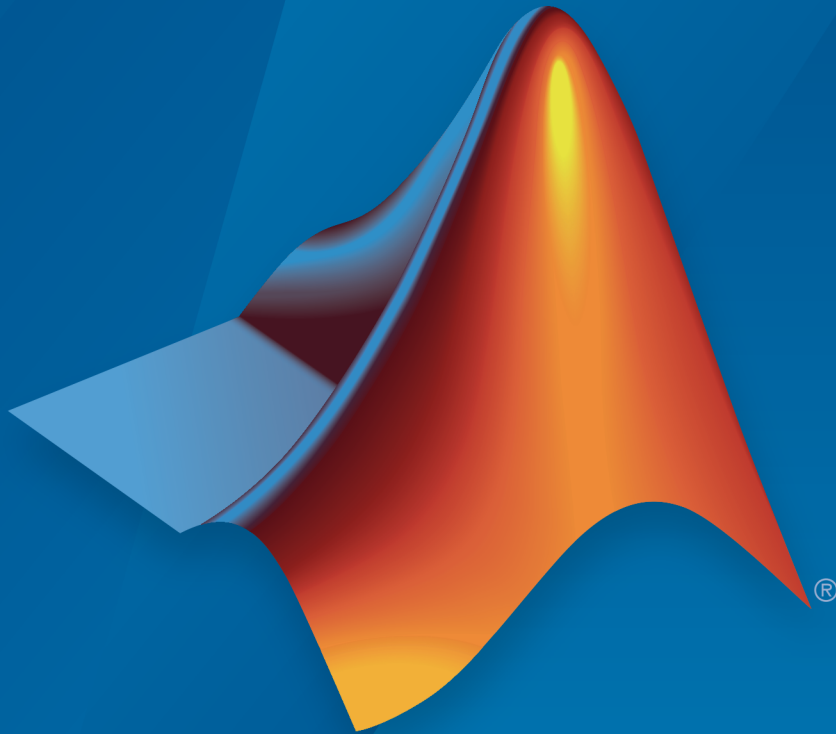


# Simulink<sup>®</sup> Desktop Real-Time<sup>™</sup>

## User's Guide



# MATLAB<sup>®</sup>

R2016a



## 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.  
3 Apple Hill Drive  
Natick, MA 01760-2098

*Simulink<sup>®</sup> Desktop Real-Time<sup>™</sup> User's Guide*

© COPYRIGHT 1999–2016 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

January 1999	First printing	New for Version 1.0 (Release 11.0)
January 2000	Second printing	Revised for Version 1.5 (Release 11.1+)
September 2000	Third printing	Revised for Version 2.0 (Release R12)
June 2001	Online only	Revised for Version 2.1 (Release R12.1)
July 2002	Online only	Revised for Version 2.2 (Release 13)
June 2004	Fourth printing	Revised for Version 2.5 (Release 14)
October 2004	Fifth printing	Revised for Version 2.5.1 (Release 14SP1)
March 2005	Online only	Revised for Version 2.5.2 (Release 14SP2)
September 2005	Online only	Revised for Version 2.6 (Release 14SP3)
March 2006	Online only	Revised for Version 2.6.1 (Release 2006a)
September 2006	Online only	Revised for Version 2.6.2 (Release 2006b)
March 2007	Online only	Revised for Version 2.7 (Release 2007a)
September 2007	Online only	Revised for Version 3.0 (Release 2007b)
March 2008	Online only	Revised for Version 3.1 (Release 2008a)
October 2008	Online only	Revised for Version 3.2 (Release 2008b)
March 2009	Online only	Revised for Version 3.3 (Release 2009a)
September 2009	Online only	Revised for Version 3.4 (Release 2009b)
March 2010	Online only	Revised for Version 3.5 (Release 2010a)
September 2010	Online only	Revised for Version 3.6 (Release 2010b)
April 2011	Online only	Revised for Version 3.7 (Release 2011a)
September 2011	Online only	Revised for Version 3.8 (Release 2011b)
March 2012	Online only	Revised for Version 4.0 (Release 2012a)
September 2012	Online only	Revised for Version 4.1 (Release 2012b)
March 2013	Online only	Revised for Version 4.2 (Release 2013a)
September 2013	Online only	Revised for Version 4.3 (Release 2013b)
March 2014	Online only	Revised for Version 4.4 (Release 2014a)
October 2014	Online only	Revised for Version 4.5 (Release 2014b)
March 2015	Online only	Revised for Version 5.0 (Release 2015a)
September 2015	Online only	Revised for Version 5.1 (Release 2015b)
March 2016	Online only	Revised for Version 5.2 (Release 2016a)



**1** | **Getting Started**

---

<b>Simulink Desktop Real-Time Product Description</b> .....	1-2
Key Features .....	1-2
<b>Real-Time Execution in Normal Mode</b> .....	1-3
<b>Real-Time Execution in Accelerator Mode</b> .....	1-5
<b>Real-Time Execution in External Mode</b> .....	1-8

**2** | **Installation and Configuration**

---

<b>Software Components</b> .....	2-2
MATLAB Environment .....	2-2
Simulink Software .....	2-2
Simulink Coder Software .....	2-2
Known Limitations .....	2-3
<b>Install Real-Time Kernel</b> .....	2-4
Install the Kernel Using MATLAB .....	2-4
Uninstall the Kernel .....	2-5
<b>Run Confidence Test</b> .....	2-7
Run the Model sldrtex_vdp .....	2-7
Display Status Information .....	2-9
Examples Library .....	2-10

<b>Prepare for Real-Time Execution</b> .....	<b>3-2</b>
<b>Create a Real-Time Application</b> .....	<b>3-4</b>
<b>Create a Simulink Model</b> .....	<b>3-5</b>
Reference a Model .....	3-8
File System Input/Output .....	3-8
<b>Configure a Model</b> .....	<b>3-9</b>
Specify a Default Configuration Set .....	3-9
Enter Configuration Parameters Manually .....	3-10
Enter Scope Parameters for Signal Tracing .....	3-11
<b>Run Normal Mode Simulation</b> .....	<b>3-15</b>
<b>Run Accelerator Mode Simulation</b> .....	<b>3-17</b>
<b>Set External Mode Code Generation Parameters</b> .....	<b>3-19</b>
<b>Set Scope Parameters</b> .....	<b>3-22</b>
<b>Prepare External Mode Application</b> .....	<b>3-25</b>
<b>Set External Mode Scope Parameters</b> .....	<b>3-26</b>
<b>Run Application from Simulink</b> .....	<b>3-29</b>
Normal Mode .....	3-29
Accelerator Mode .....	3-29
External Mode .....	3-29
<b>Run Application from MATLAB Command Line</b> .....	<b>3-31</b>
Normal Mode .....	3-31
Accelerator Mode .....	3-31
External Mode .....	3-32
<b>Inspect Simulink Desktop Real-Time Signals with Simulation Data Inspector</b> .....	<b>3-33</b>
<b>Signal Logging to the Workspace</b> .....	<b>3-36</b>

<b>Set Scope Parameters for Logging to Workspace</b> .....	3-37
<b>Set External Mode Properties for Logging to Workspace</b> ..	3-39
<b>Plot Signal Data Logged to Workspace</b> .....	3-41
<b>Signal Logging to a File</b> .....	3-43
<b>Set Scope Parameters for Logging to File</b> .....	3-44
<b>Set External Mode Properties for Logging to File</b> .....	3-47
<b>Set External Mode Data Archiving Parameters</b> .....	3-49
<b>Plot Signal Data Logged to File</b> .....	3-51
<b>Tune Block Parameters and MATLAB Variables</b> .....	3-53
<b>Change Block Parameters Using Simulink Dialog Box</b> ....	3-54

## Boards, Blocks, and Drivers

# 4

<b>Use I/O Boards</b> .....	4-2
Install and Configure I/O Boards and Drivers .....	4-2
ISA Bus Board .....	4-5
PCI Bus Board .....	4-5
PC/104 Board .....	4-6
Compact PCI Board .....	4-6
PCMCIA Board .....	4-6
<b>Use I/O Driver Blocks</b> .....	4-7
View Simulink Desktop Real-Time Library .....	4-7
Route Signals from an I/O Block .....	4-8
Configure Channel Selection .....	4-9
<b>Use Analog I/O Drivers</b> .....	4-12
Configure I/O Driver Characteristics .....	4-12
Normalize Scaling for Analog Inputs .....	4-12

Use Vector CAN Drivers .....	4-16
------------------------------	------

## Troubleshooting

### 5

Failure to Connect to Target .....	5-2
Scope Output Delayed or Missing .....	5-3
Plots Not Visible in Simulink Scope Block .....	5-4
S-Functions Using Math Functions .....	5-5
Building Older Models .....	5-6
Vendor Software Required .....	5-7
Unsupported Models .....	5-8

## Custom I/O Driver Blocks

### A

Custom I/O Driver Basics .....	A-2
Supported C Functions .....	A-2
Unsupported C Functions .....	A-3
Incompatibility with Operating System API Calls .....	A-3
I/O Register Access from S-Functions Limitation .....	A-4



# Getting Started

---

- “Simulink Desktop Real-Time Product Description” on page 1-2
- “Real-Time Execution in Normal Mode” on page 1-3
- “Real-Time Execution in Accelerator Mode” on page 1-5
- “Real-Time Execution in External Mode” on page 1-8

## Simulink Desktop Real-Time Product Description

### Run Simulink models in real time on your computer

Simulink Desktop Real-Time provides a real-time kernel for executing Simulink models on a Windows® or Mac laptop or desktop. It includes library blocks that connect to a range of I/O devices. You can create and tune a real-time system for rapid prototyping or hardware-in-the-loop simulation with your computer.

Simulink Desktop Real-Time supports real-time performance up to a 1 kHz sample rate with Simulink, and up to 20 kHz with Simulink Coder™.

### Key Features

- Real-time closed-loop execution of Simulink models
- Signal visualization and parameter tuning while model is running
- Real-time performance approaching a 1 kHz sample rate in Simulink Normal mode
- Real-time performance approaching a 20 kHz sample rate in Simulink External mode (with Simulink Coder)
- Blocks supporting more than 250 I/O devices (including analog I/O, digital I/O, counters, encoders, and frequency output) and communication protocols (including UDP, serial, and CAN)
- Connection to I/O devices installed in your computer or in a Thunderbolt expansion chassis

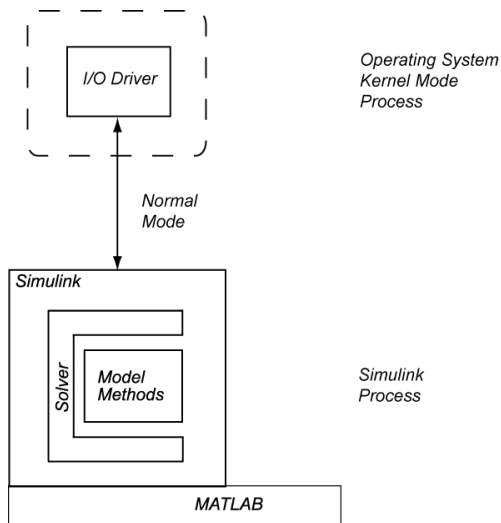
## Real-Time Execution in Normal Mode

Simulink Desktop Real-Time extends Simulink normal mode to run in real time.

The simulation algorithm for a non-real-time Simulink normal mode model runs entirely within Simulink. The model can use either a fixed-step or a variable-step solver and runs as fast as it can, given the presence of competing operating system processes. However, it is not synchronized with a real-time clock and cannot easily be used to operate real-time hardware.

You can synchronize a Simulink model with a real-time clock using Simulink Desktop Real-Time I/O blocks. In real-time normal mode, Simulink executes the simulation algorithm, while a separate operating system kernel mode process runs I/O drivers for the I/O blocks. Both the Simulink process and the kernel mode process run on the host machine, using a shared memory interface to transfer parameter data.

- Signal acquisition — You can capture and display signals from your real-time application while it is running. Simulink retrieves signal data from the I/O driver and displays it in the same **Scope** blocks you used for simulating your model in nonreal time.
- Parameter tuning — You can change parameters in your Simulink block diagram and have the new parameters take effect in your Simulink model in real time. The effects then propagate through the I/O driver to the hardware.



Because only the I/O drivers are synchronized with the real-time clock, Simulink can use either a fixed-step or a variable-step solver. The **Sample Time** setting in the Simulink Desktop Real-Time block does not change the step size of the simulation. For a fixed-step simulation, the step size is set in the **Fixed step size** box from the Configuration Parameters dialog box. For a variable-step simulation, the step size is determined automatically by Simulink or by the **Min Step Size** attribute.

In real-time normal mode, at each sample interval Simulink evaluates each real-time block and writes the input data into a buffer it passes to the kernel mode process. The kernel mode process propagates the data to the hardware, which writes response data into another buffer. At the next time tick, Simulink reads the response data and propagates it to the rest of the model.

A consequence of this kind of limited synchronization is that your simulation can be configured to miss real-time clock ticks and their associated data points. Ticks can be missed under the following circumstances:

- **Complexity of Model** — The model might be so complex that Simulink cannot keep up with the real-time kernel. In this case, the number of missed ticks increases steadily with time. Once the number of missed ticks exceeds **Maximum Missed Ticks**, an error occurs, even if **Maximum Missed Ticks** is set to a large value. This situation is marked by a rising straight line on a **Scope** connected to the optional **Missed Ticks** port.
- **Process Contention** — The model generally executes faster than required to keep up with the kernel, but process contention or some random operating system condition prevents Simulink from executing the model over some time period. In this case, the number of missed ticks jumps to some number, then decreases to zero as Simulink catches up with the kernel. This situation is marked by a sawtooth-like shape on a **Scope** connected to the **Missed Ticks** port.
- **Variable-Step Solver** — If you are using a variable-step solver, the number of ticks per algorithm step may vary during simulation. If Simulink execution does not reach the **Real-Time Sync** or I/O blocks in time to synchronize with the tick, the number of missed ticks jumps to some number, then decreases to zero as Simulink catches up with the kernel. As with process contention, this situation is marked by a sawtooth-like shape on a **Scope** connected to the **Missed Ticks** port.

## Real-Time Execution in Accelerator Mode

Simulink Desktop Real-Time extends Simulink accelerator mode to run in real time.

The simulation algorithm for a Simulink accelerator mode model is compiled to a MEX file S-function. As with normal mode, the S-function runs in the Simulink process. It is not synchronized with a real-time clock and cannot easily be used to operate real-time hardware.

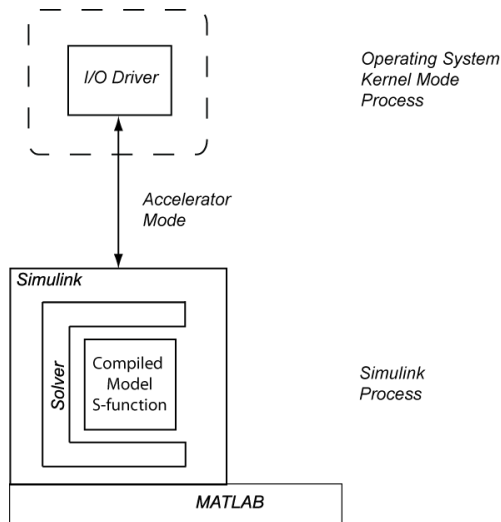
You can synchronize an accelerator mode model with a real-time clock using Simulink Desktop Real-Time I/O blocks. In real-time accelerator mode, the simulation algorithm S-function runs in the Simulink process, while a separate operating system kernel mode process runs I/O drivers for the I/O blocks. Both the Simulink process and the kernel mode process run on the host machine, using a shared memory interface to transfer parameter data.

- Signal acquisition — You can capture and display signals from your real-time application while it is running. Simulink retrieves signal data from the I/O driver and displays it in the same **Scope** blocks you used for simulating your model in nonreal time.
- Parameter tuning — You can change parameters in your Simulink block diagram and have the new parameters take effect in your Simulink model in real time. The effects then propagate through the I/O driver to the hardware.

---

**Note:** You cannot run a Simulink Desktop Real-Time model in rapid accelerator mode.

---



Because only the I/O drivers are synchronized with the real-time clock, Simulink can use either a fixed-step or a variable-step solver. The **Sample Time** setting in the Simulink Desktop Real-Time block does not change the step size of the simulation. For a fixed-step simulation, the step size is set in the **Fixed step size** box from the Configuration Parameters dialog box. For a variable-step simulation, the step size is determined automatically by Simulink or by the **Min Step Size** attribute.

In real-time accelerator mode, at each sample interval Simulink evaluates each real-time block and writes the input data into a buffer it passes to the kernel mode process. The kernel mode process propagates the data to the hardware, which writes response data into another buffer. At the next time tick, Simulink reads the response data and propagates it to the rest of the model.

A consequence of this kind of limited synchronization is that your simulation can be configured to miss real-time clock ticks and their associated data points. Ticks can be missed under the following circumstances:

- Complexity of Model — The model might be so complex that Simulink cannot keep up with the real-time kernel. In this case, the number of missed ticks increases steadily with time. Once the number of missed ticks exceeds **Maximum Missed Ticks**, an error occurs, even if **Maximum Missed Ticks** is set to a large value. This situation is marked by a rising straight line on a **SCOPE** connected to the optional **Missed Ticks** port.

- **Process Contention** — The model generally executes faster than required to keep up with the kernel, but process contention or some random operating system condition prevents Simulink from executing the model over some time period. In this case, the number of missed ticks jumps to some number, then decreases to zero as Simulink catches up with the kernel. This situation is marked by a sawtooth-like shape on a **Scope** connected to the **Missed Ticks** port.
- **Variable-Step Solver** — If you are using a variable-step solver, the number of ticks per algorithm step may vary during simulation. If Simulink execution does not reach the **Real-Time Sync** or **I/O** blocks in time to synchronize with the tick, the number of missed ticks jumps to some number, then decreases to zero as Simulink catches up with the kernel. As with process contention, this situation is marked by a sawtooth-like shape on a **Scope** connected to the **Missed Ticks** port.

## Real-Time Execution in External Mode

A higher-performance alternative to real-time normal mode is real-time external mode. In this mode, you use Simulink Coder to dynamically link generated algorithm code with I/O driver code generated from the I/O blocks. The resulting executable runs in operating system kernel mode on the host computer and exchanges parameter data with Simulink via a shared memory interface.

- Signal acquisition — You can capture and display signals from your real-time application while it is running. Signal data is retrieved from the real-time application and displayed in the same Simulink **Scope** blocks you used for simulating your model.
- Parameter tuning — You can change parameters in your Simulink block diagram and have the new parameters passed automatically to the real-time application. Simulink external mode changes parameters in your real-time application while it is running in real time.

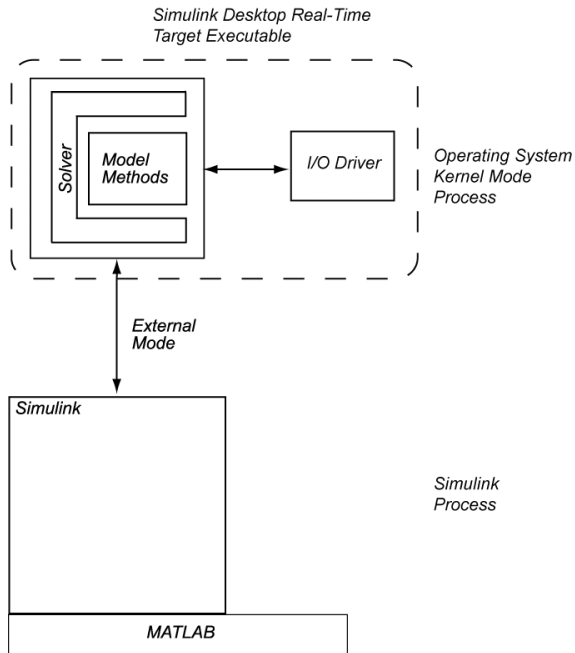
The external mode executable is fully synchronized with the real-time clock. The main role of Simulink is to read and display simulation results returned from the executable.

---

**Note:** You must use a fixed-step solver in external mode.

---





In external mode, the real-time application runs in the kernel mode process. Using I/O drivers running in that process to communicate with the hardware, it stores contiguous response data in memory accessible to Simulink until a data buffer is filled. When the buffer is filled, the real-time application continues to run while Simulink transfers the data to the MATLAB® environment through Simulink external mode. Transfer of data is less critical than maintaining deterministic real-time updates at the selected sample interval. Therefore, data transfer runs at a lower priority in the remaining CPU time after model computations are performed while waiting for another interrupt to trigger the next model update.

Data captured within one buffer is contiguous. When a buffer of data has been transferred, it is immediately plotted in a Simulink **Scope** block, or it can be saved directly to a MAT-file using the data archiving feature of the Simulink external mode.

With data archiving, each buffer of data can be saved to its own MAT-file. The MAT-file names can be automatically incremented, allowing you to capture and automatically store many data buffers. Although points within a buffer are contiguous, the time required to transfer data back to Simulink forces an intermission for data collection until

the entire buffer has been transferred and may result in lost sample points between data buffers.

# Installation and Configuration

---

- “Software Components” on page 2-2
- “Install Real-Time Kernel” on page 2-4
- “Run Confidence Test” on page 2-7

# Software Components

Simulink Desktop Real-Time is a self-targeting rapid prototyping system where the host and the target computer are the same computer.

## MATLAB Environment

The MATLAB environment provides the design and data analysis tools that you use when creating and testing Simulink models. In particular, see:

- “Importing and Exporting Data”
- “Plotting Data”

## Simulink Software

Simulink software provides an environment where you model your physical system and controller as a block diagram. You create the block diagram by using a mouse to connect blocks and a keyboard to edit block parameters. You can use Simulink Desktop Real-Time software with most Simulink blocks, including discrete-time and continuous-time systems. C code S-functions are supported by Simulink Coder code generation software.

With Simulink Desktop Real-Time software, you can remove the physical system model and replace it with Simulink Desktop Real-Time I/O driver blocks connected to your sensors and actuators. The Simulink Desktop Real-Time I/O library supports more than 200 boards.

---

**Note:** Some of the functions on a board may not be supported by Simulink Desktop Real-Time software. Check the MathWorks® web site for an updated list of supported boards and functions at [www.mathworks.com/hardware-support/simulink-desktop-real-time.html](http://www.mathworks.com/hardware-support/simulink-desktop-real-time.html).

---

## Simulink Coder Software

Simulink Coder code generation software provides the utilities to convert your Simulink models into C code and then compile the code into a real-time executable.

---

**Note:**

- Simulink Coder is required for external mode.
  - Compiler support is included as part of the product installation. No additional or external compiler is required.
  - MATLAB Coder is required for Simulink Coder installation.
- 

Simulink Desktop Real-Time software is designed for maximum flexibility during rapid prototyping. This flexibility allows parameter tuning and signal tracing during a real-time run, but increases the size of the generated code. However, Simulink Coder code generation software provides other code formats that generate more compact code for embedded applications.

## Known Limitations

- In external mode, the Simulink Desktop Real-Time software does not support the following:
  - Simscape™ or Simscape Driveline™ blocks
  - Blocks that do not run in real time
  - To File blocks
- Limitations with Simulink Coder code generation software:
  - In external mode, MATLAB S-functions are not supported.
  - When you use a continuous-time system and generate code for external mode execution with Simulink Coder code generation software, you must use a fixed-step integration algorithm.
  - The Simulink Coder product provides an API for the MATLAB Distributed Computing Server™ or Parallel Computing Toolbox™ products to perform parallel builds that reduce build time for referenced models. However, this API does not support parallel builds for models whose system target file parameter is set to `sldrt.tlc` or `sldrtert.tlc`. In other words, you cannot perform parallel builds for Simulink Desktop Real-Time.

# Install Real-Time Kernel

The Simulink Desktop Real-Time software requires a real-time kernel that interfaces with the operating system. The Simulink Desktop Real-Time kernel assigns the highest priority of execution to your real-time executable, which allows it to run without interference at the selected sample rate. During real-time execution of your model, the kernel intervenes to give the model priority to use the CPU to execute each model update at the prescribed sample times. Once a model update completes, the kernel releases the CPU to run other operating system-based applications that might need servicing.

### In this section...

“Install the Kernel Using MATLAB” on page 2-4

“Uninstall the Kernel” on page 2-5

## Install the Kernel Using MATLAB

You must install the kernel before you can run a Simulink Desktop Real-Time application. During software installation, the Simulink Desktop Real-Time software is copied onto your hard drive, but the Simulink Desktop Real-Time kernel is not automatically installed into the operating system.

Installing the kernel configures it to start running in the background each time you start your computer. The following procedure describes how to use the command `sldrtkernel -install`. (You can also use the command `sldrtkernel -setup` instead.) To install the kernel:

- 1 In the MATLAB Command Window, type:

```
sldrtkernel -install
```

The MATLAB Command Window displays one of these messages:

```
You are going to install the Simulink Desktop Real-Time kernel.  
Do you want to proceed? [y] :
```

**or:**

```
There is a different version of the Simulink Desktop Real-Time kernel installed.  
Do you want to update to the current version? [y] :
```

- 2 Type **y** to continue installing the kernel, or **n** to cancel installation without making changes.

If you type `y`, the MATLAB environment installs the kernel and displays the message:

The Simulink Desktop Real-Time kernel has been successfully installed.

- 3 If a message appears asking you to restart your computer, do so before attempting to use the kernel, or your Simulink Desktop Real-Time model will not run.
- 4 After installing the kernel, verify that it was installed by typing:

```
rtwho
```

The MATLAB Command Window should display a message that shows the kernel version number, followed by timer, driver, and other information.

Once the kernel is installed, you can leave it installed. The kernel remains idle after you have installed it, which allows the operating system to control the execution of standard applications, such as Internet browsers, word processors, the MATLAB environment, and so on. The kernel becomes active when you begin execution of your model, and becomes idle again after model execution completes.

## Uninstall the Kernel

If you encounter problems with Simulink Desktop Real-Time software, you can uninstall the kernel. Once uninstalled, the kernel is no longer active. The kernel executable file remains on your hard drive so that you can later reinstall it.

### Uninstall the Kernel Using MATLAB

To uninstall the kernel from MATLAB:

- 1 In the MATLAB Command Window, type:

```
sldrtkernel -uninstall
```

The MATLAB Command Window displays the message:

```
You are going to uninstall the Simulink Desktop Real-Time kernel.  
Do you want to proceed? [y]:
```

- 2 Type `y` to continue uninstalling the kernel, or `n` to cancel uninstallation without making changes.

If you type `y`, the MATLAB environment uninstalls the kernel by removing it from memory, then displays the message:

The Simulink Desktop Real-Time kernel has been successfully uninstalled.

- 3 After uninstalling the kernel, verify that it was uninstalled. Type:

```
rtwho
```

The MATLAB Command Window displays the message:

```
Simulink Desktop Real-Time installation is not complete.  
Please type 'sldrtkernel -setup' to complete the installation.  
Type 'help sldrtkernel' for more information.
```

### **Uninstall the Kernel Using Host Computer Command Line**

Uninstalling the MATLAB environment does not uninstall the Simulink Desktop Real-Time kernel. If you have uninstalled the MATLAB environment and need to uninstall the kernel, launch a command window and type:

```
sldrtkernel -uninstall
```

The `sldrtkernel` program uninstalls the kernel by removing it from memory, then displays the message:

```
The Simulink Desktop Real-Time kernel uninstalled successfully.
```

This procedure works only with the Windows operating system.



## Run Confidence Test

Simulink Desktop Real-Time includes several example models you can use to test your installation. These models are preconfigured with settings such as target and scope settings, sample time, and integration algorithm. To see these models, type `sldrtexamples` in the MATLAB Command Window.

---

**Note:**

- You can run examples in Simulink normal mode (initial setting), accelerator mode, or Simulink external mode.
  - You cannot run a Simulink Desktop Real-Time model in rapid accelerator mode.
- 

Once you have finished installing the Simulink Desktop Real-Time software and kernel, you should test the installation by running the model `sldrtex_vdp`. If you change your installation, you should repeat this test to confirm that the Simulink Desktop Real-Time software is still working. To open the example model, type `sldrtex_vdp` in the MATLAB Command Window, or launch MATLAB Help, open **Simulink Desktop Real-Time**, and choose **Examples > Real-Time Van der Pol Simulation**.

**In this section...**

“Run the Model `sldrtex_vdp`” on page 2-7

“Display Status Information” on page 2-9

“Examples Library” on page 2-10

### Run the Model `sldrtex_vdp`

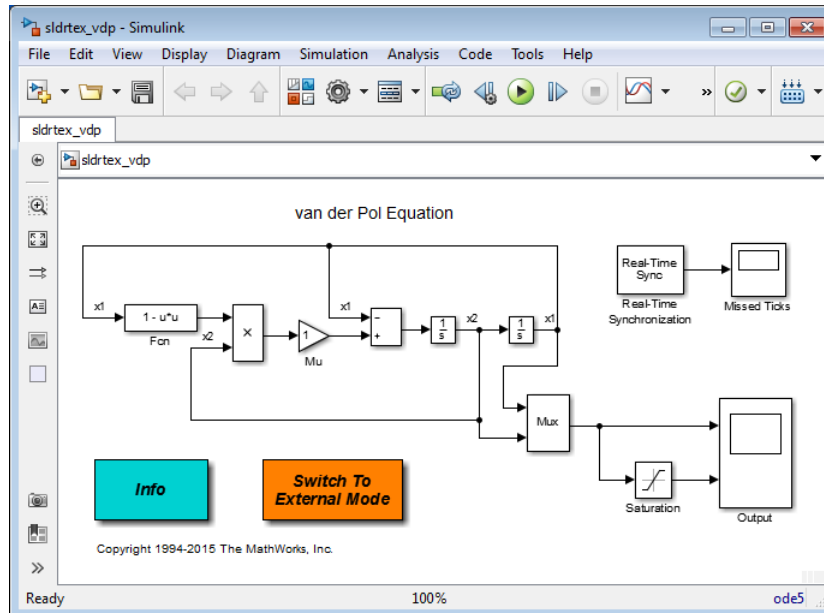
The model `sldrtex_vdp` does not have I/O blocks, so that you can run this model regardless of the I/O boards in your computer. Running this model will test the installation by running Simulink Coder code generation software, Simulink Desktop Real-Time software, and the Simulink Desktop Real-Time kernel.

After you have installed the Simulink Desktop Real-Time kernel, you can test the entire installation by building and running a real-time application. The Simulink Desktop Real-Time software includes the model `sldrtex_vdp`, which already has Simulink Coder options selected for you:

- 1 In the MATLAB Command Window, type

`sldrtex_vdp`

The Simulink model `sldrtex_vdp` window opens.




- 2 Double click **Switch To External Mode** on the model.

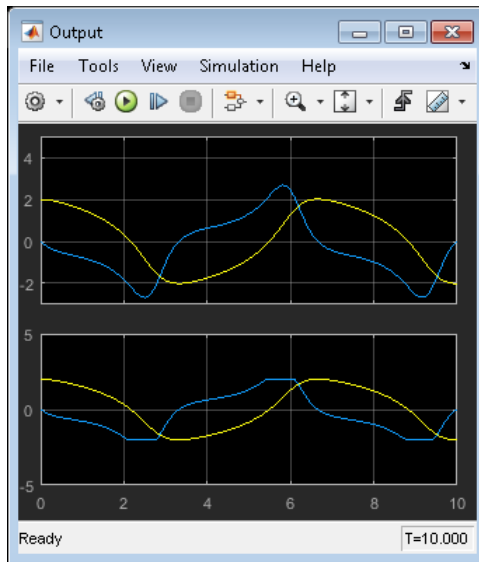
---


**Tip** The model is initially in normal mode. At this point, you might start real-time normal mode simulation by clicking **Simulation > Run**.

---

- 3 Click the Run button  on the toolbar.

After building the model and displaying messages in the Diagnostic Viewer, the target application begins running.



- 4 To stop the simulation before the end, click the Stop button  on the toolbar.

The real-time application stops running, and the Scope window stops displaying the output signals.

## Display Status Information

The Simulink Desktop Real-Time software provides the command `rtwho` for displaying the kernel version number, followed by timer, driver, and other information. To see this information, in the MATLAB Command Window type

```
rtwho
```

The command displays several lines of information in the MATLAB Command Window. Some possible lines and their interpretations are:

```
TIMERS:  Number    Period  Running
          1         0.01    Yes
```

The indicated timer(s) exist on your system with the period and run status shown for each timer.

```
DRIVERS:           Name    Address  Parameters
```

```
Humusoft AD512      0x300  []  
          ecg        0      []
```

The indicated device driver(s) are installed on your system at the address and with the parameter(s) shown for each driver.

### Examples Library

The examples library includes models with preset values and dialog boxes. These models include simple signal processing and simple control examples that use no I/O blocks, use A/D blocks only, and use both A/D and D/A blocks.

To run an example that uses I/O blocks, you must configure the Adapter block to match the I/O board installed in your computer.

---

#### Note:

- You can run examples in Simulink normal mode (initial setting), accelerator mode, or Simulink external mode.
  - You cannot run a Simulink Desktop Real-Time model in rapid accelerator mode.
- 

To see these models from the MATLAB environment:

- 1 Type `sldrtexamples` in the MATLAB Command Window.  
The Simulink Desktop Real-Time Demos window displays.
- 2 From the list, select the example to open it.

# Basic Procedures

---

- “Prepare for Real-Time Execution” on page 3-2
- “Create a Real-Time Application” on page 3-4
- “Create a Simulink Model” on page 3-5
- “Configure a Model” on page 3-9
- “Run Normal Mode Simulation” on page 3-15
- “Run Accelerator Mode Simulation” on page 3-17
- “Set External Mode Code Generation Parameters” on page 3-19
- “Set Scope Parameters” on page 3-22
- “Prepare External Mode Application” on page 3-25
- “Set External Mode Scope Parameters” on page 3-26
- “Run Application from Simulink” on page 3-29
- “Run Application from MATLAB Command Line” on page 3-31
- “Inspect Simulink Desktop Real-Time Signals with Simulation Data Inspector” on page 3-33
- “Signal Logging to the Workspace” on page 3-36
- “Set Scope Parameters for Logging to Workspace” on page 3-37
- “Set External Mode Properties for Logging to Workspace” on page 3-39
- “Plot Signal Data Logged to Workspace” on page 3-41
- “Signal Logging to a File” on page 3-43
- “Set Scope Parameters for Logging to File” on page 3-44
- “Set External Mode Properties for Logging to File” on page 3-47
- “Set External Mode Data Archiving Parameters” on page 3-49
- “Plot Signal Data Logged to File” on page 3-51
- “Tune Block Parameters and MATLAB Variables” on page 3-53
- “Change Block Parameters Using Simulink Dialog Box” on page 3-54

## Prepare for Real-Time Execution

To observe how Simulink models respond to real-world behavior, connect the real-time application to physical I/O devices. You have access to a library of I/O driver blocks that provide connections between devices and applications. To prepare for real-time execution, install I/O devices in your computer and select the corresponding Simulink Desktop Real-Time library blocks.

To prepare your computer:

- 1 Choose I/O plug-in modules from [www.mathworks.com/hardware-support/simulink-desktop-real-time.html](http://www.mathworks.com/hardware-support/simulink-desktop-real-time.html).
- 2 Acquire the modules and install them in your computer.
- 3 Refer to the vendor documentation for vendor-specific requirements.

If the hardware requires the installation of vendor software, install the vendor software on your computer.

- 4 Restart your computer and, from the MATLAB Command Window, start the Simulink Desktop Real-Time kernel.

To prepare your real-time application:

- 1 Replace Simulink I/O blocks with Simulink Desktop Real-Time blocks representing the functionality of your I/O modules.
- 2 Open the dialog box for each block and associate the block with the driver for the corresponding I/O module.

Set the other block parameters as required by your model.

- 3 In the Configuration Parameters dialog box, set the parameters in the **Code Generation** pane to support Simulink Desktop Real-Time code generation.
- 4 Set the Simulink Desktop Real-Time configuration parameters as required by your model.
- 5 In Simulink, set the simulation mode to normal, accelerator, or external mode to attain the required sample rate.

---

**Note:** You cannot run a Simulink Desktop Real-Time model in rapid accelerator mode.

---

6 Start the simulation.

### **Related Examples**

- “Create a Real-Time Application” on page 3-4

# Create a Real-Time Application

A Simulink model is a graphical representation of your physical system. You create a Simulink model for a non-real-time simulation of your system, and then you use the Simulink model to create a real-time application. This example uses `sldrtex_model`. To recreate this model, see .

To run `sldrtex_model` in real-time normal or accelerator mode:

- 1 “Configure a Model” on page 3-9
- 2 “Set Scope Parameters” on page 3-22
- 3 “Run Normal Mode Simulation” on page 3-15
- 4 “Run Accelerator Mode Simulation” on page 3-17

---

**Note:** You cannot run a Simulink Desktop Real-Time model in rapid accelerator mode.

---

To run `sldrtex_model` as a real-time application:

- 1 “Set External Mode Code Generation Parameters” on page 3-19
- 2 “Prepare External Mode Application” on page 3-25
- 3 “Set External Mode Scope Parameters” on page 3-26
- 4 “Run Application from Simulink” on page 3-29

## Related Examples

- “Create a Simulink Model” on page 3-5
- “Run Application from MATLAB Command Line” on page 3-31



## Create a Simulink Model

This procedure explains how to create a simple Simulink model. You can use this model as an example to learn other procedures that are useful with Simulink Desktop Real-Time software.

- 1 In the **MATLAB Home** tab, click the **Simulink** button.
- 2 Click **Blank Model**, and then **Create Model**.

An empty Simulink window opens.

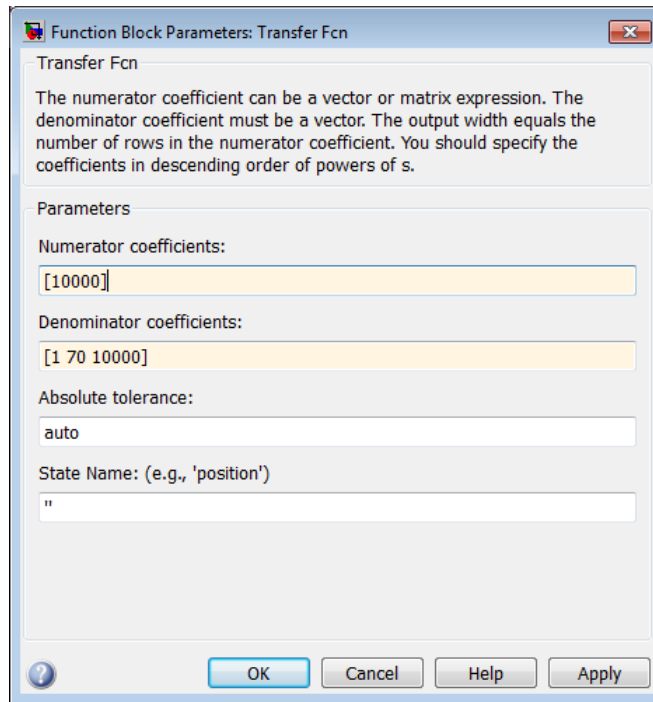
- 3 In the toolbar, open the Simulink Library Browser.
- 4 In the Library Browser:
  - Select **Simulink > Sources**, and then add a **Signal Generator** block to the model.
  - Select **Simulink > Continuous**, and then add a **Transfer Fcn** block to the model.
  - Select **Simulink > Sinks**, and then add a **Scope** block to the model.
- 5 Make the following block-to-block connections:
  - **Signal Generator** output to **Transfer Fcn** input
  - **Transfer Fcn** output to **Scope** input
- 6 Double-click the **Transfer Fcn** block. The **Block Parameters** dialog box opens. In the **Numerator** text box, enter:

[10000]

In the **Denominator** text box, enter

[1 70 10000]

Your **Block Parameters** dialog box looks like this:



- 7 Click **OK**.
- 8 Double-click the **Signal Generator** block. The Block Parameters dialog box opens. From the **Wave form** list, select **square**.

In the **Amplitude** text box, enter

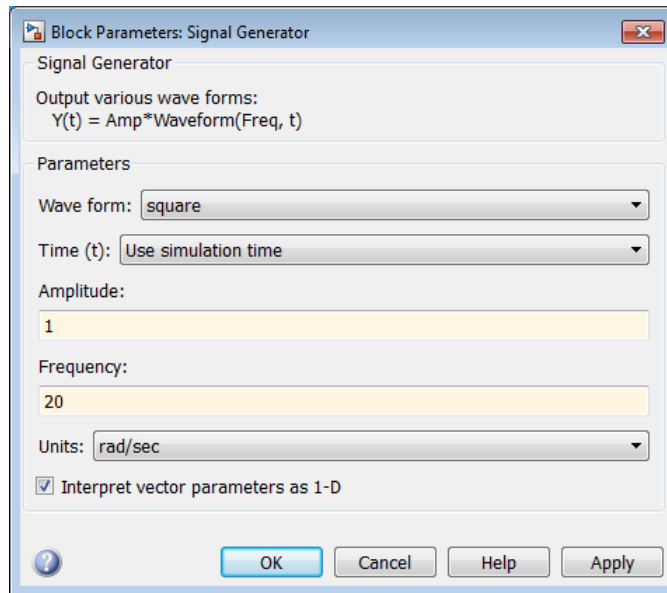
1

In the **Frequency** text box, enter

20

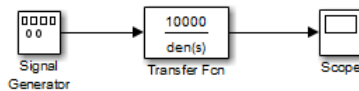
From the **Units** list, select **rad/sec**.

Your Block Parameters dialog box looks like this:



- 9 Click **OK**.

The completed Simulink block diagram looks like the figure.



- 10 From the **File** menu, click **Save As**. The Save As dialog box opens. In the **File name** text box, enter a file name for your Simulink model and click **Save**. For example, type

sldrtext\_model

The Simulink software saves your model in the file `sldrtext_model`.

To specify a default Simulink Desktop Real-Time configuration set for your model, see “Specify a Default Configuration Set” on page 3-9. If you activate this configuration set for your model, you can build your real-time application later without setting additional configuration parameters.

To manually configure your model, continue to “Enter Configuration Parameters Manually” on page 3-10, following. That section teaches you how to enter configuration parameters for your Simulink model, then leads you into procedures for entering scope parameters and running a non-real-time simulation of the model.

### Reference a Model

The Simulink Desktop Real-Time software supports model referencing. See “Overview of Model Referencing”.

### File System Input/Output

When run in external mode, the Simulink Desktop Real-Time software does not include a file system. Therefore, a Simulink model intended for use in a Simulink Desktop Real-Time application cannot use blocks, such as the `To File` or `From File` block, that generate file I/O calls such as `fopen` or `fprintf`.

If a Simulink Desktop Real-Time model contains a block that tries to perform file system I/O, an error could occur when you try to compile the model, generate code for it, or use external mode with it. Even if no error occurs, the block has no effect on either simulation or code execution.

You can use file I/O blocks in Simulink normal or accelerator mode simulation in real time. In these modes, the file I/O is not carried out by the real-time kernel, but by Simulink itself. To access files in external mode, use the `Packet Input`, `Packet Output`, `Stream Input`, or `Stream Output` blocks and select the driver **Standard Devices > File**.

To log signal data without needing a file system, use the techniques described in “Signal Logging to the Workspace” on page 3-36. See “Run Application from Simulink” on page 3-29 for information about using external mode to execute a Simulink Desktop Real-Time application.

## Configure a Model

After you create a Simulink model, you can enter configuration parameters for the model. These parameters control many properties of the model for simulation and code generation. This section contains the essential information you need when setting configuration parameters for a Simulink Desktop Real-Time application. For information about Simulink configuration, see “Manage a Configuration Set”.

A configuration set is a named set of values for model parameters, such as solver type and simulation start or stop time. Every new model is created with a default configuration set, called **Configuration**, that initially specifies default values for the model's model parameters. You can subsequently create additional configuration sets and associate them with the model, as described in “Manage a Configuration Reference”.

The easiest way to specify configuration parameters for a Simulink Desktop Real-Time model is to programmatically assign the default Simulink Desktop Real-Time configuration set, as described in “Specify a Default Configuration Set” on page 3-9. You can also set parameters manually, as described in “Enter Configuration Parameters Manually” on page 3-10. Other sections describe setting configuration patterns for specific purposes.

### Specify a Default Configuration Set

After you create a Simulink model, you can use the `sldrtconfigset` function to specify a default Simulink Desktop Real-Time configuration set for the model. In most cases, using `sldrtconfigset` provides the configuration parameter values that the model needs.

The following procedure uses the Simulink model `sldrtex_model` as an example and assumes you have already loaded that model (see “Create a Simulink Model” on page 3-5).

- 1 If you have not already saved the model, from the **File** menu, click **Save As**. The Save As dialog box opens. In the **File name** text box, enter a file name for your Simulink model and click **Save**. For example, type

```
sldrtex_model
```

The Simulink software saves your model in the file `sldrtex_model`.

- 2 In the MATLAB Command Window, type

```
sldrtconfigset('sldrtex_model')
```

The default Simulink Desktop Real-Time configuration set, `SimulinkDesktopRealTime`, is now active for the `sldrtex_model` model. Alternatively, you can set the default Simulink Desktop Real-Time configuration set by setting the Configuration Parameters **System target file** option to `sldrt.tlc`. You do not need to perform other configuration for a Simulink Desktop Real-Time application.

#### 3 Save the model.

See “Create a Real-Time Application” on page 3-4 for a description of how to build your Simulink Desktop Real-Time application.

To revert to the default configuration set, Configuration, or other configuration set you have for the model, use Model Explorer. This is an alternative tool that you can use to enter simulation parameters for a model. See the Simulink documentation for a description of how to use Model Explorer.

---

**Note:** Your model uses a Simulink Desktop Real-Time configuration set when you change the **System target file** value to a Simulink Desktop Real-Time one, such as `sldrt.tlc` or `sldrtert.tlc`. The software creates the Simulink Desktop Real-Time configuration set, `SimulinkDesktopRealTime` or `SimulinkDesktopRealTime_ERT`, only if one does not already exist.

---

## Enter Configuration Parameters Manually

The configuration parameters give information to Simulink software for running a simulation.

This procedure uses the Simulink model `sldrtex_model` as an example and assumes you have already loaded that model.

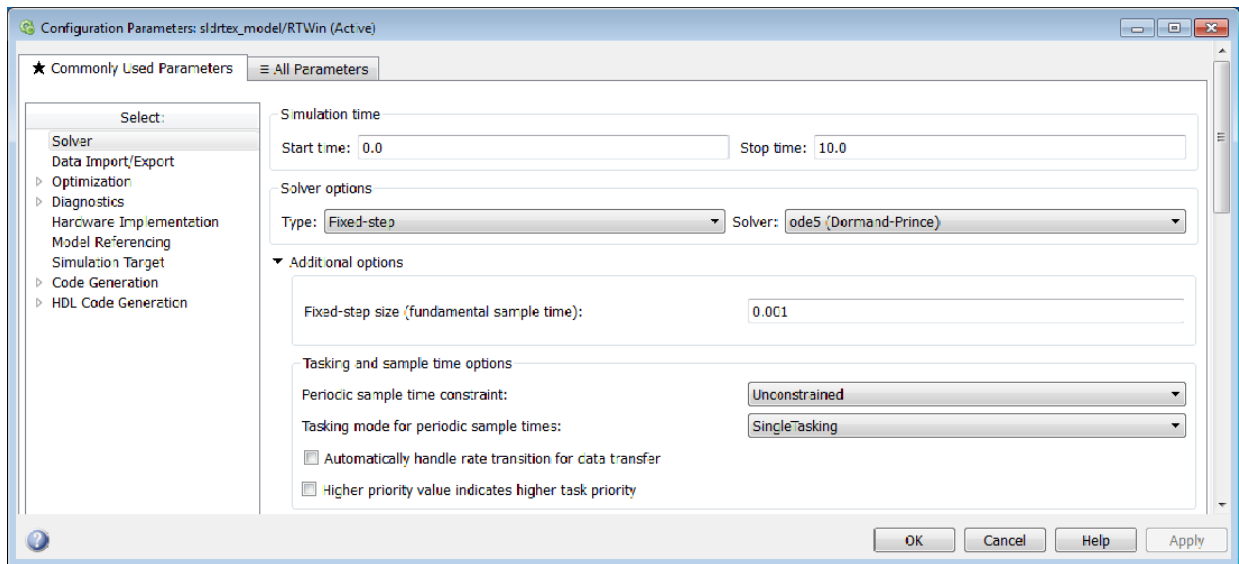
- 1 In the Simulink window, and from the **Simulation** menu, click **Model Configuration Parameters**. In the Configuration Parameters dialog box, click the **Solver** tab.

The **Solver** pane opens.

- 2 In the **Start time** field, enter `0.0`. In the **Stop time** field, enter the amount of time you want your model to run. For example, enter `10.0` seconds.

- 3 From the **Type** list, choose **Fixed-step**. Simulink Coder code generation software does not support variable step solvers.
- 4 From the **Solver** list, choose a solver. For example, choose the general purpose solver **ode5 (Dormand-Prince)**.
- 5 Under **Additional options**, in the **Fixed step size** field, enter a sample time. For example, enter **0.001** seconds for a sample rate of 1000 samples/second.
- 6 From the **Tasking Mode** list, choose **SingleTasking**. (For models with blocks that have different sample times, choose **MultiTasking**.)

Your **Solver** pane looks like this:



- 7 Do one of the following:
  - Click **Apply** to apply the changes to your model and leave the dialog box open.
  - Click **OK** to apply the changes to your model and close the dialog box.


## Enter Scope Parameters for Signal Tracing

You enter or change scope parameters to specify the  $x$ -axis and  $y$ -axis in a Scope window. Other properties include the number of graphs in one Scope window and the sample time for models with discrete blocks.

After you add a **Scope** block to your Simulink model, you can enter the scope parameters for signal tracing:

- 1 In the Simulink window, double-click the **Scope** block.

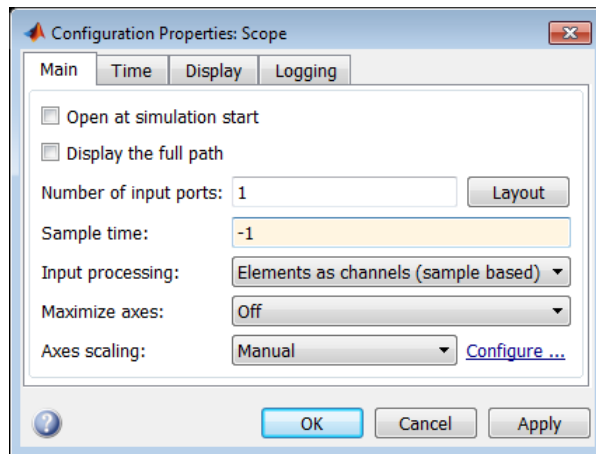
A Scope window opens.

- 2 Click the Parameters button  on the toolbar.

A Scope parameters dialog box opens.

- 3 Click the **Main** tab. In the **Sample time** text box, enter -1. Entering -1 indicates that this block inherits its value from its parent model. If you have discrete blocks in your model, enter the **Fixed step size** value you entered in the Configuration Parameters dialog box.

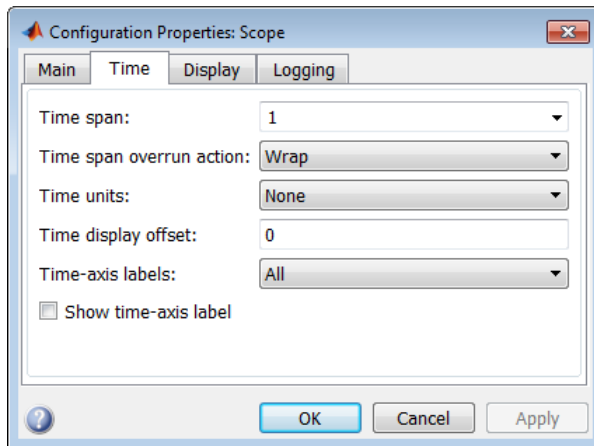
Your **Main** tab looks like this:



- 4 Click the **Time** tab. In the **Time span** box, enter 1.

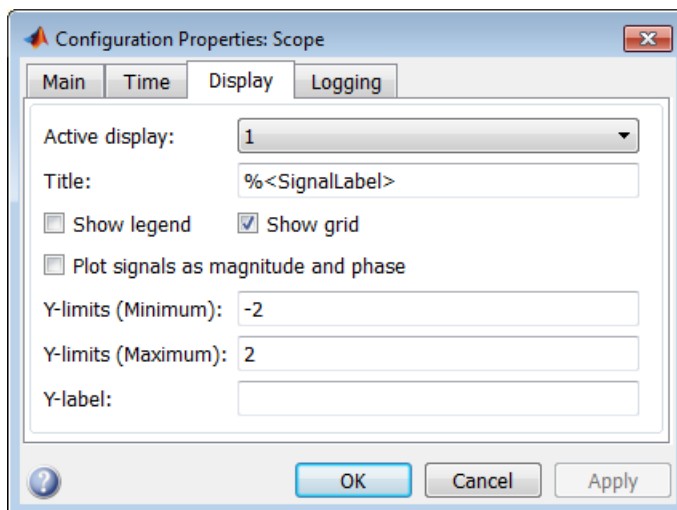
Your **Time** tab looks like this:





- 5 Click the **Display** tab. In the **Y-min** and **Y-max** text boxes, enter the range for the y-axis in the Scope window. For example, enter -2 and 2.

Your **Display** tab looks like this:



- 6 Do one of the following:
- Click **Apply** to apply the changes to your model and leave the dialog box open.


- Click **OK** to apply the changes to your model and close the dialog box.

## Run Normal Mode Simulation

You can use Simulink normal mode to run a real-time simulation. This procedure uses the Simulink model `sldrtext_model` as an example and assumes you have loaded that model.

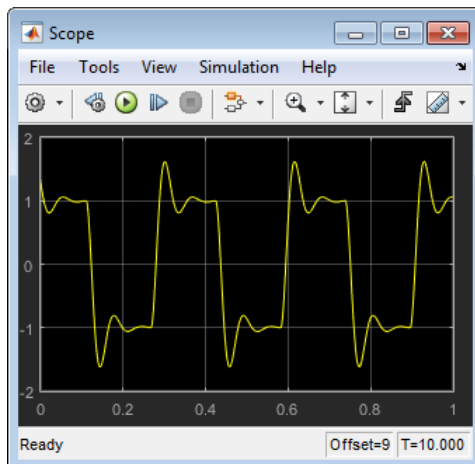
- 1 In the Simulink window, double-click the Scope block.


The Simulink software opens a Scope window with an empty graph.

- 2 Select **Simulation > Mode > Normal**.
- 3 In your Simulink Desktop Real-Time model, set values for the **Sample Time** and **Maximum Missed Ticks** block parameters to prevent missed ticks.
- 4 Select **Simulation > Run** to begin simulation or click the Run button  on the toolbar.

The Simulink software runs the simulation and plots the signal data in the Scope window.

During the simulation, the Scope window displays the samples for one time range, increases the time offset, and then displays the samples for the next time range.



- 5 To stop the simulation before the end, click the Stop button  on the toolbar.


The real-time application stops running, and the Scope window stops displaying the output signals.

## Run Accelerator Mode Simulation

You use Simulink accelerator mode to run a real-time simulation. This procedure uses the Simulink model `sldrtext_model` as an example and assumes you have loaded that model.

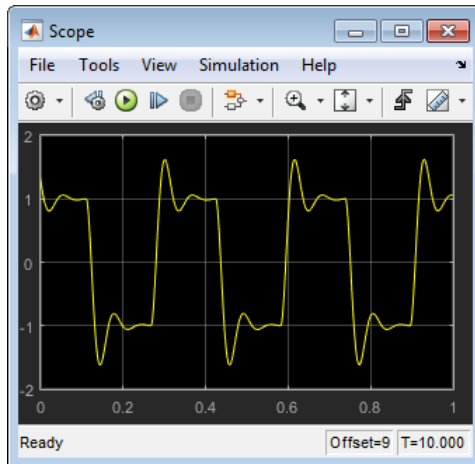
- 1 In the Simulink window, double-click the Scope block.


The Simulink software opens a Scope window with an empty graph.

- 2 Select **Simulation > Mode > Accelerator**.
- 3 In your Simulink Desktop Real-Time model, set values for the **Sample Time** and **Maximum Missed Ticks** block parameters to prevent missed ticks.
- 4 Select **Simulation > Run** to begin simulation or click the Run button  on the toolbar.

The Simulink software runs the simulation and plots the signal data in the Scope window.

During the simulation, the Scope window displays the samples for one time range, increases the time offset, and then displays the samples for the next time range.



- 5 To stop the simulation before the end, click the Stop button  on the toolbar.

The real-time application stops running, and the Scope window stops displaying the output signals.

## Set External Mode Code Generation Parameters

After you create a Simulink model, you can enter simulation parameters for use by Simulink Coder code generation software for creating C code and building a real-time application.

This procedure uses the Simulink model `sldrtext_model` as an example and assumes you have already loaded that model.

- 1 In the Simulink window, and from the **Simulation** menu, click **Model Configuration Parameters**.
- 2 Click the **Code Generation** node.

The **Code Generation** pane opens.

- 3 In the **Target selection** section, click the **Browse** button at the **System target file** list.

The **System Target File Browser** opens.

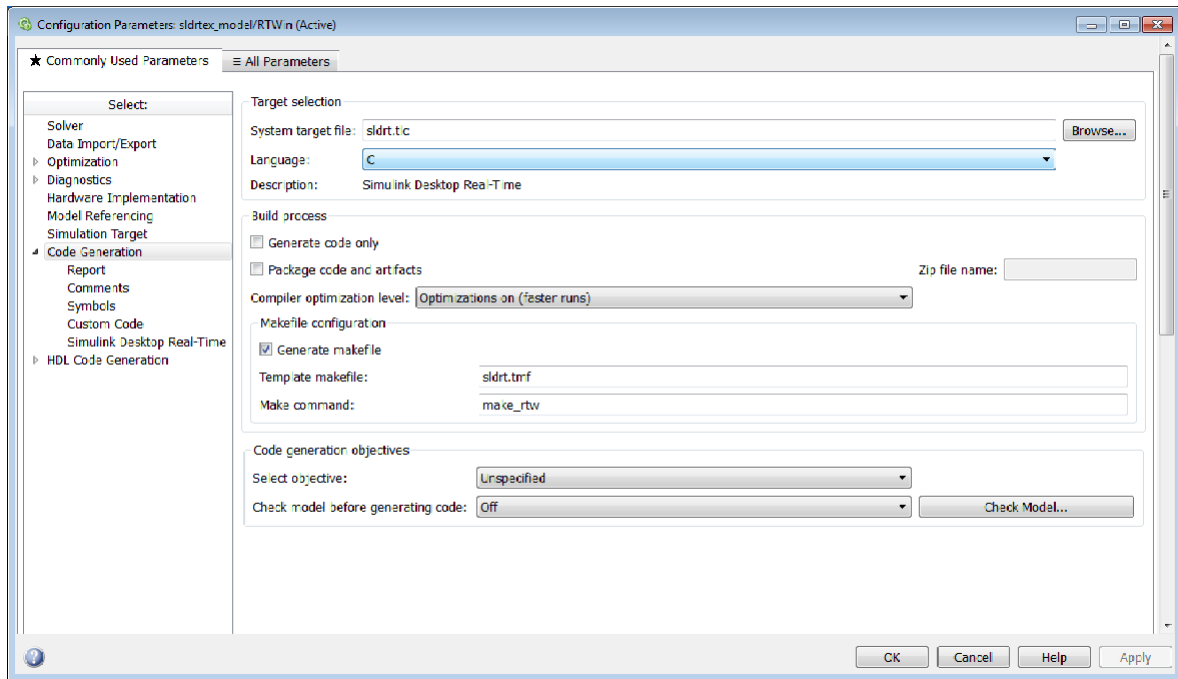
- 4 Select the system target file for building a Simulink Desktop Real-Time application, `sldrtext.tlc`, and click **OK**.

The dialog automatically enters the system target file `sldrtext.tlc`, the template makefile `sldrtext.tmf`, and the make command `make_rtw` into the **Code Generation** pane.

If you have the Embedded Coder<sup>®</sup> product, you can build an ERT target application. To build an ERT target application, in the **Target selection** section, click the **Browse** button at the **System target file** list. Click `sldrtertext.tlc`, and then click **OK**.

Although not visible in the **Code Generation** pane, when you click **OK** you also configure the external target interface MEX file `sldrtext`. This allows external mode to pass new parameters to the real-time application and to return signal data from the real-time application. The data is displayed in **Scope** blocks or saved with signal logging.

Your **Code Generation** pane looks like this:



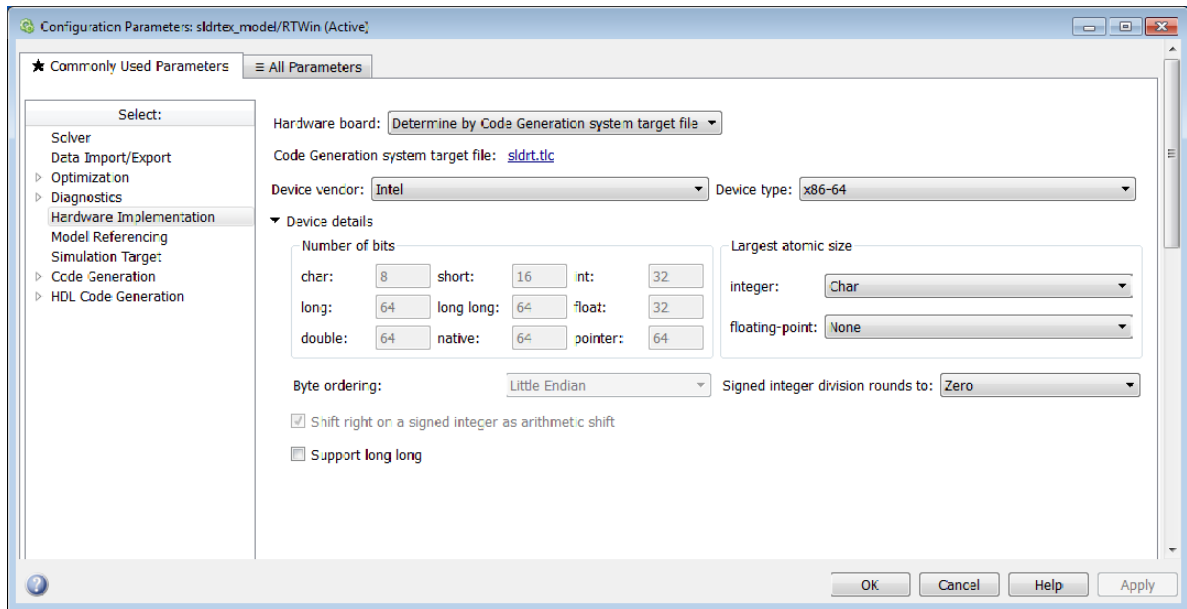
Do not set **Default parameter behavior** to **Inlined** on the **Signals and Parameters** node under **Optimization**. Inlining parameters is used for custom targets when you want to reduce the amount of RAM or ROM with embedded systems. Also, if you select inlining parameters, you disable the parameter tuning feature. Since PCs have more memory than embedded systems, you should not inline parameters.

- 5 Click the **Hardware Implementation** node. The default values are derived from the architecture of the host computer. For example, for a 64-bit Intel<sup>®</sup> machine, they are:

- **Device vendor** — Intel
- **Device type** — x86-64

Your **Hardware Implementation** pane looks like this:





6 Do one of the following:

- Click **Apply** to apply the changes to your model and leave the dialog box open.
- Click **OK** to apply the changes to your model and close the dialog box.

## Set Scope Parameters

You enter or change scope parameters to format the  $x$ -axis and  $y$ -axis in a Scope window. Other parameters include the number of graphs in a one Scope window and whether the scope is connected to a continuous or discrete model.


This procedure uses the Simulink model `sldrtex_model` as an example and assumes you have already loaded that model.

If you entered the scope parameters for running a simulation, you can skip this procedure. This information is repeated here if you did not run a simulation. This procedure is the same for normal, accelerator, and external modes.

After you add a **Scope** block to your Simulink model, you can enter the scope parameters for signal tracing:

- 1 In the Simulink window, double-click the **Scope** block.

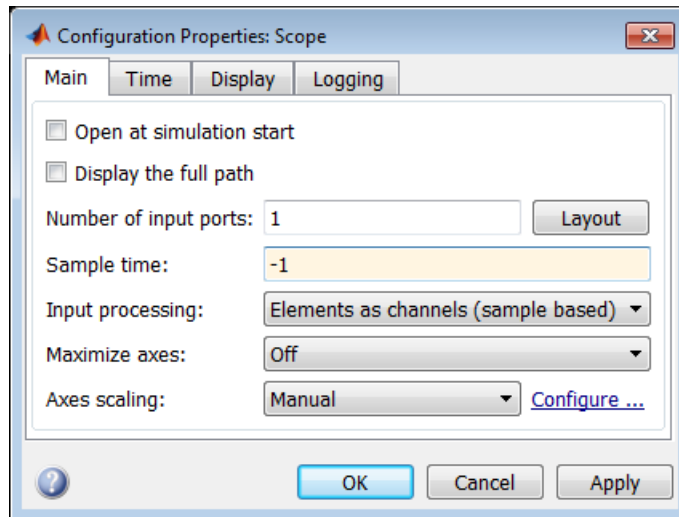
A Scope window opens.

- 2 Click the Parameters button  on the toolbar.

A Scope parameters dialog box opens.

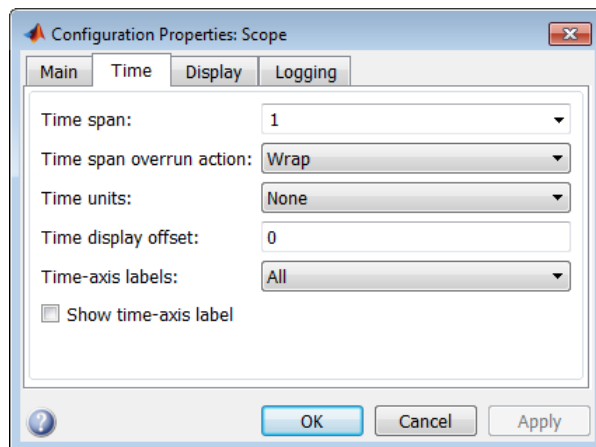
- 3 Click the **Main** tab. In the **Sample time** box, enter `-1`. Entering `-1` indicates that this block inherits its value from its parent model. If you have discrete blocks in your model, enter the **Fixed step size** value you entered in the Configuration Parameters dialog box.

Your **Main** tab looks like this:



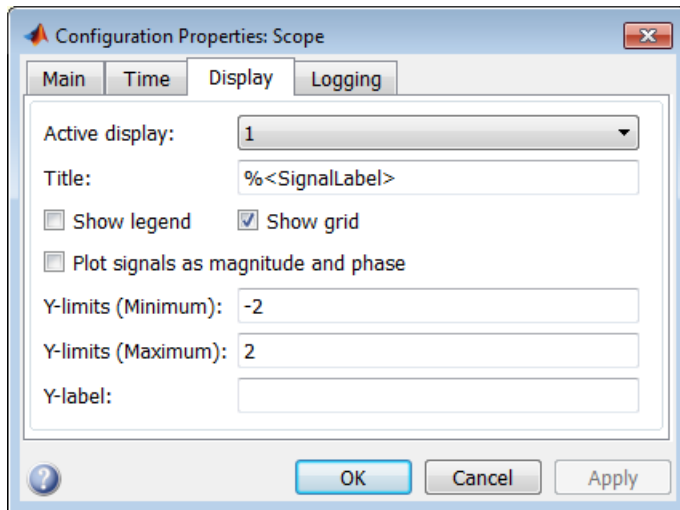
- 4 Click the **Time** tab. In the **Time span** box, enter 1.

Your **Time** tab looks like this:



- 5 Click the **Display** tab. In the **Y-min** and **Y-max** text boxes enter the range for the y-axis in the Scope window. For example, enter -2 and 2.

Your Scope **Display** tab looks like this:




- 6 Do one of the following:
- Click **Apply** to apply the changes to your model and leave the dialog box open.
  - Click **OK** to apply the changes to your model and close the dialog box.

## Prepare External Mode Application

In external mode, you must first create an executable target application. The Simulink Coder code generation software creates C code from your Simulink model, then the bundled C compiler compiles and links that C code into a real-time application.

This procedure uses the Simulink model `sldrtext_model` as an example and assumes you have already loaded that model.

- 1 Enter parameters into the Configuration Parameters dialog box for use by the Simulink Coder code generation software. See “Set External Mode Code Generation Parameters” on page 3-19.
- 2 Click the Build  icon in the toolbar.

The build process does the following:

- The Simulink Coder code generation software creates the C code source files `sldrtext_model.c` and `sldrtext_model.h`.
- The build process creates the makefile `sldrtext_model.mk` from the template makefile `sldrt.tmf`.
- The build process creates the real-time application `sldrtext_model.rwd` using the makefile `sldrtext_model.mk` created above. The file `sldrtext_model.rwd` is a binary file that we refer to as your real-time application. You can run the real-time application with the Simulink Desktop Real-Time kernel.

After you create a real-time application, you can exit the MATLAB environment, start the MATLAB environment again, and then connect and run the executable without having to rebuild.

## Set External Mode Scope Parameters

Simulink normal and accelerator modes run the simulation algorithm in Simulink and accesses the external hardware using drivers running in operating system kernel mode. The Simulink block diagram is a graphical user interface to your real-time application.

Simulink external mode connects your Simulink model to your real-time application. This connection allows you to use the Simulink block diagram as a graphical user interface as in normal or accelerator mode.

This procedure uses the Simulink model `sldrtex_model` as an example and assumes you have already loaded that model.

After you have created a real-time application, you can enter scope parameters for signal tracing with Simulink external mode:

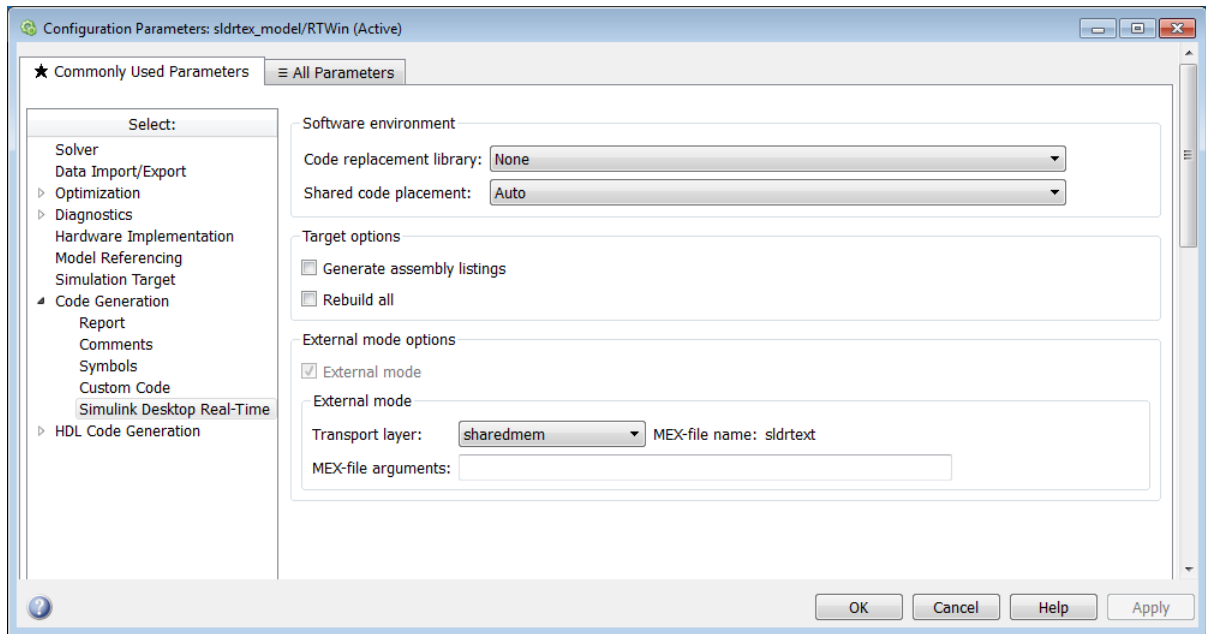
- 1 In the Simulation window, and from the **Simulation** menu, click **Model Configuration Parameters**.
- 2 Select the **Code Generation > Simulink Desktop Real-Time** node.

The **Simulink Desktop Real-Time** pane opens.

- 3 If the **External mode** check box is selected, your changes will affect the real-time application.

The **MEX-file name** label should have an entry of `sldrtex`. The MEX-file `sldrtex.mex*` is supplied with the Simulink Desktop Real-Time software to work with Simulink external mode and support uploading signal data and downloading parameter values.

The **Simulink Desktop Real-Time** pane looks like this:



- 4 Click **OK**.
- 5 In the Simulation window, click **Code > External Mode Control Panel**.

The External Mode Control Panel dialog box opens.

- 6 Click the **Signal & Triggering** button.

The External Signal & Triggering dialog box opens.

- 7 Select the **Select all** check box. From the **Source** list, choose **manual**. From the **Mode** list, select **normal**.

The X under **Signal selection** indicates that a signal is tagged for data collection, and T indicates that the signal is tagged as a trigger signal.

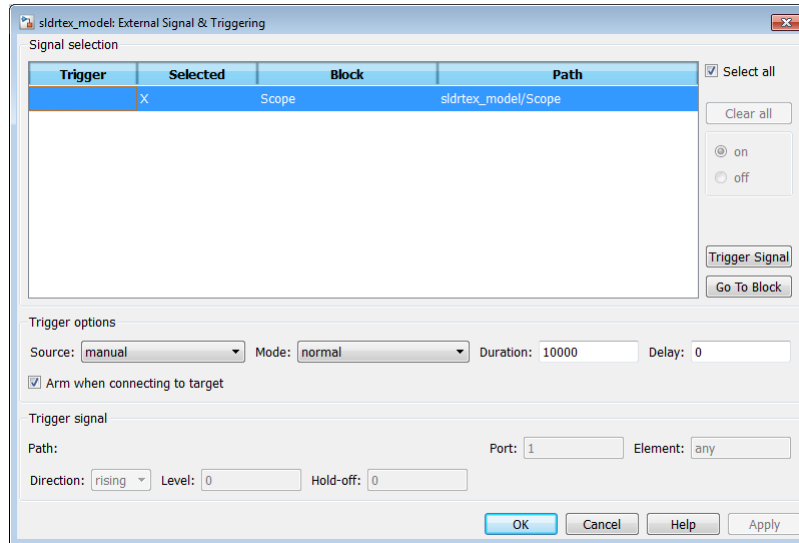
- 8 In the **Duration** field, enter the number of sample points in a data buffer. For example, to specify a sample rate of 1000 samples/second and a stop time of 10 seconds, enter

10000

- 9 Select the **Arm when connecting to target** check box.

If you do not select this check box, data is not displayed in the Scope window.

The External Signal & Triggering dialog box looks like this:



**10** Do one of the following:

- Click **Apply** to apply the changes to your model and leave the dialog box open.
- Click **Close** to apply the changes to your model and close the dialog box.

You must click the **Apply** or **Close** button on the External Signal & Triggering dialog box for the changes you made to take effect.



## Run Application from Simulink



After you build the real-time application, you can run your model in real time.

This procedure uses the Simulink model `sldrtext_model` as an example and assumes you have already created a real-time application from that model.

You cannot run a real-time application in **Rapid Accelerator** mode.

### Normal Mode



In normal mode, the procedure is as follows:

- 1 Select **Simulation > Mode > Normal**.
- 2 Select **Simulation > Run** or click the Run button  on the toolbar.
- 3 To stop the simulation before the end, click the Stop button  on the toolbar.

The real-time application stops running, and the Scope window stops displaying the output signals.

### Accelerator Mode

In accelerator mode, the procedure is as follows:

- 1 Select **Simulation > Mode > Accelerator**.
- 2 Select **Simulation > Run** or click the Run button  on the toolbar.
- 3 To stop the simulation before the end, click the Stop button  on the toolbar.

The real-time application stops running, and the Scope window stops displaying the output signals.


### External Mode

In external mode, you run your real-time application to observe the behavior of your model in real time with the generated code. The process of connecting consists of

- Establishing a connection between your Simulink model and the kernel to allow exchange of commands, parameters, and logged data.

- Running the application in real time.

The procedure is as follows:

- 1 Select **Simulation > Mode > External**.
- 2 Click **Simulation > Run** or click the Run button  on the toolbar.

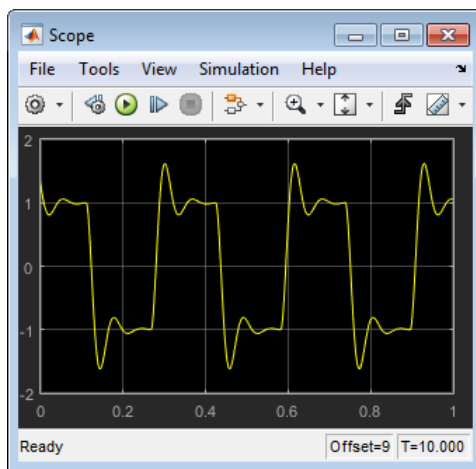
The Simulink software runs the execution and plots the signal data in the Scope window.


In this example, the Scope window displays 1000 samples in 1 second, increases the time offset, and then displays the samples for the next 1 second.

---

**Note:** Transfer of data is less critical than calculating the signal outputs at the selected sample interval. Therefore, data transfer runs at a lower priority in the remaining CPU time after real-time application computations are performed while waiting for another interrupt to trigger the next real-time application update. The result may be a loss of data points displayed in the Scope window.

---



- 3 To stop the simulation before the end, click the Stop button  on the toolbar.

The real-time application stops running, and the Scope window stops displaying the output signals.

## Run Application from MATLAB Command Line

You can use the MATLAB command-line interface as an alternative to using the Simulink GUI. Enter commands directly in the MATLAB Command Window or save them in a script file.

After you build the real-time application, you can run your model in real time.

This procedure uses the Simulink model `sldrtex_model` as an example and assumes you have already created a real-time application from that model.

### Normal Mode

- 1 In the MATLAB Command Window, type

```
set_param(gcs, 'SimulationMode', 'normal')
```

The Simulink software changes to normal mode.

- 2 Type

```
set_param(gcs, 'SimulationCommand', 'start')
```

The Simulink software starts running the real-time application.

- 3 Type

```
set_param(gcs, 'SimulationCommand', 'stop')
```

The Simulink software stops the real-time application.

### Accelerator Mode

- 1 In the MATLAB Command Window, type

```
set_param(gcs, 'SimulationMode', 'accelerator')
```

The Simulink software changes to accelerator mode.

- 2 Type

```
set_param(gcs, 'SimulationCommand', 'start')
```

The Simulink software starts running the real-time application.

- 3 Type

```
set_param(gcs, 'SimulationCommand', 'stop')
```

The Simulink software stops the real-time application.

## External Mode

- 1 In the MATLAB Command Window, type

```
set_param(gcs, 'SimulationMode', 'external')
```

The Simulink software changes to external mode.

- 2 Type

```
set_param(gcs, 'SimulationCommand', 'connect')
```

The MATLAB environment loads the real-time application, connects it to the Simulink block diagram, and displays the message

```
Model sldrtex_model loaded
```

- 3 Type

```
set_param(gcs, 'SimulationCommand', 'start')
```

The Simulink software starts running the real-time application.

- 4 Type


```
set_param(gcs, 'SimulationCommand', 'stop')
```

The Simulink software stops the real-time application.


## Inspect Simulink Desktop Real-Time Signals with Simulation Data Inspector

Use Simulink external mode to establish a communication channel between your Simulink block diagram and your real-time application. You can use Simulation Data Inspector (SDI) to stream signal data from the real-time application from models referenced at arbitrary levels within a model hierarchy. Control which signals to display by selecting them in the model.


This procedure uses the model `sldrtex_model` as an example. Make sure that you have started the Simulink Desktop Real-Time kernel.

- 1 Open `sldrtex_model`.
- 2 On the toolbar, set the simulation stop time to, for example, 30 seconds.
- 3 Click **Simulation > Mode > External**.
- 4 In the model, select the signals `Signal Generator` and `Transfer Fcn`.
- 5 On the toolbar, click the arrow next to the **Simulation Data Inspector** button , and then select **Stream Selected Signals to Data Inspector**.

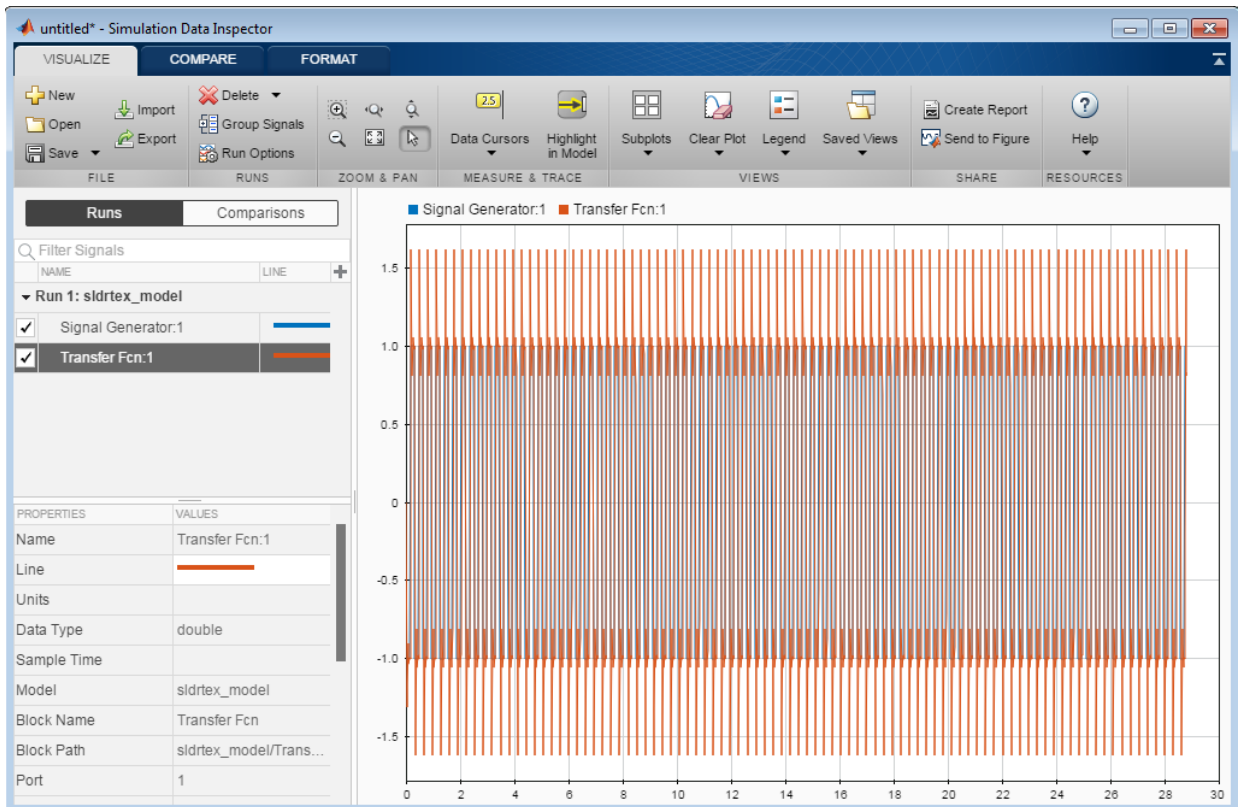
A faint **Simulation Data Inspector** icon  appears next to each signal.



- 6 To start the simulation, click the **Run** button .

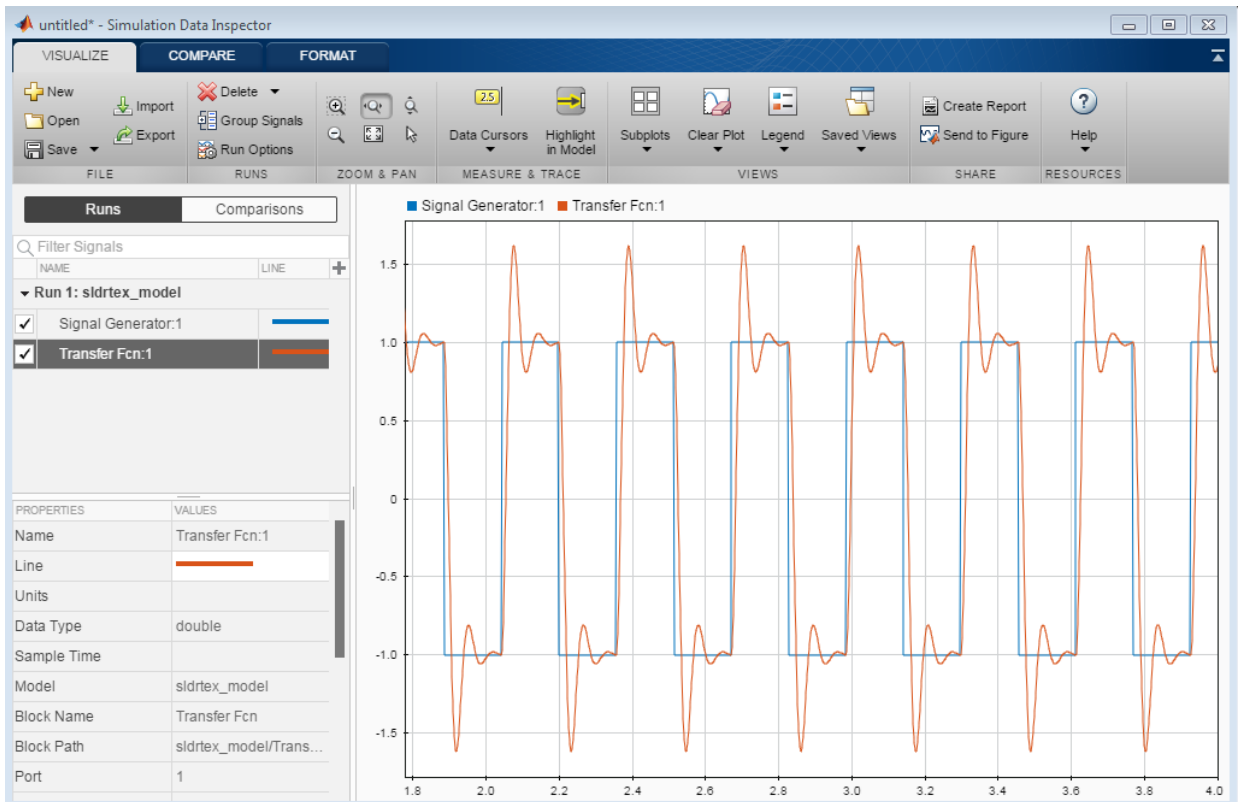
The **Simulation Data Inspector** button glows, indicating that Simulation Data Inspector has data available for viewing.

- 7 Click the glowing **Simulation Data Inspector** button .
- 8 In Simulation Data Inspector, select the signals `Signal Generator:1` and `Transfer Fcn:1`.

Simulation Data Inspector displays plotted data.



- 9 To stop the simulation, click the **Stop** button .
- 10 After the simulation, you can use the toolbar buttons to explore the data. For example, to view the simulation between seconds 2 and 4, in Simulation Data Inspector, click the **Zoom in Time** button . Drag the cursor over the range from 2 to 4.



**11** To save the Simulation Data Inspector session as a `.mat` file, click **Save**.

## Related Examples

- “Water Tank Model with Dashboard”

## More About

- “Inspect Signal Data with Simulation Data Inspector”

## Signal Logging to the Workspace

Logging signals to the workspace saves data to a variable in your MATLAB base workspace. This process allows you to use MATLAB functions for data analysis and MATLAB plotting functions for visualization. You can save data to a variable during a simulation or during an execution.

If your model contains **Outport** blocks, you cannot save signal data in Simulink external mode. Simulink supports signal logging with **Outport** blocks in normal or accelerator modes only, when Simulink runs the simulation algorithm.

---

**Tip** In external mode, do not enter or select parameters on the **Data I/O** tab in the Configuration Parameters dialog box. Instead, add a **Scope** block to your Simulink model and use it to log signal data.

---



## Set Scope Parameters for Logging to Workspace

Data is saved to the MATLAB workspace through a Simulink Scope block. Scope block parameters need to be set for data to be saved. After you create a Simulink model and add a Scope block, you can enter the scope parameters for signal logging to the MATLAB workspace.

This procedure uses the Simulink model `sldrtext_model` as an example, and assumes you have already loaded that model.


---

**Note:** If you entered the scope parameters for running a simulation, you may want to look over this procedure, because the Scope parameters dialog box is related to the External Signal and Triggering dialog box.

---

- 1 In the Simulink window, double-click the Scope block.

A Scope window opens.

- 2 Click the Parameters button  on the toolbar.

A Scope Parameters dialog box opens.

- 3 Click the **Logging** tab.

- 4 Do one of the following:

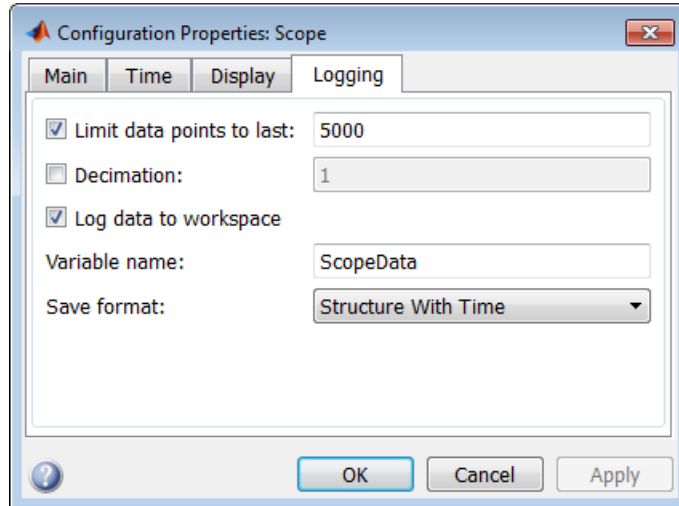
- If you are running a simulation, select the **Limit data points to last** check box, and enter the number of sample points to save.
- If you are running an execution, do not select the **Limit data points to last** check box.

The **Limit data points to last** check box is related to the **Duration** value in the External Signal and Triggering dialog box. The smaller of either value limits the number of sample points saved to the MATLAB workspace. When you are using Simulink Desktop Real-Time software, use the **Duration** value to set the number of sample points you save. To set the **Duration** value, see the next section.

- 5 Select the **Log data to workspace** check box. In the **Variable name** text box, enter the name of a MATLAB variable. The default name is `ScopeData`.

- From the **Save format** list, choose among **Structure with time**, **Structure**, **Array**, and **Dataset**. For example, to save the sample times and signal values at those times, choose **Structure with time**.

Your **Logging** pane looks like this:



- Do one of the following:
  - Click **Apply** to apply the changes to your model and leave the dialog box open.
  - Click **OK** to apply the changes to your model and close the dialog box.

When you modify a value in the Scope parameters dialog box, you must click the **Apply** or **OK** button for the changes to take effect, and you must rebuild your real-time application before connecting and starting it. If you do not rebuild, an error dialog box will open. If you do not click **Apply**, your executable will run, but it will use the old settings.

The reason why you need to rebuild is because the model checksum includes settings from the **Scope** block used for signal logging. If the model checksum does not match the checksum in the generated code, the real-time application cannot run. Rebuild your real-time application after changing Scope parameters.

## Set External Mode Properties for Logging to Workspace

Data is saved to the MATLAB workspace through a Simulink Scope block. Signal and triggering properties need to be set only when you are running a real-time application. If you are running a normal or accelerator mode simulation, you can skip this procedure.

After you create a Simulink model and add a Scope block, you can enter the signal and triggering properties for logging to the MATLAB workspace.

This procedure uses the Simulink model `sldrtext_model` as an example and assumes you have already loaded that model.

- 1 In the Simulation window, click **Code > External Mode Control Panel**.

The External Mode Control Panel dialog box opens.

- 2 Click the **Signal & Triggering** button.

The External Signal & Triggering dialog box opens.

- 3 Click the **Select all** button. From the **Source** list, choose `manual`. From the **Mode** list, choose `normal`.

The X under **Signal selection** designates that a signal has been tagged for data collection, and T designates that the signal has been tagged as a trigger signal.

- 4 In the **Duration** field, enter the number of sample points in a data buffer. For example, if you have a sample rate of 1000 samples/second and a stop time of 10 seconds, you could enter

10000

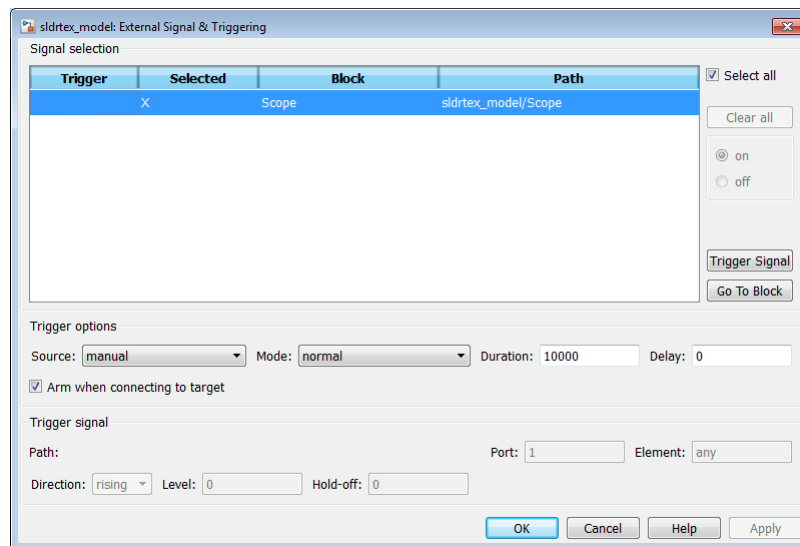
The **Duration** value is related to the **Limit data points to last** value in the Scope parameters dialog box. The smaller of either value limits the number of sample points saved to the MATLAB workspace. You should not select the **Limit data points to last** check box; instead, use the **Duration** value to set the number of sample points saved. To clear the **Limit data points to last** check box, see “Set Scope Parameters for Logging to Workspace” on page 3-37.

The **Duration** value specifies the number of contiguous points of data to be collected in each buffer of data. You should enter a **Duration** value equal to the total number of sample points that you need to collect rather than relying on a series of buffers to be continuous.

If you enter a value less than the total number of sample points, you may lose sample points during the time spent transferring values from the data buffer to the MATLAB workspace. The Simulink Desktop Real-Time software maintains point continuity only within one buffer. Between buffers, because of transfer time, some samples will be omitted.

Also, you should set the time axis for Simulink **Scope** blocks equal to the sample interval (in seconds) times the number of points in each data buffer. This setting will display one buffer of data across the entire Simulink Scope plot.

The External Signal & Triggering dialog box looks like this:



#### 5 Do one of the following:

- Click **Apply** to apply the changes to your model and leave the dialog box open.
- Click **Close** to apply the changes to your model and close the dialog box.

You must click the **Apply** or **Close** button on the External Signal & Triggering dialog box for the changes you made to take effect.

## Plot Signal Data Logged to Workspace

You can use the MATLAB plotting functions for visualizing non-real-time simulated data or real-time application data.

After running your real-time application and logging data to the MATLAB workspace, you can plot the data.

This procedure uses the Simulink model `sldrtex_model` as an example, and assumes you saved your data to the variable `ScopeData`.

- 1 In the MATLAB Command Window, type

```
ScopeData
```

The MATLAB Command Window shows the structure of the variable `ScopeData`. The variable `ScopeData` is a MATLAB structure containing fields for the time vector, signal structure, and a string containing the block name.

```
ScopeData =  
    time: [10000x1 double]  
    signals: [1x1 struct]  
    blockName: 'sldrtex_model/Scope'
```

To list the contents of the structure `signals`, type

```
ScopeData.signals
```

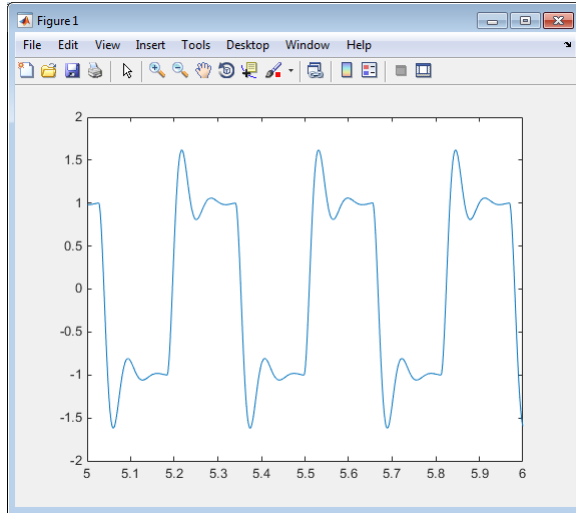
The MATLAB Command Window shows the structure of the variable `ScopeData.signals`. This structure contains one or more vectors of signal data depending on the number of signal inputs to the `Scope` block.

```
ans =  
    values: [10000x1 double]  
    dimensions: 1  
    label: ''  
    title: []  
    plotStyle: 1
```

- 2 To plot the first 1000 points, type

```
plot(ScopeData.time(1:1000),ScopeData.signals.values(1:1000))
```

The MATLAB environment plots the first 1000 samples from 0.0000 to 0.9990 second.



- 3 The variable **ScopeData** is not automatically saved to your hard disk. To save the variable **ScopeData**, type

```
save ScopeData
```

The MATLAB environment saves the scope data to the file **ScopeData.mat**.

## Signal Logging to a File

Logging signals to a file saves data to a variable in your MATLAB workspace and then saves that data to a MAT-file. This allows you to use MATLAB functions for data analysis and MATLAB plotting functions for visualization.

If your model contains **Output** blocks, you cannot save signal data in Simulink external mode. Simulink supports signal logging to a file in normal or accelerator mode only, when Simulink runs the simulation algorithm.

---

**Tip** In external mode, do not enter or select parameters on the **Data I/O** tab in the Configuration Parameters dialog box. Instead, add a **Scope** block to your Simulink model and use it to log signal data for data archiving.

---

## Set Scope Parameters for Logging to File

You save data to a file by first saving the data to the MATLAB workspace through a Simulink **Scope** block. You need to set **Scope** block parameters for data to be saved.


After you create a Simulink model and add a **Scope** block, you can enter the scope parameters for signal logging to a file.

This procedure uses the Simulink model `sldrtex_model` as an example, and assumes you have already loaded that model.

---

**Note:** If you entered the scope parameters for running a simulation, you may want to look over this procedure, because the **Scope** parameters dialog box is related to the **External Signal & Triggering** dialog box and the **Data Archiving** dialog box.

---

- 1 In the Simulink window, double-click the **Scope** block.  
A **Scope** window opens.
- 2 Click the **Parameters** button  on the toolbar.  
A **Scope** parameters dialog box opens.
- 3 Click the **Logging** tab.
- 4 Do one of the following:
  - If you are running a simulation, you can select the **Limit data points to last** check box, and enter the number of sample points to save.
  - If you are running an execution, do not select the **Limit data points to last** check box.

The **Limit data points to last** check box is related to the **Duration** value in the **External Signal & Triggering** dialog box. The smaller of either value limits the number of sample points saved to the MATLAB workspace. When using Simulink Desktop Real-Time software, you should use the **Duration** value to set the number of sample points you save. To set the **Duration** value, see “Set External Mode Properties for Logging to File” on page 3-47.

- 5 Select the **Log data to workspace** check box. In the **Variable name** text box, enter the name of a MATLAB variable. The default name is `ScopeData`.



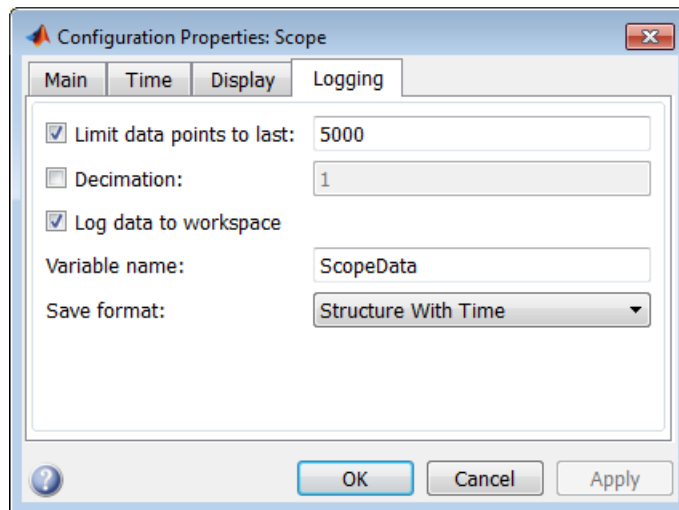
The Scope parameters dialog box is related to the Data Archiving dialog box. In the Scope parameters dialog box, you must select the **Save data to workspace** check box to be able to save data to a file, for two reasons:

- The data is first transferred from the data buffer to the MATLAB workspace before being written to a MAT-file.
- The **Variable name** entered in the Scope parameters dialog box is the same as the one in the MATLAB workspace and the MAT-file.

If you do not select the **Save data to workspace** check box, the MAT-files for data logging will be created, but they will be empty.

- 6 From the **Save format** list, choose among **Structure with time**, **Structure**, **Array**, and **Dataset**. For example, to save the sample times and signal values at those times, choose **Structure with time**.

Your **Logging** pane looks like this:



- 7 Do one of the following:
  - Click **Apply** to apply the changes to your model and leave the dialog box open.
  - Click **OK** to apply the changes to your model and close the dialog box.

You must rebuild your real-time application before connecting and starting the application with changed settings. If you do not rebuild after making changes, an error will occur.

## Set External Mode Properties for Logging to File

Data is saved to a file by first saving the data to the MATLAB workspace through a Simulink **Scope** block. Signal and triggering properties need to be set when running a real-time application.

After you create a Simulink model and add a **Scope** block, you can enter the signal and triggering properties for data logging to a file.

This procedure uses the Simulink model `sldrtext_model` as an example, and assumes you have already loaded that model.

- 1 In the Simulation window, click **Code > External Mode Control Panel**.

The External Mode Control Panel dialog box opens.

- 2 Click the **Signal & Triggering** button.

The External Signal & Triggering dialog box opens.

- 3 Click the **Select all** button. From the **Source** list, choose `manual`. From the **Mode** list, choose `normal`.

The X under **Signal selection** designates that a signal has been tagged for data collection, and T designates that the signal has been tagged as a trigger signal.

- 4 In the **Duration** field, enter the number of sample points in a data buffer. For example, if you have a sample rate of 1000 samples/second and a stop time of 10 seconds, then enter

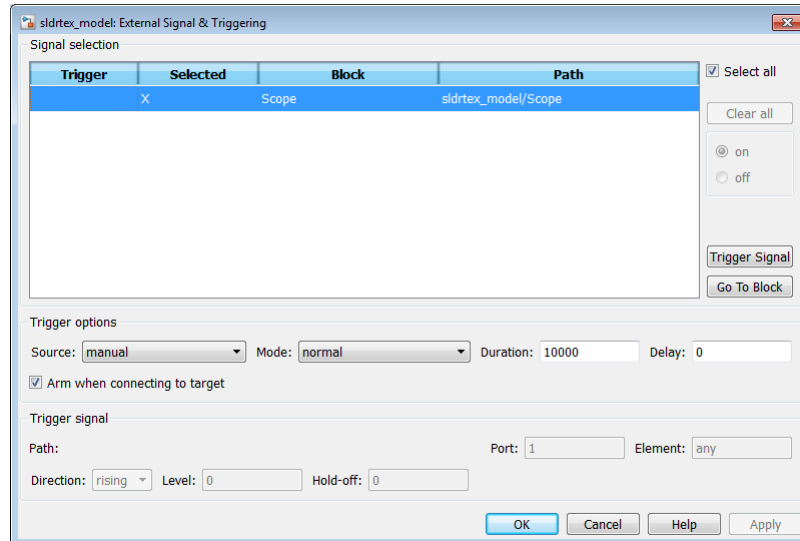
10000

The **Duration** value is related to the **Limit data points to last** value in the Scope parameters dialog box. The smaller of either value limits the number of sample points saved to the MATLAB workspace. You should not select the **Limit data points to last** check box, and use the **Duration** value to set the number of sample points saved.

The **Duration** value specifies the number of contiguous points of data to be collected in each buffer of data. You should enter a **Duration** value equal to the total number of sample points you need to collect for a run. If you enter a value less than the total number of sample points, you may lose sample points due to the time spent transferring values from the data buffer to the MATLAB workspace.

You should also set the time axis for Simulink **Scope** blocks equal to the sample interval (in seconds) times the number of points in each data buffer. This setting will display one buffer of data across the entire Simulink Scope plot.

The External Signal & Triggering dialog box looks like this:



**5** Do one of the following:

- Click **Apply** to apply the changes to your model and leave the dialog box open.
- Click **Close** to apply the changes to your model and close the dialog box.

You must click the **Apply** or **Close** button on the External Signal & Triggering dialog box for the changes you made to take effect, but you do not have to rebuild your real-time application.

## Set External Mode Data Archiving Parameters

The Data Archiving dialog box is related to the Scope parameters dialog box. In the Scope parameters dialog box, you must select the **Save data to workspace** check box to be able to save data to a file, for two reasons:

- The data is first transferred from the scope data buffer to the MATLAB workspace before being written to a MAT-file.
- The **Variable name** entered in the Scope parameters dialog box is the same as the one in the MATLAB workspace and the MAT-file. Enabling the data to be saved enables a variable named with the **Variable name** parameter to be saved to a MAT-file.

If you do not select the **Save data to workspace** check box in the Scope parameters dialog box, the MAT-files for data logging will be created, but they will be empty.

This procedure uses the Simulink model `sldrtex_model` as an example and assumes you have already loaded that model.

After you create a Simulink model, you can enter the Data Archiving Parameters for data logging to a file:

- 1 In the Simulation window, click **Code > External Mode Control Panel**.

The External Mode Control Panel dialog box opens.

- 2 Click the **Data Archiving** button.

The External Data Archiving dialog box opens. This dialog box allows you to specify data archiving options.

- 3 Select the **Enable archiving** check box.
- 4 In the **Directory** text box, enter the path to a folder on your disk. For example, if your MATLAB working folder is named `mwd`, enter

`c:\mwd`

- 5 In the **File** text box, enter the file name prefix for the data files to be saved. For example, enter

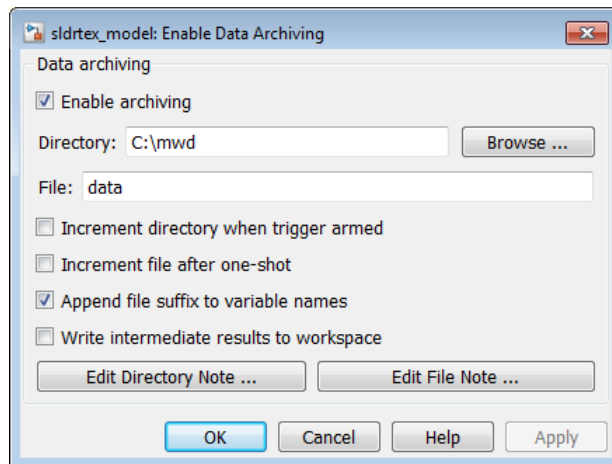
`data`

The MATLAB environment names the files `data_0.mat`, `data_1.mat`, etc. The number of files equals the total sample points. For example, if you set **Duration** to **Total sample points**, then only one file is created.

- 6 Select the **Append file suffix to variable names** check box.

Within each MAT-file, a variable is saved with the same name you entered in the **Variable name** text box (**Logging** pane on the Scope parameters dialog box). By selecting the **Append file suffix to variable names** check box, the same suffix that is added to the MAT-file is added to the variable name. For example, if you entered the variable name `ScopeData`, then within the file `data_0.mat` will be a variable `ScopeData_0`.

Your External Data Archiving dialog box looks like this:



- 7 Click the **Close** button.

The parameters you entered are applied to your model.

There is no **Apply** button with this dialog box. You must click the **Close** button for the changes you make to take effect.

## Plot Signal Data Logged to File

You can use the MATLAB plotting functions for visualization of your non-real-time simulated data or your real-time executed data.

After running your real-time application and logging data to a file, you can plot the data.

This procedure uses the Simulink model `sldrtex_model` as an example, and assumes you saved your data to the variable `ScopeData`.

- 1 In the MATLAB Command Window, type

```
ScopeData
```

The MATLAB Command Window shows the structure of the variable `ScopeData`. The variable `ScopeData` is a MATLAB structure containing the fields time vector, signal structure, and a string containing the block name.

```
ScopeData =  
    time: [10000x1 double]  
    signals: [1x1 struct]  
    blockName: 'sldrtex_model/Scope'
```

- 2 To list the MAT-files saved to your disk, type

```
dir *.mat
```

The MATLAB Command Window shows the MAT-files in your current working folder.

```
ScopeData.mat
```

- 3 To clear the MATLAB workspace and load the scope data, type

```
clear  
load ScopeData  
who
```

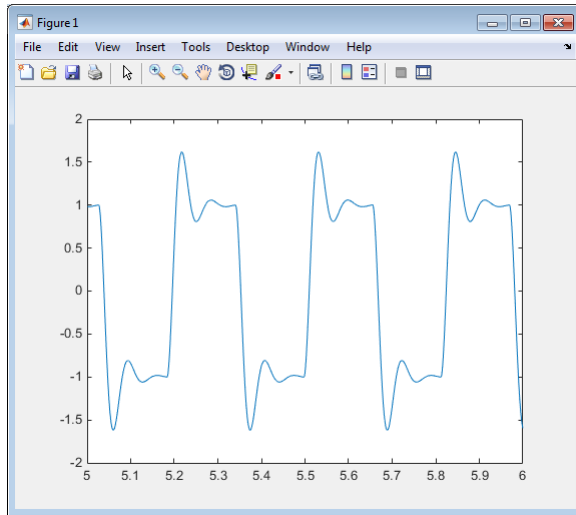
The MATLAB Command Window displays:

```
Your variables are:  
ScopeData
```

- 4 To plot the first 1000 points, type

```
plot(ScopeData.time(1:1000), ScopeData_0.signals.values(1:1000))
```

The MATLAB environment plots the first 1000 samples from 0.0000 to 0.9990 second.





## Tune Block Parameters and MATLAB Variables

In Simulink normal and accelerator modes, only the I/O drivers run in the kernel. Simulink itself manages the blocks and their parameters. Therefore, in these modes, block parameters can be tuned just as in a non-real-time model.

Simulink external mode connects your Simulink model to your real-time application. The block diagram becomes a graphical user interface to the real-time application. You can change parameter values while running the real-time application using block parameters and MATLAB variables.

- **Block parameters** — Change block parameters by changing the values in the dialog boxes associated with the Simulink blocks. Once you change a value, and click **OK**, Simulink downloads the new value to the real-time application. For more on tuning block parameters, see “Change Block Parameters Using Simulink Dialog Box” on page 3-54.
- **MATLAB variables** — Change MATLAB variables by entering the changes through the MATLAB command line, and then press **Ctrl+D** for the changes to be downloaded to your executable. An alternative method to download parameters is to click **Simulation > Update Diagram** in your Simulink window. Simply changing the value of the MATLAB variable at the MATLAB command line does not tell the Simulink software that the value has changed.

Simulink external mode also supports side-effect functions. For example, given an expression in a **Gain** block of  $2*a+b$ , the expression is evaluated and the resulting value is exported to the real-time application during execution.

When a parameter in a Simulink model is changed, the communication module `sldrtext.mex*` transfers the data to the external real-time application and changes the model parameters. Only the parameters that do not result in model structure modification can be changed. If the structure is modified, you must recompile the model. Model structure changes are detected automatically using model checksum and reported to the MATLAB Command Window to avoid conflicts.

### More About

- “Tune and Experiment with Block Parameter Values”

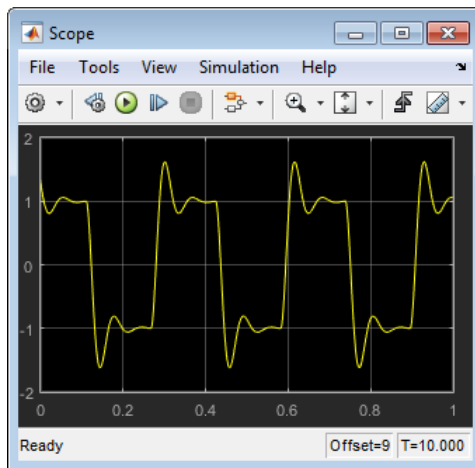
## Change Block Parameters Using Simulink Dialog Box

After you start running your real-time application, you can open Simulink blocks, change parameter values, and observe the changes to the signals. In normal or accelerator mode, Simulink transfers the new value to the real-time driver. In external mode, Simulink transfers the new value to the real-time application.

The following procedure uses the Simulink model `sldrtex_model` as an example. It assumes you have created a real-time application and are executing it.

- 1 From the **Simulation** menu, click **Run**.

The real-time execution starts running and signal data is displayed in the Scope window.

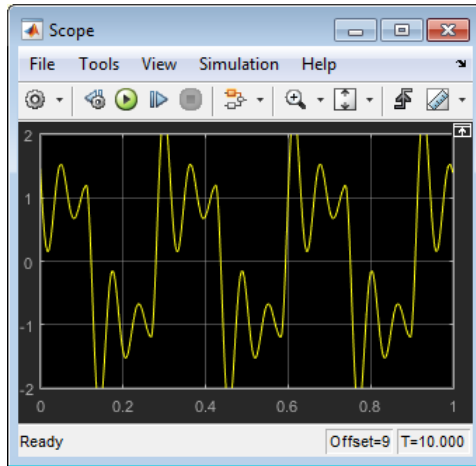


- 2 From the Simulink block diagram, double-click the Transfer Function block.

The Block Parameters: Transfer Fcn dialog box opens.

- 3 In the **Denominator** field, change 70 to 30. Click **OK**.

The effect of changing a block parameter is shown in the Scope window.





# Boards, Blocks, and Drivers

---

Simulink Desktop Real-Time software includes driver blocks for more than 200 I/O boards. These driver blocks connect the physical world to your real-time application.

- “Use I/O Boards” on page 4-2
- “Use I/O Driver Blocks” on page 4-7
- “Use Analog I/O Drivers” on page 4-12
- “Use Vector CAN Drivers” on page 4-16

# Use I/O Boards

Typically I/O boards are preset from the factory for certain base addresses, voltage levels, and unipolar or bipolar modes of operation. Boards often include switches or jumpers that allow you to change many of these initial settings. For information about setting up and installing an I/O board, read the board manufacturer documentation.

For an online list of I/O boards that Simulink Desktop Real-Time software supports, see [www.mathworks.com/hardware-support/simulink-desktop-real-time.html](http://www.mathworks.com/hardware-support/simulink-desktop-real-time.html).

### In this section...

“Install and Configure I/O Boards and Drivers” on page 4-2

“ISA Bus Board” on page 4-5

“PCI Bus Board” on page 4-5

“PC/104 Board” on page 4-6

“Compact PCI Board” on page 4-6

“PCMCIA Board” on page 4-6

## Install and Configure I/O Boards and Drivers

A Simulink Desktop Real-Time model connects to a board by including an I/O driver block. This block provides an interface to the board's device driver and the board-specific settings. The device drivers included with the Simulink Desktop Real-Time software usually provide the same flexibility of settings offered by the board manufacturer. You can enter I/O board settings by using the I/O Block Parameters dialog box; setting jumpers and switches on the board; or both. The three types of board settings are:

- **Software selectable** — Specify the desired settings in the I/O Block Parameters dialog box. The driver writes the settings you specify to the board. Examples include A/D gain inputs and selecting unipolar or bipolar D/A outputs.
- **Hardware selectable and software readable** — Specify the desired settings by configuring jumpers or switches on the board. The driver reads the settings you selected and displays them in the I/O Block Parameters dialog box.
- **Hardware selectable, but not software readable** — Set jumpers or switches on the physical board, and then enter the same settings in the I/O Block Parameters dialog box. These entries must match the hardware jumpers or switches you set on the board. Use this type of setting when the board manufacturer does not provide a

means for the I/O driver to write or read the board settings. Examples include base address, D/A gain, and differential or single-ended A/D inputs.

You can configure a Simulink Desktop Real-Time model to use an I/O board whether or not the board exists in the computer, but you will not be able to run the model until the board is installed with its jumpers and switches set. Details of installation and configuration depend on the data transfer direction and the specific board, but are similar in most cases. Details for various types of boards and drivers appear later in this topic.

The following instructions use the HUMUSOFT® AD512 I/O board as an example, configure the board for analog input, and assume that you have physically configured and installed the board in your computer before you add its driver to your model. Customize the steps to achieve the results you need.

To install and configure an I/O board and its driver,

- 1 Install the board in the computer, setting jumpers or switches according to the board documentation.
- 2 In the model window, choose **View > Library Browser** to display the Simulink Library Browser.
- 3 Drag an **Analog Input I/O driver** block into your model from the Simulink Desktop Real-Time library.
- 4 Double-click the driver block in the model.

The I/O Block Parameters dialog box opens.

- 5 Click **Install new board**. From the list that appears, point to a manufacturer, then select a board type. For example, point to **Humusoft**, then click **AD512**.

The I/O board dialog box opens. The name of this dialog box depends on which I/O board you selected.

- 6 Select one of the following, as required by the board type:
  - For an ISA bus board, enter a hexadecimal base address. This value must match the base address jumpers or switches set on the physical board. For example, to enter a base address of 0x300, in the **Address** box type

300

You can also select the base address by checking boxes **A9** through **A3**.

---

**Note:** Support for ISA bus boards will be removed in a future release. Use PCI, PC/104, Compact PCI, or PCMCIA boards instead.

---

- For a PCI bus board, enter the logical device number in the **Device order** box or check **Auto-detect**.
- 7 Set the other required block parameters using the I/O Block Parameters dialog box.
  - 8 Click **Test**.

The Simulink Desktop Real-Time kernel tries to connect to the selected board, and if it does so without an error, displays a message indicating that it found the board.

- 9 Click **OK** on the message box, and again on the I/O Block Parameters dialog box.

The I/O Block Parameters dialog box closes, and the parameter values are included in your Simulink model.

### Multiple Boards of Identical Type

When multiple boards of identical type exist, you must execute the complete installation sequence for each board, just as you would if the boards were of different types. Thus, two identical PCI boards result in two entries in the list of installed boards. The entries differ only in the logical device number shown in the **Device order** box for each board.

#### Autodetect Multiple Boards

The Autodetect feature cannot be used to locate multiple boards of the same type. You must specify their logical device numbers manually.

### Multiple Driver Blocks for One Board

Once you have used the I/O Block Parameters dialog box to add a board and configure its driver, you can add additional I/O driver blocks that connect to the same board from other locations in the model. To do this, drag the required driver block into the model, open its I/O Block Parameters dialog box, and choose the board from the list of installed boards.

#### Scope of Driver Block Parameters

I/O driver blocks that use a given board share identical parameters. You need to specify these parameters only once, when you first add the board and configure its driver. If you subsequently change a parameter in a driver block connected to a board, the same change occurs in the other driver blocks connected to that board.



## ISA Bus Board

Most ISA bus I/O boards are preset with a base address of 0x300. If you are using multiple I/O boards or other boards (for example, network cards) that already use the address 0x300, you must set your board with another base address.

In the I/O board dialog box, enter the same base address that you set on the physical board. You open the I/O board dialog box from an I/O driver Block Parameters dialog box.

---

**Note:** Support for ISA bus boards will be removed in a future release. Use PCI, PC/104, Compact PCI, or PCMCIA boards instead.

---

## PCI Bus Board

You do not have to set a base address for a PCI board. The plug-and-play feature of the operating system assigns a PCI slot number and logical device number. You can enter the logical device number in the **Device order** box, or you can let the driver determine the device number for you. The **Device order** box is in the I/O board dialog box, which you can open from an I/O driver Block Parameters dialog box.

Before you use a PCI or PCMCIA board, you should install the drivers supplied by the board manufacturer. The Simulink Desktop Real-Time software does not use these manufacturer-supplied drivers. However, they sometimes initiate the plug-and-play recognition of the board. Without these drivers installed, the board might be invisible to your computer and to the Simulink Desktop Real-Time software.

### Write PCI Bus Board Drivers

Simulink Desktop Real-Time applications cannot use DLLs and kernel-mode drivers, which are not suitable for real-time operation. The device drivers supported by the Simulink Desktop Real-Time software are listed at [www.mathworks.com/hardware-support/simulink-desktop-real-time.html](http://www.mathworks.com/hardware-support/simulink-desktop-real-time.html). If no driver is listed for the board that you want to use, you may be able to write a custom device driver.

A user-written custom device driver must program the board directly at the register level. All supported Simulink Desktop Real-Time drivers use this technique. The Simulink Desktop Real-Time software supports I/O mapped board registers for custom device drivers. The Simulink Desktop Real-Time software does not support memory-mapped board registers for custom device drivers.

If you want to use an unsupported board that you believe should be supported, contact MathWorks Technical Support at [www.mathworks.com/contact\\_TS.html](http://www.mathworks.com/contact_TS.html).

### **PC/104 Board**

Most PC/104 bus I/O boards are preset with a base address of **0x300**. If you are using multiple I/O boards or other boards (for example, network cards) that already use the address **0x300**, you must set your board with another base address.

In the I/O board dialog box, enter the same base address that you set on the physical board. You open the I/O board dialog box from an I/O driver Block Parameters dialog box.

### **Compact PCI Board**

Using a compact PCI board (PXI<sup>®</sup>, PXI Express) requires you to use a compact PC (industrial PC). In addition, you need to install the operating system, the MATLAB environment, Simulink software, and Simulink Desktop Real-Time software on the compact PC.

### **PCMCIA Board**

The plug-and-play feature of the operating system assigns a base address automatically. You can enter this address in the I/O board dialog box, or you can let the driver determine the address for you. You open the I/O board dialog box from an I/O driver Block Parameters dialog box.

Before you use a PCI or PCMCIA board, install the drivers supplied by the board manufacturer. Simulink Desktop Real-Time software does not use these manufacturer-supplied drivers. However, they sometimes initiate the plug-and-play recognition of the board. Without these drivers installed, the board might be invisible to your computer and to the Simulink Desktop Real-Time software.

## Use I/O Driver Blocks

Simulink Desktop Real-Time I/O driver blocks allow you to select and connect specific analog channels and digital lines to your Simulink model through I/O driver blocks. These blocks provide an interface to your physical I/O boards and your real-time application. They enable the C code created by Simulink Coder code generation software to map block diagram signals to the corresponding I/O channels. All I/O blocks support all applicable Simulink data types.

You can have multiple I/O blocks associated with each type of I/O board. For example, you can have one **Analog Input** block for channels 1 to 4 and another block for channels 5 to 8. Each I/O block in a model specifies its own block configuration parameters, which apply only to that instance of that block.

The capability of an I/O driver block is available only if the corresponding I/O hardware device supports that capability. For example, data-oriented devices like Serial Port and File support packet and stream I/O blocks. However, data acquisition devices do not support the packet and stream I/O blocks.

The Simulink Desktop Real-Time Library provides blocks that you can use with supported I/O boards. You can also create your own I/O blocks to work with Simulink Desktop Real-Time software. See “Custom I/O Driver Basics” on page A-2 for details.

### In this section...

“View Simulink Desktop Real-Time Library” on page 4-7

“Route Signals from an I/O Block” on page 4-8

“Configure Channel Selection” on page 4-9

## View Simulink Desktop Real-Time Library

I/O driver blocks are available in the Simulink Desktop Real-Time Library. To view this library from the MATLAB Command Window, type:

```
sldrtlib
```

To view the Simulink Desktop Real-Time Library from a model:

- 1 In the model window, choose **View > Library Browser**.

The Simulink Library Browser opens. The left pane shows a hierarchy of libraries and categories, with the Simulink library at the top. The right pane shows the blocks available in the category selected on the left.

- 2 In the left column, double-click **Simulink Desktop Real-Time**.

The Simulink Desktop Real-Time library opens.

You can add an I/O block in the library to your Simulink model by dragging it from the library to the model. After you add the block, connect it to your model as you would any other block, and provide block configuration parameter values.

### Route Signals from an I/O Block

I/O driver blocks output multiple signals as a vector instead of individual channels or lines. To connect the individual channels and lines to parts of your Simulink model, you need to separate the vector with a **Demux** block.

After you add and configure an I/O driver block in your Simulink model, you can separate and connect the output signals from the blocks:

- 1 In the Simulink window, choose **View > Library Browser**.

The Simulink Library Browser opens.

- 2 In the **Simulink** library, click **Signal Routing**. From the list in the right pane, click and drag **Demux** to your Simulink model.
- 3 Double-click the **Demux** block. The Block Parameters: Demux dialog box opens. Enter the number of lines leaving the **Demux** block. For example, if you entered three channels in the **Analog Input** block, enter **3** in the **Number of outputs** box.
- 4 Click **OK**.
- 5 Connect the **Analog Input** block to the **Demux** block input.
- 6 Connect each of the **Demux** block output lines to the input of other blocks.
- 7 In the Simulink window, click **Display > Signals & Ports > Wide Nonscalar Lines**.
- 8 Click **Display > Signals & Ports > Signal Dimensions**.

---

**Note:** In this example, inputs 1 and 2 are not connected, but they could be connected to other Simulink blocks.

---

## Configure Channel Selection

For a better understanding of how to specify device settings when using both analog and digital signals, this section uses the Keithley® Metrabyte™ DAS-1601 I/O board as an example. The following is a specification summary of the DAS-1601 board:

- **Analog input (A/D)** — 16 single-ended or 8 differential analog inputs (12-bit), polarity is switch configured as either unipolar (0 to 10 volts) or bipolar ( $\pm 10$  volts). Gain is software configured to 1, 10, 100, and 500.
- **Digital input** — Four unidirectional digital inputs
- **Analog output (D/A)** — Two analog outputs (12-bit). Gain is switch configured as 0 to 5 volts, 0 to 10 volts,  $\pm 5$  volts, or  $\pm 10$  volts
- **Digital output** — Four unidirectional digital outputs
- **Base address** — Switch configured base address

This section explores different configurations for input signals.

Once an **Analog Input** block has been placed in the model and the I/O board selected and configured, you can set up the **Analog Input** block to handle input signals.

**Single analog input** — The most basic case is for a single analog input signal that will be physically connected to the first analog input channel on the board. In the Block Parameter: Analog Input dialog box, and the **Input channels** box, enter

1 or [1]

The use of brackets is optional for a single input.

**Input vector with differential analog** — Analog channels are numbered starting with channel 1 and continue until you reach a number corresponding to the maximum number of analog signals supported by the I/O board.

In the case of the DAS-1601, when configured as differential inputs, eight analog channels are supported. The analog input lines are numbered 1 through 8. The complete input vector is

[1 2 3 4 5 6 7 8] or [1:8]

If you want to use the first four differential analog channels, enter

[1 2 3 4]

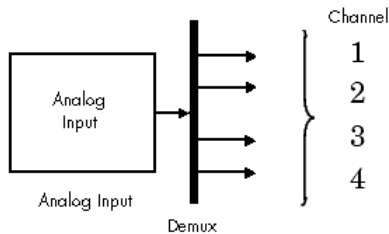
**Input vector with single-ended analog** — Now, assume your DAS-1601 board is configured to be single-ended analog input. In this case, 16 analog input channels are supported. The complete input vector is

[1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16] or [1:16]

To use the first four single-ended analog input channels, enter

[1 2 3 4] or [1:4]

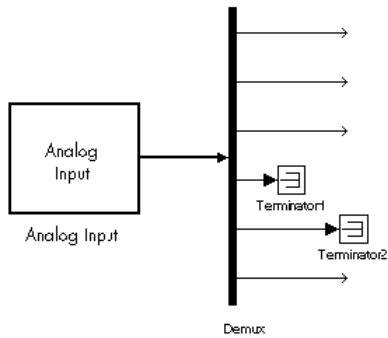
The next figure shows the resulting block diagram.



Do not specify more channels than you actually use in your block diagram. This results in additional overhead for the processor with A/D or D/A conversions. In this case, for example, even though some channels are not actually used in the block diagram, these channels are still converted.

You could attach terminator blocks to channels 4 and 5 inside your block diagram after passing the `Analog Input` block vector in to a `Demux` block. Adding terminator blocks provides you with graphical information in your block diagram to clearly indicate which channels you connected and which are available. The penalty is that even the terminated channels are converted, adding some computational overhead.

The next figure shows the block implementation.



Depending on the board and the number of channels used, I/O conversion time can affect the maximum sample rate that can be achieved on your system. Rather than converting unused channels, specify only the set of channels that are actually required by your model.

## Use Analog I/O Drivers

Control systems have unique requirements for I/O devices used with Simulink Desktop Real-Time applications. For information about writing custom I/O device drivers to work with Simulink Desktop Real-Time applications, see “Custom I/O Driver Basics” on page A-2.

In this section...
“Configure I/O Driver Characteristics” on page 4-12
“Normalize Scaling for Analog Inputs” on page 4-12

### Configure I/O Driver Characteristics

Simulink Desktop Real-Time applications use off-the-shelf I/O boards provided by many hardware vendors. These boards are often used for data acquisition independently of Simulink Desktop Real-Time software. In such environments, board manufacturers usually provide their own I/O device drivers for data acquisition purposes. This use differs significantly from the behavior of drivers provided with Simulink Desktop Real-Time software.

In data acquisition applications, data is often collected in a burst or frame consisting of many points, possibly 1,000 or more. The burst of data becomes available once the final point is available. This approach is not suitable for automatic control applications, because it results in unacceptable latency for most of the data points.

In contrast, drivers used by Simulink Desktop Real-Time applications capture a single data point at each sample interval. Considerable effort is made to minimize the latency between collecting a data point and using the data in the control system algorithm. This is why a board might specify a maximum sample rate (for data acquisition) higher than the rates achievable by Simulink Desktop Real-Time applications. For data acquisition, such boards usually acquire data in bursts and not in the point-by-point fashion required by control systems.

### Normalize Scaling for Analog Inputs

Simulink Desktop Real-Time software allows you to normalize I/O signals internal to the block diagram. Generally, inputs represent real-world values such as angular velocity,



position, temperature, pressure, and so on. This ability to choose normalized signals allows you to

- Apply your own scale factors
- Work with meaningful units without having to convert from voltages

When using an Analog Input block, you select the range of the external voltages that are received by the board, and you choose the block output signal. For example, the voltage range could be set to 0 to +5 V, and the block output signal could be chosen as Normalized unipolar, Normalized bipolar, Volts, or Raw.

If you prefer to work with units of voltage within your Simulink block diagram, you can choose Volts.

If you prefer to apply your own scaling factor, you can choose Normalized unipolar or Normalized bipolar, add a Gain block, and add an offset to convert to a meaningful value in your model.

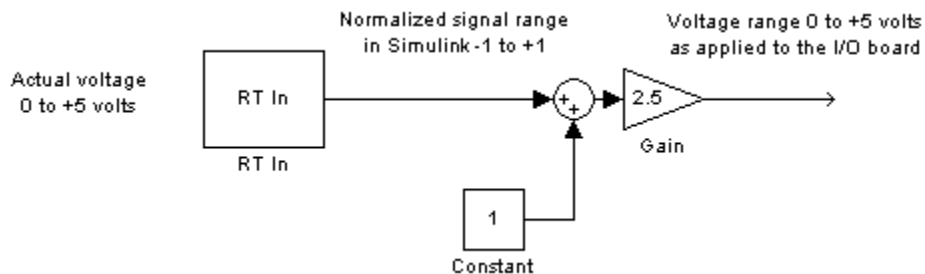
If you prefer unrounded integer values from the analog-to-digital conversion process, you can choose Raw.

### 0 to +5 Volts and Normalized Bipolar

From the Input range list, choose 0 to +5 V, and from the Block output signal list, choose Normalized bipolar. This example converts a normalized bipolar value to volts, but you could also easily convert directly to another parameter in your model.

0 to 5 volts -->  $([-1 \text{ to } 1] \text{ normalized} + 1) * 2.5$

In your block diagram, you can do this as follows.

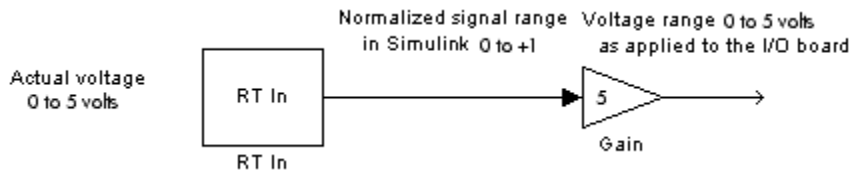


### 0 to +5 Volts and Normalized Unipolar

From the Input range list, choose 0 to +5 V, and from the Block output signal list, choose Normalized unipolar. This example converts a normalized unipolar value to volts, but you could also easily convert directly to another parameter in your model.

0 to 5 volts --> ([0 to 1] normalized \* 5.0

In your block diagram, you can do this as follows.

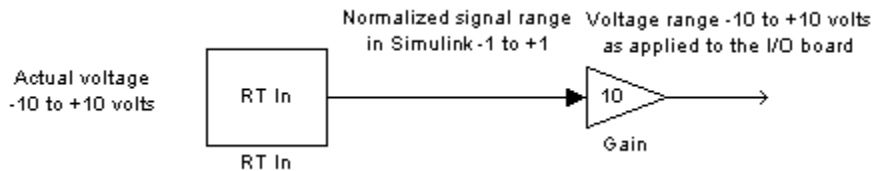


### -10 to +10 Volts and Normalized Bipolar

From the Input range list, choose -10 to +10 V, and from the Block output signal list, choose Normalized bipolar. This example converts a normalized bipolar value to volts, but you could also easily convert directly to another parameter in your model.

-10 to 10 volts --> [-1 to +1] normalized \* 10

In your block diagram, you can do this as follows.

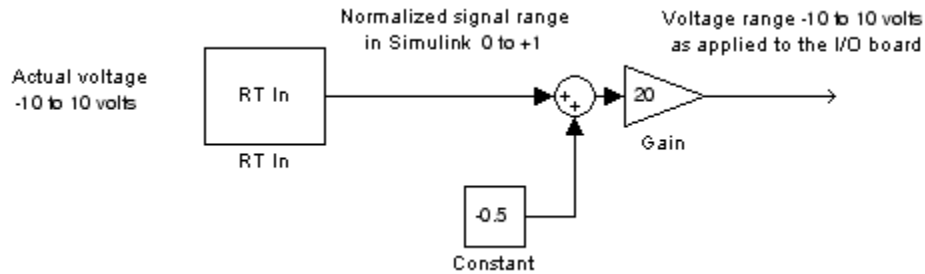


### -10 to +10 Volts and Normalized Unipolar

From the Input range list, choose -10 to +10 V, and from the Block output signal list, choose Normalized unipolar. This example converts a normalized bipolar value to volts, but you could also easily convert directly to another parameter in your model.

-10 to 10 volts --> ([0 to 1] normalized - 0.5) \* 20

In your block diagram, do this as follows.



### Normalize Scaling for Analog Outputs

Analog outputs are treated in an equivalent manner to analog inputs.

If the voltage range on the D/A converter is set to 0 to +5 volts, and the **Block input signal** is chosen as **Normalized bipolar**, then a Simulink signal of amplitude -1 results in an output voltage of 0 volts. Similarly, a Simulink signal of amplitude +1 results in an output voltage of +5 volts.

A voltage range on the D/A converter is set to -10 to +10 volts, and the **Block input signal** is chosen as **Normalized bipolar**, then a Simulink signal of amplitude -1 results in an output voltage of -10 volts. Similarly, a Simulink signal of amplitude +1 results in an output voltage of +10 volts.

This may require that you adjust your signal amplitudes using a **Gain** block, **Constant** block, and **Summer** block depending on the selected voltage range.

### Use Vector CAN Drivers

Before you can use the Vector Informatik CAN devices, you must install the drivers for them.

- 1 Install the Vector CAN devices. See the Vector Informatik GmbH documentation for installation instructions for hardware devices such as CANcaseXL, CANboardXL, and CANcardXL, drivers, support libraries, and so on.
- 2 Install the Vector XL driver library for the Windows XP, Windows Vista™, or Windows 7 operating systems. If you do not have this library, download it from the Vector Informatik GmbH web site:

[www.vector.com/vi\\_downloadcenter\\_en.html](http://www.vector.com/vi_downloadcenter_en.html)

- 3 Install the driver file.
- 4 Copy the `vxlapi.dll` file into the Windows system `root\system32` folder.
- 5 Use the Vector software to assign physical CAN channels to an application. When specifying the name for the Simulink Desktop Real-Time application, use the string `MATLAB`.

# Troubleshooting

---

Solutions have been worked out for some common errors and problems that can occur when you are using Simulink Desktop Real-Time software.

- “Failure to Connect to Target” on page 5-2
- “Scope Output Delayed or Missing” on page 5-3
- “Plots Not Visible in Simulink Scope Block” on page 5-4
- “S-Functions Using Math Functions” on page 5-5
- “Building Older Models” on page 5-6
- “Vendor Software Required” on page 5-7
- “Unsupported Models” on page 5-8

### Failure to Connect to Target

Possible Problem — When trying to run a model, the Simulation Errors dialog box displays:

```
Error occurred while executing External Mode MEX-file
```

```
'sldrtext': Incorrect version
```

Solution — This indicates that the Simulink Desktop Real-Time kernel may be out of date. Type `rtwho` at the MATLAB prompt. If you see a message like this, install the current version:

```
Incorrect kernel version is installed.  
Use 'sldrtkernel -setup' to install the correct version.
```

Possible Problem — When trying to connect to the target, the Simulation Errors dialog box displays:

```
Checksum mismatch. Target code needs to be rebuilt.
```

Solution — This indicates that the model structure has changed since the last time code was generated. You must rebuild the real-time application. If your model fails to build, delete the `.mk` and `.obj` files from the Simulink Coder project folder, and then select **Build** from the **Tools** menu.

### Related Examples

- “Install Real-Time Kernel” on page 2-4

## Scope Output Delayed or Missing

During an external mode run, you may notice either slow updates of Scope blocks or a complete failure to plot data in Scope blocks. This could indicate that the real-time application sample time is near the lower threshold for your hardware.

---

**Note:** The sample time is set in the **Fixed step size (fundamental sample time)** field in the Configuration Parameters **Solver** pane. The **Fixed step size** field appears only when **Type** is set to **Fixed-step**.

---


Plotting data has a lower priority than executing the application, so a small sample time may allow the application to run but leave insufficient resources for plotting. If the sample time is so small that the application itself cannot run, an error message is displayed and real-time execution is terminated.

To check the sample time, select a larger sample time for your application. You should also change the sample time of any I/O drivers to be the same as the new application sample time, or an integer multiple of that time. Then rebuild the model, connect to the target, and restart the real-time application.

As required, iterate changing the application sample time until scope output appears. For example, start with a sample time of 0.01 second, and confirm that your system runs and plots are displayed. Then decrease the sample time until you can both display scopes and meet your accuracy and response time requirements. You must update the I/O driver sample time and rebuild the application after changing the application sample time.

# Plots Not Visible in Simulink Scope Block

Prior to an external mode run, you must specify the following for data to plot in a Simulink Scope block:

- 1 Select **Simulation > Mode > External**.
- 2 Select one or more signals for capture (designated with X) in the External Signal & Triggering dialog box from the External Mode Control Panel.
- 3 Set **Duration \* Fixed Step Size** close to or less than the X range in the Scope block
- 4 Select required mode (**one-shot** or **normal**)
- 5 Configure signal levels to allow triggering
- 6 Set Y range on Simulink Scope block axes to large enough to span the signal amplitude
- 7 Set X range to large enough to provide required time resolution
- 8 Set **Arm when connect to target** in the External Signal & Triggering dialog box or **Arm Trigger** in the External Mode Control Panel.
- 9 Click the Run button  on the toolbar.

If you cannot see signals plotted in your Simulink Scope blocks after carrying out these steps, your system might have too little CPU time available between sample intervals to transfer data back to MATLAB, where the plotting is performed. To test for this condition, run one of the example models or run your model at a significantly slower sample rate.



## S-Functions Using Math Functions

Possible problem — When you create your own S-functions that include math functions, the S-functions compile, but you cannot build the application.

Solution — Add the Simulink Desktop Real-Time header to your S-function. For example, add

```
#include <math.h>
#include "sldrtkernel.h"
```

The header `#include <math.h>` must precede the header `#include "sldrtkernel.h"`.

### Building Older Models

If you are building an older model with Simulink Desktop Real-Time software, you might get a message like the following:

```
"Simulink Coder utilizes device specific information (e.g.,  
microprocessor word sizes) to reproduce a bit true representation  
of the diagram. This information is not specified in this model.  
If you continue, Simulink Coder will use a 32-bit generic  
target setting."
```

This is simply a warning, and you can ignore the message. To eliminate this message, you can use the `sldrtconfigset` command, as follows:

```
sldrtconfigset('<model_name>')
```

## Vendor Software Required

Simulink Desktop Real-Time provides dedicated drivers for I/O modules. Some vendors can require the installation of vendor-specific software, even if you do not use this software. A warning or error message during the build or during execution can indicate this software installation requirement. Or, the I/O module fails to initialize.

If you encounter such instances, refer to the vendor documentation for vendor-specific requirements. If the hardware requires the installation of vendor software, install the vendor software on your computer.

### Unsupported Models

In external mode, Simulink Desktop Real-Time does not support Simscape and Simscape Driveline models.

# Custom I/O Driver Blocks

---

## Custom I/O Driver Basics

In this section...
“Supported C Functions” on page A-2
“Unsupported C Functions” on page A-3
“Incompatibility with Operating System API Calls” on page A-3
“I/O Register Access from S-Functions Limitation” on page A-4

You can write custom I/O device drivers to work with Simulink Desktop Real-Time applications.

---

**Note:** Do not use Analog Input, Analog Output, Digital Input, or Digital Output drivers as starting points for creating custom device drivers.

---

### Supported C Functions

You can use ANSI<sup>®</sup> C functions that do not use the operating system in your custom blocks or I/O drivers. The following includes a partial list of supported functions:

- **Console I/O** — `printf`

The `printf` function sends output to the MATLAB Command Window when it is called from the real-time application.

- **Data conversion** — `abs`, `atof`, `atoi`, `atol`, `itoa`, `labs`, `ltoa`, `strtod`, `strtol`, `strtoul`, `ultoa`
- **Memory allocation** — `calloc`, `free`, `malloc`

Memory allocation is not an operation that can be done in real time. To work with a Simulink Desktop Real-Time application, memory management must occur before real-time simulation begins. Simulation switches into real-time after `mdlStart`, so you can allocate memory in `mdlInitializeSizes` or `mdlStart`. You cannot allocate memory in any function after `mdlStart`, such as `mdlOutputs` or `mdlUpdate`.

- **Memory manipulation** — `_memcpy`, `memcpy`, `memchr`, `memcmp`, `_memicmp`, `memmove`, `memset`

- **String manipulation** — `strcat`, `strchr`, `strcmp`, `strcpy`, `strcspn`, `_strdup`, `_stricmp`, `strlen`, `_strlwr`, `strncat`, `strncmp`, `strncpy`, `_strnset`, `strpbrk`, `strrchr`, `_strrev`, `_strset`, `strspn`, `strstr`, `strtok`, `strupr`
- **Mathematical** — `acos`, `asin`, `atan`, `atan2`, `ceil`, `cos`, `cosh`, `div`, `exp`, `fabs`, `floor`, `fmod`, `frexp`, `ldexp`, `ldiv`, `log`, `log10`, `max`, `min`, `modf`, `pow`, `rand`, `sin`, `sinh`, `sqrt`, `srand`, `tan`, `tanh`, `uldiv`
- **Character class tests and conversion** — `isalnum`, `isalpha`, `_isascii`, `isctrl`, `isdigit`, `isgraph`, `islower`, `isprint`, `ispunct`, `isspace`, `isupper`, `isxdigit`, `isxupper`, `isxlower`, `_toascii`, `tolower`, `toupper`
- **Searching and sorting** — `bsearch`, `qsort`
- **Dummy functions** — `exit`, `fprintf`

## Unsupported C Functions

If you create your own custom I/O driver blocks, you should use only C functions supported by Simulink Desktop Real-Time software. Functions that use the operating system are not supported by Simulink Desktop Real-Time. This includes functions from vendor-supplied driver libraries for the operating system, which are also not supported.

The following list includes many, but not all, of the unsupported functions:

- **File I/O** — `fopen`, `freopen`, `fclose`, `fread`, `fwrite`, `fputs`, `fputc`, `fgets`, `fgetc`, `gets`, `getc`, `getchar`, `puts`, `putc`, `putchar`, `fflush`, `setbuf`, `setvbuf`
- **Console I/O** — `fprintf`, `sprintf`, `vfprintf`, `vprintf`, `vsprintf`, `fscanf`, `scanf`, `sscanf`
- **Process management** — `spawn`, `exit`, `abort`, `atexit`
- **Signals and exceptions** — `signal`, `longjmp`, `raise`
- **Time functions** — `clock`, `time`, `difftime`, `asctime`, `ctime`, `difftime`, `gmtime`, `localtime`, `mktime`, `strftime`
- **Operating system API functions** — *No operating system API functions, such as Win64 functions, are supported.*

## Incompatibility with Operating System API Calls

The Simulink Desktop Real-Time kernel intercepts the interrupt from the system clock. It then reprograms the system clock to operate at a higher frequency for running your real-time application. At the original clock frequency, it sends an interrupt to the

operating system to allow standard applications or any software that uses the operating system API to run.

As a result, **software that uses the operating system API, such as Win64 functions, cannot be executed as a component of your real-time application.** Software you use to write I/O drivers must not make calls to the operating system API.

## **I/O Register Access from S-Functions Limitation**

Operating system drivers can access I/O registers only from the real-time kernel and not from the Simulink software. To prevent drivers from attempting to access I/O registers from Simulink S-functions, enter code fragments like the following:

```
#ifndef MATLAB_MEX_FILE
/* we are in real-time kernel, do board I/O */
#else
/* we are in Simulink, don't do board I/O */
#endif
```