MPS Software and Hardware

Installation Manual

205 Indigo Creek Drive | Rochester, NY 14626 | Phone +1.585.256.0200 | support@pt.com

www.pt.com

Document Revision History

Part Number	Date	Explanation of Changes
106p0856.72	08/21/2007	Changes made to the `commstats` command. Name of manual changed from **Software and Hardware Installation Manual** to **MPS Software and Hardware Installation Manual**.
106p0856.73	06/13/2008	Reformatted manual.
106p0856.74	03/27/2009	Removed reference to the *wcpConfig* utility.
106p0856.75	01/14/2011	Rebranded and reformatted. Added -V option to "System Statistics Monitor Program (commstats)" on page 63.

Compliance with RoHS and WEEE Directives

In February 2003, the European Union issued *Directive 2002/95/EC* regarding the Restriction of the use of certain Hazardous Substances in electrical and electronic equipment (RoHS) and *Directive 2002/96/EC* on Waste Electrical and Electronic Equipment (WEEE). This product is compliant with *Directive 2002/95/EC*. It may also fall under the *Directive 2002/96/EC*.

PT's complete position statements on the RoHS and WEEE Directives can be viewed on the Web at: http://pt.com/assets/lib/files/pdfs/rohs/policies/rohs-weee-statement.pdf.

Contents

Chapter 1: About This Guide 7

Overview 7

Text Conventions 7

Customer Support and Services 8

Customer Support Packages 8

Other Web Support 8

Return Merchandise Authorization (RMA) 8

Product Warranty 9

Chapter 2: Software Installation 11

Overview 11

Installation Instructions for UNIX Systems 12

Installation Instructions for Solaris 14

Installation Instructions for Windows 16

Windows LAN-Based Software Installation 16

Windows Embedded Based Software Installation 17

Instructions for Linux Systems 18

Installing the PT Client Software 18

Protokit Software Installation 21

GNU Tools Software Installation 22

Source-Level Debugger Installation 23

mps Configuration File 25

Solaris and Linux Operating Systems 25

Windows Operating System 25

General Usage Suggestion 26

Chapter 3: Network Configuration 27

Overview 27
Embedded NexusWare Controller 27
MPS1000 28
MPS800, MPS600 and MPS300 28
Configuring a Client 28
- Trivial File Transfer Protocol for UNIX Systems 28
- Trivial File Transfer Protocol for Windows 29
- Bootstrap Protocol 30

Configuring the Server 32
- Client’s Internet Address 34
- Server’s Internet Address 34
- Server's Line Electrical Interface 34
- Server’s Ethernet Address 34
- Server’s Logical Name 34
- Server’s (Listen) Port Numbers 35
- Download Command File Pathname 35
- Diagnostic Configuration 35
- Routing Configuration 35
- User-Defined Flash Prom 36
- Login Password 37
- Remote Configuration 37
- SNTP Configuration 40
- Write New Flash Program 40
- Enter Debugger 41
- Exit and Start Download 41
- Exit, Download Then Enter PQbug 41

Chapter 4: QMC Configuration Guide 43

Overview 43
QMC Configuration File Format 43

T1 Configuration Parameters 44
E1 Configuration Parameters 48
Running the QMC Configuration Utility 52

Chapter 5: Client Utilities 55

Overview 55
Download Program (commload) 55
Description 55
Options 55
Communications Server Update Program (commupdate) 57
Modifying the Buffer Pool Configuration 58
Update the Memory Size 60
Modifying the MPS Server-Specific Options 62
System Statistics Monitor Program (commstats) 63
Synopsis 63
Description 63
Options 63
Interpreting the System Statistics Report of a Non-NexusWare Platform 64
Interpreting the System Statistics Report of a NexusWare Platform 67
LAN Server Reset Utility (commreset) 67
Synopsis 68
Description 68
Options 68

Appendix A: Network Load Procedures 69

Overview 69
The Load Procedure 69
MPS 3000-Family Front-Panel Display 70
MPS 300/600 Front-Panel Display 71
Using the Console Port to Diagnose a Problem 71
SNMP Implementation on an MPS Server 73
MPSsystem 73

MPSip 73

Chapter 1

About This Guide

Overview

This document describes software and hardware installation procedures for PT WAN access products. PT software runs on intelligent serial communication controllers installed in either a UNIX or Windows host system, or on PT LAN-based communication servers. Some sections of this installation guide apply only to certain host, controller or server types and are not applicable to others.

The contents of each chapter in this manual are described below.

Chapter 2, "Software Installation," on page 11 covers basic software installation procedures for UNIX, Solaris, Windows, and Linux-based systems.

Chapter 3, "Network Configuration," on page 27 covers configuration of PT LAN-based servers.

Chapter 4, "QMC Configuration Guide," on page 43 covers configuration of PCI37x T1/E1/J1 Communications Controllers.

Chapter 5, "Client Utilities," on page 55 covers operation of client utilities that are used to monitor and maintain PT communications software.

Appendix A, "Network Load Procedures" on page 69 describes loading runtime software and server loading indicators.

Text Conventions

This guide uses the following conventions:

Convention	What it is Used For
`Monospace font`	Monospace font is used to represent sample code.
Italic font	Bold font is used to represent: • pathnames • filenames • UNIX commands • user input
Bold font	Italic font is used to represent: • notes that supply useful advice • general information • referenced documents
< >	Angle brackets are used to represent variables such as file names, passwords, and so on.
Regular font	Regular font is used for the ENTER and TAB keys and the SPACEBAR on your keyboard.

Customer Support and Services

PT offers a variety of standard and custom support packages to ensure customers have access to the critical resources that they need to protect and maximize hardware and software investments throughout the development, integration, and deployment phases of the product life cycle.

If you encounter difficulty in using this PT product, you may contact our support personnel by:

1. **EMAIL** (Preferred Method) – Email us at the addresses listed below or use our online email support form. Outline your problem in detail. Please include your return email address and a telephone number.
2. **TELEPHONE** – Contact us via telephone at the number listed below, and request Technical Support. Our offices are open Monday to Friday, 8:00 a.m. to 8:00 p.m. (Eastern Time).

PT Support Contact Information

	Embedded Systems and Software (Includes Platforms, Blades, and Servers)	SS7 Systems (Includes SEGway™)
Email	support@pt.com	ss7support@pt.com
Phone	+1 (585) 256-0248 (Monday to Friday, 8 a.m. to 8 p.m. Eastern Time)	+1 (585) 256-0248 (Monday to Friday, 8 a.m. to 8 p.m. Eastern Time)

If you are located outside North America, we encourage you to contact the local PT distributor or agent for support. Many of our distributors or agents maintain technical support staffs.

Customer Support Packages

Our configurable development and integration support packages help customers maximize engineering results and achieve time-to-market goals. To find out more about our Customer Support packages, visit http://www.pt.com/page/support/.

Other Web Support

Support for existing products including manuals, release notes, and drivers can be found on specific product pages at http://www.pt.com. Use the product search to locate the information you need.

Return Merchandise Authorization (RMA)

To submit a return merchandise authorization (RMA) request, complete the online RMA form available at http://pt.com/assets/lib/files/rma-request-form.doc and follow the instructions on the form. You will be notified with an RMA number once your return request is approved. Shipping information for returning the unit to PT will be provided once the RMA is issued.

Product Warranty

Performance Technologies, Incorporated, warrants that its products sold hereunder will at the time of shipment be free from defects in material and workmanship and will conform to Performance Technologies' applicable specifications or, if appropriate, to Buyer's specifications accepted by Performance Technologies in writing. If products sold hereunder are not as warranted, Performance Technologies shall, at its option, refund the purchase price, repair, or replace the product provided proof of purchase and written notice of nonconformance are received by Performance Technologies within 12 months of shipment, or in the case of software and integrated circuits within ninety (90) days of shipment and provided said nonconforming products are returned F.O.B. to Performance Technologies's facility no later than thirty days after the warranty period expires. Products returned under warranty claims must be accompanied by an approved Return Material Authorization number issued by Performance Technologies and a statement of the reason for the return. Please contact Performance Technologies, or its agent, with the product serial number to obtain an RMA number. If Performance Technologies determines that the products are not defective, Buyer shall pay Performance Technologies all costs of handling and transportation. This warranty shall not apply to any products Performance Technologies determines to have been subject to testing for other than specified electrical characteristics or to operating and/or environmental conditions in excess of the maximum values established in applicable specifications, or have been subject to mishandling, misuse, static discharge, neglect, improper testing, repair, alteration, parts removal, damage, assembly or processing that alters the physical or electrical properties. This warranty excludes all cost of shipping, customs clearance and related charges outside the United States. Products containing batteries are warranted as above excluding batteries.

THIS WARRANTY IS IN LIEU OF ALL OTHER WARRANTIES WHETHER EXPRESS, IMPLIED OR STATUTORY INCLUDING IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS. IN NO EVENT SHALL PERFORMANCE TECHNOLOGIES BE LIABLE FOR ANY INCIDENTAL OR CONSEQUENTIAL DAMAGES DUE TO BREACH OF THIS WARRANTY OR ANY OTHER OBLIGATION UNDER THIS ORDER OR CONTRACT.

Chapter 2

Software Installation

Overview

This chapter contains basic software installation instructions for UNIX and Windows host systems. Choose the appropriate section for your host system and follow the instructions in that section:

Before proceeding with the software installation, you should install your hardware. For installation of a LAN-based server, or a NexusWare controller that will use TCP/IP sockets to communicate with its host processor, refer to "Network Configuration," on page 27*.*

For installation of embedded communications controllers that will communicate over the PCI bus with its host processor, refer to the hardware *User's Guide* your received from PT.

The following topics are covered in this chapter:

- "Installation Instructions for UNIX Systems," on page 12
- "Installation Instructions for Solaris," on page 14
- "Installation Instructions for Windows," on page 16
- "Instructions for Linux Systems," on page 18
- "Installing the PT Client Software," on page 18
- "Protokit Software Installation," on page 21
- "mps Configuration File," on page 25

If your Host system has Solaris, Linux or Windows as the operating system, and the embedded mps driver will be used for controller access over the PCI or cPCI bus, be sure to read "mps Configuration File," on page 25 regarding the mps configuration file.

If you are installing the Protokit software and the GNU tools refer to the instructions listed under "Protokit Software Installation," on page 21.

The software is shipped on CDROM. The media contains both the host-resident and controller-resident portions (when applicable) of the software package. Specifically, this includes:

- A load command file and a set of executable files for the controller or server (not applicable for a NexusWare Controller). These files contain the system-level code and the protocol application. They are transferred to the controller or server during the download procedure.
- Source files containing a host device driver for the communication controller (not included for LAN-based servers). The Windows host device driver is pre-compiled.
- Source files for the host-resident download, configuration and test utilities. Some products also include a background daemon process.
- Source files for the PT Application Programming Interface (API). This is a library of subroutines to be linked with host applications that communicate with the protocol software. They provide a simple read/write interface to the controller's host driver or, for LAN-based servers, to TCP/IP.

Installation Instructions for UNIX Systems

The software installation procedure creates a main directory called *pti* under your current directory, and a number of subdirectories under *pti*. The default location for the *pti* directory structure is in */usr*. These installation instructions as well as other software documentation assume that the distribution is based at the directory */usr/pti.* If you choose to install the software in another location, remember to substitute your base directory for */usr/pti* when following instructions in this and other documents.

1. Login as root.

   Make sure that */usr/ccs/bin* is included in root's SUPATH environment variable definition in */etc/default/su*. This is necessary to enable the operating system to find the *make* utility.

2. Verify that the complete pathname to your compiler is specified for the CC environment variable in the file */usr/share/lib/make/make.rules* before executing any of the following installation procedures. For example, if your compiler is located in */opt/SUNWspro/bin/cc*

   ```
   CC = /opt/SUNWspro/bin/cc
   ```

3. Insert the CDROM into the drive and type:

   ```
   cd /cdrom/cdrom0
   ./INSTALL
   ```

   (Note that the CDROM device might have a different name on your system.)

   The CDROM install script will ask you to:

   (a) Select your board/server type from the list of board/server types provided.

   Note: *If you are installing software for a LAN server, you will be asked to select the installation package format. The choices provided are (1) pkgadd format, or (2) unix tar format. You must select choice (2) for non-Solaris systems.*

   (b) Enter the installation directory.

4. When you run the install script, you will be asked to enter your server type(s), and operating system type. This information is used to build the PT Application Programming Interface (API).

   The choices provided for server type are (1) LAN server, (2) embedded server, or (3) both. The API will build correctly for a LAN server only if your host system includes a TCP/IP package with a standard socket interface. (This is not a problem on most UNIX systems.)

   The choices provided for operating system type are:

   1. Solaris 2.x
   2. SunOS (Solaris 1.x)
   3. HP-UX for HP9000 Series 700
   4. HP-UX for HP9000 Series 800
   5. DG/UX
   6. IRIX (V4)
   7. IRIX (V5 or later)
   8. DEC-UX

9. QNX

10.Other

If you select 'other', the script prompts you for additional information in order to determine your operating system's UNIX compatibility:

The choices provided for UNIX version are (1) UNIX System V or (2) SunOS/BSD. This determines whether the *makefile* runs the *ranlib* utility on the API library. This utility is not supported on UNIX System V. If you are not sure of the correct choice for your system, check your manual pages to see if *ranlib* is supported. If so, select '2'; if not, select '1'.

The choices provided for system interface definition are (1) SVR4 or (2) SVR3. This information affects the compilation of the *MPSpoll* API routine, which uses the *poll* system call. If your system complies with SVR4, any system file descriptor can be supplied to the *poll* system call. Under SVR3, *poll* is only supported for STREAMS file descriptors. If you are not sure of the correct choice for your system, read your manual pages for the *poll* system call to determine what types of file descriptors are supported. If any system file descriptor can be used, select '1'; if only STREAMS file descriptors can be used, select '2'.

The installation script builds the portions of the software distribution that execute on the local system, with the exception of the host driver (if included). These include the API library and the download, configuration and test utilities. Except for the API archive, *libMPS.a*, which is copied to */usr/lib*, the installation script and the Makefiles it executes do not modify any files or directories outside */usr/pti*.

5. Skip this step if, either you installed the software in the default */usr* directory or, you are installing for a NexusWare (CPC388) product.

   The load command file in the main *pti* directory is named *Cmdfile.prot*, where *prot* is a filename extension identifying a specific software product or configuration. The load command file contains the names of the executable files that are loaded to the communication controller or server. This file is used during the load procedure, and must contain the correct full pathname of each file. If you did not install the software in the default directory, the pathnames in this file are incorrect. The easiest solution is to create a symbolic link to the */usr/pti* directory. In the following example, a link is created from the alternate installation directory */home/tools*:

```
ln -s /home/tools/pti /usr/pti
```

   If, for some reason, this solution is not acceptable, you must edit the command file and modify the pathname of each executable file. Below, a sample file is shown with the default pathnames and then after modification to accommodate the alternate installation directory
/home/tools. (Note that the details of your file may not match the examples shown here.)

```
BTYPE 0 PTI334
LOADCTLR 0 /usr/pti/Load/xstra.pci334.hdlc.0 L_0 0x405e00 0x405e00
LOADCTLR 0 /usr/pti/Load/hif.pci334.hdlc.0 L_1 0x40fe00 0x0
LOADCTLR 0 /usr/pti/Load/hdlcswap.mod.hdlc.0 L_2 0x413e00 0x0
LOADCTLR 0 /usr/pti/Load/hdlc.pci334.hdlc.0 L_3 0x414400 0x0
LOADCTLR 0 /usr/ptiX/Load/config.hdlc.0 L_4 0x402000 0x0
INIT L_0

BTYPE 0 PTI334
LOADCTLR 0 /home/tools/pti/Load/xstra.pci334.hdlc.0 L_0 0x405e00
    0x405e00
LOADCTLR 0 /home/tools/pti/Load/hdlcswap.mod.hdlc.0 L_2 0x413e00 0x0
LOADCTLR 0 /home/tools/pti/Load/hdlc.pci334.hdlc.0 L_3 0x414400 0x0
LOADCTLR 0 /home/tools/pti/Load/config.hdlc.0 L_4 0x402000 0x0
INIT L_0
```

Installation Instructions for Solaris

When you run the install script, it creates a main directory called *pti* and a number of subdirectories. The default location for the *pti* directory structure is in */opt*, with a symbolic link from the directory */opt/pti* to */usr/pti*. These installation instructions and other software documentation assume that the software is based at the symbolic directory path */usr/pti*.

Before performing the software installation, you should install your communication controller hardware. Then boot your system using the *-r* option so the new hardware is identified.

1. To install the software distribution, first login as root.
2. Make sure that */usr/ccs/bin* is included in root’s SUPATH environment variable definition in */etc/default/su*. This is necessary to enable the operating system to find the *make* utility.

   Verify that the complete pathname to your compiler is specified for the CC environment variable in the file */usr/share/lib/make/make.rules* before executing any of the following installation procedures. For example, if your compiler is located in */opt/SUNWspro/bin/cc:*

   ```
   CC = /opt/SUNWspro/bin/cc
   ```

3. Insert the media into the drive and inform the volume manager that you have done so.
4. Change directory to the appropriate device, and run the install script. Enter the following commands:

   ```
   cd /cdrom/cdrom0
   ./INSTALL
   ```

   The CDROM install script will ask you to:

   (a) Select your board/server type from the list of board/server types provided.

   Note: *If you are installing software for a LAN server, you will be asked to select the installation package format. The choices provided are (1) pkgadd format, or (2) unix tar format. It is recommended to select choice (1) if you are installing our software products on a Solaris system. The second option (unix tar format) is for installing products on the UNIX systems that do not support the Solaris pkgadd utility.*

   If you selected *pkgadd* format, proceed to Step 5.

   If you selected *tar package* format, the install script asks for the complete pathname of the installation directory. The script first copies the tar file to this location and then extracts it. Proceed to “Installation Instructions for UNIX Systems,” on page 12, Step 3.

5. The install script asks you to:
   - Enter the complete pathname of a temporary directory to be created for the installation. The install script automatically removes this directory when the installation is complete. You must specify the full path of a directory that does not already exist.
   - Choose a base directory path for the installation. The default location is */opt*.

- Authorize the creation of a symbolic link to the installation directory, and enter the directory path for the symbolic link. The default path is */usr/pti*. These installation instructions and other software documentation assume that the distribution is based at the directory */usr/pti*. If you choose to install the software in another location, remember to substitute your base directory for */usr/pti* when following instructions in this and other documents.
- Select your server type. The choices provided are (1) LAN server, (2) embedded server, or (3) both. The API will build correctly for a LAN server only if your host system includes a TCP/IP package with a standard socket interface. (This is not a problem on most UNIX systems.)

The install script builds the portions of the product that execute on the local system. These include the API and the download, configuration and test utilities. The API archive, *libMPS.a*, is copied to */usr/lib*. The device driver object file is copied to */usr/kernel/drv* and is loaded into the kernel.

6. Skip this step if either you accepted, in Step 5, the default symbolic link from the installation directory to */usr/pti*, or you are installing for a NexusWare (CPC388).

   The load command file in the main *pti* directory contains the names of the executable files that are loaded to the communication controller or server. This file is used during the load procedure, and must contain the correct full pathname of each executable file. If you did not allow the install script to create the default symbolic link, the pathnames in the load command file is incorrect. In this case, you must edit the command file and modify the pathname of each executable file. The command file is called *Cmdfile.prot*, where *prot* is a filename extension identifying your specific protocol or configuration.

   Below, a sample file is shown with the default pathnames and then after modification to accommodate the alternate installation directory */home/tools*. (Note that the details of your file will not exactly match the examples shown here.)

```
BTYPE 0 PTI334
LOADCTLR 0 /usr/pti/Load/xstra.pci334.hdlc.0 L_0 0x405e00 0x405e00
LOADCTLR 0 /usr/pti/Load/hif.pci334.hdlc.0 L_1 0x40fe00 0x0
LOADCTLR 0 /usr/pti/Load/hdlcswap.mod.hdlc.0 L_2 0x413e00 0x0
LOADCTLR 0 /usr/pti/Load/hdlc.pci334.hdlc.0 L_3 0x414400 0x0
LOADCTLR 0 /usr/ptiX/Load/config.hdlc.0 L_4 0x402000 0x0
INIT L_0

BTYPE 0 PTI334
LOADCTLR 0 /home/tools/pti/Load/xstra.pci334.hdlc.0 L_0 0x405e00
    0x405e00
LOADCTLR 0 /home/tools/pti/Load/hif.pci334.hdlc.0 L_1 0x40fe00 0x0
LOADCTLR 0 /home/tools/pti/Load/hdlcswap.mod.hdlc.0 L_2 0x413e00 0x0
LOADCTLR 0 /home/tools/pti/Load/hdlc.pci334.hdlc.0 L_3 0x414400 0x0
LOADCTLR 0 /home/tools/pti/Load/config.hdlc.0 L_4 0x402000 0x0
INIT L_0
```

7. Loading the Device Driver

   The Solaris host device driver is loaded during the installation procedure described in Step 5 above. Thereafter, the driver is loaded automatically whenever you reboot the system. In general, you never need to manually load or unload the device driver, but the commands to do so are shown below:

   To reload the driver for SBus, PCI, CMC, or PMC communications controllers, type:

```
cd /usr/pti/drivers/pt33x
make
```

Installation Instructions for Windows

This section discusses the installation and setup of PT software on a Windows machine. The installation process for the LAN based software differs slightly from the installation processes for the embedded based software. For both installations, the following apply:

- The PT software uses the industry standard installation software InstallShield. Administrative permission is required to run the InstallShield setup application.
- There are project files which are used to build the API and each of the client applications. They were created with the Microsoft Visual Studio toolkit. If you want to use these project files, please use this toolkit. The older C/C++ compiler build projects are still supported and provided, however they do not contain all of new functionality (IPv6) and they will not be maintained going forward.
- A client application should be linked with the PT MPS Application Programmer's Interface (API). It is called *libMPS.lib*, and it has the functionality for both the embedded controller and the LAN server. There is a project file which is used to build this library. The library should be copied to a place where the client's projects can find and link to it. By default, it is copied to *%PTI%\lib*.
- In the client directory of each protocol, there is a project file for each sample test application (which are discussed in the each protocol's programmer's guide). For example, the *%PTI%\sbsi\client* directory has two workspaces defined; *sbsisnd.dsw* and *sbsircv.dsw*. When these programs are built, "WINNT" and "ANSI_C" must be defined at compile time, and *wsoc32.lib* and *libMPS.lib* must be linked to the client application. Furthermore, since by default the API is built MT-Safe, the client application must be as well; thus "/MT" or "/MTd" must be specified on the compiler command line. See the file "README.THREADS" for building the API and client applications thread-safe. Also make sure the client application is built the same way as the API in terms of either Debug or Release.

Windows LAN-Based Software Installation

This section discusses the issues pertinent only to a Windows software installation for LAN-based PT communications servers. The software installation is started by double-clicking on the *Setup.exe* application that is on the provided CDROM.

You will see the PT welcome screen displayed, click **next**. The software licence agreement will then be displayed, read the terms and click **yes** to accept the agreement. You will then be asked to choose a destination folder for the installation. The default directory is *c:\pti*. The directory that you specify for the software to be installed will become your machine's *pti* directory and will have an environment variable called PTI to refer to it.

You must then select the hardware platform to install. For a LAN installation, choose an MPS server. Type in the name of the MPS server that the software will connect to. Next, type in the IP address of that server in dot notation. Both the name of the server and the IP address will then be added to the system's host table. If the host table already has an entry for that server name, the installation will continue without writing a new entry for that server. If you don't know the name of the server to connect to, just click next for the server name and IP address. You can add that information later to *%SystemRoot%\system32\drivers\etc\hosts*.

Finally, the installation asks if you want to reboot your computer. You must reboot your computer for the software installation to complete. Remember to reboot before you try to use the PT software.

If you are using a LAN-based PT server, (this does not include a NexusWare server) a TFTP service must be defined. The TFTP server is not standard for the Windows operating system. PT uses TFTPServer by SolarWinds (http://solarwinds.net). The command file (*Cmdfile.prot*) that is downloaded to the server via TFTP is of type binary. If you edit the file, make sure line feeds do not get attached by the editor.

The client applications communicate with the PT LAN server through TCP/IP. The installation process adds the *mps* service to the *services* file (*%SystemRoot%\system32\drivers\etc\services*). The PT LAN server IP address must be defined in the *host* table (*%SystemRoot%\system32\drivers\etc\hosts*) if the server wasn't defined during installation or if the server IP changes after the installation.

Windows Embedded Based Software Installation

This section discusses the issues pertinent only to a Windows software installation for embedded PT communications controllers. The architecture platform for the Windows system must be Intel x86 or AMD.

Note: *If you have already physically installed the embedded board(s) on your system before installing the PT Software, a Found New Hardware/Files Needed Window will appear on your Windows desktop. Disregard this window until you have installed the PT Software and reboot your system.*

The software installation is started by double-clicking on the *Setup.exe* application that is on the provided CDROM. You will see the PT welcome screen displayed, click **next**. The software licence agreement will then be displayed, read the terms and click **yes** to accept the agreement. You will then be asked to choose a destination folder for the installation. The default directory is *c:\pti*. The directory that you specify for the software to be installed will become your machine's *pti* directory and will have an environment variable called PTI to refer to it.

You must then select the hardware platform to install. Choose the embedded communications controllers to install. Finally, the installation asks if you want to reboot your computer. You must reboot your computer for the software installation to complete. Remember to reboot before you try to use the PT software.

Note: *After rebooting your system, there will be a Found New Hardware/Files Needed window on your desktop. From this window, for Windows 2000 users, specify c:\PTI\DRIVERS\WIN_2K. For all other Windows types, specify c:\PTI\DRIVERS|OTHER."*

Note: *The Hardware Installation Wizard will present the following warning: "The software you are installing... has not passed Windows Logo testing..." Select the* ***Continue Anyway*** *option.*

The host machine communicates with the communications controller through the device driver, *ntmps.sys*. During the installation, the *ntmps* device driver is set up to load automatically at machine startup. You can change this setting, or load and unload the driver manually through the **Control Panels Devices** applet.

There will be a new registry entry (ntmps) created in the HKEY_LOCAL_MACHINE / SYSTEM CurrentControlSet / Services area. Also, the event log service will be modified to enable it to log messages from the ntmps device driver. Remember to reboot the machine after the device driver is installed manually.

For the embedded Windows driver, the application that loads the card (*commload*) has to be run a little bit differently. The controller to be loaded must be specified on the command line. For instance, even if just one controller is resent in the Windows machine, the command line for the *commload* should be:

```
commload -v -c 0 Cmdfile.prot
```

(the actual name of the command file will vary depending on the protocol software). If the controller number is not specified, the *commload* program will fail. Note that the loading of modules is not applicable for a NexusWare Controller (CPC388, CPC358, and PCI384).

For Windows systems that support Power Management, the embedded device driver (ntmps) will allow both Standby and Hibernate requests if there are no open connections to a PTI communications controller. If a connection is open, the request will be rejected, otherwise data coming from the controller will be lost.

For Standby mode, since the power to the PCI bus is not affected, the ntmps driver will view each controller after the system has come out of Standby as it did before. For Hibernate mode, since the system turns off the power to the PCI bus, the ntmps driver will treat each controller as if it has just been reset.

Instructions for Linux Systems

This section describes the procedure for installing the PT software on a Linux system. The PT software is capable of being used in an embedded controller environment, a LAN network environment, or both.

If you are running a LAN-based application *only*, you may skip to the next section entitled installing the PT Client Software.

If you intend to use the PT software for embedded applications, there are certain system requirements that must be met.

The Linux Kernel must be Version 2.4 or higher. If your kernel is *prior* to Version 2.4 Linux STREAMS (LiS) is required. Please contact support@pt.com for assistance.

If your Linux Kernel Version 2.4 system already contains LiS, you may wish to select the STREAMS based. This may be the case if you have a previous installation of the PT product.

If your Linux Kernel is Version 2.6, or Version 2.4 without the LiS package installed, then you will want to select the non-STREAMS driver.

Installing the PT Client Software

The PT installation procedure will create a directory tree on your system whose top node is named *pti*. The default location in the file system where this directory tree is created is the */usr* directory (i.e. */usr/pti*). These installation instructions and other PT documents assume that the distribution is based at */usr/pti*. If you choose to install the software in another location, you must substitute the alternate directory path for */usr/pti* when following instructions in this document and others. For example, if you choose to install the software in the */var/local* directory, then you must substitute */var/local/pti* for */usr/pti* in the instructions below.

The following procedure describes how to install the PT NexusWare client software onto your Linux host.

***Note:** You should be logged in as root when performing the indicated commands.*

1. Insert the distribution CD into the CDROM drive of your computer. Change the current working directory to the appropriate device directory and run the installation script:

   ```
   root# cd /mnt/cdrom
   root# ./INSTALL
   [...]
   root#
   ```

   ***Note:** The CDROM device might have a different name on your system.*

   The installation script will ask you to:

   (a) Select your target operating system.

   (b) Enter the installation directory (e.g. */usr*).

   (c) Authorize the creation of a symbolic link to the installation directory, and enter the directory path for the symbolic link. The default path is */usr/pti*.

   (d) Select your server type from the list of controller types provided.

   (e) Select the STREAMS, or non-STREAMS based driver.

2. The installation script prompts you for the operating system type for the product is being built. This information is used to build the PT Application Programming Interface (API).

   ```
   The operating system choices for NexusWare controllers are:
   1. Solaris 2.x
   2. Linux
   3. UNIX (other)
   -> 2
   ```

   You should select the option for Linux.

3. Enter the installation directory at the prompt

   ```
   Enter the installation directory for the P057570H.
   Installation dir > /opt/pti
   Creating /opt/pti
   Copying P057570H.tar.gz to /opt/pti/P057570H
   extracting package from tar file
   …
   ```

   If the installation directory does not exist it will be created and a directory below it where the client files from the distribution media extracted.

4. Authorize the creation of a logical link.

   A symbolic link to the */opt/pti/P057570H/pti* directory will be created as */usr/pti*. The PTI software will look for its files using this symbolic link. You may chose a different location for the symbolic link, or none at all if desired.

```
To select the default of yes, press the ENTER key
[y] [y,n,?,q] <ENTER>
To select the default, press the ENTER key
[/usr/pti] [?,q] <ENTER>
```

5. Select the communications method with the server; 1) Network will a select TCP/IP socket interface. 2) Embedded will select an driver for communication over the PCI bus. 3) Both will allow either communications to be used.

   ```
   What type(s) of PTI Servers will you be connecting to?
   1. CPC388/CPC358/CPC308/CPC324 -Network
   2. PCI384/CPC388/CPC358/CPC308/CPC324 -Embedded
   3. Both
   ```

6. Select the driver that will be used for embedded communication.

   ```
   What embedded driver would you like to install?
    1. STREAMS driver
    2. non-STREAMS driver
   -> 2
   ```

   The installation script builds the portions of the software distribution that execute on the local host system only. These portions include the API library, host driver (if applicable), and the download, configuration, and test utilities. For the most part, the installation script and the make files it executes do not modify any files or directories outside of the */usr/pti* tree. The exceptions are the API library (*libMPS.a*) that is copied into */usr/lib*, the host driver objects that are copied into the LiS directory tree (if applicable), and the object modules produced by rebuilding LiS with the PT host driver included in the package (if applicable).

   Note: *(for embedded STREAMS driver applications only): If kerneld is enabled on your system, the LiS package will automatically be loaded into kernel memory when you open a file descriptor to an embedded controller. LiS will also be automatically unloaded from kernel memory when it has not been used for awhile. If kerneld is disabled, however, you must manually load the streams module into kernel memory prior to opening a file descriptor to an embedded controller, and you should unload it when it is no longer in use. The 'insmod' and 'rmmod' commands are used to load and unload a kernel module, respectively; 'lsmod' displays a list of all modules currently loaded in the kernel.*

7. Skip this step if you are installing the client software for use with a NexusWare Controller. You may also skip this step if your xSTRa based software in the default */usr* directory.

   The load command file is */usr/pti/Cmdfile.prot*, where “prot” is a filename extension identifying a specific software product or configuration. The load command file contains the names of the executable files that are loaded onto the communication controller or server. This file is used during the download procedure, and must contain the correct full pathname of each file. Since you did not install the PT software in the default directory, the pathnames in this file are incorrect and must be updated.

   The easiest solution is to create a symbolic link named */usr/pti* that points to the alternate installation directory. For purposes of example, assume that the alternate installation directory for the PT software is */opt/tools*.

   ```
   root# ln -s /opt/tools/pti /usr/pti
   root#
   ```

If for some reason this solution is not acceptable, you must edit *Cmdfile.prot* and modify the pathnames of the files. Below, a sample load command file is shown with the default pathnames, and then after modification to accommodate the alternate installation directory (*/opt/tools*). Note that the details of your command file will not exactly match the example shown here.

```
Before:
BTYPE 0 PCI334
LOADCTLR 0 /usr/pti/Load/UC.pci334.dfx.0 L_0 0x802000 0x0
LOADCTLR 0 /usr/pti/Load/hdlc.pci334.dfx.0 L_1 0x806b30 0x0
LOADCTLR 0 /usr/pti/Load/xstra.pci334.dfx.0 L_2 0x80cb40 0x80cb40
LOADCTLR 0 /usr/pti/Load/config.dfx.0 L_3 0x80a320 0x0
INIT L_2
After:
BTYPE 0 PCI334
LOADCTLR 0 /opt/tools/pti/Load/UC.pci334.dfx.0 L_0 0x802000 0x0
LOADCTLR 0 /opt/tools/pti/Load/hdlc.pci334.dfx.0 L_1 0x806b30 0x0
LOADCTLR 0 /opt/tools/pti/Load/xstra.pci334.dfx.0 L_2 0x80cb40
    0x80cb40
LOADCTLR 0 /opt/tools/pti/Load/config.dfx.0 L_3 0x80a320 0x0
INIT L_2
```

Protokit Software Installation

This section contains basic Protokit software installation instructions for Solaris 2.x host systems. This section is not applicable for NexusWare products (CPC388).

ProtoKit is shipped on two CDROMs: one containing the ProtoKit itself and the other containing the GNU software development tools, which are used to build software that runs in the *xSTRa* environment on a PT server or controller.

The instructions below explain how to install Protokit and extract the GNU tools from the distribution media. Installation of the source-level debugger is described in “Installing the PT Client Software,” on page 18.

Follow the instructions in this section, in order, before proceeding with the installation procedures described in the remainder of this document, and before installing the source-level debugger (if applicable).

If you are using ProtoKit with an SBus or PCIbus communication controller (rather than a LAN-based server), you must install the hardware before performing the software installation. After installing the communication controller(s) as described in the User's Manual you received from PT, boot your system using the *-r* option so the new hardware is identified.

To install the Protokit software return to “Installation Instructions for Solaris,” on page 14.

GNU Tools Software Installation

In order to develop software to run in the *xSTRa* environment, you must use the software development tools supplied with Protokit. These are a slightly-modified version of the GNU tools from the Free Software Foundation. As the tools are originally distributed, they generate code for the machine type on which they are executed. PT has modified the programs so that they generate code for the Motorola 68K processor or PowerQUICC processor families regardless of the machine type on which they are installed. The GNU tools for Motorola 68K are used with PT servers and controllers based on that CPU. The GNU tools for Motorola PowerQUICC are used for all PT platforms that use the PowerQUICC chip. Therefore, you will need to install the tools supplied with Protokit even if you are already using GNU.

The GNU software is on the second CDROM labeled *GNU Tools*. The distribution includes complete source for all the utilities as well as pre-built executables for Sun SPARC or Intel processors using Solaris 2.6 or newer.

There are two procedures outlined below. If you are running on a Sun SPARC or Intel processor that has Solaris 2.6 or newer, use “Installing the Tools,” on page 22. If not, use the alternate procedure “Building and Installing the Tools,” on page 23. Your host must have a C compiler available to rebuild the tools.

Before running the *build/installation* script, you must add two directories to your PATH environment variable. By default, these are */opt/gnu68k-pti/bin* and */opt/gnu68k-pti/lib* for the Motorola 68K tools and */opt/gnuppc-pti/bin* and */opt/gnuppc-pti/lib* for the Motorola PowerQUICC tools. Once the tools have been built and/or installed, all users of the tools must also add these directories to their PATH. If you elect to build the native compiler, a default installation directory of */opt/gnu* is used, which must then also be added to your PATH.

Installing the Tools

There is no need to rebuild the tools for Sun SPARC or Intel processors running Solaris 2.6 or newer. An installation script is provided to copy the executable files for the utilities to an appropriate target installation directory. The script prompts you for the installation pathname, with a default based on the GNU target type. For instance, if you are installing the GNU tools for support of Motorola 68K platforms, the default directory is */opt/gnu68k-pti*. If you choose another pathname, the install script creates a symbolic link from */opt/gnu68k-pti* to your path.

Install the CDROM labelled *GNU Tools*. To run the GNU build/installation script, type:

```
cd /cdrom/cdrom0
INSTALL.gcc
```

The script will ask you:

```
Would you like to rebuild GNU tools:[y/n]:
```

Answer **n** to this question. Installation of the GNU tools requires between 10 to 15 Mbytes of free disk space, depending on which target you are installing. The script will ask you for the target type (Motorola 68k, PowerQUICC, or native compiler) and for the installation directory pathnames. During the installation procedure, the tools are copied to their runtime locations.

Building and Installing the Tools

The GNU tools must be built from the source code if one of the following apply: 1) your host is not a SPARC or Intel processor running Solaris 2.6 or newer, or 2) you are running an older version of Solaris. Run the installation script as described above; the script will ask you:

```
Would you like to rebuild GNU tools:[y/n]:
```

In this case, you answer **y**. The script will ask you for the target type (Motorola 68k, PowerQUICC, or native compiler) and for the installation directory pathnames. The script will print out status of the build as it progresses; this is a lengthy operation on most computers.

Note: *If you have elected to build the native compiler, build it again once it has completed the first build and installation procedure. Make sure that /opt/gnu (or the installation directory name you have chosen) is in your PATH the second time you build it. The GNU software will then rebuild the native compiler using the GNU native compiler that you have just built. This is the standard GNU recommended procedure.*

Building of the GNU tools requires anywhere from 50 to 100 Mbytes of additional free disk space for *each* compiler you build. During the installation procedure, the tools are built and copied to their run-time locations. The *gnutools* directory in *pti/util* can then be removed, freeing the majority of the disk space.

Source-Level Debugger Installation

This section describes how to build/install the PT source-level debuggers, **gdb68k** or **gdbppc**. The debugger is based on the Free Software Foundation's GDB, and works in conjunction with the GNU software development tools. You must complete the standard Protokit and GNU installations as described previously before installing the source-level debugger.

As the names imply, **gdb68k** and **xgdb68k** are used to debug PT Motorola 68k platforms, while **gdbppc** and **xgdbppc** are used to debug PT Motorola PowerQUICC platforms.

The instructions below assume that you installed Protokit in the default location, */usr/pti*. If you used an alternate base directory, be sure to make the appropriate pathname substitutions as you type the commands.

The Protokit CDROM labeled *GNU Tools* contains the GDB distribution. The source for GDB is contained in the subdirectory *gnutools/gdb-4.17*. Source for an X windows graphical user interface (GUI) for GDB is also included, and is located in the directory *gnutools/xxgdb-1.06*.

The distribution includes complete source for GDB. If you purchased Protokit for Solaris 2.X, the distribution also includes pre-built executables for Sun SPARC or Intel processors using Solaris 2.6 or newer.

There are two procedures outlined below. If you are running on a Sun SPARC or Intel processor that has Solaris 2.6 or newer, use “Installing GDB,” on page 24. If not, use the alternate procedure “Building and Installing GDB,” on page 24.

To build the X windows GUI programs **xgdb68k** and **xgdbppc**, you must have X11 Release 4 or later installed on your system. You must also set your path variable to include the directory where your X11 binaries are located, in order to use the *imake* utility that is invoked by the script.

If you are using the GNU tools rather than the standard SunPro tools to build executables for your host system, edit the file *Imakefile* in *pti/util/gnutools/xxgdb-1.06* before you begin the build procedure. Modify the line that reads *#CC=gcc -v*, removing the # symbol at the start of the line. However, the *imake* program may have produced a Makefile that contains switches not supported by the GNU compiler. If so, you will have to manually pull them out of the Makefile.

xgdb68k and **xgdbppc** use the **gdb68k** and **gdbppc** utilities, respectively, and must be able to find that utility in the executable path of the current process. For example, if the **gdb68k** and **xgdb68k** utilities were installed in */opt/gnu/bin*, then that directory must be present in your PATH environment variable.

Installing GDB

There is no need to rebuild GDB for Sun SPARC or Intel processors running Solaris 2.6 or newer. An installation script is provided to copy the executable files to an appropriate target installation directory. The script prompts you for the installation pathname, with a default set to */opt/gnu/bin.*

Install the CDROM labelled *GNU Tools*. To run the GDB build/installation script, type:

```
cd /cdrom/cdrom0
INSTALL.gdb
```

The script will ask you:

```
Would you like to rebuild GDB programs:[y/n]:
```

Answer **n** to this question. The script will prompt you for target type, for installation directory, and whether you want to use the X Window GUI. During the installation procedure, the tools are copied to their runtime locations.

Building and Installing GDB

The GNU tools must be built from the source code if either of the following is true: 1) your host is not a SPARC or Intel processor running Solaris 2.6 or newer, or 2) you are running an older version of Solaris. Run the installation script as described above; the script will ask you:

```
Would you like to rebuild GDB programs:[y/n]:
```

In this case, you answer **y**. The script will prompt you for target type, for installation directory, and whether you want to use the X Window GUI. The script will print out status of the build as it progresses. During the installation procedure, the tools are built and copied to their run-time locations. The *gnutools* directory in *pti/util* can then be removed, freeing the majority of the disk space.

mps Configuration File

By default, controller numbers that the Host mps (ntmps for Windows) driver associates with the PT controllers in the system are assigned in the order that they are made available to the driver. It is the Host operating system that scans for PCI devices in the system, and then calls the mps drivers attach routine for each PT controller found. The order in which each PT controller is presented to the driver is not always deterministic.

If you wish to specifically assign controller numbers to physical PT controllers (by the bus and device numbers), the mps driver configuration file can be used. For the Solaris operating system, this file is called mps.conf, and a template can be found in */usr/kernel/drv*. For the Linux operating system, depending on the release, this file is either *modules.conf* or *modprobe.conf* and can be found in the */etc* directory. For Windows, the registry key “ntmps” has two elements that help define the assignments

Solaris and Linux Operating Systems

In this file, you can set several parameters. **ptix_mps_number_controllers** will specify the number of PT controllers in the system, or zero if the configuration file should not be used, and the default controller assignments will be performed. The **ptix_mps_controller_entries** is a list of controller/bus/device numbers for each PT controller assignment. **ptix_mps_dev_path_entries** is specific to Solaris and is used for identifying controllers across multiple PCI domains.

Note: *If a default installation of the PTI software was done on the system, the template mps.conf file will be put in the /usr/kernel/drv directory.*

The Linux configuration file, *mps_linux.conf*, can be modified and appended into the pre-existing system file, */etc/modules.conf* or */etc/modprobe.conf*, depending on your release. For the Linux case, there must be no spaces or tabs between the comma delimited entries in **ptix_mps_controller_entries**. If the Host mps driver detects an error when parsing the file, it will report the error to the messages or log file, and reject loading.

Windows Operating System

The ability to specifically assign controller numbers in the Windows Operating System is available for versions 2000 and above. Specifically, this support is not available in Windows NT. The registry key for the *ntmps* driver is:

```
HKEY_LOCAL_MACHINE+SYSTEM+CurrentControlSet+Services+ntmps
```

This key has two elements, which need to be modified. **NumberOfControllers** will specify the number of PT controllers in the system, or zero if the manual configuration should not be used, and default controller assignments will be performed. This value is set to zero at installation time. **ControllerEntries** is a list of controller/bus/device numbers for each PT controller assignment. This is a comma and space separated list and specifies “controller number, bus number, device number” on each line. At install time, sample values are issued just to show the format method (note having –1s at the end is required). If physical assignments are desired, a registry editor utility (like regedt32) must be used to define the appropriate system.

General Usage Suggestion

To come up with appropriate controller/bus/device numbers for your specific system, we recommend to insert the PTI cards in the positions you desire, and have the Host mps driver load the default way for the first time. At that point, run “commstats –i”, and then use the values it returns in the configuration file.

Chapter 3

Network Configuration

Overview

This chapter contains basic instructions to configure your hardware to operate over your TCP/IP Network. Chose the appropriate section for your hardware and follow the instructions in that section.

The following topics are covered in this chapter:

- “Embedded NexusWare Controller,” on page 27
- “MPS1000,” on page 28
- “MPS800, MPS600 and MPS300,” on page 28
- “Configuring a Client,” on page 28
- “Configuring the Server,” on page 32

Embedded NexusWare Controller

The NexusWare CompactPCI controllers can be configured as TCP/IP servers. To do so, it is critical that the configuration of their jumpers or switches match that of the chassis to be used. Typically, this would imply that the controller be configured for ‘Standalone’ mode. In Standalone mode the controller is responsible for the generation of the clocking that would otherwise be supplied from the PCI bus.

Table 3-1, “Busless Configurations for NexusWare Controllers,” contains only the switches that are required when the PCI bus is not being used. For full details on the jumpers and switches associated with your specific controller refer to your hardware *User’s Guide* from PT.

Table 3-1: Busless Configurations for NexusWare Controllers

CPC Card	Switch	Description
CPC358	K38 2-3 IN	Install on busless systems. Connects on-board clock to Tundra PowerSpan
	K43 1-2 OUT	Allows for alternate power on control signal source
	K46 1-2 IN	No external controller is necessary – power always on
CPC388	K44 3-4 IN	No external controller is necessary – power always on
	K46 1-2 IN	Install on busless systems. Connects on-board clock to Tundra PowerSpan
	K47 1-2 OUT	Allow 3V signal to be put on PCI Bus pins if no PCI Bus is present
	K48 1-2 OUT	Removed for busless systems
CPC308 CPC324	SW5-2 ON	STANDALONE provide internal PCI clock

MPS1000

This section applies to the NexusWare based server, MPS1000. It will discuss what needs to be initially performed on the unit before it can be accessed over the LAN.

On the Master controller, networking will need to be enabled. Either a hard-coded IP address, or DHCP will need to be specified in the pticonfig file. This must be done with the supplied console cable. Refer to Chapter 2 (WCP Network Configuration) of the *NexusWare WAN Protocol Configuration Guide* (106P0150) for more details on the use of the *pticonfig* file.

MPS800, MPS600 and MPS300

This section applies to PT’s LAN-based MPS servers only. It explains how to configure the server so that it automatically loads and begins execution whenever it is powered on or reset. This configuration can be accomplished by connecting a terminal (opening a window) on the server’s console port. When a terminal is connected to the server, and the server power is cycled, the server PROM code provides an opportunity to specify its configuration by waiting five seconds before beginning the download process. If during that time you press the terminal’s space bar, the PROM code prompts you for configuration information. During configuration at the console, you may either provide a complete configuration or indicate that the Bootstrap Protocol (BOOTP) is to be used to provide the complete configuration. BOOTP is supported by the MPS 800 and MPS 300/600.

During initialization, PROM code in the server uses TFTP (Trivial File Transfer Protocol) to request its runtime code from a designated client on the network. You may configure the server with up to 100 clients that are able to provide the runtime code. If a client does not respond to the TFTP request, the server tries the next client in the list. If no client responds, and the Bootstrap Protocol is not configured, the server returns to the top of the list, and begins again, retrying indefinitely. If the Bootstrap Protocol is configured, the PROM issues a Bootstrap Protocol request message starting the download process from the beginning.

“Configuring a Client” describes how to configure a client so it can respond to the server’s TFTP requests for runtime image files. “Configuring the Server” describes how to configure the server with information that it needs to access the client(s) and load the runtime system and respond to connection requests from client applications.

Configuring a Client

Trivial File Transfer Protocol for UNIX Systems

In order to load the server with a runtime system over the network, the server’s boot prom code and the client communicate using the TFTP protocol. The file */etc/inetd.conf* configures the Internet daemon and enables the various protocols. Login as root, edit this file and find the entry for TFTP. If the entry has a # character at the left margin, TFTP is not enabled. Remove the # character. Also, delete the last portion of the line that reads:

```
-s /tftpboot
```

After editing the file, either reboot the system or kill the Internet daemon (the process name is *inetd*) and restart it with the command *inetd &*. After editing the file enable TFTP. On a Solaris 10 system this is done by converting the */etc/inetd.conf* entry into a Service Management Facility (SMF) service manifest and enabling the resulting service:

```
/usr/bin/inetconv
```

For prior Solaris releases or other UNIX systems either reboot the system or kill the Internet daemon (the process name is *inetd*) and restart it with the command inetd &. If TFTP is already enabled, this step is not necessary.

In order for a client application to access the server during runtime operations, the service name and port number must be added to the file */etc/services* on the client. The entry should look something like this:

```
mps    5555/tcp   # TCP, normal mode
```

A client application uses the service name in the first column to establish a connection to the server through the PT API. You should always use the service name mps.

The port number in the second column must be followed by */tcp*, which specifies the protocol used to access the server. The port number shown above is an example, and may be changed, following these guidelines:

- Use values between 5000 and 65,535 (decimal), and check the existing entries in the */etc/services* file to be sure that the values you choose are not already assigned to another service.
- The port number assigned must be the same on all clients that access the server

The final column contains an optional comment, designated by the # character.

To establish a connection to the server, the API also requires the server's logical host name and Internet address to be added to the *hosts* database. An entry in this file looks something like the following example:

```
100.0.0.8   indiana indy hoosier
```

The first column contains the Internet address in the standard dotted format. In this example, the host name is *indiana*, with *indy* and *hoosier* as aliases, or nicknames, which can be substituted for the host name. The Internet address you use must be consistent with your network, and should be assigned by your system administrator. You can choose any logical name for the server, and any number of aliases. This entry should look the same in the *hosts* database of all clients that access the server.

Trivial File Transfer Protocol for Windows

Having a PT server boot over the LAN from a Windows machine can only be performed when the Windows machine has a TFTP server running. The client TFTP server software is not standard for the Windows operating system. Therefore, you will need to obtain and install this TFTP software in order to download the PT server. PT uses TFTPServer by Solar Winds (http://solarwinds.net).

In order for a client application to access the PT server during runtime operations, the service name and port number must be added to the services file (%SystemRoot%\system32\drivers\etc\services) on the client. The entry should look something like this:

```
mps    5555/tcp   # TCP, normal mode
```

A client application uses the service name in the first column to establish a connection to the server through the PT API. You should always use the service name mps.

The port number in the second column must be followed by */tcp*, which specifies the protocol used to access the server. The port number shown above is an example, and may be changed, following these guidelines:

- Use values between 5000 and 65,535 (decimal), and check the existing entries in the *services* database to be sure that the values you choose are not already assigned to another service.
- The port number assigned must be the same on all clients that access the server.

The final column contains an optional comment, designated by the # character.

To establish a connection to the server, the API also requires the server's logical host name and Internet address to be added to the hosts database (%SystemRoot%\system32\drivers\etc). An entry in this file looks something like the following example:

```
100.0.0.8    indiana indy hoosier
```

The first column contains the Internet address in the standard dotted format. In this example, the host name is *indiana*, with *indy* and *hoosier* as aliases, or nicknames, which can be substituted for the host name. The Internet address you use must be consistent with your network, and should be assigned by your system administrator. You can choose any logical name for the server, and any number of aliases. This entry should look the same in the hosts database of all clients that access the server.

Bootstrap Protocol

The MPS 800 and MPS 300/600 servers can be configured using the Bootstrap Protocol (BOOTP), if the client machine is a UNIX system. BOOTP is discussed in Internet Request for Comments (RFC) 951, and RFC1084. If BOOTP is to be used, the BOOTP daemon must be running at the client when the server is booted. The file */etc/inetd.conf* configures the BOOTP daemon. Login as root, edit this file and add the line:

```
bootps dgram udp wait root /usr/etc/bootpd -i /usr/etc/bootptab
```

PT does not provide a BOOTP daemon. See your system administrator if your system does not include one and you wish to use BOOTP.

You must connect a terminal or open a window to the server's console port to configure the server to use BOOTP. When power is cycled or the server reset button is pushed, and the server is configured for BOOTP, the server broadcasts a BOOTP request message. If the host BOOTP daemon is running and an entry exists in the */usr/etc/bootptab* table for the server, then the BOOTP daemon sends a BOOTP reply message to the server. This reply can provide the server with the following information:

- Server's Internet address
- Configuration server's Internet address
- Configuration server's host name
- Configuration file name
- Internet address of gateway to configuration server

In order for your BOOTP daemon to reply to the server's BOOTP request, you must edit the */usr/etc/bootptab* file to provide an entry for the server. This entry includes the server's MAC (Ethernet) address, the server's Internet address and the full path name of a server configuration file. The MAC address is assigned to the server at the factory. You can read it at the server 300/600 console using the *E* command (See "Configuring the Server," on page 32 for further information. The MPS 800 displays the MAC address when the menu is displayed. The server's Internet address is assigned by you to meet your network's needs. The configuration file is a text file created by you. The content of this file is discussed in "Configuring the Server," on page 32.

The format of the bootptab file is not defined in the BOOTP Requests for Comments, and implementations vary. The following is a typical entry in a bootptab file:

```
# Provide an entry for the PTI MPS 300 communications
# server. The hd field defines the home directory. The ht
# field defines the hardware type. The ha field defines
# the MAC (Ethernet) address of the server. The ip field
# is the host's specification of the Internet address to
# be assigned to the server. The bf field is the full path
# name of the configuration file.
#
bootpClient:\
   :hd=/tftpboot:\
   :ht=ethernet:\
   :ha=00.03.31.02.15.D0:\
   :ip=206.251.239.143:\
   :bf=/usr/pti/mps300.config:
```

This entry includes the Ethernet address of the server (00:03:31:02:15:D0). When the BOOTP daemon reads the server's BOOTP request, it extracts the source Ethernet address from the request and checks to see if there is a matching entry in bootptab. If a matching entry is found, the BOOTP daemon builds a BOOTP reply message using the *ip* and *bf* fields from the bootptab entry and writes the message to the server. The server reads the reply and uses TFTP to retrieve the *bf* file from the machine at the Internet address specified at *ip*.

You may specify a machine to provide the configuration file that is on a network different from that of the server. Consider the following bootptab entry:

```
# Provide an entry for the PTI MPS 600 communications
# server. The hd field defines the home directory. The ht
# field defines the hardware type. The ha field defines
# the MAC (Ethernet) address of the server. The ip field
# is the host’s specification of the Internet address to
# be assigned to the server. The sa field is the Internet
# address of the machine to provide the configuration
# file. The T99 field specifies the Internet address of
# the gateway the server is to use to access the
# sa machine.
# The bf field is the full path name of the configuration
# file. Since the configuration file is provided by a
# machine on a different network, use the “vendor
# specific” area of the BOOTP reply to specify the Internet
# address of a gateway. T99=0Xcefbef81 = 251.206.239.129
#
bootpClient:\
   :hd=/tftpboot:\
   :ht=ethernet:\
   :ha=00.03.31.02.15.90:\
   :ip=206.251.239.143:\
   :sa=204.250.22.30:\
   :T99=0xCEFBEF81:\
   :bf=/usr/pti/mps600.config:
```

The server’s Internet address, as defined in the bootptab entry is 206.251.239.143. When this address is expressed as a 32-bit number, the first three bits are 110. Thus, the server has a Class C Internet address and its network identifier is 206.251.239. The *sa* field specifies a machine with a Class C address and its network identifier is 204.250.22. The gateway Internet address specified by the *T99* field is 206.251.239.129. It has the same class and network identifier as the server.

The *sa* field is the Internet address of the machine that is to provide the server with its configuration file. This machine is on a different network from the server. The *T99* field is the Internet address of the gateway the MPS is to use to access the machine at the *sa* address. The *T99* field provides the gateway’s Internet address as a 32-bit hexadecimal number.

You must be logged in as super user to modify bootptab.

Configuring the Server

The server’s network configuration parameters are saved in a non-volatile flash PROM. To program the parameters, you must attach a terminal to the server’s console port with an RS232 cable. The terminal should be set to 9600 baud, 8 bits, no parity. Once the terminal is attached, switch the server on and press the terminal’s space bar within five seconds. When you do this the very first time after installing your server, you will see the message:

```
Formatting uninitialized FLASH prom...
```

After a slight delay, a configuration menu is displayed on the terminal screen. For an MPS 800 server, the menu looks like this:

```
PTI Prom Version 1.0
MPS 800 Server Configuration

Ethernet Address: 00:11:22:33:44:55

C - Client’s Internet Address
I - Server’s Internet Address
E - Server’s Line Electrical Interface
N - Server’s Logical Name
P - Server’s (Listen) Port Numbers
F - Download Command File Pathname
U - User Defined Flash Prom
R - Routing Configuration
B - Remote Configuration
T - SNTP Configuration
W - Write New Flash Program
S - Save The Configuration

D - Enter PQbug
X - Exit And Start Download
Y - Exit, Download Then Enter PQbug
->
```

For the MPS 300/600 servers, the menu looks like this:

```
MPS 300/600 Server Configuration
C - Client’s Internet Address
I - Server’s Internet Address
E - Server’s Ethernet Address
N - Server’s Logical Name
P - Server’s (Listen) Port Numbers
F - Download Command File Pathname
G - Diagnostic Configuration
U - User Defined Flash Prom
R - Routing Configuration
L - Set Login Password
B - Remote Configuration
T - SNTP Configuration
S - Save The Configuration

D - Enter PTbug
X - Exit And Start Download
->
```

Client’s Internet Address

Select *C* to enter the Internet addresses of the clients that have been configured to load the server (See “Configuring a Client,” on page 28 for further information). Addresses are entered in dotted decimal notation. You may enter up to 100 addresses, typing a carriage return after each entry, and the character *Q* to return to the menu. Insert and delete commands are available to make modification of the list easier. As you edit the list, the existing content of each location is displayed. To change the address, type the new value and press return. To insert an address preceding the currently displayed entry, type *I* followed by the new address, then return. To delete the currently displayed entry, type *D* and return. When you reach the end of the list, a marker (0.0.0.0) is displayed instead of the current value. At this point, you can enter an address to add to the end of the list, or press return to go back to the top of the list. Enter a *Q* to return to the main menu.

Server’s Internet Address

Select *I* to program the server’s Internet address, also in dotted notation. This value must match the address assigned to the server and entered in the *hosts* database. See “Configuring a Client,” on page 28 for further information.

Server's Line Electrical Interface

This option is only supported by the MPS 800 server.

Select *E* to program the server’s line electrical interface. The MPS 800 supports 8 serial ports. The ports support the RS232 and RS422 (RS530) interface standards. The assignment of interface standards to server ports is accomplished with the *i* command. There is a two port granularity on the assignment, i.e., ports zero and one must have the same assignment, ports two and three must have the same assignment, and so on.

When you type an i, you are prompted to make the assignment for each of four pairs of server ports. Type a “0” to assign RS422 (RS530) to a port pair; type a “1” to assign RS232 to a port pair.

Server’s Ethernet Address

The server’s Ethernet address is assigned by the hardware manufacturer and cannot be modified. The MPS 800 displays the assigned Ethernet address with the menu. For the MPS 300/600, you can select *E* to display it.

Server’s Logical Name

Select *N* to enter the server’s logical name. This can be the server name or any of the aliases assigned to the server in the clients’ *hosts* databases.

Server’s (Listen) Port Numbers

Select *P* to program the server’s port numbers for normal operation. First, you are prompted for the server's TCP port number, which must match the port number assigned in the clients’ *services* database for the TCP/normal mode service (See “Configuring a Client,” on page 28 for further information). Next, you are prompted for the server's UDP port number. Enter zero, since you will not be using this service with most PT software products. Finally, you are prompted for the port number used by the server for broadcast connections. Broadcast connections are supported only by certain, special-purpose server protocols. You should enter zero for the broadcast port number.

Download Command File Pathname

Select *F* to enter the pathname of the server’s load command file. This file must be present, in the same location, on all clients configured to load the server. By default, it is installed in the directory */usr/pti* for Unix-based hosts (or *C:\pti* for Windows-based hosts), and is named *Cmdfile.prot*, where *prot* is a filename extension that identifies the protocol or software configuration. Check your *pti* directory to determine the exact name of your command file.

Diagnostic Configuration

Since most PT software products do not make use of the server's diagnostic mode, you do not need to program this option.

Routing Configuration

Select *R* to enter routing information. The MPS 800 and MPS 300/600 may be booted from a host on a local network or from a host on a separate network via a gateway. Once booted, clients may access the server from both local networks and separate networks via a gateway. The routing information is stored in a table as 16 pairs of Internet addresses in decimal dot notation. The first address of each pair is a destination network, the second address is the gateway to access the network.

For example, suppose the server’s Internet address is 206.251.239.143 and the client’s Internet address is 204.250.22.30. These addresses both begin with binary bits 110 and are, therefore, class C addresses but they have different network identifiers. The server accesses the client via a gateway. Suppose the gateway on the server’s local area network has Internet address 206.251.239.129. This configuration is present in the server’s routing tables as follows:

```
204.250.22.30     206.251.239.129
```

The first entry is the address of the client. The second is the address of the gateway on the server’s local area network that the server uses to access the client.

If a client’s Internet address indicates a different network identifier but there is no entry for the client in the server’s routing tables, the server looks in the routing tables for a default gateway. If a network address of 255.255.255.255 exists in the table, then the server uses the corresponding gateway address as its default gateway.

The following entry in the routing table indicates that the host at Internet address 206.251.239.129 is a default gateway:

```
255.255.255.255   206.251.239.129
```

If your network uses subnet addressing you can provide your subnet mask to the server with an entry in the routing tables. If a network address exists in the table that is the same as the server’s Internet address, then the server interprets the corresponding gateway address as a subnet mask. For example, many sites choose to subnet class B addresses by using the third octet of to identify subnets and the fourth octet to identify hosts. This yields a subnet mask of 255.255.255.0. If the server’s Internet address is 136.19.91.35, the following entry in the routing table indicates this subnet mask:

```
136.19.91.35      255.255.255.0
```

The first entry is the Internet address of the server. This cues the server to use the second entry in the table as the subnet mask.

When you select *R*, the server prompts you to enter *d* to display the table, *x* to exit from the menu item or *m* to modify the table. If you select *m*, the server prints the Internet address of the first entry. In response to this prompt, you can enter an Internet address or type *q* to quit the menu item or *d* to delete the network/gateway entry pair. If you enter an Internet address and press *Return*, the server displays the corresponding gateway Internet address. You may then enter an address for the gateway or *d* to delete the network/gateway entry pair. If you enter an address and press ENTER, the server displays the next entry in the table.

User-Defined Flash Prom

A 1024-byte area of the flash PROM is reserved for configuring product specific parameters. Most PT software products do not require any parameters to be programmed into this area of the flash PROM. However, three features are configured by entries in this area of flash PROM.

Auto Reset

The server can be configured to perform an automatic software reset when unrecoverable errors are detected. The reset concludes with downloading the server and restarting operation. To enable the auto reset feature, the first four bytes of the user flash area (starting at offset 0) must be set to 0x00, 0x00, 0xA0 and 0xD0. If any other values are used, when the server detects an unrecoverable error it prints diagnostic information at the console device and halts.

Network Reset

The server has the ability to be reset over the Local Area Network and by default is set to allow this action. See *commreset* and *commload* in “LAN Server Reset Utility (commreset),” on page 67 and “Download Program (commload),” on page 55 for further information. If security is a concern, and only a physical reset (pushing the reset button or cycling power) can be allowed, the address 0x3fe of the user defined flash prom should be set to a value other than 0x00.

System Report

The server can be configured to generate a report at the console device on a periodic basis. The System Report includes statistics on the usage of buffers and memory by the server. To enable display of the System Report, the last byte of the user flash area (offset 0x3ff) must be modified to a nonzero hexadecimal value between 1 and 0xff, which specifies the frequency in seconds for the report. The System Report is disabled by entering a zero at this offset (the factory default).

Login Password

The Login Password is not implemented on the MPS 800.

If you wish to restrict access to the configuration menu, select *L* to set up a login password. You are prompted to enter a password, and then prompted to retype it for verification. The characters you type are not displayed on the terminal screen, and the password is encoded in the flash PROM so that it is not readable. You may use any combination of letters, digits and special characters, up to a maximum of 64. The password is terminated with a carriage return. Once you have entered a password and saved it in the flash PROM, the server always prompts you to supply your password when you attempt to access the configuration menu. If you do not type the correct password, the configuration menu is not displayed.

Remote Configuration

The MPS 800 and MPS 300/600 servers can use the Bootstrap Protocol (BOOTP) to obtain the server configuration. The configuration is defined in a text file you provide. This text file is read by the server at boot time and is an alternative to using the console to configure the server. The server uses the Trivial File Transfer Protocol (TFTP) to read the configuration file. To use BOOTP, you must provide an entry in your host’s bootptab file, and make sure your host’s BOOTP daemon is executing when the PT server is booting.

If you select *B*, the server asks you if you wish to enable remote configuration using BOOTP. If you respond *y*, and follow this with *X* to start download or *S* to save the configuration and then *X*, the server uses BOOTP to obtain its configuration. When using BOOTP, the server first clears the RAM memory version of the flash PROM information. It then initializes the RAM copy of the flash PROM using the information in the configuration file. If the configuration file is processed without error, the RAM copy of the PROM information is written back to PROM.

You cannot use both the BOOTP protocol and the console to configure the server. To use BOOTP, you must request it with the *B* option. Once you do this, the setting is preserved in flash PROM and BOOTP is used until the option is disabled.

The Bootstrap Protocol consists of two messages: a BOOTP request and a BOOTP reply. The PT server writes the BOOTP request and reads the BOOTP reply that is written by the BOOTP daemon executing on a BOOTP host. The reply includes the following information:

- The server’s Internet address
- The server’s configuration full path file name

The BOOTP daemon obtains this information from the bootptab file entry you provide. The PT server uses TFTP to read the configuration file to obtain its configuration. Each line in the file is a configuration command. The line content that follows the # character is regarded as comment. The commands may be entered in either upper or lower case. The following are the valid commands: CLIENT, SERVER, NAME, PORTS, FILE, DIAG, USER, ROUTE, SNTP, and ELEC.

These commands are used in the same way as when the server is being configured at the console. The discussion above provides details on how the server uses the configuration information. The following describes the format of the commands.

CLIENT — The CLIENT command is followed by one or more client Internet addresses separated by spaces or tabs. The format of the addresses is decimal dot notation. For example,

```
CLIENT          206.251.239.148  206.251.239.149
```

SERVER — The SERVER command defines the Internet address of the MPS 800 and MPS 300/600 server. If a SERVER command is omitted from the configuration file, the server uses the Internet address provided in the BOOTP reply. Only one address is allowed with the SERVER command. For example,

```
SERVER          206.251.239.143  # PTI MPS 600
```

NAME — The NAME command identifies the server’s name. For example,

```
NAME       diamond
```

PORTS — The PORTS command has up to three entries: the server’s tcp port, udp port and udp broadcast port. The ports are provided as decimal numbers. For example,

```
PORTS      48879 0     0
```

The above entry indicates that the server is to use the TCP protocol and to “listen” at “well known port” 48879. In addition, the entry indicates that the UDP protocol is not to be used. These are typical assignments. The port assignments must match those in the */etc/services* file.

FILE — The FILE command defines the full path name of the PT server’s download command file. For example,

```
FILE       /usr/pti/Cmdfile.x25
```

After the server reads the configuration file and stores the content in flash PROM non-volatile memory, it retrieves */usr/pti/Cmdfile.x25* to obtain a list of the files that are to be loaded into RAM.

DIAG — The DIAG command configures an MPS3000 and MPS300/600 server to run in a diagnostic mode. The command is followed by the responses to the following questions:

```
Are diagnostics supported? (y or n)
```

```
Enter diagnostic TCP port number (decimal)
Enter diagnostic UDP port number (decimal)
Invoke diagnostic mode upon exiting? (y or n)
```

If the first argument is *n*, the rest of the command is ignored. For example,

```
DIAG       n     #     Diagnostics not supported
```

USER

The USER command allows the user area of flash PROM to be set. The command is followed by a hexadecimal offset from the beginning of the user flash PROM area, and a hexadecimal value for the byte at that offset. For example, the following enables the auto reset feature:

```
# Enable auto reset
USER       0     0
USER       1     0
USER       2     0xA0
USER       3     0xD0
```

ROUTE

The ROUTE command allows the user to specify three items of information that relate to TCP/IP routing:

- The Internet address of gateways that can be used to access a different network
- The Internet address of a default gateway
- A subnet mask

Each ROUTE command is followed by two Internet addresses in decimal dot notation. The first address is that of a Internet destination. The second is the address of a gateway to be used to reach the destination. There are two special cases. If the destination address is 255.255.255.255, the server regards the gateway address as the default gateway. If the destination address is the same as that of the server, the server regards the corresponding gateway address as a subnet mask. For example:

```
ROUTE  204.250.22.30    206.251.239.129#      Route to 204.250.22.30
ROUTE  255.255.255.255  206.251.239.129#      Default gateway
ROUTE  206.251.239.143  255.255.255.240#      Subnet mask
```

The above defines a gateway to a remote Internet destination and the Internet address of a default gateway. In addition, assuming the server’s Internet address is 206.251.239.143, the last line in the example defines a subnet mask of 255.255.255.240 or 0XFFFFFFF0.

SNTP

The SNTP command allows configuration of two parameters used by the Simple Network Time Protocol (SNTP). The first parameter specifies the Internet address of the NTP server platform, and the second parameter specifies the Poll Frequency Code (See “SNTP Configuration” on page 40 for further information):

```
SNTP       206.251.239.141       5
```

ELEC	The ELEC command allows configuration of the electrical interface for MPS800 servers. The MPS800 supports 8 RS232 and RS422 (RS530) serial ports. There is a two port granularity on the assignment of electrical interfaces, i.e., ports zero and one must have the same assignment, ports two and three must have the same assignment, and so on. Thus, the ELEC command has 1 or 4 parameters, designating the line interface for ports 0/1, 2/3, 4/5, and 6/7. Each parameter must be either RS232 or RS530. In the example below, ports 0/1 and 4/5 are RS232 and ports 2/3 and 6/7 are RS530.

```
ELEC       RS232 RS530 RS232 RS530
```

SNTP Configuration

The MPS 800 and MPS 300/600 servers can use the Simple Network Time Protocol (SNTP) to maintain and synchronize a local system clock to a reference clock provided by a Network Time Protocol (NTP) or SNTP Server. This option provides the ability for PT serial protocols to then timestamp serial data traffic with the Universal Coordinated Time (UTC) time of day. When used with an accurate reference clock such as that provided by Global Positioning System (GPS), this option allows the MPS server to provide timestamping capability for received serial data traffic to the nearest millisecond.

Once the *T* option is selected, you are prompted for the Internet address of the NTP server platform. This platform must be an NTP (or SNTP) server on your local subnet in order to provide clock synchronization to the nearest millisecond. After entering this address, you are prompted for the Poll Frequency Code value. This value translates into the number of seconds between polls to the NTP server, in terms of powers of 2. For instance, the value 4 designates polls to the NTP server every 16 seconds, the value 5 designates polls to the server every 32 seconds, and so on. The valid range of entries is 4 to 14, which is the complete valid range proscribed in the SNTP specification. However, for practical use with a PT server, the subset 4 to 7 is highly recommended, and ideally, one should specify 4. If SNTP is no longer required, specifying a Poll Frequency Code of 0 will disable the SNTP feature. The factory setting is to disable SNTP.

Note: *The Universal Coordinated Time is the number of seconds (and in this case, microseconds) that have elapsed since January 1, 1970 at the zero meridian of Greenwich, England. This format is used by most computer systems to provide a time value that is independent of time zone and daylight savings time concerns. Consult the Programmer's Guide(s) for the serial protocol(s) you have purchased for more information on how timestamps are reported.*

Write New Flash Program

The MPS 800 supports updating the Boot Flash code in the field. This is not a part of the configuration process but can be helpful in certain maintenance situations. The command causes a program in the Boot Flash code to be copied to RAM. This program erases the existing Boot Flash code, reads a new Boot Flash image from the boot client via TFTP, programs the Boot Flash device with the image, and then transfers control to the Boot Flash device.

If you are provided with an updated Boot Flash image file, instructions on the usage of the update procedure will accompany the image.

Enter Debugger

The Boot Prom code includes a debugger to aid in field service. Invoking the debugger is not a part of the configuration process.

Exit and Start Download

Once all the network parameters have been configured, select *S* to program the flash PROM and save the configuration. This takes a few seconds. Then, select *X* to exit from the configuration menu and transfer control to the server's TFTP load program.

Note: *If you do not save the configuration before exiting from the configuration menu, the values you changed take effect when the server is booted, but will be lost on the next reboot or when the server is powered off.*

The server configuration parameters may be modified at any time by repeating the steps outlined in this section.

Exit, Download Then Enter PQbug

The MPS 800 Boot Prom code includes a debugger to aid in field service. If *Y* is selected, the server downloads the runtime image but instead of transferring to the image, it transfers to the debugger instead. This command is not part of the configuration process.

Note: *If you do not save the configuration before exiting from the configuration menu, the values you changed take effect when the server is booted, but will be lost on the next reboot or when the server is powered off.*

The server configuration parameters may be modified at any time by repeating the steps outlined in this section.

Chapter 4

QMC Configuration Guide

Overview

This chapter does not apply to controllers running PT’s NexusWare. Please refer to the *NexusWare WAN Protocol Configuration Guide* (106P0150) that came with your NexusWare controller for use of the NWC configuration utility.

A special configuration utility is provided for controllers running with the QUICC Multichannel Controller (QMC). All PT 37x/38x PowerQUICC controllers use the QMC. Source code for the configuration utility, *qmc.c*, is included with the release in the HDLC *config* directory. The installation procedure builds the utility *qmc* and copies it to */etc* (in the UNIX environment).

The QMC utility reads configuration parameters for each physical connection (called a span) from a text file. Sample configuration files, *t1config* and *e1config*, are also included in the HDLC *config* directory and copied to */etc* by the installation script. The sample files contain default parameters to initialize one or both spans on the controller. You should modify the appropriate configuration file according to your specific site requirements.

The following topics are covered in this chapter:

- “QMC Configuration File Format,” on page 43
- “T1 Configuration Parameters,” on page 44
- “E1 Configuration Parameters,” on page 48
- “Running the QMC Configuration Utility,” on page 52

QMC Configuration File Format

The QMC utility uses the information in the configuration file to configure spans on the controller. The parameters specified in the file determine the corresponding parameters of the *T1CONFIG* (or *E1CONFIG*) structure. This structure is defined in the file *hdlchdr.h* located in the hdlc include directory. Once all parameters have been extracted from the configuration file and validated, the QMC utility sends a *T_CONFIG_REQ* ioctl command via an MPSioctl API call to execute the reconfiguration request.

Each entry in the configuration file is composed of a parameter “keyword” followed by a corresponding parameter value (or values). The file is divided into sections, where each section configures a single span, and contains an entry for each defined parameter. The following general rules apply:

- The first keyword of each section must be “span”. This is the only required keyword. Other keywords may be listed in any order.
- Each keyword must be followed by a valid parameter on the same line, separated by white space (one or more spaces and/or tabs).
- Keywords and parameters may be entered in either upper or lower case.

- Blank lines and unrecognized entries are ignored. Comments may be included anywhere in the file (prefaced by a “#” character)

T1 Configuration Parameters

The keywords and valid parameter values for T1 controllers are listed below:

`span` specifies the physical connection on the controller. There are two connectors designated as port 0 and port 1. These are synonymous with spans 0 and 1.

`buildout` is an integer value (0-7) that specifies the amount of energy used to drive the T1 signal down the span. It is a function of distance and will provide a “hot” signal for longer distances in a DSX-1 (digital cross connect application), or provide a dB gain factor in the case of a CSU application. The following table lists the possible values for *buildout*:

ValueLine	Buildout	Application
0	0 to 133 feet/0 dB	DSX-1/CSU
1	133 to 266 feet	DSX-1
2	266 to 399 feet	DSX-1
3	399 to 533 feet	DSX-1
4	533 to 655 feet	DSX-1
5	-7.5 dB	CSU
6	-15 dB	CSU
7	-22.5 dB	CSU

The default value for *buildout* is 0.

`framing` framing defines the type of framing used on the span, which must match the type of framing used by the connecting equipment:

ESF = Extended Super Frame

SF = Super Frame

Extended Super Frame is the default and is used in most modern-day equipment. With this framing type, a CRC is generated on transmission and checked during receipt for every extended superframe. This provides a mechanism for ensuring the integrity of the span.

`encoding` defines the method of encoding to be used on the span:

B8ZS = Bipolar 8 - Zero Substitution

AMI = Alternate Mark Inversion

B8ZS encoding is a method of replacing a string of eight consecutive zero bits with a binary pattern that will maintain a one’s density on the line. This is necessary to keep the clocking synchronized between the communicating systems. All data type applications where data can be passed with 8 or more consecutive zero bits require B8ZS encoding (this is the default).

AMI encoding specifies that logic one bits alternate polarity (positive/ negative). If for any reason, the data stream is not set to B8ZS at the other end of the span, AMI encoding should be selected.

`alarms` enables/disables alarms. The alarms function is not implemented in this release.

`clocking` specifies the type of clocking used for the transmit side of the framer. The receive side clocking is always derived from the span. The transmitter, however, can derive clocking from three different sources:

EXTERNAL - refers to an external clock supplied at the inputs of the sync port located on the board. This option is used when a reference clock is being supplied by some other piece of equipment.

INTERNAL - refers to the on-board oscillator which supplies a stratum 4 clock source to the transmitter. Stratum 4 has a requirement of 30ppm (parts/million) tolerance and is used in CPE end equipment. This option is used when the board is the reference point for the clocking within the system.

RECOVERED - refers to using the recovered receive clock as the source for the transmit side. This option is used when the board is slaved to the communicating equipment at the other end.

`maxframe` specifies the maximum transmit and receive frame size for all channels (logical links, defined in timeslot parameter below) on the span.

`timeslot` specifies a channel and bit rate for each timeslot on the T1 span. In this nomenclature, a timeslot is equivalent to a DS0 (Data Signal Level 0).

The T1 span is split into 24 timeslots which are time-division multiplexed onto the bitstream. Each timeslot is 8 bits wide and occurs at 125us intervals to yield a 64Kbit stream. The first parameter on the line identifies the timeslot number (0 to 23).

Each timeslot defined is assigned a channel number (specified in the second parameter on the line). The channel number (0-23) is interpreted as a logical link number by the HDLC serial device driver as follows:

Span 0: Channels 0-23 => Logical Links 0-23

Span 1: Channels 0-23 => Logical Links 24-47

Multiple timeslots may be assigned to a given channel to produce FT1 (fractional T1 channels). Conversely, the bandwidth of a channel depends upon the number of timeslots assigned to the channel. For example, if eight 64Kbit timeslots are assigned to channel 3, the channel will be allocated 8 x 64Kbits = 512Kbits/sec or 1/3 of the T1 span. Obviously, both sides of the communicating equipment must agree to this mapping of timeslots and channels.

The next parameter on the line specifies if a masking option should be implemented to provide a 56Kbit rate rather than the full 64Kbit rate for a particular timeslot. The following values are allowed:

64K - use 64Kbit rate

56L - use 56Kbit rate (masking low-order bit)

56M - use 56Kbit rate (masking high-order bit)

In certain T1 applications, the requirement is to provide 56Kbit links as opposed to 64Kbit interface to end equipment that has a 56Kbit rate (such as a DDS service). The bit that is masked off in the 8-bit timeslot value is either the low bit (normally) or the high bit. In either case, this yields a 7-bit word which occurs every 125 us and produces the 56Kbit rate.

The final parameter on the line is an optional parameter that specifies if the channel assigned to the timeslot will be operating in HDLC or transparent (clear channel) mode:

hdlc -   HDLC channel

clear - Transparent (clear) channel

<blank> Use last value (if any) for channel (default is HDLC)

Note that this parameter applies to the *channel* and not the *timeslot*, so all timeslots assigned to a channel must effectively be running the same mode (i.e. all must be in HDLC or all transparent for a given channel). However, it is possible to designate some channels as HDLC channels and others as clear channels.

If this parameter is not specified, the mode of operation for the channel will default to the value previously assigned to the channel (if any). If this is the first timeslot assigned to the channel, the default is HDLC mode.

A sample T1 configuration file follows:

```
span           0
buildout       0
framing        ESF
encoding       B8ZS
alarms         on
clocking       internal
maxframe       2048

# timeslot assignments for span 0:
#
# 1st parameter is timeslot number
# 2nd parameter = channel (0-23) = links 0-23
# 3rd parameter = bitmask:
#       64K means use all bits in timeslot
#       56L means mask LS bit in timeslot
#       56M means mask MS bit in timeslot
# 4th parameter = operating mode:
#       hdlc (default)
#       clear

timeslot 0      0       64K      hdlc
timeslot 1      1       64K      clear
timeslot 2      2       64K      <blank>
timeslot 3      3       64K      clear
...........    ..      ...
timeslot 23     23      64K      hdlc

span           1
buildout       0
framing        ESF
encoding       B8ZS
alarms         on
clocking       internal
maxframe       2048

# timeslot assignments for span 1:
#
# 1st parameter is timeslot number
# 2nd parameter = channel(0-23) = links 24-47
# 3rd parameter = bitmask:
#       64K means use all bits in timeslot
#       56L means mask LS bit in timeslot
#       56M means mask MS bit in timeslot
# 4th parameter = operating mode:
#       hdlc (default)
#       clear
```

```
timeslot 0      0       64K      hdlc
timeslot 1      1       64K      clear
timeslot 2      2       64K      <blank>
timeslot 3      3       64K      clear
...........    ..     ...
timeslot 23     23      64K      hdlc
```

Table 4-1: T1 Configuration Details

Parameter	Possible values	Default Value
buildout	0-7	0
framing	ESF or SF	ESF
encoding	B8ZS or AMI	B8ZS
alarms	on /off	on
clocking	internal, external, or recovered	internal
maxframe	any (up to max configured buffer size)	2048

timeslot	timeslot #: 0-23
	channel #: 0-23
	bitrate: 64K/56L/56M
	mode: HDLC or clear
		timeslot n = channel n
		bitrate = 64K
		mode = HDLC

E1 Configuration Parameters

The keywords and valid parameter values for E1 controllers are listed below:

span specifies the physical connection on the controller. There are two connectors designated as port 0 and port 1. These are synonymous with spans 0 and 1.

encoding defines the method of encoding to be used on the span:

HDB3 = High Density Bipolar 3 - Zero Maximum

AMI = Alternate Mark Inversion

HDB3 encoding is a zero suppression method of replacing a string of four consecutive zero bits with a binary pattern that will maintain a one’s density on the line. This is necessary to keep the clocking synchronized between the communicating systems. All data type applications where data can be passed with 4 or more consecutive zero bits require HDB3 encoding (this is the default).

AMI encoding specifies that logic one bits alternate polarity (positive/negative). If for any reason the data stream is not set to HDB3 at the other end of the span, AMI encoding should be selected.

signaling specifies the type of signaling to be used on the span, which must match the type of signaling used by the connecting equipment:

CCS = Clear Channel Signaling

CAS = Channel Associated Signaling

Note that if CCS is used (default), the signaling channel which is in timeslot 16 can be used to pass data transparently. In this mode of operation, timeslot 16 may be used for normal operations and assigned to any logical channel.

When using CAS signaling, timeslot 16 is reserved for sending a specific framing word, and thus cannot be used for data transfer. An attempt to assign a logical channel to timeslot 16 will be rejected if CAS signaling is selected.

`crc4` enables/disables a 4-bit CRC in the E1 frame. The CRC is included in timeslot 0, which is used for framing the E1. A value “on” for this parameter enables the CRC (default), and a value of “off” disables the CRC. If the connecting equipment uses CRC, this feature must be enabled.

`alarms` enables/disables alarms.

`clocking` specifies the type of clocking used for the transmit side of the framer. The receive side clocking is always derived from the span. The transmitter, however, can derive clocking from three different sources:

EXTERNAL - refers to an external clock supplied at the inputs of the sync port located on the board. This option is used when a reference clock is being supplied by some other piece of equipment.

INTERNAL - refers to the on-board oscillator which supplies a stratum 4 clock source to the transmitter. Stratum 4 has a requirement of 30ppm (parts/million) tolerance and is used in CPE end equipment. This option is used when the board is the reference point for the clocking within the system.

RECOVERED - refers to using the recovered receive clock as the source for the transmit side. This option is used when the board is slaved to the communicating equipment at the other end.

`maxframe` specifies the maximum transmit and receive frame size for all channels (logical links, defined in timeslot parameter below) on the span.

`timeslot` specifies a channel and bit rate for each timeslot on the E1 span.

The E1 span is split into 32 timeslots which are time-division multiplexed onto the bitstream. Each timeslot is 8 bits wide and occurs at 125us intervals to yield a 64Kbit stream. The first parameter on the line identifies the timeslot number (1 to 31).

Some restrictions apply to the use of timeslots on E1 spans: Timeslot 0 is reserved for framing the E1 span and cannot be used for data. When using CAS signaling, timeslot 16 is reserved for sending a specific framing word, and thus cannot be used for data transfer. An attempt to assign a logical channel to timeslot 16 will be rejected if CAS signaling is selected.

Each timeslot defined is assigned a channel number (specified in the second parameter on the line). The channel number (0-31) is interpreted as a logical link number by the HDLC serial device driver as follows:

Span 0: Channels 0-31 => Logical Links 0-31

Span 1: Channels 0-31 => Logical Links 32-63

Multiple timeslots may be assigned to a given channel to produce FE1 (fractional E1 channels). Conversely, the bandwidth of a channel depends upon the number of timeslots assigned to the channel. For example, if eight 64Kbit timeslots are assigned to channel 3, that channel will enjoy 8 x 64 = 512Kbits/sec or 1/4 of the E1 span. Obviously, both sides of the communicating equipment must agree to this mapping of timeslots and channels.

The next parameter on the line specifies if a masking option should be implemented to provide a 56Kbit rate rather than the full 64Kbit rate for a particular timeslot. The following values are allowed:

64K - use 64Kbit rate

56L - use 56Kbit rate (masking low-order bit)

56M - use 56Kbit rate (masking high-order bit)

The bit that is masked off in the 8-bit timeslot value is either the low bit (normally) or the high bit. In either case, this yields a 7-bit word which occurs every 125 us and produces the 56Kbit rate.

The final parameter on the line is an optional parameter that specifies if the channel assigned to the timeslot will be operating in HDLC or transparent (clear channel) mode:

hdlc - HDLC channel

clear - Transparent (clear) channel

<blank>Use last value (if any) for channel (default is HDLC)

Note that this parameter applies to the *channel* and not the *timeslot*, so all timeslots assigned to a channel must effectively be running the same mode (i.e. all must be in HDLC or all transparent for a given channel). However, it is possible to designate some channels as HDLC channels and others as clear channels.

If this parameter is not specified, the mode of operation for the channel will default to the value previously assigned to the channel (if any). If this is the first timeslot assigned to the channel, the default is HDLC mode.

Warning:
Performance on the E1 card is directly affected by the number of timeslots configured. It is highly recommended that you configure only the timeslots that you will actually be using on the span. Be advised the QMC configuration utility may restrict the total number of timeslots that you may assign. You may eliminate unused timeslots by simply commenting the line in the configuration file (inserting a “#” character prior to the “timeslot” keyword).

A sample E1 configuration file follows:

```
span           0
encoding       HDB3
signaling      CCS
crc4           on
```

```
alarms         on
clocking       internal
maxframe       2048

#
# timeslot assignments for span 0:
#
# 1st parameter is timeslot number
# 2nd parameter = channel (0-31) = links 0-31
# 3rd parameter = bitmask:
#        64K means use all bits in timeslot
#        56L means mask LS bit in timeslot
#        56M means mask MS bit in timeslot
# 4th parameter = operating mode:
#        hdlc (default)
#        clear

timeslot 1      0       64K       hdlc
timeslot 2      1       64K       <blank>
timeslot 3      2       64K       clear
...........    ..      ...
timeslot 24     23      64K       hdlc


span           1
encoding       HDB3
signaling      CCS
crc4           on
alarms         on
clocking       internal
maxframe       2048
#
# timeslot assignments for span 1:
#
# 1st parameter is timeslot number
# 2nd parameter = channel(0-31) = links 32-63
# 3rd parameter = bitmask:
#        64K means use all bits in timeslot
#        56L means mask LS bit in timeslot
#        56M means mask MS bit in timeslot
# 4th parameter = operating mode:
#        hdlc (default)
#        clear

timeslot 1      0       64K       hdlc
timeslot 2      1       64K       <blank>
timeslot 3      2       64K       clear
...........    ..      ...
timeslot 24     23      64K       hdlc
```

Table 4-2: E1 Configuration Details

Parameter	Possible Values	Default Value
encoding	HDB3 or AMI	HDB3
signaling	CCS or CAS	CCS
crc4	on / off	on
alarms	on /off	on
clocking	internal, external, or recovered	internal
maxframe	any (up to max configured buffer size)	2048

timeslot	timeslot #: 1-31
	channel #: 0-31
	bitrate: 64K/56L/56M
	mode: HDLC or clear
		timeslot n = channel n-1
		bitrate = 64K
		mode = HDLC

Running the QMC Configuration Utility

The QUICC Multichannel Controller Configuration Utility (*qmc*), may be used to get or set T1/E1 configuration parameters or to obtain current status and statistics relating to the T1/E1 span. Note that running the *qmc* utility is optional - i.e. you are not required to run the *qmc* utility in order to commence data transfer options. The QMC is initialized with the default values for all parameters as outlined in the previous sections.

The *qmc* utility may be executed any time after the software has been downloaded to the controller. However, a reconfiguration of parameters (set parameters option) may only occur if there are no active host applications attached to the controller. To run the *qmc* utility, use the following command:

```
/etc/qmc   [-g] [-s] [-f filename] [-t stats time] [-v] [-z] [-c
controller]
```

`-c controller`
: This parameter specifies the controller number. If not specified, controller defaults to 0.

`-g`
: This option can be used to get and display the current configuration for each span on the controller. This is the default action taken by qmc if no option is selected.

`-s`
: This option can be used to set and display the new configuration for each span on the controller. In addition, all statistics for the card are cleared. The configuration file is specified via the “-f” parameter. Note that this option may only be executed if there are no applications using the controller.

`-f filename`
: This option is used to specify the pathname of the configuration file for a set (“-s”) operation. If this option is not included, the pathname defaults to /etc/t1config (in the UNIX environment) or t1config (for Windows NT). Presence of this option implies a “set” operation.

`-t` *`stats time`*
: This option allows periodic monitoring of T1/E1 and HDLC statistics. The monitoring interval is specified in seconds. Once qmc has executed all other requests, it enters into an indefinite loop, requesting and displaying statistics at the specified interval.

`-v` This option causes qmc to display detailed information as it performs a reconfiguration.

`-z` This option causes qmc to zero all T1/E1 statistics.

Options may be specified independently or in combination. Some examples follow:

5. Request current configuration on controller 1:

```
/etc/qmc -c 1          - OR -
/etc/qmc -g -c 1
```

6. Set new configuration on controller 0 using file *myconfig:*

```
/etc/qmc -f myconfig   - OR -
/etc/qmc -s -f myconfig
```

7. Clear all statistics on controller 0 and monitor every 3 seconds:

```
/etc/qmc -z -t 3
```

Statistics displayed by *qmc* include the status of each span, T1/E1 network statistics and HDLC statistics:

```
Stats for T1 card (controller 1)
   T1 Span 0 Status: Network is UP
   T1 Span 1 Status: Network is UP

                             Span 0     Span 1
Severely errored seconds:    0          0
Errored seconds:             1          1
Line code/bipolar violations:3          3
CRC errors:                  0          0
Out of frame errors:         0          0
Messages received:           177        386
Messages transmitted:        386        177
Receive overruns:            0          0
Receive frames too big:      0          0
Receive CRC errors:          0          0
Receive aborts:              0          0
```

The following is a list of possible statuses for the span:

Network is UP the span is synchronized and ready for data transfer operations.

Receive Carrier Loss
set when 192 consecutive zeros (T1) or 255 consecutive zeros (E1) have been detected on the span.

Receive Loss of Sync
set when the device is not synchronized to the receive T1/E1 stream.

Receive Blue Alarm (T1 only)
known also as AIS (Alarm Indication Signal) - set when over a 3ms window, five or less zeros are received. This alarm indicates that the communicating equipment has experienced either a loss of sync or loss of carrier.

Loss of Transmit Clock (T1 only)
set when the TCLK pin has not transitioned for one channel time (or 5.2 us). This error indicates that a span configured for external clocking is not receiving a reference clock.

Receive Unframed All 1s (E1 only)
known also as AIS (Alarm Indication Signal) - set when a serious condition has been detected by the connecting equipment (such as loss of sync or loss of carrier).

Chapter 5

Client Utilities

Overview

This chapter contains information on the client utilities used to monitor and maintain the software running on embedded controllers and LAN servers in the PT environment. The utilities are installed in the PT *bin* directory, and are referenced as *commload*, *commupdate, commstats*, and *commreset*.

Note: *Only the commstats utility applies to the NexusWare controllers (PCI384, CPC38x).*

The following topics are covered in this chapter:

- "Download Program (commload)," on page 55
- "Communications Server Update Program (commupdate)," on page 57
- "System Statistics Monitor Program (commstats)," on page 63
- "LAN Server Reset Utility (commreset)," on page 67

Download Program (commload)

Before your client application can access communication protocol software, you must load the software to the server or communication controller. For a LAN-based server, this is done automatically when the server is powered on or reset. The network load procedure is described in Appendix A, "Network Load Procedures" on page 69.

For an embedded communication controller, download requires the use of the *commload* utility.

Description

For embedded controllers, the *commload* program accesses the PT STREAMS driver to download one or more controllers according to instructions provided in the command file *Cmdfile.xxx* (the actual name of the file will vary depending on the protocol software). If the full pathname for C*mdfile.xxx* is not specified, *commload* expects to find the file in the current directory.

Options

`-s` *`server`*	This parameter identifies the target server. For a LAN-based server, this is the logical host name as specified in the client's *hosts* database. For a controller installed in a backplane, *server* is the pathname of the driver's cloneable special file. If this option is not specified, the server name defaults to */dev/ucsclone* for UNIX, and \\.*PTIXpqcc* for Windows-based systems.

`-S` *`service`*	For LAN-based servers, this parameter identifies the service name to be accessed, as specified in the client's *services* database. If this option is not specified, the service name defaults to *mps*. For a controller installed in a backplane, this parameter is not used.
`-c` *`ctlrnum`*	Load only controller number *ctlrnum*, ignore load instructions in the command file for any other controller. If this option is not specified on a Unix based system, all controllers in the command file will be loaded. The *ctlrnum* option is required for a Windows-based system.
`-r`	Reset (but not download) the specified controller number (-c ctlrnum). In this case, a command file should not be specified.
`-v`	Verbose. Print the status of each step of the commload process (reset, load, exec). If this option is not specified, commload generates no output unless an error is encountered.
`-i`	Will print out the version, strings of the currently loaded device driver, the PT API library and all the modules specified in the command file (*Cmdfile.xxx*). This is for informational purposes only. If the “-i” option is specified, the card will not be loaded.

To download an embedded controller on a Unix-based system, type the following command:

```
commload -v Cmdfile.prot
```

To download an embedded controller on a Windows-based system:

```
commload -v -c0 Cmdfile.prot
```

Be sure to replace the characters *prot* with your actual filename extension.

As shipped, the load command file contains instructions for loading controller zero. If you have a second controller installed in your system, you will need to create an additional load command file for that controller. The load command file looks something like this (for a UNIX based system):

```
BTYPE 0 PCI334_Ma
LOADCTLR 0 /usr/pti/Load/xstra.pci334.dfx.0 L_0 0x405e00 0x405e00
LOADCTLR 0 /usr/pti/Load/hif.pci334.dfx.0 L_1 0x410000 0x0
LOADCTLR 0 /usr/pti/Load/hdlcswap.mod.dfx.0 L_2 0x413e00 0x0
LOADCTLR 0 /usr/pti/Load/hdlc.pci334.dfx.0 L_3 0x414200 0x0
LOADCTLR 0 /usr/pti/Load/config.dfx.0 L_4 0x402000 0x0
INIT L_0
```

For a Windows-based system, the command file looks something like this:

```
BTYPE 0 PCI334_Ma
LOADCTLR 0 c:\pti\load\xstra.pci334.dfx.0 L_0 0x405e00 0x405e00
LOADCTLR 0 c:\pti\load\hif.pci334.dfx.0 L_1 0x410000 0x0
LOADCTLR 0 c:\pti\load\hdlcswap.mod.dfx.0 L_2 0x413e00 0x0
LOADCTLR 0 c:\pti\load\hdlc.pci334.dfx.0 L_3 0x414200 0x0
LOADCTLR 0 c:\pti\load\config.dfx.0 L_4 0x402000 0x0
INIT L_0
```

For a second controller, make a copy of the file and modify it as follows: On the first line, which begins with BTYPE, change the 0 that follows to a 1. On each line that begins with LOADCTLR, change the 0 that follows to a 1. An example is shown below.

```
BTYPE 1 PCI334_Ma
LOADCTLR 1 /usr/pti/Load/xstra.pci334.dfx.0 L_0 0x405e00 0x405e00
LOADCTLR 1 /usr/pti/Load/hif.pci334.dfx.0 L_1 0x410000 0x0
LOADCTLR 1 /usr/pti/Load/hdlcswap.mod.dfx.0 L_2 0x413e00 0x0
LOADCTLR 1 /usr/pti/Load/hdlc.pci334.dfx.0 L_3 0x414200 0x0
LOADCTLR 1 /usr/pti/Load/config.dfx.0 L_4 0x402000 0x0
INIT L_0
```

(For any additional controllers, repeat this procedure, but replace the zeroes with 2, 3 and so on.)

To download the second controller, use *commload* exactly as described above, but specify your new load command file name. For example:

```
commload -v Cmdfile.copy
```

To display the current versions the working PT environment and what is being loaded to the controller(s), the following **commload** command can be run:

```
commload -i Cmdfile.hdlc_pci334
```

Note, this example is using the specific command file. For this case, the output will be the following:

```
mps driver:  PTI Sbus/PCIbus Driver Version 3.5.7
PTI API Version 3.3.7
Controller 0:  PTI xSTRa Version 7.1.9
Controller 0:  PTI HIF Driver Version 4.2.6
Controller 0:  PTI HDLC QUICC Driver Version 5.1.1
```

Communications Server Update Program (commupdate)

The *commupdate* utility allows modification of the buffer pool configuration used on the controller or server.

The utility modifies one of the executable files that is loaded to the controller. This file is located in the PT *Load* directory and is called *config.prot.H* or *config.prot.n*, where *prot* is the same filename extension used for your download command file and the suffix *H* or *n* denotes a server installation (H) or the controller board instance (n) for embedded controllers. The program displays default responses in parentheses. To select the default, just press return.

To run *commupdate*, type the following:

```
commupdate
```

```
commupdate
Enter config filename below (e.g. /usr/pti/Load/config.hdlc.0)
Enter config filename:/usr/pti/Load/config.hdlc_mps800.H

MEMORY SIZE:                              16777216

Code/data/reserved areas:        455184
System memory:                   120196
Data/Message blocks:           14528040
Drivers/Stream Head:               2248
TCP memory:                      264448
TOTAL USED                                15370116
Estimated memory available                1407100
Communications Server Update Utility
1   Buffer Pool Configuration
2   Update Memory Size
3   MPS Server Specific
4   Save Configuration File Updates
0   Quit

Enter Menu Item (1): >
```

Modifying the Buffer Pool Configuration

With PT’s protocol software, a portion of memory on the controller is used for buffering data as it is transferred to and from the serial ports, and for messages traveling to and from your application. This memory is divided into several pools of fixed-size buffers, each pool having buffers of a different size. The default configuration (number of pools, buffer sizes and number of buffers) varies depending on the protocol, the platform type and the amount of on-board memory. Depending on the packet sizes used by your protocol, this default configuration may not be optimal for your application. You can configure between one and four pools, with any number of buffers in each pool. Buffers sizes may be any multiple of 16 bytes. The *commupdate* utility prints an error if the total memory required by your requested configuration exceeds the limit for your platform type.

Note: *The efficiency of memory usage is increased if you configure a number of small buffers (e.g. 64 bytes each) to be used for control messages, as well as larger buffers to hold data messages. “System Statistics Monitor Program (commstats),” on page 63 describes how it can be used to monitor buffer usage.*

Note: *Also keep in mind that some protocols will dynamically request memory for buffers other than those configured by the commupdate utility. Care must be taken to reserve a sufficient amount of memory for use by the protocol. In order to ascertain the amount of memory that is available, you should use the commstats utility to monitor memory usage under peak load conditions (with the maximum number of links enabled).*

```
Communications Server Update Utility

   1   Buffer Pool Configuration
   2   Update Memory Size
   3   MPS Server Specific
   4   Save Configuration File Updates
   0   Quit

Enter Menu Item (1): >  1
Current Configuration:
Class 1
   Size     64
   Number   14000
Class 2
   Size     528
   Number   8000
Class 3
   Size     1040
   Number   4000
Class 4
   Size     2064
   Number   2000

You may configure up to 4 classes of data blocks, where each class
contains blocks of a different size. You must configure at least one
class.

Enter number of data block classes (4) >

For each class, specify the block size and the number of blocks to
build. Configure the class with the smallest block size first, in order
up to the largest.

Note that buffer sizes will be rounded up to multiples of 16.
Class 1
Enter block size (64) >
Enter Number of blocks to build (14000) > 12000
Class 2
Enter block size (528) >
Enter Number of blocks to build (8000) > 2500
Class 3
Enter block size (1040) >
Enter Number of blocks to build (4000) > 3000
Class 4
Enter block size (2064) > 4112
Enter Number of blocks to build (2000) > 1000

New Configuration:
Class 1
   Size     64
   Number   12000
Class 2
```

```
   Size     528
   Number   2500
Class 3
   Size     1040
   Number   3000
Class 4
   Size     4112
   Number   1000

MEMORY SIZE:                           16777216

Code/data/reserved areas:        455184
System memory:                   120196
Data/Message blocks:           10060040
Drivers/Stream Head:               2248
TCP memory:                      264448
TOTAL USED                     10902116

Estimated memory available             5875100

   Communications Server Update Utility
       1   Buffer Pool Configuration
       2   Update Memory Size
       3   MPS Server Specific
       4   Save Configuration File Updates
       0   Quit

   Enter Menu Item (1): > 4

Do you really want to update ./Load/config.hdlc_mps800.H? (n)  > y

   Communications Server Update Utility

       1   Buffer Pool Configuration
       2   Update Memory Size
       3   MPS Server Specific
       4   Save Configuration File Updates
       0   Quit

       Enter Menu Item (1): > 0
```

Update the Memory Size

With some PT’s controllers, the amount of physical memory on the controller may be modified. The hardware manual that was supplied with the controller will provide the information required to alter the amount of on-board memory. This additional memory is made available for the protocol software and is used by the data and message blocks.

```
Communications Server Update Utility
```

```
   1   Buffer Pool Configuration
   2   Update Memory Size
   3   MPS Server Specific
   4   Save Configuration File Updates
   0   Quit

   Enter Menu Item (1): >  2

WARNING: If you modify the memory size, you MUST be sure that the
server/controller hardware has sufficient memory onboard!

Current memory size is 16 MB
Enter new memory size in MB (4, 8, 16, or 32):  > 16

MEMORY SIZE:                               16777216

Code/data/reserved areas:         455184
System memory:                    120196
Data/Message blocks:             10060040
Drivers/Stream Head:                2248
TCP memory:                       264448
TOTAL USED                                 10902116

Estimated memory available                 5875100

Communications Server Update Utility
   1   Buffer Pool Configuration
   2   Update Memory Size
   3   MPS Server Specific
   4   Save Configuration File Updates
   0   Quit

   Enter Menu Item (1): > 4

Do you really want to update ./Load/config.hdlc_mps800.H? (n)  > y

   Communications Server Update Utility

   1   Buffer Pool Configuration
   2   Update Memory Size
   3   MPS Server Specific
   4   Save Configuration File Updates
   0   Quit

   Enter Menu Item (1): > 0
```

Modifying the MPS Server-Specific Options

The MPS server-specific options allow for alterations of the default behavior of the MSP800 server.

1. Blink the USER LED allows the front panel USER LED may be used for a visual display, or ‘heartbeat’ that signifies the unit is operation. The LED will be illuminated in a 1 Hz 50% duty cycle.
2. Check FEC Status allows for the software to check the status of the Fast Ethernet Controller. Motorola has identified a condition where the FEC can hang when a packet > 2047 bytes arrives. The solution to this is to avoid a collision domain where such packets could be created. Enabling Check FEC Status will check the FEC with each heartbeat and reset the FEC and continue if required.
3. SNMP Support allows the user to disable SNMP if it is not required.

```
Communications Server Update Utility
   1   Buffer Pool Configuration
   2   Update Memory Size
   3   MPS Sever Specific
   4   Save Configuration File Updates
   0   Quit

   Enter Menu Item (1): 3

Currently these options are valid for the MPS800 servers only.
Blink the USER LED as a heart beat signal? [no]:  >
Check FEC Status? [no]:  >
SNMP Support? [yes]:  >

   Communications Server Update Utility

   1   Buffer Pool Configuration
   2   Update Memory Size
   3   MPS Server Specific
   4   Save Configuration File Updates
   0   Quit

Enter Menu Item (1): > 4

Do you really want to update ./Load/config.hdlc_mps800.H? (n)  > y

Communications Server Update Utility

   1   Buffer Pool Configuration
   2   Update Memory Size
   3   MPS Server Specific
   4   Save Configuration File Updates
   0   Quit

Enter Menu Item (1): > 0
```

System Statistics Monitor Program (commstats)

The *commstats* utility provides remote access to the buffer and memory usage statistics on an embedded controller or LAN server. This utility may be useful when desiring to tune factory configured buffer pools for better performance.

Synopsis

```
commstats [ -s server ] [ -S service ] [ -c ctrlrnum ] [ -I IPver ]
[ -N scopeID ] [-d] [-i] [-V]
```

Description

The *commstats* program uses the PT API to communicate with the controller (whether embedded or over a Local Area Network). For LAN servers, *commstats* allows any user on the same network the ability to query statistics information for any controller on a PT server. For embedded controllers, *commstats* allows the user the ability to monitor the statistics of any communications controller.

Options

-s *server*	This parameter identifies the target server. For a LAN-based server, this is the logical host name as specified in the client's *hosts* database, or an IP address specified in either IPv4 dotted decimal or IPv6 colon-hex notation. For a controller installed in a backplane, *server* is the pathname of the driver's cloneable special file. If this option is not specified, the server name defaults to */dev/ucsclone* for UNIX-based systems, and \\.*PTIXpqcc* for Windows-based systems.
-S *service*	For LAN-based servers, this parameter identifies the service name to be accessed, as specified in the client's *services* database. If this option is not specified, the service name defaults to *mps*. For a controller installed in a backplane, this parameter is not used.
-c *ctrlrnum*	Specify the controller to use. If this no controller is specified for the embedded case, controller 0 is used. If no controller is specified for the LAN case, controller 15 (Host board) is used.
-I *IPver*	For LAN-based NexusWare servers, this option specifies the type of IP socket (IPv4 or IPv6) that should be created for the connection to the NexusWare server. The following values are recognized (see the *MPSOPEN_IVP6_SOCKET_xxx* symbol definitions in <*/usr/pti/include/mpsipv6.h*>).

0	There is no preference for the socket type.
1	Create an IPv6 socket for the connection.
2	Try to create an IPv6 socket for the connection, but if it cannot be done then create an IPv4 socket instead.

	3	Try to create an IPv4 socket for the connection, but if it cannot be done then create an IPv6 socket instead.
	4	Create an IPv4 socket for the connection.

A default value of 0 (no preference for the socket type) is used when this option is omitted. This option is ignored for non-NexusWare LAN servers and embedded controllers.

`-N` *`scopeID`*	For LAN-based NexusWare servers, this option specifies the IPv6 scope identifier. This option is only recognized when an IPv6 socket is created for the connection to the server. The default value when this option is not specified depends on the operating system on which *commstats* is executing (see the *MPSOPEN_IPV6_SCOPEID_DFLT* symbol definition in </usr/pti/include/mpsipv6.h>): “eth0” for Linux, “hme0” for Solaris, and “8” for Windows. This option is ignored for non-NexusWare LAN servers, embedded controllers, and IPv4 sockets to a NexusWare server.
`-d`	Debug mode. Outputs detailed information about every stream currently active on the target controller or server. This option is provided as a debugging aid for software developers. Not valid for NexusWare Controllers.
`-i`	Controller identification (applies to embedded controllers only). This option causes *commstats* to display the type of controller and its geographic address (if NexusWare) Controller for every device identified by the PT host device driver during system startup. Also included in this report will be the version string of the current PT host device driver that is installed in the system.
-V	Displays the version string of the MPS-API with which the utility was built.

Note: *The number of buffers reported when using commstats will never be zero, since buffers are required on the controller/server to process the commstats request.*

For example, to request statistics from embedded controller:

```
commstats -c 0
```

To request statistics from the server *alpha*:

```
commstats -s alpha
```

Interpreting the System Statistics Report of a Non-NexusWare Platform

There are several factors governing the proper configuration of the buffer pools. There should be one pool defined of “smaller” buffers; typically, protocols require headers to be prepended to data messages, and these header buffers are usually of small size (under 64 bytes). Your provided software package may support multiple protocols, which may have different average frame sizes. An individual protocol may transfer frames of a consistent size, yet require a larger or smaller control frame to be transferred occasionally. Where supported, client requests for statistics reporting from the server may require yet another buffer size.

In addition to the buffer pool sizes, the rate at which these buffers are used must also be considered; *commstats* provides a snapshot of the current pool usage as well as the high and low watermarks for these pools. Analysis of this information can be useful in determining just how many of each buffer size should be configured to minimize the likelihood of an allocation request failing because all buffers of a given size are in use. The server will attempt to allocate a buffer from the appropriate data block pool, but if unable to obtain a buffer, will attempt to allocate the buffer from the next larger data block pool. Thus, failures to obtain buffers from a given pool may not indicate any loss of data if buffers of larger size are available to handle incoming data from the serial port.

PT protocols are designed to accept incoming data as long as sufficient memory is available for message storage, and, where supported, use protocol mechanisms to notify the remote station to refrain from sending more data if memory is unavailable to store data. If the client application does not issue read requests to the server in a timely manner, buffer pool exhaustion can also occur. Thus, *commstats* can be used as a troubleshooting tool during client application development. An example of the output from *commstats* follows, with an explanation of how to analyze it.

```
System Report for Server mps800, Controller: 0

dblk 1 (64):   current 4,      max     5,      tot     15000,  fail   0
dblk 2 (272):  current 0,      max     0,      tot     5000,   fail   0
dblk 3 (528):  current 1,      max     1,      tot     5000,   fail   0
dblk 4 (1040): current 0,      max     0,      tot     5000,   fail   0
total dblks:   current 5,      max     5,      tot     30000,  fail   0
total mblks:   current 5,      max     5,      tot     30000,  fail   0
memory avail:  current 4436688,        min     4436688
electrical interface per port:
        0 - RS232       1 - RS232       2 - RS232       3 - RS232
        4 - RS232       5 - RS232       6 - RS232       7 - RS232
```

The first line includes the host name given to the server, *mps800.* For an embedded controller, the line will indicate the name of the device (normally */dev/ucsclone*) and the controller number.

The next two lines apply to the buffer pools used by the TCP/IP stack on a PT LAN-based server (e.g., an MPS300/600 and MPS800). These lines do not apply to, and therefore are not shown for, embedded or slave controllers. The *mbuflist* pool is used by the TCP/IP stack for both sending and receiving data, and for other internal memory use. It reports the *current* number of available buffers, the low watermark for available buffers (*min*), and the *max*imum number of buffers that are available. The *mcllist* is a pool of buffers for use by the TCP/IP stack for transferring data from the PT server to the client, and reports the *current* number of available buffers, the low watermark for available buffers (*min*), and the *max*imum number of buffers that are available. As more sockets are opened or smaller frame sizes are configured, more buffers are allocated from this pool.

The remaining lines of the report apply to the STREAMS message buffers that are used by protocol and serial port driver software. The following is a line-by-line explanation of this portion of the report:

dblk	These lines report the usage for Data Block buffer pools. The Data Block provides storage for the message data transferred over the serial ports and between the TCP/IP software and the protocol software. The PT software allows the definition and configuration of up to four buffer pools. The size and number of buffers in each pool are configurable. See "Communications Server Update Program (commupdate)," on page 57 for an explanation of how to modify the factory settings of the buffer pools. Each line reports the size (in bytes) of each pool (in parentheses), the *current* number of Data Blocks in use, the high watermark (*max*) of allocated Data Blocks, the *tot*al number of buffers in the pool, and the number of times that an allocation request to this Data Block pool has been issued and has *fail*ed. Note that each *dblk* pool is displayed in order of ascending size of the frames it supports. If the *dblk* line for a pool displays failures, this is an indication that the buffer allocation was attempted from the following *dblk* pool.
total dblks	This line is a summation of the four lines that precede it in the report, and represents the aggregate Data Block pool usage for this controller. This line reports the *current* number of Data Blocks allocated by the controller, the high watermark (*max*) of allocated Data Blocks, the *tot*al number of available Data Blocks, and the number of times that an allocation request to any Data Block pool has been issued and has *fail*ed.
total mblks	This line reports the Message Block pool usage. The Message Block is a structure that is allocated for each Data Block available to the system; it is a control structure that provides linkage to the Data Block and to other Message Blocks on the same queue. The total number of Message Blocks available to the controller is equal to the total number of Data Blocks configured for the controller (i.e., the aggregate number of blocks in the four defined Data Block pools - see below). This line reports the *current* number of Message Blocks allocated by the controller, the high watermark (*max*) of allocated Message Blocks, the *tot*al number of available Message Blocks, and the number of times that an allocation request has been issued that has *failed*.
memory avail	This line reports the *current* amount of memory available and the low watermark (*min*) since the system has been operational.
electrical interface	These lines apply only if the server or controller has a serial interface. If so, the electrical interface is reported for each serial port.

A few guidelines

To determine the factory settings of the *dblk* pools, use "Communications Server Update Program (commupdate)," on page 57 to display the current settings. The *dblk* pools should only be modified if performance problems occur. You should always back up the configuration files before modifying the settings so that you can return to the factory settings.

If one or more of the *dblk* pools in the report indicate under-utilization of that pool (for instance, the *current* and *max* lines are both zero after extensive use of the controller), you might consider modifying the *dblk* pool definitions to provide more buffers of a different size *dblk* pool, and eliminate the *dblk* pool that is not being used at all.

If a *dblk* pool exhibits large numbers of failures, you may need to evaluate the frame size you are configuring for the protocol and comparing it with the current *dblk* pool definitions. If the frame size is large, the *dblk* pool of the largest number of bytes must be able to accommodate its size plus any protocol overhead.

The number of buffers to configure for a given *dblk* pool must also be determined by running the protocol for an extended period and then examining the *max* number on the report lines. If that number is much smaller than the total number configured, consider reducing the number of those buffers and providing more buffers to another *dblk* pool.

Interpreting the System Statistics Report of a NexusWare Platform

For the NexusWare controller, the *commstats* utility is used to get a simple snapshot of the system. Below is an example of the output from running commstats:

```
System Report for Server 192.139.241.146, Controller: 0

Product code for kernel: PTI Kernel Image: 810p0574.20
The date is Wed Apr 14 07:29:47 2004
Current system kernel version 2.4.18
System has been alive 2 hrs 3 mins 27 secs
Total system memory   122265600 bytes
Free system memory    90238976 bytes
Used system memory    32026624 bytes
Running Kernel 1
Vendor Id: 0x1214, Device Id: 0x3580
Board serial number: 12345678
Geographic Address: 4
IP device information:
   eth0 -> addr:192.139.241.146  Mask:255.255.255.0
electrical interface per port:
        0 - RS232       1 - RS232       2 - RS232       3 - RS232
        4 - RS232       5 - RS232       6 - RS232       7 - RS232
```

LAN Server Reset Utility (commreset)

The *commreset* utility provides a method to reset a LAN server unit from a client computer. Note that for the reset function to succeed, the server must be in an operable state. The user defined flash prom may be configured to disallow this functionality if security is a concern (See “Network Reset,” on page 36 for further information).

There are two types of reset: a *hard* reset (default) which causes the server to react as if it has been powered down and back up, and a *soft* reset which causes the server to close all active connections in an orderly fashion.

Synopsis

```
commreset [ -f ] [ -v ] [ -s server ] [ server ]
```

Description

The *commreset* program uses the PT API to communicate with the LAN server, and issues a *RESET* (hard reset) or *SOFTRESET* (soft reset) ioctl command. A *RESET* command will cause the entire server to reboot and initiate a software download. A *SOFTRESET* command will cause all active connections to the server to be closed immediately, without requiring the server to be rebooted and downloaded. Regardless of the type of reset, all client applications will have to reconnect once the server is back in an operational state.

The server name can either be specified as an option (-s), or as a parameter to *commreset*.

Options

`-f`	This parameter requests a soft reset of the server only, i.e. all connections to the server are closed. If this parameter is not specified (default), a hard reset of the server will be performed.
`-s` *`server`*	This parameter identifies the target server. This is the logical host name as specified in the client's *hosts* database.
`-S` *`service`*	This parameter identifies the service name to be accessed, as specified in the client's *services* database. If this option is not specified, the service name defaults to *mps*.
`-v`	This option causes commreset to display status information during execution of the reset request.

For example, to reset server *alpha*:

```
commreset alpha   -or-  commreset -s alpha
```

Appendix A

Network Load Procedures

Overview

After completing the hardware and software installation and the configuration procedures described in this document, the server is ready to be loaded with its runtime software. To initiate the download procedure, you can select *X* from the configuration menu, depress the reset push-button on the server's front panel (MPS 3000-family servers only), or cycle the power off and on. Any of these actions cause the server's PROM-resident boot code to automatically begin the load sequence by attempting to contact, over the Ethernet, the first Internet address in the list of clients configured to load the server. The configuration information is either entered at the console or retrieved from the Ethernet network using the Bootstrap Protocol (MPS 300 or 600 servers only) and the Trivial File Transfer Protocol. (After reset or power-on, there is an initial delay of five seconds to allow access to the configuration menu before control is transferred to the boot code.)

The following topics are covered in this appendix:

- “The Load Procedure,” on page 69
- “MPS 3000-Family Front-Panel Display,” on page 70
- “MPS 300/600 Front-Panel Display,” on page 71
- “Using the Console Port to Diagnose a Problem,” on page 71
- “SNMP Implementation on an MPS Server,” on page 73

The Load Procedure

In the first phase of the load procedure, the server uses the Address Resolution Protocol (ARP) to contact the client. The server broadcasts an ARP request containing the client's Internet address. The client responds with a packet containing its Ethernet address. This exchange informs the server of the Ethernet address of the client. If a client does not respond within a timeout period, the server moves to the next entry in the list of clients configured to load the server.

Once the server has established contact with a client, it uses the Trivial File Transfer Protocol (TFTP) to request the load command file from the client. It supplies the client with the full pathname of the file, which you stored in the server's flash PROM as part of the configuration procedures. Using TFTP, the client transfers the file to the server a block at a time. If the client cannot locate the file, it replies with an error packet. In this case, the server attempts the load from the next client in the list

The load command file contains an entry for each executable file to be loaded, and includes the full pathname of each file. In the third and final phase of the load procedure, the server uses this information to request each required executable file from the client. These files are also transferred using TFTP. The last entry in the load command file provides startup information, which the server uses to transfer control to the runtime system. If an error occurs, generally due to a missing executable file or a corrupted load command file, the server aborts the load procedure. It then locates the next client in the list and broadcasts another ARP message to restart the load procedure.

If the server cannot successfully complete the load from any of the configured clients, it returns to the top of the list and retries the procedure with each client in turn. This process continues until the server is downloaded successfully from one of the configured clients.

MPS 3000-Family Front-Panel Display

For a server in the MPS 3000 family, the *RUN* LED on the server's front panel is used to indicate status. When the boot code initiates the load procedure, the LED begins to flash. When download is complete, the LED no longer flashes, and should remain illuminated until the server is reset or powered off. If the server is not able to load successfully, the *RUN* LED continues to flash indefinitely.

The load sequence should take 10 to 15 seconds if the server is able to contact and obtain its runtime system from the first client in the list. The sequence can require much more time if the server must attempt to load from a number of clients before meeting with success.

MPS 300/600 Front-Panel Display

The MPS 300 and MPS 600 servers have a two-line LCD display on the front panel. When the server is powered on, the following message appears on the display:

```
PTI MPS
Waiting
```

When the boot code initiates the load procedure, the “Waiting” message is replaced with “Loading”. As the server obtains each executable file, it displays a character (beginning with “A” and continuing through the alphabet) on the second line following the “Loading” message. For example:

```
PTI MPS
Loading   C
```

When download is complete, the message changes to:

```
PTI MPS
Load Complete
```

If the server is not able to load successfully, the “Loading” message may remain on the LCD display indefinitely, or an error message may be displayed.

The load sequence should take 10 to 15 seconds if the server is able to contact and obtain its runtime system from the first client in the list. The sequence can require much more time if the server must attempt to load from a number of clients before meeting with success.

Using the Console Port to Diagnose a Problem

The boot code prints a number of messages on the console port as the load proceeds, describing the sequence of events and any errors that may occur. If your server is not able to load successfully (the *RUN* LED continues to flash indefinitely or the LCD display indicates an error condition), you should attach a terminal to the console port (as described in “Configuring the Server,” on page 32) to observe the output. In the example below, the server's boot code attempts to load from three clients. The first does not respond at all; the second responds but is not able to supply one of the required files; and the third completes the load successfully.

```
Sending ARP Request To 100.0.0.33
No ARP Reply

Sending ARP Request To 100.0.0.8
Received ARP Reply, Requesting Boot File /usr/pti/Load/Cmdfile.x25
File Obtained
Requesting File /usr/pti/Load/UC_tcp.ucmps.x25.H
File Obtained
Requesting File /usr/pti/Load/wan.ucmps.x25.H
File Not Found

Sending ARP Request To 100.0.0.2
Received ARP Reply, Requesting Boot File /usr/pti/Load/Cmdfile.x25
File Obtained
Requesting File /usr/pti/Load/UC_tcp.ucmps.x25.H
```

```
File Obtained
Requesting File /usr/pti/Load/wan.ucmps.x25.H
File Obtained
Requesting File /usr/pti/Load/hdlc.ucmps.x25.H
File Obtained
Requesting File /usr/pti/Load/lapb.mod.x25.H
File Obtained
Requesting File /usr/pti/Load/plp.mod.x25.H
File Obtained
Requesting File /usr/pti/Load/reset.mod.x25.H
File Obtained
Requesting File /usr/pti/Load/xSTRa.ucmps.x25.H
File Obtained
Requesting File /usr/pti/Load/config.x25.H
File Obtained
Initializing Module L_6
Init L_6 at 200BBC40
```

When BOOTP is configured on, the console log begins with messages regarding the BOOTP protocol message exchange and processing the configuration file. Each line from the configuration file is copied to the console device as it is processed. The content of this file is copied to flash PROM and then booting proceeds as shown above. The following is an example of the beginning of the console output that appears when the server is configured to use the BOOTP protocol. Following this output is output from the ARP request as described in the previous example.

```
Broadcast BOOTP Request: xid 1
BOOTP Reply: size 308, htype 1, op 2, xid 1
Bootp server: 206.251.239.140
Bootp client: 206.251.239.143
Gateway: 0.0.0.0
Remote file: /usr/pti/mps600.config
Requesting configuration file /usr/pti/msp600.config
Configuration file read complete.
process#
process# Remote configuration file for MPS diamondback.
process# The full path name of this file is an entry
process# in the /usr/etc/bootptab file.
process#
process NAME      diamondback1
process CLIENT    206.251.239.148
process FILE      /usr/pti/Cmdfile.x25
process
process SERVER    206.251.239.143  # MPS internet address
process PORTS     48879 0 0        # Must match /etc/services
process DIAG      n
process#
process# Enable auto-reset
process USER      0x00       0x00
process USER      0x01       0x00
process USER      0x02       0xA0
process USER      0x03       0xD0
process
process ROUTE     204.250.22.30    206.251.239.129 # gateway
process #Define default gateway
process ROUTE     255.255.255.255  206.251.239.129
Saving configuration... Updating DBC bytes at 200EF110

Sending ARP Request to 206.251.239.148
```

SNMP Implementation on an MPS Server

All configurable parameters saved on the non-volatile flash PROM on MPS800 servers can be viewed or changed using SNMP. A MIB Browser or SNMP Network Manager can query or set SNMP variables on the MPS Server by way of the MPS Server MIB once the Server has been Downloaded with its PT software. Most parameters configurable from the MPS Server Console can be configured in this way. The following lists the MPS Server MIB variable and its corresponding console entry:

MPSsystem

MPSsysName	Server's Logical Name
MPScmdFile	Download Command File Pathname
MPSbootp	Remote Configuration
MPSsntpServAddr	SNTP Configuration - NTP Server Address
MPSsntpFreq	SNTP Configuration - Poll Frequency Code
MPSreboot	Resets MPS Box (not a console entry)

MPSip

MPSipClientAddr1	Client's Internet Address (First entry)
MPSipClientAddr2	Client's Internet Address (Second entry)
MPSipServerAddr	Server's Internet Address
MPSiptcpPortNum	Server's (Listen) Port Numbers - Enter TCP Port
MPSipudpPortNum	Server's (Listen) Port Numbers - Enter UD Port
MPSipBroadPortNum	Server's (Listen) Port Numbers - Enter Broad cast Port
MPSipNetAddr1	Routing Configuration - Modify route table - Network1
MPSipNetAddr2	Routing Configuration - Modify route table - Network2
MPSipNetAddr3	Routing Configuration - Modify route table - Network3
MPSipGateAddr1	Routing Configuration - Modify route table - Gateway1
MPSipGateAddr2	Routing Configuration - Modify route table - Gateway2

MPSipGateAddr3	Routing Configuration - Modify route table - Gateway3