

# BlueCat Linux Board Support Guide

---

BlueCat Linux Release 4.1

DOC-0591-00

*for the Virtio VPCS926 Virtual Platform*

Product names mentioned in the *BlueCat Linux Board Support Guide for the Virtio VPCS926 Virtual Platform* are trademarks of their respective manufacturers and are used here only for identification purposes.

Copyright©1987 - 2004, 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 the *BlueCat Linux Board Support Guide for the Virtio VPCS926 Virtual Platform* 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>PREFACE</b>	.....	<b>V</b>
	For More Information .....	v
	Typographical Conventions .....	vi
	Special Notes .....	vii
	Technical Support .....	vii
	How to Submit a Support Request .....	vii
	Where to Submit a Support Request .....	viii
<b>CHAPTER 1</b>	<b>OVERVIEW</b> .....	<b>1</b>
	Features Overview .....	1
	Support of Multithreaded Applications in GDB .....	1
	Cygwin Execution Environment Version 1.3.6 .....	1
	Supported Hardware .....	2
	Available BlueCat Linux Development Tools .....	2
	Supported Cross-Development Hosts .....	2
<b>CHAPTER 2</b>	<b>DOWNLOADING AND BOOTING BLUECAT LINUX ON THE TARGET</b> .....	<b>3</b>
	Prerequisites .....	3
	Downloading and Booting Overview .....	4
	Setting up the Virtual Platform .....	4
	Installing the Virtual Platform Software .....	4
	Configuring the Virtual Ethernet Connection .....	4
	Installing BlueCat Linux into VPCS926 ROM .....	5
	Booting BlueCat Linux from a Network .....	6
	Booting a Demo System from a Network Using the OS Loader .....	6

<b>CHAPTER 3</b>	<b>KERNEL CONFIGURATION OPTIONS .....</b>	<b>9</b>
<b>CHAPTER 4</b>	<b>SUPPORTED DEMO SYSTEMS.....</b>	<b>23</b>
	Demo Systems .....	23
	developer Demo System .....	23
	osloader Demo System .....	24
	showcase Demo System .....	24
	Using Selected RPM Packages .....	26
	Using the BusyBox RPM Package .....	26
	Using the TinyLogin RPM Package .....	31
	Using the Zebra RPM Package .....	37
<b>CHAPTER 5</b>	<b>SUPPORTED DEVICE DRIVERS .....</b>	<b>45</b>
<b>CHAPTER 6</b>	<b>KNOWN PROBLEMS AND LIMITATIONS.....</b>	<b>47</b>
	VPCS926 Target Platform Problems and Limitations .....	47

---

# — *Preface*

---

## For More Information

For more information on the features of BlueCat Linux, refer to the following printed and online documentation.

- *BlueCat Linux User's Guide*

This document contains information about installing, configuring and using BlueCat Linux.

- Online information

The complete BlueCat Linux documentation set is available on the BlueCat Linux Documentation CD-ROM. Books are provided in both HTML and PDF formats.

Updates to these documents are available online at the LynuxWorks Website: <http://www.lynuxworks.com>.

Additional information about commands and utilities is provided online with the `man` command. For example, to find information about the GNU GCC compiler, use the following syntax:

```
man gcc
```

## Typographical Conventions

The typefaces used in this manual, summarized below, emphasize important concepts. All references to file names and commands are case sensitive and should be typed accurately.

Kind of Text	Examples
Body text; <i>italicized</i> for emphasis, new terms, and book titles	Refer to the <i>BlueCat Linux User's Guide</i> .
Environment variables, file names, functions, methods, options, parameter names, path names, commands, and computer data	<code>ls</code> <code>-l</code> <code>myprog.c</code> <code>/dev/null</code>
Commands that need to be highlighted within body text, or commands that must be typed as is by the user are <b>bolded</b> .	<code>login: <b>myname</b></code> <code># <b>cd</b> /usr/home</code>
Text that represents a variable, such as a file name or a value that must be entered by the user	<code>cat &lt;filename&gt;</code> <code>mv &lt;file1&gt; &lt;file2&gt;</code>
Blocks of text that appear on the display screen after entering instructions or commands	<code>Linux version 2.4.10-1</code> <code>(bin@build1) (gcc version</code> <code>2.95.3 20010315 (release)) #5</code> <code>Tue Dec 18 13:33:08 MSK 2001</code> <code>Processor: Intel StrongARM-</code> <code>IXP1200 revision 3</code> <code>Architecture: Intel IXP1200</code> <code>On node 0 totalpages: 32768</code> <code>zone(0): 32768 pages.</code> <code>zone(1): 0 pages.</code> <code>zone(2): 0 pages.</code>
Keyboard options, button names, and menu sequences	<b>Enter</b> , <b>Ctrl-C</b>

## Special Notes

The following notations highlight any key points and cautionary notes that may appear in this manual.

---

**NOTE:** These callouts note important or useful points in the text.

---



**CAUTION!** Used for situations that present minor hazards that may interfere with or threaten equipment/performance.

---

## Technical Support

LinuxWorks Support handles support requests from current support subscribers.

For questions regarding LinuxWorks products or evaluation CDs, or to become a support subscriber, our knowledgeable sales staff will be pleased to help you (<http://www.linuxworks.com/corporate/contact/sales.php3>).

### How to Submit a Support Request

When you are ready to submit a support request, please include *all* the following information:

- First name
- Last name
- Your job title
- Phone number
- Fax number
- E-mail address
- Company name
- Address
- City, state, ZIP

- Country
- LynxOS or BlueCat Linux version you are using
- Target platform (for example, PowerPC or x86)
- Board Support Package (BSP)
- Current patch revision level
- Development host OS version
- Description of problem you are experiencing

## Where to Submit a Support Request

### Web-based Support:

Log in at <http://www.linuxworks.com/support/custhelp.php3> for all support subscribers, including Europe.

### By E-mail:

Support, Europe	tech_europe@lnxw.com
Support, worldwide except Europe	support@lnxw.com
Training and courses	USA: training-usa@lnxw.com Europe: training-europe@lnxw.com

### By Phone:

Training and courses	USA: +1 408-979-4353 Europe: +33 1 30 85 06 00
Support, Europe (from our Paris, France office)	+33 1 30 85 93 96
Support, worldwide except Europe and Japan (from our San José, CA, USA headquarters)	+1 800-327-5969 or +1 408-979-3940
Support, Japan	+81 33 449 3131

**By Fax:**

Support, Europe (from our Paris, France office)	+33 1 30 85 06 06
Support, worldwide except Europe and Japan (from our San José, CA, USA headquarters)	+1 408-979-3945
Support, Japan	+81 22 449 3803



This *BlueCat Linux Board Support Guide for the Virtio VPCS926 Virtual Platform* provides information about the BlueCat Linux Board Support Package (BSP) supporting the Virtio VPCS926 Virtual Platform that emulates the LSI Logic ARM926EJ-S Processor System.

Throughout this Board Support Guide (BSG), the BSP is referred to as the “vpcs926.” The target board is referred to as the “VPCS926.”

---

## Features Overview

The following sections describe the new features of this release.

### **Support of Multithreaded Applications in GDB**

BlueCat Linux for the VPCS926 Virtual Platform contains a new version of GDB that supports debugging of multithreaded applications.

### **Cygwin Execution Environment Version 1.3.6**

This release contains a new version of the Cygwin execution environment (1.3.6). It is recommended that users remove any previous version of Cygwin and perform the installation of Cygwin version 1.3.6 as described in the *BlueCat Linux User's Guide*.

## Supported Hardware

Table 1-1 describes the hardware supported with this release. For available BlueCat Linux drivers, please see Chapter 5, “Supported Device Drivers.”

**Table 1-1: Hardware Supported**

Model	Description
Virtio VPCS926 Virtual Platform	LSI Logic ARM926EJ-S processor, 64MB SDRAM, cache, AMBA Peripheral Vectored Interrupt Controller, two ApUART ports incorporated into the processor system, ApE110 Ethernet controller integrated into the processor system.

---

## Available BlueCat Linux Development Tools

Table 1-2 indicates the availability of BlueCat Linux development tools on the cross-development platforms listed for use with the vpcs926 BSP.

**Table 1-2: BlueCat Linux Tools Availability**

Tool	Windows	Linux
CodeWarrior	N/A	N/A
SpyKer	N/A	N/A
VisualLynux	✓	N/A

---

## Supported Cross-Development Hosts

The BlueCat Linux development environment requires an installed, functional cross-development host with an Intel 386 or higher CPU. This host needs to be running one of the following development environments:

- Windows 2000/Pro with SP1 or later
- Windows XP
- PC running Red Hat Linux 7.3
- PC running Red Hat Linux 8.0
- PC running Red Hat Linux 9

# *Downloading and Booting BlueCat Linux on the Target*

This chapter provides instructions for downloading a BlueCat Linux demo system from a cross-development host onto the target and then booting the demo system on the target board.

---

## **Prerequisites**

This document is a guide for downloading and booting BlueCat Linux systems onto the user's target platform. Scenarios that use demo systems included in the BlueCat Linux distribution are presented. A basic familiarity with the target board hardware and operation is required. The user must also have an understanding of system administration for the particular cross-development host on which the BlueCat Linux Core and the BSP are installed. It is assumed that the user has the manufacturer's documentation for the target platform as well as system administration reference material for the cross-development host.

Before downloading and booting BlueCat Linux on the target platform, it is assumed that the default BlueCat Linux ARM configuration and the vpcs926 BSP have been installed on the cross-development host. This means that the user must:

1. Install the BlueCat Linux ARM Core onto the cross-development host as described in the “Installing the Default Configuration” section in Chapter 1, “Introduction and Installation” in the *BlueCat Linux User's Guide*.
2. Install the vpcs926 BSP onto the cross-development host as detailed in the “Installing Target Board Support” section in Chapter 1, “Introduction and Installation” in the *BlueCat Linux User's Guide*.
3. Activate support for the vpcs926 BSP as detailed in the “Activating Support for a Target Board” section in Chapter 1, “Introduction and Installation” in the *BlueCat Linux User's Guide*.

---

## Downloading and Booting Overview

The procedure for downloading and booting a BlueCat Linux system on the VPCS926 target consists of the following main steps:

- Setting up the Virtual Platform
- Booting a BlueCat Linux system from emulated target ROM

Once the BlueCat Linux OS loader demo system has been installed into the emulated ROM, it can be used to download and boot a BlueCat Linux system from a network.

The OS loader demo system includes the `i_osloader` and `osloader` downloadable images. In the `vpcs926` BSP, both images provide functionality identical to the BlueCat Linux OS loader. This includes the ability to download BlueCat Linux images from a TFTP host and execute them in RAM, as well as other features of the OS loader.

---

## Setting up the Virtual Platform

### Installing the Virtual Platform Software

The VPCS926 is installed on a Windows host. Double-click on the `VPCS926.exe` installation file (obtained from Virtio) and follow the on-screen installation instructions. Once the installation script is completed, the Virtual Platform will start to run. The **Virtio Innovator** application window will appear, displaying the VPCS926 Virtual Platform design.

### Configuring the Virtual Ethernet Connection

The VPCS926 Virtual Platform software bundle includes a VHub application, which allows the VPCS926 to be connected to the physical Ethernet network and/or another virtual platform. The VHub application is installed automatically by the virtual platform installer.

If, however, you want to connect the virtual platform to the real network (rather than to a virtual network that includes only virtual platforms running on the host), a separate packet protocol driver needs to be installed manually.

To install this driver on Windows 2000/XP, perform the following steps:

1. On the desktop, right-click the **My Network Places** icon and then choose **Properties**.
2. Right-click the relevant **Local Area Connection** icon and then choose **Properties**.
3. Click on **Install**, select **Protocol**, and click the **Add button**.
4. Click on **Have Disk...** and point to the `PACKET.INF` file present in the `VHub` directory. By default, this directory is:  

```
C:\Virtio\Shared\Vhub\Win2k
```
5. Finally, proceed through the dialog and finish the installation.

---

**NOTE:** After the installation, the user needs to enable the check box for **Connect virtual network to real world** in the **VHub options** pop-up window.

---

---

## Installing BlueCat Linux into VPCS926 ROM

This section provides instructions on how a BlueCat Linux embedded system can be installed into the target ROM using the mechanisms provided by the VPCS926 Virtual Platform software. These instructions are applicable to any of the BlueCat Linux demo systems. This chapter uses the `developer` demo system as an example.

To install `developer` into the target platform ROM, perform the following steps:

1. Transfer the `developer.kdi` file from the `$BLUECAT_PREFIX/demo/developer` directory to the Windows host on which the VPCS926 Virtual Platform software has been installed.
2. Start the **VPCS926 Virtual Platform**.  
A block diagram of the system is displayed.
3. Double-click on the ROM component.  
A panel displaying the ROM component properties is opened.
4. Edit the **ROM contents** property. Enter the path to the `developer.kdi` file.

5. Make sure that the **disabled** property of the ROM component has the value of **no**.
6. Click the **Save and Apply Changes** button.
7. Go back to the **VPCS926 block diagram** tab in the main application window and double-click on the **SRAM** component that is located right below the **ROM** component on the diagram.  
  
A panel displaying the **SRAM component properties** is opened.
8. Make sure that the **disabled** property of the SRAM component is set to **yes**.
9. Click the **Save and Apply Changes** button.
10. Edit the initialization script of your VPCS926 Virtual Platform configuration (refer to the *VPCS926 Virtual Platform User's Guide* for information on the platform configurations and initialization scripts) and make it look like this:

g

Once this procedure has been performed, the `developer` demo system is installed into ROM and will be booted automatically when the VPCS926 Virtual Platform is started by pressing the **F5** key.

---

## Booting BlueCat Linux from a Network

A BlueCat Linux embedded system can be booted from a network using the OS loader.

### Booting a Demo System from a Network Using the OS Loader

To boot the `developer` demo system over a network using the OS loader, perform the following steps:

1. Follow the instructions given in “Installing BlueCat Linux into VPCS926 ROM” on page 5 to install the `osloader` demo system (`osloader.kdi`) into the target platform ROM.

2. Copy the `developer.kernel` and `developer.rfs` files from the `$BLUECAT_PREFIX/demo/developer` directory to the `/tftpbboot` directory on the cross-development host.

Start the VPCS926 Virtual Platform by pressing **F5**. The BlueCat Linux OS loader prompt (`>`) appears in the TTY0 console window.

3. At the BlueCat Linux OS loader prompt, enter the following commands:

```
> set IF eth0
> set IP <target_board_IP>
> set HOST <development_host_IP>
> set KERNEL tftp developer.kernel
> set RFS tftp developer.rfs
> set CMD ramdisk_size=28472
> boot
```

where `<target_board_IP>` is the IP address of the target and `<development_host_IP>` is the IP address of the development host. These commands load the `developer` demo system from a network onto the target board and then automatically start it.



# *Kernel Configuration Options*

The vpcs926 BSP comes with a default BlueCat Linux kernel. This kernel has a number of configuration options. This chapter details these options in the tables listed in Table 3-1: “vpcs926 BSP Kernel Configuration Parameters” below.

**Table 3-1: vpcs926 BSP Kernel Configuration Parameters**

<b>Parameters</b>
Table 3-2: “Code Maturity Level Options”
Table 3-3: “Loadable Module Support”
Table 3-4: “System Type”
Table 3-5: “IXP1200 Implementations”
Table 3-6: “General Setup”
Table 3-7: “Parallel Port Support”
Table 3-8: “Memory Technology Devices”
Table 3-9: “Plug and Play Configuration”
Table 3-10: “Block Devices”
Table 3-11: “Multidevice Support (RAID and LVM)”
Table 3-12: “Networking Options”
Table 3-13: “QoS and/or Fair Queueing”
Table 3-14: “Network Device Support”
Table 3-15: “Amateur Radio Support”
Table 3-16: “IrDA (Infrared) Support”
Table 3-17: “ATA/IDE/MFM/RLL Support”
Table 3-18: “SCSI Support”

**Table 3-1: vpcs926 BSP Kernel Configuration Parameters (Continued)**

Parameters
Table 3-19: “I2O Device Support”
Table 3-20: “ISDN Subsystem”
Table 3-21: “Input Core Support”
Table 3-22: “Character Devices”
Table 3-23: “Serial Drivers”
Table 3-24: “I2C Support”
Table 3-25: “L3 Serial Bus Support”
Table 3-26: “Mice”
Table 3-27: “Joysticks”
Table 3-28: “Watchdog Cards”
Table 3-29: “Ftape, the Floppy Tape Device Driver”
Table 3-30: “Multimedia Devices”
Table 3-31: “File Systems”
Table 3-32: “Network File Systems”
Table 3-33: “Partition Types”
Table 3-34: “Multimedia Capabilities Port Driver”
Table 3-35: “Bluetooth Support”
Table 3-36: “Kernel Hacking”
Table 3-37: “Modular Advanced Power Management”
Table 3-38: “Messenger Support”

**Table 3-2: Code Maturity Level Options**

Option	Value	Description
CONFIG_EXPERIMENTAL	Y	Prompt for development and/or incomplete code/drivers
CONFIG_OBSOLETE	N	Prompt for obsolete code/drivers

**Table 3-3: Loadable Module Support**

Option	Value	Description
CONFIG_MODULES	Y	Enable loadable module support
CONFIG_MODVERSIONS	N	Set version information on all modules
CONFIG_KMOD	N	Kernel module loader

**Table 3-4: System Type**

Option	Value	Description
CONFIG_ARCH_VPCS926	Y	ARM system type

**Table 3-5: IXP1200 Implementations**

Option	Value	Description
CONFIG_CPU_ARM926T	Y	Support ARM926T processor
CONFIG_CPU_ARM926_CPU_IDLE	Y	ARM926T CPU idle
CONFIG_CPU_ARM926_I_CACHE_ON	Y	ARM926T I-Cache on
CONFIG_CPU_ARM926_D_CACHE_ON	Y	ARM926T D-Cache on
CONFIG_CPU_ARM926_WRITETHROUGH	N	Force write through caches on ARM926T
CONFIG_CPU_ARM926_ROUND_ROBIN	N	Round robin I and D cache replacement algorithm

**Table 3-6: General Setup**

Option	Value	Description
CONFIG_ANGELBOOT	N	Load kernel using Angel Debug Monitor
CONFIG_BLUECAT_IGNORE_PRINTK	N	BlueCat Linux Ignore printk

**Table 3-6: General Setup (Continued)**

Option	Value	Description
CONFIG_BLUECAT_THUMB	N	BlueCat Linux kernel support for THUMB binaries
CONFIG_BLUECAT_LOADER	N	BlueCat Linux OS loader
CONFIG_BLUECAT_SMALL_FOOTPRINT	N	BlueCat Linux small memory footprint
CONFIG_BLUECAT_FIX_PCI_DMA	N	Do not turn PCI bus-mastering on kernel startup
CONFIG_ZBOOT_ROM	N	Compressed boot loader in ROM/Flash
CONFIG_ZBOOT_ROM_TEXT	0	Compressed ROM boot loader base address
CONFIG_ZBOOT_ROM_BSS	0	Compressed ROM boot loader BSS address
CONFIG_HOTPLUG	N	Support hot-pluggable devices
CONFIG_NET	Y	Networking support
CONFIG_BLUECAT_MEMSIZE	N	Memory sizing benchmarks
CONFIG_SYSVIPC	Y	System V IPC
CONFIG_BSD_PROCESS_ACCT	N	BSD Process Accounting
CONFIG_SYSCTL	Y	Sysctl support
CONFIG_FPE_NWFPE	Y	NWFPE math emulation
CONFIG_FPE_FASTFPE	N	FastFPE math emulation (Experimental)
CONFIG_KCORE_ELF	Y	Kernel core (/proc/kcore) format
CONFIG_BINFMT_AOUT	N	Kernel support for a.out binaries
CONFIG_BINFMT_ELF	Y	Kernel support for ELF binaries
CONFIG_BINFMT_MISC	N	Kernel support for MISC binaries
CONFIG_PM	N	Power Management support (Experimental)
CONFIG_ARTHUR	N	RISC OS personality
CONFIG_ALIGNMENT_TRAP	Y	Kernel-mode alignment trap handler

---

**Table 3-7: Parallel Port Support**

Option	Value	Description
CONFIG_PARPORT	N	Parallel port support

**Table 3-8: Memory Technology Devices**

Option	Value	Description
CONFIG_MTD	N	Memory Technology Device (MTD) support

**Table 3-9: Plug and Play Configuration**

Option	Value	Description
CONFIG_PNP	N	Plug and Play support

**Table 3-10: Block Devices**

Option	Value	Description
CONFIG_BLK_DEV_FD	N	Normal PC floppy disk support
CONFIG_BLK_DEV_LOOP	Y	Loopback device support
CONFIG_BLK_DEV_NBD	N	Network block device support
CONFIG_BLK_DEV_RAM	Y	RAM disk support
CONFIG_BLK_DEV_RAM_SIZE	4096	Default RAM disk size
CONFIG_BLK_DEV_INITRD	N	Initial RAM disk ( <i>initrd</i> ) support
CONFIG_BLUECAT_RFS	Y	BlueCat Linux RFS support

**Table 3-11: Multidevice Support (RAID and LVM)**

Option	Value	Description
CONFIG_MD	N	Multiple devices driver support (RAID and LVM)

**Table 3-12: Networking Options**

Option	Value	Description
CONFIG_PACKET	Y	Packet socket
CONFIG_PACKET_MMAP	N	Packet socket: mmaped I/O
CONFIG_NETLINK_DEV	Y	Netlink device emulation
CONFIG_NETFILTER	N	Network packet filtering (replaces ipchains)
CONFIG_FILTER	N	Socket filtering
CONFIG_UNIX	Y	UNIX domain sockets
CONFIG_INET	Y	TCP/IP networking
CONFIG_IP_MULTICAST	Y	IP: multicasting
CONFIG_IP_ADVANCED_ROUTER	N	IP: advanced router
CONFIG_IP_PNP	N	IP: kernel level autoconfiguration
CONFIG_NET_IPIP	N	IP: tunneling
CONFIG_NET_IPGRE	N	IP: GRE tunnels over IP
CONFIG_IP_MROUTE	N	IP: multicast routing
CONFIG_ARPD	N	IP: ARP daemon support (Experimental)
CONFIG_INET_ECN	N	IP: TCP Explicit Congestion Notification support
CONFIG_SYN_COOKIES	N	IP: TCP syncookie support (disabled per default)
CONFIG_IPV6	N	The IPv6 protocol (Experimental)
CONFIG_KHTTPD	N	Kernel httpd acceleration (Experimental)

**Table 3-12: Networking Options (Continued)**

Option	Value	Description
CONFIG_ATM	N	Asynchronous Transfer Mode (ATM) (Experimental)
CONFIG_VLAN_8021Q	N	802.1Q VLAN support (Experimental)
CONFIG_IPX	N	The IPX protocol
CONFIG_ATALK	N	Appletalk protocol support
CONFIG_DECNET	N	DECnet support
CONFIG_BRIDGE	N	802.1d Ethernet Bridging
CONFIG_X25	N	CCITT X.25 Packet Layer (Experimental)
CONFIG_LAPB	N	LAPB Data Link Driver (Experimental)
CONFIG_LLC	N	802.2 LLC (Experimental)
CONFIG_NET_DIVERT	N	Frame Diverter (Experimental)
CONFIG_ECONET	N	Acorn Econet/AUN protocols (Experimental)
CONFIG_WAN_ROUTER	N	WAN router
CONFIG_NET_FASTROUTE	N	Fast switching (read help!)
CONFIG_NET_HW_FLOWCONTROL	N	Forwarding between high speed interfaces

**Table 3-13: QoS and/or Fair Queueing**

Option	Value	Description
CONFIG_NET_SCHED	N	QoS and/or fair queueing

**Table 3-14: Network Device Support**

Option	Value	Description
CONFIG_NETDEVICES	N	Network device support

**Table 3-15: Amateur Radio Support**

Option	Value	Description
CONFIG_HAMRADIO	N	Amateur radio support

**Table 3-16: IrDA (Infrared) Support**

Option	Value	Description
CONFIG_IRDA	N	IrDA subsystem support

**Table 3-17: ATA/IDE/MFM/RLL Support**

Option	Value	Description
CONFIG_IDE	N	ATA/IDE/MFM/RLL support

**Table 3-18: SCSI Support**

Option	Value	Description
CONFIG_SCSI	N	SCSI support

**Table 3-19: I2O Device Support**

Option	Value	Description
CONFIG_I2O	N	I2O support

---

**Table 3-20: ISDN Subsystem**

Option	Value	Description
CONFIG_ISDN	N	ISDN support

**Table 3-21: Input Core Support**

Option	Value	Description
CONFIG_INPUT	N	Input core support

**Table 3-22: Character Devices**

Option	Value	Description
CONFIG_VT	N	Virtual terminal
CONFIG_SERIAL	Y	Standard/generic (8250/16550 and compatible UARTs) serial support
CONFIG_SERIAL_CONSOLE	Y	Support for console on serial port
CONFIG_SERIAL_EXTENDED	N	Extended dumb serial driver options
CONFIG_SERIAL_NONSTANDARD	N	Nonstandard serial port support

**Table 3-23: Serial Drivers**

Option	Value	Description
CONFIG_SERIAL_8250	N	8250/16550 and compatible serial support (Experimental)
CONFIG_UNIX98_PTYS	Y	Unix98 PTY support
CONFIG_UNIX98_PTY_COUNT	256	Maximum number of Unix98 PTYs in use (0 to 2048)

**Table 3-24: I2C Support**

Option	Value	Description
CONFIG_I2C	N	I2C support

**Table 3-25: L3 Serial Bus Support**

Option	Value	Description
CONFIG_L3	N	L3 support

**Table 3-26: Mice**

Option	Value	Description
CONFIG_BUSMOUSE	N	Bus mouse support
CONFIG_MOUSE	N	Mouse support (not serial and bus mice)

**Table 3-27: Joysticks**

Option	Value	Description
CONFIG_QIC02_TAPE	N	QIC-02 tape support

**Table 3-28: Watchdog Cards**

Option	Value	Description
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 3-28: Watchdog Cards (Continued)**

Option	Value	Description
CONFIG_R3964	N	Siemens R3964 line discipline
CONFIG_APPLICOM	N	Applicom intelligent fieldbus card support

**Table 3-29: Ftape, the Floppy Tape Device Driver**

Option	Value	Description
CONFIG_FTAPE	N	Ftape (QIC-80/Travan) support
CONFIG_AGP	N	/dev/agpgart (AGP support)
CONFIG_DRM	N	Direct Rendering Manager (XFree86 DRI support)

**Table 3-30: Multimedia Devices**

Option	Value	Description
CONFIG_VIDEO_DEV	N	Video for Linux

**Table 3-31: File Systems**

Option	Value	Description
CONFIG_QUOTA	N	Quota support
CONFIG_AUTOFS_FS	N	Kernel automounter support
CONFIG_AUTOFS4_FS	N	Kernel automounter version 4 support (also supports v3)
CONFIG_REISERFS_FS	N	Reiserfs support
CONFIG_ADFS_FS	N	ADFS file system support
CONFIG_AFFS_FS	N	Amiga FFS file system support (Experimental)

**Table 3-31: File Systems (Continued)**

Option	Value	Description
CONFIG_HFS_FS	N	Apple Macintosh file system support (Experimental)
CONFIG_BFS_FS	N	BFS file system support (Experimental)
CONFIG_EXT3_FS	N	Ext3 journalling file system support (Experimental)
CONFIG_FAT_FS	N	DOS FAT file system support
CONFIG_EFS_FS	N	EFS file system support (read-only) (Experimental)
CONFIG_CRAMFS	N	Compressed ROM file system support
CONFIG_TMPFS	N	Virtual memory file system support (former shm file system)
CONFIG_RAMFS	N	Simple RAM-based file system support
CONFIG_ISO9660_FS	N	ISO 9660 CD-ROM file system support
CONFIG_MINIX_FS	N	Minix file system support
CONFIG_VXFS_FS	N	FreeVxFS file system support (VERITAS VxFS™-compatible)
CONFIG_NTFS_FS	N	NTFS file system support (read-only)
CONFIG_HPFS_FS	N	OS/2 HPFS file system support
CONFIG_PROC_FS	Y	/proc file system support
CONFIG_DEVFS_FS	N	/dev file system support (Experimental)
CONFIG_DEVPTS_FS	Y	/dev/pts file system for Unix98 PTYs
CONFIG_QNX4FS_FS	N	QNX4 file system support (read-only) (Experimental)
CONFIG_ROMFS_FS	N	ROM file system support
CONFIG_EXT2_FS	Y	Second extended file system support
CONFIG_SYSV_FS	N	System V/Xenix/V7/Coherent file system support
CONFIG_UDF_FS	N	UDF file system support (read-only)
CONFIG_UFS_FS	N	UFS file system support (read-only)

**Table 3-32: Network File Systems**

Option	Value	Description
CONFIG_CODA_FS	N	Coda file system support (advanced network file system)
CONFIG_INTERMEZZO_FS	N	InterMezzo file system support (Experimental, replicating file system)
CONFIG_NFS_FS	Y	NFS file system support
CONFIG_NFS_V3	Y	Provide NFSv3 client support
CONFIG_NFSD	N	NFS server support
CONFIG_SMB_FS	N	SMB file system support (to mount Windows shares, etc.)
CONFIG_NCP_FS	N	NCP file system support (to mount NetWare volumes)

**Table 3-33: Partition Types**

Option	Value	Description
CONFIG_PARTITION_ADVANCED	N	Advanced partition selection

**Table 3-34: Multimedia Capabilities Port Driver**

Option	Value	Description
CONFIG_MCP	N	Multimedia drivers

**Table 3-35: Bluetooth Support**

Option	Value	Description
CONFIG_BLUEZ	N	Bluetooth subsystem support

**Table 3-36: Kernel Hacking**

Option	Value	Description
CONFIG_FRAME_POINTER	N	Compile kernel with frame pointer
CONFIG_DEBUG_USER	N	Verbose user fault messages
CONFIG_DEBUG_INFO	N	Include debugging information in kernel binary
CONFIG_BLUECAT_KDBG	N	Include kdbg kernel debugger
CONFIG_DEBUG_KERNEL	N	Kernel debugging

**Table 3-37: Modular Advanced Power Management**

Option	Value	Description
CONFIG_BLUECAT_APM	N	Modular Advanced Power Management (MAPM) support

**Table 3-38: Messenger Support**

Option	Value	Description
CONFIG_BLUECAT_IOPMAN	N	Enable IOP Manager support
CONFIG_BLUECAT_MSNG	N	Enable Messenger support

This chapter provides information about BlueCat Linux demo systems supported by the vpcs926 BSP.

---

## Demo Systems

The following table lists demo systems supported by the vpcs926 BSP, their default boot devices, and their RAM and ROM requirements:

**Table 4-1: Demo Systems Supported by the vpcs926 BSP**

Demo System	Default Supported Boot Device(s)	ROM Requirements	RAM Requirements
developer	Ethernet (using the OS loader), Virtual platform ROM	3491 KB	13360 KB
osloader	Virtual platform ROM	793 KB	4819 KB
showcase	Ethernet (using the OS loader), Virtual platform ROM	2591.5 KB	12660 KB

### developer Demo System

The `developer` demo system is a package consisting of the functionalities of the `shell`, `ftp`, `ping`, `gdb`, and `vl_demo` systems. For a description of `developer` and its components, refer to Chapter 4, “BlueCat Linux Demo Systems” in the *BlueCat Linux User’s Guide*.

## osloader Demo System

`osloader` is the BlueCat Linux OS loader system used to boot a BlueCat Linux system on target boards. Refer to Chapter 4, “BlueCat Linux Demo Systems” in the *BlueCat Linux User’s Guide* for details.

## showcase Demo System

The `showcase` demo system starts and configures the Apache HTTP daemon, turning the target board into a web server. Refer to Chapter 4, “BlueCat Linux Demo Systems” in the *BlueCat Linux User’s Guide* for details. The “Modifying Target Board IP and Gateway Addresses” subsection below describes how to change the default IP address for the `showcase` demo system.

## Modifying Target Board IP and Gateway Addresses

The `showcase` demo system includes an Apache web server feature.

The Apache web server is a robust, commercial-grade, featureful, and freely-available source code implementation of an HTTP (web) server. See <http://www.apache.org> for further information.

Users may wish to define a unique IP address for their target board rather than use the default address (172.17.100.15) defined in the `showcase` demo system. To change the default IP address for the `showcase` demo system, open the `showcase` demo system’s `.bashrc.vpcs926` file with any text editor (such as `vi`) and perform the following steps:

1. Change directory to the location of the `.bashrc` file:

```
BlueCat:$ cd $BLUECAT_PREFIX/demo.vpcs926/\
showcase/local/etc
```

2. Change the `showcase` demo system’s `.bashrc.vpcs926` default target board IP address (172.17.100.15) to a new user-selected one. For example, change the target board IP address to 216.100.252.140 by editing it as follows:

Find this line:

```
TARGET_IP=172.17.100.15
```

Edit it to read as follows:

```
TARGET_IP=216.100.252.140
```

- Set a gateway IP address by entering a value at this line in the `.bashrc.vpcs926` file:

```
GATE_IP= <gateway_IP_addr>
```

- Rebuild the `showcase` demo system by entering the following commands in the `$BLUECAT_PREFIX/demo.vpcs926/showcase` directory:

```
BlueCat:$ touch showcase.spec
BlueCat:$ make rootfs
BlueCat:$ make kdi
```

- Install the updated `showcase.kdi` into the target platform ROM. Refer to “Installing BlueCat Linux into VPCS926 ROM” on page 5 for the instructions.

Now when the `showcase` demo system is downloaded onto the target board, the Apache web server BlueCat Linux web pages can be accessed at the user’s customized IP address, as shown in the figure below.

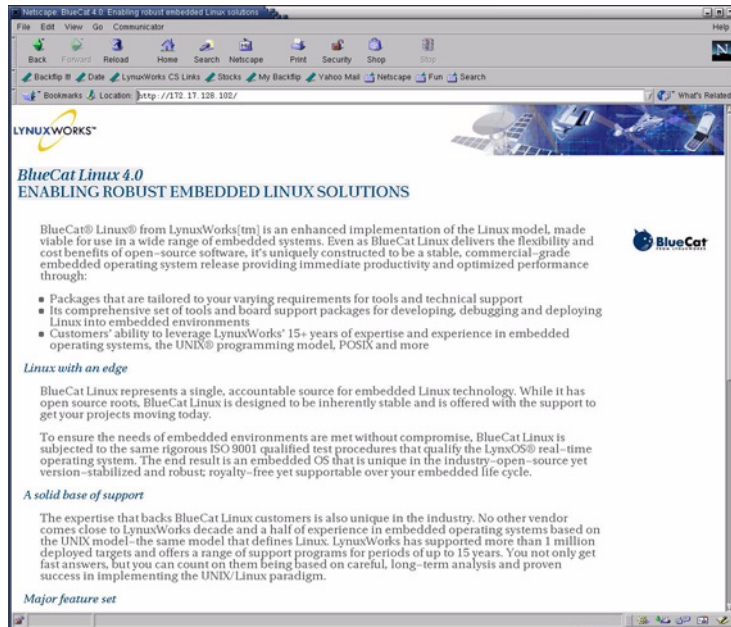


Figure 4-1: BlueCat Linux Apache Web Server Web Page

## Using Selected RPM Packages

This section describes how to use selected RPM packages that are frequently deployed in the embedded systems environment.

### Using the BusyBox RPM Package

The BusyBox RPM package combines tiny versions of many common UNIX utilities into a single small executable. It provides minimalist replacements for most UNIX utilities, such as `fileutils`, `shellutils`, `findutils`, `textutils`, `grep`, `gzip`, and `tar`. BusyBox provides a fairly complete POSIX environment for any small or embedded system.

The utilities in BusyBox generally have fewer options than their full-featured GNU counterparts. The options that are included, however, provide the expected functionality and behave much like their GNU correlates.

The following sections describe the steps necessary for creating and booting a BlueCat Linux system containing BusyBox and demonstrate the use of the BusyBox utilities.

### Creating a BlueCat Linux System for BusyBox

To create a BlueCat Linux image for BusyBox, perform the following steps:

1. Create a new directory by typing:

```
BlueCat:$ mkdir -p \  
$BLUECAT_PREFIX/demo/busybox/local
```

2. Set up the BlueCat Linux kernel configuration using the standard kernel configuration tools and copy the kernel configuration file to the `$BLUECAT_PREFIX/demo/busybox` directory. For instance, type the following commands:

```
BlueCat:$ cd $BLUECAT_PREFIX/usr/src/linux  
BlueCat:$ make xconfig
```

Select **Save and Exit** to update the `.config` file, then type the following command:

```
BlueCat:$ cp .config \  
$BLUECAT_PREFIX/demo/busybox/busybox.config
```

---

**NOTE:** The kernel `.config` file for the `developer demo` (`$BLUECAT_PREFIX/demo/developer/developer.config`) is also recommended as a starting point.

---

3. Create the BlueCat Linux Kernel Downloadable Image (busybox.kernel):

```
BlueCat:$ cd $BLUECAT_PREFIX/demo/busybox
BlueCat:$ mkkernel ./busybox.config \
./busybox.kernel ./busybox.disk
```

4. Create a `.spec` file (busybox.spec) with the following minimal directives:

```
strip on

mkdir /dev
mknod /dev/console c 5 1

mkdir /lib
mkdir -p /usr/lib
mkdir /bin
mkdir /sbin
mkdir -p /etc/rc.d
mkdir /proc

cp ./local/fstab ./local/inittab /etc
cp ./local/rc.sysinit /etc/rc.d

lcd ${BLUECAT_PREFIX}/sbin
cp reboot busybox /sbin

ln -s /sbin/busybox /sbin/init
ln -s /sbin/busybox /sbin/ifconfig
ln -s /sbin/busybox /sbin/route
ln -s /sbin/busybox /bin/mount
ln -s /sbin/busybox /bin/sh
ln -s /sbin/busybox /bin/ping

chmod 711 /etc/rc.d/rc.sys.init
chmod 755 /bin /sbin
# End of File
```

5. Create the `local/fstab` file with the following contents:

```
proc /proc proc defaults 0 0
```

6. Create the `local/inittab` file with the following contents:

```
# System initialization.
::sysinit:/etc/rc.d/rc.sysinit

::respawn:/bin/sh
```

---

**NOTE:** The first two fields in every record of the `inittab` file are ignored by the BusyBox `init`, so they must be empty. For example, the line `1:12345:respawn:/bin/sh` is not valid.

---

7. Create the `local/rc.sysinit` file with the following contents:

```
#!/bin/sh

PATH=/bin:/sbin:/usr/bin:/usr/sbin
export PATH
mount -a
```

8. Create a root file system image (`busybox.rfs`) by entering the following command:

```
BlueCat:$ mkrootfs -lv ./busybox.spec ./busybox.rfs
```

9. Create a composite BlueCat Linux image:

```
BlueCat:$ mkboot -m -k busybox.disk -f \
busybox.rfs busybox.kdi
```

Refer to the *BlueCat Linux User's Guide* for more information about a BlueCat Linux composite image.

**NOTE:** The Makefile for the `developer` demo system can be used to produce the BusyBox kernel and RFS images. To do so, modify the Makefile as follows:

i. Change the line `KDI_NAME = developer` to `KDI_NAME = busybox`.

ii. Comment out the following lines:

```
# IS_NEED_REBUILD_SRC=$(shell if ! make -q -C src; then echo this ; fi)
# cd src; make all
```

iii. Change the following lines:

```
clean      :
            rm -f *.rfs *.tar *.kernel *.disk *.kdi *.srec; cd src; make clean
```

to read as follows:

```
clean      :
            rm -f *.rfs *.tar *.kernel *.disk *.kdi *.srec
```

iv. Run the `make all` command.

v. If the following message appears:

```
make: circular busybox <- busybox dependency dropped.
make: *** No rule to make target `*.rfs`, needed by `busybox`. stop.
```

edit the Makefile to delete the following line:

```
KDI_NAME = busybox
```

Then retype the line in. There may be trailing characters on this line that cause errors in the Makefile.

---

## Booting BusyBox Images from a Network

To boot BlueCat Linux with the BusyBox utility from a network using the BlueCat Linux OS loader, perform the steps listed below. Refer to Chapter 2, “Downloading and Booting BlueCat Linux on the Target” for additional details about the BlueCat Linux OS loader.

1. Follow the instructions given in “Installing BlueCat Linux into VPCS926 ROM” on page 5 to install the `osloader` demo system (`osloader.kdi`) into the target platform ROM.
2. Copy the `busybox.kernel` and `busybox.rfs` files from the `$BLUECAT_PREFIX/demo/busybox` directory to the `/tftpboot` directory on the cross-development host.
3. Start the VPCS926 Virtual Platform by pressing **F5**.

The BlueCat Linux OS loader prompt (`>`) will appear in the TTY0 console window.

4. At the OS loader prompt, type the following commands:

```
> set IF eth0
> set IP <target_board_IP>
> set HOST <development_host_IP>
> set KERNEL tftp busybox.kernel
> set RFS tftp busybox.rfs
> set CMD ramdisk_size=28472
> boot
```

where *<target\_board\_IP>* is the IP address of the target and *<development\_host\_IP>* is the IP address of the development host.

The BusyBox utility is loaded onto the target board and then automatically started.

## Using BusyBox Utilities

This section provides examples of using the BusyBox utilities. Entering a command from the following list results in the respective output:

- `ls`

```
/ # ls
bin          etc          lost+found  sbin
dev          lib          proc        usr
```
- `cat`

```
/ # cat /etc/inittab
# System initialization.
::sysinit:/etc/rc.d/rc.sysinit
::respawn:/bin/sh
```
- `chmod`

```
/ # chmod a-x /sbin/reboot
/ # ls -la /sbin/reboot
-rw-r--r--  1 0      0      7812 May  25 14:23 /sbin/reboot
/ # chmod 755 /sbin/reboot
/ # ls -la /sbin/reboot
-rwxr-xr-x  1 0      0      7812 May  25 14:23 /sbin/reboot
```
- `echo`

```
/ # echo !!!!!!!
!!!!!!!
```
- `date`

```
/ # date
Mon Nov  3 15:22:10 UTC 2003
```

- `uname`

```

/ # uname -a
Linux (none) 2.4.18-1 #7 Thu May 25 18:35:39 MSD 2004 armv5EJ1 unknown

```
- `mount`

```

/ # mount
/dev/root on / type ext2 (rw)
proc on /proc type proc (rw)

```
- `ifconfig`

```

/ # ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:00:00:00:00:00
          BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:10 Base address:0x320

/ # ifconfig eth0 172.16.1.70
/ # ping -c 5 172.16.1.70
PING 172.16.1.70 (172.16.1.70): 56 data bytes
64 bytes from 172.16.1.70: icmp_seq=0 ttl=255 time=1.1 ms
64 bytes from 172.16.1.70: icmp_seq=1 ttl=255 time=0.5 ms
64 bytes from 172.16.1.70: icmp_seq=2 ttl=255 time=0.5 ms
64 bytes from 172.16.1.70: icmp_seq=3 ttl=255 time=0.5 ms
64 bytes from 172.16.1.70: icmp_seq=4 ttl=255 time=0.5 ms

--- 172.16.1.70 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 0.5/0.6/1.1 ms
/ #

```

## Using the TinyLogin RPM Package

The TinyLogin RPM package is a suite of tiny UNIX utilities for handling logging into, being authenticated by, changing one's password for, and otherwise maintaining users and groups on an embedded system. It also provides shadow password support to enhance system security.

The following sections describe the steps necessary for creating and booting a BlueCat Linux system containing TinyLogin and demonstrate the use of the TinyLogin utility.

## Creating a BlueCat Linux System for TinyLogin

To create a BlueCat Linux image for TinyLogin, perform the following steps:

1. Create a new directory by typing:

```
BlueCat:$ mkdir -p \  
$BLUECAT_PREFIX/demo/tinylogin/local
```

2. Set up the BlueCat Linux kernel configuration by using the standard kernel configuration tools and copy the kernel configuration file to the `$BLUECAT_PREFIX/demo/tinylogin` directory. For instance, type the following commands:

```
BlueCat:$ cd $BLUECAT_PREFIX/usr/src/linux  
BlueCat:$ make xconfig
```

Select **Save and Exit** to update the `.config` file, then type the following command:

```
BlueCat:$ cp .config \  
$BLUECAT_PREFIX/demo/tinylogin/tinylogin.config
```

---

**NOTE:** The kernel `.config` file for the developer `demo` (`$BLUECAT_PREFIX/demo/developer/developer.config`) is also recommended as a starting point.

---

3. Create the BlueCat Linux Kernel Downloadable Image (tinylogin.kernel):

```
BlueCat:$ cd $BLUECAT_PREFIX/demo/tinylogin  
BlueCat:$ mkkernel ./tinylogin.config \  
./tinylogin.kernel ./tinylogin.disk
```

4. Create a `.spec` file (tinylogin.spec) that contains the following minimal directives:

```
strip on  
  
mkdir /dev  
mknod /dev/console c 5 1  
ln -s /dev/console /dev/tty  
ln -s /dev/console /dev/ttyl  
  
mkdir /bin  
mkdir /sbin  
mkdir -p /etc/rc.d  
mkdir /proc  
mkdir /tmp  
mkdir -p /usr/bin  
  
mkdir /root
```

```

mkdir /dev/pts
mknod /dev/ptmx c 5 2

chmod 0666 /dev/ptmx

cp ./local/fstab ./local/passwd ./local/inittab      /etc
cp ./local/securetty ./local/shadow                /etc
cp ./local/rc.sysinit                              /etc/rc.d
cp ${BLUECAT_PREFIX}/etc/shells                    /etc
chmod 644 /etc/shells
cp ${BLUECAT_PREFIX}/etc/group                      /etc

lcd ${BLUECAT_PREFIX}/sbin
cp reboot init mingetty /sbin

cp ${BLUECAT_PREFIX}/usr/bin/tinylogin              /usr/bin
ln -s /usr/bin/tinylogin                           /usr/bin/passwd
ln -s /usr/bin/tinylogin                           /bin/login

lcd ${BLUECAT_PREFIX}/bin
cp mount bash ls cat hostname /bin
ln -s /bin/bash /bin/sh

chmod 711 /etc/rc.d/rc.sysinit

chmod 755 /bin /sbin /usr/bin

chmod 04755 /usr/bin/tinylogin
# End of Filee

```

---

**NOTE:** In this `.spec` file, the `/bin/login` and `/usr/bin/passwd` symbolic links point to `/usr/bin/tinylogin`. This allows the user to change his/her password by simply typing `passwd`.

---

5. Create the `local/fstab` file with the following contents:

```

none /proc      proc
none /dev/pts  devpts

```

6. Create the `local/inittab` file with the following contents:

```

id:1:initdefault:

# System initialization.
si::sysinit:/etc/rc.d/rc.sysinit

1:12345:respawn:/sbin/mingetty tty1

```

7. Create the `local/securetty` file with the following contents:

```

console
tty1

```

8. Create the `local/passwd` file with the following contents:

```
root:x:0:0:/root:/:/bin/bash
guest:x:500:10:/:/bin/bash
```

9. Create the `local/shadow` file:

```
root::10942:0:99999:7:::
guest::500:10:99999:7:::
```

10. Create the `local/rc.sysinit` file with the following contents:

```
#!/bin/sh

PATH=/bin:/sbin:/usr/bin:/usr/sbin
export PATH

mount -a
hostname myhostname
```

11. Create a root file system image (`tinylogin.rfs`) by entering the following command:

```
BlueCat:$ mkrootfs -lv ./tinylogin.spec \  
./tinylogin.rfs
```

12. Create a composite BlueCat Linux image:

```
BlueCat:$ mkboot -m -k tinylogin.disk -f \  
tinylogin.rfs tinylogin.kdi
```

---

**NOTE:** The Makefile for the `developer` demo system can be used to produce the TinyLogin kernel and RFS images. To do so, modify the Makefile as follows:

- i. Change the line `KDI_NAME = developer` to  
`KDI_NAME = tinylogin`.

- ii. Comment out the following lines:

```
# IS_NEED_REBUILD_SRC=$(shell if ! make -q -C src; then echo this ; fi)
# cd src; make all
```

- iii. Change the following lines:

```
clean      :
            rm -f *.rfs *.tar *.kernel *.disk *.kdi *.srec; cd src; make clean
```

to read as follows:

```
clean      :
            rm -f *.rfs *.tar *.kernel *.disk *.kdi *.srec
```

- iv. Run the `make all` command.

- v. If the following message appears:

```
make: circular tinylogin <- tinylogin dependency dropped.
make: *** No rule to make target `*.rfs`, needed by `tinylogin`. stop.
```

edit the Makefile to delete the following line:

```
KDI_NAME = tinylogin
```

Then retype the line in. There may be trailing characters on this line that cause errors in the Makefile.

---

## Booting the TinyLogin Images from a Network

To boot BlueCat Linux with the TinyLogin utility from a network using the BlueCat Linux OS loader, perform the steps listed below. Refer to Chapter 2, “Downloading and Booting BlueCat Linux on the Target” for additional details about the BlueCat Linux OS loader.

1. Follow the instructions given in “Installing BlueCat Linux into VPCS926 ROM” on page 5 to install the `osloader` demo system (`osloader.kdi`) into the target platform ROM.
2. Copy the `tinylogin.kernel` and `tinylogin.rfs` files from the `$BLUECAT_PREFIX/demo/tinylogin` directory to the `/tftpboot` directory on the cross-development host.

3. Start the VPCS926 Virtual Platform by pressing **F5**.

The BlueCat Linux OS loader prompt (>) will appear in the TTY0 console window.

4. At the OS loader prompt, type the following commands:

```
> set IF eth0
> set IP <target_board_IP>
> set HOST <development_host_IP>
> set KERNEL tftp tinylogin.kernel
> set RFS tftp tinylogin.rfs
> set CMD ramdisk_size=28472
> boot
```

where <target\_board\_IP> is the IP address of the target and <development\_host\_IP> is the IP address of the development host.

The TinyLogin utility is loaded onto the target board and then automatically started.

## Using the TinyLogin Utility

This section provides examples of using the TinyLogin utility:

- Changing the guest password:

```
myhostname login: guest
bash-2.04$ passwd
Changing password for guest
Enter the new password (minimum of 5, maximum of 8 characters)

Please use a combination of upper and lower case letters and numbers.
Enter new password: <new_guest_password>
Re-enter new password: <new_guest_password>
passwd[13]: password for `guest' changed by user `guest' Password
changed.
bash-2.04$ exit
```

- Changing the root password:

```
myhostname login: root
login[16]: root login on `console'

bash-2.04# passwd
Changing password for root
Enter the new password (minimum of 5, maximum of 8 characters)

Please use a combination of upper and lower case letters and numbers.
Enter new password: <new_root_password>
Re-enter new password: <new_root_password>
```

```
passwd[16]: password for `root' changed by user `root'
Password changed.
bash-2.04# exit
logout
myhostname login: root
Password: <new_root_password>
login[17]: root login on `console'

bash-2.04# exit
logout
```

- Getting the root permissions:

```
myhostname login: guest
Password: <guest_password>
bash-2.04$ tinylogin su
Password:
login[19]: root login on `console'

bash-2.04#
```

## Using the Zebra RPM Package

GNU Zebra is free software that manages a TCP/IP-based routing protocol. It takes a multiserver and multithread approach to resolve the current complexity of the Internet.

GNU Zebra supports BGP4, BGP4+, OSPFv2, OSPFv3, RIPv1, RIPv2, and RIPng.

GNU Zebra is intended to be used as a Route Server and a Route Reflector. It is not a toolkit; it provides full routing power under a new architecture. GNU Zebra is unique in design in that it has a process for each protocol.

The following sections describe the steps necessary for creating and booting a BlueCat Linux system containing Zebra and demonstrate the use of the Zebra utility.

## Creating a BlueCat Linux System for Zebra

To create a BlueCat Linux image for Zebra, perform the following steps:

1. Create a new directory by typing:

```
BlueCat:~$ mkdir -p $BLUECAT_PREFIX/demo/zebra/local
```

2. Set up the BlueCat Linux kernel configuration by using the standard kernel configuration tools and copy the kernel configuration file to the

`$BLUECAT_PREFIX/demo/zebra` directory. For instance, type the following commands:

```
BlueCat:$ cd $BLUECAT_PREFIX/usr/src/linux
BlueCat:$ make xconfig
```

Select **Save and Exit** to update the `.config` file, then type the following command:

```
BlueCat:$ cp .config \
$BLUECAT_PREFIX/demo/zebra/zebra.config
```

---

**NOTE:** In the kernel configuration, the following options must be set to **Y**:

```
CONFIG_NETLINK=Y
CONFIG_RTNETLINK=Y
```

By default, Zebra is configured to communicate with the kernel via the netlink socket.

---

3. Create the BlueCat Linux Kernel Downloadable Image (`zebra.kernel`):

```
BlueCat:$ cd $BLUECAT_PREFIX/demo/zebra
BlueCat:$ mkkernel ./zebra.config ./zebra.kernel \
./zebra.disk
```

4. Create a `.spec` file (`zebra.spec`) that contains the following minimal directives:

```
strip on

mkdir /dev
mknod /dev/console c 5 1
ln -s /dev/console /dev/tty
ln -s /dev/console /dev/tty1
# Standard 16550 serial driver device
mknod /dev/ttyS0 c 4 64
mknod /dev/ttyS1 c 4 65

mkdir -p /lib/security
mkdir -p /usr/lib
mkdir /bin
mkdir /sbin
mkdir -p /etc/rc.d
mkdir -p /etc/pam.d
mkdir -p /etc/xinetd.d
mkdir -p /etc/zebra
mkdir /proc
mkdir /tmp
mkdir -p /usr/bin
mkdir -p /usr/sbin
mkdir -p /var/run
mkdir -p /usr/libexec
```

```

mkdir -p /var/log/zebra

mkdir /root

mkdir /dev/pts
mknod /dev/ptmx c 5 2

chmod 0666 /dev/ptmx

cp ./local/fstab ./local/passwd ./local/inittab ./local/mtab /etc
cp ./local/other /etc/pam.d
cp ./local/rc.sysinit /etc/rc.d
cp ./local/hosts /etc
cp ./local/protocols /etc
cp ./local/resolv.conf /etc
cp ${BLUECAT_PREFIX}/etc/pwdb.conf /etc
cp ${BLUECAT_PREFIX}/etc/nsswitch.conf /etc
cp ${BLUECAT_PREFIX}/etc/services /etc

cp ${BLUECAT_PREFIX}/etc/security /etc

cp ./local/shadow /etc
cp ./local/pam.d /etc
cp ./local/xinetd.d/* /etc/xinetd.d
cp ./local/zebra.conf /etc/zebra/

cp ${BLUECAT_PREFIX}/lib/libnss_files-*.so /lib
cp ${BLUECAT_PREFIX}/lib/libnss_dns-*.so /lib
cp ${BLUECAT_PREFIX}/lib/libpwdb.so /lib
cp ${BLUECAT_PREFIX}/lib/security /lib

cp ./local/empty /var/log/wtmp

lcd ${BLUECAT_PREFIX}/sbin
cp reboot init mingetty ifconfig /sbin

cp ${BLUECAT_PREFIX}/lib/security/pam_permit.so /lib/security

cp ${BLUECAT_PREFIX}/etc/xinetd.conf /etc

cp ${BLUECAT_PREFIX}/usr/bin/telnet /usr/bin

cp ${BLUECAT_PREFIX}/etc/shells /etc
chmod 644 /etc/shells

cp ${BLUECAT_PREFIX}/etc/group /etc

#
# General Binaries
#
lcd ${BLUECAT_PREFIX}/bin
cp ping mount bash cat ls hostname ps /bin
cp login /bin
ln -s /bin/bash /bin/sh

cp ${BLUECAT_PREFIX}/usr/bin/vtyssh /usr/bin

# internet services utils
cp ${BLUECAT_PREFIX}/usr/sbin/xinetd /usr/sbin
cp ${BLUECAT_PREFIX}/usr/sbin/in.telnetd /usr/sbin
cp ${BLUECAT_PREFIX}/usr/sbin/zebra /usr/sbin

```

```
chmod 711 /etc/rc.d/rc.sysinit
chmod 755 /bin /sbin /usr/bin /usr/sbin
# End of File
```

### 5. Create the `local/inittab` file with the following contents:

```
id:1:initdefault:
# System initialization.
si::sysinit:/etc/rc.d/rc.sysinit
l0:0:wait:/sbin/halt
l6:6:wait:/sbin/reboot
ca::ctrlaltdel:/sbin/shutdown -t3 -r now
pf::powerfail:/sbin/shutdown -f -h +2 "Power Failure; System Shutting
Down"
pr:12345:powerokwait:/sbin/shutdown -c "Power Restored; Shutdown
Cancelled"
1:12345:respawn:/sbin/mingetty tty1
```

### 6. Create the `local/rc.sysinit` file with the following contents:

```
#!/bin/sh
PATH=/bin:/sbin:/usr/bin:/usr/sbin
export PATH
mount -a
xinetd -stayalive -reuse
hostname myhostname
zebra -d
```

### 7. Create the `local/zebra.conf` file with the following contents:

```
!
! zebra configuration file
!
hostname Router
password zebra
enable password zebra
!
! Interface's description.
!
interface lo
ip address 127.0.0.1/8
interface eth0
ip address 172.16.1.33/16
!
```

```
! Static default route.  
!  
ip route 213.240.0.0 255.255.0.0 172.16.1.70  
  
log stdout
```

---

**NOTE:** This configuration file sets the password to `zebra`. The user has to enter this password when connecting to Zebra or changing the Zebra configuration mode by entering the `enable` command at the command prompt.

---

8. Copy the `fstab`, `passwd`, `mtab`, `other`, `hosts`, `protocols`, `resolv.conf`, `shadow`, `pam.d/*`, `xinetd.d/*`, and empty files from the `$BLUECAT_PREFIX/demo/developer/local` directory to the `$BLUECAT_PREFIX/demo/zebra/local` directory.

9. Create a root file system image (`zebra.rfs`) by entering the following command:

```
BlueCat:$ mkrootfs -lv ./zebra.spec ./zebra.rfs
```

10. Create a composite BlueCat Linux image:

```
BlueCat:$ mkboot -m -k zebra.disk -f \  
zebra.rfs zebra.kdi
```

**NOTE:** The `Makefile` for the `developer` demo system can be used to produce the Zebra kernel and RFS images. To do so, modify the `Makefile` as follows:

- i. Change the line `KDI_NAME = developer` to `KDI_NAME = zebra`.
- ii. Comment out the following lines:

```
# IS_NEED_REBUILD_SRC=$(shell if ! make -q -C src; then echo this ; fi)
# cd src; make all
```

- iii. Change the following lines:

```
clean      :
            rm -f *.rfs *.tar *.kernel *.disk *.kdi *.srec; cd src; make clean
```

to read as follows:

```
clean      :
            rm -f *.rfs *.tar *.kernel *.disk *.kdi *.srec
```

- iv. Run the `make all` command.
- v. If the following message appears:

```
make: circular zebra <- zebra dependency dropped.
make: *** No rule to make target `*.rfs', needed by `zebra'. stop.
```

edit the `Makefile` to delete the following line:

```
KDI_NAME = zebra
```

Then retype the line in. There may be trailing characters on this line that cause errors in the `Makefile`.

---

## Booting the Zebra Images from a Network

To boot BlueCat Linux with the Zebra utility from a network using the BlueCat Linux OS loader, perform the steps listed below. Refer to Chapter 2, “Downloading and Booting BlueCat Linux on the Target” for additional details about the BlueCat Linux OS loader.

1. Follow the instructions given in “Installing BlueCat Linux into VPCS926 ROM” on page 5 to install the `osloader` demo system (`osloader.kdi`) into the target platform ROM.
2. Copy the `zebra.kernel` and `zebra.rfs` files from the `$(BLUECAT_PREFIX)/demo/zebra` directory to the `/tftpboot` directory on the cross-development host.
3. Start the VPCS926 Virtual Platform by pressing **F5**.

The BlueCat Linux OS loader prompt (`>`) will appear in the TTY0 console window.

4. At the OS loader prompt, type the following commands:

```
> set IF eth0
> set IP <target_board_IP>
> set HOST <development_host_IP>
> set KERNEL tftp zebra.kernel
> set RFS tftp zebra.rfs
> set CMD ramdisk_size=28472
> boot
```

where *<target\_board\_IP>* is the IP address of the target and *<development\_host\_IP>* is the IP address of the development host.

The Zebra utility is loaded onto the target board and then automatically started.

## Using the Zebra Utility

This section provides examples of using the Zebra utility:

```
myhostname login: root
bash-2.04# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:00:5E:01:00:20
          inet addr:172.16.1.33 Bcast:172.16.255.255  Mask:255.255.0.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:164 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          Interrupt:14

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0

bash-2.04# ping -c 2 172.16.1.70
PING 172.16.1.70 (172.16.1.70) from 172.16.1.33 : 56(84) bytes of data.
64 bytes from 172.16.1.70: icmp_seq=0 ttl=255 time=946 usec
64 bytes from 172.16.1.70: icmp_seq=1 ttl=255 time=470 usec

--- 172.16.1.70 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/mdev = 0.470/0.708/0.946/0.238 ms

bash-2.04# telnet localhost 2601
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.

Hello, this is zebra (version 0.91a).
Copyright 1996-2001 Kunihiro Ishiguro.
```

User Access Verification

Password:

Router> **enable**

Password:

Router# show ip route

Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF,  
B - BGP, > - selected route, \* - FIB route

C>\* **127.0.0.0/8 is directly connected, lo**

C>\* **172.16.0.0/16 is directly connected, eth0**

S>\* **213.24.0.0/16 [1/0] via 172.17.0.1, eth0**

Router#

Table 5-1 lists the device drivers supported by the vpcs926 BSP and provides important information about them.

**Table 5-1: Device Drivers Supported by the vpcs926 BSP**

<b>Hardware Device</b>	<b>Device Drivers</b>	<b>Location in Source Tree</b>	<b>Kernel Configuration Options</b>
UART	<code>serial.c</code>	<code>drivers/serial</code>	<code>CONFIG_SERIAL</code> <code>CONFIG_SERIAL_CONSOLE</code>
Ethernet ApE110	<code>ape110.c</code>	<code>drivers/net</code>	<code>CONFIG_APE110</code>



This chapter describes known problems and limitations of this release.

---

## **VPCS926 Target Platform Problems and Limitations**

- If `mkrootfs` is terminated (either by an error or by a signal), it tries to clean all its temporary files before exiting. Due to certain features of the Cygwin execution environment, however, such temporary files can remain uncleaned in the `/tmp` directory on a Windows host. It is recommended that the `/tmp` directory be regularly checked and cleaned.
- The `tc1x` RPM package is not included in the Windows-hosted distribution.
- On Windows hosts, some file permissions (including `r` and `s`) always have default values. To set permissions different from the default values, the `chmod` command should be used in the `.spec` file.

