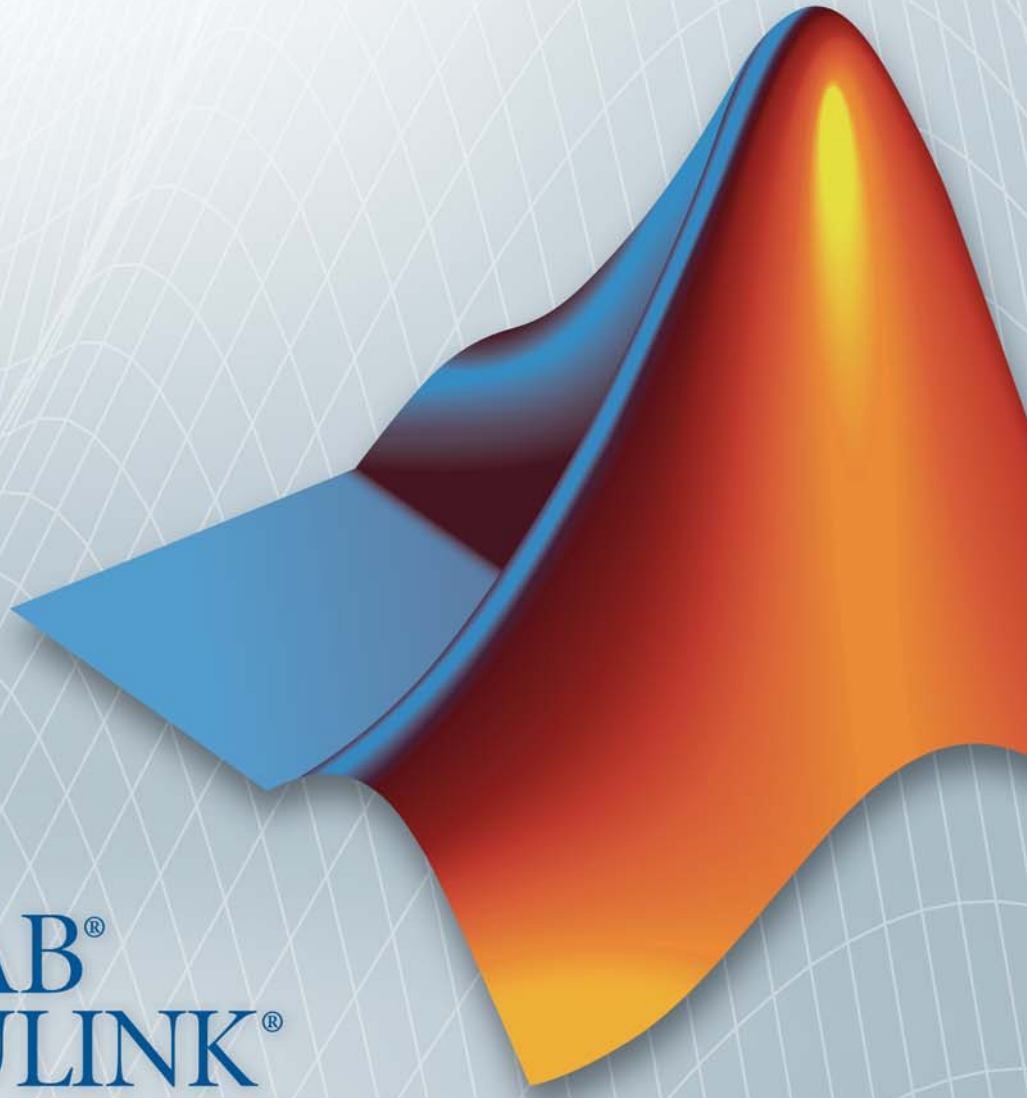
xPC Target™

Getting Started Guide

R2012b



MATLAB®
&SIMULINK®

MathWorks®

How to Contact MathWorks

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

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

508-647-7000 (Phone)

508-647-7001 (Fax)

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

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

xPC Target™ Getting Started Guide

© COPYRIGHT 2000–2012 by The MathWorks, Inc.

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

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

Trademarks

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

Patents

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

Revision History

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

Contents

1 Introduction

2 Installation and Configuration

4 Tutorial and Examples

1

Introduction

Product Description

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

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

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

Key Features

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

Using the Product

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

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

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

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

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

- MATLAB Central File Exchange for xPC Target Product (`http://www.mathworks.com/matlabcentral/fileexchange/?term=xPC+Target`)
- MathWorks Support xPC Target Web site (`http://www.mathworks.com/support/search_results.html?q=product:"xPC+Targ`

Required Knowledge

Users who read this documentation should be familiar with

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

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

If you are a new user — Begin with "Getting Started with xPC Target". This topic gives you an overview of the xPC Target features and xPC Target environment. Next, read and try the examples in "Application and Driver Scripts" on page 4-65.

If you are an experienced user — After you are familiar with using the xPC Target software, review the following topics:

- "Setting Configuration Parameters"
- "Setup and Configuration" on page 2-3
- "Command Line Setup for Single Target Computer Systems"
- "Command Line Setup for Multiple Target Computer Systems"

Product Features

In this section...

Real-Time Kernel

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

However, the xPC Target Embedded Option product requires DOS and a DOS license at boot time. For more information, see "Stand Alone Boot Method" on page 2-59.

Target Boot Options

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

- Removable boot devices

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

- Fixed boot devices

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

- Network boot

  Dedicated network

Target Computer BIOS

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

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

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

Real-Time Kernel

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

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

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

Real-Time Application

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

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

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

For more information on creating a target application, see “Create and Run Simulink Model” on page 4-16.

Signal Acquisition

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

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

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

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

For information on acquiring signal data with the xPC Target software, see:

- "User Interaction" on page 1-25
- "Monitor Signals with MATLAB Language"
- "Signal Logging Basics"
- "Signal Tracing Basics"

Parameter Tuning

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

- Interactive — The xPC Target software supports tuning of parameters while the target application is running in real time.
- Scripts and batch procedures — The xPC Target software also includes commands to change parameters during a run or between runs. By writing a script that incrementally changes a parameter and monitors a signal output and running it on the host computer, you can optimize the value of that parameter.

For information on tuning parameters with the xPC Target software, see:

- "User Interaction" on page 1-25
- "Parameter Tuning Basics"

xPC Target Embedded Option

Run xPC Target applications on standalone target computers

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

Key Features

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

Fixed-Point Support

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

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

MATLAB Compiler Support

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

Standalone applications that include xPC Target functionality have the following limitations:

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

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

BLAS Library Support

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

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

- “Introduction to Code Replacement Libraries”

- Code Replacement Library (CRL) and Embedded Targets

Note Your model accesses the XPC_BLAS library when you build your model with a Microsoft Visual C/C++ compiler.

Hardware Description

In this section...

Introduction

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

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

Host Computer

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

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

For more details on the requirements of the host computer, see "Host Computer Hardware" on page 2-4.

Target Computer

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

A target computer can be one of the following:

- Desktop Computer — This computer is booted from a special target boot kernel created by the xPC Target software and stored on a target boot disk, removable boot drive, or network boot image.

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

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

- Industrial Computer — This computer is booted from a special target boot kernel stored on a target boot disk, removable boot drive, hard disk, flash memory, or network boot image.

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

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

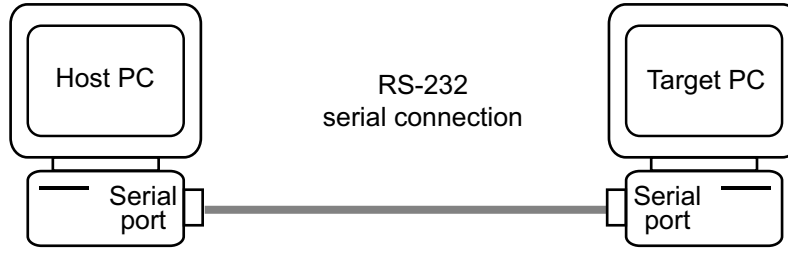
Note Do not use a laptop PC as a target computer. For target computer hardware, consider the xPC Target Turnkey solutions.

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

Host-Target Connection

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

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

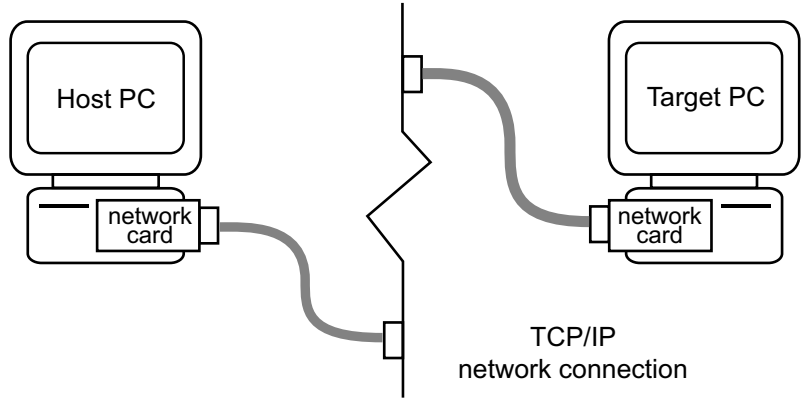


For detailed information on setting up the hardware and software for serial communication, see "Serial Communication Setup" on page 2-36.

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

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

When using a network connection, the target computer requires a supported Ethernet adapter card. The data transfer rate can be 10 megabits/second, 100 megabits/second, or 1 gigabit/second. For a list of supported cards, see `http://www.mathworks.com/products/xpctarget/-supported-hardware/xPC_Target_Supported_Ethernet_Chipsets.pdf`.



For detailed information on setting up the hardware and software for network communication, see "Network Communication Setup" on page 2-16.

Advantages of Network Communication

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

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

This manual does not include information for installing network cards or the TCP/IP protocol on your host computer. To install and configure your network cards and TCP/IP protocol, contact your system administrator.

I/O Driver Support

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

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

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

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

- 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.
- 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 both dynamic object mode and FIFO mode.

- GPIB support — Special RS-232 drivers support communication with a GPIB control module from National Instruments® to external devices with a GPIB connector.
- UDP support — Communicate with another system using the standard UDP/IP network protocol.
- 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.
- Incremental encoder — Change motion into numerical information for determining position, direction of rotation, and velocity.
- Shared memory — Use shared memory blocks with multiprocessing applications.
- LVDT — Use the North Atlantic Industries, Inc. 73LD3, 76CL1, 76LD1, and 76CL1 boards with xPC Target LVDT blocks to work with LVDT applications.
- ARINC-429 — Use the Condor Engineering CEI-X20 boards with xPC Target ARINC-429 blocks to interface with the ARINC 429 data bus.
- MIL-STD-1553 — Use the Condor Engineering PCI-1553 and QPCI-1553 series boards with xPC Target MIL-STD-1553 blocks to interface with the MIL-STD-1553 data bus.
- Audio — Use the audio blocks to work with audio applications.
- Thermocouple — Use the Measurement Computing™ PCI-DAS-TC board with xPC Target thermocouple blocks to work with thermocouple applications.

Real-Time Test Environment

In this section...

Software Setup

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

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

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

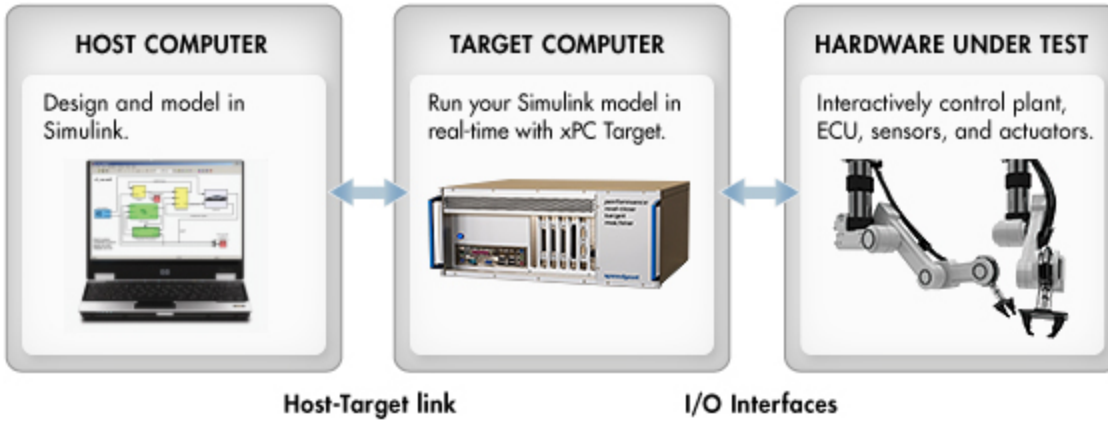
- xPC Target
- Simulink
- Simulink Coder
- MATLAB Coder
- A C compiler

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

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

Once you make the connections, you can:

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



Host-Target Communication

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

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

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

xPC Target Embedded Option

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

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

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

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

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

For more information on the xPC Target Embedded Option product, see "Stand Alone Boot Method" on page 2-59.

User Interaction

In this section...

Introduction

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

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

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

Interface	Environment Properties	Control	Signal Acquisition	Parameter Tuning
"xPC Target Explorer" on page 1-26	X	X	X	X
"MATLAB Command-Line Interface" on page 1-27	X	X	X	X
"Simulink External Mode Interface" on page 1-29		X	X	X
"Simulink with xPC Target Blocks" on page 1-30			X	
"Target Computer Command-Line Interface" on page 1-30		X	X	X
"Web Browser Interface" on page 1-31		X	X	X
"Custom GUI with xPC Target API for Microsoft .NET Framework" on page 1-31		X	X	X
"Custom GUI with xPC Target C API" on page 1-32		X	X	X
"Custom GUI with xPC Target COM API" on page 1-32		X	X	X

xPC Target Explorer

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

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 setting environment using xPC Target Explorer, see “Setup and Configuration” on page 2-3

- Control — Use the xPC Target Explorer to download and execute a model. 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 **Scopes** pane to interactively add host, target, or file scopes, add or remove signals, and display simulation results.

  For more information on using scopes with the xPC Target Explorer, see “Trace Signals with Host Scope (xPC) Blocks”.

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

MATLAB Command-Line Interface

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

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

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

The MATLAB command-line interface includes the following functions:

- Environment — Create a target boot kernel and directly change the environment properties without using a graphical interface. See:

  - "Command Line Setup for Single Target Computer Systems"
  - "Command Line Setup for Multiple Target Computer Systems"

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

  For more information, see "Execute Target Application Using MATLAB Language" on page 4-62.

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

  For more information, see "Monitor Signals with MATLAB Language" "Trace Signals with Target Scopes Using MATLAB Language" and "Log Signals Using Outport with MATLAB Language".

- 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 "Tune Parameters Using MATLAB Language".

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 "Trace Signals with Simulink External Mode".

- 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 "Trace Signals with Simulink External Mode".

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

For more information, see "Tune Parameters with Simulink External Mode".

Simulink with xPC Target Blocks

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

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

- For information on acquiring signal data with `Scope` blocks, see:
  - "Add xPC Target Scope Block" on page 4-35
  - "Set Target Scope Block Parameters" on page 4-37
  - "Signal Tracing Basics"
- For information on using xPC Target `To` and `From` blocks to transfer data to and from a Simulink user interface model, see "xPC Target Interface Blocks to Simulink Models".

Target Computer Command-Line Interface

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

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

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

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

Web Browser Interface

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

The Web browser interface includes the following functions:

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

  For more information, see “Web Browser Interface”.

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

  For more information, see “Trace Signals with Host Scope (xPC) Blocks”.

  For more information, see “Log Signals with a Web Browser”.

- 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 “Tune Parameters with a Web Browser”.

Custom GUI with xPC Target API for Microsoft .NET Framework

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

For more information, see “Using the xPC Target API for .NET Framework”.

Custom GUI with xPC Target C API

Use the C API to create a GUI application interface to a target application using any development environment that can link in a DLL.

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

For more information, see “Using the C API”.

Custom GUI with xPC Target COM API

Use the COM API to 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 “Using the COM API”.

2

Installation and Configuration

Setup and Configuration

This topic summarizes the initial hardware and software requirements, setup, and configuration using xPC Target Explorer.

Host Computer Hardware

In the xPC Target software environment, the host computer is usually your desktop computer. Here you install the xPC Target and xPC Target Embedded Option products, along with MATLAB, Simulink, and other required and optional software. A notebook computer is also a viable host computer.

Peripherals

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

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

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

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

Communication

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

- One available **Ethernet interface** (PCI, ISA, or USB) connected to a network (see "Network Communication Setup" on page 2-16 for details)
- One available **serial port** (COM1 or COM2) with a 9-pin or 25-pin D-sub connector (see "Serial Communication Setup" on page 2-36 for details)

Target Computer Hardware

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

Tip If you are using xPC Target Turnkey, you may disregard this section.

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

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

Note

- To support xPC Target, 64-bit target computers must run in 32-bit mode.
- Do not use a laptop PC as a target computer.
- The xPC Target kernel can use only 2 GB of memory. You can acquire several megabytes of data during a run, depending on how much memory you install in the target computer.

Target Computer BIOS Settings

The xPC Target kernel does not require an operating system installed on the target computer. Target computer BIOS settings such as the following are required to run the xPC Target software:

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

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

- USB communication — If you are using USB communications, enable USB ports for the target computer.
- Plug-and-Play (PnP) operating system — Disable this feature so the PCI BIOS can set up the plugged-in PCI cards. The xPC Target kernel is not a PnP operating system; this feature must be disabled or PCI devices will not work on the xPC Target product.
- Power saving modes — Disable all power saving modes.
- PCI boards — Do not detect PCI boards with class code `0xff` in the target computer BIOS. Turn this option `Off` to enable the BIOS to detect and configure all boards.
- Hyper-threading — If your target computer supports hyper-threading capabilities, do not enable these capabilities. Enabling hyper-threading can degrade the performance of the target computer.
- Multicore processor — If your target computer includes a multicore processor (see “Multicore CPU Support” on page 2-7), you can configure the xPC Target software to take advantage of the individual cores. See “Multicore Processor Configuration” and “Design Considerations”.

Note

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

- Boot order — Set the boot order for the target computer BIOS. You can boot the target computer using the following methods:
  - Boot floppy disk
  - CD/DVD bootable ROM
  - Flash drive
  - Removable hard drive
  - Dedicated network boot
  - Bootable hard drive

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

Multicore CPU Support

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

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

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

Peripherals

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

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

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

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

- You can copy files to a hard drive or flash memory and boot from that device.
- If you want to access the target computer file system on a hard drive, see "File Systems".
- The hard drive must be a parallel ATA (PATA)/Integrated Device Electronics (IDE) or serial ATA (SATA) drive. For better performance, configure this drive as a primary master.
- Verify the hard drive is not cable-selected.
- The xPC Target product supports file systems of type FAT-12, FAT-16, or FAT-32.

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

Note If a keyboard is not connected, the BIOS might display an error message (keyboard failure). With a current BIOS, you can use the BIOS setup to skip the keyboard test.

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

Communication

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

- One free supported **Ethernet adapter** (PCI, ISA, or USB) connected to a network (see "Network Communication Setup" on page 2-16 for supported Ethernet adapters).

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

- One free **serial port** (COM1 or COM2) with a 9-pin or 25-pin D-sub connector (see "Serial Communication Setup" on page 2-36 for details). Use a serial null modem cable to connect the target computer to the host computer.

PC-Compatible Form Factors

xPC Target supports the following target computer hardware form factors:

- ISA
- PCI
- PMC

- PC/104 and PC/104+
- PCIe
- CompactPCI

I/O Boards

You can install inexpensive I/O boards in the PCI or ISA slots of the target computer. These boards provide a direct interface to the sensors, actuators, or other devices for real-time control or signal processing applications. The xPC Target software supports I/O functionality via the blocks in xpclib. You can also write a custom driver.

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

Host Software Installation

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

Note

- Verify that the xPC Target and xPC Target Embedded Option products are not already installed on your host computer. If they are, uninstall them both before proceeding.
- To install products in the xPC Target family, you must have a valid license. See "License Requirements" on page 2-12.
- Microsoft .NET Framework 4.0 is required for host computer software installation. If the software detects its absence, see the section "Installing the Microsoft .NET Framework" on page 2-15.
- xPC Target requires an installed Microsoft C compiler. See "Command Line C Compiler Configuration".
- If you have purchased an xPC Target Turnkey system, install your MATLAB products on your system before installing the xPC Target Turnkey software. See your xPC Target Turnkey system user documentation for further information.
- For more about the installation file structure, including the location of examples, see "Files on the Host Computer" on page 2-13.

Continue with "Network Communication Setup" on page 2-16 or "Serial Communication Setup" on page 2-36.

License Requirements

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

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

The xPC Target family of software includes options that you can purchase and add later to the xPC Target environment. You must get a valid license for the xPC Target product, and a separate license for the xPC Target Embedded Option product.

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

Files on the Host Computer

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

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

  Tip Select a working folder outside the MATLAB root. See "Setting an Initial Working Folder" on page 2-14.

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

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

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

Setting an Initial Working Folder

You should set your initial MATLAB working folder outside the MATLAB root folder. The default MATLAB root folder is `c:\matlab`. If your MATLAB working folder is below or inside the MATLAB root, files created by Simulink and Simulink Coder are mixed with the MATLAB directories.

You can use the `cd` command to temporarily set your working folder, or set your initial working folder in the MATLAB shortcut:

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

To check your working folder, in the MATLAB Command Window, type `pwd`.

Installing the Microsoft .NET Framework

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

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

Network Communication Setup

This topic describes using network communications (TCP/IP) to connect the host computer and target computer. For serial communication, see "Serial Communication Setup" on page 2-36.

Use one of these procedures as required:

- "PCI Bus Ethernet Setup" on page 2-17
- "USB-to-Ethernet Setup" on page 2-24
- "ISA Bus Ethernet Setup" on page 2-30

Continue with "Target Boot Methods" on page 2-41.

PCI Bus Ethernet Setup

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

Follow these procedures:

1. "PCI Bus Ethernet Hardware" on page 2-18
2. "PCI Bus Ethernet Settings" on page 2-20

Continue with "Target Boot Methods" on page 2-41.

PCI Bus Ethernet Hardware

To install the PCI bus Ethernet card:

1. Acquire a supported PCI bus Ethernet card.

   For the most current network communications requirements, see `http://www.mathworks.com/products/xpctarget/-supported-hardware/xPC_Target_Supported_Ethernet_Chipsets.pdf`.

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

2. Turn off your target computer.

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

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

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

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

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

   You can directly connect your computers using a crossover UTP cable with RJ45 connectors. Both computers must have static IP addresses. If the

host computer has a second network adapter card, that card can have a DHCP address.

Continue with "PCI Bus Ethernet Settings" on page 2-20.

PCI Bus Ethernet Settings

After you install the PCI bus Ethernet card, specify the environment properties for the host and target computers.

Note You must specify these properties before you can build and download a target application.

1. Type `xpcexplr` in the MATLAB Command Window.
2. In the **Targets** pane, expand the target computer node.
3. Click the Target Properties icon on the toolbar or double-click **Properties**.

   Tip

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

4. In the **Target Properties** pane, click **Host-to-Target communication**.
5. Select **Communication type** `TCP/IP`.
6. Set **IP address** to the IP address for your target computer (for example `10.10.10.15`).

   Ask your system administrator for this value.
7. Set **Subnet mask** to the subnet mask address of your LAN (for example `255.255.255.0`).

   Ask your system administrator for this value.

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

Ask your system administrator for this value.

9 Set **Gateway** (optional) to the gateway required to access the target computer, if any. This property is set by default to `255.255.255.255`, which means that you do not use a gateway to connect to your target computer.

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

Ask your system administrator for the IP address of the required gateway.

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

---

10 Select **Bus type** `PCI`.

11 For **Target driver** `Auto` (default). If you need to select a particular driver, see `http://www.mathworks.com/products/xpctarget/-supported-hardware/xPC_Target_Supported_Ethernet_Chipsets.pdf` and select the appropriate driver from the list.

Tip

- For **Target driver** `Auto`, the software determines the target computer TCP/IP driver from the card installed on the target computer.
- If no supported Ethernet card is installed in your target computer, the software returns an error.

---

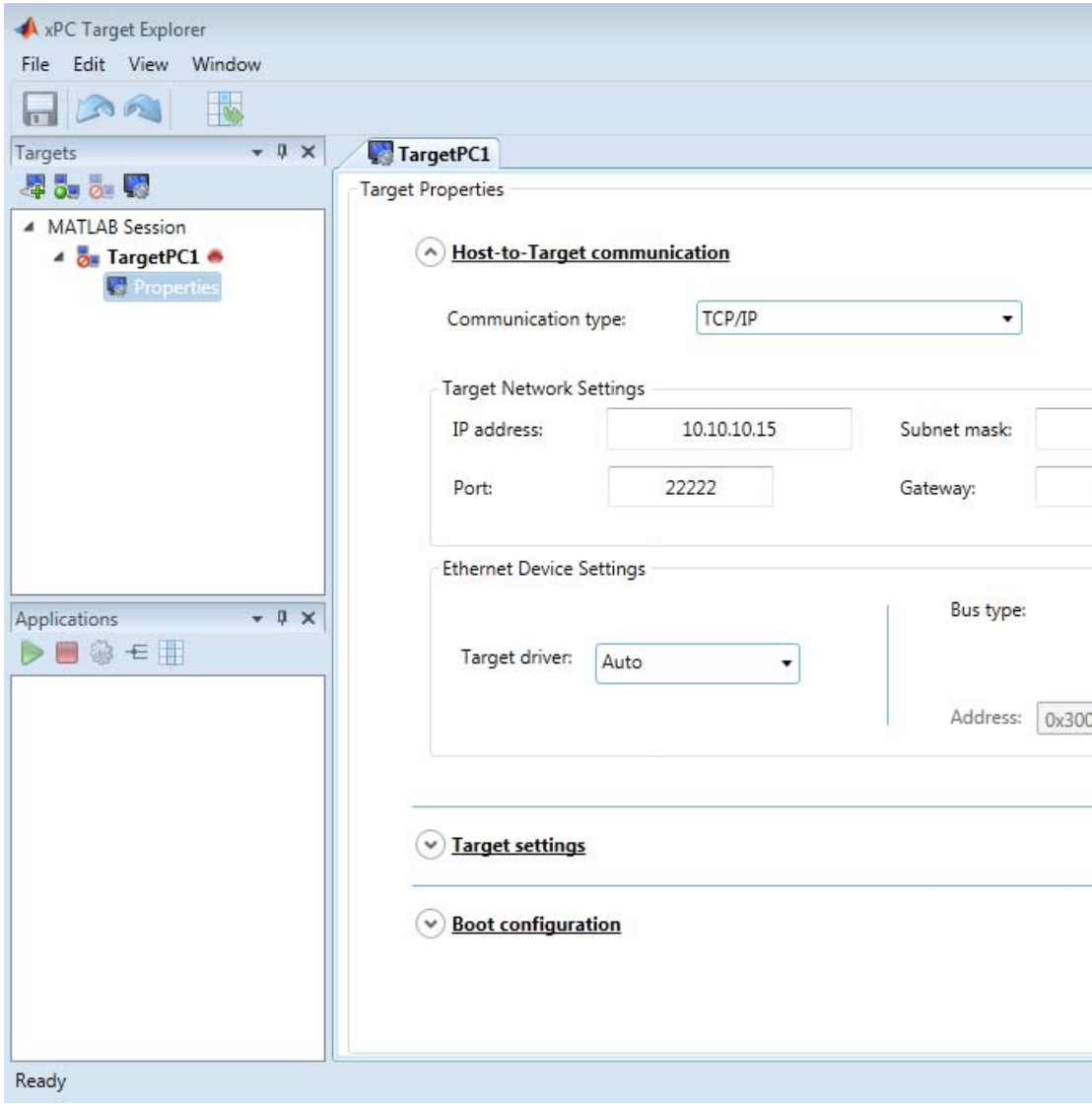
12 If the target computer has multiple Ethernet cards, follow the procedures in one of:

- "Ethernet Card Selection by EthernetIndex"
- "Ethernet Card Selection by EthernetIndex: Multiple Target Computers"

13 Click in the **Target Properties** workspace, then click the Save icon  on the toolbar, or click **File > Save**.

14 Repeat this procedure for any target computer for which you have a PCI bus Ethernet connection between the host computer and target computer.

The xPC Target Explorer window looks like this:



Continue with "Target Boot Methods" on page 2-41.

USB-to-Ethernet Setup

If the target computer has a USB 2.0 port but no PCI or ISA Ethernet card, use a USB-to-Ethernet adapter.

Follow these procedures:

1. "USB-to-Ethernet Hardware" on page 2-25
2. "USB-to-Ethernet Settings" on page 2-27

Continue with "Target Boot Methods" on page 2-41.

USB-to-Ethernet Hardware

To install the USB-to-Ethernet adapter:

1. Acquire a supported USB-to-Ethernet adapter.

   For the most current network communications requirements, see `http://www.mathworks.com/products/xpctarget/-supported-hardware/xPC_Target_Supported_Ethernet_Chipsets.pdf`.

   Note To boot the target computer from the network, you must install on the target computer a USB-to-Ethernet adapter compatible with the Preboot eXecution Environment (PXE) specification.

2. Turn off your target computer.

3. Plug an Ethernet-to-USB adapter into the USB port on the target.

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

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

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

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

   You can directly connect your computers using a crossover UTP cable with RJ45 connectors. Both computers must have static IP addresses. If the

host computer has a second network adapter card, that card can have a DHCP address.

Note Do not connect the host computer to the target computer using a USB cable. As seen from the host computer, a USB-to-Ethernet adapter plugged into the target computer USB port is an Ethernet card on the target computer.

Continue with "USB-to-Ethernet Settings" on page 2-27.

USB-to-Ethernet Settings

After you have installed the USB-to-Ethernet adapter, specify the environment properties for the host and target computers.

Note You must specify these properties before you can build and download a target application.

---

1. Type `xpcexplr` in the MATLAB Command Window.
2. In the **Targets** pane, expand the target computer node.
3. Click the Target Properties icon  on the toolbar or double-click **Properties**.

   Tip

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

   ---

4. In the **Target Properties** pane, click **Host-to-Target communication**.
5. Select **Communication type** `TCP/IP`.
6. Set **IP address** to the IP address for your target computer (for example `10.10.10.15`).

   Ask your system administrator for this value.
7. Set **Subnet mask** to the subnet mask address of your LAN (for example `255.255.255.0`).

   Ask your system administrator for this value.

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

   Ask your system administrator for this value.

9 Set **Gateway** (optional) to the gateway required to access the target computer, if any. This property is set by default to `255.255.255.255`, which means that you do not use a gateway to connect to your target computer.

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

   Ask your system administrator for the IP address of the required gateway.

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

---

10 Select **Bus type** `USB`.

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

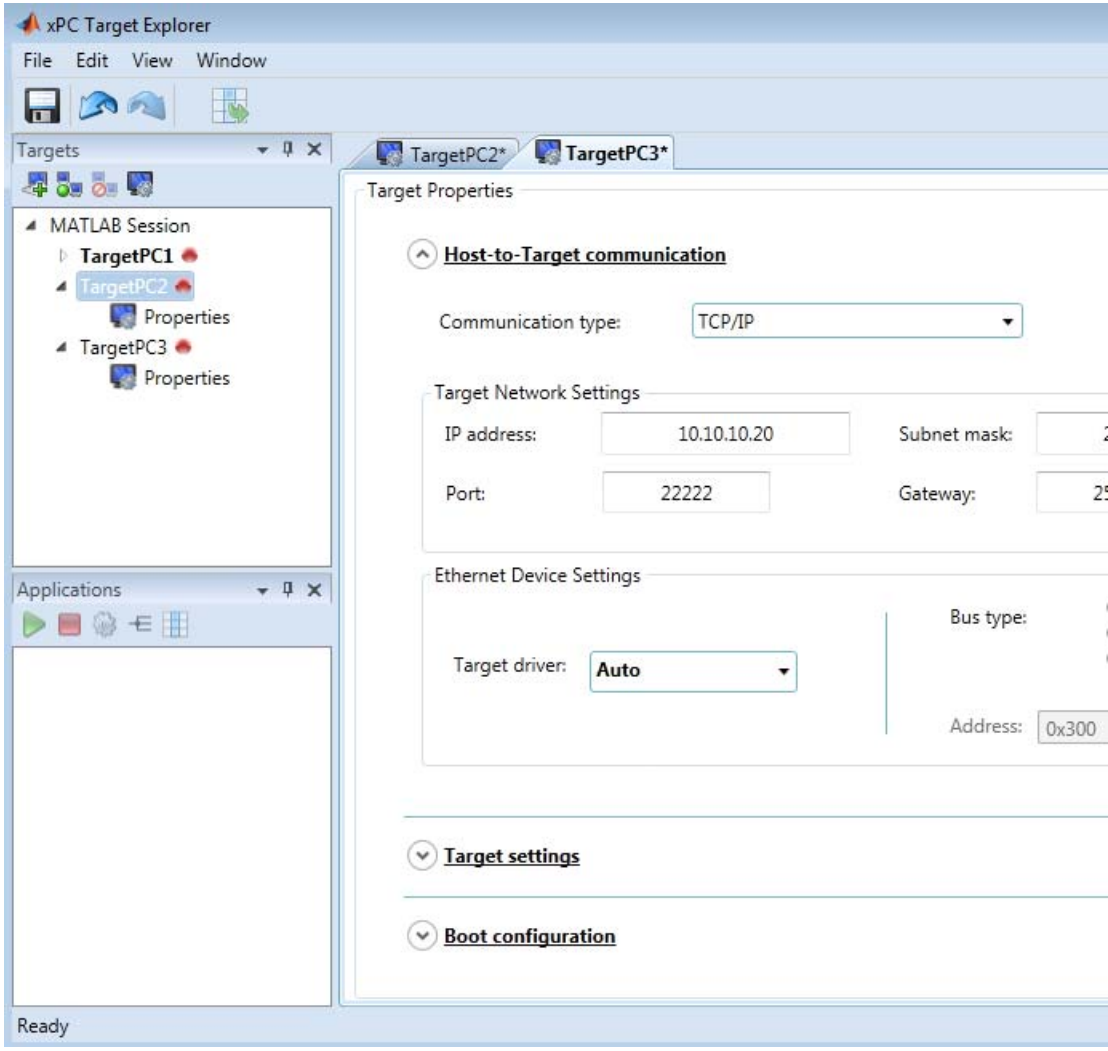
Note If **Target driver** is `Auto`, the software will default the driver to `USBAX772`, the driver most commonly used.

---

12 Click in the **Target Properties** workspace, then click the Save icon  on the toolbar, or click **File > Save**.

13 Repeat this procedure for any target computer for which you have a USB bus Ethernet connection between the host computer and target computer.

The xPC Target Explorer window looks like this:



Continue with "Target Boot Methods" on page 2-41.

ISA Bus Ethernet Setup

Your target computer might not have an available PCI bus slot or USB 2.0 port. In these cases, use an Ethernet card for an ISA bus.

1 "ISA Bus Ethernet Hardware" on page 2-31

2 "ISA Bus Ethernet Settings" on page 2-33

Continue with "Target Boot Methods" on page 2-41.

ISA Bus Ethernet Hardware

To install an ISA bus Ethernet card:

1. Acquire a supported ISA bus Ethernet card.

   For the most current network communications requirements, see `http://www.mathworks.com/products/xpctarget/-supported-hardware/xPC_Target_Supported_Ethernet_Chipsets.pdf`.

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

2. Turn off your target computer.

3. 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 must enter them in xPC Target Explorer.

   Set the IRQ line to `5` and the I/O-port base address to around `0x300`. If one of these hardware settings would lead to a conflict in your target computer, select another IRQ or I/O-port base address.

   If your ISA bus card does not contain jumpers to set the IRQ line and the base address, after installation use the utility on the installation disk supplied with your card to manually assign the IRQ line and base address.

   If you use an Ethernet card for an ISA bus within a target computer that has a PCI bus, after installation 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.

   Note Do not configure the card as a PnP-ISA device.

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

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

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

---

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

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

Continue with "ISA Bus Ethernet Settings" on page 2-33.

ISA Bus Ethernet Settings

After you have installed the ISA bus Ethernet card, specify the environment properties for the host and target computers.

Note You must specify these properties before you can build and download a target application.

---

1. Type `xpcexplr` in the MATLAB Command Window.
2. In the **Targets** pane, expand the target computer node.
3. Click the Target Properties icon  on the toolbar or double-click **Properties**.

   Tip

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

4. In the **Target Properties** pane, click **Host-to-Target communication**.
5. Select **Communication type** `TCP/IP`.
6. Set **IP address** to the IP address for your target computer (for example `10.10.10.15`).

   Ask your system administrator for this value.
7. Set **Subnet mask** to the subnet mask address of your LAN (for example `255.255.255.0`).

   Ask your system administrator for this value.

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

   Ask your system administrator for this value.

9. Set **Gateway** (optional) to the gateway required to access the target computer, if any. This property is set by default to `255.255.255.255`, which means that you do not use a gateway to connect to your target computer.

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

   Ask your system administrator for the IP address of the required gateway.

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

10. Select **Bus type** `ISA`.

11. For **Target driver**, select one of `NE2000` or `SMC91C9X`.

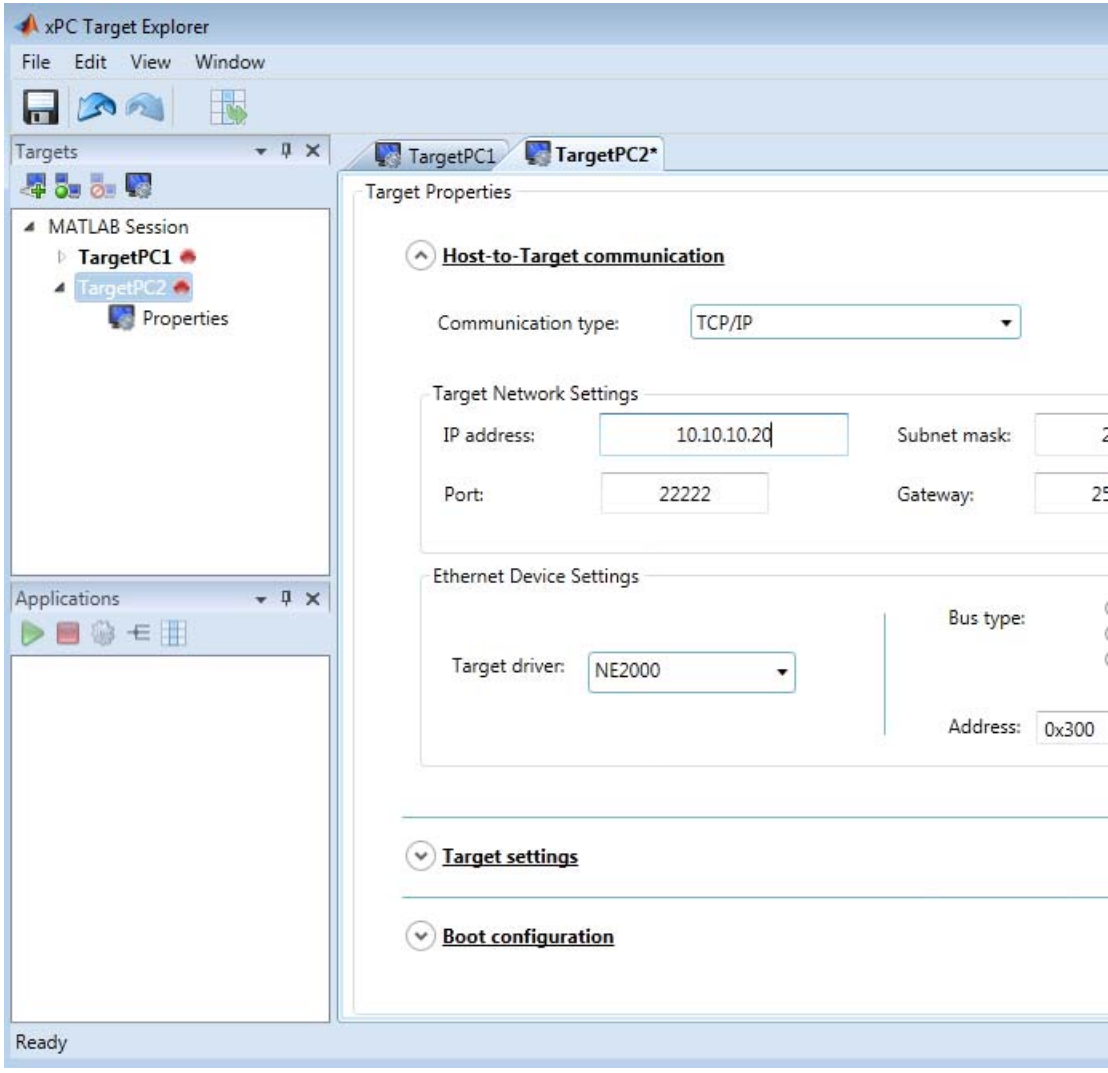
    Note **Target driver** `Auto` is not supported for **Bus type** `ISA`.

12. Set **Address** and **IRQ** to values that correspond with the jumper settings or ROM settings on your ISA bus Ethernet card.

13. Click in the **Target Properties** workspace, then click the Save icon  on the toolbar, or click **File > Save**.

14. Repeat this procedure for any target computer for which you have an ISA bus Ethernet connection between the host computer and target computer.

The xPC Target Explorer window looks like this:



Continue with "Target Boot Methods" on page 2-41.

Serial Communication Setup

This topic describes using serial communications (RS-232) to connect the host computer and target computer. For network communication, see "Network Communication Setup" on page 2-16.

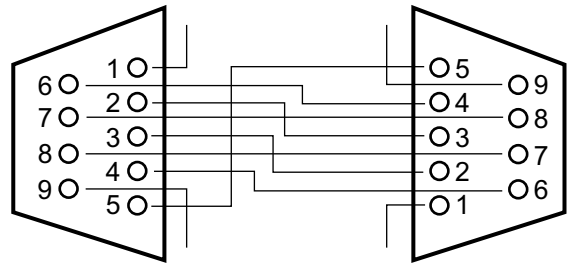
Note RS-232 Host-Target communication mode will be removed in a future release. Use TCP/IP instead.

Continue with "Target Boot Methods" on page 2-41.

RS-232 Hardware

Before you can use serial communication for host-target communication, you must install the following RS-232 hardware:

1 Acquire a null modem cable:



DB9 Female DB9 Female

2 Connect the host and target computers with the null modem cable, using either the COM1 or COM2 port.

Make a note of which port is in use on the host computer. You will need to record the host computer port in the environment property settings.

Continue with "RS-232 Settings" on page 2-38.

RS-232 Settings

After you have installed the serial communication hardware, specify the environment properties for the host and target computers.

Note

- You must specify these properties before you can build and download a target application.
- Do not use host scopes and a scope viewer on the host computer to acquire and display large blocks of data. The slowness of the RS-232 connection causes large delays for large blocks of data.
- **Boot mode** type `Network` is not supported when serial communication is used.

---

1. Type `xpcexplr` in the MATLAB Command Window.
2. In the **Targets** pane, expand the target computer node.
3. Click the Target Properties icon  on the toolbar or double-click **Properties**.

   Tip

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

   ---

4. In the **Target Properties** pane, click **Host-to-Target communication**.
5. Select **Communication type** `RS-232`.

The pane changes to display only parameters pertinent to serial communication.

6 For **Host port**, select one of `COM1` or `COM2`. The default is `COM1`. xPC Target selects the target computer port automatically.

7 Select a **Baud rate** as high as possible. The default is `115200`.

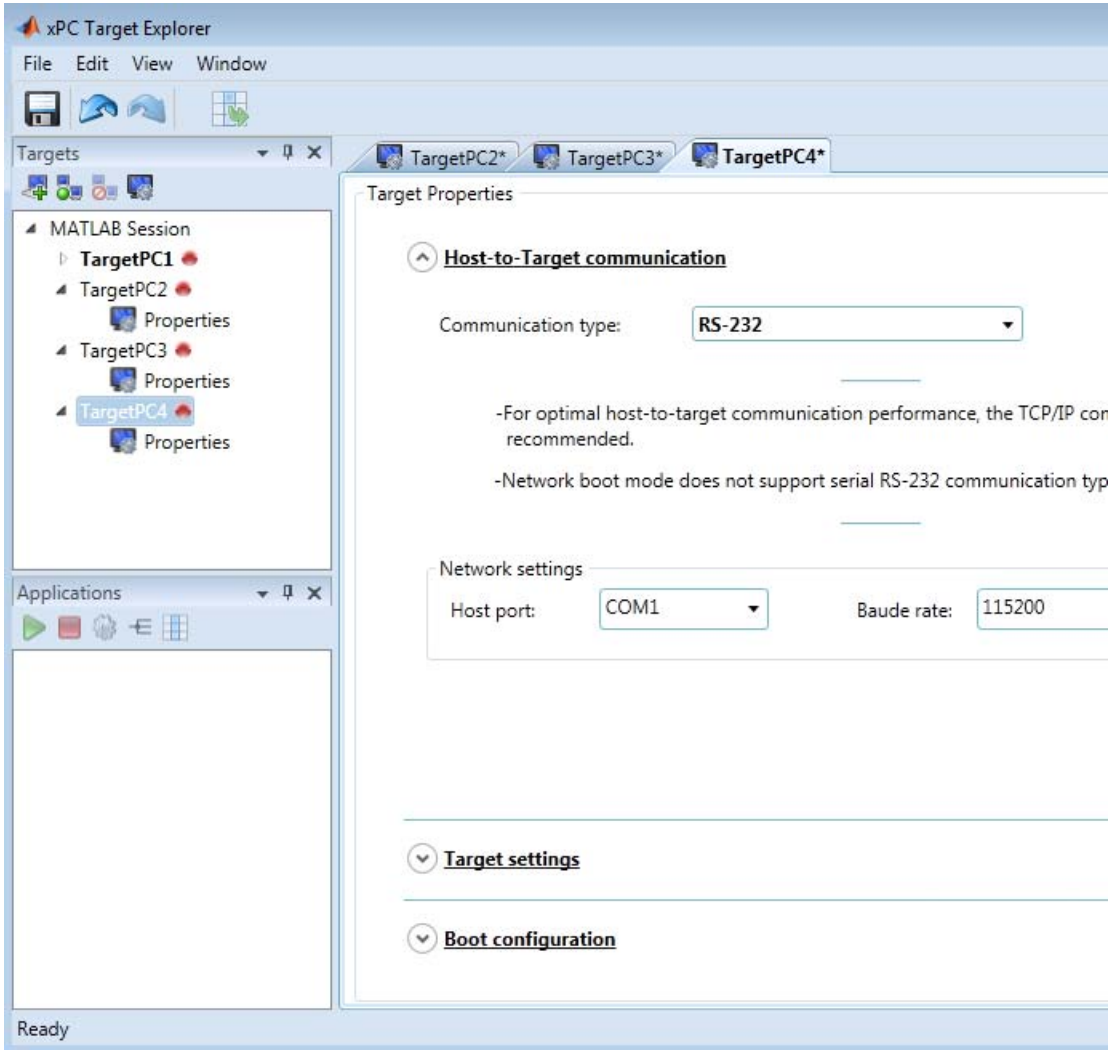
Note A baud rate less than 38400 can cause communication failures.

---

8 Click in the **Target Properties** workspace, then click the Save icon  on the toolbar, or click **File > Save**.

9 Repeat this procedure for any target computer for which you have a serial connection between the host computer and target computer.

The xPC Target Explorer window looks like this:



Continue with "Target Boot Methods" on page 2-41.

Target Boot Methods

You can boot your target computer with the xPC Target kernel using one of several methods.

Tip xPC Target Turnkey systems come with DOS Loader software preinstalled. You can set up the DOS Loader Boot Method on your host or configure another boot method. See your xPC Target Turnkey system user documentation for further information.

- "Preboot Checks" on page 2-42
- "Network Boot Method" on page 2-45
- "CD Boot Method" on page 2-43
- "DOS Loader Boot Method" on page 2-55
- "Stand Alone Boot Method" on page 2-59
- "Removable Disk Boot Method" on page 2-51

Continue with "Run Confidence Test on Configuration" on page 2-69.

Preboot Checks

Configure your xPC Target system before creating the target boot kernel. At a minimum, do the following:

1. Check the physical connections between the host computer and the target computer. If you are using TCP/IP, these are Ethernet connections that may pass through a LAN.
2. Check your target computer BIOS settings (see "Target Computer BIOS Settings" on page 2-6).
3. Check that you have write permission for your current working folder.
4. Type `xpcexplr` in the MATLAB Command Window.
5. In the **Targets** pane, expand the target computer node.
6. Click the Target Properties icon  on the toolbar or double-click **Properties**.
7. In the **Target Properties** pane, click **Host-to-Target communication**.
8. Check the host-to-target communication settings (see "Network Communication Setup" on page 2-16 and "Serial Communication Setup" on page 2-36 as required).
9. In the **Target Properties** pane, click **Boot configuration**.
10. Check that **Boot mode** is set to the required value.
11. Click in the **Target Properties** workspace, then click the Save icon  on the toolbar, or click **File > Save**.
12. Repeat this procedure for each target computer.

CD Boot Method

After you have configured the target computer environment parameters, you can use a target boot CD or DVD to load and run the xPC Target kernel. This topic describes using xPC Target Explorer to create a boot CD or DVD. To use this capability, your host computer must run under one of the following Windows systems:

- Microsoft Windows 7
- Microsoft Windows Vista™
- Microsoft Windows XP Service Pack 2 or 3 with Image Mastering API v2.0 (IMAPIv2.0), available at `http://support.microsoft.com/kb/KB932716`.

Tip To create a bootable CD/DVD using MATLAB language, see "Command Line CD Boot Method" (single target computer) or "Command Line CD Boot Method: Multiple Target Computers" (multiple target computers).

To create a boot CD or DVD:

1. Type `xpcexplr` in the MATLAB Command Window.
2. In the **Targets** pane, expand the target computer node.
3. Click the Target Properties icon  on the toolbar or double-click **Properties**.
4. Select **Boot configuration** and set **Boot mode** to `CD`.
5. Click **Create boot disk**.
6. Select the CD/DVD read/write drive in the **Select CD-ROM Drive** list.
7. When prompted, insert an empty CD or DVD in the host computer CD/DVD read/write drive.
8. Click **Burn Disk**.

   A progress bar is displayed.

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

10 Remove the CD or DVD from the drive.

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

12 Click in the **Target Properties** workspace, then click the Save icon  on the toolbar, or click **File > Save**.

13 Repeat this procedure for each target computer booting from a CD/DVD.

Continue with "Run Confidence Test on Configuration" on page 2-69.

Network Boot Method

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

Note

- Do not boot a target computer on a corporate or nondedicated network. Doing so might interfere with dynamic host configuration protocol (DHCP) servers and cause problems with the network.
- Your Ethernet card must be compatible with the Preboot eXecution Environment (PXE) specification.
- If the target computer and host computer communicate by serial communication (RS-232), you cannot boot the target computer across the network.
- If Stand Alone mode is enabled, you cannot boot the target computer across the network.

Use these procedures:

1. "Network Communication Setup" on page 2-16
2. "Network Boot Procedure" on page 2-46

Network Boot Procedure

1. Type `xpcexplr` in the MATLAB Command Window.
2. In the **Targets** pane, expand the target computer node.
3. Click the Target Properties icon  on the toolbar or double-click **Properties**.
4. Select **Boot configuration** and set **Boot mode** to `Network`.
5. If required, click **Reset**.

   This clears the MAC address. The next time the target computer restarts, by default the software automatically obtains the MAC addresses of accessible target computers and displays them for confirmation in the **xPC Target Network Boot** dialog box.
6. If you want to enter the MAC address of the target computer manually, click **Reset** and enter it in the **MAC address** box in the format `xx:xx:xx:xx:xx:xx`.

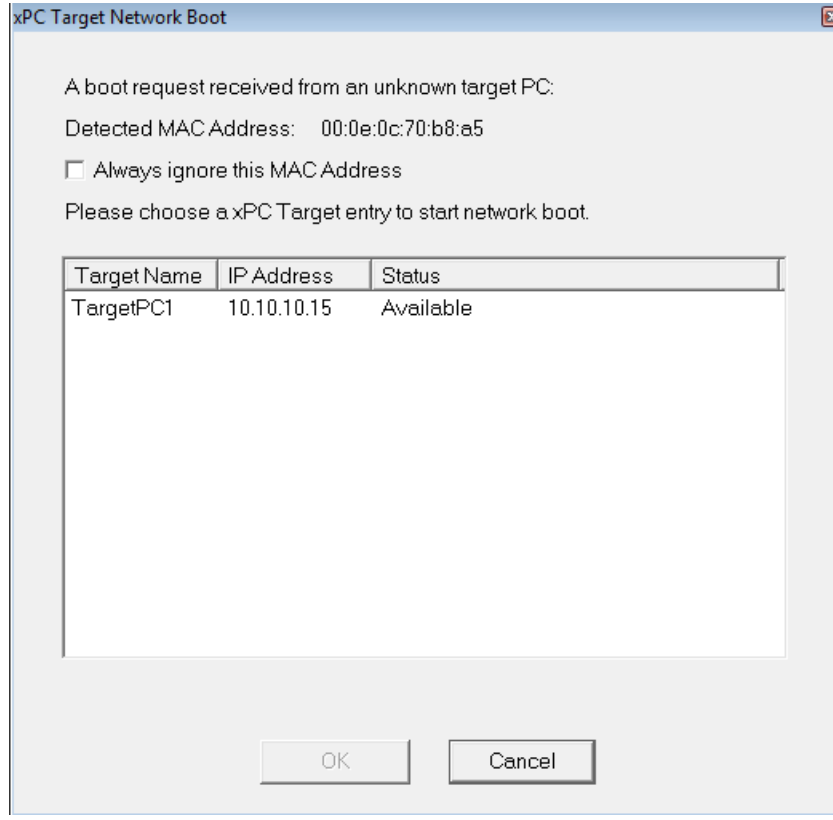
   The next time the target computer restarts, the software selects and boots the target computer that matches this MAC address. The **xPC Target Network Boot** dialog box does not appear.
7. Click **Create boot disk**.

   The software creates and starts a network boot server process on the host computer, as indicated by a minimized icon () in the bottom right system tray on the host computer.
8. Turn on the target computer.
9. Enter the target computer BIOS and set up the target computer for a LAN or network boot.

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

10 Reboot the target computer.

11 **First time or after reset:** The first time the network boot server process detects a target computer, it displays the **xPC Target Network Boot** dialog box, which contains physical target computer names and the corresponding IP addresses.



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

b Click **OK**.

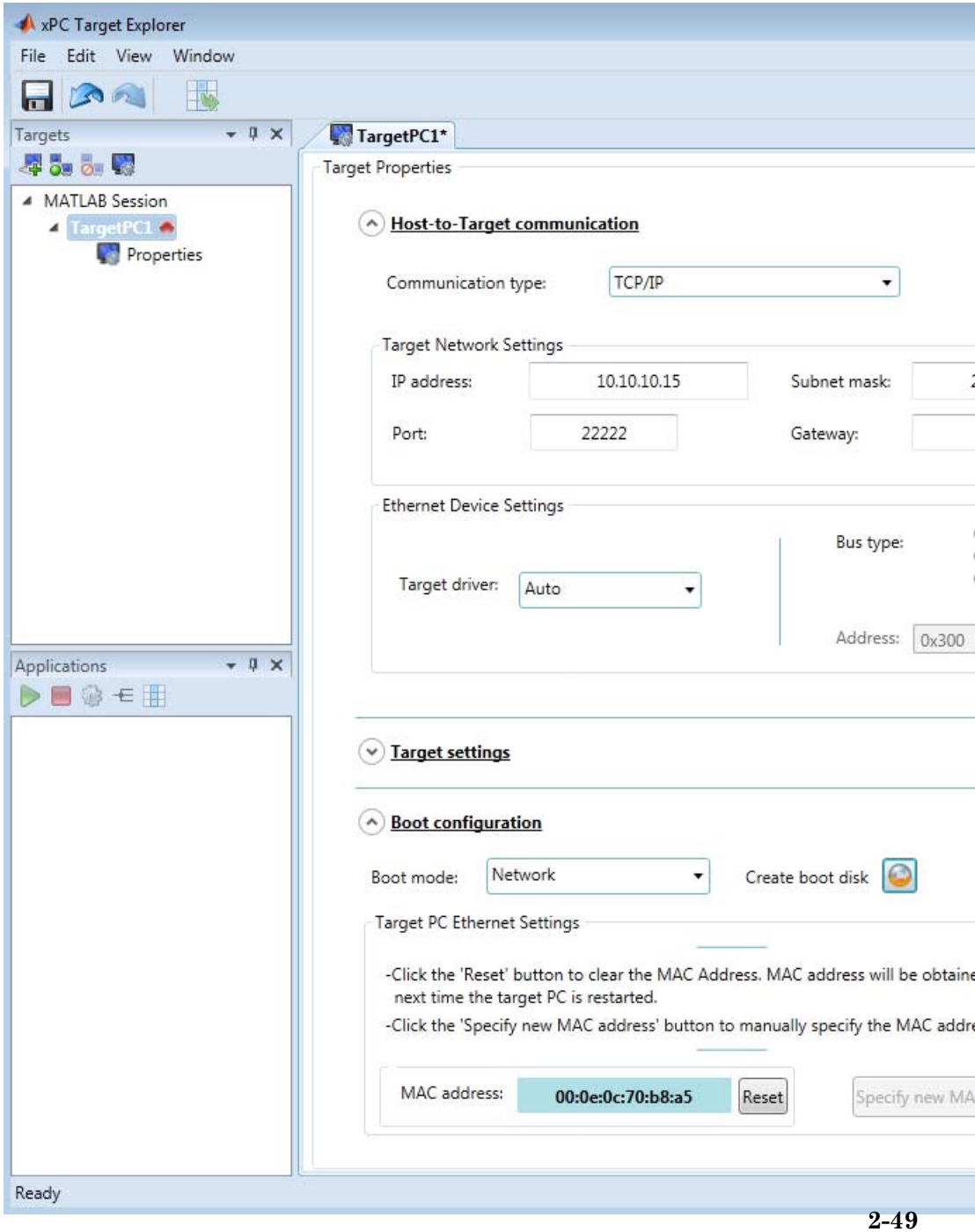
Note If you click **Cancel** instead of selecting a target computer name, the kernel will ignore the boot request for 90 seconds the next time you try to boot the target computer across the network.

The target computer receives the xPC Target kernel and boots with this kernel. The host computer network boot server displays a pop-up from the system tray, indicating that the boot server is being downloaded to the target computer.

12 Click in the **Target Properties** workspace, then click the Save icon  on the toolbar, or click **File > Save**.

13 Repeat this procedure for each target computer booting from a dedicated network.

The xPC Target Explorer window looks like this:


xPC Target Explorer
File Edit View Window
Targets
MATLAB Session
TargetPC1
Properties
Applications
TargetPC1*
Target Properties
Host-to-Target communication
Communication type:
TCP/IP
Target Network Settings
IP address:
10.10.10.15
Subnet mask:
Port:
22222
Gateway:
Ethernet Device Settings
Bus type:
Target driver:
Auto
Address:
0x300
Target settings
Boot configuration
Boot mode:
Network
Create boot disk
Target PC Ethernet Settings
-Click the 'Reset' button to clear the MAC Address. MAC address will be obtaine
next time the target PC is restarted.
-Click the 'Specify new MAC address' button to manually specify the MAC addre
MAC address:
00:0e:0c:70:b8:a5
Reset
Specify new MA
Ready

Continue with "Run Confidence Test on Configuration" on page 2-69.

Removable Disk Boot Method

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

Tip

- If you are creating a removable boot disk from a USB flash drive, you must create a bootable partition on the drive before performing this procedure. See "Creating a Bootable Partition" on page 2-53.
- To create a removable boot disk using MATLAB language, see "Command Line Removable Disk Boot Method" (single target computer) or "Command Line Removable Disk Boot Method: Multiple Target Computers" (multiple target computers).

To create a removable boot disk:

1. Type `xpcexplr` in the MATLAB Command Window.
2. In the **Targets** pane, expand the target computer node.
3. Click the Target Properties icon  on the toolbar or double-click **Properties**.
4. Select **Boot configuration** and set **Boot mode** to `Removable Disk`.
5. If you are creating a removable boot disk from a USB drive, insert the USB drive in the host computer USB port and wait for it to be recognized.
6. Click **Create boot disk**.
7. Select the drive in the **Removable Disk Selector** list.
8. If required, insert an empty removable disk in the host computer drive and click **OK**.

A progress bar is displayed.

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

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

11 Click in the **Target Properties** workspace, then click the Save icon  on the toolbar, or click **File > Save**.

12 Repeat this procedure for each target computer booting from a removable disk.

Continue with "Run Confidence Test on Configuration" on page 2-69.

Creating a Bootable Partition

Before you create a removable boot drive from a USB flash drive, you must create a bootable partition on the drive:

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

   ```
   diskpart
   ```

3. At the diskpart prompt, type

   ```
   list disk
   ```

   Make a note of the disk numbers of the existing host computer disks.
4. Insert the flash drive. Wait for the host computer to recognize the drive. Make a note of the drive device name.
5. At the diskpart prompt, type

   ```
   list disk
   ```

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

   ```
   select disk N
   ```

   All of the next steps change disk *N*.

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

7. Type

   ```
   clean
   ```

8. Type

   ```
   create partition primary
   ```

9. Type

   ```
   select partition 1
   ```

10. Type

    ```
    active
    ```

11. Type

    ```
    format fs=fat32 quick
    ```

12. Type

    ```
    exit
    ```

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

14. Remove the drive from the host computer.

DOS Loader Boot Method

DOS Loader mode allows you to boot the xPC Target kernel on a target computer from a fixed or removable device with DOS boot capability, such as a hard disk or flash memory. After booting the target computer, you can download your application from the host computer over a serial or network connection between the host and target computers.

Note

- xPC Target Turnkey systems come with DOS Loader software preinstalled. You can set up the DOS Loader Boot Method on your host or configure another boot method. See your xPC Target Turnkey system user documentation for further information.
- To run in DOS Loader mode, the target computer boot device must provide a minimal DOS environment complying with certain restrictions. For details, see "Creating a DOS System Disk" on page 2-57 and "DOS Loader Mode Restrictions" on page 2-58.
- To create a DOS Loader boot disk using MATLAB language, see "Command Line DOS Loader Boot Method" (single target computer) or "Command Line DOS Loader Boot Method: Multiple Target Computers" (multiple target computers).

To create DOS Loader files and use them to boot the target computer, use the following procedure:

1. Type `xpcexplr` in the MATLAB Command Window.
2. In the **Targets** pane, expand the target computer node.
3. Click the Target Properties icon  on the toolbar or double-click **Properties**.
4. Select **Boot configuration** and set **Boot mode** to `DOS Loader`.

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

6 Click **OK**.

A progress bar appears to indicate execution.

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

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

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

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

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

10 Select the required boot device in the BIOS of the target computer.

11 Boot the target computer.

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

12 Click in the **Target Properties** workspace, then click the Save icon  on the toolbar, or click **File > Save**.

13 Repeat this procedure for each target computer booting using DOS Loader.

Continue with "Run Confidence Test on Configuration" on page 2-69.

Creating a DOS System Disk

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

Note

- xPC Target software does not include a DOS license. You must obtain a valid DOS license for your target computer.
- MathWorks has tested the xPC Target product with FreeDOS Beta 8 (“Nikita”) distribution, MS-DOS (6.0 or higher), PC DOS, and Caldera OpenDOS. You can use a copy of any of these DOS systems to boot the target computer.

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

```
sys drive1 drive2
```

It is helpful to copy additional DOS utilities to the boot disk, such as:

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

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

DOS Loader Mode Restrictions

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

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

To satisfy these restrictions:

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

Stand Alone Boot Method

The xPC Target Embedded Option software extends the xPC Target base product with Stand Alone mode.

- For details on Stand Alone mode and its uses, see “Stand Alone Mode Embedded Option” on page 2-60.
- To run in Stand Alone mode, the target computer DOS environment must comply with certain restrictions. For details, see “Stand Alone Mode Restrictions” on page 2-62.

Use these procedures:

1. “Stand Alone Target Computer Setup” on page 2-63
2. “Stand Alone Settings” on page 2-64
3. “Stand Alone Target Application Build” on page 2-65
4. “Stand Alone Target Application Transfer” on page 2-66
5. “Stand Alone Target Application Boot Configuration” on page 2-67

Continue by rebooting the target computer and testing your application in Stand Alone mode..

Stand Alone Mode Embedded Option

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

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

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

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

Regardless of the mode, you initially boot your target computer with DOS from any boot device, then DOS starts the xPC Target kernel. After the xPC Target kernel starts, DOS is no longer available on the target computer. You must reboot the target computer without starting the xPC Target kernel.

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

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

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

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

If you want to view signals on the target computer in Stand Alone mode, you must provide a monitor for the target computer and add Scope (xPC) blocks of types `target` and `file` before building the xPC Target application.

If you do not want to view signals on the target computer, you can run your xPC Target system operates as a black box without a monitor or keyboard. Stand Alone applications are automatically set to continue running for an infinite time or until the target computer is turned off.

Stand Alone Mode Restrictions

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

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

To satisfy these restrictions:

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

There are additional restrictions on target and scope settings:

- In the xPC Target Explorer **Target settings** pane, you can only use the **Model size** values `1 MB` and `4 MB`.
- If you want to use Scope (xPC) blocks to display or record output:
  - You must use `Target` or `File` type blocks.
  - You must select the **Start scope when application starts** check box in the block parameters dialog box.

Note These restrictions hold because the host computer is not necessarily available in Stand Alone mode to issue a “start scopes” command.

Stand Alone Target Computer Setup

Stand Alone mode combines the target application with the kernel and boots them together on the target computer from the hard drive or flash memory. The host computer does not need to be connected to the target computer, but it does need to be able to boot from DOS.

Before you start, set up your system as follows:

1. Verify that your target computer hard drive is a serial ATA (SATA) or parallel ATA (PATA)/Integrated Device Electronics (IDE) drive. Verify that the hard drive is not cable-selected and that the BIOS can detect it. xPC Target supports file systems of type FAT-12, FAT-16, or FAT-32.

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

   If required, create a standard DOS boot device from a CD ROM, 3.5–inch floppy drive, flash drive, or hard drive using the procedure in "Creating a DOS System Disk" on page 2-57.

3. Verify that you have host-to-target communication configured. You use this to transfer the kernel/target application files built on the host computer to the target computer.

4. Repeat this procedure for each target computer booting Stand Alone.

Continue with "Stand Alone Settings" on page 2-64..

Stand Alone Settings

Use xPC Target Explorer to set the kernel environment properties. When you are done, you can create a Stand Alone kernel/target application.

Note For Stand Alone mode, you do not create an xPC Target boot disk or network boot image. Instead, you copy files created from the build process to the target computer hard drive.

Use the following procedure:

1. Type `xpcexplr` in the MATLAB Command Window.
2. In the **Targets** pane, expand the target computer node.
3. Click the Target Properties icon  on the toolbar or double-click **Properties**.
4. Select **Boot configuration** and set **Boot mode** to `Stand Alone`.
5. Click in the **Target Properties** workspace, then click the Save icon  on the toolbar, or click **File > Save**.
6. Repeat this procedure for each target computer booting Stand Alone.

Continue with "Stand Alone Target Application Build" on page 2-65..

Stand Alone Target Application Build

After you set xPC Target boot mode to `Stand Alone`, you can use the xPC Target, the Simulink Coder, and a C/C++ compiler in Stand Alone mode to build a combined kernel and target application with utility files.

1. In the MATLAB window, type the name of your Simulink model, for example `xpc_osc3`.

   A Simulink window opens displaying the model.

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

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

   - `autoexec.bat` — This file contains xPC Target-specific code that calls the `xpcboot.com` executable to boot the xPC Target kernel (the `*.rtb` file).
   - `xpcboot.com` — This static file loads and executes the `*.rtb` file. It is part of the xPC Target Embedded Option software.
   - `xpc_osc3.rtb` — This image contains the xPC Target kernel and the target application. It also contains applicable options such as serial or TCP/IP communications and the IP address of the target computer.

   Note

   - A `config.sys` file is not required to boot the kernel using `autoexec.bat` and `xpcboot.com`
   - In Stand Alone mode, you can only use the **Model size** values `1 MB` and `4 MB`. If the compiled target application (DLM) is larger than **Model size**, the build software will generate an error.
   -

3. Repeat this procedure for each target application being run Stand Alone.

Continue with "Stand Alone Target Application Transfer" on page 2-66.

Stand Alone Target Application Transfer

After building the kernel/target application on a host computer, transfer the files to a target computer using the xPC Target FTP functions:

1. Boot the target computer from a target computer boot disk.
2. In the MATLAB Command Window, change folder to the folder that contains the kernel/target application files.
3. Create an xPC Target FTP object in the MATLAB Command Window:

   ```
   f=xpctarget.ftp
   ```

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

   ```
   f.mkdir('xpcfiles')
   f.cd('xpcfiles')
   ```

5. Copy files to the `xpcfiles` folder:

   ```
   f.put('autoexec.bat')
   f.put('xpcboot.com')
   f.put('xpc_osc3.rtb')
   ```

6. Remove the boot disk from the target computer.

Continue with "Stand Alone Target Application Boot Configuration" on page 2-67..

Stand Alone Target Application Boot Configuration

After transferring the files to a target computer, configure the target computer to run the target application automatically upon boot:

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

   Note If the target computer that you want to boot in Stand Alone mode was previously booted from the network boot image, selecting the `Stand Alone` check box should disable the network boot capability.

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

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

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

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

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

4. Reboot the target computer.

   When you boot the target computer, the target computer goes through the following sequence of calls:

   a. `C:\autoexec.bat`

   b. `C:\xpcfiles\autoexec.bat`

c `C:\xpcfiles\xpcboot.com`

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

The target application starts executing as soon as possible after boot.

5 Repeat this procedure for each target application being run Stand Alone.

Continue by rebooting the target computer and testing your application in Stand Alone mode..

Run Confidence Test on Configuration

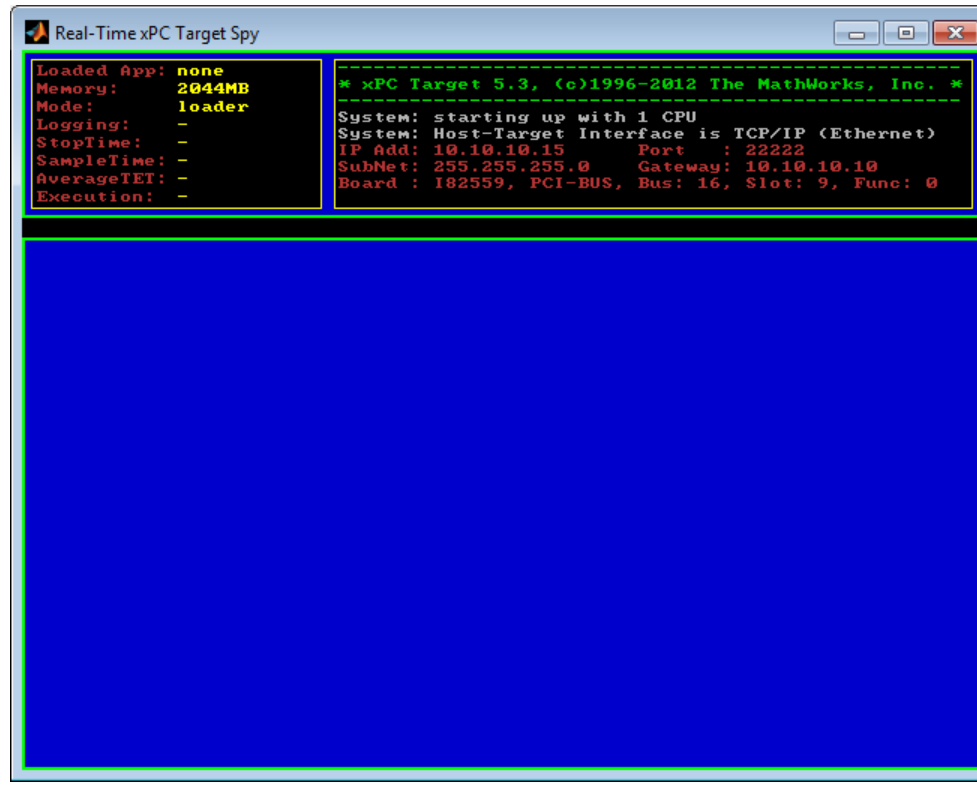
This topic describes how to use a script to test the target boot process, the connection between the host and target computers, and the basic functionality of the xPC Target software. Use this test under the following circumstances:

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

1 If necessary, create a network boot image and reboot the computer according to the procedure in "Network Boot Procedure" on page 2-46.

Note This procedure assumes you are using TCP/IP for host-target communication and for booting the target computer. For more information, see "Network Communication Setup" on page 2-16 and "Network Boot Method" on page 2-45.

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



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

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

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

---

3 Type `xpctest` in the MATLAB Command Window.

MATLAB runs the test script for the default target computer and displays messages indicating whether the test passed or failed. If you use RS-232 communication, the first test is skipped.

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

4 Evaluate the results.

- If all of the tests return `OK`, you are ready to build and download a target application to the target computer.
- If any of the tests return `FAILED`, see "Troubleshooting in xPC Target".

Continue by trying some examples in "Application and Driver Scripts" on page 4-65.

3

Basic Workflows

Rapid Prototyping

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

1 Create a Simulink or Stateflow model.

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

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

---

2 Simulate the model in nonreal time.

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

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

You may log simulation results for later comparison.

3 Configure the host and target environment.

Configure the communication method between the host and target.

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

Configure the host and target environment using:

- "xPC Target Explorer" on page 1-26
- "MATLAB Command-Line Interface" on page 1-27

4 Prepare the model for real-time execution.

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

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

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

5 Configure the build environment.

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

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

6 Connect to external hardware.

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

7 Reboot the target computer.

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

- Target computer hardware boot control.
- "MATLAB Command-Line Interface" on page 1-27

8 Build and download the target application.

Build and download the real-time application using:

- "xPC Target Explorer" on page 1-26 (load and unload only)
- "MATLAB Command-Line Interface" on page 1-27
- "Simulink External Mode Interface" on page 1-29
- "Target Computer Command-Line Interface" on page 1-30
- "Web Browser Interface" on page 1-31

9 Execute the target application in real time.

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

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

Execute using:

- "xPC Target Explorer" on page 1-26
- "MATLAB Command-Line Interface" on page 1-27
- "Simulink External Mode Interface" on page 1-29
- "Target Computer Command-Line Interface" on page 1-30
- "Web Browser Interface" on page 1-31

10 Visualize signals.

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

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

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

Visualize signals using:

- "xPC Target Explorer" on page 1-26
- "MATLAB Command-Line Interface" on page 1-27
- "Simulink External Mode Interface" on page 1-29
- "Simulink with xPC Target Blocks" on page 1-30
- "Target Computer Command-Line Interface" on page 1-30
- "Web Browser Interface" on page 1-31

11 Tune parameters.

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

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

Tune parameters using:

- "xPC Target Explorer" on page 1-26
- "MATLAB Command-Line Interface" on page 1-27
- "Simulink External Mode Interface" on page 1-29
- "Target Computer Command-Line Interface" on page 1-30
- "Web Browser Interface" on page 1-31

12 Prepare regression and stress tests.

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

Hardware in the Loop

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

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

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

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

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

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

4

Tutorial and Examples

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

Set Up and Configure xPC Target

This tutorial assumes the following configuration:

- Single PCI bus target computer
- TCP/IP host-target communication
- Network target boot method

Tip This tutorial assumes that you installed and configured a C compiler as part of xPC Target installation. If not, see "Command Line C Compiler Configuration".

---

1 "Configure Host-to-Target Communication" on page 4-4

2 "Configure Target Settings" on page 4-7

3 "Configure Boot Configuration" on page 4-9

4 "Run the Confidence Test" on page 4-11

Continue with "Create and Run a Real-Time Application" on page 4-14.

Configure Host-to-Target Communication

To run an xPC Target model on a target machine, you must connect the host and target computers in a network.

Tip The **Target Network Settings** values given are representative only. Consult your network administrator for actual values. For more on network configuration, see "Network Communication Setup" on page 2-16.

---

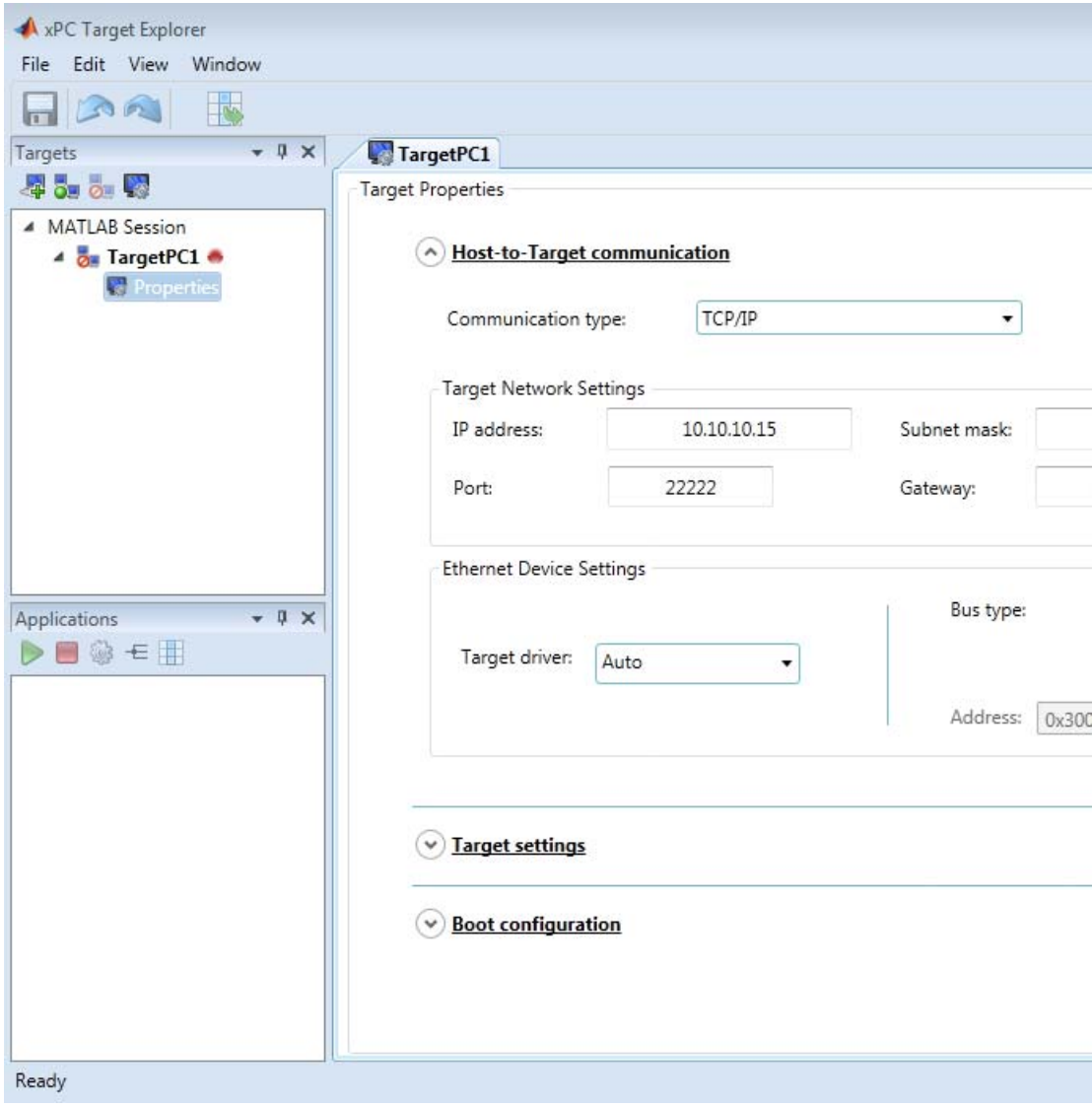
1. Type `xpcexplr` in the MATLAB Command Window.
2. In the **Targets** pane, expand the target computer node.
3. Click the Properties icon  on the toolbar or double-click **Properties**.
4. In the **Target Properties** workspace, click **Host-to-Target communication**.
5. Select **Communication type** `TCP/IP`.
6. Under **Target Network Settings**, set values such as the following.
   - **IP address**: `10.10.10.15`
   - **Port**: `22222`
   - **Subnet mask**: `255.255.255.0`
   - **Gateway**: `10.10.10.10`
7. Under **Ethernet Device Settings**, set the following values:
   - **Target driver**: `Auto`
   - **Bus type**: **PCI**

Note **Address** and **IRQ** are enabled for **Bus type ISA** only.

---

8 Click in the **Target Properties** workspace, then click the Save icon  on the toolbar, or click **File > Save**.

The dialog box looks like this:



Continue with "Configure Target Settings" on page 4-7.

Configure Target Settings

To run an xPC Target model on a target machine, you must configure the target settings.

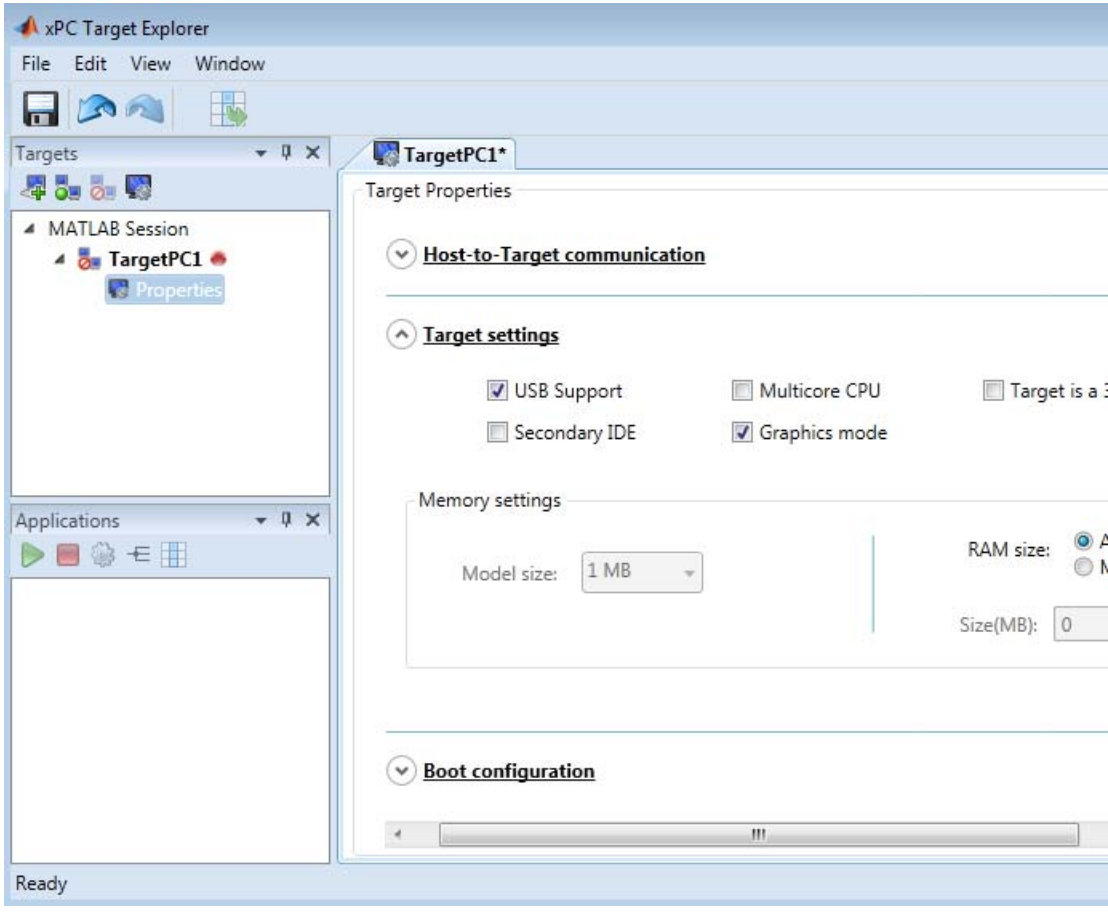
1. Type `xpcexplr` in the MATLAB Command Window.
2. In the **Targets** pane, expand the target computer node.
3. Click the Properties icon  on the toolbar or double-click **Properties**.
4. In the **Target Properties** workspace, click **Target settings**.
5. Select **USB Support** and **Graphics mode**.
6. Clear **Secondary EDE**, **Multicore CPU**, and **Target is a 386/486**.
7. Under **Memory settings**, select **Ram size Auto**.

Note Settings **Model size** and **Size(MB)** are enabled for **RAM size Manual** only.

---

8. Click in the **Target Properties** workspace, then click the Save icon  on the toolbar, or click **File > Save**.

The dialog box looks like this:



Continue with "Configure Boot Configuration" on page 4-9.

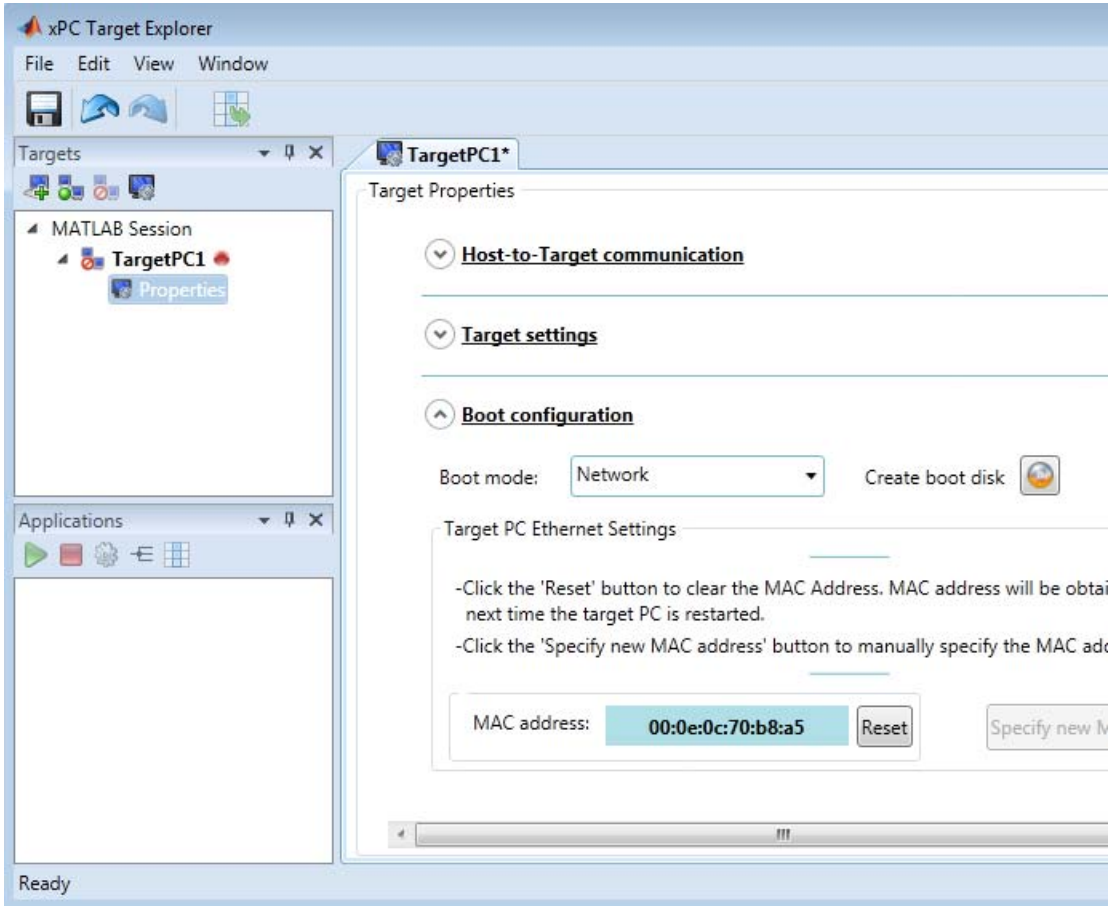
Configure Boot Configuration

To run an xPC Target model on a target machine, you must configure the target boot method.

Note For more on boot methods, see "Target Boot Methods" on page 2-41.

1. Type `xpcexplr` in the MATLAB Command Window.
2. In the **Targets** pane, expand the target computer node.
3. Click the Properties icon on the toolbar or double-click **Properties**.
4. In the **Target Properties** workspace, click **Boot Configuration**.
5. Select **Boot mode** `Network`.
6. Click **Create boot disk**.
7. Click in the **Target Properties** workspace, then click the Save icon on the toolbar, or click **File > Save**.

The dialog box looks like this:



Continue with "Run the Confidence Test" on page 4-11.

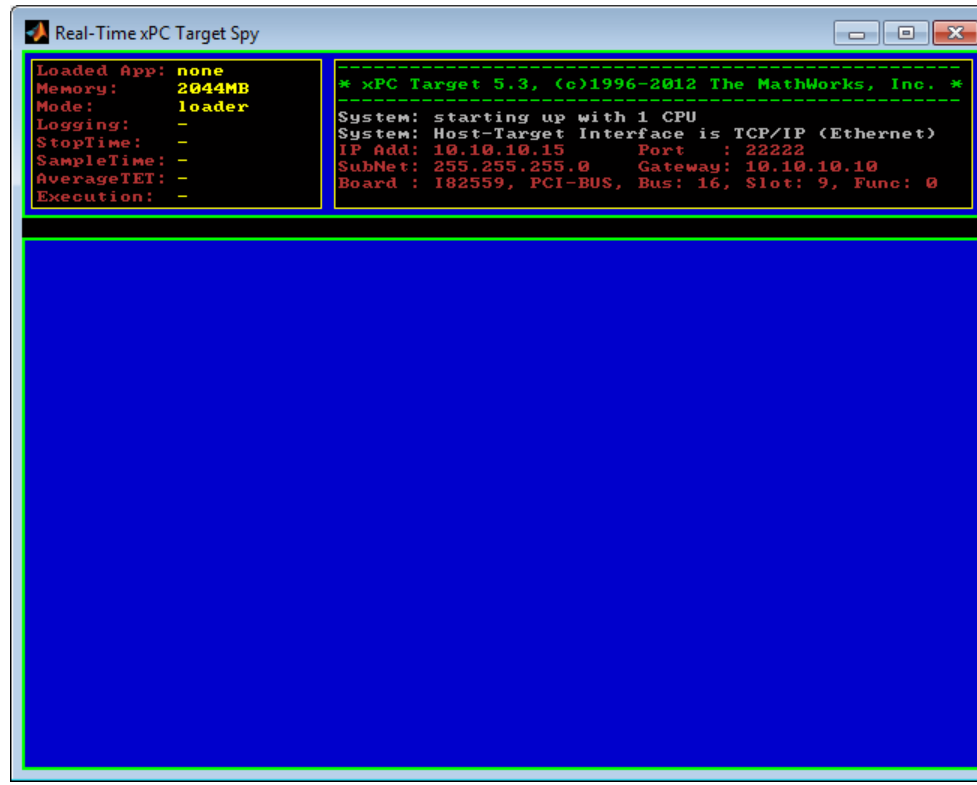
Run the Confidence Test

Validate the setup and configuration by running the confidence test:

1 If necessary, create a network boot image and reboot the computer according to the procedure in “Network Boot Procedure” on page 2-46.

Note This procedure assumes you are using TCP/IP for host-target communication and for booting the target computer. For more information, see “Network Communication Setup” on page 2-16 and “Network Boot Method” on page 2-45.

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



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

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

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

---

3 Type `xpctest` in the MATLAB Command Window.

MATLAB runs the test script for the default target computer and displays messages indicating whether the test passed or failed. If you use RS-232 communication, the first test is skipped.

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

4 Evaluate the results.

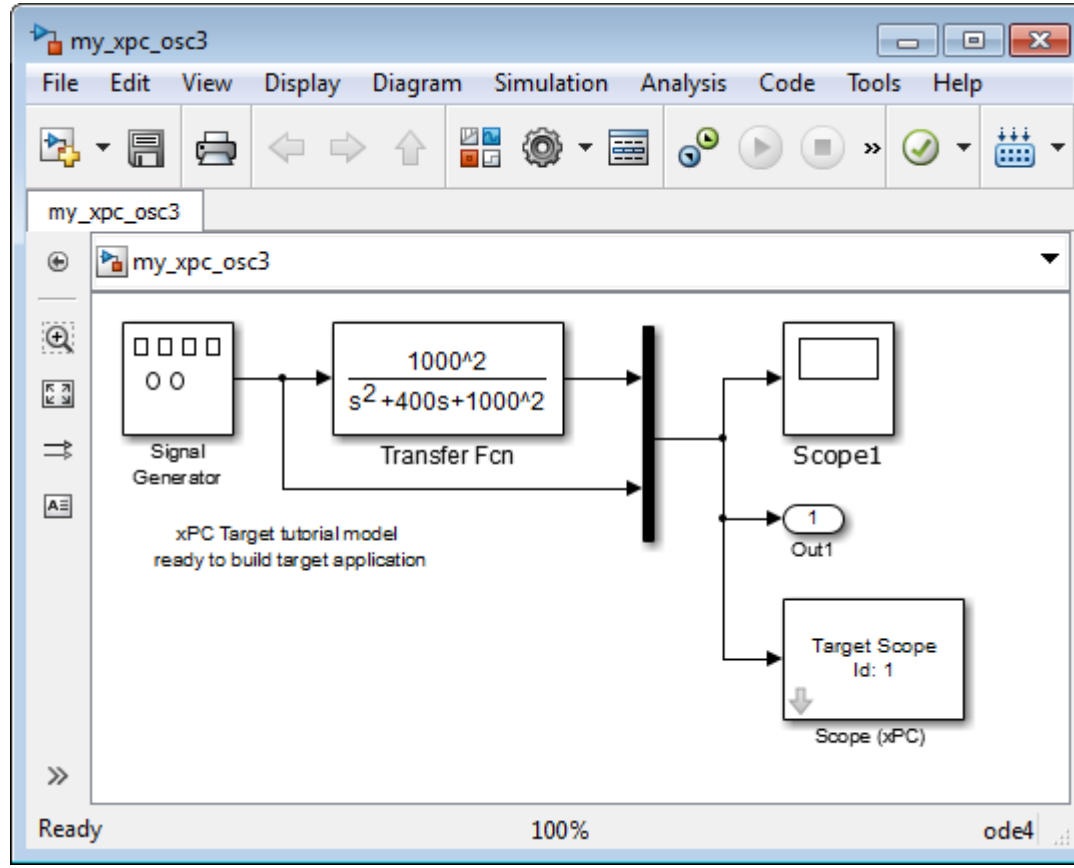
- If all of the tests return `OK`, you are ready to build and download a target application to the target computer.
- If any of the tests return `FAILED`, see "Troubleshooting in xPC Target".

Continue with "Create and Run a Real-Time Application" on page 4-14.

Create and Run a Real-Time Application

Simulink model `my_xpc_osc1` is a non-real-time model of a damped oscillator. Starting from this, the tutorial explains how to transform `my_xpc_osc1` into a xPC Target model configured to build as a target application.

The final model, `my_xpc_osc3`, looks like this:



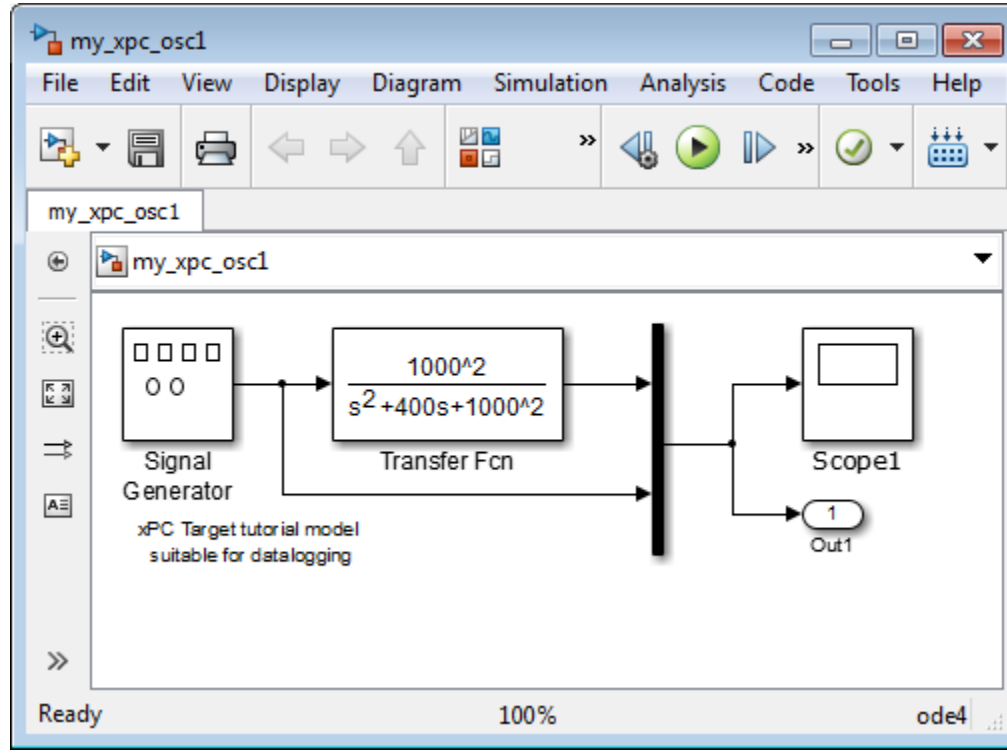
The tutorial shows how to use the xPC Target software, the Simulink Coder environment, and a third-party compiler to create a target application from `my_xpc_osc3`, build and download it, and execute it on the target computer.

Continue with "Interact with a Real-Time Application" on page 4-53.

Create and Run Simulink Model

The model includes a Transfer Fcn block and a Signal Generator block. To visualize and log signals while simulating, it also includes standard Simulink Scope and `Out1` (Outport) blocks.

The final model, `my_xpc_osc1`, looks like this:



5 "Configure Simulation Parameters" on page 4-28

6 "Simulate Using Simulink" on page 4-32

Continue with "Transform Simulink Model to Target Application" on page 4-34.

Create Simulink Model

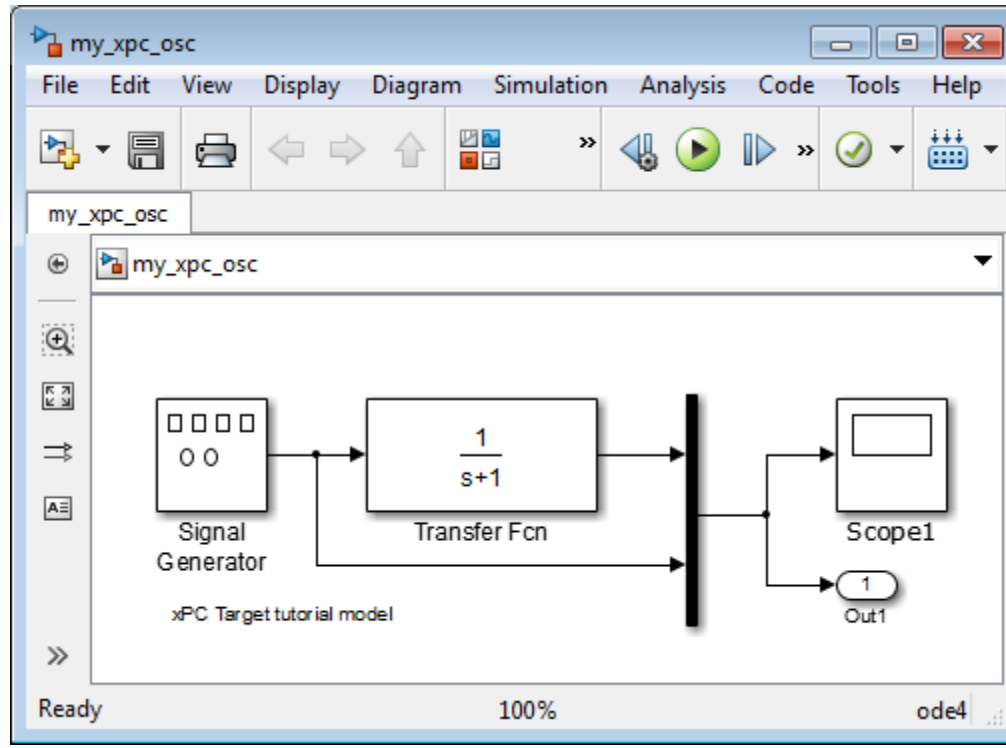
1. In the MATLAB Command Window, type `simulink`.
2. In the Simulink library window, from the **File** menu, point to **New**, and then click **Model**.
3. In the left pane, double-click **Simulink**, and then click **Continuous**.
4. Click and drag the Transfer Fcn block to the Simulink model window.
5. Click **Sources**, and add a Signal Generator block.
6. Click **Sinks**, and add a Scope block and an Out1 (Outport) block.
7. Click **Signal Routing**, and add a Mux block.
8. Make the following block-to-block connections:
   - Signal Generator output to Transfer Fcn input.
   - Signal Generator output to Mux input.
   - Transfer Fcn output to Mux input.
   - Mux output to Scope input.
   - Mux output to Out1 input.
9. As required, double-click signals and enter names.

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

---

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

    The model, `my_xpc_osc`, looks like this (unconfigured):

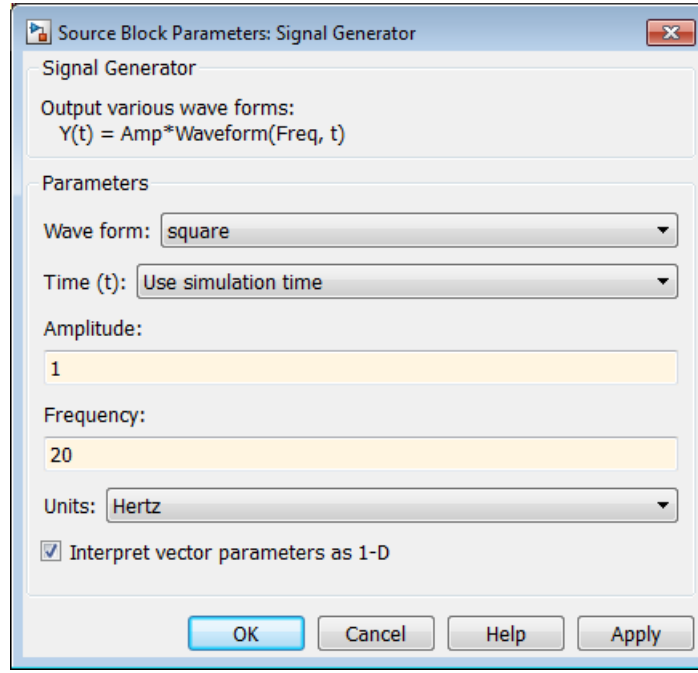


Continue with "Configure Signal Generator" on page 4-20.

Configure Signal Generator

1. In the MATLAB Command Window, type `my_xpc_osc`.
2. In the Simulink window, double-click the Signal Generator block.
3. In the Block Parameters dialog box, from the **Wave form** list, select `square`.
4. In the **Amplitude** text box, enter `1`.
5. In the **Frequency** text box, enter `20`.
6. From the **Units** list, select `Hertz`.

   The Block Parameters dialog box looks like this:



7. Do one of the following:

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

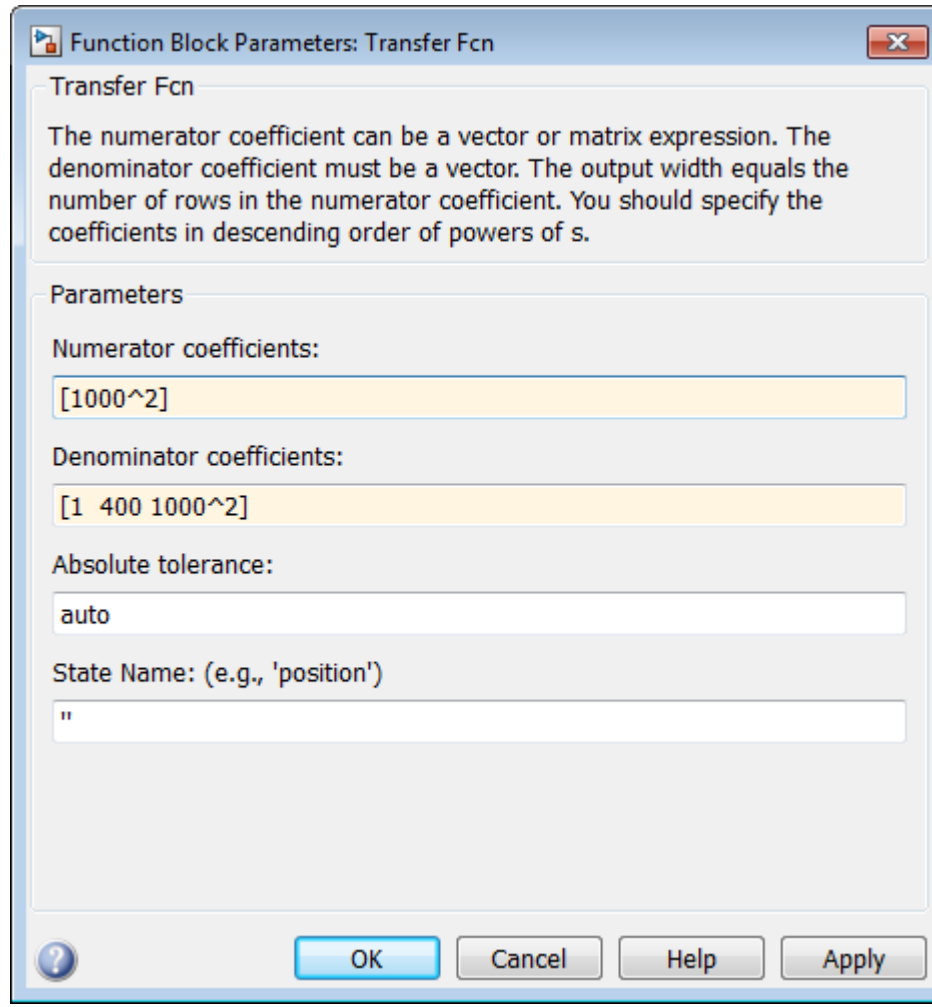
8 From the **File** menu, click **Save As** and enter a filename. For example, enter `my_xpc_osc1` and then click **OK**.

Continue with "Configure Transfer Function" on page 4-22.

Configure Transfer Function

1. In the MATLAB Command Window, type `my_xpc_osc1`. MATLAB loads the oscillator model and displays the Simulink block diagram.
2. In the Simulink window, double-click the Transfer Fcn block. The Block Parameters dialog box opens.
3. In the **Numerator** text box, enter `[1000^2]`.
4. In the **Denominator** text box, enter`[1 400 1000^2]`.

   The Block Parameters dialog box looks like this:



5 Do one of the following:

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

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

Continue with "Configure Scope Block" on page 4-25.

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

1. In the MATLAB Command Window, type `my_xpc_osc1`. MATLAB loads the oscillator model and displays the Simulink block diagram.

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

3. Click the **Parameters** icon  on the toolbar. The Scope parameters dialog box opens.

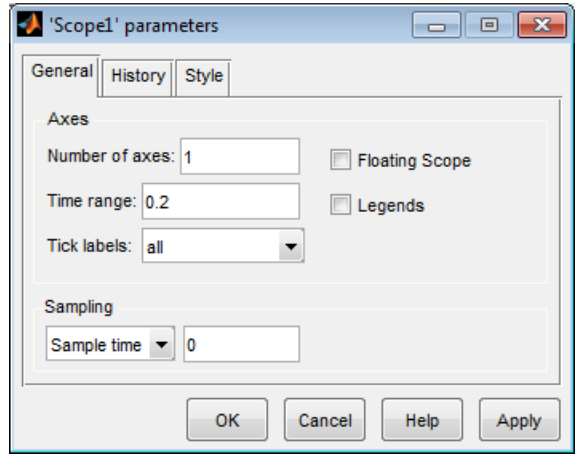
4. Click the **General** tab. In the **Number of axes** box, enter the number of graphs you want in one Scope window. For example, enter `1` for a single graph. Do not select the **floating scope** check box.

5. In the **Time range** box, enter the upper value for the time range. For example, enter `0.2` second.

6. From the **Tick labels** list, choose `all`.

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

   Your Scope parameters dialog box looks like this:



8 Do one of the following:

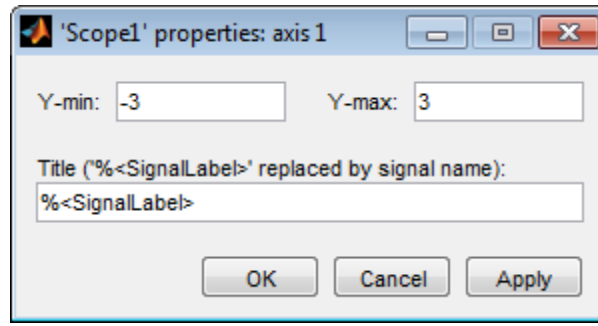
- Click **Apply** to apply the changes to the model and leave the dialog box open.
- Click **OK** to apply the changes to the model and close the dialog box.

9 In the Scope window, point to the *y*-axis and right-click.

10 From the pop-up menu, click **Axes Properties**. The Scope properties: axis 1 dialog box opens.

11 In the **Y-min** and **Y-max** text boxes, enter the range for the *y*-axis in the Scope window. For example, enter `-3` and `3`.

Your Scope axis dialog box looks like this:



12 Do one of the following:

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

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

Continue with "Configure Simulation Parameters" on page 4-28.

Configure Simulation Parameters

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

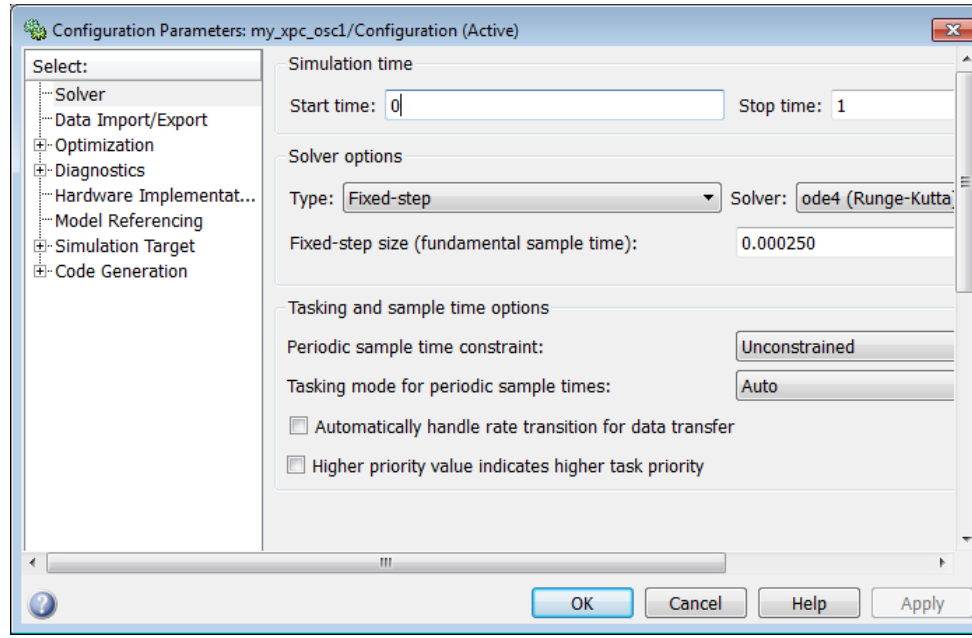
1. In the MATLAB Command Window, type `my_xpc_osc1`.
2. In the Simulink window, click **Simulation > Model Configuration Parameters**.
3. In the Configuration Parameters dialog box, click the **Solver** node.

   Simulink displays the **Solver** pane. The **Simulation section** of this pane defines the initial stop and sample time for the 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 `1` second. To change this time after creating the target application, change the target object property `tg.Stoptime` to the new time using the MATLAB command-line interface. To specify an infinite stop time, enter `inf`.
5. From the **Type** list, select `Fixed-step`. Simulink Coder does not support variable-step solvers.
6. From the **Solver** list, select a solver. For example, select the general-purpose solver `ode4 (Runge-Kutta)`.
7. In the **Fixed step size** box, enter the sample time for the target application. For example, enter `0.00025` second (`250` microseconds). You can change this value after creating the target application.

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

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

The **Solver** pane looks like this:



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

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

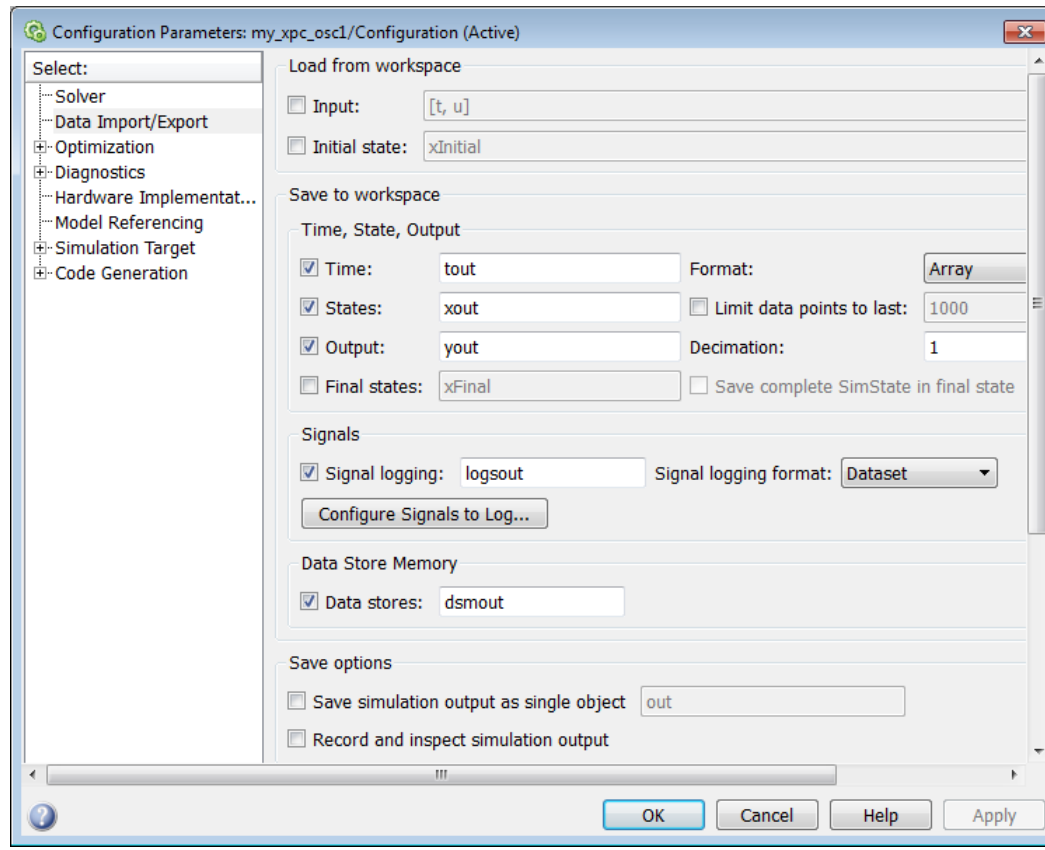
9 In the **Save to workspace** section of this pane, select the **Time**, **States**, and **Output** check boxes.

When the target application is running in non-real time, data is saved to the variables `tout` and `yout`. When the target application is running in

real time, data is saved to the target object properties `TimeLog`, `StateLog`, and `OutLog`instead of to `tout` and `yout`. However, you must still select the **Time**, **States**, and **Output** check boxes for data to be logged to the target object properties.

Note For circumstances in which you might turn one or more **Save to workspace** check boxes off, see "How Can I Improve Run-Time Performance?".

The **Data Import/Export** pane looks like this:



10 Do one of the following:

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

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

Continue with "Simulate Using Simulink" on page 4-32.

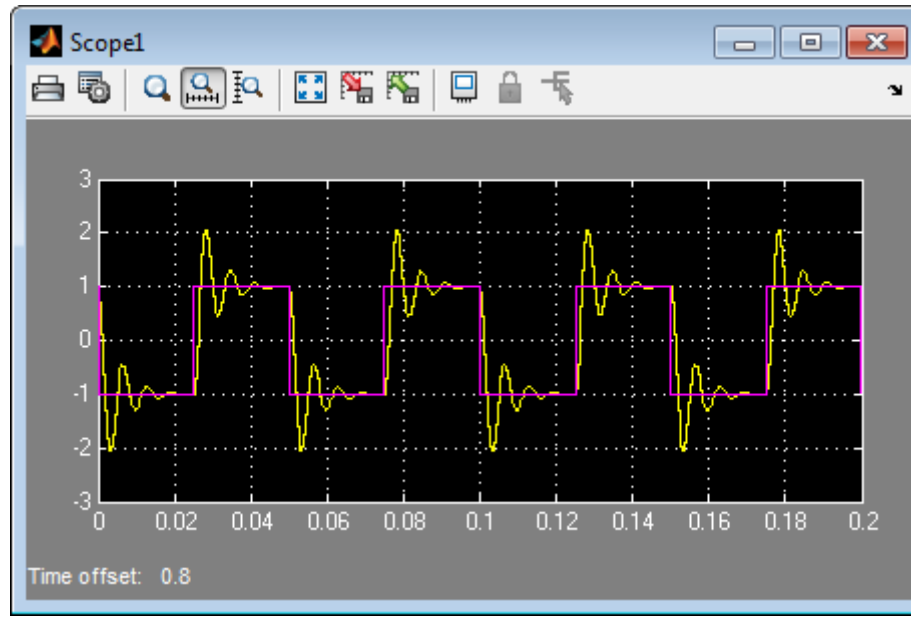
Simulate Using Simulink

You use Simulink in normal mode to observe the behavior of the model in non-real time. After you load the Simulink model, you can run a simulation.

1. In the MATLAB Command Window, type `my_xpc_osc1`. MATLAB loads the oscillator model and displays the Simulink block diagram.
2. In the Simulink window, double-click the Scope block. Simulink opens a scope window.
3. Select **Simulation > Mode > Normal**.
4. Choose **Run** to begin simulation or click the Run icon  on the toolbar.

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

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



5 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**.

Continue with "Transform Simulink Model to Target Application" on page 4-34.

Transform Simulink Model to Target Application

To run a Simulink model as a target application under xPC Target, you need to add and configure xPC Target Scope block and set Configuration Parameters for code generation and target execution.

1. "Add xPC Target Scope Block" on page 4-35
2. "Set Target Scope Block Parameters" on page 4-37
3. "Set Configuration Parameters" on page 4-41

Continue with "Boot Target Hardware" on page 4-46.

Add xPC Target Scope Block

xPC Target supports a special xPC Target There are three types of scopes, `target`, `host`, and `file`. For this tutorial, we are adding a `target` Scope block, which displays execution data on the target computer monitor.

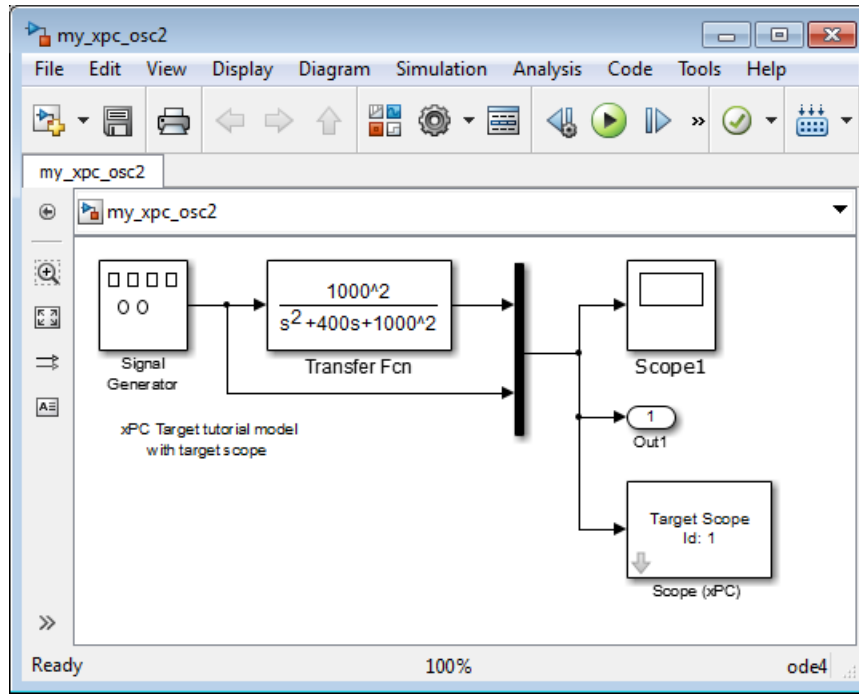
1. In the MATLAB window, type `my_xpc_osc1`. The Simulink block diagram opens for the model `my_xpc_osc1`.
2. In the Simulink window, from the **View** menu, click **Library Browser**.

   The Simulink Library Browser window opens.
3. In the left pane, browse to and double-click node **xPC Target**.

   A list of I/O block categories opens.
4. Click node **Misc**.
5. Click and drag block **Scope (xPC)** to the Simulink block diagram.

   Simulink adds a new Scope (xPC) block to the model with a scope identifier of `1`.
6. Connect the Mux output to the Scope (xPC) input.

   The model `my_xpc_osc1` looks like this:



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

Continue with "Set Target Scope Block Parameters" on page 4-37.

Set Target Scope Block Parameters

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

Note For more information about Scope (xPC) parameters, see "Trace Signals with Host Scope (xPC) Blocks".

---

1. In the MATLAB window, type `my_xpc_osc2`. The Simulink block diagram opens for the model `my_xpc_osc2`.
2. Double-click block **Scope (xPC)**.

   The block parameters dialog box opens.
3. Select **Scope type** `Target`. This value means that the scope display appears on the target computer monitor.
4. Select check box **Start scope when application starts**.

   Note This setting is a convenience for most boot modes. It is mandatory in Stand Alone mode because the host computer is not available to issue a command to start scopes.

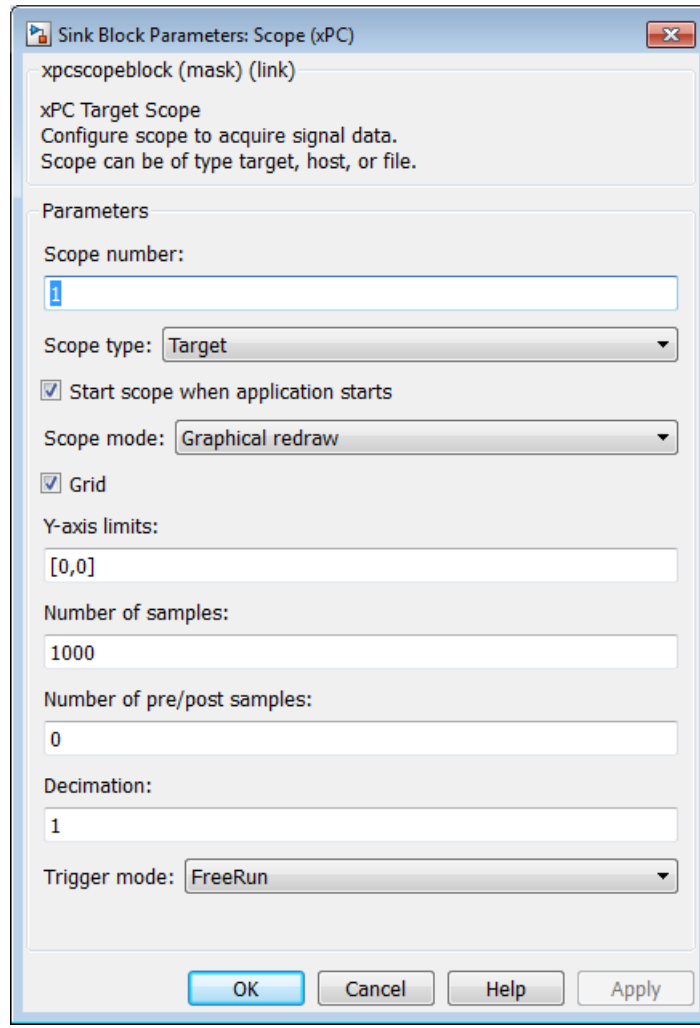
   ---
5. Select **Scope mode** `Graphical redraw`.
6. Select check box **Grid**.
7. Type `[0,0]` in text box **Y-axis limits**. This value means that display scaling is `auto`.
8. Type `1000` in text box **Number of samples**. For a **Scope mode** of `Graphical redraw`, this value means that 1000 samples are acquired before the graph is redrawn.

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

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

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

The target scope dialog box looks like this:



12 Do one of the following:

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

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

Continue with "Set Configuration Parameters" on page 4-41.

Set Configuration Parameters

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

After you open a Simulink model and boot the target computer, you can enter the simulation parameters. This procedure uses the Simulink model `my_xpc_osc2` as an example and assumes you have already opened that model (see "Create and Run Simulink Model" on page 4-16).

1. In the MATLAB window, type

   ```
   my_xpc_osc2
   ```

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

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

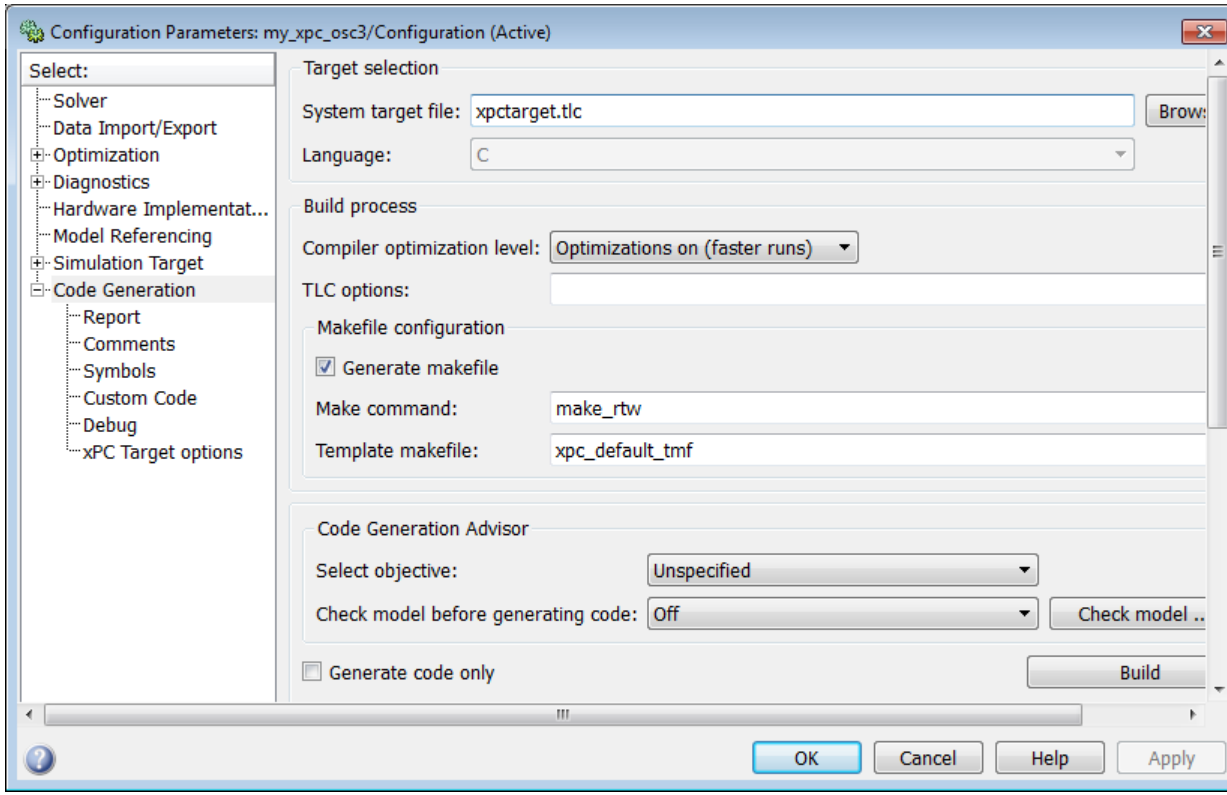
   The Configuration Parameters dialog box is displayed for the model.

3. Click the **Code Generation** node.

   The code generation pane opens.

4. To build a basic target application, in the **Target selection** section, click the **Browse** button at the **System target file** list. Click `xpctarget.tlc`, and then click **OK**.

The system target file `xpctarget.tlc`, the template makefile `xpc_default_tmf`, and the make command `make_rtw` are automatically entered into the page. The **xPC Target options** node appears in the left pane. The code generation pane looks like this:



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

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

---

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

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

6 From the **Execution mode** list, select either `Real-Time` or `Freerun`. The option `Freerun` is similar to a simulation, but with the generated code. It runs the target application as fast as it can. However, unlike a simulation, the `Freerun` mode of the xPC Target software does not support variable-step solvers.

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

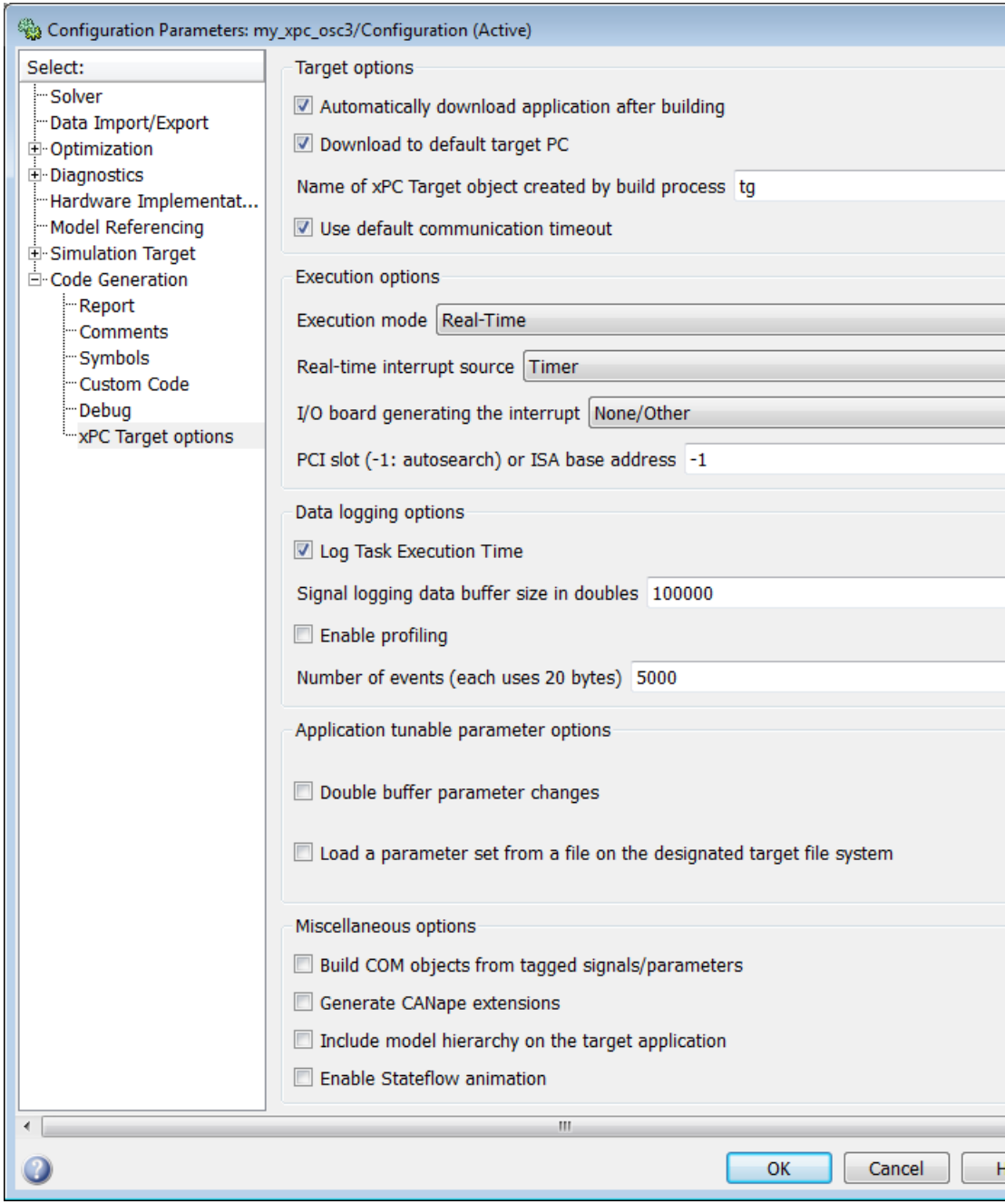
8 Select the **Log Task Execution Time** check box to log task execution times to the target object property `tg.TETlog`.

The task execution time is the time in seconds to complete calculations for the model equations and post outputs during each sample interval. If you do not select this box, the 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 **xPC Target Options** pane should now look like the figure shown.



11 Click **OK**.

12 From the **File** menu, click **Save as**. Enter a filename. For example, enter `my_xpc_osc3` and then click **Save**.

Continue with "Boot Target Hardware" on page 4-46.

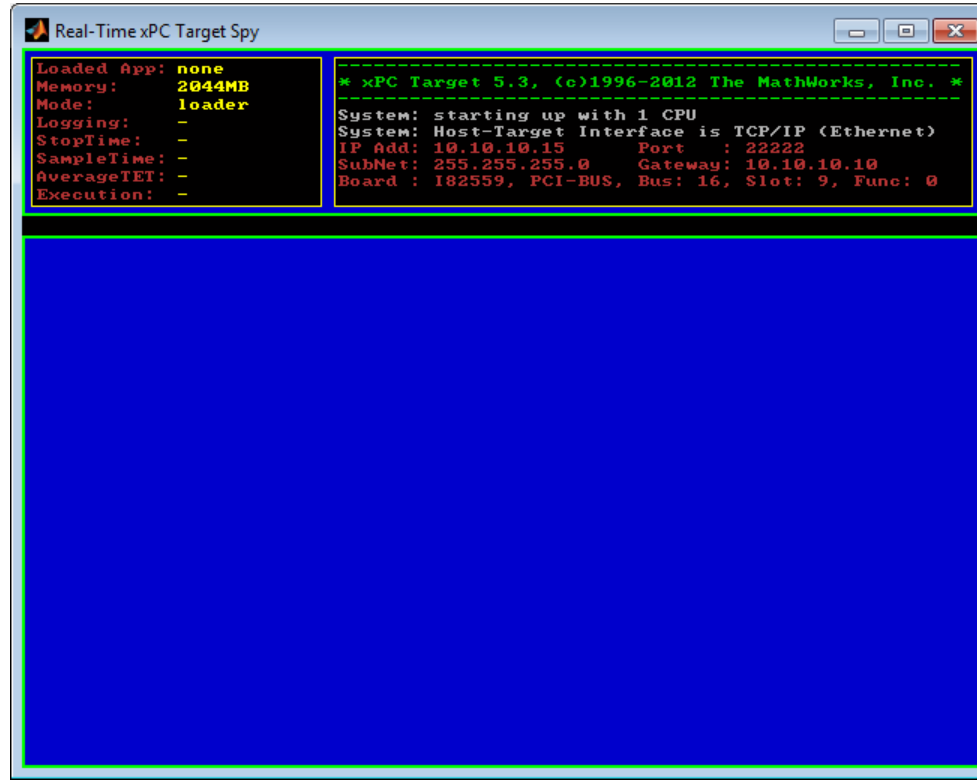
Boot Target Hardware

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

After you have configured the xPC Target product using the xPC Target Explorer and created a target boot disk for that setup, you can boot the target computer. You need to boot the target computer before building the target application because the build process automatically downloads the target application to the target computer. Be sure that you have followed the instructions from “Host-Target Configuration” before continuing.

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

   The target computer displays a screen like this:



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

Note The kernel can use only 2 GB of memory. Of the available memory, the kernel uses 3 MB for itself and reserves 1 MB for the target application.

Continue with "Build and Download Target Application" on page 4-48.

Build and Download Target Application

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

After you enter changes in the Configuration Parameters dialog box, you can build the target application. This procedure uses the Simulink model `my_xpc_osc3` as an example. To create this model , see “Create and Run Simulink Model” on page 4-16. By default, the build procedure downloads the target application to the default target computer, as designated in xPC Target Explorer. See “xPC Target Options Configuration Parameter” for further details on setting the target computer for a target application.

1. In the MATLAB window, type `my_xpc_osc3`.

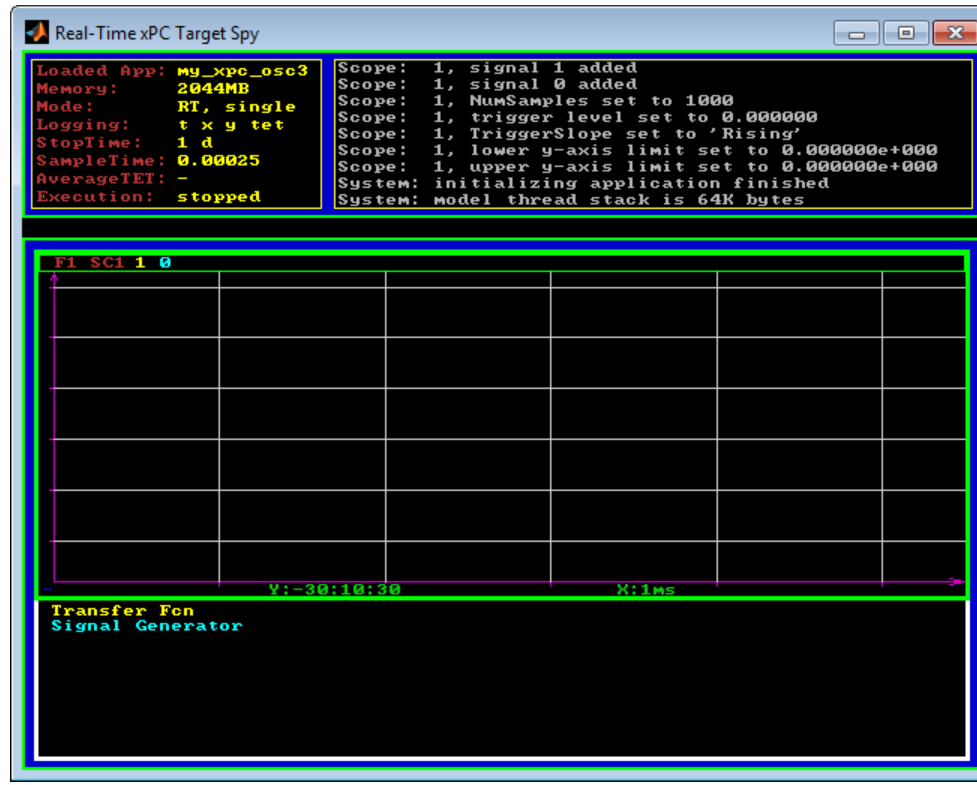
2. In the Simulink window, click the Build Model icon  on the toolbar or click **Code > C/C++ Code > Build Model**.

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

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

   After compiling, linking, and downloading, xPC Target creates a target object in the MATLAB workspace. The default name of the target object is `tg`. For more information about the target object, see “Target Driver Objects”.

   If you have a monitor connected to your target computer, the monitor screen looks like this.



3 In the MATLAB window, type

```
tg
```

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

If the software detects a error during build and download, see "Troubleshooting in xPC Target".

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

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

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

During the build process, the xPC Target software creates a target object that represents the target application running on the target computer. The target object is defined by a set of properties and associated methods. You control the target application and computer by setting the target object properties.

Continue with "Execute Target Application Using Simulink External Mode" on page 4-51.

Execute Target Application Using Simulink External Mode

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

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

After you create and download a target application to the target computer, you can run the target application. This procedure uses the Simulink model `my_xpc_osc2` as an example (see "Build and Download Target Application" on page 4-48). It assumes that you have specified the required target computer environment on the xPC Target options node of the Simulink Coder parameters dialog. In particular, you must specify the target computer to which you want to connect. See the **Use the default target PC** check box description in "xPC Target Options Configuration Parameter".

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

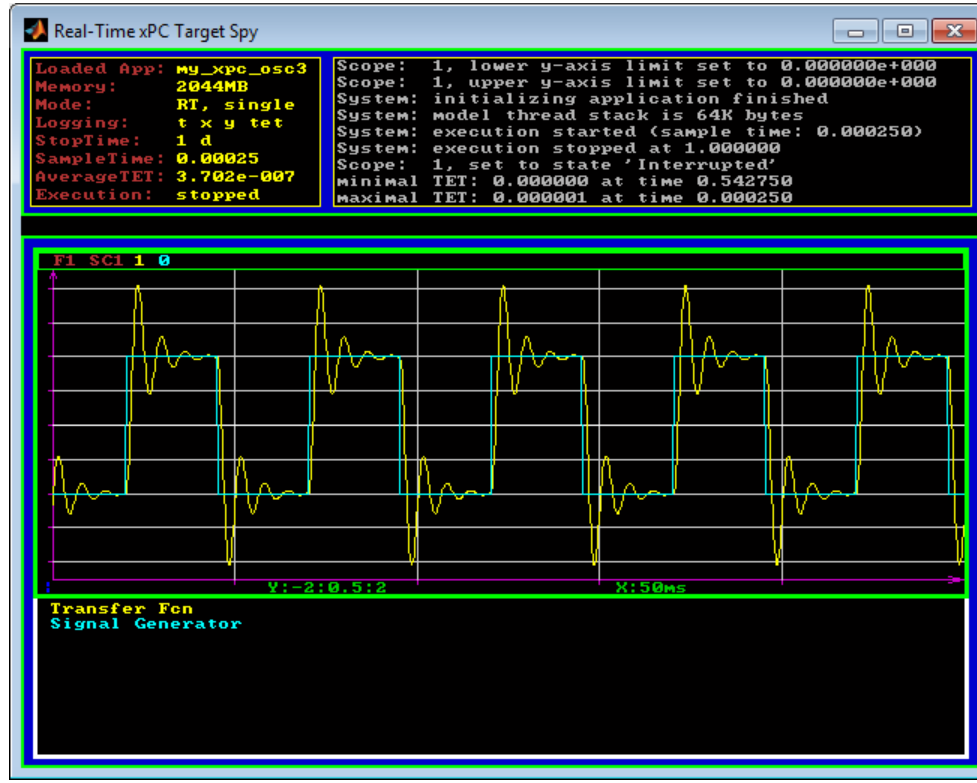
   A check mark appears next to the menu item **External**, and Simulink external mode is activated. Simulink external mode connects the Simulink model to the target application as a simple graphical user interface.

2. Click **Simulation > Connect To Target** or click the Connect To Target icon  on the toolbar.

   All the current Simulink model parameters are downloaded from the host computer to the target application.

3. Click **Simulation > Run** or click the Run icon  on the toolbar.

   The target application begins running. If you have a monitor connected to your target computer, you can view output that looks like this:



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.

Interact with a Real-Time Application

You can interact with a real-time application using the xPC Target Explorer as well as Simulink External Mode. This tutorial will focus on interaction using xPC Target Explorer.

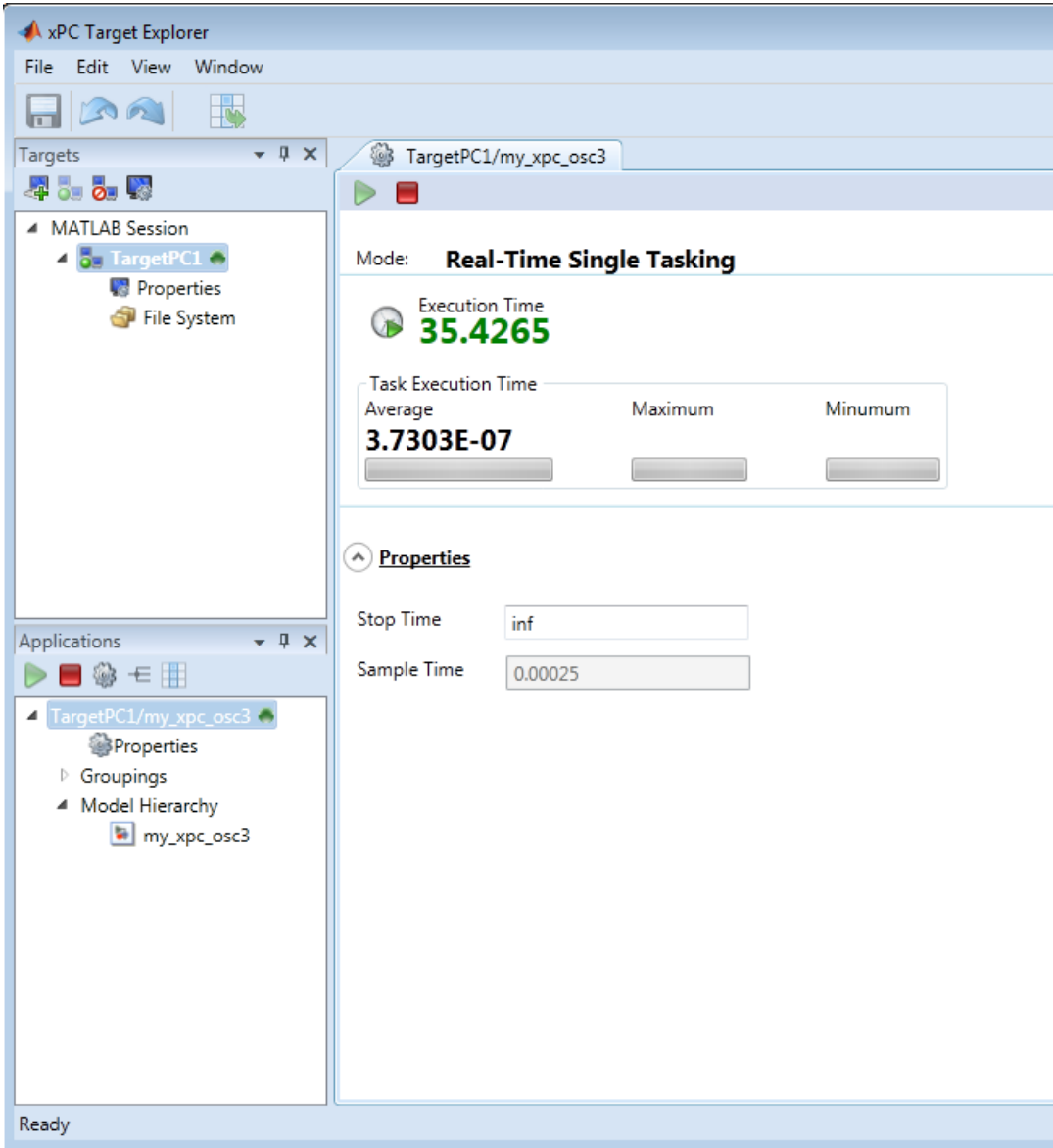
Execute Target Application Using xPC Target Explorer

This procedure assumes you have created an xPC Target boot disk and you have booted the target computer. See "Boot Target Hardware" on page 4-46. This procedure begins with a target application already downloaded to the target computer (see "Build and Download Target Application" on page 4-48).

1. In the MATLAB window, type `xpcexplr`.
2. In the **Targets** pane, right-click the target computer icon for which you have downloaded the application and click **Connect** or click the Connect icon on the toolbar.

   The target computer icon turns Connected .
3. Select the target application in the **Applications** pane.
4. To start execution, click the target application and click the Start icon on the toolbar, or right-click the target application and click **Start**.

   The application starts running. The dialog box looks like this:


xPC Target Explorer
File
Edit
View
Window
Targets
MATLAB Session
TargetPC1
Properties
File System
Applications
TargetPC1/my_xpc_osc3
Properties
Groupings
Model Hierarchy
my_xpc_osc3
TargetPC1/my_xpc_osc3
Mode:
Real-Time Single Tasking
Execution Time
35.4265
Task Execution Time
Average
Maximum
Minumum
3.7303E-07
Properties
Stop Time
inf
Sample Time
0.00025
Ready

Tip You can view the output of the Scope (xPC) block by connecting a monitor to your target computer.

5 To stop execution, click the target application and click the Stop icon on the toolbar, or right-click the target application and click **Stop**.

The target application on the target computer stops running, and the target computer displays messages like this:

```
minimal TET: 0.0000006 at time 0.001250
maximal TET: 0.0000013 at time 75.405500
```

See also "Trace Signals with Target Scopes Using xPC Target Explorer" and "Log Signals Using Outport with xPC Target Explorer".

Create a Prebuilt Target Application

To build a target application without downloading it:

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

2 Select the **xPC Target Options** node under **Code Generation**. menu.

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

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

Download a Prebuilt Target Application

To download a prebuilt application from xPC Target Explorer:

1 In the MATLAB window, type `xpcexplr`.

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

The target computer icon turns Connected .

3 Do one of the following:

- Right-click the required target computer icon, click **Load** then in the **DLM Application Selector** dialog box, enter or browse to the prebuilt application DLM file.
- From the MATLAB window or from Microsoft Windows, drag the prebuilt application DLM file to the target computer icon.

Create an Application Showing Model Hierarchy

To create a target application that shows model hierarchy in xPC Target Explorer:

1 Open the Simulink window.

2 Click **Simulation > Model Configuration Parameters**.

3 Select the **xPC Target Options** node under the **Code Generation** node.

4 Set the check box **Include model hierarchy on the target application**.

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

Change Stop Time and Sample Time

This topic describes how to change the stop time and sample time in the target application configuration. It assumes that you have already downloaded the target application to a target computer.

1. In xPC Target Explorer, expand the node of the loaded target application in the **Applications** pane.
2. Click the Properties icon on the toolbar or double-click **Properties**.
3. In the **Application Configuration** workspace, under Execution Parameters, enter a new value for **Stop time**. For example, enter `inf` and type carriage return.

   The target application now runs until you stop it.
4. Enter a new value for **Sample Time**. For example, enter `0.00010` and type carriage return.

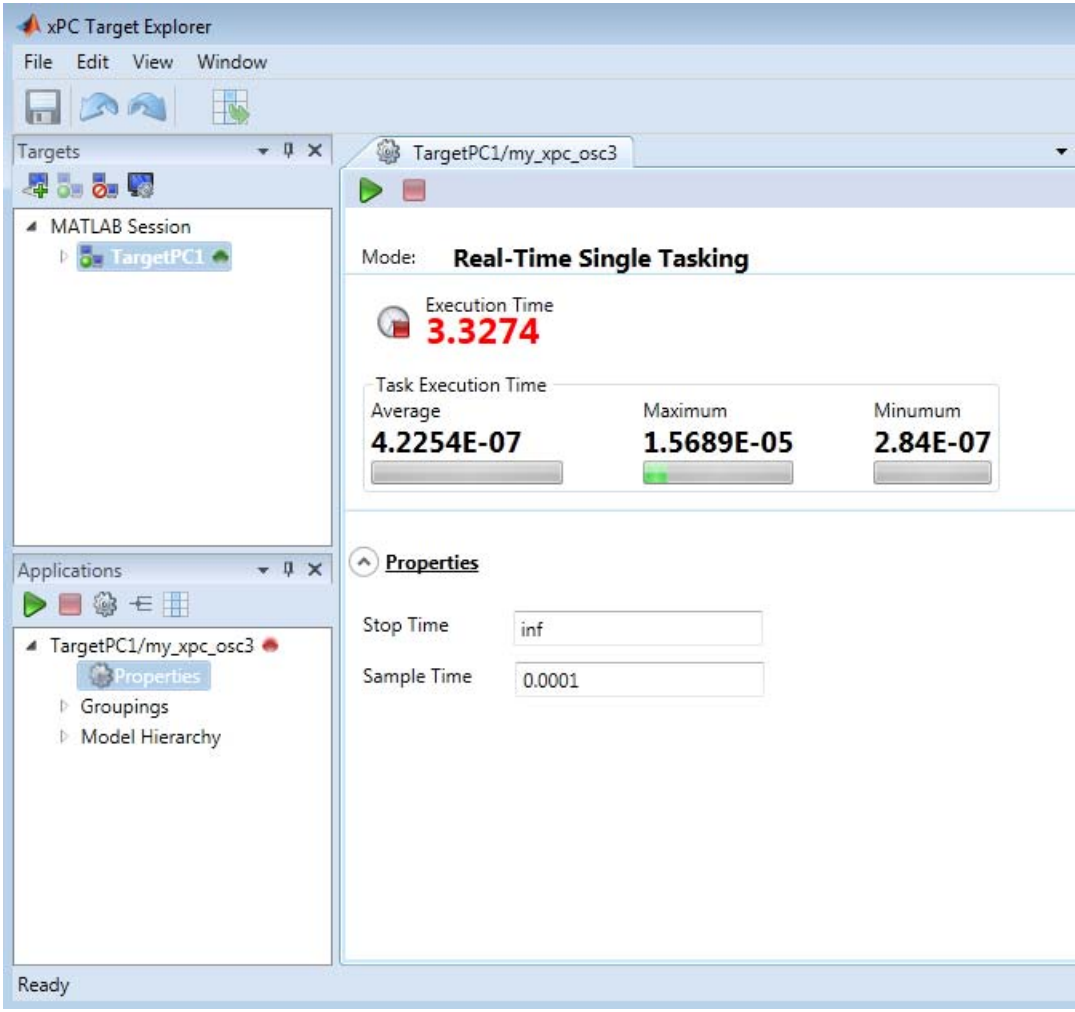
Tip See "User Interaction" on page 1-25 for limitations on changing sample times.

---

5. To start execution, click the target application and click the Start icon on the toolbar, or right-click the target application and click **Start**.

   The application starts running.
6. To stop execution, click the target application and click the Stop icon on the toolbar, or right-click the target application and click **Stop**.

   The dialog box looks like this:


xPC Target Explorer
File Edit View Window
Targets
MATLAB Session
TargetPC1
Applications
TargetPC1/my_xpc_osc3
Properties
Groupings
Model Hierarchy
TargetPC1/my_xpc_osc3
Mode: Real-Time Single Tasking
Execution Time
3.3274
Task Execution Time
Average
4.2254E-07
Maximum
1.5689E-05
Minumum
2.84E-07
Properties
Stop Time
inf
Sample Time
0.0001
Ready

Simulate Simulink Model Using MATLAB Language

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

After you load the Simulink model into the MATLAB workspace, you can run a simulation. This procedure uses the Simulink model `my_xpc_osc2` as an example and assumes you have already loaded that model. To create this model , see "Create and Run Simulink Model" on page 4-16.

1. In the MATLAB window, type

   ```
   output=sim('my_xpc_osc2','SimulationMode','normal');
   ```

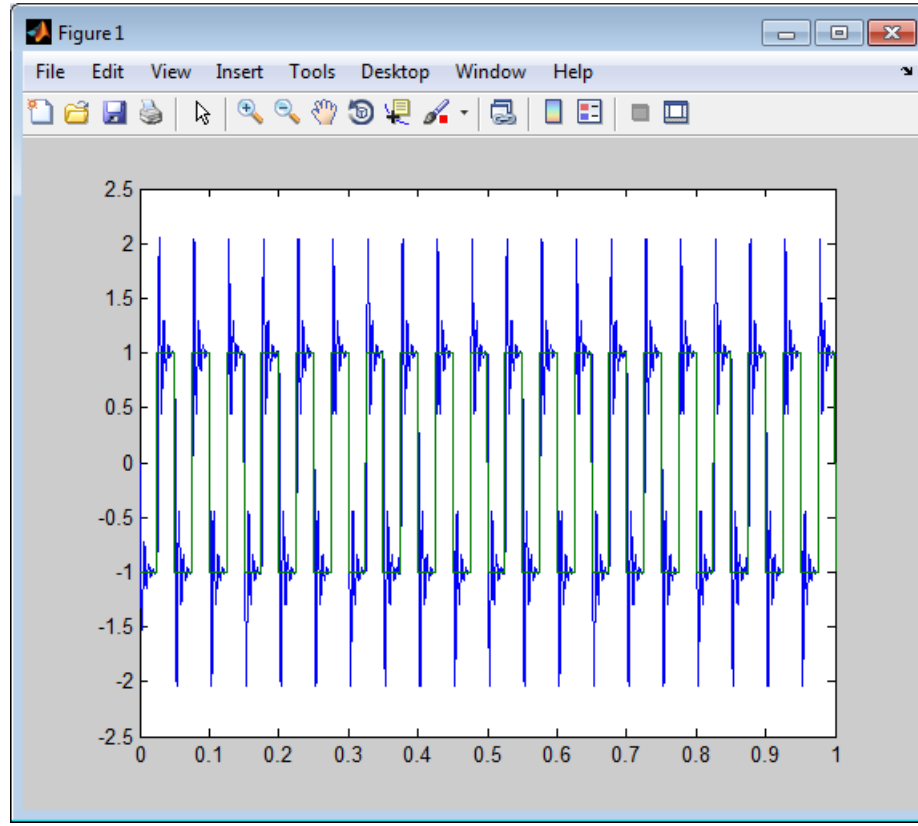
   Simulink runs a simulation in normal mode through to completion. You cannot manually stop the simulation. See the online Simulink documentation for further information on using the `sim` command.

2. After Simulink finishes the simulation, type

   ```
   plot(output.get('tout'), output.get('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 configure an Outport block, see "Configure Simulation Parameters" on page 4-28.

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

Execute Target Application Using MATLAB Language

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

After the xPC Target software downloads the target application to the target computer, you can run the target application. This procedure uses the Simulink model `my_xpc_osc3` as an example, and assumes you have created and downloaded the target application for that model. It also assumes that you have assigned `tg` to the target computer.

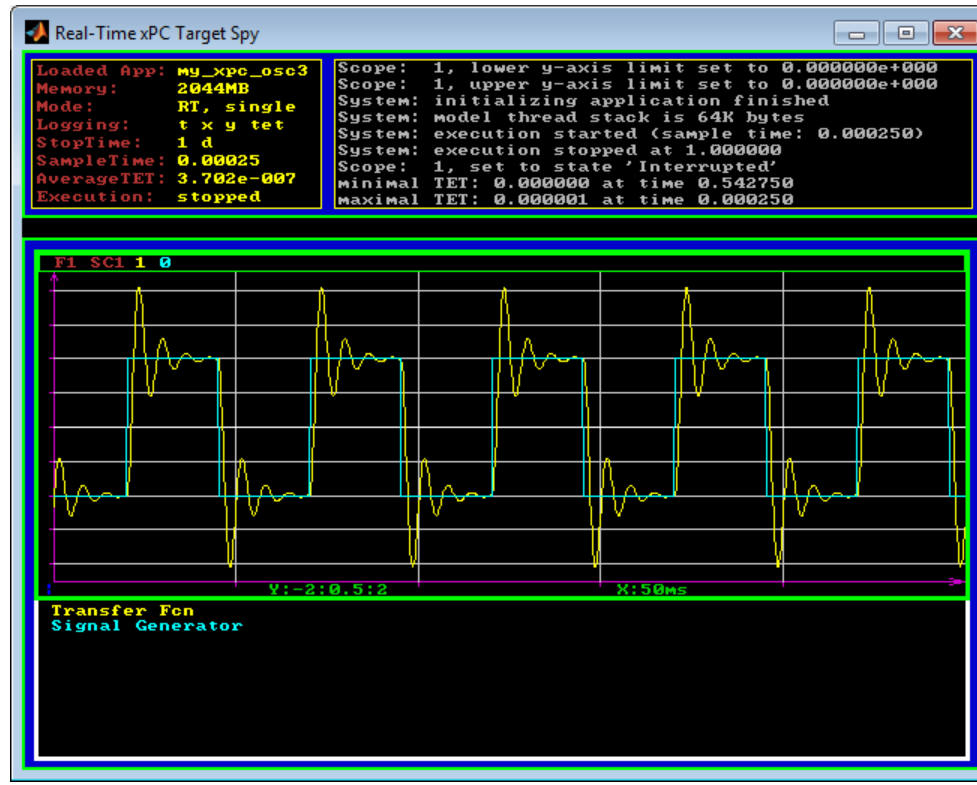
1 In the MATLAB window, type any of

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

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

```
xPC Object
  Connected    = Yes
  Application    = my_xpc_osc3
  Mode    = Real-Time Single-Tasking
  Status    = running
```

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



2 In the MATLAB window, type

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

The target application stops running.

The xPC Target software allows you to change many properties and parameters without rebuilding the 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 1-25 for further limitations on changing sample times.)

Application and Driver Scripts

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

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

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

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

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

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

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

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

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

Edit Scripts

To locate and edit the example Signal Tracing With a Host Scope in Freerun Mode:

1. Follow the link to the example and determine the example name, in this case `scfreerundemo`.
2. In the MATLAB Command Window, type

   ```
   which scfreerundemo
   ```

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

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

3. Type

   ```
   edit scfreerundemo
   ```

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

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

B

C

D

E

F

G

H

I

K

L

M

N

O

P

Q

R

S

T

U

V

W

X