

**Simulink® Real-Time™**

Reference



**MATLAB® & SIMULINK®**

R2020a



## How to Contact MathWorks



Latest news: [www.mathworks.com](http://www.mathworks.com)  
Sales and services: [www.mathworks.com/sales\\_and\\_services](http://www.mathworks.com/sales_and_services)  
User community: [www.mathworks.com/matlabcentral](http://www.mathworks.com/matlabcentral)  
Technical support: [www.mathworks.com/support/contact\\_us](http://www.mathworks.com/support/contact_us)



Phone: 508-647-7000



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

*Simulink® Real-Time™ Reference*

© COPYRIGHT 2002–2020 by The MathWorks, Inc.

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

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

### Trademarks

MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See [www.mathworks.com/trademarks](http://www.mathworks.com/trademarks) for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

### Patents

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

March 2007	Online only	New for Version 3.2 (Release 2007a)
September 2007	Online only	Updated for Version 3.3 (Release 2007b)
March 2008	Online only	Updated for Version 3.4 (Release 2008a)
October 2008	Online only	Updated for Version 4.0 (Release 2008b)
March 2009	Online only	Updated for Version 4.1 (Release 2009a)
September 2009	Online only	Updated for Version 4.2 (Release 2009b)
March 2010	Online only	Updated for Version 4.3 (Release 2010a)
April 2011	Online only	Updated for Version 5.0 (Release 2011a)
September 2011	Online only	Updated for Version 5.1 (Release 2011b)
March 2012	Online only	Revised for Version 5.2 (Release 2012a)
September 2012	Online only	Revised for Version 5.3 (Release 2012b)
March 2013	Online only	Revised for Version 5.4 (Release 2013a)
September 2013	Online only	Revised for Version 5.5 (Release 2013b)
March 2014	Online only	Revised for Version 6.0 (Release 2014a)
October 2014	Online only	Revised for Version 6.1 (Release 2014b)
March 2015	Online only	Revised for Version 6.2 (Release 2015a)
September 2015	Online only	Revised for Version 6.3 (Release 2015b)
March 2016	Online only	Revised for Version 6.4 (Release 2016a)
September 2016	Online only	Revised for Version 6.5 (Release 2016b)
March 2017	Online only	Revised for Version 6.6 (Release 2017a)
September 2017	Online only	Revised for Version 6.7 (Release 2017b)
March 2018	Online only	Revised for Version 6.8 (Release 2018a)
September 2018	Online only	Revised for Version 6.9 (Release 2018b)
March 2019	Online only	Revised for Version 6.10 (Release 2019a)
September 2019	Online only	Revised for Version 6.11 (Release 2019b)
March 2020	Online only	Revised for Version 6.12 (Release 2020a)



## Configuration Parameters

<b>Simulink Real-Time Options Pane</b> .....	<b>1-2</b>
Configuration .....	1-2
Tips .....	1-3
To get help on an option .....	1-3
<b>Execution mode</b> .....	<b>1-4</b>
Settings .....	1-4
Command-Line Information .....	1-4
<b>Real-time interrupt source</b> .....	<b>1-5</b>
Settings .....	1-5
Tips .....	1-5
Command-Line Information .....	1-5
<b>I/O board generating the interrupt</b> .....	<b>1-6</b>
Settings .....	1-6
Command-Line Information .....	1-6
<b>PCI slot (-1: autosearch) or ISA base address</b> .....	<b>1-7</b>
Settings .....	1-7
Tip .....	1-7
Command-Line Information .....	1-7
See Also .....	1-7
<b>Monitor Task Execution Time</b> .....	<b>1-8</b>
Settings .....	1-8
Command-Line Information .....	1-8
See Also .....	1-8
<b>Signal logging data buffer size in doubles</b> .....	<b>1-9</b>
Settings .....	1-9
Tips .....	1-9
Command-Line Information .....	1-9
<b>Double buffer parameter changes</b> .....	<b>1-10</b>
Settings .....	1-10
Tips .....	1-10
Command-Line Information .....	1-10
<b>Load a parameter set from a file on the designated target file system</b> ..	<b>1-11</b>
Settings .....	1-11
Dependencies .....	1-11
Command-Line Information .....	1-11
See Also .....	1-11

<b>File name</b> .....	<b>1-12</b>
Settings .....	<b>1-12</b>
Tip .....	<b>1-12</b>
Dependencies .....	<b>1-12</b>
Command-Line Information .....	<b>1-12</b>
<b>Generate INCA/CANape extensions (disables the Simulation Data Inspector and Dashboard blocks)</b> .....	<b>1-13</b>
Settings .....	<b>1-13</b>
Command-Line Information .....	<b>1-13</b>
See Also .....	<b>1-13</b>
<b>Enable Stateflow animation</b> .....	<b>1-14</b>
Settings .....	<b>1-14</b>
Command-Line Information .....	<b>1-14</b>
See Also .....	<b>1-14</b>
<b>Arm when connecting to target</b> .....	<b>1-15</b>
Settings .....	<b>1-15</b>
Command-Line Information .....	<b>1-15</b>
See Also .....	<b>1-15</b>

## TLC Options Parameters

### 2

<b>TLC Command-Line Options</b> .....	<b>2-2</b>
xPCMaxOverloads .....	<b>2-2</b>
xPCMaxOverloadLen .....	<b>2-3</b>
xPCStartupFlag .....	<b>2-3</b>
xPCModelStackSizeKB .....	<b>2-4</b>
xpcCPUClockPoll .....	<b>2-4</b>
SLRTFTZOFF .....	<b>2-4</b>

## Simulink Real-Time Explorer

### 3

## Simulink Real-Time Explorer Instrumentation

### 4

<b>Instrumentation for Real-Time Applications</b> .....	<b>4-2</b>
Instrument Selection and Binding .....	<b>4-4</b>
Display Instruments .....	<b>4-5</b>
Tuning Instruments .....	<b>4-6</b>
Layout Elements .....	<b>4-7</b>
Graphical Properties .....	<b>4-9</b>

<b>Explorer Configuration Exported to Run Outside MATLAB</b> .....	<b>4-13</b>
Execution Environment .....	<b>4-13</b>
Signal Groups .....	<b>4-13</b>
Parameter Groups .....	<b>4-13</b>
Instrument Panels .....	<b>4-13</b>
Window Layout .....	<b>4-14</b>
Limitations as Standalone Executable .....	<b>4-14</b>
<b>View Signal Waveforms with Scope Instrument</b> .....	<b>4-15</b>
Create Instrument Panel .....	<b>4-15</b>
Configure Slider Instrument for Parameter Tuning .....	<b>4-16</b>
Configure Scope Instrument with Default Triggering .....	<b>4-17</b>
Run Instrumented Model .....	<b>4-18</b>
<b>View Signal Waveforms with Signal Triggered Scope Instrument</b> .....	<b>4-21</b>
Preconditions .....	<b>4-21</b>
Configure Scope Instrument .....	<b>4-21</b>
Run Instrumented Model .....	<b>4-22</b>
<b>Add Instrument Panel to Tank Model</b> .....	<b>4-25</b>
<b>Save Environment Properties</b> .....	<b>4-35</b>
<b>Save and Load Instrument Panels</b> .....	<b>4-36</b>
<b>Save and Restore Layouts</b> .....	<b>4-37</b>
<b>Prepare Explorer Environment for Export</b> .....	<b>4-38</b>
<b>Prepare Instrument Panel Configuration for Export</b> .....	<b>4-39</b>
<b>Export Explorer Configuration</b> .....	<b>4-41</b>
<b>Unpack and Run Standalone Configuration</b> .....	<b>4-42</b>

## Simulink Real-Time Explorer Instruments

**5**

## Simulink Real-Time Instrumentation Object

**6**

<b>Instrumentation Apps for Real-Time Applications</b> .....	<b>6-2</b>
--------------------------------------------------------------	------------

## Target Computer Command-Line Interface Reference

### 7

<b>Target Computer Commands</b> .....	7-2
Target Object Function Commands .....	7-2
Target Object Property Commands .....	7-2
Scope and Video Object Function Commands .....	7-3
Scope Object Property Commands .....	7-4
Aliasing with Variable Commands .....	7-7

## Simulink Real-Time Performance Advisor Checks

### 8

<b>Simulink Real-Time Performance Advisor Checks</b> .....	8-2
To get help on an option .....	8-2
See Also .....	8-3
<b>Baseline</b> .....	8-4
See Also .....	8-4
<b>System Target File Compatibility</b> .....	8-5
See Also .....	8-5
<b>Profiling Settings</b> .....	8-6
See Also .....	8-6
<b>Check Target</b> .....	8-7
See Also .....	8-7
<b>Real-Time Performance Baseline</b> .....	8-8
See Also .....	8-8
<b>Determine minimum sample time</b> .....	8-9
See Also .....	8-9
<b>Real-Time</b> .....	8-10
See Also .....	8-10
<b>Outport Logging</b> .....	8-11
See Also .....	8-11
<b>EtherCAT Synchronous SDO</b> .....	8-12
See Also .....	8-12
<b>Concurrent execution</b> .....	8-13
See Also .....	8-13
<b>Final Validation</b> .....	8-14
See Also .....	8-14



# Configuration Parameters

---

- “Simulink Real-Time Options Pane” on page 1-2
- “Execution mode” on page 1-4
- “Real-time interrupt source” on page 1-5
- “I/O board generating the interrupt” on page 1-6
- “PCI slot (-1: autosearch) or ISA base address” on page 1-7
- “Monitor Task Execution Time” on page 1-8
- “Signal logging data buffer size in doubles” on page 1-9
- “Double buffer parameter changes” on page 1-10
- “Load a parameter set from a file on the designated target file system” on page 1-11
- “File name” on page 1-12
- “Generate INCA/CANape extensions (disables the Simulation Data Inspector and Dashboard blocks)” on page 1-13
- “Enable Stateflow animation” on page 1-14
- “Arm when connecting to target” on page 1-15

## Simulink Real-Time Options Pane

Parameter	Description
“Execution mode” on page 1-4	Specify execution mode of downloaded code.
“Real-time interrupt source” on page 1-5	Select a real-time interrupt source from the I/O board.
“I/O board generating the interrupt” on page 1-6	Specify the board interrupt source.
“PCI slot (-1: autosearch) or ISA base address” on page 1-7	Enter the slot number or base address for the I/O board generating the interrupt.
“Monitor Task Execution Time” on page 1-8	Monitor task execution times and append the results to the target object property <code>tg.TETlog</code> .
“Signal logging data buffer size in doubles” on page 1-9	Enter the maximum number of sample points that the software stores before wrapping.
“Double buffer parameter changes” on page 1-10	Use a double buffer for parameter tuning. This setting enables parameter tuning so that the process of changing parameters in the real-time application uses a double buffer.
“Load a parameter set from a file on the designated target file system” on page 1-11	Automatically load a parameter set from a file on the designated target computer file system.
“File name” on page 1-12	Specify the target computer file name from which to load the parameter set.
“Generate INCA/CANape extensions (disables the Simulation Data Inspector and Dashboard blocks)” on page 1-13	Enable real-time applications to generate data, such as A2L data, for Vector CANape® and ETAS® Inca.
“Enable Stateflow animation” on page 1-14	Enables visualization of Stateflow® chart animation.

Control the code created by Simulink Coder™ code generation software for a Simulink Real-Time application. Set up general information about building real-time applications, including target, execution, data logging, and other options.

### Configuration

The **Simulink Real-Time Options** node in the Configuration Parameters dialog box allows you to specify how the software generates the real-time application. To reveal the **Simulink Real-Time Options** node, do the following:

- 1 In the **Code Generation** pane, in the **System target file** list, select `slrt.tlc`. This setting generates system target code for Simulink Real-Time.

---

**Note** If you open a model that was originally saved with **System target file** set to `xpctarget.tlc`, the software updates the setting to `slrt.tlc`. To retain the updated setting, save the updated model.

---

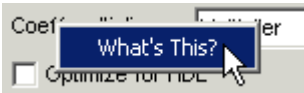
- 2 Select C for the **Language** parameter on the code generation pane.

## Tips

- The default values work for the generation of most real-time applications. If you want to customize the build of your real-time application, set the option parameters to suit your specifications.
- To access configuration parameters from the MATLAB® command line, use:
  - `gcs` — To access the current model.
  - `set_param` — To set the parameter value.
  - `get_param` — To get the current value of the parameter.

## To get help on an option

- 1 Right-click the option text label.
- 2 From the context menu, select **What's This**.



## Execution mode

Specify execution mode of downloaded code.

### Settings

**Default:** Real-Time

Real-Time

Executes downloaded code as a real-time application.

Freerun

Executes downloaded code as fast as possible.

Multirate models cannot be executed in Freerun execution mode. On the **Solver** pane in the Configuration Parameters dialog box, clear the check box for **Treat each discrete rate as a separate task**.

### Command-Line Information

**Parameter:** RL32ModeModifier

**Type:** character vector

**Value:** 'Real-Time' | 'Freerun'

**Default:** 'Real-Time'

## Real-time interrupt source

Select a real-time interrupt source from the I/O board.

### Settings

**Default:** Timer

Timer

Specifies that the board interrupt source is a timer.

Auto (PCI only)

Enables the Simulink Real-Time software to automatically determine the IRQ that the BIOS assigned to the board and use it.

3 to 15

Specifies that the board interrupt source is an IRQ number on the board.

### Tips

- The Auto (PCI only) option is available only for PCI boards. If you have an ISA board (PC/104 or onboard parallel port), set the IRQ manually.
- The Simulink Real-Time software treats PCI parallel port plugin boards like ISA boards. For PCI parallel port plugin boards, set the IRQ manually.
- Multiple boards can share an interrupt number.

### Command-Line Information

**Parameter:** RL32IRQSourceModifier

**Type:** character vector

**Value:** 'Timer' | Auto (PCI only) | '3' | '4' | '5' | '6' | '7' | '8' | '9' | '10' | '11' | '12' | '13' | '14' | '15'

**Default:** 'Timer'

## I/O board generating the interrupt

Specify the board interrupt source.

### Settings

**Default:** None/Other

Bitflow NEON

Specifies that the interrupt source is the BitFlow™ NEON video board.

GE\_Fanuc(VMIC)\_PCI-5565

Specifies that the interrupt source is the GE® Fanuc VMIC PCI-5565 board.

General Standards 24DSI12

Specifies that the interrupt source is the General Standards 24DSI12 board.

Parallel\_Port

Specifies that the interrupt source is the parallel port of the target computer.

Speedgoat\_IO321

Specifies that the interrupt source is the Speedgoat IO321 FPGA board.

Speedgoat\_IO331

Specifies that the interrupt source is the Speedgoat IO331 FPGA board.

Speedgoat\_IO333

Specifies that the interrupt source is the Speedgoat IO333 FPGA board.

None/Other

Specifies that the I/O board has no interrupt source.

### Command-Line Information

**Parameter:** xPCIRQSourceBoard

**Type:** character vector

**Value:** 'Bitflow NEON' |  
'GE\_Fanuc(VMIC)\_PCI-5565' |  
'General Standards 24DSI12' |  
'Parallel\_Port' |  
'Speedgoat\_IO321' |  
'Speedgoat\_IO331' |  
'Speedgoat\_IO333' |  
'None/Other'

**Default:** 'None/Other'

## PCI slot (-1: autosearch) or ISA base address

Enter the slot number or base address for the I/O board generating the interrupt.

### Settings

**Default:** -1

The PCI slot can be either -1 (let the Simulink Real-Time software determine the slot number) or of the form [bus, slot].

The base address is a hexadecimal number of the form 0x300.

### Tip

To determine the bus and PCI slot number of the boards in the target computer, in the Command Window, type:

```
tg = slrt;  
getPCIInfo(tg, 'installed')
```

### Command-Line Information

**Parameter:** xPCIOIRQSlot

**Type:** character vector

**Value:** '-1' | hexadecimal value

**Default:** '-1'

### See Also

"PCI Bus I/O Devices"

## Monitor Task Execution Time

Monitor task execution times and append the results to the target object property `tg.TETlog`.

Task execution time (TET) measures how long it takes the kernel to run for one base-rate time step. For a multirate model, use the profiler to find out what the execution time is for each rate.

### Settings

**Default:** on

On

Monitors task execution times and appends the results to the target object property `tg.TETlog`.

Off

Does not monitor task execution times.

### Command-Line Information

**Parameter:** RL32LogTETModifier

**Type:** character vector

**Value:** 'on' | 'off'

**Default:** 'on'

### See Also

“Signal Logging Basics”



## Signal logging data buffer size in doubles

Enter the maximum number of sample points that the software stores before wrapping.

### Settings

**Default:** 100000

The maximum value for this option cannot exceed the available target computer memory, which the Simulink Real-Time software also uses to hold other items.

### Tips

- Real-time applications use this buffer to store the time, states, outputs, and task execution time (TET) logs as defined in the Simulink model.
- The maximum value for this option derives from available target computer memory, which the Simulink Real-Time software also uses to hold other items. For example, in addition to signal logging data, the software also uses the target computer memory for the Simulink Real-Time kernel, real-time application, and scopes.

For example, assume that your model has six data items (time, two states, two outputs, and task execution time). If you enter a buffer size of 100000, the target object property `tg.MaxLogSamples` is calculated as  $\text{floor}(100000 / 6) = 16666$ . After the buffer saves 16666 sample points, it wraps and further samples overwrite the older ones.

- Suppose that you enter a logging buffer size larger than the available RAM on the target computer. When you download and initialize the real-time application, the target computer displays a message, `ERROR: allocation of logging memory failed`. To avoid this error, either install more RAM or reduce the buffer size for logging, and then restart the target computer. To calculate the maximum buffer size available for your real-time application logs, divide the amount of available RAM on your target computer by `sizeof(double)`, or 8. Enter that value for the **Signal logging data buffer size in doubles** value.

### Command-Line Information

**Parameter:** RL32LogBufSizeModifier

**Type:** character vector

**Value:** '100000' | A valid memory size

**Default:** '100000'

## Double buffer parameter changes

Use a double buffer for parameter tuning. This setting enables parameter tuning so that the process of changing parameters in the real-time application uses a double buffer.

### Settings

**Default:** off



On

Changes parameter tuning to use a double buffer.



Off

Suppresses double buffering of parameter changes in the real-time application.

### Tips

- When a parameter change request is received, the new value is compared to the old one. If the new value is identical to the old one, it is discarded, and if different, it is queued.
- At the start of execution of the next sample of the real-time task, the queued parameters are updated. This operation increases the task execution time (TET) and can cause a CPU overload error.
- Double buffering leads to a more robust parameter tuning interface, but it increases task execution time and the higher probability of overloads. Under typical conditions, keep double buffering off (default).
- If the real-time application contains MATLAB variables as the value of block parameters, the software ignores this double buffering setting. Normal parameter tuning occurs.

### Command-Line Information

**Parameter:** xpcDbfBuff

**Type:** character vector

**Value:** 'on' | 'off'

**Default:** 'off'

## Load a parameter set from a file on the designated target file system

Automatically load a parameter set from a file on the designated target computer file system.

### Settings

**Default:** off

On

Enable the automatic loading of a parameter set from the file specified by **File name** on the designated target computer file system.

Off

Suppress the automatic loading of a parameter set from a file on the designated target computer file system.

### Dependencies

This parameter enables **File name**.

### Command-Line Information

**Parameter:** xPCLoadParamSetFile

**Type:** character vector

**Value:** 'on' | 'off'

**Default:** 'off'

### See Also

“Save and Reload Parameters with MATLAB Language”

## File name

Specify the target computer file name from which to load the parameter set.

### Settings

''

### Tip

If the named file does not exist, the software loads the parameter set built with the model.

### Dependencies

To enable this parameter, set **Load a parameter set from a file on the designated target file system**.

### Command-Line Information

**Parameter:** xPCOnTgtParamSetFileName

**Type:** character vector

**Value:** A valid file name

**Default:** ''

## Generate INCA/CANape extensions (disables the Simulation Data Inspector and Dashboard blocks)

Enable real-time applications to generate data, such as A2L data, for Vector CANape and ETAS Inca.

### Settings

**Default:** off

On

Enables real-time applications to generate data, such as that for A2L, for Vector CANape and ETAS Inca.

Off

Does not enable real-time applications to generate data, such as that for A2L, for Vector CANape and ETAS Inca.

### Command-Line Information

**Parameter:** GenerateASAP2

**Type:** character vector

**Value:** 'on' | 'off'

**Default:** 'off'

### See Also

“Prepare ASAP2 Data Description File”

## Enable Stateflow animation

Enables visualization of Stateflow chart animation.

### Settings

**Default:** off

On

Enables visualization of Stateflow chart animation.

Off

Disables visualization of Stateflow chart animation.

### Command-Line Information

**Parameter:** xPCEnableSFAnimation

**Type:** character vector

**Value:** 'on' | 'off'

**Default:** 'off'

### See Also

“Animate Stateflow Charts with Simulink External Mode”

## Arm when connecting to target

Whether a button or a signal triggers data uploading (as defined by **Source**), the trigger must be armed to allow data uploading to begin. See “Configure Trigger Options” (Simulink Coder).

If the trigger **Source** is signal, monitoring of the trigger signal begins immediately. Data uploading begins when the trigger signal satisfies trigger conditions (as defined in the Trigger signal section).

If you clear **Arm when connecting to target**, manually arm the trigger by clicking the **Arm Trigger** button on the External Mode Control Panel.

### Settings

**Default:** off



On

Connecting to the target arms the trigger.



Off

Data uploading begins immediately after clicking the **Arm Trigger** button on the External Mode Control Panel.

### Command-Line Information

**Parameter:** ExtModeArmWhenConnect

**Type:** character vector

**Value:** 'on' | 'off'

**Default:** 'off'

### See Also

- “Configure Trigger Options” (Simulink Coder)
- “Role of Trigger in Signal Data Uploading” (Simulink Coder)





# TLC Options Parameters

---

## TLC Command-Line Options

TLC command-line options are model options set before code generation to configure the real-time application and the real-time kernel.

To set these options from the **Code Generation** pane in the Configuration Parameters dialog box, select **Advanced Parameters**. Type the option in the **TLC command line options** text box in this form:

```
-aoption_name1=option_value1 -aoption_nameN=option_valueN
```

Prefix each option name with -a. Do not leave spaces around the equal sign. Do not place a comma between consecutive value assignments.

To set these options from the Command Window, use the syntax:

```
set_param(model_name, ...
    'TLCOptions', ...
    '-aoption_name1=option_value1 -aoption_nameN=option_valueN')
```

For example:

```
set_param('xpcosc', ...
    'TLCOptions', ...
    '-axPCMaxOverloads=20 -axPCModelStackSizeKB=4096')
```

To read these options from the Command Window, use the syntax:

```
get_param(model_name, 'TLCOptions')
```

For example:

```
get_param('xpcosc', 'TLCOptions')
ans =
-axPCMaxOverloads=20 -axPCModelStackSizeKB=4096
```

To remove these options, use the syntax:

```
set_param(model_name, 'TLCOptions', '')
```

### In this section...

- “xPCMaxOverloads” on page 2-2
- “xPCMaxOverloadLen” on page 2-3
- “xPCStartupFlag” on page 2-3
- “xPCModelStackSizeKB” on page 2-4
- “xpcCPUClockPoll” on page 2-4
- “SLRTFTZOFF” on page 2-4

## xPCMaxOverloads

When xPCMaxOverloads is set to a value, the Simulink Real-Time software stops execution with a CPU overload at the next overload within the same application execution. For example, if

`xPCMaxOverloads` is set to 3, the software stops with a CPU overload at the fourth overload in the same application execution.

The default value of 0 means that overloads are registered on the first overload.

Allowing the target computer CPU to overload can cause incorrect results, especially for multirate models. Use these TLC command-line options on page 2-2 only for diagnosis. When your diagnosis is complete, turn off these options.

### See Also

“CPU Overload Options”

## xPCMaxOverloadLen

When `xPCMaxOverloadLen` is set to a value, the software stops execution with a CPU overload at the next overload within the same execution step. For example, if `xPCMaxOverloadLen` is set to 2, the software stops execution with a CPU overload at the third overload within the same execution step.

The default value of 0 means that overloads are registered on the first overload within the same execution step.

Specify a value that is less than or equal to the value for `xPCMaxOverloads`. If `xPCMaxOverload` is set to a value, for example 4, and `xPCMaxOverloadLen` is not defined, the real-time application stops if one of following occurs:

- The cumulative overloads since execution start is greater than 4.
- One execution step has two overloads.

Allowing the target computer CPU to overload can cause incorrect results, especially for multirate models. Use these TLC command-line options on page 2-2 only for diagnosis. When your diagnosis is complete, turn off these options.

### See Also

“CPU Overload Options”

## xPCStartupFlag

`xPCStartupFlag` temporarily disables CPU overload checking during the first few model execution steps. After the model finishes the first `xPCStartupFlag` steps, the software reenables CPU overload checking, which takes effect for the next execution of the model.

The default value of 1 means that overloads are ignored on the first step. If `xPCMaxOverloads` and `xPCMaxOverloadLen` are not set, their default setting determines the software response to overloads.

### See Also

“CPU Overload Options”

## **xPCModelStackSizeKB**

Sets the number of kilobytes of stack memory that are allocated to real-time threads on the target computer. The default value is 2048.

Target computer memory for the real-time application executable, the kernel, and other uses is limited to a maximum of 4 GB.

### **See Also**

“Troubleshoot Target Computer Stack Size”

## **xpcCPUClockPoll**

Switches the kernel from interrupt mode to polling mode. When **Execution mode** is Real-Time, a nonzero value causes the real-time application to perform a busy wait at the target computer CPU clock rate. If the value is 0 (the default) or if the option is not defined, the kernel executes in interrupt mode.

### **See Also**

“Polling Mode”

## **SLRTFTZOFF**

Configures floating-point processing:

- 0 (default) — Denormal float processing is not performed. The representation of small numbers is slightly different from the representation when the value is 1. Floating-point operations are faster. The corresponding Microsoft® Visual C++® compiler options are `/fp:fast /arch:SSE2`.
- 1 — Floating-point operations meet the IEEE® Standard for Floating Point Arithmetic (IEEE 754-2008). The corresponding Microsoft Visual C++ compiler options are `/fp:precise` with the default value of `/arch`.

### **See Also**

- <https://www.ieee.org>
- <https://www.microsoft.com>

# Simulink Real-Time Explorer

---

# Simulink Real-Time Explorer

Interact with target computer and real-time application running on target computer

## Description

Simulink Real-Time provides a single point of contact for configuring the development and target computers and interacting with a real-time application. You can monitor and trace signals, tune parameters, create and run instrument panels, and export and run an Explorer configuration as a standalone executable.

---

**Note** Do not use Simulink external mode while Simulink Real-Time Explorer is running. Use only one interface or the other.

---

Use Simulink Real-Time Explorer for the following tasks. For examples, click the links in the **More Information** column.

### Target Computer Configuration

Capability	More Information
Configure target computer communication parameters.	<ul style="list-style-type: none"><li>“PCI Bus Ethernet Setup”</li></ul>
Configure target computer configuration settings.	“Target Computer Settings”
Configure target computer startup and create boot images for target computers.	“Target Computer Boot Methods”

**Real-Time Application Access and Control**

Capability	More Information
<ul style="list-style-type: none"> <li>• Connect target computers to a development computer, and then disconnect them.</li> <li>• Load a prebuilt real-time application into a target computer, and then unload it.</li> <li>• Start and then stop running a real-time application that you downloaded to the target computer.</li> <li>• Display execution time, task execution time, and other properties of the real-time application.</li> <li>• Change stop time and sample times without regenerating code.</li> </ul>	<ul style="list-style-type: none"> <li>• “Configure and Control a Real-Time Application”</li> <li>• “Process a Real-Time Application by Using Simulink Real-Time Explorer”</li> </ul>

**Signal Access**

Capability	More Information
Filter and group hierarchical signals	“Display and Filter Hierarchical Signals and Parameters”
Monitor signals.	<ul style="list-style-type: none"> <li>• “Monitor Signals with Simulink Real-Time Explorer”</li> <li>• “Instrument a Stateflow Subsystem”</li> </ul>
<ul style="list-style-type: none"> <li>• Add host, target, or file scopes for the downloaded real-time application, and then remove them.</li> <li>• Add signals to scopes, and then remove them.</li> <li>• Configure scope properties.</li> <li>• Configure a host scope display and use it for viewing signal values.</li> <li>• Start and stop scopes.</li> <li>• Browse file scope output files on the target computer file system.</li> </ul>	<ul style="list-style-type: none"> <li>• “Create Target Scopes with Simulink Real-Time Explorer”</li> <li>• “Create Host Scopes with Simulink Real-Time Explorer”</li> <li>• “Create File Scopes with Simulink Real-Time Explorer”</li> </ul>
Create, save, and load signal groups.	“Create Signal Groups with Simulink Real-Time Explorer”

### Parameter Tuning

Capability	More Information
Filter and group hierarchical parameters	“Display and Filter Hierarchical Signals and Parameters”
Display and tune parameter values while the real-time application is running.	<ul style="list-style-type: none"> <li>• “Tune Parameters with Simulink Real-Time Explorer”</li> <li>• “Tune Parameter Structures with Simulink Real-Time Explorer”</li> </ul>
Create, save, and load parameter groups.	“Create Parameter Groups with Simulink Real-Time Explorer”

### Instruments and Instrument Panels

Capability	More Information
<p>Create, configure, save, and load graphical instrument panels for acquiring signals and tuning parameters.</p> <p>Attach parameters to instruments in instrument panels.</p> <p>Attach signals to instruments in instrument panels.</p> <p>Start and stop instrument panels and use them to interact with real-time applications.</p>	<ul style="list-style-type: none"> <li>• “Instrumentation for Real-Time Applications” on page 4-2</li> <li>• “Add Instrument Panel to Tank Model” on page 4-25</li> </ul>
Configure Scope instruments.	<ul style="list-style-type: none"> <li>• “View Signal Waveforms with Scope Instrument” on page 4-15</li> <li>• “View Signal Waveforms with Signal Triggered Scope Instrument” on page 4-21</li> <li>• “Triggering Scope Instruments”</li> </ul>

### Explorer Configuration

Capability	More Information
Save and load environment properties, instrument panels, and Explorer configuration layouts.	<ul style="list-style-type: none"> <li>• “Save Environment Properties” on page 4-35</li> <li>• “Save and Load Instrument Panels” on page 4-36</li> <li>• “Save and Restore Layouts” on page 4-37</li> </ul>
Export Explorer configuration as a standalone executable.	“Export Explorer Configuration” on page 4-41
Run standalone executable generated from an exported Explorer configuration.	“Unpack and Run Standalone Configuration” on page 4-42

### Windows

Simulink Real-Time Explorer provides tool windows and workspace windows in its main window. Tool windows include **Targets**, **Applications**, **Scopes**, **Palette**, **Panels**, **Output**, and so on. Workspace



windows include windows for interfacing with the real-time application, setting environment properties, setting application and scope properties, and so on. You can resize tool windows and drag them by their title bar. You can drag workspace windows by their tab.

### **Layouts**

To create optimal window layouts for your work, customize the position, size, and behavior of the various windows. When you customize the layout for the tool windows, Explorer retains the positions, size, and docking locations of each tool window. For example, assume that you change the docking location of the **Targets** tool window, and then close Explorer and MATLAB. The next time that you start Explorer, the target tool window is docked in that same location. Workspace window customizations do not persist across different MATLAB sessions.

### **Tab Groups**

To manage limited workspace while you are working with two or more Explorer windows, use Tab Groups. You can organize multiple workspace windows and tool windows into vertical and horizontal Tab Groups. You can drag windows from one Tab Group to another.

### **Arrange Tool Windows**

You can dock tool windows anywhere within the main window, float them as separate windows independent of the main window, and hide them.

You arrange tool windows by dragging and dropping them or by right-clicking the window title bar and clicking a menu command:

- Pin tool windows to the left or the right of the tab well.
- Dock tool windows to the edge of the main window.
- Float tool windows over or outside the main window.
- Hide tool windows along the edge of the main window.
- Display tool windows on different monitors.
- Save window placement to a custom layout and restore it.

### **Arrange Workspace Windows**

You can dock workspace windows anywhere to the left of the main window, float them as separate windows independent of the main window, and close them.

You arrange workspace windows by dragging and dropping them or by right-clicking the window tab and clicking a menu command:

- Tile workspace windows in the tab well.
- Dock workspace windows to the left edge of the main window.
- Float workspace windows over or outside the main window.
- Close workspace windows.
- Display workspace windows on different monitors.
- Save window placement to a custom layout and restore it.

### **Dock Windows with a Guide Diamond**


When you click and drag a tool window title bar or a workspace window tab, a guide diamond appears. As you drag the window, place your cursor over one of the arrows in the diamond. A shaded area appears that shows where the window will appear after you release the mouse button.

To move a dockable window without snapping it into place, press the **Ctrl** key while you drag the window.

To return a tool window or workspace window to its most recent docked location, double-click the window title bar or window tab. You can fasten tool windows to one side of a frame in the main window. When you drag a tool window to another location, a guide diamond appears to help you redock the window.

### **Close and Auto Hide Tool Windows**

You can close a tool window by clicking the **X** in the upper right of the title bar. To reopen the window, in the **View** menu, select the tool window that you want to reopen. If the tool window is closed, the tool window opens and becomes the active selected tool window. If the tool window is already open, it becomes the currently active tool window.

Tool windows support Auto Hide () , which causes a window to slide out of the way when you use a different window. When you autohide a window, its name appears on a tab at the edge of the frame of the main window. To use the window again, place your cursor over the tab or select the tool window from the **View** menu. The window slides back into view.

### **Create and Save Layouts**

You can create and save a custom layout that includes workspaces that you created for a specific target computer and real-time application. You can then switch between layouts with a single command. If you open a layout that contains workspaces, the target computer must have the same application and resources as when you originally created the layout.

### **Restore Layouts**

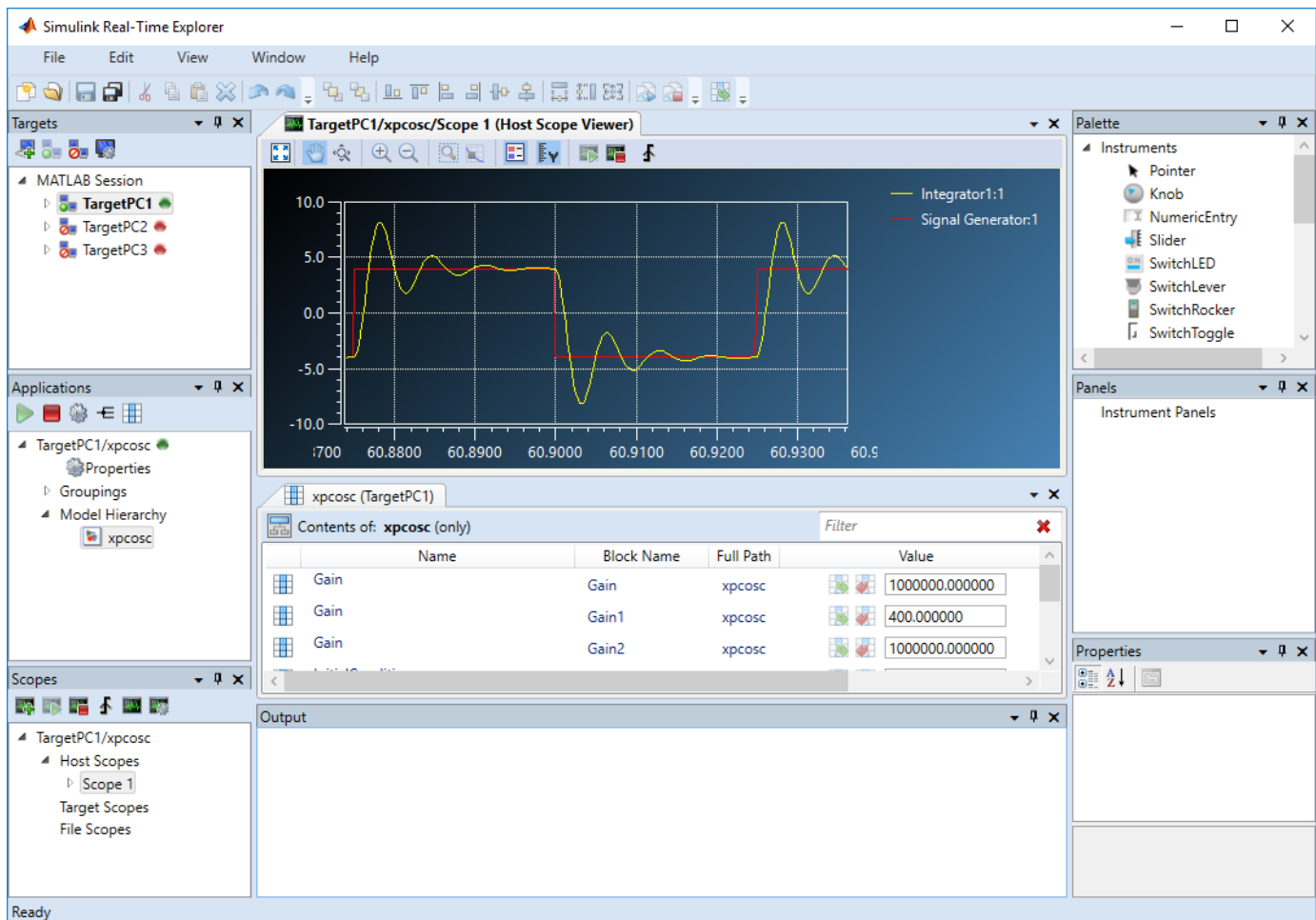
You can return Explorer to the original window layout for your settings by using the **Restore layout** command. When you run this command:

- Tool windows move to their original positions.
- Workspace windows that were previously opened close.
- Workspace windows included in the layout that are open in the original layout open.

You can restore workspace windows that depend on a specific target computer and real-time application. The target computer must be connected and contain the same application and resources as when you created the layout. Windows that fail to open produce an error message in the output window

### **Representative Layout**

The figure shows a representative layout showing several tool and workspace windows.



In this figure, the tool windows are:

- **Targets** (top left) — Lists the targets in your Simulink Real-Time hierarchy. Under each target are nodes representing the properties and file system of the target.
- **Applications** (middle left) — Lists the real-time applications running on the targets. Under each application are nodes representing the properties, signal and parameter groupings, and model hierarchy of the application. If a target is not running a real-time application, it does not appear in the list.
- **Scopes** (bottom left) — Lists the scopes defined on the active real-time applications, whether predefined or dynamically created.
- **Output** (bottom center) — Displays status and error messages.
- **Properties** (bottom right) — Lists the properties of a selected instrument or layout element.
- **Panels** (middle right) — Lists the instrument panels that have been loaded.
- **Palette** (top right) — Lists the instrument and layout elements available for constructing instrument panels.

The workspace windows are:

- **TargetPC1/xpcosc/Scope 1 (Host Scope Viewer)** (top center) — The host scope viewer window shows signal and time information from a host scope that is running on the target computer.

- **xpcosc (TargetPC1)** (mid center) — The parameter window shows tunable parameters from the real-time application that is running on the target computer.

### Exported Standalone Interface

You can export Simulink Real-Time Explorer as a standalone executable to a computer compatible with Windows®. You do not have to run MATLAB to run the standalone interface.

When you run Explorer as a standalone interface, it supports a subset of the capabilities that it supports under MATLAB.

- You cannot change the communication parameters that the interface uses to communicate with the target computers. Before you export the Simulink Real-Time Explorer configuration, configure and test the communication parameters for each target computer.
- For each instrument, the exporting software records the real-time application and target computer environment with which it is associated.
- If you rename a target computer, to maintain the connection to the real-time application, update the **TargetName** parameter for each associated instrument.
- You cannot load or unload a real-time application from the standalone executable. Before you start the executable, start the real-time application on the target computer.
- You can access only instrument panels and windows that you loaded before you exported the configuration.
- You cannot access the real-time application model hierarchy from the standalone executable.
- You can access only signals in signal groups that you loaded before you exported the configuration.
- You cannot move a signal from one signal group to another group, or create or load a new signal group.
- You can access only parameters in parameter groups that you loaded before you exported the configuration.
- You cannot move a parameter from one parameter group to another group, or create or load a new parameter group.
- You cannot save session layouts. If you close a window, you can restore the original layout using **File > Restore Original View**.

## Open the Simulink Real-Time Explorer

- In the Simulink Editor, on the **Real-Time** tab, from the **Prepare** section, click **SLRT Explorer**.
- On a computer compatible with Windows, click the standalone executable.

### Programmatic Use

In the Command Window, type `slrtexplr`.

### See Also

`SimulinkRealTime.prototype.Explorer` |  
`SimulinkRealTime.utils.TargetComputerManager` | `slrtexplr`

## **Topics**

"Signal Monitoring Basics"

"Signal Tracing Basics"

"Signal Logging Basics"

"File System Basics"

"Tunable Block Parameters and Tunable Global Parameters"

"Simulink Real-Time Scope Usage"

"Target Scope Usage"

"Host Scope Usage"

"File Scope Usage"

"Instrumentation for Real-Time Applications" on page 4-2

"Explorer Configuration Exported to Run Outside MATLAB" on page 4-13

Simulink Real-Time Target Computer Explorer tech preview

Simulink Real-Time Target Computer Manager

## **Introduced in R2014a**

## Simulink Real-Time Target Computer Explorer tech preview

Interact with target computer and real-time application running on target computer

### Description

Simulink Real-Time Target Computer Explorer provides a single point of contact for viewing connection status and interacting with a real-time application. You can monitor and trace signals, tune parameters, and stream data to the Simulation Data Inspector.

---

**Note** Do not use Simulink external mode while Simulink Real-Time Target Explorer is running. Use only one interface or the other.

---

Use Simulink Real-Time Target Computer Explorer for these tasks:

- Connect the development computer and target computer.
- Load, start, and stop a real-time application on target computer.
- View real-time application parameters and signal hierarchy.
- Select real-time application signals for streaming to the Simulation Data Inspector.
- Set real-time application stop time.

---

**Tip** In this tech preview, the `improveExplorerPerformance` function improves performance of the tech preview Explorer. This function generates a tree that represents the model hierarchy of real-time application *APP* for the Explorer. If you do not execute the `improveExplorerPerformance` function after you build *APP*, the tech preview Explorer builds the tree each time the Explorer opens and loads *APP* on the target computer. To improve prototype explorer performance, execute `improveExplorerPerformance` each time you build *APP*.

```
SimulinkRealTime.prototype.improveExplorerPerformance('APP')
```

---

## Open the Simulink Real-Time Target Computer Explorer tech preview

From the MATLAB Command Window, type:

```
SimulinkRealTime.prototype.Explorer
```

### Examples

## Select Signals and Stream Data to Simulation Data Inspector

This example shows how to connect to the target computer, load the real-time application, select signals for a signal list, start the real-time application, and view the streaming data in the Simulation Data Inspector.

Open the Simulink Real-Time Target Computer Explorer, type:

```
SimulinkRealTime.prototype.Explorer
```

In the Simulink Real-Time Target Computer Explorer, to connect to the selected target computer, click **Connect**.

To select and load a real-time application, click **Load Application** and select the `ml.datx` file.

To select signals for streaming, click the application name, select signals from the **Signals** tab, and click the **Add selected signals** button.

To run the application and generate data for streaming, click the **Run** button.

To stream the signal data, select the signals in the **Group signals to stream for SDI** list and click the **Stream Signal Group to SDI** button.

To view the streaming signals, click the **Open in SDI** button.

After viewing the data, to stop the real-time application, click the **Stop** button.

## Programmatic Use

`SimulinkRealTime.prototype.Explorer` opens the Target Computer Explorer.

## See Also

`SimulinkRealTime.prototype.Explorer` |  
`SimulinkRealTime.utils.TargetComputerManager` | `slrtexplr`

## Topics

**Simulink Real-Time Explorer**

**Simulink Real-Time Target Computer Manager**

**Introduced in R2019a**

# Simulink Real-Time Target Computer Manager

Interact with target computer

## Description

Simulink Real-Time Target Computer Manager provides a single point of contact for configuring the target computers. You configure the connection to the target computer, select target computer boot modes, and view connection status.

To configure target computers, use the Simulink Real-Time Target Computer Manager.

- Add target computer. To add a target computer definition, click the **Add** button and configure setting in the Target Computer Manager fields.
- Remove target computer. To remove the selected target computer definition, click the **Remove** button.
- Connect target computer. To connect the selected target computer, click the **Connect** button. When you select a connected target computer, the button label changes to **Disconnect**.
- Configure advanced settings. To configure the port, subnet mask, and boot mode, expand the **Advanced settings** button.
- Create boot disk. To create a boot disk, click the **Create boot disk** button.

## Open the Simulink Real-Time Target Computer Manager

From the Simulink Editor toolstrip, expand the target computer list and select **Target Computer Manager**. Or, you can open the manager from the MATLAB command window as shown in “Open the Target Computer Manager” on page 3-12

## Examples

### Open the Target Computer Manager

- In the Simulink Editor, on the **Real-Time** tab, from target computers list in the **Connect to Target Computer** section, click **Target Computer Manager**.
- In the MATLAB Command Window, type:

```
SimulinkRealTime.utils.TargetComputerManager.open
```

## Programmatic Use

`SimulinkRealTime.utils.TargetComputerManager.open` opens the Target Computer Manager.



## See Also

`SimulinkRealTime.prototype.Explorer` |  
`SimulinkRealTime.utils.TargetComputerManager` | `slrtexplr`

## Topics

**Simulink Real-Time Explorer**

**Simulink Real-Time Target Computer Explorer tech preview**

**Introduced in R2019a**



# Simulink Real-Time Explorer Instrumentation

---

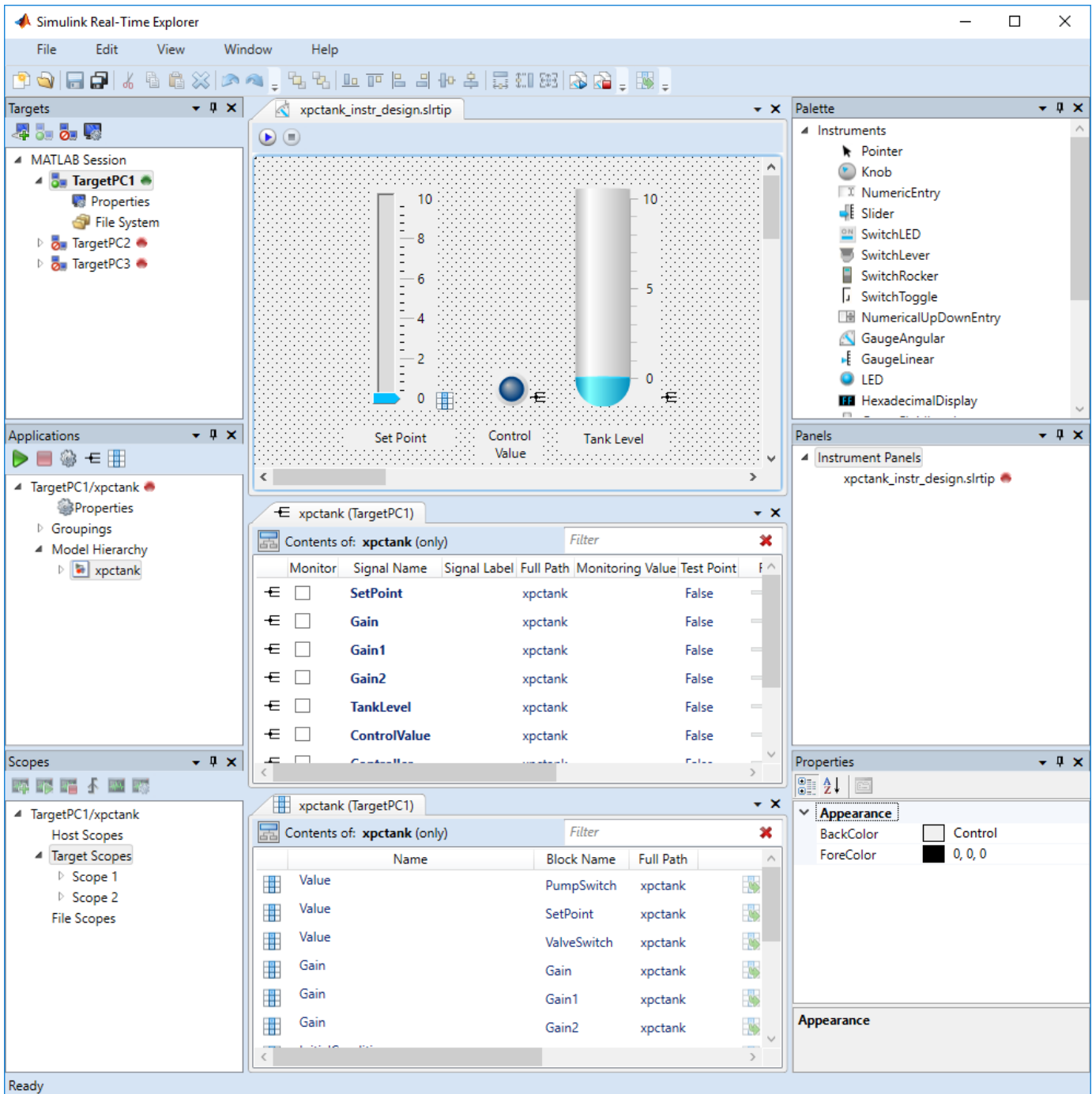
- “Instrumentation for Real-Time Applications” on page 4-2
- “Explorer Configuration Exported to Run Outside MATLAB” on page 4-13
- “View Signal Waveforms with Scope Instrument” on page 4-15
- “View Signal Waveforms with Signal Triggered Scope Instrument” on page 4-21
- “Add Instrument Panel to Tank Model” on page 4-25
- “Save Environment Properties” on page 4-35
- “Save and Load Instrument Panels” on page 4-36
- “Save and Restore Layouts” on page 4-37
- “Prepare Explorer Environment for Export” on page 4-38
- “Prepare Instrument Panel Configuration for Export” on page 4-39
- “Export Explorer Configuration” on page 4-41
- “Unpack and Run Standalone Configuration” on page 4-42

## Instrumentation for Real-Time Applications









In this section...
“Instrument Selection and Binding” on page 4-4
“Display Instruments” on page 4-5
“Tuning Instruments” on page 4-6
“Layout Elements” on page 4-7
“Graphical Properties” on page 4-9

To visualize the behavior of a real-time application running on a target computer, Simulink Real-Time Explorer provides instrument panels. An instrument panel is an Explorer workspace into which you can insert one or more instruments. In Simulink Real-Time Explorer, you can create and load instrument panels from the toolbar, the **File** menu, and the **Panels** tool window. You can display simultaneously as many panels as can fit on the screen.

After creating one or more instrument panels, you can drag instruments to the panels and drag parameters and signals to the instruments. You can add layout elements to clarify the relationships among the instruments. You can then start your real-time application from Simulink Real-Time Explorer and start the instrument panels to control the parameters and view the signal outputs.



At design time, you can manipulate the instruments by using the toolbar buttons.

Action	Icon	Notes
<b>New, Open, Save, SaveAll</b>		<p>The available operations vary with the active window.</p> <ul style="list-style-type: none"> <li>You can save an instrument panel from within only the window for that panel.</li> <li>You can create or open a signal or parameter group from within only the <b>Applications</b> window.</li> <li>You can save a group from within only the window for that group.</li> <li>You can create or open an instrument panel regardless of the active window.</li> <li>You can use the <b>SaveAll</b> button regardless of the active window.</li> </ul>
<b>Cut, Copy, Paste, Delete</b>		Applies to instruments only.
<b>Layer</b>		Applies to instruments only.
<b>Align on edges</b>		Applies to instruments only.
<b>Align on centers</b>		Applies to instruments only.
<b>Equalize sizes</b>		Applies to instruments only.
<b>Undo, Redo</b>		Available after unsaved change only.
<b>Run, Stop, Run all, Stop all</b>		Run and stop all active instrument panels.


## Instrument Selection and Binding

To instrument a real-time application, populate an instrument panel with instruments compatible with the parameters and signals that they represent.

To make an instrument interact with the real-time application, bind a signal or parameter to the instrument. You bind the signal or parameter by dragging it from the signal or parameter viewer and dropping it on the instrument. You can also drag a signal or parameter from a signal or parameter group to an instrument.

You can bind a signal to a display instrument but not to a tuning instrument. You can bind a parameter to a tuning instrument and to a display instrument.

When you bind a signal to a display instrument, the **Signal** button  appears next to the instrument.

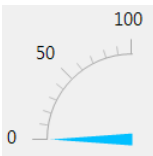
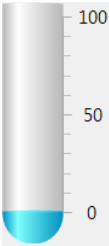
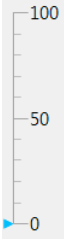


When you bind a parameter to a tuning or display instrument, the **Parameter** button  appears next to the instrument.


When the instrument panel is running, you can tune parameters by using tuning instruments. The software transmits the parameter changes to the real-time application. You can view the changed behavior by using display instruments.

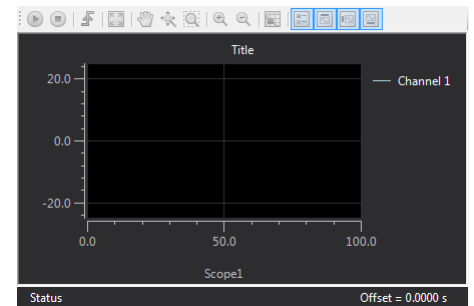
When the instrument panel is not running, you can add, remove, and lay out instruments and connect signals and parameters to them.

## Display Instruments

The table shows some of the tasks that you can do with display instruments.

Action	Requiring	Use
<ul style="list-style-type: none"> <li>Show the pressure in a container.</li> <li>Show the speed of a vehicle.</li> <li>Show current or voltage in a circuit.</li> </ul>	<ul style="list-style-type: none"> <li>Real-valued data</li> <li>Show approximate value by angular displacement</li> <li>Show enumerated value as scale units (requires enum type data for display).</li> </ul>	 <p>GaugeAngular Properties</p>
<ul style="list-style-type: none"> <li>Show the level of fluid in a container.</li> <li>Show the pressure in a pipe.</li> </ul>	<ul style="list-style-type: none"> <li>Real-valued data</li> <li>Show approximate value by vertical displacement</li> <li>Show enumerated value as scale units (requires enum type data for display).</li> </ul>	 <p>GaugeFluidLevel Properties</p>
<ul style="list-style-type: none"> <li>Show the pressure in a container.</li> <li>Show audio output power.</li> <li>Show current or voltage in a circuit.</li> </ul>	<ul style="list-style-type: none"> <li>Real-valued data</li> <li>Show approximate value by linear displacement</li> <li>Show enumerated value as scale units (requires enum type data for display).</li> </ul>	 <p>GaugeLinear Properties</p>
<ul style="list-style-type: none"> <li>Show traffic on a digital bus.</li> <li>Show the state of a state machine.</li> </ul>	<ul style="list-style-type: none"> <li>Hexadecimal-valued data</li> <li>Show values in hexadecimal format</li> </ul>	 <p>HexadecimalDisplay Properties</p>
<ul style="list-style-type: none"> <li>Show on-off state of a switch.</li> <li>Show on-off state of a bidirectional pin.</li> </ul>	<ul style="list-style-type: none"> <li>Boolean data</li> <li>Show value by light turning on and off</li> </ul>	 <p>LED Properties</p>

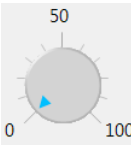
Action	Requiring	Use
<ul style="list-style-type: none"> <li>Show a temperature measurement to given precision.</li> <li>Show a voltage measurement to given precision.</li> <li>Show a date and time.</li> <li>Show a time-varying temperature measurement.</li> <li>Show a time-varying voltage measurement.</li> </ul>	<ul style="list-style-type: none"> <li>Real-valued data</li> <li>Show real values in decimal or other format</li> </ul>	 NumericDisplay Properties








Scope Properties

## Tuning Instruments

The table shows some of the tasks you can do with tuning instruments.

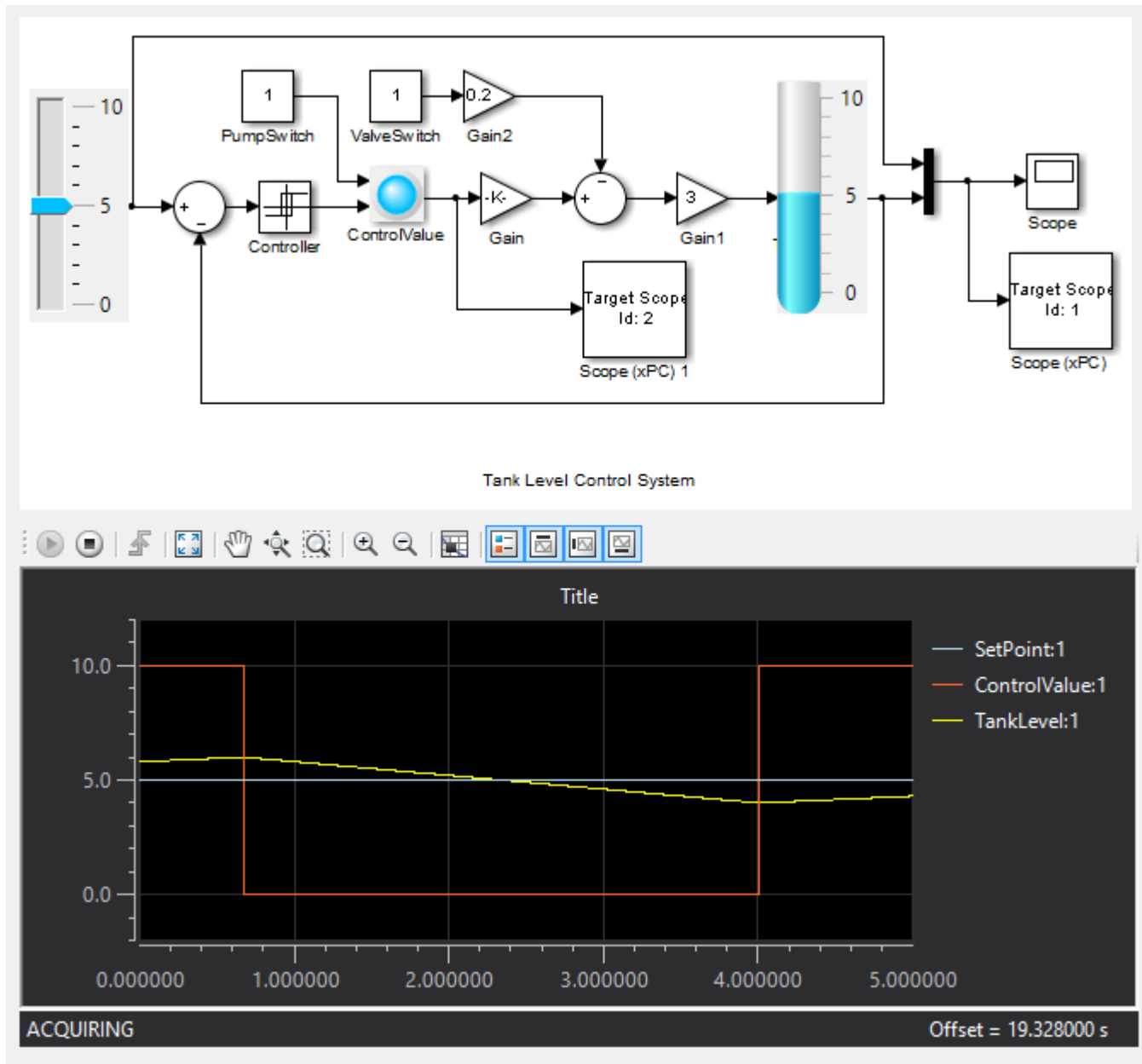
Action	Requiring	Use
<ul style="list-style-type: none"> <li>Control gain of a radio receiver.</li> <li>Control amplification of a radio transmitter.</li> </ul>	<ul style="list-style-type: none"> <li>Real-valued data</li> <li>Set approximate value by angular displacement</li> <li>Show enumerated value as scale units (requires enum type data for display).</li> </ul>	 Knob Properties
<ul style="list-style-type: none"> <li>Enter initial set point control value for thermostat.</li> <li>Enter seed value for random number generator.</li> </ul>	<ul style="list-style-type: none"> <li>Real-valued data</li> <li>Set exact numeric value</li> </ul>	<input data-bbox="1063 1323 1177 1354" type="text" value="0"/> NumericEntry Properties
<ul style="list-style-type: none"> <li>Control car radio audio volume using step-increment.</li> <li>Smoke test controller range in small number of clicks.</li> </ul>	<ul style="list-style-type: none"> <li>Real-valued data</li> <li>Set initial value and step increment</li> </ul>	<input data-bbox="1063 1470 1201 1501" type="text" value="0"/> NumericUpDownEntry Properties



Action	Requiring	Use
<ul style="list-style-type: none"> <li>Control frequency of a radio receiver.</li> <li>Control pressure valve setting.</li> </ul>	<ul style="list-style-type: none"> <li>Real-valued data</li> <li>Set approximate values by linear displacement</li> <li>Show enumerated value as scale units (requires enum type data for display).</li> </ul>	 <p>Slider Properties</p>
<ul style="list-style-type: none"> <li>Turn on a power supply.</li> <li>Close a gate valve.</li> </ul>	<ul style="list-style-type: none"> <li>Boolean data</li> <li>Turn control on or off</li> </ul>	 <p>SwitchLED Properties</p>  <p>SwitchLever Properties</p>  <p>SwitchRocker Properties</p>  <p>SwitchToggle Properties</p>

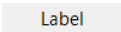


## Layout Elements

To make the relationships among the instruments clearer, you can add layout elements, such as a picture box. In the figure, a picture box shows to which signals the instruments are bound.



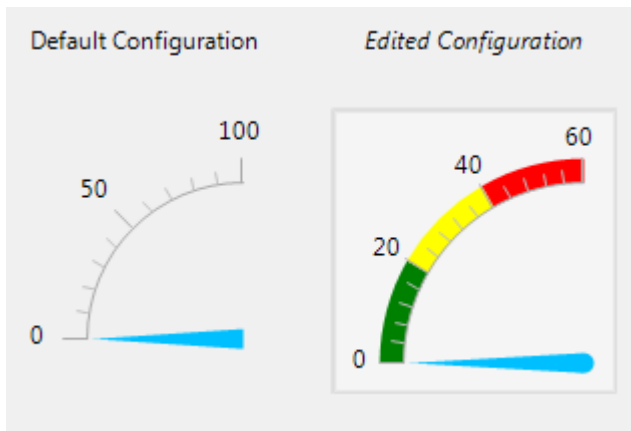
The table shows some ways that you can modify your layout with layout elements.


Action	Requiring	Use
<ul style="list-style-type: none"> <li>Group electrical instruments together and label the group.</li> <li>Group pressure instruments together and label the group.</li> </ul>	<ul style="list-style-type: none"> <li>Design-time box resizing</li> <li>Run-time static display</li> </ul>	<div style="border: 1px solid gray; padding: 5px; width: fit-content;">                     GroupBox                 </div> <p>GroupBox Properties</p>


Action	Requiring	Use
<ul style="list-style-type: none"> <li>Label an electrical instrument.</li> <li>Label a pressure instrument.</li> </ul>	<ul style="list-style-type: none"> <li>Design-time box resizing, left-right and up-down text alignment</li> <li>Run-time static display</li> </ul>	 Label Properties
<ul style="list-style-type: none"> <li>Group the electrical group with the pressure group and scroll between the groups.</li> <li>Group a picture box with a group of instruments to show what the instruments are measuring.</li> </ul>	<ul style="list-style-type: none"> <li>Design-time box resizing</li> <li>Run-time box scrolling</li> </ul>	 Panel Properties
<ul style="list-style-type: none"> <li>Insert an image of a circuit diagram on a panel behind electrical instruments.</li> <li>Insert an image of a circulation diagram on a panel behind pressure instruments.</li> </ul>	<ul style="list-style-type: none"> <li>Design-time image stretch, zoom, center, and autosize</li> <li>Run-time static display</li> </ul>	 PictureBox Properties

## Graphical Properties

You can configure the graphical properties of the instruments. The figure shows an instrument in its default configuration and with representative property changes.



To access a dialog box containing key instrument properties, select the instrument and in the top, right corner of the instrument, click the **Tasks** button .

To access the complete **Properties** list for an instrument, in the instrument panel, select the instrument. To access a dialog box for a particular property group, click the group. To the right of the group, click the continuation button .

## Instrument Properties

The table shows some of the ways that you can modify the appearance of your instruments through the **Instrument** properties.

Action	Use
Set instrument value range	<p><b>ScaleRange</b></p> <ul style="list-style-type: none"> <li>• <b>Min, Span</b> — Value bounds. <b>Max</b> is read-only.</li> <li>• <b>Reverse</b> — If <b>True</b>, reverse direction of display</li> <li>• <b>ScaleType</b> — One of <b>Linear</b>, <b>Log10</b>, <b>SplitLinearLog10</b>.</li> <li>• <b>SplitStart, SplitPercent</b> — Value and percentage of scale split for <b>ScaleType SplitLinearLog10</b> only.</li> <li>• <b>AngleMin, AngleSpan</b> — Angular value bounds. <b>AngleMax</b> is read-only. For <b>GaugeAngular</b> and <b>Knob</b> only.</li> </ul>
Set value display properties	<p><b>ScaleDisplay &gt; GeneratorStyle &gt; Auto</b></p> <p>In <b>ScaleDisplay &gt; GeneratorAuto</b>:</p> <ul style="list-style-type: none"> <li>• <b>MidIncluded</b> — If <b>True</b>, insert a tick halfway between major ticks.</li> </ul> <p>If <b>MinorCount</b> is even, space the minor ticks equally around the center tick. If <b>MinorCount</b> is odd, replace the center tick with the middle tick.</p> <ul style="list-style-type: none"> <li>• <b>FixedMinMaxMajor</b> — If <b>True</b>, the top and bottom ticks are constrained to be major ticks with min/max values defined by <b>Min</b> and <b>Span</b>.</li> <li>• <b>MinorCount</b> — Number of minor ticks between major ticks.</li> <li>• <b>MinTextSpacing</b> — Minimum space between scale ticks.</li> <li>• <b>DesiredIncrement</b> — Display of major tick values. <math>\text{number of labels} = \text{span} / (\text{desired increment} + 1)</math>. Does nothing if the required labels do not fit in the space available in the graphic.</li> </ul> <p>You can also set properties for <b>ScaleDisplay &gt; GeneratorStyle &gt; Fixed</b> and <b>ScaleDisplay &gt; GeneratorStyle &gt; Custom</b>.</p> <p><b>ScaleDisplay &gt; TextFormatting</b></p> <ul style="list-style-type: none"> <li>• <b>Style</b> — One of <b>Number</b>, <b>Thousands</b>, <b>Prefix</b>, <b>Exponent</b>, <b>Price32nds</b>, <b>DateTime</b>, <b>DateTimeUTC</b>.</li> <li>• <b>PrecisionStyle</b> — One of <b>FixedDecimalPoints</b>, <b>SignificantDigits</b>, <b>None</b>.</li> <li>• <b>Precision</b> — Number of digits to the right of the decimal point.</li> <li>• <b>UnitsText</b> — Text to display next to tick labels.</li> </ul>

Action	Use
Set pointer properties	<p><b>Pointer</b></p> <ul style="list-style-type: none"> <li>• <b>Style</b> — The pointer style depends upon the instrument. <ul style="list-style-type: none"> <li>• GaugeAngular — One of Arrow, ArrowLine, Line, Triangle.</li> <li>• GaugeLinear — One of Pointer, Triangle, TLine, ColorBar, Tube.</li> <li>• Knob — One of Dot, DotRaised, DotSunken, LineCenter, LineCustom, Triangle.</li> <li>• Slider — One of Bar, BarIndicator, BarIndicatorLight, Pointer</li> </ul> </li> <li>• <b>Size</b> — The width of the pointer, in pixels.</li> <li>• <b>Margin</b> — The margin around the pointer, in pixels.</li> </ul>
Set border properties	<p><b>Border</b></p> <ul style="list-style-type: none"> <li>• <b>Style</b> — One of Bump, Etched, Flat, Raised, RaisedInner, RaisedOuter, Sunken, SunkenInner, SunkenOuter, RoundedSides.</li> <li>• <b>Margin</b> — Border margin, in pixels</li> <li>• <b>ThicknessDesired</b> — Specified thickness, in pixels</li> </ul>
Set font properties	<p><b>ScaleDisplay &gt; TickMajor &gt; Font</b></p> <p><b>ScaleDisplay &gt; TickMid &gt; Font</b></p> <ul style="list-style-type: none"> <li>• <b>Name</b> — Font for text display.</li> <li>• <b>Font style</b> — One of Light, Semilight, Regular, Italic, Semibold, Bold, Bold Italic, Light Oblique, Semibold Oblique.</li> <li>• <b>Size</b> — Font size, in points</li> <li>• <b>Strikeout</b> — If True, add strikeout effect.</li> <li>• <b>Underline</b> — If True, add underline effect.</li> </ul>
Set color properties	<p><b>ColorSections</b></p> <p>By default, the list of color sections is empty. To add color sections, open the Color Section Collection Editor dialog box and click <b>Add</b>.</p> <ul style="list-style-type: none"> <li>• <b>Color</b> — Open the Color dialog box, which contains the color palettes <b>Custom</b>, <b>Web</b>, and <b>System</b>.</li> <li>• <b>Start</b> — Start of color section, in range units.</li> <li>• <b>Stop</b> — End of color section, in range units.</li> </ul> <p>Color properties are associated with the following instrument properties: <b>Border</b>, <b>Pointer</b>, <b>ScaleDisplay &gt; TickMajor</b>, <b>ScaleDisplay &gt; TickMid</b>, <b>ScaleDisplay &gt; TickMinor</b>.</p>

## Appearance Properties

The table shows some of the ways that you can modify the appearance of your instruments through the **Appearance** properties.

Action	Use
Set font properties	<ul style="list-style-type: none"> <li>• <b>Name</b> — Font for text display.</li> <li>• <b>Font style</b> — One of <code>Light</code>, <code>Semilight</code>, <code>Regular</code>, <code>Italic</code>, <code>Semibold</code>, <code>Bold</code>, <code>Bold Italic</code>, <code>Light Oblique</code>, <code>Semibold Oblique</code>.</li> <li>• <b>Size</b> — Font size, in points</li> <li>• <b>Strikeout</b> — If True, add strikeout effect.</li> <li>• <b>Underline</b> — If True, add underline effect.</li> </ul>
Set color properties	<b>BackColor</b> , <b>ForeColor</b> — Open the color palettes <b>Custom</b> , <b>Web</b> , and <b>System</b> .
Set image properties	<ul style="list-style-type: none"> <li>• <b>BackgroundImage</b> — Navigate to a background image file.</li> <li>• <b>BackgroundImageLayout</b> — One of <code>Tile</code>, <code>None</code>, <code>Center</code>, <code>Stretch</code>, <code>Zoom</code>.</li> </ul>

## See Also

GaugeAngular Properties | GaugeFluidLevel Properties | GaugeLinear Properties | GroupBox Properties | HexadecimalDisplay Properties | Knob Properties | LED Properties | Label Properties | NumericDisplay Properties | NumericEntry Properties | NumericUpDownEntry Properties | Panel Properties | PictureBox Properties | Scope Properties | Slider Properties | SwitchLED Properties | SwitchLever Properties | SwitchRocker Properties | SwitchToggle Properties

## Related Examples

- “View Signal Waveforms with Scope Instrument” on page 4-15
- “View Signal Waveforms with Signal Triggered Scope Instrument” on page 4-21
- “Add Instrument Panel to Tank Model” on page 4-25
- “Triggering Scope Instruments”

## More About

- **Simulink Real-Time Explorer**

## Explorer Configuration Exported to Run Outside MATLAB

You can export a Simulink Real-Time Explorer configuration as a standalone executable to run outside MATLAB on a computer compatible with Windows.

Before exporting a Simulink Real-Time Explorer configuration, set up the execution environment, signal and parameter groups, and panels. Lay out the windows the way that you want them in the standalone executable. Follow these guidelines:

### Execution Environment

For each computer on which you intend to run the standalone Simulink Real-Time Explorer executable:

- Check that the candidate computer is compatible with 64-bit Windows.
- Check that the CPU and operating system meet the requirements for executing the standalone Simulink Real-Time Explorer executable.
- Check that Microsoft .NET Framework 4.5 is installed on the candidate computer.

For each target computer on which you intend to run the real-time application:

- Check that the target computer meets the requirements for running the real-time application.
- In Simulink Real-Time Explorer, configure the target and communication settings to connect each computer that is compatible with Windows to each target computer.

You can have only one target computer node for each unique **IP address** setting.

- Configure the **Boot mode** setting for each target computer as Stand Alone.
- Optionally, rename the target computer session from TargetPCx to something more specific to your system.
- As a test, build and download a real-time application to each target computer connected to the development computer running Simulink Real-Time Explorer.

The real-time application on the target computer is the same application that you intend to access with the standalone executable.

### Signal Groups

- To access signals, add them from the model hierarchy to a signal group.
- To include a signal group in the standalone package, load it into the current session.

### Parameter Groups

- To access parameters, add them from the model hierarchy to a parameter group.
- To include a parameter in the standalone package, load it into the current session.

### Instrument Panels

- To interact with multiple target computers, create a separate instrument panel for each separate real-time application and target computer pair.

- To include an instrument panel in the standalone package, load it into the current Simulink Real-Time Explorer session.
- If you renamed the target computer, update the **TargetName** parameter for each instrument to maintain the connection to the real-time application.

### Window Layout

- You can configure which windows the software opens on startup by opening the windows and arranging them accordingly. When you export the model configuration, the software includes the windows layout in the standalone package.
- To return to the initial standalone executable layout, click **File > Restore Initial View**.

### Limitations as Standalone Executable

When you export Simulink Real-Time Explorer as a standalone executable, it supports a subset of the capabilities that it supports under MATLAB.

- You cannot change the communication parameters that the interface uses to communicate with the target computers. Before you export the Simulink Real-Time Explorer configuration, configure and test the communication parameters for each target computer.
- For each instrument, the exporting software records the real-time application and target computer environment with which it is associated.
- If you rename a target computer, to maintain the connection to the real-time application, update the **TargetName** parameter for each associated instrument.
- You cannot load or unload a real-time application from the standalone executable. Before you start the executable, start the real-time application on the target computer.
- You can access only instrument panels and windows that you loaded before you exported the configuration.
- You cannot access the real-time application model hierarchy from the standalone executable.
- You can access only signals in signal groups that you loaded before you exported the configuration.
- You cannot move a signal from one signal group to another group, or create or load a new signal group.
- You can access only parameters in parameter groups that you loaded before you exported the configuration.
- You cannot move a parameter from one parameter group to another group, or create or load a new parameter group.
- You cannot save session layouts. If you close a window, you can restore the original layout using **File > Restore Original View**.

### See Also

#### More About

- “Standalone Boot Method”



## View Signal Waveforms with Scope Instrument

### In this section...

“Create Instrument Panel” on page 4-15

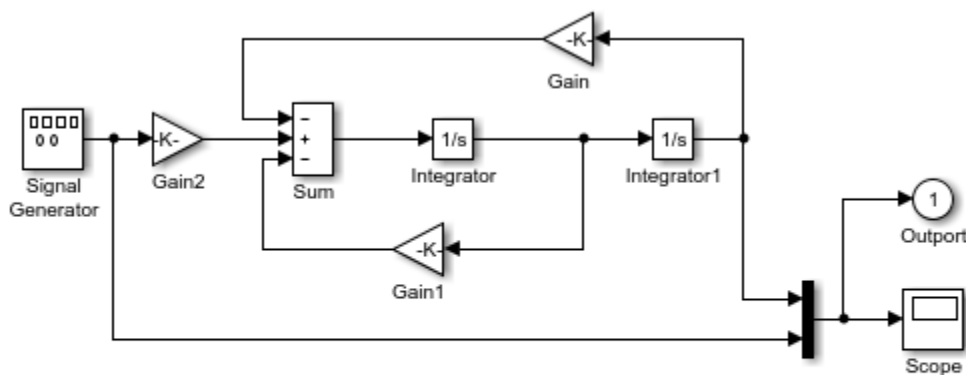
“Configure Slider Instrument for Parameter Tuning” on page 4-16

“Configure Scope Instrument with Default Triggering” on page 4-17

“Run Instrumented Model” on page 4-18

In this example, create an instrument panel for the `xpcosc` model that contains the following instruments:

- Slider — To tune the input signal amplitude (Signal Generator/Amplitude).
- Slider — To tune the oscillator feedback damping parameter (Gain1/Gain).
- Scope — To display the input signal (Signal Generator) and the oscillator output (Integrator1).




Model `xpcosc`  
Simulink Real-Time example model

Copyright 1999-2013 The MathWorks, Inc.

Start by building and downloading the real-time application to the target computer, running Simulink Real-Time Explorer, and connecting Explorer to the target computer.

### Create Instrument Panel

Create and save an instrument panel for the `xpcosc` model.

- 1 In the **Panels** pane, right-click the **Instrument Panels** node, and then click **Add New**.
- 2 Type a name for the instrument panel in the **Name** text box. Give the panel a name like `xpcosc_scope_instr_freerun.slrtip`. Type a folder location in the **Location** text box, and then press **Enter**.
- 3 Click the **Save** button .

## Configure Slider Instrument for Parameter Tuning

Select and configure an instrument to tune two parameters in the `xpcosc` model. You must have previously created the `xpcosc_scope_instr_freerun.slrtip` instrument panel.

The parameter characteristics are listed in this table.

Name	Type	Range	Purpose
Amplitude	Numeric	0–10 units	Represents the amplitude of the input signal Signal Generator. You do not have to set it to an exact value.
Gain1	Numeric	0–1000 units	Represents the damping of the oscillator feedback signal Gain1. You do not have to set it to an exact value.

The Slider instrument meets the requirement for Amplitude and Gain1. To set exact numeric values, use, for example, a NumericEntry instrument.

To select and configure the instrument:



- 1 Load the instrument panel.

In the **Panels** pane, right-click the **Instrument Panels** node and select **Existing**. From the list, select `xpcosc_scope_instr_freerun.slrtip`.

- 2 Select the instrument.


From the **Palette** pane, drag a Slider instrument to the `xpcosc_scope_instr_freerun.slrtip` instrument panel.

- 3 Bind the parameter to the instrument.

To bind the Amplitude parameter to the Slider instrument, open the Parameter workspace for model `xpcosc` (  on the toolbar). Drag the **Parameter** icon  next to parameter Amplitude to the Slider instrument.

A small copy of the **Parameter** icon appears next to the Slider instrument.

- 4 Set the instrument range.

Click the Slider instrument, and then click the **Tasks** button  in the top, right corner.

- 5 In the **Slider Tasks** dialog box, set property **Min** to 0 and property **Span** to 10.

- 6 Select and configure a label.

From the **Palette** pane, drag a Label layout item under the Slider instrument.

- 7 Click the Label element.

- 8 In the **Properties** pane, scroll down to the **Appearance** node. Set the **Text** property to Amplitude, and then press **Enter**.

- 9 Scroll down to the **TextAlign** property. Click the down arrow and click the center of the nine blocks presented.

The **TextAlign** property becomes MiddleCenter.

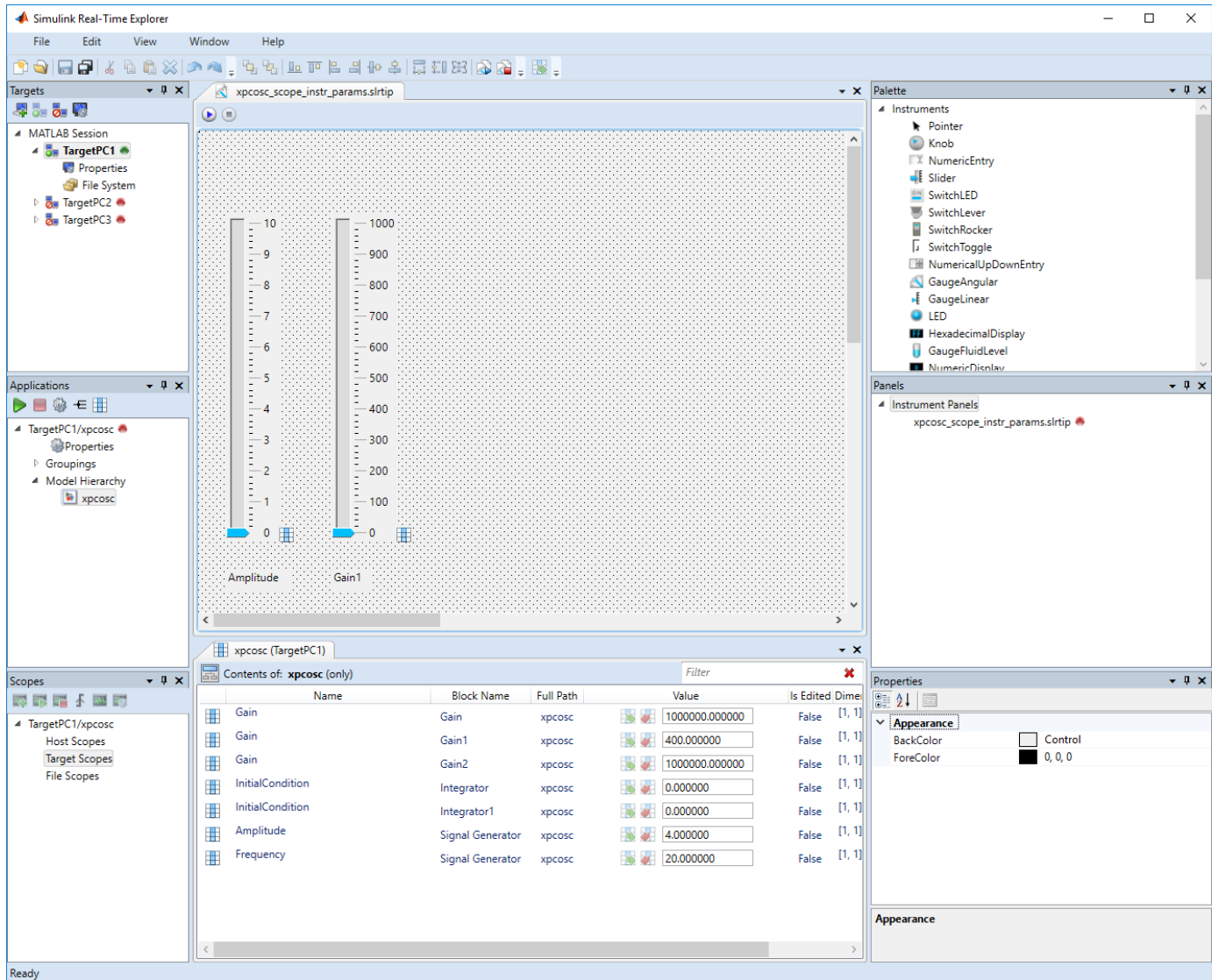
10 Repeat steps 2–9 for parameter Gain1.

Set the property **Min** to 0 and property **Span** to 1000.

Set the **Text** property to Gain1.

11 Click the **Save** button .

At the end of this task, Simulink Real-Time Explorer looks like this figure.



## Configure Scope Instrument with Default Triggering

Add a Scope instrument and configure it with the default triggering, which is a TriggerMode of SINGLESHOT and a TriggerSource of FREERUN. You must have previously created the xpcosc\_scope\_instr\_freerun.slrtip instrument panel.

The signal characteristics are listed in this table.

Name	Type	Range	Purpose
Signal Generator	Numeric	-25 : 25 units	Represents the time-varying value of the signal generator input.
Integrator1	Numeric	-25 : 25 units	Represents the time-varying value of the oscillator output.

To select and configure the Scope instrument:

- 1 Load the instrument panel.

In the **Panels** pane, right-click the **Instrument Panels** node and select **Existing**. From the list, select `xpcosc_scope_instr_freerun.slrtip`.

- 2 Select the instrument.

From the **Palette** pane, drag a Scope instrument to the `xpcosc_scope_instr_freerun.slrtip` instrument panel.

- 3 Bind the signals to the instrument.

To bind the Signal Generator signal to the Scope instrument, open the Signal workspace for model `xpcosc` (☰ on the toolbar). Click the **Signal** icon (⊞) next to signal Signal Generator and drag the icon to the Scope instrument.

The name `Signal Generator:1` appears in the Scope instrument legend.

- 4 In a similar manner to step 3, bind the Integrator1 signal to the Scope instrument
- 5 Set the instrument range.

Select the Scope instrument, and then click the **YAxesLimits** node in the **Properties** list.

- 6 Set property **Max** to 25 and **Min** to -25.



- 7 Click the **Save** button .

## Run Instrumented Model

Run the instrumented `xpcosc` model, tune the parameters, and view the output waveform. You must have previously built and downloaded the `xpcosc` model and configured the `xpcosc_scope_instr_freerun.slrtip` instrument panel.

- 1 Load the instrument panel.


In the **Panels** pane, right-click the **Instrument Panels** node and select **Existing**. From the list, select `xpcosc_scope_instr_freerun.slrtip`.

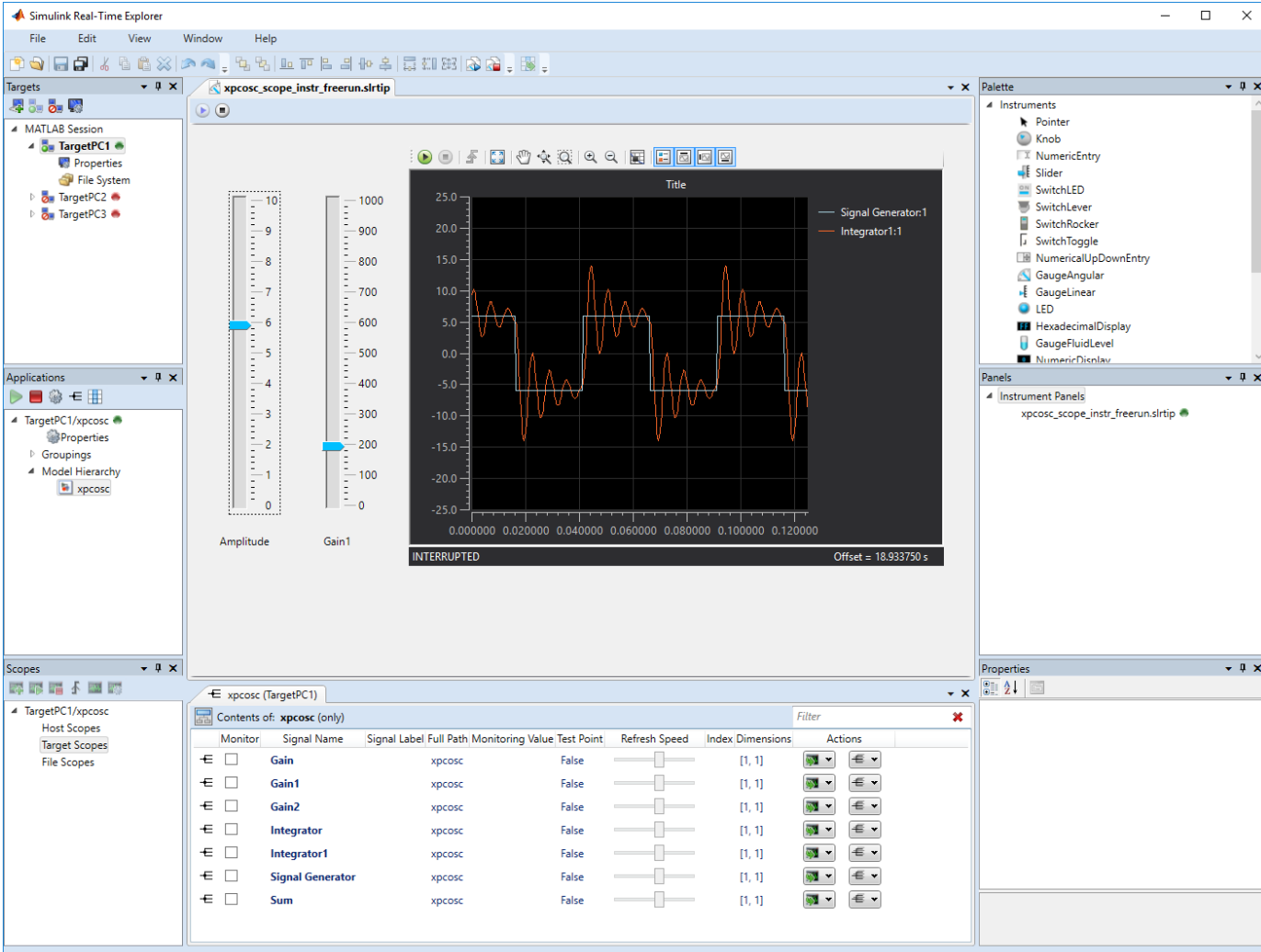
- 2 Set property **Stop time** to `inf` in the **Applications** pane (⚙ on the toolbar).
- 3 To start the instrument, in the `xpcosc_scope_instr_freerun.slrtip` instrument panel, click the **Run Instrument** button .
- 4 To start execution, in the **Applications** pane, click the real-time application, and then click the **Start** button  on the toolbar.

The Scope instrument displays one set of captured waveforms, and then stops.

- To change the amplitude and damping value, use the **Amplitude** and **Gain1** instruments.

For example, to increase the amplitude and reduce the oscillator damping, set **Amplitude** to 6 and **Gain1** to 200.















- To view the result of the parameter changes, click the **Start Acquisition** button  on the Scope toolbar.





The screenshot shows the Simulink Real-Time Explorer interface. The main window displays a scope instrument titled "xpcosc\_scope\_instr\_freerun.slrtip". The scope plot shows two signals: "Signal Generator:1" (blue) and "Integrator:1:1" (orange). The plot shows a square wave signal with a superimposed oscillation. The x-axis represents time from 0.000000 to 0.120000, and the y-axis ranges from -25.0 to 25.0. The plot is currently interrupted, with an offset of 18.933750 s.

On the left side, there are two vertical sliders: "Amplitude" (ranging from 0 to 10) and "Gain1" (ranging from 0 to 1000). The "Amplitude" slider is set to 6, and the "Gain1" slider is set to 200.

At the bottom, the "Scopes" pane shows the contents of the "xpcosc" scope. The table below lists the monitored signals:

Monitor	Signal Name	Signal Label	Full Path	Monitoring Value	Test Point	Refresh Speed	Index	Dimensions	Actions
<input type="checkbox"/>	Gain	xpcosc		False			[1, 1]		 
<input type="checkbox"/>	Gain1	xpcosc		False			[1, 1]		 
<input type="checkbox"/>	Gain2	xpcosc		False			[1, 1]		 
<input type="checkbox"/>	Integrator	xpcosc		False			[1, 1]		 
<input type="checkbox"/>	Integrator1	xpcosc		False			[1, 1]		 
<input type="checkbox"/>	Signal Generator	xpcosc		False			[1, 1]		 
<input type="checkbox"/>	Sum	xpcosc		False			[1, 1]		 

- To stop execution, in the **Applications** pane, click the real-time application, and then click the **Stop** button  on the toolbar.

- To stop the instruments, in the `xpcosc_scope_instr_freerun.slrtip` instrument panel, click the **Stop Instrument** button .

**See Also**  
Scope | Slider

### **More About**

- “View Signal Waveforms with Signal Triggered Scope Instrument” on page 4-21
- “Add Instrument Panel to Tank Model” on page 4-25
- “Triggering Scope Instruments”
- “Save and Load Instrument Panels” on page 4-36
- “Save and Restore Layouts” on page 4-37
- “Instrumentation for Real-Time Applications” on page 4-2
- “Display and Filter Hierarchical Signals and Parameters”

## View Signal Waveforms with Signal Triggered Scope Instrument

The default trigger mode of the Scope instrument is SINGLESHOT, which displays one sample sweep. To trace a waveform in a continuously updated moving display, you can change the trigger mode to REPEATED. However, a moving display is often difficult to evaluate.

In this example, to produce a continuously updated static display, we configure the scope instrument to trigger mode REPEATED with trigger source Signal.

### Preconditions

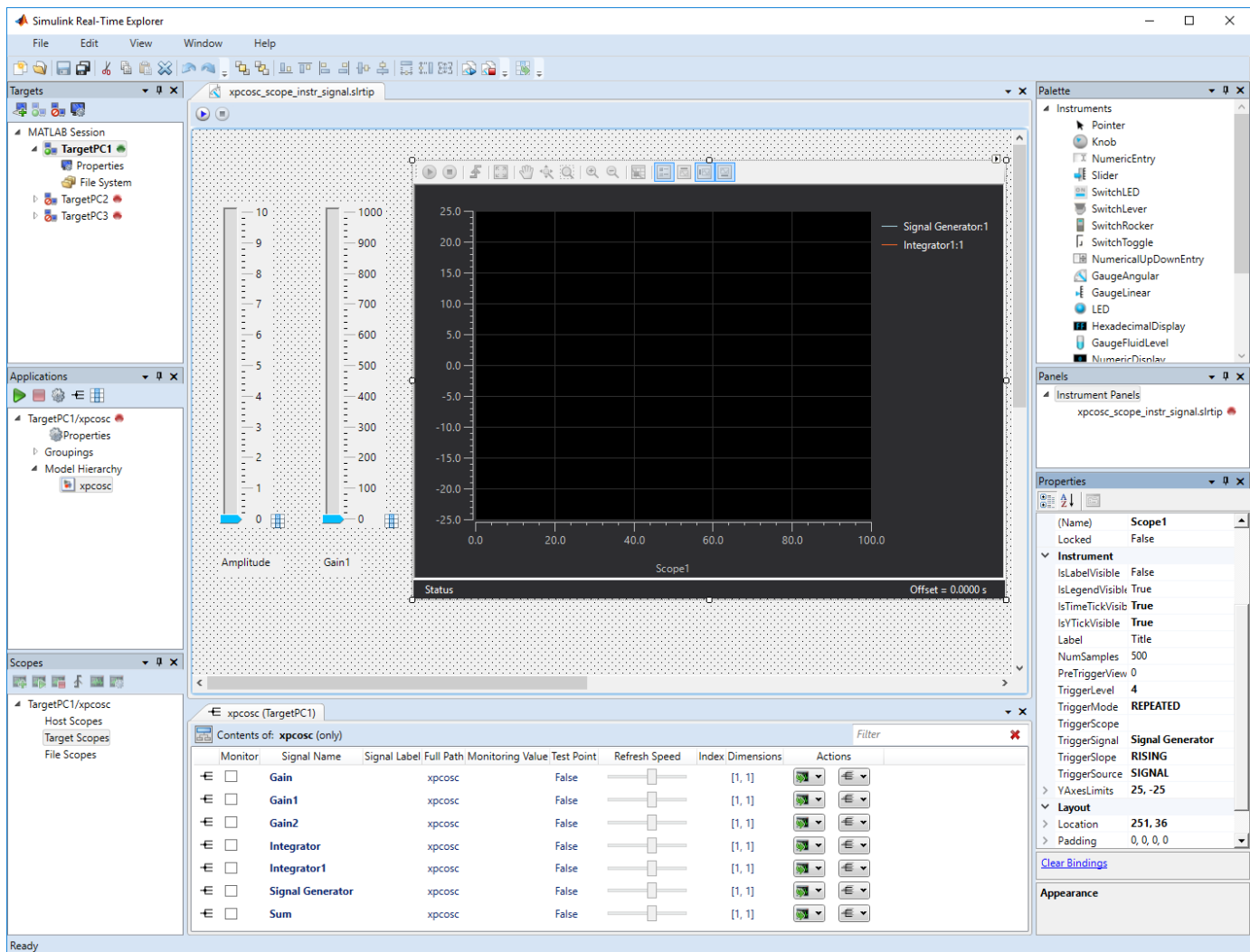
You must have previously built and downloaded the `xpcosc` model and configured the `xpcosc_scope_instr_freerun.slrtip` instrument panel. See “View Signal Waveforms with Scope Instrument” on page 4-15.

### Configure Scope Instrument

- 1 Load the instrument panel.

In the **Panels** pane, right-click the **Instrument Panels** node and select **Existing**. From the list, select `xpcosc_scope_instr_freerun.slrtip`.

- 2 Select the Scope instrument.
- 3 Open the **Instrument** properties list.
- 4 Set **TriggerMode** to REPEATED.
- 5 Set **TriggerSource** to Signal.
- 6 Set **TriggerSignal** to Signal Generator.
- 7 Set **TriggerSlope** to Rising.
- 8 Set **TriggerLevel** to 4.
- 9 Save the instrument panel under another name, such as `xpcosc_scope_instr_signal.slrtip`.



## Run Instrumented Model

Run the instrumented xpcosc model, tune the parameters, and view the output waveform with signal triggering and forced triggering. You must have previously built and downloaded the xpcosc model and configured the xpcosc\_scope\_instr\_signal.slrtip instrument panel.

- 1 Load the instrument panel.

In the **Panels** pane, right-click the **Instrument Panels** node and select **Existing**. From the list, select xpcosc\_scope\_instr\_signal.slrtip.

- 2 Set property **Stop time** to `inf` in the **Applications** pane (⚙️ on the toolbar).
- 3 To start the instruments, in the xpcosc\_scope\_instr\_signal.slrtip instrument panel, click the **Run Instrument** button (▶️).
- 4 To start execution, in the **Applications** pane, click the real-time application, and then click the **Start** button (▶️) on the toolbar.



The Scope instrument displays a continuous series of captured waveforms.


- To change the amplitude and damping value, use the **Amplitude** and **Gain1** instruments.

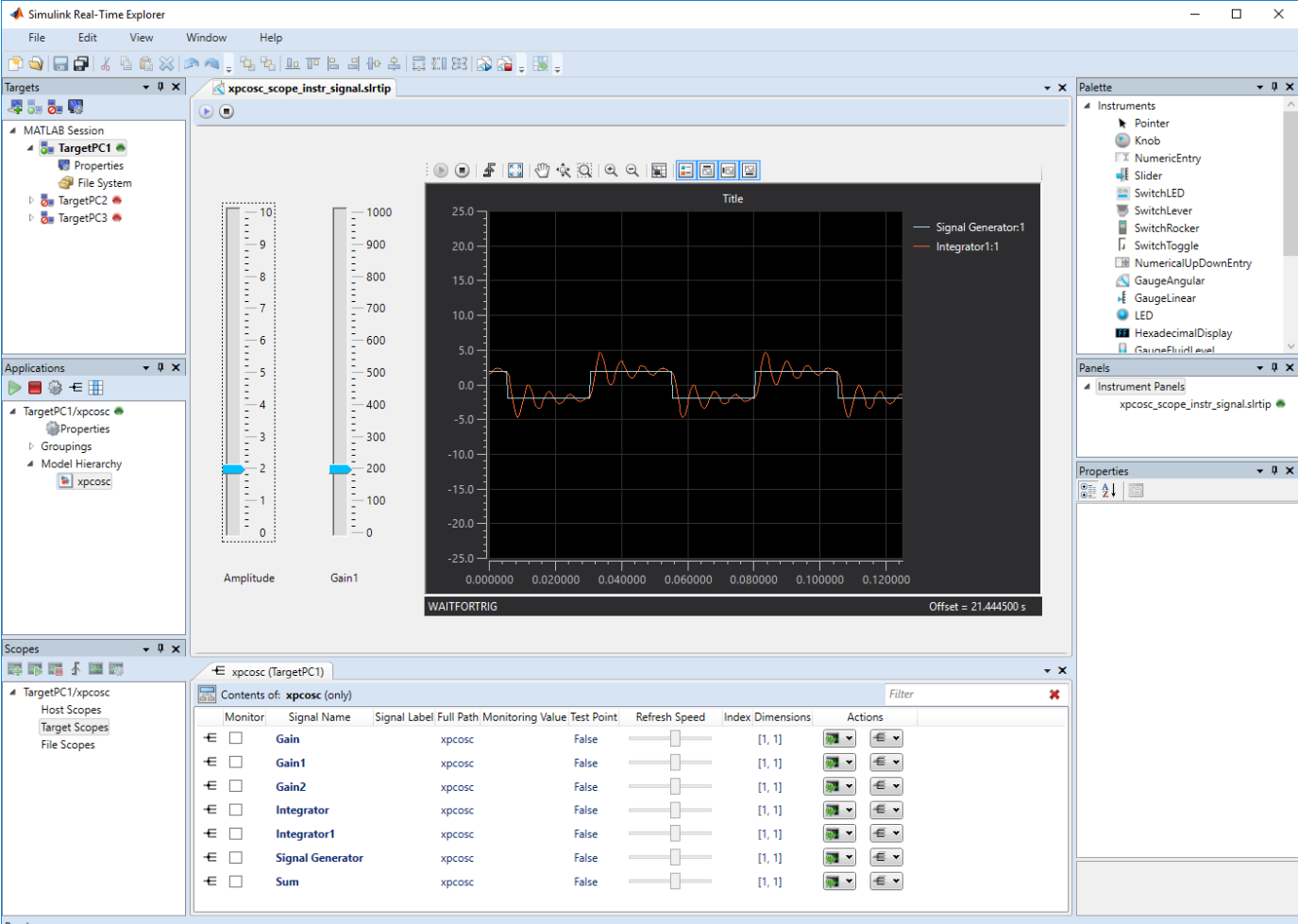
For example, to increase the amplitude and reduce the oscillator damping, set **Amplitude** to 6 and **Gain1** to 200.

The Scope instrument displays the changes in waveforms immediately.

- Reduce the amplitude below the **TriggerLevel** (4).

The Scope instrument stops displaying the changes in waveforms because the signal does not have a high enough amplitude to reach the trigger level.

- To display waveforms with an amplitude below the trigger level, click the **Force Trigger** button  on the toolbar.





The screenshot shows the Simulink Real-Time Explorer interface. The main window displays a plot titled "Title" showing two signals: "Signal Generator:1" (blue) and "Integrator:1:1" (orange). The plot shows a square wave signal with a period of approximately 0.04 seconds. The y-axis ranges from -25.0 to 25.0, and the x-axis ranges from 0.000000 to 0.120000. The plot is currently in a "WAITFORTRIG" state, as indicated by the text at the bottom of the plot area. The offset is 21.444500 s.

On the left side of the plot, there are two vertical sliders: "Amplitude" (ranging from 0 to 10) and "Gain1" (ranging from 0 to 1000). The "Amplitude" slider is currently set to approximately 2.5, and the "Gain1" slider is set to approximately 200.

At the bottom of the interface, there is a table showing the contents of the "xpcosc" instrument. The table has columns for Monitor, Signal Name, Signal Label, Full Path, Monitoring Value, Test Point, Refresh Speed, Index Dimensions, and Actions.

Monitor	Signal Name	Signal Label	Full Path	Monitoring Value	Test Point	Refresh Speed	Index Dimensions	Actions
<input type="checkbox"/>	Gain	xpcosc	xpcosc	False			[1, 1]	
<input type="checkbox"/>	Gain1	xpcosc	xpcosc	False			[1, 1]	
<input type="checkbox"/>	Gain2	xpcosc	xpcosc	False			[1, 1]	
<input type="checkbox"/>	Integrator	xpcosc	xpcosc	False			[1, 1]	
<input type="checkbox"/>	Integrator1	xpcosc	xpcosc	False			[1, 1]	
<input type="checkbox"/>	Signal Generator	xpcosc	xpcosc	False			[1, 1]	
<input type="checkbox"/>	Sum	xpcosc	xpcosc	False			[1, 1]	

- To stop execution, in the **Applications** pane, click the real-time application, and then click the **Stop** button  on the toolbar.
- To stop the instruments, in the `xpcosc_scope_instr_signal.slrtip` instrument panel, click the **Stop Instrument** button .

### **See Also**

Scope | Slider

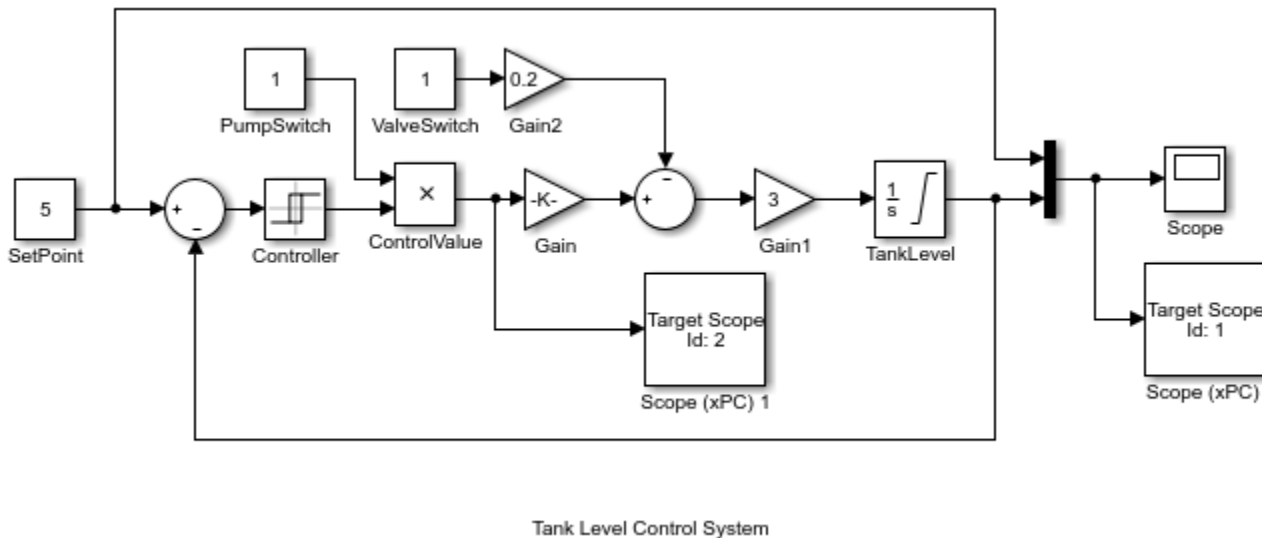
### **More About**

- “View Signal Waveforms with Scope Instrument” on page 4-15
- “Add Instrument Panel to Tank Model” on page 4-25
- “Triggering Scope Instruments”
- “Save and Load Instrument Panels” on page 4-36
- “Save and Restore Layouts” on page 4-37
- “Instrumentation for Real-Time Applications” on page 4-2
- “Display and Filter Hierarchical Signals and Parameters”

## Add Instrument Panel to Tank Model

This example shows how to create an instrument panel in Simulink® Real-Time™ Explorer for the Simulink Real-Time application built from the `xpctank` model. The instrument panel contains these instruments:

- Slider — To tune the required tank level (SetPoint).
- GaugeFluidLevel — To display the actual tank level (TankLevel).
- LED — To display the pump control status (ControlValue).



This example also shows how to update the real-time application by adding a signal with enum data type in the model and rebuilding the application. After downloading the updated real-time application to the target, you can add an instrument to display the enumerated states as the instrument units.

### Prerequisites

This example assumes knowledge of how to complete these operations:

- Open Simulink Real-Time Explorer, `slrtexplr`.
- Start the target computer.
- Connect `slrtexplr` to the target computer.
- Open the `xpctank` model.  
`open_system(docpath(fullfile(matlabroot,'toolbox','rtw','targets','xpc','xpcdemos','xpctank')))`
- Build the `xpctank` model and download the real-time application to the target computer.
- Open the parameter and signal panels for the `xpctank` model in `slrtexplr`.

### Create Instrument Panel

To provide an interactive instrument panel display for simulation, create and save an instrument panel for the `xpctank` real-time application.

To create an instrument panel in `slrtexplr`:

1. In the **Panels** pane, right-click the **Instrument Panels** node, then click **New**.
2. In the **Name** field, type a `xpctank_instr_design.slrtip` for the name.
3. In the **Location** field, select the folder for the panel file.
4. Press **Enter** and click the **Save** button.

### Configure Instrument for Parameter Tuning

To tune a parameter from the `xpctank` real-time application, select and configure an instrument in the `xpctank_instr_design.slrtip` instrument panel.

#### Parameter Characteristics

Before binding a parameter to an instrument, examine the characteristics of the parameter for compatibility with the instrument.

The parameter characteristics for the `SetPoint` block parameter are:

- Type — Numeric
- Range — 0-10 units
- Purpose — Represents the level at which the controller maintains the tank fluid level. You do not have to set the level to an exact value.

The Slider instrument is compatible with the characteristics of `SetPoint`. To set an exact numeric value, use a `NumericEntry` instrument, for example.

#### Configure Slider Instrument for SetPoint Tuning

To select and configure the Slider instrument from the `xpctank_instr_design.slrtip` instrument panel:

1. Select the instrument. From the **Palette** pane, drag a Slider instrument to the `xpctank_instr_design.slrtip` instrument panel.
2. Access the block parameter list. In the **Applications** pane, right-click `xpctank` and select **View Block Parameters**.
3. Bind the `SetPoint` parameter to the instrument. Drag the **Parameter** icon next to the parameter `SetPoint` and drop it into the Slider instrument. Next to the Slider instrument, a small copy of the **Parameter** icon appears.



4. Set the instrument range. Click the Slider instrument, and then click the **Tasks** button in the top right corner.
5. In the **Slider Tasks** dialog box, set property **Min** to 0 and property **Span** to 10.
6. Select a label. From the **Palette** pane, drag a Label layout item to underneath the Slider instrument.

7. Configure the label. Click the Label element. In the lower right corner, select the **Properties** pane.
8. Scroll down to the **Appearance** node. Set the **Text** property to Set Point, and then press **Enter**.
9. Scroll down to the **TextAlign** property. Click the down arrow and click the center block of the nine blocks presented. The **TextAlign** property becomes MiddleCenter.
10. Click the **Save** button.

At the end of this task, Simulink Real-Time Explorer displays the instrument panel, model signals panel, and model parameters panel.

### Configure Instruments for Signal Display

To display two signals from the xpctank model, select and configure instruments in the xpctank\_instr\_design.slrtip instrument panel.

#### Signal Characteristics

Before binding a signal to an instrument, examine the characteristics of the signal for compatibility with the instrument.

The signal characteristics for the TankLevel signal are:

- Type — Numeric
- Range — 0–10 units
- Purpose — Represents the current tank fluid level. You do not have to display an exact value.

The GaugeFluidLevel instrument is compatible with the characteristics of TankLevel. To display an exact numeric value, use a NumericDisplay instrument, for example.

The signal characteristics for ControlValue are:

- Type — Boolean
- Range — 1, 0
- Purpose — Represents the state of the pump (on or off).

The LED instrument is compatible with the characteristics of ControlValue.

### Configure Instruments for TankLevel and ControlValue Display

To select and configure instruments from the xpctank\_instr\_design.slrtip instrument panel:

1. Select the instrument. From the **Palette** pane, drag a GaugeFluidLevel instrument to the xpctank\_instr\_design.slrtip instrument panel.
2. Access the signal list. In the **Applications** pane, right-click xpctank and select View Signals.
3. Bind the signal to the instrument. To bind the TankLevel signal to the GaugeFluidLevel instrument, in the **Applications** pane, right-click xpctank and select View Signals. Drag the **Signal** icon next to signal TankLevel and drop it into the GaugeFluidLevel instrument. Next to the GaugeFluidLevel instrument, a small copy of the **Signal** icon appears.



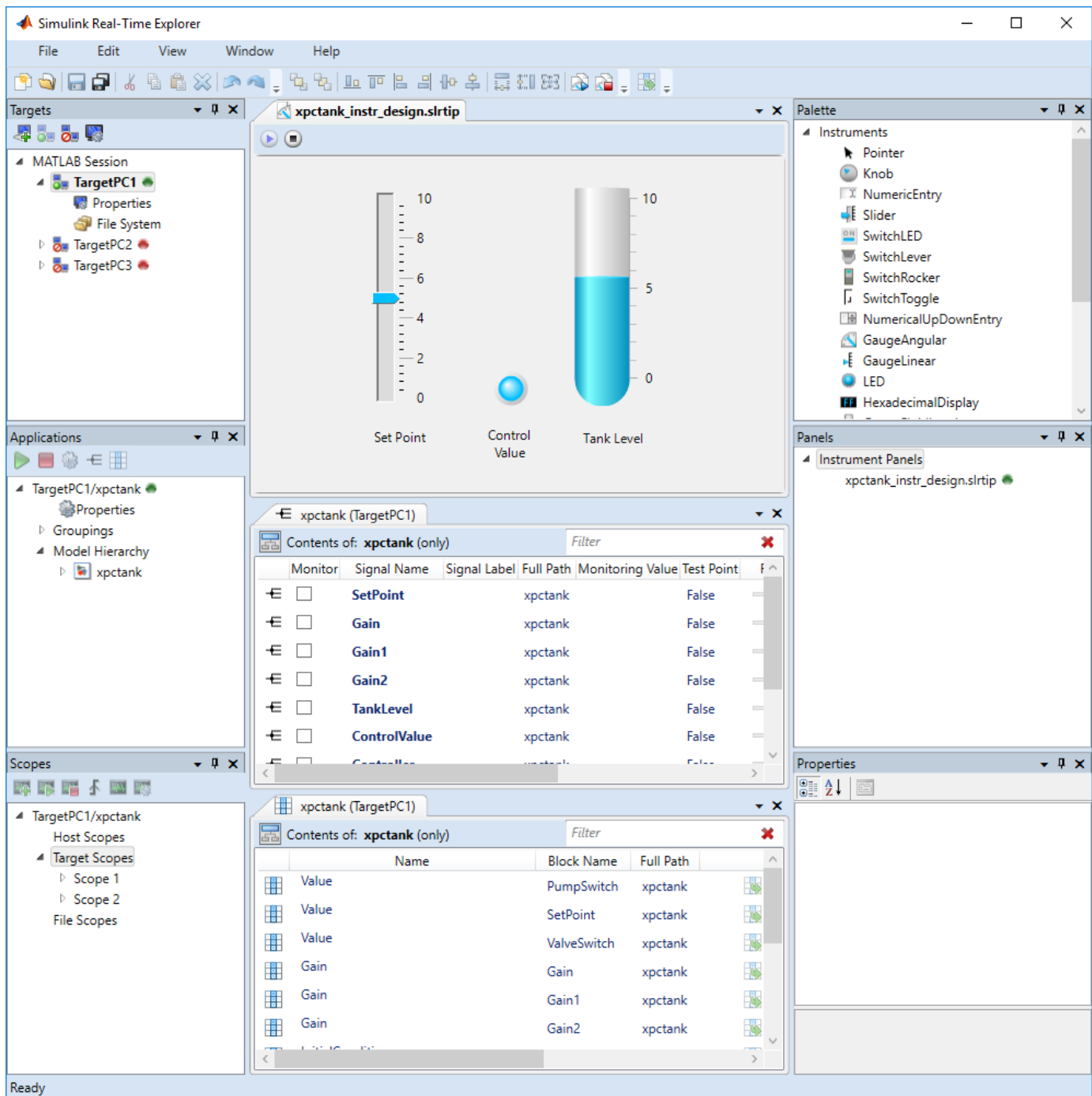
4. Set the instrument range as required. Select the GaugeFluidLevel instrument, and then click the **Tasks** button in the top right corner.
5. In the **GaugeFluidLevel Tasks** dialog box, set property **Min** to 0 and property **Span** to 10.
6. Select a label. From the **Palette** pane, drag a Label layout item to underneath the GaugeFluidLevel instrument.
7. Configure the label. Click the Label element. In the lower right corner, select the **Properties** pane.
8. Scroll down to the **Appearance** node. Set the **Text** property to Tank Level, and then press **Enter**.
9. Scroll down to the **TextAlign** property. Click the down arrow and click the center block of the nine blocks presented. The **TextAlign** property becomes MiddleCenter.
10. Click the **Save** button.

Using a similar procedure, add an LED instrument to the instrument panel and bind signal ControlValue to it. Label the LED Control Value. At the end of this task, Simulink Real-Time Explorer displays the SetPoint, TankLevel, and ControlValue on the instrument panel.

### Run the Model and Instrument Panel

You have built and downloaded the xpctank model and configured the xpctank\_instr\_design.slrtip instrument panel for the model. You can now run the model and instrument panel.

1. In the **Applications** pane, set property **Stop time** to inf.
2. To start model execution, in the **Applications** pane, click the real-time application, and then click the **Start** button.
3. To start the instrument panel, in the xpctank\_instr\_design.slrtip instrument panel, click the **Run** button.
4. Using the Slider instrument, set the tank level to the required value, such as 5. The tank level rises to and oscillates around the set point.



To stop model execution, in the **Applications** pane, click the real-time application, and then click the **Stop** button.

To stop the instruments, in the xpctank\_instr\_design.slrtip instrument panel, click the **Stop** button.

## Add Signal of enum Type and Display with Instrument

For some signals or parameters in the model, it is helpful to define their data as type enum and use the enumerated values as the scale of the instrument display.

The instruments that support enumerated values as scales for signals are:

- GaugeAngular
- GaugeLinear
- GaugeFluidLevel

The instruments that support enumerated values as scales for parameters are:

- Knob
- Slider

The LED on the instrument panel indicates the open or closed state of the valve that fills the tank.

Using a signal with enum data type, a more informative display for the state of the valve is available with a GaugeAngular instrument.

1. Open the `xpctank` model.

```
open_system(docpath(fullfile(matlabroot,'toolbox','rtw','targets','xpc','xpcdemos','xpctank')))
```

2. In MATLAB, define the enum data type with this class definition with a function call, for example: `run(docpath(fullfile(matlabroot,'examples','xpc','main','myEnumTypeDefinition.m')))`

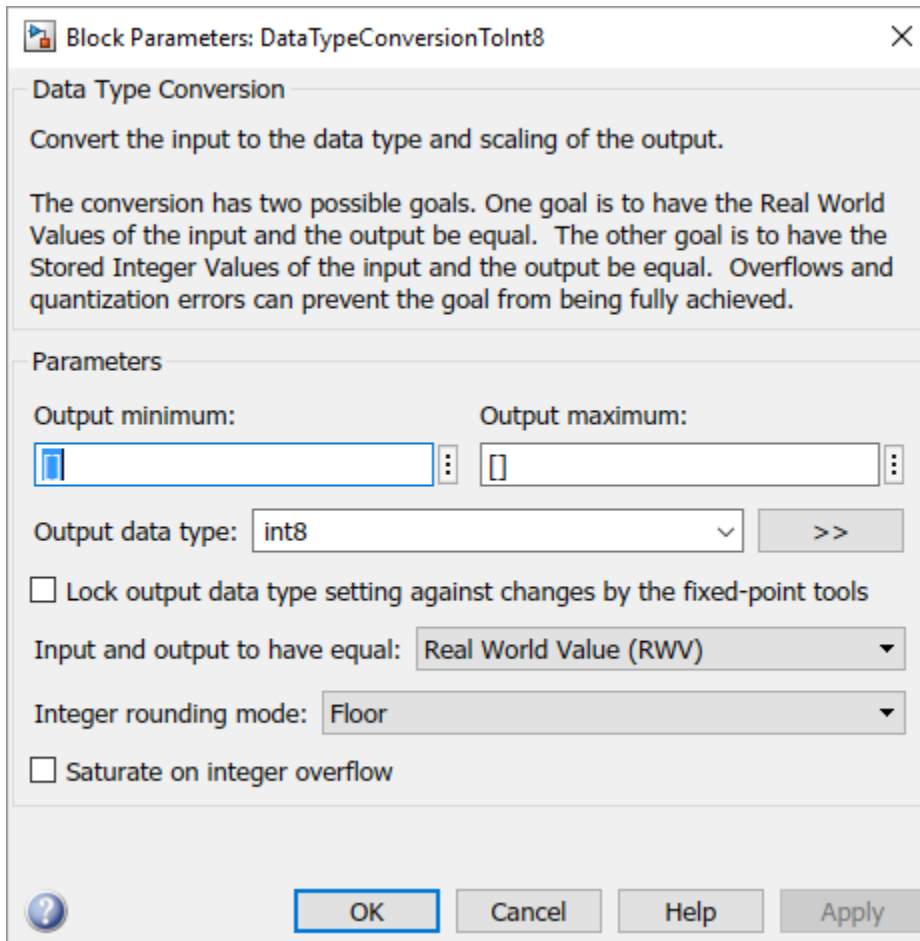
This function defines the class as:

```
Simulink.defineIntEnumType('ControlValueState', ...  
    {'Closed', 'Open'}, ...  
    [0;10], ...  
    'Description', 'Control value state', ...  
    'DefaultValue', 'Closed', ...  
    'HeaderFile', 'mycontrolvaluestate.h', ...  
    'DataScope', 'Exported', ...  
    'AddClassNameToEnumNames', true, ...  
    'StorageType', 'int8');
```

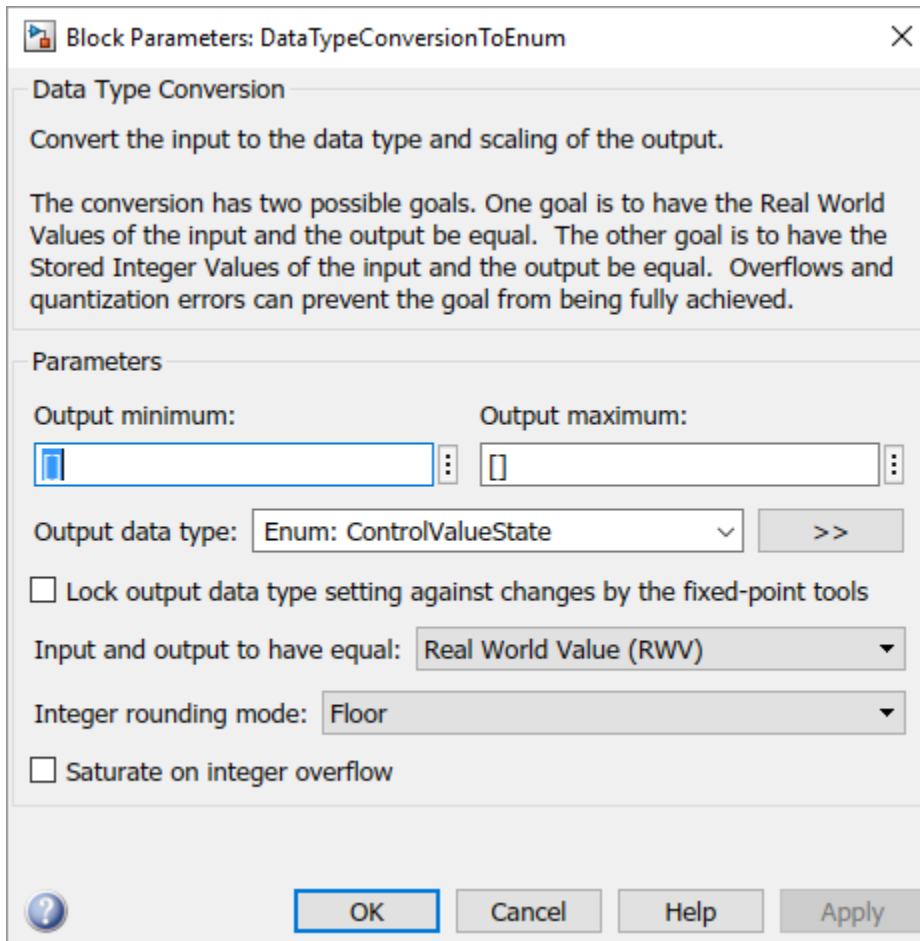
3. In Simulink, update the `xpctank` model, by adding:

- A Data Type Conversion block (DataTypeConversionToInt8) to convert the ControlValue signal data type to int8; set the block properties as shown:



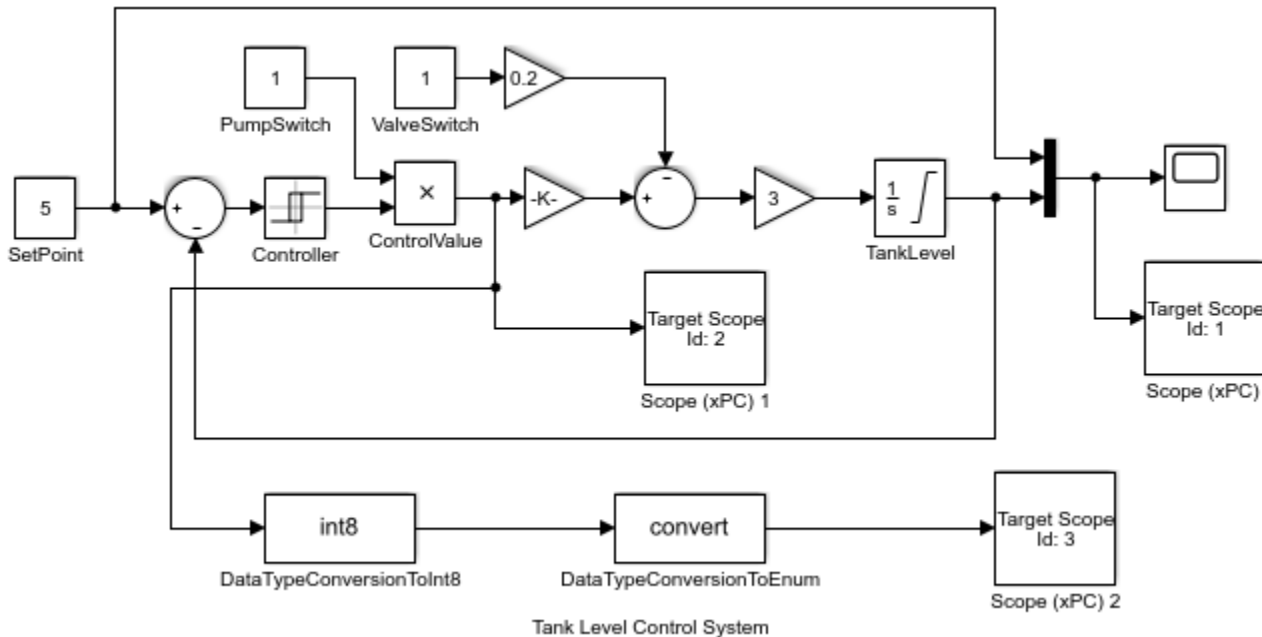


- A Data Type Conversion block (DataTypeConversionToEnum) to convert the int8 signal data type to enum; set the block properties as shown:



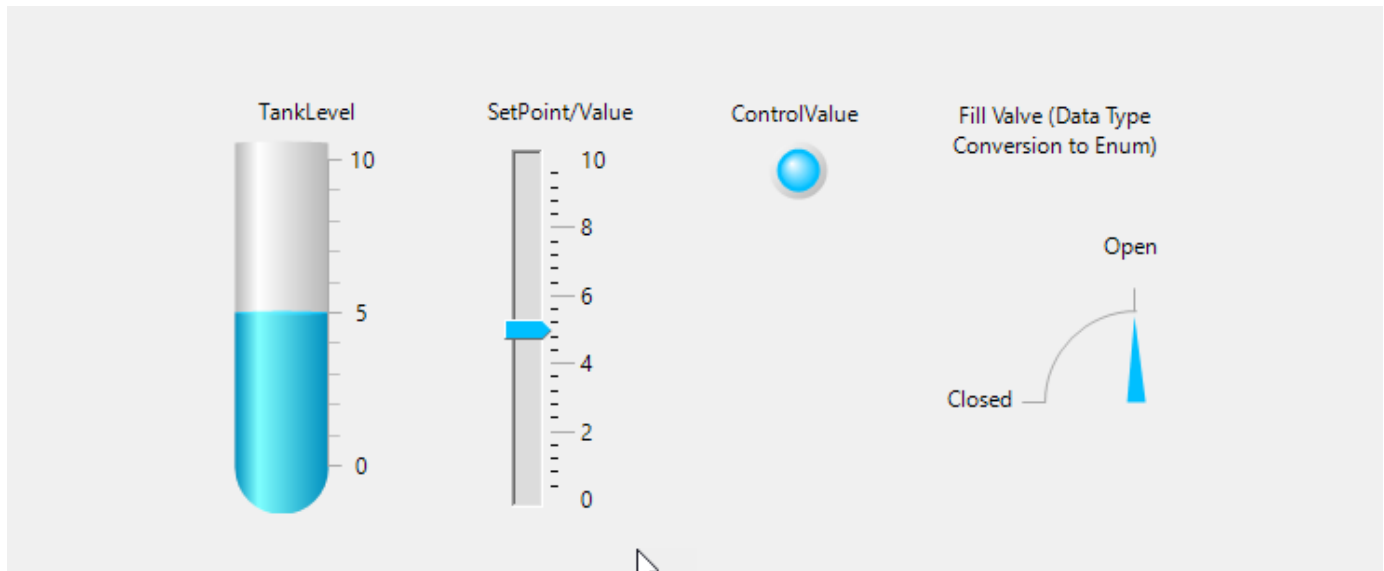
- A Target Scope block to display the output

The updated model appears as shown:



4. Build the xpctank model and download the real-time application to the target computer.
5. Open the xpctank\_instr\_design.slrtip instrument panel in slrtexplr, and open the parameter and signal panels for the xpctank model as described in **Configure Instruments for Signal Display**.
6. Add a GaugeAngular instrument to the instrument panel.
7. Bind the DataTypeConversionToEnum signal to the GaugeAngular instrument.
8. Run the model and run the instrument panel as described in **Run the Model and Instrument Panel**.

The GaugeAngular instrument displays the state of the fill valve as **Open** or **Closed**.



### See Also

GaugeFluidLevel | LED Properties | Slider

### More About


- “Get Started with Simulink Real-Time”
- “Instrumentation for Real-Time Applications” on page 4-2
- “Display and Filter Hierarchical Signals and Parameters”
- “View Signal Waveforms with Scope Instrument” on page 4-15
- “View Signal Waveforms with Signal Triggered Scope Instrument” on page 4-21
- “Save and Restore Layouts” on page 4-37
- “Save and Load Instrument Panels” on page 4-36

## Save Environment Properties

The Simulink Real-Time Explorer environment consists of the property settings you define for the **Targets** pane. You can save your settings for the next session.

- 1 Set properties in the **Targets** pane.

After you change one or more properties and press **Enter**, the **Save** button and menu item are available.

- 2 To save your environment properties, click the **Save** button  in the toolbar.

If you do not explicitly save the environment settings, Simulink Real-Time Explorer asks on exit if you want to save them.

When starting, Simulink Real-Time Explorer loads the last-saved set of environment properties.

### Save and Load Instrument Panels

As you are developing instrument panels to control your model, you can save your panels or load existing panels. You can load more than one instrument panel at a time.

- To save your instrument panel, click in the **Panels** pane, and then click the **Save** button .
- To load an existing instrument panel, in the **Panels** pane, right-click the **Instrument Panels** node and select **Existing**.

## Save and Restore Layouts

As you are configuring Simulink Real-Time session layouts, you can save your layout or restore a previous layout. Saving a layout preserves the following information:

- The position of the open panes and tabs.
- The target computer connections.
- The instrument panels that you loaded.
- The signal and parameter groups that you loaded.

Saving a layout saves the **Scopes** pane position, but it does not save the state of the scopes in the pane. In particular, if you add a scope within a Simulink Real-Time session, the software does not restore the new scope with the rest of the layout.

- To save a session layout, click **File > Save Layout**.
- To restore a session layout, click **File > Restore Layout**.

## Prepare Explorer Environment for Export

Check that each combination of a candidate computer that is compatible with Windows and a target computer works together. Each target computer must run in standalone mode.

The example uses the instrumented `xpctank` model. Before carrying out this procedure, you must have performed the steps in “Real-Time Application Instruments with Simulink Real-Time Explorer”.

For each computer on which you intend to run the standalone Simulink Real-Time Explorer executable:

- 1 Check that the candidate computer is compatible with 64-bit Windows.
- 2 Check that the CPU and operating system meet the requirements for executing the standalone Simulink Real-Time Explorer executable.
- 3 Check that Microsoft .NET Framework 4.5 is installed on the candidate computer.

For each target computer on which you intend to run the real-time application:

- 1 Check that the target computer CPU and operating system meet the requirements for running the Simulink Real-Time kernel.
- 2 Check that the Simulink Real-Time Explorer **Targets** pane contains a target computer node representing each target computer that you intend to access.

If you rename a target computer node, make the corresponding change in the **Bindings > TargetName** property for each instrument.

- 3 Check that the settings in the **Host-to-Target communication** tab match the requirements of the target computer.

You can have only one target computer node for each unique **IP address** setting.

- 4 Check that the settings in the **Target settings** tab match the capabilities of the target computer.
- 5 Prepare and copy the required kernel and real-time application files to the target computer. In the **Boot configuration** tab, set **Boot mode** to Stand Alone.
- 6 Connect the target computer to the candidate computer and restart the target computer. Check that the target computer loads the Simulink Real-Time kernel and starts the real-time application.

### See Also

#### More About

- “Development Computer Requirements”
- “PCI Bus Ethernet Setup”
- “Target Computer Settings”
- “Standalone Boot Method”
- “Explorer Configuration Exported to Run Outside MATLAB” on page 4-13



## Prepare Instrument Panel Configuration for Export

Load the instrument panels for the instrumented model. Resize and lay out Simulink Real-Time Explorer.

The example uses the instrumented `xpctank` model.

---

**Note** When you run the standalone executable, you cannot access the model hierarchy. You can access only instrument panels and windows that were open when you exported the configuration. You can access only signals and parameters that were loaded in signal and parameter groups when you exported the configuration.

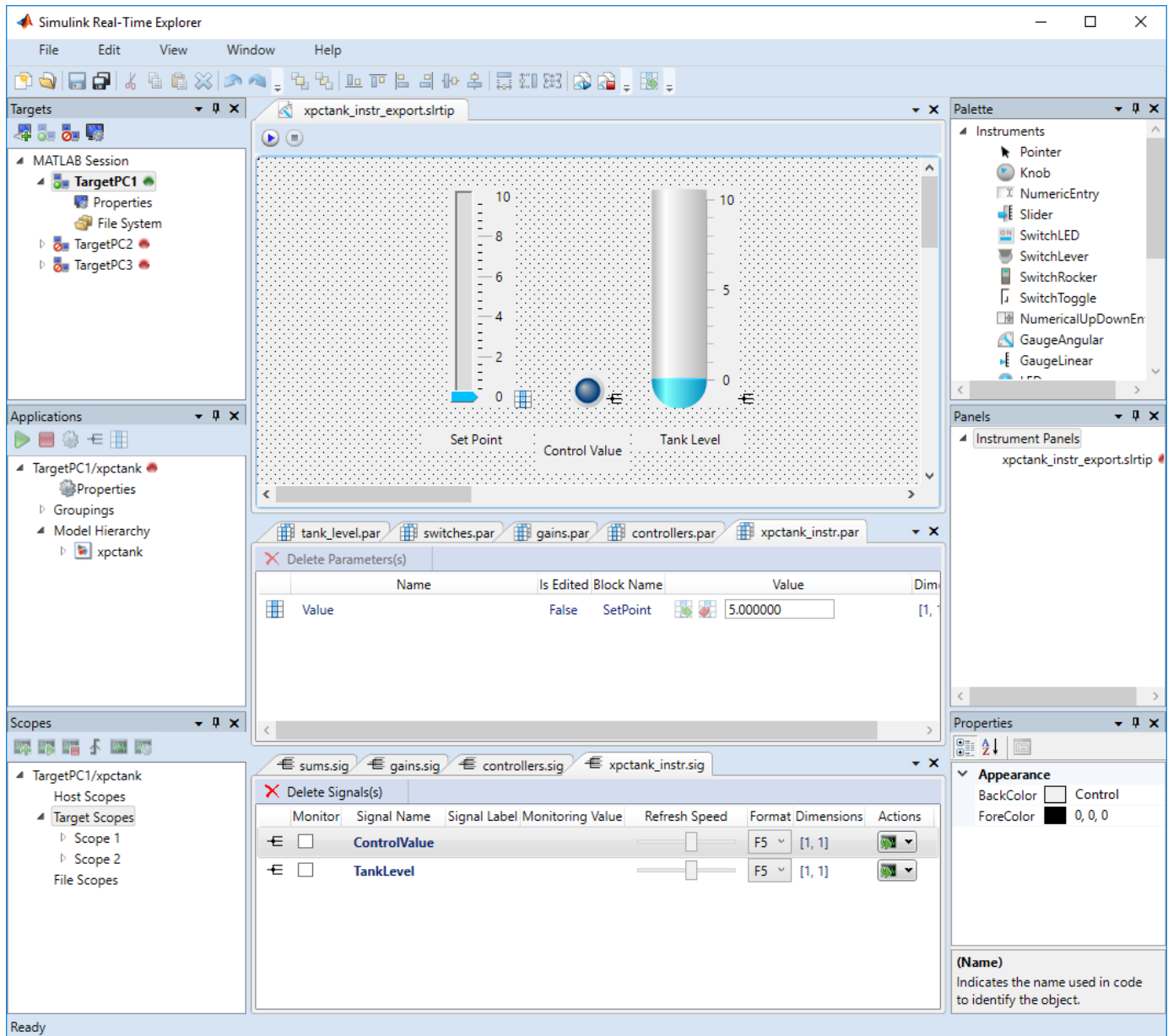
---

- 1 Load your instrument panels into Simulink Real-Time Explorer.

Here, the panel is `xpctank_instr_design.slrtip`.

- 2 Create a parameter group for the key block representing the set point (`xpctank_instr.par`).
- 3 Create parameter groups for the low-level blocks representing the tank level, switches, gains, and controllers (`tank_level.par`, `switches.par`, `gains.par`, and `controllers.par`).
- 4 Create a signal group for the key blocks representing the control value and tank level (`xpctank_instr.sig`).
- 5 Create signal groups for the low-level blocks representing the sums, gains, and controllers (`sums.sig`, `gains.sig`, and `controllers.sig`).
- 6 Open, lay out, and resize the windows that you want the standalone executable to open.
- 7 Save each instrument panel.

The `xpctank_instr_design.slrtip` configuration looks like the figure.



The next task is “Export Explorer Configuration” on page 4-41.

### See Also

### More About

- “Create Parameter Groups with Simulink Real-Time Explorer”
- “Create Signal Groups with Simulink Real-Time Explorer”
- “Instrumentation for Real-Time Applications” on page 4-2

## Export Explorer Configuration

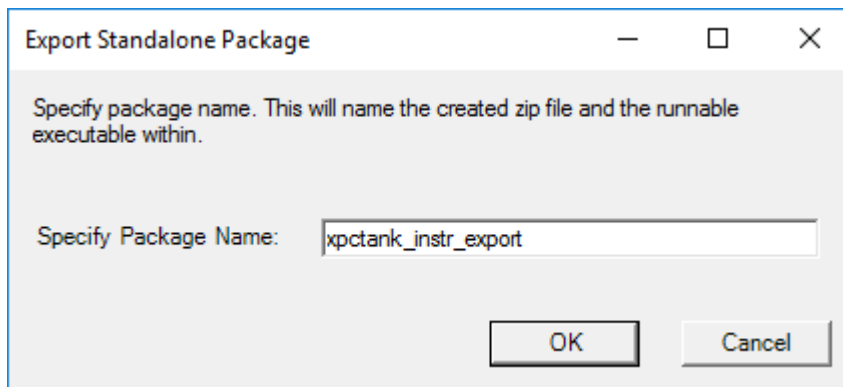
The example uses the instrumented `xpctank` model. Before carrying out this procedure, you must have performed the steps in “Prepare Instrument Panel Configuration for Export” on page 4-39.

---

**Note** When you run the standalone executable, you cannot access the model hierarchy. You can access only instrument panels and windows that were open when you exported the configuration. You can access only signals and parameters that were loaded in signal and parameter groups when you exported the configuration.

---

- 1 To export the configuration as a standalone executable, click **File > Export**.
- 2 In the **Specify Package Name** text box, type `xpctank_instr_export`.



- 3 Click **OK**.

The software generates a file named `xpctank_instr_export.zip` in the current folder.

The next task is “Unpack and Run Standalone Configuration” on page 4-42.

### See Also

### More About

- “Instrumentation for Real-Time Applications” on page 4-2

## Unpack and Run Standalone Configuration

Unpack the standalone executable onto a computer that is compatible with Windows.

The example uses the instrumented `xpctank` model. Before carrying out this procedure, you must have performed the steps in “Export Explorer Configuration” on page 4-41.

---

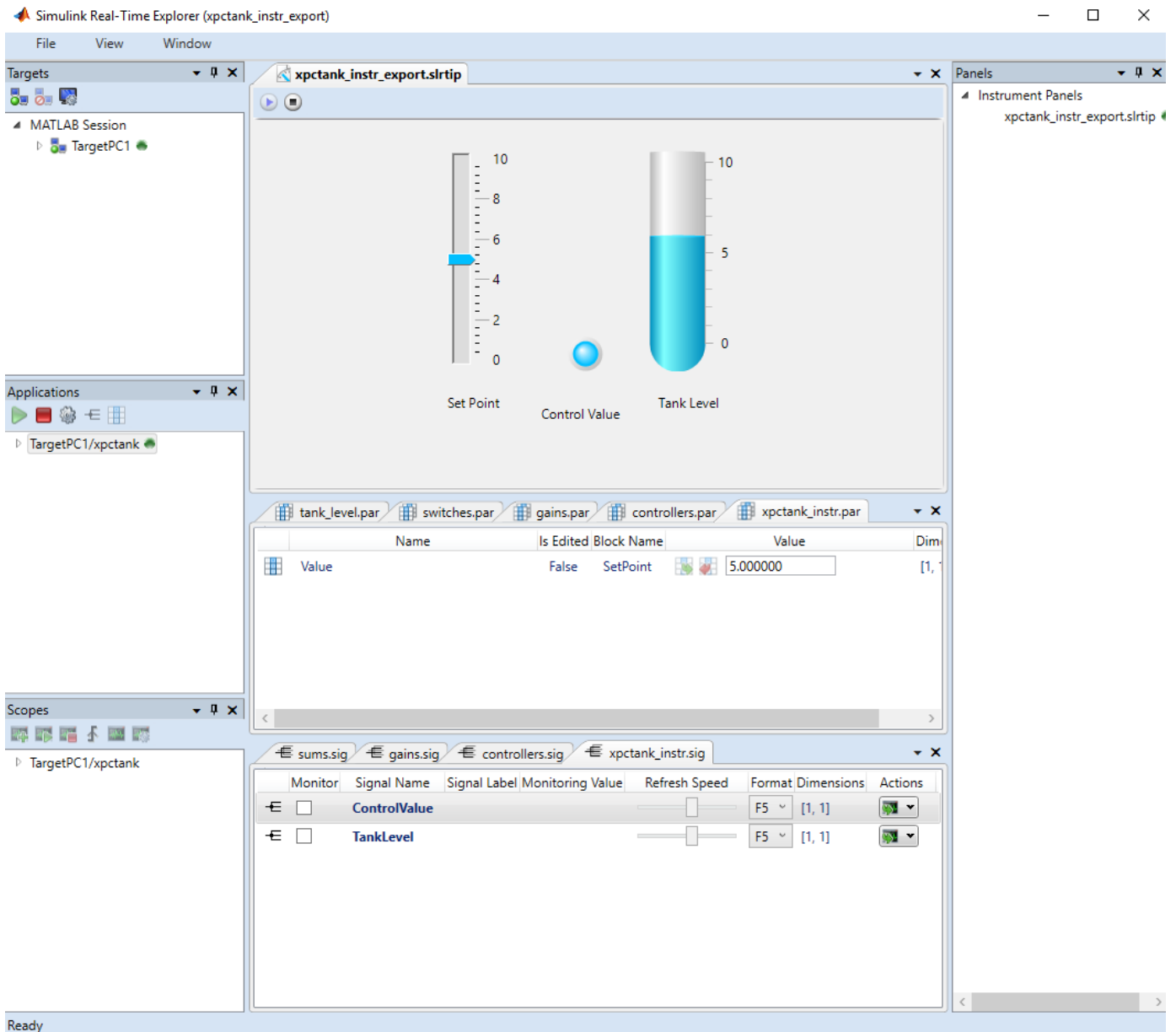
**Note** When you run the standalone executable, you cannot access the model hierarchy. You can access only instrument panels and windows that were open when you exported the configuration. You can access only signals and parameters that were loaded in signal and parameter groups when you exported the configuration.

---

- 1** Copy `xpctank_instr_export.zip` from the original folder to a folder on the computer compatible with Windows, for example `C:\workdir`.
- 2** Double-click `xpctank_instr_export.zip`.
- 3** In the unzip program dialog box, click **Extract**.
- 4** Select the extraction root folder, and then click **Extract**.
- 5** Navigate to folder `xpctank_instr_export` in the extraction root folder.
- 6** Connect the target computer to the computer that is compatible with Windows. Restart the target computer.

Check that the target computer loads the Simulink Real-Time kernel and the real-time application.

- 7** Double-click `runxpctank_instr_export.exe`.



To interact with the real-time application on the target computer, use the executable interface.

- If a signal is accessible, you can add a scope and attach the signal to the scope. Scopes that you add and remove do not change the model.
- If you remove a window, you can restore it by clicking **File > Restore Original View**.

## See Also

## More About

- “Instrumentation for Real-Time Applications” on page 4-2



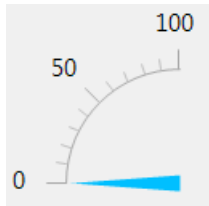
# **Simulink Real-Time Explorer Instruments**

## GaugeAngular Properties

Graphic instrument to display signal values

### Description


Use the GaugeAngular instrument to display real-valued data suitable for an angular gauge, such as pressure, speed, and current. When you bind a GaugeAngular instrument to a signal with enum type data values, the enumerated values appear as units on the instrument scale. The enumerated values are visible when you run the instrument panel. For more information, see “Add Instrument Panel to Tank Model” on page 4-25.





---

**Note** Do not use this graphic instrument for instrument panel development. Support for this instrument relies on support for the .NET API, which will be removed in a future release. For alternatives to instrument panels in Simulink Real-Time Explorer, see “Instrumentation Apps for Real-Time Applications” on page 6-2.

---

To access a dialog box containing key instrument or layout element properties, select the item. In the top, right corner, click the **Tasks** button .

To access the complete **Properties** list for an instrument or layout element, in the instrument panel, select the item. In the **Properties** list, to access a dialog box for a particular property group, click the group. To the right of the group, click the continuation button .

These properties represent a selection from the available properties. For more properties, see “Graphical Properties” on page 4-9.

### Properties

#### Instrument

#### ScaleDisplay — Subset of scale display properties

property group

Open a dialog box to configure the scale display.

#### Auto — Automatic generator display properties

property group

To set properties of the automatic display generator, under the **Generator** tab, click the **Auto** tab.



**Mid Included — Insert tick midway between major ticks**

False (default) | True

If **MinorCount** is even, space the minor ticks equally around the center tick. If **MinorCount** is odd, replace the center tick with the middle tick.

**Fixed Min/Max Majors — Constrain top and bottom tick values**

False (default) | True

If True, the top and bottom ticks are constrained as major ticks with min/max values defined by **Min** and **Span**.

**Minor Count — Number of minor ticks between major ticks**

4 (default) | numeric

**Min Text Spacing — Minimum space between scale ticks**

1 (default) | numeric

**Desired Increment — Display of major tick labels**

0 (default) | numeric

The number of labels is span / (increment + 1). If the required labels do not fit in the space available in the graphic, this setting has no effect.

**Text Formatting — Text display properties**

property group

To set text formatting properties, click the **Text Formatting** tab.

**Style — Style of tick display**

Number (default) | Thousands | Prefix | Exponent | Price32nds | DateTime | DateTimeUTC

**Precision Style — Style of expressing precision**

FixedDecimalPoints (default) | SignificantDigits | None

**Precision — Number of digits to the right of the decimal point**

0 (default) | numeric

**Units Text — Text to add to number**

text

Open a text editor and construct text to add to number. By default, the units text appears to the right of the number. If **Style** is Prefix, the text appears to the left of the number.

**ScaleRange — Subset of scale range properties**

property group

Specify range of control scale. The properties **Max** and **AngleMax** are calculated from **Min**, **Span**, **AngleMin**, and **AngleSpan**.

**Min — Minimum possible value**

0 (default) | numeric

**Span — Number of values between the minimum and maximum values**

100 (default) | numeric

**Reverse — Reverse scale direction**

False (default) | True

By default, the instrument displays values that increase in the customary directions: Clockwise, left to right, or down to up, depending on the instrument. If value is True, reverse the scale so that the values increase in the opposite directions: Counterclockwise, right to left, or up to down.

**Scale Type — Scale type: linear, logarithmic, or split**

Linear (default) | Log10 | SplitLinearLog10

Scale type, one of:

- **Linear** — Monotonic increasing or decreasing
- **Log10** — Log base 10 increasing or decreasing
- **SplitLinearLog10** — Monotonic increasing or decreasing in one part, log base 10 in the other part

**Angle Min — Starting point of scale, in degrees from bottom of circle**

180 (default) | numeric

**Angle Span — Number of degrees from minimum to full deflection**

90 (default) | numeric

**Design****Locked — Lock control against design-mode changes**

False (default) | True

When True, prevents accidental changes in size or location of the instrument instance.

**Bindings****BindingSrcPath — Name of signal or parameter**

text

Specify the hierarchical name of the signal or parameter that is bound to the instrument.

**BindingSrcType — Type of object to which the instrument is bound**

None (default) | Parameter | Signal

Specify the type of object to which the instrument is bound. The Signal binding is visible only for display instruments.

**TargetName — Name of target computer to which the control is connected**

text

Bind an instrument instance to a particular target computer. You can bind different instruments in the same instrument panel to different target computers.

**See Also****Topics**

“Instrumentation for Real-Time Applications” on page 4-2

“Add Instrument Panel to Tank Model” on page 4-25

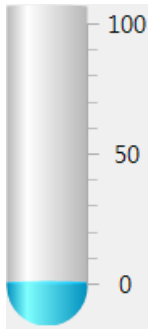
**Introduced in R2014a**

## GaugeFluidLevel Properties

Graphic instrument to display values of fluid sensor signals

### Description


Use the GaugeFluidLevel instrument to display real-valued data suitable for a fluid gauge, such as volume and pressure. When you bind a GaugeFluidLevel instrument to a signal with enum type data values, the enumerated values appear as units on the instrument scale. The enumerated values are visible when you run the instrument panel. For more information, see “Add Instrument Panel to Tank Model” on page 4-25.





---

**Note** Do not use this graphic instrument for instrument panel development. Support for this instrument relies on support for the .NET API, which will be removed in a future release. For alternatives to instrument panels in Simulink Real-Time Explorer, see “Instrumentation Apps for Real-Time Applications” on page 6-2.

---

To access a dialog box containing key instrument or layout element properties, select the item. In the top, right corner, click the **Tasks** button .

To access the complete **Properties** list for an instrument or layout element, in the instrument panel, select the item. In the **Properties** list, to access a dialog box for a particular property group, click the group. To the right of the group, click the continuation button .

These properties represent a selection from the available properties. For more properties, see “Graphical Properties” on page 4-9.

### Properties

#### Instrument

#### ScaleDisplay — Subset of scale display properties

property group

Open a dialog box to configure the scale display.

**Auto — Automatic generator display properties**

property group

To set properties of the automatic display generator, under the **Generator** tab, click the **Auto** tab.

**Mid Included — Insert tick midway between major ticks**

False (default) | True

If **MinorCount** is even, space the minor ticks equally around the center tick. If **MinorCount** is odd, replace the center tick with the middle tick.

**Fixed Min/Max Majors — Constrain top and bottom tick values**

False (default) | True

If True, the top and bottom ticks are constrained to be major ticks with min/max values defined by **Min** and **Span**.

**Minor Count — Number of minor ticks between major ticks**

4 (default) | numeric

**Min Text Spacing — Minimum space between scale ticks**

1 (default) | numeric

**Desired Increment — Display of major tick labels**

0 (default) | numeric

The number of labels is  $\text{span} / (\text{desired increment} + 1)$ . This setting has no effect if the required labels do not fit in the space available in the graphic.

**Text Formatting — Text display properties**

property group

To set text formatting properties, click the **Text Formatting** tab.

**Style — Style of tick display**

Number (default) | Thousands | Prefix | Exponent | Price32nds | DateTime | DateTimeUTC

**Precision Style — Style of expressing precision**

FixedDecimalPoints (default) | SignificantDigits | None

**Precision — Number of digits to the right of the decimal point**

0 (default) | numeric

**Units Text — Text to add to number**

text

Open a text editor and construct text to add to number. By default, the units text appears to the right of the number. If **Style** is **Prefix**, the text appears to the left of the number.

**ScaleRange — Subset of scale range properties**

property group

Specify range of control scale. The property **Max** is calculated from **Min** and **Span**.

**Min — Minimum possible value**

0 (default) | numeric

**Span — Number of values between the minimum and maximum values**

100 (default) | numeric

**Reverse — Reverse scale direction**

False (default) | True

By default, the instrument displays values that increase in the customary directions: Clockwise, left to right, or down to up, depending on the instrument. If value is True, reverse the scale so that the values increase in the opposite directions: Counterclockwise, right to left, or up to down.

**Scale Type — Scale type: linear, logarithmic, or split**

Linear (default) | Log10 | SplitLinearLog10

Scale type, one of:

- **Linear** — Monotonic increasing or decreasing
- **Log10** — Log base 10 increasing or decreasing
- **SplitLinearLog10** — Monotonic increasing or decreasing in one part, log base 10 in the other part

**Tube — Subset of tube display properties**

property group

Open a dialog box to configure the tube display.

**Width — Width of tube graphic**

50 (default) | integer

Horizontal dimension of tube graphic, in pixels

**Wall Thickness — Thickness of tube wall**

0 (default) | integer

Edge thickness of tube wall, in pixels

**Fill Color — Color of active portion of tube**

DeepSkyBlue (default) | struct

Open a dialog box with these tabs:

**Custom — Customizable color grid**

struct

**Web — List of web colors**

struct

**System — List of system colors**

struct

**Wall Color — Color of inactive portion of tube**

White Smoke (default) | struct

Open a dialog box with these tabs:

**Custom — Customizable color grid**

struct

**Web — List of web colors**

struct

**System — List of system colors**

struct

**Alignment — Alignment of tube graphic**

AutoCenter (default) | Side | Center

Horizontal position of the tube within the frame.

**Design****Locked — Lock control against design-mode changes**

False (default) | True

When True, prevents accidental changes in size or location of the instrument instance.

**Bindings****BindingSrcPath — Name of signal or parameter**

text

Specify the hierarchical name of the signal or parameter that is bound to the instrument.

**BindingSrcType — Type of object to which the instrument is bound**

None (default) | Parameter | Signal

Specify the type of object to which the instrument is bound. The Signal binding is visible only for display instruments.

**TargetName — Name of target computer to which the control is connected**

text

Bind an instrument instance to a particular target computer. You can bind different instruments in the same instrument panel to different target computers.

**See Also**

“Add Instrument Panel to Tank Model” on page 4-25

**Topics**

“Instrumentation for Real-Time Applications” on page 4-2

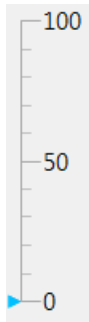
**Introduced in R2014a**

## GaugeLinear Properties

Graphic instrument to display signal values

### Description


Use the GaugeLinear instrument to display real-valued data suitable for a linear gauge, such as temperature, volume, and pressure. When you bind a GaugeLinear instrument to a signal with enum type data values, the enumerated values appear as units on the instrument scale. The enumerated values are visible when you run the instrument panel. For more information, see “Add Instrument Panel to Tank Model” on page 4-25.





---

**Note** Do not use this graphic instrument for instrument panel development. Support for this instrument relies on support for the .NET API, which will be removed in a future release. For alternatives to instrument panels in Simulink Real-Time Explorer, see “Instrumentation Apps for Real-Time Applications” on page 6-2.

---

To access a dialog box containing key instrument or layout element properties, select the item. In the top, right corner, click the **Tasks** button .

To access the complete **Properties** list for an instrument or layout element, in the instrument panel, select the item. In the **Properties** list, to access a dialog box for a particular property group, click the group. To the right of the group, click the continuation button .

These properties represent a selection from the available properties. For more properties, see “Graphical Properties” on page 4-9.

### Properties

#### Instrument

#### ScaleDisplay — Subset of scale display properties

property group

Open a dialog box to configure the scale display.



**Auto — Automatic generator display properties**

property group

To set properties of the automatic display generator, under the **Generator** tab, click the **Auto** tab.

**Mid Included — Insert tick midway between major ticks**

False (default) | True

If **MinorCount** is even, space the minor ticks equally around the center tick. If **MinorCount** is odd, replace the center tick with the middle tick.

**Fixed Min/Max Majors — Constrain top and bottom tick values**

False (default) | True

If True, the top and bottom ticks are constrained to be major ticks with min/max values defined by **Min** and **Span**.

**Minor Count — Number of minor ticks between major ticks**

4 (default) | numeric

**Min Text Spacing — Minimum space between scale ticks**

1 (default) | numeric

**Desired Increment — Display of major tick labels**

0 (default) | numeric

The number of labels is  $\text{span} / (\text{desired increment} + 1)$ . This setting has no effect if the required labels do not fit in the space available in the graphic.

**Text Formatting — Text display properties**

property group

To set text formatting properties, click the **Text Formatting** tab.

**Style — Style of tick display**

Number (default) | Thousands | Prefix | Exponent | Price32nds | DateTime | DateTimeUTC

**Precision Style — Style of expressing precision**

FixedDecimalPoints (default) | SignificantDigits | None

**Precision — Number of digits to the right of the decimal point**

0 (default) | numeric

**Units Text — Text to add to number**

text

Open a text editor and construct text to add to number. By default, the units text appears to the right of the number. If **Style** is **Prefix**, the text appears to the left of the number.

**ScaleRange — Subset of scale range properties**

property group

Specify range of control scale. The property **Max** is calculated from **Min** and **Span**.

**Min — Minimum possible value**

0 (default) | numeric

**Span — Number of values between the minimum and maximum values**

100 (default) | numeric

**Reverse — Reverse scale direction**

False (default) | True

By default, the instrument displays values that increase in the customary directions: Clockwise, left to right, or down to up, depending on the instrument. If value is `True`, reverse the scale so that the values increase in the opposite directions: Counterclockwise, right to left, or up to down.

**Scale Type — Scale type: linear, logarithmic, or split**

Linear (default) | Log10 | SplitLinearLog10

Scale type, one of:

- `Linear` — Monotonic increasing or decreasing
- `Log10` — Log base 10 increasing or decreasing
- `SplitLinearLog10` — Monotonic increasing or decreasing in one part, log base 10 in the other part

**Design****Locked — Lock control against design-mode changes**

False (default) | True

When `True`, prevents accidental changes in size or location of the instrument instance.

**Bindings****BindingSrcPath — Name of signal or parameter**

text

Specify the hierarchical name of the signal or parameter that is bound to the instrument.

**BindingSrcType — Type of object to which the instrument is bound**

None (default) | Parameter | Signal

Specify the type of object to which the instrument is bound. The `Signal` binding is visible only for display instruments.

**TargetName — Name of target computer to which the control is connected**

text

Bind an instrument instance to a particular target computer. You can bind different instruments in the same instrument panel to different target computers.

**See Also****Topics**

“Instrumentation for Real-Time Applications” on page 4-2

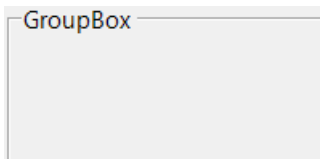
**Introduced in R2014a**

# GroupBox Properties

Nonscrollable graphical container for instruments

## Description


The **GroupBox** graphical layout element provides a container for other instruments. At design time, you can stretch and shrink the container.



---

**Note** Do not use this graphic instrument for instrument panel development. Support for this instrument relies on support for the .NET API, which will be removed in a future release. For alternatives to instrument panels in Simulink Real-Time Explorer, see “Instrumentation Apps for Real-Time Applications” on page 6-2.

---

To access the complete **Properties** list for an instrument or layout element, in the instrument panel, select the item. In the **Properties** list, to access a dialog box for a particular property group, click the group. To the right of the group, click the continuation button .

These properties represent a selection from the available properties. For more properties, see “Graphical Properties” on page 4-9.

## Properties

### Design

#### **Locked — Lock control against design-mode changes**

False (default) | True

When True, prevents accidental changes in size or location of the instrument instance.

## See Also

### Topics

“Instrumentation for Real-Time Applications” on page 4-2

**Introduced in R2014a**

# HexadecimalDisplay Properties

Instrument to display signal values

## Description


The **HexadecimalDisplay** instrument displays numeric data in hexadecimal format. This instrument is used for digital data, such as status codes and register contents.





---

**Note** Do not use this graphic instrument for instrument panel development. Support for this instrument relies on support for the .NET API, which will be removed in a future release. For alternatives to instrument panels in Simulink Real-Time Explorer, see “Instrumentation Apps for Real-Time Applications” on page 6-2.

---

To access a dialog box containing key instrument or layout element properties, select the item. In the top, right corner, click the **Tasks** button .

To access the complete **Properties** list for an instrument or layout element, in the instrument panel, select the item. In the **Properties** list, to access a dialog box for a particular property group, click the group. To the right of the group, click the continuation button .

These properties represent a selection from the available properties. For more properties, see “Graphical Properties” on page 4-9.

## Properties

### Instrument

#### **DigitCount** — Number of hex digits to display

6 (default) | numeric

Specify a number that is consistent with the width of the signal that the instrument displays.

#### **DigitLeadingStyle** — Fill digits to display

None (default) | Zeros

- None — Fill digits are blanks.
- Zeros — Fill digits are zeroes.

### Design

#### **Locked** — Lock control against design-mode changes

False (default) | True

When True, prevents accidental changes in size or location of the instrument instance.

## Bindings

### **BindingSrcPath — Name of signal or parameter**

text

Specify the hierarchical name of the signal or parameter that is bound to the instrument.

### **BindingSrcType — Type of object to which the instrument is bound**

None (default) | Parameter | Signal

Specify the type of object to which the instrument is bound. The Signal binding is visible only for display instruments.

### **TargetName — Name of target computer to which the control is connected**

text

Bind an instrument instance to a particular target computer. You can bind different instruments in the same instrument panel to different target computers.

## See Also

### **Topics**

“Instrumentation for Real-Time Applications” on page 4-2

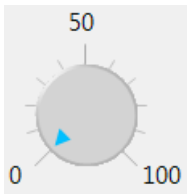
**Introduced in R2014a**

## Knob Properties

Graphic instrument to set parameter values

### Description

Use the **Knob** instrument to set real-valued data such as amplitude and frequency when an exact value is not required. When you bind a **Knob** instrument to a parameter with enum type data values, the enumerated values appear as units on the instrument scale. The enumerated values are visible when you run the instrument panel. For more information, see “Add Instrument Panel to Tank Model” on page 4-25.




---

**Note** Do not use this graphic instrument for instrument panel development. Support for this instrument relies on support for the .NET API, which will be removed in a future release. For alternatives to instrument panels in Simulink Real-Time Explorer, see “Instrumentation Apps for Real-Time Applications” on page 6-2.


---




---

**Note** Do not use this graphic instrument for instrument panel development. Support for this instrument relies on support for the .NET API, which will be removed in a future release. For alternatives to instrument panels in Simulink Real-Time Explorer, see “Instrumentation Apps for Real-Time Applications” on page 6-2.

---

To access a dialog box containing key instrument or layout element properties, select the item. In the top, right corner, click the **Tasks** button .

To access the complete **Properties** list for an instrument or layout element, in the instrument panel, select the item. In the **Properties** list, to access a dialog box for a particular property group, click the group. To the right of the group, click the continuation button .

These properties represent a selection from the available properties. For more properties, see “Graphical Properties” on page 4-9.

### Properties

#### Instrument

#### OffSwitch — Knob on/off switch configuration

parameter group

Open the parameter group list to set the off switch parameters.

**Enable — Use the knob to turn off parameter**

False (default) | True

To enable the knob to turn off the underlying parameter, set **Enabled** to True. A dot at the extreme low end of the range shows the status of the parameter.

**On — Initial state of the off switch**

False (default) | True

If True, the switch is initially turned on.

**ColorOn — Color of indicator when switch is on**

DarkSeaGreen (default) | struct

Open a dialog box with these tabs:

**Custom — Customizable color grid**

struct

**Web — List of web colors**

struct

**System — List of system colors**

struct

**ColorOff — Color of indicator when switch is off**

LightCoral (default) | struct

Open a dialog box with these tabs:

**Custom — Customizable color grid**

struct

**Web — List of web colors**

struct

**System — List of system colors**

struct

**ScaleDisplay — Subset of scale display properties**

property group

Open a dialog box to configure the scale display.

**Auto — Automatic generator display properties**

property group

To set properties of the automatic display generator, under the **Generator** tab, click the **Auto** tab.

**Mid Included — Insert tick midway between major ticks**

False (default) | True

If **MinorCount** is even, space the minor ticks equally around the center tick. If **MinorCount** is odd, replace the center tick with the middle tick.

**Fixed Min/Max Majors – Constrain top and bottom tick values**

False (default) | True

If True, the top and bottom ticks are constrained to be major ticks with min/max values defined by **Min** and **Span**.

**Minor Count – Number of minor ticks between major ticks**

4 (default) | numeric

**Min Text Spacing – Minimum space between scale ticks**

1 (default) | numeric

**Desired Increment – Display of major tick labels**

0 (default) | numeric

The number of labels is  $\text{span} / (\text{desired increment} + 1)$ . This setting does nothing if the required labels do not fit in the space available in the graphic.

**Text Formatting – Text display properties**

property group

To set text formatting properties, click the **Text Formatting** tab.

**Style – Style of tick display**

Number (default) | Thousands | Prefix | Exponent | Price32nds | DateTime | DateTimeUTC

**Precision Style – Style of expressing precision**

FixedDecimalPoints (default) | SignificantDigits | None

**Precision – Number of digits to the right of the decimal point**

0 (default) | numeric

**Units Text – Text to add to number**

text

Open a text editor and construct text to add to number. By default, the units text appears to the right of the number. If **Style** is **Prefix**, the text appears to the left of the number.

**ScaleRange – Subset of scale range properties**

property group

Specify range of control scale. The properties **Max** and **AngleMax** are calculated from **Min**, **Span**, **AngleMin**, and **AngleSpan**.

**Min – Minimum possible value**

0 (default) | numeric

**Span – Number of values between the minimum and maximum values**

100 (default) | numeric

**Reverse – Reverse scale direction**

False (default) | True

By default, the instrument displays values that increase in the customary directions: Clockwise, left to right, or down to up, depending on the instrument. If value is True, reverse the scale so that the values increase in the opposite directions: Counterclockwise, right to left, or up to down.



**Scale Type — Scale type: linear, logarithmic, or split**

Linear (default) | Log10 | SplitLinearLog10

Scale type, one of:

- **Linear** — Monotonic increasing or decreasing
- **Log10** — Log base 10 increasing or decreasing
- **SplitLinearLog10** — Monotonic increasing or decreasing in one part, log base 10 in the other part

**Angle Min — Starting point of scale, in degrees from bottom of circle**

180 (default) | numeric

**Angle Span — Number of degrees from minimum to full deflection**

90 (default) | numeric

**Design****Locked — Lock control against design-mode changes**

False (default) | True

When True, prevents accidental changes in size or location of the instrument instance.

**Bindings****BindingSrcPath — Name of signal or parameter**

text

Specify the hierarchical name of the signal or parameter that is bound to the instrument.

**BindingSrcType — Type of object to which the instrument is bound**

None (default) | Parameter | Signal

Specify the type of object to which the instrument is bound. The Signal binding is visible only for display instruments.

**TargetName — Name of target computer to which the control is connected**

text

Bind an instrument instance to a particular target computer. You can bind different instruments in the same instrument panel to different target computers.

**See Also****Topics**

“Instrumentation for Real-Time Applications” on page 4-2

**Introduced in R2014a**

## Label Properties

Graphical container for text

### Description


Use the **Label** graphical layout element to add text to the instrument layout.

Label

---

**Note** Do not use this graphic instrument for instrument panel development. Support for this instrument relies on support for the .NET API, which will be removed in a future release. For alternatives to instrument panels in Simulink Real-Time Explorer, see “Instrumentation Apps for Real-Time Applications” on page 6-2.

---

To access the complete **Properties** list for an instrument or layout element, in the instrument panel, select the item. In the **Properties** list, to access a dialog box for a particular property group, click the group. To the right of the group, click the continuation button .

These properties represent a selection from the available properties. For more properties, see “Graphical Properties” on page 4-9.

### Properties

#### Appearance

#### **Text** — Defines text displayed by label

character vector | string scalar

Type the label display text into the **Text** box.

#### **TextAlign** — Defines alignment of text displayed by the label

TopLeft (default) | TopCenter | TopRight | MiddleLeft | MiddleCenter | MiddleRight | BottomLeft | BottomCenter | BottomRight

Specifies left-to-right and top-to-bottom alignment with a 3x3 matrix.

This display represents setting TopLeft.



**Design****Locked — Lock control against design-mode changes**

False (default) | True

When True, prevents accidental changes in size or location of the instrument instance.

**See Also****Topics**

“Instrumentation for Real-Time Applications” on page 4-2

“Add Instrument Panel to Tank Model” on page 4-25

“Triggering Scope Instruments”

**Introduced in R2014a**

## LED Properties

Graphic instrument to display binary signal values

### Description


Use the **LED** instrument to display binary (1 or 0) data.





---

**Note** Do not use this graphic instrument for instrument panel development. Support for this instrument relies on support for the .NET API, which will be removed in a future release. For alternatives to instrument panels in Simulink Real-Time Explorer, see “Instrumentation Apps for Real-Time Applications” on page 6-2.

---

To access a dialog box containing key instrument or layout element properties, select the item. In the top, right corner, click the **Tasks** button .

To access the complete **Properties** list for an instrument or layout element, in the instrument panel, select the item. In the **Properties** list, to access a dialog box for a particular property group, click the group. To the right of the group, click the continuation button .

These properties represent a selection from the available properties. For more properties, see “Graphical Properties” on page 4-9.

### Properties

#### Instrument

#### **BlinkerEnable** — Cause LED to blink continuously

False (default) | True

To cause the LED to blink continuously, set to True.

#### **Indicator** — Subset of indicator display properties

property group

Open a dialog box to configure the indicator display.

#### **Active Color** — Color of indicator when signal is 1

DeepSkyBlue (default) | struct

Open a dialog box with these tabs:

#### **Custom** — Customizable color grid

struct

**Web — List of web colors**

struct

**System — List of system colors**

struct

**Inactive Color — Color of indicator when signal is 0**

[41, 41, 41] (default) | struct

Open a dialog box with these tabs:

**Custom — Customizable color grid**

struct

**Web — List of web colors**

struct

**System — List of system colors**

struct

**Design****Locked — Lock control against design-mode changes**

False (default) | True

When True, prevents accidental changes in size or location of the instrument instance.

**Bindings****BindingSrcPath — Name of signal or parameter**

text

Specify the hierarchical name of the signal or parameter that is bound to the instrument.

**BindingSrcType — Type of object to which the instrument is bound**

None (default) | Parameter | Signal

Specify the type of object to which the instrument is bound. The Signal binding is visible only for display instruments.

**TargetName — Name of target computer to which the control is connected**

text

Bind an instrument instance to a particular target computer. You can bind different instruments in the same instrument panel to different target computers.

**See Also****Topics**

“Instrumentation for Real-Time Applications” on page 4-2

**Introduced in R2014a**

# NumericDisplay Properties

Instrument to display signal values

## Description


Use the **NumericDisplay** instrument to display real-valued numeric data.





---

**Note** Do not use this graphic instrument for instrument panel development. Support for this instrument relies on support for the .NET API, which will be removed in a future release. For alternatives to instrument panels in Simulink Real-Time Explorer, see “Instrumentation Apps for Real-Time Applications” on page 6-2.

---

To access a dialog box containing key instrument or layout element properties, select the item. In the top, right corner, click the **Tasks** button .

To access the complete **Properties** list for an instrument or layout element, in the instrument panel, select the item. In the **Properties** list, to access a dialog box for a particular property group, click the group. To the right of the group, click the continuation button .

These properties represent a selection from the available properties. For more properties, see “Graphical Properties” on page 4-9.

## Properties

### Instrument

#### Text Formatting — Text display properties

property group

To set text formatting properties, click the **Text Formatting** tab.

#### Style — Style of tick display

Number (default) | Thousands | Prefix | Exponent | Price32nds | DateTime | DateTimeUTC

#### Precision Style — Style of expressing precision

FixedDecimalPoints (default) | SignificantDigits | None

#### Precision — Number of digits to the right of the decimal point

0 (default) | numeric

#### Units Text — Text to add to number

text

Open a text editor and construct text to add to number. By default, the units text appears to the right of the number. If **Style** is **Prefix**, the text appears to the left of the number.

## Design

### **Locked — Lock control against design-mode changes**

False (default) | True

When True, prevents accidental changes in size or location of the instrument instance.

## Bindings

### **BindingSrcPath — Name of signal or parameter**

text

Specify the hierarchical name of the signal or parameter that is bound to the instrument.

### **BindingSrcType — Type of object to which the instrument is bound**

None (default) | Parameter | Signal

Specify the type of object to which the instrument is bound. The Signal binding is visible only for display instruments.

### **TargetName — Name of target computer to which the control is connected**

text

Bind an instrument instance to a particular target computer. You can bind different instruments in the same instrument panel to different target computers.

## See Also

### Topics

“Instrumentation for Real-Time Applications” on page 4-2

### Introduced in R2014a

## NumericEntry Properties

Instrument to set parameter values

### Description


Use the **NumericEntry** instrument to enter real-valued data in specified formats when an exact value is required.





---

**Note** Do not use this graphic instrument for instrument panel development. Support for this instrument relies on support for the .NET API, which will be removed in a future release. For alternatives to instrument panels in Simulink Real-Time Explorer, see “Instrumentation Apps for Real-Time Applications” on page 6-2.

---

To access a dialog box containing key instrument or layout element properties, select the item. In the top, right corner, click the **Tasks** button .

To access the complete **Properties** list for an instrument or layout element, in the instrument panel, select the item. In the **Properties** list, to access a dialog box for a particular property group, click the group. To the right of the group, click the continuation button .

These properties represent a selection from the available properties. For more properties, see “Graphical Properties” on page 4-9.

### Properties

#### Instrument

#### Text Formatting — Text display properties

property group

To set text formatting properties, click the **Text Formatting** tab.

#### Style — Style of tick display

Number (default) | Thousands | Prefix | Exponent | Price32nds | DateTime | DateTimeUTC

#### Precision Style — Style of expressing precision

FixedDecimalPoints (default) | SignificantDigits | None

#### Precision — Number of digits to the right of the decimal point

0 (default) | numeric

#### Units Text — Text to add to number

text

Open a text editor and construct text to add to number. By default, the units text appears to the right of the number. If **Style** is **Prefix**, the text appears to the left of the number.



## Design

### **Locked — Lock control against design-mode changes**

False (default) | True

When True, prevents accidental changes in size or location of the instrument instance.

## Bindings

### **BindingSrcPath — Name of signal or parameter**

text

Specify the hierarchical name of the signal or parameter that is bound to the instrument.

### **BindingSrcType — Type of object to which the instrument is bound**

None (default) | Parameter | Signal

Specify the type of object to which the instrument is bound. The Signal binding is visible only for display instruments.

### **TargetName — Name of target computer to which the control is connected**

text

Bind an instrument instance to a particular target computer. You can bind different instruments in the same instrument panel to different target computers.

## See Also

### Topics

“Instrumentation for Real-Time Applications” on page 4-2

### Introduced in R2014a

## NumericUpDownEntry Properties

Instrument to set parameter values

### Description


Use the **NumericUpDownEntry** instrument to enter real-valued data and increment it by a specified amount when a step change is required.





---

**Note** Do not use this graphic instrument for instrument panel development. Support for this instrument relies on support for the .NET API, which will be removed in a future release. For alternatives to instrument panels in Simulink Real-Time Explorer, see “Instrumentation Apps for Real-Time Applications” on page 6-2.

---

To access a dialog box containing key instrument or layout element properties, select the item. In the top, right corner, click the **Tasks** button .

To access the complete **Properties** list for an instrument or layout element, in the instrument panel, select the item. In the **Properties** list, to access a dialog box for a particular property group, click the group. To the right of the group, click the continuation button .

These properties represent a selection from the available properties. For more properties, see “Graphical Properties” on page 4-9.

### Properties

#### Data

#### **DecimalPlaces — Number of decimal places to display**

0 (default) | integer

The default display is an integer (0 decimal places).

#### **Increment — Value change per arrow click**

1 (default) | numeric

Value to add or subtract in response to an up-arrow or down-arrow.

#### **Maximum — Maximum data value**

100 (default) | numeric

Values greater than the maximum are set to the maximum.

#### **Minimum — Minimum data value**

0 (default) | numeric

Values less than the minimum are set to the minimum.

## Design

### **Locked — Lock control against design-mode changes**

False (default) | True

When True, prevents accidental changes in size or location of the instrument instance.

## Bindings

### **BindingSrcPath — Name of signal or parameter**

text

Specify the hierarchical name of the signal or parameter that is bound to the instrument.

### **BindingSrcType — Type of object to which the instrument is bound**

None (default) | Parameter | Signal

Specify the type of object to which the instrument is bound. The Signal binding is visible only for display instruments.

### **TargetName — Name of target computer to which the control is connected**

text

Bind an instrument instance to a particular target computer. You can bind different instruments in the same instrument panel to different target computers.

## See Also

### Topics

“Instrumentation for Real-Time Applications” on page 4-2

### Introduced in R2014a

## Panel Properties

Scrollable graphical container for instruments

### Description


The **Panel** graphical layout element provides a container for other instruments. At design time, you can stretch and shrink the container.





---

**Note** Do not use this graphic instrument for instrument panel development. Support for this instrument relies on support for the .NET API, which will be removed in a future release. For alternatives to instrument panels in Simulink Real-Time Explorer, see “Instrumentation Apps for Real-Time Applications” on page 6-2.

---

To access a dialog box containing key instrument or layout element properties, select the item. In the top, right corner, click the **Tasks** button .

To access the complete **Properties** list for an instrument or layout element, in the instrument panel, select the item. In the **Properties** list, to access a dialog box for a particular property group, click the group. To the right of the group, click the continuation button .

These properties represent a selection from the available properties. For more properties, see “Graphical Properties” on page 4-9.

### Properties

#### Design

##### **Locked — Lock control against design-mode changes**

False (default) | True

When True, prevents accidental changes in size or location of the instrument instance.

### See Also

#### Topics

“Instrumentation for Real-Time Applications” on page 4-2

#### Introduced in R2014a

# PictureBox Properties

Graphical display for pictures

## Description


The **PictureBox** graphical display element presents graphics in bitmap, GIF, JPEG, and PNG format. At design time, you can stretch and shrink the display.





---

**Note** Do not use this graphic instrument for instrument panel development. Support for this instrument relies on support for the .NET API, which will be removed in a future release. For alternatives to instrument panels in Simulink Real-Time Explorer, see “Instrumentation Apps for Real-Time Applications” on page 6-2.

---

To access a dialog box containing key instrument or layout element properties, select the item. In the top, right corner, click the **Tasks** button .

To access the complete **Properties** list for an instrument or layout element, in the instrument panel, select the item. In the **Properties** list, to access a dialog box for a particular property group, click the group. To the right of the group, click the continuation button .

These properties represent a selection from the available properties. For more properties, see “Graphical Properties” on page 4-9.

## Properties

### Asynchronous

#### ImageLocation — Path name of image

character vector | string scalar

Enter the full or relative path to the image.

### Behavior

#### SizeMode — Policy for sizing image relative to instrument

Normal (default) | StretchImage | AutoSize | CenterImage | Zoom

- **Normal** — Image full size, can be cut off by edges of instrument
- **StretchImage** — Image stretched in the x and y directions, matching instrument aspect ratio
- **AutoSize** — Image full size, instrument changes dimensions to match the image
- **CenterImage** — Image full size, centered in instrument, can be cut off by edges of instrument
- **Zoom** — Image stretched in the x and y directions, preserving image aspect ratio

## **Design**

### **Locked — Lock control against design-mode changes**

False (default) | True

When True, prevents accidental changes in size or location of the instrument instance.

## **See Also**

### **Topics**

“Instrumentation for Real-Time Applications” on page 4-2

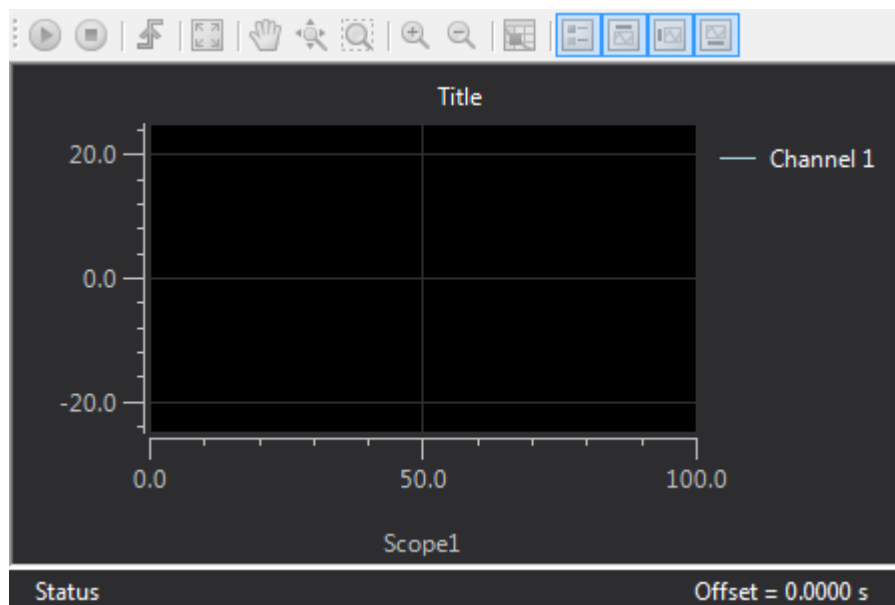
### **Introduced in R2014a**

# Scope Properties

Graphic instrument to display waveforms







## Description









Use the Scope instrument to display time-varying real-valued data, such as electronic waveforms.





**Note** Do not use this graphic instrument for instrument panel development. Support for this instrument relies on support for the .NET API, which will be removed in a future release. For alternatives to instrument panels in Simulink Real-Time Explorer, see “Instrumentation Apps for Real-Time Applications” on page 6-2.

The block provides controls that you can access at run time.

Action	Icon	Purpose
<b>Start Acquisition</b>		Enables the scope to trigger.
<b>Stop Acquisition</b>		Prevents the scope from triggering.
<b>Force trigger</b>		Triggers an immediate acquisition.
<b>AutoScale</b>		Rescales the scope axes to display the full range of the values.
<b>Axis Scroll</b>		Enables dragging the visible portion of the axes left, right, up, and down.
<b>Axis Zoom</b>		Enables stretching the axes left, right, up, and down.

Action	Icon	Purpose
<b>Zoom Box</b>		Enables selecting a box into which the display zooms.
<b>Zoom In</b>		Zooms the display inward from the current center.
<b>Zoom Out</b>		Zooms the display outward from the current center.
<b>Data-Cursor</b>		Creates a cursor that shows value and time data.
<b>Show Legend</b>		Displays the signal names and the colors of the signal traces in the top, right corner of the scope display.
<b>Show Label</b>		Displays the scope label in the top center of the scope display.
<b>Show Y-Axis</b>		Displays the Y-axis to the left of the scope display.
<b>Show X-Axis</b>		Displays the X-axis on the bottom of the scope display.

To access a dialog box containing key instrument or layout element properties, select the item. In the top, right corner, click the **Tasks** button .

To access the complete **Properties** list for an instrument or layout element, in the instrument panel, select the item. In the **Properties** list, to access a dialog box for a particular property group, click the group. To the right of the group, click the continuation button .

These properties represent a selection from the available properties. For more properties, see “Graphical Properties” on page 4-9.

## Properties

### Instrument

#### **NumSamples** — Number of contiguous samples that the scope displays

500 (default) | integer

The number of samples determines how much data the scope displays at any one time. The larger this value, the longer the scope takes to refresh.

#### **PreTriggerViewing** — Percentage of samples that the scope captures before a trigger event

0 (default) | integer

This property determines the horizontal (time axis) position of the trigger event in the display.

#### **TriggerLevel** — Level at which a signal trigger becomes active

0 (default) | integer

When the trigger signal crosses this value in the direction that you specified in **TriggerSlope**, the trigger becomes active.

#### **Dependency**

This property takes effect only if **TriggerSource** is SIGNAL.



**TriggerMode — Scope update policy**

SINGLESHOT (default) | REPEATED

Specify how often the display is updated:

- SINGLESHOT — Triggers once and stops the scope. To recapture, manually restart the scope.
- REPEATED — Triggers every time that the trigger condition is satisfied. To stop the display, manually stop the scope.

**TriggerScope — Name of triggering scope**

text

Specify the scope from which this instance of the scope instrument triggers.

**Dependency**This property takes effect only if **TriggerSource** is SCOPE.**TriggerSignal — Name of triggering signal**

text

Specify the signal from which this instance of the scope instrument triggers.

**Dependency**This property takes effect only if **TriggerSource** is SIGNAL.**TriggerSlope — Slope of signal waveform that triggers this scope**

EITHER (default) | RISING | FALLING


Specify the crossing direction at which the trigger becomes active:

- EITHER — Triggers when the signal rises and when it falls.
- RISING — Triggers only when the signal rises.
- FALLING — Triggers only when the signal falls.

Specify the crossing value with property **TriggerLevel**.**Dependency**This property takes effect only if **TriggerSource** is SIGNAL.**TriggerSource — Source of trigger event**


FREERUN (default) | MANUAL | SIGNAL | SCOPE

Specify the trigger condition:

- FREERUN — Displays data continuously without waiting for a trigger.
- MANUAL — Triggers when you click the **Force trigger** button .
- SIGNAL — Triggers when a specific signal crosses a specific value in a rising or falling direction.
- SCOPE — Triggers when a specified scope is triggered.

**Dependency**

This parameter has these dependencies:

- When **TriggerSource** is MANUAL, the **Force trigger** button  becomes available.
- When **TriggerSource** is SIGNAL, properties **TriggerSignal**, **TriggerSlope**, and **TriggerLevel** take effect.
- When **TriggerSource** is SCOPE, property **TriggerScope** takes effect.

### **YAxesLimits** — Positive and negative limits of Y-axis

max-min pair

Specify **Max** and **Min** limits to the Y-axis of the display.

#### **Bindings**

### **Signals** — List of signals that are connected to scope instrument

list of signal names

Enter signal names by using the Signal Collection Editor, which contains a **Members** list and a **Properties** list.

Example: `SetPoint`, `ControlValue`, `TankLevel`

### **TargetName** — Name of target computer to which the control is connected

text

Bind an instrument instance to a particular target computer. You can bind different instruments in the same instrument panel to different target computers.

#### **Design**

### **Locked** — Lock control against design-mode changes

False (default) | True

When True, prevents accidental changes in size or location of the instrument instance.

## **See Also**

### **Topics**

“View Signal Waveforms with Scope Instrument” on page 4-15

“View Signal Waveforms with Signal Triggered Scope Instrument” on page 4-21

“Triggering Scope Instruments”

“Instrumentation for Real-Time Applications” on page 4-2

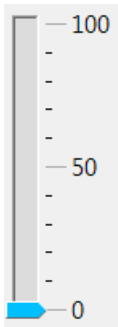
### **Introduced in R2017a**

# Slider Properties

Graphic instrument to set parameter values

## Description


Use the **Slider** instrument to set the approximate value of real-valued data, such as voltage or current. When you bind a **Slider** instrument to a parameter with enum type data values, the enumerated values appear as units on the instrument scale. The enumerated values are visible when you run the instrument panel. For more information, see “Add Instrument Panel to Tank Model” on page 4-25.





---

**Note** Do not use this graphic instrument for instrument panel development. Support for this instrument relies on support for the .NET API, which will be removed in a future release. For alternatives to instrument panels in Simulink Real-Time Explorer, see “Instrumentation Apps for Real-Time Applications” on page 6-2.

---

To access a dialog box containing key instrument or layout element properties, select the item. In the top, right corner, click the **Tasks** button .

To access the complete **Properties** list for an instrument or layout element, in the instrument panel, select the item. In the **Properties** list, to access a dialog box for a particular property group, click the group. To the right of the group, click the continuation button .

These properties represent a selection from the available properties. For more properties, see “Graphical Properties” on page 4-9.

## Properties

### Instrument

#### ScaleDisplay — Subset of scale display properties

property group

Open a dialog box to configure the scale display.

**Auto — Automatic generator display properties**

property group

To set properties of the automatic display generator, under the **Generator** tab, click the **Auto** tab.

**Mid Included — Insert tick midway between major ticks**

False (default) | True

If **MinorCount** is even, space the minor ticks equally around the center tick. If **MinorCount** is odd, replace the center tick with the middle tick.

**Fixed Min/Max Majors — Constrain top and bottom tick values**

False (default) | True

If True, the top and bottom ticks are constrained to be major ticks with min/max values defined by **Min** and **Span**.

**Minor Count — Number of minor ticks between major ticks**

4 (default) | numeric

**Min Text Spacing — Minimum space between scale ticks**

1 (default) | numeric

**Desired Increment — Display of major tick labels**

0 (default) | numeric

The number of labels is  $\text{span} / (\text{desired increment} + 1)$ . This setting has no effect if the required labels do not fit in the space available in the graphic.

**Text Formatting — Text display properties**

property group

To set text formatting properties, click the **Text Formatting** tab.

**Style — Style of tick display**

Number (default) | Thousands | Prefix | Exponent | Price32nds | DateTime | DateTimeUTC

**Precision Style — Style of expressing precision**

FixedDecimalPoints (default) | SignificantDigits | None

**Precision — Number of digits to the right of the decimal point**

0 (default) | numeric

**Units Text — Text to add to number**

text

Open a text editor and construct text to add to number. By default, the units text appears to the right of the number. If **Style** is **Prefix**, the text appears to the left of the number.

**ScaleRange — Subset of scale range properties**

property group

Specify range of control scale. The property **Max** is calculated from **Min** and **Span**.

**Min — Minimum possible value**

0 (default) | numeric

**Span — Number of values between the minimum and maximum values**

100 (default) | numeric

**Reverse — Reverse scale direction**

False (default) | True

By default, the instrument displays values that increase in the customary directions: Clockwise, left to right, or down to up, depending on the instrument. If value is True, reverse the scale so that the values increase in the opposite directions: Counterclockwise, right to left, or up to down.

**Scale Type — Scale type: linear, logarithmic, or split**

Linear (default) | Log10 | SplitLinearLog10

Scale type, one of:

- **Linear** — Monotonic increasing or decreasing
- **Log10** — Log base 10 increasing or decreasing
- **SplitLinearLog10** — Monotonic increasing or decreasing in one part, log base 10 in the other part

**Design****Locked — Lock control against design-mode changes**

False (default) | True

When True, prevents accidental changes in size or location of the instrument instance.

**Bindings****BindingSrcPath — Name of signal or parameter**

text

Specify the hierarchical name of the signal or parameter that is bound to the instrument.

**BindingSrcType — Type of object to which the instrument is bound**

None (default) | Parameter | Signal

Specify the type of object to which the instrument is bound. The Signal binding is visible only for display instruments.

**TargetName — Name of target computer to which the control is connected**

text

Bind an instrument instance to a particular target computer. You can bind different instruments in the same instrument panel to different target computers.

**See Also****Topics**

“View Signal Waveforms with Scope Instrument” on page 4-15

“View Signal Waveforms with Signal Triggered Scope Instrument” on page 4-21

“Add Instrument Panel to Tank Model” on page 4-25

“Triggering Scope Instruments”

“Instrumentation for Real-Time Applications” on page 4-2

**Introduced in R2014a**

# SwitchLED Properties

Graphic instrument to set binary parameter values

## Description


Use the **SwitchLED** instrument to set a binary (1 or 0) value.





---

**Note** Do not use this graphic instrument for instrument panel development. Support for this instrument relies on support for the .NET API, which will be removed in a future release. For alternatives to instrument panels in Simulink Real-Time Explorer, see “Instrumentation Apps for Real-Time Applications” on page 6-2.

---

To access a dialog box containing key instrument or layout element properties, select the item. In the top, right corner, click the **Tasks** button .

To access the complete **Properties** list for an instrument or layout element, in the instrument panel, select the item. In the **Properties** list, to access a dialog box for a particular property group, click the group. To the right of the group, click the continuation button .

These properties represent a selection from the available properties. For more properties, see “Graphical Properties” on page 4-9.

## Properties

### Instrument

#### **BackgroundOffValueImage** — Background image when switch is off

(none) | file path

Open a file navigator dialog box to specify a graphic file name. When the switch goes to the value 0 (off), the switch displays this image as background.

#### **BackgroundOnValueImage** — Background image when switch is on

(none) | file path

When the switch goes to the value 1 (on), the switch displays this image as background.

#### **Indicator** — Subset of indicator display properties

property group

Open a dialog box to configure the indicator display.

#### **ColorActive** — Indicator color when switch is on

DeepSkyBlue (default) | struct

Open a dialog box with these tabs:

**Custom — Customizable color grid**

struct

**Web — List of web colors**

struct

**System — List of system colors**

struct

**ColorInactive — Indicator color when switch is off**

[41, 41, 41] (default) | struct

Open a dialog box with these tabs:

**Custom — Customizable color grid**

struct

**Web — List of web colors**

struct

**System — List of system colors**

struct

**Text — Text on switch body**

ON (default) | text

Enter text to add to appear on switch body

**Design****Locked — Lock control against design-mode changes**

False (default) | True

When True, prevents accidental changes in size or location of the instrument instance.

**Misc****BindingSrcPath — Name of signal or parameter**

text

Specify the hierarchical name of the signal or parameter that is bound to the instrument.

**BindingSrcType — Type of object to which the instrument is bound**

None (default) | Parameter | Signal

Specify the type of object to which the instrument is bound. The `Signal` binding is visible only for display instruments.

**TargetName — Name of target computer to which the control is connected**

text

Bind an instrument instance to a particular target computer. You can bind different instruments in the same instrument panel to different target computers.



## **See Also**

### **Topics**

“Instrumentation for Real-Time Applications” on page 4-2

**Introduced in R2014a**

## SwitchLever Properties

Graphic instrument to set binary parameter values

### Description


Use the **SwitchLever** instrument to set a binary (1 or 0) value.





---

**Note** Do not use this graphic instrument for instrument panel development. Support for this instrument relies on support for the .NET API, which will be removed in a future release. For alternatives to instrument panels in Simulink Real-Time Explorer, see “Instrumentation Apps for Real-Time Applications” on page 6-2.

---

To access a dialog box containing key instrument or layout element properties, select the item. In the top, right corner, click the **Tasks** button .

To access the complete **Properties** list for an instrument or layout element, in the instrument panel, select the item. In the **Properties** list, to access a dialog box for a particular property group, click the group. To the right of the group, click the continuation button .

These properties represent a selection from the available properties. For more properties, see “Graphical Properties” on page 4-9.

### Properties

#### Instrument

##### Switch — Subset of switch display properties

property group

Open a dialog box to configure the switch display.

##### CoLorBase — Color palette of switch base graphic

Control (default) | struct

Open a dialog box with these tabs:

##### Custom — Customizable color grid

struct

##### Web — List of web colors

struct

##### System — List of system colors

struct

**ColorLever — Color palette of switch lever graphic**

Control (default) | struct

Open a dialog box with these tabs:

**Custom — Customizable color grid**

struct

**Web — List of web colors**

struct

**System — List of system colors**

struct

**Design****Locked — Lock control against design-mode changes**

False (default) | True

When True, prevents accidental changes in size or location of the instrument instance.

**Misc****BindingSrcPath — Name of signal or parameter**

text

Specify the hierarchical name of the signal or parameter that is bound to the instrument.

**BindingSrcType — Type of object to which the instrument is bound**

None (default) | Parameter | Signal

Specify the type of object to which the instrument is bound. The Signal binding is visible only for display instruments.

**TargetName — Name of target computer to which the control is connected**

text

Bind an instrument instance to a particular target computer. You can bind different instruments in the same instrument panel to different target computers.

**See Also****Topics**

“Instrumentation for Real-Time Applications” on page 4-2

**Introduced in R2017b**

## SwitchRocker Properties

Graphic instrument to set binary parameter values

### Description


Use the **SwitchRocker** instrument to set a binary (1 or 0) value.





---

**Note** Do not use this graphic instrument for instrument panel development. Support for this instrument relies on support for the .NET API, which will be removed in a future release. For alternatives to instrument panels in Simulink Real-Time Explorer, see “Instrumentation Apps for Real-Time Applications” on page 6-2.

---

To access a dialog box containing key instrument or layout element properties, select the item. In the top, right corner, click the **Tasks** button .

To access the complete **Properties** list for an instrument or layout element, in the instrument panel, select the item. In the **Properties** list, to access a dialog box for a particular property group, click the group. To the right of the group, click the continuation button .

These properties represent a selection from the available properties. For more properties, see “Graphical Properties” on page 4-9.

### Properties

#### Instrument

**Indicator — Subset of indicator display properties**  
property group

Open a property group to configure the indicator display.

**ColorActive — Color of indicator when switch is on**  
DeepSkyBlue (default) | struct

Open a dialog box with these tabs:

**Custom — Customizable color grid**  
struct

**Web — List of web colors**  
struct

**System — List of system colors**  
struct

**ColorInactive — Color of indicator when switch is off**

[41, 41, 41] (default) | struct

Open a dialog box with these tabs:

**Custom — Customizable color grid**

struct

**Web — List of web colors**

struct

**System — List of system colors**

struct

**Switch — Subset of switch display properties**

property group

Open a dialog box to configure the switch display.

**Color — Color palette of switch graphic**

Control (default) | struct

Open a dialog box with these tabs:

**Custom — Customizable color grid**

struct

**Web — List of web colors**

struct

**System — List of system colors**

struct

**Bevel — Switch bevel line properties**

struct

Open a dialog box to configure the lines on the movable part of the switch.

**Visible — Make bevel lines visible**

True (default) | False

If True, the bevel lines are visible, otherwise they are not visible.

**Style — 3-D appearance of bevel lines**

Raised (default) | Sunken | None

- Raised — Bevel lines lie on top of the surface.
- Sunken — Bevel lines lie below the surface.
- None — Bevel lines are not visible.,

**Thickness — Thickness of bevel lines**

X3 (default) | X2 | X4

Thickness of bevel lines, in pixels.

**Margin Edge — Space around bevel lines**

3 (default) | integer

Space around bevel lines, in pixels.

**Design****Locked — Lock control against design-mode changes**

False (default) | True

When True, prevents accidental changes in size or location of the instrument instance.

**Misc****BindingSrcPath — Name of signal or parameter**

text

Specify the hierarchical name of the signal or parameter that is bound to the instrument.

**BindingSrcType — Type of object to which the instrument is bound**

None (default) | Parameter | Signal

Specify the type of object to which the instrument is bound. The Signal binding is visible only for display instruments.

**TargetName — Name of target computer to which the control is connected**

text

Bind an instrument instance to a particular target computer. You can bind different instruments in the same instrument panel to different target computers.

**See Also****Topics**

“Instrumentation for Real-Time Applications” on page 4-2

**Introduced in R2017b**

# SwitchToggle Properties

Graphic instrument to set binary parameter values

## Description


Use the **SwitchToggle** instrument to set a binary (1 or 0) value.





---

**Note** Do not use this graphic instrument for instrument panel development. Support for this instrument relies on support for the .NET API, which will be removed in a future release. For alternatives to instrument panels in Simulink Real-Time Explorer, see “Instrumentation Apps for Real-Time Applications” on page 6-2.

---

To access a dialog box containing key instrument or layout element properties, select the item. In the top, right corner, click the **Tasks** button .

To access the complete **Properties** list for an instrument or layout element, in the instrument panel, select the item. In the **Properties** list, to access a dialog box for a particular property group, click the group. To the right of the group, click the continuation button .

These properties represent a selection from the available properties. For more properties, see “Graphical Properties” on page 4-9.

## Properties

### Instrument

**Switch — Subset of switch display properties**  
property group

Open a dialog box to configure the switch display.

**Height — Height of changing part of switch graphic**  
50 (default) |  $10 \leq \text{uint} \leq 90$

Enter an integer value with the percentage of the switch frame that is covered by the changing part of the graphic. Values below 10% or above 90% produce an error.

**Bevel — Switch bevel line properties**  
struct

Open a dialog box to configure the lines on the movable part of the switch.

**Visible — Make bevel lines visible**  
True (default) | False

If `True`, the bevel lines are visible, otherwise they are not visible.

**Style — 3-D appearance of bevel lines**

Raised (default) | Sunken | None

- **Raised** — Bevel lines lie on top of the surface.
- **Sunken** — Bevel lines lie below the surface.
- **None** — Bevel lines are not visible.,

**Thickness — Thickness of bevel lines**

X3 (default) | X2 | X4

Thickness of bevel lines, in pixels.

**Margin Edge — Space around bevel lines**

3 (default) | integer

Space around bevel lines, in pixels.

**Design****Locked — Lock control against design-mode changes**

False (default) | True

When `True`, prevents accidental changes in size or location of the instrument instance.

**Misc****BindingSrcPath — Name of signal or parameter**

text

Specify the hierarchical name of the signal or parameter that is bound to the instrument.

**BindingSrcType — Type of object to which the instrument is bound**

None (default) | Parameter | Signal

Specify the type of object to which the instrument is bound. The `Signal` binding is visible only for display instruments.

**TargetName — Name of target computer to which the control is connected**

text

Bind an instrument instance to a particular target computer. You can bind different instruments in the same instrument panel to different target computers.

**See Also****Topics**

“Instrumentation for Real-Time Applications” on page 4-2

**Introduced in R2017b**



# Simulink Real-Time Instrumentation Object

---

## Instrumentation Apps for Real-Time Applications

To visualize the behavior of a real-time application running on a target computer, you can create instrument panel apps. An instrument panel app is a user-interface application into which you can insert one or more instruments. To create an instrument panel app, use App Designer or an m-script.

- When you create an instrument panel app in the App Designer **Design View**, you add instrument components from the App Designer **Component Library** to the app. You configure each instrument by using fields in the **Component Browser**. In the App Designer **Code View**, you add callback code to handle component events, such as new streaming data or interaction with the app. For more information, see “App Building Components” (MATLAB) and “Managing Code in App Designer Code View” (MATLAB).
- When you create an instrument panel app by using an m-script, you use a programmatic approach to add each instrument to the panel as UI component. For more information, see .

To stream signal and parameter data to the instrument panel app from the real-time application, you use the `SimulinkRealTime.prototype.Instrumentation` object. After you create an instrumentation object for a real-time application, you can use instrumentation object functions to connect signals and parameters from the real-time application to instrument panel app callbacks.

When identifying parameters and output signals to stream signal to the instrument panel app from the real-time application, it can be helpful to use the hierarchical display of signals and parameters. See **Simulink Real-Time Target Computer Explorer (Tech Preview)**. For more information, see “Display and Filter Hierarchical Signals and Parameters (tech preview)”.

### See Also

**Simulink Real-Time Explorer | Simulink Real-Time Target Computer Explorer (Tech Preview)**  
| **Simulink Real-Time Target Computer Manager** |  
`SimulinkRealTime.prototype.Instrumentation`

### Related Examples

- “Add App Designer Instrument Panel App to Tank Model”
- “Add m-Script Instrument Panel App to Tank Model”

### More About

- “App Building Components” (MATLAB)
- “Managing Code in App Designer Code View” (MATLAB)
- “Display and Filter Hierarchical Signals and Parameters (tech preview)”

# Target Computer Command-Line Interface Reference

---

## Target Computer Commands

You can interact with the real-time application after it has been loaded to the target computer by using a limited set of target computer commands.

- Function commands — Interact directly with the scope or target.
- Property commands — Work with target and scope properties.
- Variable commands — Alias target computer command-line interface commands with names of your choice.

Refer to “Control Real-Time Application at Target Computer Command Line” for a description of how to use these functions and commands.

To read the target computer console log, call `SimulinkRealTime.utils.getConsoleLog`.

### Target Object Function Commands

When you are using the target computer command-line interface, target object functions are limited to starting and stopping the real-time application.

The following table lists the syntax for the target commands that you can use on the target computer. The equivalent MATLAB syntax is shown in the right column. The target object name `tg` is used as an example for the MATLAB functions. These functions assume that you have already loaded the real-time application onto the target computer.

Target Computer Command	Description	MATLAB Equivalent
<code>start</code>	Start the real-time application currently loaded on the target computer.	<code>start(tg)</code>
<code>stop</code>	Stop the real-time application currently running on the target computer.	<code>stop(tg)</code>
<code>reboot</code>	Restart the target computer.	<code>reboot(tg)</code>

### Target Object Property Commands

When you are using the target computer command-line interface, target object properties are limited to parameters, signals, stop time, and sample time.

The following table lists the syntax for the target commands that you can use to manipulate target object properties. The MATLAB equivalent syntax is shown in the right column, and the target object name `tg` is used as an example for the MATLAB functions.

To read signal and parameter and signal values, use signal and parameter names (for example, `S1`, `P1`). To both read and write parameter values, use parameter indexes (for example, `0`, `1`).

Target Computer Command	Description	MATLAB Equivalent
<code>getpar</code> <code>param_index</code>	Display the value of a block parameter using the parameter index.	<code>getparam(tg, param_index)</code>

Target Computer Command	Description	MATLAB Equivalent
setpar param_index = number	Change the value of a block parameter using the parameter index.	setparam(tg, param_index, number)
stoptime = number	With the value number, run for the specified number of seconds.	tg.StopTime = number
stoptime = Inf	With the value Inf, run the real-time application until you manually stop it or reset the target computer.	tg.StopTime = Inf
sampletime = number	Enter a new sample time.	tg.SampleTime = number
P#	Display the value of the block parameter with index #.  For example, P2 displays the value of block parameter 2.	getparam(tg, param_index)
S#	Display the value of the signal with index #.  For example, S2 displays the value of signal 2.	getsignal(tg, sig_index)

## Scope and Video Object Function Commands

When using the target computer command-line interface, you use scope object functions to start a scope and add signal traces. You can also collapse scopes and video displays into icons and expand them again. The functions `addscope` and `remscope` are target object functions that also run on the development computer.

The following table lists the syntax for the target commands that you can use on the target computer. The MATLAB equivalent syntax is shown in the right column. The target object name `tg` and the scope object name `sc` are used as an example for the MATLAB functions.

To add and remove scopes, use scope numbers (for example, 0, 1) and not scope names (for example, Scope 1, Scope 2). To add and remove signals from scopes, use signal indexes (for example, 0, 1) and not signal names (for example, S0, S1).

Target Computer Command	Description	MATLAB Equivalent
addscope	Without an argument, add a target scope and assign it the next available index.	addscope(tg, 'target')
addscope scope_index	With argument <code>scope_index</code> , add a target scope and assign it index <code>scope_index</code> .	addscope(tg, 'target', scope_index)

Target Computer Command	Description	MATLAB Equivalent
remscope scope_index	With value scope_index, remove scope scope_index.	remscope(tg, scope_index) remscope(tg)
remscope all	With value all, remove all scopes.	
startscope scope_index	With value scope_index, start the scope with index scope_index.	start(sc) start(getscope(tg))
startscope all	With value all, start all scopes.	
stopscope scope_index	With value scope_index, stop the scope with index scope_index.	stop(sc) stop(getscope(tg))
stopscope all	With value all, stop all scopes.	
addsignal scope_index = sig_index1, sig_index2, ...	With values sig_index1, sig_index2, ..., add the signals with these signal indexes to the scope with index scope_index.	addsignal(sc, sig_index_vector)
remsignal scope_index = sig_index1, sig_index2, ...	With values sig_index1, sig_index2, ..., remove the signals with these signal indexes from the scope with index scope_index.	remsignal(sc, sig_index_vector)
remsignal scope_index	Without a sig_index value, remove all the signals from the scope with index scope_index.	remsignal(sc)
show Scope scope_index	With value scope_index, expand scope scope_index from an icon.	
hide Scope scope_index	With value scope_index, collapse scope scope_index into an icon.	
show Video video_index	With value video_index, expand video display video_index from an icon.	
hide Video video_index	With value video_index, collapse video display video_index into an icon.	

## Scope Object Property Commands

When you use the target computer command-line interface, scope object properties are limited to the properties shown in the following table.

The following table lists the syntax for the target properties that you can set on the target computer. The equivalent MATLAB syntax is shown in the right column. The scope object name `sc` is used as an example for the MATLAB functions.

To change scope properties, use scope indexes (for example, 0, 1) and not scope names (for example, Scope 1, Scope 2).

If a scope is running, stop the scope before you change a scope property.

Target Computer Command	Description	MATLAB Equivalent
<code>numsamples scope_index = number</code>	Set the number of contiguous samples captured by scope <code>scope_index</code> to <code>number</code> .	<code>sc.NumSamples = number</code>
<code>decimation scope_index = 1</code>	With value 1, the scope returns all sample points.	<code>sc.Decimation = 1</code>
<code>decimation scope_index = number</code>	With value <i>n</i> , the scope returns every <i>n</i> th sample point.	<code>sc.Decimation = number</code>
<code>grid scope_index on</code> <code>grid scope_index off</code>	With value <code>on</code> , the scope grid display is visible.  With value <code>off</code> , the scope grid display is not visible.	<code>sc.Grid = 'on'</code>  <code>sc.Grid = 'off'</code>
<code>scopemode scope_index = 0</code> <code>scopemode scope_index = numerical</code> <code>scopemode scope_index = 1</code> <code>scopemode scope_index = redraw</code> <code>scopemode scope_index = 3</code> <code>scopemode scope_index = rolling</code>	With value 0 or <code>numerical</code> , scope <code>scope_index</code> displays signal values as text.  With value 1 or <code>redraw</code> , scope <code>scope_index</code> plots signal values when <code>numsamples</code> samples have been acquired.  With value 3 or <code>rolling</code> , scope <code>scope_index</code> plots signal values at every sample time.  Value 2, <code>sliding</code> , will be removed in a future release. It behaves like value 3, <code>rolling</code> .	<code>sc.DisplayMode = 'numerical'</code>  <code>sc.DisplayMode = 'redraw'</code>  <code>sc.DisplayMode = 'rolling'</code>

Target Computer Command	Description	MATLAB Equivalent
<code>triggermode scope_index = 0</code>	With value 0 or <code>freerun</code> , scope <code>scope_index</code> triggers on every sample time.	<code>sc.TriggerMode = 'freerun'</code>
<code>triggermode scope_index = freerun</code>	With value 1 or <code>software</code> , scope <code>scope_index</code> triggers from Command Window.	<code>sc.TriggerMode = 'software'</code>
<code>triggermode scope_index = 1</code>	With value 2 or <code>signal</code> , scope <code>scope_index</code> triggers when a designated signal changes state.	<code>sc.TriggerMode = 'signal'</code>
<code>triggermode scope_index = software</code>	With value 3 or <code>scope</code> , scope <code>scope_index</code> triggers when a designated scope triggers.	<code>sc.TriggerMode = 'scope'</code>
<code>triggermode scope_index = 2</code>		
<code>triggermode scope_index = signal</code>		
<code>triggermode scope_index = 3</code>		
<code>triggermode scope_index = scope</code>		
<code>numprepostsamples scope_index = number</code>	Number of samples collected before or after a trigger event.	<code>sc.NumPrePostSamples = number</code>
<code>triggersignal scope_index = sig_index</code>	If <code>triggermode</code> is <code>signal</code> , <code>triggersignal</code> identifies the block output signal to use for triggering the scope.	<code>sc.TriggerSignal = sig_index</code>
<code>triggersample scope_index = number</code>	If <code>triggermode</code> is <code>scope</code> , <code>triggersample</code> specifies which sample of the triggering scope the current scope triggers on.	<code>sc.TriggerSample = number</code>
<code>triggerlevel scope_index = number</code>	If <code>triggermode</code> is <code>signal</code> , <code>triggerlevel</code> indicates the value the signal has to cross to trigger the scope to start acquiring data.	<code>sc.TriggerLevel = number</code>



Target Computer Command	Description	MATLAB Equivalent
<pre>triggerslope scope_index = 0</pre> <pre>triggerslope scope_index = either</pre> <pre>triggerslope scope_index = 1</pre> <pre>triggerslope scope_index = rising</pre> <pre>triggerslope scope_index = 2</pre> <pre>triggerslope scope_index = falling</pre>	<p>If triggermode is signal:</p> <p>With value 0 or either, the signal triggers the scope when it crosses triggerlevel in either the rising or falling directions.</p> <p>With value 1 or rising, the signal triggers the scope when it crosses triggerlevel in the rising direction.</p> <p>With value 2 or falling, the signal triggers the scope when it crosses triggerlevel in the falling direction.</p>	<pre>sc.TriggerSlope = 'Either'</pre> <pre>sc.TriggerSlope = 'Rising'</pre> <pre>sc.TriggerSlope = 'Falling'</pre>
<pre>triggerscope scope_index = scope_index2</pre>	<p>If triggermode is scope, triggerscope identifies the scope to use for a trigger.</p>	<pre>sc.TriggerScope = scope_index2</pre>
<pre>triggerscopesample scope_index= integer</pre>	<p>If triggermode is scope, triggerscopesample specifies which sample of the triggering scope to trigger on.</p>	<pre>sc.TriggerScopeSample = integer</pre>
<pre>ylimit scope_index = min_y, max_y</pre> <pre>ylimit scope_index = auto</pre>	<p>With value min_y, max_y, change the lower and upper y-axis values to min_y and max_y.</p> <p>With value auto, the values being displayed determine the lower and upper y-axis values.</p>	<pre>sc.YLimit = [min_y, max_y]</pre> <pre>sc.YLimit = 'auto'</pre>

## Aliasing with Variable Commands

You can set a variable to a command character vector, and later use that variable to execute that command. For example, type the following on the target computer command line:

```
setvar aa = startscope 2
setvar bb = stopscope 2
```

Later, to start and stop scope 2, you can type the following:

```
aa
bb
```

The following table lists the syntax for the aliasing variable commands that you can use only on the target computer. MATLAB does not provide equivalent syntax. For a usage example, see “Alias Commands at Target Computer Command Line”.

<b>Target Computer Command</b>	<b>Description</b>
setvar variable_name = command	Set a variable to a target computer command-line character vector.
getvar variable_name	Display the value of a variable.
delvar variable_name	Delete a variable.
delallvar	Delete all variables.
showvar	Display a list of variables.

**See Also**

`SimulinkRealTime.utils.getConsoleLog`

# Simulink Real-Time Performance Advisor Checks

---

- “Simulink Real-Time Performance Advisor Checks” on page 8-2
- “Baseline” on page 8-4
- “System Target File Compatibility” on page 8-5
- “Profiling Settings” on page 8-6
- “Check Target” on page 8-7
- “Real-Time Performance Baseline” on page 8-8
- “Determine minimum sample time” on page 8-9
- “Real-Time” on page 8-10
- “Outport Logging” on page 8-11
- “EtherCAT Synchronous SDO” on page 8-12
- “Concurrent execution” on page 8-13
- “Final Validation” on page 8-14

## Simulink Real-Time Performance Advisor Checks

Parameter	Description
“Baseline” on page 8-4	Checks preconditions for real-time execution, and then measures its initial performance to establish a baseline.
“System Target File Compatibility” on page 8-5	Checks that the system target file is compatible with the real-time advisor workflow.
“Profiling Settings” on page 8-6	Checks that the execution profiling settings are compatible with the real-time advisor workflow.
“Check Target” on page 8-7	Pings the target computer and verifies that it is in a clean state.
“Real-Time Performance Baseline” on page 8-8	Collects execution-time measurements and establishes a performance baseline.
“Determine minimum sample time” on page 8-9	To determine the minimum sample time possible for the model, this check iteratively runs the model, decreasing the sample time on each run. The algorithm stops at a sample time that is greater than a sample time that can cause an overload.
“Real-Time” on page 8-10	Checks the real-time application and application setup, and recommends changes to improve performance.
“Outport Logging” on page 8-11	Checks for the use of Outport blocks for data logging. Data logging using Outport blocks can slow down execution.
“EtherCAT Synchronous SDO” on page 8-12	Checks for the use of EtherCAT Synchronous SDO blocks. EtherCAT Synchronous SDO blocks can slow down execution.
“Concurrent execution” on page 8-13	Checks if you can perform concurrent execution of the real-time application on a multicore target computer.
“Final Validation” on page 8-14	Validates the overall performance improvement that your changes make in real-time execution time and accuracy.

Use Performance Advisor checks to improve real-time application execution time. Performance Advisor runs the check and only provides recommendations. It does not modify your model.

### To get help on an option

- 1 Right-click the option text label.
- 2 From the context menu, select **What's This**.



## **See Also**

- “Improve Performance of Multirate Model”
- “Improve Simulation Performance Using Performance Advisor” (Simulink)

## **Baseline**

Checks preconditions for real-time execution, and then measures its initial performance to establish a baseline.

Performance Advisor later uses this baseline to compare performance before and after you implement the improvements that Performance Advisor recommends.

### **See Also**

- “Improve Performance of Multirate Model”
- “Performance Optimization”
- “Improve Simulation Performance Using Performance Advisor” (Simulink)

## System Target File Compatibility

Checks that the system target file is compatible with the real-time advisor workflow.

In the Configuration Parameters **Code Generation** pane, set the **System target file** setting to `slrt.tlc`.

### See Also

- “Improve Performance of Multirate Model”
- “Set Configuration Parameters”
- “Performance Optimization”
- “Improve Simulation Performance Using Performance Advisor” (Simulink)

## Profiling Settings

Checks that the execution profiling settings are compatible with the real-time advisor workflow.

In the Configuration Parameters **Verification** pane, select the **Measure task execution time** check box.

### See Also

- “Improve Performance of Multirate Model”
- “Performance Optimization”
- “Improve Simulation Performance Using Performance Advisor” (Simulink)



## Check Target

Pings the target computer and verifies that it is in a clean state.

To clear the target computer of prior simulations, select the **Automatically unload any loaded application** check box.

### See Also

- “Improve Performance of Multirate Model”
- “Performance Optimization”
- “Improve Simulation Performance Using Performance Advisor” (Simulink)

## Real-Time Performance Baseline

Collects execution-time measurements and establishes a performance baseline.

This check builds, downloads, and executes the real-time application on the target computer. When the check passes, it displays the following information:

- **Margin before CPU overload (0% indicates CPU overload)** — A table containing the real-time application task name, the sample rate, and the margin.

Margin is the minimum value of headroom for a task over all the measured samples.

Headroom is the time between the end of execution and the start of the next sample, as a percentage of sample time. For example, if the sample time is 1.2 ms and a task takes 0.93 ms to execute, the headroom is  $(1.2 - 0.93)/1.2$ , or 22.5%.

As the margin approaches 0%, the application gets closer to overloading the CPU.

- **Average CPU Usage** — A pie chart showing the average CPU resources that the real-time application uses, as a percentage of available resources.

The available CPU resources include all of the processors on a multicore target computer. For example, a single-tasking model running on a quad-core processor cannot exceed 25% CPU usage.

The background task aggregates the CPU time for all operating system tasks that are not related to application execution. These tasks include updating the target screen, communicating with the development computer, and so on. It also includes the time that the CPU is idle.

### See Also

- “Improve Performance of Multirate Model”
- “Performance Optimization”
- “Improve Simulation Performance Using Performance Advisor” (Simulink)

## Determine minimum sample time

To determine the minimum sample time possible for the model, this check iteratively runs the model, decreasing the sample time on each run. The algorithm stops at a sample time that is greater than a sample time that can cause an overload.

Random factors such as network latency can change the minimum sample time of a model. As a best practice, set the sample time for your model to a value greater than the minimum sample time returned by this check.

This check builds, downloads, and executes the real-time application.

### See Also

- “Improve Performance of Multirate Model”
- “Performance Optimization”
- “Improve Simulation Performance Using Performance Advisor” (Simulink)

## **Real-Time**

Checks the real-time application and application setup, and recommends changes to improve performance.

### **See Also**

- “Improve Performance of Multirate Model”
- “Performance Optimization”
- “Improve Simulation Performance Using Performance Advisor” (Simulink)

## Output Logging

Checks for the use of Outputport blocks for data logging. Data logging using Outputport blocks can slow down execution.

As an alternative, consider using a real-time Scope block configured as a file scope.

### See Also

- “Improve Performance of Multirate Model”
- Scope
- “Performance Optimization”
- “Improve Simulation Performance Using Performance Advisor” (Simulink)

## **EtherCAT Synchronous SDO**

Checks for the use of EtherCAT Synchronous SDO blocks. EtherCAT Synchronous SDO blocks can slow down execution.

As an alternative, consider using EtherCAT Asynchronous SDO blocks.

### **See Also**

- “Improve Performance of Multirate Model”
- “EtherCAT”
- “Performance Optimization”
- “Improve Simulation Performance Using Performance Advisor” (Simulink)

## Concurrent execution

Checks if you can perform concurrent execution of the real-time application on a multicore target computer.

To perform concurrent execution:

- Acquire a multicore target computer.
- In the Configuration Parameters **Solver** pane, select the **Allow tasks to execute concurrently on target** check box.

This check updates the model diagram.

### See Also

- “Improve Performance of Multirate Model”
- “Concurrent Execution with Multicore Target Computer”
- “Performance Optimization”
- “Improve Simulation Performance Using Performance Advisor” (Simulink)

## Final Validation

Validates the overall performance improvement that your changes make in real-time execution time and accuracy.

If you have not validated the performance improvement resulting from other checks, use this check to perform a final validation of the changes to the model.

This check builds, downloads, and executes the real-time application. When the check passes, it displays the following information for the baseline and final validation runs:

- **Margin before CPU overload (0% indicates CPU overload)** — A table containing, for each run, the real-time application task name, the sample rate, and the margin.

Margin is the minimum value of headroom for a task over all the measured samples.

Headroom is the time between the end of execution and the start of the next sample, as a percentage of sample time. For example, if the sample time is 1.2 ms and a task takes 0.93 ms to execute, the headroom is  $(1.2 - 0.93)/1.2$ , or 22.5%.

As the margin approaches 0%, the application gets closer to overloading the CPU.

- **Average CPU Usage** — Pie charts showing, for each run, the average CPU resources that the real-time application uses, as a percentage of available resources.

The available CPU resources include all of the processors on a multicore target computer. For example, a single-tasking model running on a quad-core processor cannot exceed 25% CPU usage.

The background task aggregates the CPU time for all operating system tasks that are not related to application execution. These tasks include updating the target screen, communicating with the development computer, and so on. It also includes the time that the CPU is idle.

### See Also

- “Improve Performance of Multirate Model”
- “Performance Optimization”
- “Improve Simulation Performance Using Performance Advisor” (Simulink)
- “Comparing Performance” (Simulink)