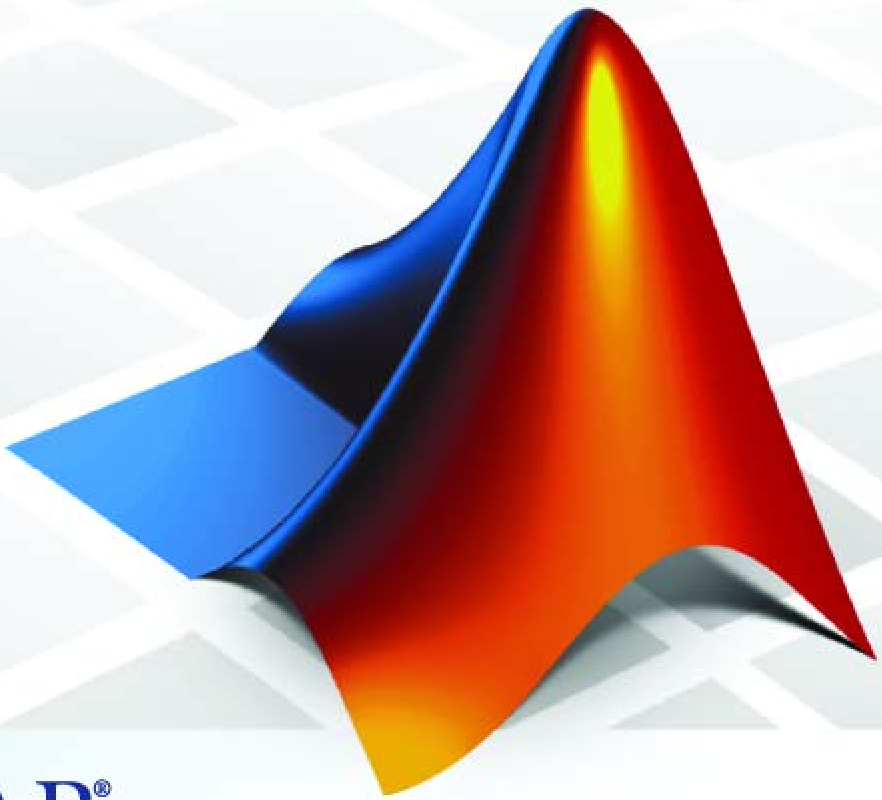
Getting Started with xPC Target 3



MATLAB®
& SIMULINK®

The MathWorks
Accelerating the pace of engineering and science

Getting Started with xPC Target

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)

Contents

1 Introduction

What Is xPC Target? **1-2**
Expected Background **1-3**

Features of xPC Target **1-4**
Real-Time Kernel **1-4**
Real-Time Application **1-6**
Signal Acquisition **1-7**
Parameter Tuning **1-8**
Fixed-Point Support **1-9**
MATLAB Compiler Support **1-9**

Hardware Environment **1-10**
Host PC **1-10**
Target PC **1-10**
Host-Target Connection **1-11**
I/O Driver Support **1-13**

Software Environment **1-16**
Host-Target Communication **1-16**
Rapid Prototyping Process **1-17**
xPC Target Embedded Option **1-19**

User Interaction **1-23**
xPC Target Explorer **1-25**
MATLAB Command-Line Interface **1-26**
Simulink External Mode Interface **1-27**
Simulink with xPC Target Scope Blocks **1-28**
Target PC Command-Line Interface **1-29**
Web Browser Interface **1-29**
Custom GUI with xPC Target API **1-30**
Custom GUI with xPC Target COM API **1-30**

2 Installation and Configuration

Required Products **2-3**
- MATLAB **2-3**
- Simulink **2-4**
- Real-Time Workshop **2-5**
- C Compiler **2-5**
- xPC Target Embedded Option **2-6**

Related Products **2-8**

System Requirements **2-9**
- Host PC Requirements **2-9**
- Target PC Requirements **2-10**

Installation on the Host PC **2-14**
- Getting or Updating Your License **2-14**
- Files on the Host PC Computer **2-15**
- Setting Your Initial Working Directory **2-16**
- Running MATLAB Remotely **2-17**
- Configuring the xPC Target Host PC for Your C Compiler **2-18**

xPC Target Explorer **2-20**
- xPC Target and Default Target PCs **2-22**

Serial Communication **2-24**
- Hardware for Serial Communication **2-24**
- Environment Properties for Serial Communication **2-25**

Network Communication **2-31**
- Advantages of Network Communication **2-31**
- Hardware for Network Communication **2-32**
- Ethernet Card Provided with xPC Target **2-33**
- Ethernet Chip Families Supported by xPC Target **2-33**
- Ethernet Card for a PCI Bus **2-36**
- Ethernet Card for an ISA Bus **2-37**
- Environment Properties for Network Communication **2-38**

Target Boot Disk **2-45**
Creating a Target Boot Disk with a Graphical User Interface **2-45**
Creating a Target Boot Disk with a Command-Line Interface **2-47**
Current Properties on the Target Boot Disk **2-48**

Testing and Troubleshooting the Installation **2-50**
Testing the Installation **2-50**
Test 1, Ping Target System Standard Ping **2-52**
Test 2, Ping Target System xPC Target Ping **2-54**
Test 3, Reboot Target Using Direct Call **2-54**
Test 4, Build and Download Application **2-55**
If You Need More Help **2-56**

Exporting and Importing xPC Target Explorer Environments **2-57**

3 Basic Tutorial

Simulink Model **3-2**
Creating a Simple Simulink Model **3-2**
Entering Parameters for the Scope Block **3-6**
Adding a Simulink Outport Block **3-10**
Entering Parameters for the Outport Blocks **3-13**
Adding an xPC Target Scope Block **3-17**
Entering Parameters for an xPC Target Scope Block **3-21**

Simulating the Model **3-38**
Simulating the Model with Simulink **3-38**
Simulating the Model from MATLAB **3-40**

xPC Target Application **3-44**
Booting the Target PC **3-44**
Troubleshooting the Boot Process **3-45**
Entering the Real-Time Workshop Parameters **3-46**
Building and Downloading the Target Application **3-51**
Troubleshooting the Build Process **3-53**

Increasing the Time-Out Value **3-54**
xPC Target Options **3-55**

Running the Target Application **3-60**
Control with xPC Target Explorer **3-60**
Control with MATLAB Commands **3-70**
Control with Simulink External Mode **3-72**

Menu Bar and Toolbar Contents and Shortcut Keys ... **3-74**

Glossary

Index

Introduction

This chapter is an overview of the functions and features of xPC Target. An introduction to these features and the xPC Target software environment helps you develop a model for working with xPC Target. This chapter includes the following sections:

What Is xPC Target? (p. 1-2)	Solution for prototyping, testing, and deploying real-time systems using a host PC and a separate target PC
Features of xPC Target (p. 1-4)	Real-time kernel, real-time application, signal acquisition, and parameter tuning
Hardware Environment (p. 1-10)	Target PC types, host and target PC connection, and I/O driver support
Software Environment (p. 1-16)	Host and target PC communication, rapid prototyping process, and deployment process
User Interaction (p. 1-23)	xPC Target GUI, MATLAB® command line, Simulink® external mode, Web browser, custom GUI with Simulink, custom GUI with xPC Target API, and custom GUI with xPC Target COM API

What Is xPC Target?

xPC Target is a solution for prototyping, testing, and deploying real-time systems using standard PC hardware. It is an environment that uses a target PC, separate from a host PC, for running real-time applications.

In this environment you use your desktop computer as a host PC with MATLAB, Simulink, and Stateflow® (optional), to create a model using Simulink blocks and Stateflow charts. After creating your model, you can run simulations in nonreal time.

xPC Target lets you add I/O blocks to your model and then use the host PC with Real-Time Workshop®, Real-Time Workshop Embedded Coder (optional), Stateflow Coder (optional), and a C/C++ compiler to create executable code. The executable code is downloaded from the host PC to the target PC running the xPC Target real-time kernel. After downloading the executable code, you can run and test your target application in real time.

- Hardware requirements — The xPC Target software requires a host PC, target PC, and, for I/O, the target PC must also have I/O boards supported by xPC Target. However, the target PC can be a desktop PC, industrial PC, PC/104, PC/104+, or CompactPCI computer.
- Software requirements — The xPC Target software requires either a Microsoft Visual C/C++ compiler (Version 6.0, 7.1, or 8.0) or an Open Watcom C/C++ compiler (Version 1.3). In addition, xPC Target requires MATLAB, Simulink, and Real-Time Workshop.
- xPC Target Embedded Option requirements — The xPC Target Embedded Option is a separate product that requires an additional license from The MathWorks. With this additional license, you can deploy an unlimited number of real-time applications for stand-alone operation. This option allows you to
  - Boot the target PC from an alternate device other than a floppy disk drive, such as a hard disk drive or flash memory.
  - Create stand-alone applications for the target PC, independent from the host PC, that can boot from a floppy drive or an alternate device.
  - Deploy stand-alone GUI applications running on the host PC to control, change parameters, and acquire signal data from a target application.

This feature uses the xPC Target API with any programming environment, or the xPC Target COM API with any programming environment, such as Visual Basic, that can use COM objects. Without the xPC Target Embedded Option, you can create, but not deploy, stand-alone GUI applications running on a host PC that does not contain your licensed copy of xPC Target, to control, change parameters, and acquire signal data from a target application.

- Documentation and help — The xPC Target software ships with the Getting Started with xPC Target documentation. 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.

Expected Background

Users who read this book should be familiar with

- Using Simulink and Stateflow to create models as block diagrams, and simulating those models in Simulink
- The concepts and use of Real-Time Workshop to generate executable code

When using Real-Time Workshop and xPC Target, you do not need to program in C or other programming languages to create, test, and deploy real-time systems.

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

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

Features of xPC Target

The xPC Target software environment includes many features to help you prototype, test, and deploy real-time systems. This section includes the following topics:

Real-Time Kernel (p. 1-4)	BIOS, BIOS-extension, kernel, and loader
Real-Time Application (p. 1-6)	Memory model and task execution time
Signal Acquisition (p. 1-7)	Signal monitoring, signal logging to the MATLAB workspace, and signal tracing on the host PC or target PC screen
Parameter Tuning (p. 1-8)	Interactive scripts and batch procedures
Fixed-Point Support (p. 1-9)	Support for Simulink fixed-point data
MATLAB Compiler Support (p. 1-9)	Support for the MATLAB Compiler

Real-Time Kernel

xPC Target does not require DOS, Windows, Linux, or any another operating system on the target PC. Instead, you boot the target PC with a boot disk that includes the xPC Target kernel.

However, the xPC Target Embedded Option requires DOS and a DOS license at boot time. For more information, see the xPC Target User's Guide documentation.

Target Boot Disk

The boot disk eliminates the need to install software, modify existing software configurations, or access the hard disk on the target PC. This arrangement allows you to use the target PC for testing real-time applications, and then when you are finished with your tests, you can use the target PC again as a desktop computer. Software is not permanently installed on the target PC unless you deliberately install a stand-alone application on the hard disk or flash memory using the xPC Target Embedded Option.

Target PC BIOS

Selecting a current BIOS allows you to customize settings for better control over the real-time behavior of the system. For example, you can suppress checking for a keyboard and switch off any power save features.

The xPC Target kernel runs only on a PC-compatible system, and a key component of every PC-compatible system is the BIOS. The BIOS is the only software component that is needed by the xPC Target kernel.

After the BIOS is loaded, it searches the target boot disk for a bootable image (executable). This bootable image includes a 16-bit part and a 32-bit part. The 16-bit part runs first because the CPU is still in real mode. It prepares the descriptor tables and, in addition to other things, switches the CPU to protected mode. Next, the 32-bit part runs. It prepares the target PC environment for running the kernel and finally starts the kernel.

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

Real-Time Kernel

After the kernel starts running, it displays a welcome message with information about the host-target connection. The kernel activates the application loader and waits to download a target application from the host PC. 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 as Windows NT applications with a flat memory model. This provides full 32-bit power without time-consuming 16-bit segment switching and DOS extenders.

Target PC Heap

The initialization code of the target application reserves the remaining unused RAM as heap. The memory available for the heap is displayed on the left side of the target screen as **Memory**. By default, it is 4 MB less than the entire RAM installed (1 MB for the application; 3 MB for the kernel). Normally, the largest part of the heap is used by signal logging because logging acquires and stores data during the entire run.

You can define the amount of memory available for data logging in the Configuration Parameters dialog box. See "Entering the Real-Time Workshop Parameters" on page 3-46.

Real-Time Application

Real-Time Workshop, Real-Time Workshop Embedded Coder, Stateflow Coder, xPC Target, and a C compiler create a real-time application (target application) from a Simulink and Stateflow model. Target applications created with Real-Time Workshop and xPC Target run in real time on a standard PC without using a Windows operating system.

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

- Memory model — The target application is compiled as a Windows NT application with the flat memory model. This executable is then converted to an image suitable for xPC Target, 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 μs (50 kHz). Model size, complexity, and target PC hardware affect maximum speed (minimal sample time) of execution.

For more information on creating a target application, see "xPC Target Application" on page 3-44.

Signal Acquisition

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

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

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

You can use xPC Target to acquire and log signals of the following data types:

- `double`
- `single`
- `uint8/int8`
- `uint16/int16`
- `uint32/int32`

xPC Target also allows you to monitor and tune signal and parameter values represented by fixed-point numbers.

For information on acquiring signal data with xPC Target, see “User Interaction” on page 1-23 in Getting Started with xPC Target and “Signal Monitoring with MATLAB”, “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 — xPC Target supports tuning of parameters while the target application is running in real time.

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

- Scripts and batch procedures — xPC Target also includes commands to change parameters during a run or between runs. By writing a script that incrementally changes a parameter and monitors a signal output and running it on the host PC, you can optimize the value of that parameter.

You can use xPC Target to tune parameters of the following data types:

- `double`
- `single`
- `uint8/int8`
- `uint16/int16`
- `uint32/int32`

For information on tuning parameters with xPC Target, see “User Interaction” on page 1-23 in Getting Started with xPC Target and “Parameter Tuning and Inlining Parameters” in the xPC Target User’s Guide documentation.

Fixed-Point Support

xPC Target 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” documentation.

MATLAB Compiler Support

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

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

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

To use these features, write an M-file that uses the MATLAB command-line interface for xPC Target (for example, then use the MATLAB Compiler).

Hardware Environment

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 xPC Target will help you to set up a development environment that meets your needs. This section includes the following topics:

Host PC (p. 1-10)	Desktop PC or notebook PC
Target PC (p. 1-10)	Desktop PC or industrial PC
Host-Target Connection (p. 1-11)	RS-232 serial or TCP/IP network
I/O Driver Support (p. 1-13)	Analog, digital, CAN, GPIB, RS-232, UDP, counters, timers, and signal conditioning

Host PC

You can use any PC that runs a Microsoft Windows platform supported by The MathWorks as the host PC. It must also contain a 3.5-inch floppy disk drive, and a free serial port or an Ethernet adapter card.

The host PC can be one of the following:

- Desktop PC
- Notebook PC

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

Target PC

xPC Target supports up to 64 target PCs with one host. A target PC can be almost any PC compatible system with a 32–bit Intel or AMD processor (386 compatible or higher). It must also contain a 3.5-inch floppy disk drive, and a free serial port or an Ethernet adapter card. Using the xPC Target Embedded Option, you can transfer files from the 3.5-inch disk to a hard disk or flash memory.

A target PC can be one of the following:

- Desktop PC — This computer is booted from a special target boot disk created by xPC Target.

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

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

- Industrial PC — This computer is booted from a special target boot disk, or, using the xPC Target Embedded Option, from a hard disk or flash memory.

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

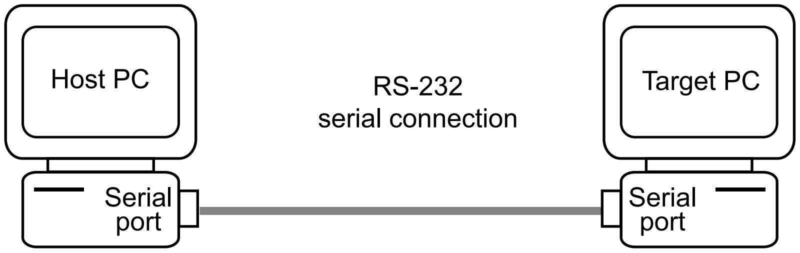
You do not need any special target hardware. However, the target PC must be a fully PC-compatible system and contain a serial port or an Ethernet controller compatible with xPC Target.

For more details on the requirements of the target PC, see "Target PC Requirements" on page 2-10.

Host-Target Connection

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

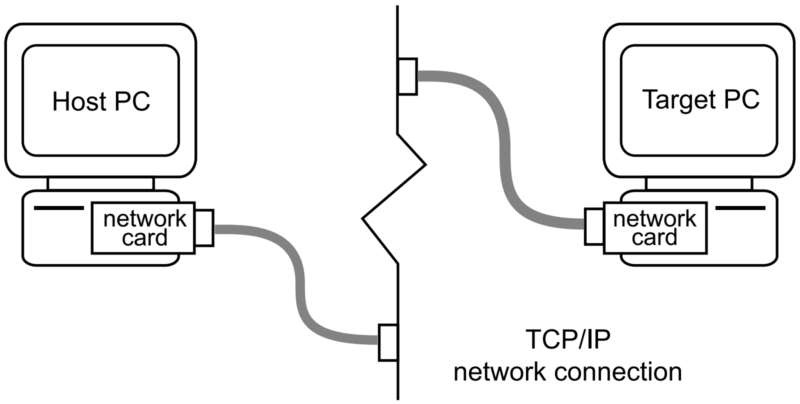
Serial — The host and target computers are connected directly with a serial cable using their RS-232 ports. This cable is wired as a null modem link that can be up to 5 meters long and with a transfer rate between 1200 and 115200 baud. A null modem cable is provided with the xPC Target software.



For detailed information on setting up the hardware and software for serial communication, see “Serial Communication” on page 2-24.

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 with Ethernet adapter cards using the TCP/IP protocol for communication.

When using a network connection, the target PC can use the Ethernet adapter card provided with xPC Target or one of the supported cards. 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 2-32.



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

I/O Driver Support

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

Note You are responsible for taking necessary precautions and implementing safeguards when interfacing hardware with xPC Target. 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 xPC Target. You drag and drop blocks from the I/O library and connect I/O drivers to your model the same way you would connect any standard Simulink block.

I/O support — The I/O device library supports approximately 300 standard boards. I/O boards plug into the target PC expansion bus, PC/104 stack, or industrial PC chassis. There is also support for modules that plug into IP or PMC carrier boards. xPC Target 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 xPC Target I/O Reference documentation.
- 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 xPC Target I/O Reference documentation.
- 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 xPC Target I/O Reference documentation.
- UDP support — Communicate with another system using the standard UDP/IP network protocol. See “UDP I/O Support” in xPC Target I/O Reference documentation.
- Counter-Timers — Use the counter-timer blocks for measuring pulse and frequency with modulation applications.
- Watchdog — Monitor an interrupt or memory location, and reset the computer if an application does not respond. See “Access IO” and “Versalogic” in xPC Target I/O Reference documentation.

- 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.
- 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 “I/O Reference”.

Software Environment

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

Host-Target Communication (p. 1-16)	Control the target application, upload signal data, and download parameter values.
Rapid Prototyping Process (p. 1-17)	Design and build a target application on a host PC and download the target application to a target PC. Run and test the target application with tuning of parameters and acquisition of data.
xPC Target Embedded Option (p. 1-19)	Boot the kernel from a device other than the floppy disk drive, but download the target application from the host PC. Or boot the kernel/application from the floppy disk drive or another device, and run the target application completely separately from the host PC.

Host-Target Communication

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

- Target application — Download a target application from the host to the target computer.
- Control — Change properties and control the target application. This includes starting and stopping the target application, changing sample and stop times, and getting information about the performance of the target application and CPU.

- Signal data — Upload signal data from the target computer for analysis after the target application is finished running, or view signal data during the run.
- Parameter values — Download parameter values to the target computer between runs or during a run.

Rapid Prototyping Process

Design and build a target application on a host PC, and then run and test the target application on a target PC. xPC Target functions include control of the target application, acquisition of signal data, and tuning of parameters while running in real time.

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

The rapid prototyping process includes the following tasks:

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

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

   ---

3. Create an executable target application — Real-Time Workshop, Stateflow Coder, xPC Target, and a C compiler create the target application that runs

on the target PC. This real-time application uses the initial parameters from the Simulink model that were available at the time of code generation.

4. Execute the target application in real time — The target PC is booted using an xPC Target boot disk that loads the xPC Target real-time kernel. After booting the target PC, you can build and download a real-time application.

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

5. Acquire signals — Acquire signal data using xPC Target scopes. xPC Target supports 10 scopes each for scopes of type `host` and `target`, and 8 scopes of type `file`, for a maximum of 28 scopes.

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

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

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

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

   - Target PC — Use commands in the target PC command window.
   - Web browser — Use the xPC Target Web browser interface.

- Simulink GUI — Add blocks to a Simulink user interface model with xPC Target From blocks. See “Graphical User Interfaces” in the xPC Target User’s Guide documentation for details.

6 Tune parameters — You can tune parameters by

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

xPC Target Embedded Option

Often, control system and digital signal processing applications are developed for use in production where a limited number of deployed systems is required. Whether you deploy one or a hundred systems, the xPC Target Embedded Option provides a convenient approach that allows you to implement your system on low-cost PC hardware.

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

The xPC Target Embedded Option has two modes of operation. In each case, the target PC boots into DOS, starts the DOS program `xpcboot.com` from `autoexec.bat`, and then starts the kernel from `xpcboot.com`:

DOSLoader Mode (p. 1-20)	Run the kernel from a device other than the floppy disk, such as a hard disk or flash disk, and download the target application from the host PC.
StandAlone Mode (p. 1-21)	Boot the device and run both the kernel and the target application from a device other than the floppy disk, such as the target PC hard disk or flash disk. The target application runs completely independently from the host PC.

In contrast, when using the normal `BootFloppy` mode, you do not need DOS to load and run the xPC Target kernel.

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

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

DOSLoader Mode

`DOSLoader` mode allows you to boot your target computer from devices other than a 3.5-inch floppy disk. For example, the boot device can be a hard disk drive or flash memory. The target application is still downloaded from the host PC.

You can also use this mode to boot from a floppy disk, but there is no advantage to using this method over using a normal target boot disk. The advantage to using `DOSLoader` mode with the extra work to create a bootable DOS disk is that you can boot from an alternative device.

1 Select `DOSLoader` mode from the `Configuration` node in the **xPC Target Hierarchy** pane of xPC Target Explorer, a graphical user interface that allows you to manage the xPC Target environment.

2 Create a new target boot disk.

3 Copy DOS system files and utilities to the target boot disk.

4 In the `autoexec.bat` file, temporarily remove the line that executes `xpcboot.com`.

5 Boot the target PC with a bootable DOS disk from the floppy disk.

6 Copy files to the alternate boot device. In the `autoexec.bat` file, add the line that loads `xpcboot.com`.

7 Remove the target boot disk and reboot the target PC from the alternate boot device.

8 Build a target application and download it to the target PC.

For more information on the xPC Target Embedded Option, see "Embedded Option" in the xPC Target User's Guide documentation.

StandAlone Mode

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

1 Select `StandAlone` mode from the `Configuration` node in the **xPC Target Hierarchy** pane of the xPC Target Explorer tool.

2 Build a kernel/target application.

3 Copy DOS system files, utilities, kernel/application files, and helper files to the target PC hard drive or flash memory.

4 Boot the target PC.

   When you boot the target PC, the target PC loads DOS, 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, see “Embedded Option” in the xPC Target User’s Guide documentation.

User Interaction

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

xPC Target Explorer (p. 1-25)	Use the xPC Target Explorer to set xPC Target environment properties, create a boot disk for the target PC, download and control a target application, add and view xPC Target scopes, and change tunable parameters.
MATLAB Command-Line Interface (p. 1-26)	Enter xPC Target functions on the host PC.
Simulink External Mode Interface (p. 1-27)	Connect a Simulink block diagram to the target application using Simulink External Mode, and then use the block diagram for parameter tuning.
Simulink with xPC Target Scope Blocks (p. 1-28)	Add xPC Target Scope blocks to your model for signal tracing.
Target PC Command-Line Interface (p. 1-29)	Enter xPC Target functions on the target PC.
Web Browser Interface (p. 1-29)	Use Microsoft Internet Explorer or Netscape Navigator to connect to the target PC from any computer on the network with the target PC.

Custom GUI with xPC Target API (p. 1-30)	Create a client application that interfaces with a target application using any development environment that can call functions from a DLL.
Custom GUI with xPC Target COM API (p. 1-30)	Create a client application that interfaces with a target application using Visual Basic or any development environment that can incorporate COM objects.

Note Some blocks (see "Blocks That Preclude Sample-Time Inheritance" in the using Simulink documentation) cannot properly handle sample time changes at run-time. For models that contain these blocks, change the sample time in the model first, then build that model. Although xPC Target allows you to change sample times at run-time, changing them at run-time for these blocks might cause incorrect results.

The following table compares the interfaces supported by xPC Target.

Interface	Environment Properties	Control	Signal Acquisition	Parameter Tuning
"xPC Target Explorer" on page 1-25	X	X	X	X
"MATLAB Command-Line Interface" on page 1-26	X	X	X	X
"Simulink External Mode Interface" on page 1-27		X		X
"Simulink with xPC Target Scope Blocks" on page 1-28			X	
"Target PC Command-Line Interface" on page 1-29		X	X	X
"Web Browser Interface" on page 1-29		X	X	X

Interface	Environment Properties	Control	Signal Acquisition	Parameter Tuning
"Custom GUI with xPC Target API" on page 1-30		X	X	X
"Custom GUI with xPC Target COM API" on page 1-30		X	X	X

xPC Target Explorer

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

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

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

  For more information on environment properties, see

  - "Serial Communication" on page 2-24 and "Network Communication" on page 2-31
  - "Working with Target PC Environments" in the xPC Target User's Guide documentation
  - The `getxpcenv` function in the xPC Target User's Guide documentation
- Control — Use the xPC Target Explorer to download a model. After the target application is downloaded to the target PC, you can use xPC Target Explorer to run it. Use xPC Target Explorer to change stop time and sample times without regenerating code, and get statistical performance information during or after the last run.
- Signal acquisition — Use the xPC Target Explorer `Model Hierarchy` node to interactively add scopes of type `host`, `target`, or `file`, 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 documentation.

- 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 documentation.

MATLAB Command-Line Interface

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

xPC Target 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 xPC Target 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 and directly change the environment properties without using a graphical interface.

  For more information on environment properties, see "Creating a Target Boot Disk with a Command-Line Interface" on page 2-47, and "Software Environment and Demos" in the xPC Target User's Guide documentation.

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

  For more information, see "Control with MATLAB Commands" on page 3-70 in Getting Started with xPC Target and "Software Environment and Demos" in the xPC Target User's Guide documentation.

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

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

- 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 MATLAB” and “Targets and Scopes in the MATLAB Interface” in the xPC Target User’s Guide documentation.

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 documentation.

- 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 documentation.

- Parameter tuning — Select external mode, and change parameters in the target application by changing parameters in the Block Parameters dialog boxes. Once you change a value and click **OK**, the new value is downloaded to the target PC and replaces the existing parameter while the target application continues to run. For more information, see “Parameter Tuning with Simulink External Mode”.

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

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

Note xPC Target does not support data acquisition through Simulink external mode. Instead, use xPC Target Scope blocks for signal acquisition. See “Simulink with xPC Target Scope Blocks” on page 1-28.

Simulink with xPC Target Scope Blocks

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

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

For information on acquiring signal data with the xPC Target, see “Adding an xPC Target Scope Block” on page 3-17, “Entering Parameters for an xPC Target Scope Block” on page 3-21, “Entering Parameters for an xPC Target Scope of Type File” on page 3-32 in Getting Started with xPC Target, and “Signal Tracing with xPC Target Scope Blocks” in the xPC Target User’s Guide documentation.

Target PC Command-Line Interface

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

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

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

  For more information, see "Using the Target PC Command-Line Interface" in the xPC Target User's Guide documentation.
- Signal acquisition — Acquiring signal data is limited to viewing signal traces and signal monitoring on the target PC screen.
- Parameter tuning — You can change only scalar parameters in your model.

Web Browser Interface

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

The Web browser interface includes the following functions:

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

  For more information, see "xPC Target Web Browser Interface" in the xPC Target User's Guide documentation.
- Signal acquisition — Signal tracing is limited to viewing a snapshot of a screen captured from the target PC screen. Add scopes of type target, 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 documentation.

- 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 documentation.

Custom GUI with xPC Target API

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

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

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

Custom GUI with xPC Target COM API

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

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

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

Installation and Configuration

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

Required Products (p. 2-3)	Products from The MathWorks and third-party products that you need to use with xPC Target
Related Products (p. 2-8)	Products from The MathWorks that allow you to use state logic in your models and animate signal data
System Requirements (p. 2-9)	Selecting a host PC and target PC
Installation on the Host PC (p. 2-14)	Getting a valid license for xPC Target, and a separate license for the xPC Target Embedded Option. Installing the xPC Target software by downloading it from the Web.
xPC Target Explorer (p. 2-20)	xPC Target Explorer, a graphical user interface for xPC Target
Serial Communication (p. 2-24)	Selecting RS-232 communication for an easy and inexpensive installation
Network Communication (p. 2-31)	Selecting TCP/IP communication for faster data transfer rates and longer connections

Target Boot Disk (p. 2-45)	Booting the xPC Target kernel on the target PC and establishing a connection with the host PC
Testing and Troubleshooting the Installation (p. 2-50)	Testing the installation
Exporting and Importing xPC Target Explorer Environments (p. 2-57)	Exporting and importing target PC node property settings in a structured format to and from the MATLAB workspace

Required Products

xPC Target is a PC-compatible product that you install on a host computer running a Microsoft Windows (32-bit) operating system. xPC Target requires the following products from The MathWorks:

MATLAB (p. 2-3)	Control and interact with the xPC Target software environment and target application using a command-line interface.
Simulink (p. 2-4)	Model dynamic physical systems and controllers using block diagrams.
Real-Time Workshop (p. 2-5)	Convert Simulink blocks and Stateflow charts into C code.
C Compiler (p. 2-5)	Use a third-party C compiler and Real-Time Workshop to build a target application.
xPC Target Embedded Option (p. 2-6)	Deploy stand-alone target applications and custom GUI applications that communicate with the target application. Note that you can create custom GUI applications without xPC Target Embedded Option.

MATLAB

MATLAB provides a command-line interface for xPC Target.

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

- Real-time application control — Download, start, and stop the target application.
- Signal acquisition and analysis — Save signal data while the target application is running and analyze the data after the application has

completed running, or display signal data while the target application is running in real time.

- Parameter tuning — Change parameters while the target application is running in real time.

Note Version 3.2 of xPC Target requires MATLAB Version 7.4.

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

Simulink

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

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

Note Real-Time Workshop supports C code S-functions. It does not support M-code S-functions.

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

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

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

Note Version 3.2 of xPC Target requires Simulink Version 6.6.

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

Real-Time Workshop

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

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

Note Version 3.2 of xPC Target requires Real-Time Workshop Version 6.6.

Real-Time Workshop documentation — For information on code generation, see the “Real-Time Workshop” documentation.

C Compiler

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

Note To configure your C compiler for xPC Target, use the xPC Target Explorer interface (see “Configuring the xPC Target Host PC for Your C Compiler” on page 2-18). Do not use the `mex -setup` command to set the C compiler for xPC Target.

In addition to the products from The MathWorks, you need to install a C compiler. Real-Time Workshop and xPC Target support the following C compilers:

- Microsoft Visual C/C++ — Version 3.2 of xPC Target requires Microsoft Visual Studio C/C++ Version 6.0, 7.1 (.NET 2003), or 8.0 and Microsoft Visual C/C++ 2005 Express Edition. For optimal performance, use the professional versions of these tools in your environment. You can also use the standard versions, but note that these are not optimized and you might experience slower performance. If you are going to write stand-alone GUI applications with the C or COM API, you can use Visual C/C++ to build your target application. To deploy a stand-alone GUI application that you create with the C or COM API, you need the xPC Target Embedded Option. Without the xPC Target Embedded Option, you can write, but not deploy, stand-alone GUI applications running on the host PC to control or change parameters and acquire signal data from a target application.

  Note If you use Microsoft Visual C/C++ 2005 Express Edition, xPC Target does not support the generation of COM objects from the xPC Target model.

- Open Watcom C/C++ — Version 3.2 of xPC Target requires Open Watcom C/C++ Version 1.3.

xPC Target Embedded Option

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

- Boot the target PC from an alternate device other than a floppy disk drive such as a hard disk drive or flash memory.

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

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

Note Version 3.2 of xPC Target requires xPC Target Embedded Option Version 3.2.

Related Products

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

For more information about any of these products, see either

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

System Requirements

The hardware and software requirements are different for the host and target computers. This section includes the following topics:

Host PC Requirements (p. 2-9)	Desktop or notebook PC
Target PC Requirements (p. 2-10)	Desktop, industrial PC, PC/104, PC/104+, or CompactPCI

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

Host PC Requirements

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

Software Requirements for the Host PC

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

Software	Description
32-bit operating system	Microsoft Windows platform supported by The MathWorks
MATLAB	MATLAB Version 7.4
Simulink	Simulink Version 6.6
Real-Time Workshop	Real-Time Workshop Version 6.6

Software	Description
C language compilers	Microsoft Visual C/C++ Professional Edition Version 6.0, 7.1, or 8.0 Microsoft Visual C/C++ 2005 Express Edition **Note** If you use Microsoft Visual C/C++ 2005 Express Edition, xPC Target does not support the generation of COM objects from the xPC Target model. Open Watcom C/C++ Version 1.3
xPC Target	xPC Target Version 3.2

Hardware Requirements for the Host PC

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

Hardware	Description
Communication	One free serial port (COM1 or COM2) with a 9-pin or 25-pin D-sub connector, or an Ethernet card connected to a network
CPU	Pentium, Athlon, or later
Peripherals	Hard disk drive with 60 MB of free space One 3.5-inch floppy disk drive
RAM	128 MB or more

Target PC Requirements

The target PC must be a PC-compatible system. You can use a second desktop computer as the target PC, but you can also use an industrial system like a PC/104 or CompactPCI as the target computer.

Software Requirements for the Target PC

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

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

Hardware Requirements for the Target PC

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

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

Hardware	Description
Chip set	PC compatible with UART, programmable interrupt controller, keyboard controller, and counter
Communication	One free serial port (COM1 or COM2) with a 9-pin or 25-pin D-sub connector, or an Ethernet card connected to a network. The xPC Target software includes a serial null modem cable and an Ethernet card for the target PC.
CPU	Intel 386/486/Pentium or AMD K5/K6/Athlon with or without a floating-point coprocessor
Keyboard and mouse	Needed to control the target PC when you create stand-alone applications Note that if a keyboard is not connected, the BIOS might display an error message (keyboard failure). With a current BIOS, you can use the BIOS setup to skip the keyboard test.

Hardware	Description
Monitor	The MathWorks recommends using a monitor, but it is not necessary. You can get all the target information using xPC Target functions on the host PC.
Peripheral	One 3.5-inch floppy disk drive. A hard disk drive is not required. Notes: • If you install the xPC Target Embedded Option, you can copy files to a hard disk or flash memory and boot from that device. • If you have a hard disk drive, and you want to access the target PC file system on that drive, see "Working with Target PC Files and File Systems" in the xPC Target User's Guide. xPC Target supports file systems of type FAT-12, FAT-16, or FAT-32.
RAM	8 MB or more

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

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

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

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

For a list of I/O functions supported by xPC Target, see “I/O Driver Support” on page 1-13.

Installation on the Host PC

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

Getting or Updating Your License (p. 2-14)	Valid License File or Personal License Password (PLP)
Files on the Host PC Computer (p. 2-15)	Directories that contain general xPC Target files
Setting Your Initial Working Directory (p. 2-16)	MATLAB working directory outside the MATLAB root directory
Running MATLAB Remotely (p. 2-17)	Active X control registration if you are running MATLAB remotely (accessing MATLAB over the network)
Configuring the xPC Target Host PC for Your C Compiler (p. 2-18)	Host PC configuration through xPC Target Explorer

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

After you install xPC Target software, you will need to set up the xPC Target environment for either serial or network communication. See “Serial Communication” on page 2-24 or “Network Communication” on page 2-31.

Getting or Updating Your License

Before you install xPC Target or the xPC Target Embedded Option, you must have a valid License File or Personal License Password (PLP). The License File or Personal License Password identifies the products you purchased from The MathWorks and are permitted to install and use.

When you purchase a product, The MathWorks sends you a License File or Personal License Password (PLP) in an e-mail message. If you have not received a PLP number, contact The MathWorks.

Internet	`http://www.mathworks.com/accesslogin` Log in to MathWorks Account using your e-mail address and password. Go to the My Licenses panel to determine your PLP number.
Telephone	508-647-7000. Ask for Customer Service.
Fax	508-647-7001. Include your license number.

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

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

Files on the Host PC Computer

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

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

  Select a working directory outside the MATLAB root. See "Setting Your Initial Working Directory" on page 2-16.
- Real-Time Workshop Build directory — The Real-Time Workshop C code files (`model.c`, `model.h`) are in a subdirectory called `modelname_xpc_rtw`.

xPC Target 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
- `xpc` — Host PC functions related to all of xPC Target, methods for target objects, and methods for scope objects

- `xpcdemos` — Simulink models and M-file demos

Setting Your Initial Working Directory

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

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

From the Desktop Icon

Your initial working directory is specified in the shortcut file you use to start MATLAB. To change this initial directory, 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 directory path you want MATLAB to use initially. Make sure you choose a directory outside the MATLAB root directory.
3. Click **OK**, and then start MATLAB. To check your working directory, in the MATLAB Command Window, type

   ```
   pwd
   ```

From Within MATLAB

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

1. In the MATLAB window, type

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

2. To check your working directory, type

   ```
   pwd or cd
   ```

To permanently set your working directory, see “From the Desktop Icon” on page 2-16.

Running MATLAB Remotely

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

1. In the MATLAB Command Window, type

   ```
   xpc_register_ocx
   ```

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

2. Close xPC Target Explorer.

3. Close MATLAB.

4. Restart MATLAB.

5. Restart xPC Target Explorer.

You are now ready to start xPC Target Explorer.

Configuring the xPC Target Host PC for Your C Compiler

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

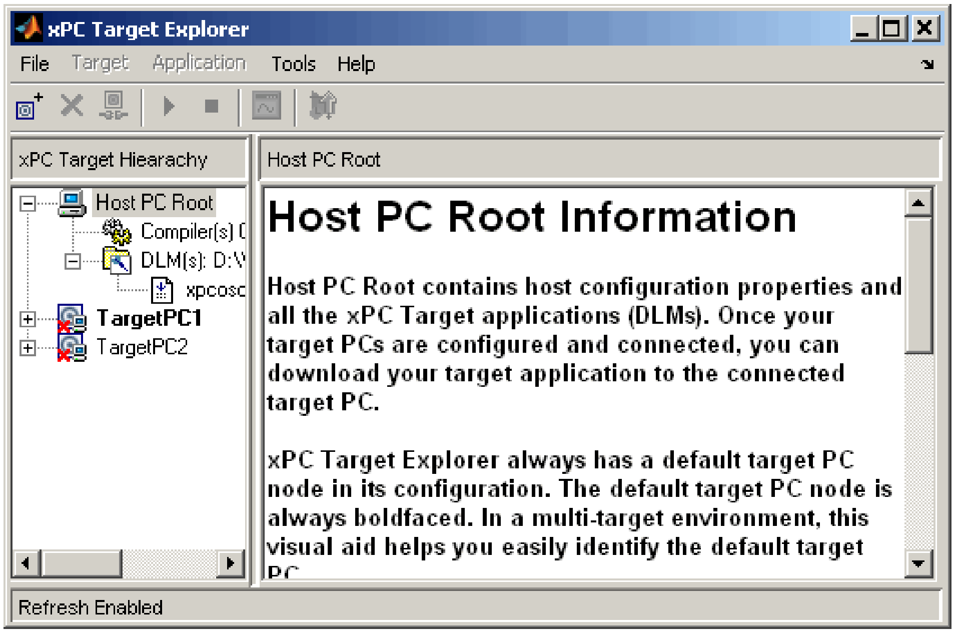
Note Do not use the `mex -setup` command to set the C compiler for xPC Target.

---

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

```
xpcexplr
```

The xPC Target Explorer window appears.



Host PC Root

Host PC Root Information

Host PC Root contains host configuration properties and all the xPC Target applications (DLMs). Once your target PCs are configured and connected, you can download your target application to the connected target PC.

xPC Target Explorer always has a default target PC node in its configuration. The default target PC node is always boldfaced. In a multi-target environment, this visual aid helps you easily identify the default target PC

xPC Target Explorer always has a default target PC node in its configuration. The default target PC node is always boldfaced. In a multitarget environment, this visual aid helps you easily identify the default target PC.

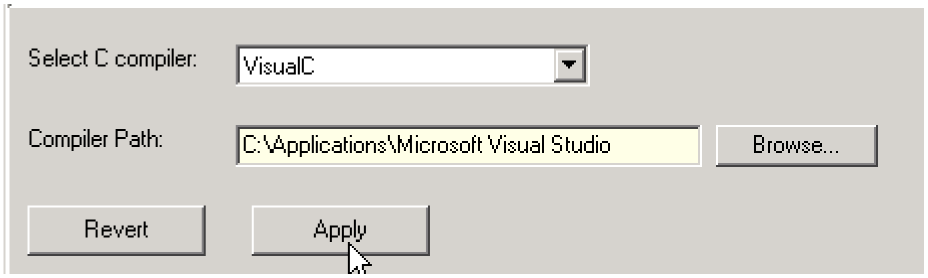
2 In the xPC Target Explorer window, select the `Compiler(s) Configuration` node.

   In the right pane, the compiler parameters appear.

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

4 Enter the path (or browse) to the compiler for **Compiler Path**. For example,

```
C:\Program Files\Microsoft Visual Studio
```



Select C compiler: VisualC

Compiler Path: 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.

---

xPC Target Explorer

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

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

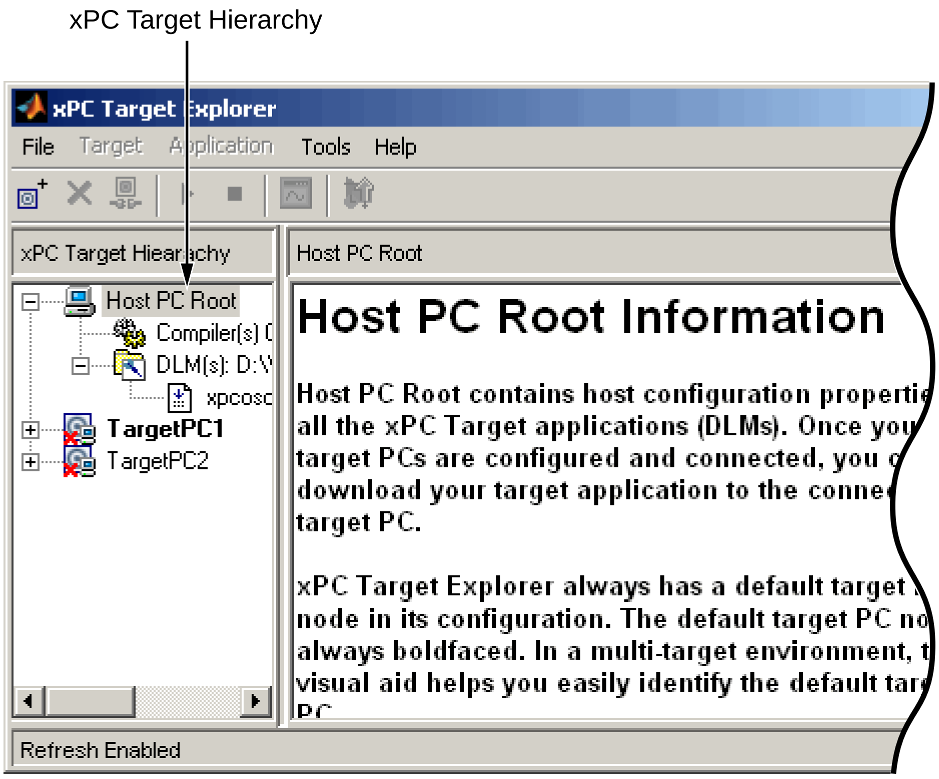
The xPC Target Explorer GUI runs on your xPC Target system host machine.

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

1 In the MATLAB Command Window, type

```
xpcexplr
```

The xPC Target Explorer window opens.



Host PC Root Information

Host PC Root contains host configuration properti
all the xPC Target applications (DLMs). Once you
target PCs are configured and connected, you c
download your target application to the conne
target PC.

xPC Target Explorer always has a default target
node in its configuration. The default target PC no
always boldfaced. In a multi-target environment, t
visual aid helps you easily identify the default tar
PC

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

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

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

xPC Target and Default Target PCs

The following are notes on default target PCs:

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

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

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

- The target PC commands `getxpcenv` and `setxpcenv` get and set environment properties of the default target PC.

Serial Communication

Before you can create and run a target application, you need to set up the connection between your host and target computers. You can use either serial or network communication. This section includes the following topics:

Hardware for Serial Communication (p. 2-24)	Use a null modem cable to connect the host PC to the target PC.
Environment Properties for Serial Communication (p. 2-25)	Enter information about the C/C++ compiler you installed on the host PC and the COM port you connected to the null modem cable.

For network communication, see “Network Communication” on page 2-31.

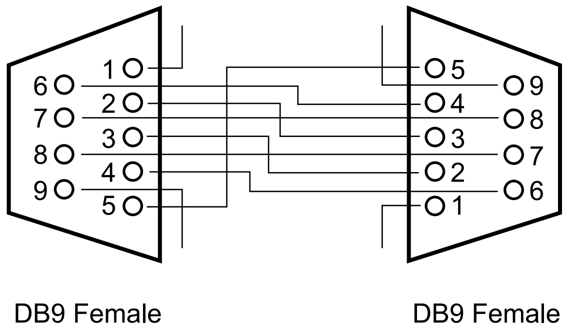
Hardware for Serial Communication

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

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

Null Model Cable Wiring

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



Environment Properties for Serial Communication

The xPC Target environment is defined by a group of properties. These properties give xPC Target 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 xPC Target, you can specify the environment properties for the host and target computers. Note that you must specify these properties before you can build and download a target application.

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

Note the following:

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

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

   ```
   xpcexplr
   ```

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

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

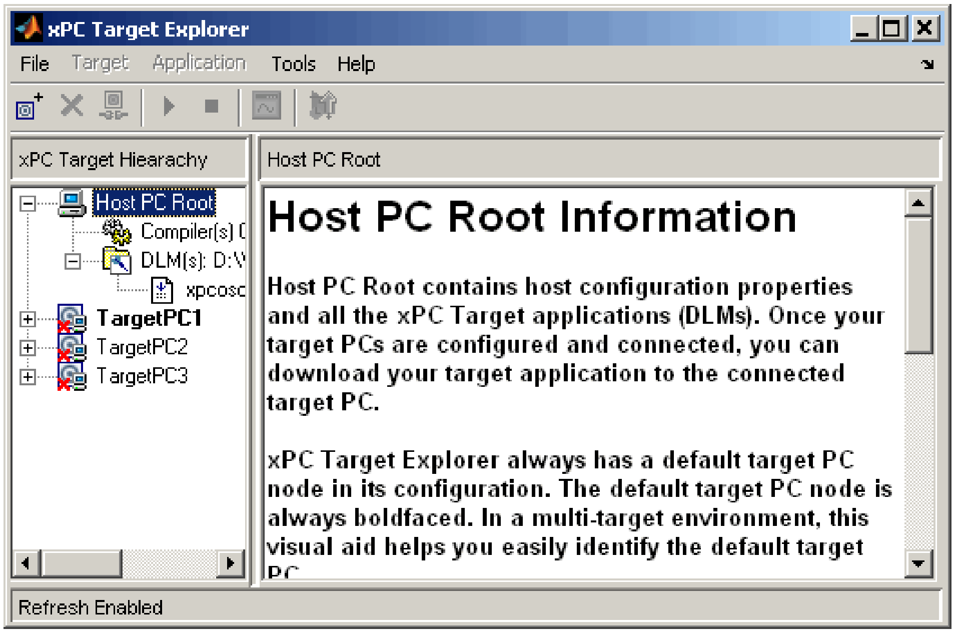
2. In the xPC Target Explorer, right-click the `Host PC` node.

3. Select **Add Target**.

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

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

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



Host PC Root

Host PC Root Information

Host PC Root contains host configuration properties and all the xPC Target applications (DLMs). Once your target PCs are configured and connected, you can download your target application to the connected target PC.

xPC Target Explorer always has a default target PC node in its configuration. The default target PC node is always boldfaced. In a multi-target environment, this visual aid helps you easily identify the default target PC

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

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

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

6 Select `Communication`.

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

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

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

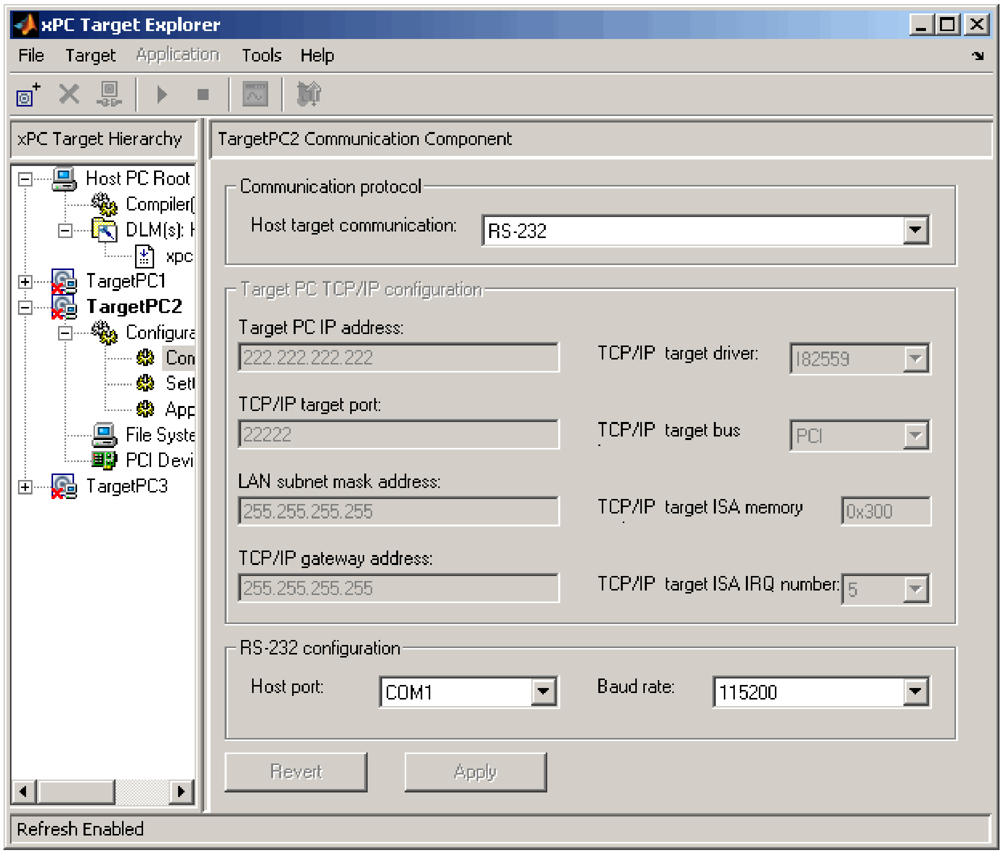
   The pane changes to one that contains only those parameters pertinent to serial communication.

8. From the **Host port** list, select either `COM1` or `COM2` for the connection on the host PC. xPC Target determines the COM port you use on the target PC automatically.

9. From the **Baud rate** list, select the baud rate for the serial connection between the host PC and this target PC. The default is `115200`. Note that for optimal performance, you should select the highest possible serial connection baud rate for xPC Target.

10. Repeat step 5 to step 9 for any target PC for which you have a serial connection between the host PC and target PC.

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



TargetPC2 Communication Component

Communication protocol

Host target communication: RS-232

Target PC TCP/IP configuration

Target PC IP address: 222.222.222.222

TCP/IP target driver: I82559

TCP/IP target port: 22222

TCP/IP target bus: PCI

LAN subnet mask address: 255.255.255.255

TCP/IP target ISA memory: 0x300

TCP/IP gateway address: 255.255.255.255

TCP/IP target ISA IRQ number: 5

RS-232 configuration

Host port: COM1

Baud rate: 115200

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

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

Your next task is to create a target boot disk. See “Target Boot Disk” on page 2-45.

Network Communication

Before you can create and run a target application, you need to set up the connection between the host and target computers. You can use either serial or network communication. This section includes the following topics:

- “Advantages of Network Communication” on page 2-31
- “Hardware for Network Communication” on page 2-32
- “Ethernet Card Provided with xPC Target” on page 2-33
- “Ethernet Chip Families Supported by xPC Target” on page 2-33
- “Ethernet Card for a PCI Bus” on page 2-36
- “Ethernet Card for an ISA Bus” on page 2-37
- “Environment Properties for Network Communication” on page 2-38

For serial communication, see “Serial Communication” on page 2-24.

Advantages of Network Communication

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

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

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

Hardware for Network Communication

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

- Network adapter card — When using xPC Target with TCP/IP, you must have a network adapter card correctly installed on both the host PC and the target PC. Connect the host and target computers with an unshielded twisted pair (UTP) cable to your local area network (LAN). The target PC network adapter card must have a static IP address. The host PC network adapter card can have a Dynamic Host Configuration Protocol (DHCP) address. The host PC can be any computer on the network. When using xPC Target with TCP/IP, you must configure the DHCP server to reserve all static IP addresses to prevent these addresses from being assigned to other systems.

  You can also directly connect your computers. Use a crossover UTP cable with RJ45 connectors to connect them. Both computers must have static IP addresses. If the host PC has a second network adapter card, that card can have a DHCP address.
- I/O boards — If you use I/O boards on your target PC, you need to install the boards correctly.

Ethernet Card Provided with xPC Target

The MathWorks supplies a PCI bus Ethernet card with the xPC Target software for you to use in a desktop target PC. The following table will help to identify the card that was shipped with your software and the parameter you need to select in the xPC Target Explorer. Both cards support a data transfer rate of 100 megabits per second (Mb/s). Note that these cards are equivalent.

Board	Identification	Setup Parameter
Intel Pro/100 M	Intel Pro/100 M or Desktop Adapter	I82559
Intel Pro/100 S	Intel Pro/100 S or Desktop Adapter	I82559

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

   ```
   xpcexplr
   ```

   The xPC Target Explorer window opens.

2. In the xPC Target Explorer **xPC Target Hierarchy** pane, select the `Communication` node of the target PC for which you want to check the boot disk. For example, select the `Communication` node for `TargetPC1`.

3. From the **Host target communication** list, select `TCP/IP`.

4. From the **TcpIp target driver** list, select `I82559`.

If you cannot use the Ethernet card provided with xPC Target, see "Ethernet Chip Families Supported by xPC Target" on page 2-33 for your TCP/IP communication options.

Ethernet Chip Families Supported by xPC Target

If you do not want to use the Ethernet card provided with xPC Target, you can use your own Ethernet card. xPC Target supports Ethernet cards with the following data transfer rates:

- 100 Mbits/second

- 10 Mbits/second

These Ethernet cards must contain an approved Ethernet chip. xPC Target supports the following chip families for 100 Mbits/second Ethernet cards:

- Intel 8255X — Fast Ethernet controller chips and other chips in the Intel 8255X family, including I82559, I82559ER, I82562EM, I82562, I82551, I82551ER, and I82550. In the xPC Target Explorer, and from the **TcpIp target driver** list, select `I82559`.

  Note, the Intel 82801 chip set includes an I82559 compatible Ethernet controller.
- Intel I8254X — Fast Gigabit Ethernet controller chips and other chips in the Intel 8254X family. In the xPC Target Explorer, and from the **TcpIp target driver** list, select `I8254x`.
- AMD79C971 PCNET — Fast Ethernet controller chips and other chips in the AMD 79C97x family. In the xPC Target Explorer, and from the **TcpIp target driver** list, select `RTLANCE`.
- Realtek RTL8139D — In the xPC Target Explorer, and from the **TcpIp target driver** list, select `R8139`.

Note The MathWorks has tested and verified Zonet ZEN3200 and AOpen AON-325.

- 3Com 3C90x — In the xPC Target Explorer, and from the **TcpIp target driver** list, select `3C90x`.
- National Semiconductor DP83815 — In the xPC Target Explorer, and from the **TcpIp target driver** list, select `NS83815`.

10 Mb/s Ethernet cards with the following chips:

- **NE2000** — In the xPC Target Explorer, select `NE2000`.
- **SMC91C9X** —- In the xPC Target Explorer, select `SMC91C9X`.

The following are cases where you might not be able to use the Ethernet card provided with xPC Target:

- You do not have an available PCI slot in your target PC.
- You do not have a PCI bus in your target PC.
- You need to use an Ethernet card other than the card provided with xPC Target.

If one of the above cases applies, purchase one of the boards from the following list. These boards are compatible with xPC Target.

Board Type	Board Number	xPC Target Driver
PCI	SMC EZ Card 10 SMC1208T (RJ45)	NE2000
	SMC EZ Card 10 SMC1208BT (RJ45, BNC)	NE2000
	SMC EZ Card 10 SMC1208BTA (RJ45, BNC, AUI)	NE2000
	Intel PRO/100 S	I82559
	Intel PRO/100 M	I82559
	Zonet ZEN3200	R8139
	AOpen AON-325	R8139
	Boards compatible with 3C90x, except the 3Com 3C905-TX Fast EtherLink XL PCI board	3C90x
	Argon 10/100MB National Semiconductor DP83815 PCI Fast Ethernet Adapter	NS83815
ISA	SMC EZ Card 10 SMC1660T (RJ45)	NE2000
	SMC EZ Card 10 SMC1660BT (RJ45, BNC)	NE2000
	SMC EZ Card 10 SMC1660BTA (RJ45, BNC, AUI)	NE2000

Board Type	Board Number	xPC Target Driver
PC/104	RealTime Devices USA CM202 (RJ45, BNC, AUI)	NE2000
	WinSystems Inc. PCM-NE2000-16 (RJ45)	NE2000
	WinSystems Inc. PCM-NE2000-16-BNC (BNC)	NE2000
SBC	Versalogic VSBC-6	SMC91C9X

Ethernet Card for a PCI Bus

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

To install the PCI bus Ethernet card supplied with the xPC Target software, use the following procedure:

1 Turn off your target PC.

2 If the target PC already has an unsupported Ethernet card, remove the card.

3 Plug the Ethernet card from The MathWorks into a free PCI bus slot.

4 Connect your target PC Ethernet card to your LAN using an unshielded twisted-pair cable.

Your next task is to set up the xPC Target environment for network communication. See "Environment Properties for Network Communication" on page 2-38.

Ethernet Card for an ISA Bus

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

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

The MathWorks does not provide an ISA bus board. For a list of known compatible network adapter cards, see “Hardware for Network Communication” on page 2-32.

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

1 Turn off your target PC.

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

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

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

3 If the target PC already has an unsupported Ethernet card, remove the card. Plug the compatible network card into a free ISA bus slot.

4 Connect the target PC network card to your local area network (LAN) using a coaxial cable or an unshielded twisted-pair cable.

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

Your next task is to set up the xPC Target environment for network communication. See “Environment Properties for Network Communication” on page 2-38.

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

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

   ```
   xpcexplr
   ```

   The xPC Target Explorer window opens.

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

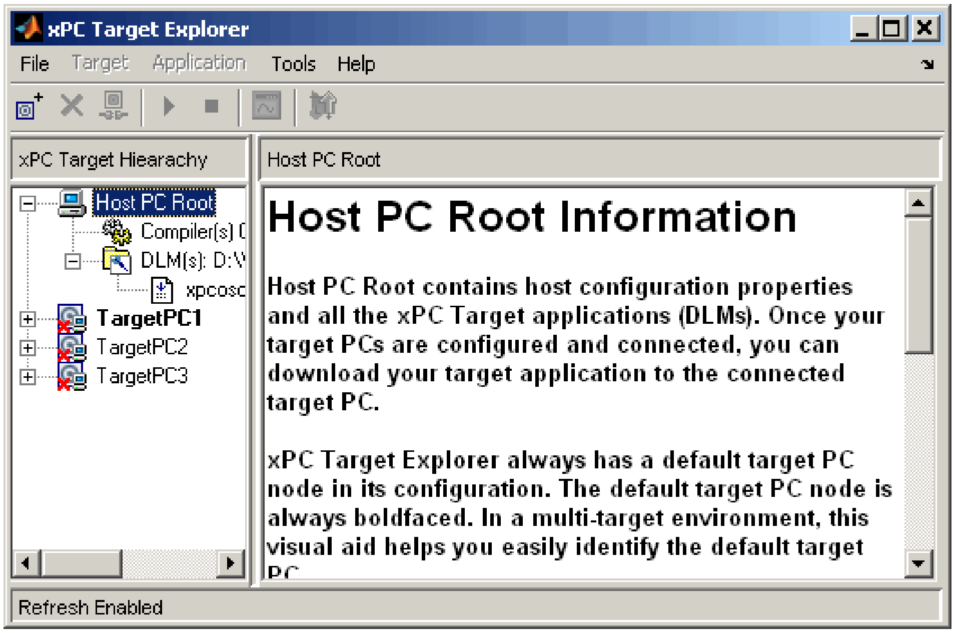
2. In the xPC Target Explorer, right-click the `Host PC` node.

3. Select **Add Target**.

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

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

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



Host PC Root Information

Host PC Root contains host configuration properties and all the xPC Target applications (DLMs). Once your target PCs are configured and connected, you can download your target application to the connected target PC.

xPC Target Explorer always has a default target PC node in its configuration. The default target PC node is always boldfaced. In a multi-target environment, this visual aid helps you easily identify the default target PC

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

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

6 Select `Communication`.

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

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

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

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

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

  Alternatively, you can obtain the LAN subnet mask address from the Network Connections dialog box on your host PC. Depending on your Microsoft 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:

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

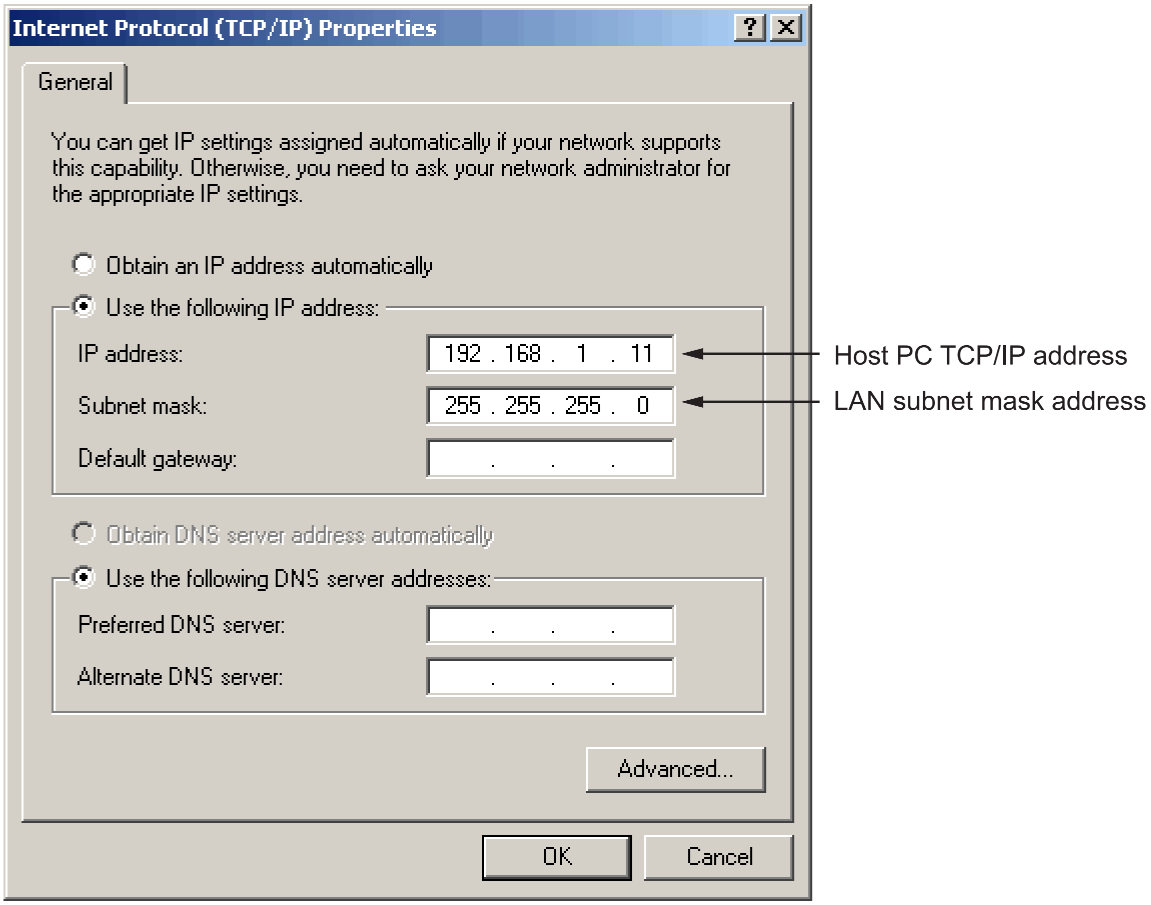
  b Right-click **Local Area Connection**, then select **Properties**.

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

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

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

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



You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

Obtain an IP address automatically [✓ Use the following IP address:]

IP address: 192 . 168 . 1 . 11

Subnet mask: 255 . 255 . 255 . 0

Default gateway: . . .

Obtain DNS server address automatically [✓ Use the following DNS server addresses:]

Preferred DNS server: . . .

Alternate DNS server: . . .

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

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

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

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

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

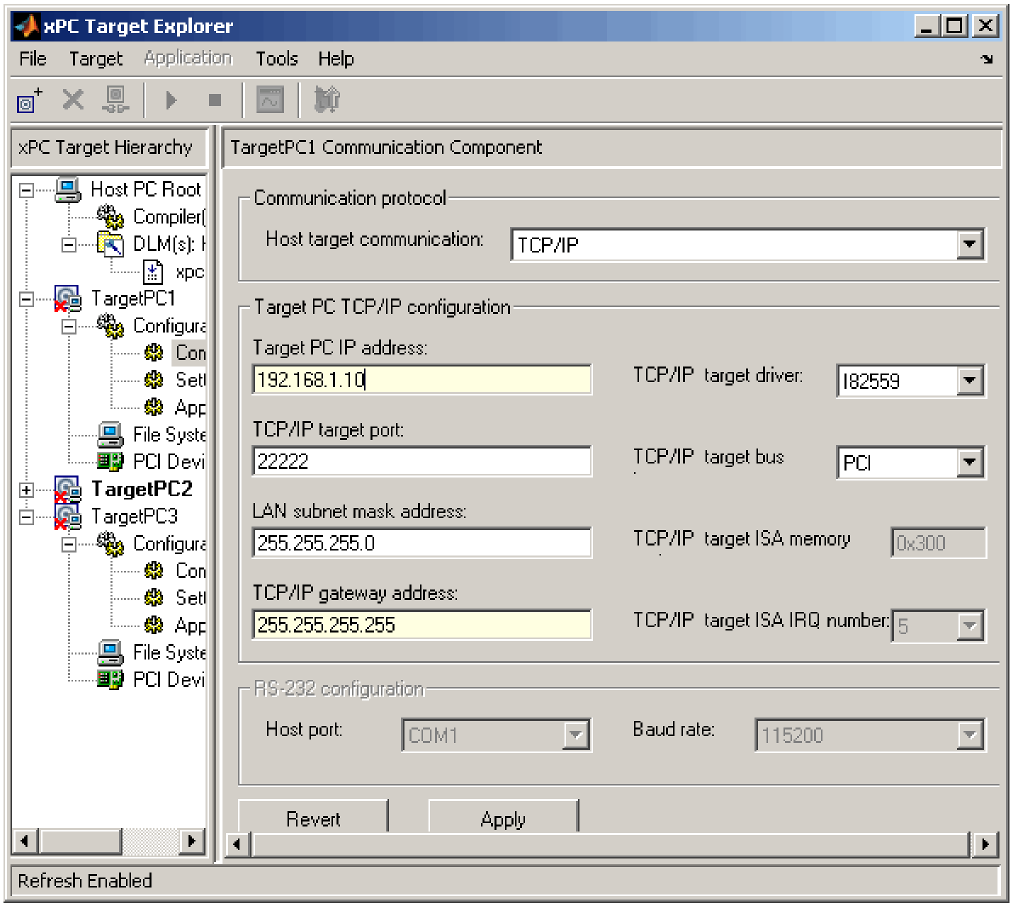
10 Enter the following properties specific to the Ethernet card on your target PC:

- **TcpIp target driver** — From the list, select `NE2000`, `SMC91C9X`, `I82559`, `RTLANCE`, `R8139`, `3C90x`, `NS83815`, or `I8254x`. This property is set by default to `NE2000`. For a crossover cable connection, select `I82559`.
- **TcpIp target bus type** — This property is set by default to PCI. If **TcpIp target bus type** is set to PCI, then the properties **TcpIp target ISA memory port** and **TcpIp target ISA IRQ number** have no effect on TCP/IP communication and are disabled (grayed out). If you are using an ISA bus Ethernet card, set **TcpIp target bus type** to `ISA` and enter values for **TcpIp ISA memory port** and **TcpIp target ISA IRQ number**.
- **TcpIp target ISA memory port** and **TcpIp target ISA IRQ number** — If you are using an ISA bus Ethernet card, you must enter values for the properties **TcpIp target ISA memory port** and **TcpIp 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 Repeat step 5 to 10 for any target PC for which you have a network connection between the host PC and target PC.

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

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



Host target communication: TCP/IP

Target PC IP address: 192.168.1.10

TCP/IP target driver: I82559

TCP/IP target port: 22222

TCP/IP target bus: PCI

LAN subnet mask address: 255.255.255.0

TCP/IP target ISA memory: 0x300

TCP/IP gateway address: 255.255.255.255

TCP/IP target ISA IRQ number: 5

Host port: COM1

Baud rate: 115200

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

Your next task is to create a target boot disk. See “Target Boot Disk” on page 2-45.

Target Boot Disk

The target boot disk includes the xPC Target kernel specific for either serial or network communication. If you installed the xPC Target Embedded Option, and you select stand-alone mode, the target boot disk includes the target application. This section includes the following topics:

- "Creating a Target Boot Disk with a Graphical User Interface" on page 2-45
- "Creating a Target Boot Disk with a Command-Line Interface" on page 2-47
- "Current Properties on the Target Boot Disk" on page 2-48

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

Creating a Target Boot Disk with a Graphical User Interface

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

To create a target boot disk for the current xPC Target environment, use the following procedure. This procedure describes how to create a target boot disk for the target TargetPC1. Alternatively, see "Creating a Target Boot Disk with a Command-Line Interface" on page 2-47.

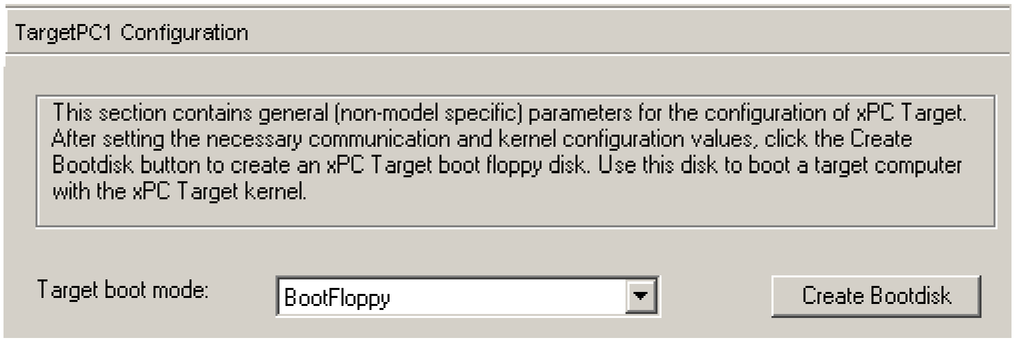
1 If the xPC Target Explorer is not open, open it now. At the MATLAB Command Window, type

```
xpcexplr
```

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

A **TargetBoot** pane appears in the rightmost pane.

3 From the **Target boot mode** list, select a boot disk mode. For example, select `BootFloppy`.

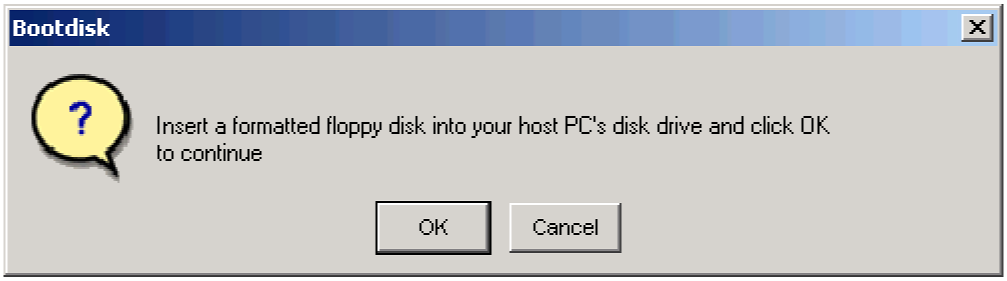

TargetPC1 Configuration

This section contains general (non-model specific) parameters for the configuration of xPC Target. After setting the necessary communication and kernel configuration values, click the Create Bootdisk button to create an xPC Target boot floppy disk. Use this disk to boot a target computer with the xPC Target kernel.

Target boot mode: BootFloppy

4 Click the **Create Bootdisk** button.

The following message box opens.

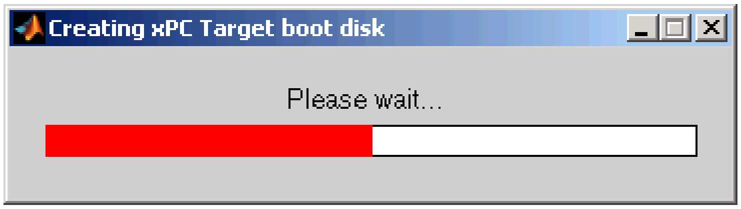

Bootdisk

Insert a formatted floppy disk into your host PC's disk drive and click OK to continue

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

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

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



6 When the 3.5-inch disk drive stops, remove the 3.5-inch disk.

7 Insert the boot disk into your target PC disk drive and reboot that PC.

Your next task is to install the software on the target PC and test your installation. See "Testing and Troubleshooting the Installation" on page 2-50.

Creating a Target Boot Disk with a Command-Line Interface

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

To create a target boot disk for the current xPC Target environment, use the following procedure:

1 In the MATLAB window, type

```
xpcbootdisk
```

xPC Target displays the following message.

```
Insert a formatted floppy disk into your host PC's
disk drive and press any key to continue.
```

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

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

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

Your next task is to install the software on the target PC and test your installation. See "Testing and Troubleshooting the Installation" on page 2-50.

Current Properties on the Target Boot Disk

To check whether a target boot disk corresponds to the current xPC Target environment, use one of the following procedures.

Note If you upgrade your xPC Target software, you need to recreate the target boot disk.

Checking the Target Boot Disk with an xPC Target GUI

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

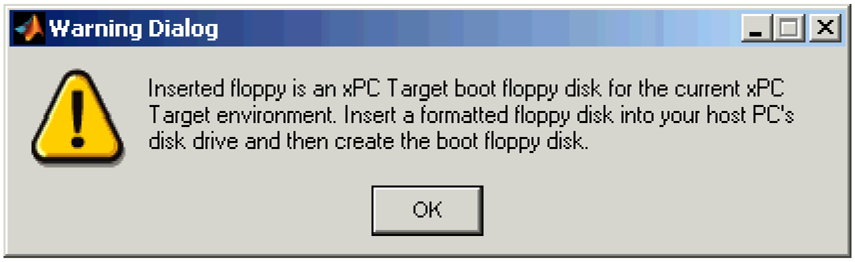
   ```
   xpcexplr
   ```

2. In the xPC Target Explorer **xPC Target Hierarchy** pane, select the `Configuration` node of the target PC for which you want to check the boot disk. For example, select the `Configuration` node for `TargetPC1`.

   A **TargetBoot** pane appears in the rightmost pane.

3. From the **TargetBoot** list, select a boot disk mode. For example, select `BootFloppy`.

   If the boot disk properties correspond to the current properties, the following warning appears and no data is written to the disk.



Inserted floppy is an xPC Target boot floppy disk for the current xPC Target environment. Insert a formatted floppy disk into your host PC's disk drive and then create the boot floppy disk.

Checking the Target Boot Disk with MATLAB Commands

1 Insert the target boot disk into your host PC drive. In the MATLAB window, type

```
xpcbootdisk
```

MATLAB displays the message

```
Insert a formatted floppy disk into your host PC's
disk drive and press a key to continue
```

2 Press any key.

If the boot disk properties correspond to the current properties, MATLAB displays the following message and no data is written to the disk.

```
Inserted floppy is a xPC Target boot floppy disk for the
current xPC Target environment
```

Testing and Troubleshooting the Installation

Use this section to test and troubleshoot connection and communication problems between your host and target computers. This section includes the following topics:

- "Testing the Installation" on page 2-50
- "Test 1, Ping Target System Standard Ping" on page 2-52
- "Test 2, Ping Target System xPC Target Ping" on page 2-54
- "Test 3, Reboot Target Using Direct Call" on page 2-54
- "Test 4, Build and Download Application" on page 2-55
- "If You Need More Help" on page 2-56

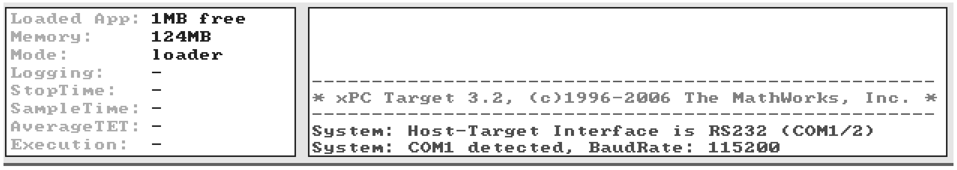
Testing the Installation

This topic describes how to install software on a target PC, boot that PC, and test the installation and connection between the host PC and the target PC. xPC Target uses a test script to test the entire installation. After you install the xPC Target software, set the environment settings, and create a target boot disk, you can test your installation. This procedure assumes that you have set environment settings with xPC Target Explorer. See "Environment Properties for Serial Communication" on page 2-25 or "Environment Properties for Network Communication" on page 2-38.

1 Insert your target boot disk into a target PC disk drive. This target boot disk contains the software to run a target PC.

2 To reboot the target PC, press the reset button on the PC.

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



```
Loaded App: 1MB free
Memory:     124MB
Mode:       loader
Logging:    -
StopTime:   -
SampleTime: -
AverageTET: -
Execution:  -
```

```
-----------------------------------------------------
* xPC Target 3.2, (c)1996-2006 The MathWorks, Inc. *
-----------------------------------------------------
System: Host-Target Interface is RS232 (COM1/2)
System: COM1 detected, BaudRate: 115200
```

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

3 In the MATLAB Current Directory window, select a directory outside the MATLAB root directory.

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

---

4 In the MATLAB Command Window, type

```
xpctest
```

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

```
### xPC Target Test Suite 3.2
### Host-Target interface is: RS232 COM1
### Test 1, Ping target system using standard ping: ... SKIPPED
### Test 2, Ping target system using xpctargetping: ... OK
### Test 3, Reboot target using direct call: ....... OK
### Test 4, Build and download xPC Target application: ... OK
### Test 5, Check host-target communication for commands: .. OK
### Test 6, Download xPC Target application using OOP: ... OK
### Test 7, Execute xPC Target application for 0.2s: ... OK
### Test 8, Upload logged data and compare with simulation:. OK
### Test Suite successfully finished
```

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

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

- “Test 1, Ping Target System Standard Ping” on page 2-52

- "Test 2, Ping Target System xPC Target Ping" on page 2-54
- "Test 3, Reboot Target Using Direct Call" on page 2-54
- "Test 4, Build and Download Application" on page 2-55

Test 1, Ping Target System Standard Ping

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

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

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

DOS should display a message similar to the following:

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

2 Check the messages on your screen.

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

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

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

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

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

To solve this problem, in the MATLAB window, type

```
xpcexplr
```

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

For a PCI bus,

- Check that **TcpIp target bus type** is set to `PCI` instead of `ISA`.

For an ISA bus,

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

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

To solve this problem, in the MATLAB window, type

```
xpcexplr
```

For the problem target PCs, enter the correct addresses. Recreate the target boot disk by inserting a floppy disk into the host disk drive, and then clicking the **Create BootDisk** button.

See also “xpctest: Test 1 Fails” in “Troubleshooting” in the xPC Target User’s Guide. If you still cannot solve your problem, see “If You Need More Help” on page 2-56.

Test 2, Ping Target System xPC Target Ping

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

1 In the MATLAB window, type

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

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

2 Check the messages in the MATLAB window.

MATLAB should respond with the following messages:

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

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

To solve this problem, create another target boot disk with a new floppy disk. See "Target Boot Disk" on page 2-45.

See also "xpctest: Test 2 Fails" in "Troubleshooting" in the xPC Target User's Guide. If you still cannot solve your problem, see "If You Need More Help" on page 2-56.

Test 3, Reboot Target Using Direct Call

This test tries to boot your target computer using an xPC Target command. If this test fails, try troubleshooting with the following procedure. This procedure assumes that you have set environment settings with xPC Target Explorer. See "Environment Properties for Serial Communication" on page 2-25 or "Environment Properties for Network Communication" on page 2-38.

1 In the MATLAB window, type

```
xpctest noreboot
```

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

```
### Test 3, Reboot target using direct call: ... SKIPPED
```

2 Observe the messages in the MATLAB window during the build process.

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

See also “xpctest: Test 3 Fails” in “Troubleshooting” in the xPC Target User’s Guide. If you still cannot solve your problem, see “If You Need More Help” on page 2-56.

Test 4, Build and Download Application

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

1 In the MATLAB window, check the error messages.

These messages help you locate where there is a problem.

2 If you get the error message

```
xPC Target loader not ready
```

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

See also “xpctest: Test 4 Fails” in “Troubleshooting” in the xPC Target User’s Guide. If you still cannot solve your problem, see “If You Need More Help” on page 2-56.

If You Need More Help

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

Internet	To contact The Mathworks Technical Support, use this form `http://www.mathworks.com/contact_TS.html`
Telephone	508-647-7000 Ask for Technical Support.

Exporting and Importing xPC Target Explorer Environments

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

This topic describes how to export target PC node property settings in a structured format to the MATLAB workspace. It assumes that you have set environment settings with xPC Target Explorer. See "Environment Properties for Serial Communication" on page 2-25 or "Environment Properties for Network Communication" on page 2-38.

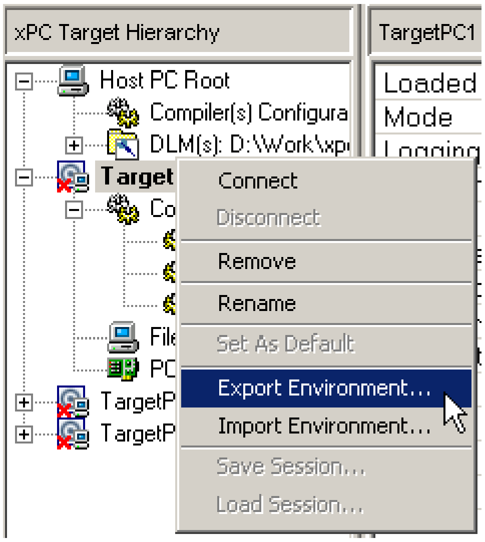
1. If the xPC Target Explorer is not open, open it now. At the MATLAB Command Window, type

   ```
   xpcexplr
   ```

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

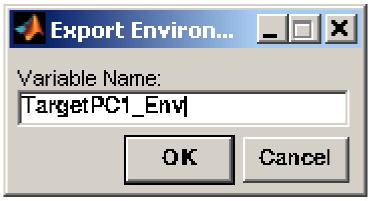
   A context-sensitive menu is displayed.

3. Select **Export Environment**.



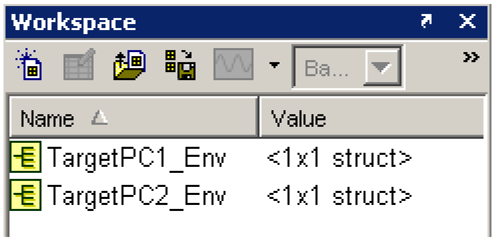
The Export Environment to Workspace dialog is displayed.

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



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

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



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

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

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

```
save targetpc1.mat TargetPC1env
```

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

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

```
load targetpc1.mat
```

3 If the xPC Target Explorer is not open, open it now.

4 In the xPC Target Explorer **xPC Target Hierarchy** pane, right-click the target PC node for which you want to import the configuration. For example, right-click `TargetPC1`.

A context-sensitive menu is displayed.

5 Select **Import Environment**.

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

Basic Tutorial

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

Simulink Model (p. 3-2)	Create a simple Simulink model with a square wave input and output blocks for tracing and logging signal data
Simulating the Model (p. 3-38)	Check the behavior of your Simulink model by simulating the model in nonreal time
xPC Target Application (p. 3-44)	Create a boot disk and boot the target PC, create a target application, and download the target application to a target PC
Running the Target Application (p. 3-60)	Test the behavior of your target application by running the application in real time
Menu Bar and Toolbar Contents and Shortcut Keys (p. 3-74)	Learn about the xPC Target Explorer menu bar and toolbar items

Simulink Model

Before you can create a target application, you need to create a Simulink model. xPC Target then uses the Simulink model, Real-Time Workshop, and a third-party compiler to create the target application. This section includes the following topics:

Creating a Simple Simulink Model (p. 3-2)	Create a simple second-order system with a sine wave input.
Entering Parameters for the Scope Block (p. 3-6)	Enter parameters for the Scope block
Adding a Simulink Outport Block (p. 3-10)	Add an Outport block to log signal data to the MATLAB workspace for analysis.
Entering Parameters for the Outport Blocks (p. 3-13)	Enter and select parameters in the Configuration Parameters dialog box.
Adding an xPC Target Scope Block (p. 3-17)	Add an xPC Target Scope block to visualize signals while running the target application.
Entering Parameters for an xPC Target Scope Block (p. 3-21)	Enter scope parameters in the Block Parameters dialog box before building the target application.

Creating a Simple Simulink Model

This tutorial uses a simple Simulink model to explain the tasks you need to do with 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
```

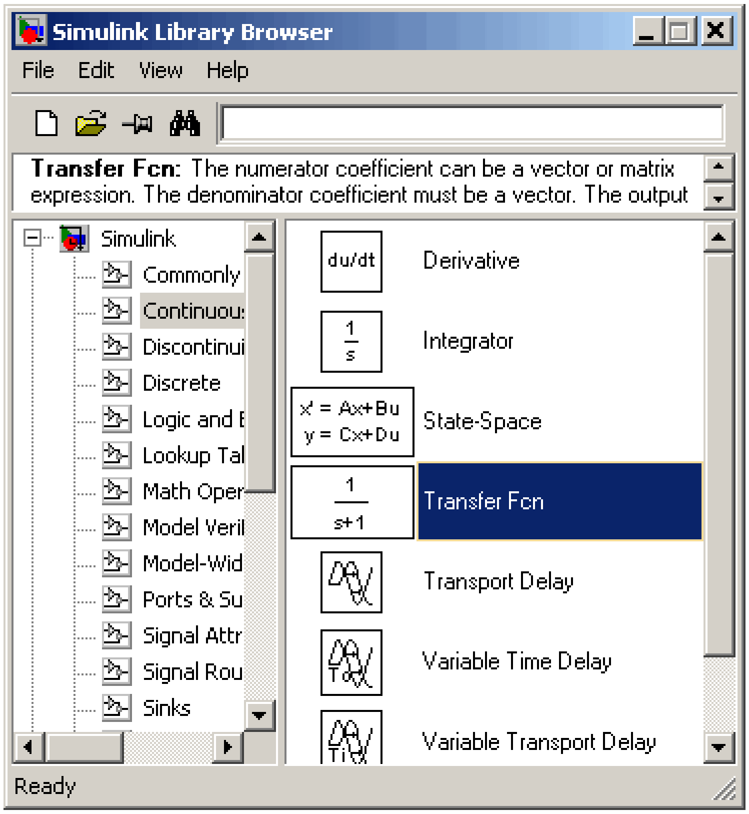
The Simulink library window opens.

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.



Transfer Fcn: The numerator coefficient can be a vector or matrix expression. The denominator coefficient must be a vector. The output

4 Click and drag the Transfer Fcn block to the Simulink model window.

5 Adjust the Numerator and the Denominator values to reflect the example.

6 In the Simulink Library Browser window, click and drag the following blocks to your model.

- Click **Sources**, and add a Signal Generator block.
- Click **Sinks**, and add a Scope block.
- Click **Signal Routing**, and add a Mux block.

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

---

7 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

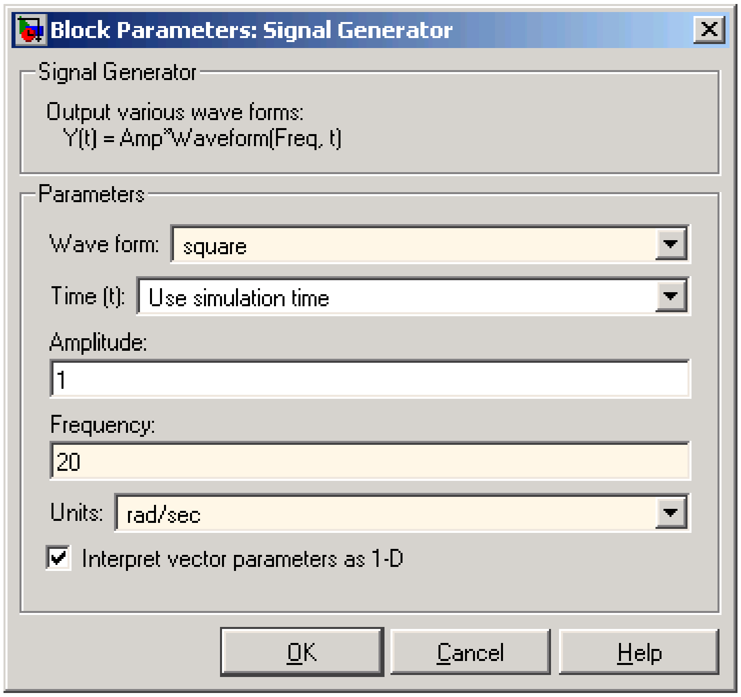
```
1
```

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.


Block Parameters: Signal Generator

Signal Generator

Output various wave forms:
  Y(t) = Amp*Waveform(Freq, t)

Parameters

Wave form: square

Time (t): Use simulation time

Amplitude:
1

Frequency:
20

Units: rad/sec

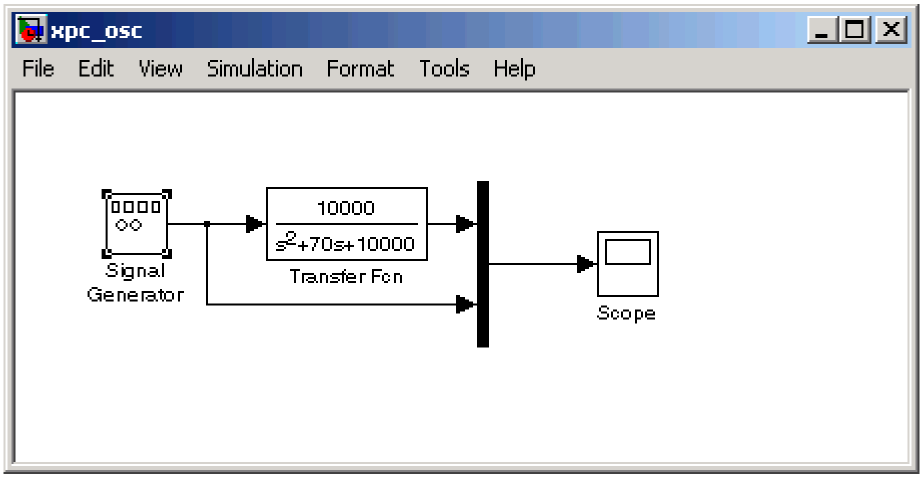
[✓] Interpret vector parameters as 1-D

8 Double-click the Transfer Fcn block.

9 Edit the **Numerator** and **Denominator** parameters to match those in the following figure.

10 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.



11 From the **File** menu, click **Save As** and enter a filename. For example, enter `xpc_osc` and then click **OK**.

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

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

Entering Parameters for the Scope Block

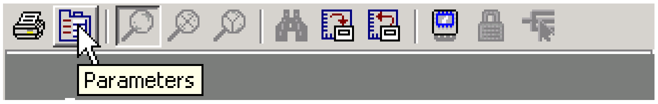
You enter or change scope parameters to specify the *x*-axis and *y*-axis in a Scope window. Other properties include the number of graphs in one Scope window and the sample time for models with discrete blocks.

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.



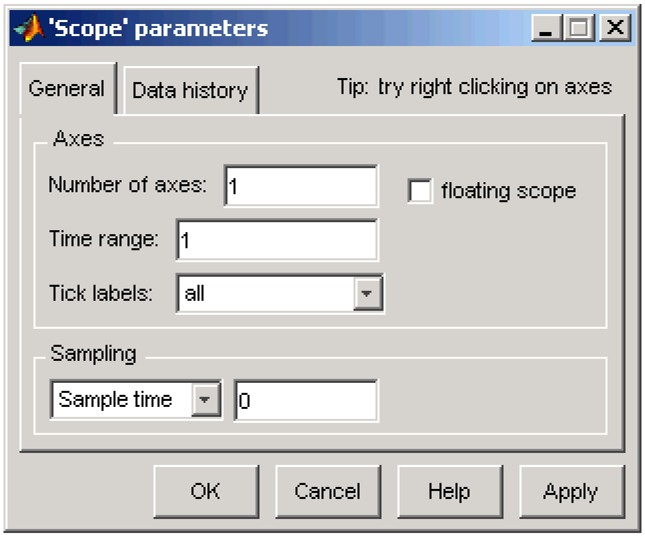
A Scope parameters dialog box opens.

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.


'Scope' parameters

Tip: try right clicking on axes

Axes

Number of axes: 1

floating scope

Time range: 1

Tick labels: all

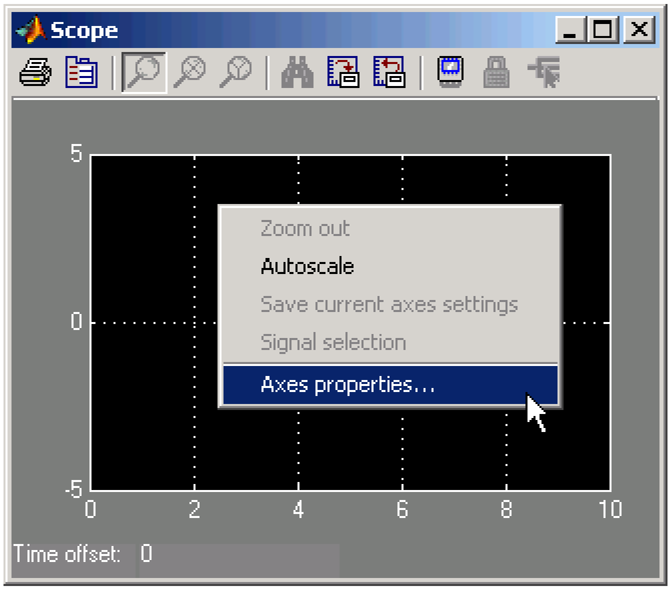
Sampling

Sample time 0

4 Do one of the following:

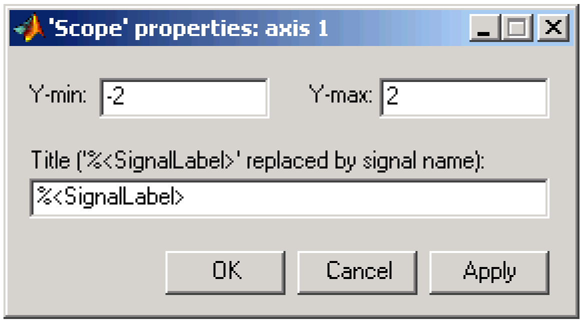
- Click **Apply** to apply the changes to your model and leave the dialog box open.
- Click **OK** to apply the changes to your model and close the Scope parameters dialog box.

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.


'Scope' properties: axis 1

Y-min: -2 Y-max: 2

Title ('%<SignalLabel>' replaced by signal name): %<SignalLabel>

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

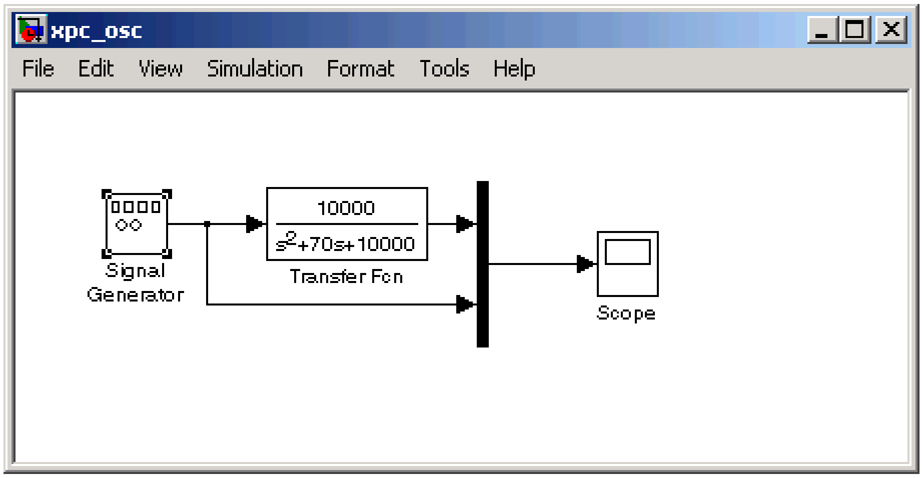
If you want to log signal data to the MATLAB workspace for analysis and later save that data to a disk, you need to add a Simulink Outport block and activate logging from the Configuration Parameters dialog box.

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

1 In the MATLAB window, type

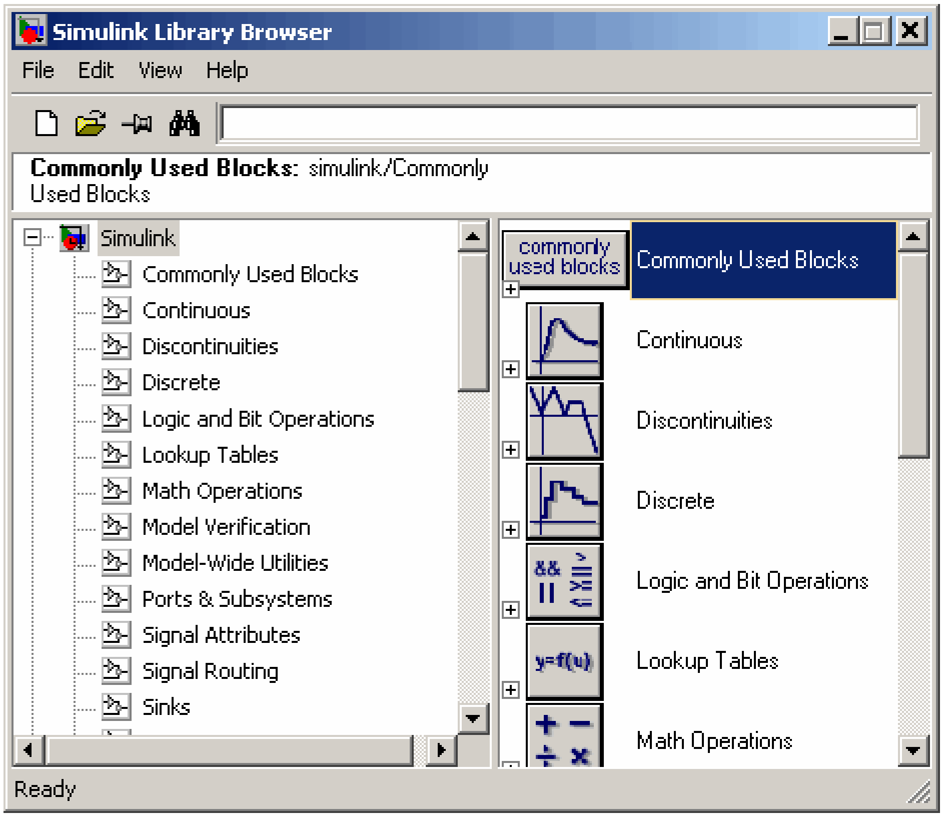
```
xpc_osc
```

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



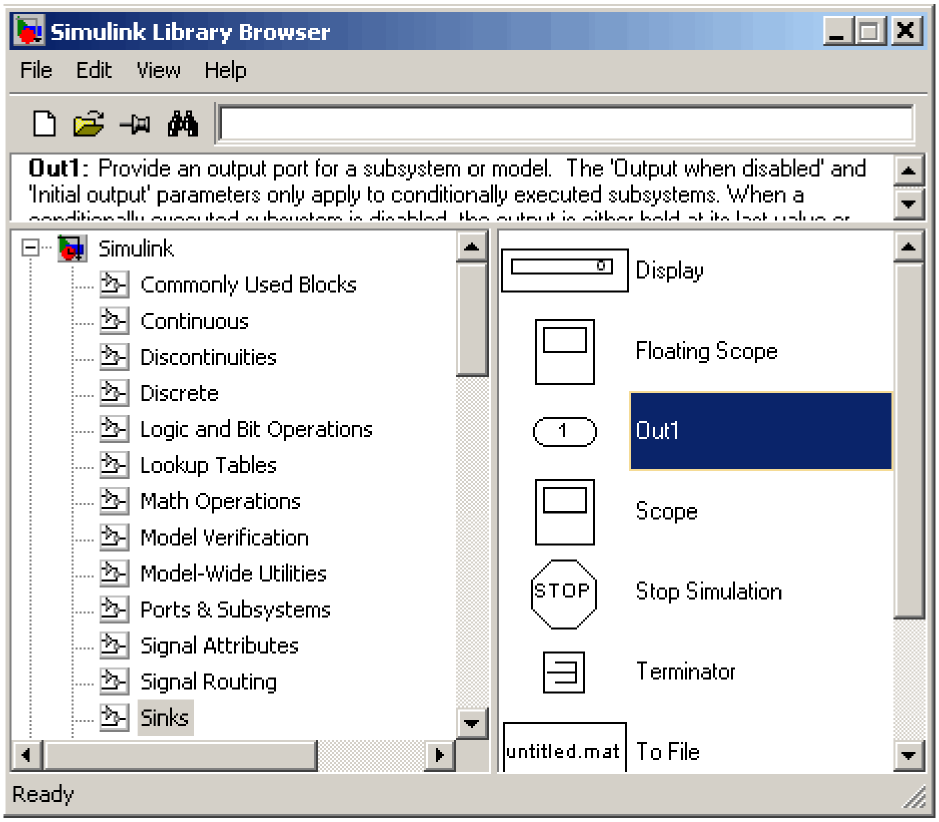
2 In the Simulink window, from the **View** menu, click **Library Browser**.

The Simulink Library Browser window opens.



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

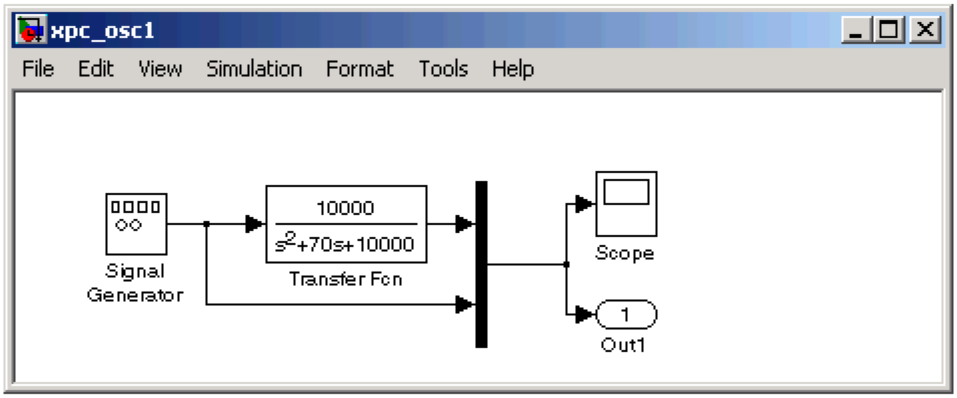
In the right pane, the browser shows a list of sink blocks.



Out1: Provide an output port for a subsystem or model. The 'Output when disabled' and 'Initial output' parameters only apply to conditionally executed subsystems. When a

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 `xpc_osc1` and then click **OK**.

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

Entering Parameters for the Outport Blocks

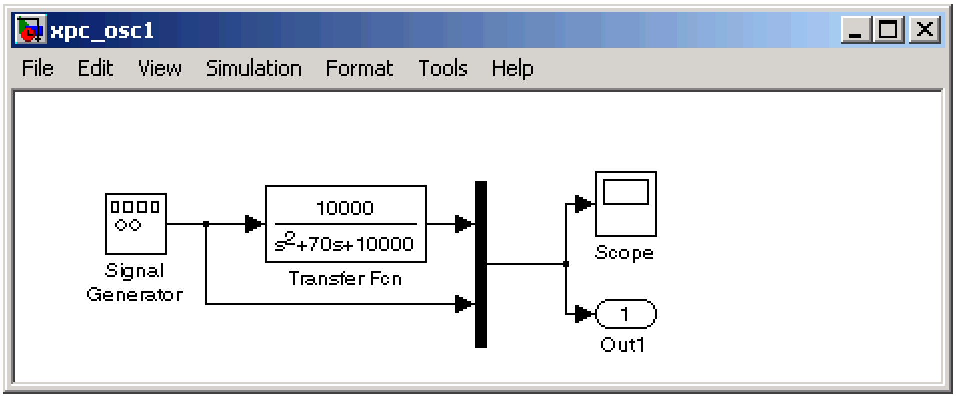
During a simulation, Simulink saves signal data to MATLAB variables using Outport blocks. The default MATLAB variables are `tout`, `xout`, and `yout`. While running a real-time application, xPC Target uses these same variables to pass signal data to target object parameters. A target object is a structure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

After you add an Outport block to your Simulink model, you can enter parameters. This procedure uses the model `xpc_osc1.mdl` with an Outport block as an example. To add an Outport block, see “Adding a Simulink Outport Block” on page 3-10.

1 In the MATLAB window, type

```
xpc_osc1
```

A Simulink window with the model `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`. Real-Time Workshop does not support variable-step solvers.

The **Solver** pane dialog changes.

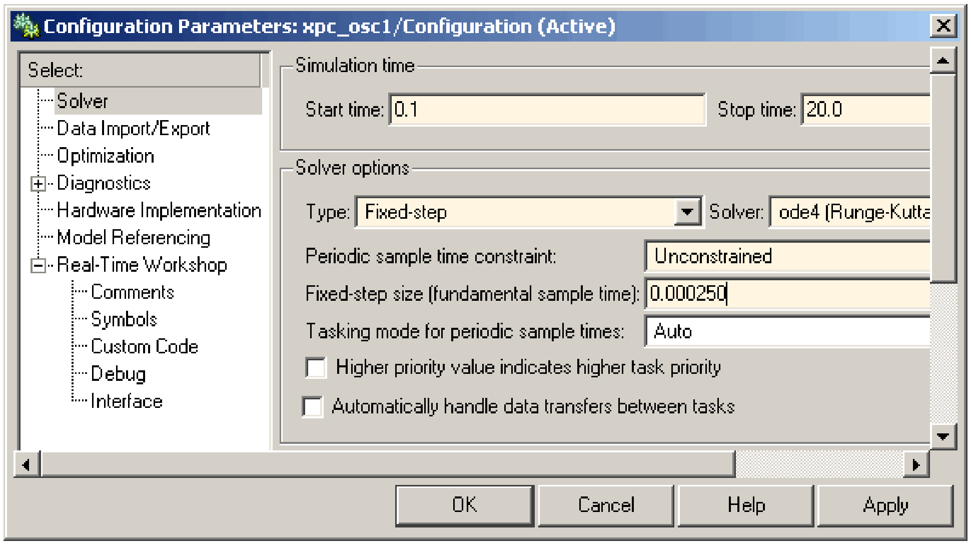
6 From the **Solver** list, select a solver. For example, select the general-purpose solver `ode4 (Runge-Kutta)`.

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

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

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

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


Configuration Parameters: xpc_osc1/Configuration (Active)

Simulation time
Start time: 0.1
Stop time: 20.0

Solver options
Type: Fixed-step
Solver: ode4 (Runge-Kutta
Periodic sample time constraint: Unconstrained
Fixed-step size (fundamental sample time): 0.000250
Tasking mode for periodic sample times: Auto
☐ Higher priority value indicates higher task priority
☐ Automatically handle data transfers between tasks

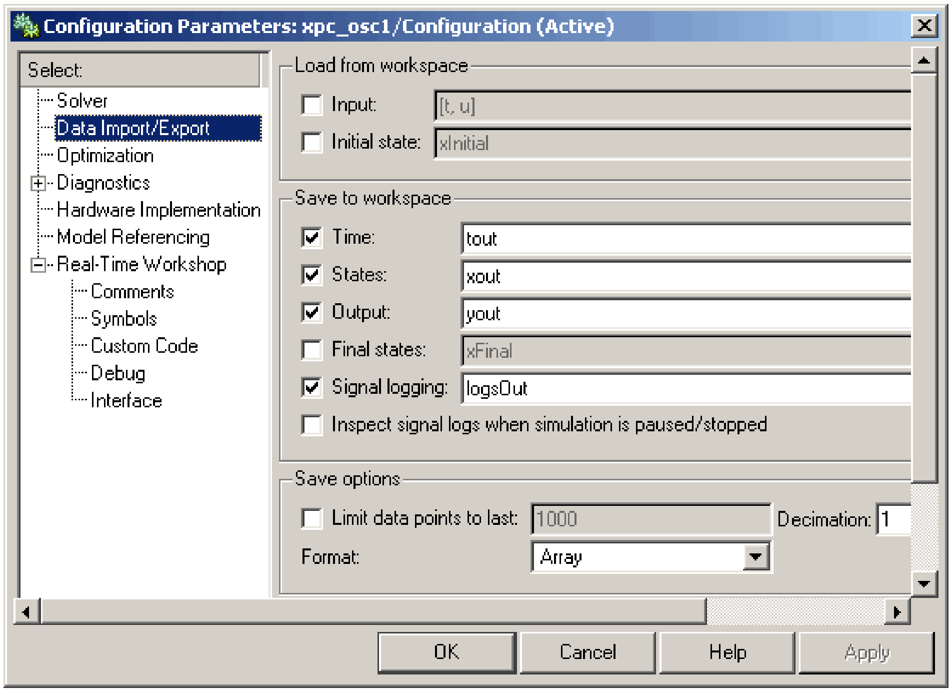
8 Click the **Data Import/Export** node.

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.


Configuration Parameters: xpc_osc1/Configuration (Active)

Select:
- Solver
- Data Import/Export
- Optimization
- Diagnostics
- Hardware Implementation
- Model Referencing
- Real-Time Workshop
  - Comments
  - Symbols
  - Custom Code
  - Debug
  - Interface

Load from workspace
- [ ] Input: [t, u]
- [ ] Initial state: xInitial

Save to workspace
- [✓] Time: tout
- [✓] States: xout
- [✓] Output: yout
- [ ] Final states: xFinal
- [✓] Signal logging: logsOut
- [ ] Inspect signal logs when simulation is paused/stopped

Save options
- [ ] Limit data points to last: 1000 Decimation: 1
- Format: Array

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

- **Many states** — If your model contains many states (for example, more than 20 states), the storage of the state vector requires a lot of target memory. If you clear the **States** check box, logging of states is turned off and more memory is available for the target application. An alternative to logging all the state signals is to select individual states of interest by adding Outport blocks to your model.

- **Small sample time** — If you choose a very short sample time, this setting might overload the CPU. If you clear the **Save to workspace** check boxes, logging is turned off and more computing time is available for calculating the model.

1 Click **OK**.

2 From the **File** menu, click **Save**. The model is saved as `xpc_osc1.mdl.`

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

Adding an xPC Target Scope Block

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

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

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

After you create a Simulink model, you can add an xPC Target Scope block. The following procedure uses the Simulink model `xpc_osc1.mdl` as an example to show how to connect an xPC Target Scope block to your model.

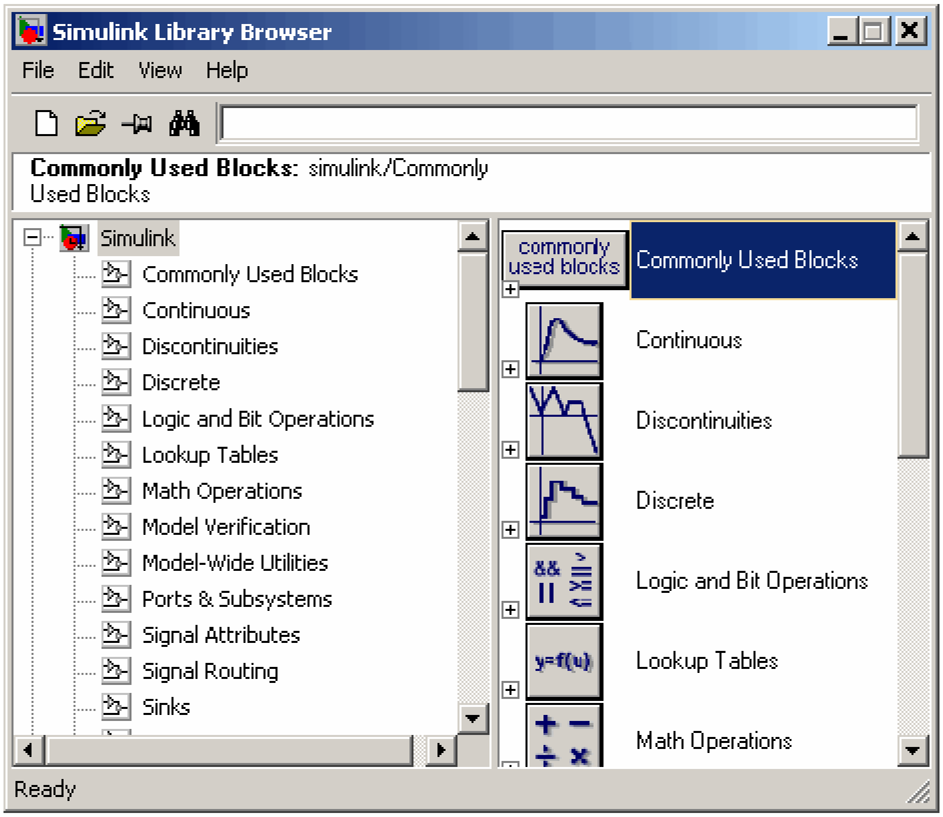
1 In the MATLAB window, type

```
xpc_osc1
```

The Simulink block diagram opens for the model `xpc_osc1.mdl`.

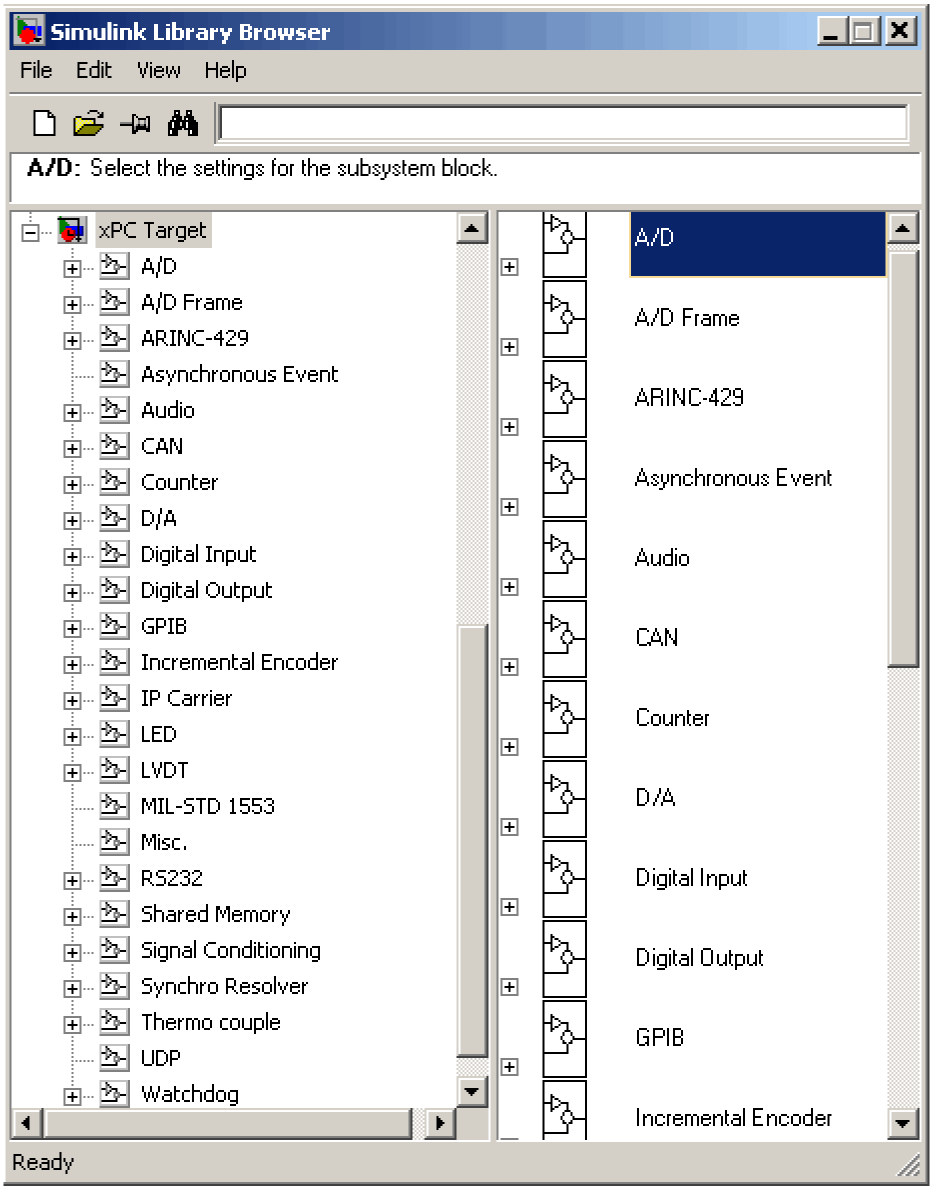
2 In the Simulink window, from the **View** menu, click **Library Browser**.

The Simulink Library Browser window opens.



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

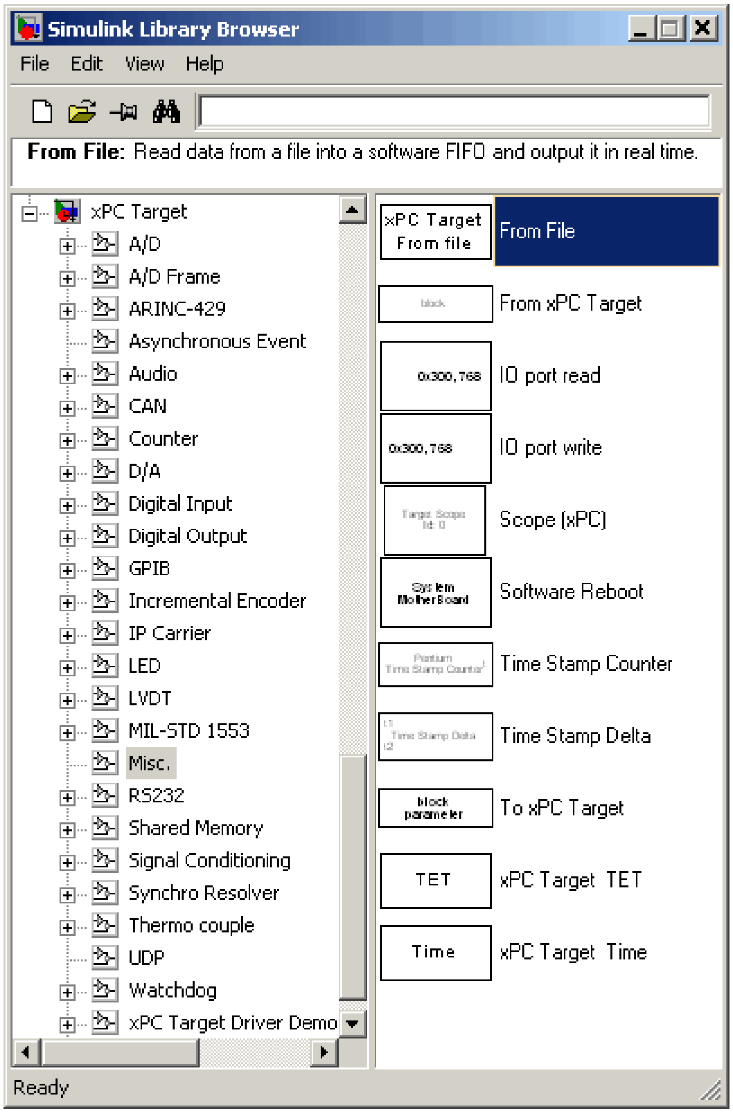
A list of I/O functions opens.



A/D: Select the settings for the subsystem block.

4 Click **Misc**.

A list of miscellaneous group blocks opens.



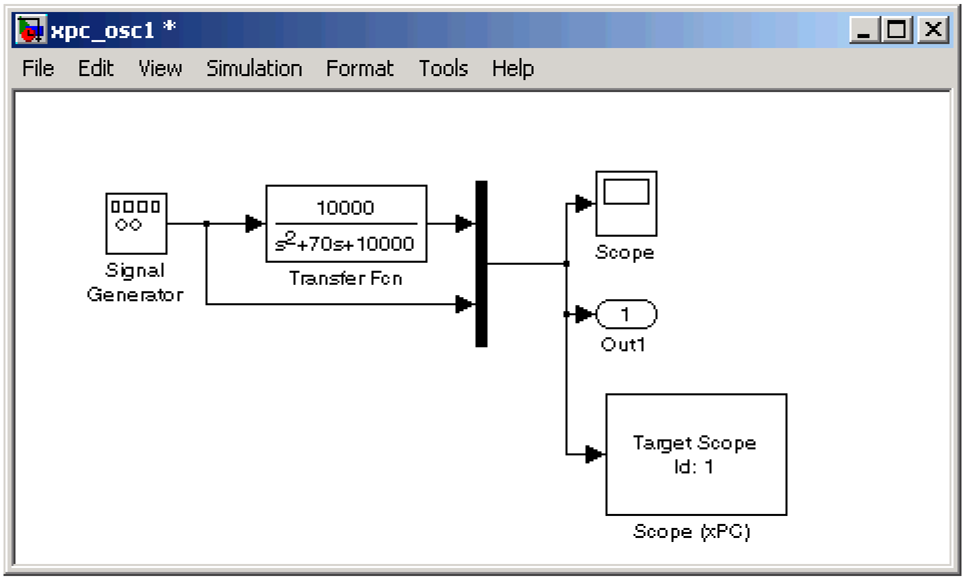
From File: Read data from a file into a software FIFO and output it in real time.

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 `xpc_osc1.mdl` should look like the figure shown.



7 From the **File** menu, click **Save As**. Enter a filename. For example, enter `xpc_osc2` and then click **OK**.

Your next task is to define the xPC Target Scope block parameters. See "Entering Parameters for an xPC Target Scope Block" on page 3-21.

Entering Parameters for an xPC Target Scope Block

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

This section describes how you can configure the xPC Target Scope block for other scope behavior.

After you add an xPC Target Scope block to your Simulink model, you can enter parameters for this block. To add an xPC Target Scope block, see "Adding an xPC Target Scope Block" on page 3-17. To enter the parameters for an xPC Target Scope block to write signal data to a file on the target PC, see "Entering Parameters for an xPC Target Scope of Type File" on page 3-32.

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

- "Entering Parameters for an xPC Target Scope of Type Target" on page 3-22
- "Entering Parameters for an xPC Target Scope of Type Host" on page 3-28
- "Entering Parameters for an xPC Target Scope of Type File" on page 3-32

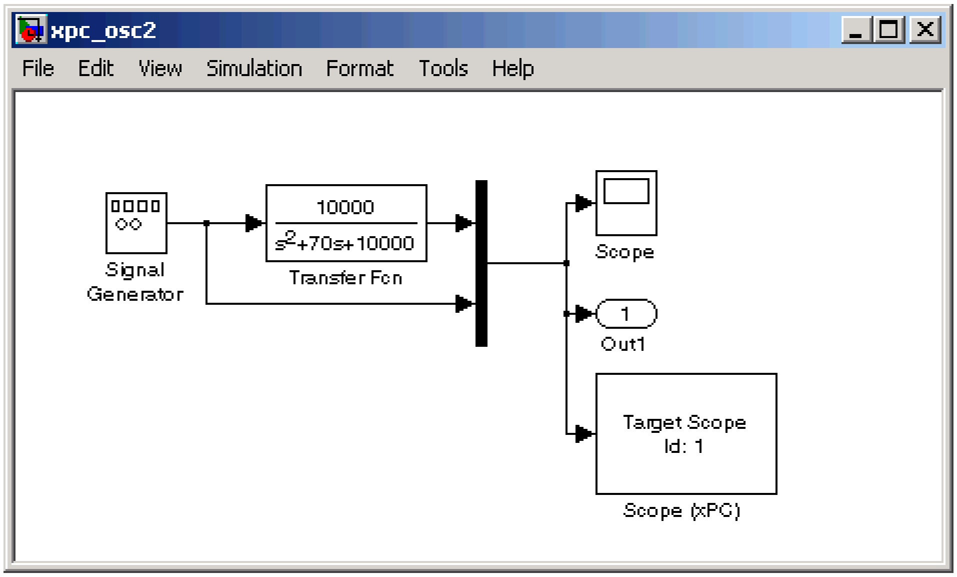
Entering Parameters for an xPC Target Scope of Type Target

This procedure uses the model `xpc_osc2.mdl` as an example.

1 In the MATLAB window, type

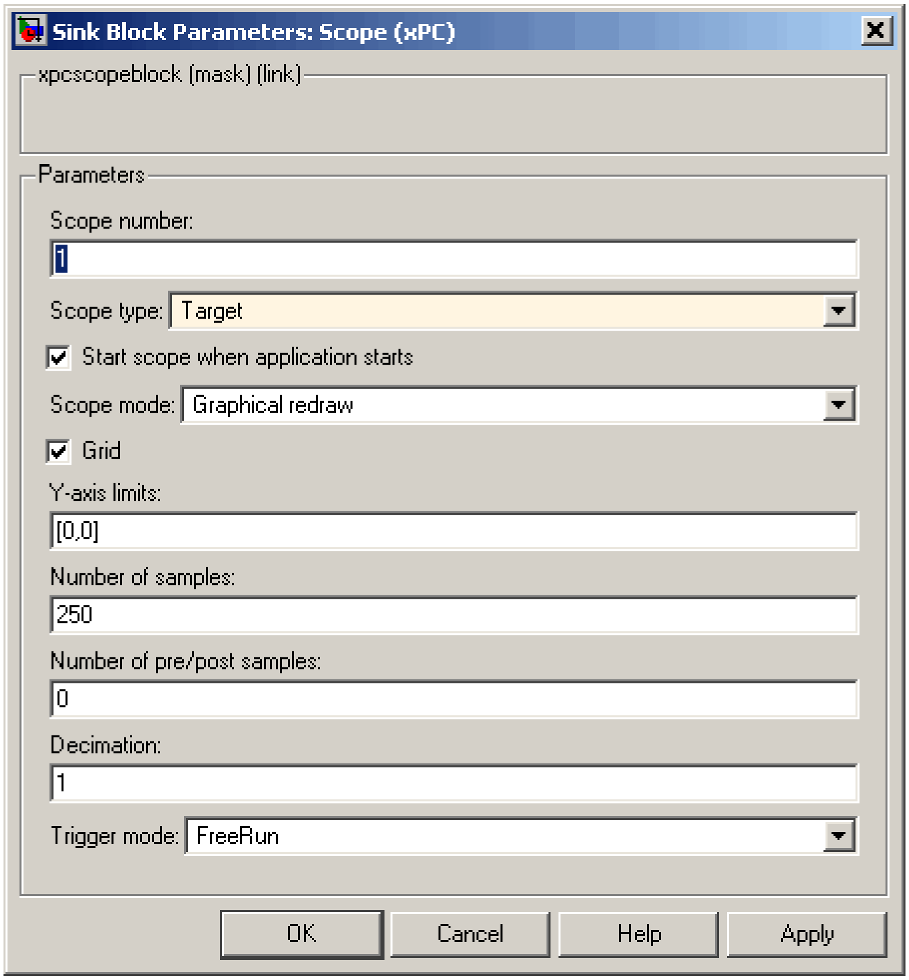
```
xpc_osc2
```

The Simulink block diagram opens for the model `xpc_osc2.mdl`.


xpc_osc2
File Edit View Simulation Format Tools Help
Signal Generator
10000
s2+70s+10000
Transfer Fcn
Scope
1
Out1
Target Scope
Id: 1
Scope (xPC)

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

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


Sink Block Parameters: Scope (xPC)

xpcscopeblock (mask) (link)

Parameters

Scope number: 1

Scope type: Target

[✓] Start scope when application starts

Scope mode: Graphical redraw

[✓] Grid

Y-axis limits: [0,0]

Number of samples: 250

Number of pre/post samples: 0

Decimation: 1

Trigger mode: FreeRun

By default, the scope of type `Target` 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, xPC Target 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] [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, xPC Target 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 scopes of type `Target` and 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 scopes of type `Target` and 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.

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 `xpc_osc2.mdl`.

Your next task is to simulate the model. See "Simulating the Model" on page 3-38.

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

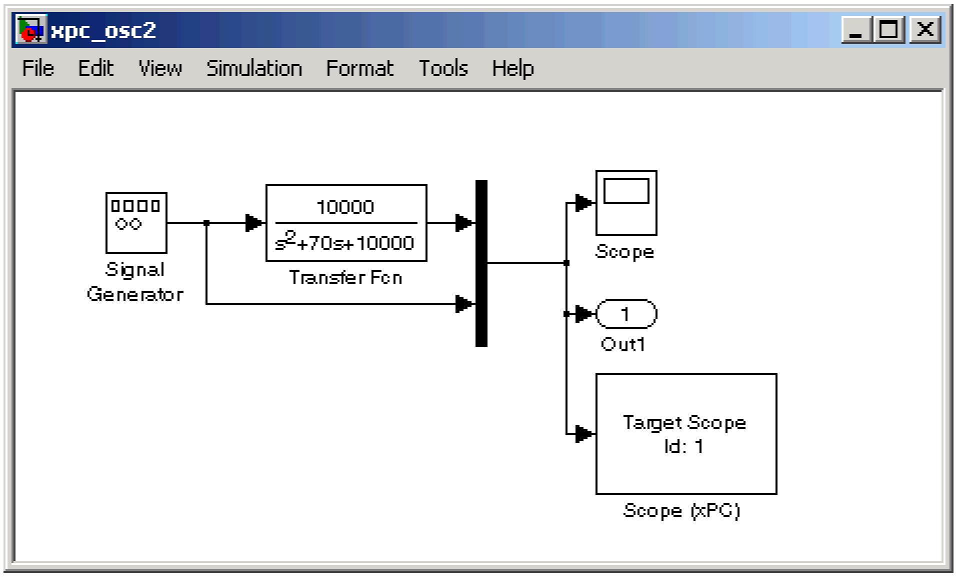
Entering Parameters for an xPC Target Scope of Type Host

This procedure uses the model `xpc_osc2.mdl` as an example.

1 In the MATLAB window, type

```
xpc_osc2
```

The Simulink block diagram opens for the model `xpc_osc2.mdl`.



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

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

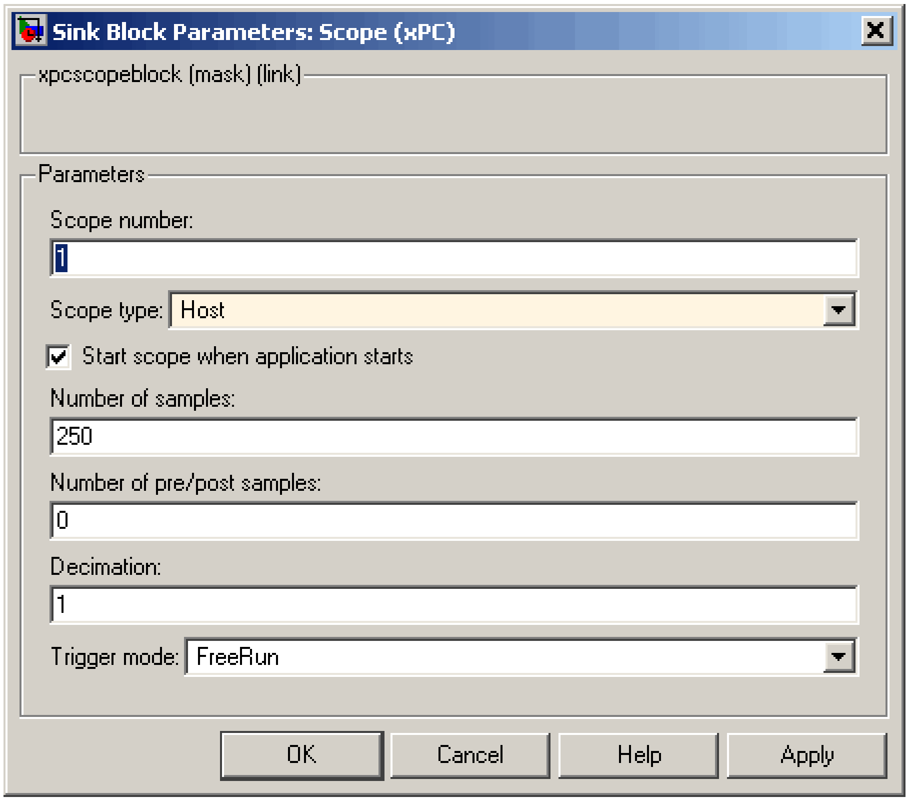
By default, the scope of type target dialog is displayed.

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.



Sink Block Parameters: Scope (xPC)

xpcscopeblock (mask) (link)

Parameters

Scope number: 1

Scope type: Host

[✓] Start scope when application starts

Number of samples: 250

Number of pre/post samples: 0

Decimation: 1

Trigger mode: FreeRun

5 Select the **Start scope when application starts** check box to start a scope when the target application is downloaded and started. With a scope of type `target`, the scope window opens automatically. With a scope of type `host`, 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 `xpc_osc2.mdl`.

Your next task is to simulate the model. See "Simulating the Model" on page 3-38.

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

Entering Parameters for an xPC Target Scope of Type File

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

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

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

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

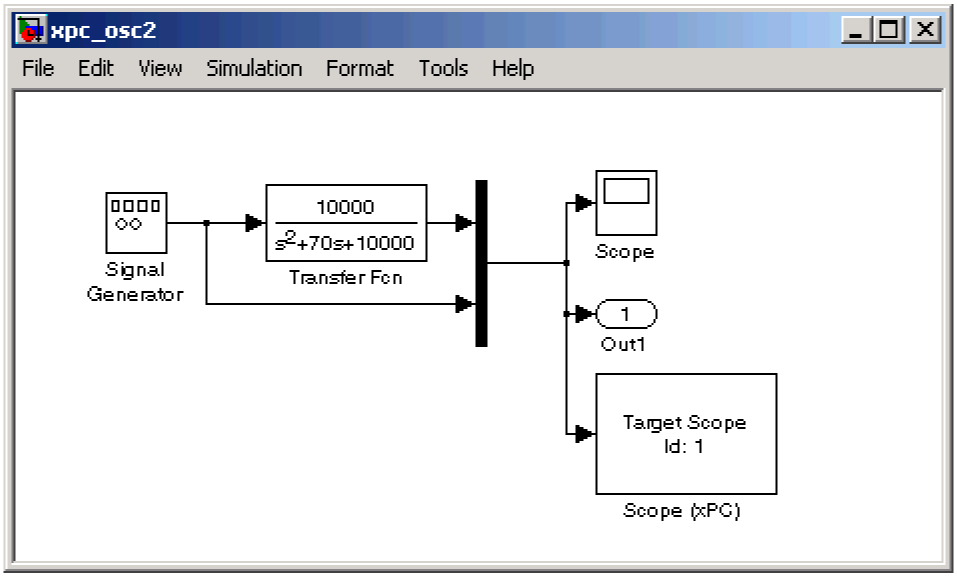
To add an xPC Target Scope block, see “Adding an xPC Target Scope Block” on page 3-17.

This procedure uses the model `xpc_osc2.mdl` as an example.

1 In the MATLAB window, type

```
xpc_osc2
```

The Simulink block diagram opens for the model `xpc_osc2.mdl`.



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

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

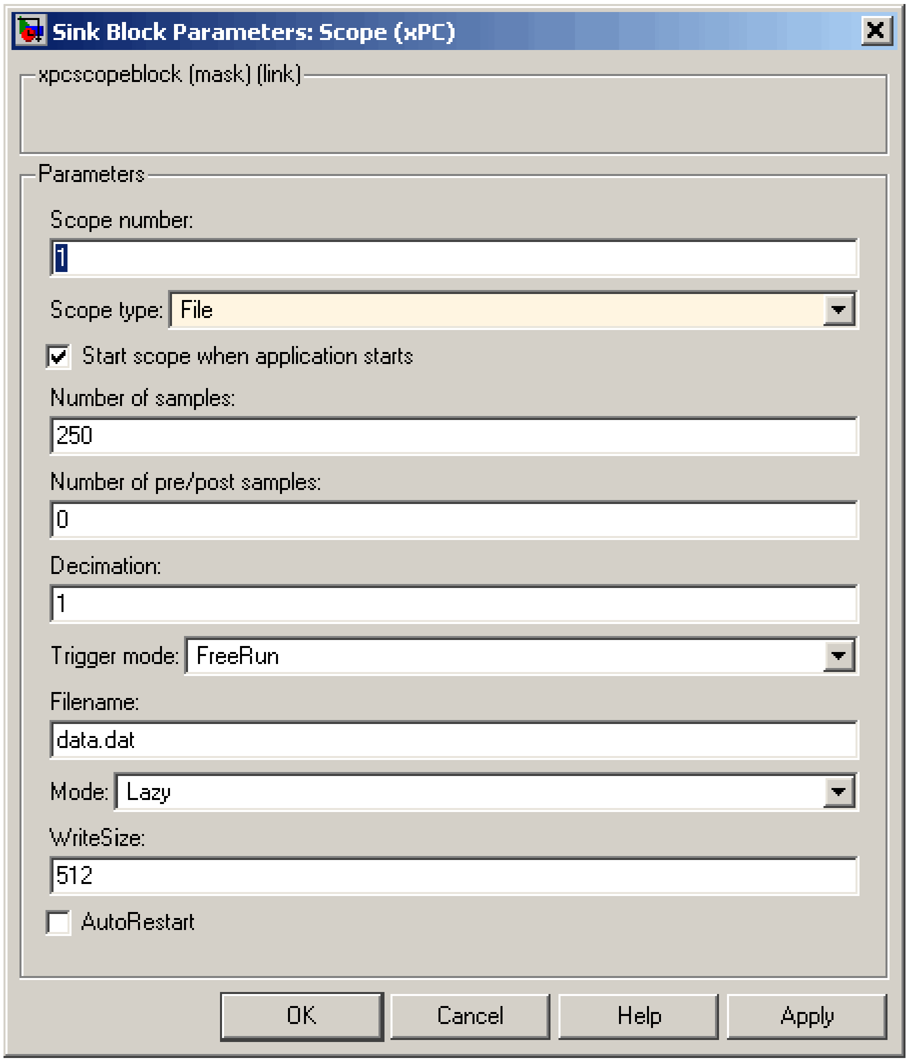
By default, the scope of type `Target` dialog is displayed.

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.



Sink Block Parameters: Scope (xPC)

xpcscopeblock (mask) (link)

Parameters

Scope number:
1

Scope type: File

[✓] Start scope when application starts

Number of samples:
250

Number of pre/post samples:
0

Decimation:
1

Trigger mode: FreeRun

Filename:
data.dat

Mode: Lazy

WriteSize:
512

[ ] AutoRestart

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 scope of type file collects data up to **Number of samples**, then starts over again, overwriting the buffer. If the **AutoRestart** box is not selected, the scope of type file collects data only up to **Number of samples**, then stops.

7 In the **Number of pre/post samples** box, enter the number of samples to save or skip. Specify a value less than `0` to save this number of samples before a trigger event. Specify a value greater than `0` to skip this number of samples after the trigger event before data acquisition begins.

8 In the **Decimation** box, enter a value to indicate that data should be collected at each sample time (`1`) or at less than every sample time (`2` or greater).

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

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

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

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

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

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

10 In the **Filename** box, enter a name for the file to contain the signal data. By default, the target PC writes the signal data to a file named `C:\data.dat`.

11 From the **Mode** list, select either `Lazy` or `Commit`. Both modes open a file, write signal data to the file, then close that file at the end of the session. With the `Commit` mode, each file write operation simultaneously updates the FAT entry for the file. This mode is slower, but the file system always knows the actual file size. With the `Lazy` mode, the FAT entry is updated only when the file is closed and not during each file write operation. This mode is faster, but if the system crashes before the file is closed, the file system might not know the actual file size (the file contents, however, will be intact). If you experience a system crash, you can expect to lose a **WriteSize** amount of data.

12 In the **WriteSize** box, enter the block size, in bytes, of the data chunks. This parameter specifies that a memory buffer of length **Number of samples** write data to the file in **WriteSize** chunks. By default, this parameter is 512 bytes, which is the typical disk sector size. Using a block size that is the same as the disk sector size provides optimal performance.

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

13 In the **Number of samples** box, enter the number of values to be acquired in a data package.

14 Select the **AutoRestart** check box to enable the scope of type file 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 scope of type file collect data up to **Number of samples**, then stop.

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

Your next task is to simulate the model. See “Simulating the Model” on page 3-38.

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

Simulating the Model

You use Simulink in normal mode to observe the behavior of your model in nonreal time. This section includes the following topics:

Simulating the Model with Simulink (p. 3-38)	Use the Simulink window interface to run a simulation.
Simulating the Model from MATLAB (p. 3-40)	Use Simulink functions in the MATLAB Command Window to run a simulation.

For procedures to run your target application in real time, see “Running the Target Application” on page 3-60.

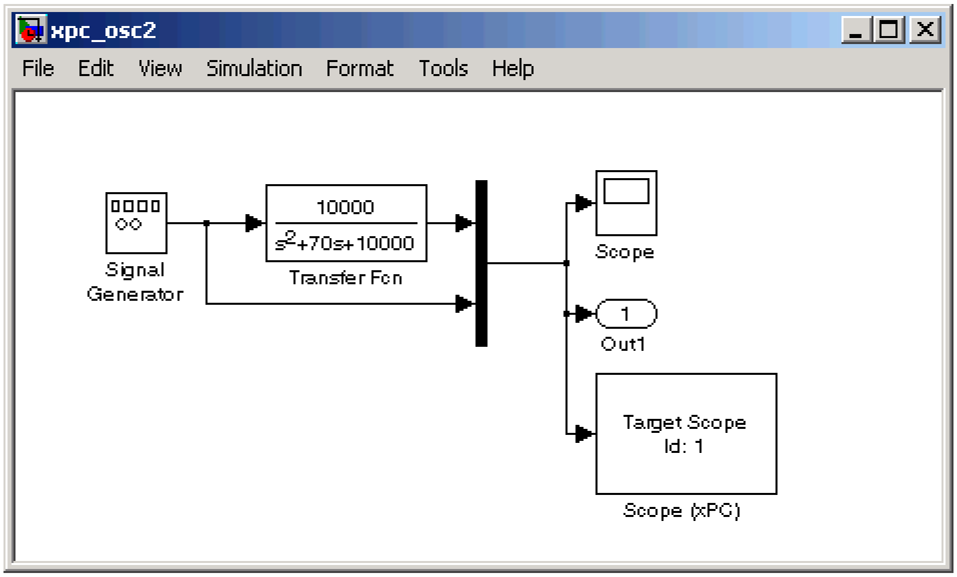
Simulating the Model with Simulink

After you load your Simulink model, you can run a simulation. This procedure uses the Simulink model `xpc_osc2.mdl` as an example and assumes you have already loaded that model. To create this model, see “Creating a Simple Simulink Model” on page 3-2.

1 In the MATLAB window, type

```
xpc_osc2
```

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

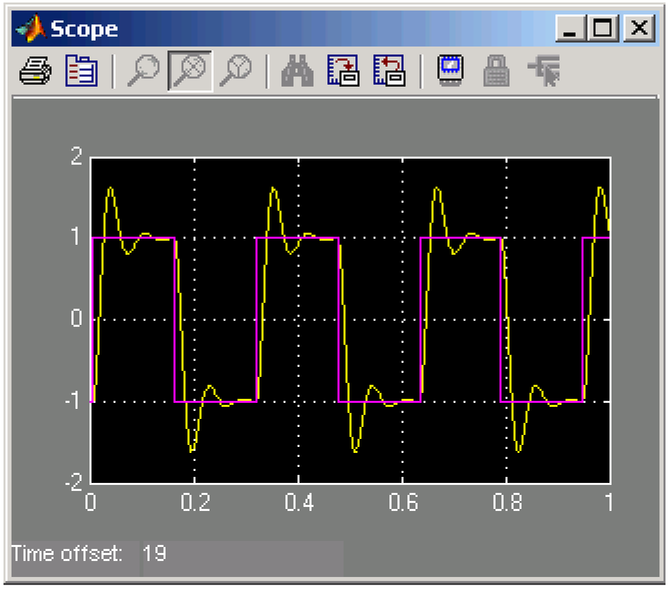


2 In the Simulink window, double-click the Scope block.

Simulink opens a scope window.

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

The **Scope1** window displays a trace of the signal data.



4 You can either let the simulation run to its stop time, or stop the simulation manually. To stop the simulation manually, from the **Simulation** menu, click **Stop**.

Your next task is to create an xPC Target application. See "xPC Target Application" on page 3-44.

Simulating the Model from MATLAB

You run a simulation of your Simulink model to observe the behavior of the model in nonreal time.

After you load your Simulink model into the MATLAB workspace, you can run a simulation. This procedure uses the Simulink model `xpc_osc2.mdl` as an example and assumes you have already loaded that model. To create this model, see "Creating a Simple Simulink Model" on page 3-2.

1 In the MATLAB window, type

```
sim('xpc_osc2')
```

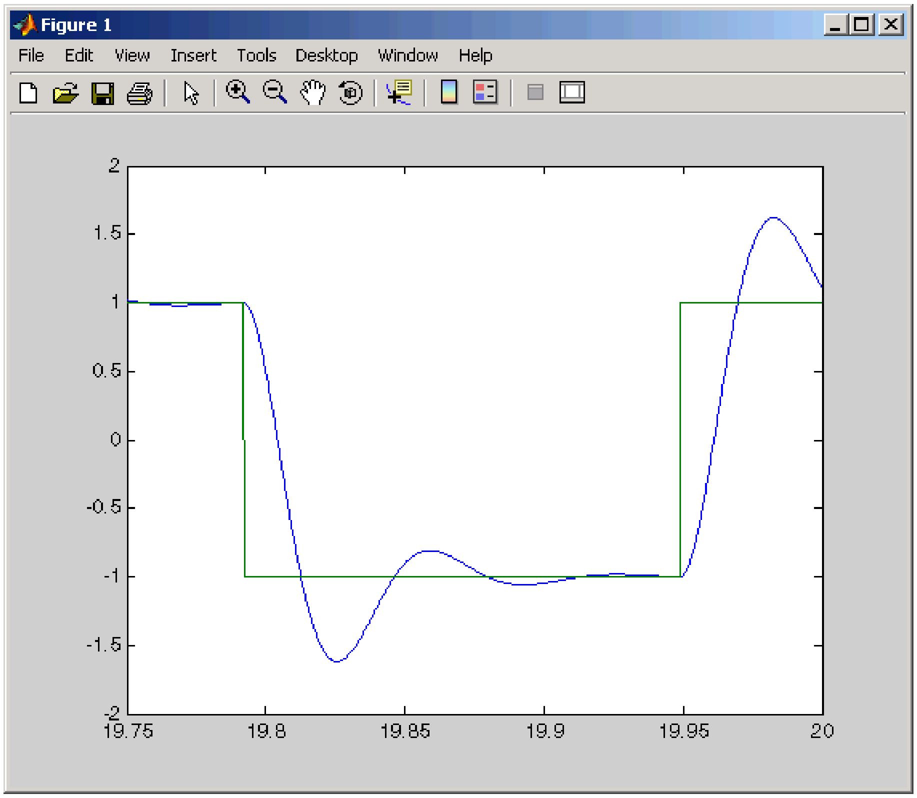
Simulink runs a simulation in normal mode through to completion. You cannot manually stop the simulation. See the “Simulink” documentation for further information on using the `sim` command.

2 After Simulink finishes the simulation, type

```
plot(tout,yout)
```

You entered the MATLAB variables `tout` and `yout` in the **Data I/O** pane on the Configuration Parameters dialog box. The signals are logged to memory through Outport blocks. To add an Outport block, see “Adding a Simulink Outport Block” on page 3-10 and “Entering Parameters for the Outport Blocks” on page 3-13.

MATLAB opens a plot window and displays the output response. The signal from the signal generator is added to the Outport block and shown in the figure below.



Note When your target application is running in real time, data is not saved to the variables `tout` and `yout`. Instead, data is saved in the target PC memory and can be retrieved through the target object properties `tg.TimeLog`, `tg.StateLog`, and `tg.OutLog`. However, in the Configuration Parameters dialog box, you must still select the **Time**, **States**, and **Output** check boxes for data to be logged to the target object properties.

Your next task is to create a target application. See “xPC Target Application” on page 3-44.

xPC Target Application

You run the target application to observe the behavior of your model in real time. This section includes the following topics:

Booting the Target PC (p. 3-44)	Start the xPC Target kernel running on the target PC.
Troubleshooting the Boot Process (p. 3-45)	Solve problems when booting the target PC
Entering the Real-Time Workshop Parameters (p. 3-46)	Enter parameters specific for building target applications.
Building and Downloading the Target Application (p. 3-51)	Use Real-Time Workshop and a third-party C compiler to create the target application.
Troubleshooting the Build Process (p. 3-53)	Solve problems when building your target application.
Increasing the Time-Out Value (p. 3-54)	Increase the time the download process waits for a target application to load onto the target PC.
xPC Target Options (p. 3-55)	Reference for all of the fields in the dialog box.

For procedures to simulate your model in nonreal time, see "Simulating the Model" on page 3-38.

Booting the Target PC

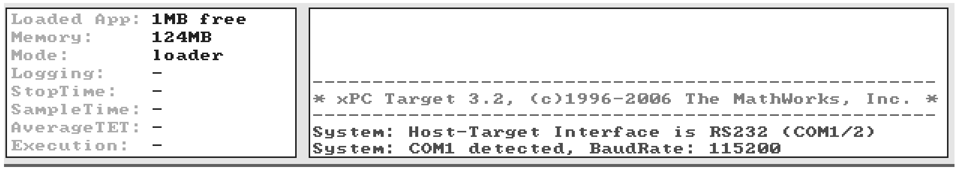
Booting the target computer loads and starts the xPC Target kernel on the target PC. The loader then waits for xPC Target to download your target application from the host PC.

After you have configured xPC Target using the xPC Target Explorer and created a target boot disk for that setup, you can boot the target PC. You need to boot the target computer before building your target application because the build process automatically downloads your target application to the target PC. Be sure that you have followed the instructions from Chapter 2, "Installation and Configuration" before continuing.

1 Insert the target boot disk into the target PC disk drive.

2 Turn on the target PC or press the **Reset** button.

The target PC displays the following screen.



```
Loaded App: 1MB free
Memory:     124MB
Mode:       loader
Logging:    -
StopTime:   -
SampleTime: -
AverageTET: -
Execution:  -
```

```
----------------------------------------------------
* xPC Target 3.2, (c)1996-2006 The MathWorks, Inc. *
----------------------------------------------------
System: Host-Target Interface is RS232 (COM1/2)
System: COM1 detected, BaudRate: 115200
```

In the example above, the status window shows that the kernel is in loader mode and waiting to load a target application. 1 MB of memory is reserved for the application, 3 MB is used by the kernel, and 28 MB is available from a total of 32 MB. The xPC Target kernel uses the 28 MB for the heap, running scopes, and acquiring data.

Your next task is to enter the simulation and real-time run parameters for Real-Time Workshop. See "Entering the Real-Time Workshop Parameters" on page 3-46.

Troubleshooting the Boot Process

Possible Problem

When booting the target PC, it might display a message like the following

```
xPC Target 3.X loading kernel..@@@@@@@@@@@@@@@@@@@@@@@
```

The target PC displays this message when it cannot read and load the kernel from the target boot disk. The probable cause is a bad disk.

Solution. Reformat the disk or use a new formatted floppy disk and create a new target boot disk.

Possible Problem

When booting the target PC, you get a message similar to the following:

```
Not a bootable medium or NTLDR is missing
```

Selecting either DOSLoader or StandAlone mode instead of BootFloppy mode can cause this message.

Solution. If the xPC Target Explorer window is not already open, open it. In the MATLAB Command Window, type xpcexplr. In the xPC Target Explorer **xPC Target Hierarchy** pane, select a target PC Configuration node. For example, select the Configuration node for TargetPC1. From the **Target boot mode** list, select Bootfloppy. Create a new boot floppy disk.

Entering the Real-Time Workshop Parameters

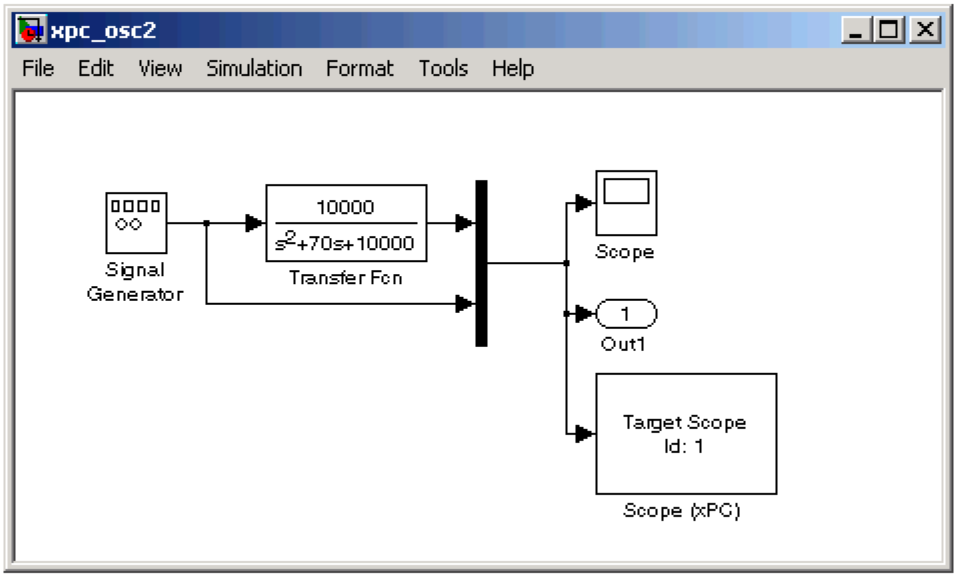
You enter the simulation and real-time run parameters in the Configuration Parameters dialog box. These parameters give information to Real-Time Workshop on how to build the target application from your Simulink model.

After you load a Simulink model and boot the target PC, you can enter the simulation parameters. This procedure uses the Simulink model xpc_osc2.mdl as an example and assumes you have already loaded that model (see "Simulink Model" on page 3-2).

1 In the MATLAB window, type

```
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**.

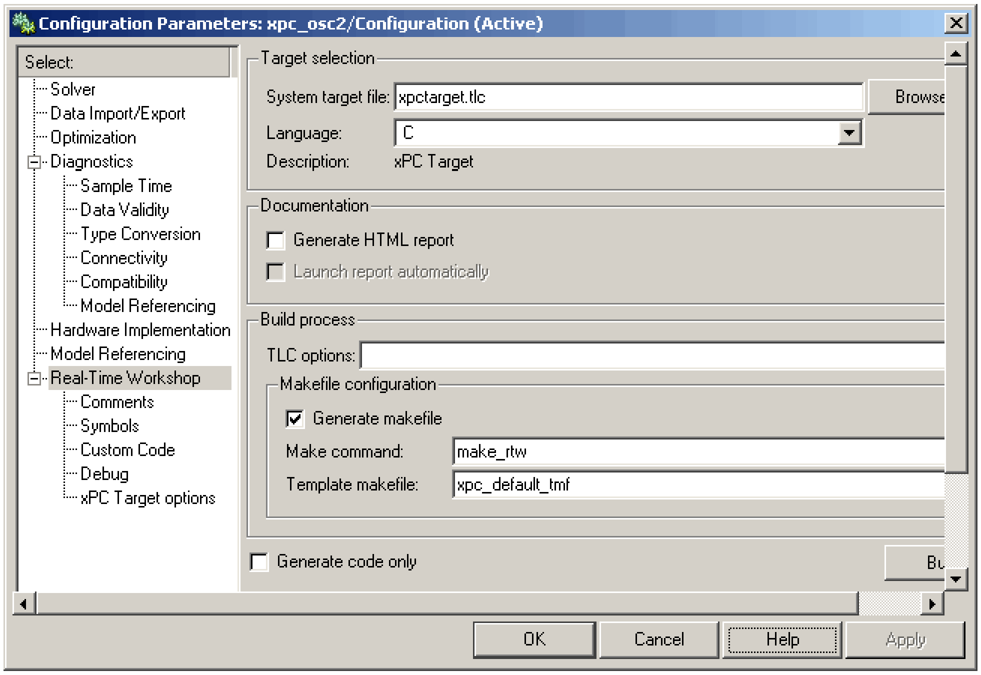
The Configuration Parameters dialog box is displayed for the model.

3 Click the **Real-Time Workshop** node.

The **Real-Time Workshop** 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 **Real-Time Workshop** pane should now look like the figure shown.



System target file: xpctarget.tlc
Language: C
Description: xPC Target
Generate HTML report ☐
Launch report automatically ☐
TLC options:
Generate makefile ☑
Make command: make_rtw
Template makefile: xpc_default_tmf
Generate code only ☐

If you have the Real-Time Workshop Embedded Coder, you can build an ERT target application. To build an ERT target application, in the **Target selection** section, click the **Browse** button at the **System target file** list. Click `xpctargetert.tlc`, and then click **OK**.

Note that if the Real-Time Workshop Embedded Coder is not installed and you select the `xpctargetert.tlc`, the build fails.

Note Leave the **Language** field set to `C`. xPC Target does not support generating C++ code.

---

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

   The associated pane is displayed.

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 xPC Target does not support variable-step solvers.

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

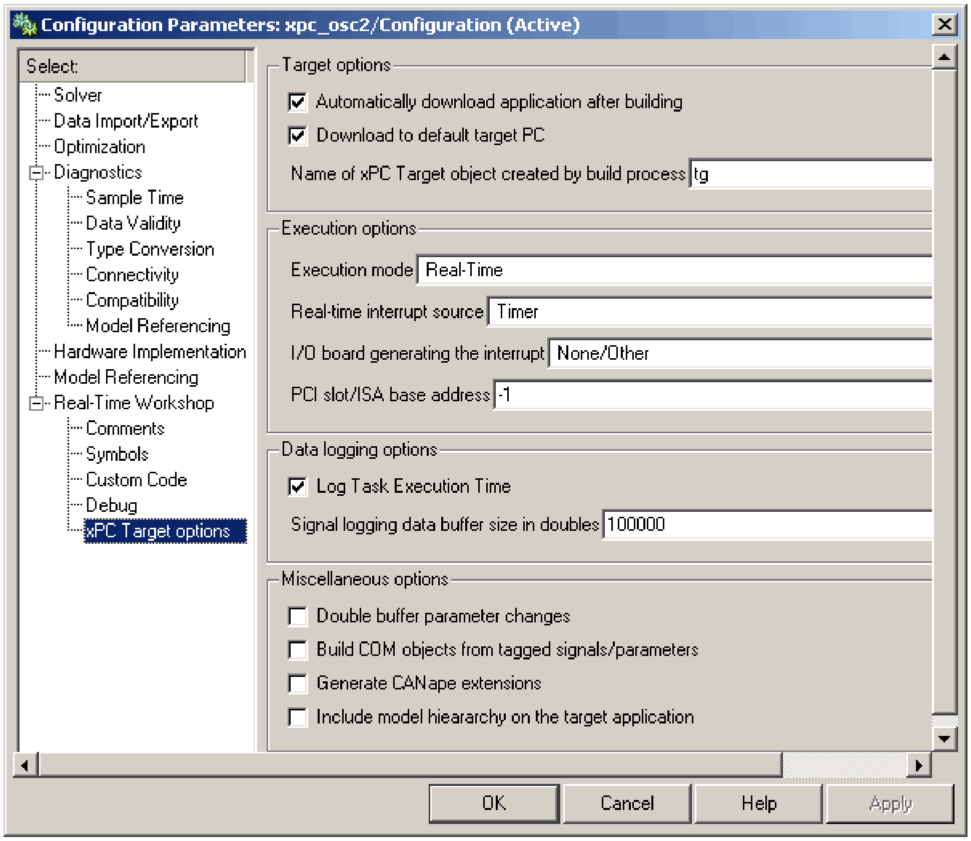
8 Select the **Log Task Execution Time** check box to log task execution times to the target object property `tg.TETlog`.

   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 **Real-Time Workshop** pane should now look like the figure shown.



Configuration Parameters: xpc_osc2/Configuration (Active)

Select:
- Solver
- Data Import/Export
- Optimization
- Diagnostics
  - Sample Time
  - Data Validity
  - Type Conversion
  - Connectivity
  - Compatibility
  - Model Referencing
- Hardware Implementation
- Model Referencing
- Real-Time Workshop
  - Comments
  - Symbols
  - Custom Code
  - Debug
  - xPC Target options

Target options

- [✓] Automatically download application after building
- [✓] Download to default target PC

Name of xPC Target object created by build process: tg

Execution options

Execution mode: Real-Time

Real-time interrupt source: Timer

I/O board generating the interrupt: None/Other

PCI slot/ISA base address: -1

Data logging options

- [✓] Log Task Execution Time

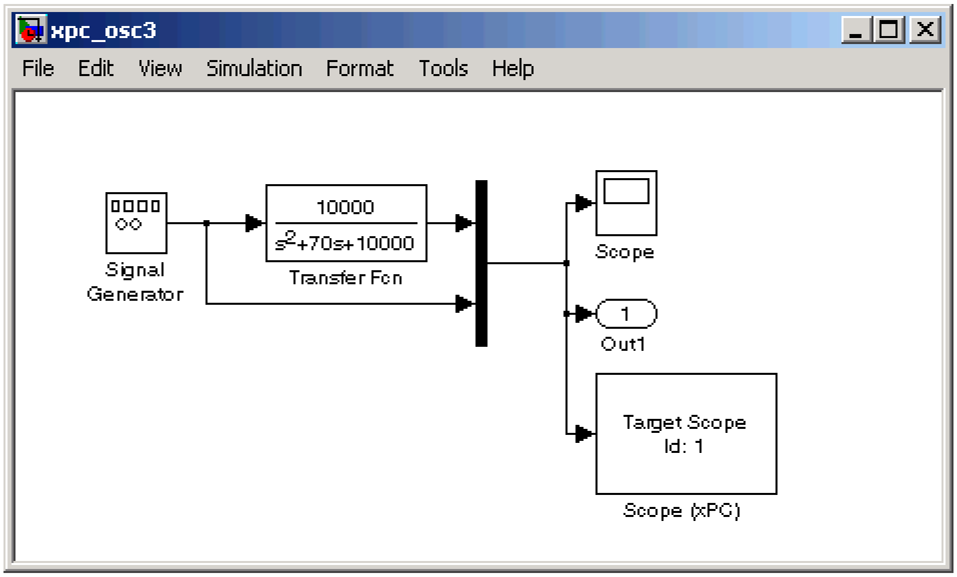
Signal logging data buffer size in doubles: 100000

Miscellaneous options

- [ ] Double buffer parameter changes
- [ ] Build COM objects from tagged signals/parameters
- [ ] Generate CANape extensions
- [ ] Include model hieararchy on the target application

11 Click **OK**.

12 From the **File** menu, click **Save as**. Enter a filename. For example, enter `xpc_osc3` and then click **Save**.



Your next task is to create (build) the target application. See “Building and Downloading the Target Application” on page 3-51.

Building and Downloading the Target Application

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

After you enter your changes in the Configuration Parameters dialog box, you can build your target application. This procedure uses the Simulink model `xpc_osc3.mdl` as an example. To create this model, see “xPC Target Application” on page 3-44. By default, the build procedure downloads the target application to the default target PC, as designated in the xPC Target Explorer. See “xPC Target Options” on page 3-55 for further details on setting the target PC for a target application.

1 In the MATLAB window, type

```
xpc_osc3
```

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

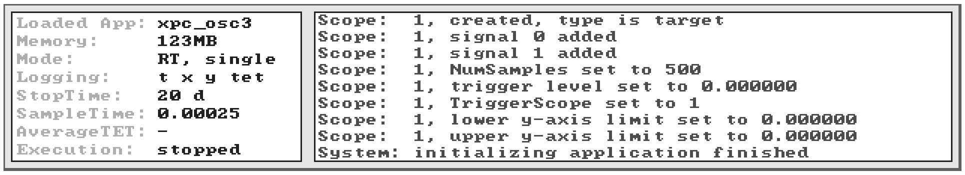
2 In the Simulink window and from the **Tools** menu, select **Real-Time Workshop**. From the **Real-Time Workshop** submenu, click **Build Model**.

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 the following lines after a successful build process:

```
### Starting Real-Time Workshop build procedure for model:
xpc_osc3
. . .
### Successful completion of xPC Target build procedure for
model: xpc_osc3
```

The target PC displays the following information:



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

3 In the MATLAB window, type

```
tg
```

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

If you do not have a successful build, see “Troubleshooting the Build Process” on page 3-53.

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

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

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

Your next task is to run the target application in real time on the target PC. See “Running the Target Application” on page 3-60.

Troubleshooting the Build Process

If the host PC and target PC are not properly connected, or you have not correctly entered the environment properties, the download process is terminated after about 5 seconds with a time-out error. Be sure that you have following the instructions outline in Chapter 2, “Installation and Configuration” before continuing.

To correct the problem, use the following procedure:

1 If the xPC Target Explorer is not already up, in the MATLAB window, type

   ```
   xpcexplr
   ```

   The xPC Target Explorer window opens.

2 For the target PC in question, check the RS-232 or TCP/IP parameters. If necessary, make changes to the communication properties and recreate the target boot disk.

   In some cases, the download might have completed successfully even though you get a time-out error. To detect this, wait until the target screen displays

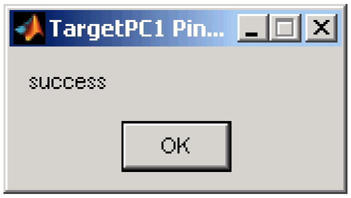
   ```
   System:initializing application finished.
   ```

3 In the xPC Target Explorer window

a Select the target PC in question

b From the menu bar, select **Target > Ping Target**.

For a working connection between the host PC and target PC, xPC Target Explorer displays a pop-up dialog box.



success

4 Right-click the target PC in question and select `Connect`.

If the connection is resumed, the connection is all right. If the connection is timing out consistently for a particular model, the time-out needs to be increased. See “Increasing the Time-Out Value” on page 3-54.

For information on setting up the xPC Target software environment, see either “Environment Properties for Serial Communication” on page 2-25 or “Environment Properties for Network Communication” on page 2-38, and then see “Target Boot Disk” on page 2-45.

Increasing the Time-Out Value

By default, if the host PC does not get a response from the target PC after downloading a target application and waiting about 5 seconds, the host PC software times out. On the other hand, the target PC responds only after downloading and initializing the target application.

Usually 5 seconds is enough time to download a target application, but in some cases it may not be sufficient. The time to download a target application mostly depends on your I/O hardware. For example, thermocouple hardware takes longer to initialize. In this case, even though the target PC is fine, a false time-out is reported.

To increase the time-out value, use the following procedure:

1. In your MATLAB working directory, create a file called `xpcdltimeout.dat`.

2. In this file, put a single integer. For example, enter

   ```
   20
   ```

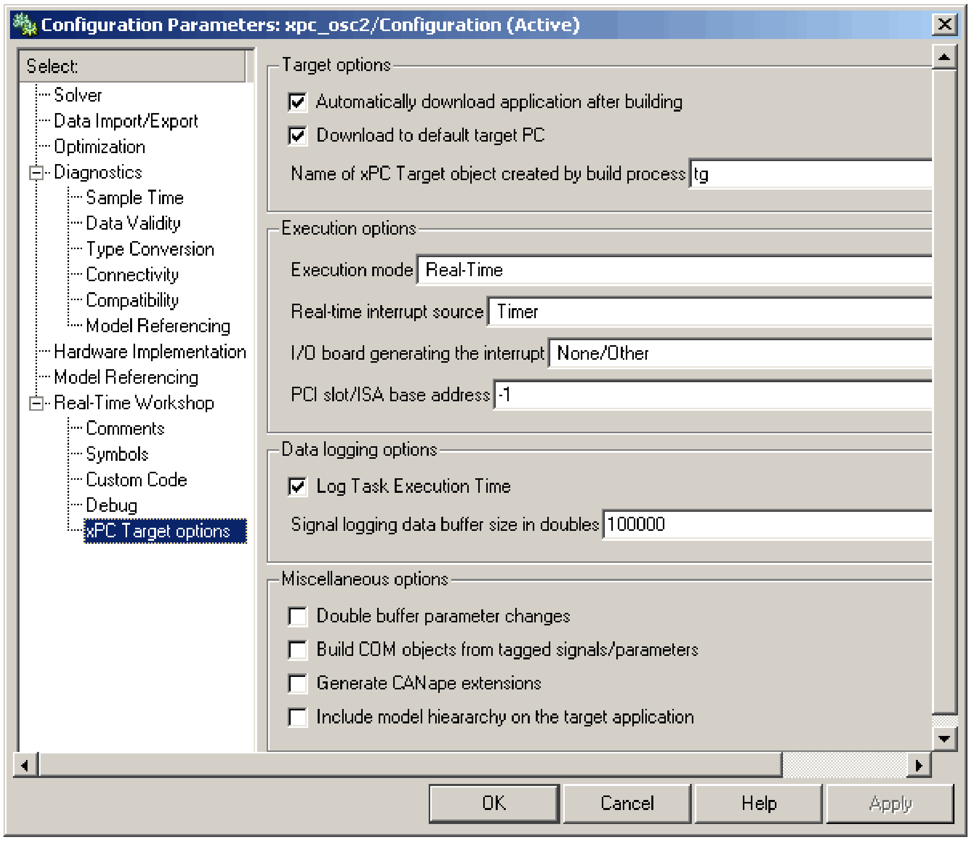
   In this case, the host PC waits for about 20 seconds before declaring that a time-out has occurred. Note that it does not take 20 seconds for every download. The host PC polls the target PC about once every second, and if a response is returned, declares success. Only in the case where a download really fails it does take the full 20 seconds.

If the file `xpcdltimeout.dat` exists, it is read once before every download. To change the time-out value, change the number in this file. Setting the time-out to `5` is the same as the default. Note also that simply removing the file does not change the time-out to the default value. xPC Target uses the last value you entered until you restart MATLAB.

xPC Target Options

Following is a description of the fields in the **xPC Target options** node. To access this node, from the model, select **Simulation > Configuration Parameters**. In the left pane, click the **xPC Target options** node.

You might need to enter and select these options before you create (build) a target application. However, this is not normally necessary because the parameters have reasonable defaults.


Configuration Parameters: xpc_osc2/Configuration (Active)

Select:
- Solver
- Data Import/Export
- Optimization
- Diagnostics
  - Sample Time
  - Data Validity
  - Type Conversion
  - Connectivity
  - Compatibility
  - Model Referencing
- Hardware Implementation
- Model Referencing
- Real-Time Workshop
  - Comments
  - Symbols
  - Custom Code
  - Debug
  - xPC Target options

Target options
- [✓] Automatically download application after building
- [✓] Download to default target PC
- Name of xPC Target object created by build process: tg

Execution options
- Execution mode: Real-Time
- Real-time interrupt source: Timer
- I/O board generating the interrupt: None/Other
- PCI slot/ISA base address: -1

Data logging options
- [✓] Log Task Execution Time
- Signal logging data buffer size in doubles: 100000

Miscellaneous options
- [ ] Double buffer parameter changes
- [ ] Build COM objects from tagged signals/parameters
- [ ] Generate CANape extensions
- [ ] Include model hieararchy on the target application

Automatically download application after building — Selecting this option allows Real-Time Workshop to build and download the target application to the target PC. If you do not select this check box, Real-Time Workshop builds the application (DLM), but does not download it to the target PC. By default, this check box is selected.

Download to default target PC — Selecting this option directs Real-Time Workshop to build and download the target application to the default target PC. This assumes that you configured a default target PC through the xPC

Target Explorer (see sections “Serial Communication” on page 2-24 and “Network Communication” on page 2-31 if you have not). By default, this check box is selected.

Specify the target PC name — If you deselect the **Download to default target PC** check box, this field is displayed.

Name of xPC Target object created by build process — Enter the name of the target object created by the build process. The default target object name is `tg`. If you select the **Download to default target PC** check box, this field is displayed.

Execution mode — From the list select either `Real-Time` or `Freerun`.

Real-time interrupt source — From the list select `Timer` or a number from `5` to `15`.

I/O board generating the interrupt — From the list, select the board interrupt source.

PCI slot/ISA base address — Enter the slot number or base address for the I/O board generating the interrupt. The PCI slot can be either `-1` (let xPC Target determine the slot number) or of the form `[bus, slot]`. The base address is a hexadecimal number of the form `0x300`.

To determine the bus and PCI slot number of the boards in the target PC, do one of the following:

- In the xPC Target Explorer, connect to the target PC in question and expand the `PCI Devices` node.
- In the MATLAB window, type `getxpcpci`.

Log Task Execution Time — Select this check box to log task execution times to the target object property `tg.TETlog`.

Signal logging data buffer size in doubles — Enter the maximum number of sample points to save before wrapping. Your target application uses this buffer to store the time, states, outputs, and task execution time logs as defined in your Simulink model.

Note that the maximum value for this option derives from available target PC memory, which xPC Target also uses to hold other items. For example, in addition to signal logging data, xPC Target also uses the target PC memory for the xPC Target kernel, target application, and scopes.

For example, the model `xpc_osc2.mdl` has six data items (one time, two states, two outputs, and one task execution time (TET)). If you enter a buffer size of `100000`, then the target object property `tg.MaxLogSamples` is calculated as `floor(100000) / 6) = 16666`. After saving 16666 sample points, the buffer wraps and further samples overwrite the older ones.

If you enter a logging buffer size larger than the available RAM on the target PC, after downloading and initializing the target application, the target PC displays a message, `ERROR: allocation of logging memory failed`. In this case you need to install more RAM or reduce the buffer size for logging. In any case you must reboot the target PC. To calculate the maximum buffer size you might have for you target application logs, divide the amount of available RAM on your target PC by 8. Enter that value for the **Signal logging data buffer size in doubles** value.

Double buffer parameter changes — Selecting this option changes parameter tuning so that the process of changing parameters in the target application uses a double buffer:

- When a parameter change request is received, the new value is compared to the old one. If the new value is identical to the old one, it is discarded, and if different, queued.
- At the start of execution of the next sample of the real-time task, all queued parameters are updated. This means that parameter tuning affects the task execution time (TET), and the very act of parameter tuning can cause a CPU overload error.

Double buffering leads to a more robust parameter tuning interface, at the cost of increased TET and a higher probability of overloads. You should normally clear this check box. It is cleared by default.

Build COM objects from tagged signals/parameters — If you select this check box, the build process creates a model-specific COM library file:

```
<model_name>COMiface.dll
```

Use this file to create custom GUI interfaces with Visual Basic or other tools that can use COM objects.

Generate CANape extensions — Selecting this check box enables target applications to generate data, such as that for A2L, for Vector CANape. By default, this check box is not selected.

Include model hierarchy on the target application — Selecting this check box includes the model hierarchy as part of the target application. If you select this, you can connect to the target PC from xPC Target Explorer without being in the target application build directory. Note, selecting this check box might increase the size of the target application, depending on the size of the model.

Running the Target Application

During the build process, xPC Target creates a target object that represents the target application running on the target PC. The target object is defined by a set of properties and associated methods. You control the target application and computer by setting the target object properties. This section includes the following topics:

Control with xPC Target Explorer (p. 3-60)	Use xPC Target Explorer to load, start, and stop a target application, and to change target application parameters.
Control with MATLAB Commands (p. 3-70)	Start and stop the target application, view the status, change the stop time and sample time, and download a new target application.
Control with Simulink External Mode (p. 3-72)	Connect a Simulink model to an xPC Target application for parameter tuning.

Control with xPC Target Explorer

This procedure assumes you have created an xPC Target boot disk and you have booted the target PC. See "Target Boot Disk" on page 2-45. While the xPC Target Explorer gives you access to build and download a target application, this procedure begins with a target application already downloaded to the target PC (see "xPC Target Application" on page 3-44). For a description of how to download a prebuilt target application, see "Downloading and Running Target Applications on a Target PC" on page 3-64.

1 In the MATLAB window, type

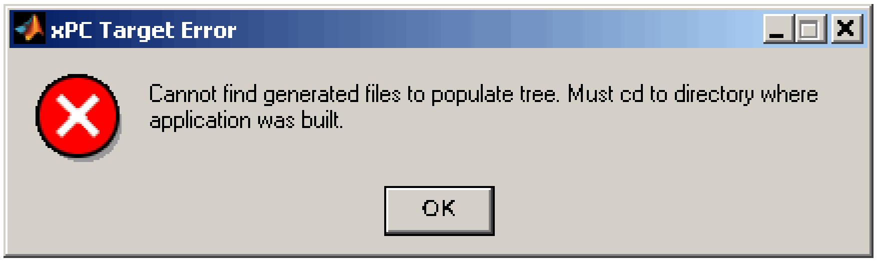
```
xpcexplr
```

The xPC Target Explorer window opens.

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

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

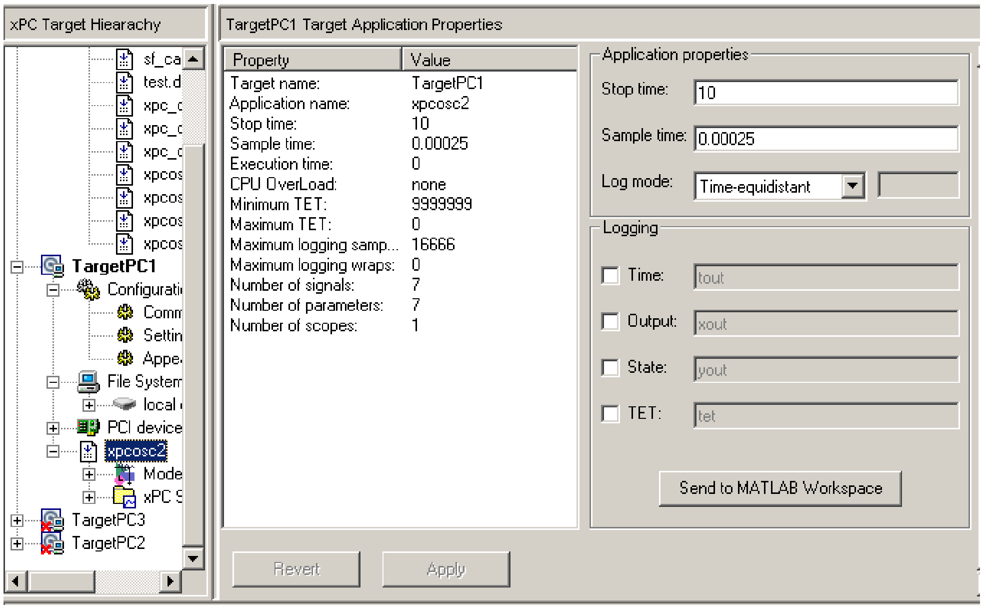
- You must be in the same directory in which you build the target application. Otherwise, xPC Target Explorer returns an error.



Cannot find generated files to populate tree. Must cd to directory where application was built.

- When you built the target application, you selected the **Include model hierarchy on the target application** check box in the xPC Target Options pane.

If you are in the appropriate directory, a node for the target application appears in the **xPC Target Hierarchy** under the target PC node and displays information about the previously loaded target application.



Property	Value
Target name:	TargetPC1
Application name:	xpcosc2
Stop time:	10
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:	1

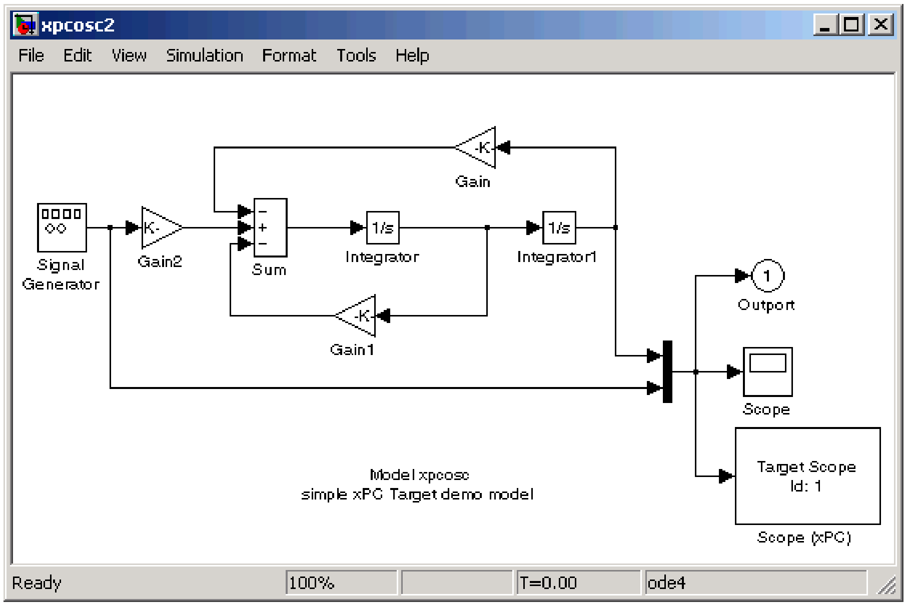
Stop time: 10
Sample time: 0.00025
Log mode: Time-equidistant

Time: tout
Output: xout
State: yout
TET: tet

Note, if, in your current directory, you have a prebuilt target application that you want to download to the target PC, left-click and drag the desired target application to the target PC to which you want to download the target application.

3 If you want to rebuild the current target application, in the xPC Target Explorer window, right-click the target application node and select `Go To Simulink Model`.

The Simulink window opens for that model.



4 To rebuild the target application, in the Simulink window and from the **Tools** menu, select **Real-Time Workshop**. From the **Real-Time Workshop** submenu, click **Build Model**.

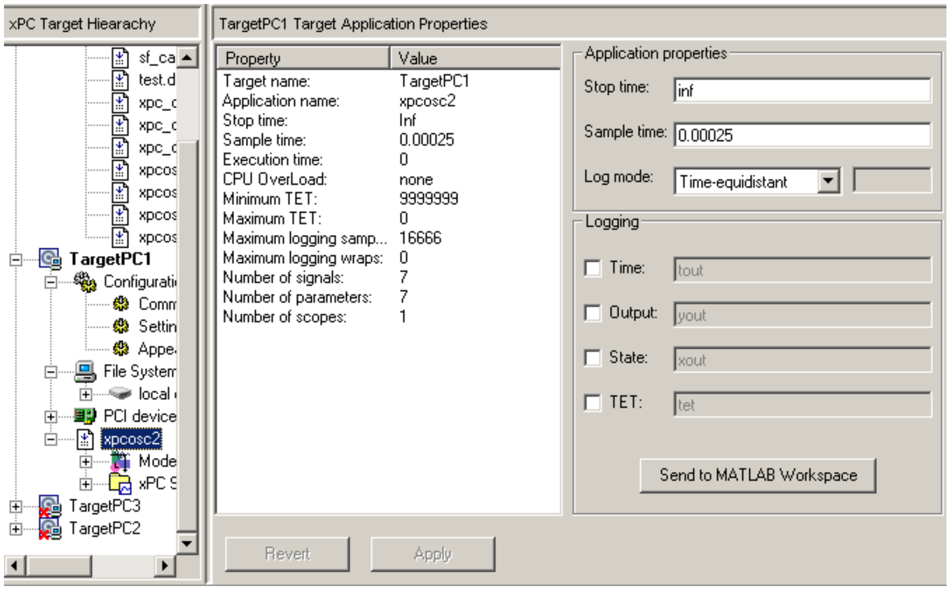
xPC Target recompiles, links, and downloads the target application to the target PC.

5 Start the target application. For example, in the xPC Target Explorer window, select the downloaded target application.

6 From the toolbar, click the **Start Application** button ▶. The target application begins running on the target PC, and stops when it reaches the stop time.

7 With the target application still selected in the Target Hierarchy pane of xPC Target Explorer, enter a new value for the **Stop time** value for the application and click **Apply**. For example,

```
inf
```



TargetPC1 Target Application Properties

Property	Value
Target name:	TargetPC1
Application name:	xpcosc2
Stop time:	Inf
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:	1

Application properties

Stop time: inf

Sample time: 0.00025

Log mode: Time-equidistant

Logging

☐ Time: tout

☐ Output: yout

☐ State: xout

☐ TET: tet

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 directory to one that contains the prebuilt target applications (DLMs) you want to download to your target PCs and how to download and run a prebuilt target application. To view the model hierarchy, you must be in the same directory in which you build the target application. This topic assumes the following:

- In your current working directory, you have a prebuilt target application that you want to download to a target PC.
- You have installed xPC Target software and booted the target PC to which you want to download a target application.

- You have a physical connection between the xPC Target Explorer host machine and the target PC to which you want to download a target application.

1 In the xPC Target Explorer, left-click the **File** menu. (Note that you can also change this directory from the MATLAB window or by right-clicking the DLM node.)

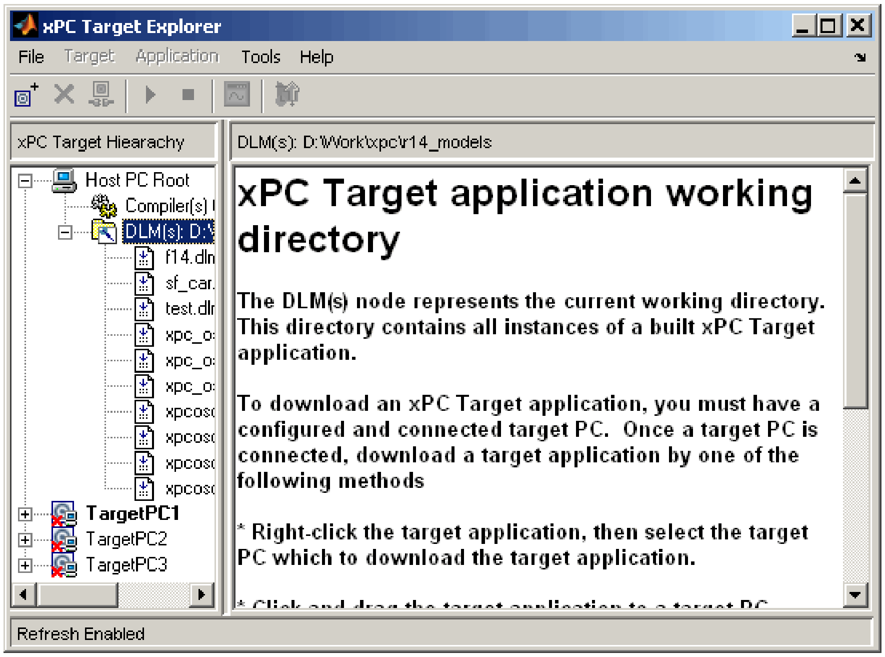
2 Select **Change current directory**.

   A browser is displayed.

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

4 Click **OK**.

   A list of the prebuilt target applications appears, as shown.



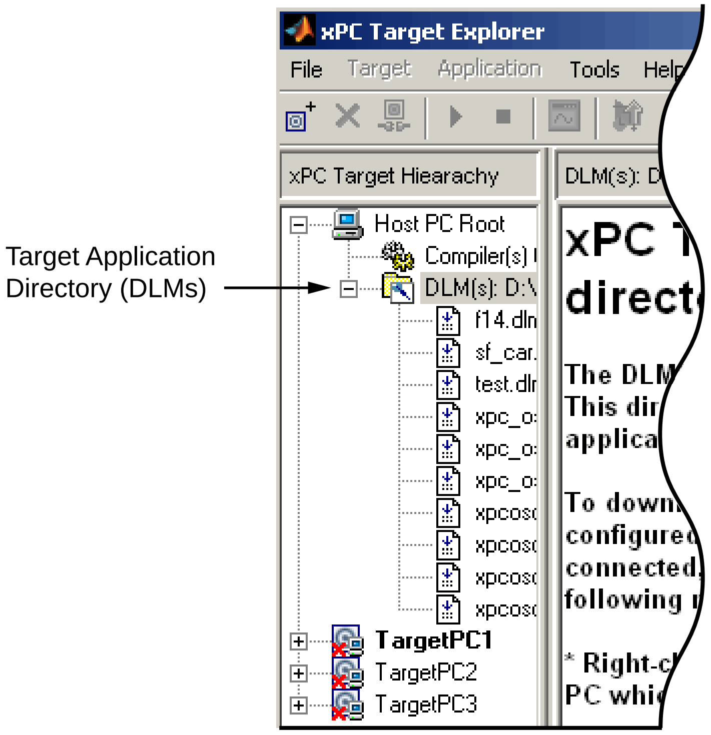
xPC Target application working directory

The DLM(s) node represents the current working directory. This directory contains all instances of a built xPC Target application.

To download an xPC Target application, you must have a configured and connected target PC. Once a target PC is connected, download a target application by one of the following methods

* Right-click the target application, then select the target PC which to download the target application.

5 In xPC Target Explorer, check that the DLM(s) node in the **xPC Target Hierarchy** has the pathname of the directory that contains the prebuilt target application you want to download to the target PC.



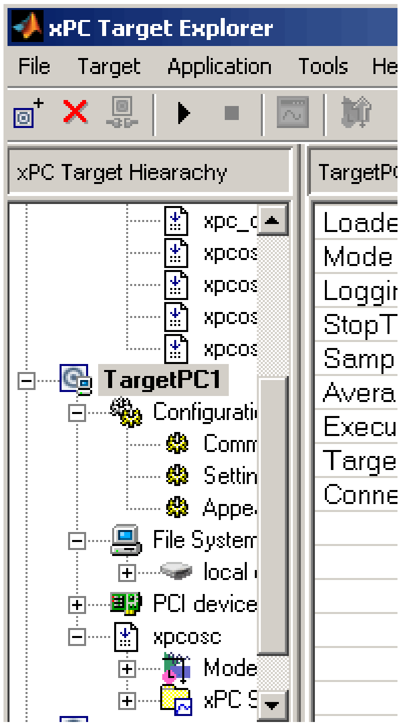
6 Right-click a target PC that you booted with xPC Target software, for example, `TargetPC1`.

7 Select **Connect**.

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

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

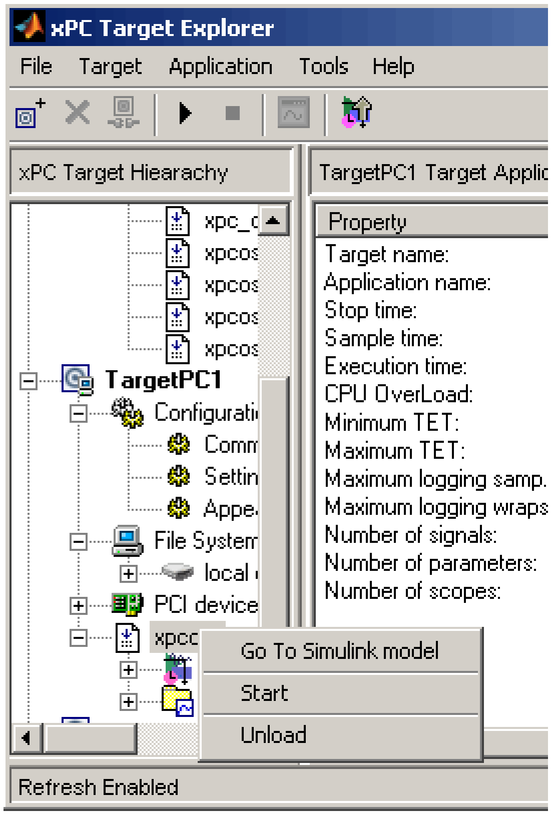
xPC Target Explorer downloads the target application to the target PC. A node for the target application appears in the **xPC Target Hierarchy** under the target PC node.



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.



TargetPC1 Target Applic

Property
Target name:
Application name:
Stop time:
Sample time:
Execution time:
CPU OverLoad:
Minimum TET:
Maximum TET:
Maximum logging samp.
Maximum logging wraps
Number of signals:
Number of parameters:
Number of scopes:

10 Select **Start**.

xPC Target Explorer starts running the loaded target application.

11 Stop the target application (described here) or let the target application run to the end. To stop the target application, right-click the target application node (for example, `xpcosc`) and select **Stop** from the list.

See also “Signal Tracing with xPC Target Explorer” and “Signal Logging with xPC Target Explorer” within “Signals and Parameters” in the xPC Target User’s Guide.

Manipulating Target Application Properties

This topic describes how to manipulate target application properties. It assumes that you have already downloaded the target application `xpcosc` to a target PC.

1 In xPC Target Explorer, select the node of the loaded target application in which you are interested. For example, `xpcosc`.

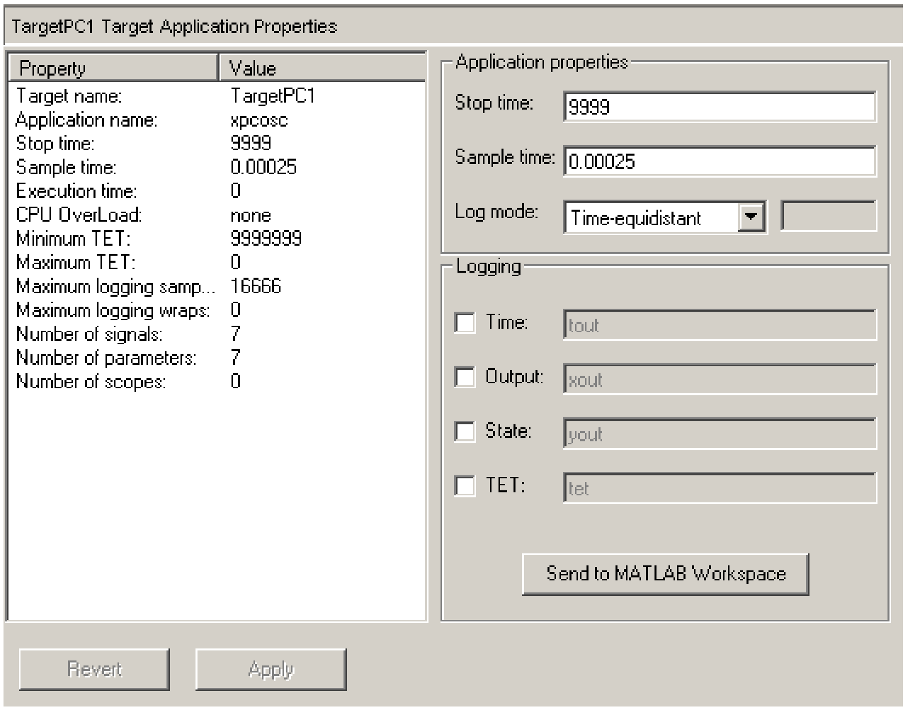
   The right pane changes to the target application properties pane for the application.

2 In this pane, you can change the following application properties:

   - **Stop time**
   - **Sample time**
   - **Log mode**

   See "User Interaction" on page 1-23 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: 9999

Sample time: 0.00025

Log mode: Time-equidistant

Logging

☐ Time: tout

☐ Output: xout

☐ State: yout

☐ TET: tet

Control with MATLAB Commands

You run your target application in real time to observe the behavior of your model with generated code.

After xPC Target downloads your target application to the target PC, you can run the target application. This procedure uses the Simulink model `xpc_osc3.mdl` as an example, and assumes you have created and downloaded the target application for that model. It also assumes that you have assigned `tg` to the appropriate target PC.

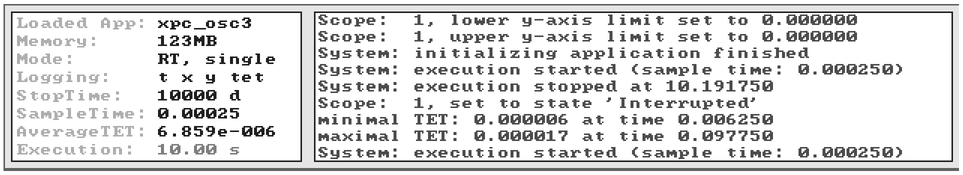
1 In the MATLAB window, type any of

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

   The target application starts running on the target PC. In the MATLAB window, the status of the target object changes from stopped to running.

```
xPC Object
  Connected    = Yes
  Application    = xpc_osc3
  Mode    = Real-Time Single-Tasking
  Status    = running
```

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



```
Loaded App: xpc_osc3
Memory:     123MB
Mode:       RT, single
Logging:    t x y tet
StopTime:   10000 d
SampleTime: 0.00025
AverageTET: 6.859e-006
Execution:  10.00 s
```

```
Scope:  1, lower y-axis limit set to 0.000000
Scope:  1, upper y-axis limit set to 0.000000
System: initializing application finished
System: execution started (sample time: 0.000250)
System: execution stopped at 10.191750
Scope:  1, set to state 'Interrupted'
minimal TET: 0.000006 at time 0.006250
maximal TET: 0.000017 at time 0.097750
System: execution started (sample time: 0.000250)
```

2 In the MATLAB window, type

```
-tg or tg.stop or stop(tg)
```

The target application stops running.

xPC Target 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 1-23 for further limitations on changing sample times.)

Control with Simulink External Mode

Control of your xPC Target application with Simulink is limited to connecting a Simulink model to a target application through external mode, and starting the target application. Using Simulink external mode is one method to tune parameters. In Simulink external mode, the model can only connect to the default target PC.

After you create and download a target application to the target PC, you can run the target application. This procedure uses the Simulink model `xpc_osc2.mdl` as an example (see "Building and Downloading the Target Application" on page 3-51). It assumes that you have specified the correct target PC environment on the xPC Target options node of the Real-Time Workshop parameters dialog. In particular, you must specify the target PC to which you want to connect. See the **Use the default target PC** check box description in "xPC Target Options" on page 3-55.

1. In the Simulink window, and from the **Simulation** menu, click **External**.

   A check mark appears next to the menu item **External**, and Simulink external mode is activated. Simulink external mode connects your Simulink model to your target application as a simple graphical user interface.

2. In the Simulink window, and from the **Simulation** menu, click **Connect to target**.

   All the current Simulink model parameters are downloaded to your target application. This downloading guarantees the consistency of the parameters between the host model and the target application.

3. From the **Simulation** menu, click **Start real-time code**.

   The target application begins running.

4. In the MATLAB window, type

```
tg.stop or -tg
```

You cannot stop the target application from the Simulink window by clicking **Stop real-time code** from the **Simulation** menu.

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

Menu Bar and Toolbar Contents and Shortcut Keys

The procedures in the xPC Target documentation for xPC Target Explorer use right-click operations. You can also perform xPC Target Explorer operations through the menu bar and toolbar. This section is a reference for the menu bar and toolbar contents.

The xPC Target Explorer menu bar has the following menus:

- **File** — General file operation options, including
  - **Add Target** — Add a target PC to the xPC Target system.
  - **Remove Target** — Remove a target PC from the xPC Target system.
  - **Change Host PC Current Directory** — Change directory to one that contains the prebuilt target application (DLM) you want to download to your target PCs.
  - **Close** — Close xPC Target Explorer.
- **Target** — General target PC operations, including
  - **Ping Target** — Test the communication between the host PC and target PCs.
  - **Connect to Target** — Connect xPC Target Explorer to selected target PC.
  - **Set As Default** — Designate the selected target PC as the default one.
  - **Import Environment** — Import an existing target PC configuration from the MATLAB workspace.
  - **Export Environment** — Save the target PC environment structure to a MAT-file.
  - **Save Session** — Save target PC application sessions to xPC Target Explorer.
  - **Load Session** — Load target PC application sessions to xPC Target Explorer.
- **Application** — General target application operations, including
  - **Start Application** — Start the selected target application.

- **Stop Application** — Stop the selected target application.
- **Add a scope** — Add a scope of type `Host`, `Target`, or `File`.
- **Delete scope** — Delete a scope.
- **View host scopes** — Display a viewer on the host PC for scopes of type `host`.

* **Tools** — General operations, including
  - **Enable/Disable Refresh** — Enable/disable the window refresh.
  - **Go to Simulink Model** — For the selected model, display the Simulink model.
* **Help** — General product help information, including
  - **Using xPC Target** -- Invoke help for xPC Target product.
  - **xPC Target Explorer Help** — Invoke help for xPC Target Explorer.
  - **About xPC Target** — Invoke general xPC Target information.

The xPC Target Explorer toolbar has buttons for some of the more commonly used operations available from the menu bar. These buttons include

Button	Description
	Add **Target** button
✕	Delete **Target** button
	Connect to **Target** button
▶	Start **Application** button
■	Stop **Application** button
	Scope **Viewer** button
	Go to **Simulink Model** button

The xPC Target Explorer has the following keyboard shortcuts:

Action	Shortcut
Add target	**Ctrl+A**
Remove target	**Ctrl+R**
Close	**Ctrl+W**
Stop/start target	**Ctrl+T**
Ping target	**Ctrl+P**
Delete scope	Select scope and click **Delete**

Glossary

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.

Index

A

advantages of network communication 2-31
analog input (A/D)
  driver support 1-13
analog output (D/A)
  driver support 1-13
API
  custom GUI 1-30

B

BIOS
  target PC 1-4
block parameters
  scope 3-21
booting
  target PC 3-44
  troubleshooting 3-45
build process
  target application 3-51
  troubleshooting 3-53

C

C compiler
  network setup 2-39
  required product 2-5
  serial setup 2-26
CAN field bus
  driver support 1-13
code generation options
  for Real-Time Workshop 3-46
  reference 3-55
COM API 1-30
command-line interface
  MATLAB 1-26
  target PC 1-29
communication
  between computers 1-16
  network 2-31
  network advantages 2-31
  serial 2-24
compiler
  required 2-5
computer
  communication 1-16
  desktop PC for host 1-10
  desktop PC for target 1-11
  host PC 1-10
  industrial PC 1-11
  notebook PC 1-10
  PC/104 and PC/104+ 1-11
  target PC 1-10
connections
  computers 1-11
  I/O boards 1-13
  real-world 1-13
controlling target applications
  with MATLAB 3-70
  with Simulink external mode 3-72
counter timers
  driver support 1-13
creating target applications 3-44
creating target boot disks 2-47
creating target objects 3-51
custom GUI 1-30
  API 1-30
  COM API 1-30

D

default target PC
  introduction 2-22
defining scope block parameters 3-21
desktop PC
  host computer 1-10
  target computer 1-11
directories
  installed 2-15
  working 2-15

    `xpc` 2-15
    `xpcdemo` 2-15
DOS loader mode
    embedded option 1-19
downloading target application 3-51

E

embedded option
    DOS loader mode 1-19
    stand-alone mode 1-19
encoder
    I/O driver support 1-13
entering simulation parameters 3-46
environment
    network communication 2-38
    serial communication 2-25
Ethernet card
    ISA bus 2-37
    PC/104 bus 2-33
    PCI bus 2-36
    SBS bus 2-33
external mode
    controlling target application 3-72
    user interaction 1-27

F

features of xPC Target
    fixed-point support 1-9
    MATLAB Compiler support 1-9
    parameter tuning 1-8
    real-time application 1-6
    real-time kernel 1-4
    signal acquisition 1-7
files
    data acquisition 3-32
    host PC 2-15
    installed 2-15
    project directory 2-15
    scopes 3-32
    working directory 2-15
    `xpc` directory 2-15
    `xpcdemos` directory 2-15
fixed-point support 1-9

G

GPIB field bus
    driver support 1-13
graphical user interface (GUI)
    custom with API 1-30
    custom with COM API 1-30

H

hardware environment
    requirements for target PC 2-10
heap
    xPC Target kernel 1-6
host computer
    see host PC 1-10
host PC 2-14
    communication 1-16
    configuring 2-18
    connections 1-11
    downloading software 2-14
    files 2-15
    hardware 2-24
    license file 2-14
    requirements 2-9

I

I/O boards
    supported by xPC Target 1-13
I/O driver support
    analog input (A/D) 1-13
    analog output (D/A) 1-13
    CAN field bus 1-13
    counter timers 1-13

digital 1-13
encoder 1-13
GPIB 1-13
RS-232 1-13
RS-422 1-13
RS-485 1-13
shared memory 1-13
UDP 1-13
industrial PC 1-11
initial working directory 2-16
installation prerequisite
obtaining a valid license 2-14
installing
Ethernet card for ISA 2-37
Ethernet card for PCI 2-36
hardware 2-24
network communication 2-31
on the host PC 2-14
serial communication 2-24
testing 2-50
ISA bus
Ethernet card 2-37

K

kernel
heap 1-6
target boot disk 1-4
target PC BIOS 1-4
xPC Target 1-4

L

license
obtaining 2-14

M

MATLAB
controlling target application 3-70
required product 2-3
MATLAB Compiler support 1-9
memory model
target application 1-6

N

network communication
advantages 2-31
environment 2-38
hardware 2-32
host PC 2-32
installing and setting up 2-31
specifying environment properties 2-38
target PC 2-32
notebook PC 1-10

O

outport block
adding to Simulink model 3-10
simulation parameters 3-13
overview
MATLAB command-line interface 1-26

P

parameter tuning
interactive 1-8
scripts 1-8
parameters
scope blocks 3-21
PC/104 bus
Ethernet card 2-33
PCI bus
Ethernet card 2-36

Q

Quatech serial drivers 1-14

R

rapid prototyping process 1-17
real-time application
    memory model 1-6
    task execution time 1-6
real-time kernel 1-4
Real-Time Workshop
    code generation options 3-55
required products
    C language compiler 2-5
    MATLAB 2-3
    Real-Time Workshop 2-5
    Simulink 2-4
    xPC Target 2-3
requirements
    host PC 2-9
    system 2-9
    target PC 2-10
RS-232
    Quatech 1-14
RS-422
    Quatech 1-14
RS-485
    Quatech 1-14
running simulation
    host PC 3-38

S

SBS bus
    Ethernet card 2-33
scope blocks
    parameters 3-21
    xPC Target 1-28
scopes
    `file` 3-32
    `host` 3-28
    `target` 3-22
serial communication
    environment 2-25
    hardware 2-24
    installing and setting up 2-24
    section overview 2-24
setting
    network communication 2-38
    serial communication 2-25
setting initial working directory 2-16
setup (network) dialog box
    C compiler 2-39
setup (serial) dialog box
    C compiler 2-26
shared memory driver support 1-13
signal acquisition
    logging 1-7
    monitoring 1-7
    tracing 1-7
simulation
    from MATLAB 3-40
    Simulink tutorial model 3-38
    with Simulink 3-38
simulation parameters
    entering 3-13
    for Real-Time Workshop 3-46
    xPC Target Scope block 3-21
Simulink external mode
    controlling target application 3-72
Simulink model
    basic tutorial 3-2
    outport block 3-10
    xPC Target Scope blocks 3-17
software environment
    requirements on host PC 2-9
    requirements on target PC 2-10
software installation 2-14
stand-alone mode
    embedded option 1-19
starting and stopping
    target application 3-70
system requirements
    host PC 2-9

target PC 1-10

T

target application 1-6
- building 3-51
- control with external mode 3-72
- creating 3-44
- downloading 3-51
- memory model 1-6
- running 3-60
- starting 3-70
- stopping 3-70
- target PC 3-60
- task execution time 1-6

target boot disk
- creating 2-47
- kernel 1-4
- with desktop PC 1-11
- with industrial PC 1-11

target PC
- boot disk 1-4
- booting 3-44
- command-line interface 1-29
- communication 1-16
- compatible target computers 2-12
- connecting 1-11
- creating boot disk 2-47
- hardware 2-24
- hardware requirements 2-10
- heap 1-6
- I/O boards 2-13
- real-time kernel 1-5
- running target application 3-60
- software installation 2-50
- software requirements 2-10
- troubleshooting installation 2-50

task execution time (TET) 1-6
- logging 3-46

testing installation 2-50

The MathWorks
- technical support 2-56
- valid license 2-14

timeout value
- changing 3-54

troubleshooting
- boot process 3-45
- build process 3-53
- timeout value 3-54

tutorial
- basic 3-1
- creating a Simulink model 3-2
- creating a target application 3-44
- running target application 3-60
- simulating a Simulink model 3-38

U

UDP driver support 1-13

user interaction
- MATLAB command-line interface 1-26
- Simulink external mode interface 1-27
- target PC command line 1-29
- Web browser 1-29
- with API 1-30
- with COM API 1-30
- xPC Target Scope blocks 1-28

V

valid license
- obtaining 2-14

W

Web browser
- user interaction 1-29

working directory
- initial 2-16
- setting initial 2-16

X

xPC Target
- features 1-4
- interaction 1-23
- introduction 1-1
- kernel 1-4
- required products 2-3
- supported I/O boards 1-13

xPC Target Explorer
- introduction 2-20
- menu bar 3-74
- shortcut keys 3-74
- toolbar 3-74

xPC Target Scope blocks
- adding to Simulink model 3-17
- interface 1-28