's maximum or less than the scale's minimum, then the block displays the maximum or minimum value, respectively. To learn how to use and customize blocks in this library, see "Linear Gauges" on page 3-7.

---

**Note** Some blocks in this library can display multiple pointers. The Multiple Scales block displays multiple pointers by default. To learn how to control multiple pointers simultaneously, see "Controlling Multiple Graphical Elements" on page 2-18.

---

## Blocks in the Library

The blocks in the Linear Gauges library fall into two categories, as listed below:

Pointer Linear Gauges	Bar Gauges
Generic Linear Gauge	Generic Bar Gauge
Min-Max Thermometer	Reverse Sliding Scale
Mixer	Scaled Bar Gauge
Mixer Scale	Tank
Multiple Scales	Thermometer

## Dialog Box

### Properties of All Library Blocks

The ActiveX Control Properties dialog box governs the appearance and functionality of the ActiveX control that is embedded in the block. The table below lists the panels of the ActiveX Control Properties dialog box for all blocks in this library.

# Linear Gauges

---

<b>Panel</b>	<b>Purpose</b>
<b>Background</b>	Configure the background and outline of the block
<b>Captions</b>	Display annotations on the block
<b>Digital</b>	Display the numerical value corresponding to the pointer
<b>Fonts</b>	Define text styles (The <b>Captions</b> , <b>Digital</b> , and <b>Ticks</b> panels use the <b>FontID</b> property to reference the styles defined here.)
<b>Library</b>	Refer to property settings as a named collection
<b>Ticks</b>	Display markers and/or numbers at intervals along the scale

The Block Parameters dialog box governs the relationship between Simulink and the ActiveX control embedded in the block. See “Block Parameters for the ActiveX Control Block” on page 3-24 for details.

## **Properties Specific to Certain Library Blocks**

The ActiveX Control Properties dialog box of the pointer linear gauges has additional panels as in the table below.

<b>Panel</b>	<b>Purpose</b>
<b>Bands</b>	Display linear or rectangular regions along the block's scale

<b>Panel</b>	<b>Purpose</b>
<b>Pointers</b>	Display one or more pointers on the block (The <b>Digital</b> panel uses the <b>PointerID</b> property to reference the pointers defined here.)
<b>Scales</b>	Define the ranges and locations of values displayed on the block (The <b>Bands</b> , <b>Pointers</b> , and <b>Ticks</b> panels use the <b>ScaleID</b> property to reference the ranges defined here.)

The ActiveX Control Properties dialog box of the bar gauges has additional panels as in the table below.

<b>Panel</b>	<b>Purpose</b>
<b>Bar</b>	Define the appearance of the linear bar
<b>General</b>	Define the range and orientation of the block's scale
<b>Knob</b>	Define the appearance of the level indicator

# Numeric Displays

---

## Purpose

Display input value using LED digits or numbered wheels

## Description

Blocks in the Numeric Displays library show the numerical values of their inputs using graphical elements that imitate either numerals composed of light-emitting diode (LED) segments, or numbered wheels.

The display of the Odometer block imitates the numbered wheels of a car's odometer.

To learn how to use and customize blocks in this library, see “Numeric Displays” on page 3-11.

---

**Note** Blocks in this library can display alphanumeric characters if their **DisplayMode** properties are set to `AlphaNumeric`. Some blocks, such as the IRIG Format block, use this alphanumeric mode by default. However, Simulink signals are always double-precision *numeric* values. When using the alphanumeric mode, you can control the display via an M-file S-function that uses the COM support features in MATLAB. For an example of controlling a gauge using an M-file S-function, see “Updating Multiple Portions of a Pie Chart” on page 2-23.

---

## Blocks in the Library

The blocks in the Numeric Displays library are

- Generic Numeric LED
- HH:MM
- HH:MM:SS
- IRIG Format
- Odometer
- PlusMinus XX.XXX
- VCR Clock

## Dialog Box

The ActiveX Control Properties dialog box governs the appearance and functionality of the ActiveX control that is embedded in the block. The table below lists the panels of the ActiveX Control Properties dialog box.

<b>Panel</b>	<b>Purpose</b>
<b>Background</b>	Configure the background and outline of the block
<b>General</b>	Define the number, appearance, and arrangement of digits on the block
<b>Library</b>	Refer to property settings as a named collection

The Block Parameters dialog box governs the relationship between Simulink and the ActiveX control embedded in the block. See “Block Parameters for the ActiveX Control Block” on page 3-24 for details.

# On Off Gauges

---

**Purpose** Display two states

**Description** Blocks in the On Off Gauges library are two-state gauges. A block input of 0 corresponds to an “off” state and a block input of 1 corresponds to an “on” state. To learn how to use and customize blocks in this library, see “On Off Gauges” on page 3-14.

## Blocks in the Library

The blocks in the On Off Gauges library are

- Dip Switch Readout
- Happy Face
- Light Bulb
- Lock
- Mailbox
- On Off Readout

## Dialog Box

The ActiveX Control Properties dialog box governs the appearance and functionality of the ActiveX control that is embedded in the block. The table below lists the panels of the ActiveX Control Properties dialog box.

Panel	Purpose
Background	Configure the background and outline of the block
General	Determine how the button’s beveling (if visible) responds to a state change
Library	Refer to property settings as a named collection



<b>Panel</b>	<b>Purpose</b>
<b>Off</b>	Associate visual (text caption or image) and/or audio cues with the button's off state
<b>On</b>	Associate visual (text caption or image) and/or audio cues with the button's on state

If you use the question-mark icon to access the online help for the ActiveX controls in this library, then note that Simulink uses the ActiveX controls only for readout, with no mouse control.

The Block Parameters dialog box governs the relationship between Simulink and the ActiveX control embedded in the block. See “Block Parameters for the ActiveX Control Block” on page 3-24 for details.

# Percent Indicators

---

**Purpose** Display percentage or ratio, using linear or circular scale

**Description** Blocks in the Percent Indicators library convert their input values to percentages or ratios. They display the percentage or ratio graphically as either a segment on a linear scale or a sector of a circle. To learn how to use and customize blocks in this library, see “Percent Indicators” on page 3-15.

---

**Note** Blocks in this library can display multiple values simultaneously using percentages or ratios. The Pie Chart block displays multiple values by default. To learn how to display multiple values simultaneously, see “Controlling Multiple Graphical Elements” on page 2-18.

---

## Blocks in the Library

The blocks in the Percent Indicators library are

- Dynamic Pie
- Generic Percent
- Pie Chart
- Simple Light Blue

## Dialog Box

The ActiveX Control Properties dialog box governs the appearance and functionality of the ActiveX control that is embedded in the block. The table below lists the panels of the ActiveX Control Properties dialog box.

Panel	Purpose
Background	Configure the background and outline of the block
Frames	Display a border on the block

<b>Panel</b>	<b>Purpose</b>
<b>Library</b>	Refer to property settings as a named collection
<b>Misc</b>	Define the shape, orientation, and range of the block's scale
<b>Portions</b>	Define the number, appearance, and labeling style of regions that the block displays

The Block Parameters dialog box governs the relationship between Simulink and the ActiveX control embedded in the block. See “Block Parameters for the ActiveX Control Block” on page 3-24 for details.

# Strip Chart

---

**Purpose** Display stream of data in real time

**Description** The Strip Chart library contains a single block, the Strip Chart block. This block displays one or more signals while the simulation runs. It also enables you to zoom in or out. To learn how to use the Strip Chart block, see “Strip Chart” on page 3-18.

**Dialog Box** The ActiveX Control Properties dialog box governs the appearance and functionality of the ActiveX control that is embedded in the block. The table below lists the panels of the ActiveX Control Properties dialog box.

<b>Panel</b>	<b>Purpose</b>
<b>Background</b>	Configure the background and outline of the block
<b>Captions</b>	Display annotations on the block
<b>Fonts</b>	Define text styles (The <b>Captions</b> and <b>Stamps</b> panels use the <b>FontID</b> or <b>Stamp FontID</b> property to reference the styles defined here.)
<b>General</b>	Define the appearance and behavior of the underlying plotting area
<b>Library</b>	Refer to property settings as a named collection
<b>Stamps</b>	Define the appearance of a symbol that you can place on the control or on an individual plot
<b>Track Bands</b>	Define the number of colored bands displayed on each individual plot, and the appearance of each band

<b>Panel</b>	<b>Purpose</b>
<b>Tracks</b>	Define the number of individual plots, and the appearance of each (The <b>Track Bands</b> and <b>Variables</b> panels use the <b>TrackID</b> property to reference the tracks defined here.)
<b>Variables</b>	Determine which variables appear in each individual plot and how each variable is displayed.
<b>X Axis</b>	Determine what the values along the <i>x</i> -axis represent and how they are displayed

The Block Parameters dialog box governs the relationship between Simulink and the ActiveX control embedded in the block. See “Block Parameters for the ActiveX Control Block” on page 3-24 for details.

# Strip Chart

---

# Examples

---

Use this list to find examples in the documentation.

## **Getting Started**

“Example: Building a Simple Model” on page 1-10

“Example of Modifying Properties” on page 1-16

## **Modifying ActiveX Control Properties**

“Modifying the Displayed Range” on page 2-11

“Modifying Multiple Tick Marks” on page 2-13

## **Controlling Multiple Graphical Elements**

“Simulating a Multiple-Needle Stopwatch” on page 2-18

“Updating Multiple Portions of a Pie Chart” on page 2-23

## **Placing ActiveX Controls in Different Windows**

“Placing ActiveX Controls in a Different Model” on page 4-2

“Placing ActiveX Controls in a Subsystem” on page 4-7

“Placing ActiveX Controls in a Figure Window” on page 4-10



## A

- accessing ActiveX control blocks
  - from MATLAB 1-5
  - from Simulink 1-6
- active area of ActiveX control blocks 1-9
- ActiveX Control block
  - generic 3-20
- ActiveX control blocks
  - accessing preconfigured 1-5
  - areas of 1-9
  - categories of preconfigured 5-1
  - manipulating 1-9
  - parameters of 3-24
  - placing in different model window 4-2
  - printing 1-13
- ActiveX controls
  - multiple tick marks on 2-13
  - placing in figure window 4-10
  - range displayed on 2-11
  - saving customizations of 2-30
  - third-party 3-21
  - using your own 3-20
  - viewing properties of 1-15
- Amp Meter block 5-2
- Analog Clock block 5-2
- Angular Gauges library 3-2
  - customizing blocks in 3-2
  - reference 5-2
- annular regions
  - on angular gauges 3-3
- applying styles 2-6
- associating gauges with signals
  - in primary model window 4-4
  - in top level of model 4-8
- ax\_strip\_sfun.m 3-18

## B

- bar gauges
  - customizing 3-9

## bars

- appearance of 3-9
  - values along 3-9
- binary LEDs 3-6
- Block Parameters dialog box 3-24
- bmp files 1-13
- Border parameter 3-28
- borders of ActiveX control blocks 1-9

## C

- captions
  - displaying on blocks 2-9
- Captions panel 2-9
- Circle Meter block 5-5
- circular percentage displays 3-15
- code generation 1-3
- colors of ActiveX controls 3-22
- Compass block 5-2
- configuring Gauges Blockset 1-4
- connections
  - input and output 2-2
- Connections parameter 3-25
  - relation to Input and Output parameters 2-2
- Control Display Properties option 1-15
- creating styles 2-5

## D

- digits
  - characteristics of 3-11
  - in Odometer block 3-13
- Dip Switch Readout block 5-12
- displaying text on blocks 2-9
- Dynamic Pie block 5-14

## E

- examples
  - building a simple model 1-10
  - modifying displayed range 2-11

- modifying gauges for bounce demo 4-3
- modifying multiple tick marks 2-13
- modifying properties 1-16
- placing controls in different model 4-2
- placing controls in figure window 4-10
- placing controls in subsystem 4-7

external mode support 1-3

## F

figure windows

- placing controls in 4-10

fonts of text captions 2-10

## G

gauges\_bounce 4-2  
gauges\_bounce\_subsys 4-7  
gauges\_offblock 4-10  
gauges\_simple 1-11  
gaugeslib 1-5  
Generic Angular Gauge block 5-2  
Generic Bar Gauge block 5-7  
Generic LED block 5-5  
Generic Linear Gauge block 5-7  
Generic Numeric LED block 5-10  
Generic Percent block 5-14  
graphical elements

- multiple 2-18

Green Rect block 5-5

## H

Happy Face block 5-12  
HH:MM block 5-10  
HH:MM:SS block 5-10  
Horizontal Meter block 5-5

## I

ID properties 2-7

applying styles using 2-8  
defining styles using 2-7  
In-block control parameter 3-27  
indexing styles using ID properties 2-7  
input connections 2-2  
Input parameter 3-25  
input ports

- unused 2-2

IRIG Format block 5-10

## L

labeling

- parts of blocks 2-9
- percentage areas 3-16
- pointers 3-8

LEDs library 3-5

- customizing blocks in 3-5
- reference 5-5

libraries of Gauges Blockset

- accessing 1-5
- summary 5-1

Library panel 2-30  
Light Bulb block 5-12  
Linear Gauges library 3-7

- customizing blocks in 3-7
- reference 5-7

linear percentage scales 3-15  
Lock block 5-12  
Lower Left block 5-2  
Lower Right block 5-2

## M

Mailbox block 5-12  
Min-Max Thermometer block 5-7  
Mixer block 5-7  
Mixer Scale block 5-7  
models

- adding ActiveX control blocks to 1-9

- associating primary and auxiliary 4-4
  - printing 1-13
- mouse events
  - unresponsiveness to 3-21
- moving ActiveX control blocks 1-9
- multiple graphical elements 2-18
- multiple portions of pie chart 2-23
- Multiple Scales block 5-7
- multiple styles 2-3

## N

- needles
  - adding and deleting 3-4
  - customizing 3-2
  - multiple 2-18
- Numeric Displays library 3-11
  - customizing blocks in 3-11
  - customizing Odometer block in 3-13
  - reference 5-10

## O

- Odometer block 5-10
  - customizing 3-13
- On Off Gauges library 3-14
  - customizing blocks in 3-14
  - reference 5-12
- On Off Readout block 5-12
- opening Gauges Blockset library
  - from MATLAB 1-5
  - from Simulink 1-6
- Other events and handlers parameter 3-26
- output ports
  - unused 2-2

## P

- parameters of ActiveX control blocks 3-24
  - Border 3-28
  - Connections 3-25

- In-block control 3-27
- Input 3-25
- Other events and handlers 3-26
- Program ID 3-25
- Update command 3-27
- Percent Indicators library 3-15
  - customizing blocks in 3-15
  - reference 5-14
- percentage regions
  - adding and deleting 3-17
  - labeling 3-16
  - shape of 3-15
- Pie Chart block 5-14
- pie chart model 2-23
- pie charts 3-15
  - multiple portions in 2-23
- PlusMinus XX.XXX block 5-10
- pointers
  - adding and deleting 3-10
  - current value of 2-12
  - customizing 3-8
- portions of pie chart
  - multiple 2-23
- preconfigured ActiveX control blocks
  - accessing 1-5
  - categories 5-1
- printing ActiveX control blocks 1-13
- Program ID parameter 3-25
- properties of ActiveX controls 1-15

## R

- radial percentage scales 3-15
- ranges displayed on controls 2-11
  - in bounce demo 4-3
- Real-Time Workshop support 1-3
- Rect Bitmap block 5-5
- Red Rect block 5-5
- Red Rectangle Plain block 5-5
- Red Star block 5-5

- resizing ActiveX control blocks 1-9
- Reverse Sliding Scale block 5-7
- Round Red LED block 5-5
- running simulations 1-12

## **S**

- S-functions
  - for controlling gauge blocks 2-23
  - using with Strip Chart block 3-18
- Scaled Bar Gauge block 5-7
- selecting ActiveX control blocks 1-9
- Simple Light Blue block 5-14
- simulations
  - running 1-12
- Stop Watch block 5-2
- stopwatch model 2-18
- Strip Chart block 5-16
- Strip Chart library 3-18
  - reference 5-16
- styles
  - applying 2-6
  - creating 2-5
  - multiple 2-3
- support for multiple styles 2-4

## **T**

- Tank block 5-7

- text
  - displaying on blocks 2-9
- Thermometer block 5-7
- third-party ActiveX controls 3-21
  - color inheritance in 3-22
  - unresponsiveness to mouse events 3-21
- tick-mark properties 2-12
- titles
  - displaying on blocks 2-9
- troubleshooting
  - configuration of Gauges Blockset 1-4
  - use of third-party ActiveX controls 3-21

## **U**

- Update command parameter 3-27
- Upper Left block 5-2
- Upper Right block 5-2

## **V**

- Vacuum block 5-2
- VCR Clock block 5-10
- Vertical Meter block 5-5
- viewing properties of controls 1-15
- Volume block 5-2