Dials & Gauges Blockset

For Use with Simulink®

Computation

Visualization

Programming

User's Guide

Version 1

How to Contact The MathWorks:

508-647-7000 Phone

508-647-7001 Fax

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

`http://www.mathworks.com`	Web
`ftp.mathworks.com`	Anonymous FTP server
`comp.soft-sys.matlab`	Newsgroup

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

Dials & Gauges Blockset User's Guide

Printing History: September 1999 New for Version 1 (Online only)
September 2000 Revised for Version 1.1 (Release 12)

Contents

1 Quick Start

2 Using Model-Based Instrumentation

3 Advanced Topics

1

Quick Start

Introduction

About the Dials & Gauges Blockset

The Dials & Gauges Blockset supplies realistic, instrument-like user interface controls for Simulink® blocks.

The instrumentation is based on ActiveX technology and is only available on Microsoft Windows platforms.

Model-Based Instrumentation

The Dials & Gauges Blockset provides you with model-based instrumentation, which allows you to add graphical instruments directly to your model, using them as you would use blocks from the Sources and Sinks libraries.

Real-Time Workshop Support

You can use Real-Time Workshop® 4.0 or later to generate code from models that include Dials & Gauges Blockset blocks.

For dials, the code you generate contains static values (i.e., the value specified at the time of code generation). Gauges are ignored during code generation, except through the use of external mode (see below). If you want to manipulate dials and view the gauges, you can do so through the Real-Time Workshop's external mode.

External Mode Support

The Dials & Gauges Blockset support for external mode allows you to incorporate dials and gauges into any target that you can connect to through external mode (e.g., the xPC Target and Real-Time Windows Target environments; see the documentation for those products for details).

For more information about external mode, see the external mode section of the *Real-Time Workshop User's Guide*.

For More Information

This chapter helps you get a quick start for working with the Dials & Gauges Blockset by illustrating how to use model-based instrumentation in a simple Simulink model.

Chapter 2, “Using Model-Based Instrumentation” discusses tasks involved in using model-based instrumentation. It provides a more thorough discussion of tasks than Chapter 1.

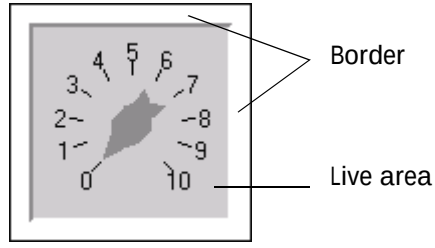
Chapter 3, “Advanced Topics” discusses advanced topics, such as importing your own ActiveX controls.

Simulink Block Diagrams and Active X Control Blocks

Before using ActiveX Control blocks with Simulink models, it is important to read this section to understand differences between using Simulink and using ActiveX Control blocks.

Selecting and Moving an ActiveX Control Block

To select an ActiveX Control block, click on its border. Once the ActiveX control has been created, the ActiveX Control block has a border and a “live” area within the border. This live area is the ActiveX control. To move a block, you must drag the block by the border, not the live area. This figure shows an ActiveX Control block and indicates the border.



Printing Limitation

The live areas of ActiveX Control blocks do not print with this release of the blockset. When you print a Simulink block with an ActiveX control, you will just see the outline of the block.

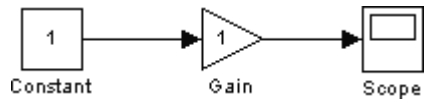
A Sample Model

The Basic Steps

Adding model-based instrumentation to your Simulink model using the ActiveX Control blocks shipped with Dials & Gauges involves a few basic steps, as introduced in the following sections:

The Model

This simple example of a model consists of a Constant block feeding into a Gain block, displaying its output on a Scope.

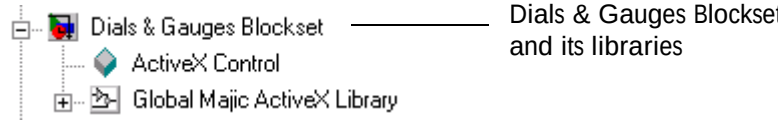


Using ActiveX Control blocks, you can provide instrument-like input and output, replacing the Constant and Scope blocks with ActiveX Control blocks.

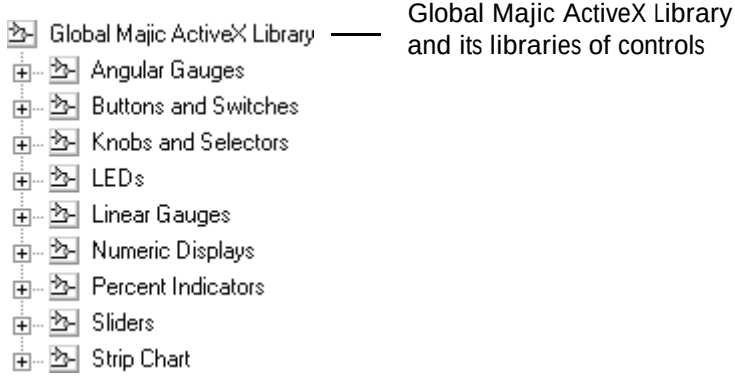
Accessing the Library of Preconfigured ActiveX Control Blocks

The Dials & Gauges Blockset contains over 40 preconfigured ActiveX Control blocks, via the Global Majic ActiveX Library. To access Global Majic ActiveX Control blocks:

1 Open the Dials & Gauges Blockset from the Simulink Library Browser by clicking on the plus sign to the left of the blockset name.



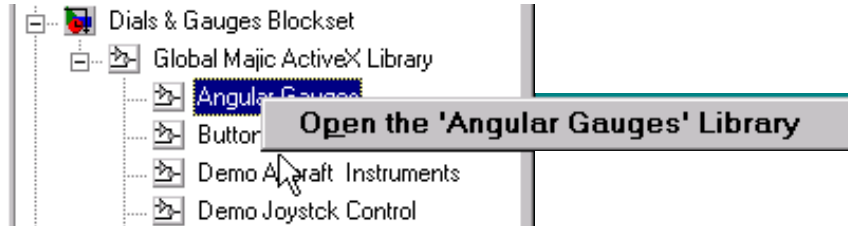
2 Open the Global Majic ActiveX Library to display its libraries of preconfigured ActiveX Control blocks.



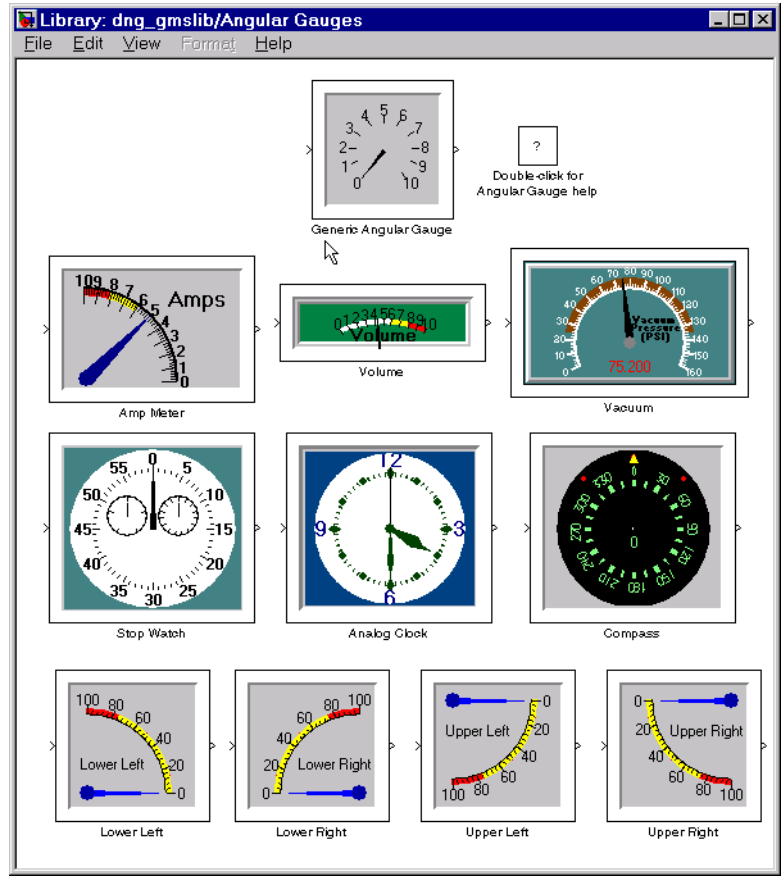
You can display a list of the ActiveX Control blocks in a library by expanding the library node in the browser tree. You can display the ActiveX Control blocks in their graphical form in a separate window by right-clicking on the library name and selecting **Open the Library**.

For example, the figure below shows the context menu that appears when you right-click on the Angular Gauges Library.

.



This figure shows the Angular Gauges Library contents.



Modifying the Model

In this example, a Generic Knob provides variable input to the Gain block, which passes its signal to a Lower Right control.

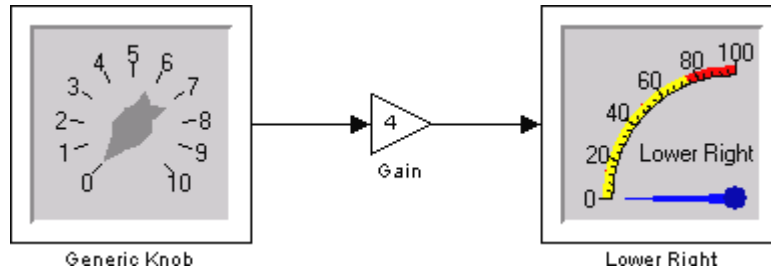
Note The model used in this example is described in "A Sample Model" on page 1-5.

---

1 Delete the Constant and Scope blocks. Change the Gain parameter to 4.

2 Copy the Generic Knob control block into the model. To do this, open the Knobs and Selectors Library, then drag the icon that appears to the left of the ActiveX Control block name (circled below) into the model.

3 From the Angular Gauges Library, drag the Lower Right gauge block into the model. Connect both ActiveX Control blocks to the Gain block. The modified model looks like this.

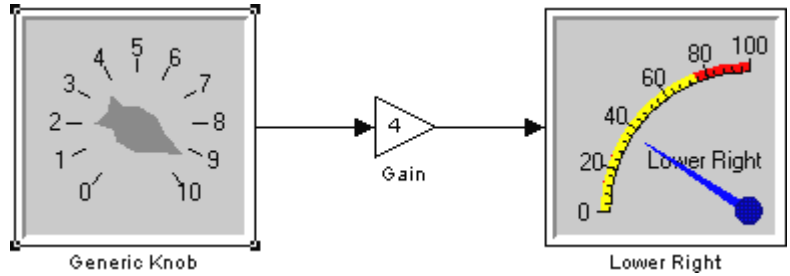


4 Set the simulation stop time to a reasonably large number to give you time to manipulate the Generic Knob's needle.

Running the Simulation

Run the simulation as you would for any Simulink model.

While the simulation is running, you can manipulate the needle of the Generic Knob and observe results on the Lower Right gauge block. This figure shows the model at the end of the simulation.

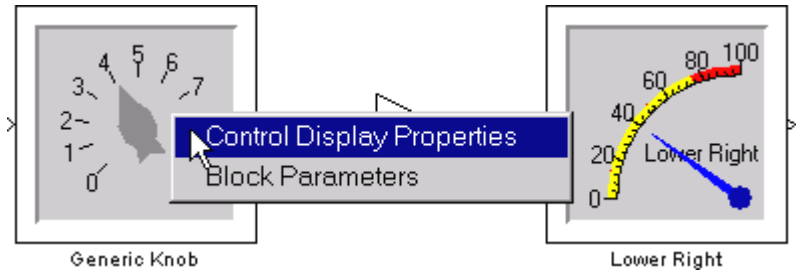


Modifying Properties On ActiveX Controls

Selecting the Control Display Properties Option

You modify ActiveX control properties by first either double-clicking on the active area of the block that contains the control or by right-clicking on the active area of the block, and then selecting the **Control Display Properties** option.

This figure shows the context menu that appears when you right-click the Generic Knob control.



Setting Properties

After selecting the **Control Display Properties** option, the **ActiveX Control Properties** dialog box appears, displaying tabs that specify the characteristics of the ActiveX control.

ActiveX controls have many properties. Changing the appearance of an ActiveX control may require changing several properties and can be quite complex. “Using ActiveX Control Blocks in a Model” on page 2-6 discusses how to make some common changes to ActiveX controls, such as changing the range of values displayed on an ActiveX control.

For information about specific properties, consult the control’s help by double-clicking on the Help icon that appears in each ActiveX Control block library. Most libraries provide one Help icon, although some provide more icons when the ActiveX Control blocks contained in the library have significant differences. Once in the Help window, click on the **Properties** link to display information about ActiveX Control block properties.

Saving the Model

When you save a model that contains ActiveX Control blocks, MATLAB® saves additional files that describe each ActiveX Control block. For example, if you save the model described in this chapter with the name `sample`, MATLAB saves the following files.

```
sample.mdl
sample@Generic_Knob.ax
sample@Lower_Right.ax
```

The files with the `.ax` extension describe the ActiveX Control blocks. Note that these files are not text files. They save the current state of the ActiveX control. If you have not changed any properties on the control, and are content with the default value for the control, you may delete or ignore the `.ax` files. They will reinitialize themselves to the exact state in which they are stored in the library.

If any ActiveX Control blocks are contained in subsystems, the subsystem name is also included in the filename. For example, if this sample model contains a subsystem named `SubSystem`, which contains the Generic Knob control, the following file is saved.

```
sample@SubSystem@Generic_Knob.ax
```

2

Using Model-Based Instrumentation

Introduction

This chapter describes how to use model-based instrumentation supported by the Dials & Gauges Blockset. This release of the blockset provides the Global Majic Systems ActiveX Library.

This chapter discusses the following topics:

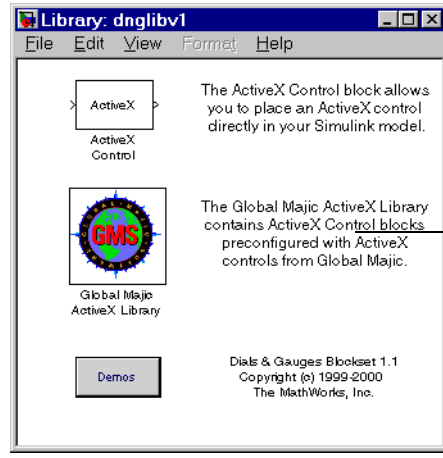
Accessing the Global Majic ActiveX Library

The Global Majic ActiveX Library is contained in the Model-Based Instrumentation Library, accessible from the Dials & Gauges Blockset. The two ways you can access the Global Majic ActiveX Library are described in these sections:

- "Using the dnglib Command" on page 2-3
- "Using the Simulink Library Browser" on page 2-4

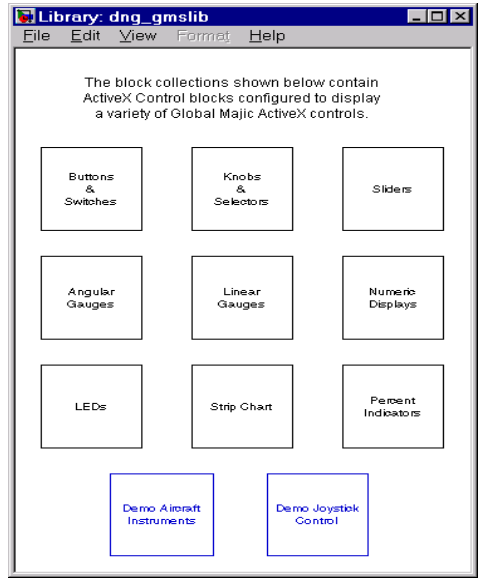
Using the dnglib Command

1 Enter the `dnglib` command in the MATLAB command window, which causes the following window to appear.



The Global Majic ActiveX Library contains libraries of controls.

2 Open the Global Majic ActiveX Library icon to access the libraries it contains.



Each subsystem gives you access to a different library of ActiveX Control blocks. Each of these libraries contains a Help block that provides access to the online help for that library.

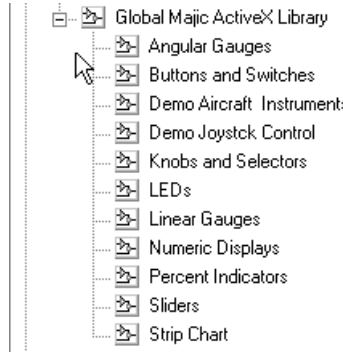
Using the Simulink Library Browser

You can use the Simulink Library Browser, instead of the `dnglib` command, to access the Global Majic ActiveX Library.

1 Open the Dials & Gauges Blockset to display the contents of the Dials & Gauges Blockset, including the ActiveX Control block and the Global Majic ActiveX Library.

Dials & Gauges Blockset contents

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



Global Majic ActiveX Library and its sublibraries of ActiveX Control blocks

Each library icon gives you access to its library of ActiveX Control blocks.

You can also view the ActiveX Control blocks in a library as instruments by right-clicking on the library name, and then selecting the **Open the Library** option.

Using ActiveX Control Blocks in a Model

This section discusses how to use ActiveX Control blocks in a model, including describing the following:

Adding ActiveX Control Blocks to a Model

To add ActiveX Control blocks to your Simulink model, drag them into the model window:

- If the Simulink Library Browser is open, you can drag the ActiveX Control block by dragging either the icon or the ActiveX Control block name.
- If the library is open (displaying ActiveX Control blocks as instruments), drag the block by its border, as described in Chapter 1, "Quick Start."

Note The rest of this section assumes that you selected the Generic Knob control block.

Modifying ActiveX Control Block Parameters

Specifying the Type of Connection and Input/Output Block Parameters

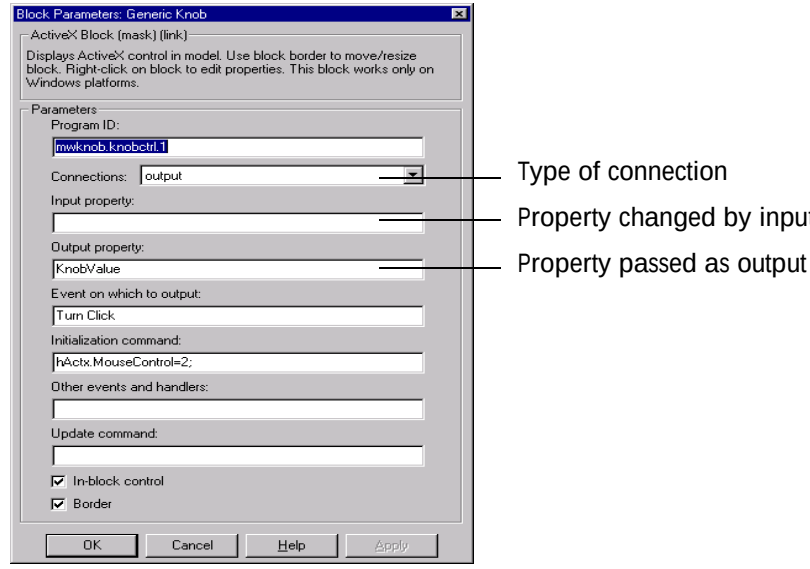
ActiveX Control blocks are drawn with an inport and an outport because all ActiveX Control blocks can be used as either input or output devices (or both). By default, however, each ActiveX Control block is set up to act as either an input device (with an output connection), or an output device (with an input connection). Unused ports are removed when the simulation starts running or when you update the block diagram.

You can determine whether an ActiveX Control block is set up to be used as an input or output device by examining the parameters of the block. Right-click on the ActiveX Control block, and then select the **Block Parameters** option.

Note For your own ActiveX controls built on top of the ActiveX Control block, you can also double-click on the border of the block to display the **Block Parameters** dialog box.

Block Parameters

The **Block Parameters** dialog box for the Generic Knob control block is shown below. Several of the fields are briefly described below.



Connections. This field determines the type of connection the ActiveX Control block is currently using:

- **input** indicates that the ActiveX Control block has an inport and receives a signal. The **Input property** indicates the ActiveX Control block's property whose value is changed by the input.

- **output** indicates that the ActiveX Control block has an outport and outputs a signal. The **Output property** indicates the ActiveX Control block's property whose value is output.
- **both** indicates that the ActiveX Control block has an inport and an outport and receives and outputs a signal.
- **neither** indicates that the ActiveX Control block has neither an inport nor an outport.

To specify a connection different than the ActiveX Control block's default setup, choose the **Connection** type and make sure that the **Input property** and **Output property** fields are filled in with the appropriate property name.

Input and Output Properties. Each ActiveX Control block stores its current value in a property. This table lists the names of those properties that are common to each ActiveX Control block in the given Global Majic ActiveX Control Library.

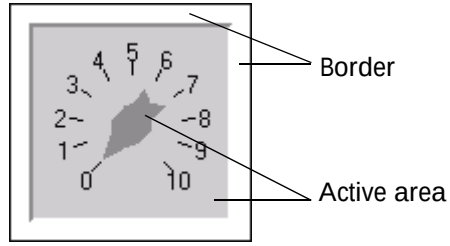
Library	Property Name
Angular Gauges	`NeedleValue`
Buttons & Switches	`Value`
Knobs & Selectors	`KnobValue` (Generic Knob) `Value` (Frequency Selector)
LEDs	`Value`
Linear Gauges	`BandStop` (Min-Max Thermometer) `PointerValue` (Others)
Numeric Displays	`Value`
Percent Indicators	`PortionValue`
Sliders	`Value`
Strip Chart	N/A (See "The Strip Chart Control Block" on page 3-15.)

Other Block Parameters. See "Summary of Dialog Box Fields and Check Boxes" on page 3-4 for information about the other fields and check boxes. You can also click the **Help** button to find out about other parameters.

Moving and Selecting ActiveX Control Blocks

The way you move and select ActiveX Control blocks is significantly different from how you move and select a Simulink block.

ActiveX Control blocks consist of a “live” area containing the actual control, and a border that surrounds that area. ActiveX Control blocks are live even when a simulation is not running; dragging the cursor anywhere within the control is interpreted as attempting to change the value of the control. This figure shows the border and the active area within the ActiveX Control block.



To move an ActiveX Control block, drag the border.

To select an ActiveX Control block, click on the border or “rubber-band” select the block.

To resize an ActiveX Control block, first select the block, and then drag one of the selection handles (as you would resize a Simulink block). You can only do this when the border is turned on.

Note Double-clicking on the border of a block that is supplied with the blockset (i.e., a built-in block) displays the ActiveX Control property sheet. Double-clicking on a *user-created* block displays the **Block Parameters** dialog box.

Modifying Properties of ActiveX Controls

ActiveX controls have many properties. Changing characteristics of the ActiveX control often requires that you change several properties. This manual does not provide details on how to modify all properties of the Global Majic Systems ActiveX controls. However, it does provide some instructions on how to make some common changes to ActiveX controls. See “Modifying ActiveX Control Properties” on page 2-11.

Running the Simulation

Run a simulation as you normally would run a Simulink model. Choose a simulation stop time large enough to give you time to manipulate input ActiveX Control blocks.

Saving the Model

When you save a model that contains ActiveX Control blocks, MATLAB saves additional files that describe the ActiveX Control blocks. For example, saving a model with the name `sample`, containing a Generic Knob control and a Lower Right gauge, saves these files.

```
sample.mdl
sample@Generic_Knob.ax
sample@Lower_Right.ax
```

If any ActiveX Control blocks are contained in subsystems, the subsystem name is indicated in the filename with the `@` character before it. For example,

```
sample@SubSystem@Generic_Knob.ax
```

Modifying ActiveX Control Properties

You modify ActiveX control properties in one of these ways:

- Double-clicking on the live part of the ActiveX Control block (not the border)
- Right-clicking on the ActiveX control, and then selecting the **Control Display Properties** option.

The **ActiveX Control Properties** dialog box appears, displaying tabs of properties for the ActiveX control that is contained by the selected ActiveX Control block.

Modifying some ActiveX control properties is straightforward. This section discusses how to modify two of the more complicated ActiveX control properties found on Global Majic controls, in the following subsections:

- "Modifying the Range Displayed on an ActiveX Control" on page 2-11
- "Modifying Multiple Tick Marks" on page 2-14

When you modify any properties of an ActiveX control, the ActiveX control is visually updated immediately. However, the changes are not permanent until you select **Save** or **Apply**; if you select **Cancel**, the changes will be undone.

For more information about properties and other characteristics of Global Majic ActiveX controls, see the online help provided by Global Majic. Each Global Majic Control library contains a Help icon that provides online help for that library. This includes a description of property categories and links to additional information, such as properties, events, and methods. To access ActiveX library help, right-click on the library name and select **Open library**.

"Saving a Customized ActiveX Control" on page 2-18 describes how to save an ActiveX control that you modify.

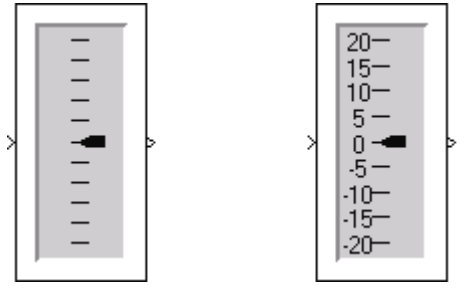
Modifying the Range Displayed on an ActiveX Control

Changing the range of values displayed on an ActiveX control involves adjusting these ActiveX control properties:

- *Scale* properties define the extent of the units displayed by the ActiveX control, the location of the ActiveX control's center, and the ActiveX control's start and stop positions.

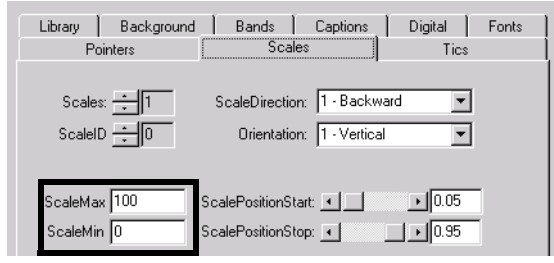
- *Needle* or *pointer* properties indicate the value.
- *Tick mark* properties define tick marks on the ActiveX control face, including start and stop values, the interval between tick marks, and label positions.

To illustrate how to use these properties to adjust the range of values displayed on an ActiveX control, 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).



Modifying the Scale Properties

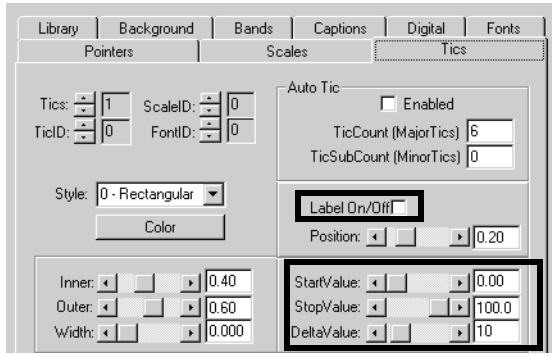
Click on the **Scales** tab to display the scales properties page. This figure shows the default scale properties for the Generic Linear Gauge.



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

Modifying the Tick Mark Properties

Click on the **Tics** tab to display the tick mark properties page. This figure shows the default tick mark properties.

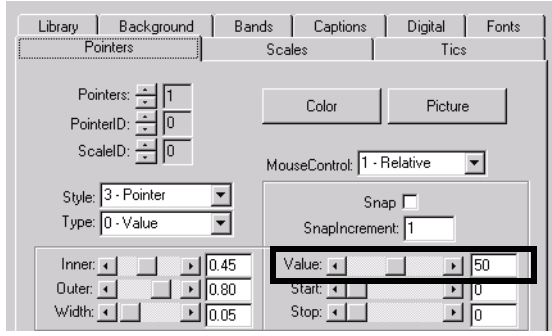


To show tick mark labels, check the **Label On/Off** check box.

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 control size.

Modifying the Needle or Pointer Properties

Click on 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.

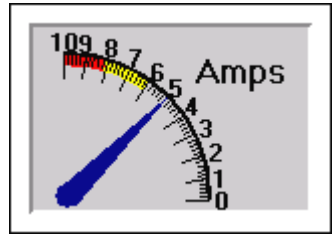
Modifying Multiple Tick Marks

Some characteristics can be repeated in an ActiveX control. For example, a single ActiveX control can display multiple needles or tick marks.

Note This release of the Dials & Gauges Blockset does not support the use of multiple needles or pointers to display more than one value. Passing a vector signal to an ActiveX Control displays only the first element of the vector. The Strip Chart control, however, is an exception to this because it *does* support vector input.

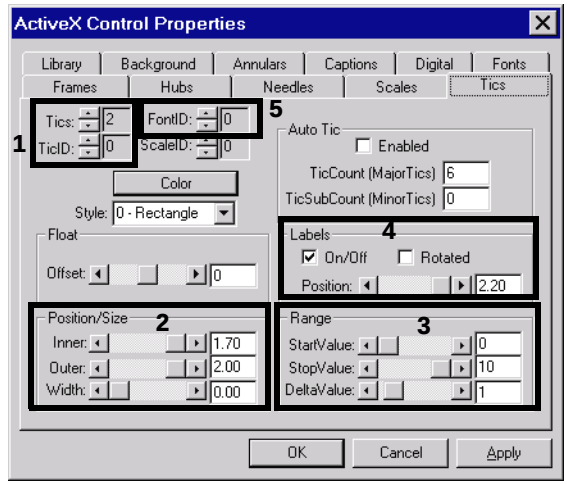
This example illustrates the use of multiple tick marks and the use of the ID property to manage them. This example, instead of modifying an ActiveX control, examines the default settings for a particular ActiveX control.

This figure shows the Amp Meter control. 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 on the Amp Meter control to display its properties dialog box. Select the **Tics** tab.



The `Tics` and `TicID` properties, in the box labeled **1**, are defined as follows:

- The `Tics` property specifies how many sets of tick marks are used by the control. For this control, this property is set to 2.
- The `TicID` 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 `TicID` 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 `TicID` 0.

Note When defining 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 of the tick marks is 0.00, the narrowest width.

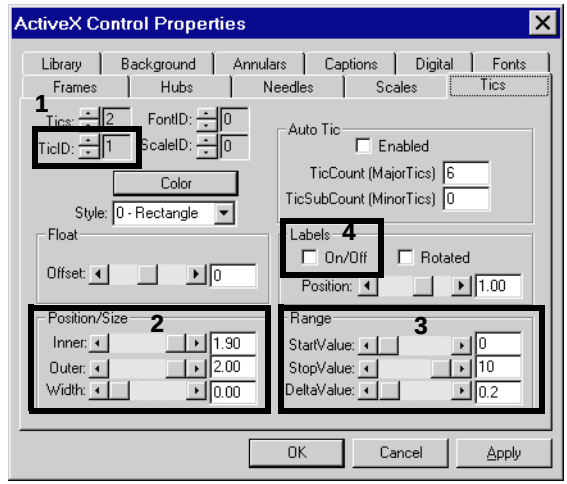
The Range properties, in the box labeled **3**, are defined as follows:

- The StartValue determines at which scale value the first tick mark is displayed. For these tick marks, the value is 0.
- The StopValue determines at which scale value the last tick mark is displayed. For these tick marks, the value is 10.
- The 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 control is used for the label. In this case, two font sets are defined. The first (FontID 0) is for the tick marks, the second (FontID 1) is for the caption, “Amps.”

To examine the second set of tick marks, change the TicID property value to 1 by clicking on the up arrow to the left of the value. The **Tics** 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` 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.

- The `StartValue` for these tick marks is `0`. The first short tick mark and the first long tick mark appear in the same place.
- The `StopValue` for these tick marks is `10`. The last short tick mark and the last long tick mark appear in the same place.
- The `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 the labels are displayed. For this set of tick marks, the labels are not displayed.

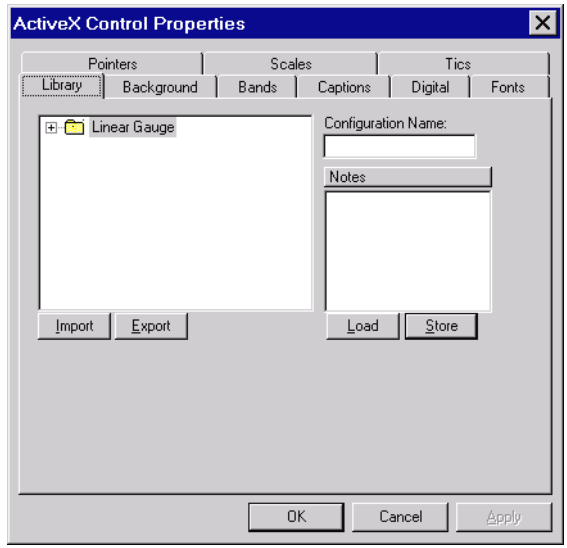
Note Be careful not to reduce the number of tick mark sets. If you do, the tick mark settings corresponding to the highest `TicID` will be removed from the control. Once they are removed, they must be recreated from the defaults.

Saving a Customized ActiveX Control

To save a modified ActiveX control for later use, follow these steps:

1 Make the modifications to the ActiveX control.

2 Select the **Library** tab of the **ActiveX Control Properties** dialog box.

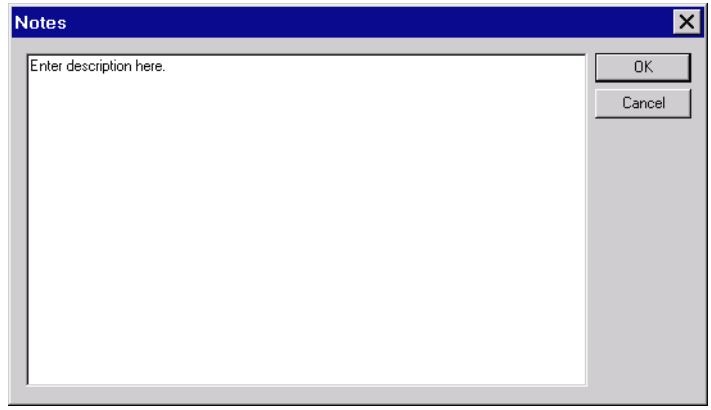


3 On this dialog box, assign a name to the modified ActiveX control by entering a new control name in the **Configuration Name** field.

Note If you leave this field blank, the new ActiveX control properties write over the previous settings, which means that you cannot access the original version except by reinstalling the blockset.

---

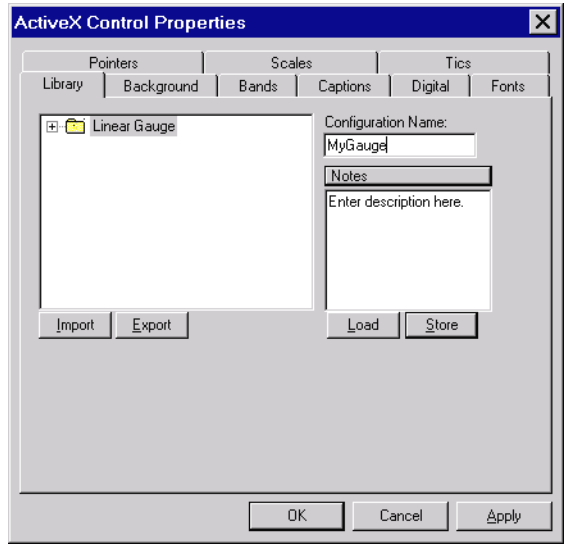
4 To provide textual information about the control, click on the **Notes** button. This dialog box appears, filled in with sample text.



5 Enter a description in the text area, then click **OK**.

6 Select the directory in which to store the modified ActiveX control by expanding the library hierarchy at the left. The new ActiveX control configuration is stored in the directory you select. Click the **Store** button, below the **Notes** field, and then click **OK** to accept all the changes and close the dialog box.

The figure below shows the dialog box with fields filled in. The modified ActiveX control configuration will be stored in the `Linear Gauge` directory.



Following this procedure enables you to save your customized copies of ActiveX controls on your computer. However, this method does not enable you to share these ActiveX control configurations.

To share these ActiveX control configurations, save the model. The ActiveX control configurations are saved in `.ax` files, which other users can access. You can also export configurations to `.gms` files for sharing. To do this, select a directory in the directory hierarchy and click **Export**. You can later use **Import** to access these configurations.

3

Advanced Topics

Introduction

This chapter discusses advanced uses for ActiveX Control blocks, in these sections:

Connecting to Your Own ActiveX Control Block

To connect your own ActiveX Control block to a Simulink model, you must use the ActiveX Control block. The ActiveX control is associated with, and is displayed on, the ActiveX Control block. Add the ActiveX Control block to the system where you want the control to appear.

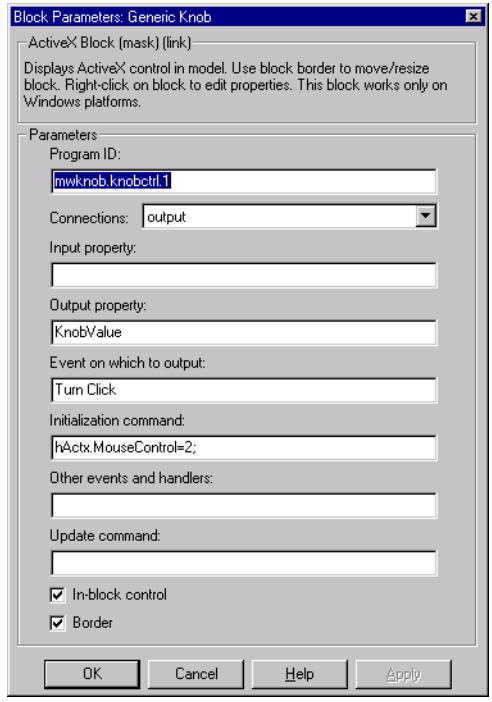
To configure the ActiveX Control block to display a specific ActiveX Control block, you need to know some of the programmatic features of the ActiveX Control block:

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

To connect an ActiveX Control block to a Simulink model:

1. Copy the ActiveX Control block to the model. The block is stored in the Model-Based Instrumentation Library.

2. Open the block to display the **Block Parameters** dialog box for the block.



3 Specify the appropriate values, described below.

Summary of Dialog Box Fields and Check Boxes

Here is a summary of the **Block Parameters** dialog box fields and check boxes. Each field and check box is described in more detail in sections following this summary:

- **Program ID** — the name of the ActiveX Control block
- **Connections** — whether the ActiveX Control block has an inport, an outport, both, or neither
- **Input property** — the name of the property that is set when the ActiveX Control block receives a signal
- **Output property** — the name of the property whose value is passed as the output signal

- **Event on which to output** — the events that will cause the value of the output to be updated
- **Initialization command** — the command that sets the initial conditions for the ActiveX Control block
- **Other events and handlers** — the events that trigger an action by the ActiveX Control block
- **Update command** — the command that gets executed when the block is updated (during the simulation)
- **In-block control** — whether the ActiveX Control block displays an ActiveX Control block or is connected to an ActiveX Control block somewhere else
- **Border** check box — whether a border appears around the control

Program ID

The **Program ID** is the name of the ActiveX Control block displayed on the block. To determine the **Program ID** of other ActiveX Control blocks, consult the documentation for the ActiveX Control block.

Connections

The **Connections** property determines whether the block has an inport, an outport, both, or neither. If the block is connected to a signal, this choice should indicate whether the signal is input to the block, output from the block, passed through the block (both), or not connected directly to the block (neither).

Input Property

The **Input property** indicates the name of the ActiveX Control block property whose value is set by the input signal. For the name of the property, see the table in “Adding ActiveX Control Blocks to a Model” on page 2-6.

Output Property

The **Output property** is the name of the ActiveX Control block property whose value is output from the block. For the name of the property, see the table in “Adding ActiveX Control Blocks to a Model” on page 2-6.

Event on Which to Output

The **Event on which to output** is a comma- or space-separated list of events that indicate a change in the output value. After the simulation starts, the output property will only be changed upon one of these events.

The table below summarizes the events associated with blocks that can have an event on which to output.

Table 3-1: Event Associated With Each Block

Block	Associated Event
Demo Joystick Control	JoyMove
Frequency Selector	Change
Generic Knob	Turn, Click
Generic Slider	Slide, Change
Generic Toggle	Click

Initialization Command

The **Initialization command** property is a string that sets the initial conditions of the ActiveX Control block. The string is evaluated during the model initialization stage.

The handle of the ActiveX Control block is `hActX`.

Other Events and Handlers

The **Other events and handlers** property 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 `nx2` 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 block library, consult the help for the library.

Update Command

The **Update command** property is the string that is evaluated by MATLAB when the block is updated during a simulation. The command is not executed when you update the diagram.

In-Block Control

The **In-block control** check box determines whether the ActiveX Control block displays an ActiveX Control block or is connected to an ActiveX Control block somewhere else. The ActiveX Control block can be in the same model window or in a different subsystem, model, or MATLAB figure.

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

If unchecked, the block is connected to the ActiveX control whose handle is specified in the **Handle location** field (this field appears when you uncheck 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 block 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 Control blocks themselves in a separate system or window, it is not necessary to have the subsystems open to see the results. For more information, see the section “Accessing an ActiveX Control Block in a Different Model Window” on page 3-10.

Border

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

Note Be careful when unchecking this box because the only way to move a block is to drag it with the border. Displaying no border will result in an ActiveX Control block that cannot be moved.

Notes on Third-Party ActiveX Control Blocks

Note that ActiveX Control blocks that try to determine their color by inheriting from the window in which they reside will not work properly in Simulink.

More specifically, ActiveX Control blocks that send the `WM_CTLCOLOR` message to their parent have this problem. `WM_CTLCOLOR` is a Microsoft Windows message sent by an ActiveX Control block to allow the parent container to determine the color used by the control.

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

Note that certain ActiveX Control blocks do not handle typical mouse events (double-click, right-click, etc.). These ActiveX Control blocks will appear "uneditable" when used with the Dials & Gauges Blockset. Double-clicking or right-clicking on these ActiveX Control blocks has no effect. To edit this type of ActiveX Control block, you must first select the block so that it is current in the Simulink diagram. Then type the following command at the MATLAB prompt.

```
propedit(get_param(gcb,'userdata'))
```

This command opens the property editor dialog for that control.

See the MATLAB ActiveX 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 funtion must take multiple arguments, of which the first one will be the handle to the ActiveX control. For example, a simple funtion to open the property editor of a control would look 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. Assume the `keypress` event is valid, the event and handler entry would look like this:

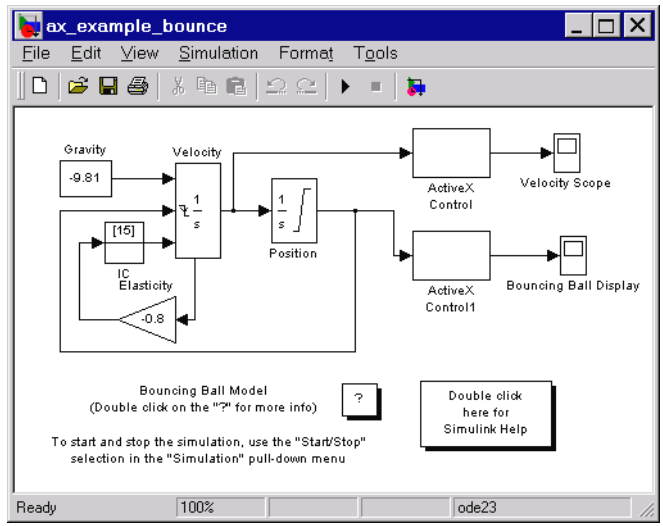
```
{'keypress','axeventhandler'}
```

To use the error checking code already written for the Dials & Gauges Blockset, you can use the same handlers we use by entering `ax_block_dclk` for events that should open the property editor (note that the editor will not open when the simulation is running). For example, to make a keystroke open the property editor (assuming the `keypress` event isvalid), enter the event and handler pair as follows:

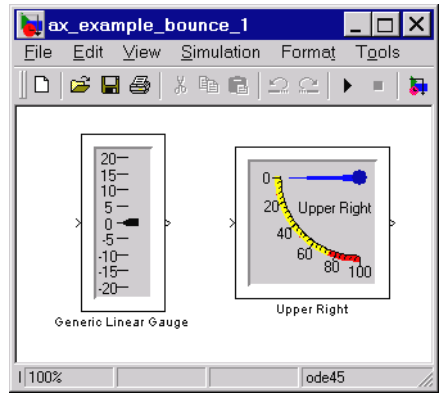
```
{'keypress','ax_block_dclk'}
```

Accessing an ActiveX Control Block in a Different Model Window

This sample model modifies the bounce demo by displaying the position and velocity signals on ActiveX Control blocks contained in another model window. Modify the model by adding two ActiveX Control blocks before the Velocity Scope and the Bouncing Ball Display Scope. The modified model looks like this.



Create a new model and copy two ActiveX gauges into it.



To connect the ActiveX Control blocks to the controls, open the ActiveX Control blocks and modify the **Block Parameters** dialog box:

1. Uncheck 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 uncheck the **In-block control** check box, the number of fields on the dialog box changes.

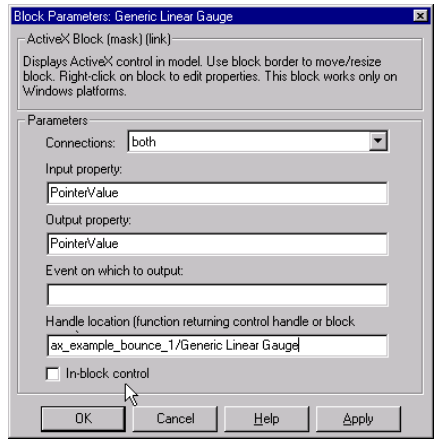
2. In the **Input property** field, specify the `NeedleValue` property for the position display and `PointerValue` for the velocity. This property controls the current value of these ActiveX gauges. Doing this passes the value of the input signal to this property.

3. In the **Output property** field, specify the same property. Doing this passes the value of this property to the scopes. The library copy of the Generic Linear Gauge has a minimum value of 0 and a maximum value of 100. For more information about changing ActiveX Control block properties, see “Modifying ActiveX Control Properties” on page 2-11.

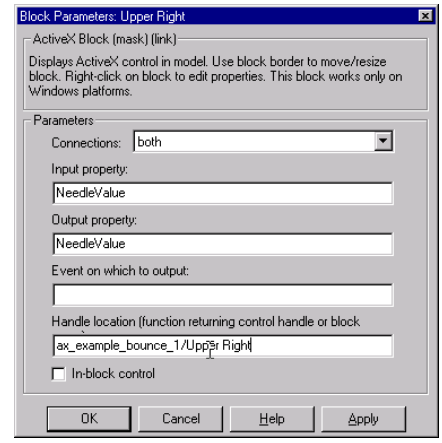
4. Specify the path of each gauge in the **Handle location** field. In this case, the new model is named `ax_example_bounce_1`.

The dialog boxes should now look like this.

For displaying the velocity

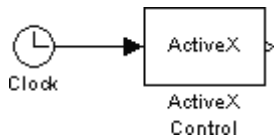


For displaying the location



Accessing an ActiveX Control Block in a Figure Window

In this example, a simple model displays the simulation time on an ActiveX Control block located in a figure window. The model looks like this.



1 Create a figure window to hold the ActiveX Control block.

```
f = figure;
```

2 Create the ActiveX Control block that is to appear in the figure window, a Generic Angular Gauge, whose program ID is `mwagauge.agaugectrl.1`. The statement also specifies the position of the ActiveX Control block in the figure window.

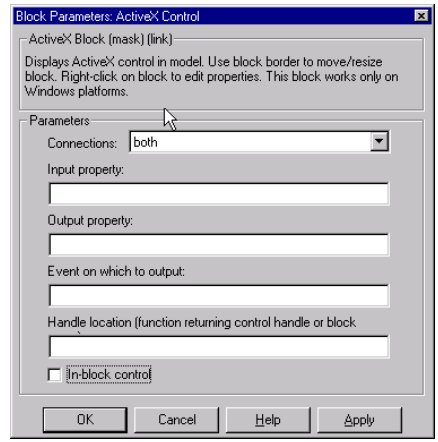
```
h = actxcontrol('mwagauge.agaugectrl.1', [100 100 100 100], f);
```

See the reference documentation for `actxcontrol`.

3 Create an M-file called `off_block` that consists of these statements.

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

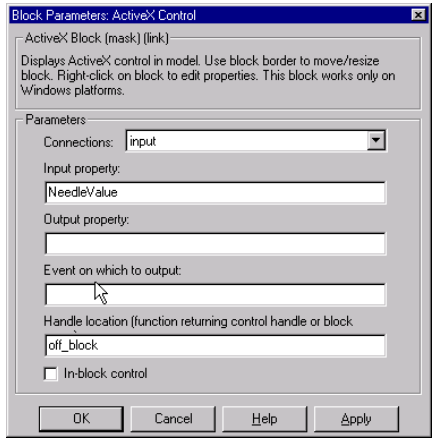
4 Open the ActiveX Control block to modify its parameters. First, uncheck the **In-block control** check box. The dialog box now looks like this.



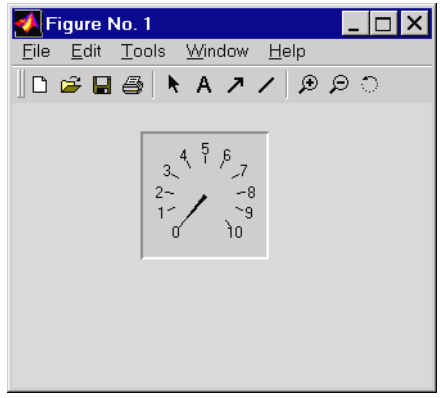
5 In the **Connections** field, select **input** to delete the outport on the ActiveX Control block.

6 In the **Input property** field, enter `NeedleValue`. When a signal is received at the ActiveX Control block's inport, this property of the ActiveX Control block referenced by the ActiveX Control block (the Generic Angular Gauge) is set with the signal value.

7 In the **Handle location** field, enter `off_block`, an M-file containing a function that returns the handle of the ActiveX Control block that is to be connected to the ActiveX Control block (the code appears below). With the fields filled in, the **Block Parameters** dialog box looks like this.



8 Click **OK**. MATLAB executes the `off_block` M-file, which returns the handle of the ActiveX Control block in the figure window. The figure window looks like this (resized).



9 Run the simulation. Notice that the clock time is passed to the Generic Angular Gauge.

The Strip Chart Control Block

The interface to the Strip Chart control block is different from the interface to the other Global Majic controls in the Dials & Gauges Blockset. The Strip Chart block has a set of properties that can be used to configure the block, which is the same for all Global Majic Control blocks. However, to plot data on the chart, you must invoke methods for the block. You can use the MATLAB command `invoke` to call or invoke methods of ActiveX Control blocks and pass arguments to those methods.

An M-file S-function provided with the Dials & Gauges Blockset plots data on the Strip Chart by using the `invoke` method. This S-function can serve as an example of communicating with any ActiveX control from the MATLAB language through an S-function. The file is named `ax_strip_sfun.m` and is located in the main Dials & 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 the Dials & Gauges Blockset provides the handle to 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.

This example can be used as a starting point for interfacing the ActiveX Control block with other ActiveX controls. Note that there are more options available with S-functions than shown in this example. See *Writing S-Functions* for more details on writing your own S-functions.

Index