

# xPC Target™

## Getting Started Guide

**R2012a**

**MATLAB®**  
& **SIMULINK®**

## How to Contact MathWorks



[www.mathworks.com](http://www.mathworks.com) Web  
[comp.soft-sys.matlab](mailto:comp.soft-sys.matlab) Newsgroup  
[www.mathworks.com/contact\\_TS.html](http://www.mathworks.com/contact_TS.html) Technical Support



[suggest@mathworks.com](mailto:suggest@mathworks.com) Product enhancement suggestions  
[bugs@mathworks.com](mailto:bugs@mathworks.com) Bug reports  
[doc@mathworks.com](mailto:doc@mathworks.com) Documentation error reports  
[service@mathworks.com](mailto:service@mathworks.com) Order status, license renewals, passcodes  
[info@mathworks.com](mailto: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 2000–2012 by The MathWorks, Inc.

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

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

### Trademarks

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

### Patents

MathWorks products are protected by one or more U.S. patents. Please see [www.mathworks.com/patents](http://www.mathworks.com/patents) for more information.

## Revision History

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



# 1

## Introduction

<b>Product Description</b> .....	1-2
Key Features .....	1-2
<b>Using the Product</b> .....	1-3
<b>Required Knowledge</b> .....	1-5
<b>Product Features</b> .....	1-6
Real-Time Kernel .....	1-6
Real-Time Application .....	1-9
Signal Acquisition .....	1-9
Parameter Tuning .....	1-10
xPC Target Embedded Option .....	1-11
Fixed-Point Support .....	1-12
MATLAB® Compiler™ Support .....	1-12
BLAS Library Support .....	1-12
<b>Hardware Description</b> .....	1-14
Introduction .....	1-14
Host Computer .....	1-14
Target Computer .....	1-15
Host-Target Connection .....	1-17
I/O Driver Support .....	1-19
<b>Real-Time Test Environment</b> .....	1-22
Software Setup .....	1-22
Host-Target Communication .....	1-23
xPC Target Embedded Option .....	1-24
<b>User Interaction</b> .....	1-26
Introduction .....	1-26
xPC Target Explorer .....	1-27
MATLAB Command-Line Interface .....	1-29

Simulink External Mode Interface .....	1-31
Simulink with xPC Target Blocks .....	1-32
Target Computer Command-Line Interface .....	1-32
Web Browser Interface .....	1-33
Custom GUI with xPC Target API for Microsoft .NET Framework .....	1-33
Custom GUI with xPC Target API .....	1-34
Custom GUI with xPC Target COM API .....	1-34

## Installation and Configuration

# 2

<b>Host Computer Configuration</b> .....	2-2
Host Computer Hardware .....	2-2
Host Computer Software .....	2-3
 <b>Target Computer Configuration</b> .....	2-5
Target Computer Hardware .....	2-5
Target Computer Software and BIOS .....	2-9
 <b>Host Software Installation</b> .....	2-12
Overview .....	2-12
xPC Target Turnkey System .....	2-12
License Requirements .....	2-13
Files on the Host Computer .....	2-13
Setting Your Initial Working Folder .....	2-14
Running MATLAB Remotely .....	2-15
Configuring the Host Computer for Your C Compiler ....	2-16
Installing the Microsoft .NET Framework .....	2-19
 <b>Network Communication</b> .....	2-20
Network Communication Overview .....	2-20
Hardware for Network Communication .....	2-20
Ethernet Card for a PCI Bus .....	2-21
Ethernet Card for an ISA Bus .....	2-22
Environment Properties for Network Communication ....	2-24
 <b>Serial Communication</b> .....	2-31
Serial Communication Overview .....	2-31

Hardware for Serial Communication .....	2-31
Environment Properties for Serial Communication .....	2-32
<b>Target Boot Method .....</b>	<b>2-38</b>
Before You Boot .....	2-38
Bootling Target Computers from CD or DVD .....	2-39
Bootling Target Computers Within a Dedicated Network ..	2-46
Bootling Target Computers from Removable Boot Drives ..	2-51
DOS Loader Boot Method .....	2-57
Embedded Target Boot Method .....	2-62
<b>Running the Confidence Test .....</b>	<b>2-72</b>

## Basic Workflows

### 3

<b>Rapid Prototyping .....</b>	<b>3-2</b>
<b>Hardware in the Loop .....</b>	<b>3-7</b>

## Tutorial and Examples

### 4

<b>Creating a Simple Simulink Model .....</b>	<b>4-2</b>
<b>Entering Parameters for the Scope Block .....</b>	<b>4-7</b>
<b>Adding a Simulink Outport Block .....</b>	<b>4-11</b>
<b>Entering Parameters for the Outport Block .....</b>	<b>4-15</b>
<b>Adding an xPC Target Scope Block .....</b>	<b>4-20</b>

<b>Entering Parameters for an xPC Target Scope Block</b> ..	<b>4-25</b>
Entering Parameters for a Target Scope .....	<b>4-25</b>
Entering Parameters for a Host Scope .....	<b>4-31</b>
Entering Parameters for a File Scope .....	<b>4-35</b>
<b>Simulating in Non-Real Time Using Simulink</b> .....	<b>4-41</b>
<b>Simulating in Non-Real Time Using MATLAB</b>	
<b>Language</b> .....	<b>4-43</b>
<b>Booting Target Hardware</b> .....	<b>4-46</b>
<b>Entering Simulation Parameters</b> .....	<b>4-48</b>
<b>Building and Downloading Target Application</b> .....	<b>4-54</b>
<b>Executing in Real Time Using xPC Target Explorer</b> ...	<b>4-56</b>
Downloading and Running Target Applications on a Target PC .....	<b>4-60</b>
Manipulating Target Application Properties .....	<b>4-65</b>
<b>Executing in Real Time Using Simulink External Mode</b> .....	<b>4-67</b>
<b>Executing in Real Time Using MATLAB Commands</b> ...	<b>4-69</b>
<b>Selected Examples</b> .....	<b>4-71</b>
Applications and Driver Examples .....	<b>4-71</b>
To Locate or Edit an Example Script .....	<b>4-73</b>



**Glossary**

---

**Index**

---



# Introduction

---

- “Product Description” on page 1-2
- “Using the Product” on page 1-3
- “Required Knowledge” on page 1-5
- “Product Features” on page 1-6
- “Hardware Description” on page 1-14
- “Real-Time Test Environment” on page 1-22
- “User Interaction” on page 1-26

## Product Description

### **Perform hardware-in-the-loop simulation and real-time rapid prototyping**

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 I/O 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 HDL Coder™.

### **Key Features**

- Real-time execution of Simulink and Stateflow models on a target computer via an optimized real-time kernel
- Support for target computer hardware, including PMC, PCI, PCIe, cPCI, and PC104 form factors
- Blocks supporting numerous I/O modules, including analog I/O, digital I/O, pulse train generation and capture, and shared memory
- Blocks supporting communication protocols and data buses, including serial, UDP/IP, CAN, J1939, ARINC 429, and MIL-STD-1553
- Ability to program FPGA boards for xPC Target Turnkey systems (with HDL Coder)
- Tools for real-time monitoring, parameter tuning, and data logging
- Standalone operation with xPC Target Embedded Option™
- APIs for developing user interfaces (Visual Basic®, C/C++, Java™, and .NET)

## Using the Product

The xPC Target environment uses a target computer, separate from a host computer, for running real-time applications. In this environment you use your desktop computer as a host computer 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 computer to the target computer 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 computer, target computer, and, for I/O, the target computer must also have I/O boards supported by the xPC Target product. The target computer 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.

---

**Note** WATCOM compiler support will be removed in a future release. Use a Microsoft compiler instead.

---

- 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 standalone operation. This option allows you to
  - Create standalone applications for the target computer, which can boot, run, and operate independent from the host computer.
  - Deploy standalone GUI applications running on the host computer 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, standalone GUI applications running on a host computer 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.

## Required Knowledge

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 4, “Tutorial and Examples”.

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*: “Target Application Environment” and “Targets and Scopes in the MATLAB Interface” for more detailed information about the commands in the xPC Target software.

## Product 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
“xPC Target Embedded Option” on page 1-11
“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 computer. Instead, you boot the target computer 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 Target Boot Method” on page 2-62 in the xPC Target™ Getting Started Guide on page 1.

### Target Boot Options

You boot and run the target computer 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 computer. This arrangement allows you to use the target computer for testing real-time applications. When you are finished with your tests, you can use the target computer again as a desktop computer. Software is not permanently installed on the target computer unless you deliberately install a standalone application on the hard disk or flash memory.

- Removable boot devices



CD, DVD, USB and SD (compact) flash drive, removable hard drive, 3.5-inch floppy disk

- Fixed boot devices

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

- Network boot

Dedicated network

## Target Computer BIOS

At the beginning of the target computer 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 computer 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 computer 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 computer. 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 computer 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  $\mu$ s (50 kHz). Model size, complexity, and target computer hardware affect maximum speed (minimal sample time) of execution.

For more information on creating a target application, see “Creating a Simple Simulink Model” on page 4-2.

## Signal Acquisition

The xPC Target real-time kernel stores signal data from the target application in RAM on the target computer. Alternatively, you can have the xPC Target real-time kernel store signal data in a file on the target computer. 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 computer, 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 computer makes a request to upload data from the target computer. You can then visualize signals by plotting data on the host computer, 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 computer or to upload the signal data and visualize it on the host computer 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 1-26 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.
- **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 computer, you can optimize the value of that parameter.

For information on tuning parameters with the xPC Target software, see “User Interaction” on page 1-26 and “Parameter Tuning and Inlining Parameters” in the *xPC Target User’s Guide*.

## **xPC Target Embedded Option**

### **Run xPC Target applications on standalone target computers**

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.

### **Key Features**

- Standalone operation with xPC Target-compatible systems, including xPC Target Turnkey systems, PC-compatible hardware, and single-board computers (SBCs)
- Automatic execution of the embedded application upon target computer startup
- Royalty-free deployment of applications generated by xPC Target

## **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, standalone applications that include xPC Target functionality.

Standalone 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, 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.

If you set up your model to xPC Target Embedded Coder (xpctargetert.tlc), you can create a custom Code Replacement Library (CRL) based upon the xPC Target BLAS (XPC\_BLAS). For more on CRLs, see:

- Code Replacement Library (CRL) and Embedded Targets
- “Code Replacement”.

---

**Note**

- Your model accesses the XPC\_BLAS 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 XPC\_BLAS library.
  - WATCOM compiler support will be removed in a future release. Use a Microsoft compiler instead.
-

## Hardware Description

In this section...
“Introduction” on page 1-14
“Host Computer” on page 1-14
“Target Computer” on page 1-15
“Host-Target Connection” on page 1-17
“I/O Driver Support” on page 1-19

### 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 Computer

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

- CD or DVD drive
- Dedicated network access
- Removable drive, such as USB and SD (compact) flash drive, removable hard drive, or 3.5-inch floppy disk

For more details on the requirements of the host computer, see “Host Computer Configuration” on page 2-2.



## Target Computer

The xPC Target software supports concurrent access to up to 64 target computers with one host. A target computer can connect to only one host computer at any given time. A target computer cannot connect to multiple host computers. A target computer can be almost any PC compatible system with a 32-bit Intel® or AMD® processor (386 compatible or higher). It must also support a free serial port or an Ethernet adapter. In addition, the target computer must contain a removable 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 removable drive or CD to a hard disk or flash memory.

A target computer 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 computer from the target boot drive or network boot image, the xPC Target software uses the resources on the target computer (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 computer, 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 computer, 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 computer must be a fully PC-compatible system and support a serial port or an Ethernet adapter compatible with the xPC Target software.

---

**Note** Do not use a laptop PC as a target computer. For target computer hardware, consider the xPC Target Turnkey solutions.

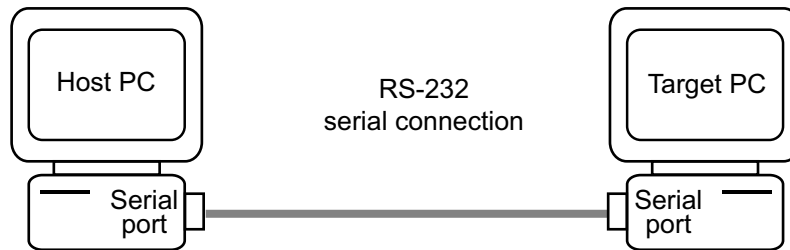
---

For more details on the requirements of the target computer, see “Target Computer Configuration” on page 2-5.

## Host-Target Connection

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



For detailed information on setting up the hardware and software for serial communication, see “Serial Communication” on page 2-31.

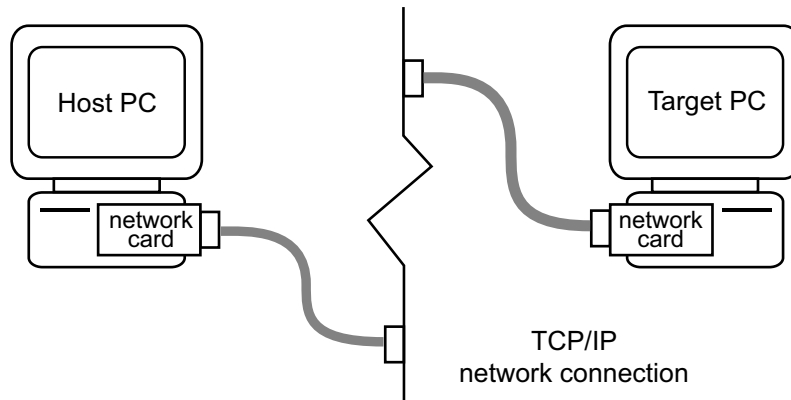
---

**Note** RS-232 Host-Target communication mode will be removed in a future release. Use TCP/IP instead.

---

**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 computer 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 2-20.



For detailed information on setting up the hardware and software for network communication, see “Network Communication” on page 2-20.

### **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. To install and configure your network cards and 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 all required 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 computer 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 “ACCES I/O” and “Versalogic” 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*.

## Real-Time Test Environment

In this section...
“Software Setup” on page 1-22
“Host-Target Communication” on page 1-23
“xPC Target Embedded Option” on page 1-24

### Software Setup

The real-time test 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.

You create a nonreal-time test environment by creating an initial model in regular Simulink.

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

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

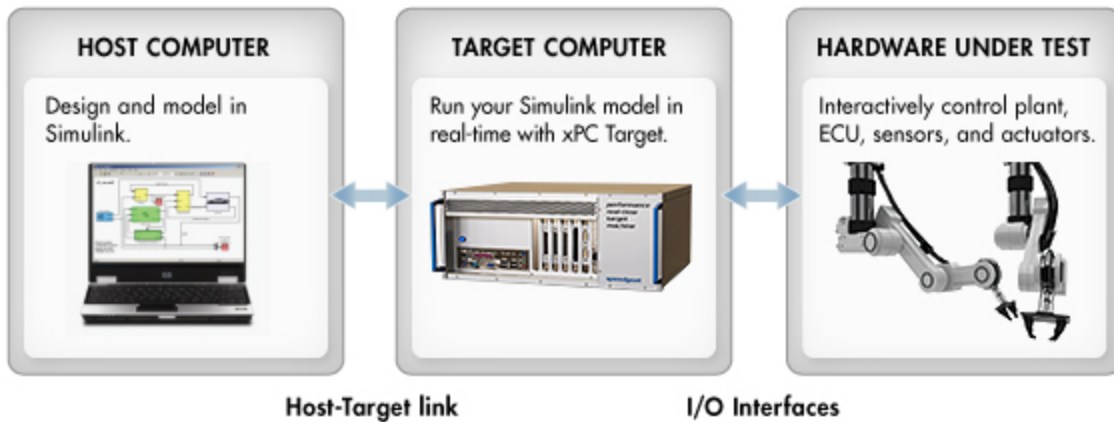
And connect the host computer to the target computer via a single TCP/IP or RS-232 connection. You then:

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

Once you make the connections, you can:



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



## Host-Target Communication

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

---

**Note** RS-232 Host-Target communication mode will be removed in a future release. Use TCP/IP instead.

---

## **xPC Target Embedded Option**

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 computer 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 computer 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 computer 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 standalone operation.

---

For more information on the xPC Target Embedded Option product, see “Embedded Target Boot Method” on page 2-62 in the xPC Target™ Getting Started Guide on page 1.

## **StandAlone Mode**

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

- 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 computer hard drive or flash memory.
- 4** Boot the target computer.

When you boot the target computer, the target computer 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 Target Boot Method” on page 2-62 in the xPC Target™ Getting Started Guide on page 1.

## User Interaction

### In this section...

“Introduction” on page 1-26

“xPC Target Explorer” on page 1-27

“MATLAB Command-Line Interface” on page 1-29

“Simulink External Mode Interface” on page 1-31

“Simulink with xPC Target Blocks ” on page 1-32

“Target Computer Command-Line Interface” on page 1-32

“Web Browser Interface” on page 1-33

“Custom GUI with xPC Target API for Microsoft .NET Framework” on page 1-33

“Custom GUI with xPC Target API” on page 1-34

“Custom GUI with xPC Target COM API” on page 1-34

## Introduction

The xPC Target environment has a modifiable interface to the target computer. You can use this interface from MATLAB or Simulink, and you can use other development environments to create standalone 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 computers.

---

**Note** Some blocks (see “Blocks Whose Outputs Depend on Inherited Sample Time” in the *Simulink User’s Guide*) cannot 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 unexpected results.

---

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

<b>Interface</b>	<b>Environment Properties</b>	<b>Control</b>	<b>Signal Acquisition</b>	<b>Parameter Tuning</b>
“xPC Target Explorer” on page 1-27	X	X	X	X
“MATLAB Command-Line Interface” on page 1-29	X	X	X	X
“Simulink External Mode Interface” on page 1-31		X	X	X
“Simulink with xPC Target Blocks ” on page 1-32			X	
“Target Computer Command-Line Interface” on page 1-32		X	X	X
“Web Browser Interface” on page 1-33		X	X	X
“Custom GUI with xPC Target API for Microsoft .NET Framework” on page 1-33		X	X	X
“Custom GUI with xPC Target API” on page 1-34		X	X	X
“Custom GUI with xPC Target COM API” on page 1-34		X	X	X

## **xPC Target Explorer**

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

---

**Note** xPC Target Explorer runs only on 32-bit host computers. On 64-bit computers, use the MATLAB command-line interface. For further information, see “Configuring Environment From the MATLAB Command Line”.

---

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 2-31 and “Network Communication” on page 2-20
  - “Target Environment Command-Line Interface” 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 computer, 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 computer. 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 Target Boot Drive with a Command-Line Interface” on page 2-54, and “Target Application Environment” in the *xPC Target User’s Guide*.

- **Control** — Reboot the target computer, 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 “Executing in Real Time Using MATLAB Commands” on page 4-69 in xPC Target™ Getting Started Guide on page 1 and “Target Application Environment” 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 computer to the host computer between runs. For standalone target computers, 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 computer and replaces the existing parameter while the target application continues to run. For more information, see “Parameter Tuning with Simulink External Mode”.

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

## Simulink with xPC Target Blocks

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

Signal acquisition — Add scopes to the target computer by adding 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 Scope blocks, see “Adding an xPC Target Scope Block” on page 4-20, “Entering Parameters for an xPC Target Scope Block” on page 4-25 , “Entering Parameters for a File Scope” on page 4-35, and “Signal Tracing”.
- For information on using xPC Target To and From blocks to transfer data to and from a Simulink user interface model, see “Execution Using Graphical User Interface Models”.

## Target Computer Command-Line Interface

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

The target computer 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 “Execution Using the Target Computer Command Line” in the *xPC Target User’s Guide*.

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

## Web Browser Interface

If the target computer 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 computer is connected to the host computer with an RS-232 cable, and is using the TCP/IP to RS-232 gateway, you can use a Web browser on the host computer.

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 “Execution Using the 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 computer 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 Reference for Microsoft .NET Framework”.

## **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 computer and communicates with the target application on the target computer using RS-232 or TCP/IP communication. A GUI application can be a console or Windows application using ActiveX<sup>®</sup> 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

---

- “Host Computer Configuration” on page 2-2
- “Target Computer Configuration” on page 2-5
- “Host Software Installation” on page 2-12
- “Network Communication” on page 2-20
- “Serial Communication” on page 2-31
- “Target Boot Method” on page 2-38
- “Running the Confidence Test” on page 2-72

## Host Computer Configuration

The xPC Target product requires two separate computers, a host computer and target computer.

---

### Note

- For overall system requirements, see <http://www.mathworks.com/products/xpctarget/requirements.html> on the xPC Target product page (<http://www.mathworks.com/products/xpctarget>).
  - For material on related products, see <http://www.mathworks.com/products/xpctarget/related.html>.
- 

In the xPC Target software environment, the host computer is usually your desktop computer. Here you install the xPC Target and xPC Target Embedded Option products, along with MATLAB, Simulink, and other required and optional software (see “Host Computer Software” on page 2-3). A notebook computer is also a viable host computer.

## Host Computer Hardware

At a minimum, the xPC Target product requires the following host computer hardware:

Hardware	Description
CPU	Intel Pentium, AMD Athlon™, or later
RAM	128 MB or more

## Peripherals

To install and run the xPC Target product, the host computer requires one **hard disk drive** with 60 MB of free space.

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

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

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

- **SD (Compact) flash drive**
- **Removable hard drive**
- **3.5-inch floppy disk drive**

## Communication

Select one of the following methods for the host computer to communicate with the target computer:

- One free **Ethernet adapter** (PCI, ISA, or USB) connected to a network (see “Network Communication” on page 2-20 for details)
- One free **serial port** (COM1 or COM2) with a 9-pin or 25-pin D-sub connector (see “Serial Communication” on page 2-31 for details)

## Host Computer Software

At a minimum, the xPC Target product requires the following software configuration on your host computer:

- 32-bit or 64-bit Windows operating system (see [http://www.mathworks.com/support/sysreq/current\\_release/](http://www.mathworks.com/support/sysreq/current_release/))
- MATLAB Version 7.14
- Simulink Version 7.9
- Simulink Coder Version 8.2
- MATLAB Coder 2.2
- Microsoft .NET Framework 4.0

---

**Tip** If Microsoft .NET Framework 4.0 is not already installed on your machine, see “Installing the Microsoft .NET Framework” on page 2-19.

---

- C language compiler (see [http://www.mathworks.com/support/-compilers/current\\_release/](http://www.mathworks.com/support/-compilers/current_release/))
- xPC Target Version 5.2

For more on installing the xPC Target software, see “Host Software Installation” on page 2-12.



## Target Computer Configuration

The target computer 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 basic system requirements, see <http://www.mathworks.com/products/xpctarget/requirements.html> on the xPC Target product page (<http://www.mathworks.com/products/xpctarget>).
  - To support xPC Target, 64-bit target computers must run in 32-bit mode.
- 

### Target Computer Hardware

At a minimum, the xPC Target product requires the target computer hardware:

Hardware	Description
CPU	Intel 386/486/Pentium or AMD K5/K6/Athlon with or without a floating-point coprocessor
Chip set	PC compatible with UART, programmable interrupt controller, keyboard controller, and counter
RAM	The xPC Target requires 8 MB or more of dynamic RAM

---

**Note**

- Do not use a laptop PC as a target computer. For target computer hardware, consider the xPC Target Turnkey solutions.
  - The xPC Target kernel can use only 2 GB of memory. You can acquire several megabytes of data during a run, depending on how much memory you install in the target computer.
-

## Multicore CPU Support

The xPC Target software can run on any target computer hardware equipped with a 32-bit or 64-bit x86 compatible CPU (386 or higher). It can allocate and manage up to 32 CPUs, including multi-core processors, multi-CPU, and simultaneous multithreads (Intel Hyper-Threads). For example, xPC Target multicore is supported on the following platforms:

- Intel Core™ 2 Duo processor
- Intel Core 2 Quad processor
- Intel Core i5 processor
- Intel Core i7 processor

You can check for multicore CPU support in the target computer BIOS.

## Peripherals

For booting the xPC Target kernel, the target computer requires one or more of:

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

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

- **SD (Compact) flash drive**
- **Removable hard drive**
- **3.5-inch floppy disk drive**
- **PXE-compatible Ethernet adapter**

A **hard drive** is not required unless you want to access the target computer file system (for file scopes).

- You can copy files to a hard drive or flash memory and boot from that device.

- If you want to access the target computer file system on a hard drive, see “Logging Signal Data with FTP and File System Objects” in the *xPC Target User’s Guide*.
- The hard drive must be a parallel ATA (PATA)/Integrated Device Electronics (IDE) or serial ATA (SATA) drive. For better performance, configure this drive as a primary master.
- Verify the hard drive is not cable-selected.
- The xPC Target product supports file systems of type FAT-12, FAT-16, or FAT-32.

A **keyboard** is required to control the target computer when you create standalone applications.

---

**Note** 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.

---

A **monitor** is required to display results on the target computer. However, you can access all the target information using xPC Target functions on the host computer.

### Communication

Select one of the following methods for the target computer to communicate with the host computer:

- One free supported **Ethernet adapter** (PCI, ISA, or USB) connected to a network (see “Network Communication” on page 2-20 for supported Ethernet adapters).

---

**Note** If you want to boot the target computer from the network, you must install on the target computer an Ethernet adapter card compatible with the **Preboot eXecution Environment (PXE)** specification.

---

- One free **serial port** (COM1 or COM2) with a 9-pin or 25-pin D-sub connector (see “Serial Communication” on page 2-31 for details). Use a serial null modem cable to connect the target computer to the host computer.

### **PC-Compatible Form Factors**

xPC Target supports the following target computer 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 computer. 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 *xPC Target I/O Reference*.

---

**Note** See the board manufacturer’s documentation for information on installing and connecting I/O boards.

---

### **Target Computer Software and BIOS**

At a minimum, xPC Target product requires the following software configuration on your target computer:

Software	Description
Operating system	None. The xPC Target kernel does not require an operating system installed on the target computer.
BIOS	PC compatible

### Target Computer BIOS Settings

Target computer BIOS settings such as the following are required to run the xPC Target software:

- RS-232 communication — If you are using RS-232 communications, enable COM ports for both host and target computers. Through the BIOS, verify that COM1 has a base address of 3F8 and an IRQ of 4. Verify that COM2 has a base address of 2F8 and an IRQ of 3. These are the default base address values. Do not change these values.

---

**Note** RS-232 host-target communication mode will be removed in a future release. Use TCP/IP instead.

---

- USB communication — If you are using USB communications, enable USB ports for the target computer.
- Plug-and-Play (PnP) operating system — Disable this feature so the PCI BIOS can set up the plugged-in PCI cards. The xPC Target kernel is not a PnP operating system; this feature must be 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 computer BIOS. Turn this option Off to enable the BIOS to detect and configure all boards.
- Hyper-threading — If your target computer supports hyper-threading capabilities, do not enable these capabilities. Enabling hyper-threading can degrade the performance of the target computer.
- Multicore processor — If your target computer includes a multicore processor (see “Multicore CPU Support” on page 2-7), you can configure

the xPC Target software to take advantage of the individual cores. See “Multicore Processor Configuration” and “Configuring Models for Targets with Multicore Processors”.

---

**Note**

- To take advantage of a multicore processor, you must disable hyper-threading in the target computer BIOS.
  - If the target computer has only a single-core processor, you cannot use the multicore capabilities of the xPC Target software.
- 

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

- Boot floppy disk
- CD/DVD bootable ROM
- Flash drive
- Removable hard drive
- Dedicated network boot
- Bootable hard drive

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

## Host Software Installation

In this section...
“Overview” on page 2-12
“xPC Target Turnkey System” on page 2-12
“License Requirements” on page 2-13
“Files on the Host Computer” on page 2-13
“Setting Your Initial Working Folder” on page 2-14
“Running MATLAB Remotely” on page 2-15
“Configuring the Host Computer for Your C Compiler” on page 2-16
“Installing the Microsoft .NET Framework” on page 2-19

### Overview

You install the xPC Target software only on the host computer. The host computer downloads the kernel software and target application to the target computer at runtime. The xPC Target software is distributed on a DVD or as a file you download from the Web.

---

**Note** Before you start, verify that the xPC Target and xPC Target Embedded Option products are not already installed on your host computer. If they are, uninstall them both before proceeding.

---

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 2-31 or “Network Communication” on page 2-20.

### 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 computer from a device other than a floppy disk or CD/DVD and deploy standalone target applications separate from the host computer.

## Files on the Host 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 2-14.

- 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 computer 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 2-14.

## 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 Host Computer for Your C Compiler

To configure the host computer 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

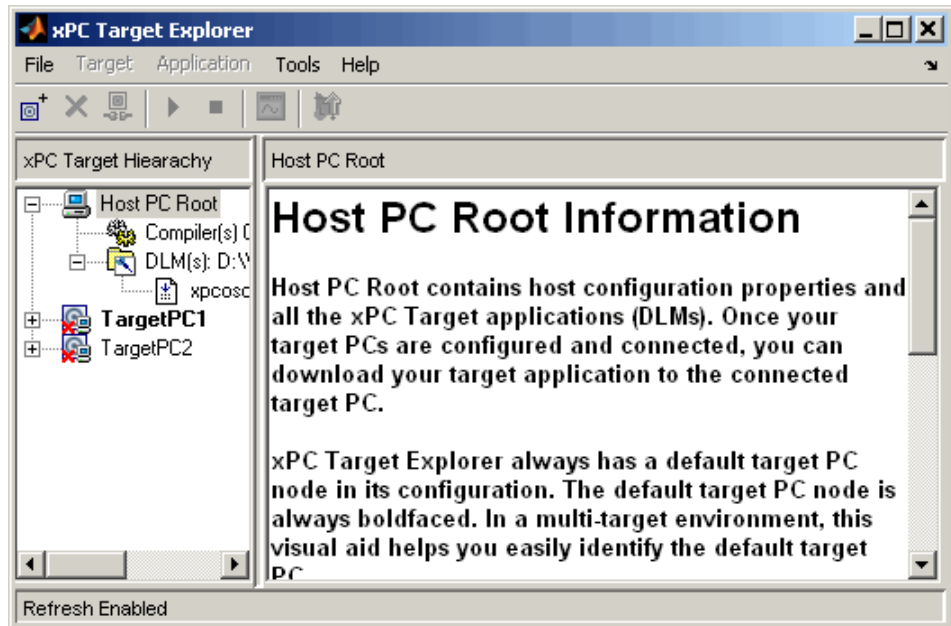
```
xpcexplr
```

---

**Note** xPC Target Explorer runs only on 32-bit host computers. On 64-bit computers, use the MATLAB command-line interface. For further information, see “Configuring Environment From the MATLAB Command Line”.

---

The xPC Target Explorer window appears.



xPC Target Explorer includes a default target computer node in its configuration, marked with boldface. In a multitarget environment, this visual aid helps you easily identify the default target computer.

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

In the right pane, the compiler parameters appear.

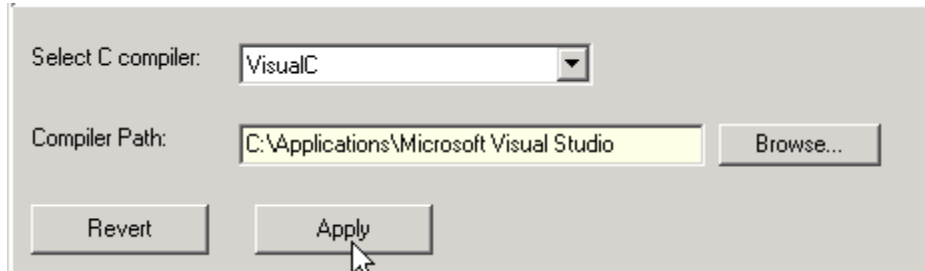
- 3 At the **Select C Compiler** drop-down list, select the compiler you have installed on the host computer. The examples in this chapter use VisualC.

---

**Note** The WATCOM 1.8 compiler will not compile all models. If a compilation fails, use a Microsoft compiler instead. WATCOM compiler support will be removed in a future release.

---

- 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 Computer for Your C Compiler with a Command-Line Interface

To configure the host computer for the C compiler:

- 1 In the MATLAB Command Window, type

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

The function queries the host computer 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:
```

---

**Note** The WATCOM 1.8 compiler will not compile all models. If a compilation fails, use a Microsoft compiler instead. WATCOM compiler support will be removed in a future release.

---

- 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 required 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...

## Installing the Microsoft .NET Framework

To install Microsoft .NET Framework from the xPC Target installation directory:

- 1 Navigate to `matlabroot\toolbox\rtw\targets\xpc\xpc\bin\dotnetfx`
- 2 Click `dotNetFx40_Full_x86_x64.exe`

## Network Communication

In this section...
“Network Communication Overview ” on page 2-20
“Hardware for Network Communication” on page 2-20
“Ethernet Card for a PCI Bus” on page 2-21
“Ethernet Card for an ISA Bus” on page 2-22
“Environment Properties for Network Communication” on page 2-24

### Network Communication Overview

This topic describes the establishment of communication between the host computer and target computer using network communications (TCP/IP). For serial communication, see “Serial Communication” on page 2-31.

### 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 should have a supported network (Ethernet) adapter (PCI, ISA, or USB) installed on the host and target computers.

---

**Note** To boot the target computer from the network, you must install on the target computer an Ethernet adapter card compatible with the Preboot eXecution Environment (PXE) specification.

---

Be sure to:

- Connect the Ethernet adapters of the host and target computers to a local area network (LAN) using unshielded twisted pair (UTP) cable.
- Assign a static IP address to the target computer network adapter card or USB-to-Ethernet converter.



For the most current network communications requirements, see

[http://www.mathworks.com/products/xpctarget/-supported-hardware/xPC\\_Target\\_Supported\\_Ethernet\\_Chipsets.pdf](http://www.mathworks.com/products/xpctarget/-supported-hardware/xPC_Target_Supported_Ethernet_Chipsets.pdf)

The host computer network adapter card can have a Dynamic Host Configuration Protocol (DHCP) address. The host computer 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 computer 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 computer, install the boards according to the manufacturer's instructions.

## **Ethernet Card for a PCI Bus**

If your target computer has a PCI bus, 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 computer.
- 2** If the target computer 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 computer Ethernet card to your LAN using an unshielded twisted-pair (UTP) cable.

Your next task is to set up the xPC Target environment for network communication. See “Environment Properties for Network Communication” on page 2-24.

### **Ethernet Card for an ISA Bus**

Your target computer might not have an available PCI bus slot or USB 2.0 port or 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.

---

**Tip** 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/-supported-hardware/xPC\\_Target\\_Supported\\_Ethernet\\_Chipsets.pdf](http://www.mathworks.com/products/xpctarget/-supported-hardware/xPC_Target_Supported_Ethernet_Chipsets.pdf)

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

- 1** Turn off your target computer.
- 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 computer, 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 computer already has an unsupported Ethernet card, remove the card. Plug the compatible network card into a free ISA bus slot.
- 4** Connect the target computer network card to your LAN using a coaxial cable or an unshielded twisted-pair cable.

If you use an Ethernet card for an ISA bus within a target computer 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 2-24.

### 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
```

---


**Note** xPC Target Explorer runs only on 32-bit host computers. On 64-bit computers, use the MATLAB command-line interface. For further information, see “Configuring Environment From the MATLAB Command Line”.

---

The xPC Target Explorer window opens.

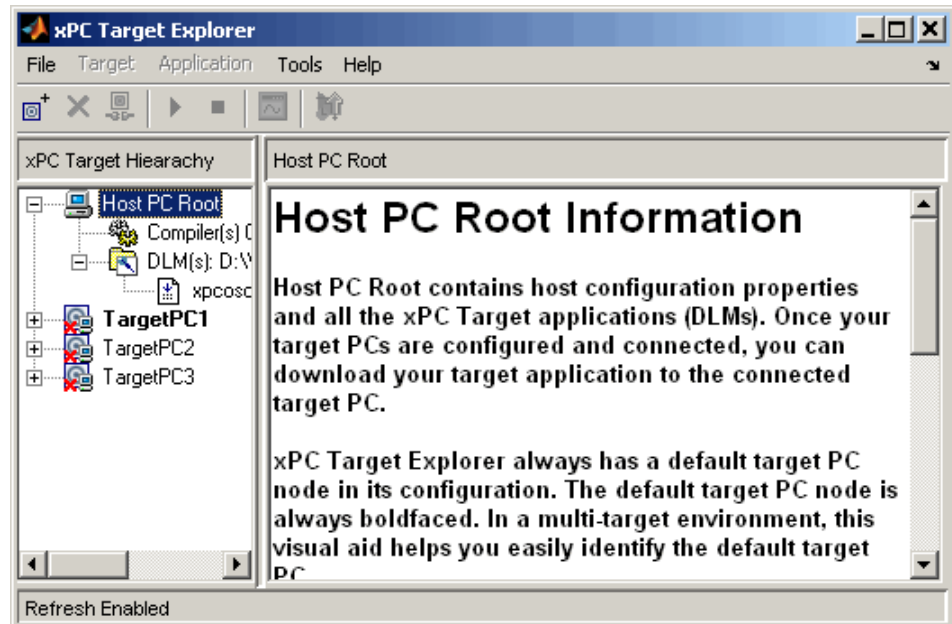
xPC Target Explorer associates network communication environment properties with the target computer.

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

A target computer 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 computer is not connected to the target computer).

- 4 Repeat and for each additional target computer you want to add to your system.

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



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

A Configuration node appears. Under this are nodes for Communication, Settings, and Appearance. The parameters for the target computer 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 values required by your LAN environment. Ask your system administrator for values for these settings.
- **Target PC IP address** — This is the IP address for your target computer. 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 computer. 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**.

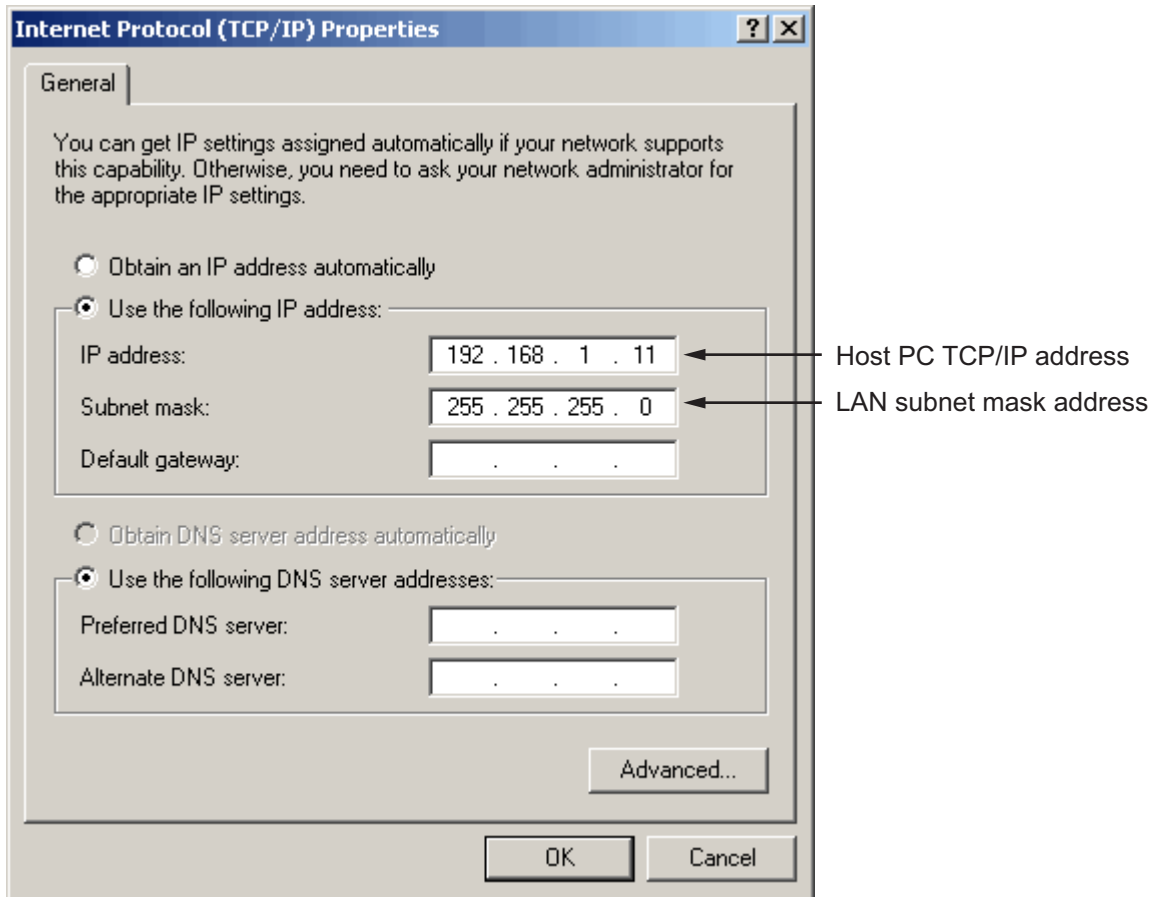
If your computers connect with a crossover cable, you might have a dialog box like the following. You can obtain your subnet mask address and TCP/IP gateway address from this dialog box.

---

**Note** The TCP/IP address is for your host computer, not your target computer. You still need to get the target computer TCP/IP address for your target computer 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 computer. 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 computer. If you connect your computers with a crossover cable, leave this property as 255.255.255.255.

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

If you communicate from a host computer located in a LAN different from your target computer, 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 required gateway.

**10** Enter the following properties specific to the Ethernet card on your target computer:

- **TCP/IP target bus type** — From the list, select: PCI, ISA, or USB.
  - The software defaults this property to PCI.
  - If you set **TCP/IP target bus type** to PCI or USB, the properties **TCP/IP ISA memory port** and **TCP/IP target ISA IRQ number** are disabled (grayed out).
- **TCP/IP target driver** — The software defaults this property to Auto. From the list, select:
  - For **TCP/IP target bus type** PCI, one of 3C90x, I8254x, I82559, NE2000, NS83815, R8139, R8168, Rhine, RTLANCE, or Auto.

If **TCP/IP target driver** is Auto, the software will determine the target computer TCP/IP driver from the card installed on the target computer.

---

**Note** If no supported Ethernet card exists in your target computer, the software returns an error.

---

- For **TCP/IP target bus type** USB, one of USBAX772, USBAX172, or Auto.

If **TCP/IP target driver** is Auto, the software will default the driver to USBAX772, the driver most commonly used.
- For **TCP/IP target bus type** ISA, one of NE2000 or SMC91C9X.

Auto is not supported for **TCP/IP target bus type** ISA.



---

**Note** To configure the software for a crossover Ethernet cable connection, select I82559.

---

- **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 computer has multiple Ethernet cards, type the following MATLAB commands to specify which card to use. These commands assume that the target computer 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 computer card.

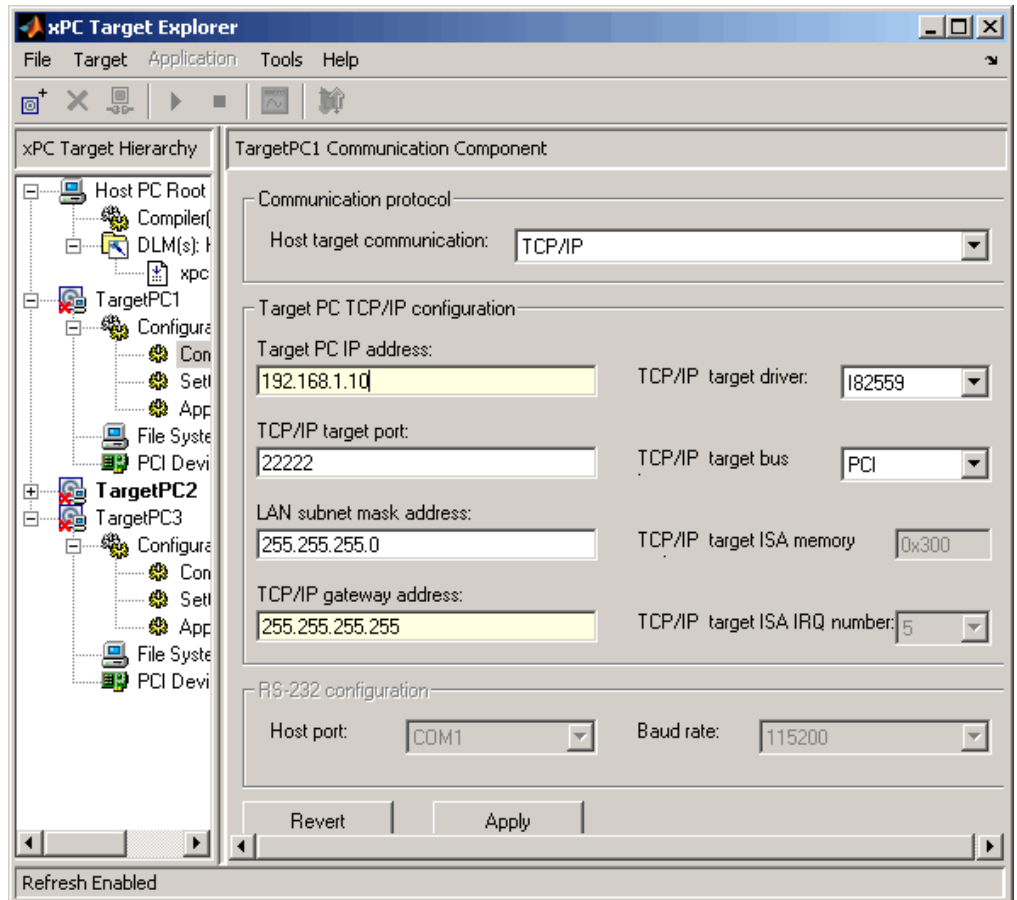
After you type this command, boot the target. The kernel selects the specified Ethernet card as the target computer card instead of selecting the default card with index number 0.

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

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

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 “Target Application Environment” in the *xPC Target User’s Guide*.

Your next task is to create a target boot drive. See “Booting Target Computers from Removable Boot Drives” on page 2-51.

## Serial Communication

In this section...
“Serial Communication Overview” on page 2-31
“Hardware for Serial Communication” on page 2-31
“Environment Properties for Serial Communication” on page 2-32

### Serial Communication Overview

---

**Note** RS-232 Host-Target communication mode will be removed in a future release. Use TCP/IP instead.

---

This topic describes the establishment of communication between the host computer and target computer using serial communications (RS-232). For network communication, see “Network Communication” on page 2-20.

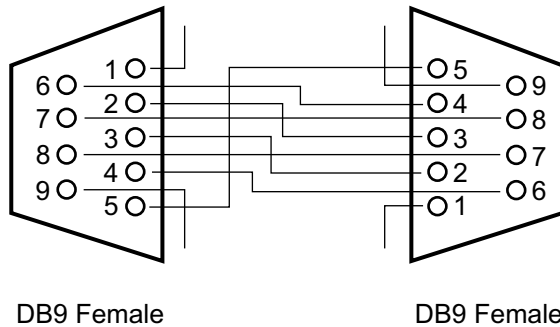
### Hardware for Serial Communication

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 a null modem cable. You can use either the COM1 or COM2 port.
- I/O boards — If you use I/O boards on the target computer, you need to install the boards according to the manufacturer’s instructions.

### Null Modem Cable Wiring

Use a null modem cable 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 computer and target computer, 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 computer (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
```

---


**Note** xPC Target Explorer runs only on 32-bit host computers. On 64-bit computers, use the MATLAB command-line interface. For further information, see “Configuring Environment From the MATLAB Command Line”.

---

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

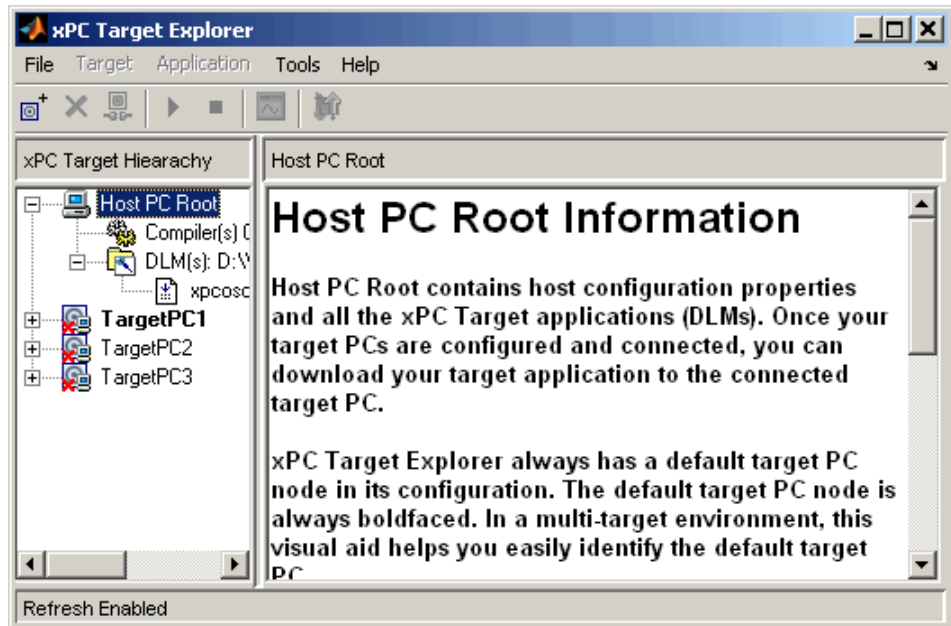
xPC Target Explorer associates serial communication environment properties with the target computer.

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

A target computer 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 computer is not connected to the target computer).

- 4 Repeat step 2 and step 3 for each additional target computer you want to add to your system.

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



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

Configuration, File System, and PCI Devices nodes appear. You work with the Configuration node to configure the target computer node for a target computer. The File System node contains the contents of a target computer file system. PCI Devices lists all PCI devices detected on the target computer. 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 computer 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 computer node, the target computer node becomes boldfaced. In a multitarget environment, this visual aid helps you easily see the target computer you are working with.

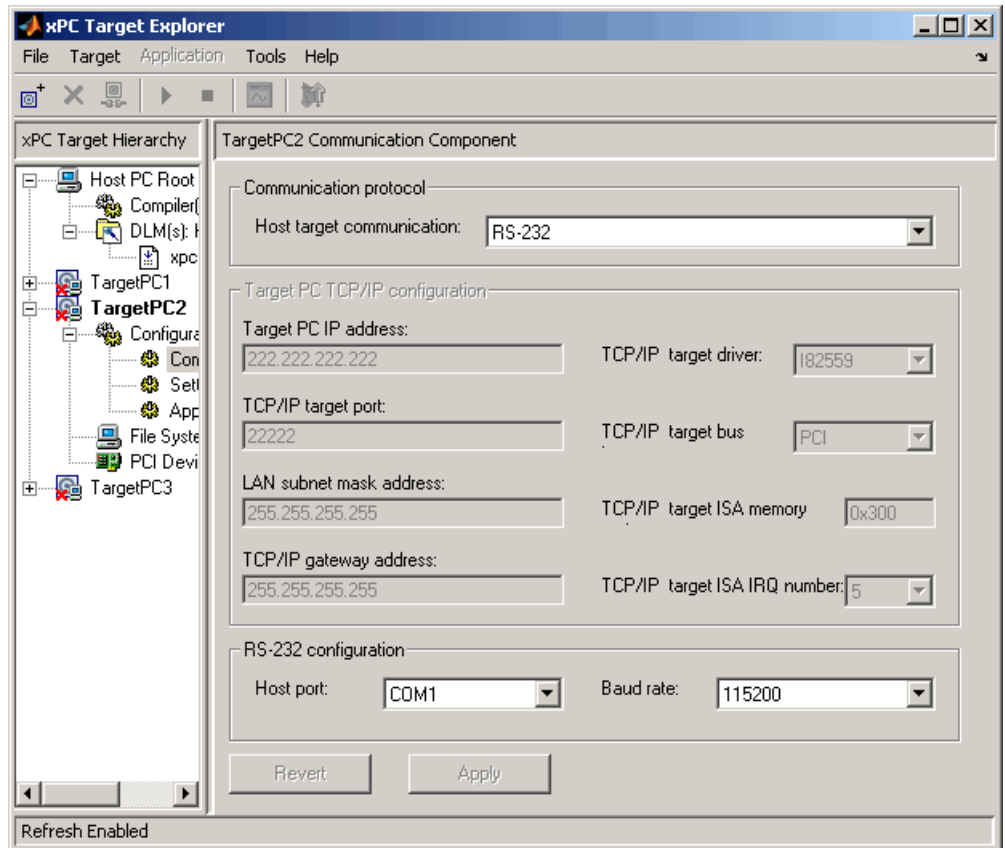
---

- 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 computer. The xPC Target software determines the COM port you use on the target computer automatically.
- 9** From the **Baud rate** list, select the baud rate for the serial connection between the host computer and this target computer. The default is 115200. Note that for better 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 computer for which you have a serial connection between the host computer and target computer.

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



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 drive and rebuild the target application from the Simulink model.

For more information on the xPC Target environment, see “Target Application Environment” in the xPC Target User’s Guide.



Your next task is to create a target boot drive. See “Booting Target Computers from Removable Boot Drives” on page 2-51.

# Target Boot Method

You can boot your target computer with the xPC Target kernel using one of the following methods from the xPC Target Explorer. Target boot drives and boot images include the xPC Target kernel specific for either serial or network communication.

---

### Tip

- xPC Target Turnkey systems come with the required software preinstalled. See your xPC Target Turnkey system user documentation for further information.
  - Before you create a target boot drive, set write permission for your current working folder. You cannot create a boot drive otherwise.
- 

In this section...
“Before You Boot” on page 2-38
“Booting Target Computers from CD or DVD” on page 2-39
“Booting Target Computers Within a Dedicated Network” on page 2-46
“Booting Target Computers from Removable Boot Drives” on page 2-51
“DOS Loader Boot Method” on page 2-57
“Embedded Target Boot Method” on page 2-62

## Before You Boot

Configure your xPC Target system before you create your boot drive or boot image. At a minimum, perform 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**

- **Network Boot**
- **Boot Floppy**
- **DOS Loader**
- **Embedded Target**
- Check the C compiler specification (see “Configuring the Host Computer for Your C Compiler” on page 2-16).
- If you are using TCP/IP communication, check your network connections. Also check the xPC Target Explorer settings (see “Network Communication” on page 2-20).
- If you are using serial communication, check your physical connections. Also check the xPC Target Explorer settings (see “Serial Communication” on page 2-31).
- Check your target computer BIOS settings (see “Target Computer BIOS Settings” on page 2-10).

## **Booting Target Computers from CD or DVD**

- “Creating a Boot CD/DVD with xPC Target Explorer” on page 2-39
- “Creating a Boot CD/DVD with a Command-Line Interface” on page 2-43

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 Computers” 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:

- Acquire an empty, writable CD or DVD.
- Acquire a CD/DVD-RW drive.

- Choose a process for creating 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 computer 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 drive.

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

```
xpcexplr
```

---

**Note** xPC Target Explorer runs only on 32-bit host computers. On 64-bit computers, use the MATLAB command-line interface. For further information, see “Configuring Environment From the MATLAB Command Line”.

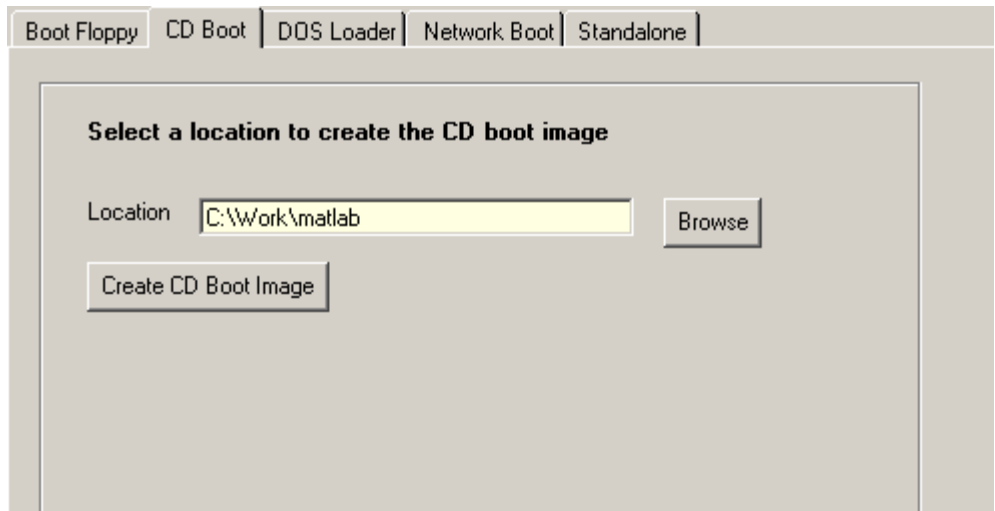
---

- 3 In the xPC Target Explorer **xPC Target Hierarchy** pane, select a target computer 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.

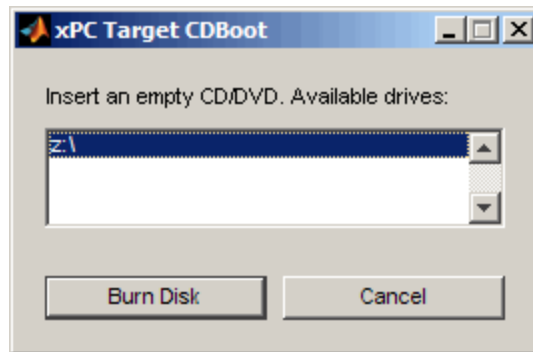
---

**Note** At this point, you have defined how xPC Target will communicate with the target computer. RS-232 Host-Target communication mode will be removed in a future release. Use TCP/IP instead.

---

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.



Insert the CD or DVD in the drive, 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 computer CD/DVD drive and reboot that computer.

Your next task is to test your installation. See “Running the Confidence Test” on page 2-72.

## 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:

- Acquire an empty, writable CD or DVD.
- Acquire a CD/DVD-RW drive.
- Choose a process for creating 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 computer 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 computer.

**2** In the MATLAB window, type

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

**3** Verify the following property settings:

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

**4** If these properties are not set to the required values, set them. For example:

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

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

```
xpcbootdisk
```

---

**Note** At this point, you have defined how xPC Target will communicate with the target computer. RS-232 Host-Target communication mode will be removed in a future release. Use TCP/IP instead.

---

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
```

Insert the CD or DVD in the drive, 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 test your installation. See “Running the Confidence Test” on page 2-72.

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

### Booting Target Computers Within a Dedicated Network

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

- Set up a dedicated network.
- Use the xPC Target Explorer to configure the target computer and host environments. You should be familiar with the following sections:
  - “Configuring the Host Computer for Your C Compiler” on page 2-16
  - “xPC Target Explorer”
  - “Network Communication” on page 2-20
  - “Booting Target Computers from Removable Boot Drives” on page 2-51

---

**Caution** Do not boot a target computer 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 Computer

- 1 Identify the target computer that you want to boot over the dedicated network.
- 2 Install the Ethernet card or USB-to-Ethernet adapter.
  - Verify that:
    - The xPC Target product supports your Ethernet card (see “Hardware for Network Communication” on page 2-20).
    - Your Ethernet card has a boot ROM that is compatible with the Preboot eXecution Environment (PXE) specification.
- 3 Connect the host computer and the target computers within the dedicated network. For example, connect one end of a crossover cable to the dedicated

network card of the host computer and connect the other end of this cable to the dedicated network card of the target computer.

- 4 Turn on the target computer.
- 5 Enter BIOS and set up the target computer for a LAN or network boot. If the BIOS allows you to change the boot order, consider setting the boot order so that the removable/boot floppy disk is the first option and the LAN is the second. This allows you to boot the target computer from a network kernel even if the target computer does not have an xPC Target boot disk or removable drive.

## Configuring for Network Booting

This procedure is similar to configuring a target boot drive. If you have previously created a target boot drive, you might not need to perform this procedure. However, you should still read the following instructions to verify that your configuration allows you to boot a target computer in the dedicated network. You can configure multiple target computers for your network.

- 1 Verify that the host computer has a network card available for the dedicated network. If no card is available, insert a new card and configure it for the dedicated network. This step includes assigning the host computer a unique IP address (for example, 10.10.10.10) in the same subnet as the target computer.
- 2 If the xPC Target Explorer is not open, open it now. At the MATLAB Command Window, type

```
xpcexplr
```

---

**Note** xPC Target Explorer runs only on 32-bit host computers. On 64-bit computers, use the MATLAB command-line interface. For further information, see “Configuring Environment From the MATLAB Command Line”.

---

- 3 Add a target computer, for example, TargetPC3.
- 4 In the **Communication Component** pane for TargetPC3,

- a In the **Host target communication** field, select TCP/IP.
- b Enter a target computer IP address in the dedicated network, for example, 10.10.10.11. The subnet of this IP address must be the same as the host computer, otherwise Network Boot will fail.
- c Enter 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 several tabs: "Boot Floppy", "CD Boot", "DOS Loader", "Network Boot" (selected), and "Standalone". The main content area is titled "Create Netboot Image and select the discovery mode for the target PC". It displays "IP Address: 10.10.10.11" and a "Create Network Boot Image" button. To the right, it says "Last created on 12-May-2008 11:11:47". Below this is a section for "Target PC Ethernet Configuration" containing 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)" which is unselected. Below the radio buttons are six empty input boxes for the 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 computer MAC address with the xPC Target Explorer target computer name, or allow the

software to automatically find target computer MAC addresses. If you want to associate your physical target computer MAC address,

- a Click **Manual**.
- b In the six fields, enter the physical target computer 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 computer. You will boot the target computer using this process.

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

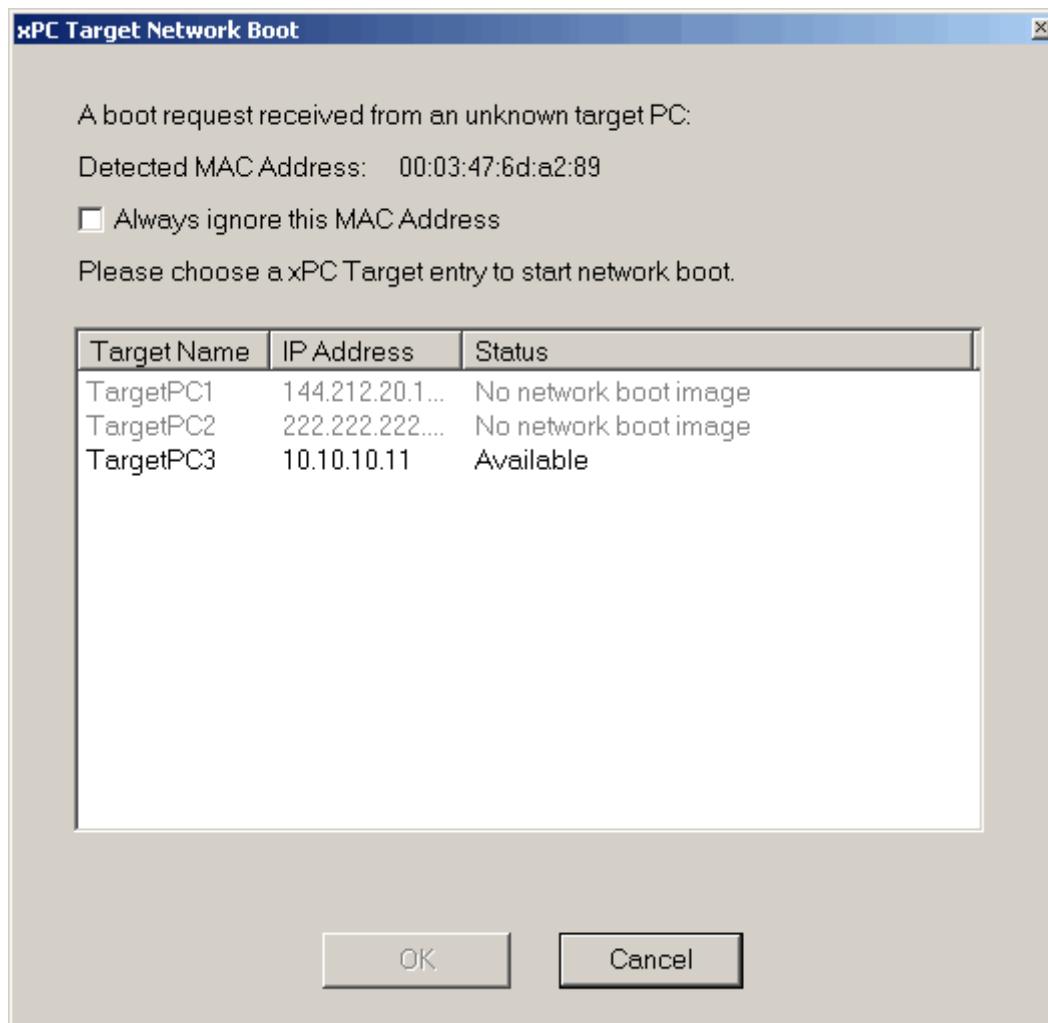
Your next task is to boot the target computer from the network. See “Booting the Target Computer From the Network” on page 2-49

## **Booting the Target Computer From the Network**

**1** Reboot the target computer.

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

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



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

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

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

Note the following behavior:

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

Your next task is to test your installation. See “Running the Confidence Test” on page 2-72.

## **Booting Target Computers from Removable Boot Drives**

You use a removable boot drive 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 drive. Note that this topic assumes you are using default environment parameter settings for the target boot drive creation. If this is not the case, see “Configuring Environment Parameters for Target Computers” in the *xPC Target User’s Guide* for further details.

### **Creating a Target Boot Drive with xPC Target Explorer**

To create a target boot drive for the current xPC Target environment, use the following procedure. This procedure describes how to create a target boot drive for the target TargetPC2. Alternatively, see “Creating a Target Boot Drive with a Command-Line Interface” on page 2-54.

---

**Tip** If you are creating a target boot drive from a USB flash drive, you must create a bootable partition on the drive before performing this procedure. See “Creating a Bootable Partition” on page 2-56.

---

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

```
xpcexplr
```

---

**Note** xPC Target Explorer runs only on 32-bit host computers. On 64-bit computers, use the MATLAB command-line interface. For further information, see “Configuring Environment From the MATLAB Command Line”.

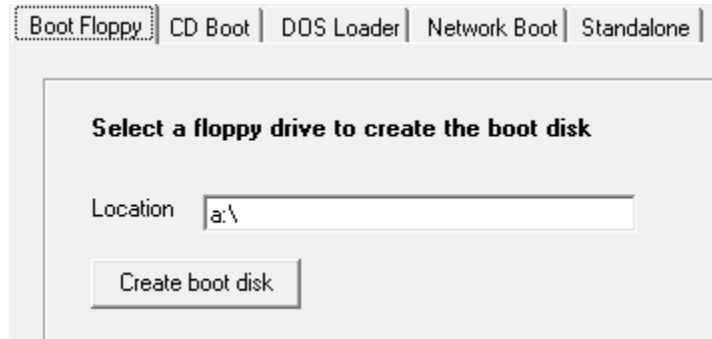
---

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

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

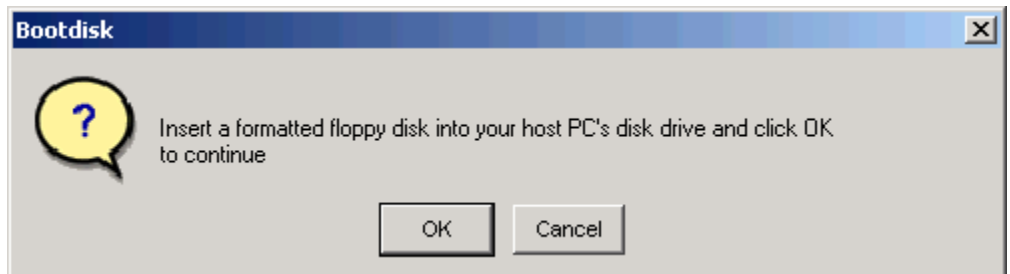


- 3 From the tab list, select the **Boot Floppy** tab.



- 4 Change the drive letter to a valid removable 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 removable drive in the host computer, and then click **OK**.

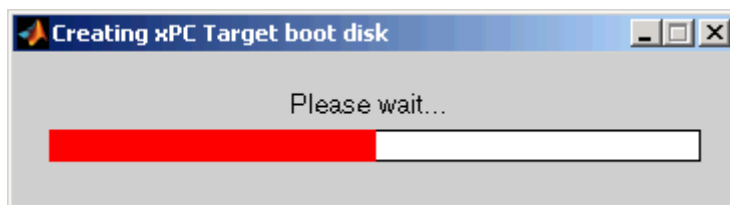
---

**Note** At this point, you have defined how xPC Target will communicate with the target computer. RS-232 Host-Target communication mode will be removed in a future release. Use TCP/IP instead.

---

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

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



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

Your next task is to test your installation. See “Running the Confidence Test” on page 2-72.

### **Creating a Target Boot Drive with a Command-Line Interface**

You use the target boot drive 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 drive. The following makes changes to the TargetPC2 environment.

---

**Tip** If you are creating a target boot drive from a USB flash drive, you must create a bootable partition on the drive before performing this procedure. See “Creating a Bootable Partition” on page 2-56.

---

- 1 In the MATLAB window, type

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

```
env=tgs.Item('TargetPC2')
```

**2** Set the following xPC Target properties as follows:

- `TargetBoot` — `BootFloppy`
- `BootFloppyLocation` — Your host computer removable drive location

**3** If these properties are not set to the required values, set them. For example:

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

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

```
xpcbootdisk
```

---

**Note** At this point, you have defined how xPC Target will communicate with the target computer. RS-232 Host-Target communication mode will be removed in a future release. Use TCP/IP instead.

---

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 removable drive in the host computer, and then press any key.

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

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

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

Your next task is to test your installation. See “Running the Confidence Test” on page 2-72.

### **Creating a Bootable Partition**

Before you create a target boot drive from a USB flash drive, you must create a bootable partition on the drive.

- 1 On the host computer, open a DOS command window.
- 2 In the DOS command window, type

```
diskpart
```

- 3 At the diskpart prompt, type

```
list disk
```

Make a note of the disk numbers of the existing host computer disks.

- 4 Insert the flash drive. Wait for the host computer to recognize the drive. Make a note of the drive device name.

- 5 At the diskpart prompt, type

```
list disk
```

Make a note of the new disk number *N*. This number is the disk number of the drive.

- 6 Type

```
select disk N
```

All of the next steps change disk *N*.

---

**Caution** Select the correct disk, or you might delete all of the data from your host computer.

---

- 7 Type

```
clean
```

**8** Type

```
create partition primary
```

**9** Type

```
select partition 1
```

**10** Type

```
active
```

**11** Type

```
format fs=fat32 quick
```

**12** Type

```
exit
```

**13** In the host computer task bar, click **Safely remove hardware and eject media** and select the device name of the flash drive.

**14** Remove the drive from the host computer.

## DOS Loader Boot Method

DOSLoader mode allows you to boot the xPC Target kernel on a target computer from a fixed or removable device with DOS boot capability, such as a hard disk or flash memory. After booting the target computer, you can download your application from the host computer over the serial or network communication between the host and target computers. See “DOSLoader Mode Setup” on page 2-58 for details.

- “DOSLoader Mode Setup” on page 2-58
- “Restrictions” on page 2-59
- “Creating a Target Application for DOSLoader Mode” on page 2-60
- “Creating DOSLoader Files with a Command-Line Interface” on page 2-60

- “Creating a DOS System Disk” on page 2-61

### DOSLoader Mode Setup

The following is a step-by-step procedure for using DOSLoader mode. This procedure assumes you have serial or network communication between your host and target computers. Also note that to use this mode, you need a minimal DOS system on the target computer boot device. See “Creating a DOS System Disk” on page 2-61 for details.

- 1 On the host computer, start a MATLAB session.
- 2 In the MATLAB Command Window, type:

```
xpcexplr
```

---

**Note** xPC Target Explorer runs only on 32-bit host computers. On 64-bit computers, use the MATLAB command-line interface. For further information, see “Configuring Environment From the MATLAB Command Line”.

---

The xPC Target Explorer window opens.

- 3 In the **xPC Target Hierarchy** pane, expand the node of the target computer that you want to set up for DOSLoader mode and select the **Configuration** node.
- 4 In the **Configuration** pane, select the **DOSLoader** tab.
- 5 In the **Location** field, enter or browse to the directory where you want to create the DOSLoader boot files and click **OK**. This location can be a local directory on the host computer or a removable storage device that you will use to boot the target computer. By default, the directory is the current working directory.
- 6 Click **Apply**.
- 7 Click **Create DOS Loader**.

---

**Note** At this point, you have defined how xPC Target will communicate with the target computer. RS-232 Host-Target communication mode will be removed in a future release. Use TCP/IP instead.

---

This operation creates the following boot files in the specified location:

```
autoexec.bat
xpcboot.com
*.rtb
```

- 8** If you create boot files on a local hard disk, copy these files to a floppy disk, CD/DVD, or any other removable storage media.
- 9** Transfer the boot files to your target computer or insert the removable media containing the boot files for booting the target computer.
- 10** Place the `autoexec.bat` file on the DOS boot path, which is typically the root directory.
- 11** Select the required boot device in the BIOS of the target computer.
- 12** Boot the target computer.

When the target computer boots, it loads DOS, which starts the `autoexec.bat` file. This file starts the xPC Target kernel (`*.rtb`). The target computer then awaits commands from the host computer.

## Restrictions

To use DOSLoader mode, your DOS environment must comply with the following restrictions:

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

To satisfy these restrictions:

- Do not use additional memory managers like `emm386` or `qemm`.

- Avoid any utilities that attempt to load in high memory (for example, `himem.sys`). If the target computer DOS environment does not use a `config.sys` file or memory manager entries in the `autoexec.bat` file, there should typically be no problems when starting the xPC Target software.

### Creating a Target Application for DOSLoader Mode

After booting the target computer using DOSLoader mode, create a target application on a host computer and download it to the target computer.

Set the Simulink and Simulink Coder parameters for code generation with the xPC Target software in your Simulink model. Next, build and download the application to the target computer.

- 1 In the MATLAB Command Window, type the name of a Simulink model. For example:

```
xpc_osc3
```

A Simulink window opens with the model.

- 2 From the **Tools** menu, select **Code Generation**, and then click **Build Model**.

The Simulink Coder and xPC Target products create a target application and download it to your target.

### Creating DOSLoader Files with a Command-Line Interface

To create DOSLoader files for the current xPC Target environment and use them to boot the target computer, use the following procedure:

- 1 For a specific target computer, retrieve the specific target computer environment object from the `xpctarget.targets` class. Then, set the value of the object properties `TargetBoot` and `DOSLoaderLocation`. For example:

```
tgs = xpctarget.targets;  
tgEnv = tgs.Item('TargetPC2');  
set(tgEnv, 'TargetBoot', `DOSLoader`);  
set(tgEnv, 'DOSLoaderLocation', 'c:\work\xpc\dosloader_files');
```



Alternatively, if you have a single target computer environment, you can use the `setxpcenv` functions, as follows to set the object properties `TargetBoot` and `DOSLoaderLocation`:

```
setxpcenv('TargetBoot','DOSLoader')  
setxpcenv('DOSLoaderLocation','c:\work\xpc\dosloader_files')
```

- 2 In the MATLAB Command Window, type:

```
xpcbootdisk
```

---

**Note** At this point, you have defined how xPC Target will communicate with the target computer. RS-232 Host-Target communication mode will be removed in a future release. Use TCP/IP instead.

---

The xPC Target software displays the following message and creates the DOSLoader files:

```
Current boot mode: DOSLoader  
xPC Target DOS Loader files are successfully created
```

- 3 Transfer the DOSLoader files as described in “DOSLoader Mode Setup” on page 2-58.

### Creating a DOS System Disk

To use the DOSLoader mode, you need a minimal DOS system on the target computer boot device. MathWorks has tested the xPC Target product with FreeDOS Beta 8 (“Nikita”) distribution, MS-DOS (6.0 or higher), PC DOS, and Caldera OpenDOS. You can use a copy of any of these DOS systems to boot the target computer.

---

**Note** xPC Target software does not include a DOS license. You must obtain a valid DOS license for your target computer.

---

To create a DOS system disk, use the following DOS command to copy the DOS system files and command interpreter from *drive1* to the boot device, *drive2*.

```
sys drive1 drive2
```

It is helpful to copy additional DOS utilities to the boot device, including:

- A DOS editor to edit files
- The format program to format a hard disk or flash memory
- The fdisk program to create partitions
- The sys program to transfer a DOS system onto another drive

Once you have created the DOS System disk, you can transfer the DOSLoader files created using `xpcexplr` or the MATLAB command line to the disk. A `config.sys` file is not required.

### **Embedded Target Boot Method**

The xPC Target Embedded Option Standalone Mode software allows you to bundle the kernel and target application into one entity on the target computer independent of the host computer. You can configure a target computer to automatically start execution of your embedded application for continuous operation each time the system is booted. You can control the target application with the command-line interface using the target computer keyboard. You can also control it from the host using custom GUIs or the Web browser interface. You can deploy host-side GUIs developed with the xPC Target C, COM and .NET APIs on any Microsoft Windows host computer without installing MATLAB software. See the *xPC Target API Guide* for more information.

### **Standalone Mode Embedded Option**

- “Introduction” on page 2-63
- “Workflow” on page 2-64
- “Restrictions” on page 2-66

**Introduction.** The xPC Target Embedded Option software extends the xPC Target base product with Standalone mode. Standalone mode enables you to deploy control systems, DSP applications, and other systems on PC hardware for use in production applications using PC hardware. Typically these production applications are found in systems where production quantities are low to moderate.

Use this mode to load the target computer with both the xPC Target kernel and a target application. This mode of operation can start the kernel on the target computer from a flash disk or hard disk. After starting the kernel on the target computer, Standalone mode also automatically starts the target application that you loaded with the kernel. Standalone mode eliminates the need for a host computer and allows you to deploy real-time applications on target computers. See “Standalone Mode Embedded Option” on page 2-62 for further details.

Regardless of the mode, you initially boot your target computer with DOS from any boot device, then the xPC Target kernel is started from DOS. The xPC Target software only needs DOS to boot the target computer and start the xPC Target kernel. DOS is no longer available on the target computer unless you reboot the target computer without starting the xPC Target kernel.

---

**Note** The xPC Target Embedded Option software requires a boot device with DOS installed. It otherwise does not have any specific requirements as to the type of boot device. You can boot the xPC Target software from any device that has DOS installed. DOS software and license are not included with the xPC Target or xPC Target Embedded Option software.

---

Without the xPC Target Embedded Option software, you can only download real-time applications to the target computer after booting the target computer from an xPC Target boot disk or network boot image.

The following are some instances where you might want to use the xPC Target Embedded Option product. You might have one of these situations if you deploy the target computer in a small or rugged environment.

- Target computer does not have removable drive.

- Target computer removable drive must be removed after setting up the target system.
- Target computer does not support network boot from host computer.

If you do not want to view signals on the target computer, you do not need a monitor for the target computer, nor do you need to add target scopes to the application. In this instance, your xPC Target system operates as a black box without a monitor or keyboard. Standalone applications are automatically set to continue running for an infinite time duration or until the target computer is turned off.

**Workflow.** The following summarizes the workflow for Standalone mode. For a more detailed procedure, see “Standalone Target Setup” on page 2-66.

- 1 Verify that the target computer has a supported version of DOS on the target computer hard drive.

---

### Note

- The xPC Target Embedded Option product does not include a DOS license. You must obtain a valid DOS license for your target computer.
  - MathWorks has tested the xPC Target software with FreeDOS Beta 8 (“Nikita”) distribution, MS-DOS (6.0 or higher), PC DOS, and Caldera OpenDOS.
- 

- 2 Create a standard boot device from CD ROM, 3.5-inch floppy drive, flash drive, or hard drive using the DOS command:

sys A:

At this time, it is helpful to copy additional DOS utilities to the boot disk, such as:

- DOS editor — to edit text files on the target computer
- format program — to format a hard disk or flash memory
- fdisk program — to create partitions

- `sys` program — to transfer a DOS system onto another drive, such as the hard disk drive
- 3 Boot the target computer.
  - 4 From the host computer MATLAB window, type `xpcexplr`.

---

**Note** xPC Target Explorer runs only on 32-bit host computers. On 64-bit computers, use the MATLAB command-line interface. For further information, see “Configuring Environment From the MATLAB Command Line”.

---

- 5 In the xPC Target Explorer **xPC Target Hierarchy** pane, select a target computer **Configuration** node.
- 6 In the configuration node, select the **Standalone** tab.
- 7 Click the **Enable Standalone Mode** check box.
- 8 Select and build a model.

This step creates a directory in the current working folder named `modelname_xpc_emb`.

- 9 Copy the contents of `model_name_emb` to the target computer hard drive.

The target computer hard drive should now contain the following files:

- DOS files — DOS operating system and utilities (see step 1).
- `*.rtb` — xPC Target kernel, plus applicable options such as serial or TCP/IP communications and the IP address of the target computer.
- `xpcboot.com` — xPC Target code that loads and executes the `*.rtb` file.
- `autoexec.bat` — xPC Target-specific code that calls the `xpcboot.com` executable to boot the xPC Target kernel.

---

**Note** A `config.sys` file is not required.

---

### 10 Boot the target computer.

When you boot the target computer, the target computer loads DOS, which then calls the xPC Target `autoexec.bat` file to start the xPC Target kernel (`*.rtb`) and the 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.

---

**Note** This mode does not require any connection between the host computer and target computer.

---

**Restrictions.** To use the Standalone mode, your DOS environment must comply with the following restrictions:

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

To satisfy these restrictions,

- Do not use additional memory managers like `emm386` or `qemm`.
- Avoid any utilities that attempt to load in high memory (for example, `himem.sys`). If the target computer DOS environment does not use a `config.sys` file or include memory manager entries in the `autoexec.bat` file, there should be no problems when running `xpcboot.com`.

### Standalone Target Setup

- “Before You Start” on page 2-67
- “Updating Environment Properties” on page 2-67
- “Creating a Kernel/Target Application” on page 2-68
- “Copying the Kernel/Target Application to the Target Computer Flash Disk” on page 2-69

**Before You Start.** Standalone mode combines the target application with the kernel and boots them together on the target computer from the hard drive (or, alternatively, flash memory). The host computer does not need to be connected to the target computer.

Before you start, set up your system as described.

- 1** Create a standard boot disk or network boot image for serial or network communication (depending on your configuration). You will need to do this so that you can copy your Standalone mode files to the target computer. See “Serial Communication” on page 2-31, “Network Communication” on page 2-20, “Booting Target Computers from Removable Boot Drives” on page 2-51, and “Target Boot Method” on page 2-38 in the Chapter 2, “Installation and Configuration” chapter of the xPC Target™ Getting Started Guide on page 1.
- 2** Boot the target computer.
- 3** Verify that your target computer hard drive is a serial ATA (SATA) or parallel ATA (PATA)/Integrated Device Electronics (IDE) drive. The xPC Target product supports file systems of type FAT-12, FAT-16, or FAT-32. Verify that the hard drive is not cable-selected and that the BIOS can detect it.

After you create the standalone target application files, you will copy them to the target computer hard drive using the File Transfer Protocol (FTP) functions of the xPC Target file system. You do not need to be familiar with the xPC Target file system before you start, but for further information on this feature, see “Logging Signal Data with FTP and File System Objects”.

**Updating Environment Properties.** The xPC Target software uses the environment properties to determine what files to create for the various target boot modes.

This procedure assumes you have serial or network communication between your host computer and a target computer.

- 1** On the host computer, start the MATLAB interface.
- 2** In the MATLAB window, type

xpcexplr

---

**Note** xPC Target Explorer runs only on 32-bit host computers. On 64-bit computers, use the MATLAB command-line interface. For further information, see “Configuring Environment From the MATLAB Command Line”.

---

The xPC Target Explorer window opens.

- 3** In the xPC Target Explorer **xPC Target Hierarchy** pane, select a target computer **Configuration** node.
- 4** Click the **Standalone** tab.

The xPC Target software updates the environment properties, and the build process is ready to create a standalone kernel/target application. See “Creating a Kernel/Target Application” on page 2-68. For Standalone mode, you do not create an xPC Target boot disk or network boot image. Instead, you copy files created from the build process to the target computer hard drive.

**Creating a Kernel/Target Application.** Use the xPC Target software with Standalone mode to create a combined kernel and target application with utility files. A combined kernel and target application allows you to disconnect your target computer from a host computer and run standalone applications.

After you set the Simulink and Simulink Coder parameters for code generation with the xPC Target software in your Simulink model, you can use the xPC Target software with Standalone mode to create a target application:

- 1** In the MATLAB window, type the name of a Simulink model. For example, type

```
xpc_osc3
```

A Simulink window opens with the model.

- 2** From the **Tools** menu, point to **Code Generation**, and then click **Build Model**.



Simulink Coder and xPC Target software create a folder `xpc_osc3_xpc_emb` with the following files:

- `autoexec.bat` — This file is automatically invoked by DOS. It then runs `xpcboot.com` and the `*.rtb` file.
- `xpc_osc3.rtb` — This image contains the xPC Target kernel and your target application.
- `xpcboot.com` — This file is a static file that is part of the xPC Target Embedded Option software.

Refer to “Copying the Kernel/Target Application to the Target Computer Flash Disk” on page 2-69 for a description of how to transfer these files to the target computer.

---

**Note** If the size of the compiled target application (DLM) exceeds the **Maximum model size** you selected in xPC Target Explorer, the software will generate an error during the build process.

---

**Copying the Kernel/Target Application to the Target Computer Flash Disk.** You build a target application on a host computer using the Simulink Coder and xPC Target products, and a C/C++ compiler. One method for transferring the files from the host computer to a target computer is to use the FTP functions of the xPC Target file system.

After you build a standalone application on a host computer, you can copy files from the host computer to the target computer hard drive or flash disk. If you have not already created these files, see “Creating a Kernel/Target Application” on page 2-68.

- 1 Verify that your target computer is still booted from a target computer boot disk.
- 2 In the MATLAB Command Window, change folder on the host computer to the folder that contains the kernel/target application files.
- 3 Create the folder `C:\xpcfiles` and copy files to that folder. For example, type

```
f=xpctarget.ftp
f.mkdir('xpcfiles')
f.cd('xpcfiles')
f.put('autoexec.bat')
f.put('xpcboot.com')
f.put('xpc_osc3.rtb')
```

- 4 If you want your standalone application to run when you reboot your target computer, remove the 3.5-inch disk or CD from the target computer, reboot the target computer, and bring up the DOS prompt. For example, if you see the message for selecting the operating system to start, select Microsoft Windows.

---

**Note** If the target computer that you want to boot in Standalone mode was previously booted from the network boot image, selecting the **Enable Standalone Mode** check box should disable the network boot capability.

---

The boot process is stopped and a DOS prompt is displayed.

- 5 At the DOS prompt, save a copy of the target computer file C:\autoexec.bat to a backup file, such as C:\autoexec\_back.wrk.
- 6 Edit the target computer file C:\autoexec.bat to include the following lines. Adding these commands to C:\autoexec.bat directs the system to execute the autoexec.bat file located in C:\xpcfiles.

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

---

**Note** Do not confuse C:\xpcfiles\autoexec.bat with C:\autoexec.bat. The file C:\xpcfiles\autoexec.bat includes the command xpcboot.com to start the xPC Target kernel and standalone application. The file C:\autoexec.bat includes the files you want the system to execute when the system starts up.

---

- 7 Reboot the target computer.

- 8** The sequence of calls during the boot process is
- a** C:\autoexec.bat
  - b** C:\xpcfiles\autoexec.bat
  - c** C:\xpcfiles\xpcboot.com
  - d** C:\xpcfiles\<<application>.rtb

The standalone target application should now be running on the target computer.

## Running the Confidence Test

This topic describes how to boot a target computer with the xPC Target operating system and test the software and the connection between the host and target computers. The xPC Target software uses a test script to test the entire installation. Use this test under the following circumstances:

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

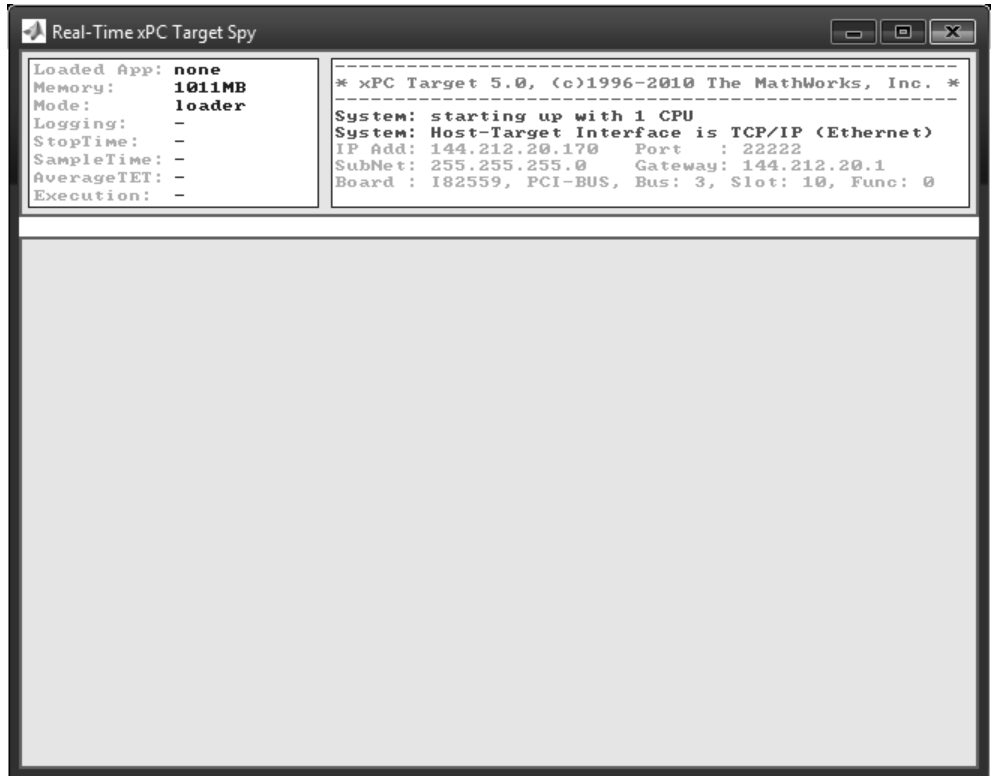
---

### Note

- This procedure assumes you are booting the target computer using a dedicated network. For more information, see “Booting Target Computers Within a Dedicated Network” on page 2-46. For alternative methods, see “Target Boot Method” on page 2-38.
- This procedure assumes that you have set environment settings with xPC Target Explorer. See “Environment Properties for Serial Communication” on page 2-32 or “Environment Properties for Network Communication” on page 2-24.
- xPC Target Explorer runs only on 32-bit host computers. On 64-bit computers, use the MATLAB command-line interface. For further information, see “Configuring Environment From the MATLAB Command Line”.

- 
- 1** Create a network boot image according to the procedure in “Configuring for Network Booting” on page 2-47.
  - 2** Reboot the computer according to the procedure in “Booting the Target Computer From the Network” on page 2-49.

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



If you have a keyboard attached to the target computer, you can activate that keyboard by typing **C**, and press the **Page Up** and **Page Down** keys to page up and down the target computer 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 computer and displays messages indicating whether the test passed or failed. 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
```

#### 5 Evaluate the results.

- If all of the tests return OK, you are ready to build and download a target application to the target computer. See Chapter 4, “Tutorial and Examples”.
- If any of the tests return FAILED, see “Confidence Test Failures”.

# Basic Workflows

---

- “Rapid Prototyping” on page 3-2
- “Hardware in the Loop” on page 3-7

## Rapid Prototyping

Use rapid prototyping to test a design with a minimal hardware plant model. In the process, you can start accumulating test data for use in later stages of production.

### 1 Create a Simulink or Stateflow model.

Create block diagrams in Simulink using simple drag-and-drop operations, and then enter values for block parameters and sample times.

---

**Note** For ease of adaptation, design and simulate the original Simulink model with real-time execution in mind. For example, when accumulating results data, set block parameters to discretized signal and fixed-step solver, and select a sample time compatible with the fixed-step solver.

---

### 2 Simulate the model in nonreal time.

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

Because the computed time vector is not connected to a 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.

You may log simulation results for later comparison.

### 3 Configure the host and target environment.

Configure the communication method between the host and target.

---

**Note** For target computer hardware, consider the xPC Target Turnkey solutions.

---

Configure the host and target environment using:



- “xPC Target Explorer” on page 1-27
- “MATLAB Command-Line Interface” on page 1-29

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

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

- Discretized signal
- Fixed-step solver
- Sample time compatible with the fixed-step solver

Add xPC Target I/O blocks representing your I/O boards. If you have a custom I/O board, you might need to create a custom driver block representing the board.

#### **5 Configure the build environment.**

Configure the build environment, including Simulink Coder options, xPC Target build options, and C compiler options, to create a target application that runs on the target computer.

At this point, you may configure the target application to run using xPC Target Embedded Option.

#### **6 Connect to external hardware.**

Install I/O boards in the target computer and connect the I/O boards to the hardware against which you want to execute the target application.

#### **7 Reboot the target computer.**

Boot the target computer with the xPC Target real-time kernel using:

- “xPC Target Explorer” on page 1-27
- “MATLAB Command-Line Interface” on page 1-29

#### **8 Build and download the target application.**

Build and download the real-time application using:

- “xPC Target Explorer” on page 1-27
- “MATLAB Command-Line Interface” on page 1-29
- “Simulink External Mode Interface” on page 1-31
- “Target Computer Command-Line Interface” on page 1-32
- “Web Browser Interface” on page 1-33

## **9 Execute the target application in real time.**

Execute the target application under command from the host computer or by booting the target computer in Standalone mode using the xPC Target Embedded Option.

The xPC Target software uses real-time resources on the target computer 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.

Execute using:

- “xPC Target Explorer” on page 1-27
- “MATLAB Command-Line Interface” on page 1-29
- “Simulink External Mode Interface” on page 1-31
- “Target Computer Command-Line Interface” on page 1-32
- “Web Browser Interface” on page 1-33

## **10 Visualize signals.**

Create xPC Target scopes and use them to acquire and display signal data from the target application.

Scopes created by xPC Target Scope blocks acquire data according to Simulink sample time rules. Scopes can gather data at the top level or in an enabled or triggered subsystem. Scopes created dynamically (from the MATLAB

Command Window or the API) sample at the base rate, irrespective of the sample time of their signals.

---

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

---

Visualize signals using:

- “xPC Target Explorer” on page 1-27
- “MATLAB Command-Line Interface” on page 1-29
- “Simulink External Mode Interface” on page 1-31
- “Simulink with xPC Target Blocks ” on page 1-32
- “Target Computer Command-Line Interface” on page 1-32
- “Web Browser Interface” on page 1-33

## 11 Tune parameters.

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

---

**Note** xPC Target does not support tuning parameters of complex or multiword data types.

---

Tune parameters using:

- “xPC Target Explorer” on page 1-27
- “MATLAB Command-Line Interface” on page 1-29
- “Simulink External Mode Interface” on page 1-31
- “Target Computer Command-Line Interface” on page 1-32
- “Web Browser Interface” on page 1-33

**12 Prepare regression and stress tests.**

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

## Hardware in the Loop

Hardware-in-the-loop (HIL) simulation builds on the test harness and simulation results of rapid prototyping to verify a physical prototype of the product. HIL simulation is especially valuable to:

- Substitute for unavailable parts of the system.
- Test the system for safety and performance.
- Minimize expensive downtime for the rest of the system.
- Test operation and failure conditions that are difficult to replicate.

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

- **Model the plant.** — Use Simulink and xPC Target to model the plant for testing the physical prototype.
- **Execute a graphical model.** — Add blocks to a Simulink user interface model with xPC Target To blocks. See “Execution Using Graphical User Interface Models” in the *xPC Target User’s Guide* for details.
- **Write a regression test harness.** — Extend the MATLAB regression tests written for rapid prototyping to cover functionality over the full parameter range.
- **Write a graphical test harness.** — Use one of the APIs (.NET, C, COM) to write a graphical test harness suitable for probing application behavior using xPC Target Embedded Option. As part of this effort, you can use MATLAB Coder to translate MATLAB regression test scripts into C for integration into the test environment.
- **Connect to a physical prototype.**—Modify the system model to replace the xPC Target design with an I/O board connected to the physical prototype.
- **Program a physical prototype.** — Use HDL Coder to generate FPGA code to program the physical prototype.
- **Configure an embedded application.** — Configure the target application to run using xPC Target Embedded Option.

- **Execute the application and display the results.** — Use the test harness to execute the embedded target application, display and log the results, and tune model parameters.

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

# Tutorial and Examples

---

This topic explains the basic functions of the xPC Target product with a simple Simulink model without I/O blocks. You can try these procedures whether or not you have I/O hardware on your target computer. Once you are familiar with the setup, build, and target execution process, you can try some of the more advanced xPC Target demos.

- “Creating a Simple Simulink Model” on page 4-2
- “Entering Parameters for the Scope Block” on page 4-7
- “Adding a Simulink Outport Block” on page 4-11
- “Entering Parameters for the Outport Block” on page 4-15
- “Adding an xPC Target Scope Block” on page 4-20
- “Entering Parameters for an xPC Target Scope Block” on page 4-25
- “Simulating in Non-Real Time Using Simulink” on page 4-41
- “Simulating in Non-Real Time Using MATLAB Language” on page 4-43
- “Booting Target Hardware” on page 4-46
- “Entering Simulation Parameters” on page 4-48
- “Building and Downloading Target Application” on page 4-54
- “Executing in Real Time Using xPC Target Explorer” on page 4-56
- “Executing in Real Time Using Simulink External Mode” on page 4-67
- “Executing in Real Time Using MATLAB Commands” on page 4-69
- “Selected Examples” on page 4-71

## 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.

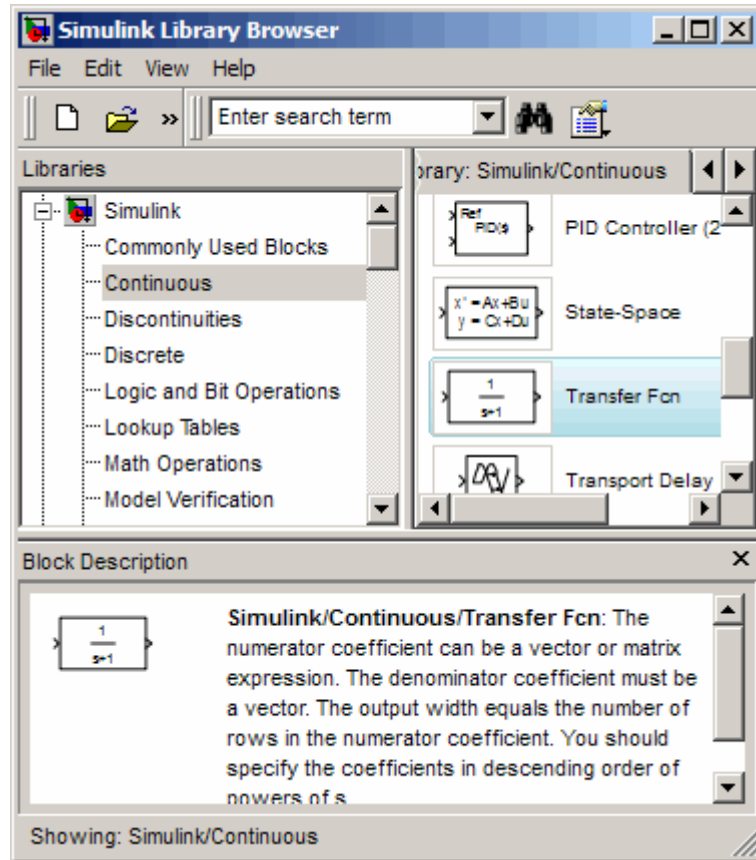
- 2** From the **File** menu, point to **New**, and then click **Model**.

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 computer GUI scope graph after you build and download the model to the target computer. By default, if you do not enter a name for the signal in this dialog box, the scope graph displays the signal identifier rather than a name.

---

- 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

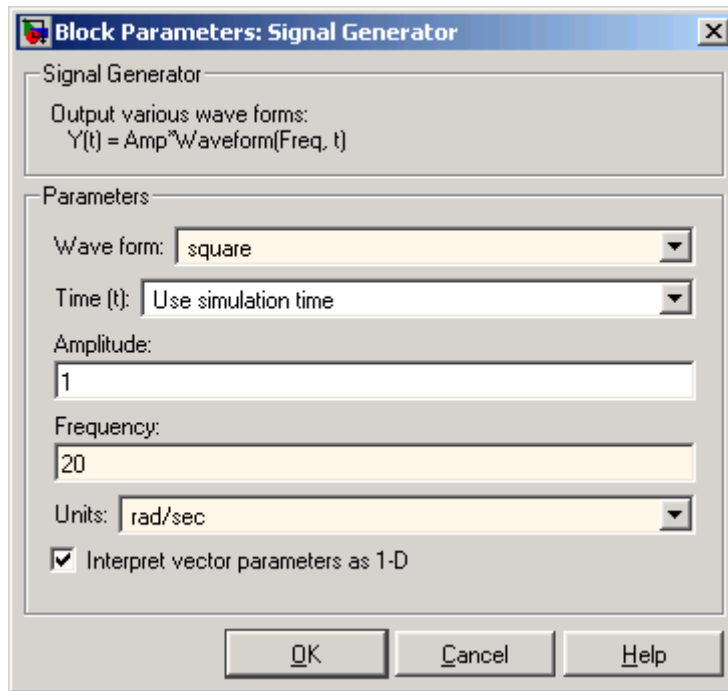
1

In the **Frequency** text box, enter

20

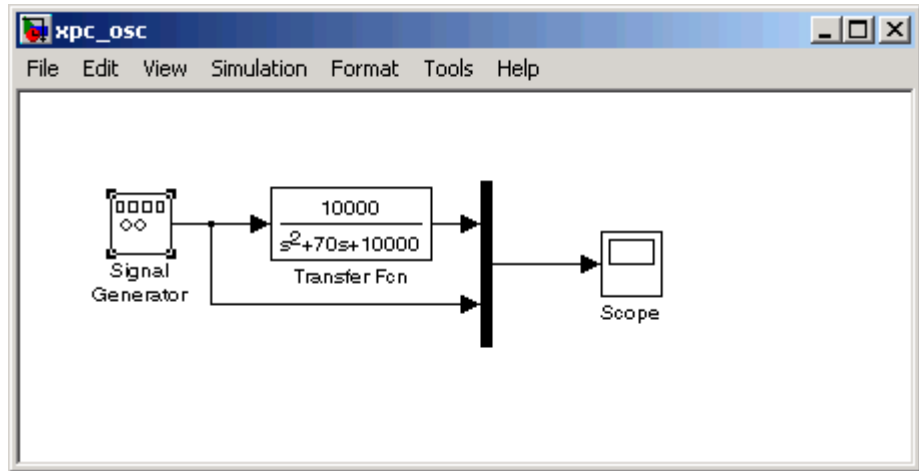
From the **Units** list, select **rad/sec**.

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 4-20. 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.



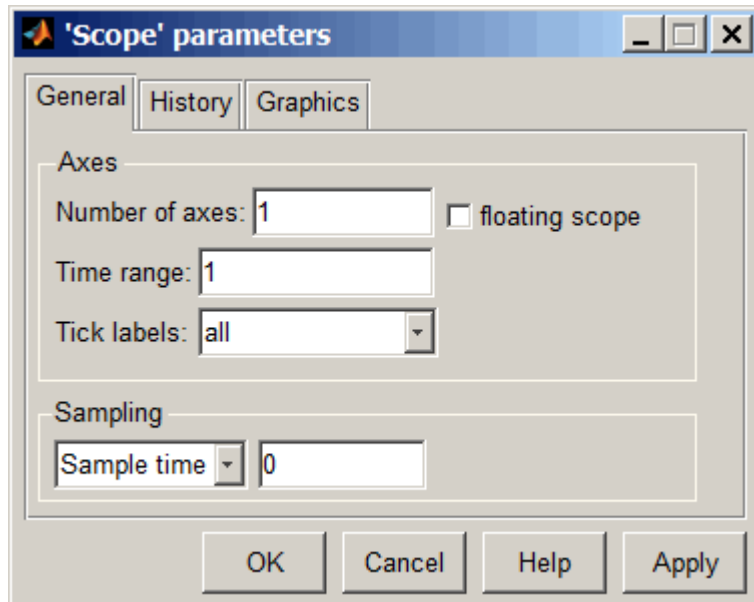
A Scope parameters dialog box opens.

- 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.

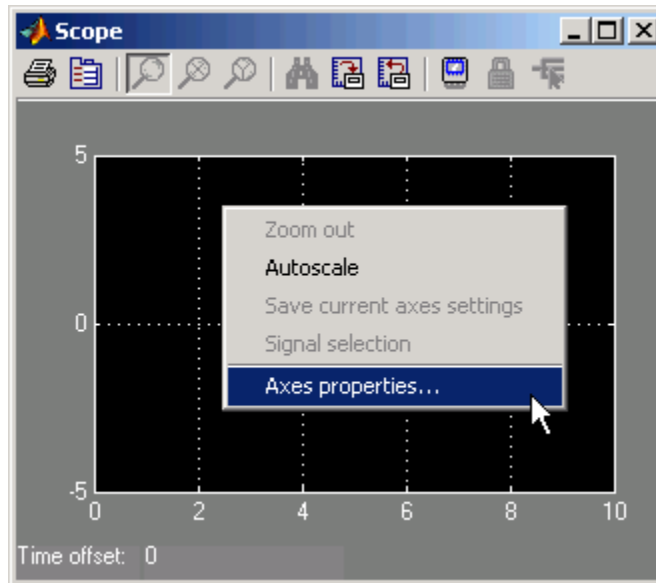
Your Scope parameters dialog box will look similar to the figure shown below.



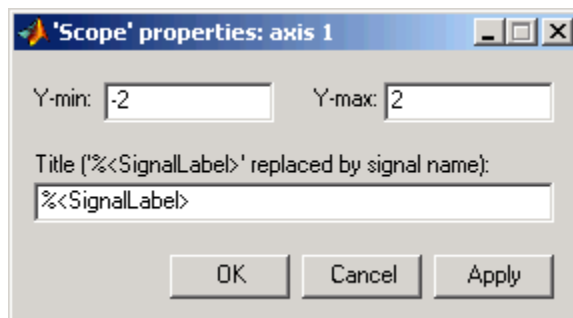
4 Do one of the following:

- 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 Outport Block

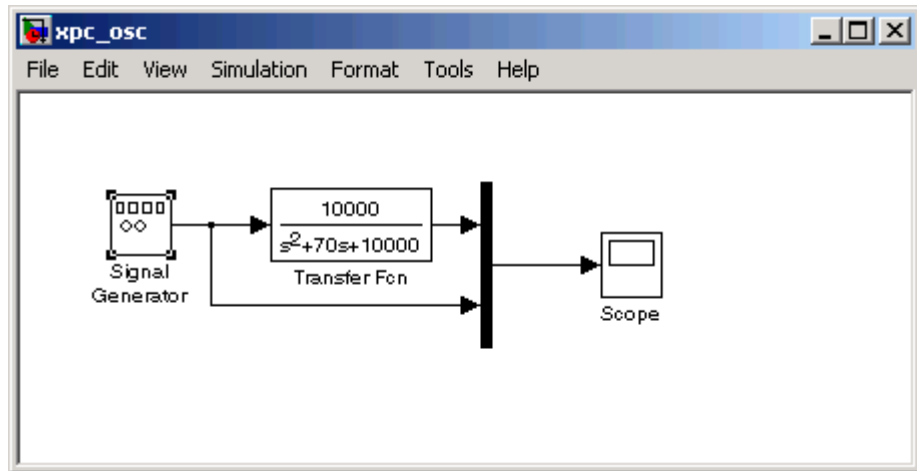
If you want to log signal data to the MATLAB workspace for analysis and later save that data to a disk, you need to add a Simulink Outport block and activate logging from the Configuration Parameters dialog box.

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 4-2.

- 1 In the MATLAB window, type

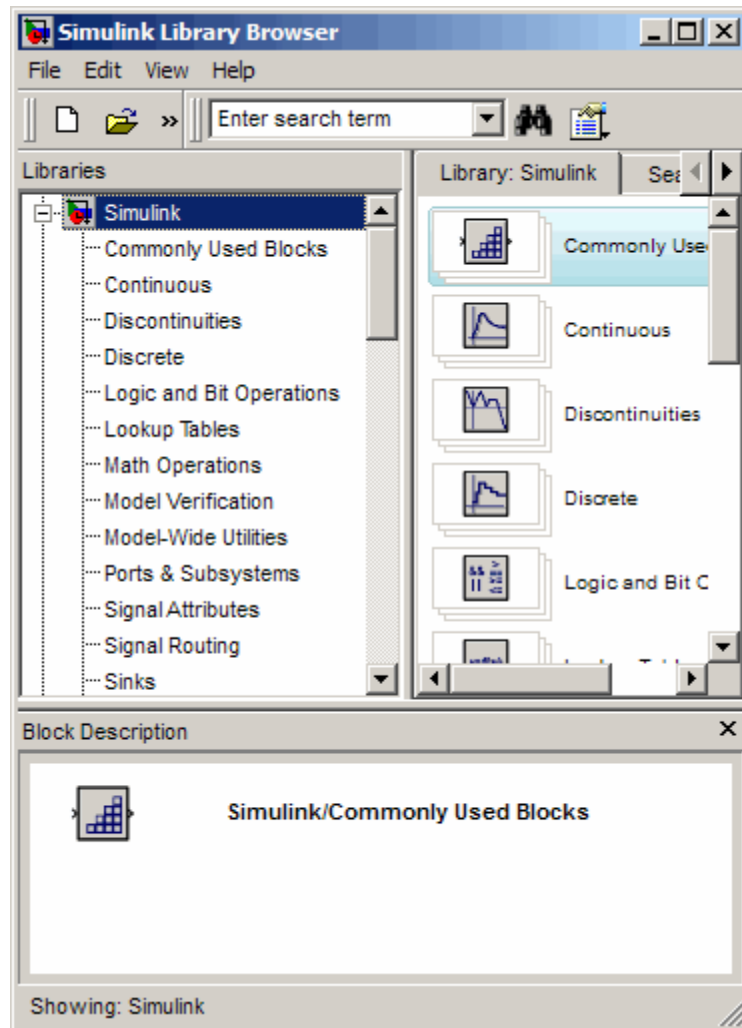
```
my_xpc_osc
```

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



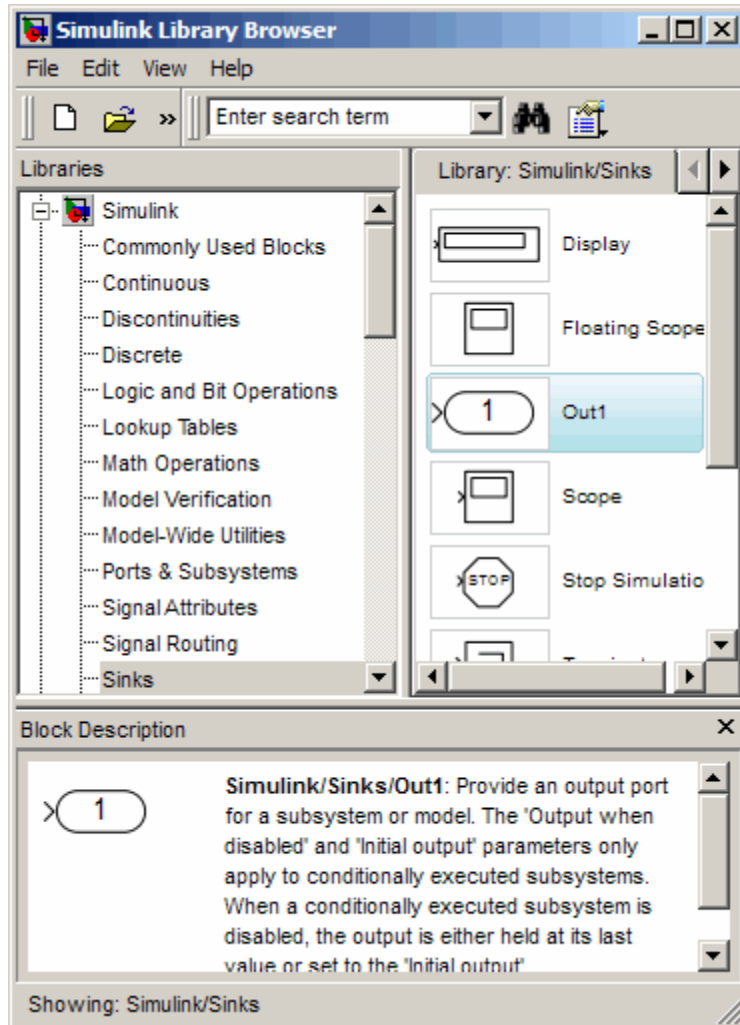
- 2 In the Simulink window, from the **View** menu, click **Library Browser**.

The Simulink Library Browser window opens.



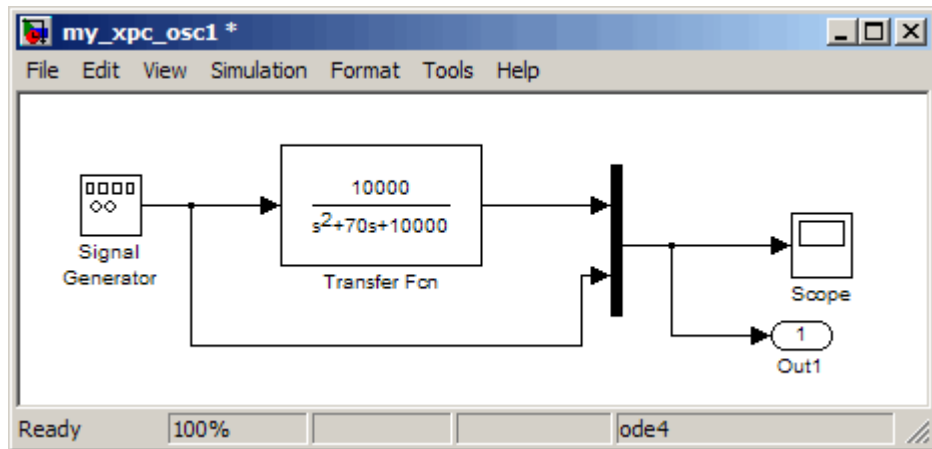
3 In the left pane, double-click **Simulink**, and then click **Sinks**.

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 Outport block. See “Entering Parameters for the Outport Block” on page 4-15.*

## Entering Parameters for the Output Block

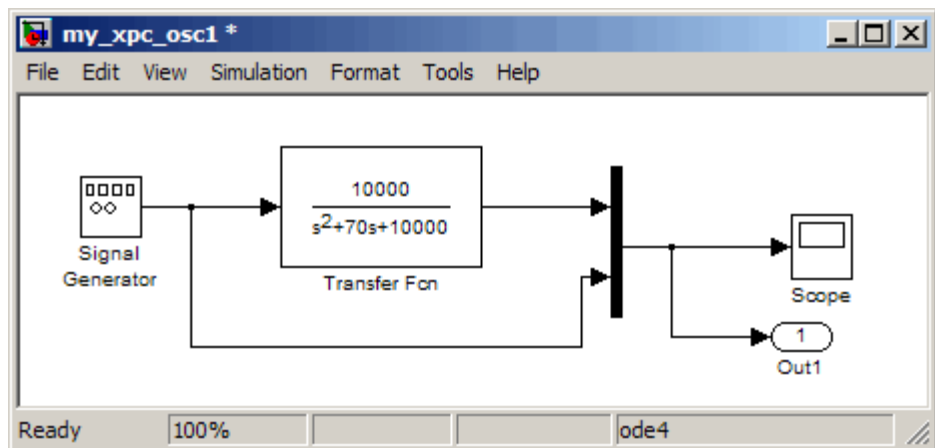
During a simulation, Simulink saves signal data to MATLAB variables using Output blocks. The default MATLAB variables are `tout`, `xout`, and `yout`. While running a real-time application, The xPC Target interface uses these same variables to pass signal data to target object parameters. A target object 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 4-11.

- 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.

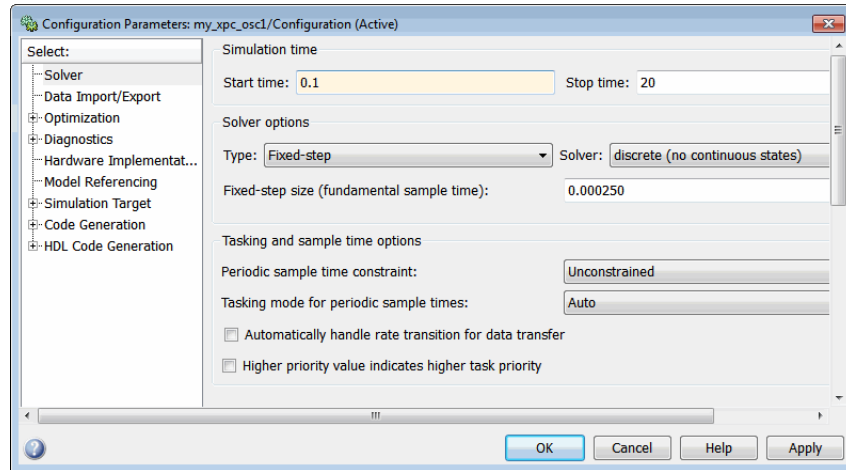
The **Solver** pane dialog changes.

- 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 computer, 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.



- 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.

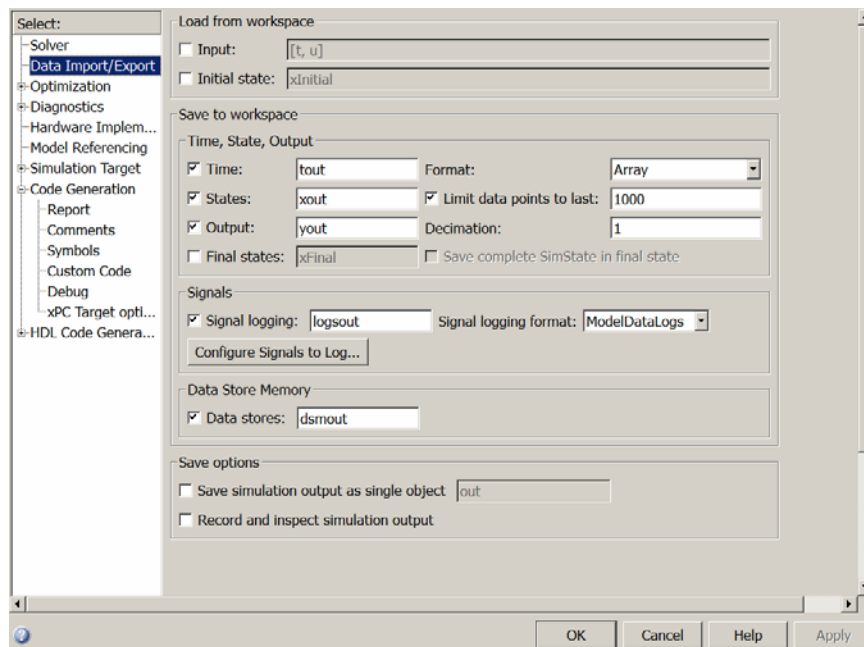
- 9 In the **Save to workspace** section of this pane, select the **Time**, **States**, and **Output** check boxes.

---

**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 Output 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 4-20.*

## 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 computer 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.

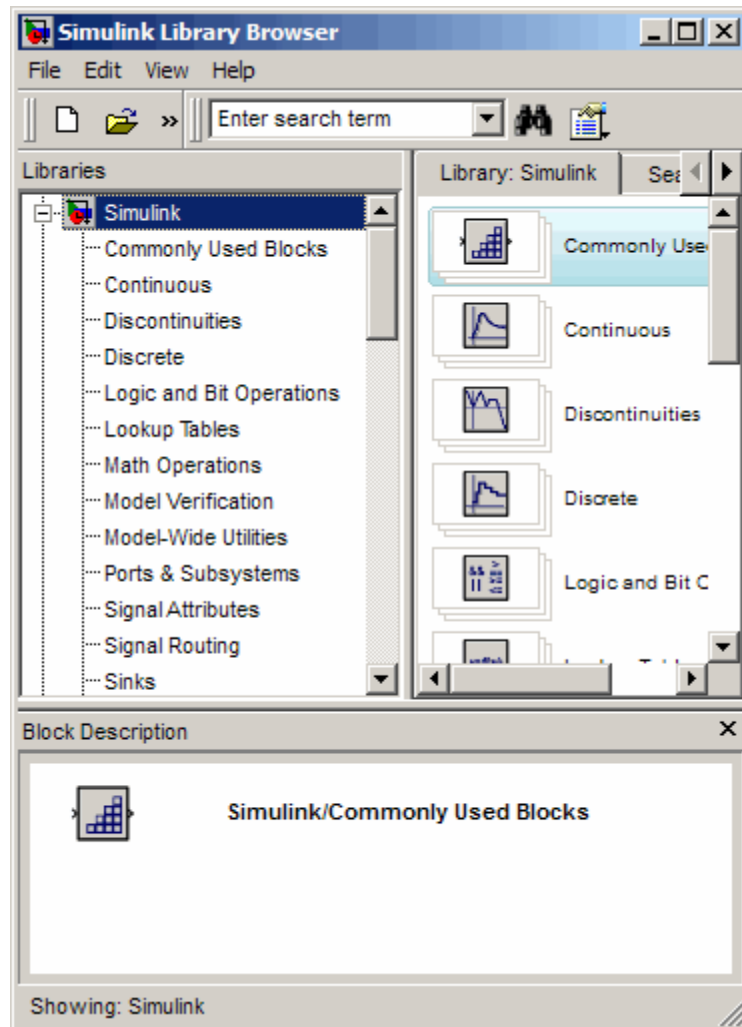
- 1 In the MATLAB window, type

```
my_xpc_osc1
```

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

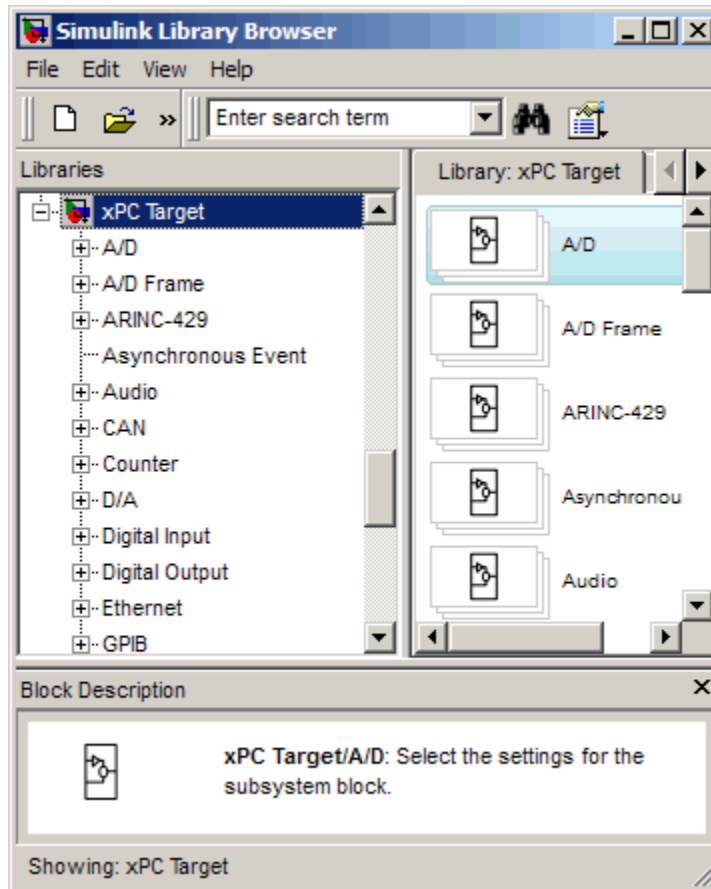
- 2 In the Simulink window, from the **View** menu, click **Library Browser**.

The Simulink Library Browser window opens.



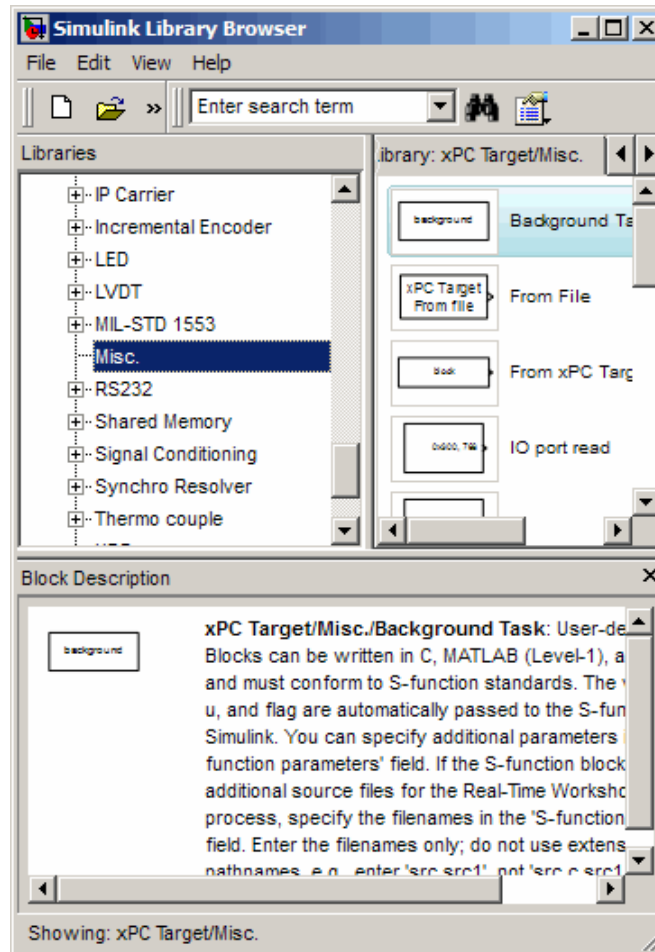
- 3 In the left pane, browse to and double-click **xPC Target**.

A list of I/O functions opens.



#### 4 Click **Misc.**

A list of miscellaneous group blocks opens.

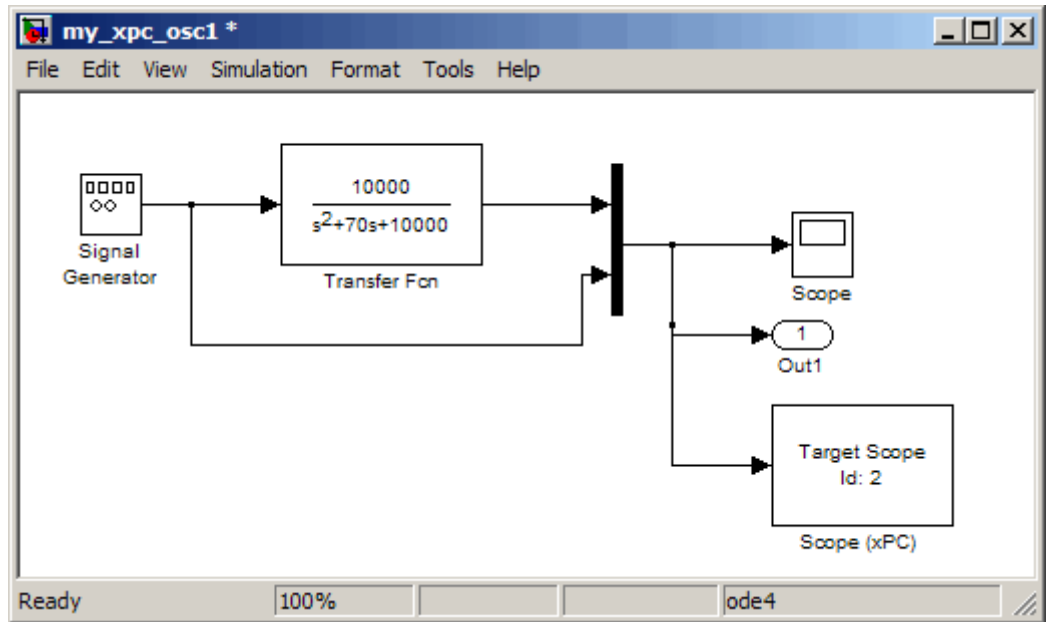


#### 5 Click and drag **Scope (xPC)** to your Simulink block diagram.

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 4-25.*

## 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 computer, the xPC Target kernel automatically creates a default scope on the target. This section describes how you can configure the xPC Target Scope block for additional 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 4-20. To enter the parameters for an xPC Target Scope block to write signal data to a file on the target computer, see “Entering Parameters for a File Scope” on page 4-35.

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:

In this section...
“Entering Parameters for a Target Scope” on page 4-25
“Entering Parameters for a Host Scope” on page 4-31
“Entering Parameters for a File Scope” on page 4-35

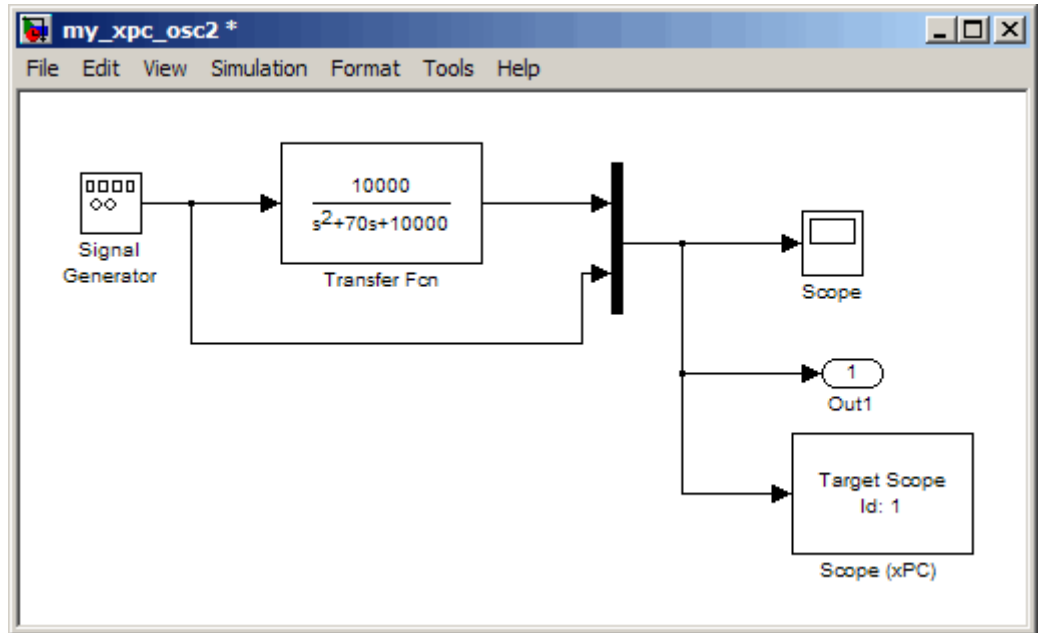
### Entering Parameters for a Target Scope

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

- 1 In the MATLAB window, type

```
my_xpc_osc2
```

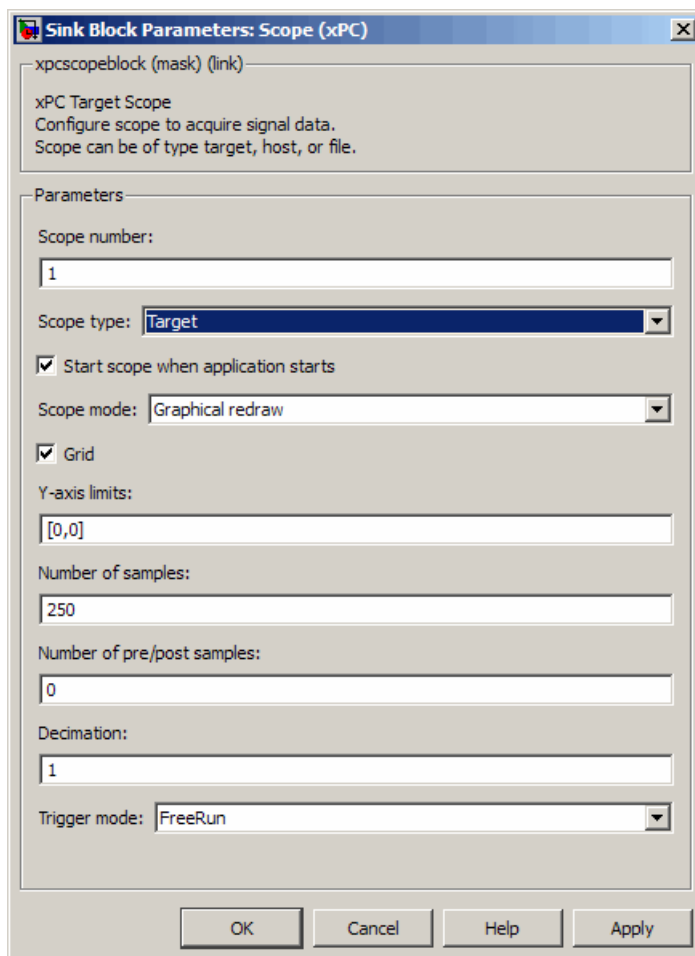
The Simulink block diagram opens for the model `my_xpc_osc2.mdl`.





- 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 in nonreal time. See “Simulating in Non-Real Time Using Simulink” on page 4-41.*

---

**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 `xpctarget.xpc.getscope`. If you use the target object method `xpctarget.xpc.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.

---

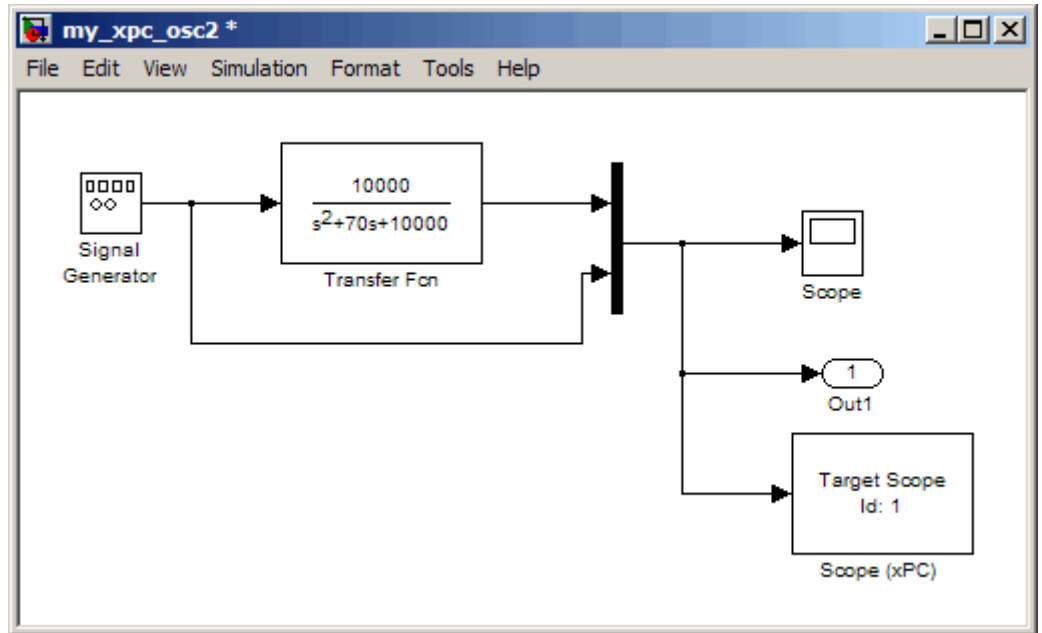
## Entering Parameters for a Host Scope

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.

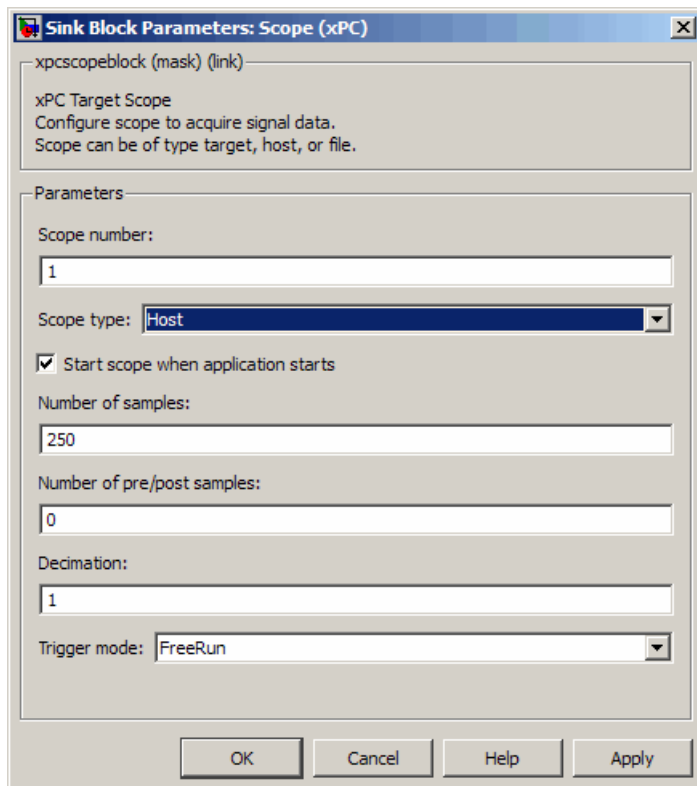
By default, the target scope dialog is displayed.

- 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 in nonreal time. See “Simulating in Non-Real Time Using Simulink” on page 4-41.*



---

**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 `xpctarget.xpc.getscope`. If you use the target object method `xpctarget.xpc.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 computer 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 computer.

---

**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 computers as standalone 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 computer 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 FTP and File System Objects” 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).

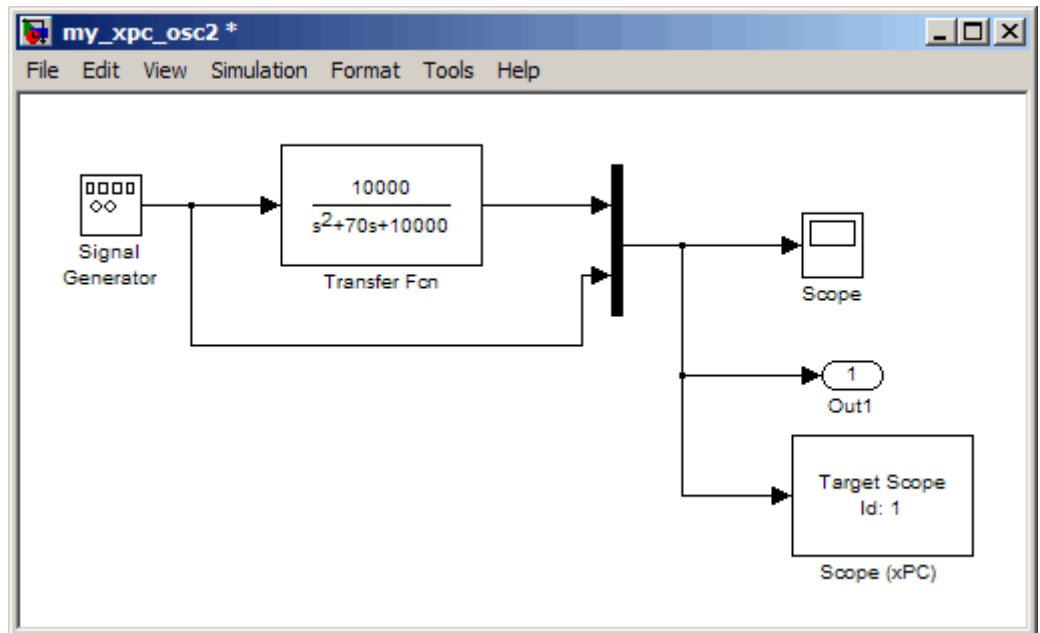
To add an xPC Target Scope block, see “Adding an xPC Target Scope Block” on page 4-20.

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.

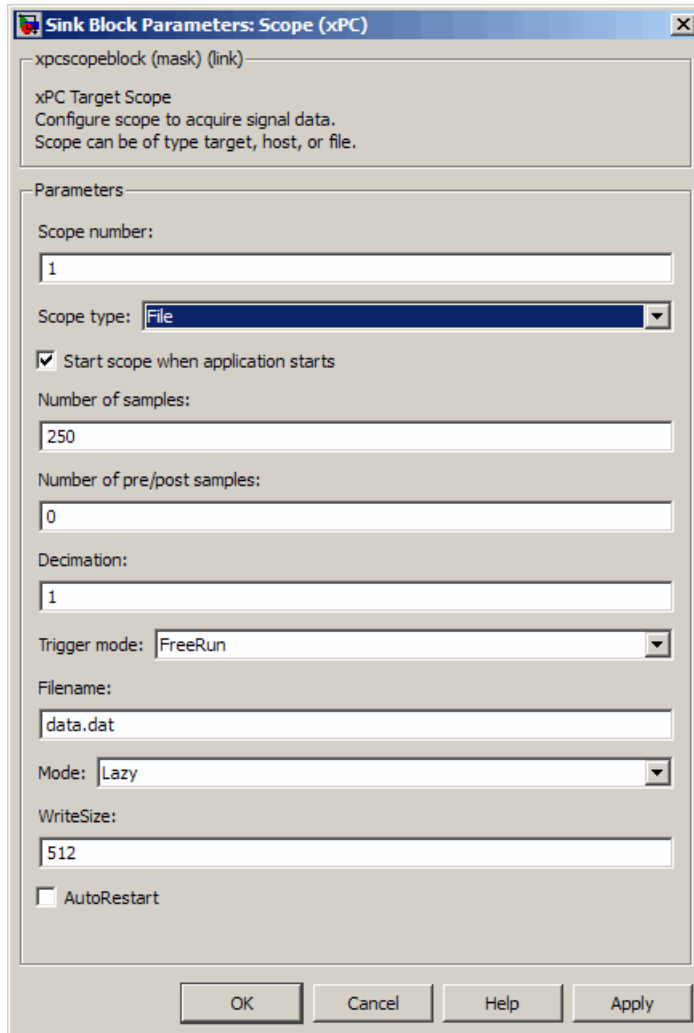
By default, the target scope dialog is displayed.

- 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 computer 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 knows the actual file size after each write. 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, the typical disk sector size. Using a block size that is the same as the disk sector size improves 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.

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 computer after you run the target application. For further details on working with these files, see “Logging Signal Data with FTP and File System Objects” in the *xPC Target User’s Guide*.

*Your next task is to simulate the model in nonreal time. See “Simulating in Non-Real Time Using Simulink” on page 4-41.*

## Simulating in Non-Real Time Using Simulink

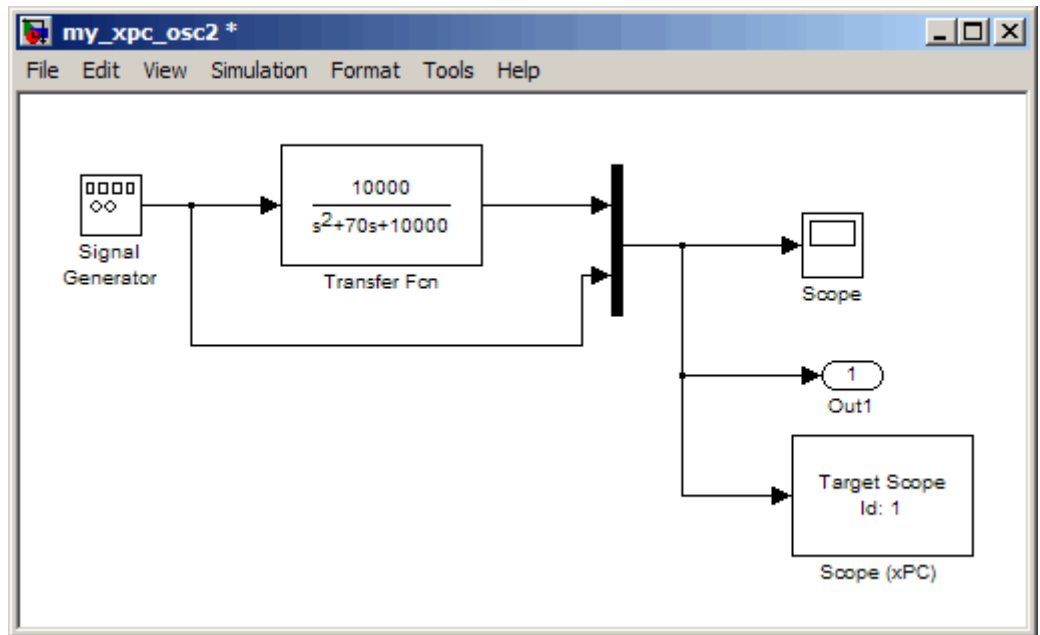
You use Simulink in normal mode to observe the behavior of your model in nonreal time. After you load your Simulink model, you can run a simulation. This procedure uses the Simulink model `my_xpc_osc2.mdl` as an example and assumes you have already loaded that model. To create this model, see “Creating a Simple Simulink Model” on page 4-2.

To run your target application in real time, start with “Booting Target Hardware” on page 4-46.

- 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 execute the simulation in real time. Start the procedure with “Booting Target Hardware” on page 4-46.*



## Simulating in Non-Real Time Using MATLAB Language

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 4-2.

- 1 In the MATLAB window, type

```
sim('my_xpc_osc2')
```

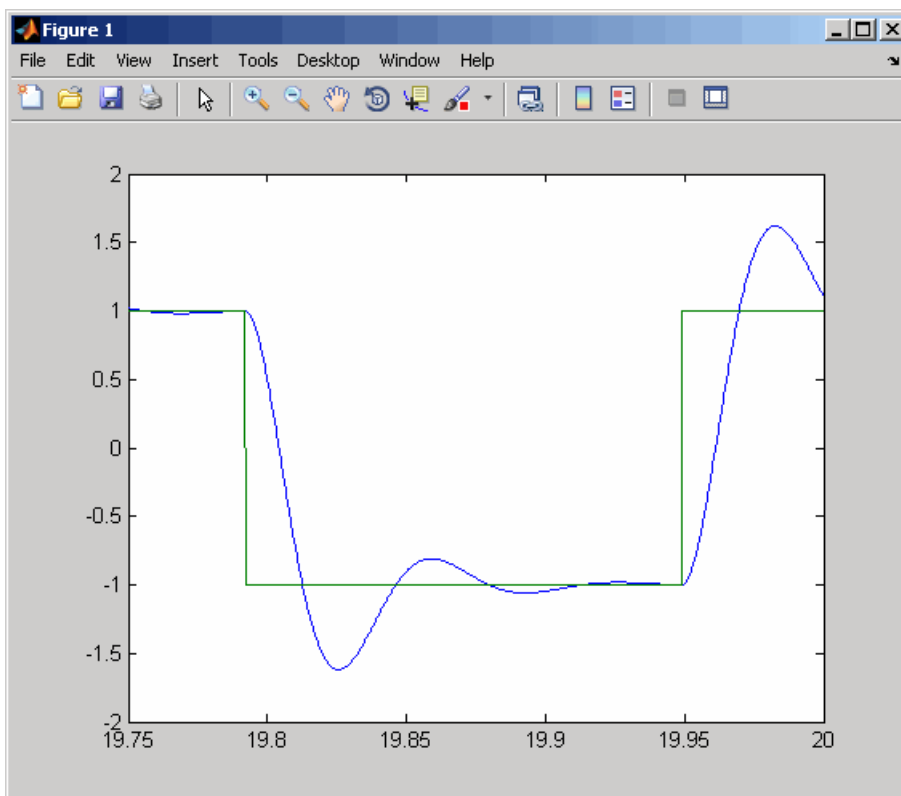
Simulink runs a simulation in normal mode through to completion. You cannot manually stop the simulation. See the online Simulink documentation for further information on using the `sim` command.

- 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 Output blocks. To add an Output block, see “Adding a Simulink Output Block” on page 4-11 and “Entering Parameters for the Output Block” on page 4-15.

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 computer memory and can be retrieved through the target object properties `tg.TimeLog`, `tg.StateLog`, and `tg.OutLog`. However, in the Configuration Parameters dialog box, you must still select the **Time**, **States**, and **Output** check boxes for data to be logged to the target object properties.

---

*Your next task is to execute the simulation in real time. Start the procedure with “Booting Target Hardware” on page 4-46.*

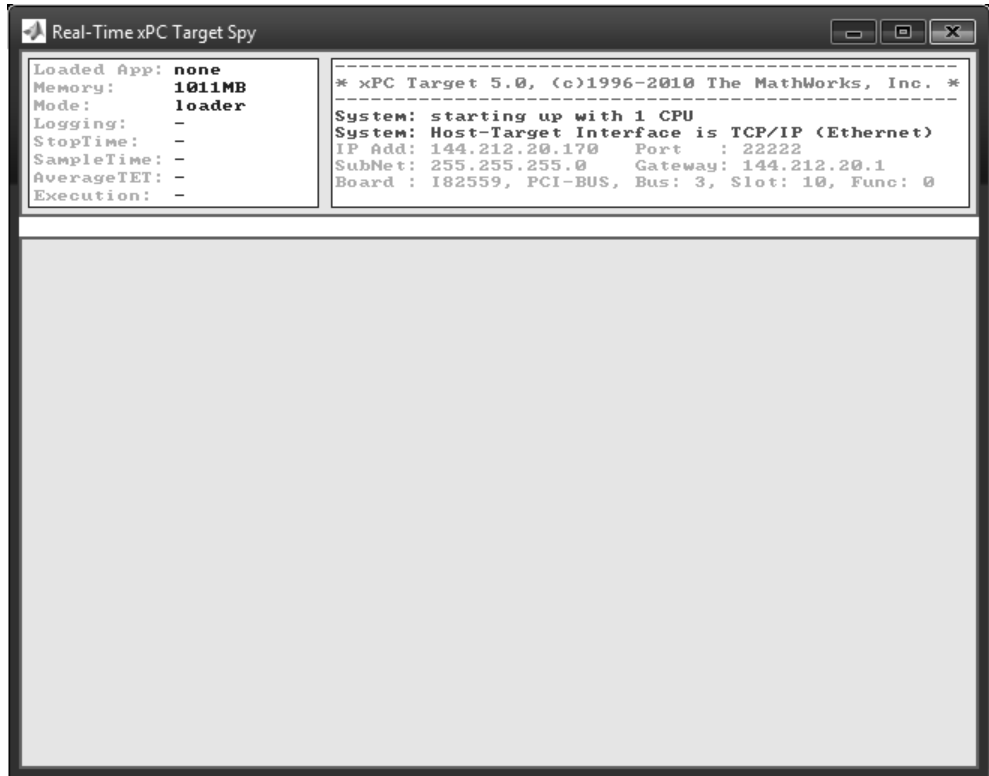
# Booting Target Hardware

Booting the target computer loads and starts the xPC Target kernel on the target computer. The loader then waits for the xPC Target software to download your target application from the host computer.

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 computer. You need to boot the target computer before building your target application because the build process automatically downloads your target application to the target computer. 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 computer disk drive.
- 2 Turn on the target computer or press the **Reset** button.

The target computer 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.

---

**Note** The xPC Target kernel can use only 2 GB of memory.

---

*Your next task is to enter the simulation and real-time run parameters for Simulink Coder. See “Entering Simulation Parameters” on page 4-48.*

## Entering Simulation Parameters

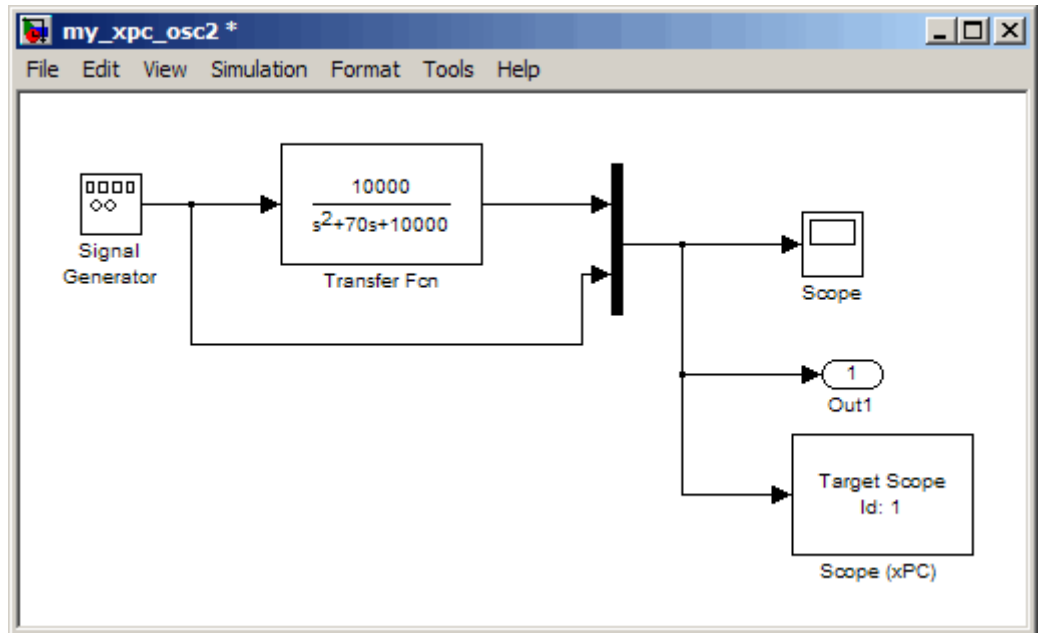
You enter the simulation and real-time run parameters in the Configuration Parameters dialog box. These parameters give information to Simulink Coder on how to build the target application from your Simulink model.

After you open a Simulink model and boot the target computer, you can enter the simulation parameters. This procedure uses the Simulink model `my_xpc_osc2.mdl` as an example and assumes you have already opened that model (see “Creating a Simple Simulink Model” on page 4-2).

- 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**.

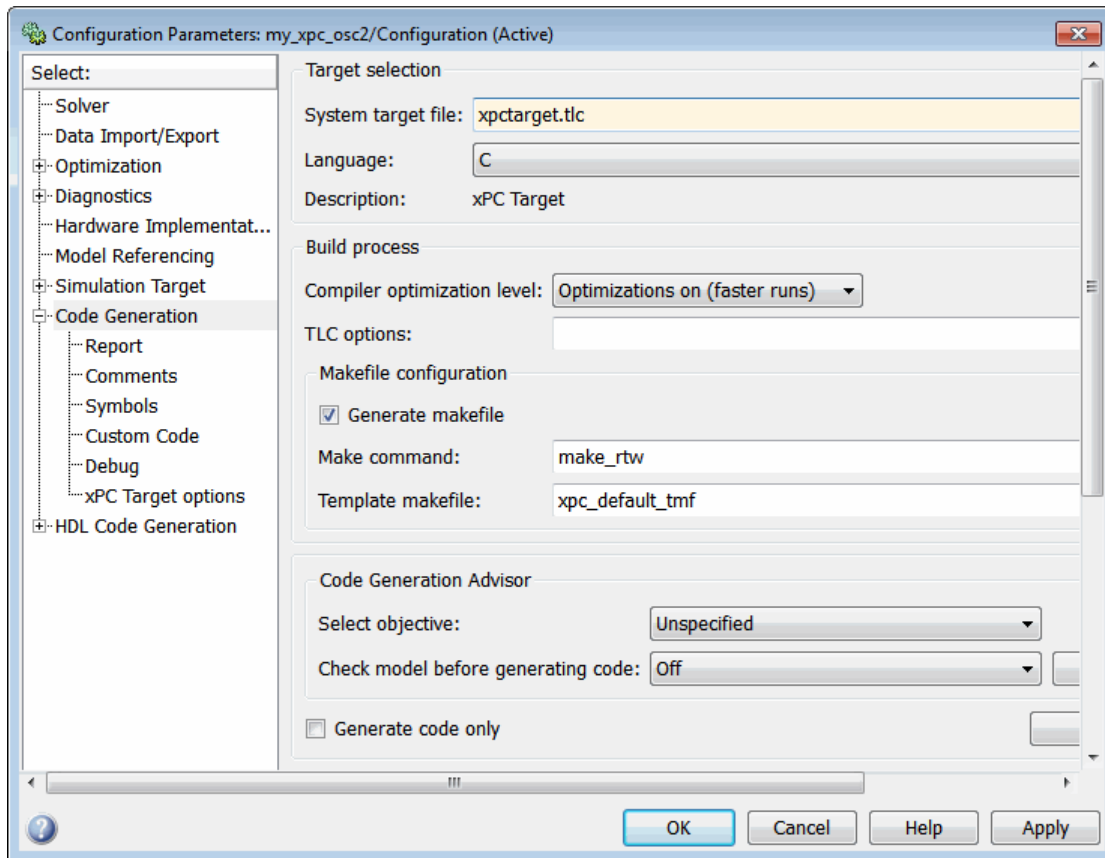
The Configuration Parameters dialog box is displayed for the model.

- 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** If you select `xpctargetert.tlc` without the Embedded Coder installed, the build fails.

---

- 5 In the left pane, click the **xPC Target options** node.

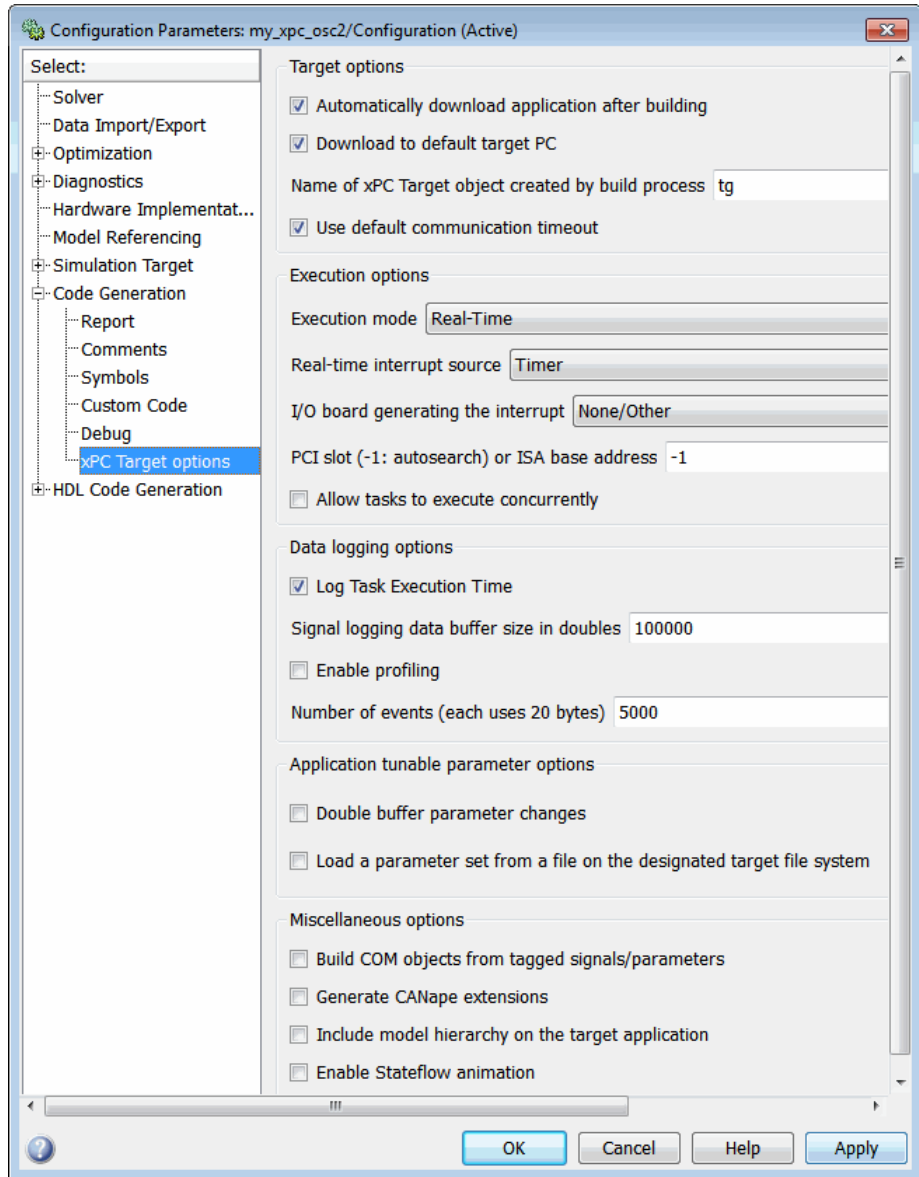
The associated pane is displayed. These are model-level configuration parameters that you can set for your model. See “Configuration Parameters” in the *xPC Target User’s Guide* for a description of the options on this node.

- 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`.

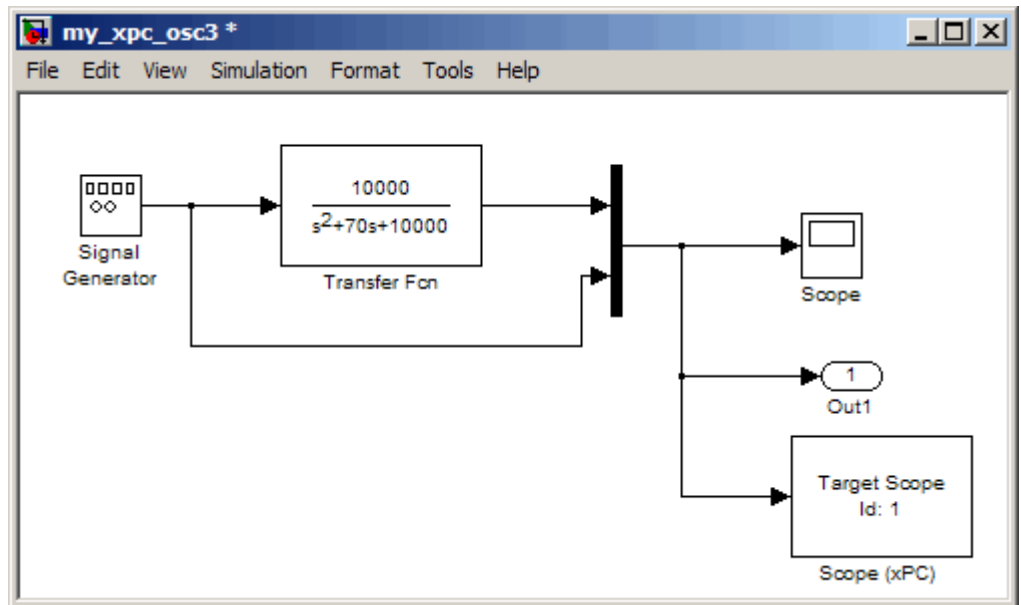
The task execution time is the time in seconds to complete calculations for the model equations and post outputs during each sample interval. If you do not select this box, your average TET value appears as Not a Number (NaN).

- 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 and download) the target application. See “Building and Downloading Target Application” on page 4-54.*

## Building and Downloading Target Application

You use the xPC Target build process to generate C code, compile, link, and download your target application to the target computer.

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 “Building and Downloading Target Application” on page 4-54. By default, the build procedure downloads the target application to the default target computer, as designated in the xPC Target Explorer. See “xPC Target Options Configuration Parameter” for further details on setting the target computer 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 completing a build process without detecting an error:

```
### Starting xPC Target build procedure for model:  
my_xpc_osc3  
. . .  
### Successful completion of xPC Target build procedure for  
model: my_xpc_osc3
```

The target computer displays the following information:

Loaded App: <b>xpc_osc3</b>	Scope: 1, created, type is target
Memory: <b>123MB</b>	Scope: 1, signal 0 added
Mode: <b>RT, single</b>	Scope: 1, signal 1 added
Logging: <b>t x y tet</b>	Scope: 1, NumSamples set to 500
StopTime: <b>20 d</b>	Scope: 1, trigger level set to 0.000000
SampleTime: <b>0.00025</b>	Scope: 1, TriggerScope set to 1
AverageTET: <b>-</b>	Scope: 1, lower y-axis limit set to 0.000000
Execution: <b>stopped</b>	Scope: 1, upper y-axis limit set to 0.000000
	System: <b>initializing application finished</b>

**3** In the MATLAB window, type

```
tg
```

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

If the software detects a error during build and download, see:

- “Model Compilation”
- “Application Download”

---

**Note** If you accidentally download a target application built with a different version of the xPC Target product than the one on the target computer, the following error message will appear on the target computer monitor and the download will fail.

Mismatch between model and kernel versions

To prevent this version mismatch, rebuild target applications with each new xPC Target release.

---

During the build process, the xPC Target software creates a target object that represents the target application running on the target computer. 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.

*Your next task is to run the target application in real time on the target computer. See “Executing in Real Time Using xPC Target Explorer” on page 4-56.*

## Executing in Real Time Using xPC Target Explorer

This procedure assumes you have created an xPC Target boot disk and you have booted the target computer. See “Booting Target Computers from Removable Boot Drives” 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 computer (see “Building and Downloading Target Application” on page 4-54). For a description of how to download a prebuilt target application, see “Downloading and Running Target Applications on a Target PC” on page 4-60.

- 1 In the MATLAB window, type

```
xpcexplr
```

---

**Note** xPC Target Explorer runs only on 32-bit host computers. On 64-bit computers, use the MATLAB command-line interface. For further information, see “Configuring Environment From the MATLAB Command Line”.

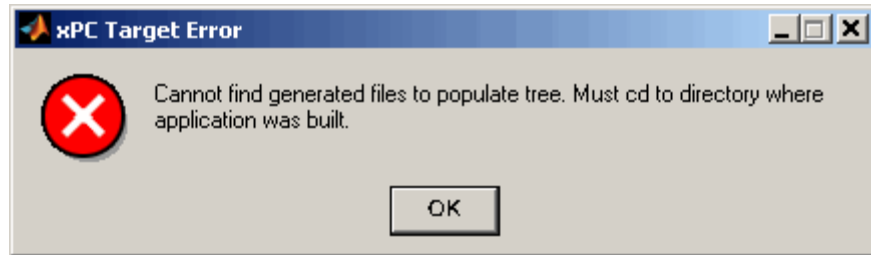
---

The xPC Target Explorer window opens.

- 2 To connect to the target computer, right-click the target computer icon for which you have downloaded the application and select **Connect**.

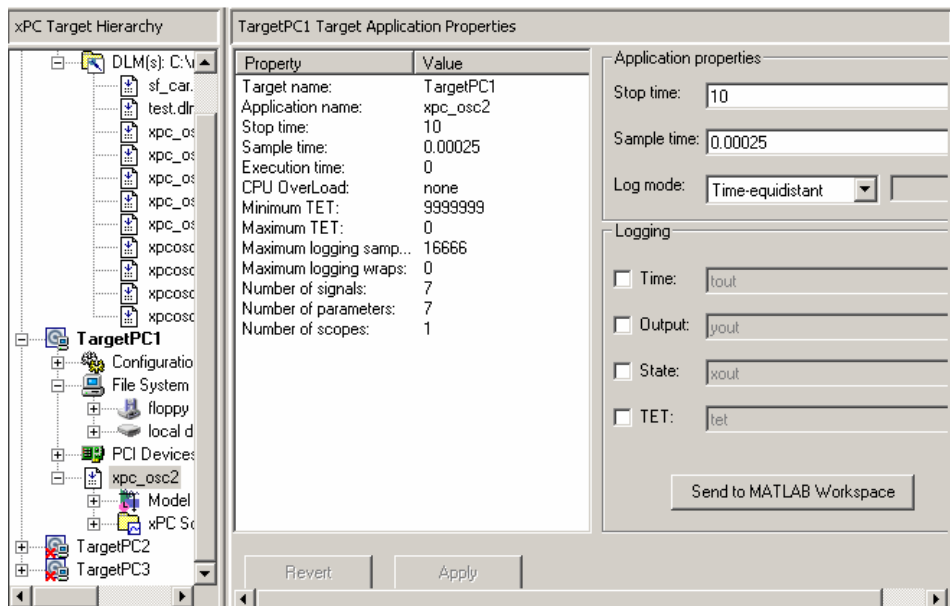
To view the model hierarchy for the downloaded application, one of the following must be true:

- 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 target application folder, a node for the target application appears in the **xPC Target Hierarchy** under the target computer node and displays information about the previously loaded target application.

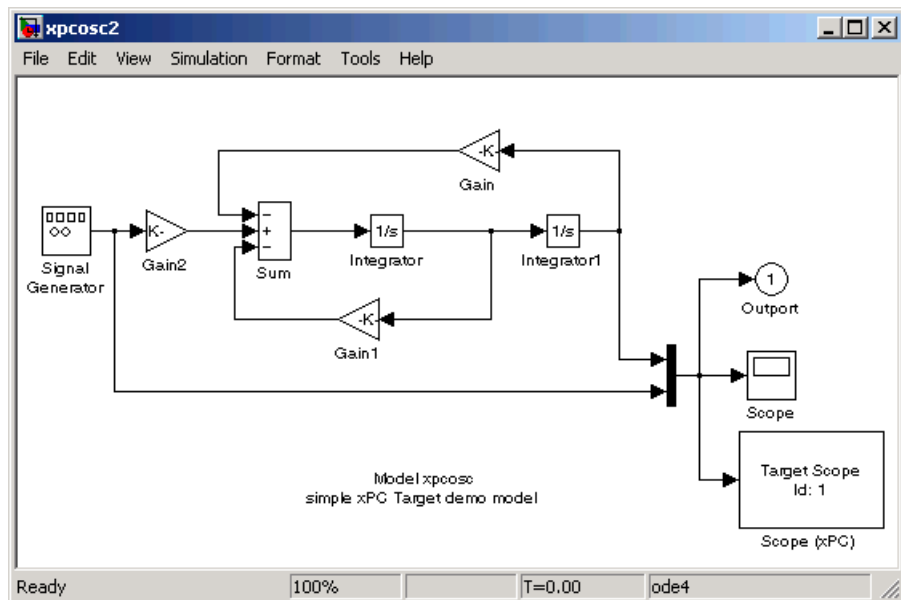


Note, if, in your current folder, you have a prebuilt target application that you want to download to the target computer, left-click and drag the desired

target application to the target computer 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**.

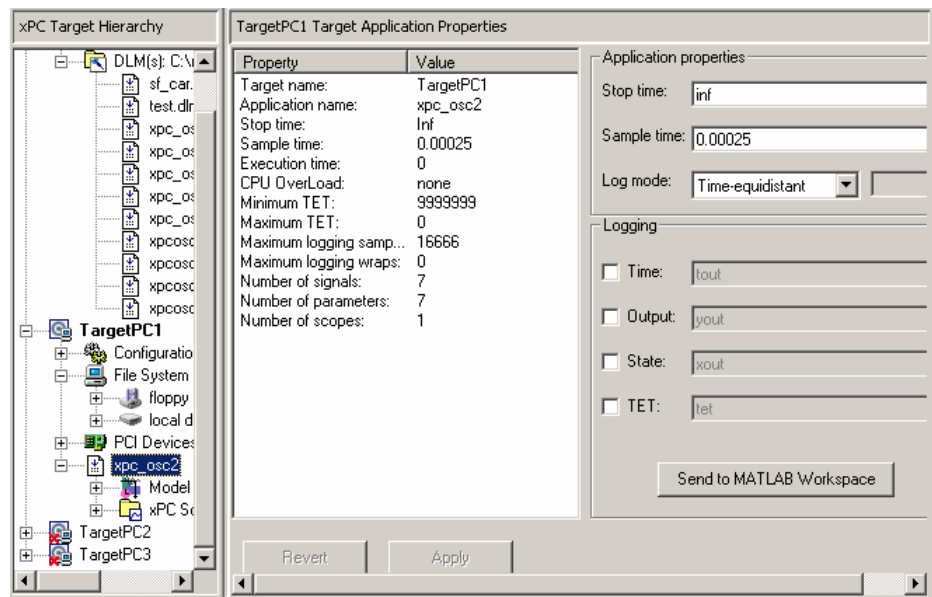
The xPC Target software recompiles, links, and downloads the target application to the target computer.


- 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 computer, 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 computers 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 computer.
- You have installed xPC Target software and booted the target computer to which you want to download a target application.
- You have a physical connection between the xPC Target Explorer host machine and the target computer 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.)

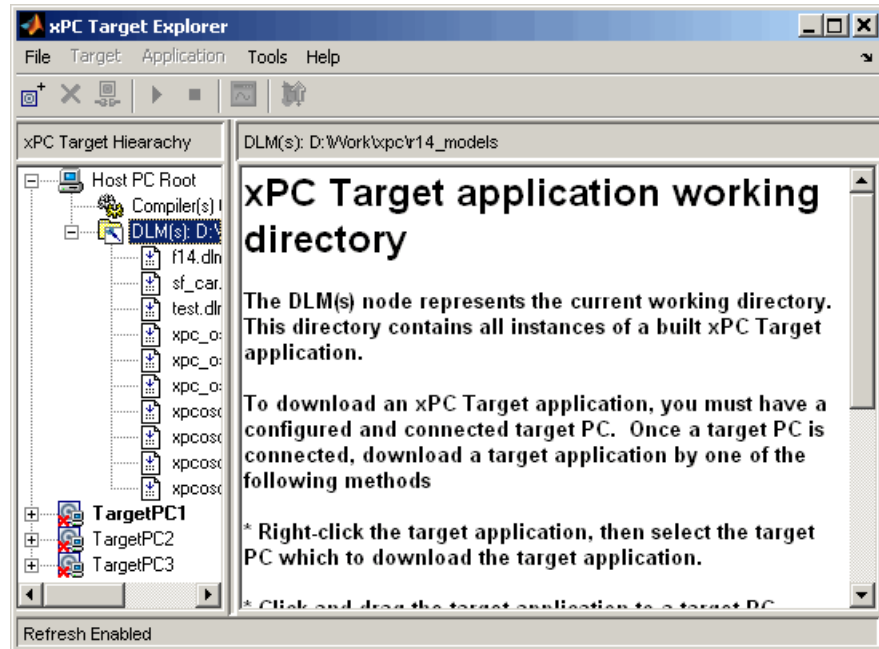
**2** Select **Change Host PC Current Working Directory**.

A browser is displayed.

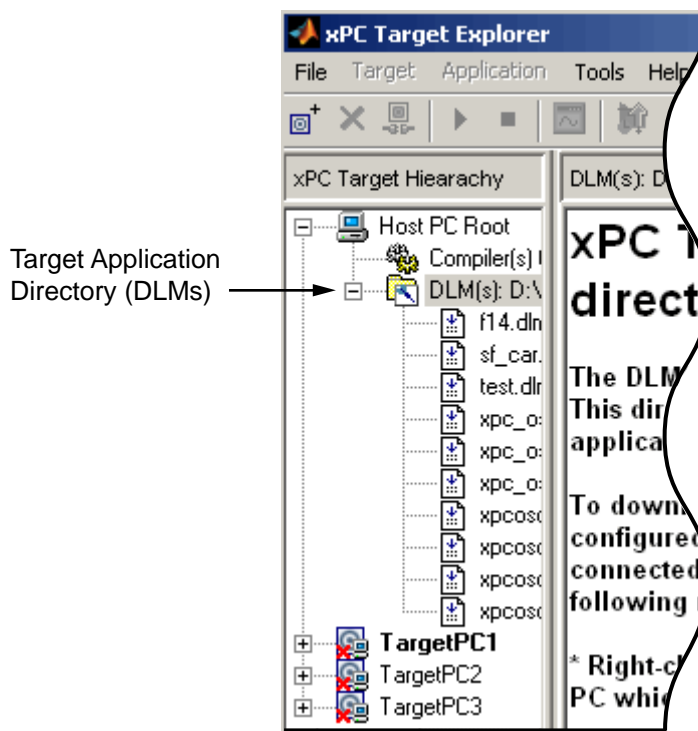
**3** Browse to the folder that contains the prebuilt target applications you want.

**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 computer.



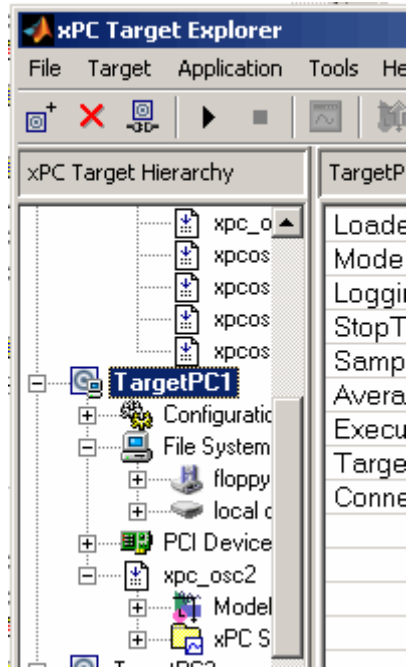
6 Right-click a target computer that you booted with xPC Target software, for example, TargetPC1.

7 Select **Connect**.

The target computer icon changes and the red X is removed . The target computer information changes to reflect file system and PCI device information.

8 Left-click and drag the desired target application to the target computer to which you want to download the target application.

xPC Target Explorer downloads the target application to the target computer. A node for the target application appears in the **xPC Target Hierarchy** under the target computer node.



Alternatively, you can now drag a prebuilt target application, DLM, to a target computer icon. If a connection does not already exist, xPC Target Explorer optionally creates a connection to that target computer.

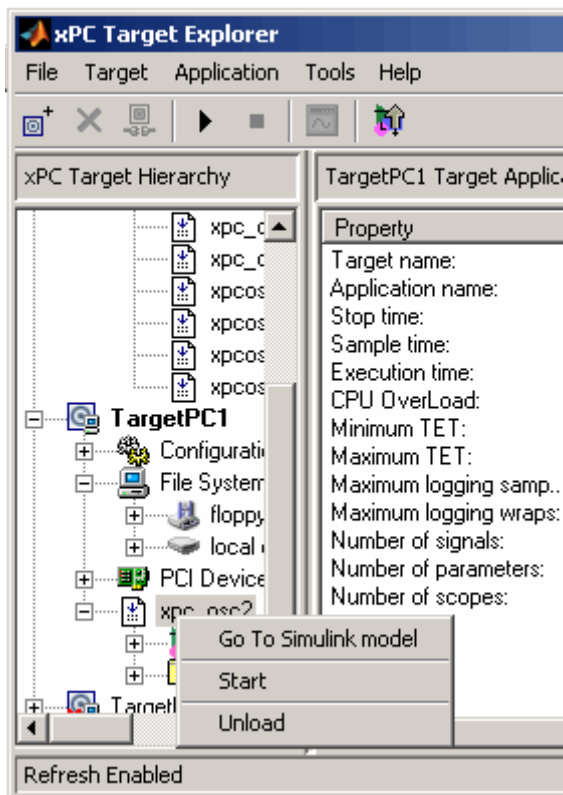
---

**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 computer.

- 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 1-26 for limitations on changing sample times.

- 3 Change the **Stop time** parameter to 9999. Click **Apply**. For example,

TargetPC1 Target Application Properties

Property	Value
Target name:	TargetPC1
Application name:	xpcosc
Stop time:	9999
Sample time:	0.00025
Execution time:	0
CPU OverLoad:	none
Minimum TET:	9999999
Maximum TET:	0
Maximum logging samp...	16666
Maximum logging wraps:	0
Number of signals:	7
Number of parameters:	7
Number of scopes:	0

Application properties

Stop time:

Sample time:

Log mode:

Logging

Time:

Output:

State:

TET:



## Executing in Real Time Using 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 computer.

---

**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 computer, you can run the target application. This procedure uses the Simulink model `my_xpc_osc2.mdl` as an example (see “Building and Downloading Target Application” on page 4-54). It assumes that you have specified the required target computer environment on the xPC Target options node of the Simulink Coder parameters dialog. In particular, you must specify the target computer to which you want to connect. See the **Use the default target PC** check box description in “xPC Target Options Configuration Parameter”.

- 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 from the host computer to your 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.

## Executing in Real Time Using 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 computer, 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 target computer.

- 1 In the MATLAB window, type any of

```
+tg or tg.start or start(tg)
```

The target application starts running on the target computer. 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 computer screen, the **Execution** line changes from stopped to running and the **AverageTET** line is periodically updated with a new value.

Loaded App: <code>xpc_osc3</code>	Scope: 1, lower y-axis limit set to <code>0.000000</code>
Memory: <code>123MB</code>	Scope: 1, upper y-axis limit set to <code>0.000000</code>
Mode: <code>RT, single</code>	System: initializing application finished
Logging: <code>t x y tet</code>	System: execution started (sample time: <code>0.000250</code> )
StopTime: <code>10000 d</code>	System: execution stopped at <code>10.191750</code>
SampleTime: <code>0.00025</code>	Scope: 1, set to state 'Interrupted'
AverageTET: <code>6.859e-006</code>	minimal TET: <code>0.000006</code> at time <code>0.006250</code>
Execution: <code>10.00 s</code>	maximal TET: <code>0.000017</code> at time <code>0.097750</code>
	System: execution started (sample time: <code>0.000250</code> )

- 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 1-26 for further limitations on changing sample times.)

## Selected Examples

In this section...
“Applications and Driver Examples” on page 4-71
“To Locate or Edit an Example Script” on page 4-73

### Applications and Driver Examples

The xPC Target examples are used to demonstrate the features of the xPC Target product. They are also scripts that you can view to understand how to write your own scripts for creating and testing target applications.

Examples fall into two categories, general applications and drivers. The following lists the general application examples.

Description	Filename
Real-time parameter tuning and data logging	Parameter Tuning and Data Logging
Freerun display mode of a host	Signal Tracing With a Host Scope in Freerun Mode
A software triggered host scope	Signal Tracing Using Software Triggering
A signal triggered host scope	Signal Tracing Using Signal Triggering
A scope triggered host scope	Signal Tracing Using Scope Triggering
Signal tracing with a target scope	Signal Tracing With a Target Scope
Pre- and posttriggering of an host scope	Pre- and Post-Triggering of a Host Scope
Time- and value-equidistant data logging	Time- and Value-Equidistant Data Logging

Description	Filename
Logging signal data to a file on the target computer	Data Logging With a File Scope
Frame signal processing  <hr/> <b>Note</b> This example requires DSP System Toolbox™ software. <hr/>	Frame Signal Processing
xPC Target software as a real-time spectrum analyzer	Spectrum Analyzer
Creating a client application to interface with the target computer	Simple Client Application With the .NET API
Concurrent execution of a model using the xPC Target profiling tool	Concurrent Execution on xPC Target

The driver examples category contains examples for a number of driver applications, including, but not limited to:

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

- Raw Ethernet
- Raw Ethernet
- RS-232
- Shared/reflective memory
- UDP
- Video

---

**Note** Because these examples illustrate the use of driver blocks in an xPC Target environment, you might need specific hardware to run them.

---

You can access xPC Target general application and driver examples through the MATLAB Online Help. In this window, **xPC Target > Demos** to list the available example categories.

## To Locate or Edit an Example Script

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

```
which scfreerundemo
```

The MATLAB interface displays the location of the MATLAB file for the example.

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

**2** Type

```
edit scfreerundemo
```

The MATLAB interface opens the MATLAB example file in a MATLAB editing window.





**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 1-18
- analog input (A/D)
  - driver support 1-19
- analog output (D/A)
  - driver support 1-19
- API
  - custom GUI 1-34
- API for Microsoft .NET framework 1-33
- applications
  - with DOSLoader mode 2-60
  - with Standalone mode 2-68

## B

- before you boot
  - checklist 2-38
- BIOS
  - target PC 1-6
- BIOS settings 2-10
- block parameters
  - scope 4-25
- boot method 2-38
- boot options
  - boot drive 2-51
  - CD 2-39
  - dedicated network 2-46
- bootable partition
  - diskpart 2-56
- booting
  - target computer 4-46
- build process
  - target application 4-54

## C

- C compiler
  - network setup 2-24
  - serial setup 2-33
- CAN field bus

- driver support 1-19
- CD
  - creating for booting 2-39
- CD target boot disk
  - creating 2-43
- code generation options
  - for Simulink Coder 4-48
- COM API 1-34
- command-line interface
  - MATLAB 1-29
  - target PC 1-32
- communication
  - between computers 1-23
  - network 2-20
  - network advantages 1-18
  - serial 2-31
- computer
  - communication 1-23
  - 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 1-17
  - I/O boards 1-19
  - real-world 1-19
- controlling target applications
  - with MATLAB 4-69
  - with Simulink external mode 4-67
- counter timers
  - driver support 1-19
- creating application with DOSLoader mode 2-60
- creating application with Standalone mode 2-68
- creating boot media
  - checklist 2-38
- creating CD target boot disks 2-43
- creating target boot drives 2-54

- creating target objects 4-54
- custom GUI 1-34
  - API 1-34
  - API for Microsoft .NET framework 1-33
  - COM API 1-34

## D

- dedicated network
  - booting within 2-46
- defining scope block parameters 4-25
- desktop PC
  - host computer 1-14
  - target computer 1-15
- directories
  - installed 2-13
  - working 2-13
  - xpc 2-13
  - xpcdemo 2-13
- DOS loader mode
  - embedded option 1-24
- DOSLoader mode 2-57
  - creating target application 2-60
  - using xpcbootdisk 2-60
- downloading target application 4-54

## E

- embedded option
  - DOS loader mode 1-24
  - DOSLoader 2-57
  - introduction 2-62
  - Standalone 2-62
  - standalone mode 1-24
- encoder
  - I/O driver support 1-19
- entering simulation parameters 4-48
- environment
  - network communication 2-24
  - serial communication 2-32

- environment properties
  - and Standalone mode 2-67
- Ethernet card
  - ISA bus 2-22
  - PC/104 bus 2-20
  - PCI bus 2-21
  - SBS bus 2-20
- Ethernet chip sets
  - supported 2-20
- external mode
  - controlling target application 4-67
  - user interaction 1-31

## 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 4-35
  - host PC 2-13
  - installed 2-13
  - project folder 2-13
  - scopes 4-35
  - working folder 2-13
  - xpc folder 2-13
  - xpcdemos folder 2-13
- fixed-point support 1-12
- floppy disk
  - creating for booting 2-51
- FreeDOS
  - copying kernel/application 2-69
- From blocks
  - xPC Target 1-32

**G**

- GPIB field bus
  - driver support 1-19
- graphical user interface (GUI)
  - custom with API 1-34
  - custom with API for Microsoft .NET framework 1-33
  - custom with COM API 1-34

**H**

- hardware environment
  - requirements for target PC 2-5
- hardware verification
  - hardware in the loop 3-7
- host computer
  - see host PC 1-14
- host PC 2-12
  - communication 1-23
  - configuring 2-16
  - connections 1-17
  - downloading software 2-12
  - files 2-13
  - hardware 2-31
  - license file 2-12
  - requirements 2-2

**I**

- I/O boards
  - supported by xPC Target 1-19
- I/O driver support
  - analog input (A/D) 1-19
  - analog output (D/A) 1-19
  - CAN field bus 1-19
  - counter timers 1-19
  - digital 1-19
  - encoder 1-19
  - GPIB 1-19
  - RS-232 1-19

- RS-422 1-19
- RS-485 1-19
- shared memory 1-19
- UDP 1-19

- industrial PC 1-15
- initial working folder 2-14
- installation prerequisite
  - obtaining a valid license 2-13
- installing
  - Ethernet card for ISA 2-22
  - Ethernet card for PCI 2-21
  - hardware 2-31
  - network communication 2-20
  - on the host PC 2-12
  - serial communication 2-31
  - testing 2-72
- ISA bus
  - Ethernet card 2-22

**K**

- kernel
  - target boot disk 1-6
  - target PC BIOS 1-6
  - with Standalone mode 2-68
  - xPC Target 1-6

**L**

- license
  - obtaining 2-13

**M**

- MathWorks
  - valid license 2-13
- MATLAB
  - controlling target application 4-69
- MATLAB Compiler support 1-12
- memory model
  - target application 1-9

**N**

- network boot 2-46
- network communication
  - advantages 1-18
  - environment 2-24
  - hardware 2-20
  - host computer 2-20
  - installing and setting up 2-20
  - specifying environment properties 2-24
  - target computer 2-20
- notebook PC 1-14

**O**

- outport block
  - adding to Simulink model 4-11
  - simulation parameters 4-15
- overview
  - MATLAB command-line interface 1-29

**P**

- parameter tuning
  - interactive 1-10
  - scripts 1-10
- parameters
  - scope blocks 4-25
- PC/104 bus
  - Ethernet card 2-20
- PCI bus
  - Ethernet card 2-21

**Q**

- Quatech serial drivers 1-19

**R**

- rapid prototyping process 3-2
- real-time application
  - memory model 1-9

- task execution time 1-9
- real-time kernel 1-6
- requirements
  - host PC 2-2
  - target PC 2-5
- RS-232
  - Quatech 1-19
- RS-422
  - Quatech 1-19
- RS-485
  - Quatech 1-19

**S**

- scope blocks
  - parameters 4-25
  - xPC Target 1-32
- scopes
  - file 4-35
  - host 4-31
  - target 4-25
- serial communication
  - environment 2-32
  - hardware 2-31
  - installing and setting up 2-31
  - section overview 2-31
- setting
  - network communication 2-24
  - serial communication 2-32
- setting initial working folder 2-14
- setup (network) dialog box
  - C compiler 2-24
- setup (serial) dialog box
  - C compiler 2-33
- shared memory driver support 1-19
- signal acquisition
  - logging 1-9
  - monitoring 1-9
  - tracing 1-9
- simulation

- from MATLAB 4-43
  - with Simulink 4-41
- simulation parameters
  - entering 4-15
  - for Simulink Coder 4-48
  - xPC Target Scope block 4-25
- Simulink external mode
  - controlling target application 4-67
- Simulink model
  - basic tutorial 4-2
  - outport block 4-11
  - scope block 4-7
  - scope parameters 4-7
  - xPC Target Scope blocks 4-20
- software environment
  - requirements on host PC 2-3
  - requirements on target PC 2-5
- software installation 2-12
- standalone mode
  - embedded option 1-24
- Standalone mode 2-62
  - copying kernel/target application 2-69
  - creating kernel/application 2-68
  - updating environment properties 2-67
- starting and stopping
  - target application 4-69
- system requirements
  - host PC 2-2
  - target PC 1-15

## T

- target application 1-9
  - building 4-54
  - control with external mode 4-67
  - copying with Standalone mode 2-69
  - downloading 4-54
  - memory model 1-9
  - starting 4-69
  - stopping 4-69

- task execution time 1-9
  - with DOSLoader mode 2-60
- target boot disk
  - creating 2-54
  - kernel 1-6
  - with desktop PC 1-15
  - with industrial PC 1-15
- target boot drive
  - creating 2-54
- target PC
  - boot disk 1-6
  - booting 4-46
  - command-line interface 1-32
  - communication 1-23
  - compatible target computers 2-9
  - connecting 1-17
  - creating boot drive 2-54
  - creating CD boot disk 2-43
  - creating CD bootable ROM 2-43
  - hardware 2-31
  - hardware requirements 2-5
  - I/O boards 2-9
  - real-time kernel 1-7
  - software installation 2-72
  - software requirements 2-5
  - troubleshooting setup 2-72
- task execution time (TET) 1-9
  - logging 4-48
- testing installation 2-72
- To blocks
  - xPC Target 1-32
- tutorial
  - basic 4-1
  - creating a Simulink model 4-2
  - simulating a Simulink model 4-41

## U

- UDP driver support 1-19
- USB bus

- Ethernet adapter 2-20
- user interaction
  - MATLAB command-line interface 1-29
  - Simulink external mode interface 1-31
  - target PC command line 1-32
  - Web browser 1-33
  - with API 1-34
  - with API for Microsoft .NET framework 1-33
  - with COM API 1-34
  - xPC Target Scope blocks 1-32

## **V**

- valid license
  - obtaining 2-13

## **W**

- Web browser

- user interaction 1-33
- working folder
  - initial 2-14
  - setting initial 2-14

## **X**

- xPC Target
  - features 1-6
  - interaction 1-26
  - introduction 1-1
  - kernel 1-6
  - supported I/O boards 1-19
- xPC Target Scope blocks
  - adding to Simulink model 4-20
  - interface 1-32