



Intel[®] IXA SDK for the IXP1200 Network Processor Family

IXA SDK 2.01 Installation and Setup Guide

December 2001



Revision History

Revision Date	Revision	Description
August 2001	2.0	IXA SDK 2.0 release.
December 2001	2.01	IXA SDK 2.01 release. Add support for Red Hat* 7.2.

Information in this document is provided in connection with Intel® products. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted by this document. Except as provided in Intel's Terms and Conditions of Sale for such products, Intel assumes no liability whatsoever, and Intel disclaims any express or implied warranty, relating to sale and/or use of Intel products including liability or warranties relating to fitness for a particular purpose, merchantability, or infringement of any patent, copyright or other intellectual property right. Intel products are not intended for use in medical, life saving, or life sustaining applications.

Intel may make changes to specifications and product descriptions at any time, without notice.

Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

The product may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Intel Corporation might have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give any license to these patents.

This document as well as the hardware and software described in it are furnished under license and may only be used or copied in accordance with the terms of the license. The information in this document is furnished for informational use only, is subject to change without notice, and should not be construed as a commitment by Intel Corporation. Intel Corporation assumes no responsibility or liability for any errors or inaccuracies that may appear in this document or any software that may be provided in association with this document. Except as permitted by such license, no part of this document may be reproduced, stored in a retrieval system, or transmitted in any form or by any means without the express written consent of Intel Corporation.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an ordering number and are referenced in this document, or other Intel literature may be obtained by calling 1-800-548-4725 or by visiting Intel's website at <http://www.intel.com>

Copyright © Intel Corporation, 2001. All rights reserved.

*Other names and brands may be claimed as the property of others.

This product includes software developed by parties other than Intel.



Contents

	About This Guide	v
	Audience	v
	In This Guide	v
	Other Sources of Information	vi
	Typographic Conventions	vii
	Installation Path	vii
	Contacting Intel	vii
	Web and Internet Sites	vii
	Customer Support Technicians	viii
Chapter 1	Before Installing the IXA SDK	1
	The IXA Development Software	1
	The IXA SDK	2
	The IXP1200 Microengine Development Environment	2
	Software Requirements	3
	Hardware and Software Configurations	3
	Software Configuration	3
	Hardware and Cabling Configurations	4
	Ethernet Connections in All Configurations	6
	System Requirements for Using This Release	6
Chapter 2	Installing and Configuring the IXA SDK for a Windows NT Development Environment (Configurations 1 and 2)	7
	Removing the Volume I Software	8
	Upgrading from IXA SDK 2.0	8
	Installing the IXP1200 Microengine Development Environment	8
	Installing the IXA SDK Development Environment	11
	Compiling the System and Library MicroACEs	13
	Installing the Embedded Linux IDE for the Intel(r) IXP1200 Network Processor	13
	Setting Up a TFTP Server	14
	Setting Up an NFS Server	15
	Installing the NFS Server	15
	Configuring the NFS Server	15
	Installing Adobe Acrobat Reader	16
	Wiring Configuration	16
	Establishing a Remote Terminal	17
	Configuring the Boot Manager and Cygmon	18

Starting the System	19
Setting Up an NFS Client.	19
Configuring the IXP1200 Microengine Development Environment.	20
Removing the IXA SDK from Windows NT	21

Chapter 3 Installing and Configuring the IXA SDK for a Linux Development Environment (Configuration 3) 23

Installing the IXA SDK Microcode Development Environment	24
Removing the Volume I Software	24
Upgrading from IXA SDK 2.0	24
Installing the IXP1200 Microengine Development Environment	25
Installing Cygwin and the Microcode Libraries	27
Compiling the System and Library MicroACES	28
Installing Adobe Acrobat Reader.	28
Installing the Linux StrongARM Development Environment	29
Installing the Development Environment	29
Using StrongARM Compiler Options.	30
Setting Up a TFTP Server for a Linux Host	30
Setting Up an NFS Server	31
Wiring Configuration	31
Establishing a Remote Terminal.	32
Configuring the Boot Manager and Cygmon	33
Starting the System	34
Using a Minicom Script for Startup.	34
Setting Up an NFS Client.	35
Configuring the IXP1200 Microengine Development Environment.	36
Removing the IXA SDK	37
Removing the IXP1200 Microengine Development Environment	37
Removing the IXA SDK from Linux	37



About This Guide

This guide describes how to install the Intel® Internet Exchange Architecture (IXA) software development kit (IXA SDK).

Audience

This guide is intended for individuals who are installing the IXA SDK as a prerequisite to developing networking applications for the IXP1200 with embedded Linux.

In This Guide

This guide includes the following chapters:

- **Chapter 1, “Before Installing the IXA SDK,”** provides software and hardware prerequisites for installing and using the IXA SDK.
- **Chapter 2, “Installing and Configuring the IXA SDK for a Windows NT Development Environment (Configurations 1 and 2),”** describes how to install the IXA SDK and configure the development environment when using a Windows NT development workstation.
- **Chapter 3, “Installing and Configuring the IXA SDK for a Linux Development Environment (Configuration 3),”** describes how to install the IXA SDK and configure the development environment when using a Linux development workstation.



Other Sources of Information

This guide is part of the Intel IXA SDK documentation set, which also includes:

- *Getting Started*, which provides a high-level introduction to the product, which provides a high-level introduction to the product family and describes supported hardware and software configurations.
- *Intel IXA SDK ACE Programming Framework Reference (IXA SDK Reference)*, which describes the Intel IXA SDK (software development kit), including the IXA API, which consists of the OMS services library, the action services library (ASL), and the microACE Resource Manager; the Network Classification Language (NCL); the IXA IDL interface definition language; and development tools.
- *Intel IXA SDK ACE Programming Framework Developer's Guide (IXA SDK Developer's Guide)*, which provides programmers with conceptual descriptions and instructions on writing networking applications for the IXP1200 family of processors using the IXA SDK.
- *Intel IXA SDK ACE Programming Framework Tutorial (IXA SDK Tutorial)*, which provides a step-by-step introduction to building a basic IXA application.
- *Intel IXA SDK ACE Programming Framework (IXA SDK 2.01) Release Notes (IXA SDK Release Notes)*, which lists information about the latest software release.
- The IXP1200 Microengine Development Environment documentation set, which includes the following documents:
 - *IXP1200 Network Processor Family Microcode Programmer's Reference Manual*
 - *IXP1200 Network Processor Family Microcode Developer Tools User's Guide*
 - *IXP1200 Network Processor Family Microcode Example Software*
 - *IXP1200 Network Processor Family Hardware Reference Manual*
 - *IXP1200 Macro Library Reference Manual*
 - *IXP1200 Macro Library Style Guide*
 - *IXDP1200 I/O Option Card Design Guide*
 - *Microengine C Compiler Language Support Reference Manual*
 - *Microengine C Compiler LIBC Reference Manual*

In addition, the Intel Web site provides valuable information on products, support, and the company. See "Contacting Intel" on page vii.

Typographic Conventions

This document uses the following typographic conventions to help you locate and identify information:

Italic text Used for new terms, emphasis, and book titles; also identifies arguments in syntax descriptions.

Bold text Identifies keywords and punctuation in syntax descriptions.

`Courier font` Identifies file names, folder names, and text that either appears on the screen or that you are required to type.



NOTE: Provides extra information, tips, and hints regarding the topic.



CAUTION: Identifies important information about actions that could result in damage to or loss of data or could cause the application to behave in unexpected ways.



WARNING!

Identifies critical information about actions that could result in equipment failure or bodily injury.

Installation Path

The pathname of the installation directory for the product is indicated by the following argument:

SDKinstallpath

The environment variable `$IXROOT` is set to the actual installation path during installation; you can use this variable if it is set in your environment.

- The default installation path for a Linux development system is `/opt/ixasdk`.
- The default installation path for a Windows NT development system is `C:\eLinuxIDE-IXP1200\cygwin\opt\ixasdk`.

Contacting Intel

You can reach Intel's automated support services 24 hours a day, every day at no charge. The services contain the most up-to-date information about Intel products. You can access installation instructions, troubleshooting information, and general product information.

Web and Internet Sites

You can use the Internet to download software updates, troubleshooting tips, installation notes, and more.

- General online support services are on the World Wide Web at:

`http://support.intel.com`



- Online support services for the IXP1200 network processor are at:

<http://support.intel.com/support/network/processor/ixp1200/>

For specific types of information and services, go to the following Web and Internet sites:

- **Corporate:** <http://www.intel.com>
- **Networking products:** <http://www.intel.com/network/>
- **Intel IXA information:** <http://developer.intel.com/design/ixa/>
- **IXA SDK:** <http://www.intel.com/design/network/products/software/index.htm>
- **IXP1200 developer site:**
<http://developer.intel.com/design/network/products/npfamily/ixp1200.htm>
- **FTP host:** download.intel.com
- **FTP directory:** [/support/network/adapter](http://support.intel.com/support/network/adapter)

**Customer
Support
Technicians**

- **United States and Canada:** 1.916.377.7000 (7:00 - 17:00 M-F Pacific Time)
- **Worldwide access:** Intel has technical support centers worldwide. Many of the centers are staffed by technicians who speak the local languages. For a list of all Intel support centers, their telephone numbers, and the times they are open, go to:
<http://support.intel.com/support/9089.htm>



Chapter 1

Before Installing the IXA SDK

This chapter specifies what you need to know or to do before you can install the Intel® IXA SDK software. It explains how to configure your development workstation for the IXA SDK, which you use to develop, debug, and deploy code targeted for the IXDP1200 Advanced Development Platform.

This chapter contains the following sections:

- “The IXA Development Software” on page 1
- “Software Requirements” on page 3
- “Hardware and Software Configurations” on page 3
- “System Requirements for Using This Release” on page 6

The IXA Development Software

This section describes the IXA SDK and the IXP1200 Microengine Development Environment, two software development environments designed specifically for IXA application developers.

- You use the IXA SDK to develop IXA applications and to write code modules targeted for the IXDP1200 Advanced Development Platform.
- You use the IXP1200 Microengine Development Environment to write microcode modules (including those for microACEs) targeted for microengines on the IXP1200.

**The IXA SDK**

The IXA SDK contains the tools, libraries, drivers, and other runtime elements that you need for developing, debugging, and delivering code targeted for the IXDP1200 Advanced Development Platform. The IXA SDK also contains languages for describing and classifying packets and for defining communication interfaces.

The IXA SDK runs on Red Hat Linux* 7.2, or on Cygwin* under Windows NT* and is available on the IXA SDK distribution CD 1. The SDK includes the following elements:

- An application programming interface, the IXA API, that includes:
 - Object Management System (OMS) services
 - Action Services Library (ASL) core services
 - ASL application services
 - MicroACE Resource Manager services
- System software, including:
 - The OMS, which acts as a name server and resource manager and provides the framework for interobject communication
 - The microACE Resource Manager, which manages shared memory and communication between the core processor and the microengines
 - An embedded Linux operating system, including a TCP/IP stack and other networking services
 - A boot manager that performs power-on initialization of the IXP1200.
- Run-time libraries, which provide execution and management services for the ASL core and application services for the OMS and Resource Manager.
- Software development tools, including:
 - Network Classification Language (NCL)
 - IXA IDL, an interface definition language
 - Linux compilers, assemblers, linkers and loaders for code in various languages (C/C++, IDL, and NCL), that enable you to develop big-endian StrongARM Linux binaries for the IXP1200
 - Linux utilities that help you run, test, and debug IXA applications
- Predefined building blocks, including:
 - Reusable library ACEs
 - Example code and sample applications
 - Predefined protocol descriptions

**The IXP1200
Microengine
Development
Environment**

The IXP1200 Microengine Development Environment supports the development and integration of microcode modules specifically compiled for microengines (high-speed packet processors) on the IXP1200. It includes microcode (assembly language for the microengines), a cycle-accurate simulator with a graphical user interface (GUI), compilers, debuggers, and application development utilities.

Software Requirements

The Volume I CD-ROM set is preinstalled on the Windows NT single-board computer that comes together with the IXDP1200 Advanced Development Platform. Before you install the software on the Volume II CD-ROM set, you must first remove the Volume I software.

If you are using an x86 Linux development workstation, you must obtain and install Red Hat Linux 7.2 before you can install and use the IXA SDK. Intel does not supply Red Hat Linux. Earlier and later versions of Red Hat Linux are not supported.

You must install the following RPMs, which are not installed by default:

```
xinetd*.rpm  
tftpd-s*.rpm
```

For additional information, go to the Red Hat Web site:

www.redhat.com

Hardware and Software Configurations

Before installing the IXA SDK, you must choose your hardware and software configuration.

Software Configuration

The following describes the possible software configurations that you can use for developing IXA SDK applications.

- The IXP1200 embedded operating system is Linux.
You must download the RAMdisk and Linux kernel onto the IXM1200 Network Processor Base Card.
- Your development environment is a combination of the Windows NT operating system and either a Linux platform or Cygwin for a Linux environment on the Windows NT platform, with the following development tools:
 - Microengine toolchain: Embedded Linux version of the IXP1200 Microengine Development Environment, running under Windows NT.
This enables you to remotely develop and debug code running on Microengines on the IXP1200. You must install this from Volume II, CD 1.
 - StrongARM toolchain: GNU/C++ cross-hosted toolchain. The toolchain utilities can be invoked from a command shell or through the optional Embedded Linux IDE.
This enables you to remotely develop and debug code running on the StrongARM on the IXP1200. You must install this from Volume II, CD 1.
 - Omni-NFS Server, optional.
A 30-day trial version of an NFS server that makes directories located on a Windows NT platform visible to application code running on the StrongARM core processor.

— Linux IDE, optional.

This supports application development, kernel debugging, and device driver development. It includes VMON as its debugger and boot loader.

Figure 1: Linux software configuration using only a Windows NT development environment for the Intel IXP1200 Advanced Development Platform

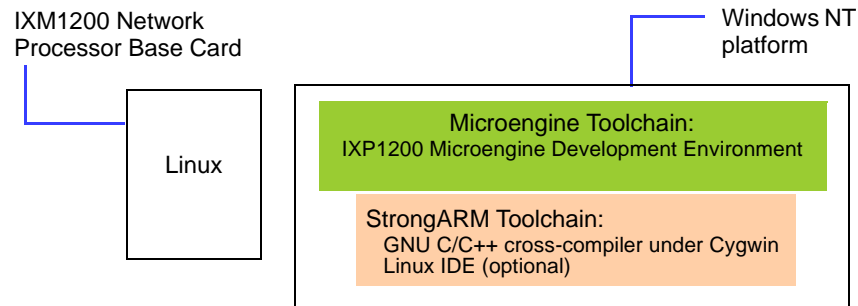
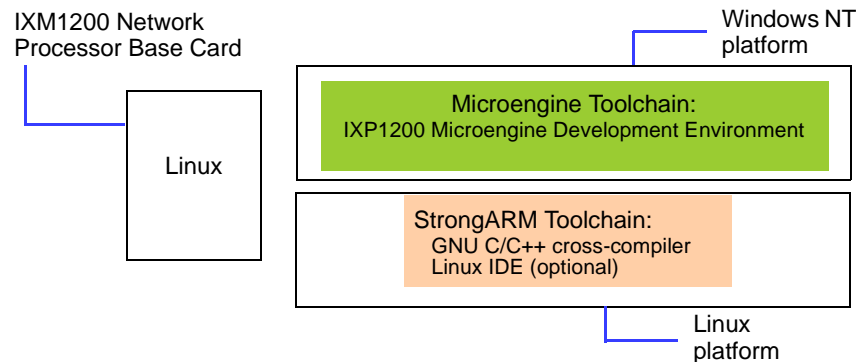


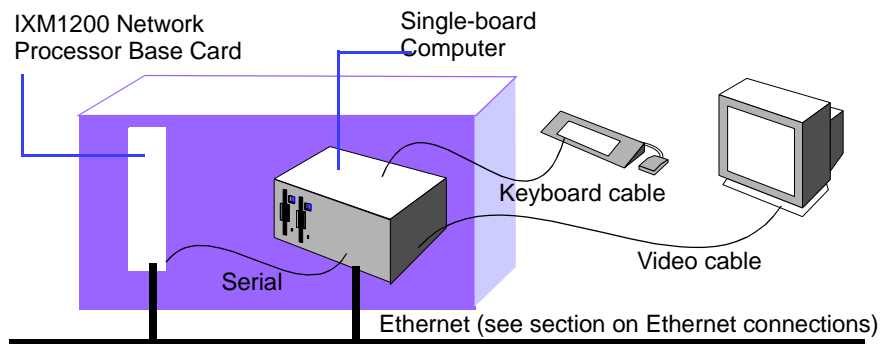
Figure 2: Linux software configuration using both Windows NT and Linux development environments for the Intel IXP1200 Advanced Development Platform



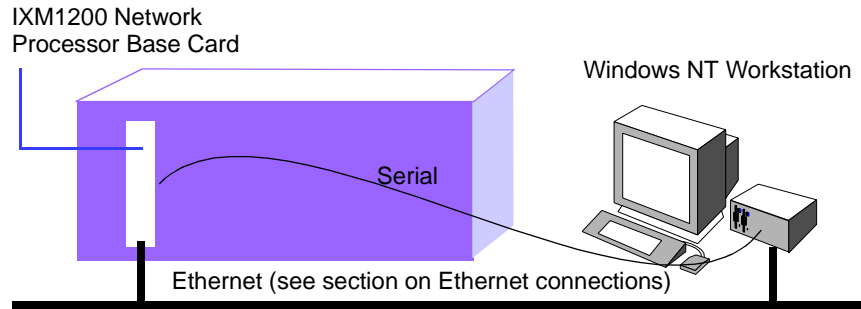
Hardware and Cabling Configurations

You can choose any of the following as your development platform:

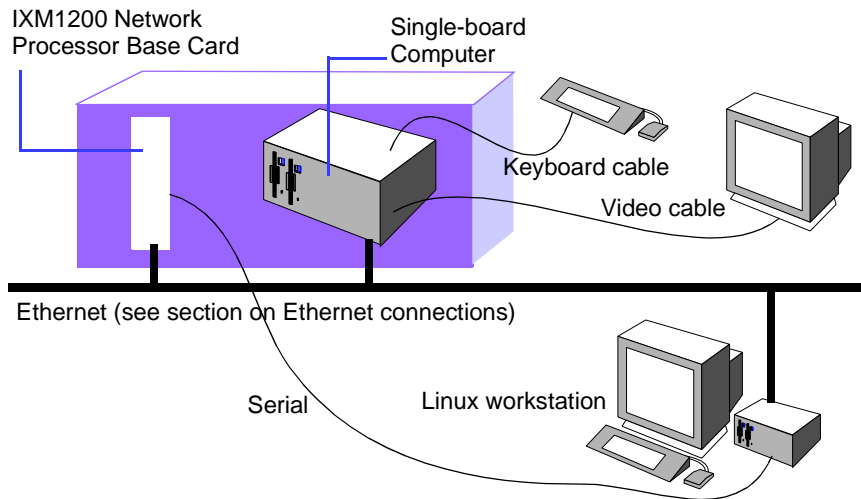
- Configuration 1: Windows NT on the single-board computer on the Intel IXP1200 Advanced Development Platform with an external keyboard and monitor attached.



- Configuration 2: An external Windows NT workstation of your own choosing, in which case the single-board computer is not used.



- Configuration 3: An external Linux workstation of your own choosing in addition to Windows NT on the single-board computer.



The connections in these configurations include:

- A serial cable from the serial port of the IXM1200 Network Processor Base Card to the COM port of the Linux development platform to provide a remote serial console for the IXM1200 Network Processor Base Card.
- Ethernet connections between the IXM1200 Network Processor Base Card and your chosen Windows NT and/or Linux development platforms as described in “Ethernet Connections in All Configurations” on page 6.
- For remote debugging of the IXM1200 Network Processor Base Card, all configurations use the Ethernet connection.
- In configurations 1 and 3, video and keyboard cables from the single-board computer to your own external monitor and keyboard.



Ethernet Connections in All Configurations

For all Ethernet configurations, you can connect any two boards or systems using either:

- A straight Ethernet cable from each system's single Ethernet port to an Ethernet hub or switch.
- An Ethernet cross-over cable that directly connects the two systems' single Ethernet ports.

Note that the IXM1200 Network Processor Base Card has only one single Ethernet port, so if you are connecting more than two systems, you must use an Ethernet hub or switch.

System Requirements for Using This Release

The following table lists the minimum system requirements for the IXA SDK 2.01 release.

Type	Minimum Requirement
Operating Environment	<ul style="list-style-type: none"> • Either <ul style="list-style-type: none"> — Windows NT 4.0 Service Pack 6 or — Red Hat Linux 7.2 • HyperTerminal or minicom remote shell utilities
Document viewer	<ul style="list-style-type: none"> • Adobe Acrobat Reader[*], to view or print IXA SDK documentation in PDF format

Chapter 2

Installing and Configuring the IXA SDK for a Windows NT Development Environment (Configurations 1 and 2)

This chapter describes the installation and configuration of the IXA SDK for the hardware configuration where you are using either the single-board Windows NT computer or an external Windows NT workstation (see Configurations 1 and 2, “Hardware and Software Configurations” on page 3).

- ✓ **NOTE:** Verify that your network is not set to DHCP. Right-click on Network Neighborhood and then click on Properties, then Protocols. In the Protocols window, double-click on TCP/IP. Select Specify an IP address.

This chapter contains the following topics:

- “Removing the Volume I Software” on page 8
- “Upgrading from IXA SDK 2.0” on page 8
- “Installing the IXP1200 Microengine Development Environment” on page 8
- “Installing the IXA SDK Development Environment” on page 11
- “Compiling the System and Library MicroACEs” on page 13
- “Installing the Embedded Linux IDE for the Intel(r) IXP1200 Network Processor” on page 13
- “Setting Up an NFS Server” on page 15
- “Installing Adobe Acrobat Reader” on page 16
- “Wiring Configuration” on page 16
- “Establishing a Remote Terminal” on page 17
- “Configuring the Boot Manager and Cygmon” on page 18
- “Starting the System” on page 19

- “Setting Up an NFS Client” on page 19
- “Configuring the IXP1200 Microengine Development Environment” on page 20
- “Removing the IXA SDK from Windows NT” on page 21

Removing the Volume I Software

If you are using the Windows NT single-board computer as a development workstation (Configuration 1, see “Hardware and Cabling Configurations” on page 4), you must remove the Volume I software, which comes preinstalled on your machine.

To remove the Volume I software, select Add/Remove Programs from the Windows NT Control Panel and remove `Intel IXP1200`.

After the software is removed, you must reboot the computer.

Upgrading from IXA SDK 2.0

If you are upgrading from the IXA SDK 2.0, you must first remove the previous version of the IXP1200 Microengine Development Environment. To do this, select Add/Remove Programs from the Windows NT Control Panel and remove `Intel IXP1200`.

After the software is removed, you must reboot the computer.

Installing the IXP1200 Microengine Development Environment

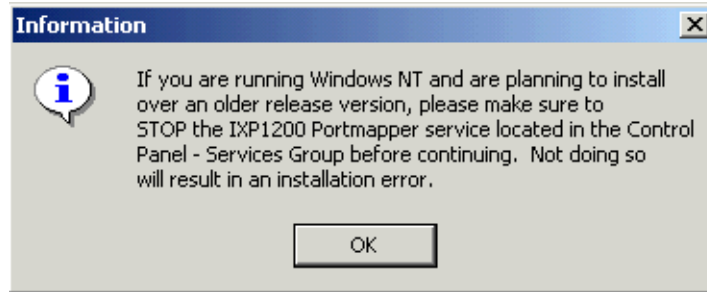
One installation script installs both the IXP1200 Microengine Development Environment and the IXA SDK Development Environment.

- ✓ **NOTE:** The IXDP1200 Advanced Development Platform comes with the Volume I CD set preinstalled. **To continue with this installation, you must first uninstall the Volume I software.** See “Removing the Volume I Software” on page 8.
- ✓ **NOTE:** **If you are upgrading from IXA SDK 2.0, you must first uninstall the IXP1200 Microengine Development Environment.** See “Upgrading from IXA SDK 2.0” on page 8.

To install the IXP1200 Microengine Development Environment:

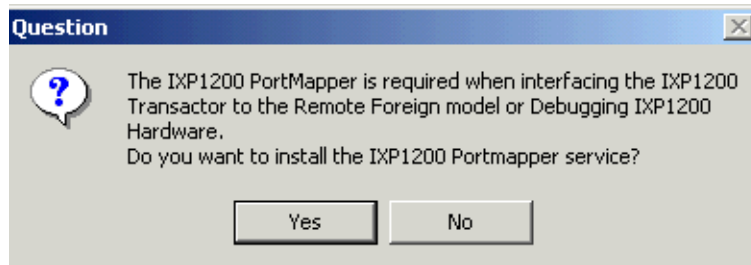
1. Exit all running applications.
2. In the `workbench` directory of the distribution CD 1, double-click `Setup.exe`. The installation begins.

- The following window appears.

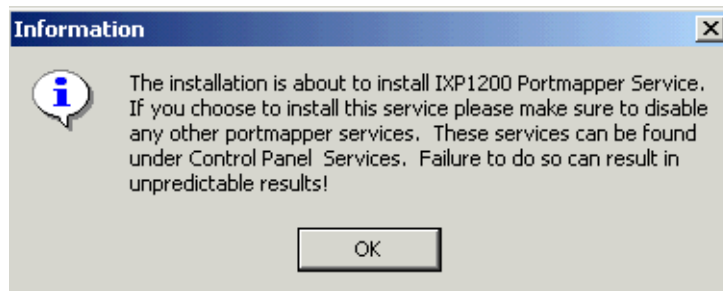


If you are running a previous version of the IXP1200 Microengine Development Environment, select Services from the Control Panel and stop the Portmapper that is currently running. Then click OK to continue.

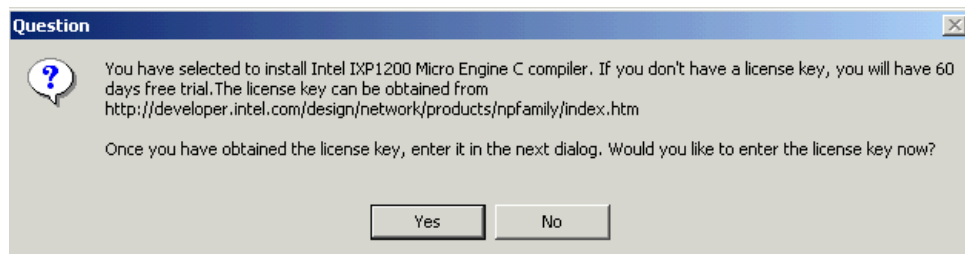
- In the Welcome IXP1200 Developers window, click Next.
- After reading the license agreement, click Yes to agree.
- In the Choose IXP1200 Development Kit Destination Location window, click Next to accept the default of `c:\IXP1200`.
- In the Select IXP1200 Development Kit Components window, ensure that all items are selected and then click Next.
- In the Select Program Folder window, click Next to accept the default of "Intel IXP1200."
- When the following window appears, click Yes to install the Portmapper.



- When the following window appears, click OK to continue.



11. When the following window appears, click Yes if you have a license for the IXP1200 MicroEngine C compiler or click No for the 60-day free trial.

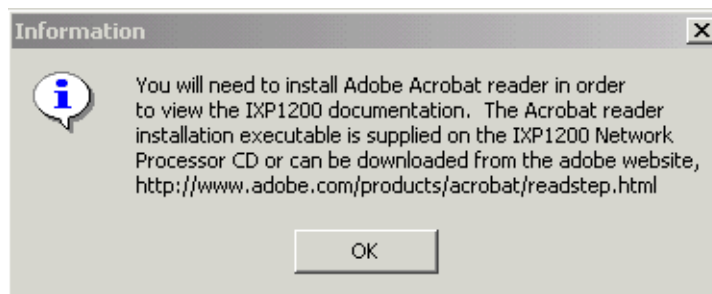


During the trial period, all features are available to you. After the trial period, your data will remain intact but the application software will no longer be available until you obtain a license for it.

12. If you click Yes at the Question window for Step 11, you must enter a password for the MicroEngine C Compiler. After you enter the password, the file installation begins. This may take several minutes.

If you entered No in Step 11, the file installation begins. This may take several minutes.

13. When the following window appears, click OK. For information on installing Adobe Acrobat Reader, see "Installing Adobe Acrobat Reader" on page 16.



14. When " Workbench Installation Complete: Starting Cygwin Setup" appears, click OK.

The installation of the IXP1200 Microengine Development Environment is complete. The installation of the development environment automatically follows.

Installing the IXA SDK Development Environment

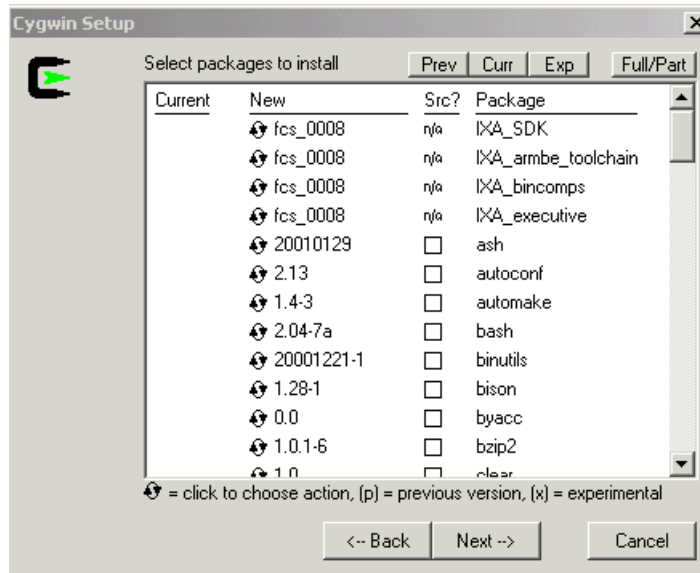
After the installation of the IXP1200 Microengine Development Environment completes, the installation of the IXA SDK development environment automatically follows. The IXA SDK development environment includes:

- Cygwin*
- Microcode source and binary files
- StrongARM* development source files

The installation continues automatically after the completion of the IXP1200 Microengine Development Environment installation.

- ✓ **NOTE:** If you have a previous version of Cygwin installed on your Windows NT computer, this version of Cygwin will become the default version.
1. When the Cygwin Setup and release information window appears, click Next.
 2. In the dialog box for the location of the installation files, choose Install from Local Directory and click Next.
 3. When prompted for Local Package Directory, click Next to accept the default `nt_sdk` directory on the CD drive.
- ✓ **NOTE:** In the next step, if you choose not to use the default Install root directory, then replace the default, `eLinuxIDE-IXP1200`, with the directory that you have chosen for the remainder of this guide.
- ✓ **NOTE:** If you plan to use the IXA IDE for Embedded Linux, you must ensure that the top-level directory (`eLinuxIDE-IXP1200` in the default case) you enter in the following step is the same as the directory that you choose during the IDE installation. The default for the IDE Installation is `eLinuxIDE-IXP1200`. For more information, see “Installing the Embedded Linux IDE for the Intel(r) IXP1200 Network Processor” on page 13.
4. In the Select Install root directory dialog box:
 - a. Choose the path in which to install the product. By default, the product is installed in `c:/eLinuxIDE-IXP1200/cygwin`.
 - b. For Default Text File Type, choose UNIX.
 - c. For Install For, choose Just me.
 - d. Click Next.

- When the following window appears, accept the default by clicking Next.



You must accept the default settings; do not click any of the buttons or part numbers. The buttons at the top of the page allow you to select previous, current, or experimental versions of the software. Clicking on the individual parts includes or excludes parts.

After you have clicked Next, files begin installing. This takes several minutes.

- Ensure that the check boxes to Create a Desktop Icon and Add to Start Menu are both selected and click Next.
- The installation is finished after the Cygwin* Setup Post-Install Script window appears and vanishes.
- In the Installation Complete window, click OK.
- A window appears as a reminder that you must install an NFS server and a TFTP server. For more information, refer to “Installing the Embedded Linux IDE for the Intel(r) IXP1200 Network Processor” on page 13 and “Setting Up an NFS Server” on page 15.
Click OK.
- In the Setup Complete window, click OK.

The installation of the IXA SDK development environment is now complete. The next two sections are optional.

Compiling the System and Library MicroACEs

Before you can run IXA SDK applications, compile the system and library microACEs. You can either do this step now or at a later time.

To compile the system and library microACEs:

1. From the Windows NT Start Menu, choose **Programs->Cygwin Solutions->Cygwin Bash Shell**

This displays a command-line shell.

2. In the Cygwin Bash Shell, type:

```
cd src/microace
make
```

This makefile takes several minutes to complete.

3. Compile the system and library microACE microblocks. At the Cygwin prompt, type:

```
cd ucbuild
make -f Makefile.win
```

This makefile takes several minutes to complete.

4. Type `exit` to close the Bash Shell.

The compiling of the microACEs is complete.

Installing the Embedded Linux IDE for the Intel(r) IXP1200 Network Processor

The embedded Linux IDE is an alternative to the GNU cross-hosted StrongARM* toolchain for Linux developers. If you choose not to install the embedded Linux IDE, proceed to "Setting Up a TFTP Server."

- ✓ **NOTE:** *Before you install the Embedded Linux IDE, you must first install IXP1200 Microengine Development Environment and the IXA SDK development environment.*

1. Insert the Embedded Linux IDE for the Intel(r) IXP1200 Network Processor CD-ROM and double-click `setup.exe`.
2. In the Welcome window, click Next.
3. Read the Software License Agreement. If you agree to the license terms, click Yes.

- ✓ **NOTE:** In the following step, if you do not accept the default destination location, `C:\eLinuxIDE-IXP1200`, the IXA SDK software will not run properly.

4. In the Choose Destination Location window, click Next to accept the default of `C:\eLinuxIDE-IXP1200`. If you do not use the default location, the system will not work properly.

5. In the Select Program Folder window, click Next to accept the default value of Intel(r) IXA IDE for Embedded Linux on IXP1200.
6. In the Setup Complete window, click Finished. The Setup Complete Window may be hidden by other open windows.
7. Remove the Embedded Linux IDE for the Intel(r) IXP1200 Network Processor CD-ROM.

Setting Up a TFTP Server

If you want to boot the IXP1200 Linux from the network, you must set up a trivial file transit protocol (TFTP) server on your Windows NT development machine. To install the TFTP server, choose either of the following:

- Install using the TFTP server provided on distribution CD 1.
- Download the software onto your system from:

http://membres.tripod.fr/phjounin/P_tftpd32.htm

To install TFTP server, TFTP32, from the distribution CD 1:

1. Create a directory called `C:\tftp-server`.
2. From the folder `nt_utils\tftpd32e` on CD 1, copy the file `tftpd32.exe` to `C:\tftp-server`.
3. Create a directory called `C:\tftpboot`.
4. Copy the files `zImage` and `ramdisk_img.gz` from `c:\eLinuxIDE-IXP1200\cygwin\opt\ixasdk\executive\bin` to `C:\tftpboot`.
5. In the `C:\tftp-server` folder, double-click `tftpd32.exe`. This starts the TFTP server. You might want to create a shortcut for this on your desktop.
6. Within the `tftpd32` application window, click Settings and change the base directory to `c:\tftpboot`.

Setting Up an NFS Server

To use the IXA SDK, you must install an NFS server. You may install any NFS servers or you may choose either of the following:

- Install the 30-day free trial of OMNI NFS Server (v 4.13) provided on CD 1.
- Purchase the OMNI NFS Server (v 4.13). For information, refer to:

<http://www.xlink.com/products.htm>

Installing the NFS Server

To install the evaluation copy from CD 1:

1. In the `nt_utils` folder of the CD 1, double-click on `nfserver.exe`
2. In the Welcome window, click Next.
3. Leave the Serial # and Password fields blank and click Next to install the evaluation copy.
4. Click Yes to continue with the installation of the evaluation copy.
5. Read the installation agreement and click Yes.
6. In the Choose Destination Location window, click Next to accept the default location, `C:\Program Files\Nfserver`.
7. In the Select Program Folder window, click Next to accept the default of `Omni-NFS Server V4.13`. Files begin to install.
8. When the installation is complete, click Finish.

Configuring the NFS Server

To configure the NFS Server:

1. Launch the NFS server. From the Windows NT Start menu, choose:
`Programs->Omni-NFS Server V4.13->NFS Server`
2. If you installed the Portmapper service for the IXP1200 Microengine Development Environment, you must disable the NFS Portmapper. You can verify that the Intel IXP1200 Portmapper is started by selecting Service from the Windows NT Control Panel.

To disable the NFS Portmapper:

- a. Click Options.
- b. Make sure that the Enable XLink Port Mapper box is *cleared (not checked)*.
- c. Click OK to return to the Configuration Center window.

3. From the Configuration Center window:
 - a. Click New.
 - b. For Drive, select C:.
 - c. For Path, enter or select:
`\eLinuxIDE-IXP1200\cygwin\opt\ixasdk\bin\arm-be`
 - d. Click OK
 - e. A window appears asking if you want to restart the NFS service from the Control Panel. Click No.
You are returned to the Configuration Center.
4. Close the NFS Server and reboot your machine.
5. When the system comes up, check under Services in the Windows NT Control Panel to ensure that the Omni-NFS Server service is started.

Installing Adobe Acrobat Reader

Documentation is provided in Adobe PDF format, viewable from the Adobe Acrobat Reader. For your convenience, this tool is provided on CD 1.

To install Adobe Acrobat Reader from CD 1:

1. In the `nt_utils` directory of the CD, double-click `rs405eng.exe`.
2. In the Acrobat Reader 4.05 Setup window, click Next.
3. Click Next to accept the default location for the program files or click Browse to select a different location.
The installation process begins copying files.
4. When the installation is complete, click OK.

Wiring Configuration

Before continuing with the configuration of the IXDP1200 Advanced Development Platform, ensure that your machines have a correct wiring configuration (see Configuration 1 and 2, "Hardware and Software Configurations" on page 3).

For Configuration 1 (single-board Windows NT computer), the following are required:

- A serial connection between the single-board Windows NT computer and the IXP1200 network processor.
- An Ethernet crossover connection between the single-board Windows NT computer and the IXP1200 network processor.



NOTE: The Ethernet port for the the IXP1200 network processor is labeled 10/100 E-NET DEBUG PORT.

For Configuration 2 (External Windows NT Workstation, single-board Computer Unused), the following are required:

- A serial connection between the external Windows NT workstation and the IXP1200 network processor.
- The external Windows NT workstation and the IXP1200 network processor must be connected through an Ethernet hub.

Establishing a Remote Terminal

To communicate with the IXP1200 network processor, you must establish a remote terminal from your development workstation.

From Windows NT, you can use the HyperTerminal application to connect to the IXP1200 network processor. HyperTerminal requires a serial connection between your development platform and the IXP1200 network processor.

To use HyperTerminal:

1. From the Windows NT Start Menu, select:
Programs->Accessories->HyperTerminal->HyperTerminal
2. The first time you run HyperTerminal, a window may appear asking for configuration information such as local area code. Enter the requested information and click OK.
3. If you are asked if you want to install a modem, click No.
4. Type a name for the new connection and press Enter.
5. Select COM1 or COM2, depending on how you have connected the serial port, and press Enter.
6. Where necessary, change the connection settings to the following and click OK.

Setting Name	Setting
Bits Per Second	38400
Data bits	8
Parity	None
Stop Bits	1
Flow Control	None

You have now started a remote shell for the IXP1200 network processor on the development workstation.


Configuring the Boot Manager and Cygmon

Initially, the IXM1200 Network Processor Base Card has no operating system; instead, it contains a Boot Manager that allows you to install one. When you power up the IXP1200 network processor, the Boot Manager is the first to run. The Boot Manager allows you to select from several operating systems; Cygmon^{*}/Linux^{*} is the default choice, which you must use with this release of the IXA SDK.

Cygmon is the tool you use to install the Linux operating system on the IXP1200 network processor. Cygmon includes boot code, a debugger for bootlevel code, and code for downloading and booting Linux via Ethernet^{*} or the serial port. Linux includes a compressed kernel (in `gzip` format) and RAMdisk.

You must configure both the Boot Manager and Cygmon before you can boot Linux. To do this, use HyperTerminal as a remote terminal to the IXP1200 network processor from your Windows NT workstation:

1. Ensure that the TFTP server is running on the Windows NT workstation. To run the TFTP server, double-click on `c:\tftp-server\tftpd32.exe`.
2. If you do not already have a HyperTerminal window open, then open a HyperTerminal window. Refer to “Establishing a Remote Terminal” on page 17.
3. Press the Reset button on the IXP1200 network processor.
4. During the autoboot countdown, type a space during the countdown before it starts booting. (If you miss this, press reset and try again.)
5. Type `c` for configure. This will start the Boot Manager.
6. At the `[BootMgr]` prompt in the Boot Manager, do the following:
 - a. At the Default Operating System prompt, select 3 for Cygmon.
 - b. At the Countdown value prompt, press Enter to accept the default.
 - c. At the Disable initial display prompt, press Enter to accept the default.
 - d. At the Default IO type prompt, press Enter to accept the default.
 - e. At the SDRAM Window Size prompt, press Enter to accept the default.
 - f. At the SDRAM Window Offset prompt, press Enter to accept the default.
 - g. At the Upstream Window Size prompt, press Enter to accept the default.
 - h. Boot by typing `b`.

 **NOTE:** To verify the IP address for the port you are using on the Windows NT computer, right-click on Network Neighborhood, click on the Adapters tab, and then double-click on the adapter you are using in order to see the IP address for the port.

7. At the `cygmon` prompt, type `bo` for boot options. This allows you to set the static IP addresses for the IXP1200 network processor and for the x86 workstation:
 - a. Enter 2 to set the local IP address. Then enter the static IP address of the IXP1200 network processor. The local IP address must have the same subnet mask as the remote IP address.
 - b. Enter 3 to set the remote IP address. Then enter the static IP address for the Windows NT server.
 - c. Enter 8 to enter a countdown value to auto-Linux boot. Enter 10 for the countdown value.
 - d. Type `s` to save this configuration in flash memory.
 - e. Type `g1` to boot Linux.

Starting the System

After you have completed the previous steps, boot Linux as follows:

1. Open a HyperTerminal window (if one is not already open) and press the Reset button on the IXP1200 network processor.
The remote shell displays the Linux login prompt.
2. Log in as `root`, using `ixp1200` as the password.
The remote shell displays the Linux command-line prompt.

3. To start the network, run the `ifup` script:

```
$ ifup <local_ipaddr> <subnet_mask> <broadcast> <gateway_ipaddr>
```

— `local_ipaddr` is the static IP address of the IXP1200 network processor .

— `subnet_mask` is the subnet mask.

— `broadcast` is the broadcast address.

— `gateway_ipaddr` is the gateway IP address.

Every time you boot, run the `ifup` script to set the addresses.

Setting Up an NFS Client

If you want to use NFS for file sharing, you must set up your development workstation as an NFS server and the IXP1200 network processor as an NFS client. For information on how to set up your Windows NT workstation as an NFS server, see “Setting Up an NFS Server” on page 15.

The IXP1200 Linux kernel and RAMdisk image are NFS-ready. To set up the IXP1200 network processor as an NFS client:

1. If you are not already logged into Linux through the remote shell, then:
 - a. Log in as `root` in a remote command shell for the IXP1200 network processor.



- b. Start the `portmap` daemon. On the current RAMdisk image the `portmap` daemon is automatically started when you run the `ifup` script.
2. Mount the remote directory on the host file system by typing, all on one line, the following command:

```
mount -t nfs
<server-ip>:C:/eLinuxIDE-IXP1200/cygwin/opt/ixasdk/bin/arm-be /nfs
```

3. Change directory to the mounted directory. Type:

```
cd /nfs
```

Configuring the IXP1200 Microengine Development Environment

You use the IXP1200 Microengine Development Environment to develop, debug, and deploy microcode specifically targeted for microengines on the IXP1200 network processor.

To use the IXP1200 Microengine Development Environment:

1. Be sure that you have installed the IXP1200 Microengine Development Environment on your Windows NT workstation according to the instructions in “Installing the IXP1200 Microengine Development Environment” on page 8.
2. If you are not running DNS on the IXP1200, add the name of the Windows NT host to the `/etc/hosts` file on the IXP1200. To do this, be sure that you are logged into Linux through a HyperTerminal window, and then enter the following commands:

- a. Use the `cat` utility:

```
cat > /etc/hosts
```

- b. Type:

```
<IP Address of NT host> <name_of_NT_host>
```

You can find the `<name of NT_host>` by looking at:

```
Network Neighborhood->Properties->Protocols->TCP/IP->DNS
```

Be sure that this name is the same as the machine name in Network ID.

- c. Type CTRL D.

If are using the Intel IXA SDK ACE Programming Framework, then use the IXP1200 Microengine Development Environment to connect to your Linux target (the IXP1200 network processor). For information on running a sample application, see the *IXA SDK Tutorial*.

If you are not using the Intel IXA SDK ACE Programming Framework, then to debug a microengine C application, type:

```
insmod ue.o; rs_udebug &
```

This starts the microcode debugger backend on the IXP1200 network processor.



Removing the IXA SDK from Windows NT

To remove the IXP1200 Microengine Development Environment, select Add/Remove Programs from the Windows NT Control Panel and remove `Intel IXP1200`.

To remove the IXA SDK, delete the `eLinuxIDE-IXP1200\cygwin` directory from your system.

To remove the Omni-NFS Server, select Add/Remove Programs from the Windows NT Control Panel and remove `Omni-NFS Server`.

To remove TFTP32, delete the `tftp-server` and `tftpboot` directories from your system.

To remove the embedded Linux IDE, select Add/Remove Programs from the Windows NT Control Panel and remove `Embedded Linux IDE for the Intel IXP1200`.



Chapter 3

Installing and Configuring the IXA SDK for a Linux Development Environment (Configuration 3)

This chapter describes the installation and configuration of the IXA SDK for the hardware configuration where you are running an external Linux workstation and a single-board Windows NT computer (see “Hardware and Software Configurations” on page 3).

You develop applications using the IXA SDK that run on both the StrongARM core processor and on the microengines of the IXP1200 network processor.

Using Configuration 3, you use an external Linux workstation as a development workstation to develop software for the StrongARM core. You use the Windows NT single-board computer to develop and deploy microcode applications targeted for the IXP1200 microengines.

- ✓ **NOTE:** To use the IXA SDK, you must complete the installation sections for both the Windows NT single-board computer and for your Linux development workstation.
- ✓ **NOTE:** Verify that your network is not set to DHCP. Right-click on Network Neighborhood and then click on Properties, then Protocols. In the Protocols window, double-click on TCP/IP. Select Specify an IP address.

The chapter includes the following sections:

- “Installing the IXA SDK Microcode Development Environment” on page 24
- “Installing the Linux StrongARM Development Environment” on page 29
- “Setting Up a TFTP Server for a Linux Host” on page 30
- “Setting Up an NFS Server” on page 31
- “Wiring Configuration” on page 31



- “Establishing a Remote Terminal” on page 32
- “Configuring the Boot Manager and Cygmon” on page 33
- “Starting the System” on page 34
- “Setting Up an NFS Client” on page 35
- “Configuring the IXP1200 Microengine Development Environment” on page 36

Installing the IXA SDK Microcode Development Environment

You use the Windows NT single-board computer to develop and deploy microcode applications targeted for the IXP1200 microengines.

This section includes the following topics:

- “Removing the Volume I Software” on page 24
- “Upgrading from IXA SDK 2.0” on page 24
- “Installing the IXP1200 Microengine Development Environment” on page 25
- “Installing Cygwin and the Microcode Libraries” on page 27
- “Compiling the System and Library MicroACEs” on page 28
- “Installing Adobe Acrobat Reader” on page 28

Removing the Volume I Software

For Configuration 3, you must first remove the Volume I software, which comes preinstalled on your Windows NT single-board computer.

To remove the Volume I software, select Add/Remove Programs from the Windows NT Control Panel and remove IXP1200.

After the software is removed, reboot the computer.

Upgrading from IXA SDK 2.0

If you are upgrading from the IXA SDK 2.0, you must first remove the previous version of the IXP1200 Microengine Development Environment on the Windows NT single-board computer.

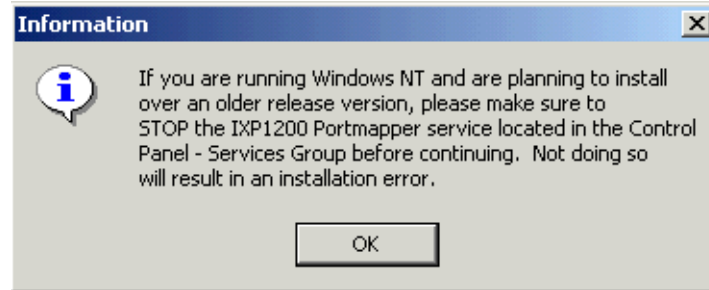
To do this, select Add/Remove Programs from the Windows NT Control Panel and remove Intel IXP1200.

After the software is removed, you must reboot the computer.

Installing the IXP1200 Microengine Development Environment

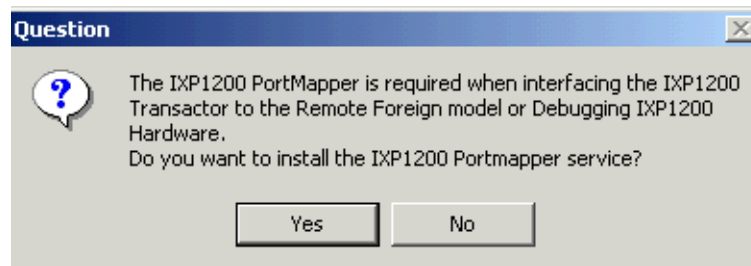
To install the IXP1200 Microengine Development Environment:

1. Exit all running applications.
2. In the `workbench` directory of CD 1, double-click `Setup.exe`. The installation begins.
3. The following window appears.

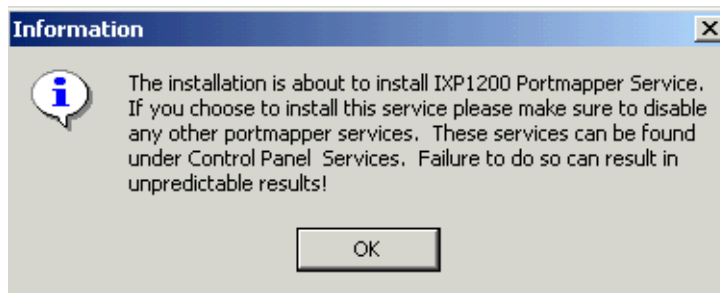


If you are running a previous version of the IXP1200 Microengine Development Environment, select Services from the Control Panel and stop the Portmapper that is currently running. Then click OK to continue.

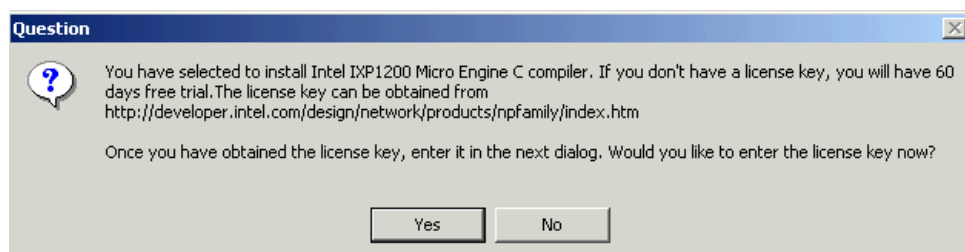
4. In the Welcome IXP1200 Developers window, click Next.
5. After reading the license agreement, click Yes to agree.
6. In the Choose IXP1200 Development Kit Destination Location window, click Next to accept the default of `c:\IXP1200`.
7. In the Select IXP1200 Development Kit Components window, ensure that all items are selected and then click Next.
8. In the Select Program Folder window, click Next to accept the default of "Intel IXP1200."
9. When the following window appears, click Yes to install the Portmapper.



10. When the following window appears, click OK to continue.

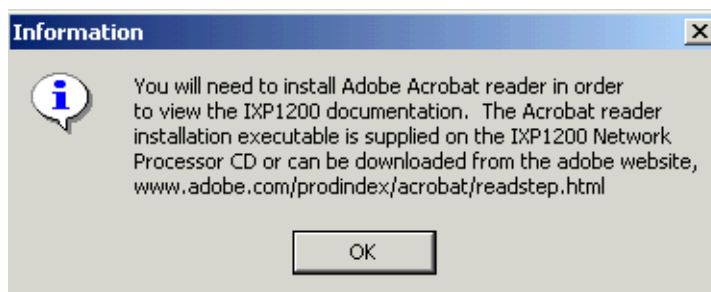


11. When the following window appears, click Yes if you have a license for the IXP1200 MicroEngine C compiler or click No for the 60-day free trial.



During the trial period, all features are available to you. After the trial period, your data will remain intact but the application software will no longer be available until you obtain a license for it.

12. If you click Yes at the Question window for Step 11, you must enter a password for the MicroEngine C Compiler. After you enter the password, the file installation begins. This may take several minutes.
- If you entered No in Step 11, the 60-day free trial installation begins. This may take several minutes.
13. When the following window appears, click OK. For information on installing Adobe Acrobat Reader, see "Installing Adobe Acrobat Reader" on page 28.



14. When "Workbench Installation Complete: Starting Cygwin Setup" appears, click OK.

Installing Cygwin and the Microcode Libraries ✓

The installation of Cygwin automatically follows the installation of the IXP1200 Microengine Development Environment.

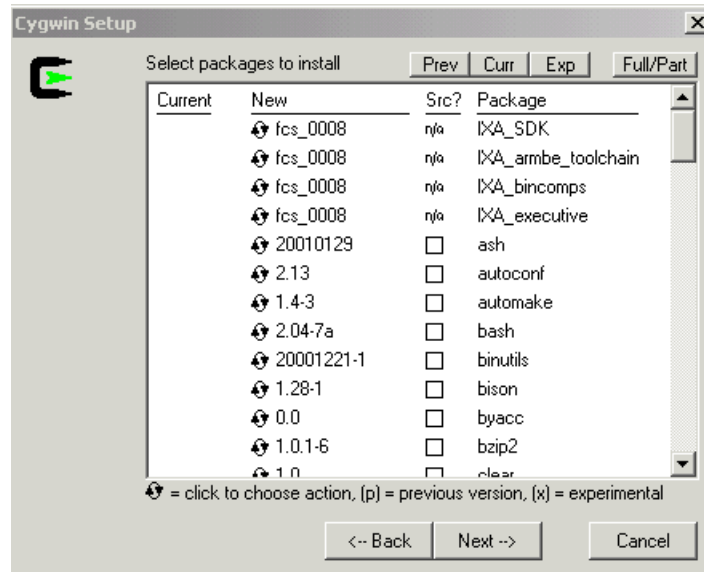
NOTE: If you have previous version of Cygwin installed on your Windows NT computer, this version of Cygwin will become the default version.

1. When the Cygwin Setup and release information window appears, click Next.
2. In the dialog box for the location of the installation files, choose Install from Local Directory and click Next.
3. When prompted for Local Package Directory, click Next to accept the default `nt_sdk` directory on the CD drive.



NOTE: In the next step, if you choose not to use the default Install root directory, then for the remainder of this guide replace the default, `eLinuxIDE-IXP1200`, with the directory that you have chosen.

4. In the Select Install root directory dialog box:
 - a. Choose the directory in which to install the product. By default, the product is installed in `c:\eLinuxIDE-IXP1200\cygwin`.
 - b. For Default Text File Type, choose UNIX.
 - c. For Install For, choose Just me.
 - d. Click Next.
5. When the following window appears, accept the default by clicking Next.



You must accept the default settings; do not click on any of the buttons or part numbers. The buttons at the top of the page allow you to select previous, current, or experimental versions of the software. Clicking on the individual parts includes or excludes parts.

After you have clicked Next, files begin installing. This takes several minutes.



6. Ensure that the check boxes to create a desktop icon and add to Start menu are both selected and click Next.
The installation is finished after the Cygwin* Setup Post-Install Script window appears and vanishes.
7. In the Installation Complete window, click OK.
8. A window appears as a reminder that you must install an NFS server and a TFTP server on your Windows NT computer. Since you are using an external Linux workstation as a development environment, you need not install an NFS server or a TFTP server on your Windows NT computer.
Click OK.
9. In the Setup Complete window, click OK.

Compiling the System and Library MicroACEs

Before you can run IXA SDK applications, you compile the system and library microACEs. You can either do this step now or at a later time.

To compile the system and library microACEs:

1. From the Windows NT Start Menu, choose **Programs->Cygnus Solutions->Cygwin Bash Shell**
This displays a command-line shell.
2. Compile the system and library microACE core components. From the Windows Desktop, click the Cygwin icon to bring up a Cygwin Bash Shell and type:

```
cd src/microace
make
```

This makefile takes several minutes to complete.

3. Compile the system and library microACE microblocks. At the Cygwin prompt, type:

```
cd ucbuild
make -f Makefile.win
```

This makefile takes several minutes to complete.

Installing Adobe Acrobat Reader

Documentation is provided in Adobe PDF format, viewable from the Adobe Acrobat Reader. For your convenience, this tool is provided on the distribution CD-ROM.

To install Adobe Acrobat Reader from the distribution CD-ROM:

1. In the `nt_utils` directory of the CD-ROM double-click `rs405eng.exe`.
2. At the Acrobat Reader 4.05 Setup window, click Next.
3. Click Next to accept the default location for the program files or click Browse to select a different location.
The installation process begins copying files.
4. When the installation is completed, click OK.

Installing the Linux StrongARM Development Environment

This section explains the software installation of the IXA SDK software. The IXA SDK is available on the IXA SDK distribution CD 1 of Volume II.

This section contains the following topics:

- “Installing the Development Environment” on page 29
- “Using StrongARM Compiler Options” on page 30
- “Using StrongARM Compiler Options” on page 30

Installing the Development Environment

After you have installed Red Hat[®] Linux 7.2 on your x86 workstation, you can install the IXA SDK.

To install the IXA SDK on your development workstation:

1. Log in as `root` in a Linux command shell on your Linux development workstation.
2. In the Linux command shell, go to the RPMS directory on the CD-ROM mount location:

```
$ cd /mnt/cdrom/rpms
```

3. Use the RPM utility to install the IXA SDK packages:

```
rpm -i ixa.sdk-2.01.D-fcs.i386.rpm
rpm -i ixa.executive-2.01.D-fcs.i386.rpm
rpm -i ixa.armbe-v4b-fcs.i386.rpm
rpm -i ixa.binaries-2.01.D-fcs.i386.rpm
```

This release does not allow you to change the root directory for the IXA SDK. The default location is `/opt/ixasdk`.

The utility installs the following components:

- The IXA SDK components.
 - The flash image, compressed Linux kernel, and compressed RAMdisk, used later in configuring the IXP1200 network processor.
 - The GNU cross-hosted toolchain. The toolchain is a set of Linux compilers, assemblers, linkers, loaders, and debuggers for code in C/C++. These tools enable you to develop big-endian StrongARM[®] Linux binaries for the IXP1200 network processor on your x86 Linux host.
 - IXA-specific utilities and compilers.
4. To get access to the GNU toolchain, update your path as follows:

```
PATH=$PATH:/usr/local/armbe/bin
```

5. To get access to the IXA-specific utilities and compilers (`tao_idl` and `nclcomp`), update your path as follows:

```
PATH=$PATH:/opt/ixasdk/bin
```

6. Set the following environment variables:

```
IXROOT=/opt/ixasdk
CONFIG=ARM_BE
```

See the `README.txt` file at the top level of the CD-ROM for a directory map and locations of documentation, including release notes.

**Using
StrongARM
Compiler
Options**

The names of all compiler utilities are prefixed with `armv4b-unknown-linux-`. For example, `armv4b-unknown-linux-gcc`.

Use the following compiler and linker options when compiling code specifically for the StrongARM processor:

- `-mbig-endian` option for `gcc`. For example:

```
armv4b-unknown-linux-gcc -mbig-endian file1.c
```
- `-EB` option for `ld`. For example:

```
armv4b-unknown-linux-ld -EB "file1.o file2.o" app
```

Setting Up a TFTP Server for a Linux Host

If you want to boot the IXP1200 Linux from the network, you must set up a trivial file transfer protocol (TFTP) server on your x86 Linux development machine. The following steps set up a TFTP server:

1. Log in as `root` in a Linux command shell on the development workstation.
2. Install the `tftp-0.17-14` and `tftp-server-0.17-14` packages from the Red Hat 7.2 CD2. For Firewall Configuration, choose No Firewall.
3. Edit the `tftp` file under `/etc/xinetd.d` to specify the following value:

```
server_args = -s /
```

4. Set the access rights for the TFTP boot directory.

```
chmod 777 /tftpboot
```

5. Find the process ID for `xinetd`:

```
ps aux | grep xinetd
```

6. Kill the process, replacing `pid` with the process ID for `xinetd`:

```
kill -9 pid
```

7. Run `xinetd` again:

```
xinetd
```

Setting Up an NFS Server

You can set up your x86 workstation as a network file system (NFS) server to expedite file sharing. To use NFS, you also need to set up the IXP1200 as an NFS client; see “Setting Up an NFS Client” on page 35.

You can use NFS to mount a directory from the hard drive of your host machine on the IXP1200. This is superior to transferring files using FTP or copying them, since you are dealing with executables (ACEs), not just source files.

For more details on using NFS, including how to optimize for speed, see:

<http://www.Linux.org/docs/ldp/howto/NFS-HOWTO.html>

Below are the steps to set up your workstation as an NFS server:

1. Log in as `root` in a Linux command shell on the development workstation.
2. Check to see whether the `nfs-utils-0.3.1-12` package is already installed on your workstation and install it if necessary.
3. Add a line to the file `/etc/exports`:

```
ip-addr:/nfsdir *(rw)
```

 - `ip-addr` is the host IP address.
 - `:/nfsdir` is the pathname of the directory that you want to export. This directory will be visible to the IXP1200.

You can use read-only permissions (`ro`), instead of read-write (`rw`).
4. Run `exportfs -a` to update the tables so that your changes will take effect.

Wiring Configuration

Before continuing with the configuration of the IXP1200 network processor, ensure that your machines have a correct wiring configuration (see Configuration 3, “Hardware and Cabling Configurations” on page 4).

For this configuration, the following must all be connected via an Ethernet hub:

- Linux development machine
- Single-board Windows NT computer
- IXP1200 network processor

✓ **NOTE:** The Ethernet port for the the IXP1200 network processor is labeled as 10/100 E-NET DEBUG PORT.

There must also be a serial connection between the Linux workstation and the IXP1200 network processor.

Establishing a Remote Terminal

To communicate with the IXP1200 network processor, you must establish a remote terminal on your development workstation.

From a Linux shell, you can use the Linux `minicom` tool to connect over the serial port to the IXP1200 network processor. Minicom allows you to write scripts that automatically enter information that you want to save. A minicom script can make it easier to boot up the IXP1200 network processor; see “Using a Minicom Script for Startup” on page 34.

The minicom tool requires a serial connection between your development platform and the IXP1200 network processor. For further details on connecting to a Linux workstation, see “Hardware and Cabling Configurations” on page 4.

For example, to use minicom from a Linux workstation:

1. Log in as `root` in a Linux command shell on the development workstation.

2. Delete the modem device:

```
$ rm -f /dev/modem
```

3. Create a soft link between the modem device and the serial port terminal (`ttys0` for `com1`, or `ttys1` for `com2`):

```
$ ln -s /dev/ttys0 /dev/modem
```

4. Create a directory for the minicom scripts:

```
$ mkdir /minicom
```

5. Run minicom:

```
$ cd /minicom
$ minicom
```

If the serial port is locked, an error message appears and the minicom shell does not start. In this case, go to `/var/lock` and remove the lock file for the port.

6. Type `CTRL A O` to get a minicom configuration menu.
7. Choose `serial port setup` and set up for 38400 8N1 and no flow control (neither hardware nor software).
8. In the minicom menu, choose `save setup as` and save it as `ixp1200`.
9. Choose `exit`.
10. Exit minicom with `CTRL A X`.
11. To select the previously-saved configuration, start minicom by typing:

```
$ minicom ixp1200
```

12. Reboot the IXP1200.

Now you can start a remote shell for the IXP1200 network processor on the development workstation. Use the remote shell to install the Linux operating system on the IXP1200; see Chapter 2, “Installing and Configuring the IXA SDK for a Windows NT Development Environment (Configurations 1 and 2).”

Configuring the Boot Manager and Cygmon

Initially, the IXM1200 Network Processor Base Card has no operating system; instead, it contains a Boot Manager that allows you to install one. When you power up the IXP1200 network processor, the Boot Manager is the first to run. The Boot Manager allows you to select from several operating systems; Cygmon^{*}/Linux^{*} is the default choice, which you must use with this release of the IXA SDK.

Cygmon is the tool you use to install the Linux operating system on the IXP1200 network processor. Cygmon includes boot code, a debugger for bootlevel code, and code for downloading and booting Linux via Ethernet^{*} or the serial port. Linux includes a compressed kernel (in `gzip` format) and RAMdisk.

You must configure both the Boot Manager and Cygmon before you can boot Linux. To do this, use `minicom` as a remote terminal to the IXP1200 network processor from your Linux development workstation:

1. Ensure that the TFTP server is running. See “Setting Up a TFTP Server for a Linux Host” on page 30
2. Open a `minicom` window. Refer to “Establishing a Remote Terminal” on page 32
3. Press the Reset button on the IXP1200 network processor.
4. During the autoboot countdown, type a space during the countdown before it starts booting. (If you miss this, press reset and try again.)
5. Type `c` for configure. This will start the Boot Manager.
6. At the `[BootMgr]` prompt in the Boot Manager, do the following:
 - a. At the Default Operating System prompt, select 3 for Cygmon.
 - b. At the Countdown value prompt, press Enter to accept the default.
 - c. At the Disable initial display prompt, press Enter to accept the default.
 - d. At the Default IO type prompt, press Enter to accept the default.
 - e. At the SDRAM Window Size prompt, press Enter to accept the default.
 - f. At the SDRAM Window Offset prompt, press Enter to accept the default.
 - g. At the Upstream Window Size prompt, press Enter to accept the default.
 - h. Boot by typing `b`.

7. At the `cygmon` prompt, type `bo` for boot options. This allows you to set the static IP addresses for the IXP1200 network processor and for the x86 workstation:
 - a. Enter 2 to set the local IP address. Then enter the static IP address of the IXP1200 network processor. The local IP address must have the same subnet mask as the remote IP address.
 - b. Enter 3 to set the remote IP address. Then enter the static IP address for the Linux development workstation.
 - c. Enter 8 to enter a countdown value to auto-Linux boot. Enter 10 for the countdown value.
 - d. Type `s` to save this configuration in flash memory.
 - e. Type `gl` to boot Linux.

Starting the System

After you have completed the previous steps, boot Linux as follows:

1. Open a minicom window (if one is not already open) and press the Reset button on the IXP1200 network processor.
The remote shell displays the Linux login prompt.
2. Log in as `root`, using `ixp1200` as the password.
The remote shell displays the Linux command-line prompt.

3. To start the network, run the `ifup` script:

```
$ ifup <local_ipaddr> <subnet_mask> <broadcast> <gateway_ipaddr>
```

- `local_ipaddr` is the static IP address of the IXP1200 network processor
- `subnet_mask` is the subnet mask
- `broadcast` is the broadcast address
- `gateway_ipaddr` is the gateway IP address

Every time you boot, run the `ifup` script to set the addresses. You can use a minicom script to do this automatically; see “Using a Minicom Script for Startup,” below.

Using a Minicom Script for Startup

You can take advantage of minicom’s scripting capabilities to automatically boot the IXP1200 network processor when you open a remote minicom shell on the development workstation.

To set up such a script:

1. In a Linux command shell on the development workstation, copy the following minicom scripts from `$IXROOT/bin` into the `/minicom` directory:
 - `bootixp`
 - `start`

2. Modify the `bootixp` script to provide the correct IP addresses and domain names for your system configuration. Make sure all variables are set correctly.

The `bootixp` script starts `minicom` using the `start` script.

The `start` script boots and logs into Linux, sets up networking by running `ifup` with the same addresses you provided in `bootixp`, and mounts the NFS directory.

3. Set permissions for the `bootixp` script:

```
$ cd /minicom
$ chmod 777 bootixp
```

To boot the IXP1200 network processor and start a `minicom` shell with the correct configuration:

1. Press the Reset button on the IXP1200 network processor.
2. In a Linux command shell, run the `bootixp` script by typing `./bootixp`

You can run a script at any time within `minicom` by typing `CTRL A G`. The `minicom` scripts automatically provide the required inputs and give you a shell prompt.

If the system hangs, you can kill the script by typing `CTRL C` in the `minicom` window.

Setting Up an NFS Client

If you want to use NFS for file sharing, you must set up your development workstation as an NFS server and the IXP1200 network processor as an NFS client. For information on how to set up your x86 workstation as an NFS server, see “Setting Up an NFS Server” on page 31.

The IXP1200 Linux kernel and RAMdisk image are NFS-ready. To set up the IXP1200 network processor as an NFS client:

1. Log in as `root` in a remote command shell for the IXP1200 network processor.
2. Start the `portmap` daemon, if necessary. On the current RAMdisk image the `portmap` daemon is automatically started when you run the `ifup` script.
3. Mount the remote directory on the host file system:

```
mount -t nfs <server-ip-or-name>:/opt/ixasdk/bin/arm-be /nfs
```

- ✓ **NOTE:** There are many versions of NFS server. The daemons are also named differently. For the IXP1200 as NFS client, use only the NFS RPM package `nfs-utils-0.3.1-13.i386.rpm` for the NFS server.



Configuring the IXP1200 Microengine Development Environment

You use the IXP1200 Microengine Development Environment to develop, debug, and deploy microcode specifically targeted for microengines on the IXP1200.

To use the IXP1200 Microengine Development Environment:

1. Install the IXP1200 Microengine Development Environment on your Linux development workstation according to the instructions in “Installing the IXP1200 Microengine Development Environment” on page 25.
2. If you are not running DNS on the IXP1200, add the name of the Windows NT host to the `/etc/hosts` file on the IXP1200. To do this, be sure that you are logged into the embedded Linux system through a minicom window, and then enter the following commands:

- a. Use the `cat` utility:

```
$ cat > /etc/hosts
```

- b. Type:

```
<IP Address of NT host> <name_of_NT_host>
```

You can find the `<name of NT_host>` by looking at:

```
Network Neighborhood->Properties->Protocols->TCP/IP->DNS
```

Be sure that this name is the same as the machine name in Network ID.

- c. Type `CTRL D`.

If are using the Intel IXA SDK ACE Programming Framework, then use the IXP1200 Microengine Development Environment to connect to your Linux target (the IXP1200 network processor). For information on running a sample application, see the *IXA SDK Tutorial*.

If you are not using the Intel IXA SDK ACE Programming Framework, then to debug a microengine C application, type:

```
insmod ue.o; rs_udebug &
```

This starts the microcode debugger daemon on the IXP1200 network processor.

Removing the IXA SDK

Removing the IXP1200 Microengine Development Environment

To remove the IXP1200 Microengine Development Environment from the Windows NT single-board computer, select Add/Remove Programs from the Windows NT Control Panel and remove Intel IXP1200.

To remove Cygwin, delete the `eLinuxIDE-IXP1200\cygwin` directory from your system.

Removing the IXA SDK from Linux

To remove the IXA SDK software, use the RPM tool with the `-e` option to remove the following packages:

```
rpm -e ixa.sdk-2.01.D-fcs
rpm -e ixa.executive-2.01.D-fcs
rpm -e ixa.armbe-v4b-fcs
rpm -e ixa.binaries-2.01.D-fcs
```







Intel Corporation
2200 Mission College Blvd.
PO Box 58119
Santa Clara, CA 95052-8119 USA
Tel: 800.628.8686
www.intel.com

Revision 2.01, December 2001