BlueCat Linux Target Support Guide

DOC-0436-00

for Cirrus Logic CS89712 Boards

Product names mentioned in *BlueCat Linux Target Support Guide for Cirrus Logic CS89712 Boards* are trademarks of their respective manufacturers and are used here for identification purposes only.

Printed in the United States of America.

Contents

CHAPTER 1 *Overview*

This *BlueCat Linux Target Support Guide for Cirrus Logic CS89712 Boards* provides information about the BlueCat Linux Target Support Package (TSP) supporting Cirrus Logic CS89712 systems on a chip with ARM CPU cores.

Throughout this Target Support Guide (TSG), the TSP is referred to as the "clee89712;" the target board as the "CS89712."

The chapters of this TSG provide the information listed below:

- **Chapter 1** is an overview of this TSG's individual chapters.
- **Chapter 2** describes BlueCat Linux downloading and booting procedures for the CS89712 target board, using the Bluecat Linux `showcase` demo system as an example.
- **Chapter 3** provides configuration option information about the clee89712 TSP's default BlueCat Linux kernel.
- **Chapter 4** summarizes BlueCat Linux demo systems supported by the clee89712 TSP.
- **Chapter 5** provides a list of clee89712 TSP-supported device drivers, with important information about each of them.
- **Chapter 6** describes Thumb, an extension of ARM architecture that enables users to achieve 32-bit performance and address space without the associated costs of going to a 32-bit system.
- **Appendix A** describes `clep_boot`, a utility used to help complete target board-specific bootstrapping operations.

CHAPTER 2

Downloading and Booting BlueCat Linux on the Target Board

This chapter provides instructions for downloading BlueCat Linux demo systems from a cross development host into a target board's flash memory via a network, and then booting that BlueCat Linux demo system on the target board.

Specifically, these instructions detail the steps needed to perform an initial download of a special version of the BlueCat Linux kernel downloadable image (KDI) called `i_osloader` onto a target board, and the subsequent downloading of the `showcase` demo system (the `showcase` demo system needs certain resources of `i_osloader` to download into the target board's flash memory).

While this chapter uses `showcase` to exemplify downloading a demo system, any other supported demo system or custom application can also be downloaded onto a target board using the same procedure.

Prerequisites

This document is intended to be used as a guide to download and boot BlueCat Linux systems on CS89712 target boards. Scenarios using demo systems included in the BlueCat Linux distribution are presented. As such, a basic familiarity with the target board hardware and operation is required before using this guide. Users must also have an understanding of system administration for the particular cross development host on which BlueCat Linux and the Target Support Package is installed. It is assumed users have 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 ARM configuration and the clee89712 Target Support Package (TSP) have been installed on the cross development host system, for example:

1. Install the BlueCat Linux ARM configuration onto the cross development host, as described in the "Installing the Default BlueCat Linux Configuration" section of "Chapter 1 - Installation" in the *BlueCat Linux User's Guide.*
2. Install the clee89712 TSP onto the cross development host as detailed in the "Installing Support for Boards" section of "Chapter 1 - Installation" in the *BlueCat Linux User's Guide.*

Downloading and Booting Overview

NOTE: *This section and those that follow provide important information about downloading BlueCat Linux into the target's flash memory, and then booting BlueCat Linux system on the same target board.*

Read all of this section and those that follow before attempting to boot a target board.

The standard procedure for downloading and booting a BlueCat Linux system onto an CS89712 target board consists of three main steps:

- Hardware Setup
- Downloading `preboot-clee89712.axf` and `osloader-clee89712.bnd`
- Using `i_osloader` to download a demo system

These three steps are performed sequentially:

1. Physically connect the hardware; see "Hardware Configurations" on page 5.
2. Download the utilities, `preboot-clee89712.axf` and `osloader-clee89712.bnd`, into a target's flash memory from the BlueCat Linux cross development host with the `clep_boot` utility via a serial port connection; see "Definitions" and "clep_boot" for descriptions.

3. Use the copy of `osloader-clee89712.bnd` that was just loaded into the target's flash memory to download `i_osloader` from a BlueCat Linux cross development host into the target's flash memory via a network connection; for a description of `i_osloader`, see "i_osloader" on page 11.

   Use the copy of `i_osloader` that was just loaded into the target's flash memory to download and boot a BlueCat Linux demo system, such as `showcase`, on the cross development host via a network connection.

Hardware Setup

Hardware Configurations

The following are two examples of supported hardware configurations, each with its own distinct cross development host and virtual terminal that can be used for downloading and booting BlueCat Linux on the target board:

- Linux-Based Cross Development Host with a Minicom Virtual Terminal
- Windows-Based Cross Development Host with a Hyperterminal Virtual Terminal

NOTE: *The baud rate and parity settings for both Hyperterminal and Minicom should be set as 9600 bps, 8 bit - No parity - 1 stop bit respectively and no hardware flow control.*

Before downloading `preboot-clee89712.axf` and `osloader-clee89712.bnd` onto the target board, choose and set up one of the hardware configurations shown below:

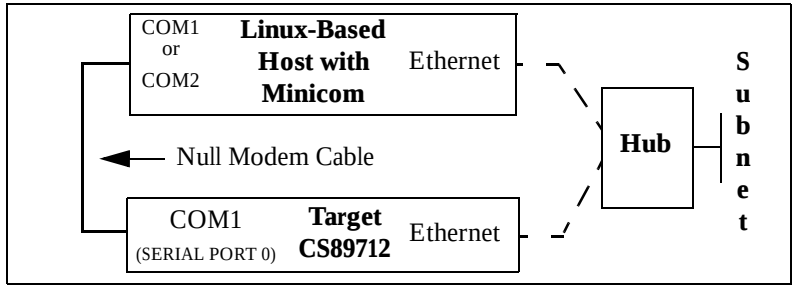

Figure 2-1: Linux-Based Cross Development Host with a Minicom Virtual Terminal

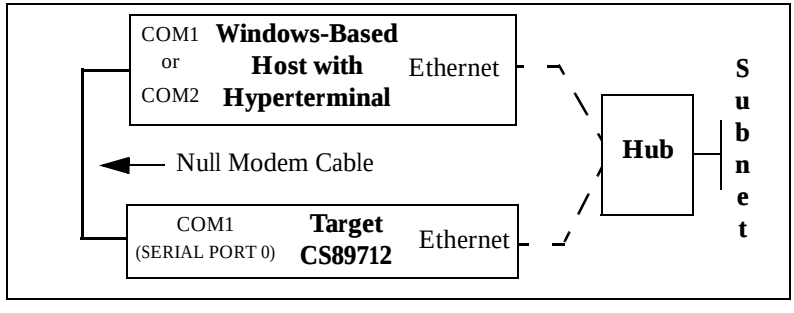

Figure 2-2: Windows-Based Cross Development Host with a Hyperterminal Virtual Terminal

Target Board Jumpers

The following target board jumpers need to be removed before starting the download process. Please consult the manufacturer's documentation to determine the location of these jumpers on the target board.

Table 2-1: Target Board Jumpers

Jumper	Initial Configuration	Description
JP31	Open	Causes the target board to be booted from the manufacturer installed firmware.

Downloading preboot-clee89712.axf and osloader-clee89712.bnd

Definitions

Utilities – preboot-clee89712.axf and osloader-clee89712.bnd

The utilities, `preboot-clee89712.axf` and `osloader-clee89712.bnd`, are located in the `$BLUECAT_PREFIX/cdt/boot` directory and are downloaded from the cross development host into the target's flash memory via a serial port connection.

The following information details `preboot-clee89712.axf` and `osloader-clee89712.bnd`:

`preboot-clee89712.axf`	Resets the CS89712's COM1 serial port data transfer rate from 9600 bps to 115200 bps.
`osloader-clee89712.bnd`	Contains the *minimum* image needed to initialize an Ethernet interface and provide a BlueCat OS Loader Shell (BLOSH) on the target board.

`preboot-clee89712.axf` and `osloader-clee89712.bnd` are downloaded from the cross development host (in that order) onto the target board using a special BlueCat Linux utility, `clep_boot`.

clep_boot

The `clep_boot` utility is used to download the `preboot-clee89712.axf` and `osloader-clee89712.bnd` utilities into the target's flash memory. `clep_boot` is run from a BlueCat Linux cross development host connected via a null modem cable from the host's COM1 or COM2 serial port to the target board's COM1 serial port (`SERIAL PORT 0`).

When `clep_boot` is invoked on the cross development host, several actions occur. `clep_boot` downloads and transfers control to `preboot-clee89712.axf`. Then `preboot-clee89712.axf` resets the CS89712 serial port's data transfer rate from 9600 bps to 115200 bps; this facilitates a faster transfer of subsequent images from the cross development host to the target board.

`clep_boot` then downloads `osloader-clee89712.bnd` from the cross development host to the target board via the same serial connection that `preboot-clee89712.axf` was transferred over, but now, at the faster 115200 bps rate. Then `osloader-clee89712.bnd` is burned into the target's flash memory, where it provides an Ethernet interface and a BLOSH prompt to download BlueCat Linux demo systems or customized BlueCat Linux systems.

Downloading preboot-clee89712.axf and osloader-clee89712.bnd

To download `preboot-clee89712.axf` and `osloaderclee89712.bnd` into the target's flash memory over a serial port connection, follow these steps:

1. Select and set up hardware in one of the configurations detailed in "Hardware Configurations" on page 5.

NOTE: *BlueCat Linux cross development hosts require write access to configure serial ports. If the cross development host system does not allow write access for regular users, run BlueCat Linux on the host as root.*

---

2. Change to the BlueCat Linux installation directory.
3. Set up the BlueCat Linux environment by entering the following command:

```
BlueCat:# . SETUP.sh clee89712
```

4. Use the `clep_boot` utility to download `preboot-clee89712.axf` and `osloader-clee89712.bnd` into the target's flash memory by entering one of the following commands as appropriate:

   For COM1:

   Linux-based hosts:

   ```
   BlueCat:# clep_boot –c /dev/ttyS0
   ```

   Windows-based hosts:

   ```
   BlueCat:# clep_boot -c COM1
   ```

   For COM2:

   Linux-based hosts:

   ```
   BlueCat:# clep_boot –c /dev/ttyS1
   ```

   Windows-based hosts:

   ```
   BlueCat:# clep_boot -c COM2
   ```

Carefully follow the `clep_boot` on-screen instructions.

`preboot-clee89712.axf` and `osloader-clee89712.bnd` should take less than two minutes to download into the target's flash memory.

CAUTION! *Neither Hyperterminal nor Minicom virtual terminal programs can be run on the host serial port connected to the target board's COM1* before *and* during `preboot-clee89712.axf` *and* `osloader-clee89712.bnd` *download. Both the Hyperterminal and Minicom programs configure the cross development host serial port to a particular baud rate, which may vary from the 115200 bps used for downloading* `osloader-clee89712.bnd`*. This difference creates a conflict, and the installation software becomes dysfunctional on the cross development host as it attempts to download* `osloader-clee89712.bnd` *onto the target board.*

Make sure no other process is running on the host serial port connected to target board's COM1, before and during the download of both `preboot-clee89712.axf` *and* `osloader-clee89712.bnd`

NOTE: *When the* `clep_boot` *utility prompts the user to short the boot jumper on the target board, short the two JP31 pins with the jumper.*

---

5. Activate a Minicom or a Hyperterminal session on the BlueCat Linux cross development host to observe screen output displays from the target board's serial port 0.
6. Reboot the target board as follows:

    a) Press the S9 power-on reset button.

    b) Press the S3 reset button.

    c) Press the S4 wakeup button.

    The screen output below appears on the Minicom or Hyperterminal, validating that the board has rebooted and that a BLOSH prompt is available:

```
(egcs-1.1.2 release)) #1 Wed Dec 27 11:50:12 MSK 2000
Calibrating delay loop... 65.33 BogoMIPS
Memory: 8964k/16M available (704k code, 224k reserved,
344k data, 4k init)
DENTRY hash table entries: 16384 (order: 5, 131072
bytes)
```

```
Buffer-cache hash table entries: 32768 (order: 5, 131072
bytes)
Page-cache hash table entries: 4096 (order: 2, 16384
bytes)
POSIX conformance testing by UNIFIX
Linux NET4.0 for Linux 2.2
Based upon Swansea University Computer Society NET3.039
NET4: Linux TCP/IP 1.0 for NET4.0
IP Protocols: ICMP, UDP, TCP
TCP: Hash tables configured (ehash 16384 bhash 16384)
Starting kswapd v 1.5
RAM disk driver initialized:  16 RAM disks of 4096K size
eth0: cs8900 rev J found at 0x300 media RJ-45, IRQ 7,
mac =  00 24 20 10 12 ba
RAMDISK: Compressed image found at block 3164160
VFS: Mounted root (ext2 filesystem).
Freeing unused kernel memory: 4k init
BlueCat Loader Shell
>
```

Upon successful completion of the `clep_boot` download process, `osloader-clee89712.bnd` automatically boots-up from the target's flash memory every time the board is rebooted by first pressing the S9 power-on reset button, then pressing the S3 reset button, and finally pressing the S4 wakeup button.

Using i_osloader to Download a Demo System

This section and its subsections provide step-by-step detailed instructions for completing the process by which any supported demo system (e.g., `showcase`) or custom BlueCat Linux system can be downloaded into the target flash memory via a network connection. This is done using the `i_osloader` utility.

Definitions

i_osloader

`i_osloader` is a special version of the OS Loader kernel downloadable image. It is used to download the desired BlueCat Linux custom or demo system (in this case, the `showcase` demo system).

`i_osloader` is located in the `$BLUECAT_PREFIX/demo.clee89712 /osloader` directory as a part of the standard BlueCat Linux distribution for CS89712 target boards.

Downloading i_osloader

At this point, the BLOSH prompt (>) is displayed on either a Minicom terminal on Linux-based cross development hosts, or a Hyperterminal on Windows-based cross development hosts. It is through commands entered at this prompt that `i_osloader` will be downloaded to the target board.

To download `i_osloader` on the target board, follow these steps:

NOTE: *These steps provide general information for many versions of Linux. Users need to consult their version of Linux's user documentation to determine any differences.*

---

1. Configure the host as a TFTP server:

   1.1 For example, to configure the TFTP daemon on a Linux server, uncomment the following line in the `/etc/inetd.conf` configuration file:

   ```
   tftp dgram udp wait root /usr/sbin/tcpd \
   in.tftpd /tftpboot
   ```

NOTE: *The backslash character,* `\`*, at the end of the first line in the previous example denotes the continuation of the line of code that cannot fit on a single line of text in this document. This character is* not *typed when entering the command; however, any space proceeding it is. This convention is used throughout this TSG, as well, with commands or lines of code that are too long fit on one line of text.*

   1.2 Send the SIGHUP signal to force the `inetd` process to re-read the configuration file, where *inetd_pid* is the process identification number of the `inetd` process:

   `BlueCat:#` **`kill -HUP`** *inetd_pid*

2. Make a `tftpboot` directory under the root directory:

   `BlueCat:#` **`mkdir /tftpboot`**

3. Copy the files listed below from the `$BLUECAT_PREFIX/demo.clee89712/osloader` directory into the `/tftpboot` directory so that they can be downloaded from there onto the target board:
   - `i_osloader.kernel`
   - `i_osloader.rfs`
4. Define the IP address for both the cross development host and target board by entering the following commands:

   > **`set IP`** *target_IP_address*
   > **`set HOST`** *host_IP_address*

5. Download `i_osloader.kernel` and `i_osloader.rfs` into the target board's RAM via the network connection by entering the following commands:

   ```
   > set IF eth0

   > set KERNEL tftp \
     /tftpboot/i_osloader.kernel

   > set RFS tftp /tftpboot/i_osloader.rfs

   > boot
   ```

The screen output below validates that both `i_osloader` files have been successfully downloaded into the target's flash memory:

```
eth0: using 10Base-T (RJ-45)
getting  tftpboot/i_osloader.kernel
................................................
................................
Received 465649
getting tftpboot/i_osloader.rfs
................................................
.....................................
Switching to new kernel...
Command line:
Uncompressing
Linux...........................................
................................................
................................................
................................................
................................................
................................................
..done.
```

This completes the download of i_osloader.

The downloading process now automatically boots `i_osloader`, displaying the screen output below:

```
S11::done.
Linux version 2.2.12-1 (build@bcm-build1) (gcc version
egcs-2.91.66 19990314/Lin
ux (egcs-1.1.2 release)) #1 Wed Feb 14 15:56:08 MSK
2001
NetWinder Floating Point Emulator V0.94.1 (c) 1998
Corel Computer Corp.
Calibrating delay loop... 65.33 BogoMIPS
Memory: 8940k/16M available (728k code, 224k reserved,
344k data, 4k init)
DENTRY hash table entries: 524288 (order: 10, 4194304
bytes)
Buffer-cache hash table entries: 32768 (order: 5,
131072 bytes)
Page-cache hash table entries: 4096 (order: 2, 16384
bytes)
POSIX conformance testing by UNIFIX
Linux NET4.0 for Linux 2.2
Based upon Swansea University Computer Society
NET3.039
NET4: Linux TCP/IP 1.0 for NET4.0
IP Protocols: ICMP, UDP, TCP
TCP: Hash tables configured (ehash 16384 bhash 16384)
Starting kswapd v 1.5
RAM disk driver initialized:  16 RAM disks of 4096K
size
eth0: cs8900 rev J found at 0x300 media RJ-45, IRQ 7,
mac =  00 24 20 10 12 ba
RAMDISK: Compressed image found at block 3155968
VFS: Mounted root (ext2 filesystem).
Freeing unused kernel memory: 4k init
BlueCat Loader Shell
> s IP 216.100.252.140
> s HOST 216.100.252.109
> s IF eth0
> s KERNEL tftp /tftpboot/i_osloader.kernel
> s RFS tftp /tftpboot/i_osloader.rfs
> b
eth0: using 10Base-T (RJ-45)
getting /tftpboot/i_osloader.kernel
................................................
.........................
...........
Received 465614
getting /tftpboot/i_osloader.rfs
................................................
................................................
..............
Switching to new kernel...
Command line:
Uncompressing Linux...done.
Now booting the kernel
Linux version 2.2.12-1 (bin@bcm-build2) (gcc version
egcs-2.91.66 19990314/Linux (egcs-1.1.2 release)) #3
Sat Feb 17 16:39:55 MSK 2001
NetWinder Floating Point Emulator V0.94.1 (c) 1998
```

```
Corel Computer Corp.
Calibrating delay loop... 65.33 BogoMIPS
Memory: 14196k/16M available (856k code, 224k
reserved, 352k data, 4k init)
DENTRY hash table entries: 1048576 (order: 11, 8388608
bytes)
Buffer-cache hash table entries: 32768 (order: 5,
131072 bytes)
Page-cache hash table entries: 4096 (order: 2, 16384
bytes)
POSIX conformance testing by UNIFIX
Linux NET4.0 for Linux 2.2
Based upon Swansea University Computer Society
NET3.039
NET4: Linux TCP/IP 1.0 for NET4.0
IP Protocols: ICMP, UDP, TCP
TCP: Hash tables configured (ehash 16384 bhash 16384)
Starting kswapd v 1.5
RAM disk driver initialized:  16 RAM disks of 4096K
size
eth0: cs8900 rev J found at 0x300 media RJ-45, IRQ 7,
mac =  00 24 20 10 12 ba
RAMDISK: Compressed image found at block 3149384
Flash mapping for EE89712 board initialized starting
at address 0xd0300000
number of CFI chips: 2
EE89712 Flash MTD driver: Configuration of partitions
is :
EE89712 Flash MTD driver: Configured 0 partitions
VFS: Mounted root (ext2 filesystem).
Freeing unused kernel memory: 4k init
BlueCat Loader Shell
> s IP 216.100.252.140
> s HOST 216.100.252.109
> s IF eth0
> s FILE tftp /tftpboot/showcase.kdi
> exec flash_fdisk /dev/mtdchar0 0-28
EE89712 Flash MTD driver: Configuration of partitions
is 0-28
EE89712 Flash MTD driver: Configured 1 partitions
> flash /dev/mtdchar1 erase
Device has 2 regions with sectors of the same size.
Total size of the device is 0x2c0000 bytes.
Erasing..... done.
eth0: using 10Base-T (RJ-45)
getting /tftpboot/showcase.kdi
.................................................
.................................................
.................................................
.................................................
Received 1942528
> Command line: ramdisk_size=32000 root=101
Uncompressing Linux...done.
Now booting the kernel
Linux version 2.2.12-1 (bin@bcm-build2) (gcc version
egcs-2.91.66 19990314/Linux
(egcs-1.1.2 release)) #4 Sat Feb 17 16:43:00 MSK 2001
NetWinder Floating Point Emulator V0.94.1 (c) 1998
Corel Computer Corp.
```

```
Calibrating delay loop... 65.33 BogoMIPS
Memory: 13776k/16M available (756k code, 224k
reserved, 328k data, 4k init)
DENTRY hash table entries: 262144 (order: 9, 2097152
bytes)
Buffer-cache hash table entries: 32768 (order: 5,
131072 bytes)
Page-cache hash table entries: 4096 (order: 2, 16384
bytes)
POSIX conformance testing by UNIFIX
Linux NET4.0 for Linux 2.2
Based upon Swansea University Computer Society
NET3.039
NET4: Linux TCP/IP 1.0 for NET4.0
IP Protocols: ICMP, UDP, TCP
TCP: Hash tables configured (ehash 16384 bhash 16384)
Starting kswapd v 1.5
RAM disk driver initialized:  16 RAM disks of 32000K
size
eth0: cs8900 rev J found at 0x300 media RJ-45, IRQ 7,
mac =  00 24 20 10 12 ba
RAMDISK: Compressed image found at block 3149404
Flash mapping for EE89712 board initialized starting
at address 0xd0300000
number of CFI chips: 2
EE89712 Flash MTD driver: Configuration of partitions
is :
EE89712 Flash MTD driver: Configured 0 partitions
VFS: Mounted root (ext2 filesystem).
Freeing unused kernel memory: 4k init
INIT: version 2.77 booting
INIT: Entering runlevel: 1
Network is configured as follows:

Target  IP address: 192.168.1.4
Gateway IP address:

eth0: using 10Base-T (RJ-45)
```

After `i_osloader` has been downloaded, booted, and the BLOSH (BlueCat Loader Shell) prompt (>) appears, the BlueCat Linux demo system or custom system can be downloaded onto the target board.

Downloading a Demo System Into Target Flash Memory

To download a BlueCat Linux demo system (in this example, `showcase`) into the target's flash memory, follow the instructions that follow.

NOTE: *These instructions can also be used to load any custom system, by replacing the filename,* `showcase.kdi`*, with the name of the custom system.*

---

1. Define the IP address for both the cross development host and target board by entering the following commands:

   > **`set IP 192.168.1.4`**

   > **`set HOST`** *host_IP_address*

   > **`set IF eth0`**

   The target board's IP address above (`192.168.1.4`) is the default address hardcoded into the `showcase.kdi` file. If the user wishes to change the address, it is necessary to rebuild `showcase.kdi`; see "Modifying Target Board IP and Gateway Addresses" on page 40 for details.

2. Copy the `showcase.kdi` file from the `$BLUECAT_PREFIX/demo.clee89712/showcase` directory into the `/tftpboot` directory so that they can be downloaded from there onto the target board.

3. Enter the following commands to configure the cross development host for downloading `showcase.kdi` into the target's flash memory, where *xx* represents the size of the partition needed to accommodate the demo system being installed:

   > **`set FILE tftp /tftpboot/showcase.kdi`**

   > **`exec flash_fdisk /dev/mtdchar0 0-28`**

NOTE: *It is important that the BlueCat Linux kernel starts at the beginning of the target's flash memory space (offset* `0x0`*). Make sure that the target's flash memory is partitioned accordingly, using the* `flash_fdisk` *tool. Keep in mind that CS89712 on-board flash memory has 78 non-uniform sized sectors. Sectors 0-7 and 39-46 are each 16 KB, while sectors 8-38 and 47-77 are 128 KB. As such, the following command would create a single partition in which a BlueCat Linux image of up to 6 MB could be downloaded:*

```
> exec flash_fdisk /dev/mtdchar0 0-61
```

Each demo system varies in size; see "Chapter 4, Supported Demo Systems" on 35 for further information. Ensure that the sector range values in the line above are set appropriately.

The following screen output validates that the first partition in the range of `0-28` in the target's flash memory has been reserved for `showcase.kdi`:

```
EE89712 Flash MTD driver: Configuration of partitions
is 0-28
EE89712 Flash MTD driver: Configured 1 partitions
```

4. Enter the following command to erase any data in the target's flash memory's first partition, then burn the `showcase.kdi` image into that partition:

   ```
   > flash /dev/mtdchar1 erase
   ```

   The screen output below appears, validating that the target flash memory's first partition has been erased and that `showcase.kdi` has been downloaded to that partition, by listing its file size in number of bytes (in this example, `1942528`):

   ```
   Device has 2 regions with sectors of the same size.
   Total size of the device is 0x2c0000 bytes.
   Erasing..... done.
   eth0: using 10Base-T (RJ-45)
   getting /tftpboot/showcase.kdi
   ................................................
   ................................................
   ........................
   Received 1942528
   >
   ```

This completes downloading the `showcase` demo system to the target's flash memory.

This process can also be used to download a custom BlueCat Linux system or application as well as other demo systems supplied with the distribution to a target's flash memory.

Booting Options

The section details two ways of booting the target board after it has been configured with a BlueCat Linux system.

Booting a Demo System from Flash Memory

At this point, it is assumed that the `showcase` demo system has successfully been downloaded into the target's flash memory. It is then booted from flash memory as follows:

Reset the target board by entering the following command:

```
> reset
```

The screen output below displays while the target board is being reset, with the `bash#` prompt validating that the demo system has booted from the target's flash memory:

```
Restarting system.
Command line: ramdisk_size=32000 root=101
Uncompressing
Now booting the kernel
Linux version 2.2.12-1 (bin@build1) (gcc version egcs-2.91.66
19990314/Linux
(egcs-1.1.2 release)) #3 Thu Jan 4 11:46:27 MSK 2001
NetWinder Floating Point Emulator V0.94.1 (c) 1998 Corel
Computer Corp.
Console: colour dummy device 80x25
Calibrating delay loop... 65.33 BogoMIPS
Memory: 12832k/16M available (860k code, 224k reserved, 412k
data, 16k init)
DENTRY hash table entries: 262144 (order: 9, 2097152 bytes)
Buffer-cache hash table entries: 32768 (order: 5, 131072
bytes)
Page-cache hash table entries: 4096 (order: 2, 16384 bytes)
POSIX conformance testing by UNIFIX
Linux NET4.0 for Linux 2.2
Based upon Swansea University Computer Society NET3.039
NET4: Linux TCP/IP 1.0 for NET4.0
IP Protocols: ICMP, UDP, TCP
TCP: Hash tables configured (ehash 16384 bhash 16384)
Starting kswapd v 1.5
Console: switching to colour frame buffer device 80x30
fb0: frame buffer device, using 75K of video memory
Initialising keyboard
```

```
audio_7110_init: mcp_major=0
RAM disk driver initialized:  16 RAM disks of 32000K size
eth0: cs8900 rev I found at 0x300
cs89x0: No EEPROM, using default setup temporarily.
 media RJ-45, IRQ 7, mac =  00 e0 98 06 91 66
RAMDISK: Compressed image found at block 3162780
Flash mapping for EE89712 board initialized starting at
address 0xd0300000
number of CFI chips: 2
EE89712 Flash MTD driver: Configuration of partitions is :
EE89712 Flash MTD driver: Configured 0 partitions
VFS: Mounted root (ext2 filesystem).
Freeing unused kernel memory: 16k init
eth0: using 10Base-T (RJ-45)

bash#
```

NOTE: *For the* `showcase` *demo system, if the IP address for the target board has been changed from the default address of* `192.168.1.4`*, the user needs to ensure that the new address is consistent throughout all files associated with the* `showcase` *demo system. This can be done by using the* `ifconfig eth0` *command.*

After it has been booted as described above, `showcase` will now automatically boot from the on-board flash memory every time the board is reset by first pressing the S9 power-on reset button, then pressing the S3 reset button, and finally pressing the S4 wakeup button.

NOTE: *After the* `showcase` *demo system has been downloaded into target's flash memory, it starts on a board power-up/reset. To boot other embedded systems, the user needs to download* `i_osloader` *into target's flash memory again.*

Booting a Demo System over a Network

To boot a demo system or a custom embedded system from a cross development host over a network connection using `i_osloader`, perform the following procedure:

1. Download `i_osloader` into the target's flash memory, as described in the steps detailed in "Downloading preboot-clee89712.axf and osloader-clee89712.bnd" on page 9 through "Downloading i_osloader" on page 12.

2. Boot the BlueCat Linux demo system of choice on the cross development host via a network connection using `i_osloader` *as* detailed in the “Booting BlueCat Linux from a Network or a Parallel Port” section in “Chapter 3 - Booting BlueCat Linux” of the *BlueCat Linux User's Guide.*

CHAPTER 3 *Kernel Configuration Options*

The clee89712 TSP 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: clee89712 TSP Default Kernel Configuration Option Tables" below:

Table 3-1: clee89712 TSP Default Kernel Configuration Option Tables

Parameters	Table Number
System and Processor Type	Table 3-2, page 24
Code Maturity Level Options	Table 3-3, page 24
Loadable Module Support	Table 3-4, page 25
General Setup	Table 3-5, page 25
Plug And Play Support	Table 3-6, page 26
Block Devices	Table 3-7, page 26
Character Devices	Table 3-8, page 26
Video for Linux	Table 3-9, page 27
Joystick Support	Table 3-10, page 27
Ftape, the Floppy Tape Device Driver	Table 3-11, page 28
Networking Options	Table 3-12, page 28
QoS and/or fair queueing	Table 3-13, page 29
Amateur Radio Support	Table 3-14, page 29
IrDA Subsystem support	Table 3-15, page 29
Network Device Support	Table 3-16, page 29
ARCnet Devices	Table 3-17, page 30

Table 3-1: clee89712 TSP Default Kernel Configuration Option

Parameters	Table Number
Ethernet (10 or 100 Mbit)	Table 3-18, page 30
Token Ring Devices	Table 3-19, page 31
WAN Interfaces	Table 3-20, page 31
SCSI Support	Table 3-21, page 32
Filesystems	Table 3-22, page 32
Network File Systems	Table 3-23, page 33
Partition Types	Table 3-24, page 33
Kernel Hacking	Table 3-25, page 33
LynuxWorks Messenger Support	Table 3-26, page 34
Modular Advanced Power Management	Table 3-27, page 34

Table 3-2: System and Processor Type

Option	Value	Description
CONFIG_ARCH_CLEE89712	Y	ARM system type
CONFIG_CPU_ARM7	Y	Optimize for ARM7 CPU
CONFIG_CPU_32	Y	Physical memory size

Table 3-3: Code Maturity Level Options

Option	Value	Description
CONFIG_EXPERIMENTAL	Y	Prompt for development and/or incomplete code/drivers
CONFIG_ALIGNMENT_TRAP	N	Enable kernel-mode alignment trap handle (experimental)
CONFIG_TEXT_SECTIONS	N	Split text into discardable sections

Table 3-4: Loadable Module Support

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

Table 3-5: General Setup

Option	Value	Description
CONFIG_NET	Y	Networking support
CONFIG_CLEE89712_PREBOOT	N	Generate code suitable for `preboot` tools
CONFIG_BLUECAT_THUMB	N	BlueCat kernel support for THUMB binaries
CONFIG_BLUECAT_LOADER	N	BlueCat OS Loader
CONFIG_BLUECAT_IGNORE_PRINTK	N	BlueCat ignore `printk`
CONFIG_BLUECAT_SMALL_FOOTPRINT	N	BlueCat small memory footprint
CONFIG_BLUECAT_MEMSIZE	N	Memory sizing benchmark
CONFIG_SYSVIPC	Y	System V IPC
CONFIG_BSD_PROCESS_ACCT	N	BSD Process Accounting
CONFIG_SYSCTL	N	Systl support
CONFIG_NWFPE	Y	Math emulation
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_ARTHUR	N	RISC OS personality
CONFIG_PARPORT	N	Parallel port support

Table 3-6: Plug and Play Support

Option	Value	Description
CONFIG_PNP	N	Plug and Play support

Table 3-7: Block Devices

Option	Value	Description
CONFIG_BLK_DEV_FD	N	Normal PC floppy disk support
CONFIG_BLK_DEV_IDE	N	Enhanced IDE/MFM/RLL disk/cdrom/tape/floppy support
CONFIG_BLK_DEV_HD_ONLY	N	Old hard disk (MFM/RLL/IDE) driver
CONFIG_BLK_DEV_LOOP	N	Loopback device support
CONFIG_BLK_DEV_NBD	N	Network block device support
CONFIG_BLK_DEV_MD	N	Multiple devices driver support
CONFIG_BLK_DEV_RAM	Y	RAM disk support
CONFIG_BLK_DEV_INITRD	N	Initial RAM disk (`initrd`) support
CONFIG_BLUECAT_RFS	Y	BlueCat RFS support
CONFIG_BLK_DEV_GENERIC_FLASH_DOC	N	M-Systems DiskOnChip
CONFIG_BLK_DEV_XD	N	XT hard disk support
CONFIG_PARIDE_PARPORT	Y	Parallel port IDE device support
CONFIG_BLK_CPQ_DA	N	Compaq SMART2 support

Table 3-8: Character Devices

Option	Value	Description
CONFIG_VT	N	Virtual terminal
CONFIG_VT_CONSOLE	N	Support for console on virtual terminal

Table 3-8: Character Devices (Continued)

Option	Value	Description
CONFIG_SERIAL	N	Standard/generic (dumb) serial support
CONFIG_SERIAL_EXTENDED	N	Extended dumb serial driver options
CONFIG_SERIAL_EE89712	Y	Support for Cirrus Logic CS89712 serial port
CONFIG_SERIAL_EE89712_CONSOLE	Y	Support for console on Cirrus Logic CS89712 serial port
CONFIG_SERIAL_NONSTANDARD	N	Non-standard serial port support
CONFIG_LCD_EE89712	Y	Support for Cirrus Logic CS89712 LCD display
CONFIG_UNIX98_PTYS	N	Unix98 PTY support
CONFIG_MOUSE	N	Mouse support (not serial mice)
CONFIG_QIC02_TAPE	N	QIC-02 tape support
CONFIG_WATCHDOG	N	Watchdog Timer Support
CONFIG_NVRAM	N	/dev/nvram support
CONFIG_RTC	Y	Enhanced Real Time Clock Support
CONFIG_DTLK	N	Double Talk PC internal speech card support

Table 3-9: Video for Linux

Option	Value	Description
CONFIG_VIDEO_DEV	N	Video for Linux

Table 3-10: Joystick Support

Option	Value	Description
CONFIG_JOYSTICK	N	Joystick support

Table 3-11: Ftape, the Floppy Tape Device Driver

Option	Value	Description
CONFIG_FTAPE	N	Ftape (QIC-80/Travan) support

Table 3-12: Networking Options

Option	Value	Description
CONFIG_PACKET	N	Packet socket
CONFIG_NETLINK	N	Kernel/User netlink socket
CONFIG_RTNETLINK	N	Routing messages
CONFIG_NETLINK_DEV	N	Netlink device emulation
CONFIG_FIREWALL	N	Network firewalls
CONFIG_FILTER	N	Socket Filtering
CONFIG_UNIX	Y	UNIX domain sockets
CONFIG_INET	Y	TCP/IP networking
CONFIG_IP_MULTICAST	N	IP: multicasting
CONFIG_IP_ADVANCED_ROUTER	N	IP: advanced router
CONFIG_IP_PNP	N	IP: kernel level autoconfiguration
CONFIG_IP_ROUTER	N	IP: optimize as router not host
CONFIG_NET_IPIP	N	IP: tunneling
CONFIG_NET_IPGRE	N	IP: GRE tunnels over IP
CONFIG_IP_MROUTE	N	IP: multicast routing
CONFIG_IP_ALIAS	N	IP: aliasing support
CONFIG_SYN_COOKIES	N	IP: TCP syncookie support (not enabled per default)
CONFIG_INET_RARP	N	IP: Reverse ARP
CONFIG_SKB_LARGE	N	IP: Allow large windows (not recommended if <16 MB of memory)

Table 3-12: Networking Options (Continued)

Option	Value	Description
CONFIG_IPX	N	The IPX protocol
CONFIG_ATALK	N	Appletalk DDP

Table 3-13: QoS and/or fair queueing

Option	Value	Description
CONFIG_NET_SCHED	N	Packet scheduler support

Table 3-14: Amateur Radio Support

Option	Value	Description
CONFIG_HAMRADIO	N	Amateur Radio support

Table 3-15: IrDA Subsystem

Option	Value	Description
CONFIG_IRDA	N	IrDA subsystem support

Table 3-16: Network Device Support

Option	Value	Description
CONFIG_NETDEVICES	Y	Network device support
CONFIG_DUMMY	N	Dummy net driver support
CONFIG_EQUALIZER	N	EQL (serial line load balancing) support
CONFIG_NET_SB1000	N	General Instruments Surfboard 1000
CONFIG_FDDI	N	FDDI driver support

Table 3-16: Network Device Support (Continued)

Option	Value	Description
CONFIG_PPP	N	PPP (point-to-point) support
CONGIG_SLIP	N	SLIP (serial line) support
CONFIG_NET_RADIO	N	Wireless LAN (non-harmradio)
CONFIG_NET_FC	N	Fibre Channel driver support
CONFIG_SBNI	N	SBNI 12-xx support

Table 3-17: ARCnet Devices

Option	Value	Description
CONFIG_ARCNET	N	ARCnet support

Table 3-18: Ethernet (10 or 100Mbit)

Option	Value	Description
CONFIG_NET_ETHERNET	Y	Ethernet (10 or 100 Mbit)
CONFIG_CS8900A	Y	Support for Cirrus Logic CS8900 Ethernet card
CONFIG_ARM_AM79C961A	N	AM79C961A support
CONFIG_NET_VENDOR_3COM	N	3COM cards
CONFIG_LANCE	N	AMD LANCE and PCnet (AT 1500 and NE2100) support
CONFIG_NET_VENDOR_SMC	N	Western Digital/SMC cards
CONFIG_NET_VENDOR_RACAL	N	Racal-Interlan (micom) NI cards
CONFIG_NET_ISA	N	Other ISA cards
CONFIG_NET_EISA	N	EISA, VLB, PCI and on board controllers
CONFIG_PCNET32	N	AMD PCnet32 (VLB and PCI) support
CONFIG_APRICOT	N	Apricot Xen-II on board Ethernet

Table 3-18: Ethernet (10 or 100Mbit) (Continued)

Option	Value	Description
CONFIG_CS89x0	N	CS89x0 support
CONFIG_DM9102	N	PCI DM9102 support
CONFIG_DE4x5	N	Generic DECchip & DIGITAL EtherWORKS PCI/EISA
CONFIG_DEC_ELCP	N	DECchip Tulip (dc21x4x) PCI support
CONFIG_DGRS	N	Pigi Intl. RightSwitch SE-X support
CONFIG_EEXPRESS_PRO100	N	EtherExpress PRO/100 support
CONFIG_NE2K_PCI	N	PCI NE2000 support
CONFIG_TLAN	N	TI ThunderLAN support (EXPERIMENTAL)
CONFIG_NET_POCKET	N	Pocket and portable adaptors

Table 3-19: Token Ring Devices

Option	Value	Description
CONFIG_TR	N	Token Ring driver support

Table 3-20: WAN Interfaces

Option	Value	Description
CONFIG_HOSTESS_SV11	N	Comtrol Hostess SV-11 support
CONFIG_COSA	N	COSA/SRP sync serial boards support
CONFIG_SEALEVEL_4021	N	Sealevel Systems 4021 support
CONFIG_DLCI	N	Frame relay DLCI support
CONFIG_SBNI	N	SBNI Leased Line Adapters

Table 3-21: SCSI Support

Option	Value	Description
CONFIG_SCSI	N	SCSI support

Table 3-22: Filesystems

Option	Value	Description
CONFIG_QUOTA	N	Quota support
CONFIG_AUTOFS_FS	N	Kernel automounter support
CONFIG_AFFS_FS	N	Amiga FFS filesystem support
CONFIG_HFS_FS	N	Apple Macintosh filesystem support (exp)
CONFIG_FAT_FS	N	DOS FAT filesystem support
CONFIG_MSDOS_FS	N	MSDOS filesystem support
CONFIG_UMSDOS_FS	N	UMSDOS: UNIX-like filesystem on top of standard MSDOS filesystem
CONFIG_VFAT_FS	N	vfat filesystem support
CONFIG_ISO9660_FS	N	ISO 9660 CD-ROM filesystem support
CONFIG_JOLIET	N	Microsoft Joliet cdrom extensions
CONFIG_MINIX_FS	N	Minix filesystem support
CONFIG_NTFS_FS	N	NTFS filesystem support (read only)
CONFIG_HPFS_FS	N	OS/2 HPFS filesystem support (read only
CONFIG_PROC_FS	Y	/proc filesystem support
CONFIG_ROMFS_FS	N	ROM filesystem support
CONFIG_EXT2_FS	Y	Second extended filesystem support
CONFIG_SYSV_FS	N	System V and Coherent filesystem support
CONFIG_UFS_FS	N	UFS filesystem support
CONFIG_BLUECAT_FFS	N	BlueCat Linux Flash File System support

Table 3-23: Network File Systems

Option	Value	Description
CONFIG_CODA_FS	N	Coda filesystem support (advanced network filesystem)
CONFIG_NFS_FS	Y	NFS filesystem support
CONFIG_SUNRPC	Y	SUN RPC support
CONFIG_LOCKD	Y	`lockd` support
CONFIG_SMB_FS	N	SMB filesystem support (to mount WfW shares etc.)
CONFIG_NCP_FS	N	NCP filesystem support (to mount NetWare volumes)

Table 3-24: Partition Types

Option	Value	Description
CONFIG_BSD_DISKLABEL	N	BSD disklabel (BSD partition tables) support
CONFIG_MAC_PARTITION	N	Macintosh partition map support
CONFIG_SMD_DISKLABEL	N	SMD disklabel (Sun partition tables) support
CONFIG_SOLARIS_X86_PARTITION	N	Solaris (x86) partition table support
CONFIG_NLS	N	Native language support

Table 3-25: Kernel Hacking

Option	Value	Description
CONFIG_FRAME_POINTER	N	Compile kernel with frame pointer (for useful debugging)
CONFIG_DEBUG_ERRORS	N	Verbose kernel error messages
CONFIG_DEBUG_USER	N	Verbose user fault messages
CONFIG_DEBUG_INFO	N	Include debugging information in kernel binary

Table 3-25: Kernel Hacking (Continued)

Option	Value	Description
CONFIG_MAGIC_SYSRQ	N	Magic SysRq key
CONFIG_BLUECAT_KDBG	N	Include kdbg kernel debugger

Table 3-26: LynuxWorks Messenger Support

Option	Value	Description
CONFIG_BLUECAT_IOPMAN	N	Enable Lynx IOP Manager support
CONFIG_BLUECAT_IOPMAN_SITKA	N	IOP Manager Sitka support
CONFIG_BLUECAT_MSNG	N	Enable Messenger Support
CONFIG_BLUECAT_MINET	N	MINET support
CONFIG_BLUECAT_MSNG_API	N	Messenger user API support

Table 3-27: Modular Advanced Power Management

Option	Value	Description
CONFIG_BLUECAT_APM	N	Modular Advanced Power Management support

CHAPTER 4

Supported Demo Systems

This chapter provides information about BlueCat Linux demo systems supported by the clee89712 Target Support Package (TSP).

Demo Systems

The following table lists demo systems supported by the clee89712 TSP, their default boot devices, and their RAM and ROM requirements:

Table 4-1: Demo Systems Supported by the clee89712 TSP

Demo System	Default Supported Boot Device(s)	ROM Requirements	RAM Requirements
`developer`	• **Ethernet** - via `osloader`	2753 KB	10240 KB
`osloader`	• **Ethernet** - via `osloader`	1399 KB	9216 KB
`showcase`	• **Ethernet** - via `osloader` • **Flash Memory**	1909 KB	9728 KB

NOTE: *While flash memory is not a default boot device for either* `osloader` *or* `developer`*, the CS89712 target board can be configured to support it by using* `make xconfig` *and selecting the Flash File System (FFS) option.*

developer Demo System

The `developer` demo system is a package consisting of the functionalities of the `shell`, `ftp`, `ping`, `gdb`, and `vl_demo` systems; refer to "Chapter 4 - BlueCat Linux Demo Systems" of the *BlueCat Linux User's Guide* for descriptions of `developer` and its component demo systems.

osloader Demo System

`osloader` is the BlueCat OS Loader system used to boot a BlueCat Linux system on target boards; refer to "Chapter 4 - BlueCat Linux Demo Systems" of the *BlueCat Linux User's Guide* for details.

showcase Demo System

This section and its subsections provide information about the `showcase` demo system. The `showcase` demo system showcases certain features specific to the CS89712 target board and configures an Apache web server. A generic description of `showcase` can be found "Chapter 4 - BlueCat Linux Demo Systems" of the *BlueCat Linux User's Guide*. Board-specific features are described below:

DEMO — An Apache web server demonstration system

SYNOPSIS — This demo system starts and configures the Apache HTTP daemon turning the target board into a web server.

REQUIREMENTS

Storage:	Medium
RAM:	Medium
Network:	Yes
Disk:	None
Special:	None
Kernel Option:	When not mounting a Flash File System (FFS) as the root file system, the `ramdisk_size=6000` option *must* be specified. If an FFS image is installed into the target's flash memory to be mounted as a root

file system (see “Downloading and Booting showcase with an FFS as the RFS” on page 38), then the `ee89712_part_conf="13-56" root=1f01` options must be specified.

DESCRIPTION

The `showcase` demo system automatically boots up in single-user mode. When the user issues the `init` command, `bash` is also automatically run, signalling successful completion by displaying the `bash#` prompt.

Network interfaces are set up in the `.bashrc.clee89712` file, located in the `$BLUECAT_PREFIX/demo.clee89712/showcase/local/etc` directory. When showcase is booted up, the IP address values in this file are automatically used. If a user wishes to reset these values, however, this can also be done by manually bringing up the network interface(s) using the `ifconfig` command with a user-selected target IP address being substituted for *target_IP_Address*, as shown in this example:

```
bash# ifconfig eth0 target_IP_Address
```

Now the Apache server is accessible from any networked system using the IP address entered by the user, serving the `index.html` page located in the `$BLUECAT_PREFIX/demo.clee89712/showcase/local/html.clee89712` directory.

Additionally, a user can also set up the kernel routing table using the `route` command, as in this example:

```
bash# route add default gw gateway_IP_Address
```

The `showcase` demo system can be downloaded into the target’s flash memory as either a BlueCat Linux image composed of a kernel image and a compressed root filesystem (as detailed in “Downloading a Demo System Into Target Flash Memory” on page 17), or as a BlueCat Linux image containing a kernel image and root FFS image (as detailed in “Downloading and Booting showcase with an FFS as the RFS” on page 38).

Downloading and Booting showcase with an FFS as the RFS

This section describes how to configure and download the `showcase` demo system into the target's flash memory with the FFS image mounted as a root file system (RFS). This procedure assumes the target board is booted and that the i_osloader (BLOSH) prompt (>) is displaying:

1. Rebuild the `showcase.kdi` BlueCat Linux image on the cross development host so it does not include a compressed RFS image. Additionally, the kernel command line in the BlueCat Linux image *must* have the `ee89712_part_conf` and `root` options *enabled*, and the flash memory partition configuration string *must* be in accordance with the argument supplied to the `flash_fdisk` command in step 2 below.

   The following commands, when issued on the cross development host, rebuild the BlueCat Linux image and set the appropriate kernel command line:

   ```
   BlueCat:# cd\
   $BLUECAT_PREFIX/demo/showcase

   BlueCat:# echo “ee89712_part_conf=\
   \”13-56\” root=1f01” | mkboot -m -k\
   showcase.kernel -c stdin showcase.kdi
   ```

2. While still on the cross development host, copy `showcase.jffs` and `showcase.kdi` to the `tftpboot` directory, using the following command:

   ```
   BlueCat:# cp showcase.jffs \
   showcase.kdi /tftpboot
   ```

3. From the target board's interface, create two partitions on the target's flash memory, one for the kernel image and the other for the root filesystem image, by entering the following command at the `i_osloader` (BLOSH) prompt on the target board:

   ```
   > exec flash_fdisk /dev/mtdchar0 \
   0-12:13-56
   ```

This command creates a 768 KB first partition for the kernel at the beginning of the target's flash memory, and a 4.6 MB second partition for the root filesystem image immediately following the first partition.

4. Set the environment variables to properly configure network access by entering the following commands:

   > **set IP** *target_IP_address*

   > **set HOST** *host_IP_address*

   > **set IF eth0**

5. Set the `FILE` environment variable so that it points to the `showcase` FFS image, by entering the following command:

   > **set FILE tftp /tftpboot/showcase.jffs**

6. Burn the FFS image into the target's flash memory, by entering the following command:

   > **flash /dev/mtdchar2 erase**

7. Set the `FILE` environment variable so that it points to the `showcase` BlueCat Linux image, for example, by entering the following command:

   > **set FILE tftp /tftpboot/showcase.kdi**

NOTE: *To download and boot other custom systems with FFS mounted as root, the kernel needs to be rebuilt with FFS and MTD (Memory Technology Device) enabled.*

8. Burn the BlueCat Linux image into the target's flash memory, by entering the following command:

   > **flash /dev/mtdchar1 erase**

9. Reset the target board by entering the following command:

```
> reset
```

NOTE: *Because the kernel must scan the entire flash partition, mounting the* `showcase` *FFS image as a root file system takes about 40 seconds during the kernel bootstrap process.*

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 (`192.168.1.4`) defined in the `showcase` demo system. To change the default IP address for the `showcase` demo system, open the `showcase` demo system's `.bashrc` 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.clee89712/ \
showcase/local/etc
```

2. Change the `showcase` demo system's `.bashrc.clee89712` default target board's IP address (`192.168.1.4`) to a new user-selected one, for example, **`216.100.252.140`**, by editing it as follows:

   Find this line:

```
TARGET_IP=192.168.1.4
```

   Edit it to read as follows:

```
TARGET_IP=216.100.252.140
```

3. Set a gateway IP address by entering a value at this line in both the `.bashrc` *and* `.bashrc.clee89712` files as well:

   ```
   GATE_IP=
   ```

4. Rebuild the `showcase` demo system by entering the following commands in the $BLUECAT_PREFIX/`demo.clee89712/showcase` directory:

   ```
   Bluecat:# touch showcase.spec

   Bluecat:# make rootfs

   Bluecat:# make kdi
   ```

5. Copy the updated `showcase.kdi` to the `tftpboot` directory to replace the old `showcase.kdi`.

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 Figure 4-1 below:

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

CHAPTER 5 Supported Device Drivers

The following table lists the device drivers supported by the clee89712 TSP and details important information about them:

Table 5-1: Device Drivers Supported by the clee89712 TSP

Hardware Device	Device Drivers	Location in Source Tree	Kernel Configuration Options
UART Two 16650-compatible devices	`serial_ep7212.c`	`drivers/char`	`CONFIG_SERIAL_EE89712` `CONFIG_SERIAL_EE89712_CONSOLE`
LCD Controller	`con_ep7212.c`	`drivers/char`	`CONFIG_LCD_EE89712`
Ethernet CS8900A	`cs8900a.c` `cs8900a.h`	`drivers/net`	`CONFIG_CS8900A`
Real Time Clock	`rtc_ep7212.c` `rtc_ep7212.h`	`drivers/char` `include/linux`	`CONFIG_RTC`

CHAPTER 6 *Thumb Support Software*

Thumb is an extension of ARM (Advanced RISC Machine) architecture that allows users to achieve 32-bit performance and address space without incurring the costs associated with going to a 32-bit system. It has 36 instruction formats drawn from the standard 32-bit ARM instruction set. These have been recoded into 16-bit wide op-codes. This results in very high code density, as Thumb instructions are half the width of ARM instructions. On execution, these new 16-bit Thumb op-codes are decompressed by the processor to their ARM instruction set equivalents. The decompressed 16-bit Thumb op-codes are then run on an ARM core.

Thumb is not just another mixed instruction-set concept. Thumb-aware cores have two separate instruction sets. This provides a unique advantage over single instruction-set cores, as Thumb provides designers with all the power of ARM's 32-bit instruction set while simultaneously maintaining the code-size advantages of the Thumb double instruction-set design. The fact that the two instruction sets that make up the Thumb double instruction-set are distinct, results in extremely simple decoding logic. This simplification enables small silicon size and maintains ARM's industry-leading low-power capability and MIPS/Watt performance.

Thumb-aware cores—such as the ARM7TDMI—have a full 32-bit architecture, similar to ARM. Because of this, designers retain 32-bit RISC performance. The combination of the two instruction-sets running on a 32-bit Thumb-aware core provides an effective solution for code size and performance issues of 16-bit systems.

NOTE: *Not all ARM cores support* Thumb*. For example, the StrongARM SA-110, supported by BlueCat Linux, does not contain a* Thumb *extension.*

Installing BlueCat Linux Thumb Support Software

BlueCat Linux Thumb support software is distributed on a separate CD-ROM from the rest of the BlueCat Linux distribution CD-ROMs. The Thumb software CD-ROM is used to install Thumb support on top of the BlueCat Linux development environment installed on the cross development host. (See the *BlueCat Linux User's Guide* for a detailed description of the BlueCat Linux installation procedure.)

To install the Thumb support software follow these steps:

1. Insert the Thumb distribution CD-ROM in the cross development host CD-ROM drive.
2. If the cross development host does not auto-mount the CD-ROM, mount the CD-ROM on the host.
3. Go to the top of your BlueCat Linux installation for the ARM target board, and enable the BlueCat Linux execution environment:

```
$ cd $HOME/bluecat

$ . SETUP.sh clee89712

BlueCat:$
```

4. Run the **`install`** installation script from the top of the Thumb distribution CD-ROM to install Thumb support cross-development and target support packages in the BlueCat Linux environment:

```
BlueCat:$ /mnt/cdrom/install
```

Upon successful completion of the steps above, Thumb support is installed in the BlueCat Linux development environment on the cross development host.

Thumb Support Directory Structure Overview

The installation procedure described in "Installing BlueCat Linux Thumb Support Software" on page 46 results in the creation of the following Thumb-specific files and directories shown in gray below:

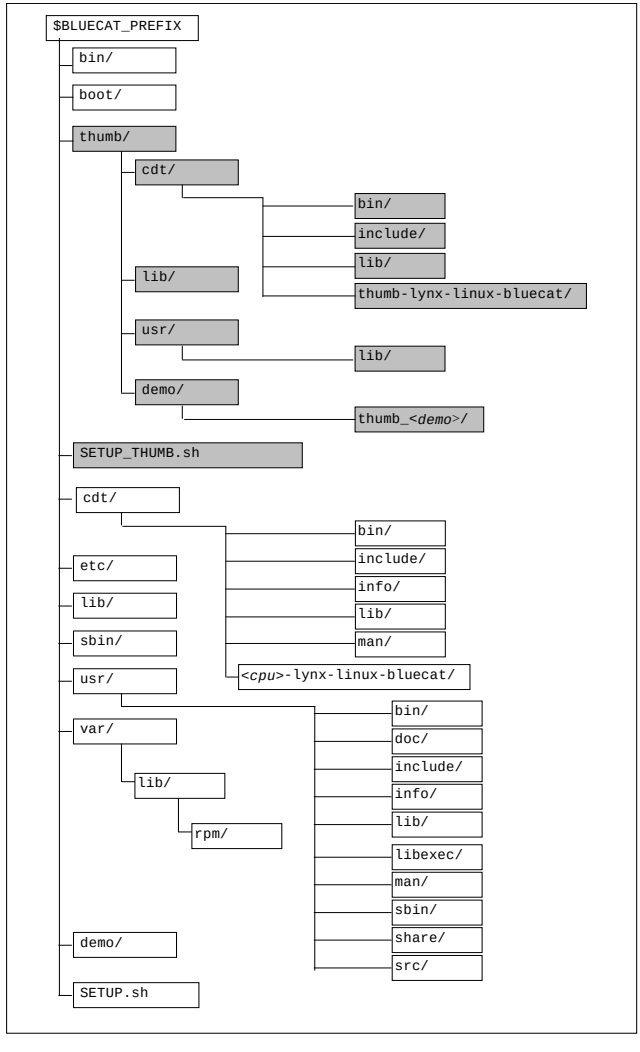

Figure 6-1: Thumb-Specific Directories and Files

The `thumb/cdt` subtree contains the Thumb cross-development tools.
The `thumb/lib` subtree contains the Thumb target board libraries.
The `thumb/demo` subtree contains the Thumb-specific demo systems.

The following table briefly describes the key components of the Thumb-specific directories in the BlueCat Linux tree:

Table 6-1: Key Components of Thumb-Specific Directories

Node	Description
`$BLUECAT_PREFIX`	Installation directory
`- thumb/`	Thumb-specific subtree
`-- cdt/`	Thumb cross-development tools
`--- bin/`	Thumb cross-development binaries
`--- include/`	Thumb cross-development `include` files
`--- lib/`	Thumb cross-development libraries
`--- thumb-lynx-linux-bluecat/`	Thumb cross-development binaries
`-- lib/`	Thumb libraries
`-- usr/`	Thumb top-level `usr` directory
`--- lib/`	More Thumb libraries
`-- demo/`	BlueCat Linux configurations
`--- thumb_`*`demo`*`/`	Thumb-specific demo system configurations
`SETUP_THUMB.sh`	Shell script for setting up Thumb environment

Thumb Cross-Development Environment

The Thumb cross-development environment does not replace the cross-development environment for the ARM mode. Instead, the cross-development tools for both execution modes are available in parallel.

Thumb cross-development tools are available in the directory `$BLUECAT_PREFIX/thumb/cdt` in the BlueCat Linux development tree.

The GNU toolkit does not support switching between compiling for the ARM instruction set and the Thumb instruction set on anything other than a

per-file basis. There are two completely different compilers. One produces the ARM assembler instructions, and the other produces the Thumb assembler instructions.

The standard development tools for the ARM mode are the default configuration, while the Thumb development tools are optional. This means that the `gcc` (`cc`, `egcs`) commands start the standard ARM compiler. To run the Thumb compiler, users can choose one of the following options:

- Enter a command specifying the full path to the compiler:

  ```
  $BLUECAT_PREFIX/thumb/cdt/ \
  thumb-lynx-linux-bluecat/bin/gcc
  ```

- Use one of the Thumb-specific commands, **`tcc`**, **`tgcc`** or **`tegcs`** to start the Thumb compiler. These Thumb-specific tools reside in the BlueCat Linux tree along with the standard cross- development tools in the `$BLUECAT_PREFIX/cdt /thumb-lynx-linux-bluecat/bin` directory. The commands are symbolic links to the actual Thumb tools, as shown here:

  ```
  tcc -> ../../../thumb/cdt/thumb-lynx-linux \
  -bluecat/bin/cc
  tegcs -> ../../../thumb/cdt/thumb-lynx-linux \
  -bluecat/bin/egcs
  tgcc -> ../../../thumb/cdt/thumb-lynx-linux \
  -bluecat/bin/gcc
  ```

  The Thumb-specific commands shown above are available to the user immediately after the BlueCat Linux Thumb-specific `SETUP_THUMB.sh` script is executed from the BlueCat environment.

  Run the `SETUP_THUMB.sh` script from the BlueCat Linux root directory by entering the following commands:

  ```
  BlueCat:$ cd $BLUECAT_PREFIX

  BlueCat:$ . SETUP.sh clee89712

  BlueCat:$ . SETUP_THUMB.sh
  ```

  Once the script above is executed, the standard commands, `cc`, `gcc` or `egcs` run the Thumb tools.

A new option, `--thumb-bc`, has been added to the standard linker `ld`. This option is used to mark the executable as a Thumb application. To produce a Thumb executable from the object files, users must specify the

`--thumb-bc` option in the command line. Use the `-static` option to build a statically linked Thumb application, for example, by entering the following command:

```
ld ... -o thumbexec ... file1.o file2.o --thumb-bc \
-static
```

If the Thumb executable is linked by calling the Thumb-specific `tgcc`, the `--thumb-bc` option is generated and passed to the `ld` command automatically.

A new option, `-mthumb-bc`, has been added to the standard assembler to specify the Thumb assembler instructions in the input file.

Placing Thumb Applications onto the Target Board

Users can combine ARM and Thumb applications on the target board and execute them in parallel. Use the `mkrootfs` cross development tools to put Thumb applications in a filesystem downloaded onto the target board.

Thumb Support in the Kernel

Users must enable Thumb support in the BlueCat Linux kernel to run Thumb applications on the target board. Thumb support is configured in the kernel using the `CONFIG_BLUECAT_THUMB` kernel configuration option.

Use the standard kernel configuration tools (for instance, `make xconfig`) to enable the Thumb support in the kernel. From the main menu, go to `General Setup`, and enable the BlueCat Linux kernel support for the Thumb binaries option.

Thumb Demo Systems

The following Thumb demo systems reside in the `$BLUECAT_PREFIX/thumb/demo` directory and describe the build and execution of Thumb applications on the 7T ARM target board:

- The `hello_thumb` demo system shows the execution of an `init`-like Thumb application on the target board.
- The `shell_thumb` demo system shows the execution of a Thumb application in parallel with ARM applications. Its functionality is identical to that of a normal `shell` demo system. This demo system starts the Thumb mode shell, and then runs all the other included utilities in the ARM mode to demonstrate how they can be executed in parallel.
- The `hello_dynamic_thumb` demo system shows the dynamically-linked `hello_world` application.
- The `gdb_thumb` demo system shows debugging of a simple Thumb application with the Thumb GDB debugger.

Building a Thumb Demo System

The demo systems included in the Thumb distribution were originally configured for a specific TSP; the `.config` filenames reflect the relevant target board's name. If users wish to run these demo systems on other ARM target boards such as, in this case, the CS89712, they must reconfigure the kernel by running `make xconfig`. For example, to build the `shell_thumb` demo system for CS89712 target boards, users would perform the following steps:

1. Copy the original `.config` file (`arm_willow.config`) to a new file with a `clee89712` identifing string by entering the following commands:

```
BlueCat:$ cp \

shell_thumb_arm_willow.config \

shell_thumb_clee89712.config
```

2. Set up the kernel to run on the CS89712 target board by entering the following command:

```
BlueCat:$ make xconfig
```

3. Enter the `System and Processor Type` submenu, and change the `CMA120/Willow` system type to `CL-EE89712-based`.
4. Return to the main menu.
5. Enter the `Character devices` submenu.
6. Deselect the `Standard/generic (dumb) serial support` option.
7. Select the `Cirrus Logic EE89712 serial port support` and `Support for console on serial port` options.
8. Return to the main menu.
9. Select the `Save and Exit` button to save the new `.config` file.
10. Build the demo system by entering the following command:

```
BlueCat:$ make
```

This series of instructions and commands successfully builds a `shell_thumb` demo system. These steps can be repeated for all other Thumb demo systems on the distribution.

For more information on downloading and booting this demo system onto a target board, refer to “Chapter 3 - Downloading and Booting BlueCat Linux” of the *BlueCat Linux User's Guide*.

APPENDIX A *clep_boot Command Reference*

This appendix describes the `clep_boot` utility.

NAME

`clep_boot` – downloads `preboot-clee89712.axf` and `osloader-clee89712.bnd` into target's flash memory.

SYNOPSIS

```
clep_boot [-h] [-v] [-c port]
```

DESCRIPTION

The `clep_boot` utility downloads `preboot-clee89712.axf and osloader-clee89712.bnd` into target's flash memory over a serial connection. To download `preboot-clee89712.axf` and `osloader-clee89712.bnd`, set up the BlueCat Linux cross development environment on the cross development host systems (`. SETUP.sh`), run `clep_boot`, and follow the on-screen instructions.

On the Linux cross development host, the `clep_boot` utility requires `write` access to the serial port special files. If the cross development host system does not allow such access for regular users, this program must be run as root.

When `clep_boot` is invoked on the cross development host several actions occur. `clep_boot` downloads and transfers control to `preboot-clee89712.axf`. Then `preboot-clee89712.axf` resets the CS89712 serial port's data transfer rate from 9600 bps to 115200 bps; this facilitates a faster transfer of `osloader-clee89712.bnd` from the cross development host to the target board.

`clep_boot` then downloads `osloader-clee89712.bnd` from the cross development host to the target board via the same serial connection that `preboot-clee89712.axf` was transferred over, but now, at the faster 115200 bps rate. `osloader-clee89712.bnd` burns itself into the target's flash memory, where it provides an Ethernet interface and a BLOSH prompt in order to download BlueCat Linux demo systems or customized BlueCat Linux kernel images.

OPTIONS

`-h`	Prints a short instruction on the program usage.
`-v`	Verbose output – If this option is specified, all messages coming from the board are printed out.
`-c` *name*	This option tells the program to use port *name* instead of the default port (where *name* is one of either `/dev/ttyS1`/`dev/ttyS0` for Linux, or `COM1`, `COM2` for Windows).

EXAMPLES

To download `osloader-clee89712.bnd` into a target's flash memory over a serial cable connected to the cross development host serial `COM2` port (the default host serial port), enter the following command:

```
BlueCat:# clep_boot
```

Follow the instructions printed on the screen.

If the cable is connected to the cross development host's serial `COM1`, enter the following commands:

- on Linux hosts:

  ```
  BlueCat:# clep_boot -c /dev/ttyS0
  ```

- on Windows hosts:

  ```
  BlueCat:$ clep_boot -c COM1
  ```

Follow the instructions displayed on the screen.