Virtex-5 FXT PowerPC 440 and MicroBlaze Edition Kit Reference Systems

UG511 (v1.2.1) July 30, 2009
## Revision History

The following table shows the revision history for this document.

<table>
<thead>
<tr>
<th>Date</th>
<th>Version</th>
<th>Revision</th>
</tr>
</thead>
<tbody>
<tr>
<td>7/29/08</td>
<td>1.0</td>
<td>Initial Xilinx release.</td>
</tr>
<tr>
<td>12/19/08</td>
<td>1.1</td>
<td>Updated to 10.1.3. Added content on Flash file system. Updated steps on programming parallel Flash. Added chapter on software application to read, write, and erase Flash.</td>
</tr>
<tr>
<td>1/26/09</td>
<td>1.1.1</td>
<td>Updated reference system links.</td>
</tr>
<tr>
<td>5/21/09</td>
<td>1.2</td>
<td>Updated to 11.1. Added steps for applying the xlltemac patch. Updated section on the MLD file set to use the 11.1 file set included in the project.</td>
</tr>
<tr>
<td>7/30/09</td>
<td>1.2.1</td>
<td>Updated ref system CID numbers; updated to current date.</td>
</tr>
</tbody>
</table>
## Table of Contents

**Preface: About This Guide**

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guide Contents</td>
<td>7</td>
</tr>
<tr>
<td>Hardware and Software Requirements</td>
<td>7</td>
</tr>
<tr>
<td>References</td>
<td>8</td>
</tr>
<tr>
<td>Additional Resources</td>
<td>8</td>
</tr>
<tr>
<td>Conventions</td>
<td>8</td>
</tr>
<tr>
<td>Typographical</td>
<td>8</td>
</tr>
<tr>
<td>Online Document</td>
<td>9</td>
</tr>
</tbody>
</table>

**Chapter 1: Hardware Platform**

<table>
<thead>
<tr>
<th>Subsection</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>11</td>
</tr>
<tr>
<td>PowerPC 440 Processor Reference System</td>
<td>11</td>
</tr>
<tr>
<td>Block Diagram</td>
<td>11</td>
</tr>
<tr>
<td>Address Map</td>
<td>12</td>
</tr>
<tr>
<td>System Configuration</td>
<td>12</td>
</tr>
<tr>
<td>MicroBlaze Processor Reference System</td>
<td>13</td>
</tr>
<tr>
<td>Block Diagram</td>
<td>13</td>
</tr>
<tr>
<td>Address Map</td>
<td>13</td>
</tr>
<tr>
<td>System Configuration</td>
<td>14</td>
</tr>
</tbody>
</table>

**Chapter 2: HelloWorld Software Application**

<table>
<thead>
<tr>
<th>Subsection</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>15</td>
</tr>
<tr>
<td>Executing the HelloWorld Software Application</td>
<td>15</td>
</tr>
<tr>
<td>Executing the HelloWorld Application Using the Pre-Built Bitstream</td>
<td>15</td>
</tr>
<tr>
<td>Executing the HelloWorld Software Application from XPS</td>
<td>18</td>
</tr>
<tr>
<td>Commands in the HelloWorld Software Application</td>
<td>19</td>
</tr>
<tr>
<td>Booting the HelloWorld Application from Serial Flash</td>
<td>20</td>
</tr>
</tbody>
</table>

**Chapter 3: LynuxWorks BlueCat Linux**

<table>
<thead>
<tr>
<th>Subsection</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>23</td>
</tr>
<tr>
<td>Executing the BlueCat Linux Images</td>
<td>23</td>
</tr>
<tr>
<td>Executing the BlueCat Linux Image with a Ramdisk File System</td>
<td>23</td>
</tr>
<tr>
<td>Executing the BlueCat Linux Image Using the Pre-Built Bitstream</td>
<td>24</td>
</tr>
<tr>
<td>Executing the BlueCat Linux Image from XPS</td>
<td>25</td>
</tr>
<tr>
<td>Executing the BlueCat Linux Image with a JFFS2 File System</td>
<td>26</td>
</tr>
<tr>
<td>Executing the BlueCat Linux Image Using the Pre-Built Bitstream</td>
<td>27</td>
</tr>
<tr>
<td>Executing the BlueCat Linux Image from XPS</td>
<td>32</td>
</tr>
<tr>
<td>Executing BlueCat Linux Commands</td>
<td>35</td>
</tr>
<tr>
<td>Web Server Demonstration</td>
<td>36</td>
</tr>
<tr>
<td>Building the BlueCat Linux Kernel Image</td>
<td>37</td>
</tr>
<tr>
<td>Installing the BlueCat Linux Distribution</td>
<td>37</td>
</tr>
<tr>
<td>Using the Provided Demo Directories</td>
<td>37</td>
</tr>
</tbody>
</table>
Chapter 4: FlashRWE Software Application

Introduction ................................................................. 53

Executing the FlashRWE Software Application ...................... 53
   Executing the FlashRWE Application Using the Pre-Built Bitstream .... 53
   Executing the FlashRWE Software Application from XPS ................. 55

Functions in the FlashRWE Software Application .................... 55
   Flash Read Menu ......................................................... 57
      Read Bytes of Flash ............................................... 57
      Check if Flash is Empty ........................................... 57
   Flash Write Menu ....................................................... 59
      Write Incrementing Numbers to Flash ............................ 59
      Write Bytes to Flash .............................................. 60
   Flash Erase Menu ....................................................... 61
      Erase Bytes of Flash .............................................. 61
      Erase Blocks of Flash ............................................ 62
      Erase the Entire Flash ........................................... 63
Preface

About This Guide

The Embedded Development HW/SW Kit - Virtex®-5 FXT PowerPC® 440 and MicroBlaze™ Processor Edition showcases various features of the Virtex-5 FXT ML507 development board. This kit includes two hardware systems with a HelloWorld software application, bootable BlueCat Linux image, and a Flash reading, writing, and erasing application. This document describes the hardware platform, the HelloWorld software application, the BlueCat Linux images, and the Flash software application.

The reference systems are available at:

MicroBlaze system:
https://secure.xilinx.com/webreg/clickthrough.do?cid=135330

PowerPC 440 system:
https://secure.xilinx.com/webreg/clickthrough.do?cid=135331

Guide Contents

This manual contains the following chapters:

• Chapter 1, “Hardware Platform,” provides an overview of the IP cores in the reference system. This chapter includes the reference system block diagram and address map.

• Chapter 2, “HelloWorld Software Application,” describes the board tests in the application, how to execute the application, and how to boot the application from SPI Flash.

• Chapter 3, “LynuxWorks BlueCat Linux,” includes information on how to execute the provided BlueCat Linux images and how to build a similar image using the BlueCat Linux development tools.

• Chapter 4, “FlashRWE Software Application” describes the available functions in the application and how to execute the application.

Hardware and Software Requirements

The hardware and software requirements are:

• Xilinx ML507 Development Board
• Xilinx Platform USB Download Cable or Parallel IV Download Cable
• RS232 Serial Cable
• Ethernet Cable
• Serial Communications Utility Program (e.g. HyperTerminal)
References

References used throughout this user guide are listed below.

1. *BlueCat Linux User’s Guide*
2. *BlueCat Linux Board Support Guide for Xilinx Spartan®-3E 1600E Boards*
3. UG083 *Getting Started Tutorial for ML401/ML402/ML403/ML405 Evaluation Platforms*
4. UG200 *Embedded Processor Block in Virtex-5 FPGAs*

Additional Resources

To find additional documentation, see the Xilinx website at:


To search the Answer Database of silicon, software, and IP questions and answers, or to create a technical support WebCase, see the Xilinx website at:


Conventions

This document uses the following conventions. An example illustrates each convention.

**Typographical**

The following typographical conventions are used in this document:

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning or Use</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Courier font</td>
<td>Messages, prompts, and program files that the system displays</td>
<td>speed grade: - 100</td>
</tr>
<tr>
<td>Courier bold</td>
<td>Literal commands that you enter in a syntactical statement</td>
<td>ngdbuild  <em>design_name</em></td>
</tr>
<tr>
<td>Helvetica bold</td>
<td>Commands that you select from a menu</td>
<td>File → Open</td>
</tr>
<tr>
<td></td>
<td>Keyboard shortcuts</td>
<td>Ctrl+C</td>
</tr>
<tr>
<td>Italic font</td>
<td>Variables in a syntax statement for which you must supply values</td>
<td>ngdbuild  <em>design_name</em></td>
</tr>
<tr>
<td></td>
<td>References to other manuals</td>
<td>See the <em>Development System Reference Guide</em> for more information.</td>
</tr>
<tr>
<td></td>
<td>Emphasis in text</td>
<td>If a wire is drawn so that it overlaps the pin of a symbol, the two nets are <em>not</em> connected.</td>
</tr>
</tbody>
</table>

|
## Conventions

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning or Use</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Square brackets [ ]</td>
<td>An optional entry or parameter. However, in bus specifications, such as bus[7:0], they are required.</td>
<td>ngdbuild [option_name] design_name</td>
</tr>
<tr>
<td>Braces {}</td>
<td>A list of items from which you must choose one or more</td>
<td>lowpwr ={on</td>
</tr>
<tr>
<td>Vertical bar</td>
<td>Separates items in a list of choices</td>
<td>lowpwr ={on</td>
</tr>
<tr>
<td>Vertical ellipsis . . .</td>
<td>Repetitive material that has been omitted</td>
<td>IOB #1: Name = QOUT’ IOB #2: Name = CLKN’ . . .</td>
</tr>
<tr>
<td>Horizontal ellipsis ...</td>
<td>Repetitive material that has been omitted</td>
<td>allow block block_name loc1 loc2 ... locn;</td>
</tr>
</tbody>
</table>

### Online Document

The following conventions are used in this document:

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning or Use</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blue text</td>
<td>Cross-reference link to a location in the current document</td>
<td>See the section “Additional Resources” for details. Refer to “Title Formats” in Chapter 1 for details.</td>
</tr>
<tr>
<td>Red text</td>
<td>Cross-reference link to a location in another document</td>
<td>See Figure 2-5 in the Virtex-II Platform FPGA User Guide.</td>
</tr>
<tr>
<td>Blue, underlined text</td>
<td>Hyperlink to a website (URL)</td>
<td>Go to <a href="http://www.xilinx.com">http://www.xilinx.com</a> for the latest speed files.</td>
</tr>
</tbody>
</table>
Chapter 1

Hardware Platform

Introduction

The Virtex-5 FXT Development Kit includes two reference systems that target the ML507 development board. One reference system is based on the PowerPC 440 (PPC440) processor and the other reference system is based on the MicroBlaze processor. Both the PPC440 processor reference system and the MicroBlaze processor reference system are created to run BlueCat Linux. The BlueCat Linux images are described in Chapter 3, “LynuxWorks BlueCat Linux”. The HelloWorld software application described in Chapter 2, “HelloWorld Software Application,” and the FlashRWE software application described in Chapter 4, “FlashRWE Software Application” can be run on either the MicroBlaze processor reference system or the PPC440 processor reference system. Both the PowerPC and MicroBlaze processor systems are described in this chapter.

PowerPC 440 Processor Reference System

Block Diagram

The block diagram for the PowerPC 440 processor reference system is shown in Figure 1-1.
Chapter 1: Hardware Platform

Address Map

The address map for the IP cores in the PowerPC 440 processor reference system is given in Table 1-1.

<table>
<thead>
<tr>
<th>Instance</th>
<th>Peripheral</th>
<th>Base Address</th>
<th>High Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>xps_bram_if_cntlr_1</td>
<td>xps_bram</td>
<td>0xFFFF0000</td>
<td>0xFFFFFFFF</td>
</tr>
<tr>
<td>LEDs_8Bit</td>
<td>xps_gpio</td>
<td>0x81400000</td>
<td>0x8140FFFF</td>
</tr>
<tr>
<td>LEDs_Positions</td>
<td>xps_gpio</td>
<td>0x81420000</td>
<td>0x8142FFFF</td>
</tr>
<tr>
<td>Push.Buttons_5Bit</td>
<td>xps_gpio</td>
<td>0x81440000</td>
<td>0x8144FFFF</td>
</tr>
<tr>
<td>DIP_Switches_8Bit</td>
<td>xps_gpio</td>
<td>0x81460000</td>
<td>0x8146FFFF</td>
</tr>
<tr>
<td>xps_timebase_wdt_1</td>
<td>xps_timebase_wdt</td>
<td>0x83A00000</td>
<td>0x83A0FFFF</td>
</tr>
<tr>
<td>xps_intc_0</td>
<td>xps_intc</td>
<td>0x81800000</td>
<td>0x8180FFFF</td>
</tr>
<tr>
<td>IIC_EEPROM</td>
<td>xps_iic</td>
<td>0x81600000</td>
<td>0x8160FFFF</td>
</tr>
<tr>
<td>Hard_Ethernet_MAC</td>
<td>xps_ll_temac</td>
<td>0x81C00000</td>
<td>0x81C0FFFF</td>
</tr>
<tr>
<td>SysACE_CompactFlash</td>
<td>xps_sysace</td>
<td>0x83600000</td>
<td>0x8360FFFF</td>
</tr>
<tr>
<td>xps_timer_1</td>
<td>xps_timer</td>
<td>0x83C00000</td>
<td>0x83C0FFFF</td>
</tr>
<tr>
<td>RS232_Uart_1</td>
<td>xps_uart16550</td>
<td>0x83E00000</td>
<td>0x83E0FFFF</td>
</tr>
<tr>
<td>DDR2_SDRAM</td>
<td>ppc440mc_ddr2</td>
<td>0x00000000</td>
<td>0x00000000</td>
</tr>
<tr>
<td>FLASH</td>
<td>xps_mch_emc</td>
<td>0xFC000000</td>
<td>0xFDFFFFFF</td>
</tr>
</tbody>
</table>

System Configuration

The PPC440 reference system uses the PowerPC 440 processor block with a processor frequency of 400 MHz. The Memory Interface Block of the processor block is connected to the PPC440MC DDR2 memory controller and is set to operate at a frequency of 200 MHz. The PLB v4.6 bus is connected to the MPLB port of the processor block, which allows the XPS peripherals to be connected as slaves on the bus. The PLB v4.6 bus frequency is 100 MHz.

More information about the PowerPC 440 processor and the embedded processor block can be found in UG200, Embedded Processor Block in Virtex-5 FPGAs.

The LocalLink connection of the XPS LL TEMAC core is connected to the Hard DMA device on the processor block. In the BlueCat Linux demonstration, the Ethernet MAC can run at 10 Mb/s, 100 Mb/s, or 1000 Mb/s, depending on the attached network.
MicroBlaze Processor Reference System

Block Diagram

The block diagram for the MicroBlaze processor reference system is shown in Figure 1-2.

![Block Diagram](image)

Figure 1-2: MicroBlaze Processor System Block Diagram

Address Map

The address map for the IP cores in the MicroBlaze processor reference system is given in Table 1-2.

Table 1-2: MicroBlaze Processor Reference System Address Map

<table>
<thead>
<tr>
<th>Instance</th>
<th>Peripheral</th>
<th>Base Address</th>
<th>High Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>dlmb_cntlr</td>
<td>lmb_bram_if_cntlr</td>
<td>0x00000000</td>
<td>0x00001FFF</td>
</tr>
<tr>
<td>ilmb_cntlr</td>
<td>lmb_bram_if_cntlr</td>
<td>0x00000000</td>
<td>0x00001FFF</td>
</tr>
<tr>
<td>debug_module</td>
<td>mdm</td>
<td>0x84400000</td>
<td>0x8440FFFF</td>
</tr>
<tr>
<td>xps_bram_if_cntlr_1</td>
<td>xps_bram</td>
<td>0x88310000</td>
<td>0x8831FFFF</td>
</tr>
<tr>
<td>LEDs_8Bit</td>
<td>xps_gpio</td>
<td>0x81400000</td>
<td>0x8140FFFF</td>
</tr>
<tr>
<td>LEDs_Positions</td>
<td>xps_gpio</td>
<td>0x81420000</td>
<td>0x8142FFFF</td>
</tr>
<tr>
<td>Push.Buttons_5Bit</td>
<td>xps_gpio</td>
<td>0x81440000</td>
<td>0x8144FFFF</td>
</tr>
<tr>
<td>DIP.Switches_8Bit</td>
<td>xps_gpio</td>
<td>0x81460000</td>
<td>0x8146FFFF</td>
</tr>
<tr>
<td>xps_timebase_wdt_1</td>
<td>xps_timebase_wdt</td>
<td>0x83A00000</td>
<td>0x83A0FFFF</td>
</tr>
<tr>
<td>xps_intc_0</td>
<td>xps_intc</td>
<td>0x81800000</td>
<td>0x8180FFFF</td>
</tr>
<tr>
<td>IIC.EEPROM</td>
<td>xps_iic</td>
<td>0x81600000</td>
<td>0x8160FFFF</td>
</tr>
<tr>
<td>Hard.Ethernet.MAC</td>
<td>xps_ll_temac</td>
<td>0x81C00000</td>
<td>0x81C0FFFF</td>
</tr>
<tr>
<td>SysACE.CompactFlash</td>
<td>xps_sysace</td>
<td>0x83600000</td>
<td>0x8360FFFF</td>
</tr>
<tr>
<td>xps_timer_1</td>
<td>xps_timer</td>
<td>0x83C00000</td>
<td>0x83C0FFFF</td>
</tr>
<tr>
<td>RS232_Uart_1</td>
<td>xps_uart16550</td>
<td>0x83E00000</td>
<td>0x83E0FFFF</td>
</tr>
<tr>
<td>FLASH</td>
<td>xps_mch_emc</td>
<td>0x83F00000</td>
<td>0x83F0FFFF</td>
</tr>
<tr>
<td>DDR2_SDRAM</td>
<td>mpmc</td>
<td>0x90000000</td>
<td>0x9FFFFFFFF</td>
</tr>
<tr>
<td>DDR2_SDRAM(SDMA)</td>
<td>mpmc</td>
<td>0x84600000</td>
<td>0x8460FFFF</td>
</tr>
</tbody>
</table>
System Configuration

The MicroBlaze processor reference system runs off a reference clock frequency of 100 MHz from the oscillator on the board. The PLBv46 bus and the MicroBlaze processor run at a frequency of 100 MHz and the DDR2 runs at 200 MHz in this system.

The MicroBlaze processor is configured with the Memory Management Unit (MMU) enabled. The MMU is enabled and is implemented in Virtual mode by setting the MicroBlaze processor parameter C_USE_MMU to 3. In Virtual mode, the MMU controls effective-address to physical-address mapping and supports memory protection. Virtual mode provides greater control over memory protection. Protection and relocation enable system software to support multitasking. This capability gives the appearance of simultaneous or near-simultaneous execution of multiple programs.

The instruction cache and data cache of the MicroBlaze processor are both enabled. The cacheable block of main memory is accessed via the XCL Port Interface Modules (PIM) of the Multi-Port Memory Controller (MPMC).

The MicroBlaze processor system uses the XPS LL TEMAC FPGA with the Virtex-5 Hard TEMAC FPGA to provide Ethernet functionality. The Ethernet MAC can run at 10 Mb/s, 100 Mb/s, or 1000 Mb/s, depending on the attached network.

The XPS MCH EMC memory controller is connected to an external Xilinx Parallel Flash device, which is used to store the hardware configuration bitstream and bootloader application, as well as the BlueCat Linux kernel image.
HelloWorld Software Application

Introduction

The HelloWorld software application is a simple application that exercises a few of the board features. When the application is run, it will first flash the LEDs and read the DIP and push button switches. The user can then select from a list of menu options, including options to select a target memory and read/write an address with necessary data. The HelloWorld software application can run on either the MicroBlaze or the PowerPC 440 processor reference system.

Two methods for downloading and running the HelloWorld software application are listed below:

- Use a debugger, such as XMD (provided as part of the EDK tools), to download the executable file directly into BRAM. This method is described in the section “Executing the HelloWorld Software Application”.
- Program Flash memory with the HelloWorld software application. This method is described in the section “Booting the HelloWorld Application from Serial Flash”. Once Flash memory is programmed, the HelloWorld software application can be run by setting the FPGA configuration mode pins to SPI mode and either powering up the development board or depressing the PROG button on the board.

Note: A warning box will appear during some of the steps in this chapter. The warning box states that “Software development features in XPS are deprecated, and will be removed in the next major release”. Click OK to safely ignore this warning. To turn off this warning completely, navigate to Edit→Preferences in XPS. Select Application Preferences and check the box that states “Do not show “Software Features Deprecated” dialog box”.

Executing the HelloWorld Software Application

To execute the HelloWorld software application, program the hardware bitstream to the Virtex-5 FX device and load the HelloWorld software application into BRAM. Program the bitstream by downloading the pre-built bitstream from the ready_for_download directory or generate and download it from XPS. Similarly, the HelloWorld executable can be downloaded from the ready_for_download directory or built and downloaded through XPS.

Executing the HelloWorld Application Using the Pre-Built Bitstream

To execute the application using the files inside the ready_for_download directory in the project root directory, follow these steps:

1. Connect the Platform USB cable or the Parallel IV JTAG cable between the host computer and the Virtex-5 FX70T ML507 development board.
2. Connect the serial cable between the host computer and the RS232 port on the Virtex-5 FX70T ML507 development board.

3. Apply power to the Virtex-5 FX70T ML507 development board.

4. Start a HyperTerminal (or similar) session on the host computer with the settings shown in Figure 2-1. Select the COM port corresponding to the connected serial port on the host computer. Set the Baud Rate to 115200, Data bits to 8 bits, Parity to None, Stop bits to 1 bit, and Flow control to None.

5. Through XPS, launch an EDK shell by selecting Project → Launch EDK Shell.

6. In the EDK shell, change directories to the ready_for_download directory in either the MicroBlaze or PowerPC processor reference system.

7. Use iMPACT to download the bitstream by using the following command:

   ```
   $ impact -batch ug511.cmd
   ```

8. Invoke XMD and connect to the processor by the following command:

   ```
   $ xmd -opt ug511.opt
   ```

9. Download the HelloWorld software application into BRAM using the following command:

   ```
   XMD% dow helloworld_executable.elf
   ```
10. To start the HelloWorld software application running, use the following XMD command:

  XMD% run

  a. After the HelloWorld software application runs, the HyperTerminal output will be as shown in Figure 2-2.

```
Walking LEDs test... Observe the LEDs...
LEDs test PASSED.
Writing pseudo random data at address... 0x20FFC
Reading pseudo random data at address... 0x20FFC
Memory Test PASSED!
Press any key to continue...
Type <Menu> for options
->
```

Figure 2-2:  **HelloWorld Output**

b. For an explanation of the available tests in the application, see the section “Commands in the HelloWorld Software Application”.

**Executing the HelloWorld Software Application from XPS**

To execute the reference system using XPS, follow these steps:

1. Perform steps 1-4 in the “Executing the HelloWorld Application Using the Pre-Built Bitstream” section.
2. Open either the MicroBlaze or PowerPC 440 processor reference system project in XPS.
3. Implement the hardware design and create the hardware bitstream by selecting **Hardware** → **Generate Bitstream** in XPS.
4. In the Applications tab, build the **helloworld** project by right-clicking on the project and selecting **Build Project**. This will create the software executable for the application.
5. Download the bitstream to the board by selecting **Device Configuration** → **Download Bitstream** in XPS.
6. After the bitstream has downloaded, launch the XMD by selecting **Debug** → **Launch XMD...** in XPS.
7. Download the HelloWorld application executable using the following command in XMD:

    dow helloworld/executable.elf

8. To run the software application, use the `run` command in XMD.
   a. After the HelloWorld software application runs, the HyperTerminal output will be as shown in Figure 2-2.
   b. For an explanation of the available commands in the application, see the section “Commands in the HelloWorld Software Application”.

### Commands in the HelloWord Software Application

After the HelloWord application is executed, type `Menu` into the terminal console to bring up the HelloWord menu of tests, which is shown in Figure 2-3.

**Note:** The Menu command options are case sensitive.

```
Type <Menu> for options
->Menu

Xilinx Virtex-5 ML507 Development kit Demo Menu:

Mem  Test DDR2 SDRAM
Led  Test the LEDs
PBT  Test the push buttons
Flash Test parallel Flash memory
Test  Perform all factory tests
wrd <addr><# bytes><data> Write DDR2 mem locations with given data
rwd <addr><# bytes>  Read f st DDR2 mem locations and print data
Menu Display Menu options
clrs Clear screen
quit Quit

Type <Menu> for options
->
```

**Figure 2-3: HelloWord Menu**

Table 2-1 lists and describes the commands that are available in the HelloWord application.

**Table 2-1: Description of the HelloWord Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mem</td>
<td>The Mem test performs a destructive 32-bit wide memory test on a 132K byte block of the DDR2 SDRAM memory. This test erases, writes, reads, and verifies the DDR2 memory in the Virtex-5 FX70T ML507 development board. The results of the test will be displayed in the HyperTerminal.</td>
</tr>
<tr>
<td>Led</td>
<td>The Led test flashes each LED with a delay so that it is visible. Once all the LEDs are flashed, it sends the test pass message to the HyperTerminal.</td>
</tr>
<tr>
<td>PBT</td>
<td>The PBT test instructs the user to push the West, South, East, North, and Center buttons and see the specified LEDs glow.</td>
</tr>
</tbody>
</table>
Booting the HelloWorld Application from Serial Flash

This section includes steps on how to program the HelloWorld application into the serial Flash. These steps includes details on how to use, create, and boot serial Flash files for the Virtex-5 FX70T ML507 Development Kit.

Flash files that have already been generated and are ready to use can be found in the `<project root directory>/ready_for_download/Flash_files/` directory.

1. Open the reference system project in XPS.
2. Disconnect the cable attached to header J1 (the header on the left side of the board) from the Xilinx download cable.
3. Connect JTAG flying wires from the Xilinx download cable to the J2 header using the pin labels as a guide on how to make the connections. The 7-pin J2 header is located to the right of the FPGA and just above the LCD panel.
4. Set the configuration address DIP switches to 00010101 (bits 4, 6, and 8 ON).
5. Remove the inserted ML507 CF card if present, and press the Prog button to erase the FPGA.
6. In XPS, compile a bitstream, download.bit, that includes the system configuration and the HelloWorld application. Compile the bitstream by marking the HelloWorld application to Initialize BRAMs, then selecting Device Configuration → Update Bitstream in XPS.
7. Copy the hardware bitstream from `<project root directory>/implementation/download.bit` to the `<project root directory>/ready_for_download/Flash_files/` directory and rename it helloworld.bit to replace the current helloworld.bit file in the directory.
8. Format the BIT file to an MCS file using the cmd file `impact -batch convert_bits_to_mcs.cmd` from the `ready_for_download/flash_files` directory.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flash</td>
<td>This test unlocks, writes, reads, and verifies the Parallel Flash memory in the Virtex-5 FX70T ML507 development board. The results of the test will be displayed in the HyperTerminal.</td>
</tr>
<tr>
<td>Test</td>
<td>This performs all the factory tests mentioned above for the Virtex-5 FX70T ML507 development board and displays the results to the HyperTerminal.</td>
</tr>
<tr>
<td>mwr &lt;addr&gt;&lt;# bytes&gt;&lt;data&gt;</td>
<td>This test writes the given data to the DDR2 memory locations specified. The address range should be within the DDR2 base address and high address.</td>
</tr>
<tr>
<td>mrd &lt;addr&gt;&lt;# bytes&gt;</td>
<td>This test reads the number of bytes specified from the DDR2 memory location given. The address range should be within the DDR2 base address and high address.</td>
</tr>
<tr>
<td>Menu</td>
<td>This command lists the menu options for the user.</td>
</tr>
<tr>
<td>cls</td>
<td>This command clears the HyperTerminal screen.</td>
</tr>
</tbody>
</table>

Table 2-1: Description of the HelloWorld Commands (Cont’d)
9. Launch iMPACT, then double-click **Direct SPI Configuration** in the iMPACT Flows window to program the SPI Flash device.
10. Right-click the Direct SPI Configuration tab, then select **Add SPI Device...**
11. Navigate to the `<project root directory>/ready_for_download/flash_files/helloworld.mcs` created above, then click Open.
12. In the Select Device Part Name drop-down dialog box, select **M25P32**, then click **OK**.
13. Click **OK** in the Device Programming Properties box.
14. The Direct SPI Configuration tabbed window displays a diagram of a single SPI PROM. Right-click on the SPI PROM, then select **Program**.
15. On the board, change the configuration address / mode DIP switches to **00000101** (bits 6 and 8 ON).
16. Press the **Prog** button. The design takes about 10 seconds to finish loading and begin to run. The serial output is shown in Figure 2-3.
LynuxWorks BlueCat Linux

Introduction

BlueCat Linux can be targeted to run on the PowerPC 440 processor or the MicroBlaze soft processor with the MMU enabled. Example BlueCat Linux images are provided that are tailored to the Virtex-5 FXT ML507 development board and the hardware systems that are described in Chapter 1, “Hardware Platform.”. Two BlueCat Linux images are provided with each hardware system, one that boots with a ramdisk root file system and one that uses a Journalling Flash File System, version 2 (JFFS2). The Virtex-5 FXT Development Kit also includes example demo directories, which allow the user to rebuild the example kernel images with the LynuxWorks BlueCat Linux development tools.

The methods for downloading and running the BlueCat Linux kernel demonstrations are listed below.

- Use a debugger, such as XMD (provided as part of the EDK tools), to download the image file directly into DDR2. This method is described in the section “Executing the BlueCat Linux Images”.
- Program Flash memory with the BlueCat Linux image. This method is described in the section “Booting the BlueCat Linux Image from Parallel Flash”. Once Flash memory is programmed, the BlueCat Linux demonstration can be run by setting the configuration mode switches to Platform Flash and either powering up the development board or depressing the PROG button on the board.

Note: A warning box will appear during some of the steps in this chapter. The warning box states that “Software development features in XPS are deprecated, and will be removed in the next major release”. Click OK to safely ignore this warning. To turn off this warning completely, navigate to Edit→Preferences in XPS. Select Application Preferences and check the box that states “Do not show “Software Features Deprecated” dialog box”.

Executing the BlueCat Linux Images

Two BlueCat Linux images are provided with each hardware system. One BlueCat Linux image uses a ramdisk root file system. The root file system is included in the image so it is a self contained image that can be booted quickly. One image uses a JFFS2 root file system. The JFFS2 file system must be written to the Flash memory before the Linux image can be booted, but it allows for persistent storage. This section details how to execute the different BlueCat Linux images.

Executing the BlueCat Linux Image with a Ramdisk File System

To boot the BlueCat Linux image, the hardware bitstream must be programmed to the Virtex-5 FXT device and the BlueCat Linux kernel image must be downloaded to the DDR2 memory. Programming the bitstream can be done by either downloading the pre-built
bitstream from the `ready_for_download` directory or generating and downloading it from XPS. The BlueCat Linux kernel image is downloaded from the `bclinux_images` directory.

**Executing the BlueCat Linux Image Using the Pre-Built Bitstream**

To execute the reference system using the files inside the `ready_for_download` directory in the system root directory, follow these steps:

1. Connect the Platform USB cable or the Parallel IV JTAG cable between the host computer and the Virtex-5 FXT ML507 development board.
2. Connect the serial cable between the host computer and the RS232 port on the Virtex-5 FXT ML507 development board.
3. Apply power to the Virtex-5 FXT ML507 development board.
4. Start a HyperTerminal (or similar) session on the host computer. Select the COM port corresponding to the connected serial port on the host computer. Set the Baud Rate to 115200, Data bits to 8 bits, Parity to None, Stop bits to 1 bit, and Flow control to None.
5. Through XPS, launch an EDK shell by selecting **Project → Launch EDK Shell**.
6. In the EDK shell, change directories to the `ready_for_download` directory.
7. Use iMPACT to download the bitstream by using the following command:
   
   ```
   $ impact -batch ug511.cmd
   ```
8. Invoke XMD and connect to the processor by the following command:

   ```
   $ xmd -opt ug511.opt
   ```
9. Download the BlueCat Linux kernel image into DDR2 memory using one of the following commands, depending on the system.

   **PPC440 System:**
   
   ```
   XMD% dow -data ../bclinux_images/v5fxt_devl_kit_demo.kdi 0x03000000
   ```

   **MicroBlaze System:**
   
   ```
   XMD% dow -data ../bclinux_images/v5fxt_devl_kit_demo.kdi 0x90000000
   ```

   **Note:** This step may take several minutes to download the BlueCat Linux image into memory.

10. To start the kernel image running and boot BlueCat Linux, use one of the following XMD commands, depending on the system.

    **PPC440 System:**
    
    ```
    XMD% con 0x03007000
    ```

    **MicroBlaze System:**
    
    ```
    XMD% con 0x90000000
    ```
a. After BlueCat Linux boots, the HyperTerminal output will be similar to the output shown in Figure 3-1.

```c
Linux version 2.6.11.4 () (gcc version 3.4.3) #1 Thu Apr 29 11:45:05 EDT 2008
Xilinx ML07 Reference System (Virtex-5 FXT)
Built 1 timezone(s)
Kernel command line: ramdisk_size=65536 xilinx_enc_part_cont=0-25 26-45 46-125:126-254
hd=-swap hda=-swap hdb=-swap hdd=-swap root=/dev/hda
Xilinx HFC @ 0x81000000 mapped to 0x0000F000
PXD hash table entries: 2048 (order: 11, 1768 bytes)
Device cache hash table entries: 65536 (order: 6, 242144 bytes)
Indext-cache hash table entries: 22720 (order: 6, 131072 bytes)
Memory: 258888K available (17246k kernel code, 46k data, 84k init, 0k highmem)
Kernel-cache hash table entries: 512
NET: Registered protocol family 1
JFFS2 version 2.2 (MAD (C)) 2001-2003 Mad Hatter Inc.
mpg00:01 at 0x816f0000 mapped to 0x0000f000 device: 10:15 using JFFS7
mpg00:02 at 0x81700000 mapped to 0x0000f000 device: 10:16 not using JFFS7
mpg00:03 at 0x81710000 mapped to 0x0000f000 device: 10:17 not using JFFS7
mpg00:04 at 0x81720000 mapped to 0x0000f000 device: 10:18 not using JFFS7
Serial: 0200-1050 driver Revision: 1.50 (4 ports) 16k sharing enabled
ttyS0 at HMC 0x8039d493 (irq = 9) in a 16550A
io scheduler noop registered
io scheduler aLOCKS registered
io scheduler cio registered
RAMDISK driver initialized: 16 512K disks of 65536K size 1024 blocksize
xilinx.sysname: Could not lock (XST_DEVICE_BUSY), giving up
XILcloc: using DMA code
XILterm: ICR address: 0x80
XILterm: buffer descriptor size: 22728 (0x9000)
XILterm: buffer descriptor init: phys: 0xb0400000, virt: 0xc0400000, size: 0x9000
XILterm: FDB detected at address 0x80000000
eth0: Xilinx TEC at 0x00100000 mapped to 0x00108000 irq=5
EMC Flash on Xilinx board: Found 1 32-byte devices at 0x06 in 12-bit bank
Intel Sharp Extended Query Table at 0x0014
cli.csrdev: 0x0001 Suspended erase on write disabled.
Using buffer write method
Registering a 32MB EMC Flash at 0x0FC00000
EMC Flash MTU driver: Configuration of partitions is 0-25:26-45:46-125:126-254
Creating 4 MTUs partitions on EMC Flash on Xilinx board:
0x00000000-0x003f0000: 'EMC Flash on Xilinx board'
0x003f0000-0x005f0000: 'EMC Flash on Xilinx board'
0x005f0000-0x007f0000: 'EMC Flash on Xilinx board'
0x007f0000-0x009f0000: 'EMC Flash on Xilinx board'
EMC Flash MTU driver: Configured 4 partitions
xilinux.0: 0x00 at 0x00100000 mapped to 0x00100000 irq=6
NET: Registered protocol family 2
IP route cache hash table entries: 4096 (order: 2, 16384 bytes)
TCP established hash table entries: 16384 (order: 5, 331072 bytes)
TCP bind hash table entries: 16384 (order: 4, 65536 bytes)
TCP hash tables configured (established 16384 bind 16384)
TCP recv registered
TCP doc registered
NET: Registered protocol family 1
NET: Registered protocol family 17
eth0: XILterm: Options: 0x816
eth0: XILterm: allocating interrupt 11 for daa node tx.
et0: XILterm: allocating interrupt 10 for daa node rx.
et0: XILterm: speed set to 1000Kbps
eth0: XILterm: Send Threshold = 32, Receive Threshold = 2
eth0: XILterm: Send Wait bound = 255, Receive Unit bound = 255
IP-Config: Incomplete network configuration information
RAMDISK: Compressed image found at block 17440
Freeing BlueCat ESP memory: 1536K freed
VFS: Mounted root (ext2 filesystem).
Freeing unused kernel memory: 94K init
Init: version 2.2.0 booting
eth0: XILterm: Options: 0x968
eth0: XILterm: allocating interrupt 11 for daa node tx.
et0: XILterm: allocating interrupt 10 for daa node rx.
et0: XILterm: speed set to 1000Kbps
eth0: XILterm: Send Threshold = 32, Receive Threshold = 2
eth0: XILterm: Send Wait bound = 255, Receive Unit bound = 255
Starting apache HTTP server
Running the DRF client
Init: Entering runlevel: 1
xshostname login: root
```

Figure 3-1: BlueCat Linux Boot Output - Ramdisk

b. Log into BlueCat Linux by using the username root.

c. For example commands to run in BlueCat Linux, see the section “Executing BlueCat Linux Commands”.

---

Virtex-5 FXT Kit Reference Systems
UG511 (v1.2.1) July 30, 2009
www.xilinx.com
Executing the BlueCat Linux Image from XPS

To execute the reference system using XPS, follow these steps:

1. Connect the Platform USB cable or the Parallel IV JTAG cable between the host computer and the Virtex-5 FXT ML507 development board.
2. Connect the serial cable between the host computer and the RS232 port on the Virtex-5 FXT ML507 development board.
3. Apply power to the Virtex-5 FXT ML507 development board.
4. Start a HyperTerminal (or similar) session on the host computer. Select the COM port corresponding to the connected serial port on the host computer. Set the Baud Rate to 115200, Data bits to 8 bits, Parity to None, Stop bits to 1 bit, and Flow control to None.
5. Open the reference system project in XPS.
6. Implement the hardware design and create the hardware bitstream by selecting Hardware → Generate Bitstream in XPS.
7. Download the bitstream to the board by selecting Device Configuration → Download Bitstream in XPS.
8. Select Debug → Launch XMD... to launch an XMD command window.
9. In XMD, download the BlueCat Linux kernel image into DDR2 memory using one of the following commands, depending on the system.
   - PPC440 System:
     
     ```
     XMD% dow -data bclinux_images/v5fxt_devl_kit_demo.kdi 0x03000000
     ```
   - MicroBlaze System:

     ```
     XMD% dow -data bclinux_images/v5fxt_devl_kit_demo.kdi 0x90000000
     ```

   **Note:** This step may take several minutes to download the BlueCat Linux image into memory.
10. To start the kernel image running and boot BlueCat Linux, use one of the following XMD commands, depending on the system.
   - PPC440 System:
     
     ```
     XMD% con 0x03007000
     ```
   - MicroBlaze System:

     ```
     XMD% con 0x90000000
     ```

     a. After BlueCat Linux boots, the HyperTerminal output will be as shown in Figure 3-1.
     b. Log into BlueCat Linux by using the username root.
     c. For example commands to run in BlueCat Linux, see the section “Executing BlueCat Linux Commands”.

Executing the BlueCat Linux Image with a JFFS2 File System

To boot the BlueCat Linux image, the hardware bitstream must be programmed to the Virtex-5 FXT device, the BlueCat Linux kernel image must be downloaded to the DDR2 memory, and the root file system must be written to the parallel Flash. Programming the bitstream can be done by either downloading the pre-built bitstream from the ready_for_download directory or generating and downloading it from XPS.
BlueCat Linux kernel image is downloaded from the bclinux_images directory. The root file system is found in the bclinux_images directory and can be programmed to the Flash device in XPS after the blocks of Flash that hold the file system have been erased.

**Executing the BlueCat Linux Image Using the Pre-Built Bitstream**

To execute the reference system using the files inside the ready_for_download directory in the system root directory, follow these steps:

1. Connect the Platform USB cable or the Parallel IV JTAG cable between the host computer and the Virtex-5 FXT ML507 development board.
2. Connect the serial cable between the host computer and the RS232 port on the Virtex-5 FXT ML507 development board.
3. Apply power to the Virtex-5 FXT ML507 development board.
4. Start a HyperTerminal (or similar) session on the host computer. Select the COM port corresponding to the connected serial port on the host computer. Set the Baud Rate to **115200**, Data bits to **8** bits, Parity to **None**, Stop bits to **1** bit, and Flow control to **None**.
5. Through XPS, launch an EDK shell by selecting **Project → Launch EDK Shell**.
6. In the EDK shell, change directories to the ready_for_download directory.
7. Use iMPACT to download the bitstream by using the following command:
   ```
   $ impact -batch ug511.cmd
   ```
8. Invoke XMD and connect to the processor by the following command:
   ```
   $ xmd -opt ug511.opt
   ```
9. Download the FlashRWE software application into BRAM using the following command:
   ```
   XMD% dow flashrwe_executable.elf
   ```
10. To start the FlashRWE software application running, use the following XMD command:
    ```
    XMD% run
    ```
    After the FlashRWE software application runs, the HyperTerminal will display the Main Menu.
11. With the FlashRWE program, erase blocks 46-125 of Flash. These blocks are the location that the BlueCat Linux kernel image will expect the JFFS2 root file system. To erase the blocks, perform the following steps, as shown in **Figure 3-2**.
    a. Enter 3 at the Main Menu.
    b. In the Flash Erase Menu, enter 2.
    c. Enter 46 as the starting block.
d. Enter 125 as the ending block.

```
Main Menu:
1 - Read Flash Contents
2 - Write to Flash
3 - Erase Flash
4 - Exit the Flash Program
Enter selection: 3

Flash Erase Menu:
1 - Erase Bytes of Flash
2 - Erase Blocks of Flash
3 - Erase the Entire Flash
4 - Exit to Main Menu
Enter selection: 2
Enter starting block (0-255) to erase: 46
Enter ending block (46-255) to erase: 125
Erasing blocks 46-125 of Flash...
Erased the Flash memory contents successfully
```

Figure 3-2: Erase Blocks 46-125 of the Flash

12. In XMD, stop and reset the processor. Then, exit XMD.
   XMD% stop
   XMD% rst
   XMD% exit

13. In XPS, select Device Configuration → Program Flash Memory.

14. In the Program Flash Memory dialog box, choose the file to program to be
   /bclinux_images/v5fxt_devl_kit_demo_flash.jffs2. Enter the offset to be
0x005C0000. The external DDR2 memory is set as the Scratch Memory. The Program Flash Memory settings are shown in Figure 3-3.

Figure 3-3: Program Flash Memory Box for the JFFS2 File System
15. In an EDK shell in the ready_for_download directory, invoke XMD and connect to
the processor by the following command:

$ xmd -opt ug511.opt

16. Download the BlueCat Linux kernel image that uses the JFFS2 filesystem into DDR2
memory using one of the following commands, depending on the system.

PPC440 System:

XMD% dow -data
   ../bclinux_images/v5fxt_devl_kit_demo_flash.kdi 0x03000000

MicroBlaze System:

XMD% dow -data
   ../bclinux_images/v5fxt_devl_kit_demo_flash.kdi 0x90000000

Note: This step may take several minutes to download the BlueCat Linux image into memory.

17. To start the kernel image running and boot BlueCat Linux, use one of the following
XMD commands, depending on the system.

PPC440 System:

XMD% con 0x03007000

MicroBlaze System:

XMD% con 0x90000000
a. After BlueCat Linux boots, the HyperTerminal output will be similar to the output shown in Figure 3-4.

```
Figure 3-4: BlueCat Linux Boot Output - JFFS2
```

b. Log into BlueCat Linux by using the username `root`.

c. For example commands to run in BlueCat Linux, see the section “Executing BlueCat Linux Commands”.

```
Figure 3-4: BlueCat Linux Boot Output - JFFS2
```
Executing the BlueCat Linux Image from XPS

To execute the reference system using XPS, follow these steps:

1. Connect the Platform USB cable or the Parallel IV JTAG cable between the host computer and the Virtex-5 FXT ML507 development board.
2. Connect the serial cable between the host computer and the RS232 port on the Virtex-5 FXT ML507 development board.
3. Apply power to the Virtex-5 FXT ML507 development board.
4. Start a HyperTerminal (or similar) session on the host computer. Select the COM port corresponding to the connected serial port on the host computer. Set the Baud Rate to 115200, Data bits to 8 bits, Parity to None, Stop bits to 1 bit, and Flow control to None.
5. Open the reference system project in XPS.
6. Implement the hardware design and create the hardware bitstream by selecting Hardware → Generate Bitstream in XPS.
7. Download the bitstream to the board by selecting Device Configuration → Download Bitstream in XPS.
8. Right click the FlashRWE software application project and select Build Project to create the executable file.
9. Select Debug → Launch XMD... to launch an XMD command window.
10. Download the FlashRWE software application into BRAM using the following command:

        XMD% dow FlashRWE/executable.elf

11. To start the FlashRWE software application running, use the following XMD command:

        XMD% run

After the FlashRWE software application runs, the HyperTerminal will display the Main Menu.
12. With the FlashRWE program, erase blocks 46-125 of Flash. These blocks are the location that the BlueCat Linux kernel image will expect the JFFS2 root file system. To erase the blocks, perform the following steps, as shown in Figure 3-5.
   a. Enter 3 at the Main Menu.
   b. In the Flash Erase Menu, enter 2.
   c. Enter 46 as the starting block.
   d. Enter 125 as the ending block.

```
Main Menu:
1 – Read Flash Contents
2 – Write to Flash
3 – Erase Flash
4 – Exit the Flash Program
Enter selection: 3

Flash Erase Menu:
1 – Erase Bytes of Flash
2 – Erase Blocks of Flash
3 – Erase the Entire Flash
4 – Exit to Main Menu
Enter selection: 2
Enter starting block (0–255) to erase: 46
Enter ending block (46–255) to erase: 125
Erasing blocks 46–125 of Flash...

Figure 3-5: Erase Blocks 46-125 of the Flash
```

13. In XMD, stop and reset the processor. Then, exit XMD.
   
   XMD% stop
   XMD% rst
   XMD% exit

14. In XPS, select **Device Configuration → Program Flash Memory.**

15. In the Program Flash Memory dialog box, choose the file to program to be `/bclinux_images/v5fxt_devl_kit_demo_flash.jffs2`. Enter the offset to be
0x005C0000. The external DDR2 memory is set as the Scratch Memory. The Program Flash Memory settings are shown in Figure 3-6.

16. Select **Debug** → **Launch XMD**... to launch an XMD command window.
17. Download the BlueCat Linux kernel image that uses the JFFS2 filesystem into DDR2 memory using one of the following commands, depending on the system.

**PPC440 System:**
```
XMD% dow -data
../bclinux_images/v5fxt_devl_kit_demo_flash.kdi 0x03000000
```

**MicroBlaze System:**
```
XMD% dow -data
../bclinux_images/v5fxt_devl_kit_demo_flash.kdi 0x90000000
```
Note: This step may take several minutes to download the BlueCat Linux image into memory.

18. To start the kernel image running and boot BlueCat Linux, use one of the following XMD commands, depending on the system.

PPC440 Processor System:

XMD% con 0x03007000

MicroBlaze Processor System:

XMD% con 0x90000000

a. After BlueCat Linux boots, the HyperTerminal output will be as shown in Figure 3-4.

b. Log into BlueCat Linux by using the username root.

c. For example commands to run in BlueCat Linux, see the section “Executing BlueCat Linux Commands”.

Executing BlueCat Linux Commands

The BlueCat Linux images provided with the development kit support many basic Linux commands. The list of commands and tools available to be run are found under the /bin directory.

The BlueCat Linux kernel images were built with networking support enabled, therefore the images support several network utilities when connected to a live network or connected directly to a remote computer.

The provided BlueCat Linux images include DHCP client support, and will try to retrieve an IP address during boot up. If unable to retrieve an IP address, the DHCP client will timeout and an IP address will need to be set manually to use the networking features.

To view the Ethernet configuration settings, use the command ifconfig. Example results of using this command for the eth0 (Ethernet) and lo (Local Loopback) ports are shown in Figure 3-7. In the figure, the board IP address is 192.168.0.126. The board IP address can be manually set or changed by issuing the command ifconfig eth0 IP_address.

<table>
<thead>
<tr>
<th>eth0</th>
<th>Link encap:Ethernet HWaddr 00:01:00:00:FE:44</th>
</tr>
</thead>
<tbody>
<tr>
<td>inet addr:192.168.0.126 Bcast:192.168.0.255 Mask:255.255.255.0</td>
<td></td>
</tr>
<tr>
<td>UP BROADCAST RUNNING MTU:1500 Metric:1</td>
<td></td>
</tr>
<tr>
<td>RX packets:83 errors:0 dropped:0 overruns:0 frame:0</td>
<td></td>
</tr>
<tr>
<td>TX packets:83 errors:0 dropped:0 overruns:0 carrier:0</td>
<td></td>
</tr>
<tr>
<td>collisions:0 txqueuelen:1000</td>
<td></td>
</tr>
<tr>
<td>RX bytes:22140 (21.6 KiB) TX bytes:21486 (20.9 KiB)</td>
<td></td>
</tr>
<tr>
<td>Interrupt:5</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>lo</th>
<th>Link encap:Local Loopback</th>
</tr>
</thead>
<tbody>
<tr>
<td>inet addr:127.0.0.1 Mask:255.0.0.0</td>
<td></td>
</tr>
<tr>
<td>UP LOOPBACK RUNNING MTU:16436 Metric:1</td>
<td></td>
</tr>
<tr>
<td>RX packets:8 errors:0 dropped:0 overruns:0 frame:0</td>
<td></td>
</tr>
<tr>
<td>TX packets:8 errors:0 dropped:0 overruns:0 carrier:0</td>
<td></td>
</tr>
<tr>
<td>collisions:0 txqueuelen:0</td>
<td></td>
</tr>
<tr>
<td>RX bytes:0 (0.0 b) TX bytes:0 (0.0 b)</td>
<td></td>
</tr>
</tbody>
</table>

Figure 3-7: Ethernet Configuration Settings

To ping a remote computer from the development board, the command string, ping -c 4 remote_computer_IP_address, is used to ping the remote computer 4 times.
To FTP from a networked computer to the board, issue the command `ftp board_IP_address`. Files can now be transferred back and forth via FTP.

To telnet from a networked computer to the board, issue the command `telnet board_IP_address`. All of the Linux commands can now be performed remotely as if the user was logged into the console on a HyperTerminal.

**Web Server Demonstration**

The kernel images provided with the kit include the ability to run the Apache Web server. During boot up, the Web server will begin to run. The user can view the Web page from the Web server by going to http://<board_ip_address>. The Web page that is served is shown in Figure 3-8. On the Web page, the user can interact with the LEDs and the switches on the board. To set the LEDs, enter in a one or two digit hexadecimal number, then press SetLEDs. This will display the binary equivalent of the number on the LEDs. The Web page will also display the value of the DIP switches. Change the DIP switches on the board, then press the ReadSwitches button to update the Web page with the new switches value in binary.

![Web Page Displayed by the Web Server](image)

**Figure 3-8:** Web Page Displayed by the Web Server
Building the BlueCat Linux Kernel Image

This section briefly describes the process for rebuilding the kernel image that is included with this reference system. To rebuild the kernel, the BlueCat Linux distribution must be obtained from LynuxWorks. For more information on the LynuxWorks BlueCat Linux distribution, see the BlueCat Linux User’s Guide for Release 5.4.

Building the BlueCat Linux kernel image for the PowerPC 440 processor system requires the BlueCat Linux core components for the PPC and the ML507 Board Support Package (BSP). For more information on the ML507 BSP, see the BlueCat Linux Board Support Guide for Xilinx ML507 Virtex-5 FXT Boards.

Building the BlueCat Linux kernel image for the MicroBlaze processor system requires the BlueCat Linux core components for the MicroBlaze processor and the Spartan-3E BSP. For more information on the Spartan-3E BSP, see the BlueCat Linux Board Support Guide for Xilinx Spartan-3E 1600E Boards.

The BlueCat Linux User’s Guide, the BlueCat Linux Board Support Guide for Xilinx ML507 Virtex-5 FXT Boards, and the BlueCat Linux Board Support Guide for Xilinx Spartan-3E 1600E Boards can be obtained from LynuxWorks at:

http://www.lynuxworks.com/support/bluecat/docs.php3

These steps assume the kernel is being built on a host system running Red Hat Enterprise Linux 4.0. All of the Linux commands must be run using a bash shell.

Installing the BlueCat Linux Distribution

These steps describe how to install the BlueCat Linux core components and the BSP. For more information on the directory structures of the LynuxWorks BlueCat Linux distribution and the installation procedures, see the BlueCat Linux User’s Guide referenced above.

1. To install the BlueCat Linux core components on the host machine, follow the steps outlined in the “Installing the Default Configuration” section in the Introduction and Installation chapter of the BlueCat Linux User’s Guide.

2. To install the BSP on the host machine, follow the steps outlined in the “Installing Target Board Support” section in the Introduction and Installation chapter of the BlueCat Linux User’s Guide.

   Note: When running the commands in these steps, bsp = ml507 for the PPC440 processor system or bsp = sp3e for the MicroBlaze processor system.

3. After the BSP is installed, support for it must be activated in the bash shell. To activate the BSP, follow the steps in the “Activating Support for a Target Board” section in the Introduction and Installation chapter of the BlueCat Linux User’s Guide.

Using the Provided Demo Directories

BlueCat Linux demo directories are provided with the reference systems which will allow the user to rebuild the BlueCat Linux images that are included. These demo directories can be found in <project root directory>/bclinux_demo/. These directories should be unzipped and placed in the BlueCat Linux distribution in $BLUECAT_PREFIX/demo/.

The provided demo directories can be built similar to the LynuxWorks BlueCat Linux demos in the BlueCat Linux development environment. In each EDK project, there are two zipped demo directories. The v5fxt_devl_kit_demo.zip file is the demo directory to recreate the BlueCat Linux image that boots with a ramdisk file system. The
v5fxt_dev1_kit_demo_flash.zip file is the demo directory to recreate the BlueCat Linux image that boots with a JFFS2 file system.

Getting the MLD File Set

The MLD file set is included in the project directory at <project root directory>/bsp/linux_bc54_v1_00_a/. This MLD file set is for building BlueCat Linux images on a Linux host computer.

Generating the BSP

With the use of the BlueCat Linux MLD, XPS can update the BlueCat Linux kernel source tree to match a specific hardware configuration. Follow these steps to generate the BSP and update the BlueCat Linux kernel source tree.

1. Open the reference system in XPS.
2. Select Software → Software Platform Settings... under XPS.
3. In the Software Platform Settings window, select linux_bc54 in the OS field, as shown in Figure 3-9.

4. Select the OS and Libraries option on the left of the Software Platform Settings window. Fill in the fields as follows:

   BLUECAT_PREFIX:
   
   <BlueCat_Linux_install_point>/usr/src/linux

   KERNEL_CONFIG:
   
   Ramdisk file system:

   Figure 3-9: Select BlueCat Linux for the OS
Building the BlueCat Linux Kernel Image

<BlueCat_Linux_install_point>/demo/v5fxt_devl_kit_demo/v5fxt_devl_kit_demo.config

Flash file system:
<BlueCat_Linux_install_point>/demo/v5fxt_devl_kit_demo_flash/v5fxt_devl_kit_demo_flash.config

An example showing these fields for the ramdisk file system is in Figure 3-10.

5. Click OK to save the changes and close the Software Platform Settings window.
6. In XPS, select Software → Generate Libraries and BSPs. This will update the BlueCat Linux kernel source tree.

Rebuilding the Kernel Image

This is the final step to create a bootable BlueCat Linux kernel image. To recreate the image provided with this reference system, follow these steps:

1. Two patches are provided with the reference systems. The cfi_patch allows the Flash memory to be used to hold a file system without chip errors. The xlltemac_patch makes the xlltemac driver compatible with v2.00.a of the XPS LL TEMAC core. The patches are located in the /bclinux_demo directory in the project. The patch files must be copied to the BlueCat Linux installation location, $BLUECAT_PREFIX, then the patches can be applied.

   BlueCat:$ cd $BLUECAT_PREFIX
   BlueCat:$ cp <project root directory/bclinux_demo/cfi_patch .

   X-Ref Target - Figure 3-10
   Figure 3-10: Set the BlueCat Linux Paths
Chapter 3: LynuxWorks BlueCat Linux

2. To force all kernel components to rebuild, clean the kernel tree using the following commands:

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

3. Navigate to the appropriate demo directory.
   Ramdisk file system:
   ```
   BlueCat:$ cd $BLUECAT_PREFIX/demo/v5fxt_devl_kit_demo
   ```
   Flash file system:
   ```
   BlueCat:$ cd $BLUECAT_PREFIX/demo/v5fxt_devl_kit_demo_flash
   ```

4. Run the following command to see the menu for the Linux kernel configuration:

```
BlueCat:$ make menuconfig
```

5. To rebuild the Linux image that uses the JFFS2 Flash file system, ensure that the kernel configuration has enabled support for JFFS2. Navigate the menu to `File Systems → Miscellaneous filesystems`. Select the menu item to include Journalling Flash File System v2 (JFFS2) support. Under the main menu, navigate to `Device Drivers → Memory Technology Devices (MTD)`. Select the menu item to include Memory Technology Device (MTD) support. Under MTD support, select the menu item to include MTD partitioning support.

6. Under the Linux kernel configuration menu, make sure that support is enabled for `General setup → System V IPC`. The Apache Web server requires this support.

7. Exit the Linux kernel configuration menu, saving the new configuration if changes were made.

8. Clean any prebuilt image files.

```
BlueCat:$ make clean
```


```
BlueCat:$ make all
```

This command produces a .kdi file which is the BlueCat Linux image and is composed of a compressed kernel image and a compressed RAM disk root file system. The image will be stored in one of the following locations, depending on which demo was built.

Ramdisk file system:
```
$BLUECAT_PREFIX/demo/v5fxt_devl_kit_demo/v5fxt_devl_kit_demo.kdi
```

Flash file system:
```
$BLUECAT_PREFIX/demo/v5fxt_devl_kit_demo_flash/v5fxt_devl_kit_demo_flash.kdi
```

10. To run the newly created kernel image, refer to the steps in the section “Executing the BlueCat Linux Images”. When downloading the kernel image through XMD into DDR
Booting the BlueCat Linux Image from Parallel Flash

To boot the BlueCat Linux image from parallel Flash, the Linux image, root filesystem, and a bootloader application must be programmed into Flash. The bootloader application copies the Linux image from Flash to DDR2 memory and boots BlueCat Linux. This section details the steps to program the BlueCat Linux image into parallel Flash. Flash files that have already been generated are provided, or the user can create new files for programming the Flash.

After programming all the files into the Flash memory as detailed in the steps in this section, the Flash memory will have the address map shown in Table 3-1. The files do not necessarily take the entire space assigned. Each file starts on a block boundary so that each can be erased individually without affecting the other files in the device.

**Note:** Before starting these steps, make sure that the standalone OS is chosen under Software Platform Settings. If changing the OS to standalone, make sure to set stdout and stdin in the OS and Libraries settings.

**Table 3-1: Parallel Flash Address Mapping**

<table>
<thead>
<tr>
<th>File</th>
<th>Start Address Offset</th>
<th>End Address Offset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bootloader/System Configuration (BIN)</td>
<td>0x00000000</td>
<td>0x0033FFFF</td>
</tr>
<tr>
<td>BlueCat Linux Image (KDI)</td>
<td>0x00340000</td>
<td>0x005BFFFF</td>
</tr>
<tr>
<td>Flash File System (JFFS2)</td>
<td>0x005C0000</td>
<td>0x00FBFFFF</td>
</tr>
<tr>
<td>Unused Space</td>
<td>0x00FC0000</td>
<td>0x01FFFFFFFFFF</td>
</tr>
</tbody>
</table>

Programming the Flash with the Provided Files

Flash files that have already been generated and are ready to use can be found in the `<project root directory>/ready_for_download/flash_files/` directory. A bootloader, `bootloader_bclinux`, is also provided in the reference system for bootloading the BlueCat Linux image.

**Programming the KDI File**

1. Open the reference system project in XPS.
2. Connect the USB or Parallel IV programming cable, the serial cable, and the power supply to the ML507 V5FXT development board. Power on the board.
3. The target board must be configured with the project bitstream before XPS can program the Flash memory. In an EDK shell, change directories to the `ready_for_download` directory. Use iMPACT to download the bitstream by using the following command:
   
   $ impact -batch ug511.cmd

4. In XPS, select **Device Configuration → Program Flash Memory**.
5. In the Program Flash Memory dialog box, choose the file to program to be `/bclinux_images/v5fxt_dev1_kit_demo_flash.kdl` under the project root directory. Change the program offset to 0x00340000, which will allow enough room for the bootloader and will not overwrite the location that BlueCat Linux expects the Flash file system. The external DDR2 memory is set as the Scratch Memory. The Program Flash Memory settings are shown in Figure 3-11.

**Note:** The bootloader provided in the `bootloader_bclinux` directory assumes the image has been programmed at an offset of 0x00340000.

![Program Flash Memory Dialog Box](UG511_03_16_121208)

**Figure 3-11:** Program Flash Memory Dialog Box for BlueCat Linux Image

6. Click the OK button. This will program the flash memory with the BlueCat Linux image.

**Note:** The Program Flash Memory application will take a long time as the application must program the image of size approximately 2 MB.
Programming the BIN File

The next step is to program the BIN file for the system. The BIN file is used to program the Flash with the system bitstream and bootloader loaded in the bitstream.

1. From the XPS menu, select Device Configuration → Program Flash Memory.
2. In the Program Flash Memory dialog box, choose the file to program to be bootloader_bclinux.bin under the ready_for_download/flash_files directory of the project. Change the program offset to 0x00000000. The external DDR2 memory is set as the Scratch Memory. The Program Flash Memory settings are shown in Figure 3-12.

3. Click the OK button. This will program the flash memory with the bitstream and bootloader loaded in the bitstream.

Figure 3-12: Program Flash Memory Box for the Bootloader and Bitstream BIN File
Chapter 3: LynuxWorks BlueCat Linux

Note: The Program Flash Memory application will take a long time as the application must program the BIN file of size approximately 3 MB.

Programming the Root File System

1. In an EDK shell, change directories to the ready_for_download directory.
2. Invoke XMD and connect to the processor by the following command:
   `$ xmd -opt ug511.opt`
3. Download the FlashRWE software application into BRAM using the following command:
   `XMD% dow flashrwe_executable.elf`
4. To start the FlashRWE software application running, use the following XMD command:
   `XMD% run`
   After the FlashRWE software application runs, the HyperTerminal will display the Main Menu.
5. With the FlashRWE program, erase blocks 46-125 of Flash. These blocks are the location that the BlueCat Linux kernel image will expect the JFFS2 root file system. To erase the blocks, perform the following steps, as shown in Figure 3-13.
   a. Enter 3 at the Main Menu.
   b. In the Flash Erase Menu, enter 2.
   c. Enter 46 as the starting block.
   d. Enter 125 as the ending block.

6. In XMD, stop and reset the processor. Then, exit XMD.
   `XMD% stop`
   `XMD% rst`
   `XMD% exit`

---

Main Menu:
1 - Read Flash Contents
2 - Write to Flash
3 - Erase Flash
4 - Exit the Flash Program
Enter selection: 3

Flash Erase Menu:
1 - Erase Bytes of Flash
2 - Erase Blocks of Flash
3 - Erase the Entire Flash
4 - Exit to Main Menu
Enter selection: 2
Enter starting block (0-255) to erase: 46
Enter ending block (46-255) to erase: 125
Erasing blocks 46-125 of Flash...

..........................
Erased the Flash memory contents successfully

Figure 3-13: Erase Blocks 46-125 of the Flash
7. In XPS, select **Device Configuration → Program Flash Memory**.

8. In the Program Flash Memory dialog box, choose the file to program to be `<project root directory>/bclinux_images/v5fxt_devl_kit_demo_flash.jffs2`. Enter the offset to be `0x005C0000`. The external DDR2 memory is set as the Scratch Memory. The Program Flash Memory settings are shown in **Figure 3-14**.

![Program Flash Memory](image)

**Figure 3-14**: Program Flash Memory Box for the JFFS2 File System

9. Click the OK button. This will program the flash memory with the JFFS2 file.

**Note**: The Program Flash Memory application will take a long time as the application must program the JFFS2 file of size approximately 7 MB.

### Running the Design

1. Once the Flash is programmed with all the files documented in the previous steps, change the configuration address / mode DIP switches to **00001001**.
2. Press the Prog button. The design takes about 10 seconds to finish loading and begin to run.

Generating New Flash Files and Programming the Flash

Instead of using the pregenerated files, the user can generate new files for programming the Flash device. This section details the steps for creating new Flash files and programming them into the Flash device.

Programming the KDI File and Creating the Bootloader

1. Open the reference system project in XPS.
2. Connect the USB or Parallel IV programming cable, the serial cable, and the power supply to the ML507 V5FXT development board. Power on the board.
3. The target board must be configured with the project bitstream before XPS can program the SREC file into the flash memory. Select Device Configuration → Download Bitstream in XPS.
4. In XPS, select Device Configuration → Program Flash Memory. In the Program Flash Memory dialog box, choose the file to program to be /bclinux_images/v5fxt_devl_kit_demo_flash.kdi under the project root directory. Change the program offset to 0x00340000, which will allow enough room for the bootloader and will not overwrite the location that BlueCat Linux expects the Flash file system.
The external DDR2 memory is set as the Scratch Memory. A bootloader is created by clicking the Create Flash Bootloader Application check box in the Program Flash Memory dialog box. The Program Flash Memory settings are shown in Figure 3-15.

**Note:** The bootloader provided in the bootloader_bclinux directory assumes the image has been programmed at an offset of 0x00340000.

5. Click the OK button. This will program the flash memory with the BlueCat Linux image and will create a bootloader software application project.

**Note:** The Program Flash Memory application will take a long time as the application must program the image of size approximately 2 MB.

6. After creating the bootloader files, add the following lines, shown in Figure 3-16, in the bootloader.c file:

```
#include "xparameters.h"
#include "xuartns550_1.h"
```

**Figure 3-15:** Program Flash Memory Dialog Box for BlueCat Linux Image
Chapter 3: LynuxWorks BlueCat Linux

7. It is strongly suggested to disable the bootloader from displaying its progress. This is done by commenting out the following line in the `bootloader.c` generated file:

```c
#define VERBOSE
```

The line to comment out to allow non-verbose bootloading is shown in Figure 3-16.

```
/* define VERBOSE */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <kconfig.h>
#include "portab.h"
#include "errors.h"
#include "srec.h"
#include "xparameters.h"
#include "xuartns550_i.h"

/* define GW */

/* Comment the following line, if you want a smaller and faster bootloader which will be silent */
/* define VERBOSE */
```

**Figure 3-16: Code to Select Non-Verbose Bootloading**

8. The bootloader must be modified to copy the KDI image from Flash into DDR2 for the BlueCat Linux demonstration to fully operate. This is done by adding lines of code to the `bootloader.c` file that the EDK generates. To modify the newly created bootloader, add the following pieces of code:

a. Code is required to define the location in Flash where the KDI image resides and the location in DDR to put the KDI image. This code is shown in Figure 3-17. The KDI_FLASH_LOC parameter should be set to the location in Flash memory of where the KDI image will be placed. The KDI_DDR_LOC parameter should be set to the location in DDR2 memory where the KDI image is to be copied. The KDI_LENGTH parameter should be set to the length of the KDI image in bytes.

   **Note:** Figure 3-17 shows the code for the PPC440 system. Some of the parameters have different values for the MicroBlaze processor system and can be seen in the example code provided for the MicroBlaze processor system.

```
/* Add for copying the contents of flash to DDR */
#define KDI_DDR_LOC XPAR_DDR2_SDRAM_BASEADDR + 0x01000000  //Location to put KDI into the DDR
#define KDI_FLASH_LOC XPAR_FLASH_MEM0_BASEADDR + 0x00020000 //Location of the KDI in FLASH
#define KDI_LENGTH 0x01000000 //Size (in bytes) of the KDI with the Flash file system

/* Declarations for copying the contents from flash to DDR */

uint8_t *kdi_ddr_ptr;
uint8_t *kdi_flash_ptr;
static uint32_t ddr_loc = KDI_DDR_LOC;
static uint32_t flash_loc = KDI_FLASH_LOC;
```

**Figure 3-17: BlueCat Linux Bootloader Code Definitions and Declarations**

b. Comment the SREC function `load_exec` in the source file as the KDI image is loaded as it is to the FLASH memory and so these functions are not required. Code is also required for the bootloader to copy the KDI image from flash into DDR2 when the bootloader runs. This code is shown in Figure 3-18. After modifying the C file, be sure to recompile the software application.

   **Note:** Figure 3-18 shows the code for the PPC440 system. The MicroBlaze processor system has a different address to execute from and therefore the code is slightly different.
Booting the BlueCat Linux Image from Parallel Flash

The bootloader code for the MicroBlaze processor can be found in the MicroBlaze processor reference system.

```c
/* srinfo sr_data = sr_data_buf;
while (length)
    if ((ret = flash_get_sec_line (sr_buf)) != 0)
        return ret;
    if ((ret = decode_sec_line (sr_buf, &srinfo)) != 0)
        return ret;
#endif
display_progress (sec_line);
switch (srinfo_type) {
    case SEC_TYPE_0:
        break;
    case SEC_TYPE_1:
    case SEC_TYPE_2:
    case SEC_TYPE_3:
        memcpy ((void*)srinfo.addr. (void*)srinfo.sr_data. srinfo.dlen):
        break;
    case SEC_TYPE_5:
    case SEC_TYPE_7:
    case SEC_TYPE_8:
    case SEC_TYPE_9:
        laddr = (void*)srinfo.addr;
        done = 1;
    ret = 0;
    break;
}
/* Copy the KDI from FLASH into the DDR */
kd3_ddr_ptr = (*uint8_t*)ddr_loc;
kd3_flash_ptr = (*uint64_t*)flash_loc;
memcpy (kd3_ddr_ptr, kd3_flash_ptr, KDI_LENGTH);
addr = (void*) (ddr_loc+0x7000);
#endif
print ("Executing program on address: ");
putnum ((uint32_t) addr);
print ("\n");
#endif
(laddr);
```

Figure 3-18: BlueCat Linux Bootloader Code to Copy the KDI Image

9. In XPS, compile a bitstream download.bit, that includes the system configuration and the bootloader_bclinux application. This is done by marking the bootloader application to Initialize BRAM’s, then selecting Device Configuration → Update Bitstream in XPS.

Creating and Programming the BIN File

Once the bootloader application is created, the next step is to create the BIN file for the system. The BIN file is used to program the Flash with the system bitstream and bootloader loaded in the bitstream.

1. In XPS, open an EDK shell.
2. Create a BIN file to program the Flash device by issuing the following command in the project root directory:
   ```bash
   $ promgen -w -p bin -o bootloader_bclinux.bin -u 0 implementation/download.bit
   ```
3. From the XPS menu, select Device Configuration → Program Flash Memory.
4. In the Program Flash Memory dialog box, choose the file to program to be `<project root directory>/bootloader_bclinux.bin`. Change the program offset to `0x00000000`. The external DDR2 memory is set as the Scratch Memory. The Program Flash Memory settings are shown in Figure 3-19.

![Program Flash Memory](image)

**Figure 3-19: Program Flash Memory Box for the Bootloader and Bitstream BIN File**

5. Click the OK button. This will program the flash memory with the bitstream and bootloader loaded in the bitstream.

**Note:** The Program Flash Memory application will take a long time as the application must program the BIN file of size approximately 3 MB.

**Programming the Root File System**

1. In an EDK shell, change directories to the `ready_for_download` directory.
2. Invoke XMD and connect to the processor by the following command:
Booting the BlueCat Linux Image from Parallel Flash

3. Download the FlashRWE software application into BRAM using the following command:
   $ xmd -opt ug511.opt

4. To start the FlashRWE software application running, use the following XMD command:
   XMD% run

   After the FlashRWE software application runs, the HyperTerminal will display the Main Menu.

5. With the FlashRWE program, erase blocks 46-125 of Flash. These blocks are the location that the BlueCat Linux kernel image will expect the JFFS2 root file system. To erase the blocks, perform the following steps, as shown in Figure 3-20.
   a. Enter 3 at the Main Menu.
   b. In the Flash Erase Menu, enter 2.
   c. Enter 46 as the starting block.
   d. Enter 125 as the ending block.

   | Main Menu: |
   | 1 - Read Flash Contents |
   | 2 - Write to Flash |
   | 3 - Erase Flash |
   | 4 - Exit the Flash Program |
   | Enter selection: 3 |

   | Flash Erase Menu: |
   | 1 - Erase Bytes of Flash |
   | 2 - Erase Blocks of Flash |
   | 3 - Erase the Entire Flash |
   | 4 - Exit to Main Menu |
   | Enter selection: 2 |
   | Enter starting block (0-255) to erase: 46 |
   | Enter ending block (46-255) to erase: 125 |
   | Erasing blocks 46-125 of Flash... |
   | Erased the Flash memory contents successfully |

   Figure 3-20: Erase Blocks 46-125 of the Flash

6. In XMD, stop and reset the processor. Then, exit XMD.
   XMD% stop
   XMD% rst
   XMD% exit

7. In XPS, select Device Configuration → Program Flash Memory.

8. In the Program Flash Memory dialog box, choose the file to program to be /bclinux_images/v5fxt_devl_kit_demo_flash.jffs2. Enter the offset to be
0x005C0000. The external DDR2 memory is set as the Scratch Memory. The Program Flash Memory settings are shown in Figure 3-21.

9. Click the OK button. This will program the flash memory with the JFFS2 file.

**Note:** The Program Flash Memory application will take a long time as the application must program the JFFS2 file of size approximately 7 MB.

Running the Design

1. Once the Flash is programmed with all the files documented in the previous steps, change the configuration address / mode DIP switches to **00001001**.
2. Press the **Prog** button. The design takes about 10 seconds to finish loading and begin to run.
FlashRWE Software Application

Introduction

The FlashRWE software application provides functions to read, write, and erase, the parallel Flash device. When the application is run, it will print a menu with options to enter the read, write, or erase menus. The FlashRWE application uses the xilflash library. The FlashRWE software application can run on either the MicroBlaze processor or the PowerPC 440 processor reference system.

To download and run the FlashRWE software application:

- Download the executable file directly into BRAM using a debugger, such as XMD, which is provided as part of the EDK tools. The process is described in the section “Executing the FlashRWE Software Application”.

Note: A warning box will appear during some of the steps in this chapter. The warning box states that “Software development features in XPS are deprecated, and will be removed in the next major release”. Click OK to safely ignore this warning. To turn off this warning completely, navigate to Edit → Preferences in XPS. Select Application Preferences and check the box that states “Do not show “Software Features Deprecated” dialog box”.

 Executing the FlashRWE Software Application

To execute the FlashRWE software application, program the hardware bitstream to the Virtex-5 FX device and load the FlashRWE software application into BRAM. Program the bitstream by downloading the pre-built bitstream from the ready_for_download directory or generate and download it from XPS. Similarly, the FlashRWE executable can be downloaded from the ready_for_download directory or built and downloaded through XPS.

Executing the FlashRWE Application Using the Pre-Built Bitstream

To execute the application using the files in the ready_for_download directory in the project root directory, follow these steps:

1. Connect the Platform USB cable or the Parallel IV JTAG cable between the host computer and the Virtex-5 FX70T ML507 development board.
2. Connect the serial cable between the host computer and the RS232 port on the Virtex-5 FX70T ML507 development board.
3. Apply power to the Virtex-5 FX70T ML507 development board.
4. Start a HyperTerminal (or similar) session on the host computer with the settings shown in Figure 4-1. Select the COM port corresponding to the connected serial port on the host computer. Set the Baud Rate to 115200, Data bits to 8 bits, Parity to None, Stop bits to 1 bit, and Flow control to None.

![HyperTerminal Settings](UG511_04_01_121208)

**Figure 4-1: HyperTerminal Settings**

5. Through XPS, launch an EDK shell by selecting **Project → Launch EDK Shell.**

6. In the EDK shell, change directories to the `ready_for_download` directory in either the MicroBlaze processor or the PowerPC processor reference system.

7. Use iMPACT to download the bitstream by using the following command:

   ```bash
   $ impact -batch ug511.cmd
   ```

8. Invoke XMD and connect to the processor by using the following command:

   ```bash
   $ xmd -opt ug511.opt
   ```

9. Download the FlashRWE software application into BRAM by using the following command:

   ```bash
   XMD% dow flashrwe_executable.elf
   ```
10. Start the FlashRWE software application running by using the following XMD command:
   XMD% run
   a. After the FlashRWE software application runs, the HyperTerminal will display the Main Menu, as shown in Figure 4-2, page 55.
   b. For an explanation of the available functionality in the application, see the section “Functions in the FlashRWE Software Application”.

**Executing the FlashRWE Software Application from XPS**

To execute the reference system using XPS, follow these steps:

1. Perform steps 1-4 in the “Executing the FlashRWE Application Using the Pre-Built Bitstream” section.
2. Open either the MicroBlaze processor or the PowerPC 440 processor reference system project in XPS.
3. Implement the hardware design and create the hardware bitstream by selecting **Hardware → Generate Bitstream** in XPS.
4. In the Applications tab, build the FlashRWE project by right-clicking on the project and selecting **Build Project**. This will create the software executable for the application.
5. Download the bitstream to the board by selecting **Device Configuration → Download Bitstream** in XPS.
6. After the bitstream has downloaded, launch the XMD by selecting **Debug → Launch XMD...** in XPS.
7. Download the FlashRWE application executable file using the following command in XMD:
   XMD% dow FlashRWE/executable.elf
8. To run the software application, use the **run** command in XMD.
   a. After the FlashRWE software application runs, the HyperTerminal will display the Main Menu, as shown in Figure 4-2, page 55.
   b. For an explanation of the available commands in the application, see the section “Functions in the FlashRWE Software Application”.

**Functions in the FlashRWE Software Application**

After the FlashRWE application is executed, the Main Menu will be printed out to the terminal application. The Main Menu is shown in Figure 4-2. The Main Menu allows the options to enter into the read, write, and erase submenus.

```
Main Menu:
1 - Read Flash Contents
2 - Write to Flash
3 - Erase Flash
4 - Exit the Flash Program
Enter selection:
```

*Figure 4-2: Main Menu*
Table 4-1 lists and describes the functions that are available in the FlashRWE application.

<table>
<thead>
<tr>
<th>Submenu</th>
<th>Function</th>
<th>Description</th>
<th>Parameters Entered by User</th>
<th>Document Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flash Read</td>
<td>Read Bytes of Flash</td>
<td>Read and print out individual bytes of data from the parallel Flash device.</td>
<td>Address offset, Number of bytes</td>
<td>“Read Bytes of Flash,” page 57</td>
</tr>
<tr>
<td></td>
<td>Check if Flash is Empty</td>
<td>Checks all bytes of data in the parallel Flash device and determines if the Flash device is fully erased.</td>
<td>None</td>
<td>“Check if Flash is Empty,” page 57</td>
</tr>
<tr>
<td></td>
<td>Exit to Main Menu</td>
<td>Exit the submenu and return to the main menu</td>
<td>None</td>
<td>N/A</td>
</tr>
<tr>
<td>Flash Write</td>
<td>Write Incrementing Numbers to Flash</td>
<td>Write incrementing numbers, 0x0-0xFF to bytes in the Flash.</td>
<td>Address offset, Number of bytes</td>
<td>“Write Incrementing Numbers to Flash,” page 59</td>
</tr>
<tr>
<td></td>
<td>Write Bytes to Flash</td>
<td>Write bytes of data to Flash</td>
<td>Address offset, Bytes of data, Number of times to write the data</td>
<td>“Write Bytes to Flash,” page 60</td>
</tr>
<tr>
<td></td>
<td>Exit to Main Menu</td>
<td>Exit the submenu and return to the main menu</td>
<td>None</td>
<td>N/A</td>
</tr>
<tr>
<td>Flash Erase</td>
<td>Erase Bytes of Flash</td>
<td>Erases the block(s) of Flash that the specified bytes reside in</td>
<td>Address offset, Number of bytes</td>
<td>“Erase Bytes of Flash,” page 61</td>
</tr>
<tr>
<td></td>
<td>Erase Blocks of Flash</td>
<td>Erases a single block or a range of consecutive blocks of Flash</td>
<td>Starting block, Ending block</td>
<td>“Erase Blocks of Flash,” page 62</td>
</tr>
<tr>
<td></td>
<td>Erase the Entire Flash</td>
<td>Erases the entire Flash device</td>
<td>None</td>
<td>“Erase the Entire Flash,” page 63</td>
</tr>
<tr>
<td></td>
<td>Exit to Main Menu</td>
<td>Exit the submenu and return to the main menu</td>
<td>None</td>
<td>N/A</td>
</tr>
<tr>
<td>None (Main menu)</td>
<td>Exit the Flash Program</td>
<td>Quits the FlashRWE program</td>
<td>None</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Flash Read Menu

The Flash Read Menu is displayed after entering 1 at the Main Menu prompt. The Flash Read Menu has options to read bytes of Flash, to check if the Flash is empty, or to exit back to the Main Menu. The Flash Read Menu is shown in Figure 4-3.

<table>
<thead>
<tr>
<th>Flash Read Menu:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 - Read Bytes of Flash</td>
</tr>
<tr>
<td>2 - Check if Flash is Empty</td>
</tr>
<tr>
<td>3 - Exit to Main Menu</td>
</tr>
</tbody>
</table>

Enter selection:

Figure 4-3: Flash Read Menu

Read Bytes of Flash

The Read Bytes of Flash function can be executed by entering 1 at the Flash Read Menu prompt. Once executed, this function reads individual bytes of data out of the Flash device and displays the address and data.

The following are the parameters the user must supply values for:

- Address offset: Offset into the Flash from which to start reading the first byte of data
- Number of bytes: Total number of bytes to read and display data

Example input and output from this function is shown in Figure 4-4. In the figure, 32 bytes of data is read starting from the address 0xFCCC0000. The example output was created while the software application was running on the PPC440 hardware system. The output from the application running on the MicroBlaze processor hardware system will be the same except the addressing will be offset from 0x8C000000 instead of 0xFC000000.

<table>
<thead>
<tr>
<th>Flash Read Menu:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 - Read Bytes of Flash</td>
</tr>
<tr>
<td>2 - Check if Flash is Empty</td>
</tr>
<tr>
<td>3 - Exit to Main Menu</td>
</tr>
</tbody>
</table>

Enter selection: 1
Enter address offset (0-1FFFFFFF): CC0000
Enter number of bytes to read (0-128): 32
0xFCCC0000: 00 01 02 03 04 05 06 07
0xFCCC0008: 08 09 0A 0B 0C 0D 0E 0F
0xFCCC0010: 10 11 12 13 FF FF FF FF
0xFCCC0018: FF FF FF FF FF FF FF

Figure 4-4: Read Bytes of Flash Example Input/Output

Check if Flash is Empty

The Check if Flash is Empty function can be executed by entering 2 at the Flash Read Menu prompt. Once executed, this function checks if the entire Flash device is completely erased and empty. This function reads all the individual bytes of data in the Flash device and compares the data to 0xFF. If all bytes of data are 0xFF, the Flash device is completely erased. Otherwise, a set of the locations that were not empty and the data in those locations is outputted to the terminal. The total number of locations found that were not empty is also outputted. This function takes approximately 8 seconds to run for the PPC440 system and 28 seconds to run for the MicroBlaze system.
Figure 4-5 shows example output from the function when the Flash is empty. Figure 4-6 shows example output from the function when the Flash is not empty. The example output was created while the software application was running on the PPC440 hardware system. The output from the application running on the MicroBlaze processor hardware system will be the same except the addressing will be offset from 0x8C000000 instead of 0xFC000000.

**Flash Read Menu:**
1 - Read Bytes of Flash
2 - Check if Flash is Empty
3 - Exit to Main Menu
Enter selection: 2
Checking if the Flash is empty...

| Flash is not erased at 0x8C100000: Data is 0 |
| Flash is not erased at 0x8C100001: Data is 1 |
| Flash is not erased at 0x8C100002: Data is 2 |

... 

| Flash is completely empty |

---

**Figure 4-5:** Check if Flash is Empty Example Output - Flash is Empty

**Figure 4-6:** Check if Flash is Empty Example Output - Flash is not Empty
Flash Write Menu

The Flash Write Menu is displayed after entering 2 at the Main Menu prompt. The Flash Write Menu has options to write incrementing numbers to Flash, write bytes to Flash, or exit back to the Main Menu. The Flash Write Menu is shown in Figure 4-7.

<table>
<thead>
<tr>
<th>Flash Write Menu:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 - Write Incrementing Numbers to Flash</td>
</tr>
<tr>
<td>2 - Write Bytes to Flash</td>
</tr>
<tr>
<td>3 - Exit to the Main Menu</td>
</tr>
<tr>
<td>Enter selection:</td>
</tr>
</tbody>
</table>

Figure 4-7: Flash Write Menu

Write Incrementing Numbers to Flash

The Write Incrementing Numbers to Flash function can be executed by entering 1 at the Flash Write Menu prompt. Once executed, this function writes incrementing numbers (0x0-0xFF) to bytes in the Flash. After reaching 0xFF, the function wraps back to 0x0 and continues writing incrementing numbers to bytes in the Flash. If there is data found in the Flash at the bytes specified for writing, the function will prompt the user to either erase the block(s) of Flash where the bytes reside or to cancel the write request.

The following are the parameters for which the user must supply values:

- Address offset: Offset into the Flash from which to start writing the first byte of data
- Number of bytes: Total number of bytes to write incrementing numbers

Figure 4-8 shows an example output from the function when the bytes to write are already empty. In Figure 4-8, 64 bytes of data is being written to address 0xFC100000. Figure 4-9 shows an example output from the function when the bytes to write have previously stored data. In Figure 4-9, 20 bytes of data were inputted to be written to address 0xFDAl00000, but 6 bytes of Flash were found to have previous data. To proceed with the write, the user must allow the entire block (0xFDAl00000-0xFDAlFFFF) to be erased. The example output was created while the software application was running on the PowerPC 440 processor hardware system. The output from the application running on the MicroBlaze processor hardware system will be the same, except that the addressing will be offset from 0x8C000000 instead of 0xFC000000.

<table>
<thead>
<tr>
<th>Flash Write Menu:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 - Write Incrementing Numbers to Flash</td>
</tr>
<tr>
<td>2 - Write Bytes to Flash</td>
</tr>
<tr>
<td>3 - Exit to the Main Menu</td>
</tr>
<tr>
<td>Enter selection: 1</td>
</tr>
<tr>
<td>Enter address offset (0-1FFFFF): 100000</td>
</tr>
<tr>
<td>Enter number of bytes to write (0-2048): 64</td>
</tr>
</tbody>
</table>

Completed writing incrementing numbers to Flash

Figure 4-8: Write Incrementing Numbers Example Input and Output - Empty Bytes
### Chapter 4: FlashRWE Software Application

#### Write Bytes to Flash

The Write Bytes to Flash function can be executed by entering 2 at the Flash Write Menu prompt. Once executed, this function writes bytes of data to Flash. If there is data found in the Flash at the bytes specified for writing, the function will prompt the user to either erase the block(s) of Flash where the bytes reside or to cancel the write request.

The following are the parameters for which the user must supply values:
- **Address offset**: Offset into the Flash from which to start writing the first byte of data
- **Bytes of data**: The actual data to write to bytes in Flash
- **Number of times**: The number of times to write the specified bytes of data

*Figure 4-10* shows example input and output from the function when the bytes to write are already empty. In *Figure 4-10*, the data string 0x21436587 is being written 3 times starting at address 0xFDA00000. *Figure 4-11* shows example input and output from the function when the bytes to write have previously stored data. In *Figure 4-11*, the data string 0xFEDCBA was input to be written 2 times to the address 0xFC100000, but 6 bytes of Flash were found to have previous data. To proceed with the write, the user must allow the entire block (0xFC100000-0xFC11FFFF) to be erased. The example output was created while the software application was running on the PowerPC 440 processor hardware system. The output from the application running on the MicroBlaze processor hardware system will be the same except that the addressing will be offset from 0x8C000000 instead of 0xFC000000.

<table>
<thead>
<tr>
<th>Flash Write Menu:</th>
<th>1 – Write Incrementing Numbers to Flash</th>
<th>2 – Write Bytes to Flash</th>
<th>3 – Exit to the Main Menu</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enter selection:</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Enter address offset (0–1FFFFFF):</td>
<td>1A000000</td>
<td>21436587</td>
<td></td>
</tr>
<tr>
<td>Enter number of bytes to write (0–2048):</td>
<td>20</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Completed writing bytes of data to Flash</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Figure 4-10*: Write Bytes Example Input and Output - Empty Bytes

<table>
<thead>
<tr>
<th>Flash Write Menu:</th>
<th>1 – Write Incrementing Numbers to Flash</th>
<th>2 – Write Bytes to Flash</th>
<th>3 – Exit to the Main Menu</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enter selection:</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Enter address offset (0–1FFFFFF):</td>
<td>1A000000</td>
<td>21436587</td>
<td></td>
</tr>
<tr>
<td>Enter number of times to write the data (0–32):</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Completed writing bytes of data to Flash</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Figure 4-10*: Write Bytes Example Input and Output - Full Bytes

*Figure 4-11*: Write Bytes Example Input and Output - Empty Bytes
Flash Write Menu
1 - Write Incrementing Numbers to Flash
100000
2 - Write Bytes to Flash
0
3 - Exit to the Main Menu
Enter selection:

Figure 4-11: Write Bytes Example Input and Output - Full Bytes

Flash Erase Menu

The Flash Erase Menu is displayed after entering 3 at the Main Menu prompt. The Flash Erase Menu has options to erase bytes of Flash, erase blocks of Flash, erase the entire Flash, or exit back to the Main Menu. The Flash Erase Menu is shown in Figure 4-12.

<table>
<thead>
<tr>
<th>Flash Erase Menu:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 - Erase Bytes of Flash</td>
</tr>
<tr>
<td>2 - Erase Blocks of Flash</td>
</tr>
<tr>
<td>3 - Erase the Entire Flash</td>
</tr>
<tr>
<td>4 - Exit to Main Menu</td>
</tr>
<tr>
<td>Enter selection:</td>
</tr>
</tbody>
</table>

Figure 4-12: Flash Erase Menu

Erase Bytes of Flash

The Erase Bytes of Flash function can be executed by entering 1 at the Flash Erase Menu prompt. Once executed, this function erases the block(s) of Flash in which the specified bytes reside.

The following are the parameters for which the user must supply values:

- Address offset: Offset into the Flash to the first byte desired to be erased
- Number of bytes: Total number of bytes to erase

Caution! Because Flash memory must be erased in blocks, the number of bytes actually erased may be larger than the inputted number of bytes to erase, depending on the address offset and number of bytes specified.
Chapter 4: FlashRWE Software Application

An example input and output from this function is shown in Figure 4-13. In the figure, it is specified to erase 2048 bytes at address 0xFC010000. However, because of Flash memory characteristics, the entire block in which these bytes reside, will be erased, namely the address range 0xFC000000-0xFC01FFFF. The example output was created while the software application was running on the PPC440 hardware system. The output from the application running on the MicroBlaze processor hardware system will be the same except that the addressing will be offset from 0x8C000000 instead of from 0xFC000000.

<table>
<thead>
<tr>
<th>Flash Erase Menu:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 - Erase Bytes of Flash</td>
</tr>
<tr>
<td>2 - Erase Blocks of Flash</td>
</tr>
<tr>
<td>3 - Erase the Entire Flash</td>
</tr>
<tr>
<td>4 - Exit to Main Menu</td>
</tr>
</tbody>
</table>

Enter selection: 1
Enter address offset (0-1FFFFFF): 10000
Enter number of bytes to erase (0-393216): 2048
Erased the Flash memory contents successfully

Figure 4-13: Erase Bytes of Flash Example Input and Output

Erase Blocks of Flash

The Erase Blocks of Flash function can be executed by entering 2 at the Flash Erase Menu prompt. Once executed, this function erases the specified range of block(s) of Flash. If the maximum number of blocks (256 blocks) is specified to be erased, this function will take approximately 4 minutes to complete.

The following are the parameters for which the user must supply values:

- Starting block: First block in the range of blocks to be erased
- Ending block: Last block in the range of blocks to be erased

An example input and output from this function is shown in Figure 4-14. In the figure, blocks 35-104 have been erased, which corresponds to the address range 0xFC46000000-0xFCD1FFFF. The example output was created while the software application was running on the PowerPC 440 processor hardware system. The output from the application running on the MicroBlaze processor hardware system will be the same except the addressing will be offset from 0x8C000000 instead of 0xFC000000.

<table>
<thead>
<tr>
<th>Flash Erase Menu:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 - Erase Bytes of Flash</td>
</tr>
<tr>
<td>2 - Erase Blocks of Flash</td>
</tr>
<tr>
<td>3 - Erase the Entire Flash</td>
</tr>
<tr>
<td>4 - Exit to Main Menu</td>
</tr>
</tbody>
</table>

Enter selection: 2
Enter starting block (0-255) to erase: 35
Enter ending block (35-255) to erase: 104
Erasing blocks 35-104 of Flash...

Erased the Flash memory contents successfully

Figure 4-14: Erase Blocks of Flash Example Input and Output
Erase the Entire Flash

The Erase the Entire Flash function can be executed by entering 3 at the Flash Erase Menu prompt. Once executed, this function erases the entire Flash device. The function will take approximately 4 minutes to complete.

Example output from this function is shown in Figure 4-15.

```
Flash Erase Menu:
1 – Erase Bytes of Flash
2 – Erase Blocks of Flash
3 – Erase the Entire Flash
4 – Exit to Main Menu
Enter selection: 3
Erasing the entire Flash...

Erased the Flash memory contents successfully
```

**Figure 4-15:** Erase the Entire Flash Example Output