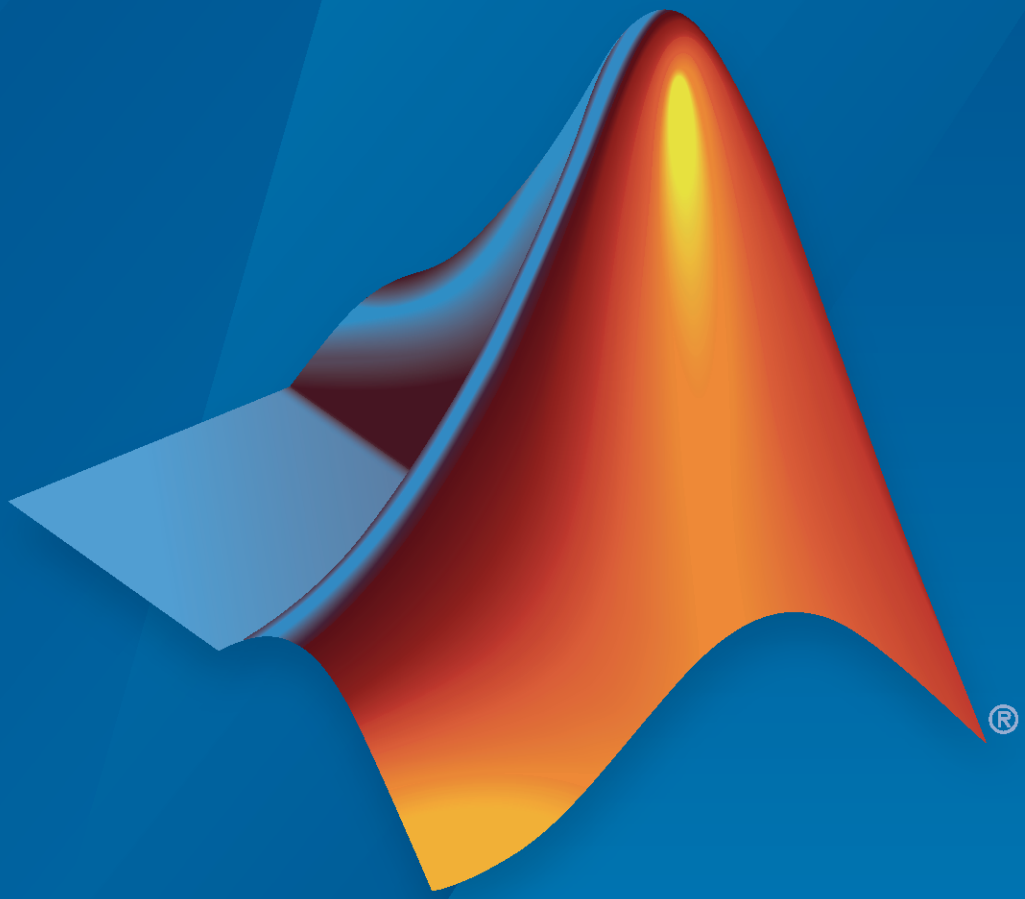
Simulink® Real-Time™

Reference



MATLAB® & SIMULINK®

R2024b

MathWorks®

How to Contact MathWorks

Latest news: `www.mathworks.com`

Sales and services: `www.mathworks.com/sales_and_services`

User community: `www.mathworks.com/matlabcentral`

Technical support: `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

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)
September 2020	Online only	Revised for Version 7.0 (Release 2020b)
March 2021	Online only	Revised for Version 7.1 (Release 2021a)
September 2021	Online only	Revised for Version 7.2 (Release 2021b)
March 2022	Online only	Revised for Version 8.0 (Release 2022a)
September 2022	Online only	Revised for Version 8.1 (Release 2022b)
March 2023	Online only	Revised for Version 8.2 (Release 2023a)
September 2023	Online only	Revised for Version 23.2 (R2023b)
March 2024	Online only	Revised for Version 24.1 (R2024a)
September 2024	Online only	Revised for Version 24.2 (R2024b)

Contents

1 Configuration Parameters

Simulink Real-Time Options Pane **1-2**
- Configuration **1-2**
- Tips **1-2**
- To get help on an option **1-2**

2 TLC Options Parameters

TLC Command-Line Options **2-2**

3 Apps

4 Target Computer Status Monitor

Target Computer Status Monitor **4-2**
- Display Status Monitor **4-2**
- Display Status Monitor by Using PuTTY **4-2**

5 Target Computer Command-Line Interface Reference

Target Computer Command-Line Interface **5-2**
- Target Object Commands **5-2**
- Target Computer RTOS System Commands **5-3**

Configuration Parameters

Simulink Real-Time Options Pane

Parameter	Description
Log level	Selects filtering level that limits RTOS system messages that appear in the system log.
Force polling mode	Enables polling mode —instead of interrupt-driven mode— for clocking the real-time application.
Max file log runs	Selects the number of file log run to retain when logs are stored on the target computer.
Compile with GCC -ffast-math	Enables the GCC compiler `-ffast-math` option when compiling real-time application code.

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 `slrealtime.tlc` or another Simulink Real-Time STF. 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 `slrealtime.tlc`. To retain the updated setting, save the updated model.

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**.


What's This?

Log level

Selects filtering level for system log messages

Model Configuration Pane: Code Generation / Simulink Real-Time Options

Description

Selects filtering level that limits Simulink Real-Time target computer system messages that appear in the system log. To open the viewer tab in Simulink Real-Time Explorer and view the system log from the target computer `tg`, in the MATLAB Command Window, type:

```
slrtExplorer
```

For more information, see `slrtExplorer`.

Dependencies

The `SLRTLogLevel` configuration parameter sets the initial value for the `loglevel` option when you build the real-time application.

For more information, see `Application` object.
Option: `loglevel`

Settings

`info` | `trace` | `debug` | `warning` | `error` | `fatal`

Default: `info`

`info`
: Select to include all system messages in the system log.

`trace`
: Select to include all system memory trace messages in the system log.

`debug`
: Select to include all system debug messages in the system log.

`warning`
: Select to include all system warning messages in the system log.

`error`
: Select to include all system error messages in the system log.

`fatal`
: Select to include all system fatal messages in the system log.

Recommended Settings

Application	Setting
Debugging	`debug`
Traceability	`trace`
Efficiency	`info`
Safety precaution	`info`

Programmatic Use

Parameter: `SLRTLogLevel`
Type: character vector
Value: `'info'` | `'trace'` | `'debug'` | `'warning'` | `'error'` | `'fatal'`
Default: `'info'`

Version History

Introduced in R2020b

See Also

Force polling mode | **Max file log runs** | **Compile with GCC -ffast-math** | "Simulink Real-Time Options Pane" on page 1-2 | `Application`

Force polling mode

Enables polling mode — instead of interrupt-driven mode — for clocking the real-time application

Model Configuration Pane: Code Generation / Simulink Real-Time Options

Description

Enables polling mode — instead of interrupt-driven mode — for clocking the real-time application. Polling mode can be useful for reducing sample time jitter. But, enabling this option causes the real-time application to consume a CPU core completely to clock and execute the base rate.

Dependencies

The `SLRTForcePollingMode` configuration parameter sets the initial value for the `pollingThreshold` option when you build the real-time application. Enabling `SLRTForcePollingMode` sets the `pollingThreshold` to a value above the base sample rate. This setting forces clocking the real-time application in polling mode.
Option: `pollingThreshold`

Settings

`off` | `on`

Default: `off`

`off`

When `Force polling mode` is disabled, the real-time application is clocked by a timer interrupt, unless the base sample rate is equal to or below the polling threshold (100 µs). If the base sample rate is less than or equal to the threshold, the real-time application is clocked in polling mode.

`on`

When `Force polling mode` is enabled, the real-time application is always clocked in polling mode.

Recommended Settings

Application	Setting
Debugging	`off`
Traceability	`off`
Efficiency	`off`
Safety precaution	`off`

Programmatic Use

Parameter: `SLRTForcePollingMode`
Type: character vector

Value: `'off'` | `'on'`
Default: `'off'`

Version History

Introduced in R2020b

See Also

Log level | **Max file log runs** | **Compile with GCC -ffast-math** | "Simulink Real-Time Options Pane" on page 1-2 | `Application`

Max file log runs

Selects the number of file log runs to retain for the real-time application

Model Configuration Pane: Code Generation / Simulink Real-Time Options

Description

Selects the number of file log runs to retain for the real-time application when logs are stored on the Speedgoat® target computer instead of uploaded to the MATLAB development computer after each simulation run. The logs are stored if auto-import is disabled, or the target is not connected to the host at stop time.

Note You can inadvertently delete existing file logs for an installed real-time application on the target computer if you use the `slrealtime.Application` function to change the Options for `FileLogMaxRuns` and then reload the application. To change the number of stored logs without deleting existing logs, load the real-time application and then change the FileLogMaxRuns option by using the `start(tg)` function.

Dependencies

The `SLRTFileLogMaxRuns` configuration parameter sets the initial value for the `fileLogMaxRuns` option when you build the real-time application.
Option: `fileLogMaxRuns`

Settings

1 | int

Default: 1

`int`

Select the number of file log runs to retain for the real-time application when logs are stored on the target computer instead of uploaded to the development computer after each simulation run.

Recommended Settings

Application	Setting
Debugging	1
Traceability	1
Efficiency	1
Safety precaution	1

Programmatic Use

Parameter: `SLRTFileLogMaxRuns`
Type: int

Value: int
Default: 1

Version History

Introduced in R2020b

See Also

Log level | **Force polling mode** | **Compile with GCC -ffast-math** | “Simulink Real-Time Options Pane” on page 1-2 | `Application`

Compile with GCC -ffast-math

Enables the GCC compiler `-ffast-math` option when compiling real-time application code

Model Configuration Pane: Code Generation / Simulink Real-Time Options

Description

Enables the GCC compiler `-ffast-math` option when compiling real-time application code. This option is disabled by default for Simulink Real-Time models.

By enabling the **Compile with GCC -ffast-math** option, you provide the compiler with more flexibility to optimize floating-point math at the expense of deviating from the IEEE-754 floating-point standard.

For more information about the `-ffast-math` option, see the Semantics of Floating-Point Math in GCC.

- gcc.gnu.org/wiki/FloatingPointMath/

Dependencies

None

Settings

`off` | `on`

Default: `off`

`off`

When `UseGCCFastMath` is disabled, Simulink Real-Time compiles real-time application code without the compiler `-ffast-math` option.

`on`

When `UseGCCFastMath` is enabled, Simulink Real-Time compiles real-time application code with the compiler `-ffast-math` option.

Recommended Settings

Application	Setting
Debugging	`on`
Traceability	`on`
Efficiency	`on`
Safety precaution	`off`

Programmatic Use

Parameter: `UseGCCFastMath`

Type: character vector
Value: `'off'` | `'on'`
Default: `'off'`

Version History

Introduced in R2020b

See Also

Log level | **Force polling mode** | **Max file log runs** | “Simulink Real-Time Options Pane” on page 1-2 | `Application`

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 RTOS.

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')
```

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

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

To remove these options, use the syntax:

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

Note At this time, no TLC options for Simulink Real-Time are supported.

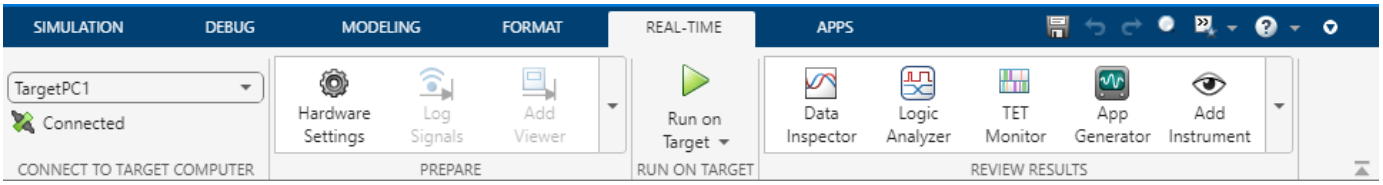
Apps

Simulink Real-Time

Generate real-time applications for simulations that run on a target computer and interface with I/O devices in the target computer

Description

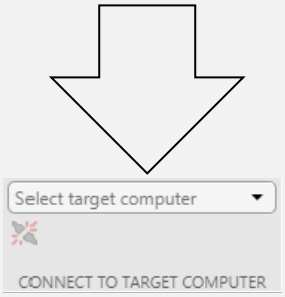
Use the **Simulink Real-Time** app to configure a model to build and run real-time applications on a Speedgoat target computer. The app configures the model to use the Simulink Real-Time code generation target and other configuration parameters for code generation. When you open the app, a **Real-Time** tab is added to the toolstrip. The **Real-Time** tab represents groups of tasks in the Simulink Real-Time workflow.



After you use the app to configure the model for Simulink Real-Time, you can perform these and more tasks from the **Real-Time** tab in the Simulink Editor.

Use the actions in the **Connect to Target Computer** section to select and connect to a target computer.

Connect to Target Computer Actions

UI Control	Description
 Select ta rget to acti vate Simulink Real-Time	From the target computers list on the **Real-Time** tab, select the target computer to which you want to connect. For more information, see the `connect` function.

Use the actions in the **Prepare** section to configure the model and tune parameters.

Prepare Actions

UI Control	Description
	Use the **Hardware Settings** button to configure model to run on target computer. The Configuration Parameters dialog box opens.
	Use the **Log Signals** button to send signal to the Simulation Data Inspector and workspace. Select one or more signals before using this button.
	Use the **Add Viewer** button to add a display of the selected signals. Select one or more signals before using this button.
	Use the **Test Point** button to allocate memory and make signals observable when using a Floating Scope. Select one or more signals before using this button.
	Use the **Signal Table** button to show a table that manages signal logging and viewing. The **Signal Table** tab opens at the bottom of the Simulink Editor.
	Use the **Configure Logging** button to configure signal logging. The **Data Inport/Export** tab of the Configuration Parameters dialog box opens.
	Use the **SLRT Explorer** button to open the Simulink Real-Time Explorer app. For more information, see Simulink Real-Time Explorer.
	Use the **Library Browser** to open the Simulink block library. The block library browser opens. See the Simulink Real-Time blocks and the Speedgoat I/O Blockset.
	Use the **Control Panel** button to launch the external mode control panel. The control panel opens.
	Use the **Connect Inputs** button to link sets of signals from files and workspace to root Inport blocks. The Root Inport Mapper opens.

UI Control	Description
	Use the **Hold Updates** button to communicate changes of multiple parameters at once. For more information, see *Tune Parameters by Using Hold Updates and Update All Parameters* in “Tune Parameters by Using Simulink® External Mode”.
	Use the **Update All Parameters** button to update all real-time application parameters on target computer. For more information, see *Tune Parameters by Using Hold Updates and Update All Parameters* in “Tune Parameters by Using Simulink® External Mode”.
Remove Hardware Configuration Remove hardware configuration from this model	Use the **Remove Hardware Configuration** selection to remove the Simulink Real-Time configuration from this model. The code generation target is set to `grt.tlc`.

Use the actions in the **Run on Target** section build the model, deploy the real-time application to the target computer, and run the real-time application.

Run on Target Actions

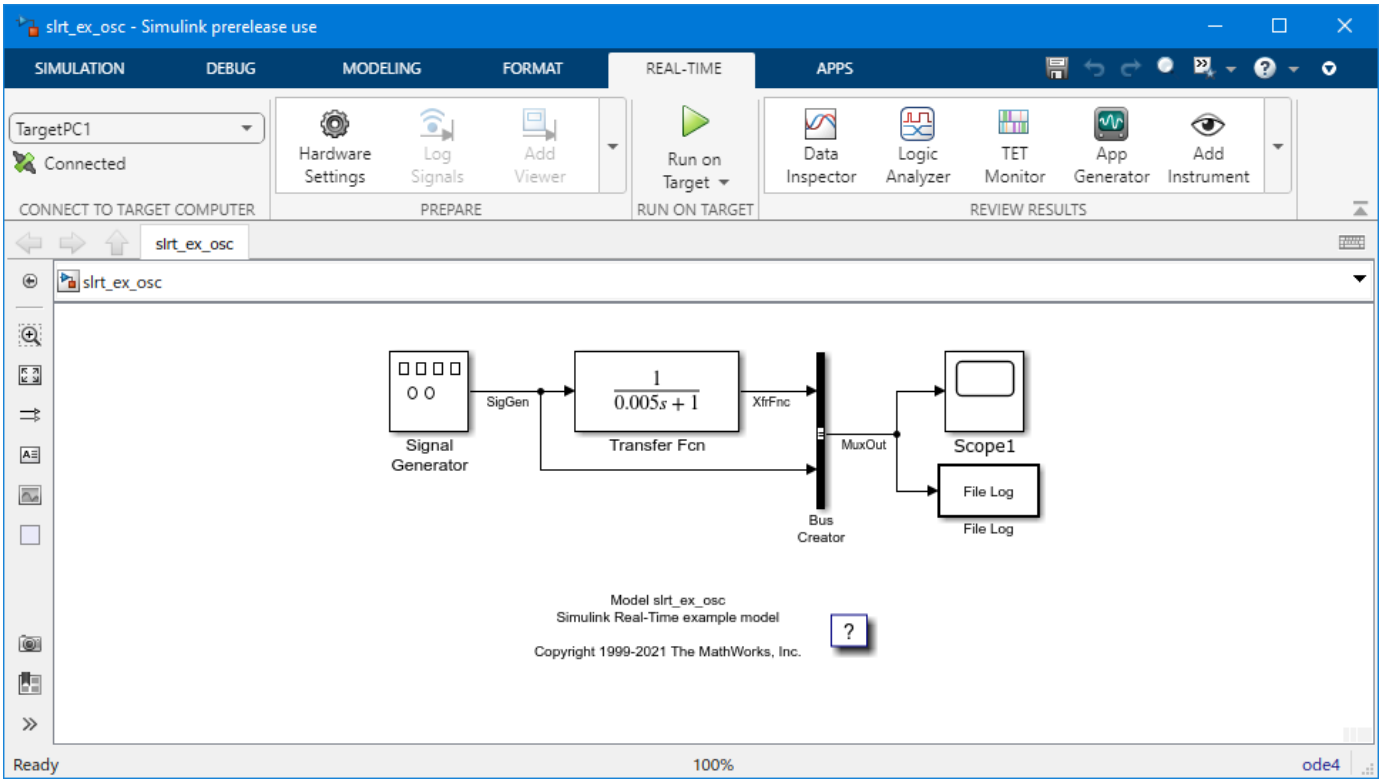
UI Control	Description
	Use the **Run on Target** button to start an application on target computer, observe outputs, and tun parameters. One-click builds and deploys real-time application when model changes are found. For more information, see “Build and Download Real-Time Application by Using Run on Target”.
	Use the **Build Application** button to generate a real-time application from a Simulink model. For more information, see “Execute Real-Time Application in Simulink External Mode by Using Step-by-Step Commands”.
	Use the **Deploy to Target** button to install a real-time application on a target computer. For more information, see “Execute Real-Time Application in Simulink External Mode by Using Step-by-Step Commands”.
	Use the **Connect to Model** button to connect a Simulink model to a real-time application on a target computer. For more information, see “Execute Real-Time Application in Simulink External Mode by Using Step-by-Step Commands”.
	Use the **Start Application** to start a real-time application. Observe output and tune parameters by using Simulink model. For more information, see “Execute Real-Time Application in Simulink External Mode by Using Step-by-Step Commands”.
	Use the **Restart Application** button to restart a real-time application on a target computer. For more information, see “Execute Real-Time Application in Simulink External Mode by Using Step-by-Step Commands”.
	Use the **Stop Application** button to stop a real-time application on a target computer. For more information, see “Execute Real-Time Application in Simulink External Mode by Using Step-by-Step Commands”.
	Use the **Disconnect Model** button to disconnect a Simulink model from a real-time application on a target computer. For more information, see “Execute Real-Time Application in Simulink External Mode by Using Step-by-Step Commands”.
AutoImportFileLog	Select the **AutoImportFileLog** checkbox to import the file log data on application stop. For more information, see the `AutoImportFileLog` argument for the `start` function or `stop` function. You also can select this option by using the **Run** or **Stop** button in Simulink Real-Time Explorer.

Use the actions in the **Review Results** section instrument the model and observe outputs.

Review Results Actions

UI Control	Description
	Use the **Data Inspector** button to view logged data in the Simulation Data Inspector.
	Use the **Start Recording** button to start signal logging and streaming. For more information, see `startRecording` function and “Real-Time Signal Logging and Streaming Basics”.
	Use the **Stop Recording** button to stop signal logging and streaming. For more information, see `stopRecording` function and “Real-Time Signal Logging and Streaming Basics”.
	Use the **Logic Analyzer** button to visualize, measure, and analyze transitions and states over time in the Logic Analyzer.
	Use the **TET Monitor button** to view real-time application task execution time. For more information, see Simulink Real-Time TET Monitor.
	Use the **App Generator** button to launch the App Generator to create an App Designer instrument panel. For more information, see Simulink Real-Time App Generator.
 	Use the **Add Instrument** button to select signals from the Simulink model for viewing in the Simulation Data Inspector. After adding and instrument to the model, the button label changes from **Add Instrument** to **Configure Instrument**. For more information, see “Add Instruments to Real-Time Application from Simulink Model”.
	Use the **Remove Instrument** button to stop recording selected signal values in the Simulation Data Inspector. For more information, see “Add Instruments to Real-Time Application from Simulink Model”.
	Use the **Highlight Instrument** button to highlight selected signals. For more information, see “Add Instruments to Real-Time Application from Simulink Model”.

UI Control	Description
	Use the **Import Instrument** button to import an instrument from a file. For more information, see “Add Instruments to Real-Time Application from Simulink Model”.
	Use the **Export Instrument** button to export an instrument to a file. For more information, see “Add Instruments to Real-Time Application from Simulink Model”.



Open the Simulink Real-Time App

In the **Apps** gallery, under **Real-Time Simulation and Testing**, click **Simulink Real-Time**. The **Real-Time** tab opens.

Examples

- “Tune Parameters by Using Simulink® External Mode”
- “Execute Real-Time Application in Simulink External Mode by Using Step-by-Step Commands”
- “Add Instruments to Real-Time Application from Simulink Model”

Version History

Introduced in R2020b

R2023a: UI enhancements for file log import options

In the Simulink Editor, on the **Real-Time** tab, you can enable the **AutoImportFileLog** check box from the **Run on Target** button, **Start Application** button, or **Stop Application** button. This check box corresponds to the `AutoImportFileLog` option of the `start` function and `stop` function.

R2022a: Added App Generator

The Simulink Real-Time App Generator creates App Designer instrument panels from the Simulink model or from generated application (MLDATX) files.

R2022a: AutoImportFileLog Available from Run on Target

The **AutoImportFileLog** check box is available from the **Run on Target button** on the **Real-Time** tab in the Simulink Editor. This checkbox corresponds to the `AutoImportFileLog` option of the start function.

R2021b: Bind mode for signals to instruments in Simulink Editor

You can bind signals to instruments (also referred to as instrumenting a signal) by using the **Add Instrument** button to enter bind mode in the Simulink Editor. You can select signals in the model and stream signal data for those signals from the real-time application to the Simulation Data Inspector.

R2021b: Changed Batch Mode Button Label to Hold Updates

On the **Real-Time** tab, the **Batch Mode** button label changed to **Hold Updates**.

See Also

Functions
`connect` | `start`

Apps
Simulink Real-Time Explorer | Simulink Real-Time TET Monitor | Simulink Real-Time App Generator

Topics
“Tune Parameters by Using Simulink® External Mode”
“Execute Real-Time Application in Simulink External Mode by Using Step-by-Step Commands”
“Add Instruments to Real-Time Application from Simulink Model”

Simulink Real-Time Explorer

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

Description

Simulink Real-Time Explorer provides a single point of contact for viewing connection status and interacting with a real-time application. You can visualize real-time signals, tune parameters, and stream real-time signals 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 Explorer for these tasks:

- Connect the MATLAB development computer and Speedgoat 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.
- View task execution time (TET).

For examples, click the links in the **More Information** column.

Target Computer Configuration

Capability	More Information
Configure target computer configuration settings. View target computer disk usage.	“Target Computer Settings”

Real-Time Application Access and Control

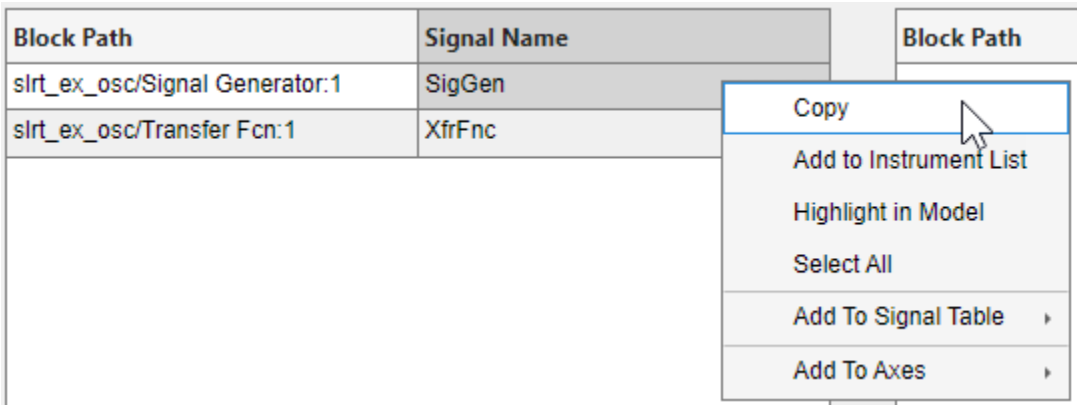
Capability	More Information
• Connect target computers to a development computer, and then disconnect them. • Load a prebuilt real-time application into a target computer. • Start and then stop running a real-time application that you downloaded to the target computer. • Display execution time, task execution time, and other properties of the real-time application. • Change stop time without regenerating code. • Start and stop signal logging and streaming by using **Start Recording** and **Stop Recording** buttons. • Import file logs from the target computer by using the **Import File Log** button.	• "Real-Time Application and Target Computer Modes" • "Configure and Control Real-Time Application by Using Simulink Real-Time Explorer" • "Real-Time Signal Logging and Streaming Basics"

Real-Time Application Management

Capability	More Information
By using **Configuration** tab or right-click menu in **Targets** tree: • View real-time applications available on target computer, including detailed properties. • Delete real-time applications from target computer. • Select a real-time application as startup application.	"Connect, Load Application, and Start" on page 3-15

Signal Access

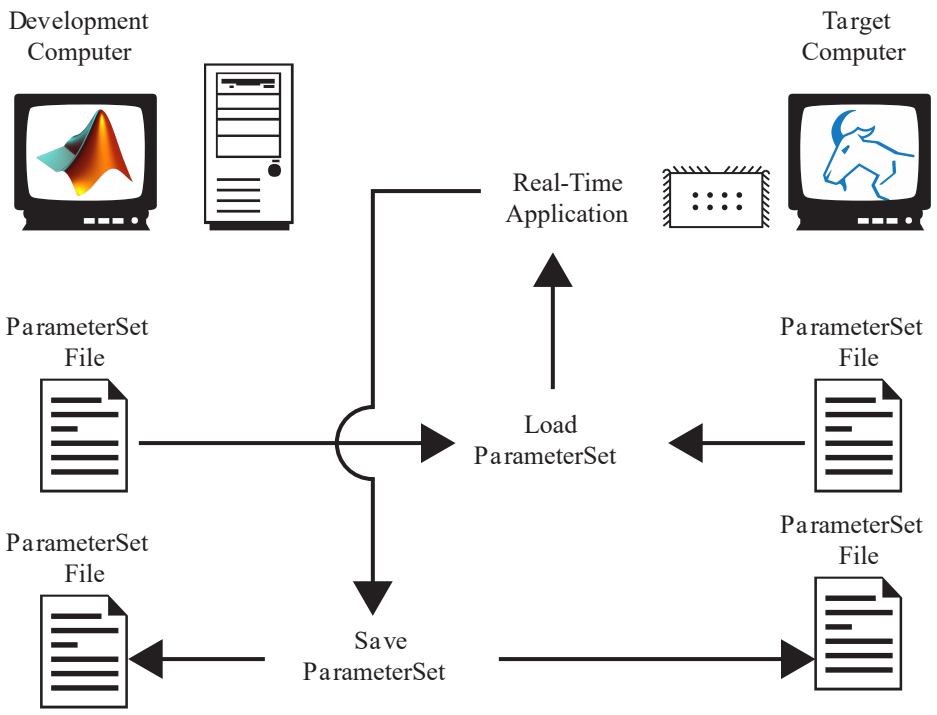
Capability	More Information
Filter and group hierarchical signals.	"Filter Hierarchical List of Signals and Parameters in Simulink Real-Time Explorer"
Visualize signals.	• "Stream Real-Time Signals by Using Simulink Real-Time Explorer" • "Add an Instrument to a Stateflow Subsystem" • "Customize Instrumentation Display" on page 3-19
Create, save, and load signal groups.	"Export and Import Signals in Instrument by Using Simulink Real-Time Explorer"
Copy signal block paths, add to instrument list, highlight in model, add to signal table, or add to axes.	See "Simulink Real-Time Explorer, Signals Tab, Signal Context Menu" on page 3-11.



Simulink Real-Time Explorer, Signals Tab, Signal Context Menu

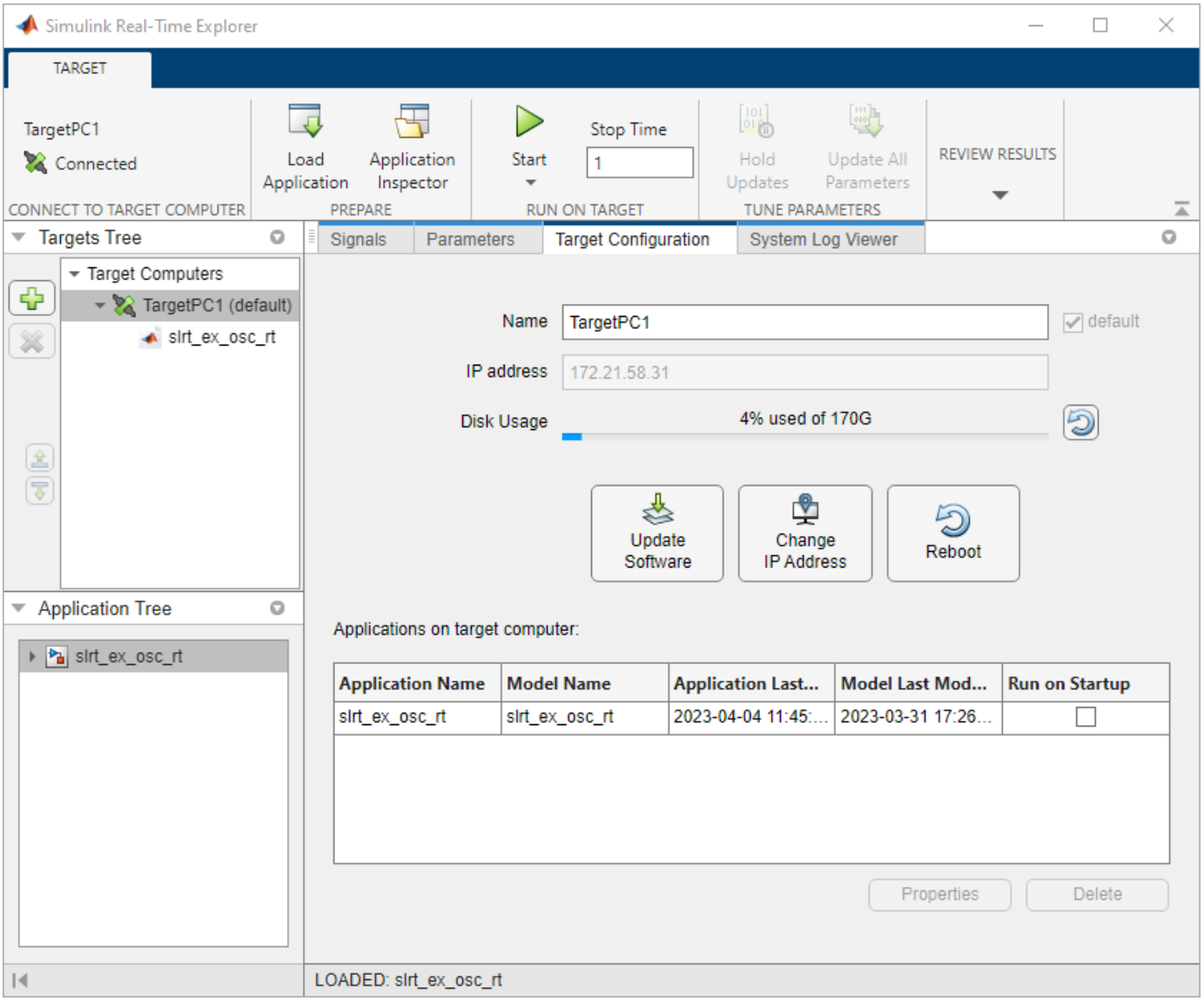
Parameter Tuning

Capability	More Information
Filter and group hierarchical parameters.	"Filter Hierarchical List of Signals and Parameters in Simulink Real-Time Explorer"
Display and tune parameter values while the real-time application is running.	• "Tune Parameters by Using Simulink® Real-Time™ Explorer" • "Customize Instrumentation Display" on page 3-19
When the ECU page and XCP page selections do not match, the mismatch disables the explorer **Parameter** table. You can enable parameter table operation by coordinating ECU page and XCP page selection in the real-time application. Use the explorer **Enable Parameter Table** button. This button is context sensitive and appears when explorer detects a page selection mismatch.	• `copyPage` • `getECUPage` • `getNumPages` • `getXCPPage` • `setECUAndXCPPage` • `setECUPage` • `setXCPPage`
Refresh cached parameter table values by clicking the **Refresh Values** button.	Use the **Refresh Values** button for instances in which the parameter table data becomes disabled (for example when page switching occurs),
Use the **Hold Updates** button and **Update All Parameters** button to change multiple parameter values simultaneously. These buttons in Explorer operate in the same way as these buttons on the **Real-Time** tab in the Simulink Editor.	*Tune Parameters by Using Hold Updates and Update All Parameters* in "Tune Parameters by Using Simulink® External Mode"
Use the **Save Param Set** or **Load Param Set** buttons on the **Parameters** tab to save or load a parameter set file for the current real-time application. You can save or load the parameter set file from the development computer or target computer. See the figure.	"Save and Reload Parameters by Using Simulink Real-Time Explorer"



Observe Task Execution Time and Target Computer Status

Capability	More Information
Open the **TET Monitor** tab and monitor task execution time.	• Simulink Real-Time TET Monitor • "Customize Instrumentation Display" on page 3-19
Open the **System Log Viewer** tab and observe the target computer system messages.	• `slrtLogViewer` • "Target Computer Status Monitor" on page 4-2
Open the **System Log Viewer** tab and use the log controls to filter messages, clear the display, clear the messages, or retrieve messages from the target computer.	• "Filter, Clear, or Retrieve System Messages" on page 3-17



Open the Simulink Real-Time Explorer App

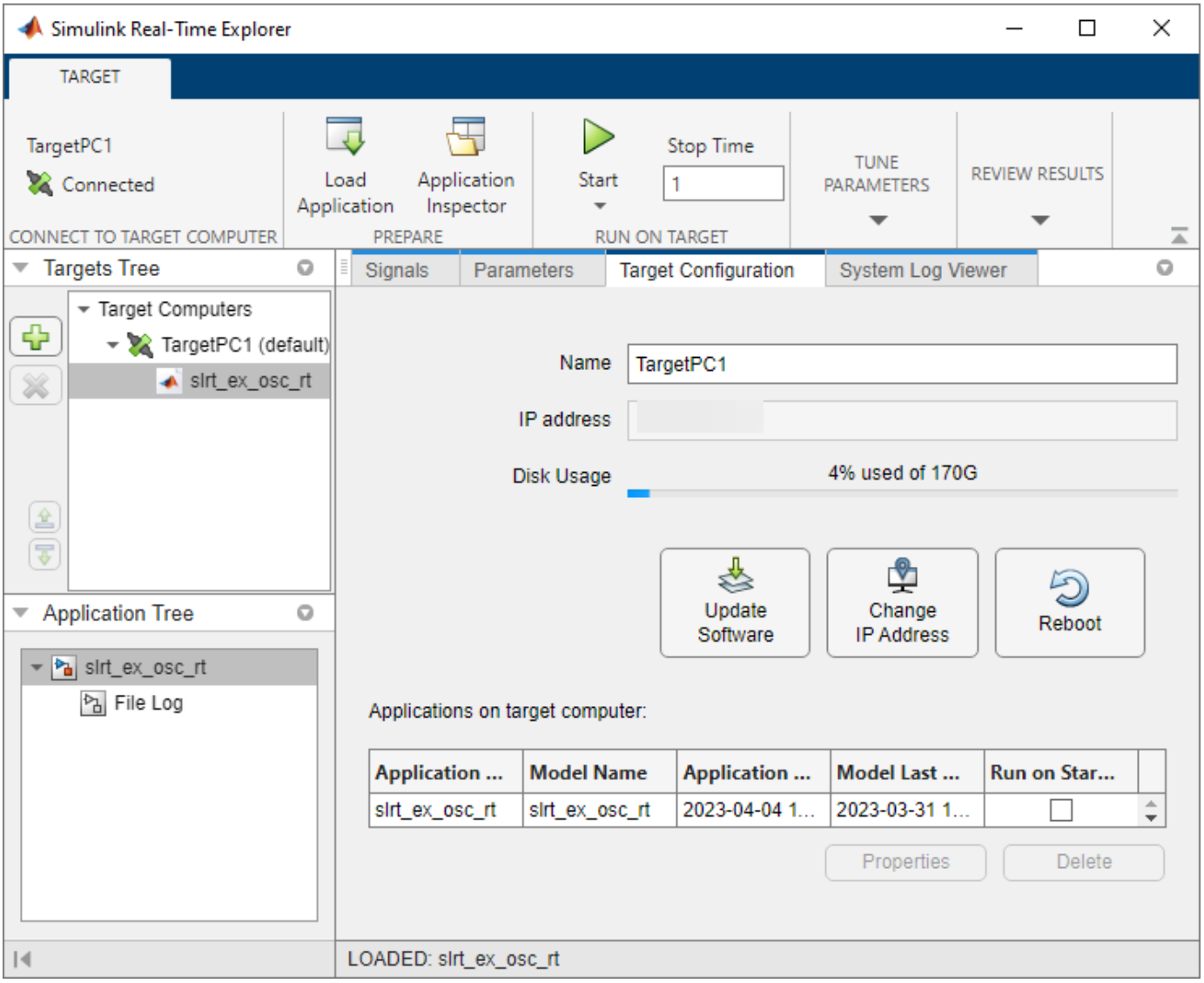
From the Simulink Editor, in the **Real-Time** tab, select **Prepare** > **SLRT Explorer**. Or, from the MATLAB Command Window, type:

```
slrtExplorer
```

Examples

Configure Target, Update, and Reboot

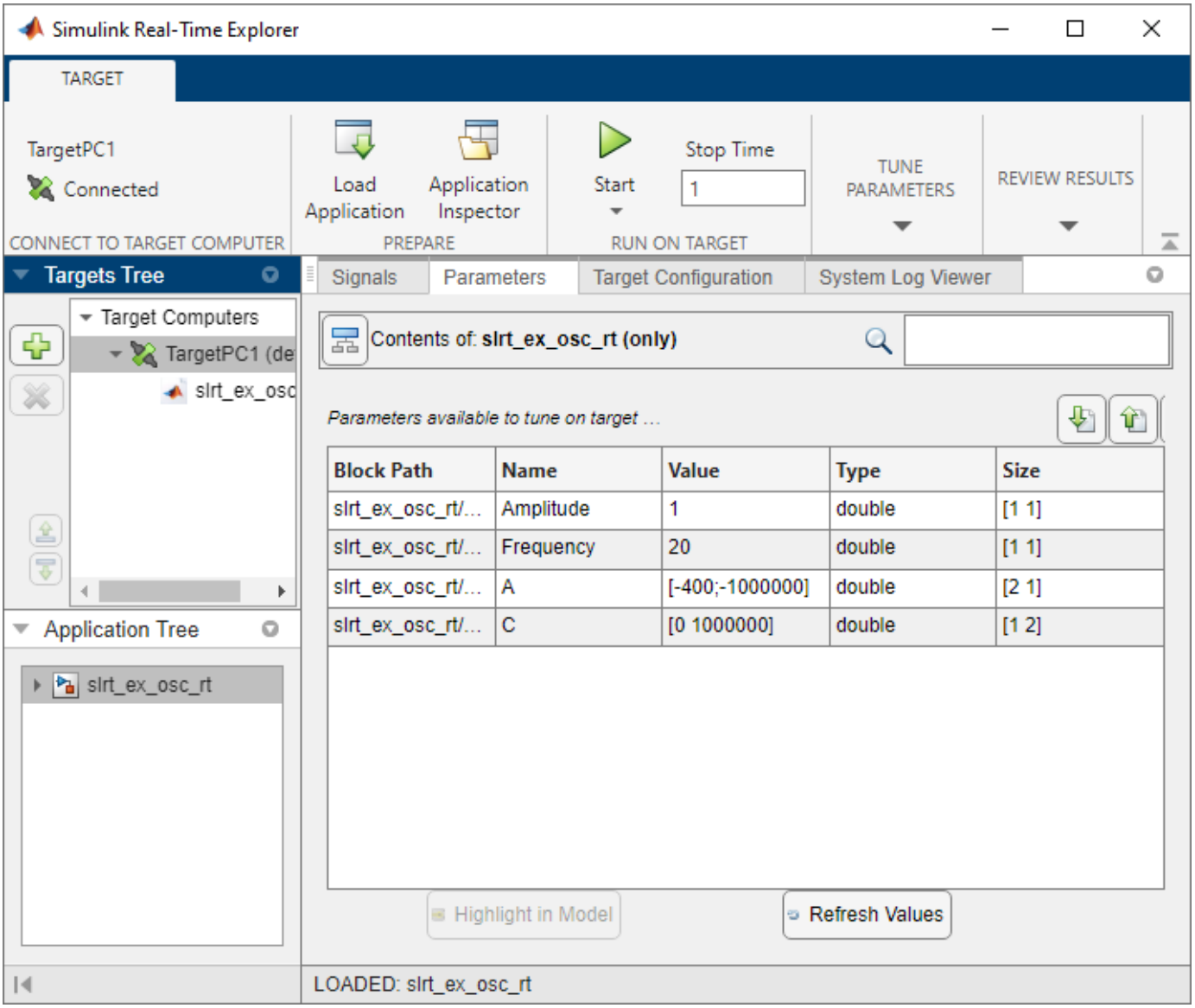
This example shows how to change the IP address of the target computer, update the target computer software, and reboot the target computer.



1. Open the Simulink Real-Time Explorer.
2. Select the target computer in the **Targets Tree** panel.
3. To change the IP address of the target computer, click the **Change IP Address** button.
4. To update the target computer software, click the **Update Software** button.
5. To reboot the target computer, click the **Reboot** button.

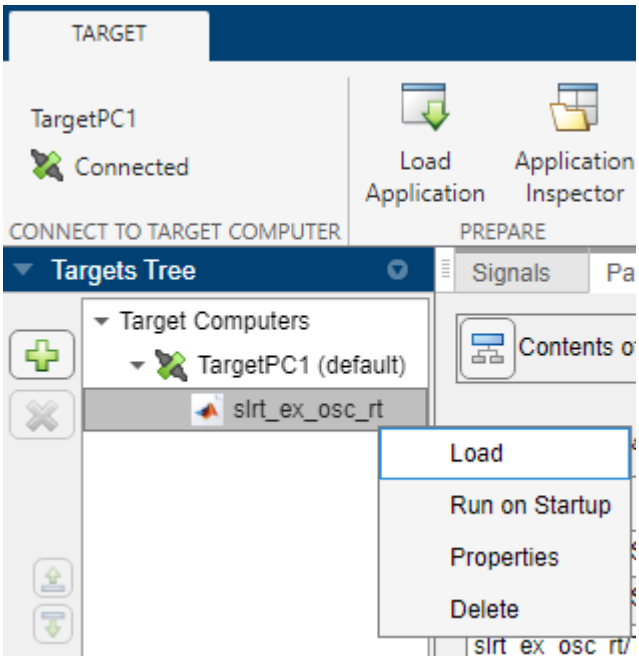
Connect, Load Application, and Start

This example shows how to connect to the target computer, load the real-time application, set the stop time, and start the real-time application.



1. Open the Simulink Real-Time Explorer.
2. Select the target computer in the **Targets Tree** panel.
3. To connect to the target computer if not already connected, click **Disconnected** toggling it to **Connected**.
4. To select and load a real-time application, click **Load Application** and select the MLDATX file.

Note You can select and load a real-time application by using the context menu. Right-click on the application and select **Load**.



5 To select the application stop time, type a value (in seconds) in the **Stop time** field.

6 To start the application, click the **Start** button.

In Explorer, clicking the **Start** button is equivalent to executing this command for target object `tg`:

```
start(tg, 'ReloadOnStop', true, 'AutoImportFileLog', true)
```

Note To change the `ReloadOnStop` and `AutoImportFileLog` operation of the **Start** button, you can:

- Select **Start** > **ReloadOnStop**
- Select **Start** > **AutoImportFileLog**.

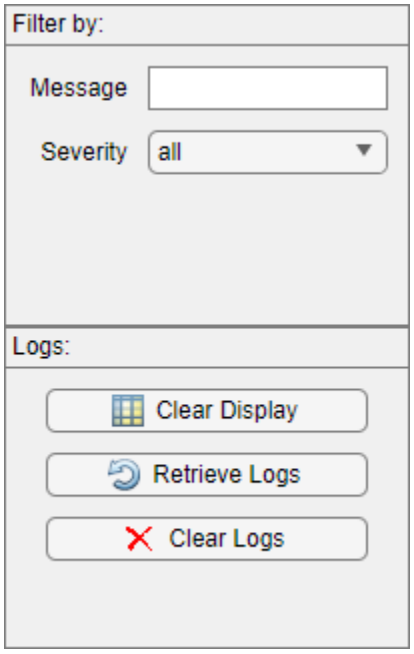
Note To change the `AutoImportFileLog` operation of the **Stop** button, you can:

- Select **Stop** > **AutoImportFileLog**.

7 To stop the real-time application, click the **Stop** button.

Filter, Clear, or Retrieve System Messages

This example shows how to filter the messages displayed by the **System Log Viewer** tab. You also can clear the log display in explorer, clear the log messages from explorer, or retrieve the log messages from the target computer. The tab provides filter selections for message severity and buttons that let you **Clear Display**, **Retrieve Logs**, or **Clear Logs**. These operations also are available in a right-click context menu for the message area of the tab.



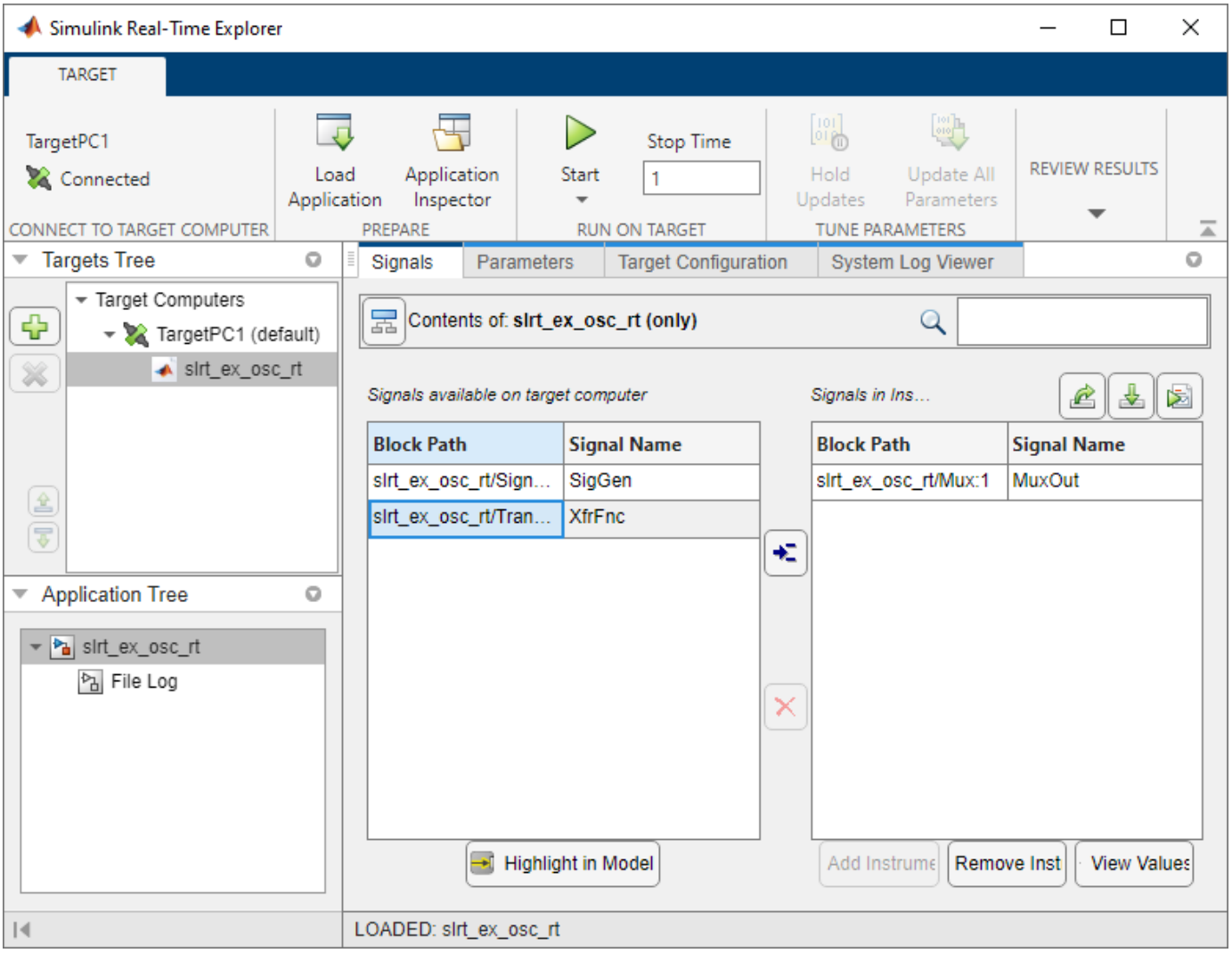
1. Open the Simulink Real-Time Explorer.
2. Connect to a target computer. Load and run a real-time application. For more information, see “Connect, Load Application, and Start” on page 3-15.
3. Select the **System Log Viewer** tab.
4. Use the tab controls to filter the system log message display by message severity.

   While the target computer status is `'loaded'`, `'loading'`, or `'running'`, you can clear the display of log messages or retrieve log messages from the target computer.

   While the target computer status is `'stopped'`, you can clear the log messages from the **System Log Viewer**.
5. Right click on a message in the **System Log Viewer**, and use the right-click context menu to clear the display, retrieve logs, or clear logs.

Select Signals and Stream Data to the 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.



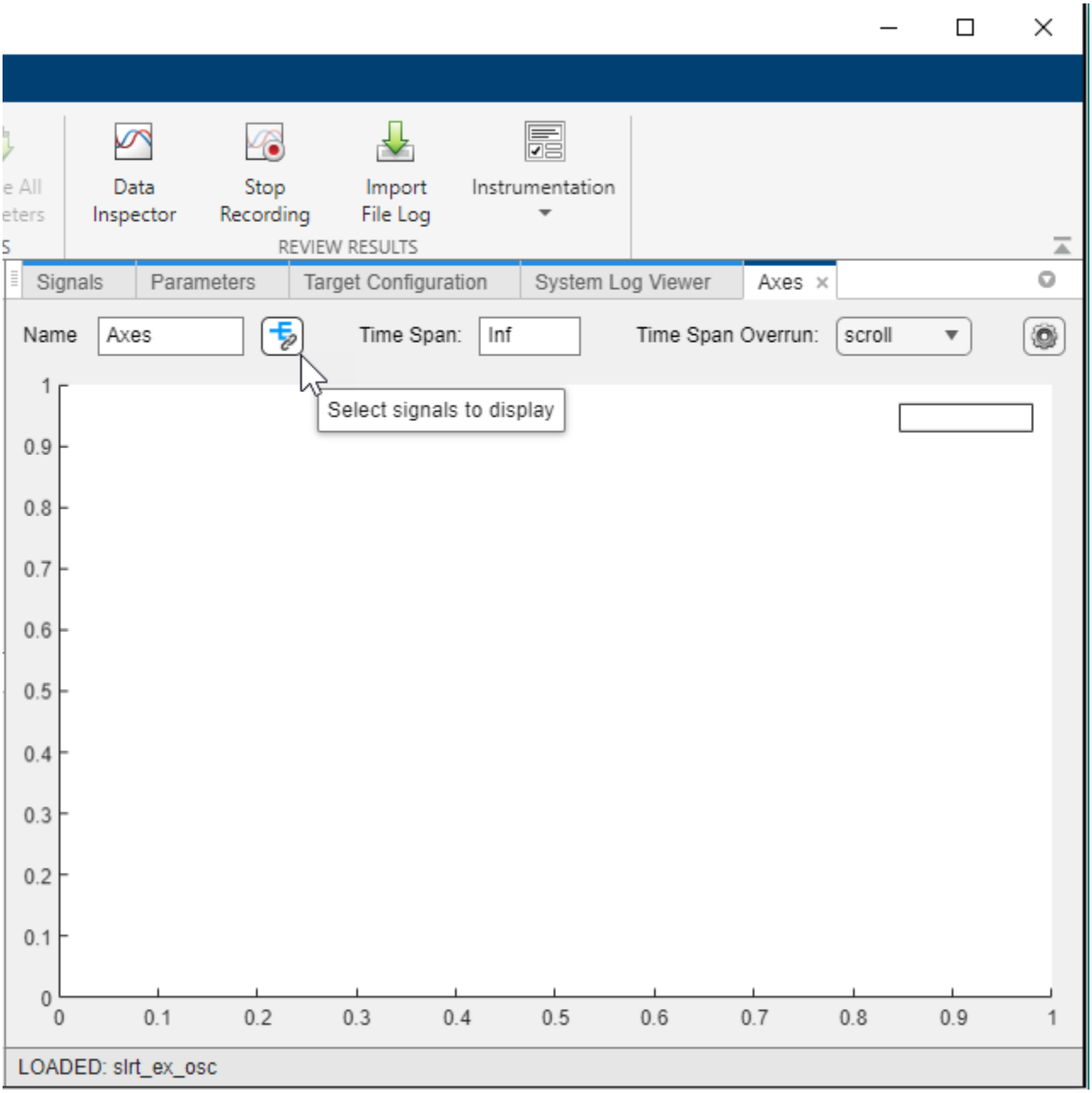
1. Open the Simulink Real-Time Explorer.
2. To connect to the target computer if not already connected, click **Disconnected** toggling it to **Connected**.
3. To select and load a real-time application, click **Load Application** and select the MLDATX file.
4. To select signals for streaming, click the application name, select signals from the **Signals** tab, and click the **Add selected signals** button.
5. To run the application and generate data for streaming, click the **Start** button.
6. To stream the signal data, select the signals in the **Signals in instrument** list and click the **Add Instrument** button.
7. To view the streaming signals, click the **Data Inspector** button.
8. To stop streaming and logging signal data, click the **Stop Recording** button. This button also stops signal logging.
9. After viewing the data, to stop the real-time application, click the **Stop** button.

Customize Instrumentation Display

To customize the instrumentation display in explorer, use the **Instrumentation** button to:

- Add signal tables, parameter tables, and axes as custom instrumentation tabs in explorer.
- Bind signals and parameters to the custom instrumentation tabs by using the **Signal Selector** button and the **Parameter Selector** button.
- Configure the display of signals that you bind to custom axes.
- Add a task execution time (TET) monitor tab to explorer.
- Export or import custom instrumentation display configurations in explorer.

This example shows how to add and configure custom instrumentation (an axes display).



1 Open the Simulink Real-Time Explorer.

2. Select the target computer in the **Targets Tree** panel.
3. To connect to the target computer if not already connected, click **Disconnected** toggling it to **Connected**.
4. To select and load a real-time application, click **Load Application** and select the MLDATX file.
5. To add an **Axes** tab, pull down the options for the **Instrumentation** button and select the **Axes** button.
6. Use the controls in the **Axes** tab to name the tab and set features for the axes. Bind signals to the axes by using the **Signal Selector** button.
7. To export the custom instrumentation configuration, pull down the options for the **Instrumentation** button and select the **Export Configuration** button.
8. To import the custom instrumentation configuration in another explorer session, pull down the options for the **Instrumentation** button and select the **Import Configuration** button.

Programmatic Use

`slrtExplorer` opens the Simulink Real-Time Explorer. Operations in the Simulink Real-Time Explorer UI correspond to Simulink Real-Time commands. For example, the explorer **Start** button corresponds to the `start` function.

Version History

Introduced in R2020b

R2024a: Filter, clear, or retrieve messages in system log viewer

You can use controls in the **System Log Viewer** tab of Simulink Real-Time Explorer to filter message display by severity, clear the message display, clear message logs, or retrieve message logs. A right-click on the message area opens a context menu with these controls.

Added color-coding by severity for messages makes the message display in explorer and on the target computer easier to read. Messages with log level `'Error'` or `'Fatal'` appear in red text. Messages with log level `'Warning'` appear in yellow text. Messages with log level `'Trace'`, `'Debug'`, or `'Info'` appear in black text.

R2024a: Apply line style properties in signal selector

You can apply line style properties to signals that you add to Axes in Simulink Real-Time Explorer. To apply these properties, click the edit icon in the **Line Style** column of the signal in the **Signal Selector** dialog box.

R2023b: Customize Instrumentation and use Signals context menu

In R2023b, the **Instrumentation** button lets you add signal table, parameter tables, and axes to the tabs in Simulink Real-Time Explorer. You also can export and import your instrumentation configuration. For more information, see “Customize Instrumentation Display” on page 3-19.

In R2023b, the **Signals** context menu is available when you select a signal in the Signals tab and right-click on the signal. From this menu, you can copy signal block paths, add to instrument list,

highlight in model, add to signal table, or add to axes. For more information, see “Simulink Real-Time Explorer, Signals Tab, Signal Context Menu” on page 3-11.

R2023b: Import previous file log while recording

In R2023b, you can import a previously stored file log while recording file log data. The change to Simulink Real-Time Explorer is that the **Import File Log** button remains enabled while recording is active. You also can use the file log `import` function while recording file log data.

R2023a: Parameter Set Support

The Simulink Real-Time Explorer supports save or load of a parameter set to or from the development computer or target computer. This support is similar to the operation of the `loadParamSet` and `saveParamSet` functions.

R2023a: UI enhancements for file log import options

In Simulink Real-Time Explorer, you can enable the **AutoImportFileLog** check box from the **Run** button or **Stop** button. This check box corresponds to the `AutoImportFileLog` option of the `start` function and `stop` function.

R2022b: Added Recording Buttons and Import File Log Button

Added **Start Recording** button and **Stop Recording** button to the **Real-Time** tab in the Simulink Editor and in the Simulink Real-Time Explorer. The **Start Recording** button and **Stop Recording** button have the same functionality as the `startRecording` function and `stopRecording` function.

R2022a: Parameter Table Caching and Parameter Tuning

The **Parameters** tab supports caching parameter table data. By caching the data, updates to parameter data in the table is improved. This improvement is noticeable for real-time applications that have a substantial number of parameters.

The **Parameters** tab supports a **Hold Updates** button and **Update All Parameters** button to change multiple parameter values simultaneously.

R2021b: Disk Usage Display

The target computer disk usage appears on the **Target Configuration** tab and appears on the target computer status monitor.

R2021a: Added Right-Click Menu for Application Options and Add Options to Start Button

The Explorer provides access to real-time application actions and properties through an application context menu and access to real-time application start options through the Start button. When you right-click an application name in the **Targets Tree**, the Explorer displays a menu of actions for the application. The application **Start** button provides `ReloadOnStop` and `AutoImportFileLog` options.

See Also

`slrtExplorer` | `slrtLogViewer` | `slrtTETMonitor`

Topics

“Target Computer Settings”
“Real-Time Application and Target Computer Modes”
“Configure and Control Real-Time Application by Using Simulink Real-Time Explorer”
“Filter Hierarchical List of Signals and Parameters in Simulink Real-Time Explorer”
“Stream Real-Time Signals by Using Simulink Real-Time Explorer”
“Export and Import Signals in Instrument by Using Simulink Real-Time Explorer”
“Tune Parameters by Using Simulink® Real-Time™ Explorer”

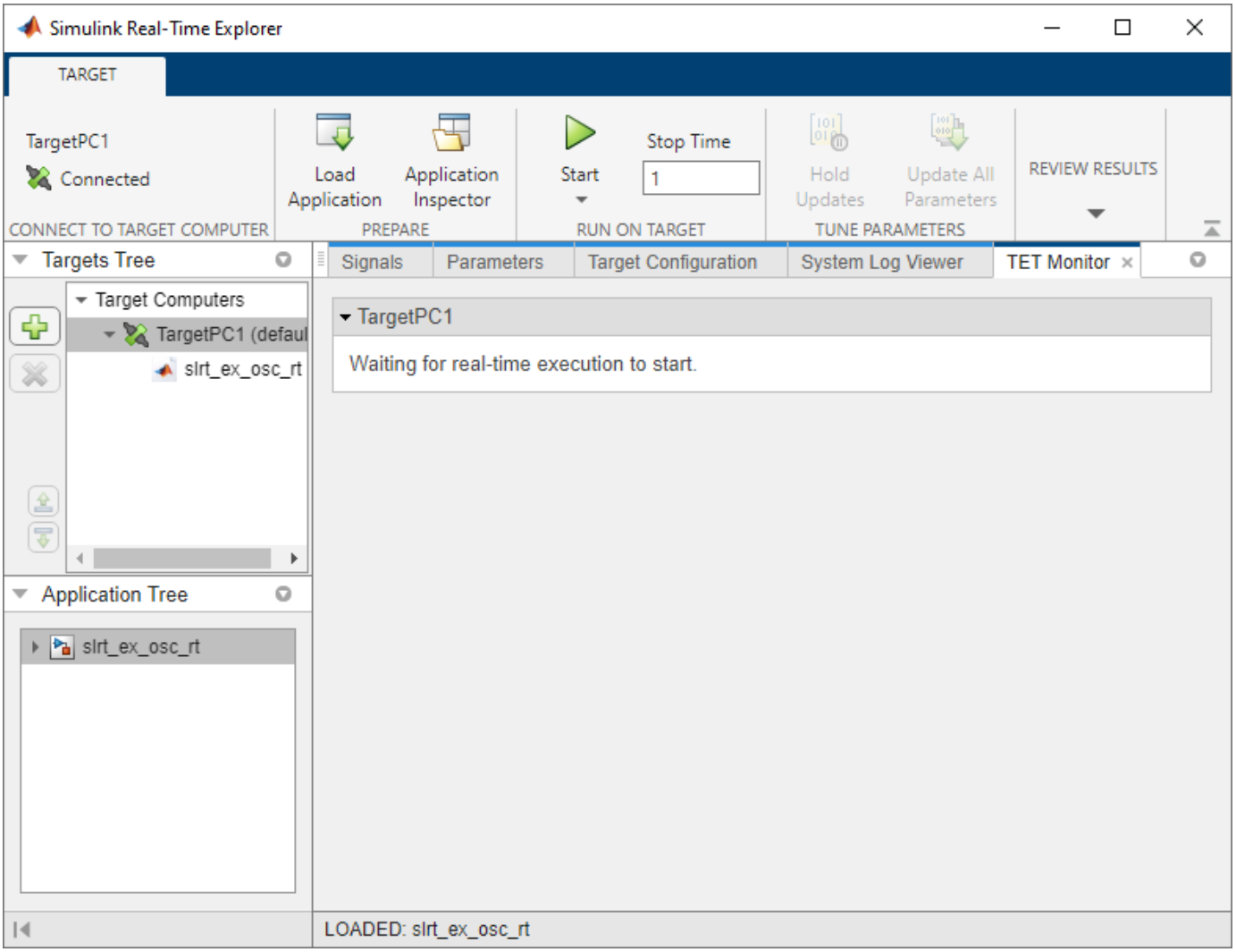
Simulink Real-Time TET Monitor

Observe task execution time for the real-time application running on target computer

Description

Simulink Real-Time Task Execution Time (TET) Monitor lets you view the task execution time for the real-time application running on the Speedgoat target computer.

You can open the TET monitor at any time. Depending on the current state of connected target computers, the monitor displays TET data for each real-time application task. Changes to the target computer state are updated in the TET monitor.



Open the Simulink Real-Time TET Monitor App

From the Simulink Editor, in the **Real-Time** tab, select **TET Monitor**. Or, from the MATLAB Command Window, type:

```
slrtTETMonitor
```

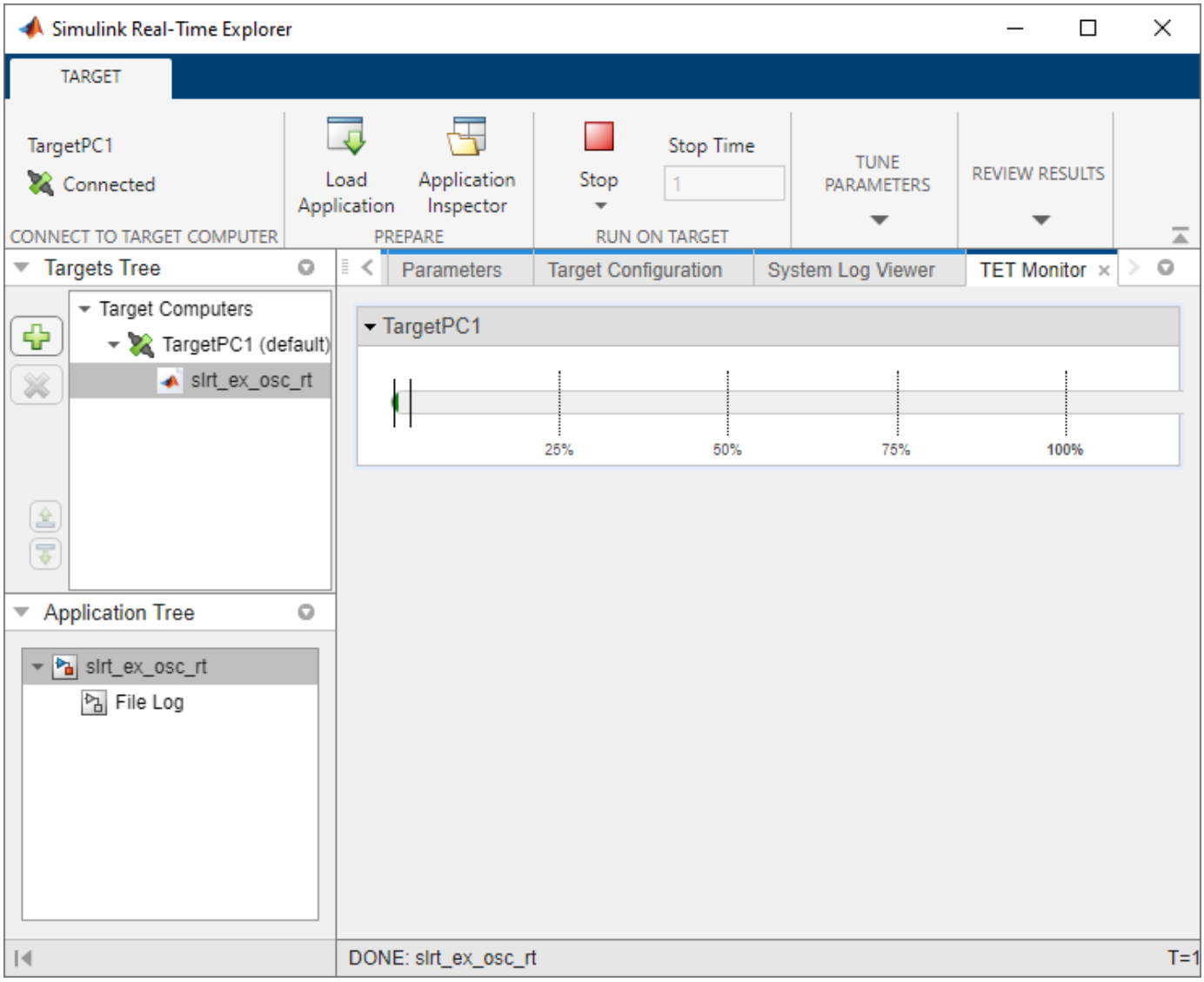
Examples

Open TET Monitor and View Status

In the “Post-Process Real-time Signals Streamed to the Simulation Data Inspector” example, use these additional steps to display the TET monitor.

1. Open the `slrt_ex_osc` model.
2. Build the real-time application, load it on the target computer, and start the application. In Simulink Editor **Real-Time** tab, click **Run on Target**.
3. Open the TET monitor. In the **Real-Time** tab, click **TET Monitor**. Or, in the Command Window, enter:

   ```
   slrtTETMonitor
   ```
4. When you run the real-time application, the TET monitor displays status.



Programmatic Use

`slrtTETMonitor` opens the Simulink Real-Time TET Monitor.

Version History

Introduced in R2020b

R2021b: Added TET Monitor to Explorer

The TET monitor appears as a tab in Simulink Real-Time Explorer instead of operating as a separate tool. You can open this tab in Simulink Real-Time explorer by using the **TET Monitor** button or by using the `slrtTETMonitor` function to open this tab.

See Also

`slrtTETMonitor` | SLRT Overload Options | `slrtExplorer`

Topics

“Post-Process Real-time Signals Streamed to the Simulation Data Inspector”
“Real-Time Signal Logging and Parameter Tuning”
“Real-Time Application and Target Computer Modes”
“Configure and Control Real-Time Application by Using Simulink Real-Time Explorer”
“Execution Profiling for Real-Time Applications”

Simulink Real-Time App Generator

Generate instrument panel app to interact with target computer and real-time application running on target computer

Description

Simulink Real-Time App Generator helps you generate an instrument panel app that interacts with the Speedgoat target computer and real-time application running on the target computer. You can select signals and parameters in your model to represent as instrument panel controls and configure the controls before generating the app.

To use the Simulink Real-Time App Generator, open the App Generator from the **Real-Time** tab in the Simulink Editor and use the App Generator for these tasks:

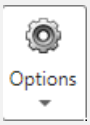
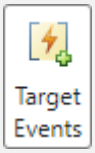
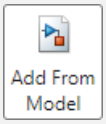
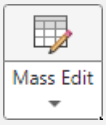
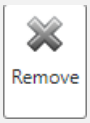
- Open a model file SLX or real-time application file MLDATX, and create an instrument panel app.
- Select signals and parameters to add to an instrument panel app.
- Configure controls for instrument panel app.
- Validate instrument bindings against real-time application MLDATX file.
- Create instrument panel app.
- Save an App Generator session file MAT, and open it in a future App Generator session.

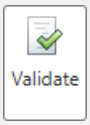
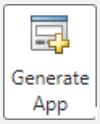
By using the parts of the Simulink Real-Time App Generator, you can instrument signals and parameters from your real-time application, binding them to controls in a generated instrument panel. The generator lets you configure options for generating the instrument panel and configure the properties of the instrument panel controls. Use these parts of the App Generator to configure and generate an instrument panel:

- “Designer Toolstrip Operations” on page 3-28
- “Signals and Parameters Pane Operations” on page 3-30
- “Bindings Tab Operations” on page 3-31
- “Property Panel Operations” on page 3-31

Designer Toolstrip Operations

The App Generator controls on the **Designer** toolstrip let you add, remove, edit, and validate signal and parameter bindings between the real-time application to controls in the generated instrument panel app.

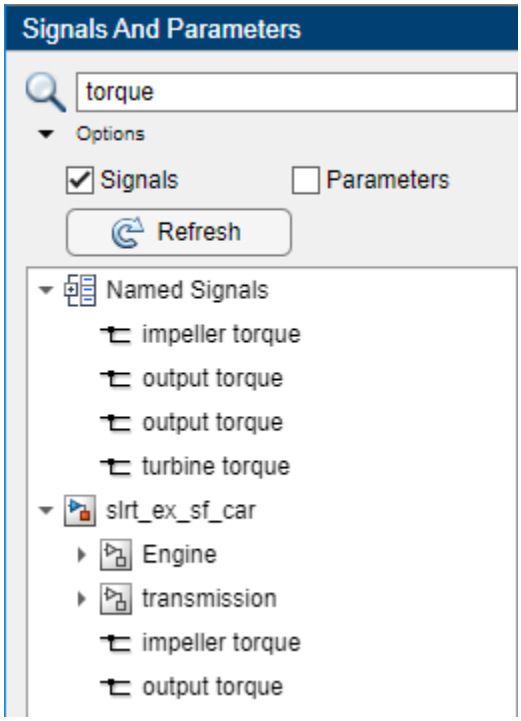
Controls	App Generator Operations
	Use the **Options** button pull down menu to configure options for instrument panel app generation. The options include: • **Toolstrip** — Provide target computer controls at top of instrument panel. • **Menu** — Provide target computer controls in a menu on the instrument panel. • **Status bar** — Add status bar to the instrument panel. • **TET Monitor** — Add task execution time monitor to the instrument panel. • **Instrumented signals** — Include a plot of instrumented signals on a set of axes in the instrument panel. When you enable this option, all instrumented signals are included on a single axes. This all-signals axes control is not configurable. You also can add axes controls for individual signals. The individual-signal axes controls are configurable in the property panel. See “Property Panel Operations” on page 3-31. Instrumented nonvirtual bus signals are not included on the axes generated by the **Instrumented signals** option. You can add individual axes for nonvirtual bus signals. • **Dashboard blocks** — Import dashboard blocks from model to the instrument panel. • **Use Grid Layout** — Arrange components using grids on the instrument panel. • **Callback** — Process signals before displaying on the instrument panel. • **Configure Components** — Configure options for Simulink Real-Time App Designer components and select name for instrument panel app. • **Settings** — Enable automatic validation of instrument-to-control bindings during app generation.
	Use the **Target Events** button to add callback code for target computer events. The App Generator adds the callback code for events when generating the app, and you can edit the callback code further by using the **Code View** in the App Designer. For more information, see “Add Callbacks for Target Computer Events” on page 3-36.
	Use the **Add From Model** button to open a model in bind mode and use the model to select signals and parameters for the generated instrument panel. The App Generator Bind Mode message box remains active until you close bind mode for the model in the Simulink Editor.
	When you select a single signal or a parameter from the **Bindings** tab, you can highlight that selected item in the model. If you select a workspace variable from the Bindings tab, clicking the **Highlight in Model** button opens the model base workspace in Model Explorer.
	When you select multiple signals or parameters from the **Bindings** tab, you can use the **Mass Edit** button to edit the instrument binding configurations for the selected items. You can change the control type, change the control name, and select whether to make the control name unique.
	Use the **Remove** button to delete the selected instrument bindings from the **Bindings** tab for instrument panel generation.

Controls	App Generator Operations
	For the real-time application MLDATX file, use the **Validate** button to check the instrument binding configurations in the **Bindings** tab. If the MLDATX file is not available, the App Generator issues an error.
	Use the **Generate App** button to create an App Designer instrument panel for the real-time application by using the instrument binding configurations from the **Bindings** tab and by using the selected instrument panel options from the **Options** button.
	After generating an App Designer instrument panel MLAPP file, you can add instrument bindings with their controls to the instrument panel. To enable the **Add to App** button: **1** Add a signal or parameter to the **Bindings** tab. **2** Select the added signal or parameter in the **Bindings** tab. **3** Click the **Add to App** button and select the MLAPP file for the instrument panel to which you would like to add the control. **Tip** The **Add to App** workflow helps you add instrument bindings with their controls to a generated instrument panel. This approach lets you add to a generated instrument panel that you have customized in App Designer.

Signals and Parameters Pane Operations

The Signals and Parameters pane helps you instrument the signals, parameters, and data in a real-time application. The tree hierarchy in the pane divides into **Model and External Data**, **Named Signals**, and the real-time application model signals and parameters. To add signals, parameters, or data to the instrument panel, select items from the **Signals And Parameters** pane, and click the **Add selected** button.

At the top of this pane, the filter box and filter options help you search for signals and parameters. The figure shows a search for signals in the `slrt_ex_sf_car` model that contain the text `torque`.

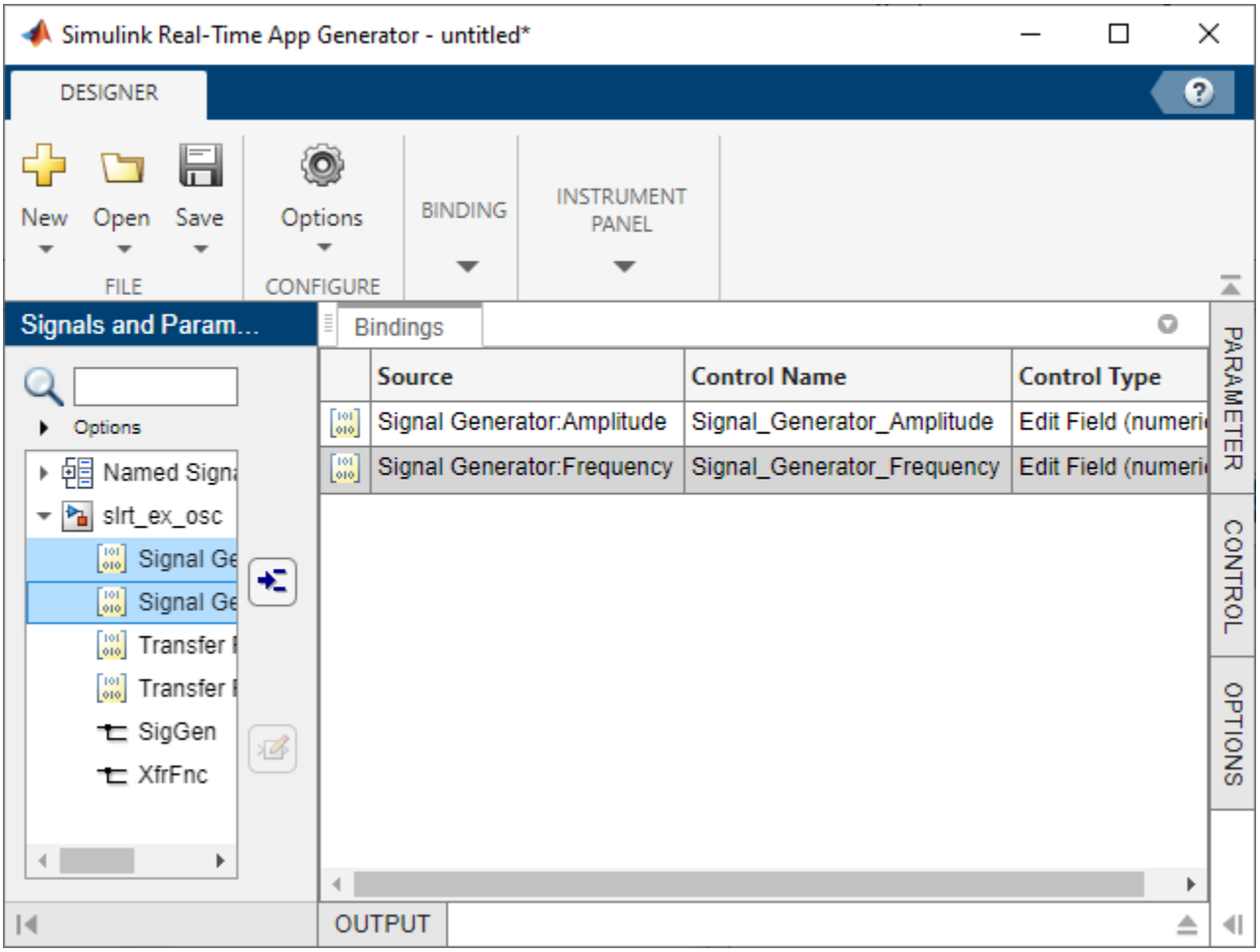


Bindings Tab Operations

When you select a signal, parameter, or variable in the **Bindings** tab, the **Parameter** or **Signal** configuration pane opens. This pane lets you configure the App Designer control in the instrument panel for the selected item. If you select multiple signals or parameters, you can use the **Mass Edit** button to configure the controls for the group of selected items.

Property Panel Operations

When you select a signal, parameter, or variable in the **Bindings** tab, the **Parameter** or **Signal** property panel opens. This panel lets you configure the App Designer control in the instrument panel for the selected item. You can change the displayed fields in the **Parameter** or **Signal** property panel by using the selection for the **Control Type** field.



Open the Simulink Real-Time App Generator App

From the Simulink Editor, in the **Real-Time** tab, select **Review Results** > **App Generator**. Or, in the MATLAB Command Window, type:

```
slrtAppGenerator
```

If you open the App Generator from a model, the App Generator populates the **Signals and Parameters** pane with information from the model.

Examples

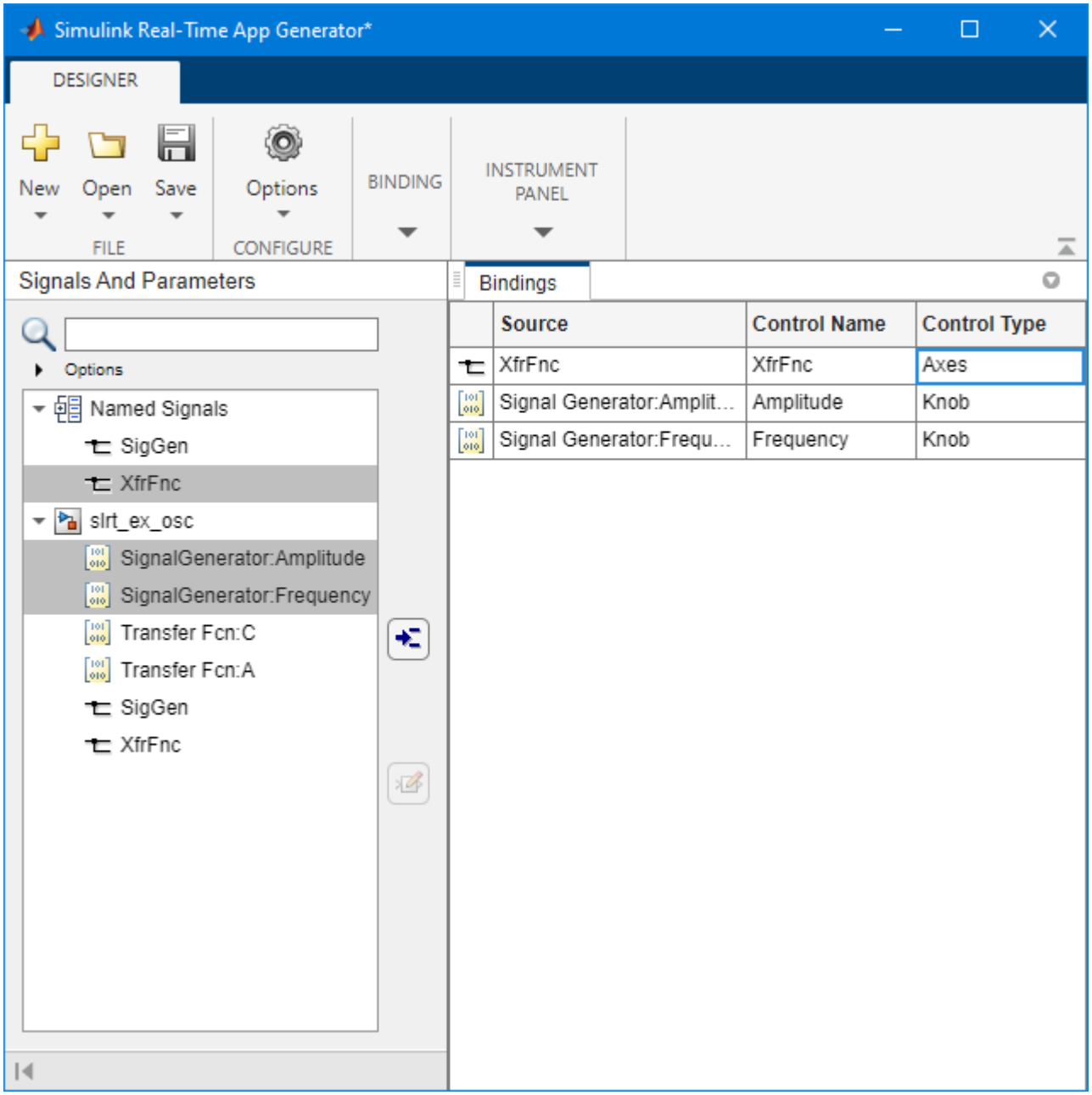
Configure Instrument Panel Controls and Create App

This example shows how to configure instrument panel controls for signals and parameter, then create an instrument panel app by using the App Generator. This example uses example model `slrt_ex_osc`.

1. Open example model `slrt_ex_osc`. In the MATLAB Command Window, type:

   ```
   openExample('slrealtime/SlrtAddIOBlocksToSimulinkModelExample', ...
       'supportingFile','slrt_ex_osc.slx')
   ```
2. Build the model, creating a real-time application file MLDATX.
3. Open the Simulink Real-Time App Generator. In the **Real-Time** tab, select **Review Results** > **App Generator**.
4. To create an instrument panel from the real-time application file MLDATX, select **New** > **New**, click **No** to remove the current session, and select the MLDATX file. For information about the difference between developing an instrument panel from a model SLX file or a real-time application MLDATX file, see the “Tip About MLDATX and SLX Files.” on page 3-0 .
5. Add signals and parameters to the instrument panel app. From the **Signals and Parameters** pane, select the `Amplitude` parameter, the `Frequency` parameter, and the `XfrFnc` signal. Click the **Add Selection** button.

   
6. Configure each control by clicking on its **Control Type** entry and editing the selections for the control. This figure shows a possible configuration for this instrument panel.



7 To create the instrument panel app, click the **Generate App** button.

After creating the app, you can open it in App Designer to further customize the instrument panel.

The App Generator adds controls to your instrument panel that let the panel interface with the real-time application. These controls include the target computer selector, connect button, load application button, start/stop button, stop time field, and system log. Any instrumented signals from the model are added in an axis component. For more information, see “Create App Designer Instrument Panels by Using Simulink Real-Time Components”.

8 **Tip About MLDATX and SLX Files.** You can develop an instrument panel app in the App Generator from a model SLX file (if you start the App Generator from the Real-Time tab in the

Simulink Editor) or from a real-time application MLDATX file. It is recommended that you develop the instrument panel based on the MLDATX file, because—when developing from the MLDATX file—the App Generator only lists the signals and parameters that are present in the generated code. If you develop the instrument panel based on the SLX file, the App Generator can list more signals than are present in the generated code. These signals include virtual signals and signals to Scope blocks.

Open Real-Time Application and Create App

This example shows how to open a real-time application in the App Generator, add signals and parameters to an instrument panel app from the real-time application, and add signals and parameters to the instrument panel app from the model that corresponds to the real-time application..

1. Open the App Generator. In the MATLAB Command Window, type:

   ```
   slrtAppGenerator
   ```
2. To create a new instrument panel app, click the **New** button and select the real-time application file `slrt_ex_osc.mldatx`. You created this file in “Configure Instrument Panel Controls and Create App” on page 3-32.
3. Add signals and parameters to the instrument panel app. From the **Signals and Parameters** pane, select the `Amplitude` parameter, the `Frequency` parameter, and the `XfrFnc` signal. Click the **Add Selection** button.

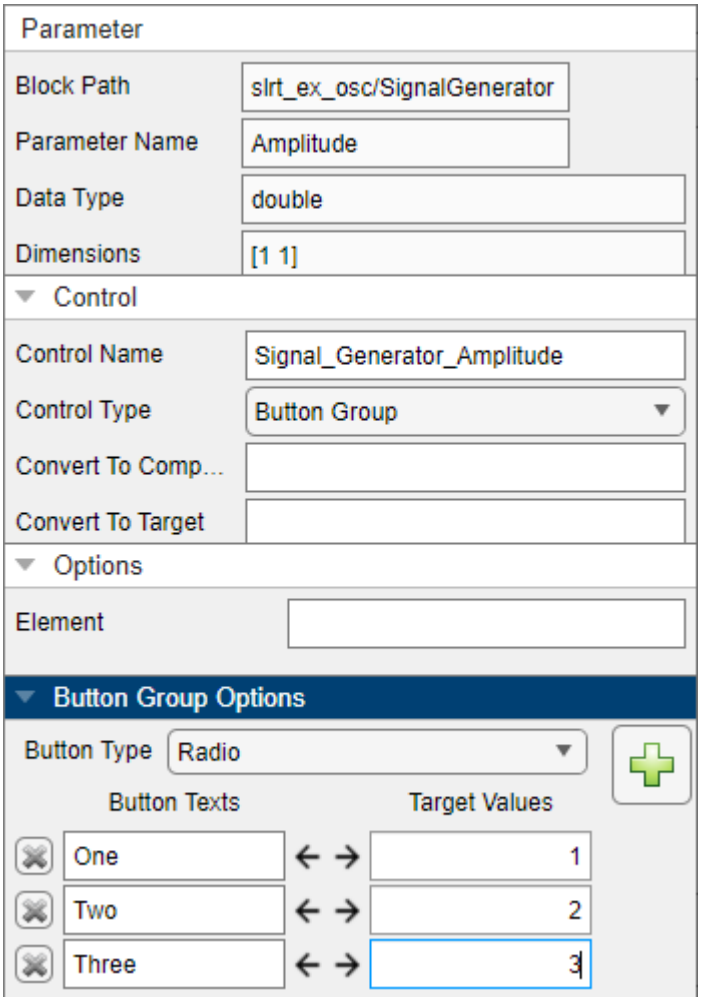
   
4. To add signals and parameters from the model that corresponds to the real-time application, click the **Add From Model** button.

   The App Generator opens the model and puts the model in bind mode for signal and parameter selection. For more information about bind mode, see “Add Instruments to Real-Time Application from Simulink Model”.
5. To return to the App Generator, close bind mode in the model.
6. To create the instrument panel app, click the **Generate App** button.

Bind Parameter to Toggle or Radio Button Group

Because conversion functions are needed to transform the value of a `uibuttongroup` `'SelectObject'` property to a usable real-time parameter value (and vice-versa), it is recommended that you use the App Generator to add button groups for parameters. The App Generator adds the needed conversion code as part of the instrument panel generation process.

1. Follow the steps in “Open Real-Time Application and Create App” on page 3-35, stopping before generating the instrument panel.
2. Select and add a parameter from the App Generator **Signals and Parameters** pane to the **Bindings** tab.
3. Select the parameter in the **Bindings** tab and change the **Control Type** to **Button Group**.
4. Configure the control in the **Properties** panel. The figure shows an example parameter setup from the `slrt_ex_osc` model.



5 Generate the instrument panel and open it in App Designer.

The generated instrument panel has a button group to apply values to the parameter. To examine the generated conversion functions, open the **Code View**.

Add Callbacks for Target Computer Events

1 Follow the steps in “Open Real-Time Application and Create App” on page 3-35, stopping before generating the instrument panel.
2 Click the **Target Events** button.
3 Select the `Connected` event from the list, click in the callback field, and add the function command:

```
disp('Connected to target computer')
```

4 Select the `Loaded` event from the list, click in the callback field, and add the function command:

```
disp('Loaded MLDATX on target computer')
```

5 To add these callbacks, click **Apply**. To close the **Callback Editor** dialog box, click **OK**.
6 Generate the instrument panel and open it in App Designer.

The generated instrument panel has callback code for the target computer events. To examine the generated conversion functions, open the **Code View** and see the added methods.

```
methods
    function ConnectedFcn(app, event)
        disp('Connected to target computer')
    end
    function LoadedFcn(app, event)
        disp('Loaded MLDATX on target computer')
    end
end
```

When you run the app and connect, your `ConnectedFcn` callback executes automatically. When you load a real-time application, your `LoadedFcn` callback executes automatically.

- “Create App Designer Instrument Panels by Using App Generator”

Programmatic Use

`slrtAppGenerator` opens the Simulink Real-Time App Generator. Operations that the Simulink Real-Time App Generator UI adds to an instrument panel app correspond App Designer controls that are customized for your real-time application. For more information about these controls, see slrealtime.ui Properties.

Version History

Introduced in R2022a

R2024a: Apply added App Generator and App Designer support

The Simulink Real-Time `LatchButton` and `MomentaryButton` components provide push button support for App Designer apps and App Generator generated apps.

Information is available for the `slrealtime.instrument.LineStyle` object `Label` property. In the App Generator, for an `Axes` component, this property appears as **Line Label**. In App Designer, the `Label` property is set by using the `slrealtime.instrument.LineStyle` object.

To ease adding callback code in generated apps, the Target Events button in the App Generator lets you add callback code for target computer events and outputs your event callback code in the generated app. For more information, see “Add Callbacks for Target Computer Events” on page 3-36.

R2023b: Apply added App Generator and App Designer Support

Additional Simulink Real-Time App Generator features ease the App Designer workflow and provide more signal binding information. These additions include:

- Detect whether an app is open in App Designer and close the app when Add to App or Generate App is selected in the App Generator.
- Display data type and dimension information in the App Generator properties pane.
- Create bindings for each selected bus element with App Generator when you create a binding for a bus signal and select multiple bus elements.

- App Generator automatically adds labels to app components.
- Add an `ImportFileLogButton` control to an app in App Designer.

R2023a: Button group support for parameter values

Button Group support — you can bind a real-time parameter to a button group of toggle or radio buttons. The Simulink Real-Time App Generator adds the conversion functions that are needed to transform the value of the button group to a usable real-time parameter value (and vice versa). For more information, see “Bind Parameter to Toggle or Radio Button Group” on page 3-35.

See Also

`slrtAppGenerator` | `slrtExplorer` | `slrtLogViewer` | `slrtTETMonitor` | slrealtime.ui Properties

Topics

“Create App Designer Instrument Panels by Using App Generator”
“Target Computer Settings”
“Real-Time Application and Target Computer Modes”
“Configure and Control Real-Time Application by Using Simulink Real-Time Explorer”
“Filter Hierarchical List of Signals and Parameters in Simulink Real-Time Explorer”
“Stream Real-Time Signals by Using Simulink Real-Time Explorer”
“Export and Import Signals in Instrument by Using Simulink Real-Time Explorer”
“Tune Parameters by Using Simulink® Real-Time™ Explorer”

Simulation Data Inspector

Inspect and compare data and simulation results to validate and iterate model designs

Description

The Simulation Data Inspector visualizes and compares multiple kinds of data.

Using the Simulation Data Inspector, you can inspect and compare time series data at multiple stages of your workflow. This example workflow shows how the Simulation Data Inspector supports all stages of the design cycle:

1. “View Simulation Data in Simulation Data Inspector” or “Import Data from Workspace or File into Simulation Data Inspector”

   Run a simulation in a model configured to log data to the Simulation Data Inspector, or import data from the workspace or a file. You can view and verify model input data or inspect logged simulation data while iteratively modifying your model diagram, parameter values, or model configuration. For more information about logging simulation data, see “Save Simulation Data”.

2. “Inspect Simulation Data”

   Plot signals on multiple subplots, zoom in and out on specified plot axes, and use data cursors to understand and evaluate the data. You can choose from several visualizations, such as time, array, map, sparklines, and XY plots. For more information on presenting data effectively, see “Create Plots Using the Simulation Data Inspector”.

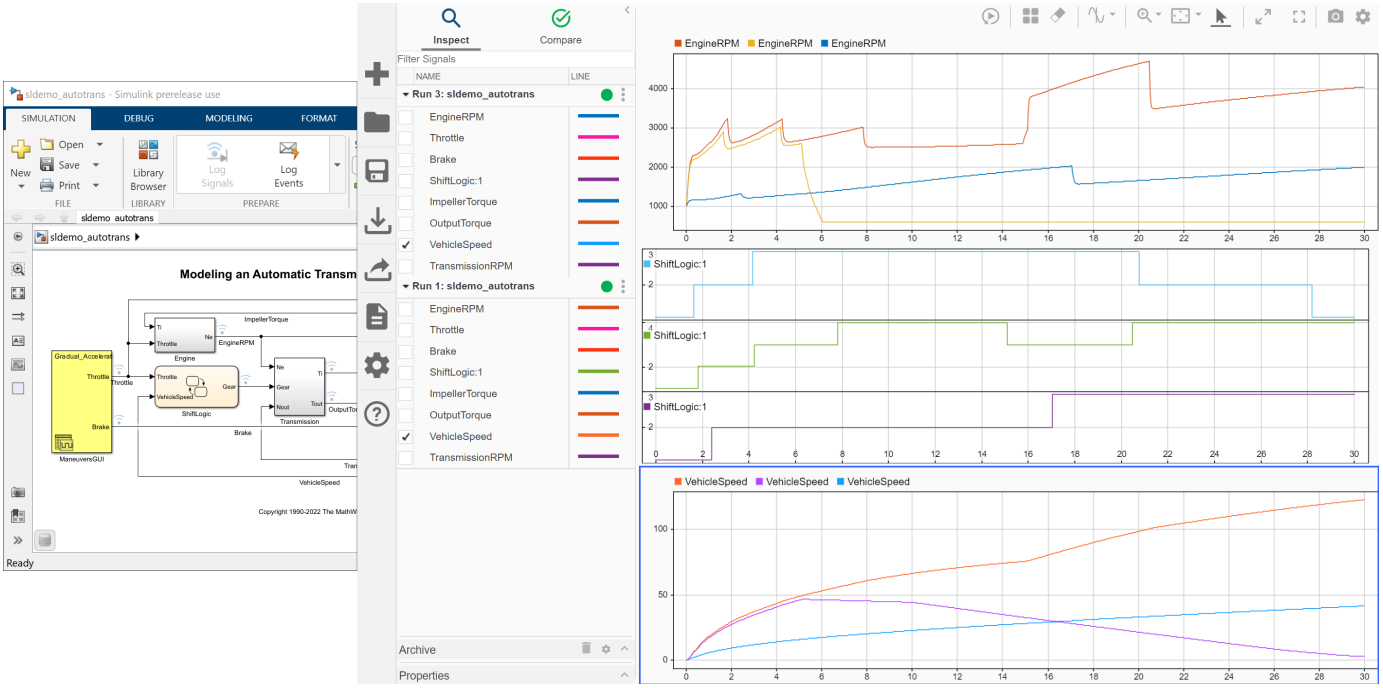
3. “Compare Simulation Data”

   Compare individual signals or simulation runs and analyze your comparison results with relative, absolute, and time tolerances. The compare tools in the Simulation Data Inspector facilitate iterative design and allow you to highlight signals that do not meet your tolerance requirements. For more information about the comparison operation, see “How the Simulation Data Inspector Compares Data”.

4. “Save and Share Simulation Data Inspector Data and Views”

   Share your findings with others by saving Simulation Data Inspector data and views.

You can also harness the capabilities of the Simulation Data Inspector from the command line. For more information, see “Inspect and Compare Data Programmatically”.



Open the Simulation Data Inspector

- Simulink Toolstrip: On the **Simulation** tab, under **Review Results**, click **Data Inspector**.
- From a model: click the streaming badge on a signal to open the Simulation Data Inspector and plot the signal.
- MATLAB command prompt:
  - Type `Simulink.sdi.view` to open the Simulation Data Inspector.
  - Use the `Simulink.sdi.plot` to open the Simulation Data Inspector and plot data.

Examples

Create Run and View Data

Create a run, add data to it, and then view the data in the Simulation Data Inspector.

Create Data for Run

Create two `timeseries` objects to contain data for a sine signal and a cosine signal. Give each `timeseries` object a descriptive name.

```
time = linspace(0,20,100);

sine_vals = sin(2*pi/5*time);
sine_ts = timeseries(sine_vals,time);
sine_ts.Name = "Sine, T=5";

cos_vals = cos(2*pi/8*time);
```

```
cos_ts = timeseries(cos_vals,time);
cos_ts.Name = "Cosine, T=8";
```

Create Run and Add Data

Use the `Simulink.sdi.view` function to open the Simulation Data Inspector.

```
Simulink.sdi.view
```

To import data into the Simulation Data Inspector from the workspace, create a `Simulink.sdi.Run` object using the `Simulink.sdi.Run.create` function. Add information about the run to its metadata using the `Name` and `Description` properties of the `Run` object.

```
sinusoidsRun = Simulink.sdi.Run.create;
sinusoidsRun.Name = "Sinusoids";
sinusoidsRun.Description = "Sine and cosine signals with different frequencies";
```

Use the `add` function to add the data you created in the workspace to the empty run.

```
add(sinusoidsRun,"vars",sine_ts,cos_ts);
```

Plot Data in Simulation Data Inspector

Use the `getSignalByIndex` function to access `Simulink.sdi.Signal` objects that contain the signal data. You can use the `Simulink.sdi.Signal` object properties to specify the line style and color for the signal and plot the signal in the Simulation Data Inspector. Specify the `LineColor` and `LineDashed` properties for each signal.

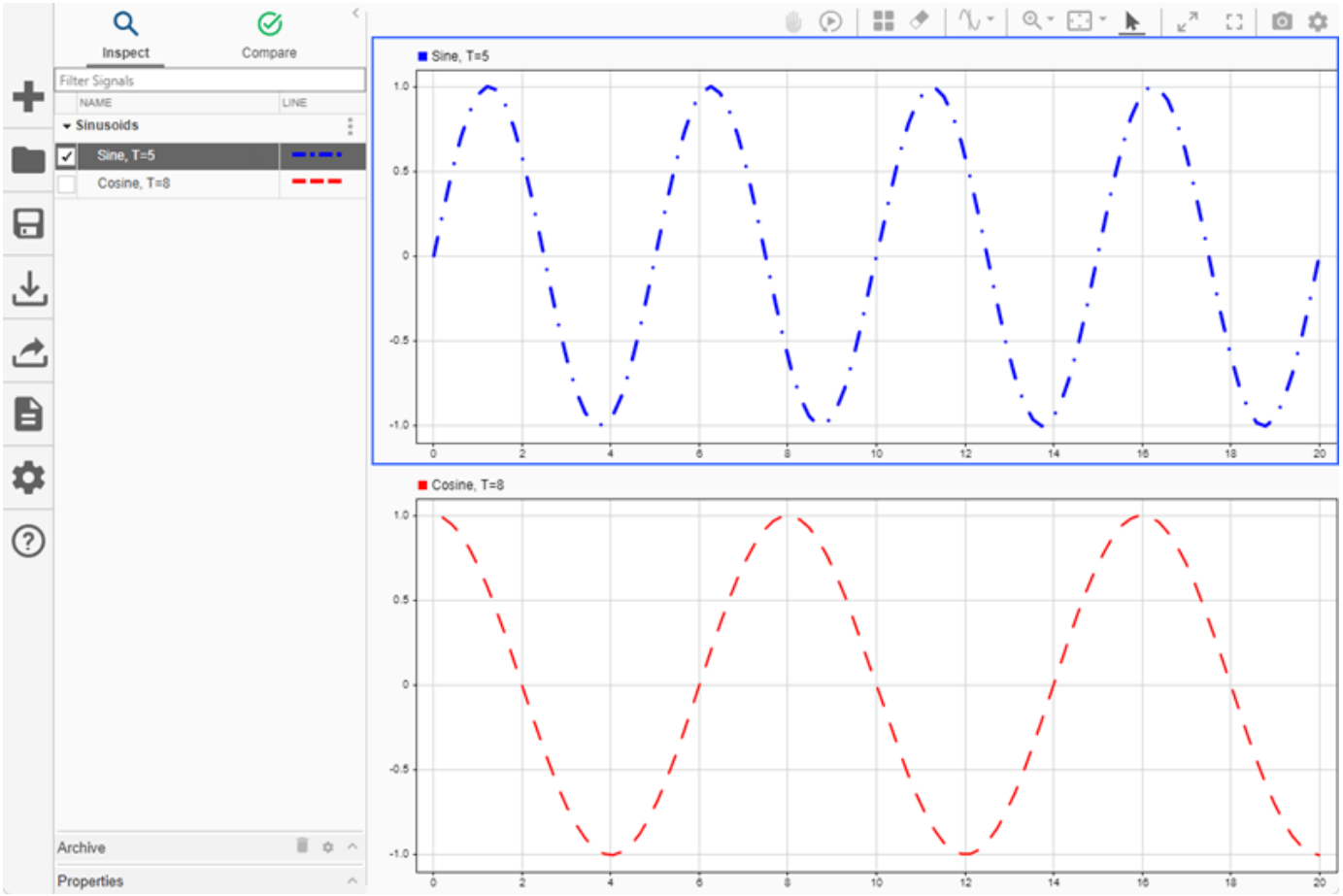
```
sine_sig = getSignalByIndex(sinusoidsRun,1);
sine_sig.LineColor = [0 0 1];
sine_sig.LineDashed = "-.";

cos_sig = sinusoidsRun.getSignalByIndex(2);
cos_sig.LineColor = [1 0 0];
cos_sig.LineDashed = "--";
```

Use the `Simulink.sdi.setSubPlotLayout` function to configure a 2-by-1 subplot layout in the Simulation Data Inspector plotting area. Then, use the `plotOnSubplot` function to plot the sine signal on the top subplot and the cosine signal on the lower subplot.

```
Simulink.sdi.setSubPlotLayout(2,1);

plotOnSubPlot(sine_sig,1,1,true);
plotOnSubPlot(cos_sig,2,1,true);
```



Close Simulation Data Inspector and Save Data

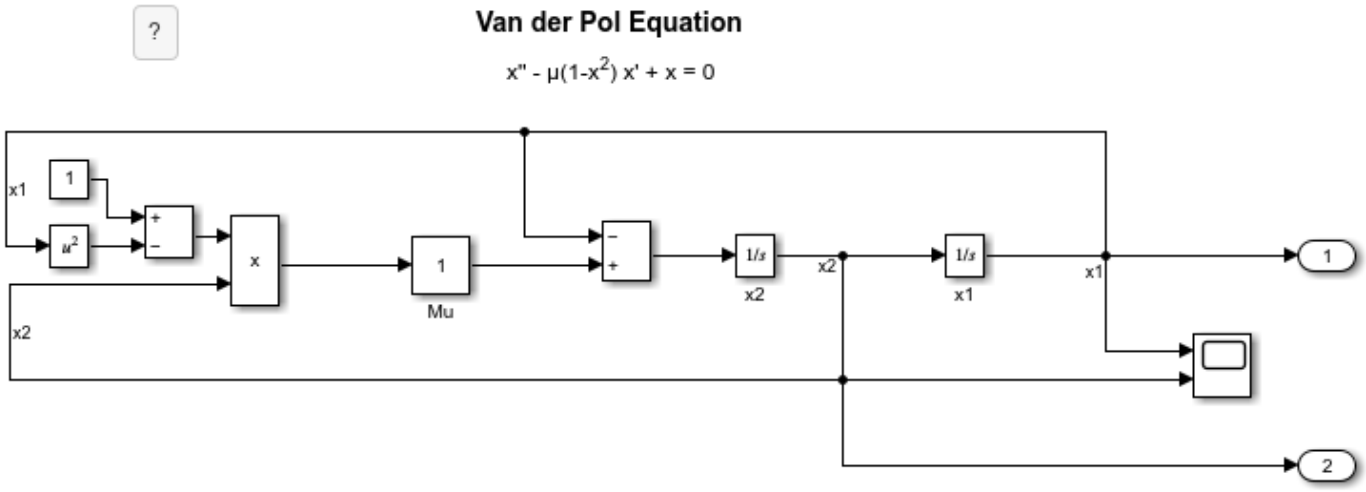
When you finish inspecting the plotted signal data, you can close the Simulation Data Inspector and save the session to an MLDATX file.

```
Simulink.sdi.close("sinusoids.mldatx")
```

Plot Simulation Result in Simulation Data Inspector

You can use the `Simulink.sdi.plot` function to plot simulation results in the Simulation Data Inspector. Open the model `vdp`, which models the second-order Van der Pol differential equation. For more information about the model, see “Van der Pol Oscillator”.

```
mdl = "vdp";
open_system(mdl)
```

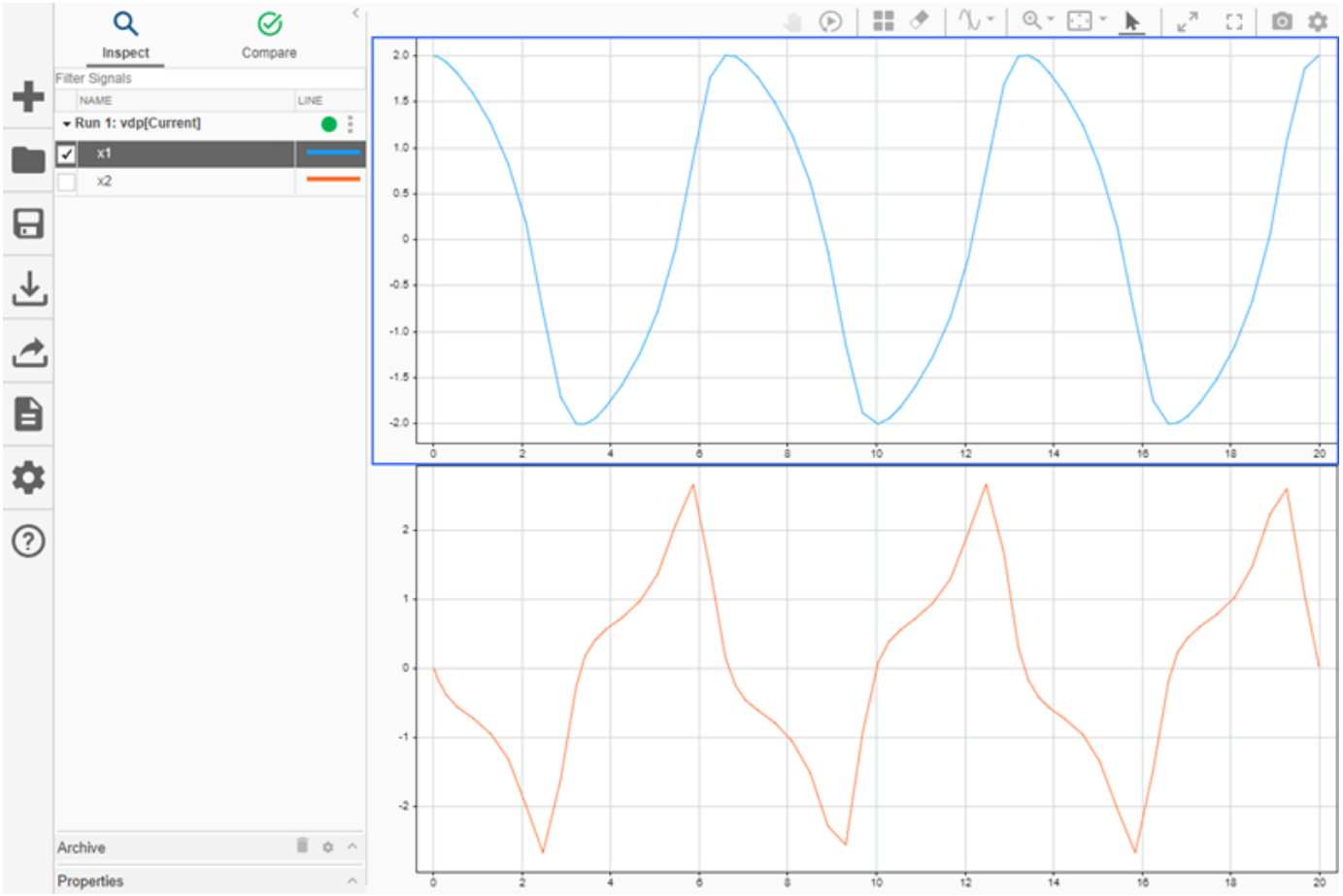


Simulate the model. The model logs two signals: `x1` and `x2`. Simulation results are stored in a single `SimulationOutput` object named `out`.

```
out = sim(mdl);
```

Use the `Simulink.sdi.plot` function to open the Simulation Data Inspector and plot the results.

```
Simulink.sdi.plot(out);
```



Apply Tolerance to Signal in Multiple Runs

You can use the Simulation Data Inspector programmatic interface to modify a parameter for the same signal in multiple runs. This example adds an absolute tolerance of `0.1` to a signal in all four runs of data.

First, clear the workspace and load the Simulation Data Inspector session with the data. The session includes logged data from four simulations of a Simulink® model of a longitudinal controller for an aircraft.

```
Simulink.sdi.clear
Simulink.sdi.load('AircraftExample.mldatx');
```

Use the `Simulink.sdi.getRunCount` function to get the number of runs in the Simulation Data Inspector. You can use this number as the index for a for loop that operates on each run.

```
count = Simulink.sdi.getRunCount;
```

Then, use a for loop to assign the absolute tolerance of `0.1` to the first signal in each run.

```
for a = 1:count
    runID = Simulink.sdi.getRunIDByIndex(a);
```

```
    aircraftRun = Simulink.sdi.getRun(runID);
    sig = getSignalByIndex(aircraftRun,1);
    sig.AbsTol = 0.1;
end
```

- “View Simulation Data in Simulation Data Inspector”
- “Inspect Simulation Data”
- “Compare Simulation Data”
- “Iterate Model Design Using the Simulation Data Inspector”
- “Save Simulation Data”

Programmatic Use

`Simulink.sdi.view` opens the Simulation Data Inspector from the MATLAB command line.

Version History

Introduced in R2010b

See Also

Functions

`Simulink.sdi.clear` | `Simulink.sdi.clearPreferences` | `Simulink.sdi.snapshot`

Topics

“View Simulation Data in Simulation Data Inspector”
“Inspect Simulation Data”
“Compare Simulation Data”
“Iterate Model Design Using the Simulation Data Inspector”
“Save Simulation Data”

Simulink Real-Time Application Property Inspector

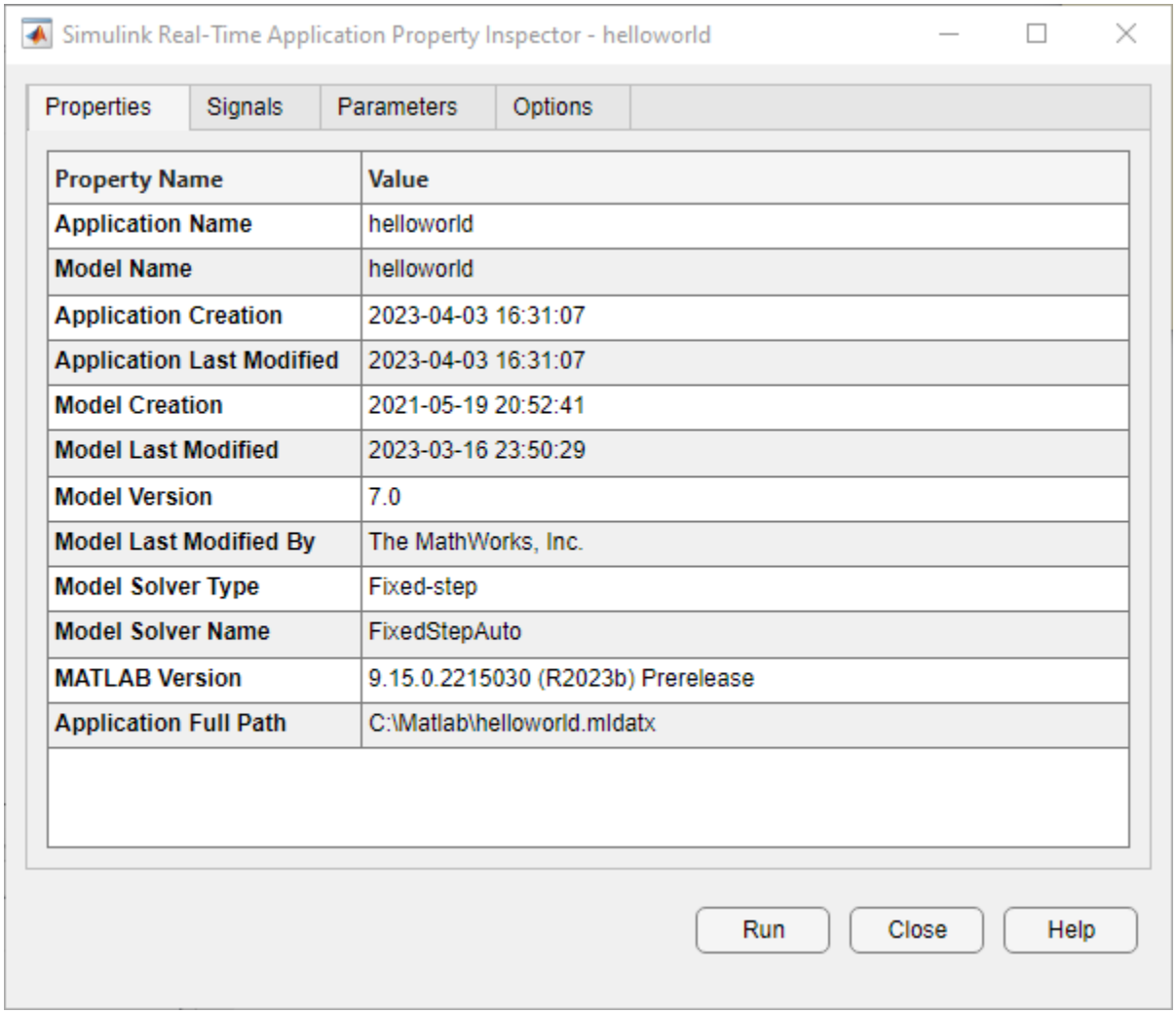
View and modify application properties

Description

Simulink Real-Time Application Property Inspector tool provides an user interface to view and modify the properties of a real-time application. You can view Application information, Signals, and Parameters. You can modify the application properties `Max fil log runs` and `Stop time`.

Use Simulink Real-Time Application Property Inspector for these tasks:

- View the model and application information.
- View the list of signals present in application.
- View the list of parameters present in application.
- View and edit the `Max file log runs` and `Stop time` fields.
- Run the application on the Speedgoat target computer.



Open the Simulink Real-Time Application Property Inspector App

- In the MATLAB Current Folder browser, select the `MLDATX` application file, right click, and select `Open`.
- In the MATLAB Current Folder browser, double-click the `MLDATX` application file.
- From the Simulink Editor, in the **Real-Time** tab, select **Prepare** > **SLRT Explorer**. Select **Prepare** > **Application Inspector** on the SLRT Explorer.
- From the MATLAB Command Window, type:

  ```
  slrtApplicationInspector(<myApplication.mldatx>)
  ```

Examples

View Application Signals and Parameters

This example shows how to view the application properties using the Simulink Real-Time Application Property Inspector.

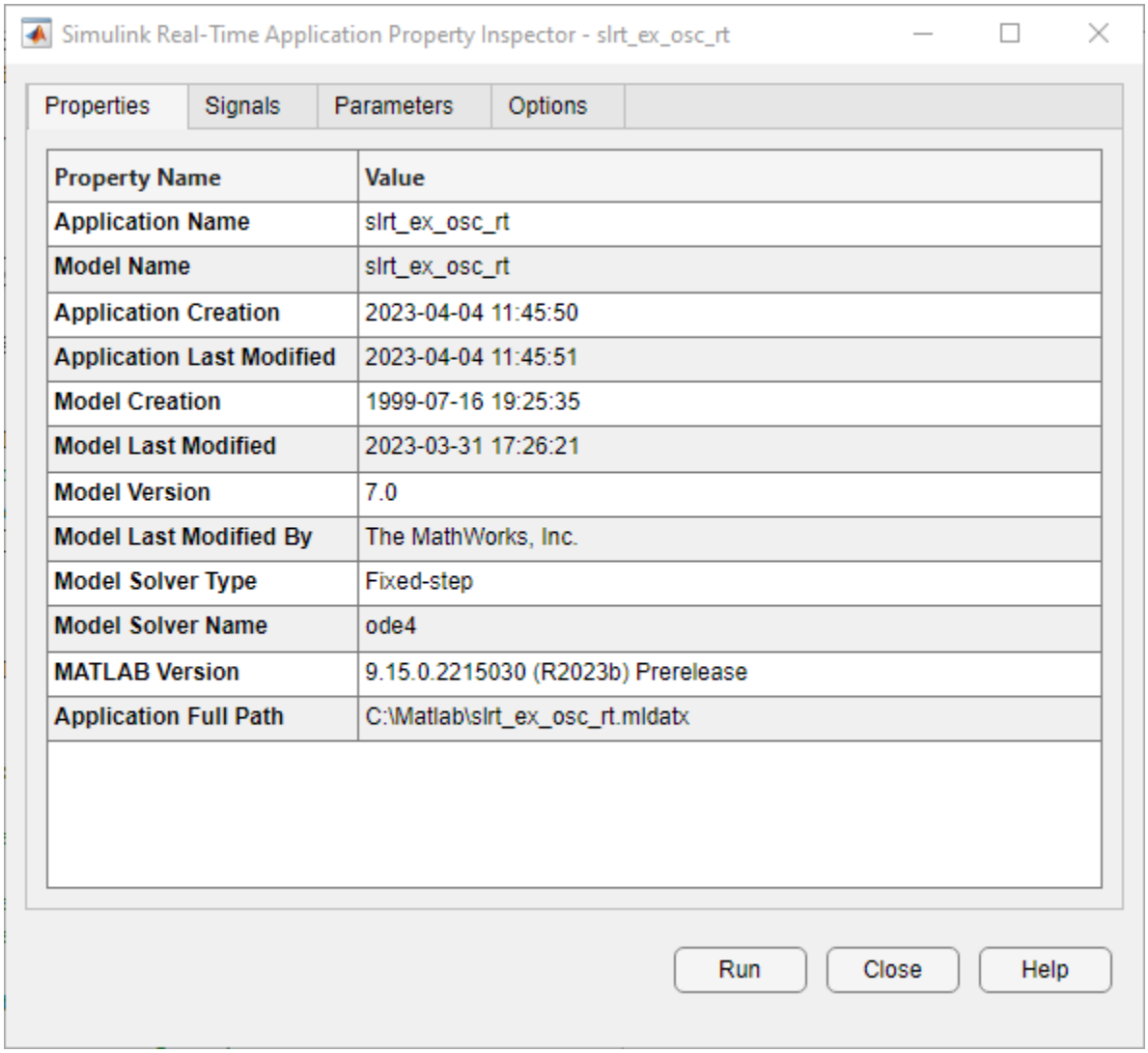
Open an example Simulink model, for example, `slrt_ex_osc_rt`. In the Command Window, type:

```
openExample('slrealtime/SlrtCreateAndRunRealTimeAppFromSimulinkModelExample', ...
    'supportingFile','slrt_ex_osc_rt.slx')
```

Build the model. In the Simulink Editor, on the **Real-Time** tab, click **Run on Target > Build Application**.

Open Simulink Real-Time Explorer. In the Simulink Editor, on the **Real-Time** tab, click **Prepare > SLRT Explorer**.

Open Simulink Real-Time Application Property Inspector. In the SLRT Explorer, click **Prepare > Application Inspector**.



Programmatic Use

`slrtApplicationInspector` opens the Simulink Real-Time Application Property Inspector.

Version History

Introduced in R2023b

See Also

Simulink Real-Time Explorer | `slrtApplicationInspector` | `Application`

Target Computer Status Monitor

Target Computer Status Monitor

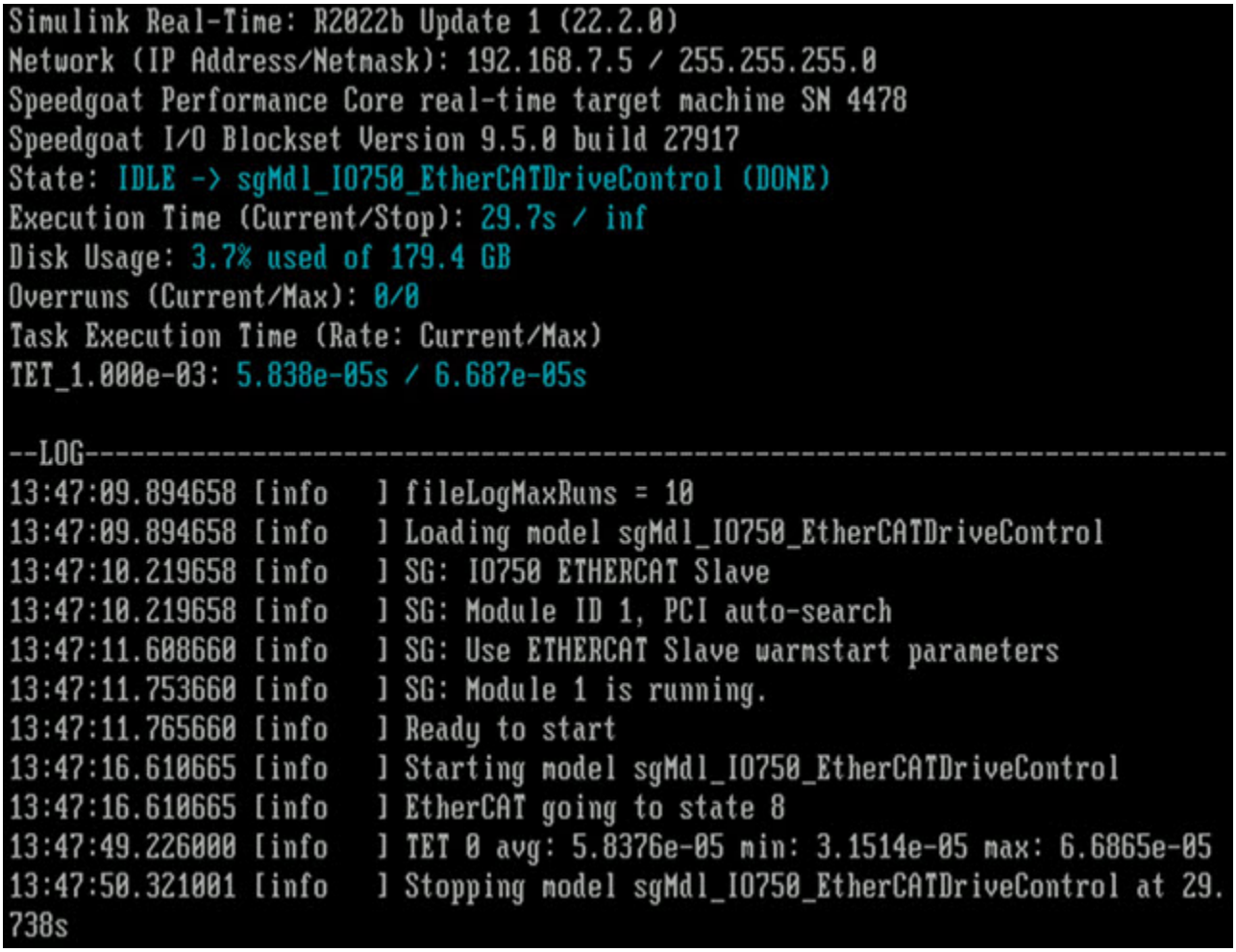
The status monitor application on the Speedgoat target computer displays the status of the real-time application, disk usage, and other target computer status information.

The target computer display supports multiple sessions. You can choose to display the status monitor (default, session 1) or display the target computer command-line interface (session 2).

Display Status Monitor

Start the target computer.

The target computer displays session 1 (default) and the target computer status monitor.



Display Status Monitor by Using PuTTY

To view the status monitor from the development computer, use PuTTY to open an SSH client and start the status monitor application `statusmonitor` on the target computer. Keyboard commands for the status monitor include:

- Q (quit)
- Up arrow (scroll up in the log)
- Down arrow (scroll down in the log)

For more information about PuTTY, see “Execute Target Computer RTOS Commands at Target Computer Command Line”.

To display the target computer command-line interface, switch to display session 2:

1. Start the target computer.

   The target computer displays session 1 and the target computer status monitor.
2. To switch to session 2 and use the target computer command-line interface, on the target computer keyboard (console), press **Ctrl+Alt+2**.
3. To switch back to session 1 (status monitor), on the target computer keyboard (console), press **Ctrl+Alt+1**.

5

Target Computer Command-Line Interface Reference

Target Computer Command-Line Interface

You can load, run, stop, and check the status of a real-time application by using the Speedgoat target computer command-line interface commands.

By default, the target computer displays the session 1 screen with the target computer status monitor. For information about switching to the session 2 screen with the command-line interface, see “Target Computer Status Monitor” on page 4-2.

To read the target computer console log, view the log in the `slrtLogViewer`.

Target Object Commands

When you are using the target computer command-line interface, target object functions support loading, starting, stopping, and checking the status of the real-time application.

For a description of how to use these commands, see “Control Real-Time Application at Target Computer Command Line”.

Note To run user commands, log in as user `slrt` by using password `slrt`. To run the system commands (for example, `date`, `ntpdate`, `ntpd`, `rtc`, or setting the time zone), login as user `root` by using password `root`.

These commands are `Target` object commands that you can use through the command-line interface on the target computer. Each command appears with its equivalent MATLAB syntax. In the descriptions, `Target` is the target object name, and app_name is the real-time application MLDATX file name.

- **Target:** `slrealtime listApplications`

  MATLAB: `getInstalledApplications(Target)`

  When run from the development computer in the MATLAB Command Window, the `getInstalledApplications` command returns a list of the real-time application that are installed on the target computer.
- **Target:** `slrealtime load --AppName app_name`

  MATLAB: `load(Target,'app_name')`

  When run from the development computer in the MATLAB Command Window, the `load` command deploys the real-time application to the target computer and loads the application. When run from the target computer command interface, the `load` command loads the application.
- **Target:** `slrealtime start`

  MATLAB: `start(Target)`

  The `start` command runs the real-time application that is loaded on the target computer.
- **Target:** `slrealtime stop`

  MATLAB: `stop(Target)`

The `stop` command stops the real-time application that is running on the target computer.

- **Target:** `slrealtime install --AppName app_name`

  The `slrealtime install` command installs the real-time application MLDATX file for standalone operation on the target computer. The MLDATX file should have been previously downloaded to the target computer and be either in the current directory or specified by using absolute path.

  MATLAB: `install(Target,'app_name')`

  The `install` command installs the real-time application for standalone operation on the target computer. The command uses the MATLAB path to find the real-time application MLDATX file to install.

- **Target:** `slrealtime saveParamSet --FileName filename`

  Target: `slrealtime saveParamSet -F filename`

  MATLAB: `saveParamSet(Target,filename)`

  The `saveParamSet` command saves the parameter set from the loaded real-time application on the target computer to the specified `filename`.

- **Target:** `slrealtime loadParamSet --FileName filename`

  Target: `slrealtime loadParamSet -F filename`

  MATLAB: `loadParamSet(Target,filename)`

  The `loadParamSet` command loads the parameter set into the real-time application on the target computer from the specified `filename`.

- **Target:** `shutdown -S reboot`

  MATLAB: `reboot(Target)`

  The `reboot` command reboots the target computer.

  If you prefer to safely shutdown the RTOS before turning off power to the target computer, you can use the command: `shutdown -S system`

Target Computer RTOS System Commands

The target computer uses the QNX Neutrino Real-Time Operating System (RTOS). You can run system commands on the target computer from the development computer by using an SSH utility, such as PuTTY. Or, you can run system commands on the target computer from its keyboard (console). Target computer RTOS system command information is available in the Utilities Reference in the QNX Momentics IDE 7.1 User's Guide. All commands that this reference identifies as **Runs on: QNX Neutrino** are supported on the target computer.

Some RTOS commands are required for configuring the target computer. These commands include:

- `date` — set date and time
- `ntpdate` — set the local date and time from NTP server
- `ntpd` — start NTP daemon

- `rtc` — set date from hardware clock

Note To run user commands, log in as user `slrt` by using password `slrt`. To run the system commands (for example, `date`, `ntpdate`, `ntpd`, `rtc`, or setting the time zone), login as user `root` by using password `root`.

For a description of how to use these commands, see “Execute Target Computer RTOS Commands at Target Computer Command Line”.

See Also

More About

- “Control Real-Time Application at Target Computer Command Line”
- “Execute Target Computer RTOS Commands at Target Computer Command Line”

External Websites

- QNX Momentics IDE 7.1 User’s Guide
- QNX Momentics IDE 7.1 User’s Guide, Utilities Reference