

BlueCat Linux Board Support Guide

BlueCat Linux Release 4.1

DOC-0584-00

for Motorola PQ2FADS-ZU Boards

Product names mentioned in the *BlueCat Linux Board Support Guide for Motorola PQ2FADS-ZU Boards* are trademarks of their respective manufacturers and are used here only for identification purposes.

Copyright ©1987 - 2003, 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 Motorola PQ2FADS-ZU 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

	Typographical Conventions	v
	Special Notes	vi
	Technical Support	vi
CHAPTER 1	OVERVIEW	1
CHAPTER 2	DOWNLOADING AND BOOTING BLUECAT LINUX ON THE TARGET	3
	Prerequisites	3
	Downloading and Booting Overview	4
	Setting up Hardware	5
	Configuring On-Board Switches and Jumpers	5
	Connecting the Target Board Serial Ports to the Host	5
	Connecting the Target Board Ethernet Card to the Host	6
	Installing LynuxWorks Boot Loader into Flash	6
	Setting Up the LynuxWorks Boot Loader Firmware	6
	Downloading a BlueCat Linux System into Flash	7
	Downloading a BlueCat Linux System into Flash Using LynuxWorks Boot Loader	7
	Booting a BlueCat Linux System from Flash	8
	Booting a BlueCat Linux System from a Network	8
	Booting a BlueCat Linux System from a Network Using LynuxWorks Boot Loader	9
	Booting a BlueCat Linux System from a Network Using the OS Loader	9

CHAPTER 3	KERNEL CONFIGURATION OPTIONS	11
CHAPTER 4	SUPPORTED DEMO SYSTEMS.....	31
	Demo Systems	31
	developer Demo System	31
	osloader Demo System	32
	showcase Demo System	32
	Using Selected RPM Packages	32
	Using the BusyBox RPM Package	32
	Using the TinyLogin RPM Package	37
	Using the Zebra RPM Package	42
CHAPTER 5	SUPPORTED DEVICE DRIVERS	49
CHAPTER 6	NEW FEATURES	51
	Supported Cross-Development Hosts	51
	JFFS2 Support	51
	Support of Multithreaded Applications in GDB	52
	Cygwin Execution Environment Version 1.3.6	52
CHAPTER 7	KNOWN PROBLEMS AND LIMITATIONS.....	53
	PQ2FADS Target Board Problems and Limitations	53

Preface

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; *italicized* for emphasis, new terms, and book titles

Refer to the *BlueCat Linux User's Guide*.

Environment variables, file names, functions, methods, options, parameter names, path names, commands, and computer data

```
ls
-l
myprog.c
/dev/null
```

Commands that need to be highlighted within body text, or commands that must be typed as is by the user are **bolded**.

```
login: myname
# cd /usr/home
```

Text that represents a variable, such as a file name or a value that must be entered by the user

```
cat <filename>
mv <file1> <file2>
```

Blocks of text that appear on the display screen after entering instructions or commands

```
Loading file /tftpboot/shell.kdi
into 0x4000
.....
File loaded. Size is 1314816
Copyright 2002 LynuxWorks, Inc.
All rights reserved.

LynxOS (ppc) created Mon Jan 17
17:50:22 GMT 2002
user name:
```

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

LynuxWorks Technical Support is available Monday through Friday (holidays excluded) between 8:00 AM and 5:00 PM Pacific Time (U.S. Headquarters) or between 9:00 AM and 6:00 PM Central European Time (Europe).

LynuxWorks U.S. Headquarters

email support@lnxw.com

phone (408) 979-3940
(800) 327-5969

fax (408) 979-3945

LynuxWorks Europe

email tech_europe@lnxw.com

phone (+33) 1 30 85 06 00

fax (+33) 1 30 85 06 06

World Wide Web

website <http://www.lynuxworks.com>

customer support <http://www.lynuxworks.com/support/custhelp.php3>

The *BlueCat Linux Board Support Guide for Motorola PQ2FADS-ZU Boards* provides information about the BlueCat Linux Board Support Package (BSP) for the Motorola PQ2FADS-ZU board. The PQ2FADS-ZU boards are designed to aid hardware and software developers of the PowerQUICC II family by providing an evaluation platform for MPC82xx derivatives. This BlueCat Linux port targets the MPC8280 (HiP7) processor.

Throughout this Board Support Guide (BSG), the BSP is referred to as the “pq2fads” and the board as the “PQ2FADS” or simply as the “target board.”

The chapters of this BSG provide the information listed below:

- *Chapter 1* is an overview of this BSG’s individual chapters.
- *Chapter 2* describes BlueCat Linux downloading and booting procedures for the PQ2FADS target board using the BlueCat Linux `showcase` and `developer` demo systems as examples.
- *Chapter 3* provides configuration information about the pq2fads BSP’s default BlueCat Linux kernel.
- *Chapter 4* describes BlueCat Linux demo systems supported by the pq2fads BSP.
- *Chapter 5* provides a list of pq2fads BSP-supported device drivers, with important information about each of them.
- *Chapter 6* describes new features of this release.
- *Chapter 7* describes known limitations and workarounds for this release.

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 to downloading and booting BlueCat Linux systems onto the user's target board. 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 board as well as system administration reference material for the cross-development host.

Before downloading and booting BlueCat Linux on the target board, it is assumed that the default BlueCat Linux PowerPC configuration and the pq2fads BSP have been installed on the cross-development host. This means that the user must:

1. Install the BlueCat Linux PowerPC Core onto the cross-development host, as described in the "Installing the Default Configuration" section in Chapter 1, "Installation" in the *BlueCat Linux User's Guide*.
2. Install the pq2fads BSP onto the cross-development host as detailed in the "Installing Target Board Support" section of Chapter 1, "Installation" in the *BlueCat Linux User's Guide*.
3. Activate support for the pq2fads BSP as detailed in the "Activating Support for a Target Board" section of Chapter 1, "Installation" in the *BlueCat Linux User's Guide*.

Downloading and Booting Overview

The procedure for downloading and booting a BlueCat Linux system on the PQ2FADS target consists of the following main steps:

- Setting up hardware
- Downloading and booting a BlueCat Linux system from target Flash memory or a network

Downloading and booting a BlueCat Linux system can be performed using:

- LynuxWorks Boot Loader
- OR
- BlueCat Linux OS loader

The LynuxWorks Boot Loader is a firmware intended to act as a boot loader for BlueCat Linux and LynxOS. Boot Loader is able to load BlueCat Linux embedded systems from Flash or over a network and to program them to Flash.

The BlueCat Linux OS loader demo system includes the `i_osloader` and `osloader` downloadable images. `osloader` is the image with the base functionality of the BlueCat Linux OS loader configured in. This includes the ability to download BlueCat Linux images from a TFTP host, execute them in RAM, and other important features.

`i_osloader` is extended with support for the Journalling Flash File System (JFFS) and can thus be used to download a desired BlueCat Linux custom or demo system into the target board's Flash memory. Please refer to Chapter 3, "Downloading and Booting BlueCat Linux" in the *BlueCat Linux User's Guide* for a discussion of the OS loader.

Setting up Hardware

Configuring On-Board Switches and Jumpers

Prior to using the board, the following on-board switches and jumpers need to be configured:

Table 2-1: On-Board Switches and Jumpers Configuration

Switch/Jumper	Setting	Description
JP7	2-3	Hard Reset Configuration is taken from BCSR
JP9	2-3	Local SDRAM Enabled
SW4.1	Off	Normal Boot Loader mode
SW4.2	On	Normal Boot Loader mode

Connecting the Target Board Serial Ports to the Host

The target board has two serial ports arranged in a single stack. The upper port in the stack is named P1A, and the lower port in the stack is named P1B. The P1A serial port is used both by the LynuxWorks Boot Loader and the BlueCat Linux system console.

Before using the board, at least the first serial port needs to be connected to the development host. It is recommended that the user connect the target serial connector to COM1 on the host.

The serial port settings on the host must be as follows:

- The serial port connected to the P1A target serial port has a baud rate of 9600.
- The serial port connected to the P1B target serial port can have any baud rate.

Throughout this chapter, the terminal window connected to the first serial connector is referred to as the “Boot Loader console” or the “BlueCat Linux console,” depending on the context.

Connecting the Target Board Ethernet Card to the Host

The Motorola PQ2FADS board provides 10BaseT and 100BaseTX Ethernet connections via Category 5 Unshielded Twisted Pair cables using two on-board RJ-45 connectors, which are referenced to as P3 and P4 in the manufacture's documentation.

The Ethernet ports on the target board are used to provide a standard network connection for the board and, in particular, to load BlueCat Linux embedded systems onto the board from a network. However, note that the LynuxWorks Boot Loader supports the P3 Ethernet port only.

The Ethernet ports on the PQ2FADS board are used to connect to a LAN.

It is also required that the user set up networking on the host system. In particular, the user must choose a unique IP address for the development host as well as for the target board. These addresses are referred to as `<host_IP>` and `<target_IP>`, respectively. For more information on how to set up networking on the host, please refer to system administration reference material.

TFTP must be enabled on the host. For more information, refer to "Setting Up a TFTP Server" in Chapter 3, "Downloading and Booting BlueCat Linux" in the *BlueCat Linux User's Guide*.

Installing LynuxWorks Boot Loader into Flash

To install the LynuxWorks Boot Loader onto the PQ2FADS board, the composite Boot Loader image (`pq2fads.bin`) needs to be programmed to the last megabyte of the Flash memory using either an external programming tool or a COP debugger.

Setting Up the LynuxWorks Boot Loader Firmware

To set up the LynuxWorks Boot Loader firmware for BlueCat Linux operations, perform the following steps:

1. Reset the target board.

2. At the LynuxWorks Boot Loader console, enter the following commands:

```

pq2fads> set autoboot 0
pq2fads> set boot_tftp_host_ip development_host_IP
pq2fads> set boot_tftp_client_ip target_board_IP
pq2fads> set flash_tftp_host_ip \
development_host_IP
pq2fads> set flash_tftp_client_ip target_board_IP
pq2fads> save

```

where *target_board_IP* is the IP address of the target and *development_host_IP* is the IP address of the development host.

3. Reset the target board.

Downloading a BlueCat Linux System into Flash

This section provides instructions on how a BlueCat Linux embedded system can be downloaded into the target Flash memory using the LynuxWorks Boot Loader firmware. Refer also to the *BlueCat Linux User's Guide* for additional details about the BlueCat Linux OS loader.

Downloading a BlueCat Linux System into Flash Using LynuxWorks Boot Loader

To download `developer` into the primary Flash of the target board using LynuxWorks Boot Loader, perform the following steps:

1. Copy the `developer.kdi` file from the `$BLUECAT_PREFIX/demo/developer` directory to the `/tftpboot` directory on the development host.
2. Reset the target board.
3. At the LynuxWorks Boot Loader console, enter the following commands:

```

pq2fads> set flash_device tftp
pq2fads> set flash_tftp_file developer.kdi
pq2fads> set flash_target flash0
pq2fads> set flash_offset 0
pq2fads> flash

```

After these commands have been performed, the `developer` demo system is programmed into the primary Flash and can be booted as described in “Booting a BlueCat Linux System from Flash” below.

Booting a BlueCat Linux System from Flash

To boot a demo installed into the primary Flash memory, perform the steps listed below. For a detailed information on how to install the demo system to Flash, refer to “Downloading a BlueCat Linux System into Flash Using LynuxWorks Boot Loader” on page 7.

1. Reset the target board.
2. At the LynuxWorks Boot Loader console, type the following:

```
pq2fads> set boot_device flash0
pq2fads> set boot_flash_offset 0
pq2fads> set boot_os BlueCat
pq2fads> boot
```

These commands will start the demo system programmed into the primary Flash at offset 0.

The PQ2FADS board can be configured to start a demo system programmed into Flash automatically at the board power-up. Use the following commands to prepare the PQ2FADS board to boot BlueCat Linux from primary Flash automatically:

```
pq2fads> set boot_device flash0
pq2fads> set boot_flash_offset 0
pq2fads> set boot_os BlueCat
pq2fads> set autoboot 1
pq2fads> save
```

The demo system programmed into primary Flash is started by the Boot Loader monitor automatically on board power-up.

Booting a BlueCat Linux System from a Network

A BlueCat Linux demo system can be booted from a network using either the RedBoot firmware or LynuxWorks Boot Loader firmware.

Booting a BlueCat Linux System from a Network Using LynuxWorks Boot Loader

To boot the `developer` demo system over a network using the LynuxWorks Boot Loader firmware, perform the following steps:

1. Copy the `developer.kdi` file from the `$BLUECAT_PREFIX/demo/developer` directory to the `/tftpboot` directory on the cross-development host.
2. Reset the target board.
3. At the LynuxWorks Boot Loader console, enter the following commands:

```
pq2fads> set boot_device tftp
pq2fads> set boot_tftp_file developer.kdi
pq2fads> boot
```

These commands load the `developer` demo system from a network onto the target board and then automatically start it.

Booting a BlueCat Linux 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. Copy the `osloader.srec` file from the `$BLUECAT_PREFIX/demo/osloader` directory to the `/tftpboot` directory on the development host.
2. Copy the `showcase.kernel` and `showcase.rfs` files from the `$BLUECAT_PREFIX/demo/showcase` directory to the `/tftpboot` directory on the cross-development host.
3. Reset the target board.

4. At the LynuxWorks Boot Loader console, enter the following commands:

```
pq2fads> set boot_device tftp
pq2fads> set boot_tftp_file osloader.kdi
pq2fads> set boot_os BlueCat
pq2fads> boot
```

These commands start the osloader demo system from RAM. As a result, the BlueCat Linux OS loader prompt will appear in the BlueCat Linux console.

5. 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 showcase.kernel
> set RFS tftp showcase.rfs
> set CMD ramdisk_size=16384
> boot
```

where *target_board_IP* is an IP address of the target and *development_host_IP* is an IP address of the development host.

These commands load the showcase demo system from a network onto the target board and then automatically start it.

Kernel Configuration Options

The pq2fads 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: “BlueCat Linux Default Configuration for the pq2fads BSP Distribution” below.

Table 3-1: BlueCat Linux Default Configuration for the pq2fads BSP Distribution

Table Number and Configuration Parameter
Table 3-2: “Code Maturity Level Options”
Table 3-3: “Loadable Module Support”
Table 3-4: “Platform Support”
Table 3-5: “General Setup”
Table 3-6: “Parallel Port Support”
Table 3-7: “Memory Technology Devices (MTD)”
Table 3-8: “RAM/ROM/Flash Chip Drivers”
Table 3-9: “Mapping Drivers for Chip Access”
Table 3-10: “Self-Contained MTD Device Drivers”
Table 3-11: “NAND Flash Device Drivers”
Table 3-12: “Plug and Play Configuration”
Table 3-13: “Block Devices”
Table 3-14: “Multidevice Support (RAID and LVM)”
Table 3-15: “Networking Options”
Table 3-16: “QoS and/or Fair Queueing”
Table 3-17: “ATA/IDE/MFM/RLL Support”

Table 3-1: BlueCat Linux Default Configuration for the pq2fads BSP Distribution (Continued)

Table Number and Configuration Parameter
Table 3-18: "SCSI Support"
Table 3-19: "Network Device Support"
Table 3-20: "ARCnet Devices"
Table 3-21: "Ethernet (10 or 100Mbit)"
Table 3-22: "Ethernet (1000 Mbit)"
Table 3-23: "Wireless LAN (Non-Ham Radio)"
Table 3-24: "Token Ring Devices"
Table 3-25: "WAN Interfaces"
Table 3-26: "Amateur Radio Support"
Table 3-27: "IrDA (Infrared) Support"
Table 3-28: "ISDN Subsystem"
Table 3-29: "Old CD-ROM Drivers (not SCSI, not IDE)"
Table 3-30: "Console Drivers"
Table 3-31: "Frame Buffer Support"
Table 3-32: "Input Core Support"
Table 3-33: "Character Devices"
Table 3-34: "Serial Drivers"
Table 3-35: "I2C Support"
Table 3-36: "L3 Serial Bus Support"
Table 3-37: "Mice"
Table 3-38: "Joysticks"
Table 3-39: "Watchdog Cards"
Table 3-40: "Ftape, the Floppy Tape Device Driver"
Table 3-41: "Multimedia Devices"
Table 3-42: "File Systems"
Table 3-43: "Network File Systems"

Table 3-1: BlueCat Linux Default Configuration for the pq2fads BSP Distribution (Continued)

Table Number and Configuration Parameter
Table 3-44: "Partition Types"
Table 3-45: "Sound"
Table 3-46: "MPC8260 CPM Options"
Table 3-47: "USB Support"
Table 3-48: "Bluetooth Support"
Table 3-49: "Kernel Hacking"
Table 3-50: "Modular Advanced Power Management"

Table 3-2: Code Maturity Level Options

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

Table 3-3: Loadable Module Support

Options	Value	Description
CONFIG_MODULES	Y	Enable loadable module support
CONFIG_MODVERSIONS	Y	Set version information on all module symbols
CONFIG_KMOD	Y	Kernel module loader

Table 3-4: Platform Support

Options	Value	Description
CONFIG_6xx	Y	Processor type
CONFIG_8260	Y	MPC8260 CPM support

Table 3-4: Platform Support (Continued)

Options	Value	Description
CONFIG_PQ2FADS	Y	Machine type
CONFIG_EST8260	N	Support for EST8260
CONFIG_SMP	N	Symmetric multiprocessing support

Table 3-5: General Setup

Options	Value	Description
CONFIG_HIGHMEM	N	High memory support (experimental)
CONFIG_PCI	N	Enable PCI
CONFIG_PC_KEYBOARD	N	PC PS/2-style keyboard
CONFIG_BLUECAT_IGNORE_PRINTK	N	BlueCat Linux Ignore printk
CONFIG_BLUECAT_LOADER	N	BlueCat Linux OS loader
CONFIG_BLUECAT_SMALL_FOOTPRINT	N	BlueCat Linux small memory footprint
CONFIG_NET	Y	Networking support
CONFIG_BLUECAT_MEMSIZE	N	Memory sizing benchmarks
CONFIG_SYSCTL	N	Sysctl support
CONFIG_SYSVIPC	N	System V IPC
CONFIG_BSD_PROCESS_ACCT	N	BSD Process Accounting
CONFIG_BINFORMAT_MISC	N	Kernel support for MISC binaries
CONFIG_HOTPLUG	N	Support for hot-pluggable devices

Table 3-6: Parallel Port Support

Options	Value	Description
CONFIG_PARPORT	N	Parallel port support
CONFIG_GEN_RTC	N	Generic /dev/rtc emulation
CONFIG_CMDLINE_BOOL	N	Default boot loader kernel arguments

Table 3-7: Memory Technology Devices (MTD)

Options	Value	Description
CONFIG_MTD	Y	Memory Technology Device (MTD) support
CONFIG_MTD_DEBUG	N	Debugging
CONFIG_MTD_PARTITIONS	Y	MTD partitioning support
CONFIG_MTD_CONCAT	Y	MTD concatenating support
CONFIG_MTD_REDBOOT_PARTS	N	RedBoot partition table parsing
CONFIG_MTD_CMDLINE_PARTS	N	Command line partition table parsing
CONFIG_MTD_CHAR	Y	Direct char device access to MTD devices
CONFIG_MTD_BLOCK	Y	Caching block device access to MTD devices
CONFIG_FTL	N	Flash Translation Layer (FTL) support
CONFIG_NFTL	N	NAND Flash Translation Layer (NFTL) support

Table 3-8: RAM/ROM/Flash Chip Drivers

Options	Value	Description
CONFIG_MTD_CFI	N	Detect Flash chips by Common Flash Interface (CFI) probe
CONFIG_MTD_JEDECPROBE	N	Detect JEDEC JESD21c-compatible Flash chips
CONFIG_MTD_RAM	N	Support for RAM chips in bus mapping
CONFIG_MTD_ROM	N	Support for ROM chips in bus mapping
CONFIG_MTD_ABSENT	N	Support for absent chips in bus mapping
CONFIG_MTD_OBSOLETE_CHIPS	Y	Older (theoretically obsoleted now) drivers for non-CFI chips
CONFIG_MTD_AMDSTD	N	AMD-compatible Flash chip support (non-CFI)

Table 3-8: RAM/ROM/Flash Chip Drivers (Continued)

Options	Value	Description
CONFIG_MTD_SHARP	Y	Pre-CFI Sharp chip support
CONFIG_MTD_JEDEC	N	JEDEC device support

Table 3-9: Mapping Drivers for Chip Access

Options	Value	Description
CONFIG_MTD_PQ2FADS	Y	Flash chip mapping on the Motorola PQ2FADS-ZU board
CONFIG_MTD_PQ2FADS_PART	:	Partitions layout

Table 3-10: Self-Contained MTD Device Drivers

Options	Value	Description
CONFIG_MTD_SLRAM	N	Uncached system RAM
CONFIG_MTD_MTDDRAM	N	Test driver using RAM
CONFIG_MTD_BLKMTD	N	MTD emulation using block device
CONFIG_MTD_DOC1000	N	M-Systems Disk-On-Chip 1000
CONFIG_MTD_DOC2000	N	M-Systems Disk-On-Chip 2000 and Millennium
CONFIG_MTD_DOC2001	N	M-Systems Disk-On-Chip Millennium-only alternative driver (see help)

Table 3-11: NAND Flash Device Drivers

Options	Value	Description
CONFIG_MTD_NAND	N	NAND device support

Table 3-12: Plug and Play Configuration

Options	Value	Description
CONFIG_PNP	N	Plug and Play support

Table 3-13: Block Devices

Options	Value	Description
CONFIG_BLK_DEV_FD	N	Normal PC floppy disk support
CONFIG_BLK_DEV_LOOP	N	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 (initrd) support
CONFIG_BLUECAT_RFS	Y	BlueCat Linux RFS support

Table 3-14: Multidevice Support (RAID and LVM)

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

Table 3-15: Networking Options

Options	Value	Description
CONFIG_PACKET	N	Packet socket
CONFIG_NETLINK_DEV	N	Netlink device emulation
CONFIG_NETFILTER	N	Network packet filtering (replaces ipchains)
CONFIG_FILTER	N	Socket filtering

Table 3-15: Networking Options (Continued)

Options	Value	Description
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	Y	IP: kernel level autoconfiguration
CONFIG_IP_PNP_DHCP	N	IP: DHCP support
CONFIG_IP_PNP_BOOTP	Y	IP: BOOTP support
CONFIG_IP_PNP_RARP	Y	IP: RARP support
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	Y	IP: TCP syncookie support (disabled per default)
CONFIG_IPV6	N	The IPv6 protocol (Experimental)
CONFIG_KHTTPD	N	Kernel httpd acceleration (Experimental)
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)

Table 3-15: Networking Options (Continued)

Options	Value	Description
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 3-16: QoS and/or Fair Queueing

Options	Value	Description
CONFIG_NET_SCHED	N	QoS and/or fair queueing

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

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

Table 3-18: SCSI Support

Options	Value	Description
CONFIG_SCSI	N	SCSI support

Table 3-19: Network Device Support

Options	Value	Description
CONFIG_NETDEVICES	Y	Network device support

Table 3-20: ARCnet Devices

Options	Value	Description
CONFIG_ARCNET	N	ARCnet support
CONFIG_DUMMY	N	Dummy net driver support
CONFIG_BONDING	N	Bonding driver support
CONFIG_EQUALIZER	N	EQL (serial line load balancing) support
CONFIG_TUN	N	Universal TUN/TAP device driver support
CONFIG_ETHERTAP	N	Ethertap network tap (Obsolete)

Table 3-21: Ethernet (10 or 100Mbit)

Options	Value	Description
CONFIG_NET_ETHERNET	Y	Ethernet (10 or 100Mbit)
CONFIG_MACE	N	MACE (Power Mac Ethernet) support
CONFIG_BMAC	N	BMAC (G3 Ethernet) support
CONFIG_GMAC	N	GMAC (G4/iBook Ethernet) support
CONFIG_NET_VENDOR_3COM	N	3COM cards
CONFIG_NET_VENDOR_SMC	N	Western Digital/SMC cards
CONFIG_NET_VENDOR_RACAL	N	Racal-Interlan (Micom) NI cards
CONFIG_NET_POCKET	N	Pocket and portable adapters

Table 3-22: Ethernet (1000 Mbit)

Options	Value	Description
CONFIG_FDDI	N	Fiber Distributed Data Interface (FDDI) driver support
CONFIG_HIPPI	N	High Performance Parallel Interface (HIPPI) driver support (Experimental)

Table 3-22: Ethernet (1000 Mbit) (Continued)

Options	Value	Description
CONFIG_PPP	N	Point-to-Point Protocol (PPP) support
CONFIG_SLIP	N	Serial Line Internet Protocol (SLIP) support

Table 3-23: Wireless LAN (Non-Ham Radio)

Options	Value	Description
CONFIG_NET_RADIO	N	Wireless LAN (non-ham radio)

Table 3-24: Token Ring Devices

Options	Value	Description
CONFIG_NET_FC	N	Fibre Channel driver support
CONFIG_SHAPER	N	Traffic Shaper (Experimental)

Table 3-25: WAN Interfaces

Options	Value	Description
CONFIG_WAN	N	WAN interfaces support

Table 3-26: Amateur Radio Support

Options	Value	Description
CONFIG_HAMRADIO	N	Amateur radio support

Table 3-27: IrDA (Infrared) Support

Options	Value	Description
CONFIG_IRDA	N	IrDA subsystem support

Table 3-28: ISDN Subsystem

Options	Value	Description
CONFIG_ISDN	N	ISDN support

Table 3-29: Old CD-ROM Drivers (not SCSI, not IDE)

Options	Value	Description
CONFIG_CD_NO_IDESCSI	N	Support for non-SCSI/IDE/ATAPI CDROM drives

Table 3-30: Console Drivers

Options	Value	Description
CONFIG_VGA_CONSOLE	N	Support for VGA console

Table 3-31: Frame Buffer Support

Options	Value	Description
CONFIG_FB	N	Support for frame buffer devices (Experimental)

Table 3-32: Input Core Support

Options	Value	Description
CONFIG_INPUT	N	Input core support

Table 3-33: Character Devices

Options	Value	Description
CONFIG_VT	N	Virtual terminal
CONFIG_SERIAL	N	Standard/generic (8250/16550 and compatible UARTs) serial support
CONFIG_SERIAL_NONSTANDARD	N	Nonstandard serial port support

Table 3-34: Serial Drivers

Options	Value	Description
CONFIG_SERIAL_8250	N	8250/16550 and compatible serial support (Experimental)
CONFIG_UNIX98_PTYS	N	Unix98 PTY support

Table 3-35: I2C Support

Options	Value	Description
CONFIG_I2C	N	I2C support

Table 3-36: L3 Serial Bus Support

Options	Value	Description
CONFIG_L3	N	L3 support

Table 3-37: Mice

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

Table 3-38: Joysticks

Options	Value	Description
CONFIG_QIC02_TAPE	N	QIC-02 tape support

Table 3-39: Watchdog Cards

Options	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
CONFIG_R3964	N	Siemens R3964 line discipline
CONFIG_APPLICOM	N	Applicom intelligent fieldbus card support

Table 3-40: Ftape, the Floppy Tape Device Driver

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

Table 3-41: Multimedia Devices

Options	Value	Description
CONFIG_VIDEO_DEV	N	Video for Linux

Table 3-42: File Systems

Options	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)
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_JFFS_FS	Y	Journalling Flash File System (JFFS) support
CONFIG_JFFS_FS_VERBOSE	0	JFFS debugging verbosity (0 = quiet, 3 = noisy)
CONFIG_JFFS_PROC_FS	N	JFFS stats available in /proc filesystem
CONFIG_JFFS2_FS	Y	Journalling Flash File System v2 (JFFS2) support
CONFIG_JFFS2_FS_DEBUG	0	JFFS2 debugging verbosity (0 = quiet, 2 = noisy)
CONFIG_JFFS2_FS_NAND	N	JFFS2 support for NAND chips

Table 3-42: File Systems (Continued)

Options	Value	Description
CONFIG_CRAMFS	N	Compressed ROM file system support
CONFIG_TMPFS	Y	Virtual memory file system support (former shm file system)
CONFIG_RAMFS	N	Simple RAM-based file system support
CONFIG_ISO9660_FS	N	ISO 9660 CDROM 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_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 3-43: Network File Systems

Options	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

Table 3-43: Network File Systems (Continued)

Options	Value	Description
CONFIG_NFS_V3	N	Provide NFSv3 client support
CONFIG_ROOT_NFS	Y	Root file system on NFS
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 3-44: Partition Types

Options	Value	Description
CONFIG_PARTITION_ADVANCED	N	Advanced partition selection

Table 3-45: Sound

Options	Value	Description
CONFIG_SOUND	N	Sound card support

Table 3-46: MPC8260 CPM Options

Options	Value	Description
CONFIG_SCC_CONSOLE	Y	Enable SCC console
CONFIG_SCC1_CONSOLE	Y	Console on
CONFIG_SCC_ENET	N	CPM SCC Ethernet
CONFIG_FCC_ENET	Y	CPM FCC Ethernet
CONFIG_FCC1_ENET	N	Ethernet on FCC1
CONFIG_FCC2_ENET	Y	Ethernet on FCC2

Table 3-46: MPC8260 CPM Options (Continued)

Options	Value	Description
CONFIG_FCC3_ENET	Y	Ethernet on FCC3
CONFIG_USE_MDIO	Y	Use MDIO for PHY configuration
CONFIG_FCC_LXT970	N	Support LXT970 PHY
CONFIG_FCC_LXT971	N	Support LXT971 PHY
CONFIG_FCC_QS6612	N	Support QS6612 PHY
CONFIG_FCC_AMD79C873	N	Support AMD79C873 PHY
CONFIG_FCC_DM9131	N	Support DM9131 PHY
CONFIG_FCC_DM9161	Y	Support DM9161 PHY
CONFIG_DCACHE_DISABLE	N	Disable data cache

Table 3-47: USB Support

Options	Value	Description
CONFIG_USB	N	Support for USB

Table 3-48: Bluetooth Support

Options	Value	Description
CONFIG_BLUEZ	N	Bluetooth subsystem support

Table 3-49: Kernel Hacking

Options	Value	Description
CONFIG_MAGIC_SYSRQ	N	Magic SysRq key
CONFIG_XMON	N	Include xmon kernel debugger
CONFIG_BLUECAT_KDBG	N	Include kdbg kernel debugger

Table 3-50: Modular Advanced Power Management

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

This chapter provides information about BlueCat Linux demo systems supported by the pq2fads BSP.

Demo Systems

Table 4-1 lists the demo systems supported in the pq2fads BSP distribution, the boot devices supported by each demo system, and their respective RAM and ROM requirements.

Table 4-1: Demo Systems Supported by pq2fads BSP

Demo	Boot Devices Supported by Default	ROM Requirements	RAM Requirements
developer	Network (using LynuxWorks Boot Loader) Flash	3.5 MB	14560 KB
osloader	Network (using LynuxWorks Boot Loader) Flash	640 KB	3460 KB
showcase	Network (using LynuxWorks Boot Loader) Flash	2.5 MB	11759 KB

developer Demo System

The `developer` demo system is a package consisting of the functionalities of the `shell`, `ftp`, `ping`, `gdb`, and `v1_demo` systems. For descriptions of `developer` and its component demo systems, 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 the target board. 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.

Using Selected RPM Packages

The following sections describe 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 and boot 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 a 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 step listed below. Refer to Chapter 2, “Downloading and Booting BlueCat Linux on the Target” for additional details about the BlueCat Linux OS loader.

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 an IP address of the target and *development_host_IP* is an 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          8068 Oct 24 11:33 /sbin/reboot
/ # chmod 755 /sbin/reboot
/ # ls -la /sbin/reboot
-rwxr-xr-x  1 0      0          8068 Oct 24 11:33 /sbin/reboot
```
- `echo`

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

```
/ # date
Fri Oct 24 16:13:46 UTC 2003
```
- `uname`

```
/ # uname -a
Linux (none) 2.4.18-1 #2 Fri Oct 24 15:33:40 MSD 2003 ppc unknown
```
- `mount`

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

```

• ifconfig

/ # ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 08:00:3E:66:15:59
          BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:18 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:2332 (2.2 kb)  TX bytes:0 (0.0 b)
          Base address:0x8500

/ # ifconfig eth0 172.17.2.15
/ # ping -c 5 172.17.0.1
PING 172.17.0.1 (172.17.0.1): 56 data bytes
64 bytes from 172.17.0.1: icmp_seq=0 ttl=255 time=0.4 ms
64 bytes from 172.17.0.1: icmp_seq=1 ttl=255 time=0.1 ms
64 bytes from 172.17.0.1: icmp_seq=2 ttl=255 time=0.1 ms
64 bytes from 172.17.0.1: icmp_seq=3 ttl=255 time=0.1 ms
64 bytes from 172.17.0.1: icmp_seq=4 ttl=255 time=0.1 ms

--- 172.17.0.1 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 0.1/0.1/0.4 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 a 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/tty1

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 step listed below. Refer to Chapter 2, “Downloading and Booting BlueCat Linux on the Target” for additional details about the BlueCat Linux OS loader.

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
> boot
```

where *target_board_IP* is an IP address of the target and *development_host_IP* is an 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
```

- 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.

- Create a BlueCat Linux Kernel Downloadable Image (`zebra.kernel`):

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

- 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/ttyl
# 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/vtysh /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

10:0:wait:/sbin/halt
16:6: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 192.168.4.12/16

!
```

```
! Static default route.  
!  
ip route 80.240.0.0 255.255.0.0 192.168.4.121  
  
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 step listed below. Refer to Chapter 2, “Downloading and Booting BlueCat Linux on the Target” for additional details about the BlueCat Linux OS loader.

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
> boot
```

where `target_board_IP` is an IP address of the target and `development_host_IP` is an 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 08:00:3E:66:15:59
      inet addr:172.17.2.15  Bcast:172.17.255.255  Mask:255.255.0.0
      UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
      RX packets:3  errors:0  dropped:0  overruns:0  frame:0
      TX packets:0  errors:0  dropped:0  overruns:0  carrier:0
      collisions:0  txqueuelen:100
      Base address:0x8500

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 2 172.17.0.1
PING 172.17.0.1 (172.17.0.1) from 172.17.2.15 : 56(84) bytes of data.
64 bytes from 172.17.0.1: icmp_seq=0 ttl=255 time=346 usec
64 bytes from 172.17.0.1: icmp_seq=1 ttl=255 time=142 usec

--- 172.17.0.1 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/mdev = 0.142/0.244/0.346/0.102 ms
bash-2.04# ping -c 3 172.17.0.3
PING 172.17.0.3 (172.17.0.3) from 172.17.2.15 : 56(84) bytes of data.
64 bytes from 172.17.0.3: icmp_seq=0 ttl=255 time=333 usec
64 bytes from 172.17.0.3: icmp_seq=1 ttl=255 time=135 usec
64 bytes from 172.17.0.3: icmp_seq=2 ttl=255 time=138 usec

--- 172.17.0.3 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max/mdev = 0.135/0.202/0.333/0.092 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>* 172.17.0.0/16 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 pq2fads BSP and provides important information about them.

Table 5-1: Device Drivers Supported by the pq2fads BSP

Hardware Device	Device Drivers	Location in Source Tree	Kernel Configuration Options	Notes
Flash 8MB using SIMM module	sharp.c pq2fads.c	drivers/mtd/chps drivers/mtd/maps	CONFIG_MTD_SHARP CONFIG_MTD_PQ2FADS	Supported via JFFS/JFFS2 and Flash management
Ethernet FCC2 and FCC3 on MPC8260 using external Davicom DM9161 transceiver	fcc_enet.c	arch/ppc/8260_io	CONFIG_FEC_ENET	
Serial Two async serial ports with RS-232 interface at SCC1 & SCC2	uart.c	arch/ppc/8260_io	CONFIG_SCC_CONSOLE	

This chapter describes the new features of this release.

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

JFFS2 Support

This release provides support for Journalling Flash File System version 2 (JFFS2).

The following BlueCat Linux components were updated to support this feature:

- The BlueCat Linux kernel
The BlueCat Linux kernel contains the latest version of JFFS2 available from public domain.

- The BlueCat Linux `mkrootfs` utility

A new `-j` option is used for `mkrootfs` to create a JFFS2 partition. Refer to the `mkrootfs` man page for more information.

- The BlueCat Linux `osloader` demo system

`osloader` is extended with support for the JFFS2 and now can be used to download a desired BlueCat Linux custom or demo system into the target board's Flash memory using either JFFS or JFFS2. A JFFS2 partition is created in the same way as a JFFS partition. Refer to *BlueCat Linux User's Guide* for details.

Support of Multithreaded Applications in GDB

This release 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*.

This chapter describes known problems and limitations of this release.

PQ2FADS Target Board Problems and Limitations

The following are known problems and limitations of this release:

- The 8MB SDRAM on the local bus is not used by the BlueCat Linux kernel because the instruction cache does not work with this region of memory. This is described by the following hardware errata in *MPC8260 PowerQUICC II™ Design Checklist, Application Note AN2290/D*:

“The local bus does not burst when accessed from the 603e core or from an external master through the 60x bus bridge. Accesses to the local bus are not snooped by the 603e core. For this reason, regions that are accessed across the 60x bridge to local bus must be marked as cache inhibited in the 603e core MMU.”

- The gdb Insight graphical environment has constant problems on the Windows host. An attempt to view local variables or insert watchpoints may cause hanging of Insight. Use the command line gdb debugger should you encounter such problems.
- Use the following command in order to use Ethernet in the `i_osloader` demo system:

```
make -f Makefile.i xconfig
```

Enable your network card, then type the following command:

```
make -f Makefile.i all
```

The demo now will have a correct Ethernet configuration and can be used to boot other BlueCat Linux demos over the network.

- 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.