Simulink® Real-Time™

Getting Started Guide

MATLAB®&SIMULINK®

MathWorks®

How to Contact MathWorks

	Latest news:	www.mathworks.com
	Sales and services:	www.mathworks.com/sales_and_services
	User community:	www.mathworks.com/matlabcentral
	Technical support:	www.mathworks.com/support/contact_us
	Phone:	508-647-7000

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

Revision History

November 2000	First printing	New for Version 1 (Release 12)
June 2001	Online only	Revised for Version 1.2 (Release 12.1)
September 2001	Online only	Revised for Version 1.3 (Release 12.1+)
July 2002	Second printing	Revised for Version 2 (Release 13)
September 2003	Online only	Revised for Version 2.0.1 (Release 13SP1)
June 2004	Third printing	Revised for Version 2.5 (Release 14)
August 2004	Online only	Revised for Version 2.6 (Release 14+)
October 2004	Fourth printing	Revised for Version 2.6.1 (Release 14SP1)
November 2004	Online only	Revised for Version 2.7 (Release 14SP1+)
March 2005	Online only	Revised for Version 2.7.2 (Release 14SP2)
September 2005	Online only	Revised for Version 2.8 (Release 14SP3)
March 2006	Online only	Revised for Version 2.9 (Release 2006a)
May 2006	Fifth printing	Revised for Version 3.0 (Release 2006a+)
September 2006	Online only	Revised for Version 3.1 (Release 2006b)
March 2007	Online only	Revised for Version 3.2 (Release 2007a)
September 2007	Online only	Revised for Version 3.3 (Release 2007b)
March 2008	Online only	Revised for Version 3.4 (Release 2008a)
October 2008	Sixth printing	Revised for Version 4.0 (Release 2008b)
March 2009	Online only	Revised for Version 4.1 (Release 2009a)
September 2009	Online only	Revised for Version 4.2 (Release 2009b)
March 2010	Online only	Revised for Version 4.3 (Release 2010a)
September 2010	Seventh printing	Revised for Version 4.4 (Release 2010b)
April 2011	Online only	Revised for Version 5.0 (Release 2011a)
September 2011	Online only	Revised for Version 5.1 (Release 2011b)
March 2012	Online only	Revised for Version 5.2 (Release 2012a)
September 2012	Online only	Revised for Version 5.3 (Release 2012b)
March 2013	Online only	Revised for Version 5.4 (Release 2013a)
September 2013	Online only	Revised for Version 5.5 (Release 2013b)
March 2014	Online only	Revised for Version 6.0 (Release 2014a)
October 2014	Online only	Revised for Version 6.1 (Release 2014b)
March 2015	Online only	Revised for Version 6.2 (Release 2015a)
September 2015	Online only	Revised for Version 6.3 (Release 2015b)
March 2016	Online only	Revised for Version 6.4 (Release 2016a)
September 2016	Online only	Revised for Version 6.5 (Release 2016b)

Contents

Basic Workflows

3

Tutorial and Examples

4

Glossary

Introduction

Simulink Real-Time Product Description

Build, run, and test real-time applications

Simulink Real-Time lets you create real-time applications from Simulink models and run them on dedicated target computer hardware connected to your physical system. It supports real-time simulation and testing, including rapid control prototyping, DSP and vision system prototyping, and hardware-in-the-loop (HIL) simulation.

With Simulink Real-Time you can extend your Simulink models with driver blocks, automatically generate real-time applications, define instrumentation, and perform interactive or automated runs on a dedicated target computer equipped with a real-time kernel, multicore CPU, I/O and protocol interfaces, and FPGAs.

Simulink Real-Time and Speedgoat target computer hardware are expressly designed to work together to create real-time systems for desktop, lab, and field environments. Simulink Real-Time can also be used with custom target computer and I/O hardware.

Key Features

- Automatic generation of real-time applications from Simulink models targeting dedicated CPUs, I/O and protocol hardware, and FPGAs (with HDL Coder™)
- Multitasking and multicore real-time kernel with microsecond granularity and concurrent execution support
- Speedgoat target computer hardware integration for turnkey desktop, lab, and field use
- Driver blocks for I/O, including analog, digital, pulse train, encoders, transformers, passive components, serial, audio, shared memory, and reconfigurable FPGA
- Driver blocks for protocols and data buses, including Raw Ethernet, real-time UDP, CAN, EtherCAT®, Ethernet/IP, Lin, SAE J1939, FlexRay™, Camera Link®, USB video, ARINC 429, and MIL-STD-1553
- Simulink Real-Time Explorer with gigabit Ethernet connection to multiple target computers for management, execution, and instrumentation of real-time applications
- Standalone operation of real-time applications with high-resolution signal display
- MATLAB® functions for test scripting, and APIs for developing standalone client applications and user interfaces (Visual Basic®, C/C++, Java®, and .NET)

Alternative Configuration and Control Methods

The Simulink Real-Time environment has a modifiable interface to the target computer. You can use this interface from MATLAB or Simulink, and you can use other development environments to create custom standalone client applications independent of MATLAB. Because of this open environment, there are several ways to interact with your real-time application from the development and target computers.

The following table compares the interfaces supported by the Simulink Real-Time product.

Interface	Environment Properties	Control	Signal Acquisition	Parameter Tuning	Runnable Outside MATLAB
"Simulink Real-Time Explorer" on page 1-5	X	X	X	X	X
"MATLAB Command-Line Interface" on page 1-8	X	X	X	X	
"Simulink External Mode Interface" on page 1-10		X	X	X	
"Simulink with Simulink Real-Time Blocks" on page 1-11			X		
"Target Computer Command-Line Interface" on page 1-12		X	X	X	X

Interface	Environment Properties	Control	Signal Acquisition	Parameter Tuning	Runnable Outside MATLAB
"Custom UI with Simulink Real-Time API for Microsoft .NET Framework" on page 1-13		X	X	X	X
"Custom UI with Simulink Real-Time C API" on page 1-14		X	X	X	X

Simulink Real-Time Explorer

The Simulink Real-Time software offers a graphical user interface (GUI) for configuring the development and target computers and interacting with a real-time application. To open Simulink Real-Time Explorer:

- In the Simulink Editor, click **Tools** > **Simulink Real-Time**.
- In the Command Window, type `slrtexplr`

Simulink Real-Time Explorer allows you to do the following.

- Configure target computer environment settings.
- Load, unload, and run real-time applications.
- Create, edit, save, load, and run instrument panels.
- Monitor signals.
- Create and configure host, target, and file scopes.
- Display and tune parameters.
- Create, save, and load signal and parameter groups.
- Export Simulink Real-Time Explorer configuration as a separate executable.

You can run Simulink Real-Time Explorer in two modes:

- Within MATLAB running on a development computer.
- As a standalone executable running on a computer compatible with Windows®.

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

- You cannot change the communication parameters that the interface uses to communicate with the target computer. Before you export the Simulink Real-Time Explorer configuration, configure and test the communication parameters.

  To access more than one target computer, in the **Targets** window, configure a separate **Session** record for each target computer.

- For each instrument, the exporting software records the real-time application and target computer environment with which it is associated. To interact with multiple

target computers, create separate instrument panels for each separate real-time application and target computer combination.

- If you rename a target computer, update the **TargetName** parameter for each associated instrument to maintain the connection to the real-time application.

- You cannot load or unload a real-time application from the standalone executable. Before you start the executable, start the real-time application on the target computer.

- You can access only instrument panels and windows that you loaded before you exported the configuration.

- You cannot access the real-time application model hierarchy from the standalone executable.

- You can access only signals in signal groups that you loaded before you exported the configuration.

- You cannot move a signal from one signal group to another group, or create or load a new signal group.

- You can access only parameters in parameter groups that you loaded before you exported the configuration.

- You cannot move a parameter from one parameter group to another group, or create or load a new parameter group.

- You cannot save session layouts. If you close a window, you can restore the original layout using **File** > **Restore Original View**.

More About

- "Monitor Signals with Simulink Real-Time Explorer"
- "Create Target Scopes with Simulink Real-Time Explorer"
- "Create Host Scopes with Simulink Real-Time Explorer"
- "Create File Scopes with Simulink Real-Time Explorer"

- "Tune Parameters with Simulink Real-Time Explorer"
- "Instrumentation for Real-Time Applications"
- "Explorer Configuration Exported to Run Outside MATLAB"
- "Guidelines for Exporting Explorer Configuration"

MATLAB Command-Line Interface

You can interact with the Simulink Real-Time environment through the MATLAB command-line interface. Enter Simulink Real-Time functions in the MATLAB window on the development computer. You can also write your own MATLAB scripts that use Simulink Real-Time functions for batch processing.

The Simulink Real-Time software has more than 90 MATLAB functions for controlling the real-time application from the development computer. These functions define, at the most basic level, what you can do with the Simulink Real-Time environment.

The GUIs provided with the Simulink Real-Time product are for completing the most common tasks. They use the Simulink Real-Time functions but do not extend their functionality. The command-line interface provides an interactive environment that you can extend.

The MATLAB command-line interface includes the following functions:

- Environment — Create a target boot kernel and directly change the environment properties without using a graphical interface.
- Control — Restart the target computer, download a real-time application, start and stop real-time applications, and change start and sample times without regenerating code. Get task execution time information during or after the last run. Add and remove scopes, add/remove signals to scopes, and define triggers for scope display.
- Signal acquisition — Trace signals for viewing while the real-time application is running and monitor signal values without time information. Transfer logged signal data to the MATLAB workspace by uploading from the target computer to the development computer between runs. For standalone target computers, if you write signal data to a file, use `SimulinkRealTime.fileSystem` functions to copy that file to the development computer.
- Parameter tuning — Change parameters while the real-time application is running, and use Simulink Real-Time functions to change parameters in between runs.

For more information, see:

- "Command-Line PCI Bus Ethernet Setup" or "Command-Line USB-to-Ethernet Setup"
- "Command-Line Target Computer Settings"
- "Command-Line Target Boot Methods"

- "Execute Real-Time Application with MATLAB Language" on page 4-53.
- "Monitor Signals with MATLAB Language"
- "Configure Target Scopes with MATLAB Language"
- "Configure File Scopes with MATLAB Language"
- "Log Signal Data with Outport Block and MATLAB Language".
- "Tune Parameters with MATLAB Language".

Simulink External Mode Interface

Use Simulink in external mode to connect your Simulink block diagram to your real-time application. The block diagram becomes a graphical user interface to the real-time application. By changing parameters in the Simulink blocks, you also change parameters in the real-time application.

The Simulink external mode interface includes the following functions:

- Control — Control is limited to connecting the Simulink block diagram to the real-time application, and starting and stopping the real-time application.
- Signal acquisition — You can use Simulink external mode to establish a communication channel between your Simulink block diagram and your real-time application. The block diagram becomes a graphical user interface to your real-time application and Simulink scopes can acquire signal data from the real-time application.
- Parameter tuning — Select external mode, and change parameters in the real-time application by changing parameters in the Block Parameters dialog boxes. Once you change a value and click **OK**, the new value is downloaded to the target computer and replaces the existing parameter while the real-time application continues to run.

For more information, see:

- "Execute Real-Time Application with Simulink External Mode" on page 4-35
- "Trace Signals with Simulink External Mode".
- "Tune Parameters with Simulink External Mode".

Simulink with Simulink Real-Time Blocks

An alternative to interactively adding scopes to the target computer is to add Simulink Real-Time `Scope` blocks to your Simulink model. After the download process, these blocks create scopes on the target computer during initialization of the real-time application. You can display data on either the development computer or target computer. You can also save signal data (log real-time data stream) to a file in the target computer file system and transfer that file to another computer. Finally, you can use `To` and `From` blocks to transfer data to and from a Simulink user interface model.

Signal acquisition — Add scopes to the target computer by adding Simulink Real-Time Scope blocks to your Simulink model. In the Block Parameters dialog box, select the scope mode and set the trigger.

For information on acquiring signal data with `Scope` blocks, see:

- "Add Simulink Real-Time Scope Block" on page 4-24
- "Set Target Scope Block Parameters" on page 4-25
- "Signal Tracing Basics"

For information on using Simulink Real-Time `To` and `From` blocks to transfer data to and from a Simulink user interface model, see "Simulink Real-Time Interface Blocks to Simulink Models".

Target Computer Command-Line Interface

You can interact with the Simulink Real-Time environment through the target computer command window. Enter commands in the command line on the target computer. This interface is useful with standalone real-time applications that are not connected to the development computer.

The target computer command-line interface includes the following functions:

- Control — Start and stop the real-time application, and change the stop time and sample time.
- Signal acquisition — Acquiring signal data is limited to viewing signal traces and signal monitoring on the target computer screen.
- Parameter tuning — You can change only scalar parameters in your model.

For more information, see "Control Real-Time Application at Target Computer Command Line".

Custom UI with Simulink Real-Time API for Microsoft .NET Framework

Use the .NET API Simulink Real-Time framework to develop custom applications such as human-machine interface (HMI) software and batch run scripts that use the Simulink Real-Time software. The Simulink Real-Time .NET object model provides objects that you can interact with. The Simulink Real-Time software arranges the Simulink Real-Time .NET objects in a hierarchical order. Each of these objects has methods and properties that allow you to manipulate and interact with it. This document presents this reference using the C# language.

For more information, see "Using the Simulink Real-Time API for Microsoft .NET Framework".

Custom UI with Simulink Real-Time C API

Use the C API to create a user interface to a real-time application using a development environment that can link in a DLL.

Use the user interface application to control the real-time application, tune parameters, and acquire signal data. The custom user interface runs on the development computer and communicates with the real-time application on the target computer using TCP/IP communication. A user interface application can be a console or Windows application using ActiveX® components.

For more information, see "Using the C API".

2

Installation and Configuration

Setup and Configuration

The initial requirements for installing Simulink Real-Time are:

- "Development Computer Requirements" on page 2-3
- "Development Computer Communication Setup" on page 2-5
- "Development Computer Software Installation" on page 2-10

If you are not using a Speedgoat target computer, see "Custom Target Computer Configuration".

The key setup and configuration steps using MATLAB language and Simulink Real-Time Explorer are:

1. "Command-Line C Compiler Configuration"
2. "PCI Bus Ethernet Setup" on page 2-12 or "USB-to-Ethernet Setup" on page 2-16
3. "Target Computer Settings" on page 2-21
4. "Target Boot Methods" on page 2-23

The key checkout step using MATLAB language is "Run Confidence Test on Configuration" on page 2-43.

Development Computer Requirements

To install and run Simulink Real-Time, MATLAB, Simulink, and other required and optional software, the development computer must be a 64-bit PC-compatible system with Microsoft® .NET Framework 4.5 or later installed. See `Installing the .NET Framework` (`msdn.microsoft.com/en-us/library/5a4x27ek(v=vs.110).aspx`).

In the Simulink Real-Time software environment, the development computer is usually your desktop computer. A notebook computer is also a viable development computer.

In this section...
"Peripherals" on page 2-3
"Communication with Target Computer" on page 2-3

Peripherals

To install and run the Simulink Real-Time product, the development computer requires one hard disk drive with 60 MB of free space.

For producing target computer boot media, the development computer requires one or more of:

- CD-RW drive or DVD-RW drive.
- USB drive (Universal Serial Bus drive).

  You can use a USB drive as a removable boot drive and for data transfer to and from the target computer. For data storage, Simulink Real-Time supports 1-LUN (logical unit) USB drives, 2-LUN USB drives, and 4-LUN card readers.
- SD (Compact) flash drive.
- Removable hard drive.
- 3.5-inch floppy disk drive.

Communication with Target Computer

For communicating with the target computer, the development computer requires one available Ethernet interface (PCI or USB) connected to a network. You must configure the development computer Ethernet port to use Internet Protocol Version 4 (TCP/.IPv4) only.

More About

- "Development Computer Communication Setup" on page 2-5
- "PCI Bus Ethernet Setup" on page 2-12
- "USB-to-Ethernet Setup" on page 2-16

External Websites

- www.microsoft.com

Development Computer Communication Setup

To communicate with the target computer using Ethernet, you must perform a one-time configuration of your development computer Ethernet port to use Internet Protocol Version 4 (TCP/.IPv4) only:

1 On your development computer, from the **Control Panel**, navigate to **Network and Sharing Center**.

2 Click the **Local Area Connection** for the Ethernet adapter you are using. It is most likely listed as `Unidentified network`.

3 Click the **Properties** button.

4 In the **Networking** pane, clear all check boxes.

5 Select the **Internet Protocol Version 4 (TCP/IPv4)** check box.

6 Click the **Properties** button.

7 Select the **Use the following IP address** check box.

8 Configure the IP address boxes as follows:

   IP address — `10.10.10.100`

   Subnet mask — `255.255.255.0`

   Default gateway — leave blank

9 Click the **OK** button.

Target Computer Requirements

The target computer must be a 32-bit or 64-bit PC-compatible system.

At a minimum, Simulink Real-Time requires the following target computer components.

Hardware Component	Requirement
CPU	Intel® Core™, or a model that supports the SSE2 instruction set
RAM	512 MB or more of RAM (2 GB for improved performance)
Hard drive	A hard drive

- Simulink Real-Time is designed to work with desktop target computers only. Do not use a laptop as a target computer.
- The target computer must support the Advanced Programmable Interrupt Controller (APIC) interface.

In this section...
"Peripherals" on page 2-6
"Communications" on page 2-7
"I/O Boards" on page 2-7
"Multicore Processors" on page 2-7

Peripherals

To run the Simulink Real-Time kernel, the target computer requires one or more of:

- CD-RW drive or DVD-RW drive
- USB drive (Universal Serial Bus drive)

  You can use a USB drive as a removable boot drive and for data transfer to and from the target computer. For data storage, Simulink Real-Time supports 1-LUN (logical unit) USB drives, 2-LUN USB drives, and 4-LUN card readers.

- SD (Compact) flash drive
- Hard drive configured to boot into DOS

- PXE-compatible Ethernet adapter

To take full advantage of the Simulink Real-Time product features, install a FAT32 formatted hard drive.

Communications

For the target computer to communicate with the development computer, you need one free Ethernet adapter (PCI or USB) connected to a network (see "PCI Bus Ethernet Setup" on page 2-12 or "USB-to-Ethernet Setup" on page 2-16).

If you want to start the target computer from the network, the Ethernet adapter must be compatible with the Preboot eXecution Environment (PXE) specification.

I/O Boards

I/O boards provide a direct interface to the sensors, actuators, or other devices for real-time control or signal processing system models. Simulink Real-Time supports I/O functionality via the blocks in `slrtlib`.

For information on Speedgoat and MathWorks® I/O blocks, communication protocol blocks, and custom blocks, see:

- "Speedgoat I/O Configuration"
- "I/O Modules"
- "Communication Protocols"
- "Custom I/O Drivers".

For information on installing and connecting I/O modules, see the manufacturer documentation.

Multicore Processors

The Simulink Real-Time kernel can run on target computers equipped with a 32-bit or 64-bit x86 compatible CPU. It can manage target computers with up to 32 logical processors (cores), such as the Intel Core 2 Duo, Intel Core 2 Quad, and later processors.

BIOS Settings

For a custom target computer configuration, to allow the Simulink Real-Time kernel to run on the target computer, make the following changes to the BIOS settings:

·   **Disable Plug-and-Play (PnP)** — Disable the PnP operating system feature so that the PCI BIOS can set up the plugged-in PCI cards. The Simulink Real-Time kernel is not a PnP operating system. If this feature is enabled, PCI devices do not work with Simulink Real-Time.

·   **Disable power saving mode** — Disable all power saving modes.

·   **Disable PCI class `0xff` board detection** — Do not detect PCI boards with class code `0xff` in the target computer BIOS. Turn this option `Off` to enable the BIOS to detect and configure PCI boards.

·   **Disable hyper-threading** — If your target computer supports hyper-threading capabilities, disable these capabilities. Enabling hyper-threading can degrade the performance of the target computer.

Depending upon your configuration, you can also make the following target computer BIOS settings:

·   **Configure boot order** — Set the boot order for the target computer BIOS. You can start the target computer with these methods:

    ·   CD/DVD bootable ROM

    ·   Flash drive

    ·   Removable hard drive

    ·   Dedicated network boot

    ·   Bootable hard drive

    Configure your target computer BIOS to use your preferred boot order.

·   **Enable multicore processor support** — If your target computer includes a multicore processor (see "Multicore Processors" on page 2-7), you can configure the Simulink Real-Time software to take advantage of the individual cores. See "Multicore Processor Configuration" and "Multicore Programming with Simulink".

    To take advantage of a multicore processor, disable hyper-threading in the target computer BIOS.

- **Enable USB communication** — If you are using USB communications, enable USB ports for the target computer.

Development Computer Software Installation

You install the Simulink Real-Time software only on the development computer. The development computer downloads the kernel software and real-time application to the target computer at runtime. The Simulink Real-Time software is distributed on a DVD or as a file you download from the internet.

Verify that you have met the following requirements:

- You must have uninstalled prior installations of the Simulink Real-Time product.
- You must have a valid license for the required products. See "License Requirements" on page 2-10.
- You must have Microsoft .NET Framework 4.5 or later installed on the development computer. The installation software will print an error message if the framework is not installed. See `Installing the .NET Framework` (msdn.microsoft.com/en-us/library/5a4x27ek(v=vs.110).aspx).
- You must have a Microsoft C compiler installed on the development computer. See "Command-Line C Compiler Configuration".

For more information about the file structure after installation, including the location of examples, see "Location of Files on the Development Computer" on page 2-10.

License Requirements

Before you install the Simulink Real-Time product, you must have a valid File Installation Key and License File. The File Installation Key identifies the products that you purchased from MathWorks. The License File activates the installation.

If you have not received either of these items, go to the License Center at the MathWorks web site.

Simulink Real-Time includes standalone mode. With standalone mode, you can start the target computer from a device other than a floppy disk or CD/DVD and create standalone real-time applications separate from the development computer.

Location of Files on the Development Computer

Files are located in the:

- MATLAB working folder — Simulink models (`model`), Simulink Real-Time applications (`model.dlm`)

  Select a working folder outside the MATLAB root.
- Simulink Coder™ Build folder — The Simulink Coder C code files (`model.c`, `model.h`) are in a subfolder called `modelname_xpc_rtw`.

The Simulink Real-Time software uses files located under these folders:

- *matlabroot*`\toolbox\slrt\`

  - `blocks` — Drivers and associated blocks
  - `slrt` — Development computer functions related to the Simulink Real-Time software, methods for target objects, and methods for scope objects
  - `slrtexamples` — Simulink models and MATLAB code examples
- *matlabroot*`\toolbox\rtw\targets\xpc\`

  - `target` — Files and functions related to the Simulink Real-Time kernel and build process, including drivers to support I/O blocks
  - `xpc` — Development computer functions related to the Simulink Real-Time software, methods for target objects, and methods for scope objects
  - `xpcdemos` — Simulink models and MATLAB code examples

External Websites

- www.microsoft.com

PCI Bus Ethernet Setup

If your target computer has a PCI bus, use an Ethernet card designed for the PCI bus. The PCI bus has a faster data transfer rate than the other bus types.

In this section...
"PCI Bus Ethernet Protocol Hardware" on page 2-12
"PCI Bus Ethernet Settings" on page 2-13

PCI Bus Ethernet Protocol Hardware
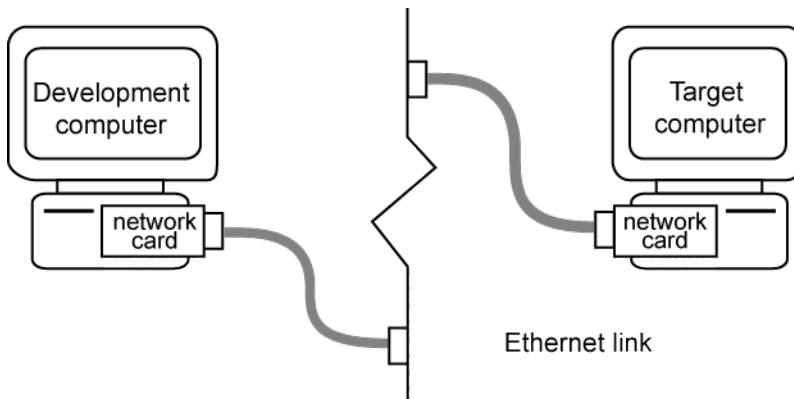
To install PCI bus Ethernet protocol interface hardware:

1 Acquire a supported PCI bus Ethernet card.

    If you want to start the target computer from the network, the Ethernet adapter must be compatible with the Preboot eXecution Environment (PXE) specification.

2 Turn off your target computer.

3 If the target computer already has an unsupported Ethernet card, remove the card.

4 Plug the supported Ethernet card into a free PCI bus slot.

5 Assign a static IP address to the target computer Ethernet card.

    Although the target computer Ethernet card must have a static IP address, the development computer network adapter card can have a Dynamic Host

Configuration Protocol (DHCP) address and can be accessed from the network. You must configure the DHCP server to reserve static IP addresses to prevent these addresses from being assigned to other systems.

6   Connect your target computer Ethernet card to your LAN using an unshielded twisted-pair (UTP) cable.

    You can directly connect your computers using a crossover UTP cable with RJ45 connectors. Both computers must have static IP addresses. If the development computer has a second network adapter card, that card can have a DHCP address.

PCI Bus Ethernet Settings

After you install the PCI bus Ethernet card, before you can build and download a real-time application, you must specify the environment properties for the development and target computers.

1   Ask your system administrator for the following information for your target computer:

    ·   IP address

    ·   Subnet mask address

    ·   Port number (optional)

    ·   Gateway (optional)

2   In the MATLAB Command Window, type `slrtexplr`.

3   In the **Targets** pane, expand the target computer node.

4   On the toolbar, click the **Target Properties** button .

    ·   To add a node representing another target computer, in the **Targets** pane, click the **Add Target** button .

    ·   To remove a node representing a target computer, right-click the node and select **Remove**.

5   In the **Target Properties** pane, click **Host-to-Target communication**.

6   Set **IP address** to the IP address for your target computer (for example, `10.10.10.15`).

7   Set **Subnet mask** to the subnet mask address of your LAN (for example, `255.255.255.0`).

8 Set **Port** (optional) to a value greater than 20000 and less than 65536. This property is set by default to 22222, a value higher than the reserved area (telnet, ftp, and so on).

9 Set **Gateway** (optional) to the gateway required to access the target computer, if a gateway is required. This property is set by default to 255.255.255.255, which means that you do not use a gateway to connect to your target computer. If you connect your computers with a crossover cable, leave this property as 255.255.255.255.

If you communicate with the target computer from within your LAN, you might not need to change this setting. If you communicate from a development computer located in a LAN different from your target computer (especially via the Internet), you must define a gateway and enter its IP address in this box.

10 Select **Bus type** PCI.

11 For **Target driver** Auto (default).

For **Target driver** Auto, the software determines the target computer TCP/IP driver from the card installed on the target computer. If a supported Ethernet card is not installed in your target computer, the software returns an error.

12 If the target computer has multiple Ethernet cards, see "Ethernet Card Selection by Index".

13
Press **Enter**, then click the **Save** button on the toolbar.

The Simulink Real-Time Explorer window looks like this figure.
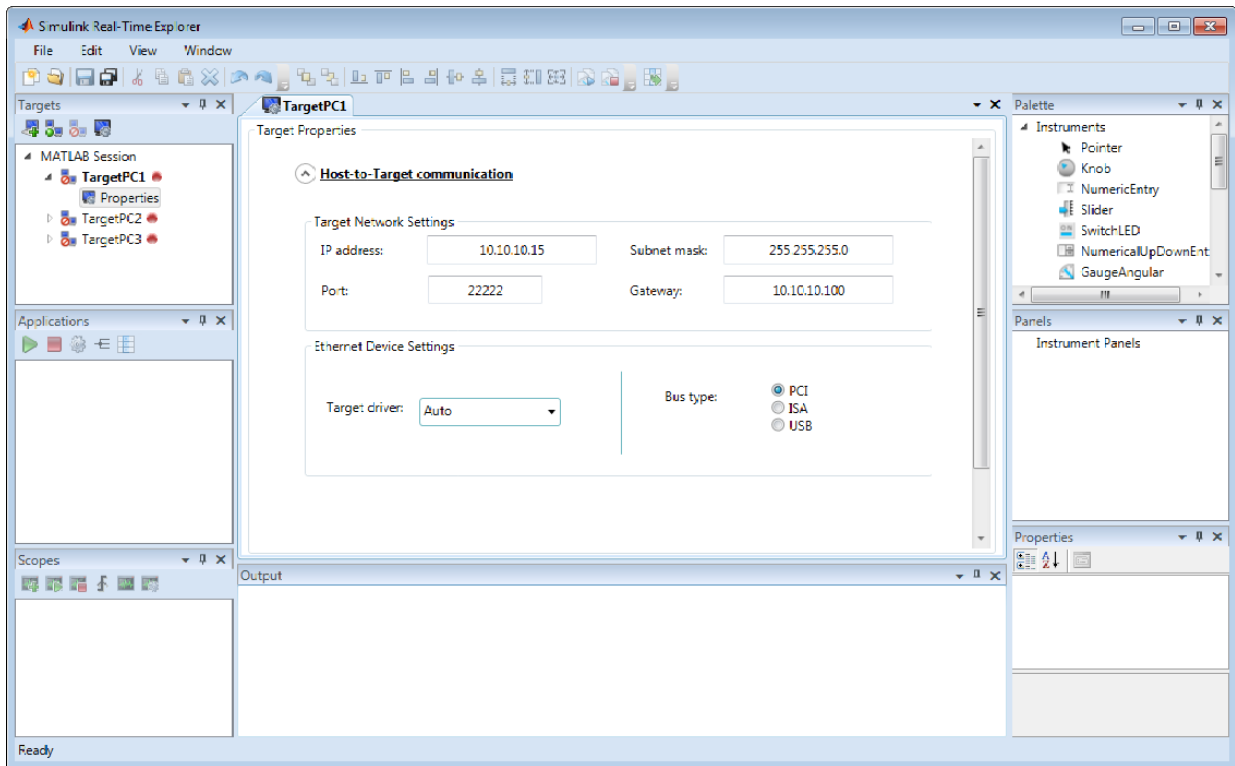
Repeat this procedure as required for each target computer.

USB-to-Ethernet Setup

If the target computer has a USB 2.0 port but does not have a PCI Ethernet card, use a USB-to-Ethernet adapter.

In this section...
"USB-to-Ethernet Protocol Hardware" on page 2-16
"USB-to-Ethernet Settings" on page 2-18

USB-to-Ethernet Protocol Hardware

You can plug the USB-to-Ethernet adapter into the development computer or the target computer. The setup is slightly different for each location.

Development Computer USB Port
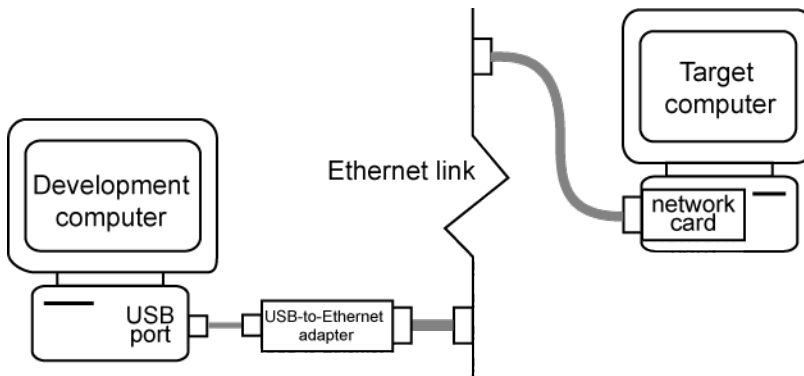
To connect the USB-to-Ethernet protocol interface hardware to the USB port on the development computer:

1 Acquire a supported USB-to-Ethernet adapter.

2 Plug the USB-to-Ethernet adapter into the USB port in the development computer.

Do not connect the development computer USB port to a target computer USB port using a plain USB cable. A USB-to-Ethernet adapter plugged into the development computer USB port behaves like an Ethernet card installed in the development computer.

3 If you want to start the target computer from the network, verify that the Ethernet card in the target computer is compatible with the Preboot eXecution Environment (PXE) specification.

4 Connect the USB-to-Ethernet adapter from your development computer to your LAN using an unshielded twisted-pair (UTP) cable.

5 Connect the target computer Ethernet card to your LAN using another UTP cable.

You can directly connect your computers using a crossover UTP cable with RJ45 connectors. Both computers must have static IP addresses. If the development computer has a second network adapter, that adapter can have a DHCP address.

Target Computer USB Port
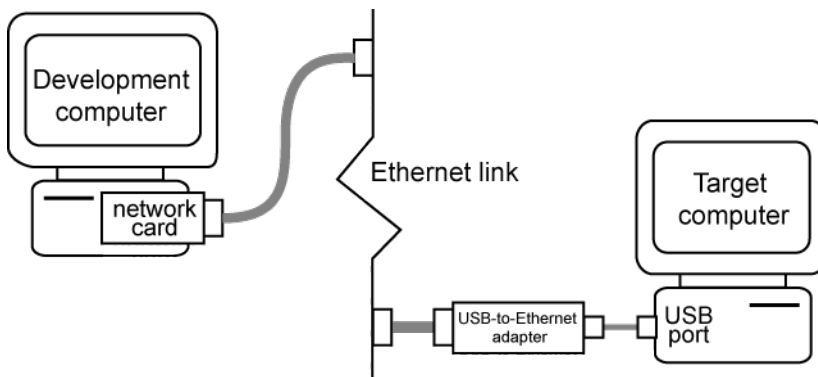
To connect the USB-to-Ethernet protocol interface hardware to the USB port on the target computer:

1 Acquire a supported USB-to-Ethernet adapter.

2 Turn off your target computer.

3 Prepare a boot drive according to the instructions in "Target Boot Methods" on page 2-23.

You cannot use the network boot method with this hardware configuration.

4 Plug the USB-to-Ethernet adapter into the USB port in the target.

Do not connect the development computer USB port to the target computer USB port using a plain USB cable. A USB-to-Ethernet adapter plugged into the target computer USB port behaves like an Ethernet card installed on the target computer.

5 Connect the USB-to-Ethernet adapter to your LAN using an unshielded twisted-pair (UTP) cable.

6 Assign a static IP address to the target computer USB-to-Ethernet adapter.

Although the target computer Ethernet adapter must have a static IP address, the development computer network adapter can have a Dynamic Host Configuration Protocol (DHCP) address and can be accessed from the network. You must configure the DHCP server to reserve static IP addresses to prevent these addresses from being assigned to other systems.

7 Connect your development computer Ethernet card to your LAN using an unshielded twisted-pair (UTP) cable.

You can directly connect your computers using a crossover UTP cable with RJ45 connectors. Both computers must have static IP addresses. If the development computer has a second network adapter, that adapter can have a DHCP address.

USB-to-Ethernet Settings

After you have installed the USB-to-Ethernet adapter, before you can build and download a real-time application, you must specify the environment properties for the development and target computers.

1 Ask your system administrator for the following information for your target computer:

- IP address
- Subnet mask address
- Port number (optional)
- Gateway (optional)

2 In the MATLAB Command Window, type `slrtexplr`.

3 In the **Targets** pane, expand the target computer node.

4 On the toolbar, click the **Target Properties** button .

- To add a node representing another target computer, in the **Targets** pane, click the **Add Target** button .
- To remove a node representing a target computer, right-click the node and select **Remove**.

5   In the **Target Properties** pane, click **Host-to-Target communication**.

6   Set **IP address** to the IP address for your target computer (for example, `10.10.10.15`).

7   Set **Subnet mask** to the subnet mask address of your LAN (for example, `255.255.255.0`).

8   Set **Port** (optional) to a value greater than `20000` and less than `65536`. This property is set by default to `22222`, a value higher than the reserved area (`telnet`, `ftp`, and so on).

9   Set **Gateway** (optional) to the gateway required to access the target computer, if a gateway is required. This property is set by default to `255.255.255.255`, which means that you do not use a gateway to connect to your target computer. If you connect your computers with a crossover cable, leave this property as `255.255.255.255`.

    If you communicate with the target computer from within your LAN, you might not need to change this setting. If you communicate from a development computer located in a LAN different from your target computer (especially via the Internet), you must define a gateway and enter its IP address in this box.

10 Select **Bus type** USB.

11 For **Target driver**, select one of USBAX772, USBAX172, or Auto.

    If **Target driver** is Auto, the software sets the driver to USBAX772, the driver most commonly used.

12 Press **Enter**, then click the **Save** button on the toolbar.

The Simulink Real-Time Explorer window looks like this figure.
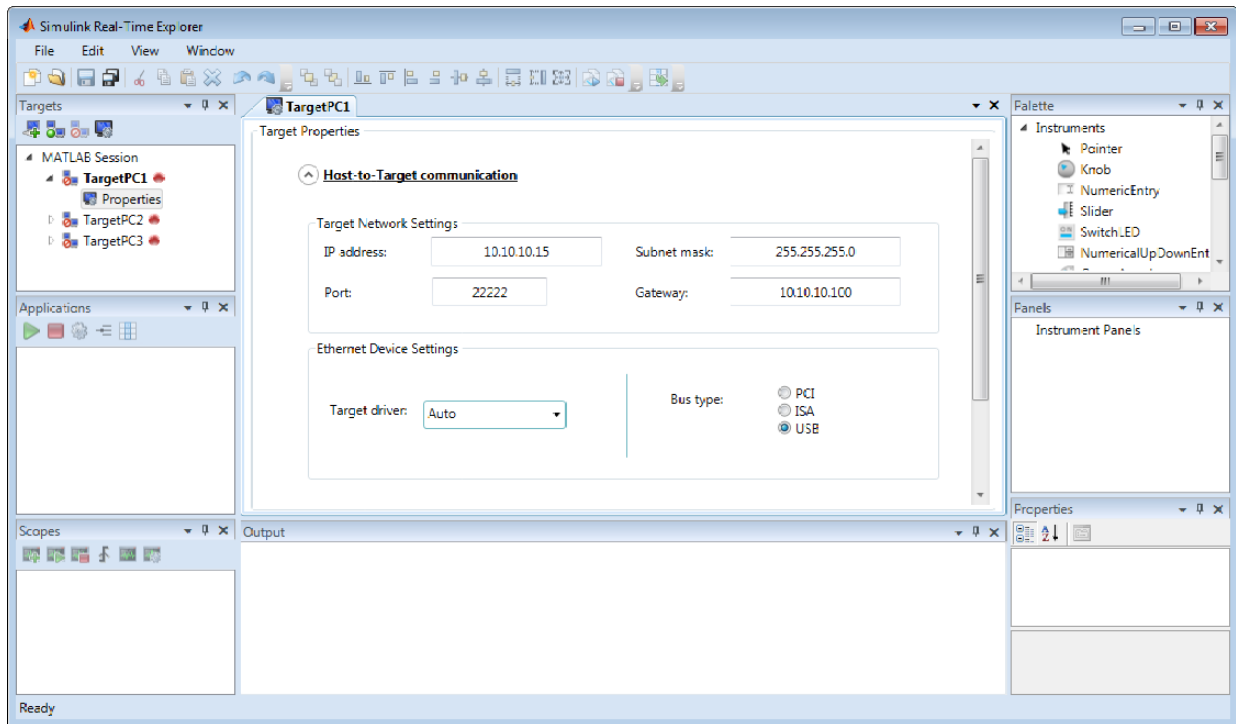
Repeat this procedure as required for each target computer.

Target Computer Settings

To run a Simulink Real-Time model on a target computer, you must configure the target settings to match the capabilities of the target computer.

1 In the MATLAB Command Window, type `slrtexplr`.

2 In the **Targets** pane, expand the target computer node.

3 On the toolbar, click the **Properties** button .

4 In the **Target Properties** workspace, click **Target settings**.

5 Execute the following target computer settings as required:

   • **Graphics mode** — Leave this check box selected if you want to display information, such as a target scope, in graphic format. If you want to display information as text, clear it.

      To use the full features of a target scope, install a keyboard on the target computer.

   • **USB Support** — Leave this check box selected if you want to use a USB port on the target computer; for example, to connect a USB mouse. Otherwise, clear it.

   • **Secondary IDE** — Select this check box only if you want to use the disks connected to a secondary IDE controller.

   • **Multicore CPU** — Leave this check box selected if your target computer has multicore processors that the kernel can use for background tasks. Otherwise, clear it.

   • **RAM size** — Leave **Auto** selected if you want the real-time application to read the target computer BIOS and determine the amount of memory installed. Select **Manual** if the real-time application cannot read the BIOS. You must enter the amount of memory, in megabytes, into the **Size(MB)** text box.

      The **RAM size** setting defines the total amount of installed RAM in the target computer used for the kernel, real-time application executable, data logging, and other functions that use the heap.

      The **Size(MB)** text box is enabled for **RAM size Manual** only.

      Target computer memory for the real-time application executable, the kernel, and other uses is limited to a maximum of 3 GB.

2-21

- **Model size** — Select a value from the list that reserves the required amount of memory on the target computer for the real-time application executable. The remaining memory is used by the kernel and by the heap for data logging.

  Selecting too high a value leaves less memory for data logging. Selecting too low a value does not reserve enough memory for the real-time application and creates an error. You can approximate the size of the real-time application by the size of the DLM file produced by the build process.

  Setting **Model size** is enabled for **Boot mode** `Stand Alone` only.

  You can use only **Model size** values `1 MB` and `4 MB`.

6 Press **Enter**, then click the **Save** button 💾 on the toolbar.

The Simulink Real-Time Explorer window looks like this figure.
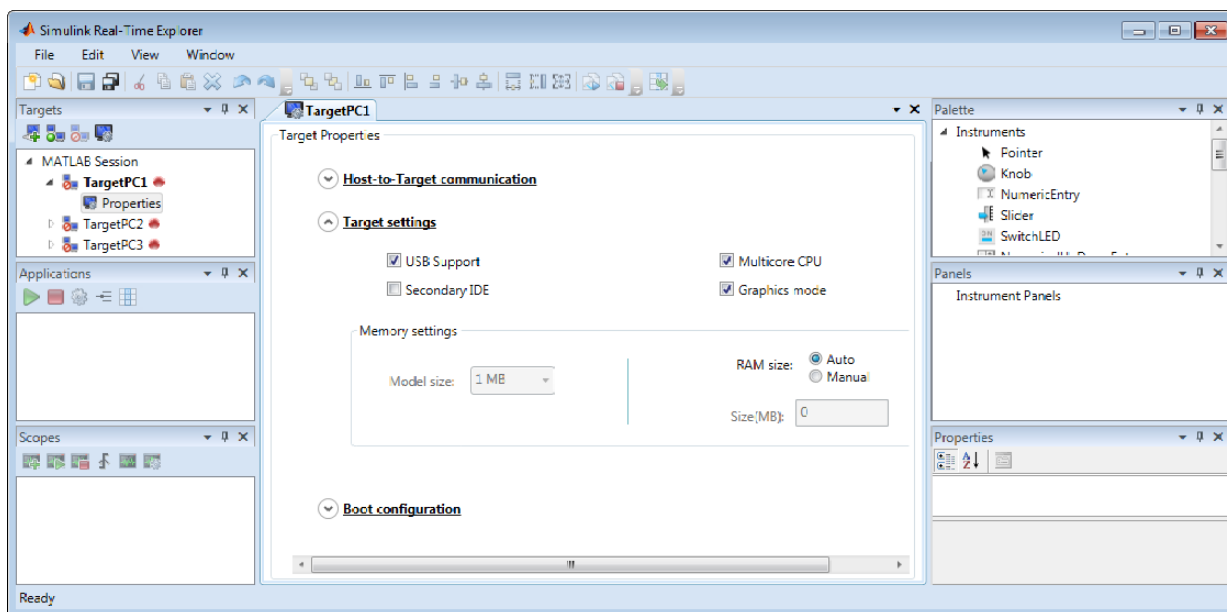
Repeat this procedure as required for each target computer.

Target Boot Methods

You can start your target computer with the Simulink Real-Time kernel using one of several methods.

Speedgoat systems come with `DOS Loader` software preinstalled. You can set up the `DOS Loader` boot method on your development computer or configure another boot method. See your Speedgoat system documentation or follow the link from "Speedgoat Real-Time Target Machines" for further information.

1   Before creating a boot kernel, perform "Kernel Creation Prechecks" on page 2-24.

2   Select one of the following methods:

- "Network Boot Method" on page 2-26
- "CD/DVD Boot Method" on page 2-25
- "DOS Loader Boot Method" on page 2-32
- "Removable Disk Boot Method" on page 2-30
- "Standalone Boot Method" on page 2-36

3   For boot methods other than `Stand Alone`, perform "Run Confidence Test on Configuration" on page 2-43.

For boot method `Stand Alone`, create a model-specific confidence test, restart the target computer, and run that confidence test. The default confidence test is not intended for standalone execution.

Kernel Creation Prechecks

Configure your Simulink Real-Time system before creating the target boot kernel. At a minimum, do the following:

1   Check the physical connections between the development and target computers. The Ethernet connections may pass through a LAN.

2   Check your target computer BIOS settings (see "BIOS Settings" on page 2-8).

3   Check that you have write permission for your current working folder.

4   In the MATLAB Command Window, type `slrtexplr`.

5   In the **Targets** pane, expand the target computer node.

6   On the toolbar , click the **Target Properties** button .

7   In the **Target Properties** pane, click **Host-to-Target communication**.

8   Check the link settings. See "PCI Bus Ethernet Setup" on page 2-12 or "USB-to-Ethernet Setup" on page 2-16.

9   In the **Target Properties** pane, click **Boot configuration**.

10   Check that **Boot mode** is set to the required value.

11
Press **Enter**, then click the **Save** button on the toolbar.

Repeat this procedure as required for each target computer.

CD/DVD Boot Method

After you have configured the target computer environment parameters, you can use Simulink Real-Time Explorer to create a boot CD or DVD that loads and runs the Simulink Real-Time kernel. Your development computer must run under Microsoft Windows 7 or later.

To create a bootable CD/DVD using MATLAB language, see "Command-Line CD/DVD Boot Method".

To create a boot CD or DVD:

1 In the MATLAB Command Window, type `slrtexplr`.

2 In the **Targets** pane, expand the target computer node.

3 On the toolbar , click the **Target Properties** button .

4 Select **Boot configuration** and set **Boot mode** to CD.

5 Click **Create boot disk**.

6 In the **Select CD-ROM Drive** list, select the CD/DVD read/write drive.

7 When prompted, insert an empty CD or DVD in the development computer CD/DVD read/write drive.

8 Click **Burn Disk**. A progress bar is displayed.

9 When the write operation has finished, click **OK** in the status dialog box.

10 Remove the CD or DVD from the drive.

11 Insert the bootable CD/DVD into your target computer CD/DVD drive and restart the target computer.

12 Press **Enter**, then click the **Save** button on the toolbar.

Repeat this procedure as required for each target computer.

The next task is "Run Confidence Test on Configuration" on page 2-43.

Network Boot Method

After you have configured the target computer environment parameters, you can use a dedicated Ethernet network to load and run the Simulink Real-Time kernel. You do not need a boot CD or a removable boot drive.

There are the following limitations:

- Do not use the network boot method on a corporate or nondedicated network. Doing so might interfere with dynamic host configuration protocol (DHCP) servers and cause problems with the network.

- Your Ethernet card must be compatible with the Preboot eXecution Environment (PXE) specification.

- If **Boot mode** Stand Alone is enabled, you cannot start the target computer across the network.

Before you start, establish an Ethernet connection between the development and target computers, using the procedures in "PCI Bus Ethernet Setup" on page 2-12 or "USB-to-Ethernet Setup" on page 2-16.

1 In the MATLAB Command Window, type slrtexplr.

2 In the **Targets** pane, expand the target computer node.

3 On the toolbar , click the **Target Properties** button .

4 Select **Boot configuration** and set **Boot mode** to Network.

5 If required, click **Reset**.

   Clicking **Reset** clears the MAC address. The next time the target computer starts, by default, the software automatically obtains the MAC addresses of accessible target computers and displays them for confirmation in the Simulink Real-Time Network Boot dialog box.

6 To enter the MAC address of the target computer manually, click **Reset**. In the **MAC address** box, enter the address in the format xx:xx:xx:xx:xx:xx.

   The next time the target computer starts, the software selects and starts the target computer that matches this MAC address. The Simulink Real-Time Network Boot dialog box does not open.

7 Click **Create boot disk**.

The software creates and starts a network boot server process on the development computer. You see a minimized icon () in the bottom right system tray on the development computer.

8   Turn on the target computer.

9   Enter the target computer BIOS and set up the target computer for a LAN or network boot.

   If the BIOS allows you to change the boot order, consider setting the boot order so that the floppy disk, CD, or USB drive come before the LAN option. Doing so allows you to start the target computer from a network kernel even if the target computer does not have a kernel boot disk or removable drive.

10 Restart the target computer.

11 The first time the network boot server process detects a target computer, it displays the Simulink Real-Time Network Boot dialog box. This dialog box contains physical target computer names and the corresponding IP addresses.

Simulink Real-Time Network Boot

Boot request received from an unknown computer:

Detected MAC Address: 90:e2:ba:17:5d:15

☐ Always ignore this MAC Address

Please choose a target to start network boot.

Target Name	IP Address	Status
TargetPC1	10.10.10.15	Available
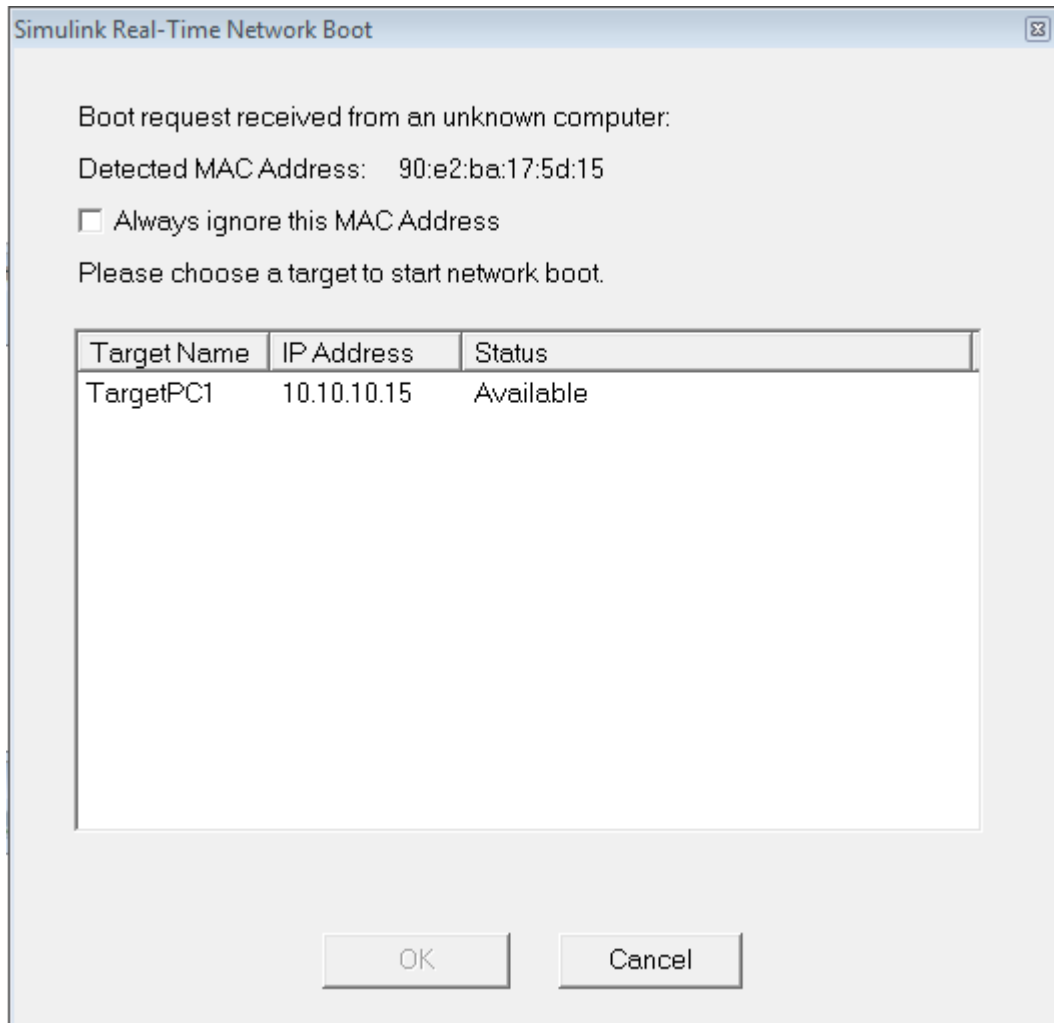
OK   Cancel

a   Select the target computer name for the physical target computer.

b   Click **OK**.

The development computer network boot server displays a message indicating that the boot server is being downloaded to the target computer.

If you click **Cancel** instead of selecting a target computer name, the kernel ignores the start request for 90 seconds the next time you try to start the target computer across the network.

12 Press **Enter**, then click the **Save** button 🖫 on the toolbar.

The Simulink Real-Time Explorer window looks like this figure.
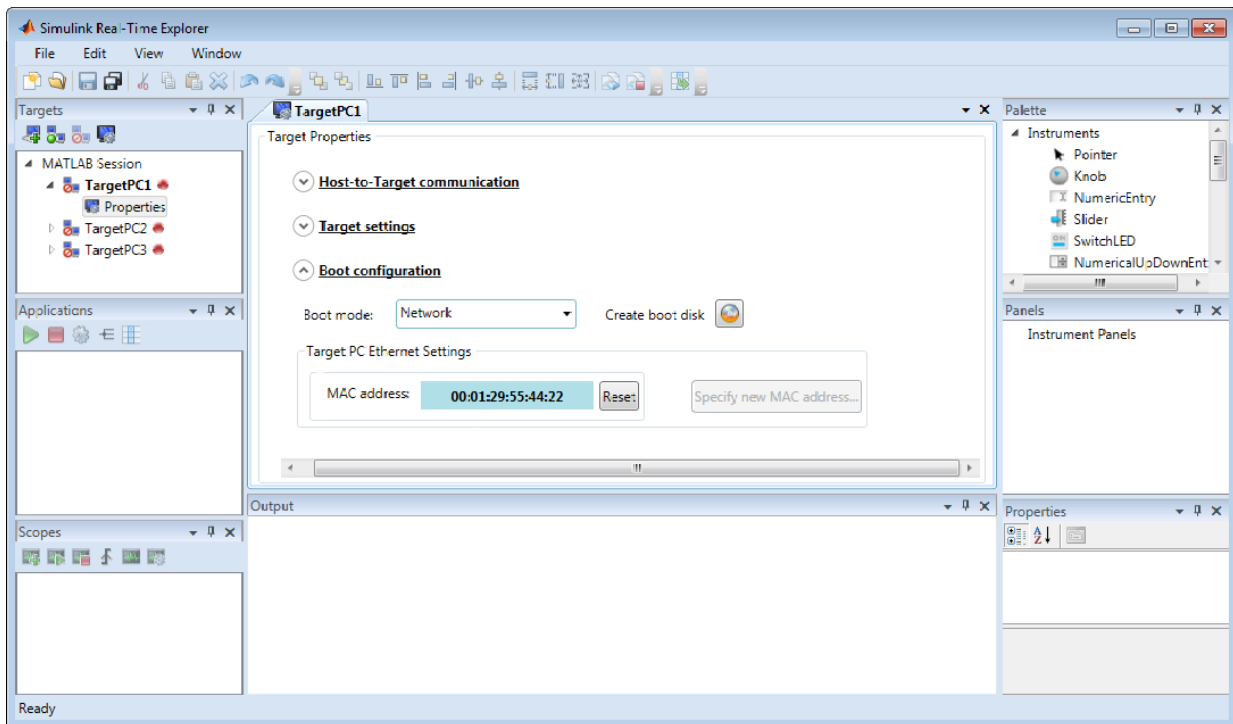
Repeat this procedure as required for each target computer.

Tip To configure network boot using MATLAB language, see "Command-Line Network Boot Method".

The next task is "Run Confidence Test on Configuration" on page 2-43.

Removable Disk Boot Method

After you have configured the target computer environment parameters, you can use a removable drive or USB flash drive to load and run the Simulink Real-Time kernel. This topic describes using Simulink Real-Time Explorer to create a removable boot disk.

If you are creating a removable boot drive from a USB flash drive, before performing this procedure, create a bootable partition on the drive. See "Create a Bootable Partition" on page 2-31.

To create a removable boot disk using MATLAB language, see "Command-Line Removable Disk Boot Method".

To create a removable boot disk:

1 In the MATLAB Command Window, type slrtexplr.

2 In the **Targets** pane, expand the target computer node.

3 On the toolbar , click the **Target Properties** button

4 Select **Boot configuration** and set **Boot mode** to Removable Disk.

5 If you are creating a removable boot disk from a USB drive, insert the USB drive in the development computer USB port. Wait for it to be recognized.

6 Click **Create boot disk**.

7 In the **Removable Disk Selector** list, select the drive.

8 If required, insert an empty removable disk in the development computer drive and click **OK**. A progress bar is displayed.

9 When the write operation has finished, remove the removable disk from the drive or USB port.

10 Insert the removable boot disk into your target computer drive or USB port. Restart the target computer.

11 Press **Enter**, then click the **Save** button on the toolbar.

Repeat this procedure as required for each target computer.

The next task is "Run Confidence Test on Configuration" on page 2-43.

Create a Bootable Partition

Before you create a removable boot drive from a USB flash drive, use the Windows command `diskpart` to create a bootable partition on the drive. Consult the `diskpart` documentation for more details.

1 On the development computer, open a DOS command window.

2 In the DOS command window, type:

   ```
   diskpart
   ```

3 At the diskpart prompt, type:

   ```
   list disk
   ```

   Note the disk numbers of the existing development computer disks.

4 Insert the flash drive. Wait for the development computer to recognize the drive. Note the drive device name.

5 At the diskpart prompt, type:

   ```
   list disk
   ```

   Note the new disk number *N*. This number is the disk number of the drive.

6 Type:

   ```
   select disk N
   ```

7 **Caution** The `clean` command deletes all of the data from disk *N*. Select the correct disk, or you might delete all of the data from your development computer.

   Type:

   ```
   clean
   create partition primary
   select partition 1
   active
   format fs=fat32 quick
   exit
   ```

8 In the development computer task bar, click **Safely remove hardware and eject media** and select the flash drive name.

9 Remove the drive from the development computer.

DOS Loader Boot Method

DOS Loader mode allows you to start the Simulink Real-Time kernel on a target computer from a fixed or removable device with DOS boot capability, such as a hard disk or flash memory. After starting the target computer, you can download your real-time application from the development computer over a serial or Ethernet link between the development and target computers.

Speedgoat systems come with DOS Loader software preinstalled. You can set up the DOS Loader boot method on your development computer or configure another boot method. See your Speedgoat system documentation or follow the link from "Speedgoat Real-Time Target Machines" for further information.

To run in DOS Loader mode, the target computer boot device must provide a minimal DOS environment complying with certain restrictions. For details, see "Create a DOS System Disk" on page 2-34 and "DOS Loader Mode Restrictions" on page 2-35.

To create a DOS Loader boot disk using MATLAB language, see "Command-Line DOS Loader Boot Method".

To create and use DOS Loader files:

1   In the MATLAB Command Window, type slrtexplr.

2   In the **Targets** pane, expand the target computer node.

3   On the toolbar , click the **Target Properties** button .

4   Select **Boot configuration** and set **Boot mode** to DOS Loader.

5   In the **Location** field, enter or browse to the folder where you want to create the DOS Loader boot files. This location can be a local folder on the development computer or a removable storage device that you use to start the target computer. By default, the folder is the current working folder.

6   Click **OK**. A progress bar appears to indicate execution.

    This operation creates the following boot files in the specified location:
        autoexec.bat
        xpcboot.com
        *.rtb

7   If you create boot files on a local hard disk, copy these files to a floppy disk, CD/DVD, or other removable storage media.

8   Transfer the boot files to your target computer or insert the removable media containing the boot files into the target computer drive or USB port.

9   Verify that `autoexec.bat` file is on the DOS boot path (typically the root folder).

10   In the BIOS of the target computer, select the required boot device.

11   Start the target computer.

When the target computer starts, it loads DOS, which executes the `autoexec.bat` file. This file starts the Simulink Real-Time kernel (*.rtb). The target computer then awaits commands from the development computer.

12

Press **Enter**, then click the **Save** button on the toolbar.

Repeat this procedure as required for each target computer.

The next task is "Run Confidence Test on Configuration" on page 2-43.

Create a DOS System Disk

To use the DOS-based modes such as `DOS Loader` and `Stand Alone`, you need a minimal DOS system on the target computer boot device.

Simulink Real-Time software does not include a DOS license. You must obtain a valid DOS license for your target computer.

To create a DOS system disk:

1 Acquire a copy of a DOS system, such as FreeDOS, MS-DOS®, PC DOS, or Caldera OpenDOS.

2 Use the following DOS command to copy the DOS system files and command interpreter from *drive1* to the boot device, *drive2*:

   `sys drive1 drive2`

3 Copy additional DOS utilities to the boot disk, including:

   - DOS editor — to edit text files on the target computer.
   - `format` program — to format a hard disk or flash memory.
   - `fdisk` program — to create partitions.
   - `sys` program — to transfer a DOS system onto another drive, such as the hard disk drive.

   A `config.sys` file is not required.

4 Transfer additional files to the disk as required.

DOS Loader Mode Restrictions

To use DOS Loader mode, the target computer DOS environment must comply with the following restrictions:

- The CPU must execute in real mode.
- While loaded in memory, the DOS partition must not overlap the address range of a real-time application.

To satisfy these restrictions:

- Avoid additional memory managers, such as `emm386` or `qemm`.
- Avoid utilities that attempt to load in high memory (for example, `himem.sys`).
- Avoid including memory manager entries in the `autoexec.bat` file.
- Avoid using a `config.sys` file.

Standalone Boot Method

You can configure the Simulink Real-Time software to run as a standalone real-time application. For more information on **Boot mode** `Stand Alone`, see "Standalone Mode" on page 2-41.

To run in `Stand Alone` mode, the target computer and its DOS environment must meet specific requirements and restrictions.

In this section...
"Target Computer Requirements" on page 2-36
"DOS Environment Restrictions" on page 2-37
"Standalone Settings" on page 2-37
"Real-Time Application Build" on page 2-38
"Real-Time Application Transfer" on page 2-39
"Real-Time Application Boot Configuration" on page 2-39

Target Computer Requirements

To run in `Stand Alone` mode, the target computer must meet specific requirements.

In `Stand Alone` mode, the real-time application is combined with the kernel on a hard drive or flash memory. The kernel and real-time application start together on the target computer. The development computer does not need to be connected to the target computer, but it must be able to start from DOS.

For each target computer, verify that the system meets the following requirements:

- The target computer hard drive must be a serial ATA (SATA) or parallel ATA (PATA)/Integrated Device Electronics (IDE) drive.
- The target computer must not cable select the hard drive.
- The target computer BIOS must be able to detect the hard drive.
- The hard drive must be formatted with one of these file systems:

  - FAT-12
  - FAT-16

· FAT-32

- You must install a supported version of DOS on the target computer hard drive. If required, create a standard DOS boot device from a CD ROM, 3.5–inch floppy drive, flash drive, or hard drive. See "Create a DOS System Disk" on page 2-34.

- You must configure the Ethernet link. This communication transfers the kernel and real-time application files built on the development computer to the target computer.

DOS Environment Restrictions

To use `Stand Alone` mode, the target computer DOS environment must comply with the following restrictions:

- The CPU must execute in real mode.

- While loaded in memory, the DOS partition must not overlap the address range of a real-time application.

To satisfy these restrictions:

- Avoid additional memory managers, such as `emm386` or `qemm`.

- Avoid utilities that attempt to load in high memory (for example, `himem.sys`).

- Avoid including memory manager entries in the `autoexec.bat` file.

- Avoid using a `config.sys` file.

There are additional restrictions on target and scope settings:

- In the Simulink Real-Time Explorer **Target settings** pane, you can use only **Model size** values `1 MB` and `4 MB`. Setting **Model size** is enabled for **Boot mode** `Stand Alone` only.

- If you want to use real-time `Scope` blocks to display or record output:

  - You must use only `Target` or `File` type blocks.

  - You must select the **Start scope when application starts** check box in the block parameters dialog box.

Standalone Settings

Use Simulink Real-Time Explorer to set the kernel environment properties. When you are done, you can create a standalone kernel and real-time application.

For **Boot mode** `Stand Alone`, you do not create a Simulink Real-Time boot disk or network boot image. Instead, you copy files created from the build process to the target computer hard drive.

1 In the MATLAB Command Window, type `slrtexplr`.

2 In the **Targets** pane, expand the target computer node.

3 On the toolbar , click the **Target Properties** button .

4 Select **Boot configuration** and set **Boot mode** to `Stand Alone`.

5 Press **Enter**, then click the **Save** button on the toolbar.

Repeat this procedure as required for each target computer.

Real-Time Application Build

After you set Simulink Real-Time boot mode to `Stand Alone`, you can use Simulink Real-Time, Simulink Coder, and a C/C++ compiler in `Stand Alone` mode to build a combined kernel and real-time application with utility files.

1 In the Command Window, open your Simulink model, for example, `ex_slrt_rt_osc` (matlab: open_system(docpath(fullfile(docroot, 'toolbox', 'xpc', 'examples', 'ex_slrt_rt_osc')))). Simulink Editor opens displaying the model.

2 From the **Code** menu, click **C/C++ Code** > **Build Model**.

The Simulink Coder and Simulink Real-Time software create a folder, `ex_slrt_rt_osc_slrt_emb`, containing the following files:

- `autoexec.bat` — Contains Simulink Real-Time-specific code that calls the `xpcboot.com` executable to start the Simulink Real-Time kernel (the `*.rtb` file).

- `xpcboot.com` — Loads and executes the `*.rtb` file. This static file is part of the Simulink Real-Time software.

- `ex_slrt_rt_osc.rtb` — Contains the Simulink Real-Time kernel and the real-time application. This image also contains applicable options, such as serial or Ethernet communications and the COM port or IP address of the target computer.

You do not need a `config.sys` file to start the kernel using `autoexec.bat` and `xpcboot.com`.

The **Model size** parameter gives an upper bound on the size of the compiled real-time application (DLM). It is only enabled in `Stand Alone` mode. You can use only **Model size** values `1 MB` and `4 MB`. If the DLM is larger than **Model size**, the build software generates an error.

Repeat this procedure as required for each real-time application.

Real-Time Application Transfer

After building the kernel and real-time application on a development computer, transfer the files to a target computer using the Simulink Real-Time FTP functions:

1   Start the target computer from a target computer boot disk.

2   In the Command Window, change to the folder that contains the kernel and real-time application files.

3   In the Command Window, create a Simulink Real-Time FTP object:

```
fsys = SimulinkRealTime.fileSystem
```

4   Use the FTP object to create the folder `C:\xpcfiles` on the target computer:

```
mkdir(fsys, 'xpcfiles')
```

5   Move to the `xpcfiles` folder:

```
cd(fsys, 'xpcfiles')
```

6   Copy these files to the `xpcfiles` folder:

```
tg = slrt;
SimulinkRealTime.copyFileToTarget(tg,'autoexec.bat')
SimulinkRealTime.copyFileToTarget(tg,'xpcboot.com')
SimulinkRealTime.copyFileToTarget(tg,'ex_slrt_rt_osc.rtb')
```

7   Remove the boot disk from the target computer.

Repeat this procedure as required for each target computer.

Real-Time Application Boot Configuration

After transferring the files to a target computer, configure the target computer to run the real-time application automatically upon start.

2-39

1 Restart the target computer and bring up the DOS prompt.

If the target computer that you want to start in `Stand Alone` mode was previously started from the network boot image, select the `Stand Alone` check box to disable the network boot capability.

2 At the DOS prompt, save a copy of the target computer `C:\autoexec.bat` file to a backup file, such as `C:\autoexec_back.wrk`.

3 Edit the target computer file `C:\autoexec.bat` to include the following lines:

```
cd C:\xpcfiles
autoexec
```

Adding these commands to `C:\autoexec.bat` directs the system to execute the `autoexec.bat` file located in `C:\xpcfiles`.

4 Restart the target computer.

The target computer executes the following sequence of calls:

a `C:\autoexec.bat`

b `C:\xpcfiles\autoexec.bat`

c `C:\xpcfiles\xpcboot.com`

d `C:\xpcfiles\<application>.rtb`

Repeat this procedure as required for each target computer.

Note: Do not confuse `C:\xpcfiles\autoexec.bat` with `C:\autoexec.bat`. `C:\xpcfiles\autoexec.bat` includes the command `xpcboot.com`, which starts the Simulink Real-Time kernel and the standalone real-time application. `C:\autoexec.bat` includes the files you want the system to execute when the system starts.

Continue by restarting the target computer and testing your real-time application in `Stand Alone` mode. The confidence test is not intended for standalone execution. You must create and execute your own confidence test for `Stand Alone` mode.

Standalone Mode

In **Boot mode** `Stand Alone` you can create control, DSP , and other system models for use in production systems running on target computers. Typically these production systems are manufactured in low to moderate quantities.

In `Stand Alone` mode, you can bundle the kernel and real-time application into one entity on the target computer, independent of the development computer. You can configure a target computer to automatically start executing your real-time application for continuous operation each time the system is started. If the target computer has a keyboard, you can control the real-time application using the target computer command-line interface. You can create custom interfaces with the Simulink Real-Time C and .NET APIs and run them on Microsoft Windows development computers, without installing MATLAB software.

Note: `Stand Alone` mode does not require a connection between the development and target computers.

Use this mode to load the target computer with both the Simulink Real-Time kernel and a real-time application. This mode of operation can start the kernel on the target computer from a flash disk or hard disk. After starting the kernel on the target computer, `Stand Alone` mode also automatically starts the real-time application that you loaded with the kernel. `Stand Alone` mode eliminates the need for a development computer and allows you to run real-time applications on target computers.

Regardless of the mode, you initially start your target computer with DOS from a boot device. DOS starts the Simulink Real-Time kernel. After the Simulink Real-Time kernel starts, DOS is no longer available on the target computer. To make DOS available, you must restart the target computer with DOS without starting the Simulink Real-Time kernel.

Simulink Real-Time standalone mode requires a boot device with DOS installed. Otherwise, it does not have specific requirements as to the type of boot device. DOS software and license are not included with Simulink Real-Time. If DOS is not installed, you must first start the target computer from a Simulink Real-Time boot disk or network boot image, and only then download and run the real-time application.

In the following instances, you might want to use standalone mode. If you use the target computer in a small or rugged environment, you might have one of these situations:

- The target computer does not have a removable drive.
- After setting up the target system, you must remove the target computer removable drive.
- The target computer does not support network boot from the development computer.

If you want to view signals on the target computer in `Stand Alone` mode, you must provide a monitor for the target computer. Before building the real-time application, you must also add real-time `Scope` blocks of types `target` and `file`.

If you do not want to view signals on the target computer, you can run your Simulink Real-Time system as a black box without a monitor or keyboard. Standalone real-time applications are automatically set to continue running for an infinite time or until the target computer is turned off.

Run Confidence Test on Configuration

This topic describes how to use a script to test the target boot process, the connection between the development and target computers, and the basic functionality of the Simulink Real-Time software. Use this test under the following circumstances:

- To validate your initial product installation.
- As the first step in a troubleshooting procedure.

You must be using TCP/IP for starting the target computer.

1 Create a network boot image and restart the target computer. See "Network Boot Method" on page 2-26.

2 In the current folder window, select a folder outside the MATLAB root folder.

You cannot save build files within the MATLAB root folder because Simulink Coder does not allow this action. If you select a current folder within the MATLAB tree, the Simulink Real-Time test procedure fails when trying to build a real-time application.

3 In the Command Window, type `slrttest`.

MATLAB runs the test script for the default target computer and displays messages indicating whether the test passed or failed.

```
### Simulink Real-Time Test Suite
### Host-Target interface is: TcpIp
### Test 1, Ping target computer 'TargetPC1' using system ping: ... OK
### Test 2, Ping target computer 'TargetPC1' using SLRTPINGTARGET: ... OK
### Test 3, Software reboot the target computer 'TargetPC1': ........... OK
### Test 4, Build and download a Simulink Real-Time application using model ...
    slrttestmdl to target computer 'TargetPC1': ... OK
### Test 5, Check host-target command communications with 'TargetPC1': ... OK
### Test 6, Download a pre-built Simulink Real-Time application to target ...
    computer 'TargetPC1': ... OK
### Test 7, Execute the Simulink Real-Time application for 0.2s: ... OK
### Test 8, Upload logged data and compare with simulation results: ... OK
### Test Suite successfully finished
```

4 Evaluate the results.

- If the tests return OK, you are ready to build and download a real-time application to the target computer.
- If one or more tests return FAILED, see "Troubleshooting in Simulink Real-Time".

The next task is "Application and Driver Scripts" on page 4-56.

3

Basic Workflows

Rapid Control Prototyping Process

Use rapid control prototyping to test a design with a physical system plant model.

1 **Create a Simulink or Stateflow® model.**

   Create block diagrams in Simulink by dragging blocks to your model. If possible, set model and block parameters to use a fixed-step solver and specify a sample time compatible with the real-time requirements of your model.

2 **Simulate the model as a non-real-time application.**

   Simulink uses a computed time vector to step the model. After computing the outputs for a given time value, Simulink immediately repeats the computation for the next time value until it reaches the stop time.

   Because the computed time vector is not connected to a physical clock, the outputs are calculated as fast as your computer can run. The elapsed time of the simulation can differ significantly from the elapsed time of the real system.

   You can log simulation results for later comparison.

3 **Configure the development and target computers.**

   Configure the communication method between the development and target computers.

   Configure the development and target computers using:

   - "Simulink Real-Time Explorer" on page 1-5
   - "MATLAB Command-Line Interface" on page 1-8

4 **Prepare the model for real-time execution.**

   Set the model Configuration Parameters to values compatible with real-time execution:

   - In the **Code Generation** pane, set **System target file** to `slrt.tlc` or `slrtert.tlc`.
   - In the **Solver** pane:

      - Set **Type** to `Fixed-step`.

- Set **Fixed-step size** to a step size compatible with the real-time requirements of your model.

Add Simulink Real-Time I/O blocks representing your I/O boards. If you have a custom I/O board, create a custom driver block representing the board. See "Add I/O Blocks to Simulink Model".

To visualize the simulation results, add real-time scope blocks. See "Add Simulink Real-Time Scope Block" on page 4-24.

5 **Configure the build environment.**

To create a real-time application that runs on the target computer, configure the build environment. The environment includes Simulink Coder options, Simulink Real-Time build options, and C compiler options.

6 **Connect to hardware under test.**

Install I/O modules in the target computer and connect the modules to the hardware under test.

7 **Restart the target computer.**

Restart the target computer with the Simulink Real-Time real-time kernel using:

- Target computer restart button.
- "MATLAB Command-Line Interface" on page 1-8

8 **Build and download the real-time application.**

Build and download the real-time application using:

- "Simulink Real-Time Explorer" on page 1-5 (load and unload only)
- "MATLAB Command-Line Interface" on page 1-8
- "Simulink External Mode Interface" on page 1-10
- "Target Computer Command-Line Interface" on page 1-12

9 **Execute the real-time application.**

Execute the real-time application under command from the development computer or by restarting the target computer in standalone mode.

The Simulink Real-Time software uses real-time resources on the target computer. Based on your sample rate, the Simulink Real-Time software uses interrupts to step

the model at the sample rate. With each new interrupt, the real-time application computes the block outputs from your model.

Execute using:

- "Simulink Real-Time Explorer" on page 1-5
- "MATLAB Command-Line Interface" on page 1-8
- "Simulink External Mode Interface" on page 1-10
- "Target Computer Command-Line Interface" on page 1-12

10 Visualize signals.

Create real-time scopes and Simulink Real-Time Explorer instruments. Use them to acquire and display signal data from the real-time application.

Scopes created by real-time scope blocks acquire data according to Simulink sample time rules. Scopes can gather data at the top level or in an enabled or triggered subsystem. Scopes created dynamically (from the MATLAB Command Window or the API) sample at the base rate, irrespective of the sample time of their signals.

To create instrument panels, use Simulink Real-Time Explorer to create instrument panels. You can drag graphical instruments to the instrument panels, and drag signals to the instruments to display signal data.

Visualize signals using:

- "Simulink Real-Time Explorer" on page 1-5
- "MATLAB Command-Line Interface" on page 1-8
- "Simulink External Mode Interface" on page 1-10
- "Simulink with Simulink Real-Time Blocks" on page 1-11
- "Target Computer Command-Line Interface" on page 1-12

11 Tune parameters.

Tune observable parameters such as time delays, input and output amplitudes, and input and output frequencies.

Note: Simulink Real-Time does not support parameters of multiword data types.

Tune parameters using:

- "Simulink Real-Time Explorer" on page 1-5
- "MATLAB Command-Line Interface" on page 1-8
- "Simulink External Mode Interface" on page 1-10
- "Target Computer Command-Line Interface" on page 1-12

12 Prepare regression and stress tests.

Write MATLAB scripts that perform parameter sweep and extreme-value testing in a repeatable manner, accumulating results as known good data for later use.

Hardware-in-the-Loop Simulation Process

Use hardware-in-the-loop (HIL) simulation to verify a physical prototype of the product with a Simulink Real-Time plant model. HIL simulation builds on the test harness and simulation results of rapid control prototyping. You can test the system for safety and performance and test operation and failure conditions that are difficult to replicate. You can substitute for unavailable parts of the system and minimize expensive downtime for the rest of the system.

With HIL simulation, you perform one or more of the following tasks:

- **Model the plant.** — Use Simulink and Simulink Real-Time to model the plant for testing the physical prototype. To debug the plant model, run it with known-good data from rapid control prototyping.

- **Connect to a physical prototype.**—Modify the system model to replace the Simulink Real-Time design with an I/O board connected to the physical prototype.

- **Program the physical prototype.** — Use HDL Coder to generate FPGA code to program the physical prototype.

- **Test the physical prototype.** — Visualize signals and tune parameters using Simulink Real-Time scopes and instruments. To write regression tests. parameter sweep tests, and edge tests, use data from rapid control prototyping.

As required, perform the following additional tasks:

- **Create instrument panels in Simulink Real-Time Explorer** — Configure Simulink Real-Time Explorer with real-time scopes and with signal display and parameter tuning graphical instruments.

- **Configure a standalone application.** — Configure the real-time application to run standalone.

- **Create a standalone program from a Simulink Real-Time Explorer configuration** — Export the Simulink Real-Time Explorer configuration as a standalone program.

- **Execute the real-time application and display the results.** — Use the exported Simulink Real-Time Explorer configuration to execute the standalone real-time application. To display signal values and tune parameters, use the scopes and graphical instruments.

- **Write a regression test harness.** — Extend the MATLAB regression tests written for rapid control prototyping to cover functionality over the full parameter range.

- **Write a graphical test harness.** — Use one of the APIs (.NET, C) to write a graphical test harness suitable for probing real-time application behavior from the development computer. As part of this effort, you can use MATLAB Coder to translate MATLAB regression test scripts into C for integration into the test environment.

As a test engineer, you can build on the plant model and HIL test scripts to create repeatable product tests for volume manufacturing. For example, you can extend and categorize the regression tests into smoke, go/no-go, and diagnostic tests. You can then use MATLAB Coder to translate MATLAB regression test scripts into C for integration into a production test environment.

4

Tutorial and Examples

This topic describes Simulink Real-Time functionality with a simple Simulink model without I/O blocks. You can try these procedures whether or not you have I/O modules on the target computer. Once you are familiar with the setup, build, and target execution process, you can try some of the more advanced Simulink Real-Time examples.

Set Up and Configure Simulink Real-Time

You must have the following configuration:

- Single PCI bus target computer
- Network target boot method

You must have installed and configured a C compiler as part of Simulink Real-Time installation. If not, see "Command-Line C Compiler Configuration".

In this section...
"Configure Link Between Development and Target Computers" on page 4-2
"Configure Target Settings" on page 4-4
"Configure Boot Configuration" on page 4-5
"Run the Confidence Test" on page 4-6

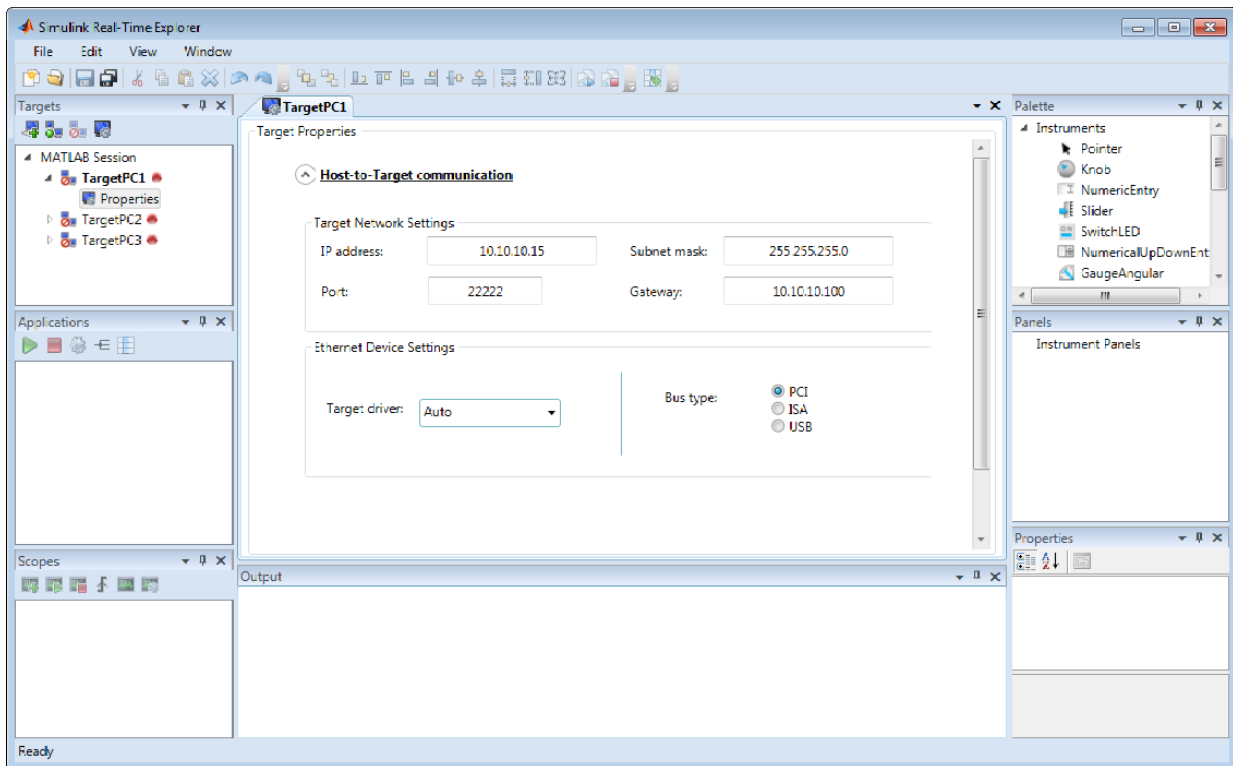## Configure Link Between Development and Target Computers

To run a Simulink Real-Time model on a target computer, you must connect the development and target computers in a network.

The **Target Network Settings** values are representative only. Consult your network administrator for actual values. For more on network configuration, see "PCI Bus Ethernet Setup" on page 2-12 and "USB-to-Ethernet Setup" on page 2-16.

1 In the Command Window, type slrtexplr.

2 In the **Targets** pane, expand the target computer node.

3 On the toolbar, click the **Properties** button .

4 In the **Target Properties** workspace, click **Host-to-Target communication**.

5 Under **Target Network Settings**, set values such as the following.

- **IP address**: 10.10.10.15
- **Port**: 22222
- **Subnet mask**: 255.255.255.0

- **Gateway**: `10.10.10.10`

6 Under **Ethernet Device Settings**, set the following values:

- **Target driver**: `Auto`
- **Bus type**: **PCI**

7 Press **Enter**, then click the **Save** button 🖫 on the toolbar.

The dialog box looks like this figure.

Note: In R2017a, support for using ISA bus Ethernet cards to communicate between the development and target computers will cease to function. Use PCI bus or USB bus Ethernet cards instead.

Configure Target Settings

To run a Simulink Real-Time model on a target computer, you must configure the target settings.

1 In the Command Window, type `slrtexplr`.

2 In the **Targets** pane, expand the target computer node.

3 On the toolbar, click the **Properties** button .

4 In the **Target Properties** workspace, click **Target settings**.

5 Select **USB Support**, **Multicore CPU**, and **Graphics mode**.

6 Clear **Secondary IDE**.

7 Under **Memory settings**, select **Ram size Auto**.

8 Press **Enter**, then click the **Save** button on the toolbar.

The dialog box looks like this figure.
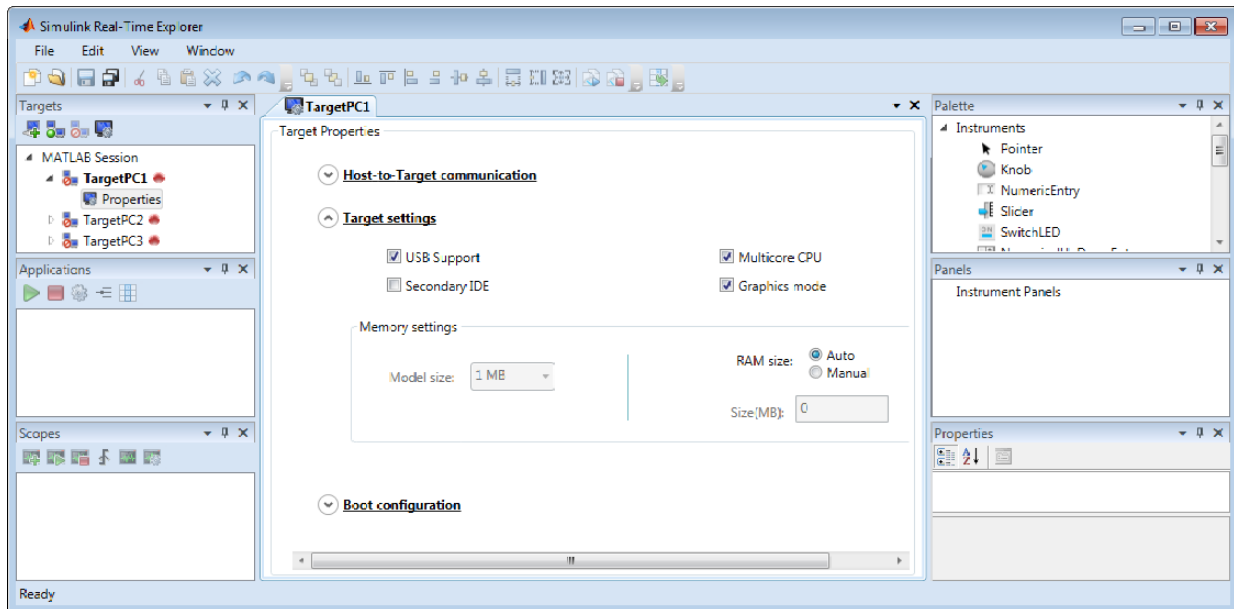
Note:

- Setting **Size(MB)** is enabled for **RAM size Manual** only.

- Setting **Model size** is enabled for **Boot mode** Stand Alone only.

Configure Boot Configuration

To run a Simulink Real-Time model on a target computer, you must configure the target boot method. For more on boot methods, see "Target Boot Methods" on page 2-23.

1 In the Command Window, type slrtexplr.

2 In the **Targets** pane, expand the target computer node.

3 On the toolbar, click the **Properties** button ![icon].

4 In the **Target Properties** workspace, click **Boot Configuration**.

5 Select **Boot mode** Network.

6 Click **Create boot disk**.

7
   Press **Enter**, then click the **Save** button ![icon] on the toolbar.

The dialog box looks like this figure.
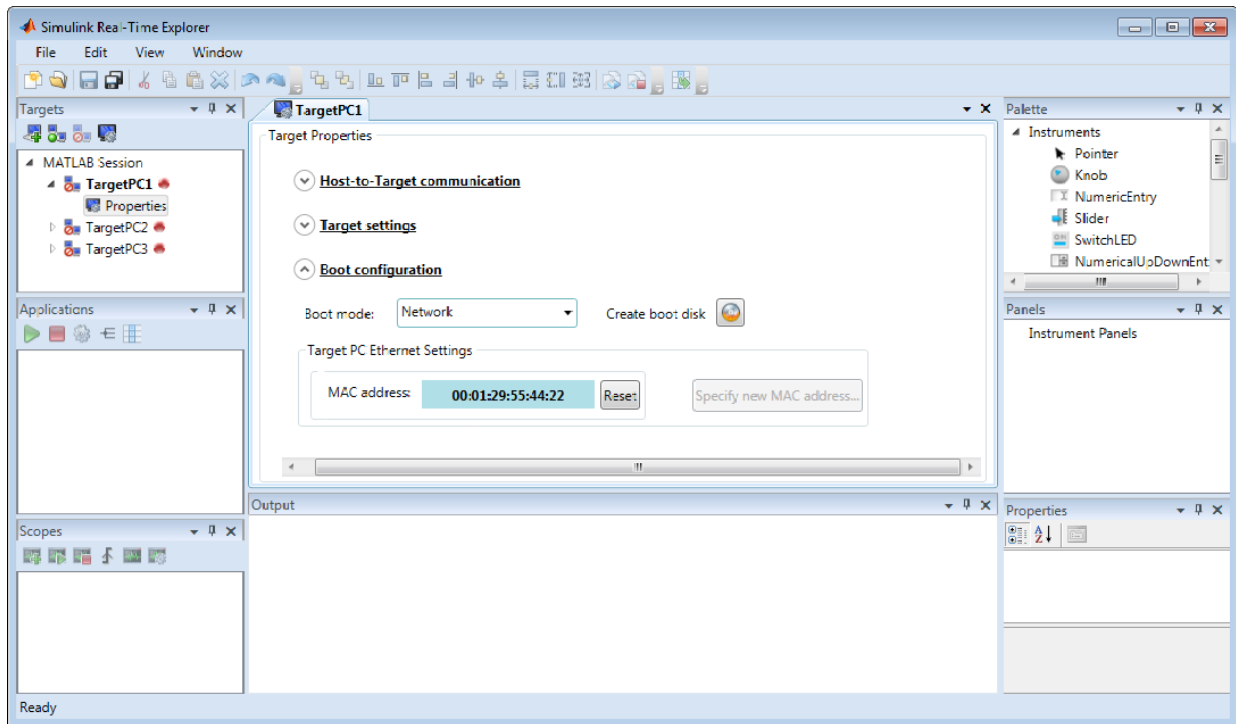
Run the Confidence Test

Validate the setup and configuration by running the confidence test.

1 Create a network boot image and restart the target computer. See "Network Boot Method" on page 2-26.

2 In the current folder window, select a folder outside the MATLAB root folder.

You cannot save build files within the MATLAB root folder because Simulink Coder does not allow this action. If you select a current folder within the MATLAB tree, the Simulink Real-Time test procedure fails when trying to build a real-time application.

3 In the Command Window, type `slrttest`.

MATLAB runs the test script for the default target computer and displays messages indicating whether the test passed or failed.

```
### Simulink Real-Time Test Suite
```

```
### Host-Target interface is: TcpIp
### Test 1, Ping target computer 'TargetPC1' using system ping: ... OK
### Test 2, Ping target computer 'TargetPC1' using SLRTPINGTARGET: ... OK
### Test 3, Software reboot the target computer 'TargetPC1': ........... OK
### Test 4, Build and download a Simulink Real-Time application using model ...
    slrttestmdl to target computer 'TargetPC1': ... OK
### Test 5, Check host-target command communications with 'TargetPC1': ... OK
### Test 6, Download a pre-built Simulink Real-Time application to target ...
    computer 'TargetPC1': ... OK
### Test 7, Execute the Simulink Real-Time application for 0.2s: ... OK
### Test 8, Upload logged data and compare with simulation results: ... OK
### Test Suite successfully finished
```

4 Evaluate the results.

- If the tests return OK, you are ready to build and download a real-time application to the target computer.

- If one or more tests return FAILED, see "Troubleshooting in Simulink Real-Time".

The next task is "Create and Run Real-Time Application from Simulink Model" on page 4-23.

Create and Run Simulink Model

The example model is a non-real-time model of a damped oscillator. It includes a `Transfer Fcn` block and a `Signal Generator` block. To visualize and log signals while simulating, it also includes standard Simulink `Scope` and `Out1` (`Outport`) blocks.

The final model is ex_slrt_nrt_osc (matlab: `open_system(docpath(fullfile(docroot, 'toolbox', 'xpc', 'examples', 'ex_slrt_nrt_osc'))))`:

Tutorial model
with Simulink scope
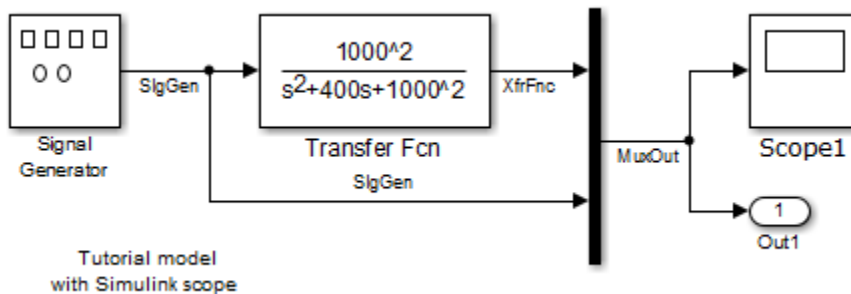
Create Simulink Model

The initial model is a non-real-time model of a damped oscillator, ex_slrt_osc (matlab: `open_system(docpath(fullfile(docroot, 'toolbox', 'xpc', 'examples', 'ex_slrt_osc'))))`.

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 and an Out1 (Outport) block to the model.
- Select **Simulink** > **Signal Routing**, and then add a Mux block to the model.

**5** Make the following block-to-block connections:

- Signal Generator output to Transfer Fcn input.
- Signal Generator output to Mux input.
- Transfer Fcn output to Mux input.
- Mux output to Scope input.
- Mux output to Out1 input.

6 As required, double-click signals and enter names.

If you provide a name for a signal in the `Signal name` property of the Signal Properties dialog box, that name appears in the target computer scope graph after you build and download the model to the target computer. By default, if you do not enter a name for the signal in this dialog box, the scope graph displays the signal identifier rather than a name.

7 From the **File** menu, click **Save As** and enter a file name. For example, enter `ex_slrt_osc` and then click **OK**.

The unconfigured model is `ex_slrt_osc` (matlab: open_system(docpath(fullfile(docroot, 'toolbox', 'xpc', 'examples', 'ex_slrt_osc')))).

Tutorial model
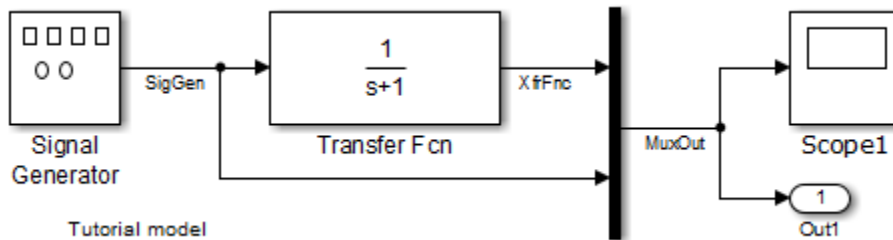
Configure Signal Generator

1   In the Command Window, open `ex_slrt_osc`.

     MATLAB loads the oscillator model and displays the Simulink block diagram.

2   In Simulink Editor, double-click the Signal Generator block.

3   In the Block Parameters dialog box, from the **Wave form** list, select `square`.

4   In the **Amplitude** text box, enter `1`.

5   In the **Frequency** text box, enter `20`.

6   From the **Units** list, select `Hertz`.
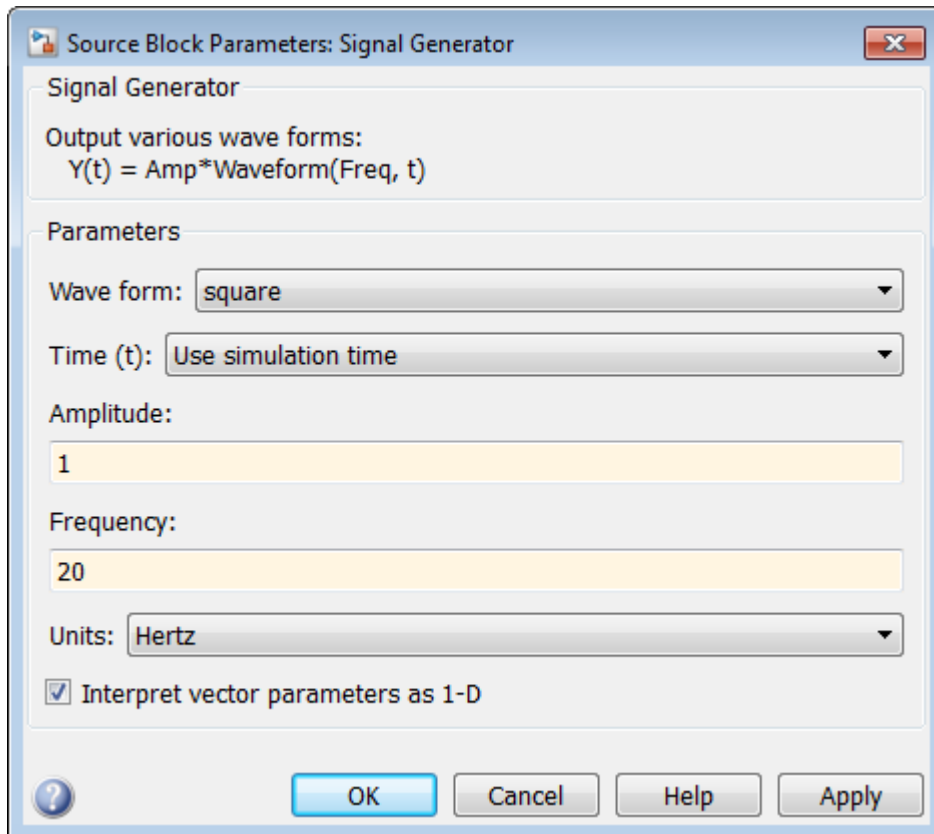
7   Do one of the following:

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

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

8   From the **File** menu, click **Save As** and enter a file name.

For example, enter `ex_slrt_nrt_osc` (matlab:
open_system(docpath(fullfile(docroot, 'toolbox', 'xpc',
'examples', 'ex_slrt_nrt_osc')))).

Click **OK**.

Configure Transfer Function

This procedure uses example `ex_slrt_nrt_osc` (`matlab:`
`open_system(docpath(fullfile(docroot, 'toolbox', 'xpc', 'examples',`
`'ex_slrt_nrt_osc'))))`.

1 In the Command Window, open `ex_slrt_nrt_osc`.

   MATLAB loads the oscillator model and displays the Simulink block diagram.

2 In Simulink Editor, double-click the Transfer Fcn block. The Block Parameters
   dialog box opens.

3 In the **Numerator** text box, enter `[1000^2]`.

4 In the **Denominator** text box, enter `[1 400 1000^2]`.

Function Block Parameters: Transfer Fcn

Transfer Fcn

The numerator coefficient can be a vector or matrix expression. The denominator coefficient must be a vector. The output width equals the number of rows in the numerator coefficient. You should specify the coefficients in descending order of powers of s.

Parameters

Numerator coefficients:

[1000^2]

Denominator coefficients:

[1 400 1000^2]

Absolute tolerance:

auto

State Name: (e.g., 'position')

''

OK   Cancel   Help   Apply

5 Do one of the following:

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

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

6 From the **File** menu, click **Save**.

Configure Scope Block

The example model is a non-real-time model of a damped oscillator, `ex_slrt_nrt_osc` (matlab: `open_system(docpath(fullfile(docro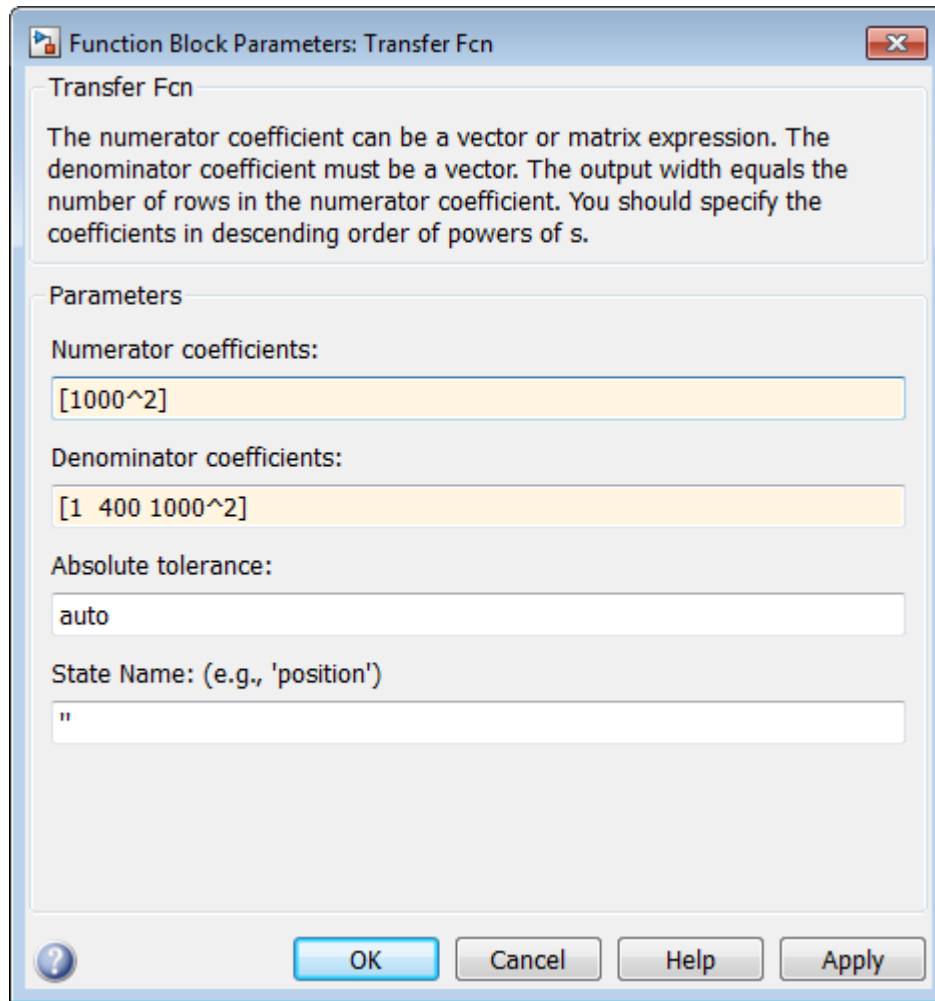ot, 'toolbox', 'xpc', 'examples', 'ex_slrt_nrt_osc'))))`. You enter or change scope parameters to specify the *x*-axis and *y*-axis in a Scope window. Other properties include the number of graphs in one Scope window and the sample time for models with discrete blocks.

For more information, see "Trace Signals with Simulink External Mode".

1　In the Command Window, type `ex_slrt_nrt_osc`.

　　MATLAB loads the oscillator model and displays the Simulink block diagram.

2　In Simulink Editor, double-click the Scope block.

3
　　On the toolbar, click the **Parameters** button ⚙ .

4　In the Scope dialog box, 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 the model has discrete blocks, enter the **Fixed step size** you entered in the Configuration Parameters dialog box.

　　Your **Main** tab looks like this figure.
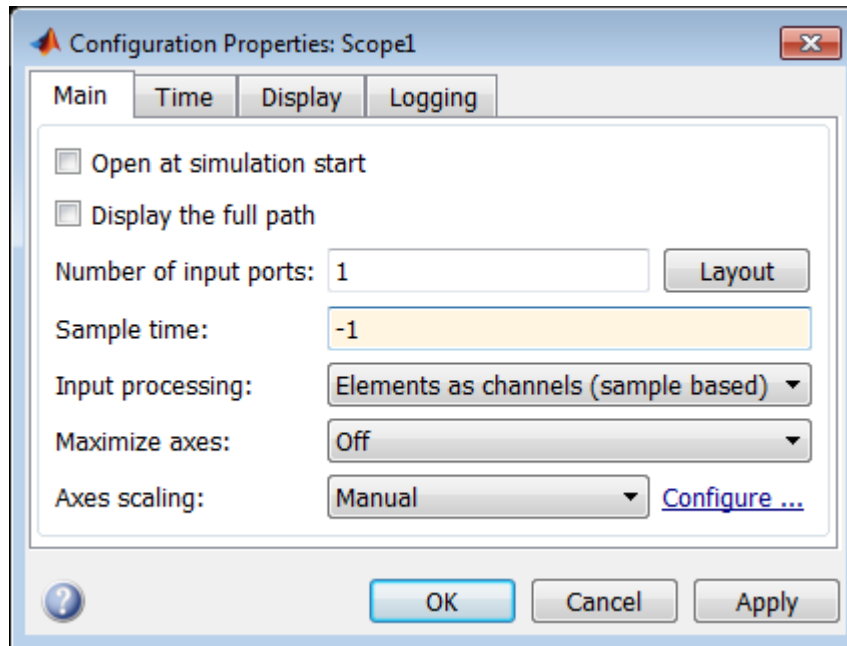
5 In the Scope dialog box, click the **Time** tab. In the **Time span** box, enter the upper value for the time range. For example, enter `0.2` second.

Your **Time** tab looks like this figure.
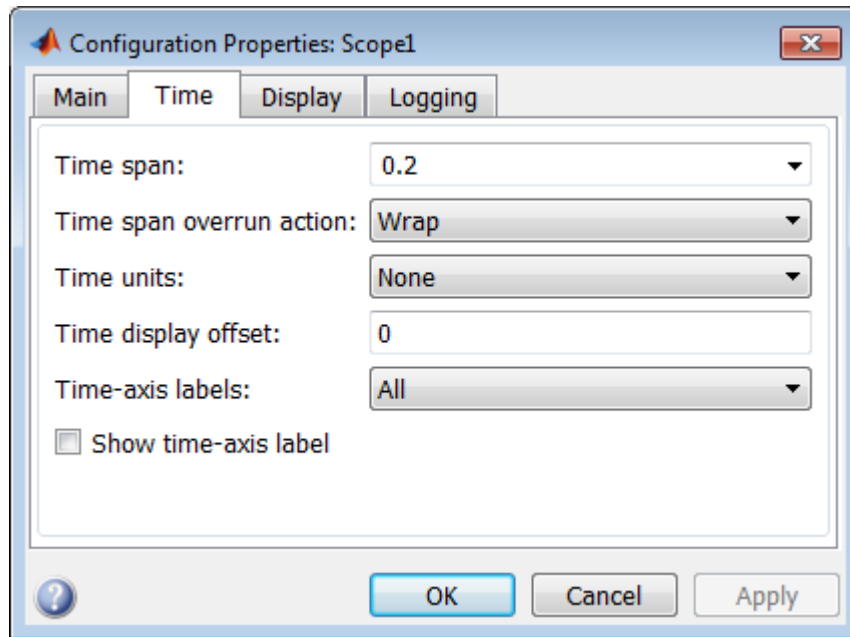
6 In the Scope dialog box, click the **Display** tab. In the **Y-min** and **Y-max** text boxes, enter the range for the *y*-axis in the Scope window. For example, enter -3 and 3.
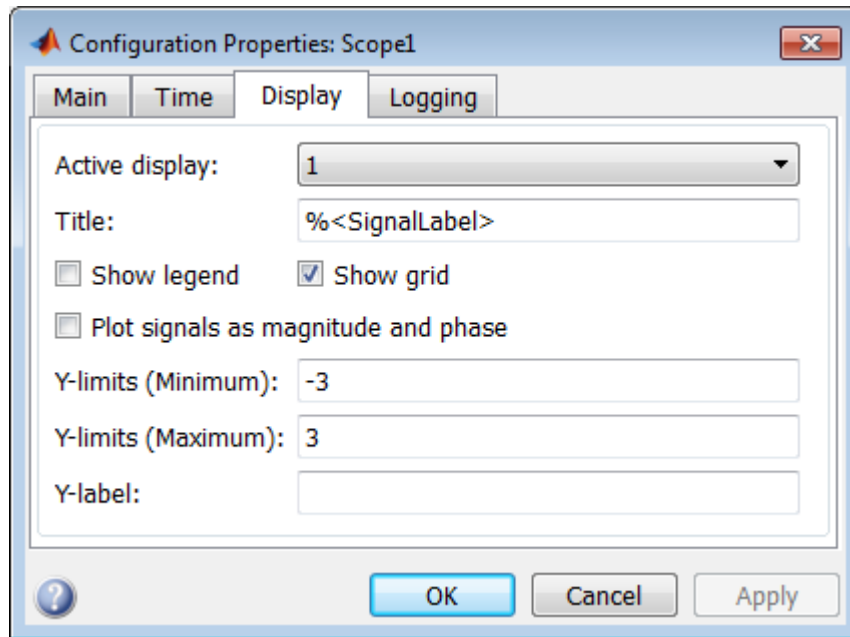
Your **Display** tab looks like this figure.

7   Do one of the following:

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

8   From the **File** menu, click **Save**.

Configure Simulation Parameters

The example model is a non-real-time model of a damped oscillator, `ex_slrt_nrt_osc` (matlab: `open_system(docpath(fullfile(docroot, 'toolbox', 'xpc', 'examples', 'ex_slrt_nrt_osc')))`). During a simulation, Simulink saves signal data to MATLAB variables using `Outport` blocks. The default MATLAB variables are `tout`, `xout`, and `yout`. While running a real-time application, the Simulink Real-Time interface uses these same variables to pass signal data to target object parameters. A target object is a structure in the MATLAB workspace that the Simulink Real-Time software uses to interact with a real-time application. The default target object is `tg`. The default parameters are `Time`, `tg.States`, and `tg.Output`.

1   In the Command Window, type `ex_slrt_nrt_osc`.

    MATLAB loads the oscillator model and displays the Simulink block diagram.

2   In Simulink Editor, click **Simulation** > **Model Configuration Parameters**.

3   In the Configuration Parameters dialog box, click **Solver**.

    Simulink displays the **Solver** pane.

4   In the **Simulation time** section, enter 0 seconds in the **Start time** box. In the
    Stop time box, enter an initial stop time. For example, enter 1 second. To change
    this time after creating the real-time application, change the target object property
    `tg.Stoptime` to the new time using the MATLAB command-line interface. To
    specify an infinite stop time, enter `inf`.

5   From the **Type** list, select `Fixed-step`. Simulink Coder does not support variable-
    step solvers.

6   From the **Solver** list, select a solver. For example, select the general-purpose solver
    `ode4 (Runge-Kutta)`.

7   Under **Additional Options**, in the **Fixed step size** box, enter the sample time for
    the real-time application. For example, enter `0.00025` seconds (`250` microseconds).
    You can change this value after creating the real-time application.

    If you find that `0.000250` seconds results in overloading the CPU on the target
    computer, try a larger **Fixed step size**, such as `0.0002` seconds.

    If the model contains both continuous and discrete states, the sample times of the
    discrete states can only be multiples of **Fixed step size**. If the model contains no
    continuous states, enter `'auto'` in **Fixed step size**. The sample time is taken from
    the model.

    The **Solver** pane looks like this figure.
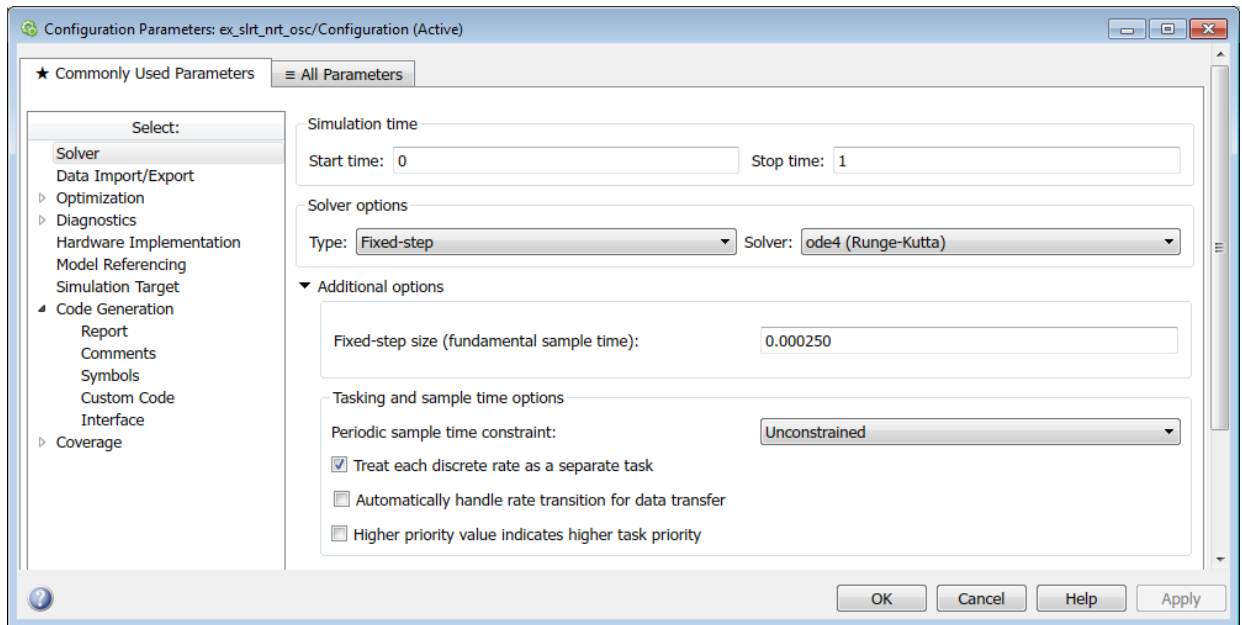
8 Click **Data Import/Export**.

The **Data Import/Export** pane opens. This pane defines the model signals logged during a simulation of the model or while running the real-time application.

9 Select the **Time**, **States**, and **Output** check boxes.

When the real-time application is running in `Freerun` mode, data is saved to the variables `tout` and `yout`. When the real-time application is running in `Real-Time` mode, data is saved to the target object properties `TimeLog`, `StateLog`, and `OutLog`. However, you must still select the **Time**, **States**, and **Output** check boxes for data to be logged to the target object properties.

You can improve runtime performance by turning one or more of the check boxes off. See "Improve Run-Time Performance".

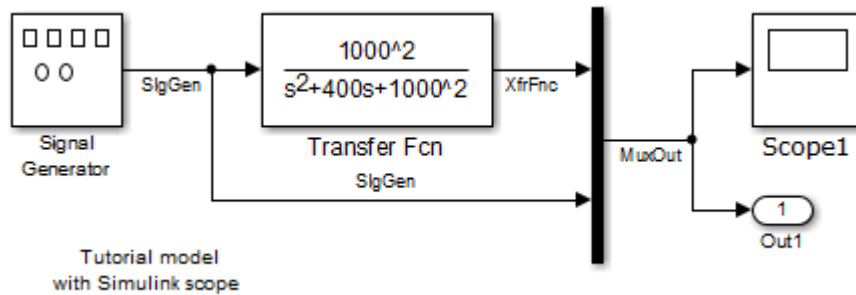The **Data Import/Export** pane looks like this figure.
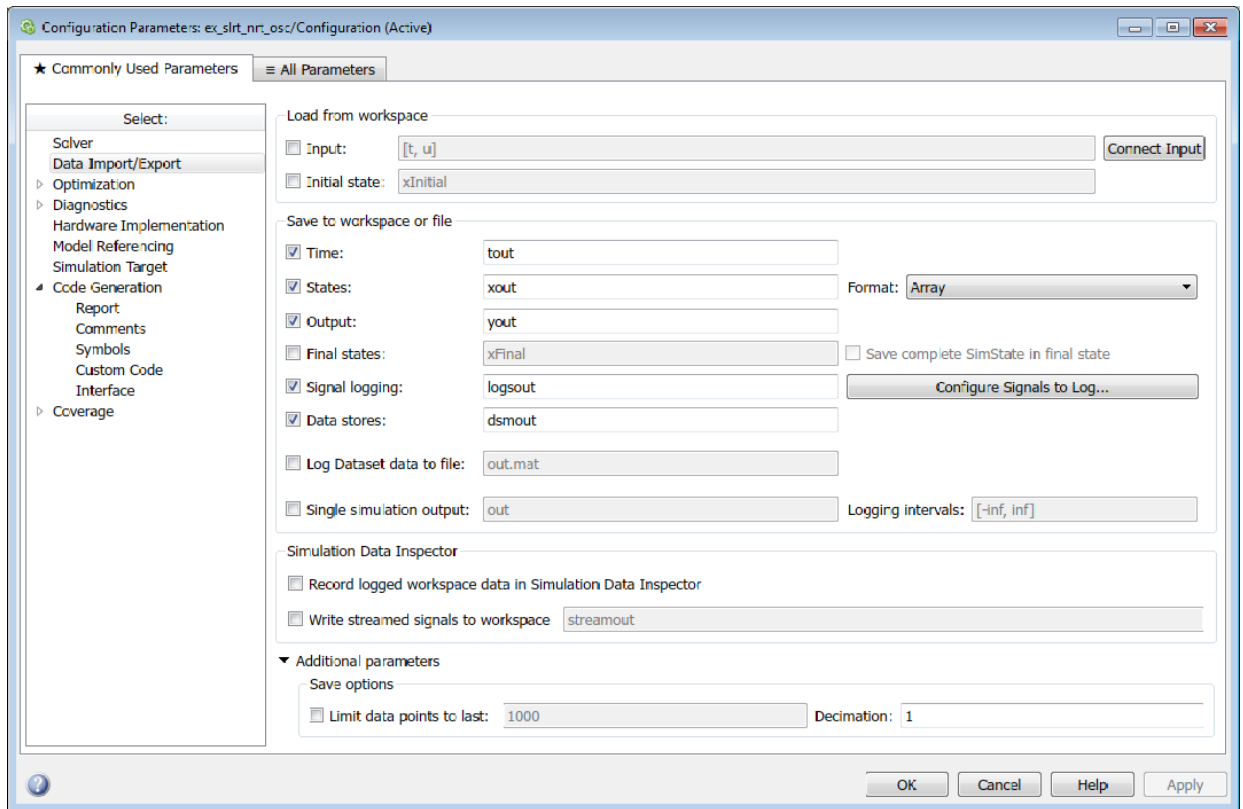
10 Do one of the following:

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

11 From the **File** menu, click **Save**.

The final model looks like this figure.

$$\frac{1000^2}{s^2+400s+1000^2}$$

Tutorial model
with Simulink scope

Simulate with Simulink

The example model is a non-real-time model of a damped oscillator, `ex_slrt_nrt_osc`
(`matlab: open_system(docpath(fullfile(docroot, 'toolbox', 'xpc',`
`'examples', 'ex_slrt_nrt_osc'))))`). You use Simulink in normal mode to observe
the non-real-time behavior of the model. After you load the Simulink model, you can run
a simulation.

1   In the Command Window, type `ex_slrt_nrt_osc`.

    MATLAB loads the oscillator model and displays the Simulink block diagram.

2   In Simulink Editor, double-click the Scope block.

3   In the Scope window, select **Simulation** > **Mode** > **Normal**.

4   On the toolbar, click the **Run** button ▶ to begin simulation.

    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.
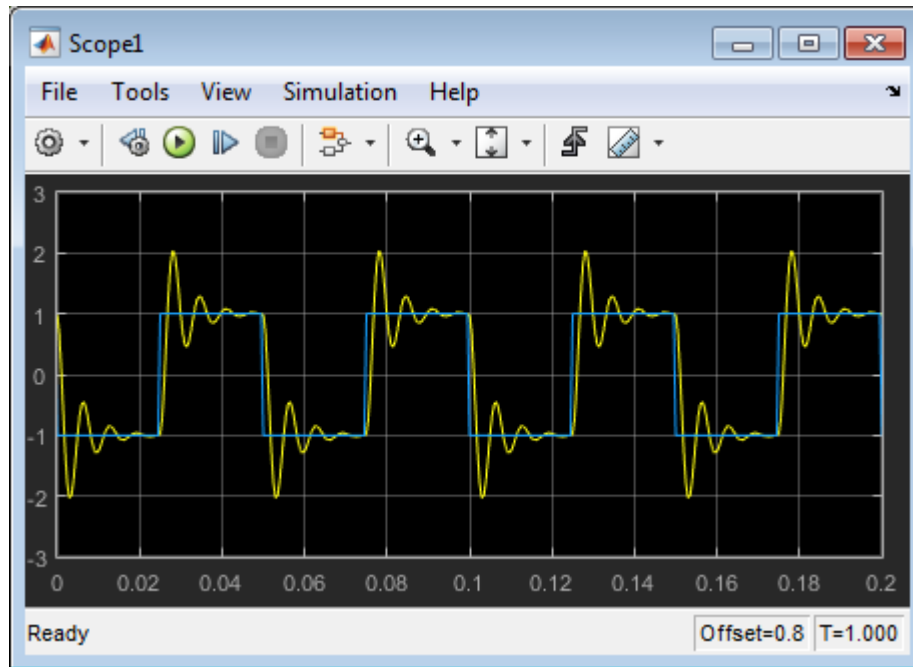
5   You can either let the simulation run to its stop time, or stop the simulation
    manually. To stop the simulation manually, on the toolbar, click the **Stop** button
    .

The next task is "Create and Run Real-Time Application from Simulink Model" on page
4-23.

See Also

matlab: open_system(docpath(fullfile(docroot, 'toolbox', 'xpc', 'examples', 'ex_slrt_osc')))
| matlab: open_system(docpath(fullfile(docroot, 'toolbox', 'xpc', 'examples',
'ex_slrt_nrt_osc'))) | Mux | Outport | Scope | Signal Generator | Transfer Fcn

Create and Run Real-Time Application from Simulink Model

The example model is a non-real-time model of a damped oscillator, `ex_slrt_nrt_osc` (`matlab: open_system(docpath(fullfile(docroot, 'toolbox', 'xpc', 'examples', 'ex_slrt_nrt_osc')))`). This tutorial describes how to transform `ex_slrt_nrt_osc` into a Simulink Real-Time model configured to build as a real-time application. It describes how to build, download, and execute the real-time application on the target computer.

The final model is `ex_slrt_rt_osc` (`matlab: open_system(docpath(fullfile(docroot, 'toolbox', 'xpc', 'examples', 'ex_slrt_rt_osc')))`):
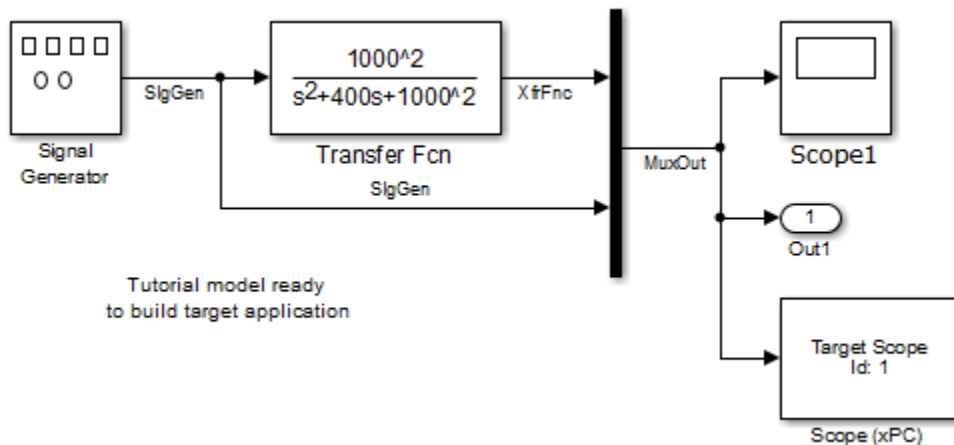
In this section...

Transform Simulink Model to Real-Time Application

To run a Simulink model as a real-time application under Simulink Real-Time, you must add and configure the Simulink Real-Time Scope block and set configuration parameters for code generation and target execution.

Add Simulink Real-Time Scope Block

Simulink Real-Time supports a real-time Scope block. There are three types of Simulink Real-Time scopes: `target`, `host`, and `file`. This tutorial uses the `target` scope, which displays execution data on the target computer monitor.

1 In the Command Window, type `ex_slrt_nrt_osc`.

   MATLAB loads the oscillator model and displays the Simulink block diagram.

2 In Simulink Editor, from the **View** menu, click **Library Browser**.

   The Simulink Library opens.

3 In the left pane, browse to and double-click node **Simulink Real-Time**.

   A list of I/O block categories opens.

4 Click node **Displays and Logging**.

5 Click and drag block Scope to the Simulink block diagram.

   Simulink adds a new real-time Scope block to the model with a scope identifier of `1`.

6 Connect the Mux output to the Scope block input.

7 From the **File** menu, click **Save As**. Enter a file name. For example, enter `ex_slrt_ucf_osc` and then click **OK**.

   The unconfigured model is `ex_slrt_ucf_osc` (matlab: open_system(docpath(fullfile(docroot, 'toolbox', 'xpc', 'examples', 'ex_slrt_ucf_osc')))).

$$\frac{1000^2}{s^2+400s+1000^2}$$

Tutorial model
with target scope
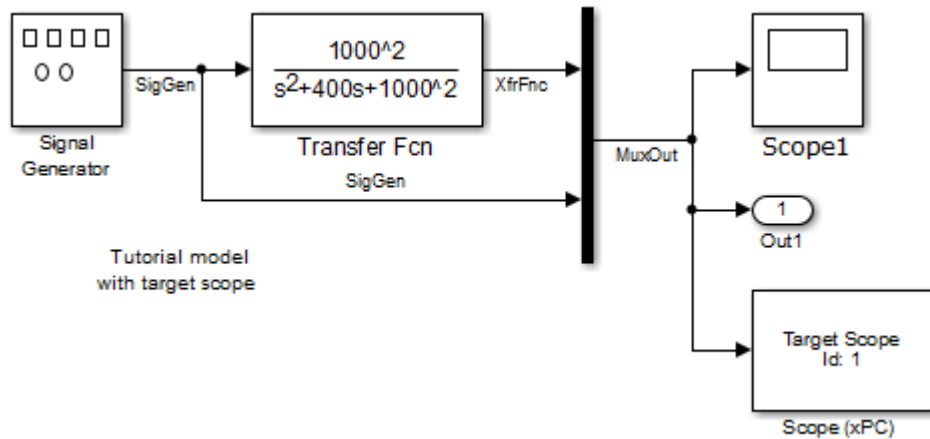
Set Target Scope Block Parameters

Scope block parameters define the signals to trace on the scope and trigger modes. The Simulink Real-Time Scope block dialog box changes depending on which scope type you are configuring: `target`, `host`, or `file`. For this tutorial, configure a `target` scope.

For more information about real-time `target` Scope parameters, see "Configure Real-Time Target Scope Blocks".

1 In the Command Window, type `ex_slrt_ucf_osc`.

MATLAB loads the oscillator model and displays the Simulink block diagram.

2 Double-click block Scope.

3 Select **Scope type** `Target`. This value means that the scope display appears on the target computer monitor.

4 In the block parameters dialog box, select the **Start scope when application starts** check box.

This setting is mandatory in `Stand Alone` mode because the development computer is not available to issue a command to start scopes.

5 Select **Scope mode** `Graphical redraw`.

6 Select the **Grid** check box.

7 Type [0,0] in the **Y-axis limits** text box. This value means that display scaling is auto.

8 Type 1000 in the **Number of samples** text box. For a **Scope mode** of Graphical redraw, this value means that 1000 samples are acquired before the graph is redrawn.

9 Type 0 in the **Number of pre/post samples** text box. This value means that samples are not saved before a trigger or skipped after a trigger.

10 Type 1 in the **Decimation** text box. This value means that data is collected at each sample interval.

11 Select **Trigger mode** FreeRun. This value means that the trigger event is automatic and no external trigger specification is required.

The target scope dialog box looks like this figure.

Sink Block Parameters: Scope (xPC)

xpcscopeblock (mask) (link)

Simulink Real-Time Scope
Configure scope to acquire signal data.
Scope can be of type target, host, or file.

Parameters

Scope number:

1

Scope type: Target

☑ Start scope when application starts

Scope mode: Graphical redraw

☑ Grid

Y-axis limits:

[0,0]

Number of samples:

1000

Number of pre/post samples:

0

Decimation:

1

Trigger mode: FreeRun

OK   Cancel   Help   Apply

12 Do one of the following:

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

**13** From the **File** menu, click **Save**.

Set Configuration Parameters

The example model, ex_slrt_ucf_osc, is a non-real-time model of a damped oscillator. You enter the simulation and real-time run parameters in the Configuration Parameters dialog box. These parameters give information to Simulink Coder on how to build the real-time application from the Simulink model.

After you open a Simulink model and start the target computer, you can enter the simulation parameters.

1 In the Command Window, type ex_slrt_ucf_osc.

2 In Simulink Editor, click **Simulation** > **Model Configuration Parameters**.

3 In the Configuration Parameters dialog box, click the **Solver** node.

4 In the **Solver options** section, from the **Type** list, select Fixed-step.

5 From the **Solver** list, select a solver. For example, select the general-purpose solver ode4 (Runge-Kutta).

6 Under **Additional Options**, in the **Fixed-step size (fundamental sample time)** box, enter the sample time for the real-time application. For example, enter 0.00025 seconds (250 microseconds). You can change this value after creating the real-time application.

If you find that 0.000250 seconds results in overloading the CPU on the target computer, try a larger **Fixed-step size** value, such as 0.0002 seconds.

The sample times of the model blocks can only be multiples of **Fixed-step size**. If you enter 'auto' in **Fixed-step size**, the fundamental sample time is calculated from the sample times of the model blocks.
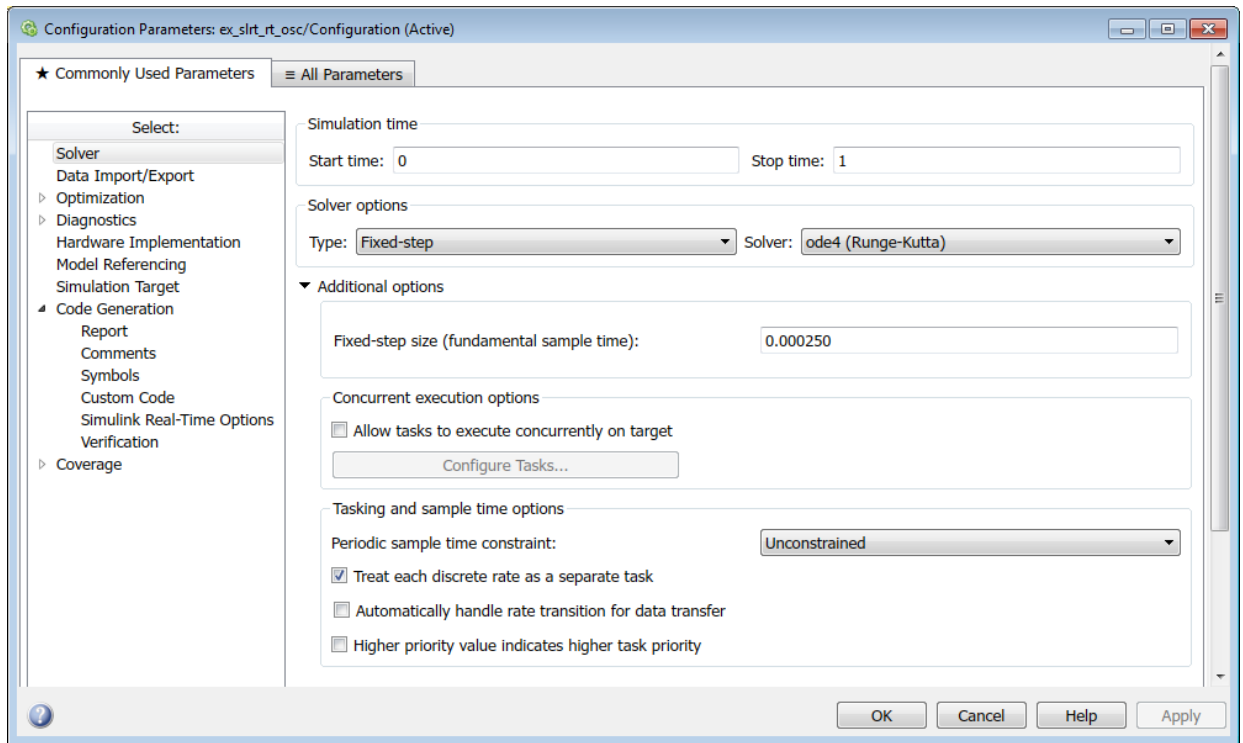
7 In the Configuration Parameters dialog box, click the **Code Generation** node.

The code generation pane opens.

8 To build a basic real-time application, in the **Target selection** section, click **Browse** at the **System target file** list. Click `slrt.tlc`, and then click **OK**.

The system target file `slrt.tlc`, the template makefile `xpc_default_tmf`, and the make command `make_rtw` are automatically entered into the page. The **Simulink Real-Time Options** node appears in the left pane.
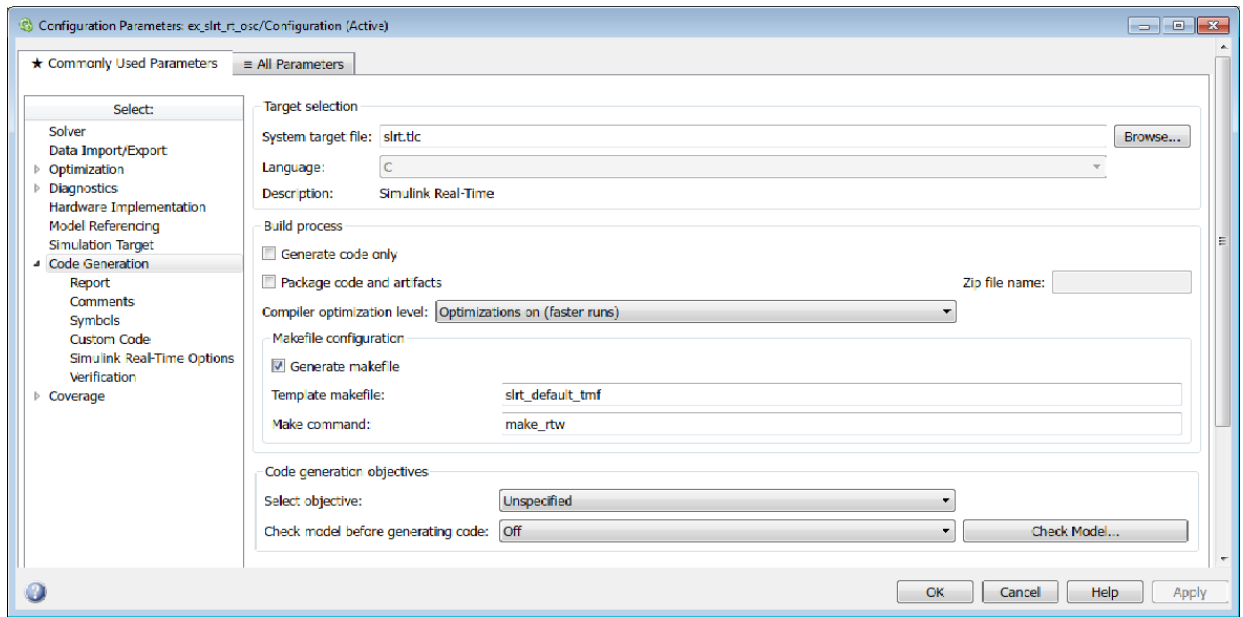
9   Click **OK**.

10   From the **File** menu, click **Save As**.

Enter a file name. For example, enter `ex_slrt_rt_osc` (matlab:
open_system(docpath(fullfile(docroot, 'toolbox', 'xpc',
'examples', 'ex_slrt_rt_osc')))).
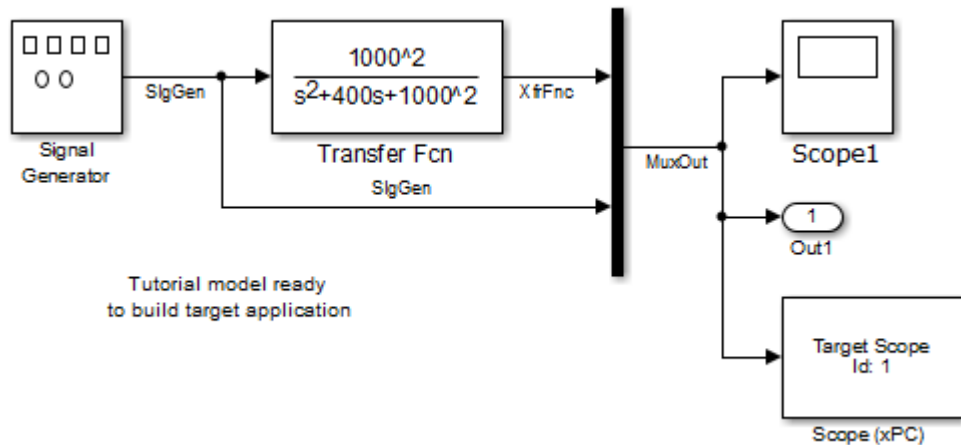
Start Target Computer

Starting the target computer loads and starts the Simulink Real-Time kernel on the target computer. The loader then waits for the Simulink Real-Time software to download the real-time application from the development computer.

After you have configured the Simulink Real-Time product using the Simulink Real-Time Explorer and created a target boot disk for that setup, you can start the target computer. Before building the real-time application, you must start the target computer because the build process automatically downloads the real-time application to the target computer. Be sure that you have followed the instructions from "System Configuration".

1 Insert the target boot disk into the target computer disk drive.
2 Turn on the target computer or press the **Reset** button.

   The target computer displays a screen like this screen.
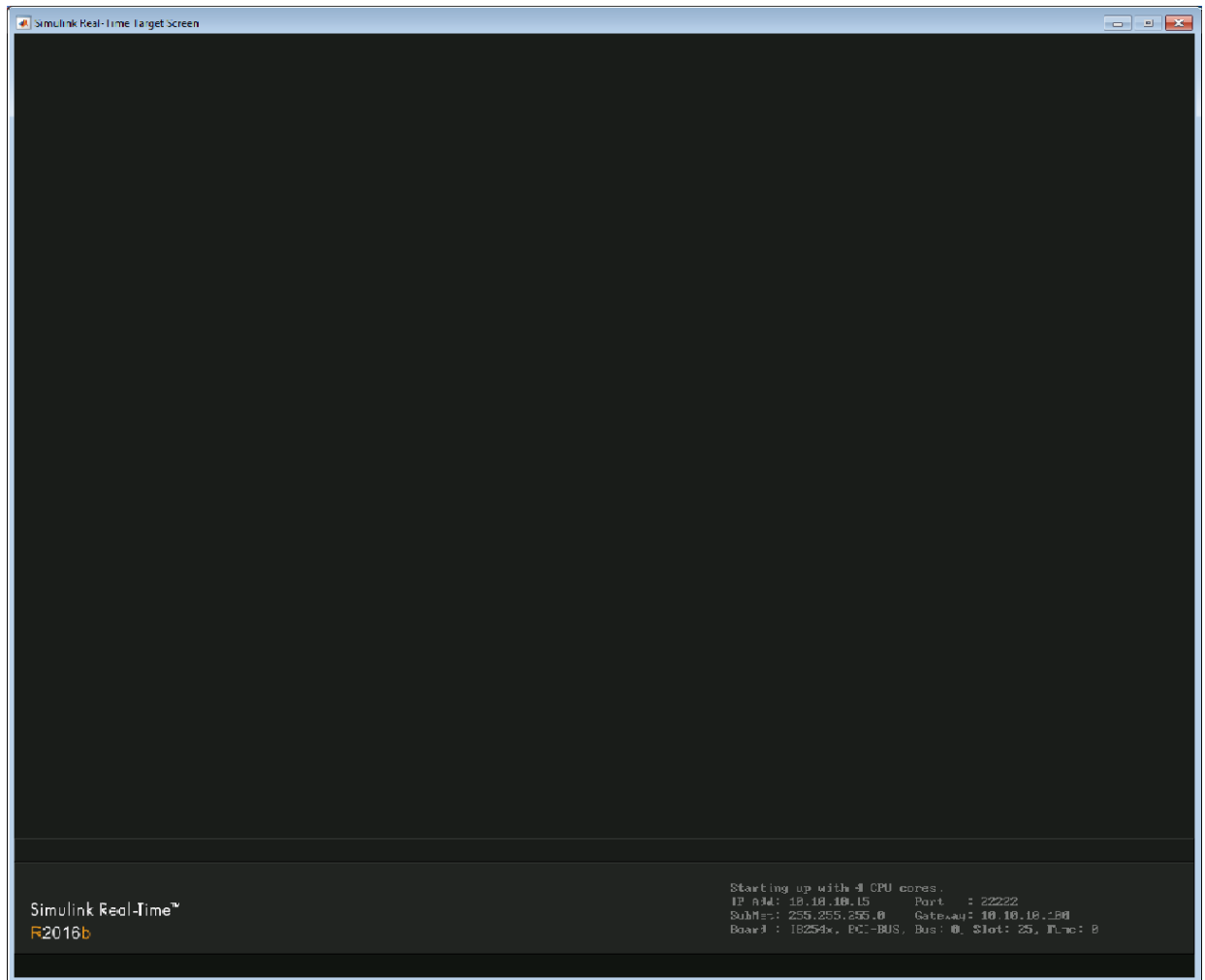
If you have a keyboard attached to the target computer, you can activate that keyboard by typing **C**. Press the **Page Up** and **Page Down** keys to page up and down in the target computer monitor.

The status window shows that the kernel is in loader mode and waiting to load a real-time application. The memory value is the number of bytes of target computer memory available for the heap, for running scopes, and for data acquisition buffers.

Note: Target computer memory for the real-time application executable, the kernel, and other uses is limited to a maximum of 3 GB.

Build and Download Real-Time Application

The example model is a real-time model of a damped oscillator, `ex_slrt_rt_osc` (matlab: `open_system(docpath(fullfile(docroot, 'toolbox', 'xpc', 'examples', 'ex_slrt_rt_osc'))))`. You use the Simulink Real-Time build process to generate C code, compile, link, and download the real-time application to the target computer.

After you enter changes in the Configuration Parameters dialog box, you can build the real-time application. By default, the build procedure downloads the real-time application to the default target computer, as designated in Simulink Real-Time Explorer. For further details on setting the target computer for a real-time application, see "Code Generation Configuration Parameters".

1 In the Command Window, type `ex_slrt_rt_osc`.

   MATLAB loads the oscillator model and displays the Simulink block diagram.

2 In Simulink Editor, on the toolbar, click the **Build Model** button ▦ on the toolbar.

   On the development computer, after completing a build without detecting an error, MATLAB displays lines like the following:

   ```
   ### Starting Simulink Real-Time build procedure for model:
   ex_slrt_rt_osc
   . . .
   ### Successful completion of Simulink Real-Time build procedure for
   model: ex_slrt_rt_osc
   ```

   After compiling, linking, and downloading the real-time application, Simulink Real-Time creates a target object in the MATLAB workspace. The default name of the target object is `tg`. For more information about the target object, see "Real-Time Application Objects".

If you have a monitor connected to your target computer, the monitor screen looks like this screen.
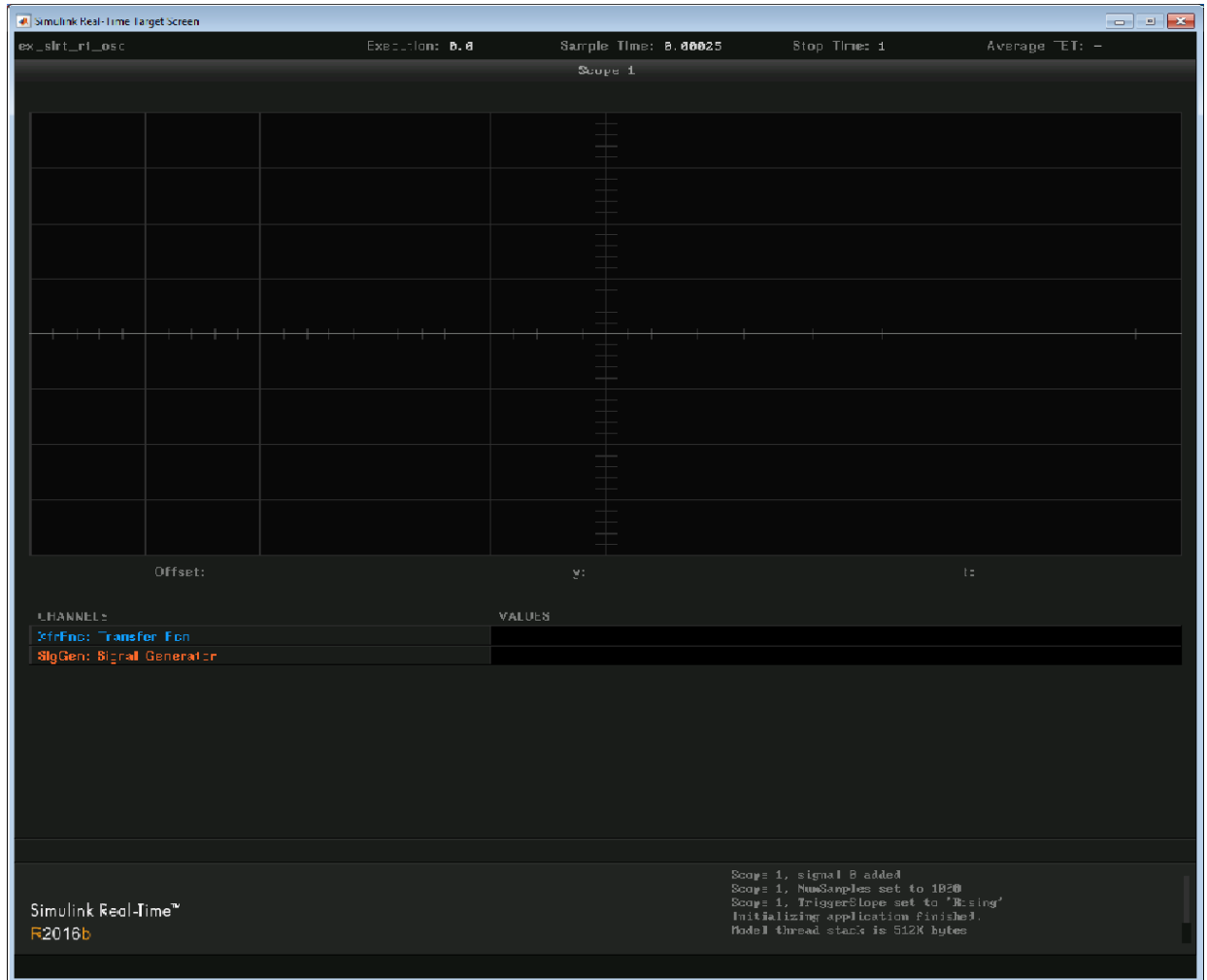
3 In the Command Window, type:

```
tg
```

MATLAB displays a list of properties for the target object `tg`.

If the software detects a error during build and download, see "Troubleshooting in Simulink Real-Time".

If you accidentally download a real-time application built with a different version of the Simulink Real-Time product than the one on the target computer, the following error message appears on the target computer monitor. The download fails.

```
Mismatch between model and kernel versions
```

To prevent this version mismatch, rebuild real-time applications with each new Simulink Real-Time release.

During the build process, the Simulink Real-Time software creates a target object that represents the real-time application running on the target computer. The target object is defined by a set of properties and associated methods. You control the real-time application and computer by setting the target object properties.

Execute Real-Time Application with Simulink External Mode

Control of the real-time application with Simulink is limited to connecting a Simulink model to a real-time application through external mode, and then starting the real-time application. Using Simulink external mode is one method to tune parameters. By default, the model connects to the default target computer, as specified in Simulink Real-Time Explorer.

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

After you build and download a real-time application to the target computer, you can run the real-time application. This procedure uses the Simulink model `ex_slrt_rt_osc` (matlab: `open_system(docpath(fullfile(docroot, 'toolbox', 'xpc', 'examples', 'ex_slrt_rt_osc'))))`. See "Build and Download Real-Time Application" on page 4-33. You must already have specified the required target computer environment on the **Simulink Real-Time Options** node of the Simulink Real-Time parameters dialog box. In particular, you must specify the target computer to which you want to connect. See the **Build for default target computer** check box description in "Code Generation Configuration Parameters".

1   Select **Simulation** > **Mode** > **External**.

A check mark appears next to the menu item **External**. Simulink external mode is activated. Simulink external mode connects the Simulink model to the real-time application as a simple graphical user interface.

2   On the toolbar, click the **Connect To Target** button ⊙.

The current Simulink model parameters are downloaded from the development computer to the real-time application.

3   On the toolbar, click the **Run** button ▶.

The real-time application begins running. The Simulink Scope block show output like the figure.
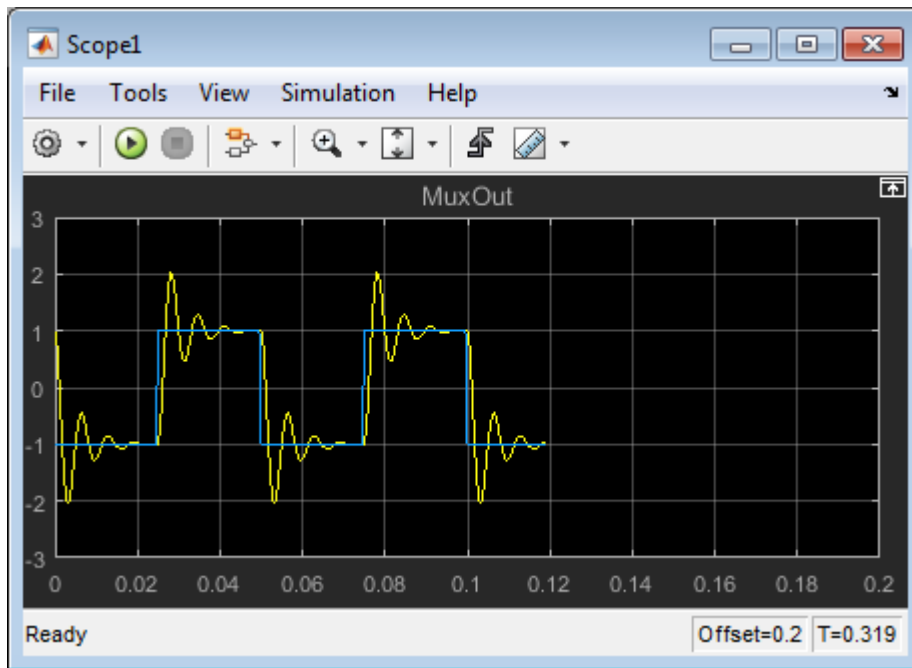
4   To stop execution, on the toolbar, click the **Stop** button ■.

See Also

Mux | Scope | Scope | SimulinkRealTime.target.viewTargetScreen

More About

- "Create and Run Simulink Model" on page 4-8

Configure and Control a Real-Time Application

You can configure and control a real-time application using the Simulink Real-Time Explorer as well as Simulink external mode. This tutorial focuses on interaction using Simulink Real-Time Explorer. The model is a real-time model of a damped oscillator, ex_slrt_rt_osc (matlab: open_system(docpath(fullfile(docroot, 'toolbox', 'xpc', 'examples', 'ex_slrt_rt_osc')))).

In this section...
"Execute Real-Time Application with Simulink Real-Time Explorer" on page 4-38
"Change Stop Time and Sample Time" on page 4-41

Execute Real-Time Application with Simulink Real-Time Explorer

To perform this procedure, you must have already created a Simulink Real-Time boot disk and started the target computer. See "Start Target Computer" on page 4-31). This procedure begins with the real-time application ex_slrt_rt_osc (matlab: open_system(docpath(fullfile(docroot, 'toolbox', 'xpc', 'examples', 'ex_slrt_rt_osc')))) already downloaded to the target computer. See "Build and Download Real-Time Application" on page 4-33.

1   In the Simulink Editor, click **Tools** > **Simulink Real-Time**.

2   In the **Targets** pane, click the target computer icon for which you have downloaded the real-time application and click the **Connect** button 🔧 on the toolbar.

The target computer icon turns to **Connected** ●.

3   In the **Applications** pane, select the real-time application.

4   To start execution, click the real-time application and click the **Start** button ▶ on the toolbar.

The application starts running. The dialog box looks like this figure.
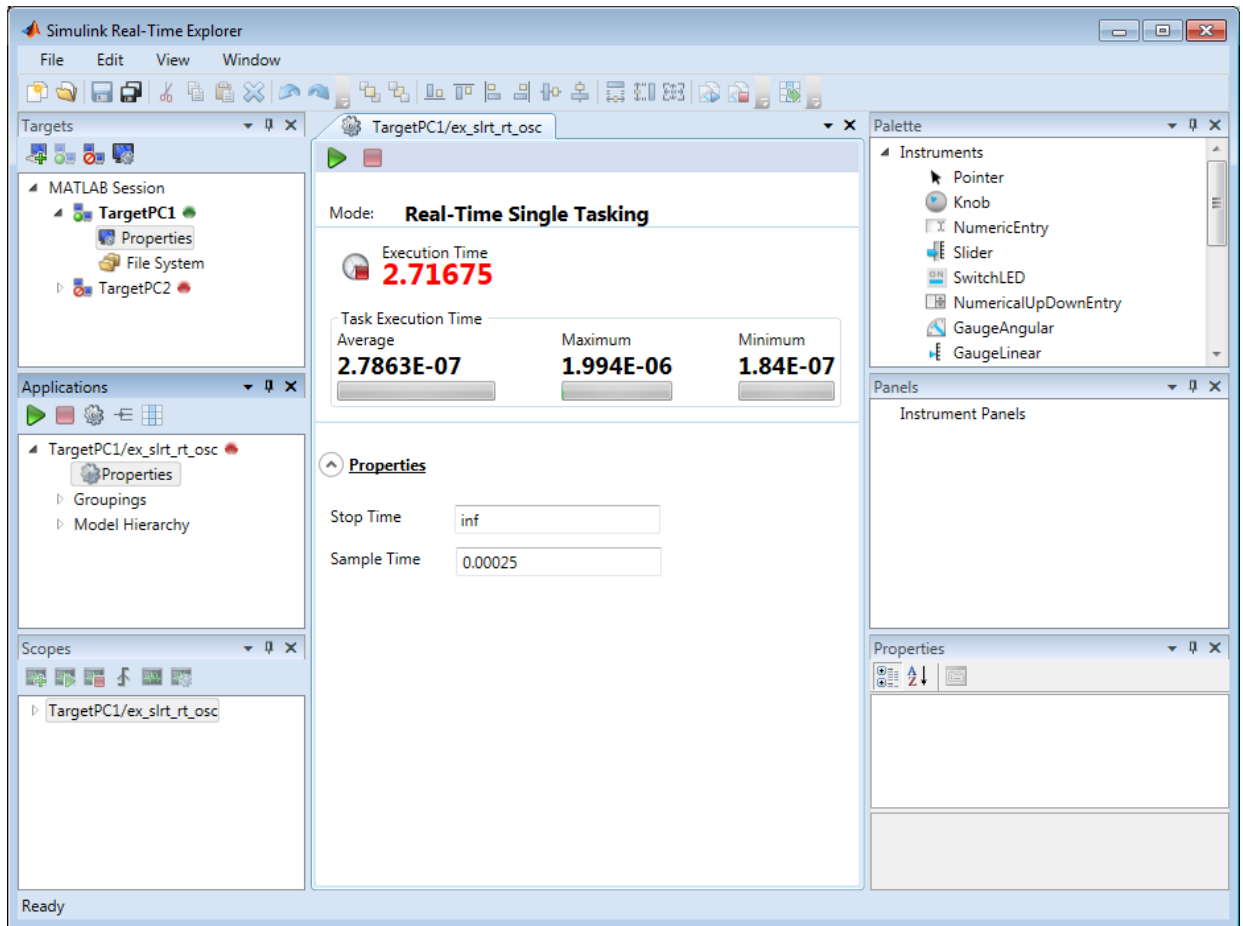
If you have a monitor connected to your target computer, you can view output that looks like this output.

5   To stop execution, click the real-time application and click the **Stop** button ■ on the toolbar.

The real-time application on the target computer stops running. The target computer displays messages like these messages.

```
minimal TET: 0.0000006 at time 0.001250
```

```
maximal TET: 0.0000013 at time 75.405500
```

Change Stop Time and Sample Time

This procedure is for changing the stop time and sample time in the real-time application configuration. You must have already downloaded the real-time application ex_slrt_rt_osc (matlab: open_system(docpath(fullfile(docroot, 'toolbox', 'xpc', 'examples', 'ex_slrt_rt_osc')))) to a target computer.

1   In Simulink Real-Time Explorer, expand the node of the loaded real-time application in the **Applications** pane.

2   On the toolbar, click the **Properties** button .

3   In the **Application Configuration** workspace, under the **Properties** arrow, enter a new value for **Stop time**. For example, enter inf and press **Enter**.

    The real-time application now runs until you stop it.

4   Enter a new value for **Sample Time**. For example, enter 0.00010 and press **Enter**.

Note: Some blocks produce incorrect results when you change their sample time at run time. If you include such blocks in your model, the software displays a warning message during model build. To avoid incorrect results, change the sample time in the original model, and then rebuild and download the model.

---

5   To start execution, click the real-time application and click the **Start** button on the toolbar.

6   To stop execution, click the real-time application and click the **Stop** button on the toolbar.

    The dialog box looks like this figure.
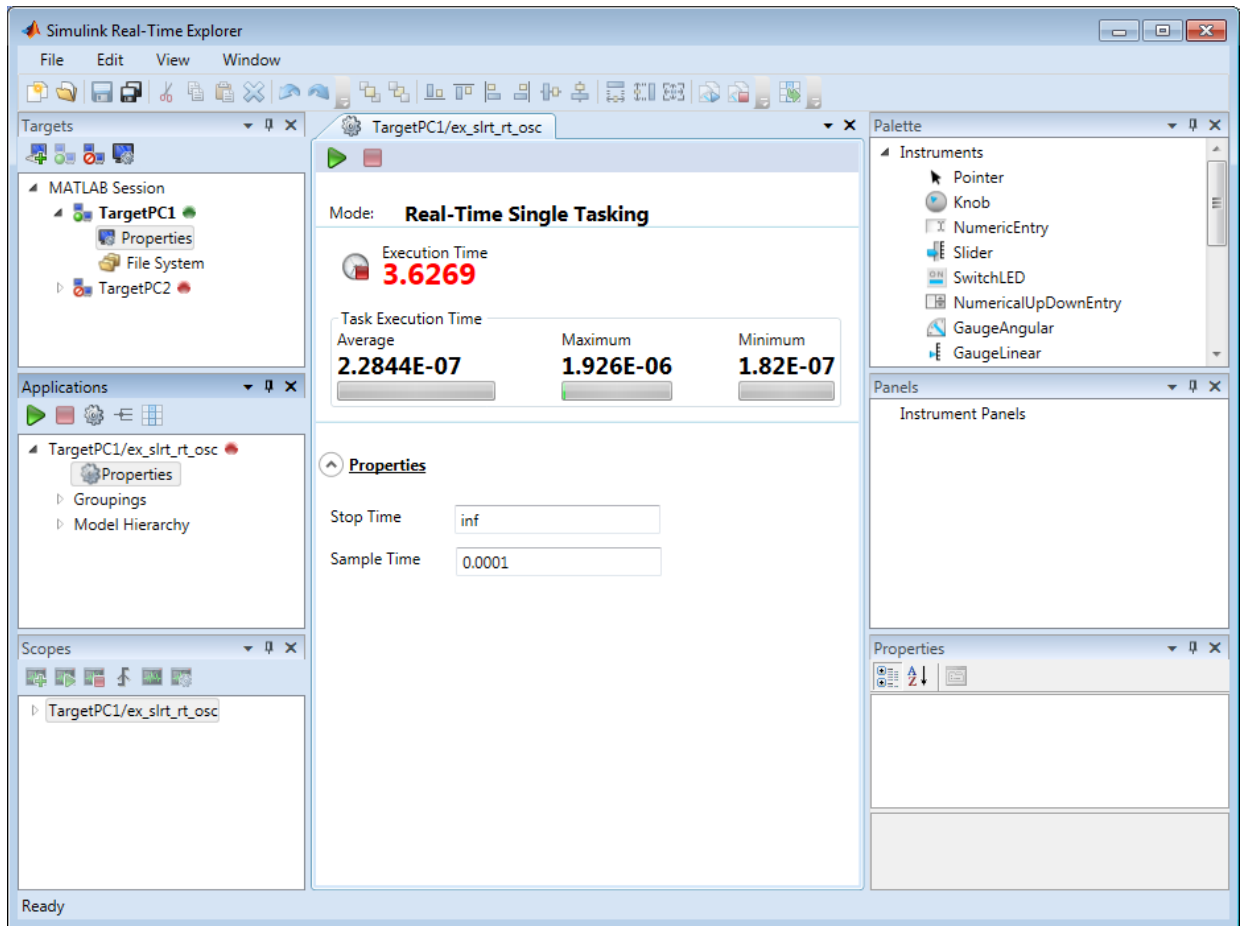
If you choose a sample time that is too small, a CPU overload can occur. If a CPU overload occurs, the target object property `CPUOverload` changes to `detected`. In that case, change **Fixed step size** to a larger value and rebuild the model.

See Also

`Scope` | `SimulinkRealTime.target.viewTargetScreen`

More About

- "Create Target Scopes with Simulink Real-Time Explorer"
- "Log Signal Data with Outport Block and Simulink Real-Time Explorer"
- "Blocks Whose Outputs Depend on Inherited Sample Time"

Configure a Real-Time Application for Embedded Coder

If you have Embedded Coder software, you can build an ERT real-time application. This example is based on the model ex_slrt_rt_osc (matlab: open_system(docpath(fullfile(docroot, 'toolbox', 'xpc', 'examples', 'ex_slrt_rt_osc')))).

- Open model ex_slrt_rt_osc, and then click **Simulation** > **Model Configuration Parameters**.

- In the Configuration Parameters dialog box, click the **Code Generation** node.

- To build an ERT real-time application, in the **Target selection** section, click **Browse** at the **System target file** list. Click slrtert.tlc, and then click **OK**.

  If you select slrtert.tlc without Embedded Coder installed, the build fails.

- In the left pane, click the **Simulink Real-Time Options** node.

  The associated pane is displayed. These are model-level configuration parameters that you can set for the model. For a description of the options on this node, see "Code Generation Configuration Parameters".

- From the **Execution mode** list, select either Real-Time or Freerun.

  Freerun mode execution is similar to a simulation, but uses the generated code. The target computer runs the real-time application as fast as possible, which may be faster than real-time execution. However, Freerun mode execution does not support variable-step solvers.

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

- From the **Real-time interrupt source** list, select a source. The default value is Timer.

- Select the **Log Task Execution Time** check box to log task execution times to the target object property tg.TETlog.

  The task execution time is the time in seconds to complete calculations for the model equations and post outputs during each sample interval. If you do not select this box, the average TET value appears as Not a Number (NaN).

- In the **Signal logging buffer size in doubles** box, enter the maximum number of sample points to save before wrapping, for example, 100000. This buffer includes the time, states, outputs, and task execution time logs.
- In the **Name of Simulink Real-Time object created by build process** box, enter the name of the target object created by the build process. The default target object name is `tg`.
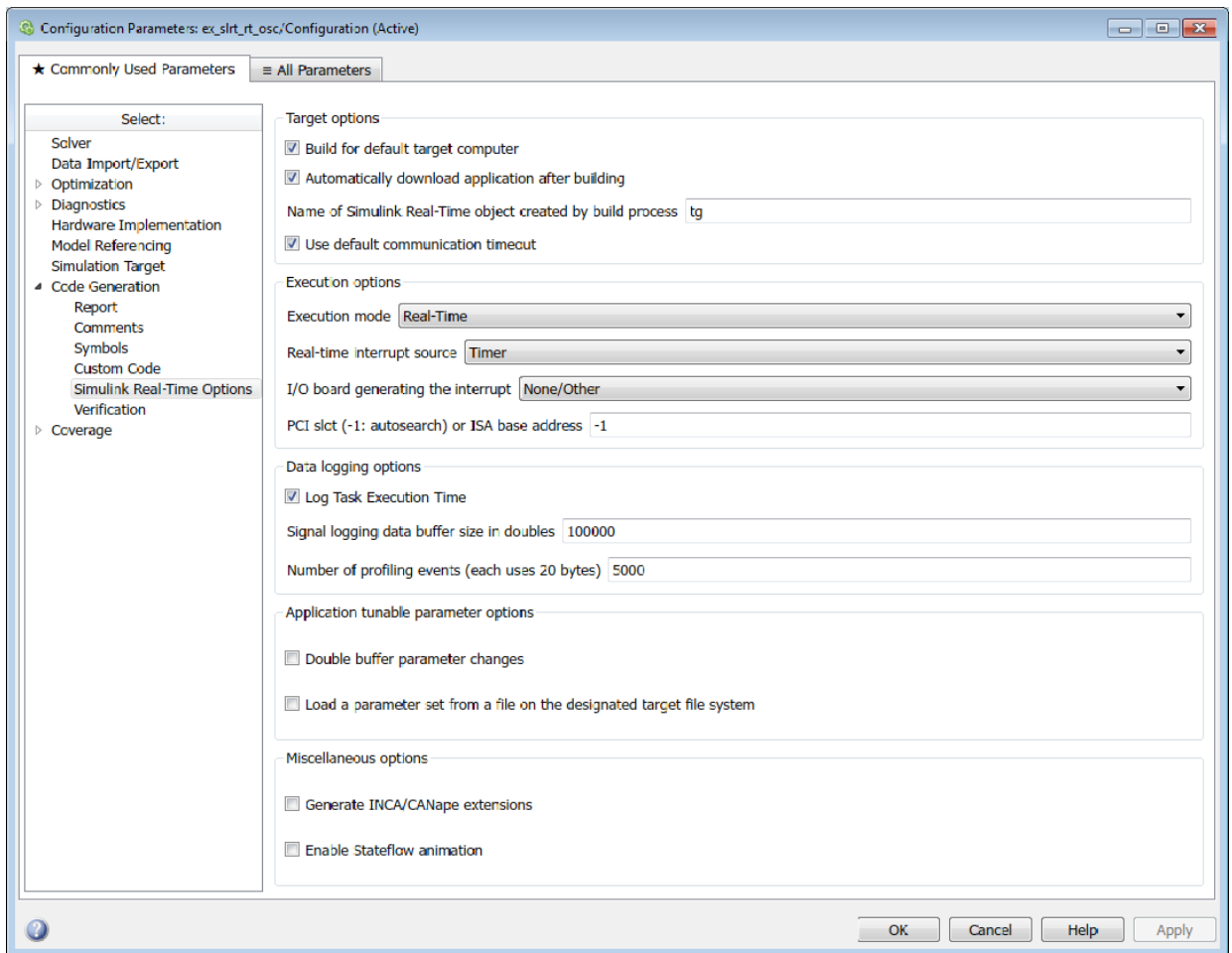
- Click **OK**.
- From the **File** menu, click **Save as**.

Enter a file name. For example, enter `ex_slrt_ert_osc` (`matlab:` `open_system(docpath(fullfile(docroot, 'toolbox', 'xpc', 'examples', 'ex_slrt_ert_osc'))))`,

Click **Save**.

Create a Prebuilt Real-Time Application

1   In Simulink Editor, click **Simulation** > **Model Configuration Parameters**.

2   Under the **Code Generation** node, select the **Simulink Real-Time Options** node.

3   Clear the **Automatically download application after building** check box.

4   From the **Code** menu, click **C/C++ Code** > **Build Model**.

Download a Prebuilt Real-Time Application

- In the Simulink Editor, click **Tools** > **Simulink Real-Time**.

- In the **Targets** pane, click the required target computer icon and click the **Connect** button on the toolbar.

  The target computer icon turns **Connected** .

- Do one of the following:

  - From the Command Window or from Microsoft Windows, drag the prebuilt real-time application DLM file to the target computer icon.

  - Right-click the required target computer icon and click **Load**. The **DLM Application Selector** dialog box opens. Enter or browse to the prebuilt application DLM file.

Simulate Simulink Model with MATLAB Language

Run a simulation of the Simulink model to observe the non-real-time behavior of the model.

After you load the Simulink model into the MATLAB workspace, you can run a simulation. This procedure uses the Simulink model ex_slrt_nrt_osc (matlab: open_system(docpath(fullfile(docroot, 'toolbox', 'xpc', 'examples', 'ex_slrt_nrt_osc')))). You must have loaded this model and entered the MATLAB variables tout and yout in the **Data I/O** pane in the Configuration Parameters dialog box.

1   In the Command Window, type:

```
output=sim('ex_slrt_nrt_osc','SimulationMode','normal');
```

The signal from the signal generator is logged to memory through the Outport block.

Simulink runs a simulation in normal mode through to completion. You cannot manually stop the simulation. For further information on the sim command, see the Simulink documentation.

2   After Simulink finishes the simulation, type:

```
plot(output.get('tout'), output.get('yout'))
```

MATLAB opens a plot window and displays the output response as shown in this figure.
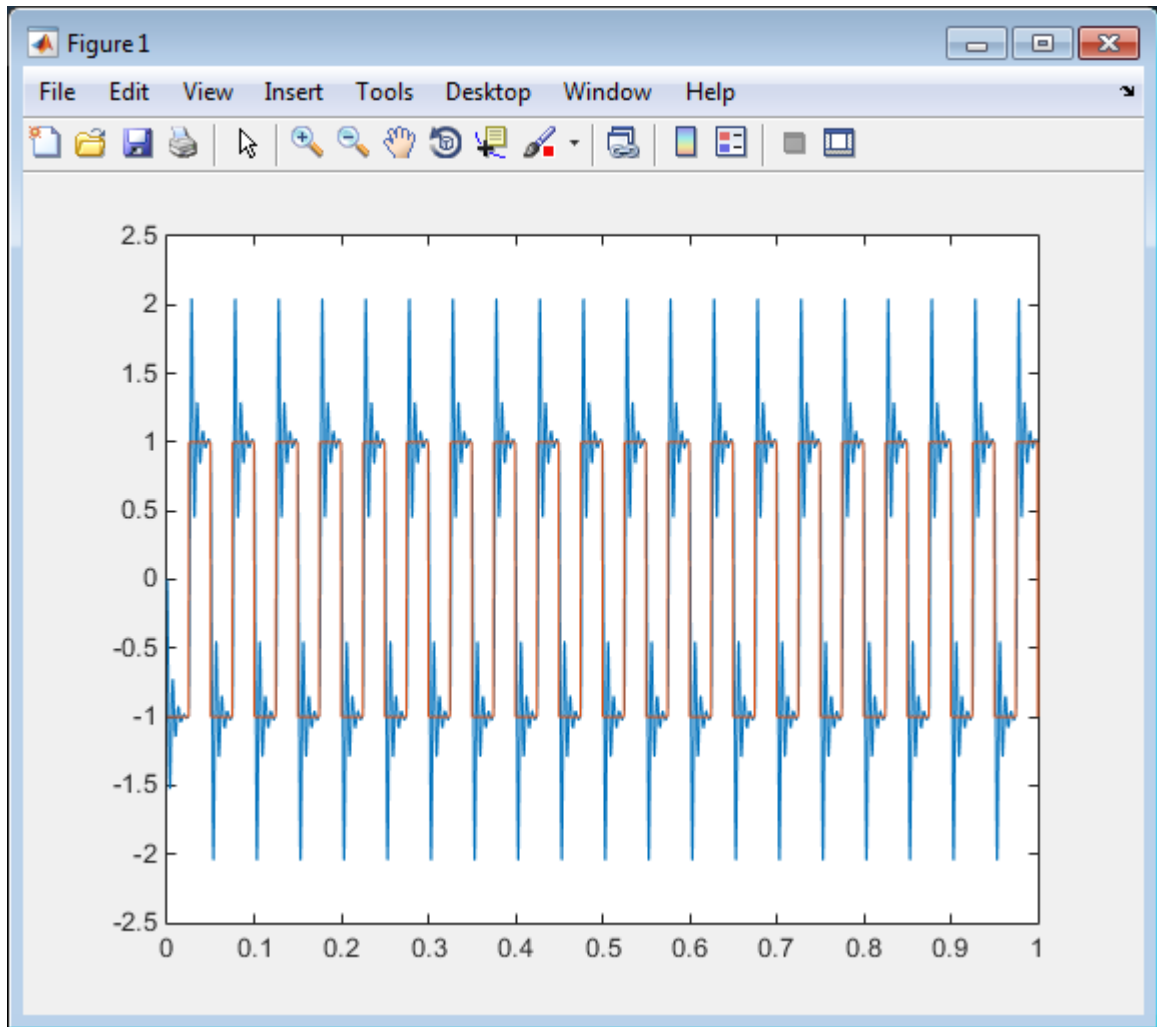
When the real-time application is running in `Real-Time` mode, data is not saved to the variables `tout` and `yout`. Instead, data is saved in the target computer memory and can be retrieved through the target object properties `tg.TimeLog`, `tg.StateLog`, and `tg.OutLog`. However, in the Configuration Parameters dialog box, you must still select the **Time**, **States**, and **Output** check boxes for data to be logged to the target object properties.

See Also
Outport

Prepare Real-Time Application with MATLAB Language

You can configure your real-time application using MATLAB language and build and download it to the target computer. This example uses the ex_slrt_rt_osc (matlab: open_system(docpath(fullfile(docroot, 'toolbox', 'xpc', 'examples', 'ex_slrt_rt_osc')))). It assumes the default target computer is configured for network boot mode and communication has been established.

1 Check that the connection to the target computer is functioning.

   ```
   slrtpingtarget
   ```

2 Load the model.

   ```
   model_name = 'ex_slrt_rt_osc';
   open_system(model_name);
   ```

3 Set the start and stop time parameters.

   ```
   set_param(model_name,'StartTime','0');
   set_param(model_name,'StopTime','10');
   ```

4 Build and download the model.

   ```
   rtwbuild(model_name);
   pause(30);
   ```

5 Assign tg to the target computer.

   ```
   tg = slrt;
   ```

The next step is to execute the real-time application. See "Execute Real-Time Application with MATLAB Language" on page 4-53.

Execute Real-Time Application with MATLAB Language

Run the real-time application with generated code to observe the real-time behavior of the model. This procedure uses the Simulink model `ex_slrt_rt_osc` (`matlab: open_system(docpath(fullfile(docroot, 'toolbox', 'xpc', 'examples', 'ex_slrt_rt_osc')))`). You must have already carried out the steps in "Prepare Real-Time Application with MATLAB Language" on page 4-52.

1 In the Command Window, type:

```
start(tg)
```

The real-time application starts running on the target computer. In the Command Window, the status of the target object changes from stopped to running.

```
Target: TargetPC1
  Connected    = Yes
  Application  = ex_slrt_rt_osc
  Mode    = Real-Time Single-Tasking
  Status    = running
```

2 On the target computer screen, the **Execution** line changes from stopped to running. The **AverageTET** line is periodically updated with a new value.

3 In the Command Window, type:

```
stop(tg)
```

The real-time application stops running.

The Simulink Real-Time software allows you to change many properties and parameters without rebuilding the real-time application. Two of these properties are `StopTime` and `SampleTime`.

4 Change the stop time. For example, to change the stop time to 1000 seconds, type:

```
tg.StopTime = 1000
```

5 Change the sample time. For example, to change the sample time to 0.01 seconds, type:

```
tg.SampleTime = 0.01
```

Note: Some blocks produce incorrect results when you change their sample time at run time. If you include such blocks in your model, the software displays a warning message during model build. To avoid incorrect results, change the sample time in the original model, and then rebuild and download the model.

If you choose a sample time that is too small, a CPU overload can occur. If a CPU overload occurs, the target object property `CPUOverload` changes to `detected`. In that case, change **Fixed step size** to a larger value and rebuild the model.

See Also
`SimulinkRealTime.target.viewTargetScreen`

More About
· "Blocks Whose Outputs Depend on Inherited Sample Time"

Application and Driver Scripts

The Simulink Real-Time examples show the features of the Simulink Real-Time product. They are also scripts that you can view to understand how to write your own scripts for creating and testing real-time applications.

Examples include general applications examples and driver examples. You can access the general application and driver examples through MATLAB Help. In the help window, click **Simulink Real-Time** > **Demos** to see the available example categories.

In this section...
"General Examples" on page 4-56
"Driver Examples" on page 4-57
"Edit Scripts" on page 4-58

General Examples

The general application examples include models and scripts showing:

Description	File Name
Real-time parameter tuning and data logging	"Parameter Tuning and Data Logging"
Freerun display mode of a host scope	"Signal Tracing With a Host Scope in Freerun Mode"
A software-triggered host scope	"Signal Tracing Using Software Triggering"
A signal-triggered host scope	"Signal Tracing Using Signal Triggering"
A scope-triggered host scope	"Signal Tracing Using Scope Triggering"
Signal tracing with a target scope	"Signal Tracing With a Target Scope"
Pre- and post-triggering of a host scope	"Pre- and Post-Triggering of a Host Scope"

Description	File Name
Time- and value-equidistant data logging	"Time- and Value-Equidistant Data Logging"
Logging signal data to a file on the target computer	"Data Logging With a File Scope"
Frame signal processing **Note:** This example requires DSP System Toolbox™ software.	"Frame Signal Processing"
Simulink Real-Time software as a real-time spectrum analyzer	"Spectrum Analyzer"
Creating a custom client application to interface with the target computer	"Simple Client Application With the .NET API"
Concurrent execution of a model using the Simulink Real-Time profiling tool	"Concurrent Execution on Simulink® Real-Time™"

Driver Examples

The driver examples include models and scripts showing:

- Analog and digital I/O
- ARINC 429
- Asynchronous events
- Audio
- CAN (CAN_MESSAGE data types)
- Counters, timers, pulse width modulators (PWM)
- Digital signal processing
- Encoders
- EtherCAT
- FlexRay
- FPGA
- J1939
- MIL-STD-1553

- Precision Time Protocol (PTP)
- Raw Ethernet
- RS-232
- Shared/reflective memory
- UDP
- Video

Note: Because these examples illustrate the use of driver blocks in a Simulink Real-Time environment, you might need specific I/O modules to run them.

Edit Scripts

To locate and edit an example such as "Signal Tracing With a Host Scope in Freerun Mode":

1 Follow the link to the example and determine its MATLAB file name, in this case `scfreerundemo`.

2 In the Command Window, type:

```
which scfreerundemo
```

MATLAB displays the location of the MATLAB example file.

```
C:\MATLAB\toolbox\rtw\targets\xpc\xpcdemos\scfreerundemo.m
```

3 Type:

```
edit scfreerundemo
```

MATLAB opens the file in the MATLAB Editor.

application	See *real-time application*.
build process	Process of generating a real-time application from your Simulink model, compiling, linking, and downloading the generated code to create a *real-time application*.
custom program	A program written in MATLAB, C, or .NET that provides a standalone interface to the *real-time application*.
execution	Running the *real-time application* on the target computer in either `Real-Time` or `Freerun` mode.
executable code	See real-time application.
kernel	Real-time software component running on the target computer that manages the downloaded *real-time application*.
model	Simulink and/or Stateflow model.
parameter tuning	Process of changing block parameters and downloading the new values to a *real-time application* while it is running or not running.
sample rate	Rate the *real-time application* is stepped in samples/second. Reciprocal of the *sample time*.
sample time	Interval, in seconds, between the execution of *real-time application* steps.
signal logging	Acquiring and saving signal data created during a real-time execution.
signal monitoring	Getting the values of one or more signals without time information.
signal tracing	Acquiring and displaying packages of signal data during real-time execution.
simulation	Running a non-real-time simulation of the Simulink and Stateflow model on the development computer.

real-time application   Executable code generated from a Simulink and Stateflow
                              model, which can be executed by the Simulink Real-Time
                              kernel on the target computer.