

xPC Target™ 5

Getting Started Guide

MATLAB®
& SIMULINK®

How to Contact MathWorks



www.mathworks.com Web
comp.soft-sys.matlab Newsgroup
www.mathworks.com/contact_TS.html Technical Support



suggest@mathworks.com Product enhancement suggestions
bugs@mathworks.com Bug reports
doc@mathworks.com Documentation error reports
service@mathworks.com Order status, license renewals, passcodes
info@mathworks.com Sales, pricing, and general information



508-647-7000 (Phone)



508-647-7001 (Fax)



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

For contact information about worldwide offices, see the MathWorks Web site.

xPC Target™ Getting Started Guide

© COPYRIGHT 2000–2011 by The MathWorks, Inc.

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

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

Trademarks

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

Patents

MathWorks products are protected by one or more U.S. patents. Please see www.mathworks.com/patents for more information.

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)

Introduction

1

Product Overview	1-2
Expected Background	1-5
xPC Target Features	1-6
Real-Time Kernel	1-6
Real-Time Application	1-9
Signal Acquisition	1-9
Parameter Tuning	1-10
Fixed-Point Support	1-12
MATLAB® Compiler Support	1-12
BLAS Library Support	1-12
Hardware Environment	1-14
Introduction	1-14
Host PC	1-14
Target PC	1-15
Host-Target Connection	1-16
I/O Driver Support	1-18
Software Environment	1-21
Introduction	1-21
Host-Target Communication	1-21
Rapid Prototyping Process	1-22
Hardware-in-the-Loop Process	1-25
The xPC Target Embedded Option Product	1-26
User Interaction	1-28
Introduction	1-28
xPC Target Explorer	1-29
MATLAB Command-Line Interface	1-30
Simulink External Mode Interface	1-32
Simulink with xPC Target Scope Blocks	1-33
Target PC Command-Line Interface	1-33

Web Browser Interface	1-34
Custom GUI with xPC Target API for Microsoft .NET Framework	1-34
Custom GUI with xPC Target API	1-35
Custom GUI with xPC Target COM API	1-35

Installation and Configuration

2

Required Products	2-2
MATLAB	2-2
Simulink	2-3
Simulink® Coder	2-4
MATLAB® Coder	2-4
C Compiler	2-4
xPC Target Embedded Option Product	2-6
 Related Products	 2-7
 System Requirements	 2-8
Introduction	2-8
Host PC Requirements	2-8
Target PC Requirements	2-9
 Installation on the Host PC	 2-15
Overview	2-15
xPC Target Turnkey System	2-15
License Requirements	2-16
Files on the Host PC Computer	2-16
Setting Your Initial Working Folder	2-17
Running MATLAB Remotely	2-18
Configuring the xPC Target Host PC for Your C Compiler	2-19
 xPC Target Explorer	 2-22
Introducing xPC Target Explorer	2-22
The xPC Target Product and Default Target PCs	2-24

Network Communication	2-26
Network Communication Overview	2-26
Hardware for Network Communication	2-26
Ethernet Card for a PCI Bus	2-27
Ethernet Card for an ISA Bus	2-27
Environment Properties for Network Communication	2-29
Serial Communication	2-36
Serial Communication Overview	2-36
Hardware for Serial Communication	2-36
Environment Properties for Serial Communication	2-37
xPC Target Boot Options	2-43
Introduction	2-43
Bootting Target PCs from CD or DVD	2-45
Bootting Target PCs from Boot Floppy Disk	2-51
Bootting Target PCs Within a Dedicated Network	2-54
Configuring the xPC Target Environment for 64-Bit	
MATLAB Systems	2-61
Setup Requirements	2-61
Configuring the Compiler	2-62
Configuring Communications	2-63
Configuring Boot Methods	2-65
Testing and Troubleshooting the Installation	2-71
Testing the Installation from a Boot Disk or Boot CD	2-71
Test 1, Ping Target System Standard Ping	2-74
Test 2, Ping Target System xPC Target Ping	2-75
Test 3, Reboot Target Using Direct Call	2-76
Test 4, Build and Download Application	2-77
If You Need More Help	2-78
Exporting and Importing xPC Target Explorer	
Environments	2-79

Simulink Model	3-2
Creating a Simple Simulink Model	3-2
Entering Parameters for the Scope Block	3-6
Adding a Simulink Outport Block	3-10
Entering Parameters for the Outport Blocks	3-13
Adding an xPC Target Scope Block	3-17
Entering Parameters for an xPC Target Scope Block	3-22
Simulating the Model	3-39
Simulating the Model with Simulink	3-39
Simulating the Model from MATLAB	3-41
xPC Target Application	3-45
Booting the Target PC	3-45
Troubleshooting the Boot Process	3-47
Entering the Simulink® Coder Parameters	3-47
Building and Downloading the Target Application	3-52
Troubleshooting the Build Process	3-54
Increasing the Time-Out Value	3-55
xPC Target Options Node	3-56
Running the Target Application	3-57
Introduction	3-57
Control with xPC Target Explorer	3-57
Control with MATLAB Commands	3-67
Control with Simulink External Mode	3-69
Parallel Model Reference Builds Using Remote Workers	3-71
Menu Bar and Toolbar Contents and Shortcut Keys ..	3-72

Glossary

Index

Introduction

- “Product Overview” on page 1-2
- “Expected Background” on page 1-5
- “xPC Target Features” on page 1-6
- “Hardware Environment” on page 1-14
- “Software Environment” on page 1-21
- “User Interaction” on page 1-28

Product Overview

xPC Target™ enables you to execute Simulink® and Stateflow® models on a target computer for rapid control prototyping, hardware-in-the-loop (HIL) simulation, and other real-time testing applications. It provides a library of IO device drivers, a real-time kernel, and an interface for real-time monitoring, parameter tuning, and data logging.

xPC Target Turnkey combines xPC Target with a variety of high-performance, real-time target computers for a complete, fully assembled, real-time testing solution. You can program FPGA boards for xPC Target Turnkey systems using code generated by Simulink® HDL Coder™.

Learn more about xPC Target Turnkey (<http://www.mathworks.com/products/xpctarget/supported-hardware/-index.html>).

The xPC Target environment uses a target PC, separate from a host PC, for running real-time applications. In this environment you use your desktop computer as a host PC with MATLAB®, Simulink, and Stateflow (optional) software, to create a model using Simulink blocks and Stateflow charts. After creating your model, you can run simulations in nonreal time within the Simulink environment.

Use xPC Target software with Simulink® Coder™, Embedded Coder™ (optional), and a C/C++ compiler to create executable code that represents your models. The executable code is downloaded from the host PC to the target PC running the xPC Target real-time kernel. After downloading the executable code, you can run and test your target application in real time. Additionally, xPC Target software lets you add I/O blocks to your model to connect and communicate with your hardware under test

- **Hardware requirements** — The xPC Target software requires a host PC, target PC, and, for I/O, the target PC must also have I/O boards supported by the xPC Target product. The target PC can be a desktop PC, industrial PC, PC/104, PC/104+, or CompactPCI computer.
- **Software requirements** — The xPC Target software requires either a Microsoft® Visual C/C++ compiler or an Open Watcom C/C++ compiler.

In addition, the xPC Target software requires MATLAB, Simulink, and Simulink Coder software.

- xPC Target Embedded Option™ requirements — The xPC Target Embedded Option product is separate from the xPC Target product. It requires an additional license from MathWorks. With this additional license, you can deploy an unlimited number of real-time applications for stand-alone operation. This option allows you to
 - Create stand-alone applications for the target PC, which can boot, run, and operate independent from the host PC.
 - Deploy stand-alone GUI applications running on the host PC to control, change parameters, and acquire signal data from a target application. This feature uses: the xPC Target API with any programming environment; the xPC Target COM API with any programming environment, such as Microsoft® Visual Basic®, that can use COM objects; the xPC Target API for Microsoft .NET Framework. Without the xPC Target Embedded Option product, you can create, but not deploy, stand-alone GUI applications running on a host PC that does not contain your licensed copy of the xPC Target software, to control, change parameters, and acquire signal data from a target application.
- Documentation and help — The xPC Target software ships with the *xPC Target Getting Started Guide*. This guide and the remaining documentation are available online through the MATLAB Help browser window, or as PDF files that you can view online or print.

For additional information on using the xPC Target product, see the following MathWorks Web site resources:

- MATLAB Central File Exchange for xPC Target Product.
- MathWorks Support xPC Target Web site (<http://www.mathworks.com/support/product/XP>). The xPC Target documentation is also available from this site.

Expected Background

Users who read this book should be familiar with

- Using the Simulink and Stateflow products to create models as block diagrams, and simulating those models in Simulink
- The concepts and use of Simulink Coder software to generate executable code

When using the Simulink Coder and xPC Target products, you do not need to program in C or other programming languages to create, test, and deploy real-time systems.

If you are a new user — Begin with Chapter 1, “Introduction”. This chapter gives you an overview of the xPC Target features and xPC Target environment. Next, read and try the examples in Chapter 3, “Basic Tutorial”.

If you are an experienced user — After you are familiar with using the xPC Target software, read or browse the following chapters in the *xPC Target User’s Guide*: “Software Environment and Demos” and “Targets and Scopes in the MATLAB Interface” for more detailed information about the commands in the xPC Target software.

xPC Target Features

In this section...
“Real-Time Kernel” on page 1-6
“Real-Time Application” on page 1-9
“Signal Acquisition” on page 1-9
“Parameter Tuning” on page 1-10
“Fixed-Point Support” on page 1-12
“MATLAB® Compiler Support” on page 1-12
“BLAS Library Support” on page 1-12

Real-Time Kernel

The xPC Target software does not require Microsoft DOS, Microsoft Windows®, Linux®, or any another operating system on the target PC. Instead, you boot the target PC with boot media that includes the xPC Target kernel.

However, the xPC Target Embedded Option product requires DOS and a DOS license at boot time. For more information, see “Embedded Option” in the *xPC Target User’s Guide*.

Target Boot Options

You boot and run the target PC with one of the following boot options. These boot options eliminate the need to install software, modify existing software configurations, or access the hard disk on the target PC. This arrangement allows you to use the target PC for testing real-time applications. When you are finished with your tests, you can use the target PC again as a desktop computer. Software is not permanently installed on the target PC unless you deliberately install a stand-alone application on the hard disk or flash memory.

- Removable boot devices
 - CD, DVD, 3.5-inch floppy disk
- Fixed boot devices

Hard drives (IDE or serial ATA (SATA)) or flash disks

- Network boot
 - Dedicated network

Target PC BIOS

At the beginning of the target PC boot process, the BIOS is loaded. Among other tasks, the BIOS searches for a bootable image (executable). This bootable image includes 16-bit and 32-bit tasks. The 16-bit task runs first because the CPU is in real mode by default. It prepares the descriptor tables and switches the CPU to protected mode. Next, the 32-bit task runs. It prepares the target PC environment for running the kernel and finally starts the real-time kernel.

You might need to enter the BIOS to customize settings for optimal real-time behavior of the system. For example, you can suppress checking for a keyboard or switch off any power save features. Enabled power features can generate system management interrupts (SMIs). These features and support can also degrade real-time performance.

After loading the kernel, the target PC does not make calls to the BIOS or DOS functions. The resources on the CPU motherboard (for example, interrupt controller, UART, and counters) are addressed entirely through I/O addresses.

Real-Time Kernel

After the kernel starts running, it displays a welcome message with information confirming the host-target connection. The kernel activates the application loader and waits to download a target application from the host PC. Upon download, the loader receives the code, copies the different code sections to their designated addresses, and sets the target application ready to start. You can now use xPC Target functions and other utilities to communicate with the target application.

It is important to note that after the CPU switches to protected mode (32-bit), none of the xPC Target components switches the CPU back to real mode (16-bit).

The generated real-time application and the real-time kernel are compiled with a flat memory model. This provides full 32-bit power without time-consuming 16-bit segment switching and Microsoft DOS extenders.

Real-Time Application

The Simulink Coder, Embedded Coder, MATLAB® Coder™, and xPC Target products, and a C compiler, create a real-time application (target application) from a Simulink and Stateflow model. Target applications created with the Simulink Coder and xPC Target software run in real time on a standard PC using an xPC Target real-time kernel.

The target application runs in real time on the target PC and has the following characteristics:

- **Memory model** — The target application is compiled as an application with a flat memory model. This executable is then converted to an image suitable for the xPC Target software, and it provides full 32-bit power without time-consuming 16-bit segment switching and DOS extenders. It also does not rely on DOS or any other Microsoft operating system.
- **Task execution time** — The target application is capable of high-speed, real-time task execution. A small block diagram can run with a sample time as fast as 20 μ s (50 kHz). Model size, complexity, and target PC hardware affect maximum speed (minimal sample time) of execution.

For more information on creating a target application, see “xPC Target Application” on page 3-45.

Signal Acquisition

The xPC Target real-time kernel stores signal data from the target application in RAM on the target PC. Alternatively, you can have the xPC Target real-time kernel store signal data in a file on the target PC. In either case, you can use this signal data to analyze and visualize signals. The xPC Target product supports the following types of signal acquisition:

- **Signal monitoring** — This is the process of acquiring signal data without time information. In this mode, you can get the current values of one or more signals. The data is not acquired in the real-time task but in the background task. The advantage of this process is that collecting data does not add any computational load to running the real-time application.

For example, if you have a LED gauge in a Simulink model on the host PC, you could use signal monitoring to display the status of the signal.

- **Signal logging** — This is the process of acquiring signal data while a target application is running, and then visualizing the collected data after the target application stops running. The data is collected in the real-time task and acquired samples are associated with a time stamp. After the run finishes or you manually stop the run, the host PC makes a request to upload data from the target PC. You can then visualize signals by plotting data on the host PC, or you can save data to a disk.
- **Signal tracing** — This is the process of acquiring and visualizing signal data while a target application is running. The data is collected in the real-time task and acquired samples are associated with a time stamp. It allows you to acquire signal data and visualize it on the target PC or to upload the signal data and visualize it on the host PC while the target application is running. The flexibility of this acquisition type is very similar to the behavior of a digital oscilloscope.

For information on acquiring signal data with the xPC Target software, see “User Interaction” on page 1-28 in the xPC Target™ Getting Started Guide on page 1 and “Signal Monitoring with the MATLAB Interface”, “Signal Logging” and “Signal Tracing” in the *xPC Target User’s Guide* documentation.

Parameter Tuning

Most Simulink blocks have parameters (such as the amplitude and frequency of a sine wave) that you can change before or while your target application is running:

- **Interactive** — The xPC Target software supports tuning of parameters while the target application is running in real time.

Note Opening a dialog box for a source block causes Simulink to pause. While Simulink is paused, you can edit the parameter values. You must close the dialog box to have the changes take effect and allow Simulink to continue.

- **Scripts and batch procedures** — The xPC Target software also includes commands to change parameters during a run or between runs. By writing a script that incrementally changes a parameter and monitors a signal

output and running it on the host PC, you can optimize the value of that parameter.

For information on tuning parameters with the xPC Target software, see “User Interaction” on page 1-28 and “Parameter Tuning and Inlining Parameters” in the *xPC Target User’s Guide*.

Fixed-Point Support

The xPC Target software supports Simulink fixed-point data. This enables you to

- Monitor and log signals of fixed-point data types
- Tune parameters of fixed-point data types

For more information on using fixed-point data, see the “Simulink Fixed Point”.

MATLAB Compiler Support

The xPC Target software supports the MATLAB® Compiler™. With this capability, you can use the MATLAB Compiler to take MATLAB files as input and generate redistributable, stand-alone applications that include xPC Target functionality.

Stand-alone applications that include xPC Target functionality have the following limitations:

- No MATLAB Compiler support, which results in no access to the xPC Target library (`xpc.lib`).
- No xPC Target Explorer, or other xPC Target graphical user interface support.
- No code generation functionality.

To use these features, create a file that uses the MATLAB Compiler command-line interface for the xPC Target software (for example, then use the MATLAB Compiler).

BLAS Library Support

The xPC Target software supports the Basic Linear Algebra Subprograms (BLAS) library. This library speeds up large matrix (up to 16 x 16) operations in target applications.

Note Your model accesses this library if you build your model with a Microsoft Visual C/C++ compiler. If you build your model with the Open Watcom compiler, the xPC Target software does not use the BLAS library.

Hardware Environment

In this section...
“Introduction” on page 1-14
“Host PC” on page 1-14
“Target PC” on page 1-15
“Host-Target Connection” on page 1-16
“I/O Driver Support” on page 1-18

Introduction

The hardware environment consists of a host computer, target computer, I/O boards in the target computer, and a serial or network connection between the host and target computers. Knowing the different types of computers and I/O supported by the xPC Target software will help you to set up a real-time testing environment that meets your needs.

For a complete, fully assembled, real-time testing solution, see xPC Target Turnkey. xPC Target Turnkey combines the xPC Target software with a variety of high-performance real-time target computers.

Host PC

You can use any PC that runs a Windows operating system supported by MathWorks as the host PC. It must also contain an available serial port or Ethernet adapter. In addition, to provide a means to boot the target PC, the host PC must have at least:

- A CD or DVD drive
- The ability to belong to a dedicated network
- A 3.5-inch floppy disk drive

For more details on the requirements of the host PC, see “Host PC Requirements” on page 2-8.

Target PC

The xPC Target software supports concurrent access to up to 64 target PCs with one host. A target PC can connect to only one host PC at any given time. A target PC cannot connect to multiple host PCs. A target PC can be almost any PC compatible system with a 32-bit Intel® or AMD® processor (386 compatible or higher). It must also contain a free serial port or an Ethernet adapter. In addition, the target PC must contain a 3.5-inch floppy disk drive, CD or DVD drive, or have the ability to belong to a dedicated network. Using the xPC Target Embedded Option software, you can transfer files from the 3.5-inch disk or CD to a hard disk or flash memory. Do not use a laptop PC as a target PC.

A target PC can be one of the following:

- **Desktop PC** — This computer is booted from a special target boot disk or network boot image created by the xPC Target software.

When you boot the target PC from the target boot disk or network boot image, the xPC Target software uses the resources on the target PC (CPU, RAM, and serial port or network adapter) without changing the files already stored on the hard drive.

After you are done using your desktop computer as a target PC, you can reboot your computer without the target boot image and resume normal use of your desktop computer.

- **Industrial PC** — This computer is booted from a special target boot disk or network boot image, or from a hard disk or flash memory.

When using an industrial target PC, you can select PC/104, PC/104+, CompactPCI, or single-board computer (SBC) hardware.

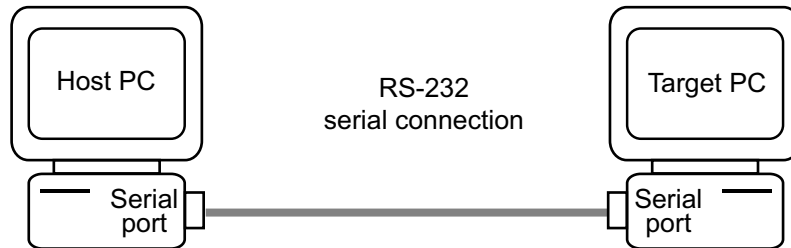
You do not need any special target hardware. However, the target PC must be a fully PC-compatible system and contain a serial port or an Ethernet controller compatible with the xPC Target software.

For more details on the requirements of the target PC, see “Target PC Requirements” on page 2-9.

Host-Target Connection

The xPC Target product supports two connection types and communication protocols between the host PC and the target PC: serial and network.

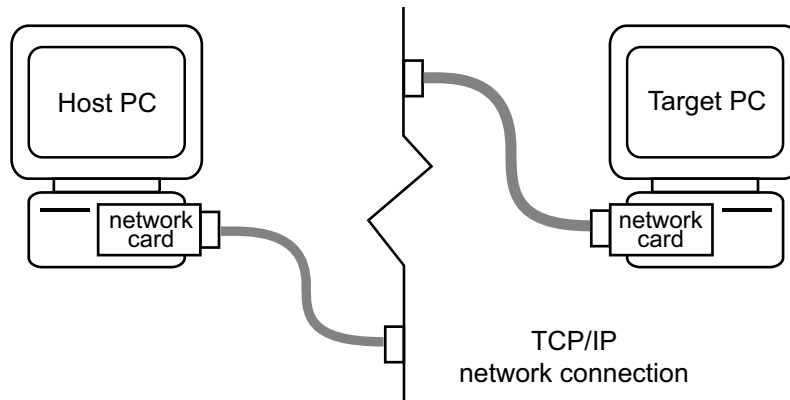
Serial — The host and target computers are connected directly with a serial cable using their RS-232 ports. This cable is wired as a null modem link that can be up to 5 meters long and with a transfer rate between 1200 and 115200 baud. A null modem cable is provided with the xPC Target software.



For detailed information on setting up the hardware and software for serial communication, see “Serial Communication” on page 2-36.

Network — The host and target computers are connected through a network. The network can be a LAN, the Internet, or a direct connection using a crossover Ethernet cable. Both the host and target computers are connected to the network via Ethernet adapters using the TCP/IP protocol for communication.

When using a network connection, the target PC requires a supported Ethernet adapter card. The data transfer rate can be 10 megabits/second, 100 megabits/second, or 1 gigabit/second. For a list of supported cards, see “Hardware for Network Communication” on page 2-26.



For detailed information on setting up the hardware and software for network communication, see “Network Communication” on page 2-26.

Advantages of Network Communication

A host-to-target connection using network TCP/IP communication has advantages over serial RS-232 communication:

- Higher data throughput — Network communication using Ethernet can transfer data up to 100 Mbit/second instead of the maximum data transfer rate of 115 kBaud with serial communication.
- Longer distances between host and target computer — By using repeaters and gateways you do not restrict the distance between your host and target computers to the length of a serial cable. Communication over the Internet is also possible.

This manual does not include information for installing network cards or the TCP/IP protocol on your host computer. For correct installation and setup of your network cards and the TCP/IP protocol, contact your system administrator.

I/O Driver Support

The xPC Target product supports a wide range of third-party I/O boards. The list of supported I/O boards includes ISA, PCI, PCIe, PMC, PC/104, PC/104+, and CompactPCI hardware. The drivers are represented by Simulink blocks. Your interaction with the I/O boards is through these Simulink blocks and the parameter dialog boxes. MathWorks does not manufacture the boards.

Note You are responsible for taking necessary precautions and implementing safeguards when interfacing hardware with the xPC Target product. You are also solely responsible for the content of your models that controls such hardware.

I/O board library — The I/O board library contains Simulink blocks for the xPC Target product. You drag and drop blocks from the I/O library and connect I/O drivers to your model the same way you would connect any standard Simulink block.

I/O support — The I/O device library supports approximately 300 standard boards. I/O boards plug into the target PC expansion bus, PC/104 stack, or industrial PC chassis. There is also support for modules that plug into IP or PMC carrier boards. The xPC Target block library supports the following I/O functions:

- Analog input (A/D) and analog output (D/A) — Connect sensors and actuators to your target application.
- Digital input and output — Connect to switches, on/off devices, and communicate information in parallel.
- RS-232/422/485 support — Use the COM1 or COM2 ports for serial communication with external devices. You can also access multiple RS-232, RS-422, and RS-485 serial ports using Quatech® and Commtech devices. See “Serial Communications Support” in the *xPC Target I/O Reference*.
- CAN support — You can use CAN-AC2, CAN-AC2-PCI, and CAN-AC2-104 boards from Softing® GmbH AG with xPC Target CAN blocks to interface with a CAN field bus network. This interface provides communication through a CAN network between target applications and remote sensors and actuators.

The xPC Target CAN blocks are compatible with CAN specifications 2.0A and 2.0B and use dynamic object mode. See “CAN I/O Support” and “CAN I/O Support for FIFO” in the *xPC Target I/O Reference*.

- GPIB support — Special RS-232 drivers support communication with a GPIB control module from National Instruments® to external devices with a GPIB connector. See “GPIB I/O Support” in the *xPC Target I/O Reference*.
- UDP support — Communicate with another system using the standard UDP/IP network protocol. See “UDP I/O Support” in the *xPC Target I/O Reference*.
- Counter-Timers — Use the counter-timer blocks for measuring pulse and frequency with modulation applications.
- Watchdog — Monitor an interrupt or memory location, and reset the computer if an application does not respond. See “Access” and “Versallogic” in the *xPC Target I/O Reference*.
- Incremental encoder — Change motion into numerical information for determining position, direction of rotation, and velocity.
- Shared memory — Use shared memory blocks with multiprocessing applications.
- LVDT — Use the North Atlantic Industries, Inc. 73LD3, 76CL1, 76LD1, and 76CL1 boards with xPC Target LVDT blocks to work with LVDT applications.
- ARINC-429 — Use the Condor Engineering CEI-X20 boards with xPC Target ARINC-429 blocks to interface with the ARINC 429 data bus.
- MIL-STD-1553 — Use the Condor Engineering PCI-1553 and QPCI-1553 series boards with xPC Target MIL-STD-1553 blocks to interface with the MIL-STD-1553 data bus.
- Audio — Use the audio blocks to work with audio applications. See General Standards PMC66-16AO16 and General Standards PMC-24DSI12 in the *xPC Target I/O Reference*.
- Thermocouple — Use the Measurement Computing™ PCI-DAS-TC board with xPC Target thermocouple blocks to work with thermocouple applications.

For information on using specific I/O driver blocks and advanced I/O support, see the *xPC Target I/O Reference*.

Software Environment

In this section...
“Introduction” on page 1-21
“Host-Target Communication” on page 1-21
“Rapid Prototyping Process” on page 1-22
“Hardware-in-the-Loop Process” on page 1-25
“The xPC Target Embedded Option Product” on page 1-26

Introduction

The software environment is a place to design, build, and test a target application in nonreal time and real time. It also includes communication between the host and target computers.

Host-Target Communication

Whether using a serial connection (RS-232) or a network connection (TCP/IP), information is exchanged between the host PC and target PC. This information includes

- Target application — Download a target application from the host to the target computer.
- Control — Change properties and control the target application. This includes starting and stopping the target application, changing sample and stop times, and getting information about the performance of the target application and CPU.
- Signal data — Upload signal data from the target computer for analysis after the target application is finished running, or view signal data during the run.
- Parameter values — Download parameter values to the target computer between runs or during a run.

Rapid Prototyping Process

You create a real-time testing environment for Simulink models by connecting a host PC, target PC, and the hardware you want to test. You run the following software on the host PC:

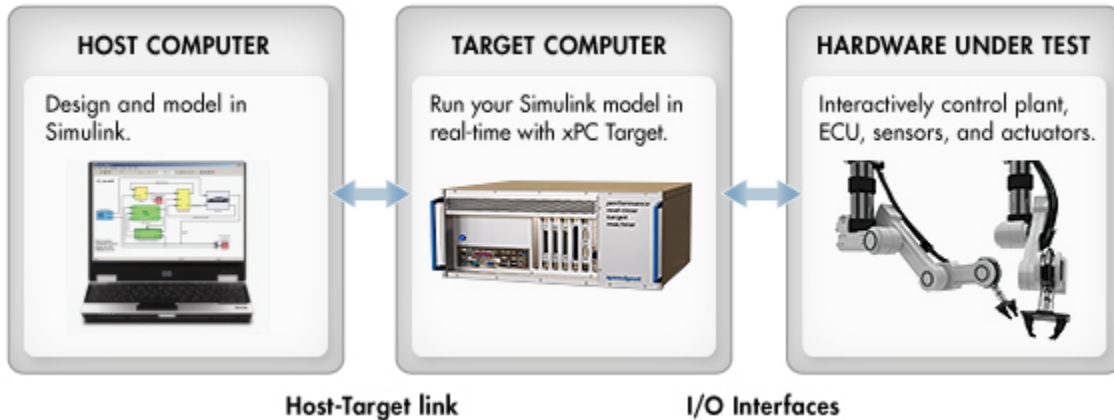
- xPC Target
- Simulink
- Simulink Coder
- MATLAB Coder
- A C compiler

And connect the host PC to the target PC via a single TCP/IP or RS-232 connection. You then:

- 1** Connect the target PC to the hardware you want to test.
- 2** Download code generated by Simulink Coder from a Simulink model to the target PC via the communications connection.

Once you make the connections, you can:

- Access and interactively control the target PC and target application.
- Tune parameters before, during, and after real-time execution.
- Acquire, monitor, and log signal data.



Note Opening a dialog box for a source block causes Simulink to pause. While Simulink is paused, you can edit the parameter values. You must close the dialog box to have the changes take effect and allow Simulink to continue.

The rapid prototyping process includes the following tasks:

- 1** Create a Simulink or Stateflow model — You create block diagrams in Simulink using simple drag-and-drop operations, and then you enter values for the block parameters and select sample rates. If you use continuous-time components, you also need to select an integration algorithm.
- 2** Simulate the model in nonreal time — Simulink uses a computed time vector to step the model. After the outputs are computed for a given time value, Simulink immediately repeats the computations for the next time value. This process is repeated until it reaches the stop time.

Note Because this computed time vector is not connected to a hardware clock, the outputs are calculated in nonreal time as fast as your computer can run. The time to run a simulation can differ significantly from real time.

- 3** Create an executable target application — The Simulink Coder, MATLAB Coder, xPC Target products, and a C compiler, create the target application that runs on the target PC. This real-time application uses the initial parameters from the Simulink model that were available at the time of code generation.
- 4** Install I/O boards in target PC and make connections to other hardware as appropriate — Install and connect to the hardware against which you want to execute and test the target application in real time.
- 5** Execute the target application in real time — The target PC is booted using an xPC Target boot image that loads the xPC Target real-time kernel. After booting the target PC, you can build and download a real-time application.

The xPC Target software uses real-time resources on the target PC hardware. Based on your selected sample rate, the xPC Target software uses interrupts to step the model at the selected rate. With each new interrupt, the target application computes all the block outputs from your model.

- 6** Acquire signals — Acquire signal data using xPC Target scopes.

Scopes created by xPC Target Scope blocks acquire data according to Simulink sample time rules. This includes non-regular execution, such as if the scope is in an enabled or triggered subsystem. Note that scopes created dynamically (from the MATLAB Command Window or the API) sample at the base rate, irrespective of the sample time of their signals.

You can create xPC Target scopes and acquire data from the target application by

- xPC Target — Use the xPC Target Remote Control Tool and Scope Manager windows to create scopes, and use the Simulink viewer to add signals.
- MATLAB — Enter commands in the MATLAB Command Window.
- Simulink — Add xPC Target Scope blocks to your Simulink model.

Note xPC Target software does not support normal Simulink Scope blocks in external mode. Instead, use xPC Target Scope blocks.

- Target PC — Use commands in the target PC command window.
 - Simulink GUI — Add blocks to a Simulink user interface model with xPC Target From blocks. See “Graphical User Interfaces” in the *xPC Target User’s Guide* documentation for details.
- 7** Tune parameters — You can tune parameters by
- xPC Target GUI — Use the xPC Target Remote Control Tool and Simulink viewer to change parameters.
 - MATLAB interface — Enter commands in the MATLAB window.
 - Simulink interface — Use your Simulink model with external mode.
 - Target PC — Use commands in the target PC command window.
 - Web browser — Use the xPC Target Web browser interface.
 - Simulink GUI — Add blocks to a Simulink user interface model with xPC Target To blocks. See “Graphical User Interfaces” in the *xPC Target User’s Guide* for details.

Hardware-in-the-Loop Process

Hardware-in-the-loop (HIL) simulation lets you test the implementation of your embedded system, such as a control system, in real time using test benches and simulations from your design phase. HIL simulation is especially valuable when:

- The system is not yet built.
- Safety and performance concerns require testing the system prior to human involvement.
- You need to minimize expensive downtime for the real system.
- You need to test operation and failure conditions that are difficult to physically replicate.

Using your system-level model, which consists of the embedded control system and the plant model, you can reuse your plant model, which executes in the Simulink environment, and use the Simulink Coder software to run these models on real-time testing equipment. By automatically generating code from your models with the Simulink Coder software, you focus on testing, not programming.

The HIL process is similar to the rapid prototyping process (“Rapid Prototyping Process” on page 1-22).

The xPC Target Embedded Option Product

xPC Target Embedded Option enables applications generated with xPC Target to run on a target computer without being connected to a host computer. You can run your applications on a standalone target computer for data acquisition, calibration, testing, and small-batch production scenarios. You can distribute the applications royalty-free to any number of target computers.

When you have completed developing and testing, you can use the target application as a real-time system that runs on a dedicated target PC without needing to connect to the host computer.

The xPC Target Embedded Option product has one mode of operation, StandAlone. In this case, the target PC boots into the Microsoft DOS environment, starts the DOS program `xpcboot.com` from `autoexec.bat`, and then starts the kernel from `xpcboot.com`:

When using Boot Floppy or CD Boot, you do not need DOS environment to load and run the xPC Target kernel. DOSLoader mode, like StandAlone mode, boots the target PC into DOS, starts the DOS program `xpcboot.com` from `autoexec.bat`, and then starts the kernel from `xpcboot.com`.

Note The xPC Target Embedded Option software is a separate product that requires an additional license from MathWorks. With this additional license you can deploy an unlimited number of real-time applications for stand-alone operation.

For more information on the xPC Target Embedded Option product, see “Embedded Option” in the *xPC Target User’s Guide*.

StandAlone Mode

StandAlone mode combines the target application with the kernel and boots them together on the target PC from a hard disk drive or flash memory. The host PC does not have to be connected to the target PC.

- 1** Select StandAlone mode from the Configuration node in the **xPC Target Hierarchy** pane of the xPC Target Explorer tool.
- 2** Build a kernel/target application.
- 3** Copy DOS system files, utilities, kernel/application files, and helper files to the target PC hard drive or flash memory.
- 4** Boot the target PC.

When you boot the target PC, the target PC loads DOS environment, which then calls the xPC Target `autoexec.bat` file to start the xPC Target kernel (`*.rtb`) and associated target application. If you set up the boot device to run the xPC Target `autoexec.bat` file upon startup, the target application starts executing as soon as possible. The xPC Target application executes entirely in protected mode using the 32-bit flat memory model.

For more information on the xPC Target Embedded Option product, see “Embedded Option” in the *xPC Target User’s Guide*.

User Interaction

In this section...

“Introduction” on page 1-28
“xPC Target Explorer” on page 1-29
“MATLAB Command-Line Interface” on page 1-30
“Simulink External Mode Interface” on page 1-32
“Simulink with xPC Target Scope Blocks” on page 1-33
“Target PC Command-Line Interface” on page 1-33
“Web Browser Interface” on page 1-34
“Custom GUI with xPC Target API for Microsoft .NET Framework” on page 1-34
“Custom GUI with xPC Target API” on page 1-35
“Custom GUI with xPC Target COM API” on page 1-35

Introduction

The xPC Target environment has a modifiable interface to the target PC. You can use this interface from MATLAB or Simulink, and you can use other development environments to create stand-alone client applications independent of MATLAB. Because of this open environment, there are several ways to interact with your target application from the host and target PCs.

Note Some blocks (see “Blocks Whose Outputs Depend on Inherited Sample Time” in the *Simulink User’s Guide*) cannot properly handle sample time changes at run-time. For models that contain these blocks, change the sample time in the model first, then build that model. Although the xPC Target product allows you to change sample times at run-time, changing them at run-time for these blocks might cause incorrect results.

The following table compares the interfaces supported by the xPC Target product.

Interface	Environment Properties	Control	Signal Acquisition	Parameter Tuning
“xPC Target Explorer” on page 1-29	X	X	X	X
“MATLAB Command-Line Interface” on page 1-30	X	X	X	X
“Simulink External Mode Interface” on page 1-32		X		X
“Simulink with xPC Target Scope Blocks” on page 1-33			X	
“Target PC Command-Line Interface” on page 1-33		X	X	X
“Web Browser Interface” on page 1-34		X	X	X
“Custom GUI with xPC Target API for Microsoft .NET Framework” on page 1-34		X	X	X
“Custom GUI with xPC Target API” on page 1-35		X	X	X
“Custom GUI with xPC Target COM API” on page 1-35		X	X	X

xPC Target Explorer

The xPC Target software offers a graphical user interface (GUI) for configuring the host and target PCs and interacting with a target application. To open the xPC Target GUI, in the MATLAB Command Window, type `xpcexplr`.

The xPC Target Explorer is an all-in-one user interface that includes the following functionality.

- **Environment** — Use the xPC Target Explorer to change properties in the xPC Target environment.

For more information on environment properties, see

- “Serial Communication” on page 2-36 and “Network Communication” on page 2-26
- “Working with Target PC Environments” in the *xPC Target User’s Guide*
- The `getxpcenv` function in the *xPC Target User’s Guide*
- Control — Use the xPC Target Explorer to download a model. After the target application is downloaded to the target PC, you can use xPC Target Explorer to run it. Use xPC Target Explorer to change stop time and sample times without regenerating code, and get statistical performance information during or after the last run.
- Signal acquisition — Use the xPC Target Explorer Model Hierarchy node to interactively add host, target, or file scopes, and add or remove signals.

For more information on using scopes with the xPC Target Explorer, see “Signals and Parameters” in the *xPC Target User’s Guide*.

- Parameter tuning — Use the xPC Target Explorer Model Hierarchy node to change tunable parameters in your target application.

For more information, see “Signals and Parameters” in the *xPC Target User’s Guide*.

MATLAB Command-Line Interface

You can interact with the xPC Target environment through the MATLAB command-line interface. Enter xPC Target functions in the MATLAB window on the host PC. You can also write your own MATLAB scripts that use xPC Target functions for batch processing.

The xPC Target software has more than 90 MATLAB functions for controlling the target application from the host computer. These functions define, at the most basic level, what you can do with the xPC Target environment.

The GUIs provided with the xPC Target product are for completing the most common tasks. They use the xPC Target 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 boot disk or network boot image and directly change the environment properties without using a graphical interface.

For more information on environment properties, see “Creating a 3.5-Inch Target Boot Disk with a Command-Line Interface” on page 2-53, and “Software Environment and Demos” in the *xPC Target User’s Guide*.

- Control — Reboot the target PC, download a target application, start and stop target applications, and change start and sample times without regenerating code. Get statistical performance information during or after the last run. Add and remove scopes, add/remove signals to scopes, and define triggers for scope display.

For more information, see “Control with MATLAB Commands” on page 3-67 in *xPC Target™ Getting Started Guide* on page 1 and “Software Environment and Demos” in the *xPC Target User’s Guide*.

- Signal acquisition — Trace signals for viewing while the target application is running and monitor signal values without time information. Transfer logged signal data to the MATLAB workspace by uploading from the target PC to the host PC between runs. For stand-alone target PCs, if you write signal data to a file, use the `ftp` utility to transfer that file to a remote PC.

For more information, see “Signal Monitoring with the MATLAB Interface” “Signal Tracing with the MATLAB Interface” and “Signal Logging in the MATLAB Interface”, and “Targets and Scopes in the MATLAB Interface” in the *xPC Target User’s Guide*.

- Parameter tuning — Change parameters while the target application is running, and use *xPC Target* functions to change parameters in between runs.

For more information, see “Parameter Tuning with the MATLAB Interface” and “Targets and Scopes in the MATLAB Interface” in the *xPC Target User’s Guide*.

Simulink External Mode Interface

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

The Simulink external mode interface includes the following functions:

- **Control** — Control is limited to connecting the Simulink block diagram to the target application, and starting and stopping the target application.
For more information, see “Signal Tracing with Simulink External Mode” in the *xPC Target User’s Guide*.
- **Signal acquisition** — You can use Simulink external mode to establish a communication channel between your Simulink block diagram and your target application. The block diagram becomes a graphical user interface to your target application and Simulink scopes can acquire signal data from the target application. For more information, see “Signal Tracing with Simulink External Mode” in the *xPC Target User’s Guide*.
- **Parameter tuning** — Select external mode, and change parameters in the target 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 PC and replaces the existing parameter while the target application continues to run. For more information, see “Parameter Tuning with Simulink External Mode”.

Note Opening a dialog box for a source block causes Simulink to pause. While Simulink is paused, you can edit the parameter values. You must close the dialog box to have the changes take effect and allow Simulink to continue.

For more information, see “Parameter Tuning with Simulink External Mode” in the *xPC Target User’s Guide*.

Simulink with xPC Target Scope Blocks

An alternative to interactively adding scopes to the target PC is to add xPC Target Scope blocks to your Simulink model. After the download process, these blocks create scopes on the target PC during initialization of the target application. You can choose to display data on either the host PC or target PC. You can also choose to save signal data (log real-time data stream) to a file in the target PC file system and transfer that file to another PC.

Signal acquisition — Add scopes to the target PC by adding xPC Target 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 the xPC Target product, see “Adding an xPC Target Scope Block” on page 3-17, “Entering Parameters for an xPC Target Scope Block” on page 3-22, “Entering Parameters for a File Scope” on page 3-33, and “Signal Tracing with xPC Target Scope Blocks” in the *xPC Target User’s Guide*.

Target PC Command-Line Interface

You can interact with the xPC Target environment through the target PC command window. Enter commands in the command line on the target PC. This interface is useful with stand-alone applications that are not connected to the host PC.

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

- Control — Start and stop the target application, and change the stop time and sample time.

For more information, see “Using the Target PC Command-Line Interface” in the *xPC Target User’s Guide*.

- Signal acquisition — Acquiring signal data is limited to viewing signal traces and signal monitoring on the target PC screen.
- Parameter tuning — You can change only scalar parameters in your model.

Web Browser Interface

If the target PC is connected to a network (TCP/IP), you can use a Web browser to interact with the target application from any computer connected to the network. If the target PC is connected to the host PC with an RS-232 cable, and is using the TCP/IP to RS-232 gateway, you can use a Web browser on the host PC.

The Web browser interface includes the following functions:

- Control — Start and stop the target application, and change the stop time and sample time.

For more information, see “xPC Target Web Browser Interface” in the *xPC Target User’s Guide*.

- Signal acquisition — Signal tracing is limited to viewing a snapshot of a screen captured from the target PC screen. Add target scopes, add or remove signals, and set triggering modes. You can also monitor signal values.

For more information, see “Signal Logging with a Web Browser” in the *xPC Target User’s Guide*.

- Parameter tuning — Change parameters in an HTML form, and then submit that form to make the changes in your target application.

For more information, see “Parameter Tuning with a Web Browser” in the *xPC Target User’s Guide*.

Custom GUI with xPC Target API for Microsoft .NET Framework

Use the .NET API xPC Target framework to develop solutions (applications, human-machine interface (HMI) software, batch runs) that use the xPC Target software. The xPC Target .NET object model provides objects that you can interact with. The xPC Target software arranges the xPC Target .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 “xPC Target API for Microsoft .NET Framework Functions”.

Custom GUI with xPC Target API

Create a GUI application interface to a target application using any development environment that can link in a DLL.

Use the GUI application to control the application, tune parameters, and acquire signal data from a target application. The custom GUI runs on the host PC and communicates with the target application on the target PC using RS-232 or TCP/IP communication. A GUI application can be a console or Windows application using ActiveX[®] components.

For more information, see the *xPC Target API Guide*.

Custom GUI with xPC Target COM API

Create a GUI application that interfaces with a target application using Visual Basic[®] or any development environment that can incorporate COM objects. These COM objects connect graphic elements to parameters for parameter tuning, and they connect signals for acquiring data from your target application. To create a custom GUI application connected to an xPC Target application, use the following process:

- 1 Create a Simulink model.
- 2 Optionally, tag parameters and signals in the Simulink model.
- 3 Build the target application.
- 4 If you tag parameters and signals, build the model-specific COM library.
- 5 Create a GUI application that references the COM library.

For more information, see the *xPC Target API Guide*.

Installation and Configuration

The software environment for xPC Target uses two separate computers. Because of this complexity, installation and configuration are more involved. This chapter includes the following sections:

- “Required Products” on page 2-2
- “Related Products” on page 2-7
- “System Requirements” on page 2-8
- “Installation on the Host PC” on page 2-15
- “xPC Target Explorer” on page 2-22
- “Network Communication” on page 2-26
- “Serial Communication” on page 2-36
- “xPC Target Boot Options” on page 2-43
- “Configuring the xPC Target Environment for 64-Bit MATLAB Systems” on page 2-61
- “Testing and Troubleshooting the Installation” on page 2-71
- “Exporting and Importing xPC Target Explorer Environments” on page 2-79

Required Products

In this section...
“MATLAB” on page 2-2
“Simulink” on page 2-3
“Simulink® Coder” on page 2-4
“MATLAB® Coder” on page 2-4
“C Compiler” on page 2-4
“xPC Target Embedded Option Product” on page 2-6

MATLAB

MATLAB provides a command-line interface for the xPC Target product.

With the xPC Target software, you have full control of the target computer and target application using xPC Target functions and the command-line interface or MATLAB scripts. You use the xPC Target functions for

- Real-time application control — Download, start, and stop the target application.
- Signal acquisition and analysis — Save signal data while the target application is running and analyze the data after the application has completed running, or display signal data while the target application is running in real time.
- Parameter tuning — Change parameters while the target application is running in real time.

Note xPC Target Version 5.0 requires MATLAB Version 7.12.

- MATLAB documentation — For information on using MATLAB and its functions, see the online MATLAB documentation.

Simulink

Simulink provides an environment where you model your dynamic physical system and controller as a block diagram. You create the block diagram by using a mouse to connect blocks and a keyboard to edit block parameters.

You can use the xPC Target product with most Simulink blocks, including discrete-time and continuous-time systems. When you use a continuous-time system and generate code with Simulink Coder, you must use a fixed-step integration algorithm.

xPC Target I/O driver blocks — You can replace the model of your physical system with I/O driver blocks connected to the actual physical system, or you can replace the model of your controller with the actual controller. The xPC Target I/O library supports more than 400 driver blocks. As additional drivers become available, you can download updates from the MathWorks Web site at

<http://www.mathworks.com/support/product/XP/productnews/productnews.html>

The I/O device drivers are written as Simulink C code S-functions.

Note xPC Target Version 5.0 requires Simulink Version 7.7.

Simulink documentation — For information on using Simulink, see the online Simulink documentation. It explains how to connect blocks to build models and change block parameters. It also provides a reference that describes each block in the standard Simulink library.

Simulink Coder

Simulink Coder provides the utilities to convert your Simulink models into C code and then, with a third-party C/C++ compiler, compile the code into a real-time executable.

Features of Simulink Coder include support for multirate systems, as well as loop-rolling and S-function inlining, which allow you to optimize your code for size and efficiency. With the xPC Target product, you can build and download your target application to the target computer using the `build` command in Simulink Coder.

Note xPC Target Version 5.0 requires Simulink Coder Version 8.0 .

Simulink Coder documentation — For information on code generation, see the online Simulink Coder documentation.

MATLAB Coder

MATLAB Coder generates standalone C and C++ from MATLAB code. For more information, see the *MATLAB Coder User's Guide*.

C Compiler

The C compiler creates executable code from the C code generated from Simulink Coder and the C code S-functions you create. The xPC Target product uses this executable code to create an executable image (target application) that runs with the xPC Target kernel on the target computer.

Note To configure your C compiler for the xPC Target software, use the xPC Target Explorer interface (see “Configuring the xPC Target Host PC for Your C Compiler” on page 2-19). Be aware that the `mex -setup` command does not set the C compiler for the xPC Target product.

In addition to the MathWorks products, you need to install a C compiler. The Simulink Coder and xPC Target products support the C compilers listed here:
http://www.mathworks.com/support/compilers/current_release/

xPC Target Embedded Option Product

You do not need this product for rapid prototyping, but with this additional license, you can

- Boot the target PC and automatically launch the application.
- Deploy a stand-alone GUI application that you create with the xPC Target C, COM, or .NET Framework API. Without the xPC Target Embedded Option product, you can create, but not deploy, stand-alone GUI applications running on the host PC to control, change parameters, and acquire signal data from a target application.

The xPC Target Embedded Option product is a separate entity that requires an additional license from MathWorks. Information about the xPC Target Embedded Option product is included with the xPC Target documentation.

Note xPC Target Version 5.0 works with xPC Target Embedded Option Version 5.0.

Related Products

MathWorks provides several products that are relevant to the kinds of tasks you can perform with the xPC Target software.

For more information about any of these products, see either

- The online documentation for that product if it is installed on your system
- The MathWorks Web site at <http://www.mathworks.com/products/xpctarget/related.jsp>.

System Requirements

In this section...
“Introduction” on page 2-8
“Host PC Requirements” on page 2-8
“Target PC Requirements” on page 2-9

Introduction

The hardware and software requirements are different for the host and target computers.

Note that the BIOS settings of a target PC can affect how it works with the xPC Target software. If you experience problems using the xPC Target product with the target or host PC, you should check the system BIOS settings on the target PC. These settings are beyond the control of the xPC Target software. Refer to “Target PC BIOS” in the *xPC Target User’s Guide* for guidelines on BIOS settings.

Host PC Requirements

The host PC is usually your desktop computer where you install the MATLAB, Simulink, Stateflow, Simulink Coder, xPC Target, and xPC Target Embedded Option products. A notebook computer is also a viable host PC.

Software Requirements for the Host PC

The following table lists the minimum software the xPC Target product requires on your host PC. For a list of optional software products related to the xPC Target product, see <http://www.mathworks.com/products/xpctarget/related.jsp>.

Software	Description
32-bit operating system	Windows operating system version supported by MathWorks
MATLAB Product	MATLAB Version 7.12

Software	Description
Simulink Product	Simulink Version 7.7
Simulink Coder Product	Simulink Coder Version 8.0
C language compilers	http://www.mathworks.com/support/-compilers/current_release/
xPC Target Product	xPC Target Version 5.0

Hardware Requirements for the Host PC

The following table lists the minimum resources the xPC Target product requires on the host PC.

Hardware	Description
Communication	Select one of the following methods to communicate with the target PC: <ul style="list-style-type: none"> • One supported Ethernet adapter connected to a network (see “Network Communication” on page 2-26) for supported Ethernet cards and chip sets) • One free serial port (COM1 or COM2) with a 9-pin or 25-pin D-sub connector (see “Serial Communication” on page 2-36 for details)
CPU	Pentium, Athlon, or later
Peripherals	Hard disk drive with 60 MB of free space One 3.5-inch floppy disk drive One CD-RW or DVD-RW drive
RAM	128 MB or more

Target PC Requirements

The target PC must be a 32- or 64-bit PC-compatible system. For example, you can use a second desktop computer or an industrial system like a PC/104 or CompactPCI as the target computer.

Note For target PCs, 64-bit systems run in 32-bit mode.

Software Requirements for the Target PC

The following table lists the minimum software the xPC Target product requires on your target PC system.

Software	Description
Operating system	None. The xPC Target kernel has no effect on any operating system installed on the target PC.
BIOS	PC compatible

Hardware Requirements for the Target PC

The following table lists the minimum resources the xPC Target product requires on the target PC system.

Note Do not use a laptop PC as a target PC.

Hardware	Description
Chip set	PC compatible with UART, programmable interrupt controller, keyboard controller, and counter
Communication	<p>Select one of the following methods to communicate with the host PC:</p> <ul style="list-style-type: none"> • One supported Ethernet adapter connected to a network (see “Network Communication” on page 2-26 for supported Ethernet adapters). • One free serial port (COM1 or COM2) with a 9-pin or 25-pin D-sub connector (see “Serial Communication” on page 2-36 for details). The xPC Target software includes a serial null modem cable for the target PC.
CPU	Intel 386/486/Pentium or AMD K5/K6/Athlon with or without a floating-point coprocessor
Keyboard	<p>Needed to control the target PC when you create stand-alone applications</p> <p>Note that if a keyboard is not connected, the BIOS might display an error message (keyboard failure). With a current BIOS, you can use the BIOS setup to skip the keyboard test.</p>
Monitor	MathWorks recommends using a monitor, but it is not necessary. You can get all the target information using xPC Target functions on the host PC.

Hardware	Description
Peripheral	<p>One 3.5-inch floppy disk or CD/DVD drive. A hard disk drive is not required unless you want to access the target PC file system (for file scopes).</p> <p>Notes:</p> <ul style="list-style-type: none">• You can copy files to a hard disk or flash memory and boot from that device.• If you have a hard disk drive, and you want to access the target PC file system on that drive, see “Logging Signal Data with Target PC Files and File Systems” in the <i>xPC Target User’s Guide</i>. The xPC Target product supports file systems of type FAT-12, FAT-16, or FAT-32.• The hard drive must be a parallel ATA (PATA)/Integrated Device Electronics (IDE) or serial ATA (SATA) drive. For best performance, configure this drive as a primary master.• Ensure that the hard drive is not cable-selected.
RAM	8 MB or more

Random Access Memory (RAM) — The xPC Target product works with PC-compatible computers that use inexpensive dynamic RAM, unlike many non-PC-compatible target computers that use expensive static RAM. You can acquire several megabytes of data during a run depending on how much memory you install in the target PC.

PC-compatible target computers — The xPC Target product supports the following PC-compatible hardware (form factors):

- ISA
- PCI
- PMC
- PC/104 and PC/104+
- PCIe
- CompactPCI

I/O boards — You can install inexpensive I/O boards in the PCI or ISA slots of the target PC. These boards provide a direct interface to the sensors, actuators, or other devices for real-time control or signal processing applications.

The xPC Target software supports the I/O functionality listed in “I/O Driver Support” on page 1-18.

The xPC Target Software and the Target PC BIOS

The BIOS settings of a PC system can affect how the PC works with the xPC Target software. As a general rule, ensure that the target PC BIOS has at least the following settings:

- RS-232 communication — If you are using RS-232 communications, ensure that COM ports are enabled for both host and target PCs. Also, ensure through the BIOS that COM1 has a base address of 3F8 and an IRQ of 4. COM2 must have a base address of 2F8 and an IRQ of 3. These are the default base address values. Do not change these values.
- Plug-and-Play (PnP) operating system — Disable this feature to ensure that the PCI BIOS sets up the plugged-in PCI cards properly. The xPC Target kernel is not a PnP operating system; you must ensure that this feature is disabled or PCI devices will not work on the xPC Target product.
- Power Saving modes — Disable all power saving modes.

- PCI boards — Do not detect PCI boards with class code 0xff in the target PC BIOS. Set this option to Off to enable the BIOS to detect and configure all boards.
- Hyper-threading — If your target PC supports hyper-threading capabilities, do not enable these capabilities. Enabling hyper-threading can degrade the performance of the target PC.

In addition, check the boot up order for the target PC BIOS. You can boot up the target PC using the following methods:

- Boot floppy disk
- CD/DVD bootable ROM
- Dedicated network boot

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

xPC Target Software and Multicore Support

If your target PC has a multicore processor, the xPC Target software enables you to take advantage of the individual cores. Check the target PC BIOS to see if it has a multicore processor.

- If the target PC has a multicore processor, you can configure the xPC Target software to take advantage of the individual cores. See “Configuring Environment Parameters for Target PCs” in the *xPC Target User’s Guide*.

To take advantage of a multicore processor, be sure to disable hyper-threading in the target PC BIOS.

- If the target PC has only a single core processor, you cannot use the multicore capabilities of the xPC Target software.

Installation on the Host PC

In this section...

“Overview” on page 2-15

“xPC Target Turnkey System” on page 2-15

“License Requirements” on page 2-16

“Files on the Host PC Computer” on page 2-16

“Setting Your Initial Working Folder” on page 2-17

“Running MATLAB Remotely” on page 2-18

“Configuring the xPC Target Host PC for Your C Compiler” on page 2-19

Overview

You install the xPC Target software entirely on the host PC. Installing software on the target PC is not necessary. The xPC Target software is distributed on a DVD or as a file you download from the Web.

Note Before you start, ensure that the xPC Target and xPC Target Embedded Option products are not already installed on your host PC. Uninstall both before proceeding if necessary.

After you install the product, you will need to set up the xPC Target environment for either serial or network communication. See “Serial Communication” on page 2-36 or “Network Communication” on page 2-26.

xPC Target Turnkey System

If you have purchased an xPC Target Turnkey system, install your MATLAB products on your system before installing xPC Target Turnkey software. See your system user documentation for further information.

License Requirements

Before you install the xPC Target or the xPC Target Embedded Option products, you must have a valid File Installation Key and License File. The File Installation Key identifies the products you purchased from MathWorks and are permitted to install and use. The License File activates the installation.

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

The xPC Target family of software includes options that you can purchase and add later to the xPC Target environment.

xPC Target Embedded Option product — With the xPC Target Embedded Option product, you can boot the target PC from a device other than a floppy disk or CD/DVD and deploy stand-alone target applications separate from the host PC.

Files on the Host PC Computer

When using the xPC Target software, you might find it helpful to know where files are located:

- MATLAB working folder — Simulink models (`model.mdl`), xPC Target applications (`model.dlm`)

Select a working folder outside the MATLAB root. See “Setting Your Initial Working Folder” on page 2-17.

- Simulink Coder Build folder — The Simulink Coder C code files (`model.c`, `model.h`) are in a subfolder called `modelname_xpc_rtw`.

The xPC Target software uses the directories and files located in `matlabroot\toolbox\rtw\targets\xpc\`

- `target` — Files and functions related to the xPC Target kernel and build process, including drivers to support I/O blocks
- `xpc` — Host PC functions related to all of the xPC Target software, methods for target objects, and methods for scope objects
- `xpcdemos` — Simulink models and MATLAB code demos

Setting Your Initial Working Folder

You should set your MATLAB working folder outside the MATLAB root folder. The default MATLAB root folder is `c:\matlab`.

If your MATLAB working folder is below or inside the MATLAB root, files created by Simulink and Simulink Coder are mixed with the MATLAB directories. This mixing of files could cause file management problems.

From the Desktop Icon

Your initial working folder is specified in the shortcut file you use to start MATLAB. To change this initial folder, use the following procedure:

- 1 Right-click the MATLAB desktop icon or, from the program menu, right-click the MATLAB shortcut.
- 2 Click **Properties**. In the **Start in** text box, enter the folder path you want MATLAB to use initially. Make sure you choose a folder outside the MATLAB root folder.
- 3 Click **OK**, and then start MATLAB. To check your working folder, in the MATLAB Command Window, type

```
pwd
```

From Within MATLAB

To temporarily set your MATLAB working folder, use the following procedure:

- 1 In the MATLAB Command Window, type

```
cd c:\<MATLAB working folder>
```

- 2 To check your working folder, type

```
pwd or cd
```

To permanently set your working folder, see “From the Desktop Icon” on page 2-17.

Running MATLAB Remotely

If you are running MATLAB remotely (accessing MATLAB over the network), register Active X controls before you start xPC Target Explorer.

- 1 In the MATLAB Command Window, type

```
xpc_register_ocx
```

This function registers the Active X controllers that xPC Target Explorer requires.

- 2 Close xPC Target Explorer.
- 3 Close MATLAB.
- 4 Restart MATLAB.
- 5 Restart xPC Target Explorer.

You are now ready to start xPC Target Explorer.

Configuring the xPC Target Host PC for Your C Compiler

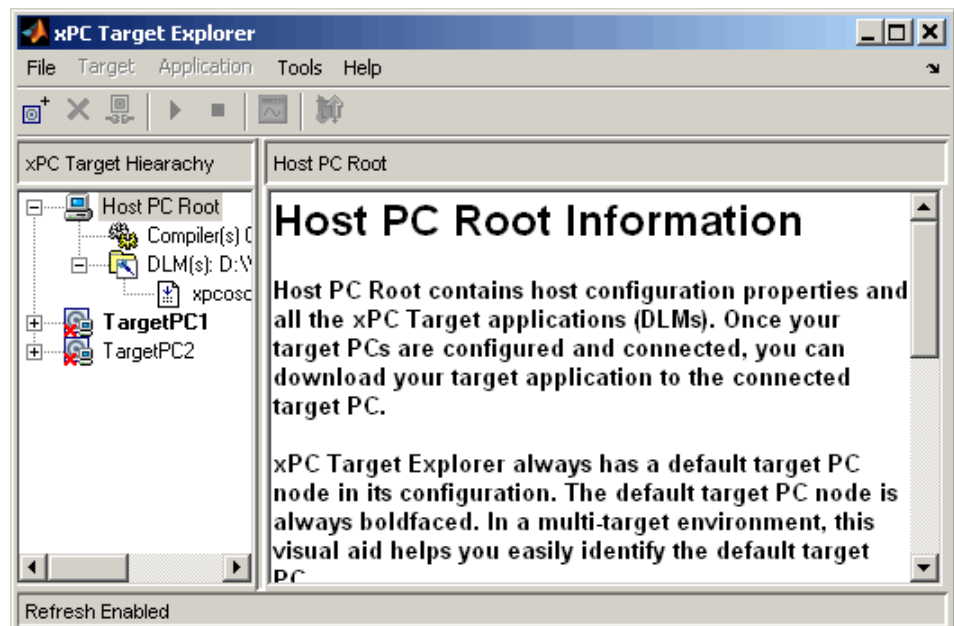
To configure the host PC for your compiler, use xPC Target Explorer.

Note Do not use the `mex -setup` command to set the C compiler for the xPC Target software.

- 1 If xPC Target Explorer is not already open, in the MATLAB Command Window, type

```
xpcexplr
```

The xPC Target Explorer window appears.



xPC Target Explorer always has a default target PC node in its configuration. The default target PC node is always boldfaced. In a

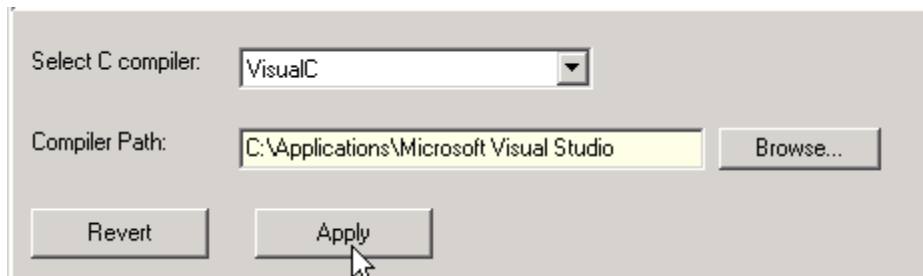
multitarget environment, this visual aid helps you easily identify the default target PC.

- 2 In the xPC Target Explorer window, select the Compiler(s) Configuration node.

In the right pane, the compiler parameters appear.

- 3 At the **Select C Compiler** drop-down list, select the compiler you have installed on the host PC. The examples in this chapter use VisualC.
- 4 Enter the path (or browse) to the compiler for **Compiler Path**. For example,

```
C:\Applications\Microsoft Visual Studio
```



- 5 Click **Apply** to apply the changes.

Note xPC Target Explorer dialogs highlight a field and enable the **Revert** and **Apply** buttons when you make changes. To apply changes, click **Apply**. A prompt is displayed if you leave a dialog without first saving changes. If you want the original entry to be displayed, click the **Revert** button and do not click the **Apply** button. If you click **Apply**, you cannot retrieve the original entries.

Configuring the xPC Target Host PC for Your C Compiler with a Command-Line Interface

To configure the host PC for the C compiler:

- 1 In the MATLAB Command Window, type

```
xpcsetCC('setup')
```

The function queries the host PC for C compilers that the xPC Target environment supports. It returns output like the following:

```
Select your compiler for xPC Target.
```

```
[1] Microsoft Visual C++ Compilers 2008 Professional Edition (SP1) in
```

```
    c:\Program Files (x86)\Microsoft Visual Studio 9.0
```

```
[2] Microsoft Visual C++ Compilers 2010 Professional in
```

```
    C:\Program Files (x86)\Microsoft Visual Studio 10.0
```

```
[0] None
```

```
Compiler:
```

- 2 At the Compiler prompt, enter the number for the compiler you want to use. For example, 2.

The function verifies that you have selected the correct compiler:

```
Verify your selection:
```

```
Compiler: Microsoft Visual C++ Compilers 2010 Professional
```

```
Location: C:\Program Files (x86)\Microsoft Visual Studio 10.0
```

```
Are these correct [y]/n?
```

- 3 Type y or press **Enter** to verify the selection.

The function finishes the dialog.

```
Done...
```

xPC Target Explorer

In this section...
“Introducing xPC Target Explorer” on page 2-22
“The xPC Target Product and Default Target PCs” on page 2-24

Introducing xPC Target Explorer

xPC Target Explorer is a graphical user interface for the xPC Target product. It provides a single point of contact for almost all interactions. Through xPC Target Explorer, you can perform basic operations, such as

- Configure the host PC for the xPC Target software
- Add and configure target PCs for the xPC Target software, up to 64 target PCs
- Create boot disks for particular target PCs
- Connect the target PCs for your xPC Target system to the host PC
- Download a prebuilt target application, DLM, to a target PC
- Start and stop the application that has been downloaded to the target
- Add host, target, or file scopes to the downloaded target application
- Monitor signals
- Add signals to xPC Target scopes and remove them
- Start and stop scopes
- Adjust parameter values for the signals while the target application is running

Note If you run xPC Target software on a 64-bit system, you must use the command-line interface. You must set up the xPC Target environment from the MATLAB command line. For further information, see “Configuring the xPC Target Environment for 64-Bit MATLAB Systems” on page 2-61.

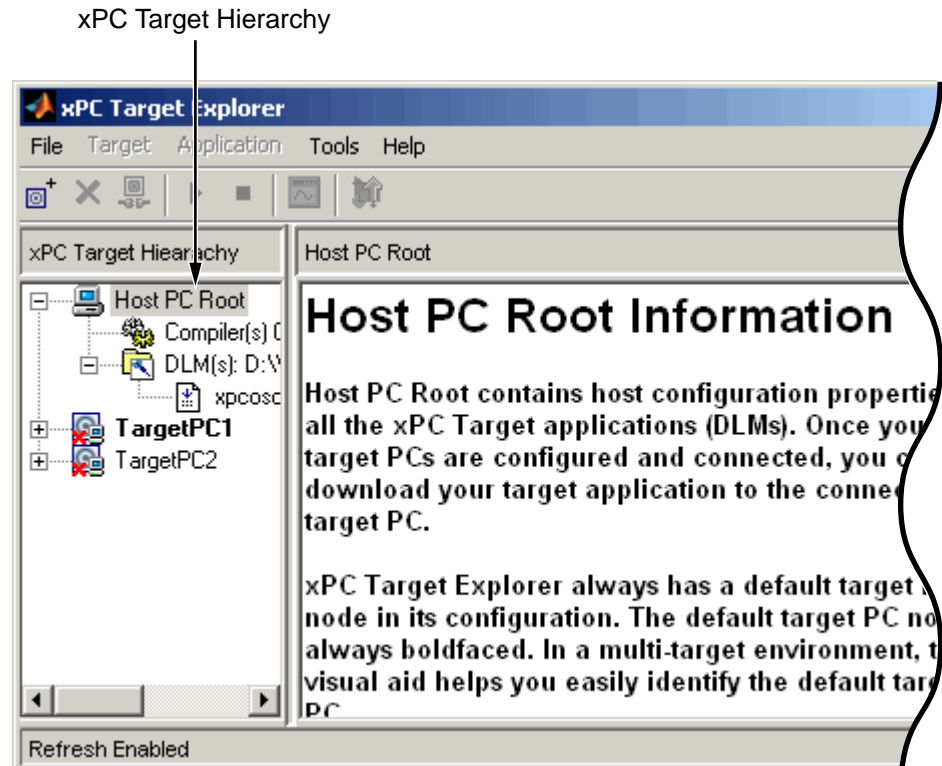
The xPC Target Explorer GUI runs on your xPC Target system host machine.

You can interact with xPC Target Explorer through menus or a toolbar. You can also right-click objects and select actions from the context menu for those objects. The tutorials in the xPC Target documentation describe procedures using mouse operations.

1 In the MATLAB Command Window, type

```
xpcexplr
```

The xPC Target Explorer window opens.



You can also start xPC Target Explorer from the Simulink model window (**Tools > Code Generation > xPC Target Explorer**).

You can dock or undock the xPC Target Explorer window using the arrow in the upper-right corner. Note the contents of the left pane of the xPC Target Explorer. This is the **xPC Target Hierarchy** pane. If you resize or move the window, the xPC Target software remembers the new size and location in subsequent restarts of xPC Target Explorer.

This pane contains all the objects in your xPC Target hierarchy. As you add objects to your system, xPC Target Explorer adds corresponding nodes to the **xPC Target Hierarchy** pane. The foremost node is the **Host PC Root** node. It represents the host PC. The right pane displays information that reflects an item selected in the left pane.

Note that, by default, xPC Target Explorer starts with two target PC objects. The first target PC object is highlighted as the default.

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

The xPC Target Product and Default Target PCs

The following are notes on default target PCs:

- When you first start xPC Target Explorer, it has a default node, **TargetPC1**. You configure this node for a target PC, then connect the node to the target PC. If you later build a target application from a Simulink model, the xPC Target software builds and downloads that application for the default target PC. You can add other target PC nodes and designate one of them as the default target PC instead of the first one. To set a target PC node as the default, right-click that node and select **Set As Default** from the context-sensitive menu. The default target PC node is always boldfaced. In a multitarget environment, this visual aid helps you easily see the target PC you are working with.

If you delete a default target PC node, the target PC node preceding it becomes the default target PC node. The last target PC node is always the default target PC node and cannot be deleted.

- If you want to use the xPC Target command-line interface to work with the target PC, you must indicate which target PC the command is interacting with. If you do not identify a particular target PC, the xPC Target software expects xPC Target Explorer to contain this information.
- The xPC Target product provides a default target PC to help you work with the MATLAB command-line interface, maintain compatibility with previous releases, and work with Simulink external mode, as follows:
 - When you define a default target PC, the MATLAB command-line interface works as in prior releases. For example, when you instantiate the target object constructor `xpc` without any arguments (for example, `tg=xpc`) the constructor uses the environment properties of the default target PC to communicate with the appropriate target PC.
 - The target PC environment object, `xpctarget.targets`, manages collective and individual target PC environments. See “Working with Target PC Environments” in the xPC Target user’s guide documentation for details.
 - The target PC commands `getxpcenv` and `setxpcenv` get and set environment properties for a single target PC system.

Network Communication

In this section...
“Network Communication Overview ” on page 2-26
“Hardware for Network Communication” on page 2-26
“Ethernet Card for a PCI Bus” on page 2-27
“Ethernet Card for an ISA Bus” on page 2-27
“Environment Properties for Network Communication” on page 2-29

Network Communication Overview

This topic describes the establishment of communication between the host PC and target PC using network communications (TCP/IP). For serial communication, see “Serial Communication” on page 2-36.

Hardware for Network Communication

You must install the following hardware before you install the xPC Target software and configure it for network communication:

- Network (Ethernet) adapter card — When using the product with TCP/IP, you must have a network (Ethernet) adapter card correctly installed on both the host PC and the target PC. Be sure to:
 - Connect the host and target computers with an unshielded twisted pair (UTP) cable to your (LAN).
 - Assign a static IP address to the target PC network adapter card.

For the most current network communications requirements, see

http://www.mathworks.com/products/xpctarget/-xPC_Target_Supported_Ethernet_Chipsets.pdf

The host PC network adapter card can have a Dynamic Host Configuration Protocol (DHCP) address. The host PC can be any computer on the network. When using the product with TCP/IP, you must configure the DHCP server to reserve all static IP addresses to prevent these addresses from being assigned to other systems.

You can also directly connect your computers. Use a crossover UTP cable with RJ45 connectors to connect them. Both computers must have static IP addresses. If the host PC has a second network adapter card, that card can have a DHCP address.

- I/O boards — If you use I/O boards on your target PC, you need to install the boards correctly.

Ethernet Card for a PCI Bus

If your target PC has a PCI bus, MathWorks recommends that you use an Ethernet card for the PCI bus. The PCI bus has a faster data transfer rate and requires minimal effort to configure.

To install the PCI bus Ethernet card, use the following procedure:

- 1** Turn off your target PC.
- 2** If the target PC already has an unsupported Ethernet card, remove the card.
- 3** Plug the supported Ethernet card into a free PCI bus slot.
- 4** Connect your target PC Ethernet card to your LAN using an unshielded twisted-pair cable.

Your next task is to set up the xPC Target environment for network communication. See “Environment Properties for Network Communication” on page 2-29.

Ethernet Card for an ISA Bus

Your target PC might not have an available PCI bus slot, or your target PC might not contain a PCI bus (older motherboards, passive ISA backplanes, or PC/104 computers). In these cases, you can use an Ethernet card for an ISA bus.

If you are using an ISA bus, you need to reserve, from the BIOS, an interrupt for this board.

For a list of known compatible network adapter chip families, see

http://www.mathworks.com/products/xpctarget/-xPC_Target_Supported_Ethernet_Chipsets.pdf

To install an ISA bus Ethernet card, use the following procedure:

- 1 Turn off your target PC.
- 2 On your ISA bus card, assign an IRQ and I/O-port base address by moving the jumpers or switches on the card. Write down these settings, because you need to enter them in the xPC Target Explorer.

You should set the IRQ line to 11 and the I/O-port base address to around 0x300. If one of these hardware settings would lead to a conflict in your target PC, select another IRQ or I/O-port base address.

Note If your ISA bus card does not contain jumpers to set the IRQ line and the base address, use the utility on the installation disk supplied with your card to manually assign the IRQ line and base address. Do not configure the card as a PnP-ISA device.

- 3 If the target PC already has an unsupported Ethernet card, remove the card. Plug the compatible network card into a free ISA bus slot.
- 4 Connect the target PC network card to your local area network (LAN) using a coaxial cable or an unshielded twisted-pair cable.

If you use an Ethernet card for an ISA bus within a target PC that has a PCI bus, you must reserve the chosen IRQ line number for the Ethernet card in the PCI BIOS. Refer to your BIOS setup documentation to set up the PCI BIOS.

Your next task is to set up the xPC Target environment for network communication. See “Environment Properties for Network Communication” on page 2-29.

Environment Properties for Network Communication

The xPC Target environment is defined by a group of properties. These properties give xPC Target information about the software and hardware that it works with. You might change some of these properties often, while others you will rarely want to change.

After you have installed the xPC Target software, you can specify the environment properties for the host and target computers. Note that you must specify these properties before you can build and download a target application.


- 1 If xPC Target Explorer is not already started, in the MATLAB Command Window, type

```
xpcexplr
```

The xPC Target Explorer window opens.

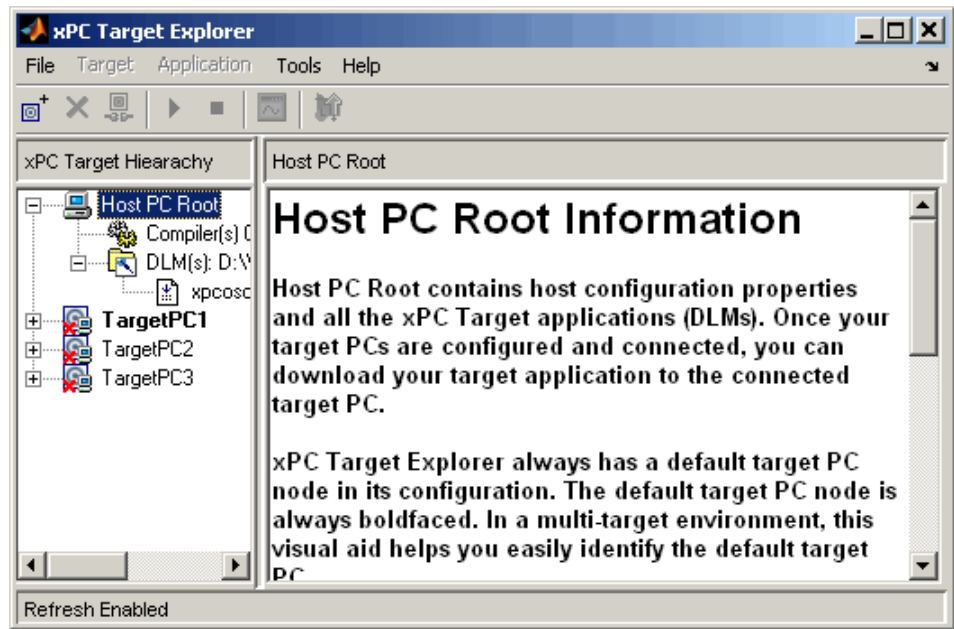
xPC Target Explorer associates network communication environment properties with the target PC.

- 2 In the xPC Target Explorer, right-click the Host PC node.
- 3 Select **Add Target**.

A target PC node named TargetPC1 appears in the **xPC Target Hierarchy**, at the same level as the Host PC node. It appears with the icon  (note the X to denote that the host PC is not connected to the target PC).

- 4 As necessary, repeat and for each additional target PC you want to add to your system.

Additional target PC nodes appear in the **xPC Target Hierarchy**. As you add other target PCs, the PC number is incremented. The following figure illustrates two target PC nodes.



- 5 In the xPC Target Explorer, expand a target PC node.

A Configuration node appears. Under this are nodes for Communication, Settings, and Appearance. The parameters for the target PC node are grouped in these categories.

- 6 Select Communication.

The **Communication Component** pane appears to the right.

- 7 From the **Host target communication** list, select TcpIp.

The pane changes to one that contains only those parameters pertinent to network communication.

- 8 You must enter the network properties with the correct values according to your LAN environment. Ask your system administrator for values for these settings.

- **Target PC IP address** — This is the IP address for your target PC. An example of an IP address is 192.168.0.10.
- **LAN subnet mask address** — This is the subnet mask address of your LAN. An example of a subnet mask address is 255.255.255.0.

Alternatively, you can obtain the LAN subnet mask address from the Network Connections dialog box on your host PC. Depending on your Windows platform, you can access this dialog box in a number of ways. For example, on a Windows XP Professional system, you can use this sequence:

1 Select **Start > Settings > Control Panel**, then double-click **Network Connections**.

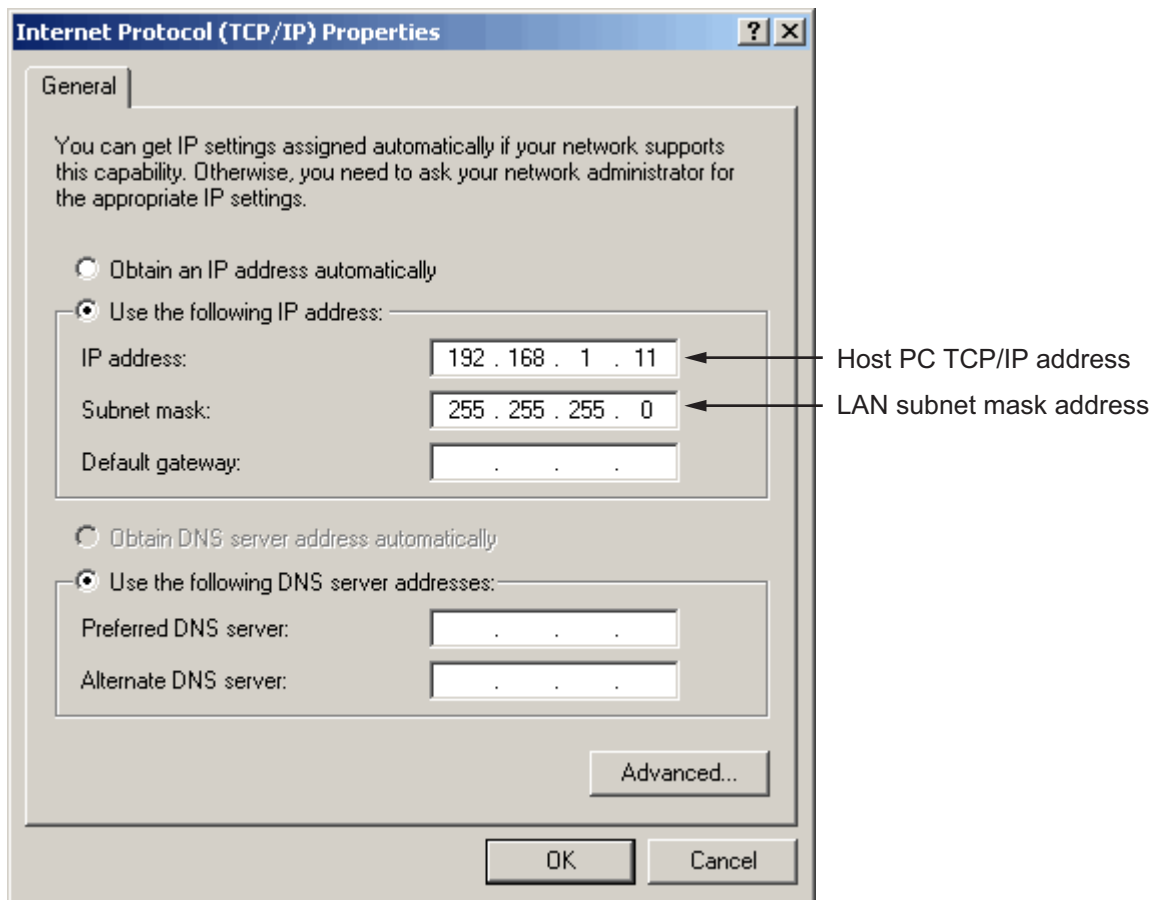
2 Right-click **Local Area Connection**, then select **Properties**.

3 Select **Internet Protocol (TCP/IP)**, then click **Properties**.

If your computers connect with a crossover cable, you might have a dialog box like the following. You can obtain your subnet mask address and TCP/IP gateway address from this dialog box.

Note The TCP/IP address is for your host PC, not your target PC. You still need to get the target PC TCP/IP address for your target PC from your system administrator.

The default gateway address is blank in this dialog box. However, in the xPC Target Explorer, you must enter 255.255.255.255 for the gateway value in the **TCP/IP gateway address** property.



9 Optionally, enter the following properties, depending on your specific circumstances:

- **TCP/IP target port** — This property is set by default to 22222. This value should not cause any problems, because this number is higher than the reserved area (telnet, ftp, ...) and it is only relevant on the target PC. If necessary, you can change this property value to any value higher than 20000 and less than 65536.
- **TCP/IP gateway address** — This property is set by default to 255.255.255.255. This means that you do not use a gateway to connect

to your target PC. If you connect your computers with a crossover cable, leave this property as 255.255.255.255.

If you communicate with the target PC from within your LAN, you might not need to define a gateway and change this setting.

If you communicate from a host PC located in a LAN different from your target PC, you need to define a gateway and enter its IP address. This is especially true if you want to work over the Internet. Ask your system administrator for the IP address of the appropriate gateway.

- 10 Enter the following properties specific to the Ethernet card on your target PC:
 - **TCP/IP target driver** — From the list, select NE2000, SMC91C9X, I82559, RTLANCE, R8139, 3C90x, NS83815, or I8254x. This property is set by default to Auto. For a crossover cable connection, select I82559.
 -

Note To allow the software to determine your TCP/IP target driver, select Auto. If no supported Ethernet card exists in your target PC, the software returns an error.

TCP/IP target bus type — This property is set by default to PCI. If **TCP/IP target bus type** is set to PCI, then the properties **TCP/IP target ISA memory port** and **TCP/IP target ISA IRQ number** have no effect on TCP/IP communication and are disabled (grayed out). If you are using an ISA bus Ethernet card, set **TCP/IP target bus type** to ISA and enter values for **TCP/IP ISA memory port** and **TCP/IP target ISA IRQ number**.

- **TCP/IP target ISA memory port** and **TCP/IP target ISA IRQ number** — If you are using an ISA bus Ethernet card, you must enter values for the properties **TCP/IP target ISA memory port** and **TCP/IP target ISA IRQ number**. The values of these properties must correspond to the jumper settings or ROM settings on your ISA bus Ethernet card.

- 11** If the target PC has multiple Ethernet cards, type the following commands to specify which card to use. These commands assume that the target PC name is `nonDefaultTarget`.

```
allTargets = xpctarget.targets;
myTargetEnv = allTargets.Item('nonDefaultTarget');
set(myTargetEnv, 'EthernetIndex', '#');
```

indicates a single digit to specify the index number for the Ethernet card. For example, `set(myTargetEnv, 'EthernetIndex', '2');` selects the Ethernet card with index number 2 as the target PC card.

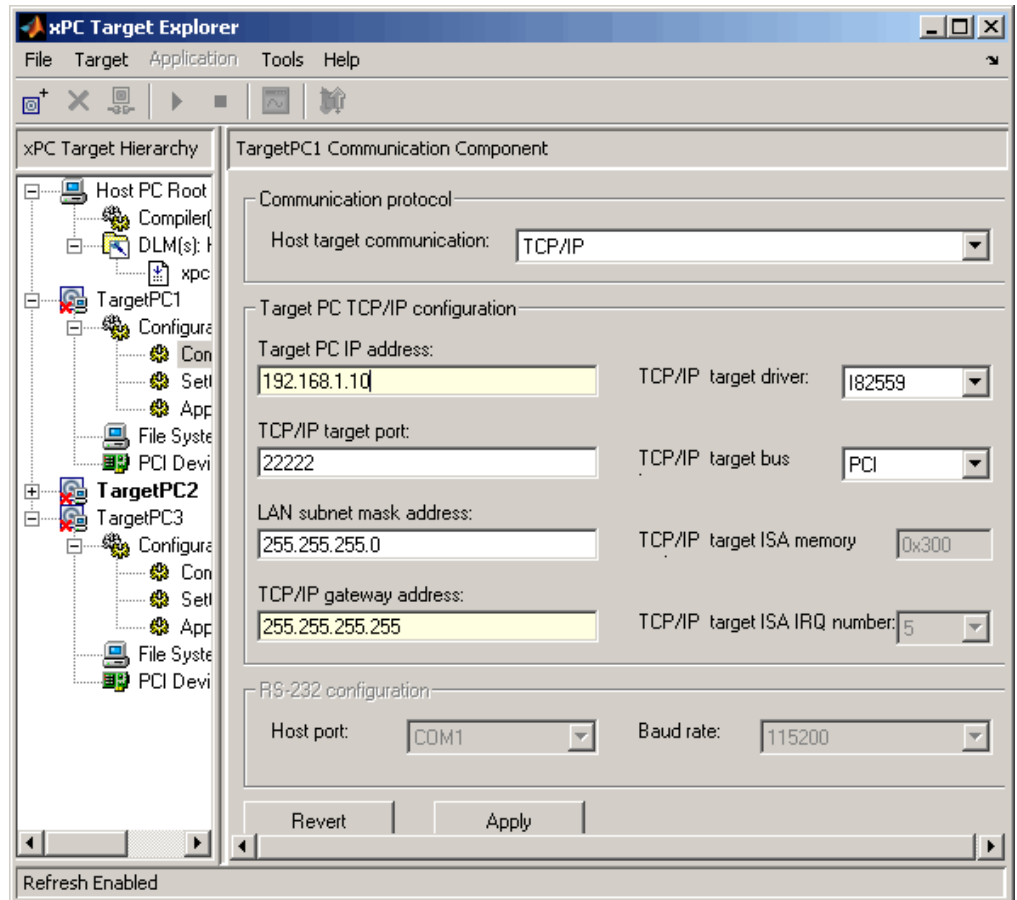
This command ensures that upon booting the target, the kernel selects the appropriate Ethernet card instead of selecting the default card with index number 0 as the target PC card.

Alternatively, if you have a single target PC system, you can use `setxpcenv('EthernetIndex', '#')`. For example, `setxpcenv('EthernetIndex', '2')` selects the Ethernet card with index number 2 as the target PC card.

- 12** Repeat step 5 to 10 for any target PC for which you have a network connection between the host PC and target PC.

The xPC Target software updates the environment with new properties as you enter them.

The following figure illustrates the **Communication Component** pane for a network connection.



For more information on the xPC Target environment, see “Software Environment and Demos” in the *xPC Target User’s Guide*.

Your next task is to create a target boot disk. See “Booting Target PCs from Boot Floppy Disk” on page 2-51.

Serial Communication

In this section...
“Serial Communication Overview” on page 2-36
“Hardware for Serial Communication” on page 2-36
“Environment Properties for Serial Communication” on page 2-37

Serial Communication Overview

This topic describes the establishment of communication between the host PC and target PC using serial communications (RS-232). For network communication, see “Network Communication” on page 2-26.

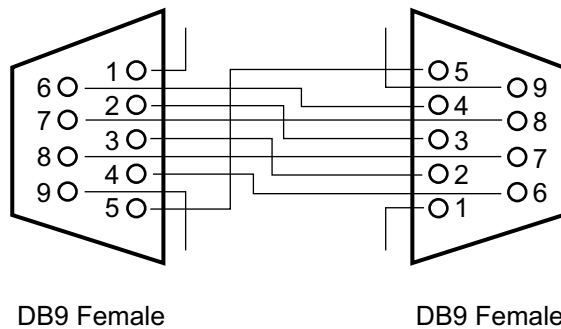
Hardware for Serial Communication

Before you install the xPC Target software and configure it for serial communication, you must install the following hardware:

- Null modem cable — Connect the host and target computers with the null modem cable supplied by MathWorks with the xPC Target software. You can use either the COM1 or COM2 port.
- I/O boards — If you use I/O boards on the target PC, you need to install the boards correctly. See the manufacturer’s literature for installation instructions.

Null Modem Cable Wiring

xPC Target software ships with a null modem cable that you can use to connect the host and target computers for serial communications. The following diagram illustrates the wiring for this cable for a 9-pin DB9 connector.



Environment Properties for Serial Communication

The xPC Target environment is defined by a group of properties. These properties give to the xPC Target software information about the software and hardware products that it works with. You might change some of these properties often, while others you will rarely want to change.

After you have installed the xPC Target product, you can specify the environment properties for the host and target computers. Note that you must specify these properties before you can build and download a target application.

The following procedure describes how to set up serial communication environment properties through the xPC Target Explorer.

Note the following:

- If you have a serial connection between your host PC and target PC, and you use a baud rate that is less than the maximum possible baud rate, you might experience communication failures. If you do experience these failures, use a baud rate greater than 19200.
- If you have an RS-232 connection, you might not want to use host scopes and a scope viewer on the host PC (Host Scope Viewer) to acquire and display large blocks of data. The slowness of the RS-232 connection causes large delays in performance for large blocks of data.

- 1 If xPC Target Explorer is not already started, in the MATLAB Command Window, type


```
xpcexplr
```

The xPC Target Explorer window opens. Note that xPC Target Explorer automatically provides a default target PC node, **TargetPC1**.

xPC Target Explorer associates serial communication environment properties with the target PC.

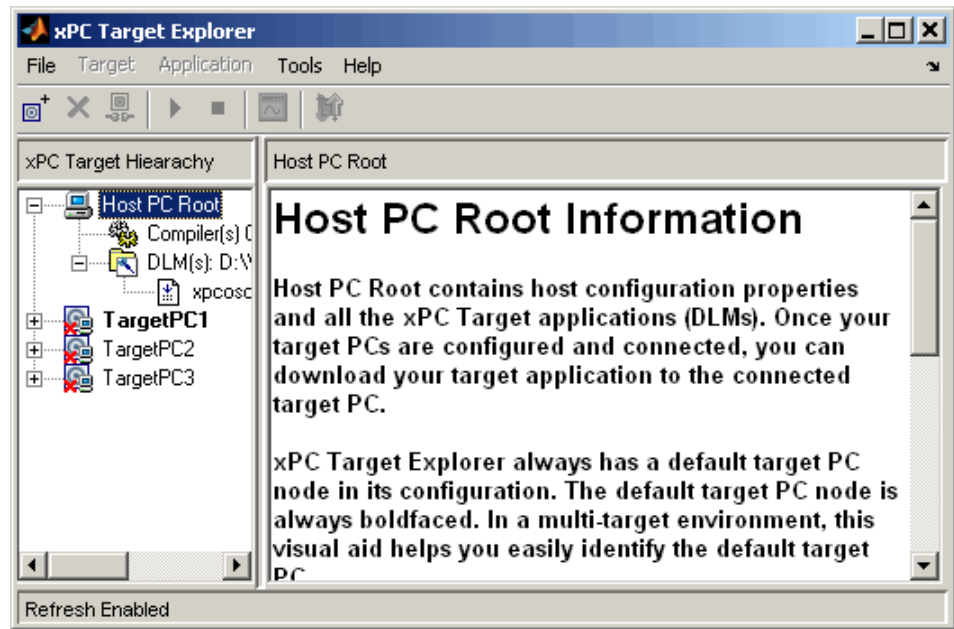
- 2 In the xPC Target Explorer, right-click the **Host PC** node.

- 3 Select **Add Target**.

A target PC node, named **TargetPC2**, appears in the **xPC Target Hierarchy**, at the same level as the **Host PC** node. It appears with the icon  (note the X to denote that the host PC is not connected to the target PC).

- 4 As necessary, repeat step 2 and step 3 for each additional target PC you want to add to your system.

Additional target PC nodes appear in the **xPC Target Hierarchy**. As you add other target PCs, the PC number is incremented. The following figure illustrates two target PC nodes.



- 5 In the xPC Target Explorer, expand a target PC node.

Configuration, File System, and PCI Devices nodes appear. You work with the Configuration node to configure the target PC node for a target PC. The File System node contains the contents of a target PC file system. PCI Devices lists all PCI devices detected on the target PC. In this procedure, you work with the Configuration node.

Under the Configuration node are nodes for Communication, Settings, and Appearance. The parameters for the target PC node are grouped in these categories. These nodes make up the target environment settings.

- 6 Select Communication.

The **Communication Component** pane appears to the right.

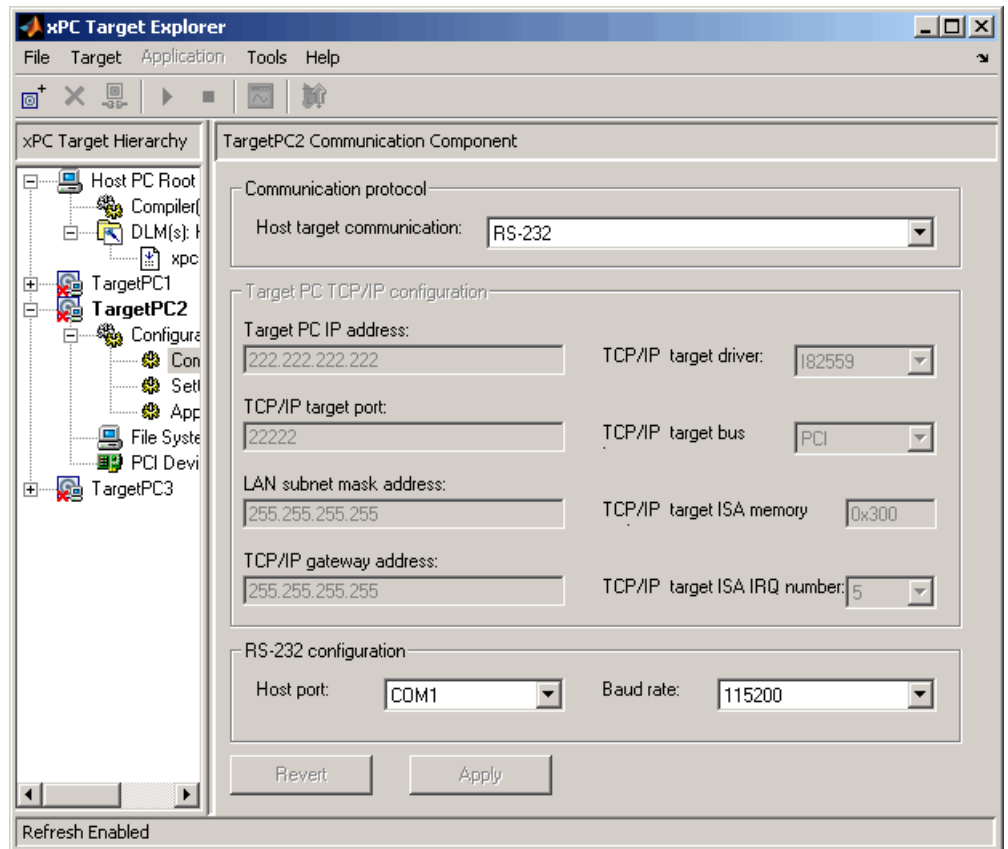
Note When you first select a subnode under a target PC node, the target PC node becomes boldfaced. In a multitarget environment, this visual aid helps you easily see the target PC you are working with.

- 7 From the **Host target communication** list, select RS232.

The pane changes to one that contains only those parameters pertinent to serial communication.

- 8 From the **Host port** list, select either COM1 or COM2 for the connection on the host PC. The xPC Target software determines the COM port you use on the target PC automatically.
- 9 From the **Baud rate** list, select the baud rate for the serial connection between the host PC and this target PC. The default is 115200. Note that for optimal performance, you should select the highest possible serial connection baud rate for the xPC Target software.
- 10 Repeat step 5 to step 9 for any target PC for which you have a serial connection between the host PC and target PC.

The following figure illustrates the xPC Target Explorer settings for the serial connection of one target PC.



You do not have to exit and restart MATLAB after making changes to the xPC Target environment, even if you change the communication between the host and target from RS-232 to TCP/IP. However, you do have to recreate the target boot disk and rebuild the target application from the Simulink model.

For more information on the xPC Target environment, see "Software Environment and Demos" in the xPC Target User's Guide.

Your next task is to create a target boot disk. See “Booting Target PCs from Boot Floppy Disk” on page 2-51.

xPC Target Boot Options

In this section...

“Introduction” on page 2-43

“Booting Target PCs from CD or DVD” on page 2-45

“Booting Target PCs from Boot Floppy Disk” on page 2-51

“Booting Target PCs Within a Dedicated Network” on page 2-54

Introduction

You can boot your target PC with the xPC Target kernel using one of the following ways from the xPC Target Explorer:

- **CD Boot** — Boot the target PC with a CD/DVD bootable ROM
- **Boot Floppy** — Boot the target PC with a 3.5-inch floppy.
- **Network Boot** — Boot the target PC from a dedicated network boot image.

Target boot disks and boot images include the xPC Target kernel specific for either serial or network communication. If you installed the xPC Target Embedded Option, and you select stand-alone mode, the target boot files include the target application (see “Embedded Option” in the *xPC Target User’s Guide*). If you want to boot from a device other than a CD, DVD, 3.5-inch disk, or dedicated network boot image, see “Booting from a DOS Device” in the *xPC Target User’s Guide*.

Note Before you create a target boot disk, ensure that you have write permission for your current working folder. You cannot create a boot disk otherwise.

xPC Target Turnkey

xPC Target Turnkey systems have necessary software preinstalled. See your xPC Target Turnkey system user documentation for further information.

Before You Boot

Ensure that you have appropriately configured your xPC Target system before you create your boot disk or boot image. At a minimum, ensure that you have performed the following configurations. You can optionally set the other xPC Target Explorer configuration options; however, their default values should suffice.

- Confirm that the boot tab on the Configuration pane is set to your desired boot mode:
 - **CD Boot**
 - **Boot Floppy**
 - **Network Boot**

For information on other boot modes, see the following topics in the *xPC Target User's Guide*:

- “Booting from a DOS Device”, for DOSLoader mode
- “Embedded Option”, for StandAlone mode
- Check the C compiler specification (see “Configuring the xPC Target Host PC for Your C Compiler” on page 2-19).
- If you are using TCP/IP communication, check that your network connections are correct. Also check the xPC Target Explorer settings (see “Network Communication” on page 2-26).
- If you are using serial communication, check that your physical connections are correct. Also check the xPC Target Explorer settings (see “Serial Communication” on page 2-36).
- Check your target PC BIOS settings (see “The xPC Target Software and the Target PC BIOS” on page 2-13 in this chapter and “BIOS Settings”).

Booting Target PCs from CD or DVD

- “Creating a Boot CD/DVD with xPC Target Explorer” on page 2-45
- “Creating a Boot CD/DVD with a Command-Line Interface” on page 2-48

You use the target boot CD or DVD to load and run the xPC Target kernel. After you make changes to the xPC Target environment properties, you must create a target boot CD or DVD. This topic assumes you are using default environment parameter settings for the boot CD or DVD creation. If this is not the case, see “Configuring Environment Parameters for Target PCs” in the *xPC Target User’s Guide* for further details.

Creating a Boot CD/DVD with xPC Target Explorer

Use the following procedure to create a boot CD or DVD for the current xPC Target environment. This procedure describes how to create a target boot CD for the target TargetPC1. Before you start:

- Ensure that you have an empty, writable CD or DVD.
- Ensure that you have a CD/DVD-RW drive.
- Ensure that you can create a boot CD or DVD. You can create a boot CD or DVD in one of the following ways:
 - The xPC Target Explorer **Create CD Boot Image** can create a boot CD or DVD for you. To use this capability, your host PC must have one of the following Windows systems:
 - Microsoft Windows 7
 - Microsoft Windows Vista™
 - Microsoft Windows XP Service Pack 2 or 3 with Image Mastering API v2.0 (IMAPIv2.0), available at <http://support.microsoft.com/kb/KB932716>.
 - You can use third-party CD/DVD writing software to write ISO image files. Use this method if you do not have Microsoft Windows 7, Microsoft Windows Vista, or Microsoft Windows XP Service Pack 2 or 3.

Note Standard Microsoft Windows software (such as Windows Explorer or Windows Media Player) does not write ISO image files to CD/DVD.

Warning Writing the CD ISO image to a CD or DVD is not the same as copying the ISO image to a CD or DVD. When you write an ISO image to a CD or DVD, you create a bootable CD or DVD from the ISO image by burning the image to the CD or DVD. When you copy the ISO image, you just copy the ISO image to the CD or DVD as data; you cannot use a copied CD or DVD as a boot disk.

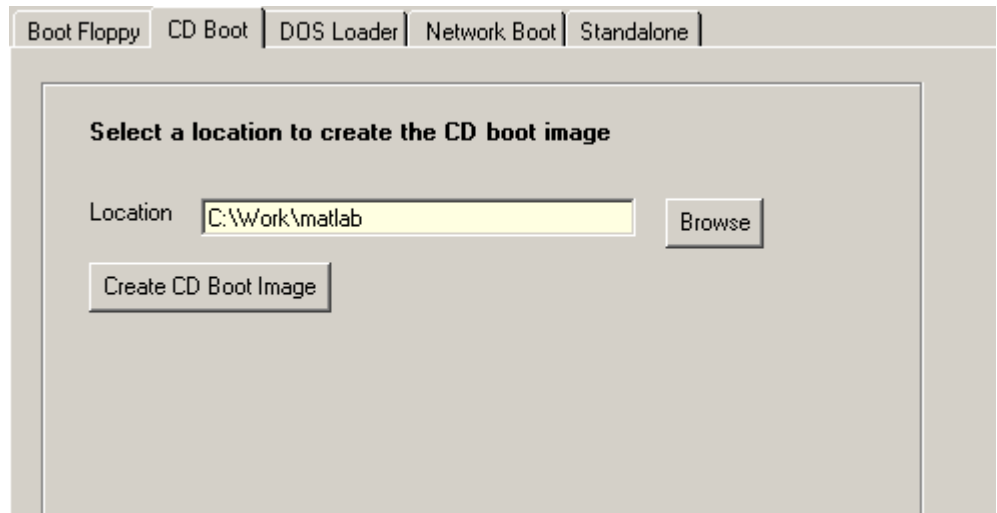
- 1 Insert the empty CD or DVD in the host PC.
- 2 If the xPC Target Explorer is not open, open it now. At the MATLAB Command Window, type

```
xpcexplr
```

- 3 In the xPC Target Explorer **xPC Target Hierarchy** pane, select a target PC Configuration node. For example, select the Configuration node for TargetPC1.

A **TargetPC1 Configuration** pane appears in the rightmost pane. This pane contains a series of tabs.

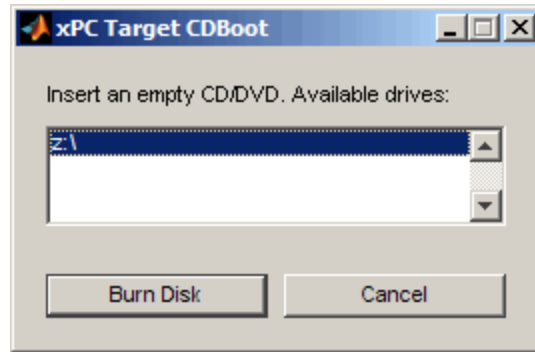
- 4 Select the **CD Boot** tab.
- 5 In the location parameter, enter a path in which you want xPC Target Explorer to write the xPC Target CD/DVD boot ISO image. For example, enter `C:\Work\matlab`.



- 6 Click the **Apply** button.
- 7 Click the **Create CD Boot Image** button.

The software creates a CD/DVD image file named `cdboot.iso` in this location.

- 8 Perform one of the following depending on your software:
 - If you have Microsoft Windows 7, Microsoft Windows Vista, or Microsoft Windows XP Service Pack 2 or 3 with Image Mastering API v2.0 (IMAPIv2.0), xPC Target Explorer prompts you to insert a CD/DVD.



Select the appropriate drive, insert the CD or DVD, then click **Burn Disk**.

- If you do not have Microsoft Windows 7, Microsoft Windows Vista, or Microsoft Windows XP Service Pack 2 or 3 with Image Mastering API v2.0 (IMAPIv2.0), use your third-party CD creation software to write the `cdboot.iso` image to the empty CD/DVD.
- 9 Insert the bootable CD/DVD into your target PC CD/DVD drive and reboot that PC.

Your next task is to install the software on the target PC and test your installation. See “Testing and Troubleshooting the Installation” on page 2-71.

Creating a Boot CD/DVD with a Command-Line Interface

You use the boot CD/DVD to load and run the xPC Target kernel. After you make changes to the xPC Target environment properties, you must create a CD/DVD bootable ROM. Before you start:

- Ensure that you have an empty, writable CD or DVD.
- Ensure that you have a CD/DVD-RW drive.
- Ensure that you can create a bootable CD or DVD. You can create a boot CD or DVD in one of the following ways:
 - The xPC Target software can create a boot CD or DVD for you. To use this capability, your host PC must have one of the following Windows systems:

- Microsoft Windows 7
- Microsoft Windows Vista
- Microsoft Windows XP Service Pack 2 or 3 with Image Mastering API v2.0 (IMAPIv2.0), available at <http://support.microsoft.com/kb/KB932716>.
- Third-party CD/DVD writing software can write ISO image files for you. Use this method if you do not have Microsoft Windows 7, Microsoft Windows Vista, or Microsoft Windows XP Service Pack 2 or 3.

Note Standard Microsoft Windows software (such as Windows Explorer or Windows Media Player) does not write ISO image files to CD/DVD.

To create a boot CD/DVD for the TargetPC2 environment:

1 Insert the empty CD or DVD in the host PC.

2 In the MATLAB window, type

```
tgs=xpctarget.targets
tgs.makeDefault('TargetPC2')
env=tgs.Item('TargetPC2')
```

3 Ensure that the following xPC Target properties are set as follows:

- TargetBoot — CDBoot
- CDBootImageLocation — Your host PC CD/DVD disk drive location

4 If these properties are not set with the correct values, set them. For example:

```
env.TargetBoot='CDBoot'
env.CDBootImageLocation='c:\work\xpc\cdimage'
```

5 In the MATLAB Command Window, type

```
xpcbootdisk
```

The xPC Target software displays the following message and creates the CD/DVD boot ISO image.

```
Current boot mode: CDBoot
CD boot image is successfully created
```

6 Perform one of the following, depending on your software:

- If you have Microsoft Windows 7, Microsoft Windows Vista, or Microsoft Windows XP Service Pack 2 or 3 with Image Mastering API v2.0 (IMAPIv2.0), xpcbootdisk prompts you to insert a CD/DVD.

```
Insert an empty CD/DVD. Available drives:
[1] z:\
[0] Cancel Burn
```

Select the appropriate drive, insert the CD or DVD, then press the **Enter** key.

- If you do not have Microsoft Windows 7, Microsoft Windows Vista, or Microsoft Windows XP Service Pack 2 or 3 with Image Mastering API v2.0 (IMAPIv2.0), use your third-party software to write the `cdboot.iso` image to the empty CD/DVD.

7 When the CD/DVD drive stops, remove the CD/DVD.

Your next task is to install the software on the target PC and test your installation. See “Testing and Troubleshooting the Installation” on page 2-71.

Alternatively, if you have a single target PC system, you can use the `setxpcenv` function.

Booting Target PCs from Boot Floppy Disk

You use the 3.5-inch target boot disk to load and run the xPC Target kernel. After you make changes to the xPC Target environment properties, you need to create or update a 3.5-inch boot disk. Note that this topic assumes you are using default environment parameter settings for the target boot disk creation. If this is not the case, see “Configuring Environment Parameters for Target PCs” in the *xPC Target User’s Guide* for further details.

Creating a Target Boot Disk with xPC Target Explorer

To create a target boot disk for the current xPC Target environment, use the following procedure. This procedure describes how to create a target boot disk for the target TargetPC2. Alternatively, see “Creating a 3.5-Inch Target Boot Disk with a Command-Line Interface” on page 2-53.

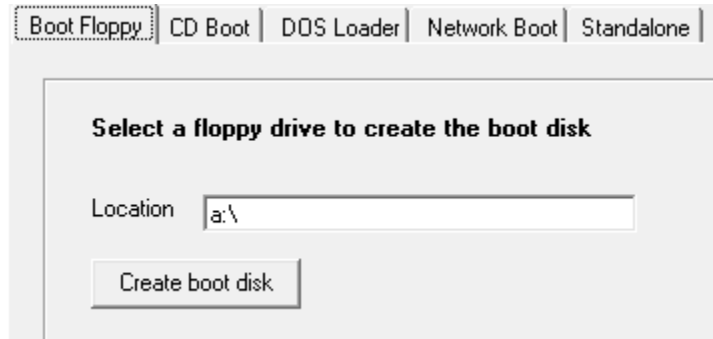
- 1 If the xPC Target Explorer is not open, open it now. At the MATLAB Command Window, type

```
xpcexplr
```

- 2 In the xPC Target Explorer **xPC Target Hierarchy** pane, select a target PC Configuration node. For example, select the Configuration node for TargetPC2.

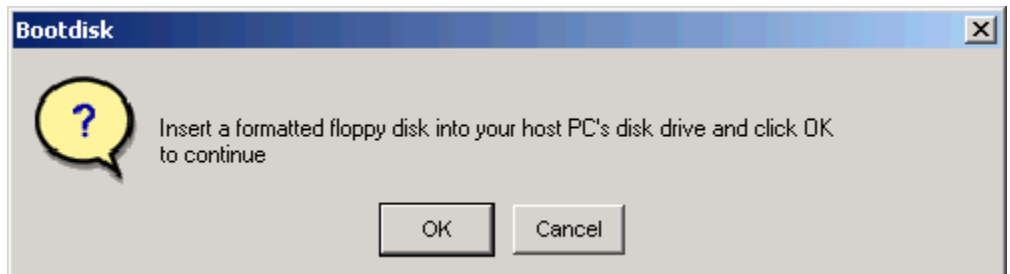
A configuration pane for that target PC appears in the rightmost pane.

- 3 From the tab list, select the **Boot Floppy** tab.



- 4 As necessary, change the drive letter of the 3.5-inch drive to a valid floppy drive. Include the backslash, such as a:\.
- 5 Click the **Apply** button.
- 6 Click the **Create boot disk** button.

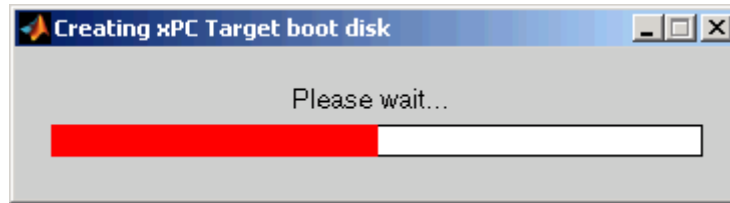
The following message box opens.



- 7 Insert a formatted 3.5-inch floppy disk into the host PC disk drive, and then click **OK**.

All data on the disk is erased as the xPC Target software writes the xPC Target kernel and other required files to the 3.5-inch disk.

The xPC Target software displays the following dialog box while creating the boot disk. The process takes about 1 to 2 minutes.



- 8 When the disk drive stops, remove the disk.
- 9 Insert the boot disk into your target PC disk drive and reboot that PC.

Your next task is to install the software on the target PC and test your installation. See “Testing and Troubleshooting the Installation” on page 2-71.

Creating a 3.5-Inch Target Boot Disk with a Command-Line Interface

You use the target boot disk to load and run the xPC Target kernel. After you make changes to the xPC Target environment properties, you need to create or update a boot disk. The following makes changes to the TargetPC2 environment.

- 1 In the MATLAB window, type

```
tgs=xpctarget.targets
tgs.makeDefault('TargetPC2')
env=tgs.Item('TargetPC2')
```

- 2 Ensure that the following xPC Target properties are set as follows:

- TargetBoot — BootFloppy
- BootFloppyLocation — Your host PC 3.5-inch disk drive location

- 3 If these properties are not set with the correct values, set them. For example:

```
env.TargetBoot='BootFloppy'
env.BootFloppyLocation='a:'
```

- 4 In the MATLAB Command Window, type

```
xpcbootdisk
```

The xPC Target software displays the following message.

```
Current boot mode: BootFloppy
Insert a formatted floppy disk into your host PC's
disk drive and press a key to continue
```

- 5 Insert a formatted floppy disk into the host PC disk drive, and then press any key.

The write procedure starts and, while creating the boot disk, the MATLAB Command Window displays the following status information.

```
Creating xPC Target boot disk ... Please wait
xPC Target boot disk successfully created.
```

Your next task is to install the software on the target PC and test your installation. See “Testing and Troubleshooting the Installation” on page 2-71.

Alternatively, if you have a single target PC system, you can create a target boot disk with the `setxpcenv` function.

Your next task is to install the software on the target PC and test your installation. See “Testing and Troubleshooting the Installation” on page 2-71.

Booting Target PCs Within a Dedicated Network

This topic describes how to boot target PCs on a dedicated network. You do not need a boot disk. You do need to set up the host PC and target PC. This topic assumes that you know how to:

- Set up a dedicated network.
- Use the xPC Target Explorer to configure the target PC and host environments. You should be familiar with the following sections:
 - “Configuring the xPC Target Host PC for Your C Compiler” on page 2-19
 - “xPC Target Explorer” on page 2-22
 - “Network Communication” on page 2-26

- “Booting Target PCs from Boot Floppy Disk” on page 2-51

Caution Do not boot a target PC on a corporate or nondedicated network. Doing so might interfere with dynamic host configuration protocol (DHCP) servers, which will cause problems with the network.

Setting Up the Target PC

- 1 Identify the target PC that you want to boot over the dedicated network.
- 2 Install the Ethernet card.
 - Ensure that:
 - The xPC Target product supports your Ethernet card (see “Hardware for Network Communication” on page 2-26).
 - Your Ethernet card has a boot ROM that is compatible with the Preboot eXecution Environment (PXE) specification.
- 3 Connect the host PC and the target PCs within the dedicated network. For example, connect one end of a crossover cable to the dedicated network card of the host PC and connect the other end of this cable to the dedicated network card of the target PC.
- 4 Turn on the target PC.
- 5 Enter BIOS and set up the target PC for a LAN or network boot. If there is a boot order, consider setting the boot order so that the removable/boot floppy disk is the first option and the LAN is the second. Doing so ensures that if there is no xPC Target boot disk in the target PC, you can still boot the target PC from a kernel on the network.

Configuring for Network Booting

This procedure is similar to configuring a target boot disk. If you have previously created a target boot disk, you might not need to perform this procedure. However, you should still read the following instructions to ensure that your configuration is appropriate for booting a target PC in the dedicated network. You can configure multiple target PCs for your network.

- 1** Ensure that the host PC has a network card available for the dedicated network. As necessary, insert a second network card and configure that card for the dedicated network. This step includes assigning the host PC a unique IP address (for example, 10.10.10.10) in the same subnet as the target PC.
- 2** If the xPC Target Explorer is not open, open it now. At the MATLAB Command Window, type


```
xpcexplr
```
- 3** Add a target PC, for example, TargetPC3 (if necessary).
- 4** In the **Communication Component** pane for TargetPC3,
 - a** In the **Host target communication** field, select TCP/IP.
 - b** Enter a target PC IP address in the dedicated network, for example, 10.10.10.11. Ensure that the subnet of this IP address is the same as the host PC, otherwise Network Boot will fail.
 - c** Enter appropriate values for the remaining fields.
 - d** Click the **Apply** button.
- 5** In the **TargetPC3 Configuration** pane, select **Network Boot**.

The screenshot shows a configuration window with a tabbed interface. The 'Network Boot' tab is selected. The main area is titled 'Create Netboot Image and select the discovery mode for the target PC'. It displays 'IP Address: 10.10.10.11'. Below this is a 'Create Network Boot Image' button and a timestamp 'Last created on 12-May-2008 11:11:47'. A section titled 'Target PC Ethernet Configuration' contains a 'MAC Address: Empty' field with a 'Reset MAC Address' button. Two radio buttons are present: 'Auto (MAC Address will be obtained next time the targetPC is restarted)' which is selected, and 'Manual (Use the following MAC Address)'. Below the manual option are six empty text boxes for entering a hexadecimal MAC address. At the bottom of the window are 'Revert' and 'Apply' buttons.

The **Target PC Ethernet Configuration** section of the configuration pane allows you to either associate a physical target PC MAC address with the xPC Target Explorer target PC name, or allow the software to automatically find target PC MAC addresses. If you want to associate your physical target PC MAC address,

- a** Click **Manual**.
 - b** In the six fields, enter the physical target PC MAC address (in hexadecimal).
- 6** Click the **Create Network Boot Image** button.

The software creates and starts a network boot server process on the host PC. You will boot the target PC using this process.

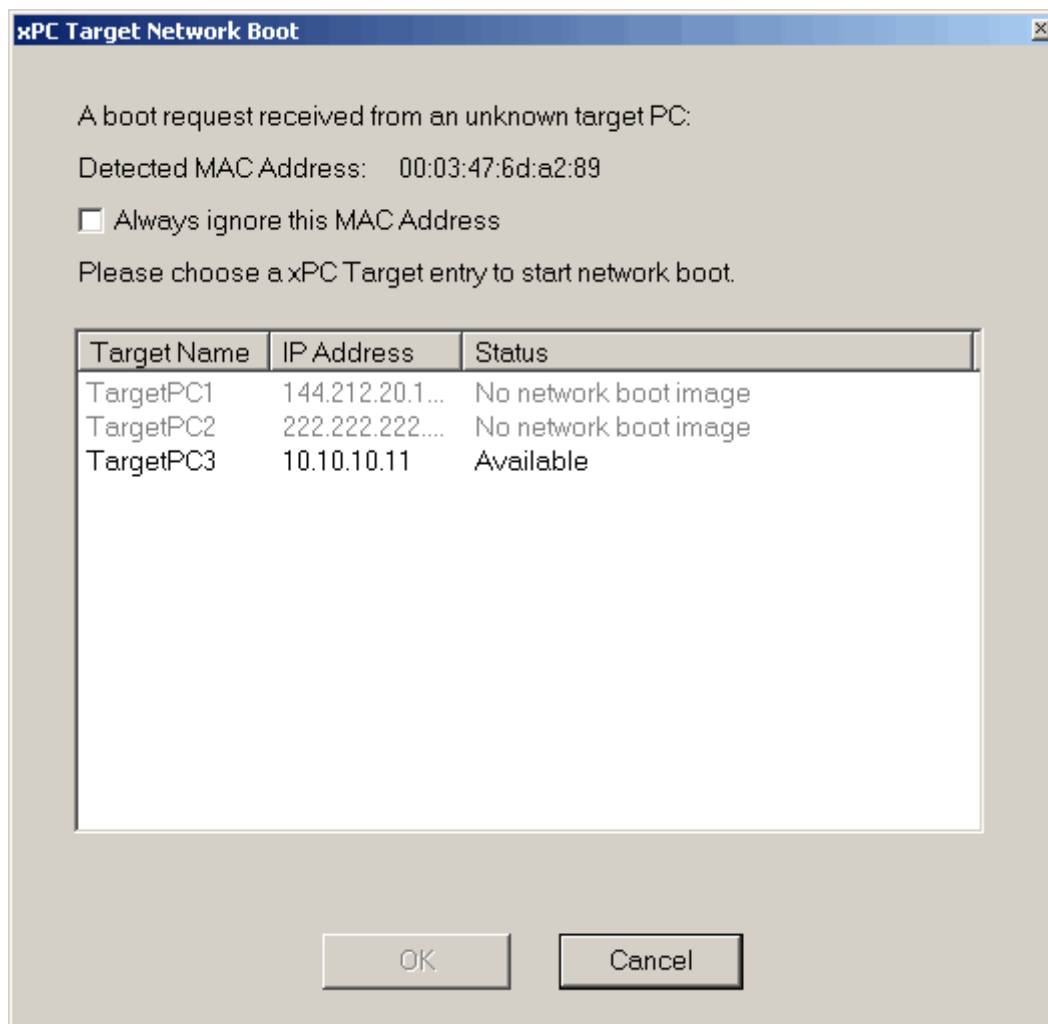
A minimized icon () representing the network boot server process appears on the bottom right host PC system tray.

Booting the Target PC

1 Reboot the target PC.

The host PC network boot server displays a pop-up from the system tray indicating that the boot server is being downloaded to the target PC.

If the target PC is not already associated with a physical target PC MAC address, the first time that the network boot server process detects a viable target PC, it displays a dialog that contains the target PC names and the IP addresses for those names. From this list, select the physical target PC you want to associate with the target PC name.



- 2 Select the target PC name with which you want to associate the physical target PC.

The target PC receives the xPC Target kernel and boots with this kernel.

If you click the **Cancel** button instead of selecting a target PC name, the next time you try to boot the target PC across the network, the kernel will ignore the target PC boot request for 90 seconds.

Note the following behavior:

- If the target PC name has a MAC address, and there is a physical target PC whose MAC address matches the target PC name MAC address, the software matches the two and the **xPC Target Network Boot** dialog does not display.
- If the connection between the target PC and host PC is an RS-232 one, you cannot boot the target PC across the network.
- If the StandAlone mode is enabled, you cannot boot the target PC across the network.

Configuring the xPC Target Environment for 64-Bit MATLAB Systems

In this section...

- “Setup Requirements” on page 2-61
- “Configuring the Compiler” on page 2-62
- “Configuring Communications” on page 2-63
- “Configuring Boot Methods” on page 2-65

Setup Requirements

If you run xPC Target software on a 64-bit system, you must use the command-line interface. You must set up the xPC Target environment from the MATLAB command line. In this case, use these topics to set up and work with the xPC Target environment.

To set up the environment, you use the set (env object) and target environment object (see “Working with Target PC Environments” in the *xPC Target User’s Guide*) to define host and target PC properties. These properties give xPC Target information about the software and hardware configuration. Once you have the properties set, you will rarely need to change some of them. Others, you might change often.

Setup involves:

- “Configuring the Compiler” on page 2-62
- “Configuring Communications” on page 2-63
- “Configuring Boot Methods” on page 2-65

These instructions provide the minimum configuration for running your target PC with the xPC Target environment. For a complete list of configuration properties, see set (env object).

The command examples assume that you have one target PC to configure, TargetPC1. If you want to add another to your environment, type the following in the MATLAB Command Window:

```
tgs=xpctarget.targets;  
get(tgs);  
tgs.Add;  
get(tgs);
```

xPC Target adds and names the target by an increment of 1. To configure the new target, reference its name in your configuration commands, for example, TargetPC2.

For further information on working in the xPC Target environment from the MATLAB Command Window, see the following sections in the *xPC Target User's Guide*:

- “Target and Scope Objects”
- “Targets and Scopes in the MATLAB Interface”
- “Working with Target PC Environments”

Configuring the Compiler

On the host PC:

- 1 In the MATLAB Command Window, type:

```
xpcsetCC('setup')
```

This function queries the host PC for C compilers that the xPC Target environment supports. It returns output like the following:

```
Select your compiler for xPC Target.
```

```
[1] Microsoft Visual C++ Compilers 2008 Professional Edition (SP1) in  
    c:\Program Files (x86)\Microsoft Visual Studio 9.0
```

```
[2] Microsoft Visual C++ Compilers 2010 Professional in  
    C:\Program Files (x86)\Microsoft Visual Studio 10.0
```

```
[0] None
```

```
Compiler:
```

- 2 At the **Compiler** prompt, enter the number for the compiler that you want to use. For example, 2.

The function verifies your selection:

```
Verify your selection:
```

```
Compiler: Microsoft Visual C++ Compilers 2010 Professional  
Location: C:\Program Files (x86)\Microsoft Visual Studio 10.0
```

```
Are these correct [y]/n?
```

- 3 Type **y** or press **Enter** to verify the selection.

The function finishes the dialog.

```
Done...
```

Now you can proceed to “Configuring Communications” on page 2-63 for the xPC Target environment.

Configuring Communications

Set the properties for the xPC Target environment according to the communications method that your host and target PCs use:

- “Network Communications” on page 2-63
- “Serial Communications” on page 2-64

Network Communications

On the host PC, set the properties that your host and target PCs require for network connections.

- 1 In the MATLAB Command Window, type the following commands to create and environment object for your target:

```
tgs=xpctarget.targets  
tgs.makeDefault('TargetPC1')  
env=tgs.Item('TargetPC1')
```

- 2 Ensure that these properties are set:

Property	Value
HostTargetComm	TcpIp.
TcpIpTargetAddress	Target PC IP address, for example, 192.168.0.10.
TcpIpTargetPort	22222 (default).
TcpIpSubNetMask	Subnet mask address of your LAN, for example 255.255.255.0.
TcpIpGateway	Gateway IP address, 255.255.255.255 (default).
TcpIpTargetDriver	Ethernet driver. Auto (default) allows the software to choose the correct driver for the target PC system.
TcpIpTargetBusType	Target PC bus type, PCI (default).
TcpIpTargetISAMemPort and TcpIpTargetISAIRQ	If you are using an ISA bus Ethernet card, enter values for these properties. The values of these properties must correspond to the jumper settings or ROM settings on your ISA bus Ethernet card.

For more information about these settings, see “Network Communication” on page 2-26.

- 3 If any of the values are incorrect, change them. For example:

```
env.HostTargetComm='TcpIp'
```

When the communication properties are set, see “Configuring Boot Methods” on page 2-65.

Serial Communications

On the host PC, set the properties that your host and target PCs require for serial connections:

- 1 In the MATLAB Command Window, type the following commands to create and environment object for your target:

```
tgs=xpctarget.targets
tgs.makeDefault('TargetPC1')
env=tgs.Item('TargetPC1')
```

- 2 Ensure that these properties are set:

Property	Value
HostTargetComm	RS232.
RS232HostPort	COM1 (default).
RS232Baudrate	115200 (default).

For more information on these settings, see “Serial Communication” on page 2-36.

- 3 If any of the values are incorrect, change them. For example:

```
env.HostTargetComm='RS232'
```

When the communication properties are set, see “Configuring Boot Methods” on page 2-65.

Configuring Boot Methods

Choose a boot method that your host and target PCs support:

- “Booting Target PCs from CD or DVD with a Command-Line Interface” on page 2-66
- “Booting Target PCs from 3.5-Inch Target Boot Disks” on page 2-68
- “Booting Target PCs Within a Dedicated Network with a Command-Line Interface” on page 2-69

Booting Target PCs from CD or DVD with a Command-Line Interface

You use a boot CD/DVD to load and run the xPC Target kernel. After you make changes to the xPC Target environment properties, you must create a CD/DVD bootable ROM. Before you start:

- Ensure that you have an empty, writable CD or DVD.
- Ensure that you have a CD/DVD-RW drive.
- Ensure that you can create a bootable CD or DVD. You can create a boot CD or DVD in one of the following ways:
 - The xPC Target software can create a boot CD or DVD for you. To use this capability, your host PC must have one of the following Windows systems:
 - Microsoft Windows 7
 - Microsoft Windows Vista
 - Microsoft Windows XP Service Pack 2 or 3 with Image Mastering API v2.0 (IMAPIv2.0), available at <http://support.microsoft.com/kb/KB932716>.
 - Third-party CD/DVD writing software can write ISO image files for you. Use this method if you do not have Microsoft Windows 7, Microsoft Windows Vista, or Microsoft Windows XP Service Pack 2 or 3.

Note Standard Microsoft Windows software (such as Windows Explorer or Windows Media Player) does not write ISO image files to CD/DVD.

To create a boot CD/DVD for the TargetPC1 environment:

- 1 Insert the empty CD or DVD in the host PC.
- 2 In the MATLAB Command Window, type

```
tgs=xpctarget.targets
tgs.makeDefault('TargetPC1')
env=tgs.Item('TargetPC1')
```


- 3** Ensure that the following xPC Target properties are set as follows:

Property	Value
TargetBoot	CDBoot.
CDBootImageLocation	Your host PC CD/DVD disk drive location.

- 4** If any of the values are incorrect, change them. For example:

```
env.TargetBoot='CDBoot'
env.CDBootImageLocation='c:\work\xpc\cdimage'
```

- 5** In the MATLAB Command Window, type:

```
xpcbootdisk
```

The xPC Target software displays the following message and creates the CD/DVD boot ISO image.

```
Current boot mode: CDBoot
CD boot image is successfully created
```

- 6** Perform one of the following, depending on your software:

- If you have Microsoft Windows 7, Microsoft Windows Vista, or Microsoft Windows XP Service Pack 2 or 3 with Image Mastering API v2.0 (IMAPIv2.0), xpcbootdisk prompts you to insert a CD/DVD.

```
Insert an empty CD/DVD. Available drives:
[1] z:\
[0] Cancel Burn
```

Select the appropriate drive, insert the CD or DVD, then press the **Enter** key.

- If you do not have Microsoft Windows 7, Microsoft Windows Vista, or Microsoft Windows XP Service Pack 2 or 3 with Image Mastering API v2.0 (IMAPIv2.0), use your third-party software to write the `cdboot.iso` image to the empty CD/DVD.

- 7** When the CD/DVD drive stops, remove the CD/DVD.

Your next task is to install the software on the target PC and test your installation. See “Testing and Troubleshooting the Installation” on page 2-71.

Booting Target PCs from 3.5-Inch Target Boot Disks

You use a boot disk to load and run the xPC Target kernel. After you make changes to the xPC Target environment properties, you must create or update the boot disk.

- 1** In the MATLAB Command Window, type:

```
tgs=xpctarget.targets
tgs.makeDefault('TargetPC1')
env=tgs.Item('TargetPC1')
```

- 2** Ensure that the following properties are set as follows:

Property	Value
TargetBoot	BootFloppy.
BootFloppyLocation	Your host PC 3.5-inch disk drive location.

- 3** If any of the values are incorrect, change them. For example:

```
env.TargetBoot='BootFloppy'
env.BootFloppyLocation='a:'
```

- 4** In the MATLAB Command Window, type:

```
xpcbootdisk
```

The following message appears:

```
Current boot mode: BootFloppy
Insert a formatted floppy disk into your host PC's
disk drive and press a key to continue
```

- 5** Insert a formatted floppy disk into the host PC disk drive, and then press any key.

The write procedure starts and, while creating the boot disk, the MATLAB Command Window displays the following status information:

```
Creating xPC Target boot disk ... Please wait
xPC Target boot disk successfully created.
```

Your next task is to install the software on the target PC and test your installation. See “Testing and Troubleshooting the Installation” on page 2-71.

Booting Target PCs Within a Dedicated Network with a Command-Line Interface

If you boot target PCs on a dedicated network, you do not need a boot disk. You do need to set up the host PC and target PC. Before you start using the command-line interface to perform this operation, see “Booting Target PCs Within a Dedicated Network” on page 2-54. On 64-bit MATLAB systems, instead of using xPC Target Explorer, use the command-line interface:

- 1 In the MATLAB Command Window, type:

```
tgs=xpctarget.targets
tgs.makeDefault('TargetPC1')
env=tgs.Item('TargetPC1')
```

- 2 Ensure that the following property is set as follows:

Property	Value
TargetBoot	NetworkBoot.

- 3 If any of the values are incorrect, change them. For example:

```
env.TargetBoot='NetworkBoot'
```

- 4 Set a TCP/IP address. Ensure that the subnet of this IP address is the same as the host PC. Otherwise your network boot will fail. For example, type:

```
env.TcpIpTargetAddress='10.10.10.11'
```

- 5 Set the target PC MAC address (in hexadecimal).

```
env.TargetMACAddress='01:23:45:67:89:ab'
```


- 6 In the MATLAB Command Window, type:

```
xpcnetboot
```

The following message appears:

```
Current boot mode: NetworkBoot
```

The software creates and starts a network boot server process on the host PC. You will boot the target PC using this process.

A minimized icon () representing the network boot server process appears on the bottom right of the host PC system tray.

Testing and Troubleshooting the Installation

In this section...

“Testing the Installation from a Boot Disk or Boot CD” on page 2-71

“Test 1, Ping Target System Standard Ping” on page 2-74

“Test 2, Ping Target System xPC Target Ping” on page 2-75

“Test 3, Reboot Target Using Direct Call” on page 2-76

“Test 4, Build and Download Application” on page 2-77

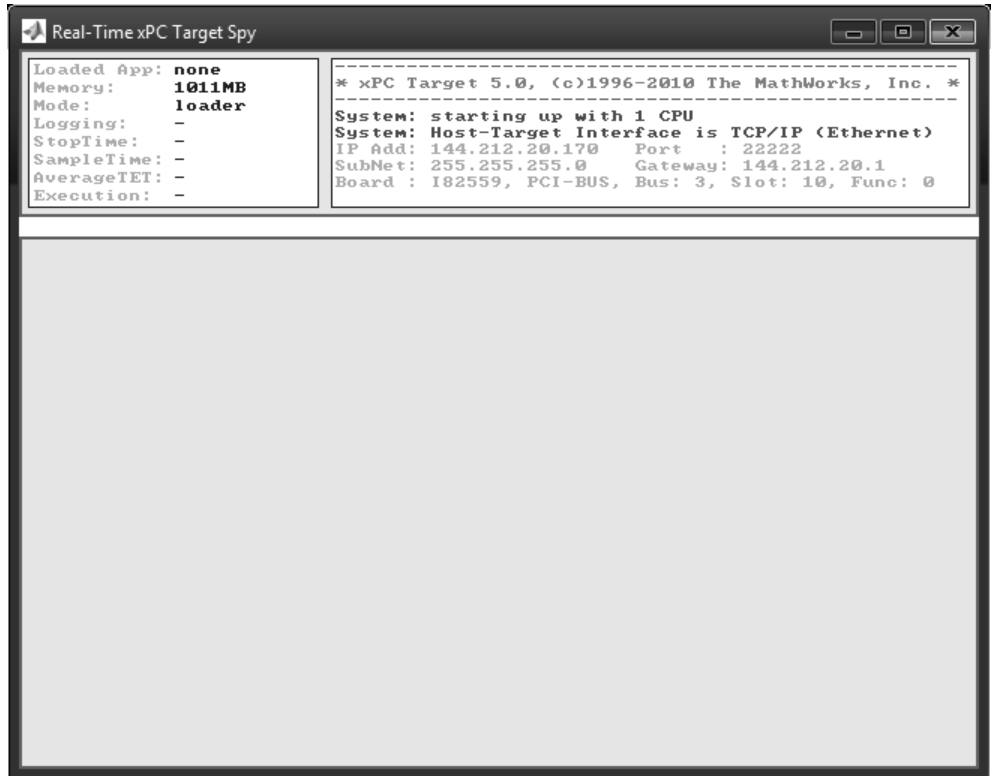
“If You Need More Help” on page 2-78

Testing the Installation from a Boot Disk or Boot CD

This topic describes how to install software on a target PC, boot that PC, and test the installation and connection between the host PC and the target PC. The xPC Target software uses a test script to test the entire installation. After you install the software, set the environment settings, and create a target boot disk, you can test your installation. This procedure assumes that you have set environment settings with xPC Target Explorer. See “Environment Properties for Serial Communication” on page 2-37 or “Environment Properties for Network Communication” on page 2-29.

- 1** Insert your target boot disk into a target PC disk drive or CD drive. This target boot disk contains the software to run a target PC.
- 2** To reboot the target PC, press the reset button on the PC.

After loading the BIOS, the software boots the kernel and displays the following on the target PC monitor.



If you have a keyboard attached to the target PC, you can activate that keyboard by typing **C**, and press the **Page Up** and **Page Down** keys to page up and down the target PC monitor.

- 3 In the MATLAB Current Folder window, select a folder outside the MATLAB root folder.

Note During the build process, Simulink Coder does not allow files to be saved within the MATLAB tree root. If you select a current folder within the MATLAB tree, the xPC Target test procedure fails when trying to build a target application.

4 In the MATLAB Command Window, type

```
xpctest
```

MATLAB runs the test script for the default target PC and displays messages indicating the success or failure of a test. If you use RS-232 communication, the first test is skipped.

```
### xPC Target Test Suite 5.0
### Host-Target interface is: TCP/IP (Ethernet)
### Test 1, Ping target system using system ping: ... OK
### Test 2, Ping target system using xpctargetping: ... OK
### Test 3, Software reboot the target PC: ..... OK
### Test 4, Build and download an xPC Target application using model xpcosc: ... OK
### Test 5, Check host-target command communications: ... OK
### Test 6, Download a pre-built xPC Target application: ... OK
### Test 7, Execute xPC Target application for 0.2s: ... OK
### Test 8, Upload logged data and compare with simulation results:. OK
### Test Suite successfully finished
```

If all of the tests succeed, you are ready to build and download a target application to the target PC. See Chapter 3, “Basic Tutorial”.

If any of the tests fails, see the appropriate test section:

- “Test 1, Ping Target System Standard Ping” on page 2-74
- “Test 2, Ping Target System xPC Target Ping” on page 2-75
- “Test 3, Reboot Target Using Direct Call” on page 2-76
- “Test 4, Build and Download Application” on page 2-77

Note This topic describes tests 1 to 4. For further details on these tests, or for a description of tests 5 to 8, see “Troubleshooting xpctest Results” in the “Frequently Asked Questions” chapter of the *xPC Target User’s Guide*.

Test 1, Ping Target System Standard Ping

If you are using a network connection, this is a standard system ping to your target computer. If this test fails, try troubleshooting with the following procedure:

- 1 Open a DOS shell and type the IP address of the target computer:

```
ping xxx.xxx.xxx.xxx
```

DOS should display a message similar to the following:

```
Pinging xxx.xxx.xxx.xxx with 32 bytes of data:  
Replay form xxx.xxx.xxx.xxx: bytes=32 time<10 ms TTL=59
```

- 2 Check the messages on your screen.

Ping command fails — If the DOS shell displays the following message,

```
Pinging xxx.xxx.xxx.xxx with 32 byte of data:  
Request timed out.
```

the ping command failed, and the problem might be with your network cables.

To solve this problem, check your network cables. You might have a faulty network cable, or if you are using a coaxial cable, the terminators might be missing.

Ping command fails, but cables are okay — If the cables are okay, the problem might be that you entered an incorrect property in xPC Target Explorer.

To solve this problem, in the MATLAB Command Window, type

```
xpcexplr
```

For the problem target PC, check that **Target PC IP address**, **LAN subnet mask address**, and **TCP/IP gateway address** have the correct values. Change the TCP/IP options as necessary, then create a new boot floppy disk. On the target PC, reboot with the new boot floppy disk.

For a PCI bus,

- Check that **TCP/IP target bus type** is set to PCI instead of ISA.

For an ISA bus,

- Check that **TCP/IP target bus type** is set to ISA instead of PCI.
- Check that **TCP/IP target ISA memory port** is set to the correct I/O port base address and check that the address does not lead to a conflict with another hardware resource.
- Check that **TCP/IP target ISA IRQ number** is set to the correct IRQ line and check that the line number does not lead to a conflict with another hardware resource.
- If the target PC motherboard contains a PCI chip set, check whether the IRQ line used by the ISA bus Ethernet card is reserved within the BIOS setup.

Ping succeeds, but test 1 with the command `xpctest` fails — The problem might be that you have incorrect IP and gateway addresses entered in xPC Target Explorer.

To solve this problem, in the MATLAB Command Window, type

```
xpcexplr
```

For the problem target PCs, enter the correct addresses. Recreate the target boot disk.

See also “`xpctest`: Test 1 Fails” in “Frequently Asked Questions” in the xPC Target User’s Guide. If you still cannot solve your problem, see “If You Need More Help” on page 2-78.

Test 2, Ping Target System xPC Target Ping

This test is an xPC Target ping to your target computer. If this test fails, try troubleshooting with the following procedure:

- 1 In the MATLAB Command Window, type

```
tg=xpctarget.xpc('argument-list')
```

where `argument-list` is the connection information that indicates which target PC you are working with. If you do not specify any arguments, the software assumes that you are communicating with the default target PC.

- 2 Check the messages in the MATLAB Command Window.

MATLAB should respond with the following messages:

```
xPC Object
  Connected           = Yes
  Application         = loader
```

Target object does not connect — If you do not get the preceding messages, the problem might be that you have a bad target boot disk.

To solve this problem, create another target boot disk with a new floppy disk. See “Booting Target PCs from Boot Floppy Disk” on page 2-51.

See also “xpctest: Test 2 Fails” in “Frequently Asked Questions” in the xPC Target User’s Guide. If you still cannot solve your problem, see “If You Need More Help” on page 2-78.

Test 3, Reboot Target Using Direct Call

This test tries to boot your target computer using an xPC Target command. If this test fails, try troubleshooting with the following procedure. This procedure assumes that you have set environment settings with xPC Target Explorer. See “Environment Properties for Serial Communication” on page 2-37 or “Environment Properties for Network Communication” on page 2-29.

- 1 In the MATLAB Command Window, type

```
xpctest('-noreboot')
```

This command reruns the test without using the `reboot` command and displays the message

```
### Test 3, Software reboot the target PC: ... SKIPPED
```

- 2 Observe the messages in the MATLAB Command Window during the build process.

Reboot fails, but build okay when reboot skipped — If the command `xpctest` skips the `reboot` command but successfully builds and loads the target application, the problem could be that some target hardware does not support the xPC Target `reboot` command. In this case, you cannot use this command to reboot your target computer. You need to reboot using a hardware reset button.

See also “`xpctest: Test 3 Fails`” in “Frequently Asked Questions” in the xPC Target User’s Guide. If you still cannot solve your problem, see “If You Need More Help” on page 2-78.

Test 4, Build and Download Application

This test tries to build and download the model `xpcosc.mdl`. If this test fails, try troubleshooting with the following procedure:

- 1 In the MATLAB Command Window, check the error messages.

These messages help you locate where there is a problem.

- 2 If you get the error message

```
xPC Target loader not ready
```

Reboot your target computer. This error message is sometimes displayed even if the target screen shows the loader is ready.

See also “`xpctest: Test 4 Fails`” in “Frequently Asked Questions” in the xPC Target User’s Guide. If you still cannot solve your problem, see “If You Need More Help” on page 2-78.

If You Need More Help

If you cannot solve your problem, contact MathWorks directly for help.

Internet To contact Mathworks Technical Support, use this form
http://www.mathworks.com/contact_TS.html

Telephone 508-647-7000

Ask for Technical Support.

Note MathWorks Technical Support might ask you to use the `getxpcinfo` function to retrieve diagnostic information for your xPC Target configuration. This function writes the diagnostic information to the `xpcinfo.txt` file in the current folder. This file might contain information sensitive to your organization. Review the contents of this file before sending to MathWorks.

Exporting and Importing xPC Target Explorer Environments

The xPC Target Explorer consists of the property settings you define for the Configuration node, for example, for the host communication method and so forth. When you have settings that you are happy with, you can save them as variables in the MATLAB workspace.

This topic describes how to export target PC node property settings in a structured format to the MATLAB workspace. It assumes that you have set settings with xPC Target Explorer. See “Environment Properties for Serial Communication” on page 2-37 or “Environment Properties for Network Communication” on page 2-29.

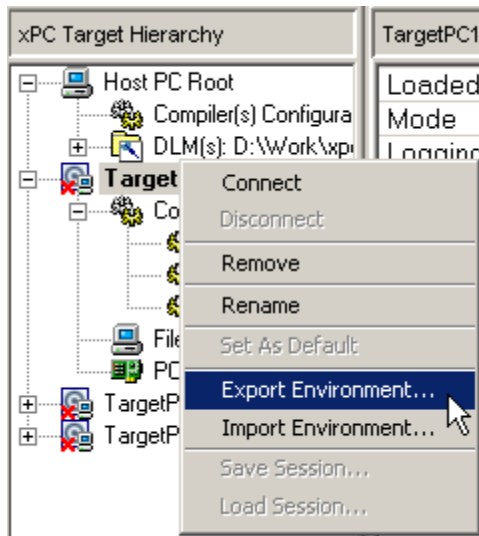
- 1 If the xPC Target Explorer is not open, open it now. At the MATLAB Command Window, type

```
xpcexplr
```

- 2 In the xPC Target Explorer **xPC Target Hierarchy** pane, right-click the **Configuration** node of the target PC for which you want to save the configuration. For example, right-click TargetPC1.

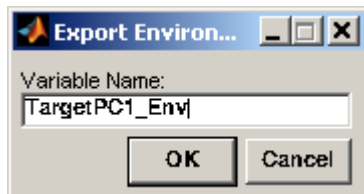
A context-sensitive menu is displayed.

- 3 Select **Export Environment**.



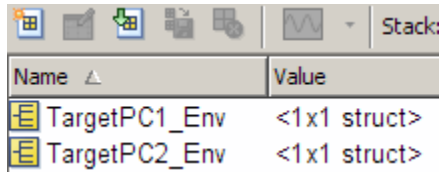
The Export Environment to Workspace dialog is displayed.

- 4 In the Export Environment to Workspace dialog box, enter a unique name. For example, type TargetPC1env.



- 5 Add configuration variables for as many target PC configurations as you need.

The following illustrates a MATLAB workspace with two xPC Target Explorer environment configuration variables.



Name	Value
TargetPC1_Env	<1x1 struct>
TargetPC2_Env	<1x1 struct>

You can save the target PC environment structure to a MAT-file. This enables you to reimport the configuration defined in the structure into a future xPC Target Explorer session.

- 1 To save the variable `TargetPC1env` in the MAT-file `targetpc1.mat`, in the MATLAB Command Window, type

```
save targetpc1.mat TargetPC1env
```

MATLAB saves the file `targetpc1.mat` in the current folder.

- 2 In the same MATLAB session, or in a different one, load the contents of `targetpc1.mat` into the MATLAB workspace. Type

```
load targetpc1.mat
```

- 3 If the xPC Target Explorer is not open, open it now.
- 4 In the xPC Target Explorer **xPC Target Hierarchy** pane, right-click the target PC node for which you want to import the configuration. For example, right-click `TargetPC1`.

A context-sensitive menu is displayed.

- 5 Select **Import Environment**.
- 6 In the Import Environment Structure dialog, select a previously exported configuration. For example, select `TargetPC1env`.

Basic Tutorial

This chapter explains the basic functions of the xPC Target product by using a simple Simulink model. Because this model does not have I/O blocks, you can try these procedures whether or not you have I/O hardware on your target PC. This chapter includes the following sections:

- “Simulink Model” on page 3-2
- “Simulating the Model” on page 3-39
- “xPC Target Application” on page 3-45
- “Running the Target Application” on page 3-57
- “Parallel Model Reference Builds Using Remote Workers” on page 3-71
- “Menu Bar and Toolbar Contents and Shortcut Keys” on page 3-72

Simulink Model

In this section...

“Creating a Simple Simulink Model” on page 3-2

“Entering Parameters for the Scope Block” on page 3-6

“Adding a Simulink Outport Block” on page 3-10

“Entering Parameters for the Outport Blocks” on page 3-13

“Adding an xPC Target Scope Block” on page 3-17

“Entering Parameters for an xPC Target Scope Block” on page 3-22

Creating a Simple Simulink Model

Before you can create a target application, you need to create a Simulink model. The software xPC Target then uses the Simulink model, the Simulink Coder environment, and a third-party compiler to create the target application. This tutorial uses a simple Simulink model to explain the tasks you need to do with the software xPC Target. If you are an experienced Simulink user, you can skip creating this model.

The model includes a transfer function and a signal generator block. If you want to visualize signals while simulating your model, you need to add a standard Simulink Scope block.

1 In the MATLAB Command Window, type

```
simulink
```

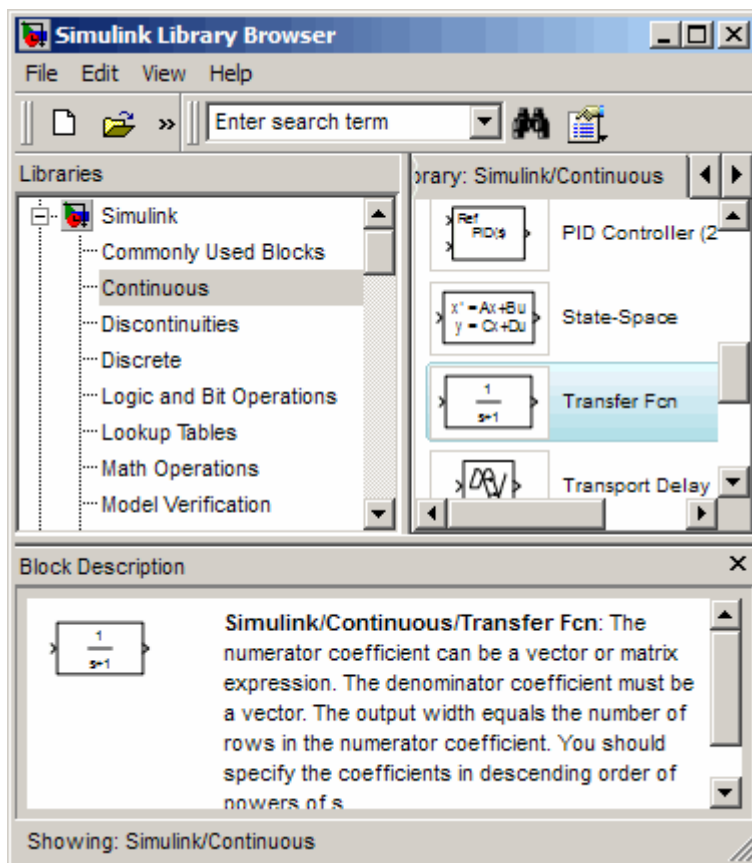
The Simulink library window opens.

2 From the **File** menu, point to **New**, and then click **Model**.

A blank Simulink model window opens.

3 In the left pane, double-click **Simulink**, and then click **Continuous**.

In the right pane, the Simulink library shows a list of blocks.



- 4 Click and drag the Transfer Fcn block to the Simulink model window.
- 5 In the Simulink Library Browser window, click and drag the following blocks to your model.
 - Click **Sources**, and add a Signal Generator block.
 - Click **Sinks**, and add a Scope block.
 - Click **Signal Routing**, and add a Mux block.

Note 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 PC GUI scope graph after you build and download the model to the target PC. 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.

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

In the **Amplitude** text box, enter

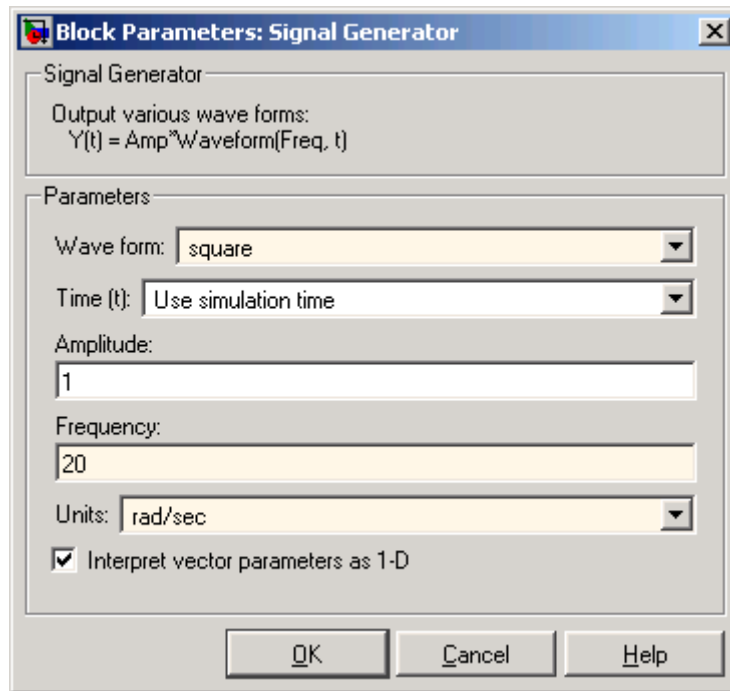
1

In the **Frequency** text box, enter

20

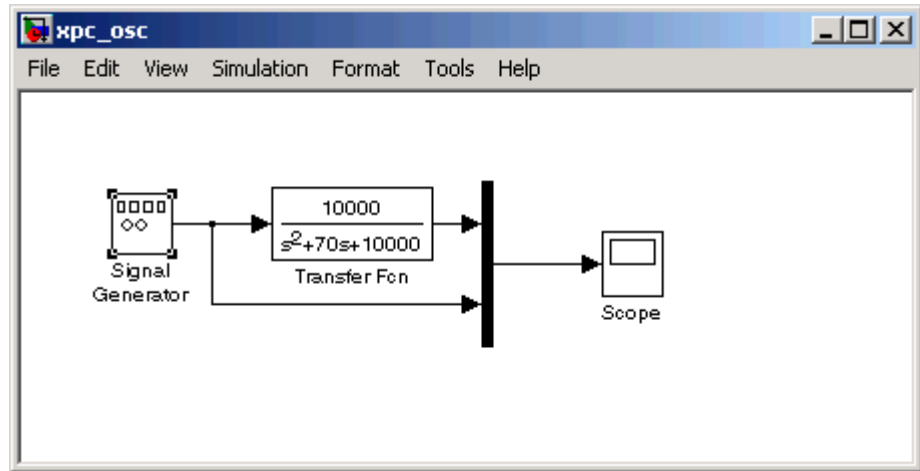
From the **Units** list, select **rad/sec**.

Your Block Parameters dialog box will look similar to the following figure.



- 7 Double-click the Transfer Fcn block.
- 8 Edit the **Numerator** and **Denominator** parameters to match those in the following figure.
- 9 Connect the Signal Generator block to the Transfer Fcn block, and connect the input and output signals to the scope block using the Mux block.

Your model should look similar to the figure shown.



- 10 From the **File** menu, click **Save As** and enter a filename. For example, enter `my_xpc_osc` and then click **OK**.

You can use either a Simulink Scope block or an xPC Target Scope block to visualize signals from an xPC Target application running in real time. The topics for this example describe how to use the xPC Target Scope block. See “Adding an xPC Target Scope Block” on page 3-17. See “Signal Tracing with Simulink External Mode” for a description of how to use a Simulink Scope block to visualize target application signals.

For information on creating a Simulink model and adding signal and scope blocks, see the online Simulink documentation.

Entering Parameters for the Scope Block

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

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

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

A Scope window opens.

- 2 Click the **Parameters** button.



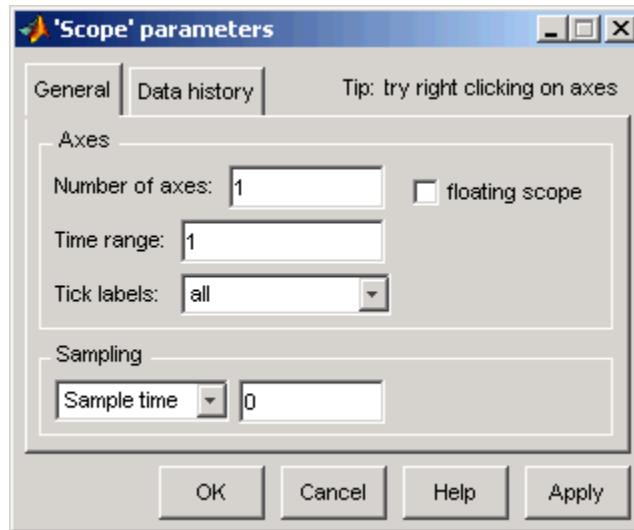
A Scope parameters dialog box opens.

- 3 Click the **General** tab. In the **Number of axes** box, enter the number of graphs you want in one Scope window. For example, enter 1 for a single graph. Do not select the **floating scope** check box.

In the **Time range** box, enter the upper value for the time range. For example, enter 1 second. From the **Tick labels** list, choose all.

From the **Sampling** list, choose **Sample time** and enter 0 in the text box. Entering 0 indicates that Simulink evaluates this block as a continuous-time block. If you have discrete blocks in your model, enter the **Fixed step size** you entered in the Configuration Parameters dialog box.

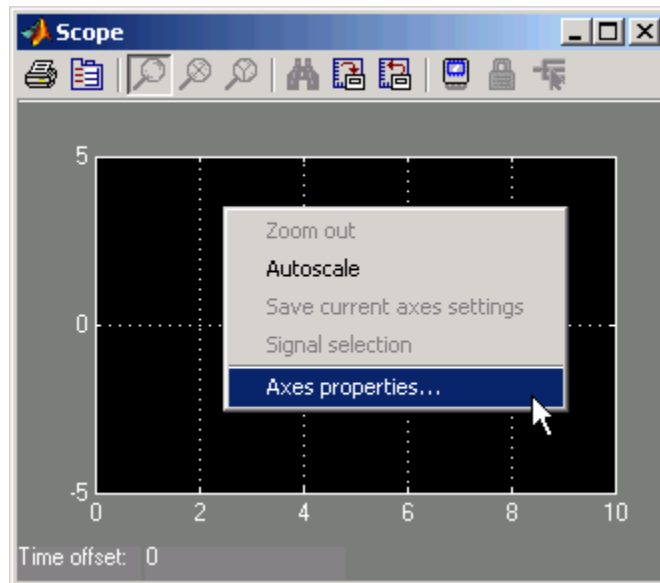
Your Scope parameters dialog box will look similar to the figure shown below.



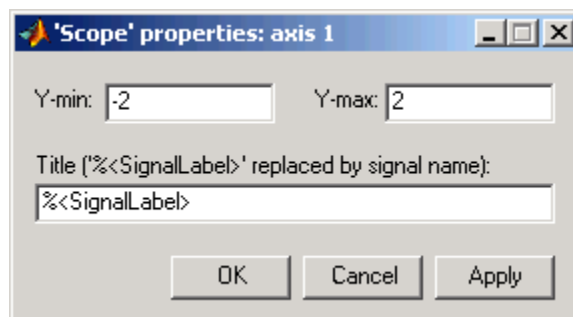
4 Do one of the following:

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

- 5 In the Scope window, point to the y -axis shown in the figure below, and right-click.



- 6 From the pop-up menu, click **Axes Properties**.
- 7 The Scope properties: axis 1 dialog box opens. In the **Y-min** and **Y-max** text boxes, enter the range for the y -axis in the Scope window. For example, enter -2 and 2 as shown in the figure below.



- 8 Do one of the following:

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

Adding a Simulink Output Block

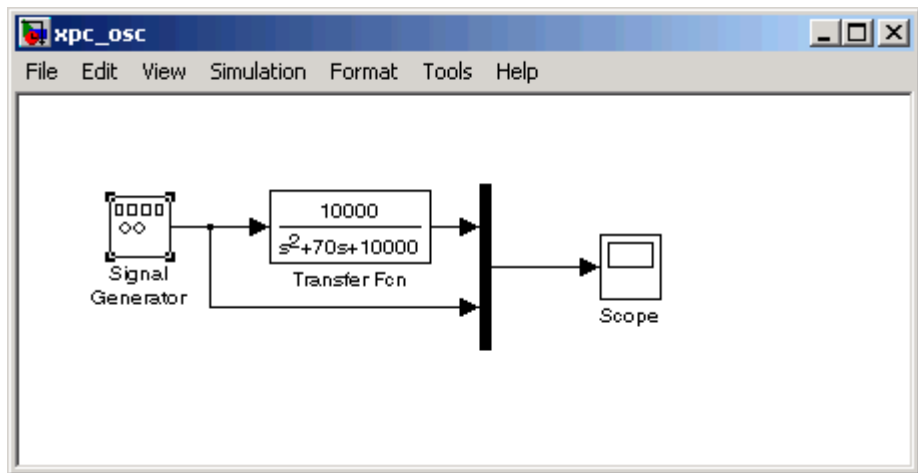
If you want to log signal data to the MATLAB workspace for analysis and later save that data to a disk, you need to add a Simulink Output block and activate logging from the Configuration Parameters dialog box.

The following procedure uses the Simulink model `my_xpc_osc.mdl` as an example. To create this model, see “Creating a Simple Simulink Model” on page 3-2.

- 1 In the MATLAB window, type

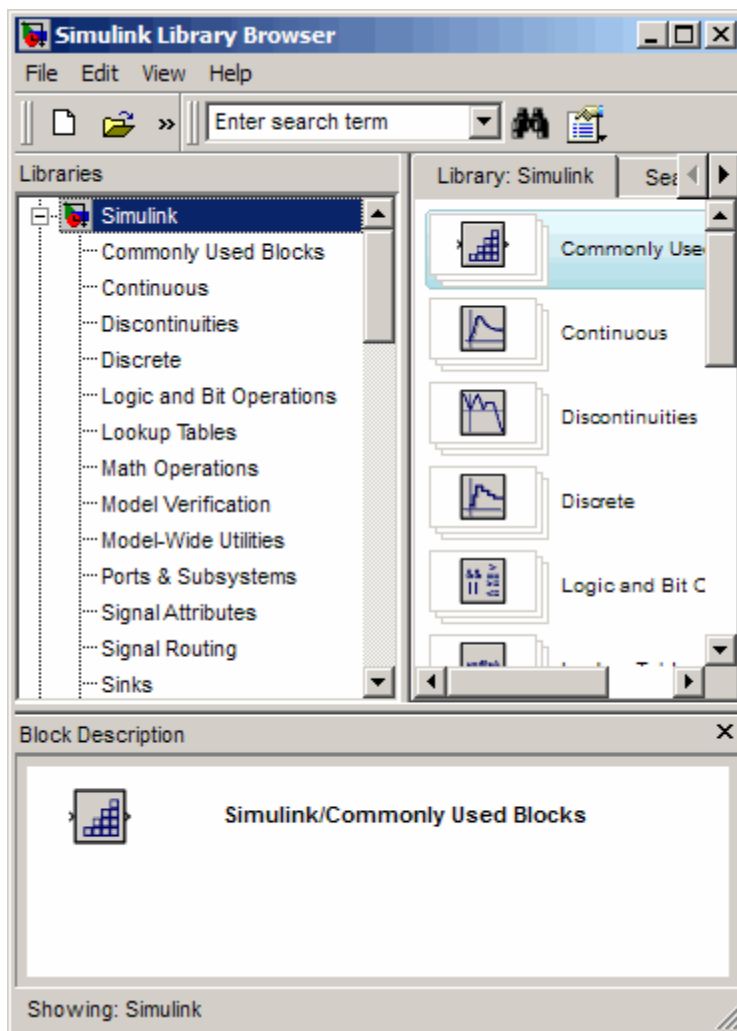
```
my_xpc_osc
```

The Simulink block diagram opens for the model `my_xpc_osc.mdl`.



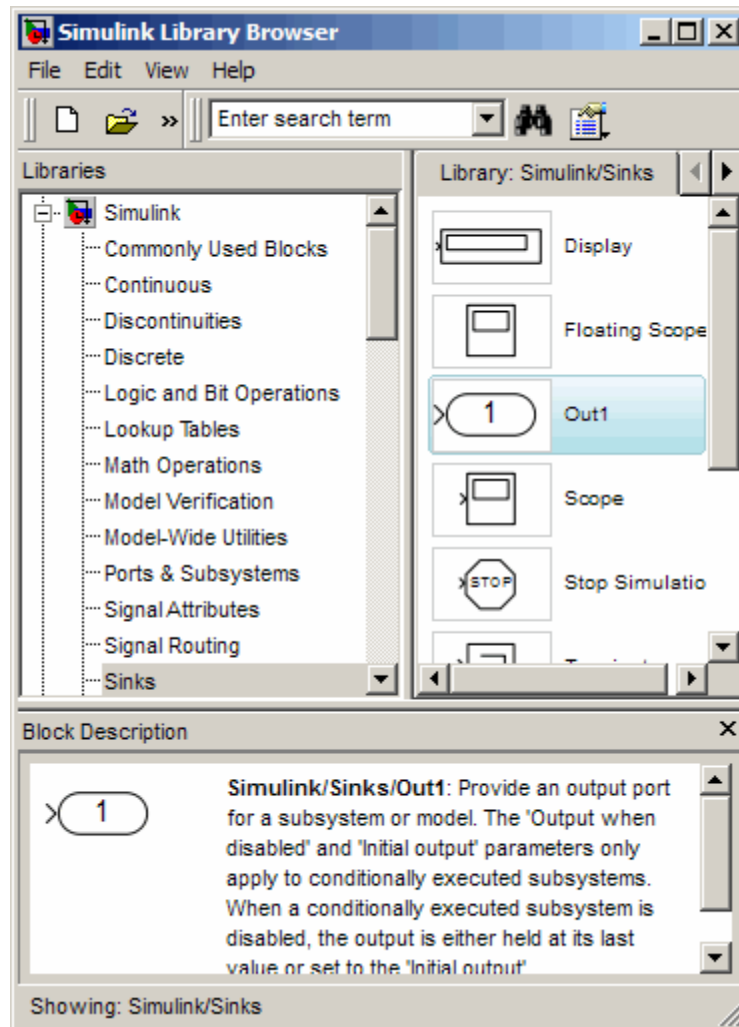
- 2 In the Simulink window, from the **View** menu, click **Library Browser**.

The Simulink Library Browser window opens.



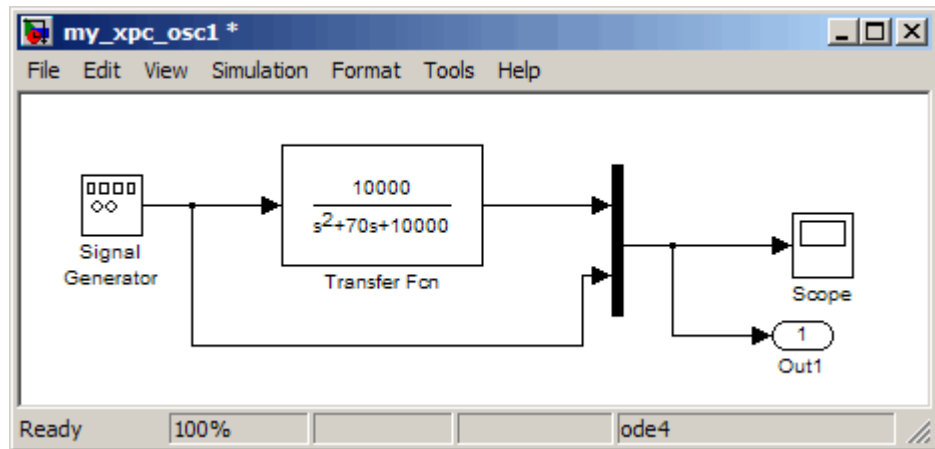
- 3 In the left pane, double-click **Simulink**, and then click **Sinks**.

In the right pane, the browser shows a list of sink blocks.



- 4 Click and drag the Out1 block to your model and connect it to the output of the Mux block.

Your model should look similar to the figure shown.



- 5 From the **File** menu, click **Save As** and enter a filename. For example, enter `my_xpc_osc1` and then click **OK**.

Your next task is to enter parameters for the Output block. See “Entering Parameters for the Output Blocks” on page 3-13.

Entering Parameters for the Output Blocks

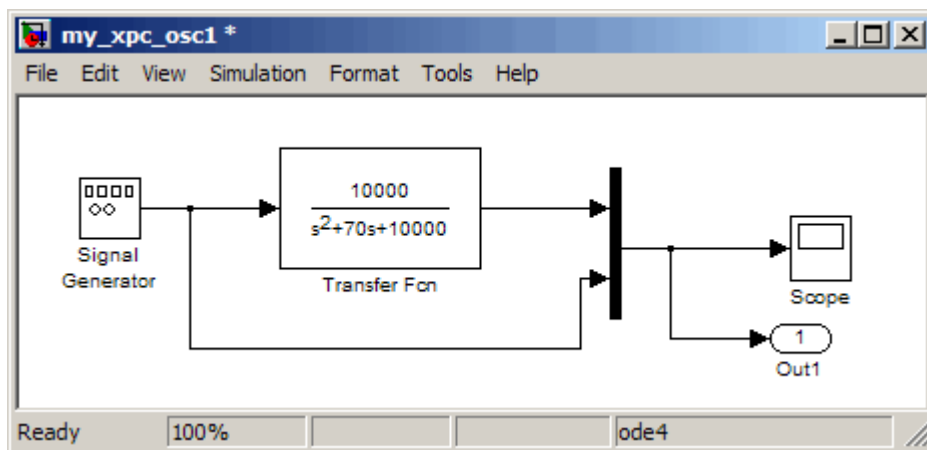
During a simulation, Simulink saves signal data to MATLAB variables using Output blocks. The default MATLAB variables are `tout`, `xout`, and `yout`. While running a real-time application, The xPC Target interface uses these same variables to pass signal data to target object parameters. A target object is a structure in the MATLAB workspace that the xPC Target software uses to interact with a target application. The default target object is `tg`, and the default parameters are `Time`, `tg.States`, and `tg.Output`.

After you add an Output block to your Simulink model, you can enter parameters. This procedure uses the model `my_xpc_osc1.mdl` with an Output block as an example. To add an Output block, see “Adding a Simulink Output Block” on page 3-10.

- 1 In the MATLAB window, type

```
my_xpc_osc1
```

A Simulink window with the model my_xpc_osc1 opens.



- 2 In the Simulink window, from the **Simulation** menu, click **Configuration Parameters**.

The Configuration Parameters dialog box is displayed for the model.

- 3 Click the **Solver** node.

Simulink displays the **Solver** pane. The **Simulation** section of this pane defines the initial stop and sample time for your target application.

- 4 In the **Solver options** section, enter 0 seconds in the **Start time** box. In the **Stop time** box, enter an initial stop time. For example, enter 20 seconds. To change this time after creating your target 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.

The **Solver** pane dialog changes.

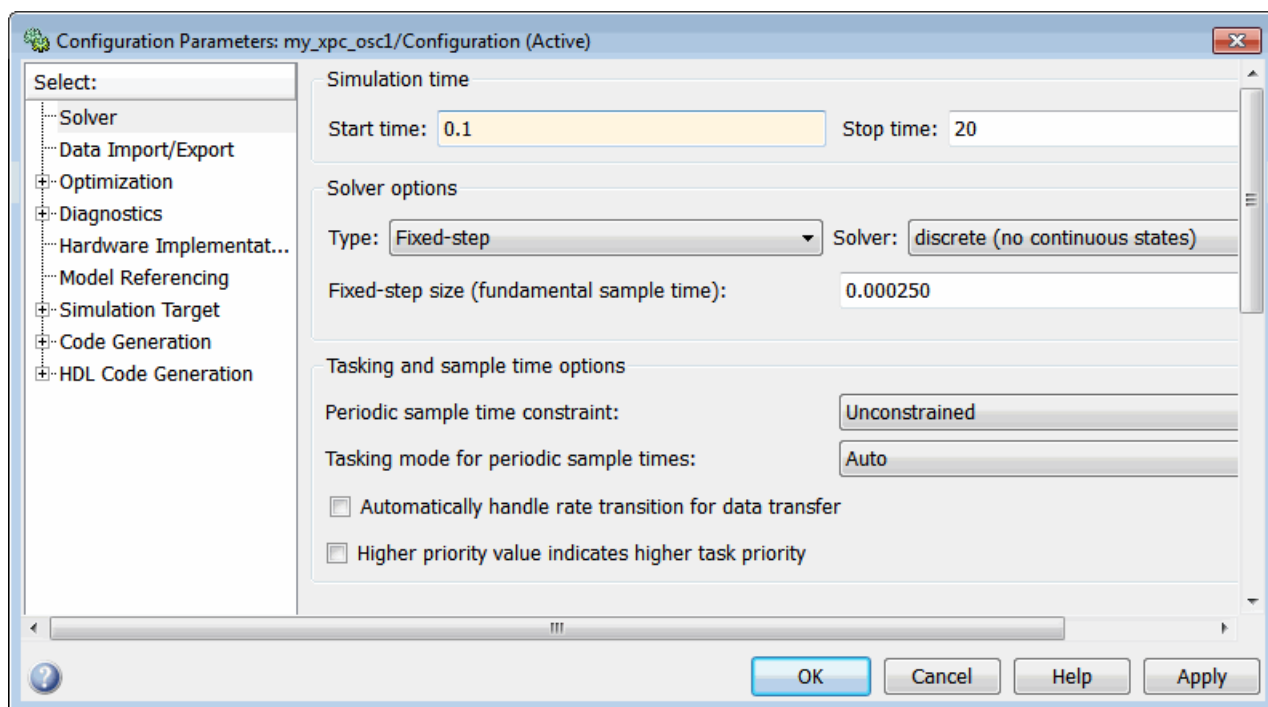
- 6 From the **Solver** list, select a solver. For example, select the general-purpose solver `ode4` (Runge-Kutta).

- 7** In the **Fixed step size** box, enter the sample time for the target application. For example, enter 0.00025 second (250 microseconds). You can change this value after creating the target application.

If you find that 0.000250 second results in overloading the CPU on the target PC, try a larger **Fixed step size** such as 0.002 seconds.

If your model contains discrete states, which would lead to a hybrid model with both continuous and discrete states, the sample times of the discrete states can only be multiples of the **Fixed step size**. If your model does not contain any continuous states, enter 'auto', and the sample time is taken from the model.

The **Solver** pane should look similar to the figure shown below.



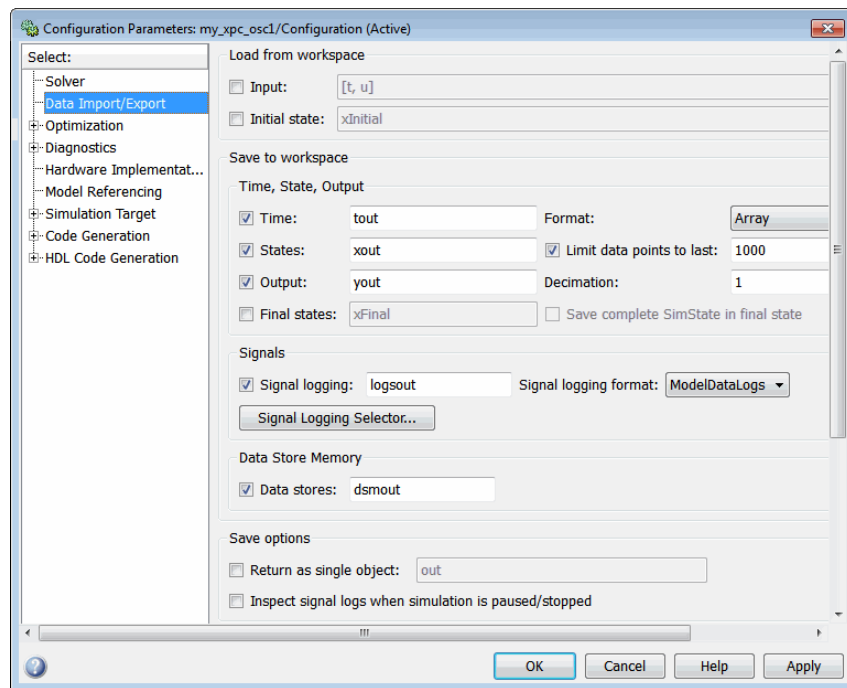
- 8** Click the **Data Import/Export** node.

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

- 9 In the **Save to workspace** section of this pane, select the **Time**, **States**, and **Output** check boxes.

Note When your target application is running in real time, data is not saved to the variables `tout` and `yout`. Instead, 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.

The **Data Import/Export** pane should look similar to the figure shown.



Normally you select all the **Save to workspace** check boxes. However, you might want to consider clearing some or all of them in the following cases:

- **Many states** — If your model contains many states (for example, more than 20 states), the storage of the state vector requires a lot of target memory. If you clear the **States** check box, logging of states is turned off and more memory is available for the target application. An alternative to logging all the state signals is to select individual states of interest by adding Outport blocks to your model.
- **Small sample time** — If you choose a very short sample time, this setting might overload the CPU. If you clear the **Save to workspace** check boxes, logging is turned off and more computing time is available for calculating the model.

1 Click **OK**.

2 From the **File** menu, click **Save**. The model is saved as `my_xpc_osc1.mdl`.

Your next task is to add an xPC Target Scope block to your Simulink model. See “Adding an xPC Target Scope Block” on page 3-17.

Adding an xPC Target Scope Block

The xPC Target software does not support the standard Simulink Scope blocks, but it does support xPC Target Scope blocks, which have unique capabilities when you use them with an xPC Target application. Do not confuse these xPC Target Scope blocks with standard Simulink Scope blocks.

Adding xPC Target Scope blocks to your Simulink model can save you time. When you build the model, the resulting target application contains the xPC Target Scope blocks you added to the model. After you download the target application, the xPC Target Scope block automatically displays on the target PC monitor. If you do not add xPC Target Scope blocks to your model and you want to monitor signals on the target application without rebuilding it, you need to add xPC Target scopes and define and select signals for the scopes (see “Signals and Parameters” in the xPC Target User’s Guide). The signal information is saved with your model.

Note If you want to monitor an output signal from a Constant block by connecting it to an xPC Target Scope block, you must add a test point for the Constant block output signal.

After you create a Simulink model, you can add an xPC Target Scope block. The following procedure uses the Simulink model `my_xpc_osc1.mdl` as an example to show how to connect an xPC Target Scope block to your model.

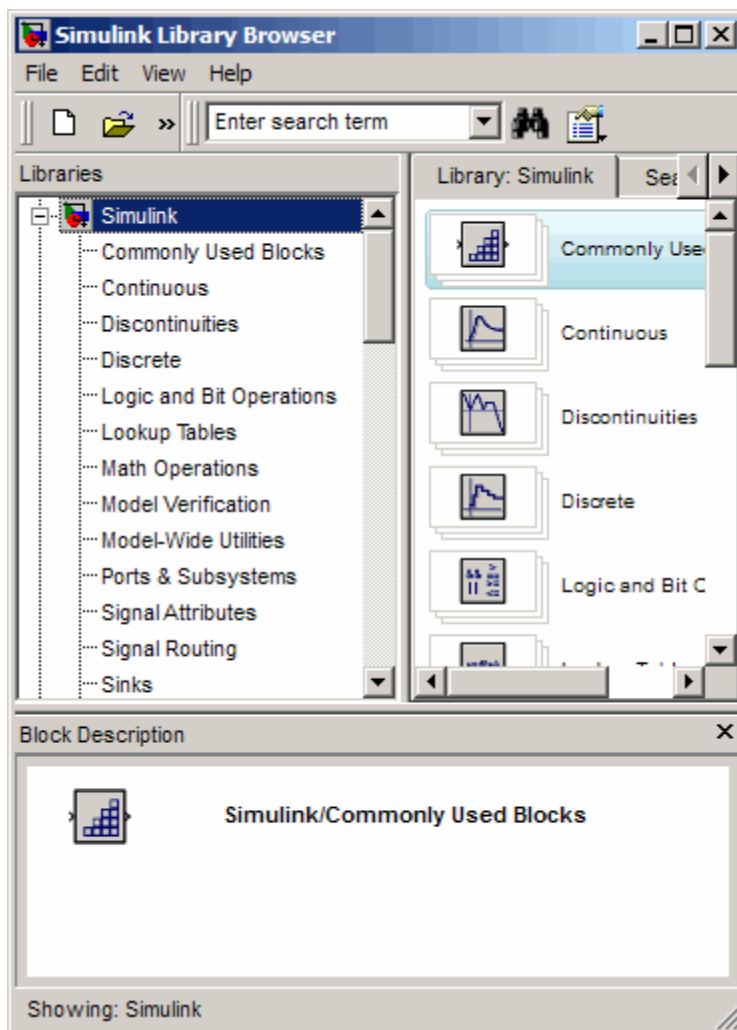
1 In the MATLAB window, type

```
my_xpc_osc1
```

The Simulink block diagram opens for the model `my_xpc_osc1.mdl`.

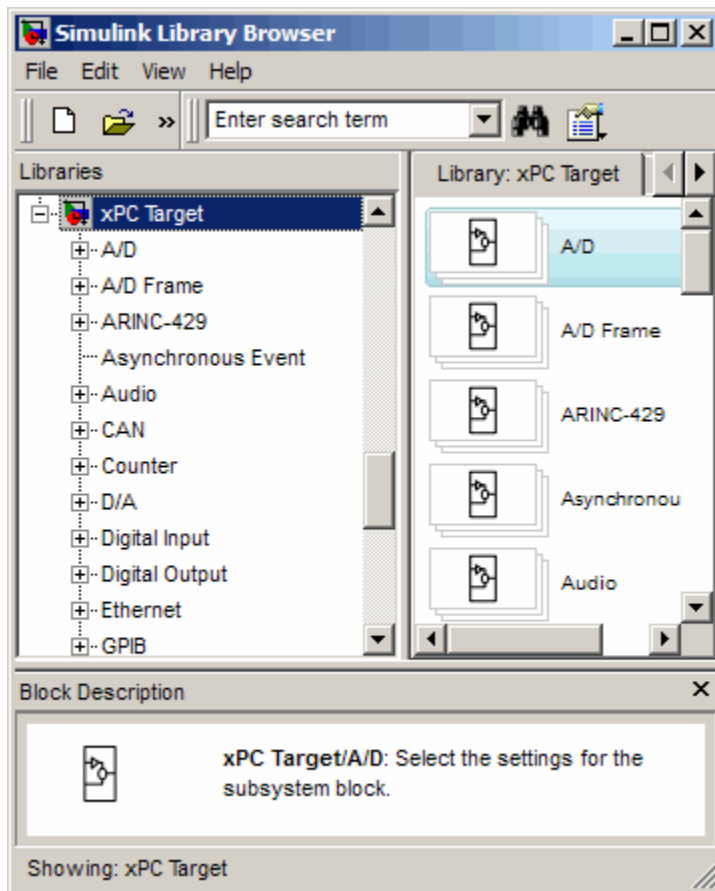
2 In the Simulink window, from the **View** menu, click **Library Browser**.

The Simulink Library Browser window opens.



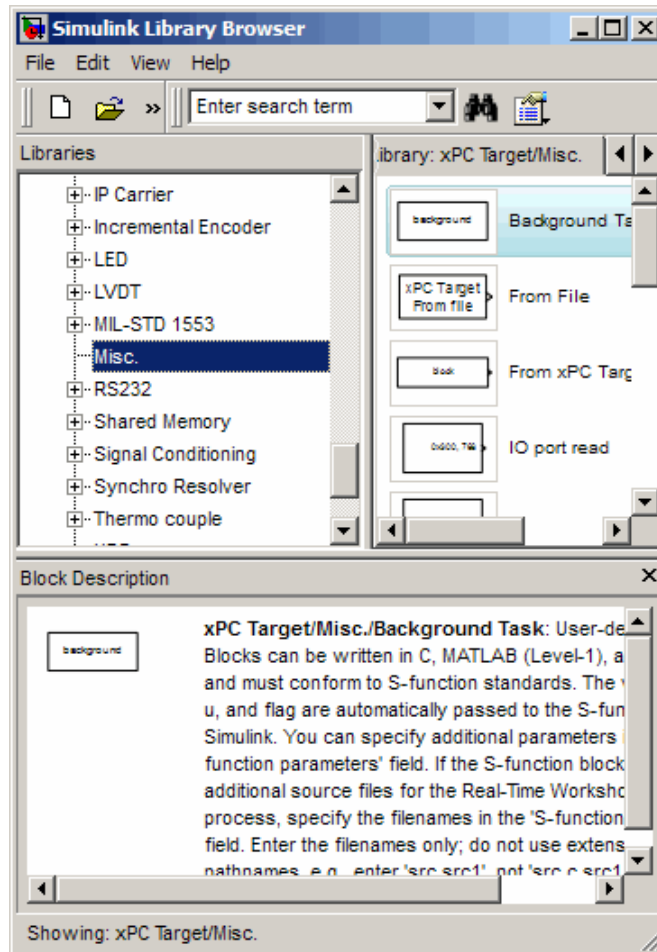
- 3 In the left pane, browse to and double-click **xPC Target**.

A list of I/O functions opens.



4 Click **Misc.**

A list of miscellaneous group blocks opens.

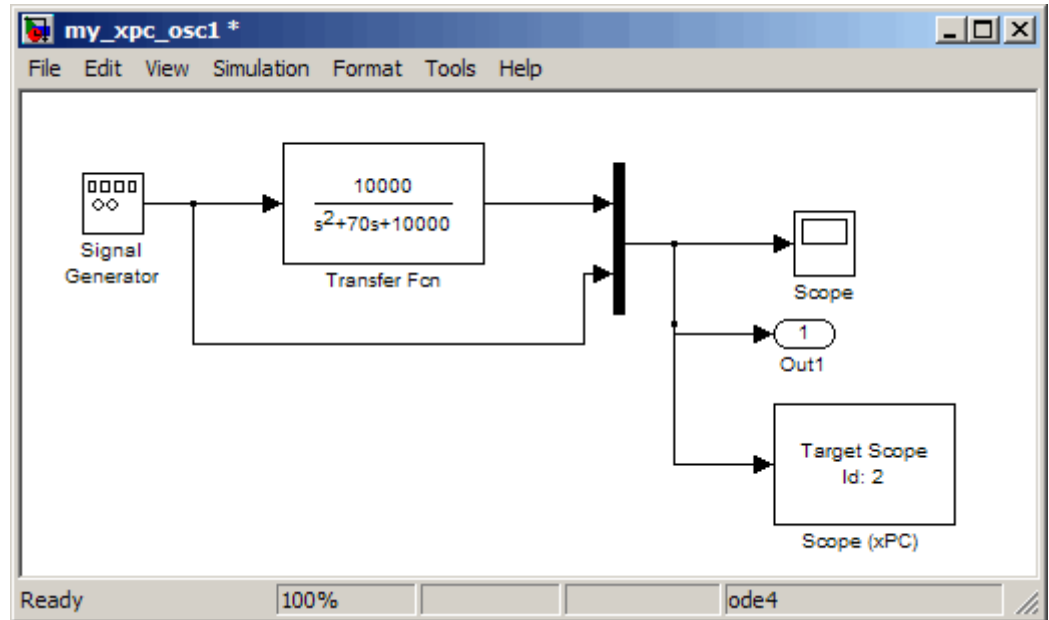


5 Click and drag **Scope (xPC)** to your Simulink block diagram.

Simulink adds a new Scope block to your model with a scope identifier of 1.

- 6 Connect the xPC Target Scope block to the Simulink Scope block.

The model `my_xpc_osc1.mdl` should look like the figure shown.



- 7 From the **File** menu, click **Save As**. Enter a filename. For example, enter `my_xpc_osc2` and then click **OK**.

Your next task is to define the xPC Target Scope block parameters. See “Entering Parameters for an xPC Target Scope Block” on page 3-22.

Entering Parameters for an xPC Target Scope Block

xPC Target Scope block parameters define the signals to trace on the scope and trigger modes. When the target application is downloaded to the target PC, the xPC Target kernel automatically creates the scope on the target. No additional definitions are necessary if you want a default scope on the target. This section describes how you can configure the xPC Target Scope block for other scope behavior.

After you add an xPC Target Scope block to your Simulink model, you can enter parameters for this block. To add an xPC Target Scope block, see “Adding an xPC Target Scope Block” on page 3-17. To enter the parameters for an xPC Target Scope block to write signal data to a file on the target PC, see “Entering Parameters for a File Scope” on page 3-33.

There are three types of scopes, `target`, `host`, and `file`. The xPC Target Scope block dialog changes depending on which scope type you are configuring. The following sections describe the procedure depending on the scope type:

- “Entering Parameters for a Target Scope” on page 3-23
- “Entering Parameters for a Host Scope” on page 3-29
- “Entering Parameters for a File Scope” on page 3-33

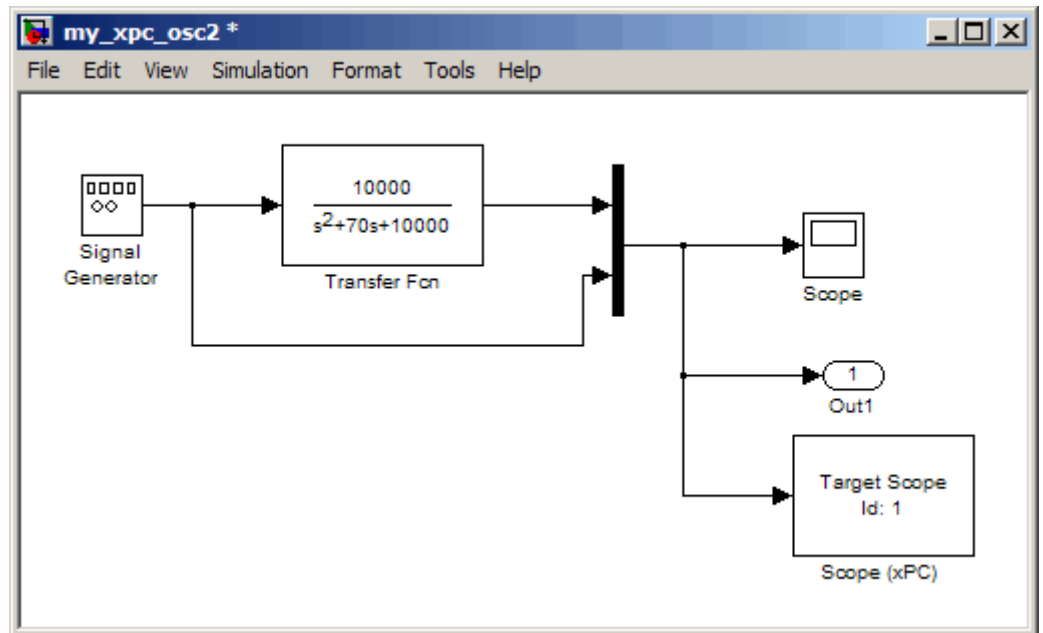
Entering Parameters for a Target Scope

This procedure uses the model `my_xpc_osc2.mdl` as an example.

- 1 In the MATLAB window, type

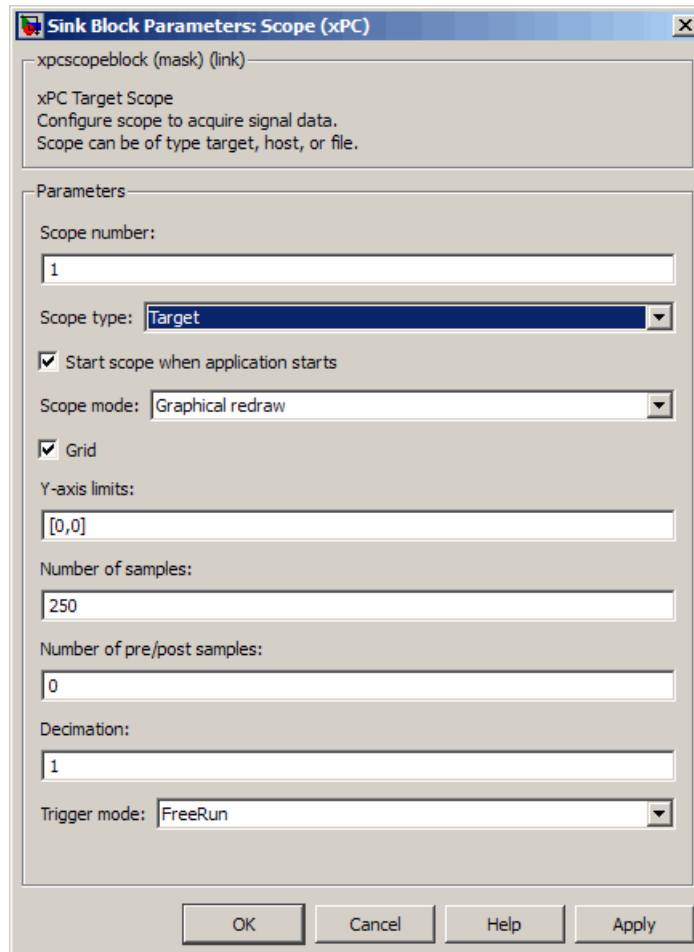
```
my_xpc_osc2
```

The Simulink block diagram opens for the model `my_xpc_osc2.mdl`.



- 2 Double-click the block labeled Scope (xPC).

The Block Parameters: Scope (xPC) dialog box opens.



By default, the target scope dialog is displayed.

- 3 In the **Scope number** box, a unique number to identify the scope is displayed. This number is incremented each time you add a new xPC Target Scope block. Normally, you do not want to edit this value.

This number identifies the xPC Target Scope block and the scope screen on the host or target computers.

- 4 From the **Scope type** list, select Target if it is not already selected.

The updated dialog box is displayed.

- 5 Select the **Start scope when application starts** check box to start a scope when the target application is downloaded and started. The scope window opens automatically.
- 6 From the **Scope mode** list, select Numerical, Graphical redraw, Graphical sliding, or Graphical rolling.

If you have a scope type of Target and a scope mode of Numerical, the scope block dialog adds a **Numerical format** box to the dialog. You can define the display format for the data. See step 7 for a description of how to complete the **Numerical format** box. If you choose not to complete the **Numerical format** box, the xPC Target software displays the signal using the default format of %15.6f, which is a floating-point format, with no label.

- 7 In the **Numerical format** box, enter a label and associated numeric format type in which to display signals. By default, the entry format is floating-point, %15.6f. The **Numerical format** box takes entries of the format

```
'[LabelN] [%width.precision][type] [LabelX]'
```

where

- LabelN is the label for the signal. You can use a different label for each signal or the same label for each signal. This argument is optional.
- width is the minimum number of characters to offset from the left of the screen or label. This argument is optional.
- precision is the maximum number of decimal places for the signal value. This argument is optional.
- type is the data type for the signal format. You can use one or more of the following types:

Type	Description
%e or %E	Exponential format using e or E
%f	Floating point
%g	Signed value printed in f or e format depending on which is smaller
%G	Signed value printed in f or E format depending on which is smaller

- `LabelX` is a second label for the signal. You can use a different label for each signal or the same label for each signal. This argument is optional.

Enclose the contents of the **Numerical format** field in single quotation marks.

For example,

```
'Foo %15.2f end'
```

For a whole integer signal value, enter 0 for the precision value. For example,

```
'Foo1 %15.0f end'
```

For a line with multiple entries, delimit each entry with a command and enclose the entire string in single quotation marks. For example,

```
'Foo2 %15.6f end,Foo3 %15.6f end2'
```

You can have multiple **Numerical format** entries, separated by a comma. If you enter one entry, that entry applies to each signal (scalar expansion). If you enter fewer label entries than signals, the first entry applies to the first signal, the second entry applies to the second signal, and so forth, and the last entry is scalar expanded for the remaining signals. If you have two entries and one signal, the software ignores the second label entry and applies the first entry. You can enter as many format entries as you have signals for the scope. The format string has a maximum length of 100 characters, including spaces, for each signal.

- 8** Select the **Grid** check box to display grid lines on the scope. Note that this parameter is only applicable for target scopes with scope modes of type Graphical redraw, Graphical sliding, or Graphical rolling.
- 9** In the **Y-Axis limits** box, enter a row vector with two elements where the first element is the lower limit of the y-axis and the second element is the upper limit. If you enter 0 for both elements, then scaling is set to auto. Note that this parameter is only applicable for target scopes with scope modes of type Graphical redraw, Graphical sliding, or Graphical rolling.
- 10** In the **Number of samples** box, enter the number of values to be acquired in a data package.

If you select a **Scope mode** of Graphical redraw, this parameter specifies the number of values to be acquired before the graph is redrawn.

If you select a **Trigger mode** other than FreeRun, this parameter can specify the number of samples to be acquired before the next trigger event.

If you select a **Scope mode** of Numerical, the block updates the output every **Number of samples**.

- 11** In the **Number of pre/post samples** box, enter the number of samples to save or skip. Specify a value less than 0 to save this number of samples before a trigger event. Specify a value greater than 0 to skip this number of samples after the trigger event before data acquisition begins.
- 12** In the **Decimation** box, enter a value to indicate that data should be collected at each sample time (1) or at less than every sample time (2 or greater).
- 13** From the **Trigger mode** list, select FreeRun.

If you select FreeRun or Software Triggering, the trigger event is an automatic one. No external trigger specification is required.

If you select Signal Triggering, then, in the **Trigger signal** box, enter the index of a signal. In the **Trigger level** box, enter a value for the signal to cross before triggering. From the **Trigger slope** list, select either, rising, or falling. You do not need to specify scope triggering.

If you select **Scope Triggering**, then in the **Trigger scope number** box, enter the scope number of a Scope block. If you use this trigger mode, you must also add a second Scope block to your Simulink model. You do not need to specify signal triggering.

If you select **Scope Triggering** and want the scope to trigger on a specific sample of the other scope, enter a value in the **Sample to trigger on** box. The default value is 0 and indicates that the triggering scope and the triggered (current) scope start simultaneously. For more information on this field, see “Triggering One Scope with Another Scope to Acquire Data” in the xPC Target User’s Guide.

14 Click **OK**.

15 From the **File** menu, click **Save As**. The model is saved as `my_xpc_osc2.mdl`.

Your next task is to simulate the model. See “Simulating the Model” on page 3-39.

Note As soon as the target application is built and downloaded, the xPC Target kernel creates a scope. If you want to change xPC Target Scope parameters after building the target application or while it is running, you need to assign the scope to a MATLAB variable. To assign the scope object, use the target object method `getscope`. If you use the target object method `getscope` to remove a scope created during the build and download process, and then you restart the target application, the xPC Target kernel recreates the scope.

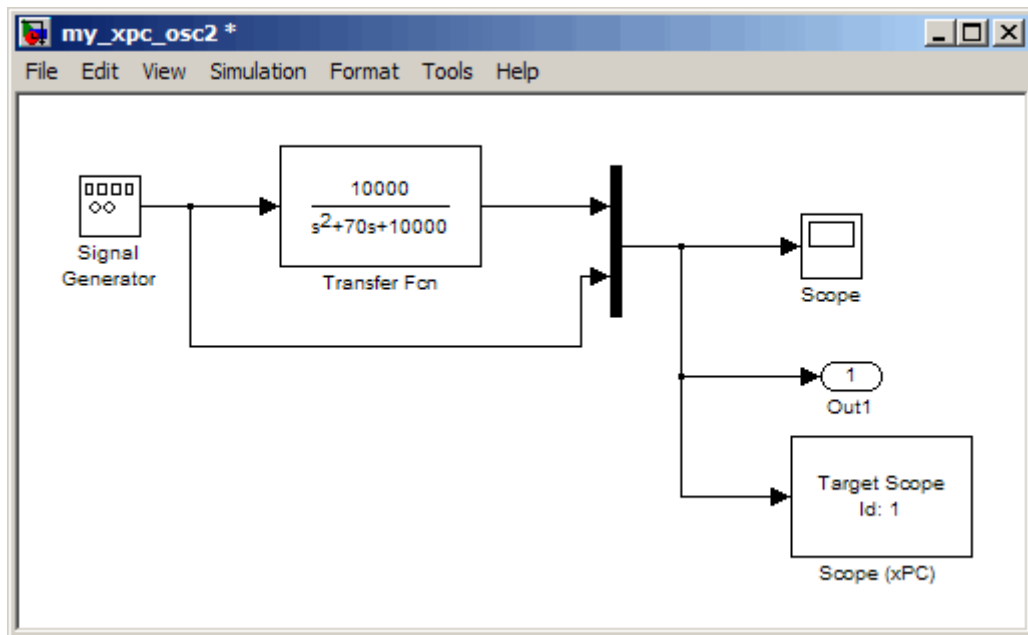
Entering Parameters for a Host Scope

This procedure uses the model `my_xpc_osc2.mdl` as an example.

1 In the MATLAB window, type

```
my_xpc_osc2
```

The Simulink block diagram opens for the model my_xpc_osc2.mdl.



- 2 Double-click the block labeled **Scope (xPC)**.

The Block Parameters: Scope (xPC) dialog box opens.

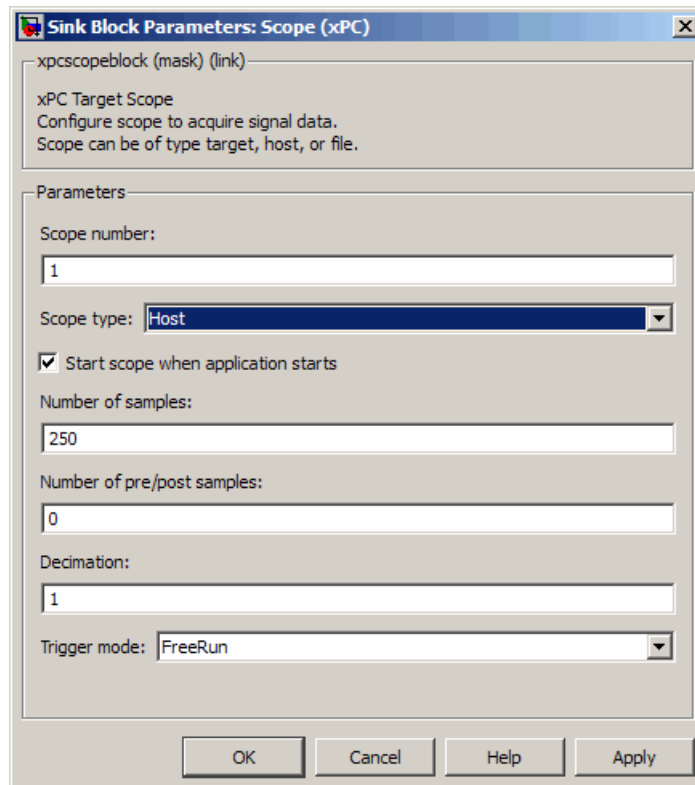
By default, the target scope dialog is displayed.

- 3 In the **Scope number** box, a unique number to identify the scope is displayed. This number is incremented each time you add a new xPC Target scope. Normally, you do not want to edit this value.

This number identifies the xPC Target Scope block and the scope screen on the host or target computers.

- 4 From the **Scope type** list, select Host.

The updated dialog box is displayed.



- 5** Select the **Start scope when application starts** check box to start a scope when the target application is downloaded and started. With a target scope, the scope window opens automatically. With a host scope, you can open a host scope viewer window from xPC Target Explorer.
- 6** In the **Number of samples** box, enter the number of values to be acquired in a data package.
- 7** In the **Number of pre/post samples** box, enter the number of samples to save or skip. Specify a value less than 0 to save this number of samples before a trigger event. Specify a value greater than 0 to skip this number of samples after the trigger event before data acquisition begins.

- 8** In the **Decimation** box, enter a value to indicate that data should be collected at each sample time (1) or at less than every sample time (2 or greater).
- 9** From the **Trigger mode** list, select FreeRun.

If you select FreeRun or Software Triggering, the trigger event is an automatic one. No external trigger specification is required.

If you select Signal Triggering, then in the **Trigger signal** box, enter the index of a signal. In the **Trigger level** box, enter a value for the signal to cross before triggering. From the **Trigger slope** list, select either, rising, or falling. You do not need to specify scope triggering.

If you select Scope Triggering, then in the **Trigger scope number** box, enter the scope number of a Scope block. If you use this trigger mode, you must also add a second Scope block to your Simulink model. You do not need to specify signal triggering.

If you select Scope Triggering and want the scope to trigger on a specific sample of the other scope, enter a value in the **Sample to trigger on** box. The default value is 0 and indicates that the triggering scope and the triggered (current) scope start simultaneously. For more information on this field, see “Triggering One Scope with Another Scope to Acquire Data” in the xPC Target User’s Guide.

- 10** Click **OK**.
- 11** From the **File** menu, click **Save As**. The model is saved as my_xpc_osc2.mdl.

Your next task is to simulate the model. See “Simulating the Model” on page 3-39.

Note As soon as the target application is built and downloaded, the xPC Target kernel creates a scope. If you want to change xPC Target Scope parameters after building the target application or while it is running, you need to assign the scope to a MATLAB variable. To assign the scope object, use the target object method `getscope`. If you use the target object method `remscope` to remove a scope created during the build and download process, and then you restart the target application, the xPC Target kernel recreates the scope.

Entering Parameters for a File Scope

In addition to logging signal data via a host scope, you can also have the xPC Target software save signal data to a file on the target PC C:\ hard drive or 3.5-inch disk drive.

After you add an xPC Target Scope block to your Simulink model, you can configure this block to save a file on the target PC.

Note The signal data file can quickly increase in size. You should examine the file size between runs to gauge the growth rate for the file. If the signal data file grows beyond the available space on the disk, the signal data might be corrupted.

Saving signal data to files is most useful when you are using target PCs as stand-alone xPC Target systems. To access the contents of the signal data file that a file scope creates, use the xPC Target file system object (`xpctarget.fs`) from a host PC MATLAB window. To view or examine the signal data, you can use the `readxpcfile` utility in conjunction with the `plot` function. For further details on the `xpctarget.fs` file system object and the `readxpcfile` utility, see “Logging Signal Data with Target PC Files and File Systems” in the *xPC Target User’s Guide*. Saving signal data to files lets you recover signal data from a previous run in the event of system failure (such as a system crash).

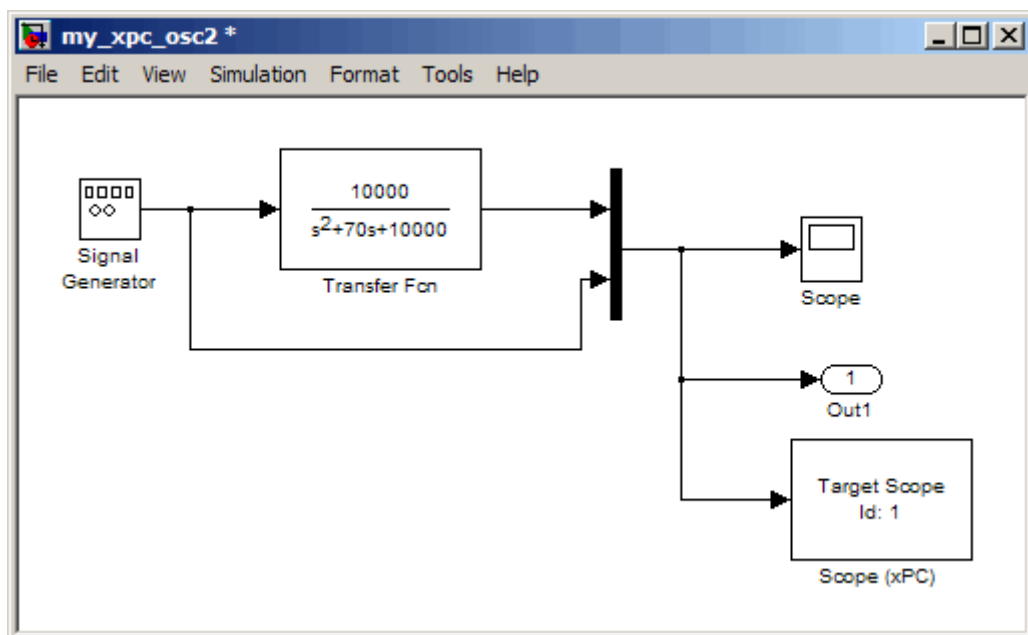
To add an xPC Target Scope block, see “Adding an xPC Target Scope Block” on page 3-17.

This procedure uses the model `my_xpc_osc2.mdl` as an example.

- 1 In the MATLAB window, type

`my_xpc_osc2`

The Simulink block diagram opens for the model `my_xpc_osc2.mdl`.



- 2 Double-click the block labeled **Scope (xPC)**.

The Block Parameters: Scope (xPC) dialog box opens.

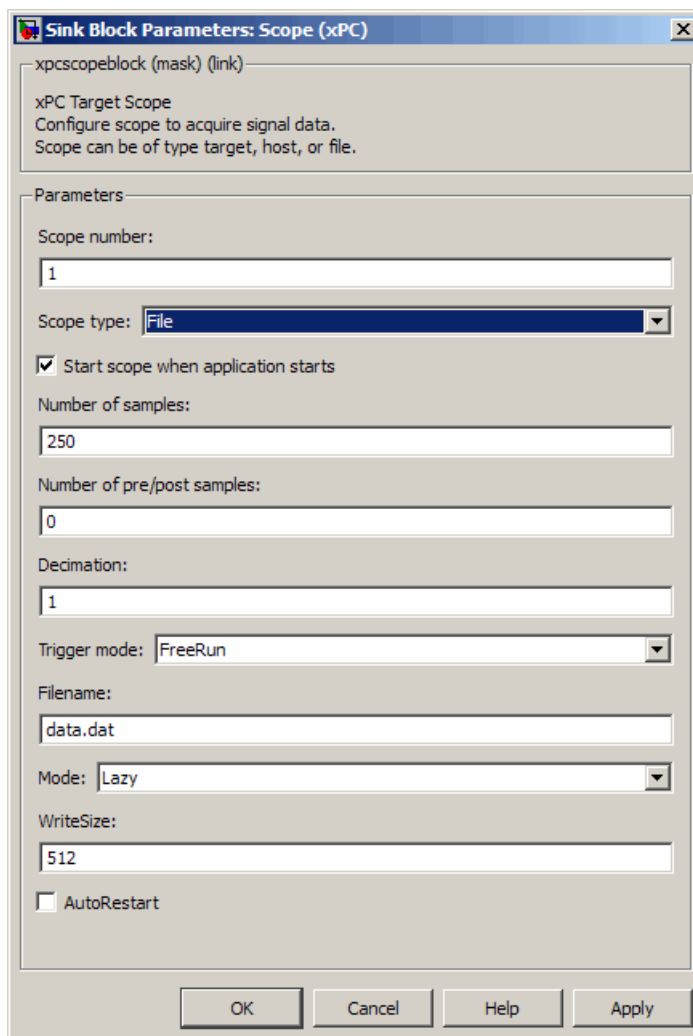
By default, the target scope dialog is displayed.

- 3 In the **Scope number** box, a unique number to identify the scope that is displayed. This number is incremented each time you add a new xPC Target scope. Normally, you do not want to edit this value.

This number identifies the xPC Target Scope block and the scope screen on the host or target computer.

- 4 From the **Scope type** list, select **File**.

The updated dialog box is displayed.



- 5 Select the **Start scope when application starts** check box to start a scope when the target application is downloaded and started. The scope window opens automatically.

- 6** In the **Number of samples** box, enter the number of values to be acquired in a data package. This parameter works in conjunction with the **AutoRestart** check box. If the **AutoRestart** box is selected, the file scope collects data up to **Number of samples**, then starts over again, overwriting the buffer. If the **AutoRestart** box is not selected, the file scope collects data only up to **Number of samples**, then stops.
- 7** In the **Number of pre/post samples** box, enter the number of samples to save or skip. Specify a value less than 0 to save this number of samples before a trigger event. Specify a value greater than 0 to skip this number of samples after the trigger event before data acquisition begins.
- 8** In the **Decimation** box, enter a value to indicate that data should be collected at each sample time (1) or at less than every sample time (2 or greater).

Note This value is the same as **Decimation** in the MATLAB interface.

- 9** From the **Trigger mode** list, select FreeRun, Software Triggering, Signal Triggering, or Scope Triggering.

If you select FreeRun or Software Triggering, the trigger event is an automatic one. No external trigger specification is required.

If you select Signal Triggering, then in the **Trigger signal** box, enter the index of a signal. In the **Trigger level** box, enter a value for the signal to cross before triggering. From the **Trigger slope** list, select either, rising, or falling. You do not need to specify scope triggering.

If you select Scope Triggering, then in the **Trigger scope number** box, enter the scope number of a Scope block. If you use this trigger mode, you must also add a second Scope block to your Simulink model. You do not need to specify signal triggering.

If you want the scope to trigger on a specific sample of the other scope, enter a value in the **Sample to trigger on** box. The default value is 0 and indicates that the triggering scope and the triggered (current) scope start

simultaneously. For more information on this field, see “Triggering One Scope with Another Scope to Acquire Data” in the *xPC Target User’s Guide*.

- 10** In the **Filename** box, enter a name for the file to contain the signal data. By default, the target PC writes the signal data to a file named `C:\data.dat`.
- 11** From the **Mode** list, select either `Lazy` or `Commit`. Both modes open a file, write signal data to the file, then close that file at the end of the session. With the `Commit` mode, each file write operation simultaneously updates the FAT entry for the file. This mode is slower, but the file system always knows the actual file size. With the `Lazy` mode, the FAT entry is updated only when the file is closed and not during each file write operation. This mode is faster, but if the system crashes before the file is closed, the file system might not know the actual file size (the file contents, however, will be intact). If you experience a system crash, you can expect to lose a **WriteSize** amount of data.
- 12** In the **WriteSize** box, enter the block size, in bytes, of the data chunks. This parameter specifies that a memory buffer of length **Number of samples** write data to the file in **WriteSize** chunks. By default, this parameter is 512 bytes, which is the typical disk sector size. Using a block size that is the same as the disk sector size provides optimal performance.

If you experience a system crash, you can expect to lose a **WriteSize** amount of data.

- 13** In the **Number of samples** box, enter the number of values to be acquired in a data package.
- 14** Select the **AutoRestart** check box to enable the file scope to collect data up to **Number of samples**, then start over again, appending the new data to the end of the signal data file. Clear the **AutoRestart** check box to have the file scope collect data up to **Number of samples**, then stop.

If the named signal data file already exists, the xPC Target software overwrites the old data with the new signal data.

Your next task is to simulate the model. See “Simulating the Model” on page 3-39.

The following sections describe how to simulate your model and run the target application. With file scopes, the xPC Target software generates a signal data file on the target PC after you run the target application. For further details on working with these files, see “Logging Signal Data with Target PC Files and File Systems” in the *xPC Target User’s Guide*.

Simulating the Model

In this section...
“Simulating the Model with Simulink” on page 3-39
“Simulating the Model from MATLAB” on page 3-41

Simulating the Model with Simulink

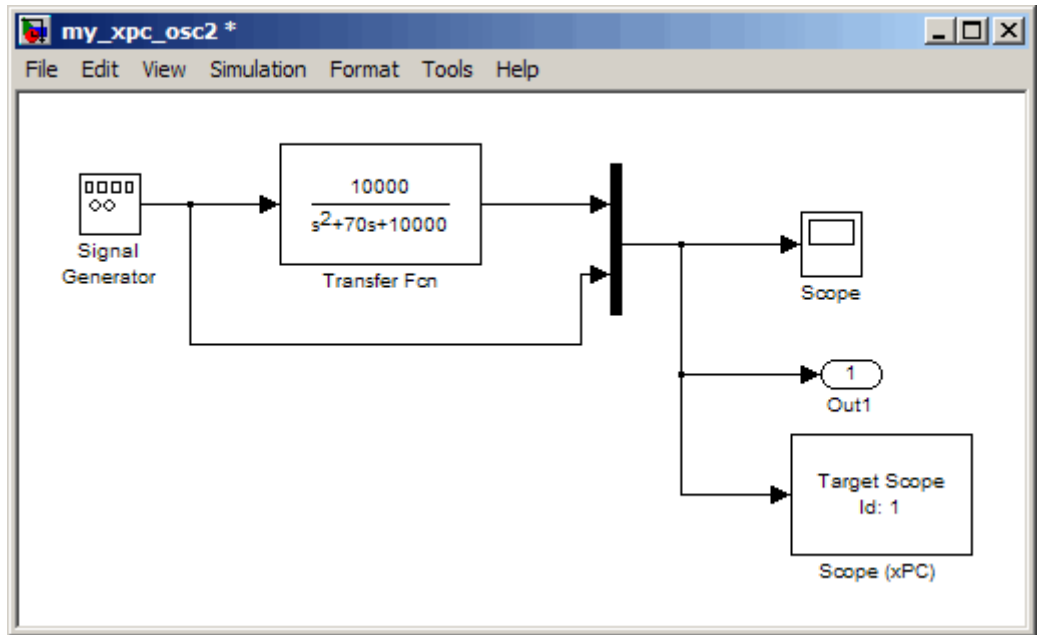
You use Simulink in normal mode to observe the behavior of your model in nonreal time. After you load your Simulink model, you can run a simulation. This procedure uses the Simulink model `my_xpc_osc2.mdl` as an example and assumes you have already loaded that model. To create this model, see “Creating a Simple Simulink Model” on page 3-2.

For procedures to run your target application in real time, see “Running the Target Application” on page 3-57.

- 1 In the MATLAB window, type

```
my_xpc_osc2
```

MATLAB loads the oscillator model and displays the Simulink block diagram, as shown below.



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

Simulink opens a scope window.

3 From the **Simulation** menu, click **Normal**, and then click **Start**.

The **Scope1** window displays a trace of the signal data.



- 4 You can either let the simulation run to its stop time, or stop the simulation manually. To stop the simulation manually, from the **Simulation** menu, click **Stop**.

Your next task is to create an xPC Target application. See “xPC Target Application” on page 3-45.

Simulating the Model from MATLAB

You run a simulation of your Simulink model to observe the behavior of the model in nonreal time.

After you load your Simulink model into the MATLAB workspace, you can run a simulation. This procedure uses the Simulink model `my_xpc_osc2.mdl` as an example and assumes you have already loaded that model. To create this model, see “Creating a Simple Simulink Model” on page 3-2.

- 1 In the MATLAB window, type

```
sim('my_xpc_osc2')
```

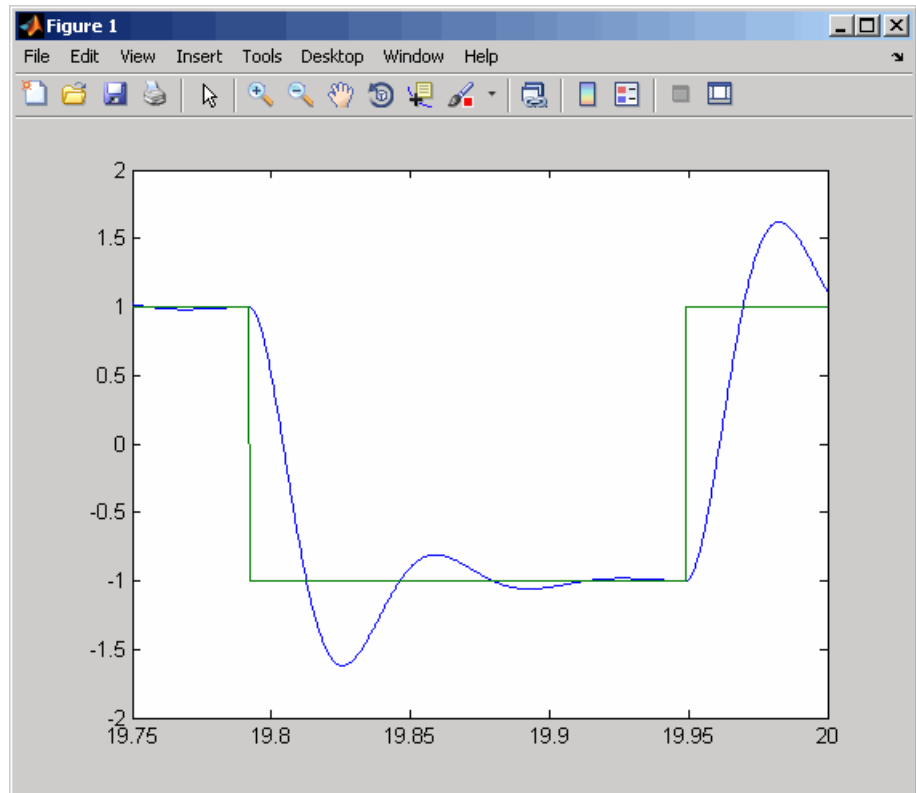
Simulink runs a simulation in normal mode through to completion. You cannot manually stop the simulation. See the online Simulink documentation for further information on using the `sim` command.

2 After Simulink finishes the simulation, type

```
plot(tout,yout)
```

You entered the MATLAB variables `tout` and `yout` in the **Data I/O** pane on the Configuration Parameters dialog box. The signals are logged to memory through Outport blocks. To add an Outport block, see “Adding a Simulink Outport Block” on page 3-10 and “Entering Parameters for the Outport Blocks” on page 3-13.

MATLAB opens a plot window and displays the output response. The signal from the signal generator is added to the Output block and shown in the figure below.



Note When your target application is running in real time, data is not saved to the variables `tout` and `yout`. Instead, data is saved in the target PC 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.

Your next task is to create a target application. See “xPC Target Application” on page 3-45.

xPC Target Application

In this section...

“Booting the Target PC” on page 3-45

“Troubleshooting the Boot Process” on page 3-47

“Entering the Simulink® Coder Parameters” on page 3-47

“Building and Downloading the Target Application” on page 3-52

“Troubleshooting the Build Process” on page 3-54

“Increasing the Time-Out Value” on page 3-55

“xPC Target Options Node” on page 3-56

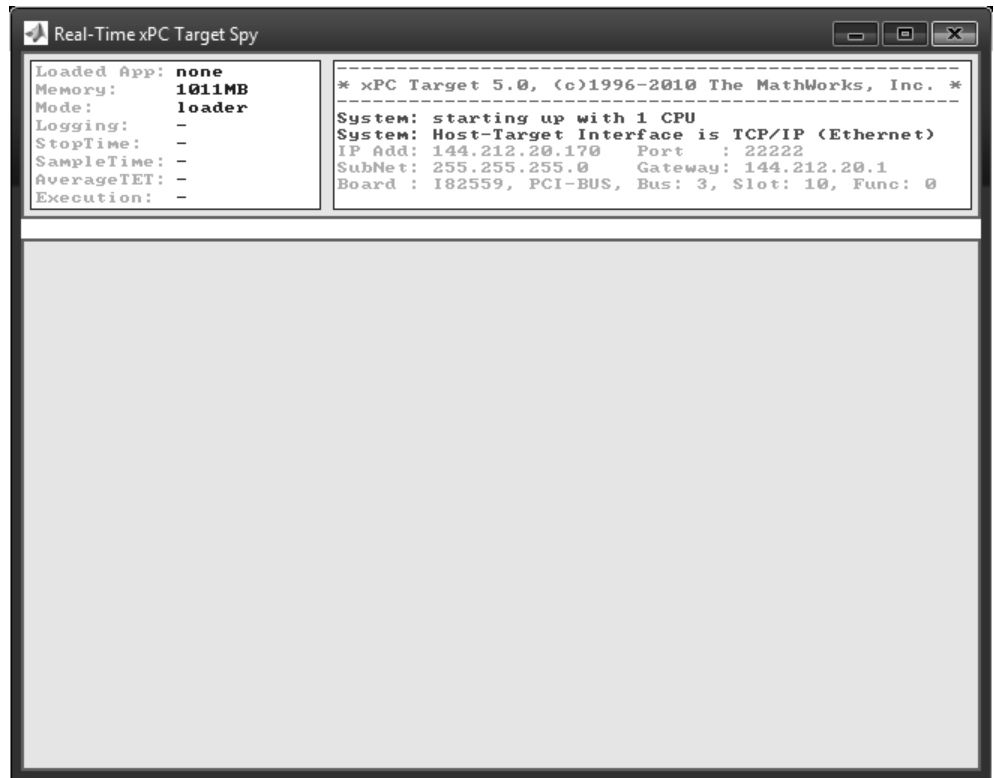
Booting the Target PC

Booting the target computer loads and starts the xPC Target kernel on the target PC. The loader then waits for the xPC Target software to download your target application from the host PC.

After you have configured the xPC Target product using the xPC Target Explorer and created a target boot disk for that setup, you can boot the target PC. You need to boot the target computer before building your target application because the build process automatically downloads your target application to the target PC. Be sure that you have followed the instructions from Chapter 2, “Installation and Configuration” before continuing.

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

The target PC displays the following screen.



In the example above, the status window shows that the kernel is in loader mode and waiting to load a target application. 1 MB of memory is reserved for the application, 3 MB is used by the kernel, and 28 MB is available from a total of 32 MB. The xPC Target kernel uses the 28 MB for the heap, running scopes, and acquiring data.

Your next task is to enter the simulation and real-time run parameters for Simulink Coder. See “Entering the Simulink® Coder Parameters” on page 3-47.

Troubleshooting the Boot Process

Possible Problem

When booting the target PC, it might display a message like the following

```
xPC Target 4.X loading kernel..@@@@@@@@@@@@@@@@@@@@
```

The target PC displays this message when it cannot read and load the kernel from the target boot disk. The probable cause is a bad disk.

Solution. If you have a 3.5-inch target boot disk, reformat the disk or use a new formatted floppy disk and create a new target boot disk. If you have a CD target boot disk, create a new disk.

Possible Problem

When booting the target PC, you get a message similar to the following:

```
Not a bootable medium or NTLDR is missing
```

Selecting either **DOS Loader** or **Standalone** mode instead of **Boot Floppy** mode can cause this message.

Solution. If the xPC Target Explorer window is not already open, open it. In the MATLAB Command Window, type `xpcexplr`. In the xPC Target Explorer **xPC Target Hierarchy** pane, select a target PC Configuration node. For example, select the Configuration node for TargetPC1. In the configuration pane, select your desired boot mode. Create a new boot disk or network boot image.

Entering the Simulink Coder Parameters

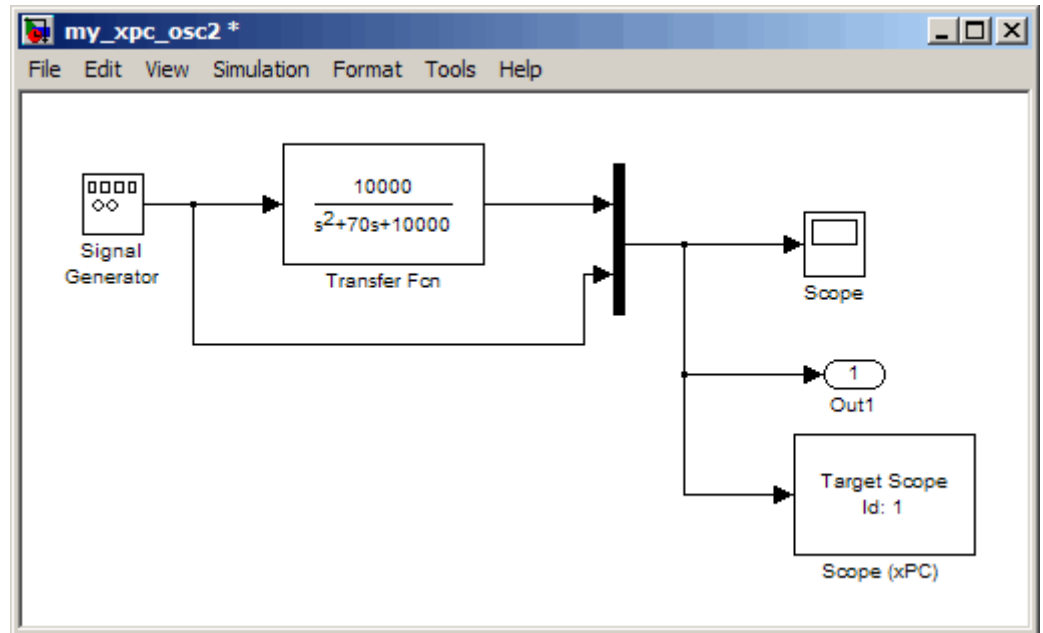
You enter the simulation and real-time run parameters in the Configuration Parameters dialog box. These parameters give information to Simulink Coder on how to build the target application from your Simulink model.

After you load a Simulink model and boot the target PC, you can enter the simulation parameters. This procedure uses the Simulink model `my_xpc_osc2.mdl` as an example and assumes you have already loaded that model (see “Simulink Model” on page 3-2).

- 1 In the MATLAB window, type

```
my_xpc_osc2
```

MATLAB loads the oscillator model and displays the Simulink block diagram, as shown below.



- 2 In the Simulink window, and from the **Simulation** menu, click **Configuration Parameters**.

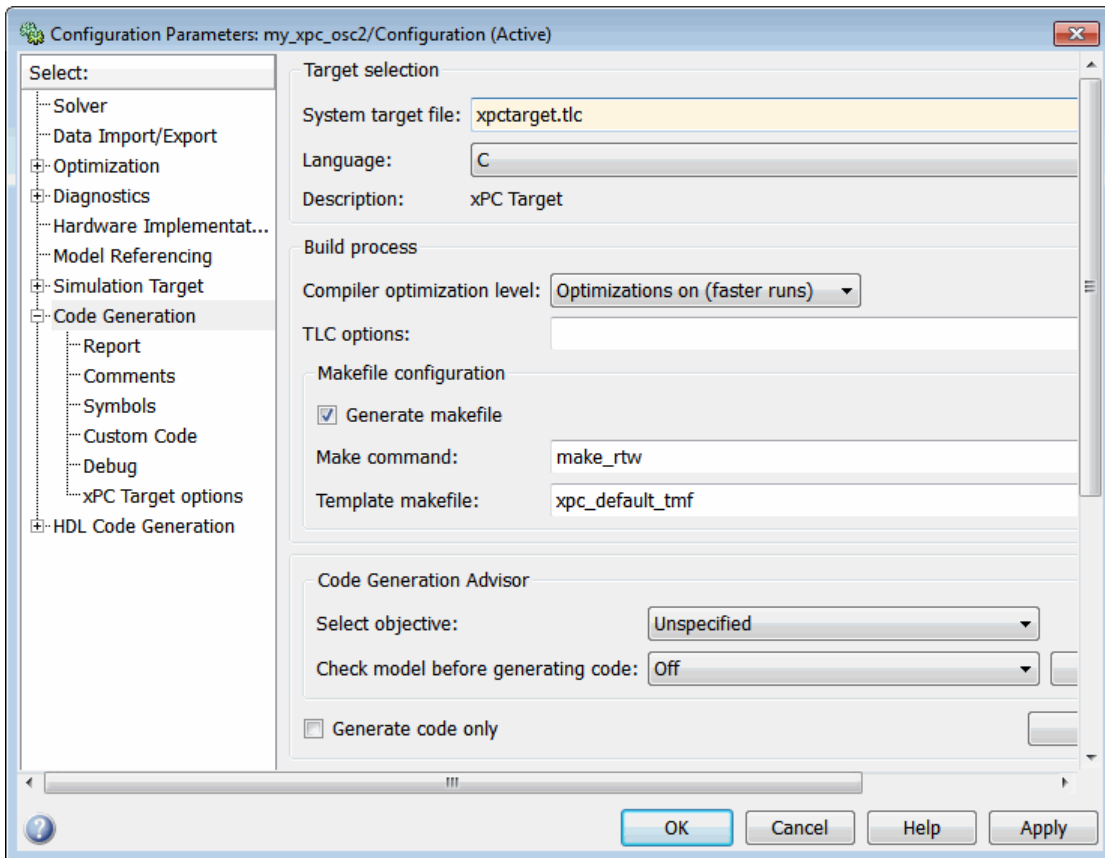
The Configuration Parameters dialog box is displayed for the model.

- 3 Click the **Code Generation** node.

The code generation pane opens.

- 4 To build a basic target application, in the **Target selection** section, click the **Browse** button at the **System target file** list. Click `xpctarget.tlc`, and then click **OK**.

The system target file `xpctarget.tlc`, the template makefile `xpc_default_tmf`, and the make command `make_rtw` are automatically entered into the page. The **xPC Target options** node appears in the left pane. The code generation pane should now look like the figure shown.



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

Note that if the Embedded Coder is not installed and you select `xpctargetert.tlc`, the build fails.

- 5** In the left pane, click the **xPC Target options** node.

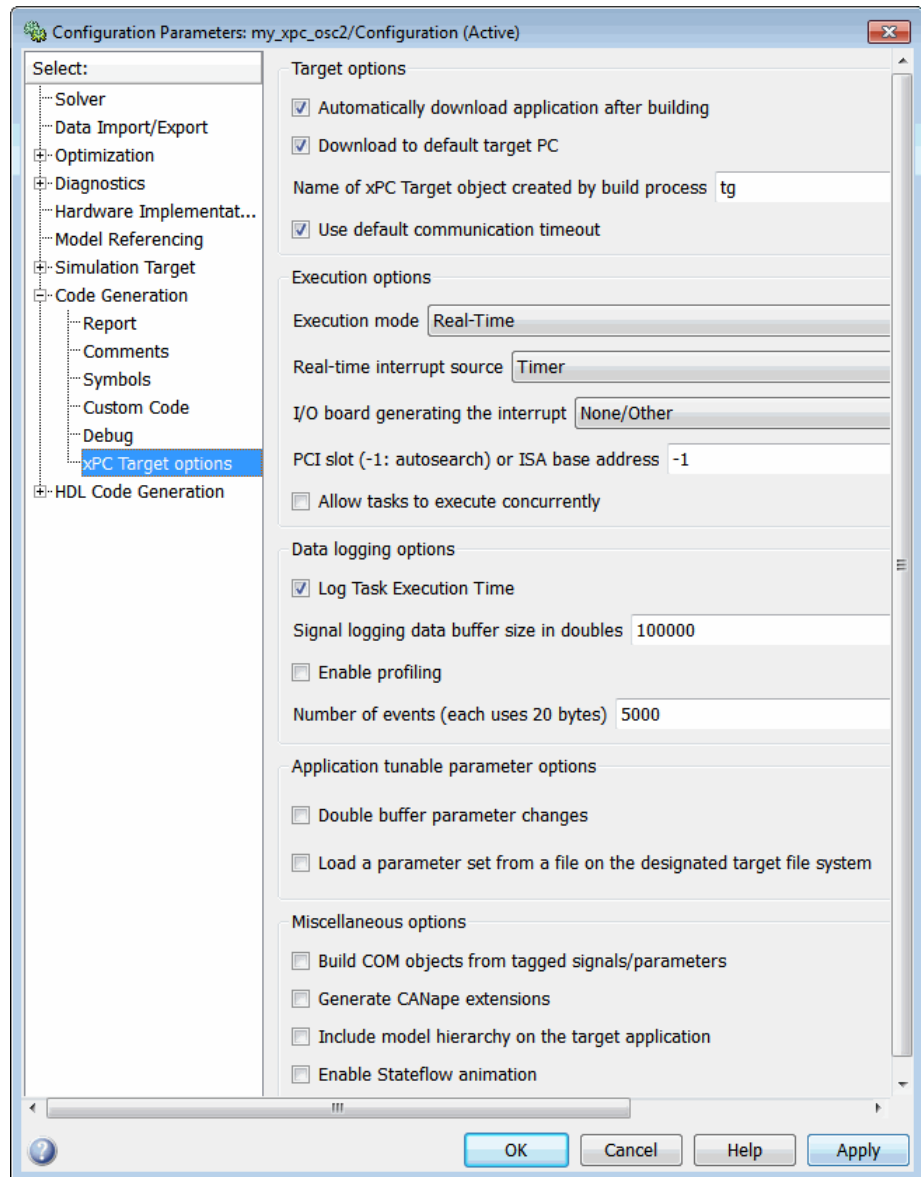
The associated pane is displayed. These are model-level configuration parameters that you can set for your model. See “Configuration Parameters” in the *xPC Target User’s Guide* for a description of the options on this node.

- 6** From the **Execution mode** list, select either **Real-Time** or **Freerun**. The option **Freerun** is similar to a simulation, but with the generated code. It runs the target application as fast as it can. However, unlike a simulation, the **Freerun** mode of the xPC Target software does not support variable-step solvers.
- 7** From the **Real-time interrupt source** list, select a source. The default value is **Timer**.
- 8** Select the **Log Task Execution Time** check box to log task execution times to the target object property `tg.TETlog`.

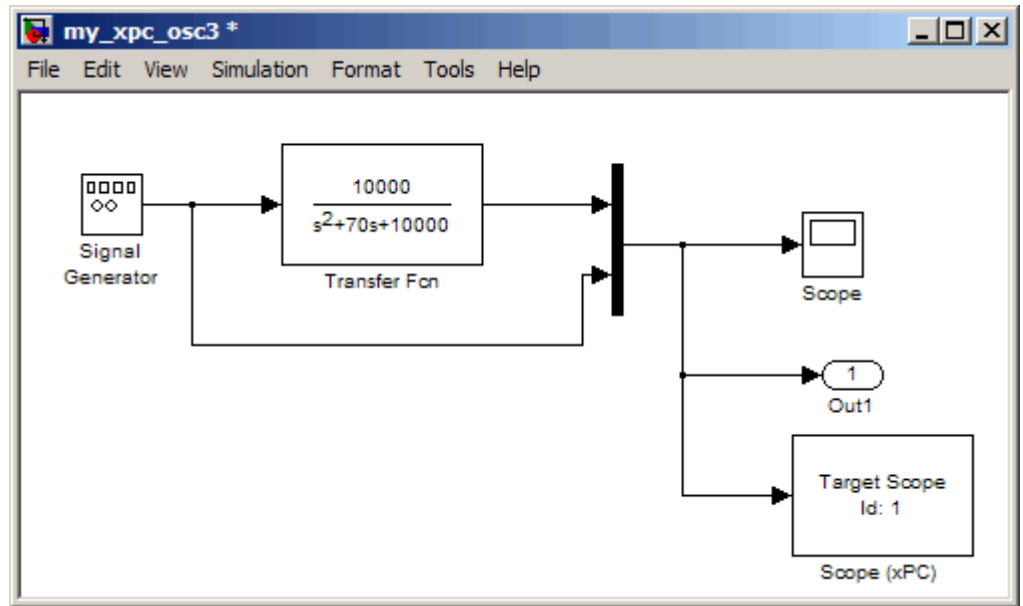
The task execution time is the time in seconds to complete calculations for the model equations and post outputs during each sample interval. If you do not select this box, your average TET value appears as Not a Number (NaN).

- 9** 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.
- 10** In the **Name of xPC Target 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`.

The **Code Generation** pane should now look like the figure shown.



- 11 Click **OK**.
- 12 From the **File** menu, click **Save as**. Enter a filename. For example, enter `my_xpc_osc3` and then click **Save**.



Your next task is to create (build) the target application. See “Building and Downloading the Target Application” on page 3-52.

Building and Downloading the Target Application

You use the xPC Target build process to generate C code, compile, link, and download your target application to the target PC.

After you enter your changes in the Configuration Parameters dialog box, you can build your target application. This procedure uses the Simulink model `my_xpc_osc3.mdl` as an example. To create this model, see “xPC Target Application” on page 3-45. By default, the build procedure downloads the target application to the default target PC, as designated in the xPC Target Explorer. See “xPC Target Options Node” on page 3-56 for further details on setting the target PC for a target application.

- 1 In the MATLAB window, type

```
my_xpc_osc3
```

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

- 2 In the Simulink window and from the **Tools** menu, select **Code Generation**. From the code generation submenu, click **Build Model**.

After the compiling, linking, and downloading process, a target object is created in your MATLAB workspace. The default name of the target object is `tg`. For more information about the target object, see “Targets and Scopes in the MATLAB Interface” in the *xPC Target User’s Guide*.

On the host computer, MATLAB displays lines like the following after a successful build process:

```
### Starting xPC Target build procedure for model:
my_xpc_osc3
. . .
### Successful completion of xPC Target build procedure for
model: my_xpc_osc3
```

The target PC displays the following information:

Loaded App: xpc_osc3	Scope: 1, created, type is target
Memory: 123MB	Scope: 1, signal 0 added
Mode: RT, single	Scope: 1, signal 1 added
Logging: t x y tet	Scope: 1, NumSamples set to 500
StopTime: 20 d	Scope: 1, trigger level set to 0.000000
SampleTime: 0.00025	Scope: 1, TriggerScope set to 1
AverageTET: -	Scope: 1, lower y-axis limit set to 0.000000
Execution: stopped	Scope: 1, upper y-axis limit set to 0.000000
	System: initializing application finished

- 3 In the MATLAB window, type

```
tg
```

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

If you do not have a successful build, see “Troubleshooting the Build Process” on page 3-54.

Note If you accidentally download a target application built with a different version of the xPC Target product than the one on the target PC, the following error message will appear on the target PC monitor and the download will fail.

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

To prevent this version mismatch, always rebuild target applications with each new xPC Target release.

Your next task is to run the target application in real time on the target PC. See “Running the Target Application” on page 3-57.

Troubleshooting the Build Process

If the host PC and target PC are not properly connected, or you have not correctly entered the environment properties, the download process is terminated after about 5 seconds with a time-out error. Be sure that you have followed the instructions outlined in Chapter 2, “Installation and Configuration” before continuing.

To correct the problem, use the following procedure:

- 1** If the xPC Target Explorer is not already up, in the MATLAB window, type

```
xpcexplr
```

The xPC Target Explorer window opens.

- 2** For the target PC in question, check the RS-232 or TCP/IP parameters. If necessary, make changes to the communication properties and recreate the target boot disk.

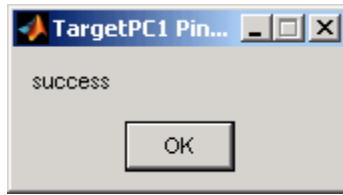
In some cases, the download might have completed successfully even though you get a time-out error. To detect this, wait until the target screen displays

```
System:initializing application finished.
```

- 3** In the xPC Target Explorer window

- a Select the target PC in question
- b From the menu bar, select **Target > Ping Target**.

For a working connection between the host PC and target PC, xPC Target Explorer displays a pop-up dialog box.



- 4 Right-click the target PC in question and select Connect.

If the connection is resumed, the connection is all right. If the connection is timing out consistently for a particular model, the time-out needs to be increased. See “Increasing the Time-Out Value” on page 3-55.

For information on setting up the xPC Target software environment, see either “Environment Properties for Serial Communication” on page 2-37 or “Environment Properties for Network Communication” on page 2-29, and then see “Booting Target PCs from Boot Floppy Disk” on page 2-51.

Increasing the Time-Out Value

By default, if the host PC does not get a response from the target PC after downloading a target application and waiting about 5 seconds, the host PC software times out. On the other hand, the target PC responds only after downloading and initializing the target application.

Usually 5 seconds is enough time to download a target application, but in some cases it may not be sufficient. The time to download a target application mostly depends on your I/O hardware. For example, thermocouple hardware takes longer to initialize. In this case, even though the target PC is fine, a false time-out is reported.

You can increase the time-out value in one of the following ways.

- At the model level, open the **Simulink > Configuration Parameters** dialog box and navigate to the **xPC Target options** node. Clear the **Use default communication timeout** parameter and enter a new desired time-out value in the **Specify the communication timeout in seconds** parameter. For example, enter 20 to increase the value to 20 s.
- At the target application level, use the target application set method to set the `CommunicationTimeOut` property to the desired time-out value. For example, to increase the value to 20 s:

```
tg.set('CommunicationTimeOut',20)
```

For both methods, the host PC waits for about 20 seconds before declaring that a time-out has occurred. Note that it does not take 20 seconds for every download. The host PC polls the target PC about once every second, and if a response is returned, declares success. Only in the case where a download really fails does it take the full 20 seconds.

xPC Target Options Node

The configuration parameters **xPC Target Options** node appears when you select and apply one of the xPC Target options to the Simulink model **Configuration Parameter > Code Generation > System target file** parameter:

- `xpctarget.tlc`
Generate code for an xPC Target target.
- `xpctargetert.tlc`
Generate code for an xPC Target target using the required Embedded Coder software.

The **xPC Target Options** node allow you to specify how the software generates the target application. You might need to enter and select these options before you create (build) a target application. However, the default values of these options are reasonable for target application creation. See “Configuration Parameters” in the *xPC Target User’s Guide* for a description of the options on this node.

Running the Target Application

In this section...

“Introduction” on page 3-57

“Control with xPC Target Explorer” on page 3-57

“Control with MATLAB Commands” on page 3-67

“Control with Simulink External Mode” on page 3-69

Introduction

During the build process, the xPC Target software creates a target object that represents the target application running on the target PC. The target object is defined by a set of properties and associated methods. You control the target application and computer by setting the target object properties.

For procedures to simulate your model in nonreal time, see “Simulating the Model” on page 3-39.

Control with xPC Target Explorer

This procedure assumes you have created an xPC Target boot disk and you have booted the target PC. See “Booting Target PCs from Boot Floppy Disk” on page 2-51. While the xPC Target Explorer gives you access to build and download a target application, this procedure begins with a target application already downloaded to the target PC (see “xPC Target Application” on page 3-45). For a description of how to download a prebuilt target application, see “Downloading and Running Target Applications on a Target PC” on page 3-61.

- 1 In the MATLAB window, type

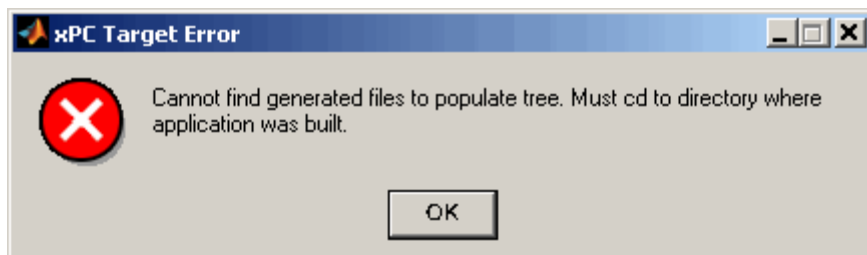
```
xpcexplr
```

The xPC Target Explorer window opens.

- 2 To connect to the target PC, right-click the target PC icon for which you have downloaded the application and select **Connect**.

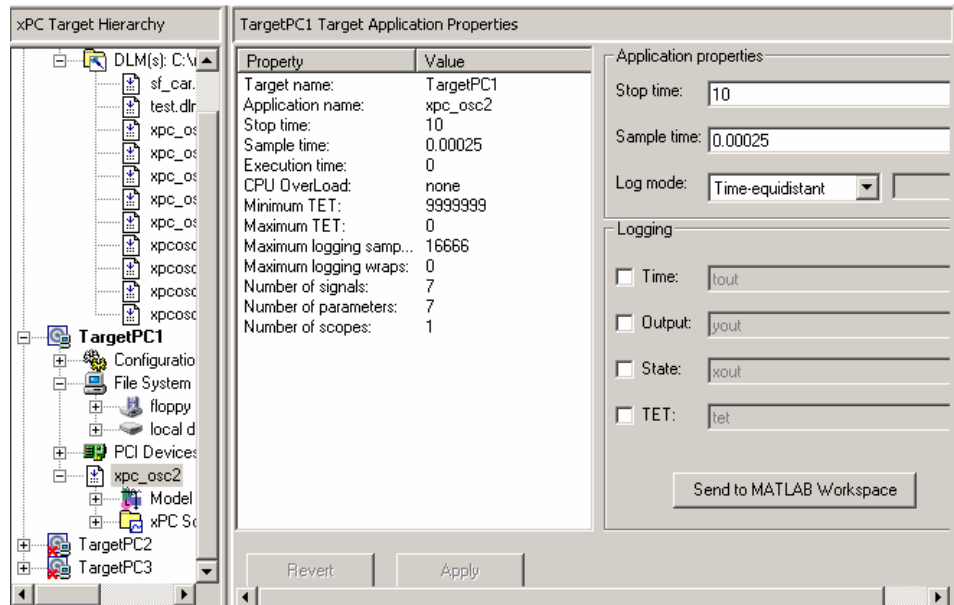
To view the model hierarchy for the downloaded application, one of the following must be true:

- You must be in the same folder in which you build the target application. Otherwise, xPC Target Explorer returns an error.



- When you built the target application, you selected the **Include model hierarchy on the target application** check box in the xPC Target Options pane.

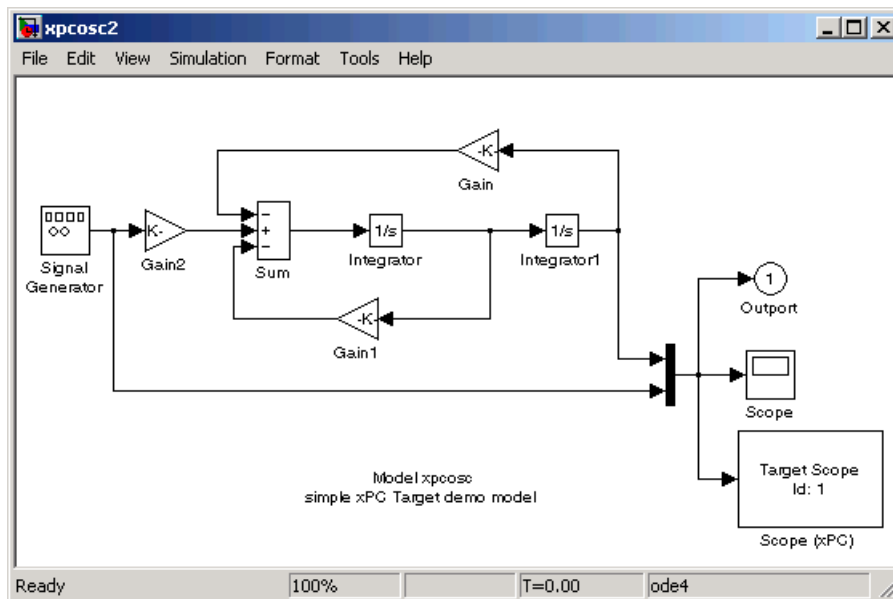
If you are in the appropriate folder, a node for the target application appears in the **xPC Target Hierarchy** under the target PC node and displays information about the previously loaded target application.



Note, if, in your current folder, you have a prebuilt target application that you want to download to the target PC, left-click and drag the desired target application to the target PC to which you want to download the target application.


- 3 If you want to rebuild the current target application, in the xPC Target Explorer window, right-click the target application node and select **Go To Simulink Model**.

The Simulink window opens for that model.

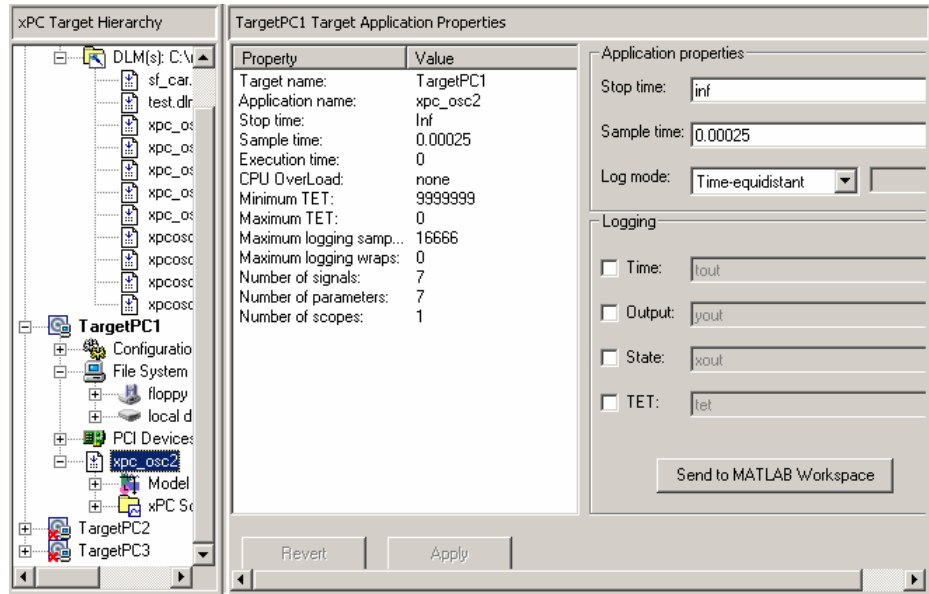


- 4 To rebuild the target application, in the Simulink window and from the **Tools** menu, select **Code Generation**. From the code generation submenu, click **Build Model**.

The xPC Target software recompiles, links, and downloads the target application to the target PC.

- 5 Start the target application. For example, in the xPC Target Explorer window, select the downloaded target application.
- 6 From the toolbar, click the **Start Application** button . The target application begins running on the target PC, and stops when it reaches the stop time.
- 7 With the target application still selected in the Target Hierarchy pane of xPC Target Explorer, enter a new value for the **Stop time** value for the application and click **Apply**. For example,

inf



8 Again, click the **Start Application** button. The target application now runs until you stop it.

9 From the toolbar, click the **Stop Application** button .

See also “Signal Tracing with xPC Target Explorer” and “Signal Logging with xPC Target Explorer” in the *xPC Target User’s Guide*.

Downloading and Running Target Applications on a Target PC

This topic describes how to change folder to one that contains the prebuilt target applications (DLMs) you want to download to your target PCs and how to download and run a prebuilt target application. To view the model hierarchy, you must be in the same folder in which you build the target application. This topic assumes the following:

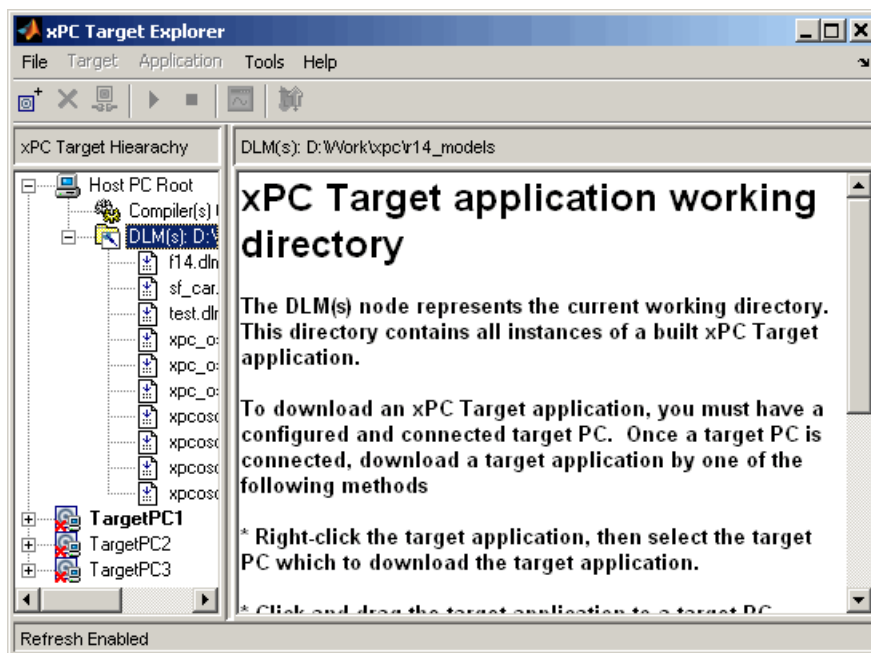
- In your current working folder, you have a prebuilt target application that you want to download to a target PC.
- You have installed xPC Target software and booted the target PC to which you want to download a target application.

- You have a physical connection between the xPC Target Explorer host machine and the target PC to which you want to download a target application.
- 1 In the xPC Target Explorer, left-click the **File** menu. (Note that you can also change this folder from the MATLAB window or by right-clicking the DLM node.)
 - 2 Select **Change Host PC Current Working Directory**.

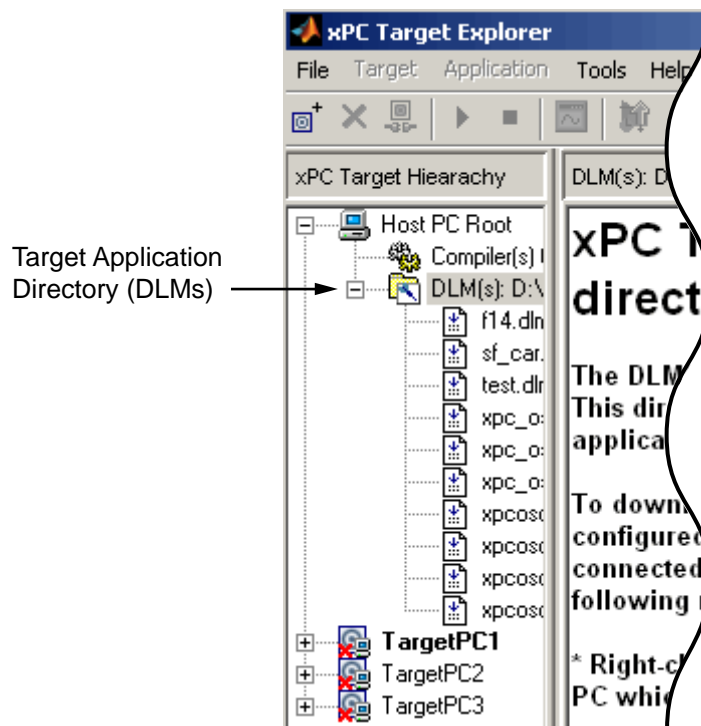
A browser is displayed.

- 3 Browse to the folder that contains the prebuilt target applications you want.
- 4 Click **OK**.


A list of the prebuilt target applications appears, as shown.



- 5 In xPC Target Explorer, check that the DLM(s) node in the **xPC Target Hierarchy** has the pathname of the folder that contains the prebuilt target application you want to download to the target PC.

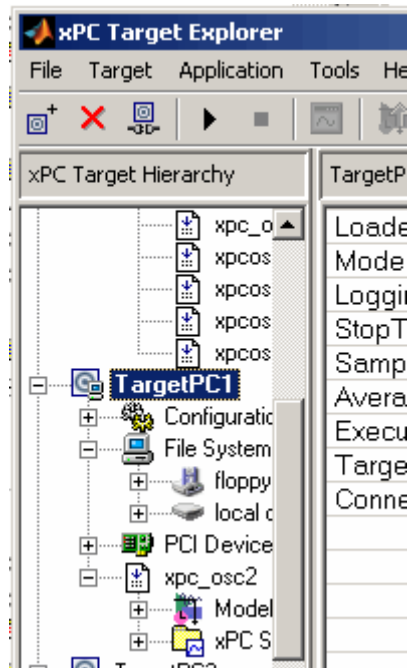


- 6 Right-click a target PC that you booted with xPC Target software, for example, TargetPC1.
- 7 Select **Connect**.

The target PC icon changes and the red X is removed . The target PC information changes to reflect file system and PCI device information.

- 8 Left-click and drag the desired target application to the target PC to which you want to download the target application.

xPC Target Explorer downloads the target application to the target PC. A node for the target application appears in the **xPC Target Hierarchy** under the target PC node.

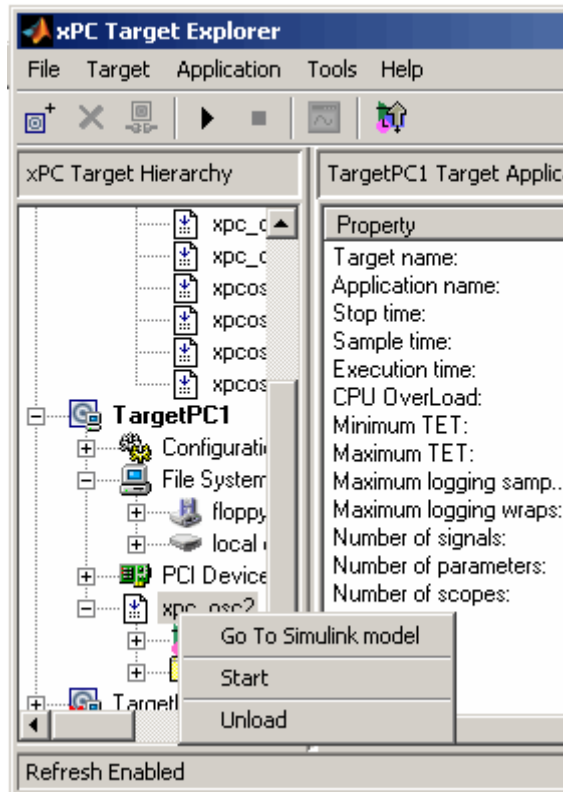


Alternatively, you can now drag a prebuilt target application, DLM, to a target PC icon. If a connection does not already exist, xPC Target Explorer optionally creates a connection to that target PC.

Note If you want to rebuild or revisit the model, click the **Go to Simulink Model** button. The Simulink model for the target application appears.

- 9 In xPC Target Explorer, right-click the downloaded target application node. For example, xpcosc.

The context menu appears and lists the operations you can perform on the target application.



10 Select **Start**.

xPC Target Explorer starts running the loaded target application.

11 Stop the target application (described here) or let the target application run to the end. To stop the target application, right-click the target application node (for example, xpcosc) and select **Stop** from the list.

See also “Signal Tracing with xPC Target Explorer” and “Signal Logging with xPC Target Explorer” within “Signals and Parameters” in the *xPC Target User’s Guide*.

Manipulating Target Application Properties

This topic describes how to manipulate target application properties. It assumes that you have already downloaded the target application `xpcosc` to a target PC.

- 1** In xPC Target Explorer, select the node of the loaded target application in which you are interested. For example, `xpcosc`.

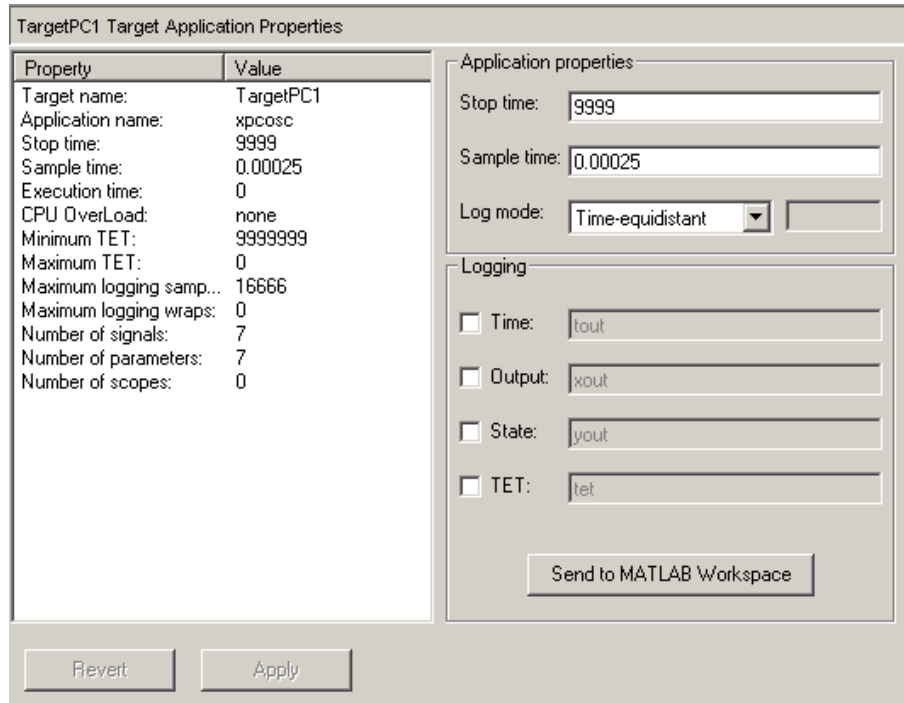
The right pane changes to the target application properties pane for the application.

- 2** In this pane, you can change the following application properties:

- **Stop time**
- **Sample time**
- **Log mode**

See “User Interaction” on page 1-28 for limitations on changing sample times.

- 3** Change the **Stop time** parameter to 9999. Click **Apply**. For example,



Control with MATLAB Commands

You run your target application in real time to observe the behavior of your model with generated code.

After the xPC Target software downloads your target application to the target PC, you can run the target application. This procedure uses the Simulink model `my_xpc_osc3.mdl` as an example, and assumes you have created and downloaded the target application for that model. It also assumes that you have assigned `tg` to the appropriate target PC.

- 1 In the MATLAB window, type any of

`+tg` or `tg.start` or `start(tg)`

The target application starts running on the target PC. In the MATLAB window, the status of the target object changes from stopped to running.

```

xPC Object
  Connected    = Yes
  Application  = my_xpc_osc3
  Mode        = Real-Time Single-Tasking
  Status      = running

```

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

Loaded App: xpc_osc3	Scope: 1, lower y-axis limit set to 0.000000
Memory: 123MB	Scope: 1, upper y-axis limit set to 0.000000
Mode: RT, single	System: initializing application finished
Logging: t x y tet	System: execution started (sample time: 0.000250)
StopTime: 10000 d	System: execution stopped at 10.191750
SampleTime: 0.00025	Scope: 1, set to state 'Interrupted'
AverageTET: 6.859e-006	minimal TET: 0.000006 at time 0.006250
Execution: 10.00 s	maximal TET: 0.000017 at time 0.097750
	System: execution started (sample time: 0.000250)

2 In the MATLAB window, type

```
-tg or tg.stop or stop(tg)
```

The target application stops running.

The xPC Target software allows you to change many properties and parameters without rebuilding your target application. Two of these properties are StopTime and SampleTime.

3 Change the stop time. For example, to change the stop time to 1000 seconds, type either

```
tg.StopTime = 1000 or set(tg,'StopTime',1000)
```

4 Change the sample time. For example, to change the sample time to 0.01 seconds, type either

```
tg.SampleTime = 0.01 or set(tg, 'SampleTime', 0.01)
```

Although you can change the sample time between different runs, you can only change the sample time without rebuilding the target application under certain circumstances.

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 the **Fixed step size** in the **Solver** node to a larger value and rebuild the model. (See “User Interaction” on page 1-28 for further limitations on changing sample times.)

Control with Simulink External Mode

Control of your xPC Target application with Simulink is limited to connecting a Simulink model to a target application through external mode, and starting the target application. Using Simulink external mode is one method to tune parameters. In Simulink external mode, the model can only connect to the default target PC.

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

After you create and download a target application to the target PC, you can run the target application. This procedure uses the Simulink model `my_xpc_osc2.mdl` as an example (see “Building and Downloading the Target Application” on page 3-52). It assumes that you have specified the correct target PC environment on the xPC Target options node of the Simulink Coder parameters dialog. In particular, you must specify the target PC to which you want to connect. See the **Use the default target PC** check box description in “xPC Target Options Node” on page 3-56.

1 In the Simulink window, and from the **Simulation** menu, click **External**.

A check mark appears next to the menu item **External**, and Simulink external mode is activated. Simulink external mode connects your Simulink model to your target application as a simple graphical user interface.

2 In the Simulink window, and from the **Simulation** menu, click **Connect to target**.

All the current Simulink model parameters are downloaded to your target application. This downloading guarantees the consistency of the parameters between the host model and the target application.

3 From the **Simulation** menu, click **Start real-time code**.

The target application begins running.

4 In the MATLAB window, type

```
tg.stop or -tg
```

You cannot stop the target application from the Simulink window by clicking **Stop real-time code** from the **Simulation** menu.

Note Opening a dialog box for a source block causes Simulink to pause. While Simulink is paused, you can edit the parameter values. You must close the dialog box to have the changes take effect and allow Simulink to continue.

Parallel Model Reference Builds Using Remote Workers

The xPC Target software supports the ability to build models that contain referenced models in parallel when possible. This capability enables you to more quickly build and download xPC Target applications to the target PC.

You can build models using a local PC, or using PCs running Windows systems (remote workers). Depending on how you want to build, the following products must be installed:

Build on...	Requires...
Local host PC	Parallel Computing Toolbox™
Remote workers running Windows	<ul style="list-style-type: none"> Parallel Computing Toolbox MATLAB® Distributed Computing Server™

See “Reducing Build Time for Referenced Models” in the *Simulink Coder User’s Guide* for a description of how to build referenced models in parallel with a local host or a pool of remote workers.

Additionally, if you want to build models using a pool of remote workers:

- 1 Ensure that each remote worker has the same compiler configuration. For example, all remote PCs must have the same compiler version and installation path as the host PC..
- 2 Ensure that each remote worker has the xPC Target software installed.
- 3 After you call the `matlabpool` command to start the pool of remote workers, in the MATLAB Command Window of the host PC, call the `pctRunOnAll` command. This command configures the compiler for all the remote workers. For example type a sequence like the following all on one line:

```
pctRunOnAll('setxpcenv(''CCompiler'', 'VisualC'', 'CompilerPath'',
'C:\Program Files\Microsoft Visual Studio 9.0'))
```

- 4 Build and download your model.

Menu Bar and Toolbar Contents and Shortcut Keys






The procedures in the xPC Target documentation for xPC Target Explorer use right-click operations. You can also perform xPC Target Explorer operations through the menu bar and toolbar. This section is a reference for the menu bar and toolbar contents.



The xPC Target Explorer menu bar has the following menus:

- **File** — General file operation options, including
 - **Add Target** — Add a target PC to the xPC Target system.
 - **Remove Target** — Remove a target PC from the xPC Target system.
 - **Change Host PC Current Directory** — Change folder to one that contains the prebuilt target application (DLM) you want to download to your target PCs.
 - **Close** — Close xPC Target Explorer.
- **Target** — General target PC operations, including
 - **Ping Target** — Test the communication between the host PC and target PCs.
 - **Connect to Target** — Connect xPC Target Explorer to selected target PC.
 - **Set As Default** — Designate the selected target PC as the default one.
 - **Import Environment** — Import an existing target PC configuration from the MATLAB workspace.
 - **Export Environment** — Save the target PC environment structure to a MAT-file.
 - **Save Session** — Save target PC application sessions to xPC Target Explorer.
 - **Load Session** — Load target PC application sessions to xPC Target Explorer.
- **Application** — General target application operations, including
 - **Start Application** — Start the selected target application.

- **Stop Application** — Stop the selected target application.
- **Add a scope** — Add a host scope, Target, or File.
- **Delete scope** — Delete a scope.
- **View host scopes** — Display a viewer on the host PC for host scopes.
- **Tools** — General operations, including
 - **Enable/Disable Refresh** — Enable/disable the window refresh.
 - **Change Refresh Rate** — Change the refresh rate for the display of signals in xPC Target Explorer. In the Refresh Rate dialog box, enter the desired refresh rate in seconds. Note that setting the refresh rate to less than 0.2 seconds (default) might cause target PC CPU overloads or degrade MATLAB performance.
 - **Go to Simulink Model** — For the selected model, display the Simulink model.
- **Help** — General product help information, including
 - **Using xPC Target** -- Invoke help for xPC Target product.
 - **xPC Target Explorer Help** — Invoke help for xPC Target Explorer.
 - **About xPC Target** — Invoke general xPC Target information.

The xPC Target Explorer toolbar has buttons for some of the more commonly used operations available from the menu bar. These buttons include

Button	Description
	Add Target button
	Delete Target button
	Connect to Target button
	Start Application button
	Stop Application button

Button	Description
	Scope Viewer button
	Go to Simulink Model button

The xPC Target Explorer has the following keyboard shortcuts:

Action	Shortcut
Add target	Ctrl+A
Remove target	Ctrl+R
Close	Ctrl+W
Stop/start target	Ctrl+T
Ping target	Ctrl+P
Delete scope	Select scope and click Delete

application

See *target application*.

build process

Process of generating a target application from your Simulink model, compiling, linking, and downloading the generated code to create a *target application*.

execution

Running the *target application* on the target PC in real time.

executable code

See target application.

kernel

Real-time software component running on the target PC that manages the downloaded *target application*.

model

Simulink and/or Stateflow model.

parameter tuning

Process of changing block parameters and downloading the new values to a *target application* while it is running or not running.

sample rate

Rate the *target application* is stepped in samples/second. Reciprocal of the *sample time*.

sample time

Interval, in seconds, between the execution of *target 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 simulation of the Simulink and Stateflow model on the host PC in nonreal time.

target application

Executable code generated from a Simulink and Stateflow model, which can be executed by the xPC Target kernel on the target PC.

A

- advantages of network communication 1-17
- analog input (A/D)
 - driver support 1-18
- analog output (D/A)
 - driver support 1-18
- API
 - custom GUI 1-35
- API for Microsoft .NET framework 1-34

B

- before you boot
 - checklist 2-44
- BIOS
 - target PC 1-6
- BIOS settings 2-13
- block parameters
 - scope 3-22
- boot options 2-43
 - boot floppy disk 2-51
 - CD 2-45
 - dedicated network 2-54
- booting
 - target PC 3-45
 - troubleshooting 3-47
- build process
 - target application 3-52
 - troubleshooting 3-54

C

- C compiler
 - network setup 2-29
 - required product 2-4
 - serial setup 2-38
- CAN field bus
 - driver support 1-18
- CD
 - creating for booting 2-45

- CD target boot disk
 - creating 2-48
- code generation options
 - for Simulink Coder 3-47
 - reference 3-56
- COM API 1-35
- command-line interface
 - MATLAB 1-30
 - target PC 1-33
- communication
 - between computers 1-21
 - network 2-26
 - network advantages 1-17
 - serial 2-36
- compiler
 - required 2-4
- computer
 - communication 1-21
 - desktop PC for host 1-14
 - desktop PC for target 1-15
 - host PC 1-14
 - industrial PC 1-15
 - notebook PC 1-14
 - PC/104 and PC/104+ 1-15
 - target PC 1-15
- connections
 - computers 1-16
 - I/O boards 1-18
 - real-world 1-18
- controlling target applications
 - with MATLAB 3-67
 - with Simulink external mode 3-69
- counter timers
 - driver support 1-18
- creating boot media
 - checklist 2-44
- creating CD target boot disks 2-48
- creating target applications 3-45
- creating target boot disks 2-53
- creating target objects 3-52

- custom GUI 1-35
 - API 1-35
 - API for Microsoft .NET framework 1-34
 - COM API 1-35

D

- dedicated network
 - booting within 2-54
- default target PC
 - introduction 2-24
- defining scope block parameters 3-22
- desktop PC
 - host computer 1-14
 - target computer 1-15
- directories
 - installed 2-16
 - working 2-16
 - xpc 2-16
 - xpcdemo 2-16
- DOS loader mode
 - embedded option 1-26
- downloading target application 3-52

E

- embedded option
 - DOS loader mode 1-26
 - stand-alone mode 1-26
- encoder
 - I/O driver support 1-18
- entering simulation parameters 3-47
- environment
 - network communication 2-29
 - serial communication 2-37
- Ethernet card
 - ISA bus 2-27
 - PC/104 bus 2-26
 - PCI bus 2-27
 - SBS bus 2-26

- Ethernet chip sets
 - supported 2-26
- exporting and importing
 - xPC Target Explorer 2-79
- external mode
 - controlling target application 3-69
 - user interaction 1-32

F

- features of xPC Target
 - fixed-point support 1-12
 - MATLAB Compiler support 1-12
 - parameter tuning 1-10
 - real-time application 1-9
 - real-time kernel 1-6
 - signal acquisition 1-9
- files
 - data acquisition 3-33
 - host PC 2-16
 - installed 2-16
 - project folder 2-16
 - scopes 3-33
 - working folder 2-16
 - xpc folder 2-16
 - xpcdemos folder 2-16
- fixed-point support 1-12
- floppy disk
 - creating for booting 2-51

G

- GPIO field bus
 - driver support 1-18
- graphical user interface (GUI)
 - custom with API 1-35
 - custom with API for Microsoft .NET framework 1-34
 - custom with COM API 1-35

H

- hardware environment
 - requirements for target PC 2-9
- hardware-in-the-loop process 1-25
- host computer
 - see host PC 1-14
- host PC 2-15
 - communication 1-21
 - configuring 2-19
 - connections 1-16
 - downloading software 2-15
 - files 2-16
 - hardware 2-36
 - license file 2-15
 - requirements 2-8

I

- I/O boards
 - supported by xPC Target 1-18
- I/O driver support
 - analog input (A/D) 1-18
 - analog output (D/A) 1-18
 - CAN field bus 1-18
 - counter timers 1-18
 - digital 1-18
 - encoder 1-18
 - GPIO 1-18
 - RS-232 1-18
 - RS-422 1-18
 - RS-485 1-18
 - shared memory 1-18
 - UDP 1-18
- industrial PC 1-15
- initial working folder 2-17
- installation prerequisite
 - obtaining a valid license 2-16
- installing
 - Ethernet card for ISA 2-27
 - Ethernet card for PCI 2-27

- hardware 2-36
 - network communication 2-26
 - on the host PC 2-15
 - serial communication 2-36
 - testing 2-71

ISA bus

- Ethernet card 2-27

K

- kernel
 - target boot disk 1-6
 - target PC BIOS 1-6
 - xPC Target 1-6

L

- license
 - obtaining 2-16

M

- MathWorks
 - technical support 2-78
 - valid license 2-16
- MATLAB
 - controlling target application 3-67
 - required product 2-2
- MATLAB Compiler support 1-12
- memory model
 - target application 1-9

N

- network boot 2-54
- network communication
 - advantages 1-17
 - environment 2-29
 - hardware 2-26
 - host PC 2-26
 - installing and setting up 2-26

- specifying environment properties 2-29
- target PC 2-26
- notebook PC 1-14

O

- outport block
 - adding to Simulink model 3-10
 - simulation parameters 3-13
- overview
 - MATLAB command-line interface 1-30

P

- parameter tuning
 - interactive 1-10
 - scripts 1-10
- parameters
 - scope blocks 3-22
- PC/104 bus
 - Ethernet card 2-26
- PCI bus
 - Ethernet card 2-27

Q

- Quatech serial drivers 1-18

R

- rapid prototyping process 1-22
- real-time application
 - memory model 1-9
 - task execution time 1-9
- real-time kernel 1-6
- required products
 - C language compiler 2-4
 - MATLAB 2-2
 - Simulink 2-3
 - Simulink Coder 2-4
 - xPC Target 2-2

- requirements
 - host PC 2-8
 - system 2-8
 - target PC 2-9

RS-232

- Quatech 1-18

RS-422

- Quatech 1-18

RS-485

- Quatech 1-18

running simulation

- host PC 3-39

S

SBS bus

- Ethernet card 2-26

scope blocks

- parameters 3-22
- xPC Target 1-33

scopes

- file 3-33
- host 3-29
- target 3-23

serial communication

- environment 2-37
- hardware 2-36
- installing and setting up 2-36
- section overview 2-36

setting

- network communication 2-29
- serial communication 2-37

setting initial working folder 2-17

setup (network) dialog box

- C compiler 2-29

setup (serial) dialog box

- C compiler 2-38

shared memory driver support 1-18

signal acquisition

- logging 1-9

- monitoring 1-9
- tracing 1-9
- simulation
 - from MATLAB 3-41
 - Simulink tutorial model 3-39
 - with Simulink 3-39
- simulation parameters
 - entering 3-13
 - for Simulink Coder 3-47
 - xPC Target Scope block 3-22
- Simulink Coder
 - code generation options 3-56
- Simulink external mode
 - controlling target application 3-69
- Simulink model
 - basic tutorial 3-2
 - outport block 3-10
 - xPC Target Scope blocks 3-17
- software environment
 - requirements on host PC 2-8
 - requirements on target PC 2-9
- software installation 2-15
- stand-alone mode
 - embedded option 1-26
- starting and stopping
 - target application 3-67
- system requirements
 - host PC 2-8
 - target PC 1-15

T

- target application 1-9
 - building 3-52
 - control with external mode 3-69
 - creating 3-45
 - downloading 3-52
 - memory model 1-9
 - running 3-57
 - starting 3-67

- stopping 3-67
- target PC 3-57
- task execution time 1-9
- target boot disk
 - creating 2-53
 - kernel 1-6
 - with desktop PC 1-15
 - with industrial PC 1-15
- target PC
 - boot disk 1-6
 - booting 3-45
 - command-line interface 1-33
 - communication 1-21
 - compatible target computers 2-13
 - connecting 1-16
 - creating boot disk 2-53
 - creating CD boot disk 2-48
 - creating CD bootable ROM 2-48
 - hardware 2-36
 - hardware requirements 2-9
 - I/O boards 2-13
 - real-time kernel 1-7
 - running target application 3-57
 - software installation 2-71
 - software requirements 2-9
 - troubleshooting installation 2-71
- task execution time (TET) 1-9
 - logging 3-47
- testing installation 2-71
- timeout value
 - changing 3-55
- troubleshooting
 - boot process 3-47
 - build process 3-54
 - timeout value 3-55
- tutorial
 - basic 3-1
 - creating a Simulink model 3-2
 - creating a target application 3-45
 - running target application 3-57

simulating a Simulink model 3-39

U

UDP driver support 1-18

user interaction

 MATLAB command-line interface 1-30

 Simulink external mode interface 1-32

 target PC command line 1-33

 Web browser 1-34

 with API 1-35

 with API for Microsoft .NET framework 1-34

 with COM API 1-35

 xPC Target Scope blocks 1-33

V

valid license

 obtaining 2-16

W

Web browser

 user interaction 1-34

working folder

 initial 2-17

 setting initial 2-17

X

xPC Target

 features 1-6

 interaction 1-28

 introduction 1-1

 kernel 1-6

 required products 2-2

 supported I/O boards 1-18

xPC Target Explorer

 exporting and importing 2-79

 introduction 2-22

 menu bar 3-72

 shortcut keys 3-72

 toolbar 3-72

xPC Target Scope blocks

 adding to Simulink model 3-17

 interface 1-33