

# BlueCat Linux Target Support Guide

---

DOC-0372-00

for Radstone PPC4A VME Boards

Product names mentioned in *BlueCat Linux Target Support Guide for Radstone PPC4A VME Boards* are trademarks of their respective manufacturers and are used here for identification purposes only.

Copyright ©1987-2001, LynuxWorks, Inc. All rights reserved.  
U.S. Patents 5,469,571; 5,594,903

Printed in the United States of America.

All rights reserved. No part of *BlueCat Linux Target Support Guide for Radstone PPC4A VME Boards* may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photographic, magnetic, or otherwise, without the prior written permission of LynuxWorks, Inc.

LynuxWorks, Inc. makes no representations, express or implied, with respect to this documentation or the software it describes, including (with no limitation) any implied warranties of utility or fitness for any particular purpose; all such warranties are expressly disclaimed. Neither LynuxWorks, Inc., nor its distributors, nor its dealers shall be liable for any indirect, incidental, or consequential damages under any circumstances.

(The exclusion of implied warranties may not apply in all cases under some statutes, and thus the above exclusion may not apply. This warranty provides the purchaser with specific legal rights. There may be other purchaser rights which vary from state to state within the United States of America.)

---

# *Contents*

---

<b>CHAPTER 1</b>	<b>OVERVIEW .....</b>	<b>1</b>
	Chapter Outline .....	1
	Additional Documentation .....	2
<b>CHAPTER 2</b>	<b>BOOTING BLUECAT LINUX ON PPC4A TARGETS .....</b>	<b>3</b>
	Installing the PPC4A Distribution .....	3
	Installing the Distribution on a Linux Host .....	3
	Installing the PPC4A Source on a Linux Host .....	5
	Installing the Distribution on a Windows Host .....	6
	Installing the PPC4A Source on a Windows Host .....	8
	Setting up the PPC4A Board Hardware .....	9
	Configuring the Kernel .....	9
	Using a Permedia-II Graphic Mezzanine Card .....	10
	Redirecting Console Messages to COM1 on the PPC4A Board .....	11
	Redirecting Console Messages to PPC4A VGA Console .....	12
	Booting BlueCat Linux on a PPC4A Target Board .....	12
	Booting from a SCSI Disk .....	13
	Booting over a Network .....	14
	Setting up a TFTP Server on a Windows Host .....	15
	Setting up Softlinks to Download Demo Systems over a TFTP Server .....	15
	Booting from Target Flash Memory .....	15
	Autobooting from Target Flash Memory .....	17
	Uninstallation .....	18

<b>CHAPTER 3</b>	<b>KERNEL CONFIGURATION PARAMETERS .....</b>	<b>21</b>
<b>CHAPTER 4</b>	<b>SUPPORTED DEMO SYSTEMS .....</b>	<b>35</b>
	Running BlueCat Linux Demos Systems on PPC4A .....	37
	Running an X-Server-Based Demo System .....	37
	Frame Buffer Support for using the X-Server .....	38
	Running a Demo from Hard Disk .....	39
	Using xdemo1 .....	42
	Enabling glx Support in the X-Server .....	43
	Testing glx Clients on the Target Board .....	44
	Running the gdb Demo under X-Server .....	45
	Running GDB in Graphical Mode .....	45
	Running a Graphical GDB Session using Data Display .....	
	Debugger .....	47
	nfsroot Demo System .....	49
	kdbg Demo System .....	50
	UVME Demo System – uvme .....	50
	Setting Parameters .....	51
	Running the uvme Demo .....	52
	BlueCat Linux Driver for Interfacing with the Tundra Universe-II PCI-VME Bridge .....	53
	Creating a New Specification File .....	57
<b>CHAPTER 5</b>	<b>SUPPORTED DEVICE DRIVERS .....</b>	<b>59</b>
	Driver for the Network Chipset .....	60
<b>APPENDIX A</b>	<b>INSTALL_PPC4A.SH .....</b>	<b>61</b>
	Command Reference .....	61
<b>APPENDIX B</b>	<b>UNINSTALL_PPC4A.SH .....</b>	<b>65</b>
	Command Reference .....	65

---

---

<b>APPENDIX C</b>	<b>X-SERVER REGENERATION .....</b>	<b>67</b>
	Procedure .....	67



The *BlueCat Linux Target Support Guide (TSG) for Radstone PPC4A VME-based PowerPC Boards* provides information about the BlueCat Linux Target Support Package (TSP) for PPC4A boards. BlueCat Linux supports the PPC4A-750 board with single MPC750 processors—Core Frequency upto 366 MHz and Bus Clock frequencies upto 100 MHz.

The PPC4A Target Support Package (TSP) contains all the binary and source files necessary to build and run a customized BlueCat Linux embedded system on the PPC4A target board. These files are:

- `ppc4a_linux_bin.tgz`
- `ppc4a_linux_src.tgz`
- `ppc4a_win_bin.tgz`
- `ppc4a_win_src.tgz`
- `install_ppc4a.sh`

Throughout this TSG, the board is referred to as the PPC4A or the target board, and the TSP as the “ppc4a.” Unless otherwise specified, this TSG covers all details related to Linux and Windows cross development hosts.

---

## Chapter Outline

- **Chapter 1** contains an overview of this TSG’s individual chapters, a list of TSP files, and notes on additional documentation.
- **Chapter 2** describes procedures for downloading/copying BlueCat Linux and booting it from the PPC4A target board.

- **Chapter 3** contains information about the configuration of the prebuilt BlueCat Linux kernel contained in the ppc4a TSP, as well as a list of parameters and options for kernel reconfiguration.
- **Chapter 4** lists and describes the BlueCat Linux demo systems supported by the ppc4a TSP.
- **Chapter 5** contains a list of supported device drivers and their requirements.

---

## Additional Documentation

For additional information refer to the following manuals:

- *Radstone SilverChip (PPC Boot Firmware) Manual*, Issue 5, Radstone Technology Publication No. RT5078
- *Radstone PPC4A Manual*, Rev. A, Radstone Technology Publication No. PPC4-0HH
- <http://www.radstone.co.uk> (PDF document for the Permedia 2 PCI Graphics Accelerator card)

# *Booting BlueCat Linux on PPC4A Targets*

This chapter describes the BlueCat Linux installation and boot procedure on the Radstone PPC4A VME board.

---

## Installing the PPC4A Distribution

Before booting BlueCat Linux on the PPC4A target board, it must first be correctly installed on a Linux or a Windows cross development host.

### Installing the Distribution on a Linux Host

*The installation of PPC4A binary is performed using the base release of BlueCat Linux (pci\_mcp750) for MCP750 target boards.*

The PPC4A binary distribution for Linux is provided as a tar file, `ppc4a_linux_bin.tgz`, containing the following files:

- `demo_trg-cpci_mcp750-1.0.-1.ppc.rpm`
- `glut_trg-3.7-1.ppc.rpm`
- `kernel_trg-bcboot-2.2.12-1.ppc.rpm`
- `kernel_trg-doc-2.2.12-1.ppc.rpm`
- `kernel_trg-headers-2.2.12-1.ppc.rpm`
- `kernel_trg-source-2.2.12-1.ppc.rpm`
- `Mesa_trg-3.4-1.ppc.rpm`
- `XFree86_trg-41-1.0.ppc.rpm`
- `mkboot_cdt-1.0-1.i386.rpm`
- `mkboot_trg-1.0-1.ppc.rpm`

- flex\_trg-2.5.4a-1.ppc.rpm

Use the following steps to install the PPC4A binary distribution from the tar file on a Linux host:

1. Install the BlueCat Linux base distribution (cpci\_mcp750) on the Linux cross development host (see “Installing BlueCat Linux” in Chapter 1 of the *BlueCat Linux User’s Guide* for detailed instructions).

2. The following files from the ppc4a TSP are required:

- install\_ppc4a.sh
- ppc4a\_linux\_bin.tgz

3. Change to the temporary directory where the binary files for the ppc4a TSP have been stored. For example,

```
# cd /home/temp
```

It is assumed that the /home/temp directory contains the files install\_ppc4a.sh and ppc4a\_linux\_bin.tgz required for installation.

4. Run the install program provided:

```
# ./install_ppc4a.sh -d/home/BC
```

This execution of the install\_ppc4a.sh script assumes that the base distribution (cpci\_mcp750) has been installed in /home/BC. This installs the ppc4a TSP in /home/BC and generates the uninstall\_ppc4a.sh and SETUP.sh files in the /home/BC directory.

This command also retains all the base distribution demos, which are not supported by the ppc4a TSP by default.

For more information on install\_ppc4a.sh, see Appendix A.

5. When installation is complete, the following message is displayed:

```
Installation Complete...Execute .SETUP.sh
```

6. The screen output above is a prompt to change to the BlueCat Linux installation directory and execute the SETUP.sh script:

```
# cd /home/BC
# . SETUP.sh
```

---

**NOTE:** *If required, the installed distribution can be removed using the `uninstall_ppc4a.sh` script file. Refer to Appendix B for details.*

---

## Installing the PPC4A Source on a Linux Host

Use the following steps to install a PPC4A source RPM from a tar file on a Linux cross development host:

1. Get a copy of the Linux source tar file, `ppc4a_linux_src.tgz`.

The tar file `ppc4a_linux_src.tgz` has following source RPM files:

```
- demo_trg-cpci_mcp750-1.0.-1src.rpm
- glut_trg-3.7-1.src.rpm
- kernel_trg-bcboot-2.2.12-1.src.rpm
- Mesa_trg-3.4-1.nosrc.rpm
- XFree86_trg-41-1.0.nosrc.rpm
```

2. Copy the tar file to a temporary folder, such as `/tmp/ppc4a`.

```
# cp ppc4a_linux_src.tgz /tmp/ppc4a
```

3. Change to the temporary folder, execute the following commands:

```
# cd /tmp/ppc4a
# tar -xvzf ppc4a_linux_src.tgz
```

Upon completion of the command, the source RPMs listed above are placed in the `/tmp/ppc4a` directory.

4. Change to the directory in which BlueCat Linux distribution for PPC4A has been installed and enable the BlueCat Linux environment (if not already enabled).

```
# cd /home/BC
# . SETUP.sh
```

5. Install a source RPM file with the `rpm` command as follows:

```
# rpm -i /tmp/ppc4a/\
kernel_trg-2.2.12-1.src.rpm
```

Upon completion of the command, the `kernel_trg.spec` file (the RPM specification file) is placed in the `$BLUECAT_PREFIX/cdt/src/bluecat/SPECS` directory and the corresponding source tar files are placed in the `$BLUECAT_PREFIX/cdt/src/bluecat/SOURCES` directory.

For further details, refer to the section “Installing Sources of BlueCat Linux RPM Packages” in Chapter 1 of the *BlueCat Linux User’s Guide*

Any source RPM provided can be installed in a similar fashion.

## Installing the Distribution on a Windows Host

*The installation of the BlueCat Linux binary files for the PPC4A target board is performed using the base release of BlueCat Linux for Windows for MCP750 boards.*

The binary distribution of PPC4A for Windows cross development hosts is distributed as a tar file, `ppc4a_win_bin.tgz`, which contains the following files:

- `demo_trg-cpci_mcp750-1.0.-1.ppc.rpm`
- `glut_trg-3.7-1.ppc.rpm`
- `kernel_trg-bcboot-2.2.12-1.ppc.rpm`
- `kernel_trg-doc-2.2.12-1.ppc.rpm`
- `kernel_trg-headers-2.2.12-1.ppc.rpm`
- `kernel_trg-source-2.2.12-1.ppc.rpm`
- `Mesa_trg-3.4-1.ppc.rpm`
- `XFree86_trg-41-1.0.ppc.rpm`
- `mkboot_cdt-1.0-1.i386.rpm`
- `mkboot_trg-1.0-1.ppc.rpm`
- `flex_trg-2.5.4a-1.ppc.rpm`

Use the following procedure to install the BlueCat Linux binaries for the PPC4A target board on a Windows cross development host.

1. Install the BlueCat Linux base distribution (cpci\_mcp750) on a Windows host (see “Installing BlueCat Linux” in Chapter 1 of the *BlueCat Linux User’s Guide* for detailed instructions.)
2. The following files from the ppc4a TSP are required for installation of the BlueCat Linux binary on a Windows host:

- install\_ppc4a.sh
- ppc4a\_win\_bin.tgz

3. Open a `bash` window by running the `cygwin.bat` script, which comes as part of the base (cpci\_mcp750) distribution.
4. Change to the temporary directory where the binary files for the ppc4a TSP are stored. For example,

```
# cd /home/temp
```

5. Run the `install_ppc4a.sh` program provided as follows:

```
# ./install_ppc4a.sh -d/home/BC
```

Assuming that the base (cpci\_mcp750) has been installed in `/home/BC`, this command installs the PPC4A binaries in `/home/BC` and generates and copies the `uninstall_ppc4a.sh` and `SETUP.sh` files to `/home/BC`.

The `install_ppc4a.sh` script run with the options above also retains the base distribution (for MCP750 boards) demo systems, which are not supported by the ppc4a TSP by default.

For more information about `install_ppc4a.sh`, see Appendix A.

6. When installation is complete, the following message is displayed:

```
Installation Complete...Execute .SETUP.sh
```

7. The screen output above is a prompt to change to the BlueCat Linux installation directory and execute the `SETUP.sh` script (if not already under the BlueCat Linux environment):

```
# cd /home/BC
# . SETUP.sh
```

---

**NOTE:** *If required, the distribution that has been installed can be removed using the `uninstall_ppc4a.sh` script in the `/home/BC` directory. Refer to [Appendix B](#) for details.*

---

## Installing the PPC4A Source on a Windows Host

Use the following steps to install the PPC4A source distribution from the tar file, `ppc4a_win_src.tgz`, on a Windows host:

1. Get a copy of the tar file provided for the Windows host, `ppc4a_win_src.tgz`. The tar file has the following source RPM files:
  - `demo_trg-cpci_mcp750-1.0.-1src.rpm`
  - `glut_trg-3.7-1.src.rpm`
  - `kernel_trg-bcboot-2.2.12-1.src.rpm`
  - `Mesa_trg-3.4-1.nosrc.rpm`
  - `XFree86_trg-41-1.0.nosrc.rpm`
2. Copy the tar file to a temporary folder, such as `/tmp/ppc4a`.

```
# cp ppc4a_win_src.tgz /tmp/ppc4a
```
3. Change to the temporary folder.

```
# cd /tmp/ppc4a
```
4. Execute the following command:

```
# tar -xvzf ppc4a_win_src.tgz
```

Upon completion of the command, the source RPMs listed above are placed in the `/tmp/ppc4a` directory.
5. Change to the directory in which BlueCat Linux binary distribution for ppc4a has been installed and enable the BlueCat Linux environment (if not already enabled).

```
# cd /home/BC
# . SETUP.sh
```
6. Install an RPM file as follows:

```
# rpm -i /tmp/ppc4a/\
kernel_trg-2.2.12-1.src.rpm
```

Upon completion of the command, the `kernel_trg.spec` file (the RPM specification file) is placed in the `$BLUECAT_PREFIX/cdt/src/bluecat/SPECS` directory and the corresponding source tar files are placed in the `$BLUECAT_PREFIX/cdt/src/bluecat/SOURCES` directory.

For further details, refer to the section “Installing Sources of BlueCat Linux RPM Packages” in Chapter 1 of the *BlueCat Linux User's Guide*

Any source RPM file provided can be installed in a similar fashion.

---

## Setting up the PPC4A Board Hardware

This sections details the hardware setup required on the PPC4A target board before booting BlueCat Linux on it.

---

**NOTE:** *The BlueCat Linux kernel configuration menu provides the choice of making the network driver work with either the 10BaseT or 100BaseT network. (By default the network-based demos provided in BlueCat Linux are configured for 10BaseT networks.)*

---

### Configuring the Kernel

To configure the kernel for working on a 100BaseT or 10BaseT network, use the following steps:

1. To enable the 100BaseT (half-duplex) network support as part of BlueCat Linux kernel startup, run `make xconfig`.
2. Click on:

```
Ethernet (10 or 100Mbit) ->
Generic DECchip $ DIGITAL
EtherWorks PCI/EISA -> Force 100BaseT media
```

If `y` is selected for this option, the `CONFIG_BLUECAT_DE4X5_PARM` variable is defined and the default network becomes a 100BaseT network.

If `n` is selected, the `CONFIG_BLUECAT_DE4X5_PARM` variable is undefined and the default network becomes a 10BaseT network.

- A suboption specifying full-duplex or half-duplex (Half duplex 100 Mbit media) is provided.

If `y` is selected for this suboption, a `CONFIG_BLUECAT_DE4X5_PARM_FDX` variable is defined and the network is configured for half-duplex.

If `n` is selected, a full duplex option is set. If the kernel is generated with the above configuration, it sets the default network as 100BaseT media.

## Using a Permedia-II Graphic Mezzanine Card

For the Permedia-II Graphic Mezzanine Card (PMCGA1) to work properly on a PPC4A target board, the following procedure must be executed before the card is inserted in a PPC4A board. Otherwise, the SilverChip firmware will not start up.

1. Ensure that the PMCGA1 board is not inserted in the PPC4A board.
2. Connect a serial console to the `COM1` serial port of the PPC4A target board to see the SilverChip monitor messages.
3. On entering the SilverChip monitor mode, set the following flag:

```
> set UNIVERSE_AUTOBAR 1
```

---

**NOTES:** *If the parameter `UNIVERSE_AUTOBAR` is set, the SilverChip firmware ignores the PCI I/O space request. This is needed to make the Universe-II compatible with the PMCGA1 card.*

*The Universe normally requests 4Kbytes of PCI I/O space. This is an illegal request when legacy PCI devices are present on the bus. Devices should request up to 256 bytes only, and the PCI configuration code should allocate 256 byte blocks on 1Kbyte boundaries. This requirement is to allow devices that have their address space aliased (i.e., duplicated) to several different areas of the PCI I/O space. The PMCGA1 board has this PCI address aliasing and, therefore, will not work if the Universe-II is allowed to allocate 4Kbytes of I/O space.*

---

## Redirecting Console Messages to COM1 on the PPC4A Board

If the PPC4A target board does not have a Graphic Mezzanine board (PMCGA1) connected, proceed as follows to enable display of all console messages on the monitor:

1. Connect a null modem cable between the `COM1` port of the PPC4A target board and the cross development host.
2. Run any terminal emulation software such as `kermit` (downloadable from the Web) or `/usr/bin/minicom` provided with the Red Hat Linux distribution.
3. In the SilverChip firmware:

```
> set USESERIAL 1
```
4. For the appropriate demo system (note that not all demos can be run with the console as the `COM1` serial port), make sure the following kernel options are set by running `make xconfig`:

```
General setup -> Support for VGA console -> N
General setup -> Support for frame buffer devices -> N

Character devices -> Virtual terminal -> N
Character devices -> Support for console on virtual terminal -> N
```

Character devices -> Standard/generic (dump)  
serial support -> Y  
Character devices -> Support for console on  
serial port -> Y  
Character devices -> Extended dump serial  
driver options -> N

## Redirecting Console Messages to PPC4A VGA Console

If the PPC4A board has a Graphic Mezzanine board (PMCGA1) connected to it, proceed as follows to enable display of all console messages on the monitor:

1. If console messages have been redirected to go only to the serial console, disable this option using the following SilverChip command:

```
> unset USESERIAL
```

2. For the appropriate demo system, ensure that the following kernel options are set by running `make xconfig`:

General setup -> Support for VGA console -> Y  
General setup -> Support for frame buffer  
devices -> N

Character devices -> Virtual terminal -> Y  
Character devices -> Support for console on  
virtual terminal -> N  
Character devices -> Standard/generic (dump)  
serial support -> Y  
Character devices -> Support for console on  
serial port -> Y  
Character devices -> Extended dump serial  
driver options -> N

---

## Booting BlueCat Linux on a PPC4A Target Board

Several boot options for BlueCat Linux are available on the PPC4A target board. Each option and its procedure is detailed below.

---

## Booting from a SCSI Disk

Use the following procedure to boot BlueCat Linux on a PPC4A target board from a SCSI disk attached to the target board:

---

**NOTE:** *For a BlueCat Linux embedded system to boot successfully from a hard disk, a hardware device driver for the specific SCSI disk controller must be configured in the kernel.*

---

1. Install a BlueCat Linux embedded system on a SCSI disk. (See “Booting from a Hard Disk on a PowerPC Target” in Chapter 2 of the *BlueCat Linux User's Guide* for details.)
2. Enter into Radstone's SilverChip (PPC Boot Firmware) monitor mode by entering `m` at the startup screen.
3. When in the SilverChip monitor mode, use the `a` command to find the SCSI Device ID for the SCSI boot disk. Ensure that SilverChip firmware detects the SCSI peripherals correctly at startup of the board.

When the startup menu (SilverChip firmware) appears, the boot option can be configured (see the *Radstone SilverChip Manual*, Issue 5, for details).

4. Select **Run setup** followed by **Manage startup**.
5. Select **Add a boot selection** followed by **Other**.
6. Select **New System Partition**, then **SCSI Hard Disk**.
7. Specify SCSI disk details. For example, `Partition (0)` indicates that BlueCat Linux will start from the first partition on the SCSI disk with `SCSI ID 0`.
8. Specify the file to be loaded, or enter nothing if the given partition is configured to boot directly using the `mkboot` system command of BlueCat Linux.
9. Return to the main screen on the SilverChip monitor and choose the boot option just entered for booting from the SCSI disk.
10. Start booting from SCSI disk.

## Booting over a Network

Use the following procedure to boot the BlueCat Linux kernel on a PPC4A target board over a network:

1. From the SilverChip monitor screen, choose to add a new boot option for booting over a network.
2. Select **Run setup** followed by **Manage startup**.
3. Select **Add a boot selection** followed by **Other**.
4. If it has been created, select the network configuration. If not, choose **New system partition** to create a new network setup.
5. Select **Network** to configure the network. This involves specifying the IP address of the client, the TFTP server, and the gateway.

For example,

```
Enter Server Name: 1.0.0.1
Enter Client IP Address: 1.0.0.4
Enter Gateway IP Address: 1.0.0.254
Name of file to load:
/tftpboot/ppc4a/demo/install.prp
Enter a name for this boot selection:
BlueCatInstall demo net booting
```

---

**NOTE:** *Specify the full path of the image in the TFTP server. The image file should always have a .prp extension, as the PPC4A target board recognizes only the PReP as a valid bootable partition, and loads only files with a .prp extension.*

---

6. Return to the main screen on the SilverChip monitor and select the boot option just entered for booting from the network:
7. Start net booting the BlueCat Linux `install demo` system.

## Setting up a TFTP Server on a Windows Host

On Windows machines, the Trivial File Transfer Protocol (TFTP) is not supported by default. The TFTP server must specifically be installed on a Windows cross development host. The user may download any of the shareware available for the Windows environment.

## Setting up Softlinks to Download Demo Systems over a TFTP Server

Softlinks to various demo system files allows having multiple copies of BlueCat Linux on the cross development host.

To create softlinks to all the available demo system files, an option is provided as part of the installation script. For more information on how to invoke this option, please refer to Appendix A, “install\_cpci730.sh.”

## Booting from Target Flash Memory

There is 16 MB of flash memory available on a PPC4A board, organized into two banks of 8 MB each, starting at `0xff000000` and `0xff800000`, respectively. The top 1 MB is reserved for the SilverChip Firmware (`0xffff00000` to `0xfffffffff`). Therefore, the BlueCat Linux image can effectively use 15 of the 16 MB of target flash memory.

The SilverChip monitor `copy` command can program this 16 MB of flash memory with the BlueCat Linux image, which can then be used to boot the PPC4A target board.

The user system flash area of 8 MB consists of memory location `0xf000000` to `0xff7fffff`, and the system flash area of 7 MB consists of memory location `0xff800000` to `0xffefffff`. The memory occupied by the SilverChip firmware spans from location `0xffff00000` to `0xfffffffff` (see figure below).

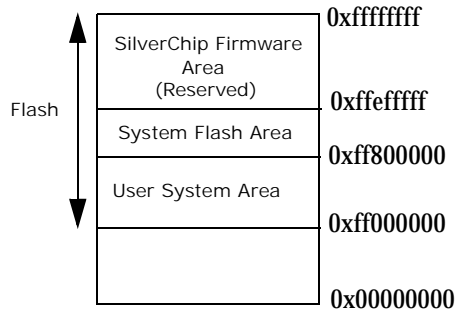


Figure 2-1: Flash Memory Organization

Use the following procedure to program and boot the BlueCat Linux kernel from the target flash memory:

1. Enter into the SilverChip monitor mode by entering `m` at the startup screen.
2. When in monitor mode, use the `set` command to set the parameters given below:

```
> set ClientIPAddr 1.0.0.5
> set myserver tftp:(1.0.0.1)
> copy \
myserver:/tftpboot/demo/ppc4a/hello.prp \
flash(0xff000000)
```

At this stage, the SilverChip firmware erases the PPC4A flash memory and reprograms it with the image downloaded from the TFTP server. The SilverChip monitor displays the progress of the `copy` command status. In the example above, the image file `hello.prp` is kept on a TFTP server under the path `/tftpboot`.

3. To boot from the PPC4A flash memory use the following command:

```
> exec(0xff007000)
```



**CAUTION!** *When copying data to the target flash memory using the SilverChip `copy` command, the default action is to erase to the end of the flash device (i.e., the next 8 MB boundary) before programming the data. This means that care must be taken not to erase the SilverChip (0xFFFF0000 to 0xFFFFFFFF) when using the area 0xFF800000 to 0xFFEFFFFFFF. To control how much of the PPC4A flash memory is erased, a length (in bytes) can be specified on the destination parameter. For example:*

```
> copy myserver:/tftpboot/big1.dat
flash(0xff000000) length(0x800000)

> copy myserver:/tftpboot/big2.dat
flash(0xff800000) length(0x700000)
```

*The second command above erases only 7 MB, thus preserving the SilverChip.*

---

**NOTE:** *As shown above, BlueCat Linux images greater than 8 MB have to be programmed with two separate `copy` commands. The image must be divided into two constituent files. This can be achieved with the Linux `DD` command, as follows:*

```
dd if=big.prp of=big1.dat bs=1 count=8192k
dd if=big.prp of=big2.dat bs=1 skip=8192k
```

---

## Autobooting from Target Flash Memory

To autoboot from the target flash memory directly, set up the following parameters under the SilverChip firmware:

1. Enter into the SilverChip monitor mode by entering `m` at the startup screen.
2. Select **Run Setup** followed by **Manage startup**.
3. Select **Run Setup**.
4. Select **Setup autoboot** followed by **Yes**.
5. Select **Enter Countdown value (in seconds):** and enter a delay value in seconds. Then, the autoboot from PPC4A flash memory starts.
6. Enter a **Countdown value in seconds: 10**.
7. Select **Run Setup** followed by **Manage startup**.
8. Select **Add a boot selection**.

9. Select **Other** followed by **New System Partition** and **Exec Image**.
10. On hitting the Enter key, enter the address at the following prompt as follows:  

```
“Enter Exec Image Address:” 0xff007000
```
11. Return to the main screen on the SilverChip monitor and choose the boot option just entered for booting from **Exec Image**. (Enter the **Exec Image** option at the SilverChip set prompt)

With the setup mentioned above, if the board is restarted, booting starts automatically from target flash memory.

---

**NOTE:** *To disable the autoboot option, during the 10 second countdown of the autoboot, press any key to get into the SilverChip startup menu and under the Setup autoboot option, select No.*

---

---

## Uninstallation

The `uninstall_ppc4a.sh` file is copied to the `/home/BC` directory as part of the installation process.

Once the installation is complete, the base distribution (`cpci_mcp750` for MCP750 boards) can be restored by using the `uninstall_ppc4a.sh` script file.

To uninstall the ppc4a TSP, proceed as follows:

1. Mount the base CD-ROM, that is, `cpci_mcp750` (if it is not already automounted) as follows:
  - For a Linux cross development host:

```
mount /mnt/cdrom
```
  - For a Windows cross development host:

```
mount -f e:/ /mnt/cdrom
```
2. Change to the directory where the TSP is installed as follows:

```
# cd /home/BC
```

3. Run the `uninstall_ppc4a.sh` program as follows:

```
# ./uninstall_ppc4a.sh -m/mnt/cdrom \  
-d/home/BC
```

This restores the base distribution.

For more information on `uninstall_ppc4a.sh`, see Appendix B.

After uninstallation, following message appears:

```
Uninstallation complete...Execute \  
. SETUP.sh on a new shell
```



---

# *Kernel Configuration Parameters*

This chapter provides tables showing the configuration of the prebuilt BlueCat Linux kernel contained in the ppc4a Target Support Package (TSP). Kernel configuration parameters and their options are also listed for the user's information.

**Table 3-1: Kernel Configuration Parameters for the ppc4a TSP**

Parameters	Table Number/Page
Platform Support	Table 3-2, page 22
General Setup	Table 3-3, page 23
Plug and Play Support	Table 3-4, page 24
Block Devices	Table 3-5, page 24
Networking Devices	Table 3-6, page 25
QoS and/or Fair Queueing	Table 3-7, page 26
SCSI Support	Table 3-8, page 27
SCSI Low-Level Drivers	Table 3-9, page 27
ARCnet Drivers	Table 3-10, page 28
Ethernet (10 or 100 Mbit)	Table 3-11, page 28
Token Ring Devices	Table 3-12, page 29
WAN Interfaces	Table 3-13, page 30
Amateur Radio Support	Table 3-14, page 30
ISDN Subsystem	Table 3-15, page 30
Old CD-ROM Drivers (not SCSI or IDE)	Table 3-16, page 30

**Table 3-1: Kernel Configuration Parameters for the ppc4a TSP (Continued)**

Parameters	Table Number/Page
Character Devices	Table 3-17, page 30
Mice	Table 3-18, page 31
Video for Linux	Table 3-19, page 31
Joystick Support	Table 3-20, page 32
Floppy Tape Device Driver, Ftape	Table 3-21, page 32
USB Drivers	Table 3-22, page 32
Filesystems	Table 3-23, page 32
Network Filesystems	Table 3-24, page 33
Partition Types	Table 3-25, page 33
Sound	Table 3-26, page 34
Kernel Hacking	Table 3-27, page 34

**Table 3-2: Platform Support**

Option	Value	Description
CONFIG_PPC	Y	PowerPC architecture
CONFIG_6xx	Y	6xx/7xx processor type
CONFIG_PREP	Y	PREP machine
CONFIG_SMP	N	Symmetric multiprocessing support

**Table 3-3: General Setup**

Option	Value	Description
CONFIG_EXPERIMENTAL	N	Prompts for development and/or incomplete code drivers
CONFIG_MODULES	Y	Enables loadable module support
CONFIG_MODVERSIONS	Y	Sets version information on all symbols for modules
CONFIG_KMOD	Y	Kernel module loader
CONFIG_PCI_QUIRKS	N	PCI quirks
CONFIG_PCI_OLD_PROC	Y	Backward-compatible <code>/proc/pci</code>
CONFIG_NET	Y	Networking support
CONFIG_BLUECAT_IGNORE_PRINTK	N	BlueCat Linux ignore <code>printk</code>
CONFIG_BLUECAT_SMALL_FOOTPRINT	N	BlueCat Linux small memory footprint
CONFIG_BLUECAT_MEMSIZE	N	Memory usage statistics
CONFIG_SYSCTL	Y	Sysctl support
CONFIG_SYSVIPC	Y	System V IPC
CONFIG_BSD_PROCESS_ACCT	Y	BSD Process Accounting
CONFIG_BINFMT_MISC	N	Kernel support for MISC binaries
CONFIG_BINFMT_JAVA	N	Kernel support for JAVA binaries (Obsolete)
CONFIG_PARPORT	N	Parallel port support to access the device
CONFIG_VGA_CONSOLE	Y	Support for VGA console
CONFIG_FB	Y	Support for frame buffer devices
CONFIG_PMAC_PBOOK	N	Power management support for Apple PowerBooks
CONFIG_MAC_KEYBOARD	N	Support for PowerMac keyboard
CONFIG_MAC_FLOPPY	N	Support for PowerMac floppy disk
CONFIG_MAC_SERIAL	N	Support for PowerMac serial parts
CONFIG_ADBMOUSE	N	Support for PowerMac ADB mouse
CONFIG_PROC_DEVICETREE	N	Support for open firmware device tree in <code>/proc</code>

**Table 3-3: General Setup (Continued)**

Option	Value	Description
CONFIG_TOTALMP	N	Support for TotalImpact TotalMP
CONFIG_BOOTX_TEXT	N	Support for early boot text console (BootX only)
CONFIG_MOTOROLA_HOTSWAP	N	Support for Motorola Hot Swap
CONFIG_COMDLIN_BOOL	N	PREP boot loader kernel arguments

**Table 3-4: Plug and Play Support**

Option	Value	Description
CONFIG_PNP	N	Plug and Play support

**Table 3-5: Block Devices**

Option	Value	Description
CONFIG_BLK_DEV_FD	N	Normal PC floppy disk support
CONFIG_BLK_DEV_IDE	Y	Enhanced IDE/MFM/RLL disk/ CD-ROM/tape/floppy support
CONFIG_BLK_DEV_HD_IDE	N	Uses old disk-only driver on primary interface
CONFIG_BLK_DEV_IDEDISK	N	Includes IDE/ATA-2 DISK support
CONFIG_BLK_DEV_IDECD	N	Includes IDE/ATAPI CDROM support
CONFIG_BLK_DEV_IDETAPE	N	Includes IDE/ATAPI TAPE support
CONFIG_BLK_DEV_IDEFLOPPY	Y	Includes IDE/ATAPI FLOPPY support
CONFIG_BLK_DEV_IDESCSI	N	SCSI emulation support
CONFIG_BLK_DEV_CMD640	N	CMD640 chipset bugfix/support
CONFIG_BLK_DEV_RZ1000	N	RZ1000chipset bugfix/support
CONFIG_BLK_DEV_IDEPCI	Y	Generic PCI IDE chipset support
CONFIG_BLK_DEV_IDEDMA	N	Generic PCI bus-master DMA support
CONFIG_BLK_DEV_OFFBOARD	N	Boots off-board chipset first support

**Table 3-5: Block Devices (Continued)**

Option	Value	Description
CONFIG_BLK_DEV_OPTI621	N	OPTi 82C621 chipset enhanced support (Experimental)
CONFIG_BLK_DEV_SL82C105	N	Winbond SL82c105 support
CONFIG_IDE_CHIPSETS	N	Other IDE chipset support
CONFIG_BLK_DEV_LOOP	Y	Loopback device support
CONFIG_BLK_DEV_NBD	N	Network block device support
CONFIG_BLK_DEV_MD	N	Multiple devices driver support
CONFIG_BLK_DEV_RAM	Y	RAM disk support
CONFIG_BLK_DEV_INITRD	N	Initial RAM disk (initrd) support
CONFIG_BLUECAT_RFS	Y	BlueCat RFS support
CONFIG_BLK_DEV_GENERIC_ FLASH_DOC	N	M-System DiskOnChip
CONFIG_BLK_DEV_XD	N	XT hard disk support
CONFIG_BLK_DEV_DAC960	N	Mylex DAC960/DAC1 100 PCI RAID Controller support
CONFIG_PARIDE_PARPORT	N	Parallel port IDE device support
CONFIG_BLK_CPQ_DA	N	Compaq SMART2 support

**Table 3-6: Networking Options**

Option	Value	Description
CONFIG_PACKET	N	Packet socket
CONFIG_NETLINK	N	Kernel/User netlink socket
CONFIG_FIREWALL	N	Network firewalls
CONFIG_FILTER	N	Socket filtering
CONFIG_UNIX	Y	UNIX domain sockets
CONFIG_INET	Y	TCP/IP networking
CONFIG_IP_MULTICAST	N	IP: Multicasting
CONFIG_IP_ADVANCED_ROUTER	N	IP: Advanced router

**Table 3-6: Networking Options (Continued)**

Option	Value	Description
CONFIG_IP_PNP	N	IP: Kernel level autoconfiguration
CONFIG_IP_ROUTER	N	IP: Optimize as router not host
CONFIG_NET_IPIP	N	IP: Tunneling
CONFIG_NET_IPGRE	N	IP: GRE tunnels over IP
CONFIG_IP_ALIAS	N	IP: Aliasing support
CONFIG_SYN_COOKIES	Y	IP: TCP syncookie support (Not enabled per default)
CONFIG_INET_RARP	N	IP: Reverse ARP
CONFIG_SKB_LARGE	Y	IP: Allows large windows (not recommended if <16 MB of memory)
CONFIG_IPV6	N	The IPv6 protocol (Experimental)
CONFIG_IPX	N	The IPX protocol
CONFIG_ATALK	N	Appletalk DDP
CONFIG_X25	N	CCITT X.25 Packet Laser (Experimental)
CONFIG_LAPB	N	LAPB Data Link Driver (Experimental)
CONFIG_BRIDGE	N	Bridging (Experimental)
CONFIG_LLC	N	802.2 LLC (Experimental)
CONFIG_ECONET	N	Acorn Econet/AUN protocols (Experimental)
CONFIG_WAN_ROUTER	N	WAN router
CONFIG_NET_FASTROUTE	N	Fast switching between high speed interfaces
CONFIG_NET_HW_FLOWCONTROL	N	Forwarding between high speed interfaces
CONFIG_CPU_IS_SLOW	N	CPU is too slow to handle full bandwidth

**Table 3-7: QoS and/or Fair Queueing**

Option	Value	Description
CONFIG_NET_SCHED	N	QoS and/or fair queueing

**Table 3-8: SCSI Support**

Option	Value	Description
CONFIG_SCSI	Y	SCSI support

**Table 3-9: SCSI Low-Level Drivers**

Option	Value	Description
CONFIG_SCSI_NCR53C8XX	Y	SCSI low-level drivers NCR53C8XX controllers
CONFIG_SCSI_SYM53C8XX	Y	SCSI low-level drivers SYM53C8XX controllers
CONFIG_SCSI_G_NCR5380_PORT	Not Set	SCSI low-level drivers NCR5380/53c400 mapping method (use Port for T130B) (PORT)
CONFIG_NETDEVICES	Y	Network device support
CONFIG_DUMMY	N	Dummy net driver support
CONFIG_EQUALIZER	N	EQL (serial line load balancing) support
CONFIG_NET_SB1000	N	General Instruments Surfboard 1000
CONFIG_FDDI	N	FDDI driver support
CONFIG_HIPPI	N	HIPPI driver support (Experimental)
CONFIG_PPP	Y	PPP (point-to-point) support
CONFIG_SLIP	N	SLIP (serial line) support
CONFIG_NET_RADIO	N	Wireless LAN (non-ham radio)
CONFIG_NET_FC	N	Fibre Channel driver support
CONFIG_RCPCI	N	Red Greek Hardware VPN (Experimental)
CONFIG_SHAPER	N	Traffic Shaper (Experimental)
CONFIG_SBNI	N	SBNI 12-xx support

Table 3-10: ARCnet Drivers

Option	Value	Description
CONFIG_ARCNET	N	ARCnet support

Table 3-11: Ethernet (10 or 100 Mbit)

Option	Value	Description
CONFIG_NET_ETHERNET	Y	Ethernet (10 or 100 MBit)
CONFIG_MACE	N	MACE (Power Mac Ethernet ) support
CONFIG_BMAC	N	BMAC (G3 Ethernet) support
CONFIG_NET_VENDOR_3COM	N	3COM cards
CONFIG_LANCE	N	AMD LANCE and PCnet (AT 1500 and NE2100) support
CONFIG_NET_VENDOR_SMC	N	Western Digital/SMC cards
CONFIG_NET_VENDOR_RACAL	N	Racal-Interlan (micom) NI cards
CONFIG_RTL8139	N	RealTek 8129/8139 (not 8019/8029!) support
CONFIG_SIS900	N	SiS 900 PCI Fast Ethernet Adapter support
CONFIG_YELLOWFIN	N	Packet Engines Yellowfin Gigabit-NIC support
CONFIG_NET_ISA	N	Other ISA cards
CONFIG_NET_EISA	N	EISA, VLB, PCI and on board controllers
CONFIG_PCNET32	N	AMD PCnet32 (VLB and PCI) support
CONFIG_ACEMIC	N	Alteon AceNIC/3Com 3C985/NetGear GA620 Gigabit support
CONFIG_AC3200	N	Ansel Communications EIA 3200 support (Experimental)
CONFIG_APRICOT	N	Apricot Xen-II on board Ethernet
CONFIG_CS89x0	N	CS89x0 support
CONFIG_DM9102	N	DM9102 PCI Fast Ethernet Adapter support (Experimental)

**Table 3-11: Ethernet (10 or 100 Mbit) (Continued)**

Option	Value	Description
CONFIG_DE4X5	Y	Generic DECchip & DIGITAL EtherWORKS PCI/EISA
CONFIG_BLUECAT_DE4X5_WORKAROUND	N	Fix for buggy SROM on Motorola MCP(N)750 cPCI board
CONFIG_DEC_ELCP	N	DECchip Tulip (dc21x4x) PCI support
CONFIG_DGRS	N	Digi Intl. RightSwitch SE-Xsupport
CONFIG_EEXPRESS_PRO100	N	Ether ExpressPro/100 support
CONFIG_LINE390	N	Mylex EISA LNE390A/B support (Experimental)
CONFIG_NET3210	N	Novell/Eagle/Microdyne NE3210 EISA support (Experimental)
CONFIG_NE2K_PCI	N	PCI NE2000 support
CONFIG_TLAN	N	TI ThunderLAN support
CONFIG_VIA_RHINE	N	VIA Rhine support
CONFIG_ES3210	N	Racal-Interlan EISA ES3210 support (Experimental)
CONFIG_EPIC100	N	SMC EtherPower II (Experimental)
CONFIG_ZNET	N	Zenith-Note support (Experimental)
CONFIG_NET_POCKET	N	Pocket and portable adaptors
CONFIG_BLUECAT_DE4X5_PARM	Y	Force network to 100BaseT
CONFIG_BLUE_DE4X5_PARM_FDX	Y	Force network to 100BaseT half-duplex
CONFIG_BLUECAT_DE4X5_PARM_FDX	Y	Force network to 100BaseT full-duplex

**Table 3-12: Token Ring Devices**

Option	Value	Description
CONFIG_TR	N	Token Ring driver support

**Table 3-13: Wan Interfaces**

Option	Value	Description
CONFIG_HOSTESS_SV11	N	Control Hostess SV-11 support
CONFIG_COSA	N	COSA/SRP sync serial boards support
CONFIG_SEALEVEL_4021	N	Sealevel Systems 4021 support
CONFIG_DLCI	N	Frame relay DLCI support

**Table 3-14: Amateur Radio Support**

Option	Value	Description
CONFIG_HAMRADIO	N	Amateur radio support

**Table 3-15: ISDN Subsystem**

Option	Value	Description
CONFIG_ISDN	N	ISDN subsystem support

**Table 3-16: Old CD-ROM Drivers (not SCSI or IDE)**

Option	Value	Description
CONFIG_CD_NO_IDESCSI	N	Supports non-SCSI/IDE/ATARI CD-ROM drivers

**Table 3-17: Character Devices**

Option	Value	Description
CONFIG_VT	N	Virtual terminal
CONFIG_VT_CONSOLE	N	Support for console on virtual terminal
CONFIG_SERIAL	Y	Standard/generic (dumb) serial support
CONFIG_SERIAL_CONSOLE	Y	Support for console on serial port

**Table 3-17: Character Devices (Continued)**

Option	Value	Description
CONFIG_SERIAL_EXTENDED	N	Extended dumb serial driver options
CONFIG_SERIAL_NONSTANDARD	N	Non-standard serial port support
CONFIG_UNIX98_PTYS	N	UNIX98 PTY support
CONFIG_MOUSE	Y	Mouse support (not serial mice)
CONFIG_QIC02_TAPE	N	QIC-02 tape support
CONFIG_WATCHDOG	N	Watchdog Timer support
CONFIG_NVRAM	N	/dev/nvram support
CONFIG_RTC	N	Enhanced Real-Time Clock support
CONFIG_DTLK	N	Double Talk PC internal speech card support

**Table 3-18: Mice**

Option	Value	Description
CONFIG_ATIXL_BUSMOUSE	N	ATIXL busmouse support
CONFIG_BUSMOUSE	Y	Logitech busmouse support
CONFIG_MS_BUSMOUSE	Y	Microsoft busmouse support
CONFIG_PSMOUSE	Y	PS/2 mouse (aka <i>auxiliary device</i> ) support
CONFIG_82C710_MOUSE	N	C&T 82C710 mouse port support (as on TI Travelmate)
CONFIG_PC110_PAD	N	PC110 digitizer pad support

**Table 3-19: Video for Linux**

Option	Value	Description
CONFIG_VIDEO_DEV	N	Video for Linux

**Table 3-20: Joystick Support**

Option	Value	Description
CONFIG_JOYSTICK	N	Joystick support

**Table 3-21: Floppy Tape Device Driver, Ftape**

Option	Value	Description
CONFIG_FTAPE	N	Ftape (QIC-80/Travan) support
CONFIG_FT_NORMAL_DEBUG	Not Set	Debugging output
CONFIG_FT_FULL_DEBUG	Not Set	Not debugging

**Table 3-22: USB Driver**

Option	Value	Description
CONFIG_USB	N	Support for USB (Experimental)

**Table 3-23: Filesystems**

Option	Value	Description
CONFIG_QUOTA	N	Quota support
CONFIG_QUOTA	N	Quota support
CONFIG_AUTOFKS_FS	N	Kernel automounter support
CONFIG_ADFS_FS	N	ADFS filesystem support (Read-only) (Experimental)
CONFIG_AFFS_FS	N	Amiga FFS filesystem support
CONFIG_HFS_FS	N	Apple Macintosh filesystem support (Experimental)
CONFIG_FAT_FS	N	DOS FAT filesystem support
CONFIG_ISO9660_FS	N	ISO 9660 CD-ROM filesystem support
CONFIG_JOLIET	N	Microsoft Joliet CD-ROM extensions
CONFIG_MINIX_FS	N	Minix filesystem support

**Table 3-23: Filesystems (Continued)**

Option	Value	Description
CONFIG_NTFS_FS	N	NTFS filesystem support (Read-only)
CONFIG_HPFS_FS	N	OS/2 HPFS filesystem support (Read-only)
CONFIG_PROC_FS	Y	/proc filesystem support
CONFIG_QNX4FS_FS	N	QNX filesystem support (Experimental)
CONFIG_ROMFS_FS	N	ROM filesystem support
CONFIG_EXT2_FS	Y	Second extended filesystem support
CONFIG_SYSV_FS	N	System V and Coherent filesystem support
CONFIG_UFS_FS	N	UFS filesystem support
CONFIG_EFS_FS	N	SGI EFS filesystem support (Read-only) (Experimental)

**Table 3-24: Network Filesystems**

Option	Value	Description
CONFIG_CODA_FS	N	Coda filesystem support (advanced network filesystem)
CONFIG_NFS_FS	Y	NFS filesystem support
CONFIG_NFSD	N	NFS server support
CONFIG_SMB_FS	N	SMB filesystem support (to mount WfW shares, etc.)
CONFIG_NCP_FS	N	NCP filesystem support (to mount NetWare volumes)

**Table 3-25: Partition Types**

Option	Value	Description
CONFIG_BSD_DISKLABEL	N	BSD disklabel (BSD partition tables) support
CONFIG_MAC_PARTITION	N	Macintosh partition map support
CONFIG_SMD_DISKLABEL	N	SMD disklabel (Sun partition tables) support

**Table 3-25: Partition Types (Continued)**

Option	Value	Description
CONFIG_SOLARIS_X86_PARTITION	N	Solaris (x86) partition table support
CONFIG_UNIXWARE_DISKLABEL	N	UnixWare slices support (Experimental)

**Table 3-26: Sound**

Option	Value	Description
CONFIG_SOUND	Y	Sound card support

**Table 3-27: Kernel Hacking**

Option	Value	Description
CONFIG_MAGIC_SYSRQ	N	Magic Sys Rq key
CONFIG_KDBG	N	Includes <code>kgdb</code> kernel debugger
CONFIG_BLUECAT_KDBG	Y	Includes BlueCat Linux kernel debugger
CONFIG_XMON	N	Includes <code>xmon</code> kernel debugger

## Supported Demo Systems

Table 4-1 lists the demo systems supported in the ppc4a Target Support Package (TSP). The boot devices supported by the prebuilt demo systems included in the distribution are also shown in Table 4-1.

**NOTE:** *Please refer to tables 4-5 and 4-6 (p.100) in the BlueCat Linux User's Guide for details on storage size and memory size, respectively.*

Table 4-1: Demo Systems Supported by PPC4A

Demo	Requirements	Boot Devices Supported by Default
gdb	Storage: Tiny RAM: Small Network: Yes Disk: None Special: Host and target machines must be connected by a serial line to use remote debugging via a serial line	<b>Network</b> using firmware <b>Flash</b>
hello	Storage: Tiny RAM: Tiny Network: None Disk: None Special: None	<b>Network</b> using firmware <b>Flash</b>

Table 4-1: Demo Systems Supported by PPC4A (Continued)

Demo	Requirements	Boot Devices Supported by Default
kdbg	Storage: Tiny RAM: Small Network: None Disk: None Special: Host and target machines must be connected by a serial line to use remote debugging via a serial line	<b>Network</b> using firmware <b>Flash</b>
memsize	Storage: Tiny RAM: Small Network: None Disk: None Special: None	<b>Network</b> using firmware <b>Flash</b>
osloader	Storage: Tiny RAM: Tiny Network: Yes Disk: None Special: None	<b>Network</b> using firmware <b>Flash</b> <b>SCSI</b>
ping	Storage: Tiny RAM: Small Network: Yes Disk: None Special: None	<b>Network</b> using firmware <b>Flash</b>
rcp	Storage: Tiny RAM: Small Network: None Disk: None Special: None	<b>Network</b> using firmware <b>Flash</b>
shell	Storage: Tiny RAM: Small Network: None Disk: None Special: None	<b>Network</b> using firmware <b>Flash</b>

Table 4-1: Demo Systems Supported by PPC4A (Continued)

Demo	Requirements	Boot Devices Supported by Default
uvme	Storage: Tiny RAM: Small Network: None Disk: None Special: Requires two PPC4A boards for testing purposes	<b>Network</b> using firmware <b>Flash</b>
xdemo1	Storage: Large RAM: Large Network: None Disk: Yes Special: Requires PS/2 mouse and keyboard	<b>Network</b> using firmware <b>SCSI</b>

## Running BlueCat Linux Demos Systems on PPC4A

This section and its subsections detail the setup and procedure required for running BlueCat Linux demo systems on the PPC4A target board.

### Running an X-Server-Based Demo System

This section is applicable for X-Server-based demo systems such as `xdemo1` and `xdemo2`. These demo systems can be run from a SCSI hard disk or over the network. For example, the `xdemo1.kdi` file can be run from any directory over the network using TFTP.

To run these demo systems from hard disk, additional files called `i_xdemo1` and `i_shell` have been provided as part of `xdemo1`, which, when generated, provides the `ftp` option to transfer files from the host. (This option is used instead of `tftp_i`.)

The demo systems require a minimum of 300MB free hard disk space.

## Frame Buffer Support for using the X-Server

To enable frame buffer support with the PMCGA1 card, do the following:

---

**NOTE:** *The frame buffer support is available to provide a generic graphic console with mouse support. The sequence below assumes that the graphic card used is PMCGA1 with a monitor supporting a 1024X768 resolution. Depending on the monitor used, modify the resolution and frequency details.*

---

1. Run `make xconfig` and enable the following options:
  - General setup-> Support for VGA console-> Y
  - General setup-> Support for frame buffer devices >Y
  
  - Console drivers-> Permedia2 support (experimental) ->Y
  - Console drivers-> enable FIFO disconnect feature->N
  - Console drivers-> generic Permedia2 PCI board support -> Y
  - Console drivers-> Open Firmware frame buffer device support -> N
  - Console drivers-> Matrox acceleration -> N
  - Console drivers-> ATI Mach64 display support -> N
  - Console drivers-> Virtual Frame Buffer support -> N
  - Console drivers-> Advanced low level drive options -> Y
  - Console drivers-> 2 bpp packed pixels support -> N
  - Console drivers-> 4 bpp packed pixels support -> N
  - Console drivers-> 8 bpp packed pixels support -> Y
  - Console drivers-> 16 bpp packed pixels support -> Y
  - Console drivers-> 24 bpp packed pixels support -> Y
  - Console drivers-> 32 bpp packed pixels support -> Y

The following options can be enabled or disabled depending on the font selection:

Console drivers -> Select compiled-in fonts

Console drivers -> VGA 8X8 font

Console drivers -> VGA 8X16 font

Console drivers -> Sparc console 8X16 font

Console drivers -> Sparc console 12X22 font

(not supported by all drivers)

2. Pass the following arguments to the kernel as part of the kernel downloadable image (KDI) generation only if the frame buffer support is required.

```
vga=792 video=pm2fb:mode:1024x768-\
75,SUN12x22,ypan
```

For details about how to pass arguments to the kernel, refer to the *BlueCat Linux User's Guide* for the `mkboot` system command.

3. If a new KDI is generated using the options given above, the frame buffer driver provides a graphical console on the screen with a resolution of 1024x768 for a given font.

## Running a Demo from Hard Disk

Use the following procedure to run a demo system from a SCSI hard disk.

The X-Server-based demo systems include, by default, PS/2 mouse drivers.

The example below explains how `xdemo1` can be run by installing it on a SCSI disk with a screen resolution of 1024x768 dpi.

1. Set up the BlueCat Linux environment if not already enabled:

```
# . SETUP.sh
```

2. Create a modified version of the `shell` demo system with the `make` command:

```
# cd $BLUECAT_PREFIX/demo/shell
# make -f i_Makefile clean all
```

This creates an image with the filename `i_shell.kdi`.

3. To access the file, create a symbolic link:

```
# cd /tftpboot/ppc4a/demo
# ln -s $BLUECAT_PREFIX/demo/shell/ \
i_shell.kdi i_shell.prp
```

4. Attach the SCSI hard disk to the target system.
5. Create the root filesystem on the host machine for the `xdemo1` demo system:

```
# cd $BLUECAT_PREFIX/demo/xdemo1
# make -f i_Makefile all
```

This generates `i_xdemo1.tar` and `i_xdemo1.disk`. By default, the root filesystem is not available immediately after installation of BlueCat Linux. The above command creates a root filesystem for `xdemo1`.

6. Boot the target board by using `i_shell.prp` from the SilverChip firmware menu:

```
/tftpboot/i_shell.prp (-> i_shell.kdi)
```

---

**NOTE:** Refer to “Installing BlueCat Linux on a Hard Disk from the install Demo System” in the BlueCat Linux User’s Guide for more details.

---

7. Create two partitions on the SCSI disk with the `fdisk` command:

```
# fdisk /dev/sda
```

Set the partitions according to the following table:

Table 4-2: SCSI Disk Partitions

Device	Start	End	ID	System
/dev/sda1	1	4	41	PreP Boot
/dev/sda2	5	last cylinder	83	Linux

8. Format the partition `/dev/sda2` and mount it for copying the root filesystem to the disk:

```
# mke2fs /dev/sda2
# mount /dev/sda2 /mnt
# cd /mnt
```

9. In the `i_shell` environment, use FTP to transfer the `xdemo` files:

```
# ifconfig eth0 target_IP
# ftp host_IP
ftp> cd /tftpboot
ftp> bin
ftp> prompt
ftp> mget i_xdemo1.disk i_xdemo1.tar
ftp> bye
```

10. In the `i_shell` environment, untar the `i_xdemo1.tar` file and change the ownership permissions with the following commands:

```
# tar xvf i_xdemo1.tar
# cd /mnt/usr/X11R6/bin
# /mnt/bin/chown root:0 XFree86 xterm \
xload xhost x init
```

---

**NOTE:** *Irrespective of how BlueCat for PPC4A is installed (with user or root permissions), the permission of the binaries XFree86, X, xterm, xdm, and xload needs to be set to root:0. Normally, as part of the build of X-Server, these files have bin:bin permissions only.*

---

11. Install the kernel image on the SCSI disk and make it bootable with the `mkboot` command:

```
# cd /mnt
# mkboot -b -k /mnt/i_xdemo1.disk -r 802
-c/mnt/c2.txt /dev/sda1
```

---

**NOTE:** *The file `c2.txt` contains default parameters for the root partition as `/dev/sda1` and screen resolution of 1024x768 dpi. This file may require changing if the installation is done on a differently partitioned disk or for a different screen resolution.*

---

12. Flush the filesystem buffers and reboot the system.

```
# sync
# sync
# reboot -f
```

## Using `xdemo1`

When the system reboots:

1. Select `start multi() scsi()` from the hard disk on the SilverChip firmware.
2. The system boots and a login session opens. Log in as `root` with password `root`.
3. Create runtime links with `ldconfig`:

```
# ldconfig
```

4. Start the X-Server with a default `xterm`.

```
# xinit
```

5. Using the mouse, click on the `xterm` window to activate it.
6. Use the `xhost` command to allow other systems to connect to the X-Server:

```
# xhost +
```

7. Open a new virtual terminal with the ALT and Function keys. ALT + F2, for example.
8. Log in as `root` with the password `root`.
9. Configure the Network interface parameters to display the X-Server:

```
# ifconfig lo 127.0.0.1
# export DISPLAY=127.0.0.1:0.0
```

10. Run an X-Windows application. For example,

```
# xeyes
```

To connect to other X-Servers on other hosts on the same network, use the `xhost` command while the X-Server is running.

```
# xhost + hostname
# ifconfig eth0 target_IP
# export DISPLAY=target_IP:0.0
```

The `xdemo1` demo system runs when the PPC4A target board is rebooted and run using a kernel image and filesystem from the SCSI disk. (See “Booting from a SCSI Disk” on page 13 for details.)

## Enabling glx Support in the X-Server

Glx is the glue between OpenGL/Mesa and the X-Server. It is an interface to provide X-Server the capability to run high acceleration, high resolution, 2D/3D graphics with rendering.

The library file `libglx.a` provided as part of the X-Server is used to load the glx module.

Any application developed to run under glx needs the graphics libraries provided either by OpenGL or Mesa. BlueCat Linux provides support for Mesa. BlueCat Linux also provides the C language based toolkit `glut-3.7` for developing glx applications.

Therefore, to run glut clients, Mesa/OpenGL need to be installed so that the shared libraries `libGL.so`, `libGLU.so`, and `libglut.so` are accessible.

To enable glx support in the X-Server:

1. Add `Load glx` to the modules section of the `$BLUECAT_PREFIX/demo/xdemo1/i_local/etc/\XF86Config` file.
2. After the starting the server, verify that `glx` and `sgi-glx` are listed in extensions of command output by entering the following command:

```
# xdpinfo
```

The command output is as follows:

```
name of display: :0.0
version number: 11.0
vendor string: The XFree86 Project, Inc
vendor release number: 4000
maximum request size: 262140 bytes
motion buffer size: 256
bitmap unit, bit order, padding: 32, MSBFirst, 32
image byte order: MSBFirst
number of supported pixmap formats: 6
supported pixmap formats: 6
depth 1, bits_per_pixel 1, scanline_page 32
depth 4, bits_per_pixel 8, scanline_page 32
depth 8, bits_per_pixel 8, scanline_page 32
depth 15, bits_per_pixel 16, scanline_page 32
depth 16, bits_per_pixel 16, scanline_page 32
depth 24, bits_per_pixel 32, scanline_page 32
keycode range: minimum 8, maximum 255
focus: PointerRoot
number of extensions: 10
DOUBLE-BUFFER
GLX
LEB
MIT-SHM
```

```
SECURITY
SGI-GLX
XC-APPGROUP
XFree86-Bigfont
XInputExtension
XTEST
default screen number: 0
number of screens: 1
screen #0:
dimensions: 1024x768 pixels (347x260 millimeters)
resolution: 75x75 dots per inch
depths (1): 16
...
```

## Testing glx Clients on the Target Board

To run glx clients, install Mesa or OpenGL, which provide shared libraries (`libGL.so`, `libGLU.so`, and `libglut.so`).

Shared libraries for Mesa are provided under the BlueCat Linux binary distribution; the RPM file for Mesa is `Mesa_trg-3.4-1.ppc.rpm`.

To recreate the binary RPM `Mesa_trg-3.4-1.ppc.rpm`, the source tar balls may need to be downloaded from [www.mesa3d.org](http://www.mesa3d.org) as these are not included in the BlueCat Linux source RPM distribution.

The spec file provided as part of `Mesa_trg-3.4-1.nosrc.rpm` can be used to recreate the binary RPM.

For testing the functionality of glx, an additional RPM file with some clients precompiled has been provided as part of the BlueCat Linux installation. This is the `glut_trg-3.7-1.ppc.rpm` file. This file provides glx demo systems under `$BLUECAT_PREFIX/usr/X11R6/bin/glx/`.

To test the glx clients on the target board, install the X-Server on the hard disk (using `i_xdemo1` demo system files) and after starting up the X-Server, use the commands below:

```
# cd /usr/X11R6/bin/glx/test
# ./bigtest
```

The `bigtest` is a client that opens two windows with an animated graphical image.

---

**NOTE:** *Other demo systems in the `/user/X11R6/bin/glx/` directory can be tested as mentioned above.*

---

## Running the gdb Demo under X-Server

The GDB in BlueCat Linux 3.0 can be run in text as well as graphical mode.

### Running GDB in Graphical Mode

This involves configuring `gdbserver` on the target and establishing a connection to it from the host.

#### Configuring the Target

To configure the target for TFTP download:

1. Using SilverChip monitor commands, specify the KDI name and IP address as follows:

```
Enter Server Name: host_IP_addr
Enter Client IP Address: target_IP_addr
Enter Gateway IP Address: gateway_IP_addr
Name of file to load:
/tftpboot/ppc4a/demo/gdb.prp
```

2. Boot the target board using the configuration above and enter the following commands at the shell prompt to start `gdbserver`:

```
# ifconfig eth0 target_IP_addr
# gdbserver 1.0.0.1:2500 file_to_debug
```

where `1.0.0.1` is the host IP address, `2500` is a free port number provided as part of the `gdb` demo system.

#### Connecting to the Target from the Host

To connect to the `gdbserver` running on the target, run the GDB client on the cross development host as follows:

1. Enable the BlueCat Linux environment if not already enabled.

```
# . SETUP.sh
```

2. Run `xhost +` from another shell.

```
# xhost +
```

3. Set the local `DISPLAY` variable:

```
# DISPLAY= host_IP_addr:0.0
```

4. Run GDB:

```
# gdb &
```

This activates a source window.

5. Change to the BlueCat Linux `gdb` source directory:

```
# cd $BLUECAT_PREFIX/demo/gdb/src
```

6. Select the program to debug by clicking on: **File -> Open -> Load New Executable**. Select `test_prog` -> **Open**.

The source code appears on the screen. The status bar at the bottom of the screen displays `test_prog.c`, `main`, and `SOURCE` in the respective pop-up lists.

7. Set the target board by clicking on **File -> Target Settings**. This will open a window entitled **Target Selection**. From this window, select **Remote/TCP**.

- Enter the target board IP address in the **Hostname** box, `192.168.10.108:2500`, for example.
- Enter the port number in the **port** box, `2500` (also specified with the `gdbserver` command), for example.
- Click on **More Options** and disable the following:

**Download program**

**Run Program**

**Continue from last stop**

and click on **OK**.

8. Run the program by selecting: **Run -> Run**  
A red **STOP** icon then flashes on the right side of the window. Line 12 in the source code is highlighted in green and a red marker shows the breakpoint.
9. Continue as follows:

```
Control -> Continue
```

This completes program execution and the following output is displayed on the cross development host console:

```
=====
Process ./test_prog created; pid = 97
Remote debugging using 192.168.10.108:2345
Do you know that 2 + 2 =
Child exited with retcode = 0
Child exited with status 1
Killing inferior
GDBserver restarting
Process ./test_prog created; pid = 98
=====
```

10. Exit the session by clicking on: **File -> Exit**

11. The following message appears:

```
A debugging is active. Do you still want
to close the debugger?
```

Click on **Yes**.

This closes the `gdbserver` on the host with the following output:

```
readchar: Got
bash#
```

## Running a Graphical GDB Session using Data Display Debugger

The following information pertains to running a graphical GDB session using the Data Display Debugger (DDD), from the cross development host to the target board with `gdbserver (gdb.prp)` on the host. DDD provides the graphical interface to GDB.

### Resolving Shared Library Requirements for DDD

The DDD is a cross development tool that runs on the i386 host. If it is run outside the BlueCat Linux environment, it fails with a segmentation fault (core dump).

If run under BlueCat Linux, the following error message may appear:

```
ddd: error in loading shared libraries: libXpm.so.4: cannot
open shared object file: No such file or directory
```

The reason for the error message is that the shared library file `libXpm.so.4` is a link to `libXpm.so.4.11` and is normally present in the path `$BLUECAT_PREFIX/cdt/X11R6/lib`. As the utility `ldconfig`

fails to create a symbolic link to this path, a link needs to be created manually as shown below:

```
# cd $BLUECAT_PREFIX/cdt/X11R6/lib
# ln -s libXpm.so.4.11 libXpm.so.4
```

### Procedure for Running GDB with DDD

Open a GDB session as follows.

1. Boot the target board by using the `gdb.kdi` image file provided under the `gdb` demo system directory. After the PPC4A target board boots with this image file, it will display a console prompt. At the prompt, enter the following commands:

```
# ifconfig eth0 IP_addr
# gdbserver host_IP_addr: any_free_port_no. \  
file_to_debug
```

For example:

```
gdbserver 192.168.10.108:2500 \  
./test_prog
```

The following message appears:

```
Process ./test_prog created; pid = 19
```

2. On the cross development host, execute DDD:

```
# cd $BLUECAT_PREFIX/demo/gdb/src
# ddd &
```

The DDD: The Data Display Debugger window appears.

3. Open a file by clicking on File -> Open Program -> Files.
4. Select `test_prog` and click on Open.  
The file appears in the DDD working field.
5. Connect to the target board by entering the following command in the `gdb` command window:

```
(gdb) target remote 192.168.10.245:2500
```

The following output appears in the DDD window:

```
Remote debugging using 192.168.10.245:2500
0x30017274 in ?? ()
```

The following message appears on the target console:

```
Remote debugging using 192.168.10.108:2500
```

6. Run the program as follows:

**Program -> Run**

7. Click on **Continue**.

The command executed is `cont`. The following output appears at the `gdb` prompt:

```
gdb (gdb)
Continuing.
Program exited normally.
(gdb)
```

The following output appears on the target console:

```
Do you know that 2 + 2 = 4
Child exited with retcode = 0
Child exited with status 1
GDBserver exiting
bash#
```

---

**NOTE:** For details on running the text-based GDB session from the cross development host to target board with `gdbserver`, see the BlueCat Linux User's Guide.

---

## nfsroot Demo System

The `nfsroot` demo system by default requires that a BOOTP server be set up for obtaining an IP address automatically when `nfsroot` starts up. If the same demo system needs to be run with a fixed IP address without a BOOTP server, follow the sequence below:

1. On the cross development host, enable the BlueCat Linux environment by running `SETUP.SH` and change directory to the `nfsroot` demo system.
2. Modify the following kernel configuration parameters using `make xconfig`:

```
Network Options -> IP: kernel level
autoconfiguration -> Y
Network Options -> BOOTP support -> N
Network Options -> RARP support -> N
```

```
Network File Systems -> NFS filesystem
Support -> Y
Network File System -> Root file system on
NFS -> Y
```

3. Modify the `MakeFile` with the `mkboot` command, and specify the IP address of the PPC4A board and the host machine as shown below:

```
echo "root=/dev/nfs nfsroot-\
1.0.3.1:/nfsroot

ip=1.0.3.2:1.3.3.1:::" > cl.txt

mkboot -m -k $kdi_name.disk -c cl.txt
$kdi_name.kdi
```

4. Create a new KDI image by running `make all`.

## kdbg Demo System

By default, the `COM1` serial port of the PPC4A target board is reserved for console messages.

The SilverChip monitor contains special flags to redirect all the console messages directly to the `COM1` port or to the graphic console (if a graphic card like the PMCGA1 is connected and properly configured).

For testing `kdbg` on PPC4A, the serial `COM2` port is used by BlueCat Linux `kdbg` to send data to the host server via a null modem serial cable.

Only the `COM2` serial port can be used on the target board side for communication with the host machine. The serial cable can be connected to `COM1` or `COM2` on the host machine. This configuration must be completed on the host system to display the data sent from the target board.

## UVME Demo System – uvme

### DEMO

The BlueCat Linux driver for interfacing with the Tundra Universe-II PCI-VME bridge

## SYNOPSIS

The UVME demo system contains the driver for Tundra Universe-II chip PCI-VME bridge ASIC (see “Initial Register Programming” on page 54 for driver details). This demo system can be used to install the driver as a module, run the test program, and demonstrate some read/write operations that can be performed using the test program.

## REQUIREMENTS

Storage: Small  
RAM: Small  
Network: None  
Disk: None

Special: Two boards are needed to test this demo, the first of which acts as the system controller.

## DESCRIPTION

Before starting the demo system, ensure that the Tundra Universe-II board has been configured so that the VME memory of the second board is visible by the system controller board (see the *Radstone SilverChip Manual* for details).

## Setting Parameters

Set the following parameters for running the `uvme` demo system using the SilverChip monitor commands:

- First Board (system controller)

```
> set VME_ID 0
> set VME0
0x100,0x80421000,0x08000000,0x08100000,0x
08000000
```

This creates a VME master window at `0x10000000` of size `0x100000` and permits PCI accesses from `0xC8000000` and upward for VME accesses at `0x10000000` and above.

- Second Board (non-system controller)

```
> set VME_ID 1
> set VME0
0xF00,0x80F20000,0x10000000,0x10100000,0x
70000000
```

This creates a VME slave window at the start of the local RAM and the VME bus at 0x10000000. The local address of the RAM is 0x80000000. The size of the window is 0x100000.

With the above setup, the system controller board can see the second board's RAM at 0x10000000 on the VME bus (A32 space).

## Running the uvme Demo

To run the `uvme` demo system, use the following procedure:

1. Load the `uvme.kdi` file provided under `$BLUECAT_PREFIX/demo/uvme` on the PPC4A system controller. The system boots up in a single user mode.

2. Under the shell prompt, load the driver as a module:

```
bash# insmod uvme.o
bash# lsmod
```

`lsmod` displays output that the module has been loaded properly.

3. On the slave board, initialize the physical memory at location 0x80000000 to some value using the SilverChip monitor command as follows:

```
> e 0x80000000
0x80000000:0x00000000 0x12f4
```

where the present value is 0x00000000 at memory location 0x80000000. The value entered is 0x12f4 (4 bytes in hex), starting at location 0x80000000.

4. To terminate the entry of values, type a period (.).
5. To verify the value entered, use the `d` command as follows:

```
> d 0x80000000
```

This displays the memory dump of a block of memory as follows:

```

0x80000000:000012f4 00000000 00000000 00000000
0x80000010:00000000 00000000 00000000 00000000
.
.
0x80000070: 00000000 00000000 00000000 00000000

```

6. Run the test program. The argument to the test program `uvme_sample` is the four bytes of data in hex.

```

bash# /bin/uvme_exmpl 5678
*** simple read/write test ***
KERN_INFO uvmeopen called
KERN_INFO uvmeioctl called
KERN_INFO uvme: test pci_addr = c8000000
KERN_INFO uvme: read access at c8004000 is 0
KERN_INFO uvme: write access at c8004000 is 34120000
KERN_INFO uvme: close called
*** test results ***
The input data to write to slave board =5678
    data "12f4" read from VME space
    data "5678" wrote to VME space
*****

```

## BlueCat Linux Driver for Interfacing with the Tundra Universe-II PCI-VME Bridge

A driver for interfacing with the Tundra Universe-11 PCI-VME Bridge has been provided in this release of BlueCat Linux as an installable module. It executes a simple memory write/read between the board with the Universe-II or any other VME boards connected to the VME backplane.

This driver can be included as part of the kernel using the configuration screen (by running `make xconfig`) and enabling the following option:

**Universe-II support -> Enable Tundra Universe-II  
PCI/VME support**

The `uvme` demo system has been provided to explain memory/read/write operations along with a sample test program to test it.

Driver installation is performed in three steps:

1. Initialization
2. PCI bus device configuration
3. Initial Universe-II register programming

## Initialization

During initialization, `uvme` registers itself as a character device driver accessible using `/dev/uvme`.

## PCI Bus Configuration

After the driver is registered, `uvme` then configures itself as a PCI bus-based device. The Universe-II bridge PCI configuration registers are initialized at boot time and `uvme` reads these only via calls to the standard BlueCat Linux PCI API. Specifically, these are the device ID, interrupt request line (IRQ), and the base physical address of the I/O port. This port address is then remapped to a kernel virtual address for access. After the IRQ is read, the topmost interrupt handler is registered for this line.

## Initial Register Programming

The final `uvme` installation step is register programming with initial static values. This includes setting the PCI slave image registers, map registers and enabling interrupts.

---

**NOTE:** *The driver does not initialize VME slave image registers. This must be done by SilverChip firmware.*

---

## File Operations

There are three file operations registered for `/dev/uvme`: *open*, *close*, and *ioctl*. The *open* and *close* functions simply return success. The *ioctl* function implements the following commands.

`REGISTER_POSTED_WRITE`

Registers the current process as the generator of posted writes.

`DMA_PGM`

Programs the DMA registers for DMA transfer.

`DMA_GO`

Initiates DMA transfer.

`DMA_STOP`

Stops DMA transfer when all the buffered data is written.

DMA\_HALT

Stops the DMA transfer after completion of the current packet transfer.

PRINT\_CONF\_REGS

Prints configuration registers.

PRINT\_DMA\_REGS

Prints DMA control/status registers.

PRINT\_IMAGE\_REGS

Prints PCI and VME slave image registers.

PRINT\_INT\_CTL\_REGS

Prints interrupt control registers.

PRINT\_ERR\_REGS

Prints error registers.

UVME\_TEST

Used for simple read/write tests.

### Interrupt Processing

UVME has the ability to handle two types of interrupts: a VME bus error, and the DMA transfer status.

### VME Bus Error

When a VME bus error interrupt is received, the VME bus error interrupt handler disables further interrupts and sets error bits in the interrupt control and error log registers.

If a posted writer process is registered, a `SIGBUS` signal is sent to that process. Otherwise, only a message is printed to the system console.

### DMA Transfer Status

When a DMA interrupt is received, the DMA interrupt handler checks to see if the interrupt is to notify `uvme` about a completed, stopped, or halted

transfer. Based on the request, it clears the appropriate status bits from the DMA control/status registers and interrupt control registers.

Finally, the process waiting on the DMA transfer is signaled.

### UVME Functionality Tests

A small user program called `uvme_exmp1` has been provided to drive the functionality tests. This program opens `/dev/uvme` and performs various *ioctl* calls on this device.

A preconfigured image that can be used for testing is provided in the `uvme` demo system. For the test program to run, the major and minor numbers assigned to the character-based `/dev/uvme` node file are 199 and 1, respectively.

### Read/Write Test

The read/write test uses the PCI slave image 0 to access a slave's RAM. The slave's VME slave image is configured by the firmware to correspond to the master's PCI slave image 0.

The `UVME_TEST` calls the *IOCTL* with the `UVME_TEST` command and passes a pointer to a structure that contains the old and new value fields (both integers).

The driver then reads the first word (four bytes) that is at the VME address mapped through the PCI slave image 0. It returns this in the new value field in the user structure. The new value in the user structure is then written to this location.

### DMA Test

The DMA test uses PCI slave image 0 to initiate a DMA transfer to a slave's RAM. The `dma_info` structure is filled with local PCI slave image 0 offset, the VME destination address (PCI slave image 0 BS + TO), and the like. It makes two *ioctl* calls with the commands `DMA_PGM` and `DMA_GO`.

After issuing the `DMA_GO` command, the process sleeps until the driver receives a DMA interrupt from the Universe device. If there is an error on the transfer, the process is killed with a bus error signal.

## Creating a New Specification File

A utility script (tcl script) has been provided under the `i_shell` demo system, which copies all the files of a given RPM package (RPM files with `trg` tags) to a given `*.spec` file. An example is provided below:

1. Mount the BlueCat Linux base distribution (`cpcli_mcp750`) CD-ROM to obtain the required RPM file:

```
# mount /dev/cdrom /mnt/cdrom
```

2. As an example, a new `*.spec` file is created using the existing `*.spec` file under the `shell` demo system. Change to the `shell` demo system directory, create a backup of the existing `.spec` file, and open it using a standard editor.

```
# cd $BLUECAT_PREFIX/demo/shell
```

```
# cp i_shell.spec i_shell.spec.bak
```

```
# vi i_shell.spec
```

3. At the desired location insert a `# RPMs` tag to mark the place where the files of a given RPM package are inserted. The text should be inserted exactly as shown below, including the filename `i_shell.spec`. It is the modified `i_shell.spec` file that is used to include the list of files in a given RPM file.

```
# perl insert_rpm_to_spec.pl \  
i_shell.spec \  
/mnt/cdrom/BlueCat_ppc/target/\  
RPMs/vim_trg-minimal-5.4-1.ppc.rpm \  
> tempfile
```

The script provided opens the `i_shell.spec` file and looks for the text `# RPMs`. It inserts the list of files of a given RPM package and a new file is created as `tempfile`.

```
# mv tempfile i_shell.spec
```

---

**NOTE:** *Before creating a new root filesystem using this specification file, ensure that the additional RPM packages are installed on the cross development host.*

---

See the *BlueCat Linux User's Guide* for more details on installing additional RPM packages.



## Supported Device Drivers

Table 5-1 shows the device drivers supported by the ppc4a Target Support Package (TSP).

Table 5-1: Device Drivers Supported by ppc4a TSP

Hardware Device	Device Driver	Location in Source Tree	Kernel Configuration Options	Notes
<b>Ethernet Controller</b> DEC 21143	Generic DECchip driver	drivers/net de4x5.c	CONFIG_DE4X5	Driver tested
<b>SCSI Controller</b> SYM 53C860 UltraSCSI/narrow	Generic SCSI driver	drivers/scsi/ scsi.c drivers/scsi/ scsi_syms.c	CONFIG SCSI CONFIG_BLK_DEV SCSI CONFIG SCSI_SYM53C8XX	Driver tested
<b>SCSI Hard Disk</b>	SCSI disk driver	drivers/block/ seagate.c	CONFIG_SERIAL	Driver tested on a Seagate drive
<b>FDD Controller</b> One floppy port	Standard PC floppy disk driver	drivers/block/ floppy.c	CONFIG_BLK_DEV_FD	Driver tested
<b>Serial Ports</b> – Two PC16550/INS8250 N-B/PC16450-compatible serial ports with RS-232 interface	Standard serial driver	drivers/char/ serial.c	CONFIG_SERIAL	Driver tested

Table 5-1: Device Drivers Supported by ppc4a TSP (Continued)

Hardware Device	Device Driver	Location in Source Tree	Kernel Configuration Options	Notes
<b>Parallel Port</b> – One parallel printer port	Parallel port driver	drivers/misc/*.c	CONFIG_PARPORT	Driver tested
<b>Keyboard/Mouse</b> PS/2	PC keyboard driver	drivers/char/pc_keyb.c	CONFIG_PCmouse	Driver tested

---

## Driver for the Network Chipset

The PPC4A target board comes with an on-board PCI Ethernet controller (DC21143 chipset). Under BlueCat Linux, the driver tested for this chipset is `de4x5.c` (the option **Generic DECchip & DIGITAL EtherWORKS PCI/EISA** under `make xconfig`).

The `tulip.c` driver is not tested for this Ethernet controller.

# *install\_ppc4a.sh*

---

## Command Reference

### UTILITY

Installs the TSP on the specified base distribution

### SYNOPSIS

```
install_ppc4a.sh [ -i -v -xy/n \  
-tabsolute_path_to_create_links ] \  
-dabsolute_path_of_base_directory
```

### LOCATION

- |    |   |
|----|---|
| -i | Specification to run in info mode   |
| -v | Specification to run in verbose mode  |
| -x | Specifies whether or not the <code>install</code> command retains base (cpci_mcp750) demo systems, which are not supported on PPC4A by default. If <code>y</code> , then demo systems are retained. |
| -d | Specifies the absolute path where base (cpci_mcp750) is installed   |
| -t | Specifies the absolute path to create links to demo system files.   |

---

**NOTE:** *It is important to note that there is no space between the options and their arguments, for example, between -x and “y/n,” -t and “absolute\_path\_to\_create\_links,” and -d and “absolute\_path\_of\_base\_directory.”*

---

## DESCRIPTION

The `install_ppc4a.sh` program also copies the `uninstall_ppc4a.sh` file for uninstallation of the ppc4a TSP and restoring the base (cpci\_mcp750).

This program runs in three modes:

- Info Mode
- Normal Mode
- Verbose Mode

### Info Mode

In this mode, information only about the files related to the current distribution is displayed. To activate *Info Mode* use the `-i` option. This displays all RPMs to be deleted from and installed onto the base (for MCP750 boards) distribution.

```
# ./install_ppc4a.sh -i -dabsolute_path_of_base_directory
```

### Normal Mode

In this mode, the script installs the ppc4a TSP on a given base. All messages are sent to the console in this mode; no messages are logged.

To access Normal Mode, use the following command:

```
# ./install_ppc4a -dabsolute_path_of_base_directory
```

If the `-x` option is specified with `y`, then the `install_ppc4a.sh` command will retain the base (cpci\_mcp750) demo systems, which are not supported by the ppc4a TSP.

If `-x` is not specified, the `install_ppc4a.sh` script will default to a `-xn` specification, which will not retain the demo systems provided in the base (cpci\_mcp750) distribution.

If the `-t` option is specified, the `install_ppc4a.sh` command creates softlinks (under the directory specified with this option) for the demo system files under `/tftpboot`.

```
# ./install_ppc4a.sh -tabsolute_path_to_create_links  
-dabsolute_path_of_base_directory
```

For example,

```
# ./install_ppc4a -t/tftpboot/ppc4a -d/home/BC
# ls -la /tftpboot/ppc4a/demo/hello*

hello.kernel-> /home/bcl/demo/hello/hello.kernel
hello.rfs-> /home/bcl/demo/hello/hello.rfs
hello.disk-> /home/bcl/demo/hello/hello.disk
hello.tar-> /home/bcl/demo/hello/hello.tar
hello.prp-> /home/bcl/demo/hello/hello.kdi
```

### Verbose Mode

This mode is used in conjunction with the `install` option. The detailed installation messages are logged in a file instead of being displayed on the console. In this mode, installation is completed with a log file, `install_ppc4a.sh.log`, being sent to base directory (`/home/BC`).

For *Verbose Mode*, specify the `-v` option as follows:

```
# ./install_ppc4a.sh -v -dabsolute_path_of_base_directory
```



# *uninstall\_ppc4a.sh*

---

## Command Reference

### UTILITY

Uninstalls the TSP from the specified base distribution.

### SYNOPSIS

```
uninstall_ppc4a.sh [ -i ] -m mountpoint \  
-d absolute_path_of_installed_tsp
```

### OPTIONS

- i            Specification to run in Info Mode
- m            Specifies the mountpoint of the base  
              (cpci\_mcp750) CD-ROM
- d            Specifies the absolute path where the TSP is  
              installed.

---

**NOTE:** *It is important to note that there is no space between the options and their arguments, for example, between -m and "mountpoint," and -d and "absolute\_path\_of\_installed\_tsp."*

---

### DESCRIPTION

This program runs in two modes:

- Info Mode
- Uninstall Mode

#### Info Mode

This option displays information regarding the TSP to be installed and the RPMs to be deleted and inserted, without any actual insertion or deletion of RPMs.

```
./uninstall_ppc4a.sh -i -mmountpoint \  
-dabsolute_path_of_installed_tsp
```

#### Uninstall Mode

For uninstalling the TSP from the base, issue the following command:

```
./uninstall_ppc4a.sh -mmountpoint \  
-dabsolute_path_of_installed_tsp
```

---

## Procedure

The X-Server supported in BlueCat Linux is Version 4.0.1. The source tar balls required for building this X-Server are available at <http://www.xfree86.org>.

An additional cross development file required to build the X-Server is provided in BlueCat Linux. This file is provided as part of the BlueCat Linux source RPM file, `XFree86_trg-41-1.0.nosrc.rpm`.

A typical sequence to regenerate the X-Server package from the source distribution is shown below:

1. Change to the directory where BlueCat Linux for ppc4a is installed as follows:

```
# cd /home/BC
```

2. Enable the BlueCat Linux environment (if not already enabled) using the following command:

```
# . SETUP.sh
```

3. Install a source RPM file as follows:

```
# rpm -i /tmp/ppc4a/\
XFree86_trg-41.1.0-1.nosrc.rpm
```

4. Before starting a new build of this installed source RPM, check that `$BLUECAT_PREFIX/cdt/src/bluecat/SOURCES` contains the four tar balls:

- X401src-1.tgz
- X401src-2.tgz
- X401src-3.tgz
- X401-cross-build-1.0.tgz

and that `$BLUECAT_PREFIX/cdt/src/bluecat/SPECS` contains the specification file `Xserver-4.0.1_trg.spec`.

---

**NOTE:** *The files X401src-1.tgz, X401src-2.tgz, and X401src-3.tgz are not provided as part of the BlueCat Linux source distribution. They must be downloaded from [www.xfree86.org](http://www.xfree86.org) and placed in the `$BLUECAT_PREFIX/cdt/src/bluecat/SOURCES` directory.*

---

5. If everything is in place, proceed with the build as follows:

```
# cd $BLUECAT_PREFIX/cdt/src/bluecat/\
SPECS
# rpm -ba XServer-4.0.1_trg.spec > \
rpm_build.log 2>&1
```

6. Check that the `$BLUECAT_PREFIX/cdt/src/bluecat\`  
`/RPMS/ppc/XFree86_trg-41.1.0-1.ppc.rpm` file  
bears the latest time stamp.

7. Before installing the newly created binary RPM, the installed RPM for the X-server must be deleted as follows:

```
# rpm -e -vv --nodeps \
XFree86_trg-41-1.0> rpm_erase.log 2>&1
```

8. Install the new X-server RPM as follows:

```
# rpm -i --nodeps --force \
$BLUECAT_PREFIX/cdt/src/bluecat/\
RPMS/ppc/XFree86_trg-41-1.0.ppc.rpm > \
rpm_install.log 2>&1
```

The updated time stamp of files in the `bin`, `lib`, and `include` directories in `$BLUECAT_PREFIX/usr/X11R6` indicate the success of RPM installation. The new installation copies its own files into the `$BLUECAT_PREFIX/usr/X11R6` directory.