PCI Express Demos for the LatticeECP3 Versa Evaluation Board

User's Guide

March 2012
UG46_01.1

Introduction

This guide describes how to start using the LatticeECP3 Versa Evaluation Board, a low-cost platform for demonstrating the PCI Express reference design for evaluating solutions for your own specific application.

This guide will familiarize you with the process of setting up your PCI Express development environment. This document assumes you do not have any associated tools installed on your system. For Linux users, corresponding installation instructions are provided in Appendix A.

The demos discussed in this document include the PCI Express Basic Demo, PCI Express Throughput Demo and PCI Express Scatter-Gather DMA Demos (EBR or DDR3 based).

Learning Objectives

After you complete the steps in this guide, you will be able to do the following:

- Set up the LatticeECP3 Versa Evaluation Board properly and become familiar with its main features.
- Install all applicable development tools and the PCI Express demonstration applications.
- Establish communication with the LatticeECP3 Versa Evaluation Board through the PCI Express link.
- Run the PCI Express Basic demo, that allows you to run the preset LED light sequence, interactively light LED segments, and familiarize yourself with other features of the software.
- Run the PCI Express Throughput demo, that allows you to see the performance of the Lattice PCI Express SERDES hardware and PCI Express Endpoint IP core in terms of maximum data rates for writes/reads to and from your system memory.
- Run the DMA demos and observe how the Scatter-Gather DMA IP core, together with the PCI Express Endpoint IP core, demonstrate data transfer between the DDR3 memory controlled by the DDR3 SDRAM Controller IP core, the Lattice FPGA and system memory using the LatticeECP3 Versa Evaluation Board.
- Use what each demo teaches you about designing Lattice PCI Express solutions.
- Become familiar with an approach that will enable you to modify and rebuild the PCI Express Basic demo for your own purposes.
- Become familiar with the software development tools and major design flow steps employed in this kit.
- Use other existing documentation in conjunction with this guide.

You can obtain more detailed information on specific board features by referring to EB62, LatticeECP3 Versa Evaluation Board User's Guide.

In addition to reading this guide, you should visit the LatticeECP3 Versa Evaluation Board web page on the Lattice web site and familiarize yourself with the set of other documents related to PCI Express.

This document assumes that you have already installed Lattice Diamond® design software and are familiar with basic tasks. If not, please refer to the Diamond Help system.

Related Documentation

In addition to using this guide to help you get started developing the PCI Express solution on your device, you can refer to other documents applicable to this guide that may contain more detailed information that is beyond the scope of this guide.

All of the following documents can be obtained on the Lattice web site:

- EB62, LatticeECP3 Versa Evaluation Board User's Guide – Describes board features, power requirements, device programming, clock management, and board schematics in detail.

- IPUG75, PCI Express 1.1 x1, x4 IP Core User's Guide – Describes the features that the x1 and x4 Endpoint IP Cores support and provides a functional description of the IPs, parameters, signals, port lists, timing diagrams, memory maps, and step-by-step procedures for creating the core in IPexpress.
- UG06, PCI Express Scatter-Gather DMA Demo Verilog Source Code User's Guide – Provides details of the Verilog code used for the Lattice PCI Express Scatter-Gather DMA demo application.
- UG07, PCI Express Throughput Demo Verilog Source Code User's Guide – Provides details of the Verilog code used for the Lattice PCI Express Throughput Demo (also known as the Stored FIFO Interface (SSIF) Demo).
- UG15, PCI Express Basic Demo Verilog Source Code User's Guide – Provides details of the Verilog code used for the Lattice PCI Express Basic demo, a block diagram of the design, and descriptions of design modules. Instructions for building the demo design in Diamond or ispLEVER are also included.
- LatticeECP3 Versa Evaluation Board web page – Visit this web page on the Lattice web site for updates to this and other related documents. You can download kit installation files from this page.

Hardware Requirements

To install the kit design and run the demo software, a single computer with a PCI Express x16, x8, x4, or x1 slot is required. You must also have a powered USB port. All of the other hardware and drivers are included in the kit.

Note: Up to 4Gb of memory (system RAM) is supported for 64-bit systems.

Software Requirements

Please be aware of the following software requirements to ensure you obtain the expected results for the procedures described in this guide:

- For Windows, the Lattice PCI Express IP demo is compatible with Microsoft Windows 7/Vista/XP/2000, or Windows Server 2003 32-bit platforms.
- For Linux, the supplied files were built and tested for Red Hat Workstation 4.0, Update 4 on a 32-bit machine; however, instructions in Appendix A are provided for building necessary drivers for any kernel version you may be running. Refer to Appendix A for Linux installation instructions.
- The Verilog HDL demo design projects in this kit are built with Diamond.
- If you are using ispVM device configuration software for your downloads, ensure you are using the latest version to ensure proper bitstream downloading results. You can obtain this software on the ispVM System Software web page.
- To develop PCI Express designs, your computer must meet minimum system requirements as described in the Diamond Installation Guide.

Installing the LatticeECP3 Versa Evaluation Board PCI Express Demos

This section provides Windows installation instructions for the LatticeECP3 PCI Express demo files and demo applications. See Appendix A for corresponding Linux kit installation instructions.

To install the LatticeECP3 Versa Evaluation Board in Windows:

1. Using a web browser, go to the LatticeECP3 Versa Evaluation Board Setup (Windows) page, and download the Windows setup.exe file: **DK-ECP3-PCIE-setup.exe**.
2. Double-click on the **DK-ECP3-PCIE-setup.exe** file.
3. If the **Install Program as Other User** dialog appears, choose to install as the current user if you have Administrator privileges or select another user with those privileges and click **OK**.
4. Click **Next** to start the installation. You must have administrative privileges to install the kit.

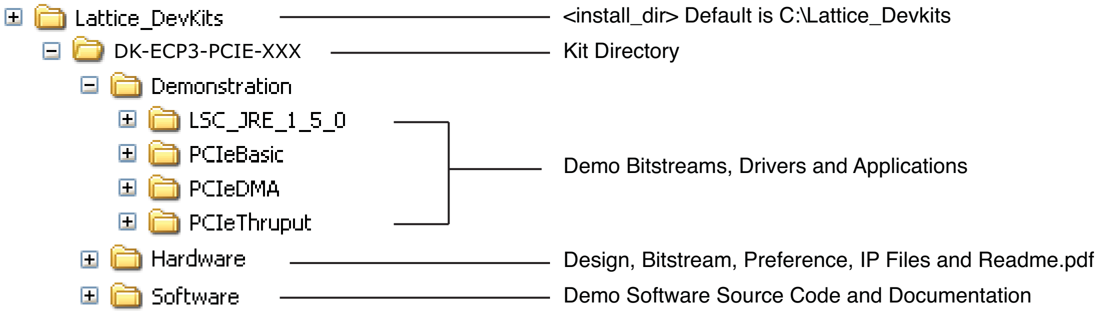
5. Click **Yes** to accept the license agreement.

6. Click **Next** to install the kit in the default **C:\Lattice_DevKits** location on your hard drive or install in the desired location by using the **Browse** button.

7. If desired, click **Yes** to accept the prompt to create shortcut desktop icons for the kit demo applications.

8. Click **Finish** to complete the kit installation. Figure 1 shows the directory structure of the installed evaluation board.

Figure 1. Installed LatticeECP3 Versa Evaluation Board Directory Structure (Windows)



Lattice_DevKits — <install_dir> Default is C:\Lattice_Devkits

DK-ECP3-PCIE-XXX — Kit Directory

Demonstration

LSC_JRE_1_5_0
PCIeBasic
PCIeDMA
PCIeThruput — Demo Bitstreams, Drivers and Applications

Hardware — Design, Bitstream, Preference, IP Files and Readme.pdf

Software — Demo Software Source Code and Documentation

Figure 1 shows the default installation path for Windows. Please note that whenever the kit directory or <kit_dir> is referred to in this document, it refers to the **<install_dir>\DK-ECP3-PCIE-XXX\** file path, where the default **<install_dir> path is C:\Lattice_Devkits** and XXX is the kit revision number.

After you install the development kit, you will also see Start Menu shortcuts for running demo applications if you opted to create them during the installation. Please see the following sections for further sequential instructions.

Hardware Installation

The procedures in this section provide step-by-step instructions for installing hardware and drivers to ensure proper board and PC communication and operation.

After board setup, you can install the hardware. You must have administrative privileges on Windows to perform this installation. See Appendix A for Linux instructions.

Caution: Lattice is not liable for any loss of data or damages that may result from the installation of the hardware and execution of the kit demo software tools. Do not install and operate on mission-critical systems.

To install the LatticeECP3 Versa Evaluation Board for Windows:

1. Shut down Windows, turn off the PC and unplug the power cord.

   IMPORTANT: This step is necessary because PC power supplies have standby voltages that are present even when the PC power light and fan are turned off. Unplugging the PC is the safest way to ensure the board will not be "hot-swapped".

2. Locate an available PCI Express slot. The board can be installed in any slot that is larger than the finger edges in use, x1, x4, or x16.

3. Ensure that the board is not connected to any external power supply before proceeding.

4. Using ESD precautions, install the LatticeECP3 Versa Evaluation Board in the PCI Express slot in the x1 position.

5. Power-on the PC and observe that it boots normally to the Windows login screen. If anything abnormal occurs, refer to Appendix B.

6. Log in as a user with administrative privileges. During the login process Windows will detect the new hardware and ask if you want to install it.

Installing Drivers

This section describes installation of the LatticeECP3 Versa Evaluation Board device driver software on a Windows PC. This procedure pertains specifically to the PCI Express Basic demo application. For the PCI Express Throughput and DMA demos, you will need to load the appropriate drivers from the respective demo folders.

Also note that this procedure describes installation on the Windows 7 and Windows XP platforms only. This may vary slightly on Windows 2000 or Windows Server 2003. See Appendix A for corresponding Linux instructions.

Note: The Found New Hardware popup dialog in Windows appears when the PC is first booted with the board installed. If this screen does not appear, the board was not properly detected by the PC BIOS or by Windows. Refer to Appendix B for more information.

To install the LatticeECP3 Versa Evaluation Board drivers on Windows 7:

1. Go to **Computer properties Device Manager**. Select the PCI Express device under the **Other Devices** tab.

2. Right-click and choose **Update Driver Software**.

3. Use the **Browse** tab to navigate to where you have installed the demo package, locate the **Demonstration\PCIeBasic** directory path on the top level of the directory, and select the **Driver** folder.

   Note: For the PCI Express Throughput and DMA demos used in this kit, you must install the appropriate drivers located in similar directory paths in the PCIeThruput and PCIeDMA folders, respectively.

4. Click **Next**. Allow the software to install by selecting **Install** on the pop-up window. Windows now copies the driver files and will display a screen indicating this. Upon completion, a capital “I” representing initialization will be displayed on the 14-segment LED.

To verify proper driver installation and device recognition on Windows:

1. Go back to **Device Manager**. The board (**LSC_PCIexpress**) should be in the list of hardware devices in your system.

2. Right-click on the Lattice evaluation board icon and select **Properties** to show the resources assigned to the device and the driver information.

Memory ranges corresponding to the configured BAR registers will be assigned to the board. If this is all present, then the demo program is able to run and access the hardware on the board.

To install the LatticeECP3 Versa Evaluation Board drivers on Windows XP:

1. In the **Found New Hardware** popup dialog, choose the **Install from specific location** option and click **Next**.

2. Use the **Browse** button to navigate to where you installed the demo package, locate the **Demonstration\PCIeBasic** directory path on the top level of the kit directory, and select the **Driver** folder.

   Note: For the PCI Express Throughput and DMA demos used in this kit, you must install the appropriate drivers located in similar directory paths in the PCIeThruput and PCIeDMA folders, respectively.

3. Click **Next**. Windows now copies the driver files and will display a screen indicating this. Upon completion, a capital “I” representing initialization will be displayed on the 14-segment LED.

To verify proper driver installation and device recognition on Windows:

1. Right click on the **My Computer** icon on your Windows desktop and choose **Properties** from the popup menu. Confirm the installation and verify that Windows properly detects the LatticeECP3 Versa Evaluation Board hardware.
2. In the System Properties dialog, choose the **Hardware** tab and click the **Device Manager** button. The board (LSC_PCIexpress) should be in the list of hardware devices in your system.
3. Right-click on the **Lattice evaluation board icon** and select **Properties** to show the resources assigned to the device and the driver information.

   Memory ranges corresponding to the configured BAR registers will be assigned to the board. If this is all present, then the demo program is able to run and access the hardware on the board.

Installing Hardware into a Different Slot

Windows identifies PCI/PCI Express hardware devices using the bus, slot, vendor ID, and device ID fields. If you install the board into a different slot, the slot number will change. This will cause Windows to display the Found New Hardware popup screen when the system powers up.

The full procedure described above for installing the driver is unnecessary since the driver has already been installed. If the board is installed in a new slot, simply choose to allow Windows to search for the driver or the Install the Software Automatically (Recommended) option and install automatically. Windows will then associate the newly created device registry tag (bus, slot, vendor and device ID) with the lscpcie.sys driver and the demo GUI will work with the board in the new slot.

Now that your board is set up and hardware is installed on your computer, you can proceed on to the next section that describes software installation, execution, and tasks to complete the demo.

Verifying Correct Board Operation

The section lists checks you should make to ensure proper functioning of the board. Also refer to related documentation on this board described in EB62, LatticeECP3 Versa Evaluation Board User’s Guide.

There are four status LED lights on the board that will go through a light sequence when the device is first powered on. To verify the PCI Express link is functioning properly, examine these indicators at the time of powering up. The PCI Express demonstration software used later in the kit verifies board operation. In addition, you can also check that the status LED lights are functioning at normal conditions in the sections below.

Note: All boards leave the manufacturer fully tested. See EB62, LatticeECP3 Versa Evaluation Board User’s Guide for details.

LED Definitions

The Status LEDs on the LatticeECP3 Versa Evaluation Board are located vertically along the left edge, middle portion of the board.

The LEDs are in the following order and have the following functions, as shown in the tables below.

Table 1. LED Order and Functionality

LED Name	LED Number	Color	Usage
U0	D29 (x1)	Blue	User LED. Off by default.
U1	D28(x1)	Blue	User LED. Off by default.
U2	D27 (x1)	Red	User LED. Off by default.
U3	D26 (x1)	Red	User LED. Off by default.

Table 1. LED Order and Functionality (Continued)

LED Name	LED Number	Color	Usage
DL_UP	D21 (x1)	Green	Data Link up, ready for packets at Transaction Layer (PCI enumeration of config registers).
L0	D22 (x1)	Green	L0 state active (training sequences completed; PHY Layer up and ready for flow control).
POLL	D24 (x1)	Yellow	Pollng state inactive.
PLL	D25 (x1)	Yellow	PLL locked to PCI Express 100MHz clock.
14-Segment Display	On board	Red	14-seg will display states with letter codes as described below: I = Initialization; driver is loaded and initializes the board O = Open; demo program is run and driver opens access to board. C = Close; demo exits and driver closes access to board. R = Remove; driver reads that board is removed from system, as in uninstalled device. E = Error; driver detects board hardware error during initialization.
14-Segment Display Decimal Point	On display in lower right	Red	Blinks to indicate PCI Express transaction is accessing registers or memory in demo IP. Indicates PCI Express bus transactions.

Table 2. Normal x1 Link Status

PLL	POLL	L0	DL_UP
Yellow	Yellow	Green	Green

Running the PCI Express Basic Demo

Once you have installed your LatticeECP3 Versa Evaluation Board in your computer and installed all necessary software, you can run the PCI Express Basic demo which consists of hardware, IP and software. This part of the document describes what you need to know to get started and successfully complete this demo.

Before You Begin

Before beginning this demo, you must do the following:

- Use ispVM to download the bitstream for this demo to the board.

  You can find the x1 bitstream and the XCF file necessary for ispVM in the **<kit_dir>\Demonstration\PCIeBasic\Bitstreams** directory path.

  For general information on ispDOWNLOAD cable and ispVM configuration software usage, see EB62, LatticeECP3 Versa Evaluation Board User's Guide.

- Install the board drivers for the application.

  You can find the driver files necessary for proper demo installation in the **<kit_dir>\Demonstration\PCIeBasic\Driver** directory path. See the procedure described in the Installing Drivers section of this document.

Resource References

Please be aware of supplementary companion documentation when using this demo.

Hardware Resources

The PCI Express Basic Demo x1 bitstream is built from the Diamond project located in the **Hardware\PCIe_x1\Versa_PCIeBasic\Implementation\ecp3-35EA_PCIeBasic** directory. The Verilog source code is located in the project Source\ directory.

The Verilog design architecture is explained in UG15, Lattice PCI Express Basic Demo Verilog Source Code User's Guide. This document describes the purpose and functionality of the Verilog modules used in PCIe Basic Demo design.

Software Resources

The PCI Express Basic demo uses the lscpcie2.sys device driver. The source code for this device driver is located in Software\lscpcie2_Win7\Driver or Software\lscpcie2_Win2kXP\drvr. The architecture of the lscpcie2 device driver is explained in the lscpcie2 Driver Reference Manual which can be accessed through the Software\PCIeDocIndex.html link.

The PCI Express Basic demo application source code is located in Software\PCIeBasic_Win2kXP\BasicGUI\DemoUI. This directory contains the Java project source code to create the user interface. The GUI also uses the PCIeAPI_Lib_Win2kXP API library. Both of these codes are common for Windows 7 and Windows XP.

The architecture of the PCIe Basic Demo application is explained in the PCIe Basic Demo Reference Manual and the PCIe API Reference Manual which can be accessed through the Software\PCIeDocIndex.html link.

Basic Demo Operations Overview

The PCI Express Basic demo shows the capabilities of the Lattice FPGA and the PCI Express Endpoint IP core functioning in a PCI Express slot in a Windows PC. The demo is easy to use and requires no test equipment.

This demo software allows you to access memory and registers on the board and provides real-time interaction with the LatticeECP3 Versa Evaluation Board hardware to demonstrate a functional PCI Express communications path between the application and driver software (running on the PC CPU) and the FPGA IP. Device driver and application source code are available so you can modify and extend the behavior of the tests or use them as a starting point for new PCIe designs.

If you experience any problems running this demo, please refer to Appendix B.

Running the PCI Express Basic Demo Software

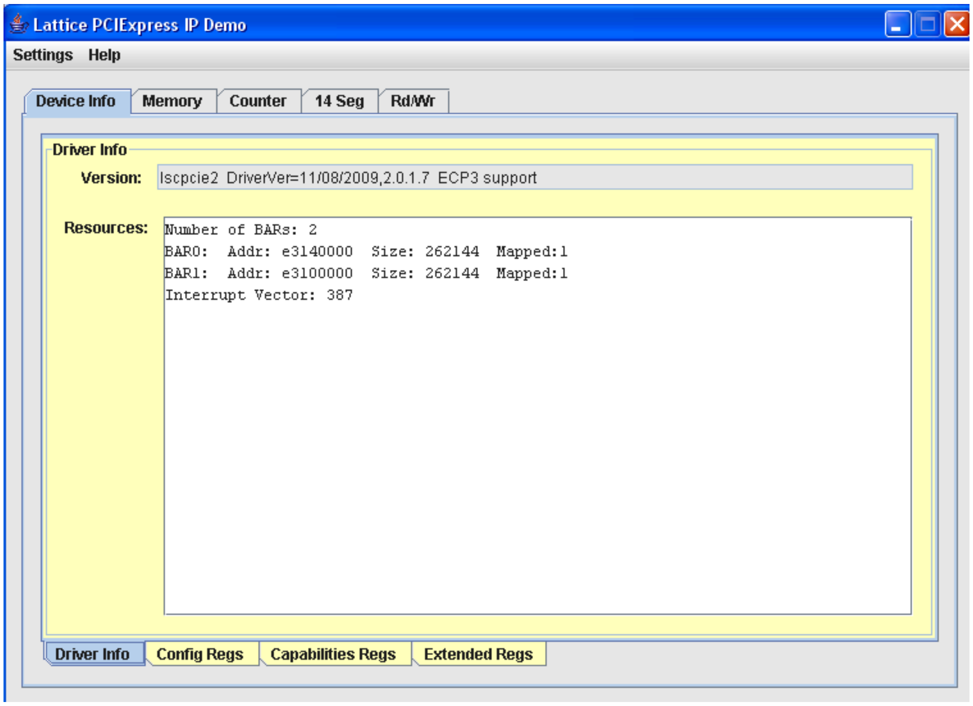
This section describes how to run the PCI Express Basic demo software after installation. You can access the PCI Express Basic demo software from the Windows Start Menu.

To run the PCI Express Basic demo software from your PC:

1. From the Windows desktop, choose **Start > Programs > LatticeECP3 Versa Evaluation Board > PCIe Basic Demo**.

The graphical user interface opens the PCI Express Basic demo software with the Device Info tab activated as shown in Figure 2.

Figure 2. PCI Express Demo Device Info Page



Version: lscpcie2 DriverVer=11/08/2009,2.0.1.7 ECP3 support

Resources:

```
Number of BARs: 2
BAR0:  Addr: e3140000  Size: 262144  Mapped:1
BAR1:  Addr: e3100000  Size: 262144  Mapped:1
Interrupt Vector: 387
```

The Device Info page displays information about the device driver and the device's PCI configuration registers. The data displayed is for informational purposes only and cannot be edited. Descriptions of all of the information you can view in this page are available in the Touring the PCI Express Basic Demo Interface section of this document.

Touring the PCI Express Basic Demo Interface

This section describes the pages and features of the PCI Express Basic demo software interface.

1. In the Device Info page, click on the **Device Info** sub tabs and observe the structure of the information that is displayed in each. Table 3 describes the information available for viewing by clicking each of the sub tabs at the bottom of the dialog.

Table 3. Device Info Page Sub Tab Descriptions

Sub Tab Page	Information Description
Driver Info	Obtained from the board PCI Config space registers by the Lattice PCI Express driver when the demo is started. Displays Windows resources assigned to the device driver to access the board.
Config Regs	Displays the standard PCI Config type 0 registers with each field annotated. Displaying this page causes the driver to issue PCI Config Type 0 read requests and re-displays the register values.
Capabilities Regs	Displays the PCI Express capabilities structures that are found in the register range 0x40 to 0xff. The applicable bit fields of the registers are parsed and displayed in readable format.
Extended Regs	Displays PCI Express extended configuration registers which are not used in this demo. These are inaccessible through the PC.

2. Now click on the 14 Seg tab to see the contents of the 14 Segment Control page. In this page, you will be running a demonstration LED sequence and controlling the display on your board from this console. See Figure 3.

   The 14-Segment Control page provides a way to interactively light segments on the display. You can preset character sequences from this page or select single characters and run them to light the display.

   The states of the LED segments are converted to a 14-bit word value (each segment is controlled by a bit) and written to the LED control register in the GPIO portion of the IP in the FPGA. This demonstrates a memory write across the PCI Express bus.

3. In the 14 Segment Control page, click the **Run** button. Notice on the board how the sequence of LED lights run in a certain pattern on the display.

   The 14-segment display has two test modes. In the first mode demonstrated here, a pre-set sequence of segments are lit and characters are written to the display.

   This LED sequence run takes approximately 30 seconds to complete. You must observe the 14-segment LEDs to see if it is operating correctly. The correct sequence is:

   a. Light all segments, one at a time, around the perimeter.

   b. Light all inner segments in a clock-wise order.

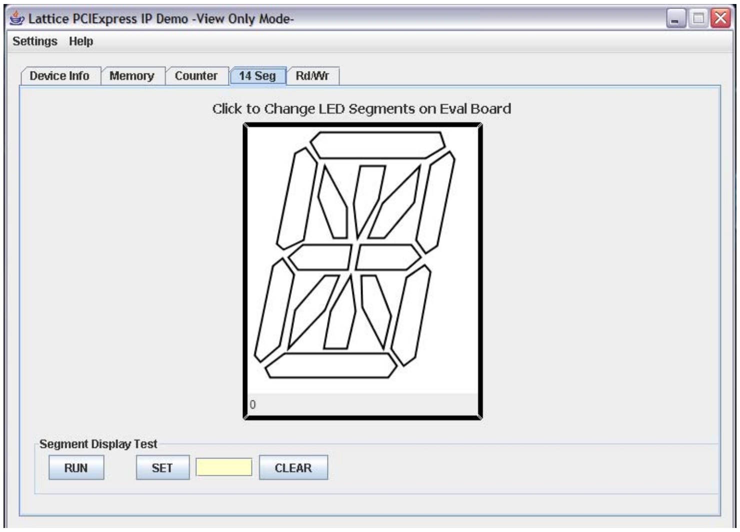
   c. Turn off all inner segments in reverse order.

   d. Turn off all outer segments in reverse order.

   e. Write the characters "LATTICE*" one at a time to the display.

   f. The "*" will be displayed when the test ends.

Figure 3. PCI Express Basic Demo 14 Segment Control Page



Lattice PCIExpress IP Demo -View Only Mode-

Click to Change LED Segments on Eval Board

See Table 4 for details about features on the 14-Segment Control page.

Table 4. 14 Segment Control Page Features

Feature	Description
LED Display	Allows you to interactively change the LED display using mouse clicks to toggle segments on and off.
RUN	Starts an LED light sequence or command operation.
SET	Sets a user-defined LED light command operation based on input characters in the text box.
CLEAR	Turns off all segments in the display.

4. Now click on any segment in the interactive segment display in the 14 Segment Control page. Notice that any selection will immediately cause the corresponding segment on the LED to light on your board's LED display.

   Clicking on a segment will turn it on or off (toggles). The 14-bit value written to the LED register in the FPGA is shown on the bottom left.

5. Now type any character in the text box and click the **Set** button. The character will be configured in the display.

   This second mode of operation allows a single character to be sent to the display. Any printable ASCII character can be displayed (lower case is displayed as upper case). You cannot write a blank character using Set.

6. Click the **Clear** button. This turns off all segments of the LED display. Right-clicking on the background area behind the segments will clear the entire display.

   The interactive 14 Segment Control page demonstration you just performed illustrates that a functional PCI Express communications path exists between the application and driver software that is running on the CPU and the FPGA IP.

7. Next, click on the **Memory** tab to open the Memory page. The Memory tab has various memory access tests that can be run to show that the IP is accessible from host software via the PCI Express bus. See Figure 4.

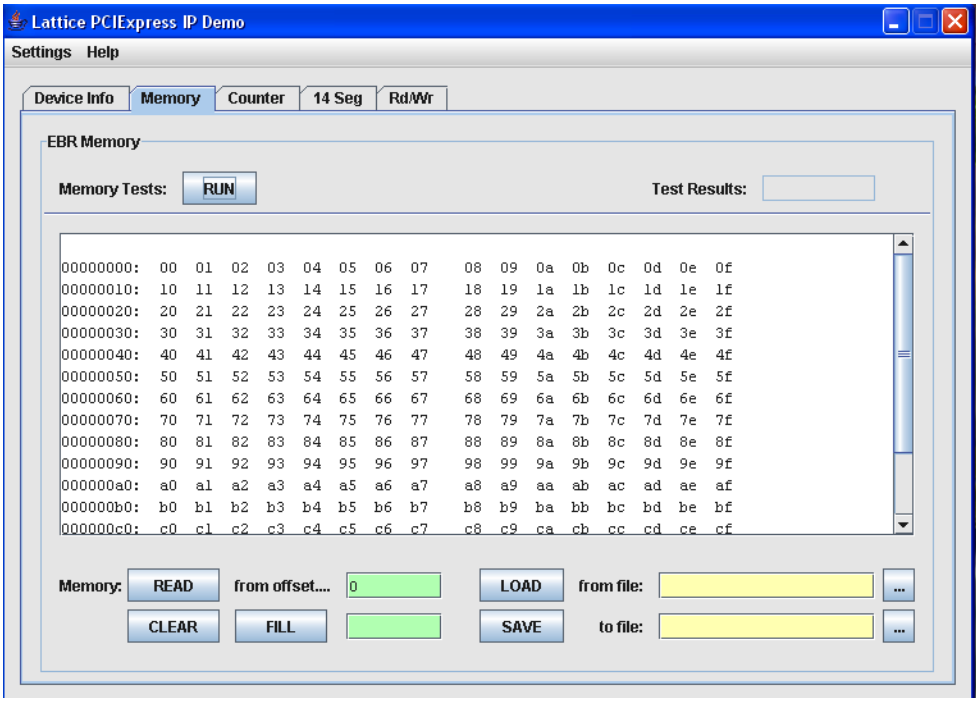
   The page contains text boxes for entering data to be sent to device registers in the FPGA design. These text boxes are color coded to indicate the data format they accept. See Table 5 for details about these codes.

Table 5. Memory Page Text Box Color Codes

Color Code	Description
Green	Indicates hex value fields. Do not include any prefixes (0x) or suffixes (H). Only digits are allowable.
Yellow	Indicates character string fields, e.g., ones containing file names, paths, or letter values.
Blue	Indicates decimal (base 10) value fields.

The Memory Page features allow you to test the access to the 16KB of EBR internal to the FPGA. Accesses are done on a byte basis. All 16KB memory locations are accessed successively, testing the PCI Express link to the memory interface. See Table 6 to for a list of the actions can be performed in this page.

Figure 4. PCI Express Basic Demo Memory Page



```
00000000:  00  01  02  03  04  05  06  07   08  09  0a  0b  0c  0d  0e  0f
00000010:  10  11  12  13  14  15  16  17   18  19  1a  1b  1c  1d  1e  1f
00000020:  20  21  22  23  24  25  26  27   28  29  2a  2b  2c  2d  2e  2f
00000030:  30  31  32  33  34  35  36  37   38  39  3a  3b  3c  3d  3e  3f
00000040:  40  41  42  43  44  45  46  47   48  49  4a  4b  4c  4d  4e  4f
00000050:  50  51  52  53  54  55  56  57   58  59  5a  5b  5c  5d  5e  5f
00000060:  60  61  62  63  64  65  66  67   68  69  6a  6b  6c  6d  6e  6f
00000070:  70  71  72  73  74  75  76  77   78  79  7a  7b  7c  7d  7e  7f
00000080:  80  81  82  83  84  85  86  87   88  89  8a  8b  8c  8d  8e  8f
00000090:  90  91  92  93  94  95  96  97   98  99  9a  9b  9c  9d  9e  9f
000000a0:  a0  a1  a2  a3  a4  a5  a6  a7   a8  a9  aa  ab  ac  ad  ae  af
000000b0:  b0  b1  b2  b3  b4  b5  b6  b7   b8  b9  ba  bb  bc  bd  be  bf
000000c0:  c0  c1  c2  c3  c4  c5  c6  c7   c8  c9  ca  cb  cc  cd  ce  cf
```

from offset.... 0

Table 6 provides descriptions of the Memory page features.

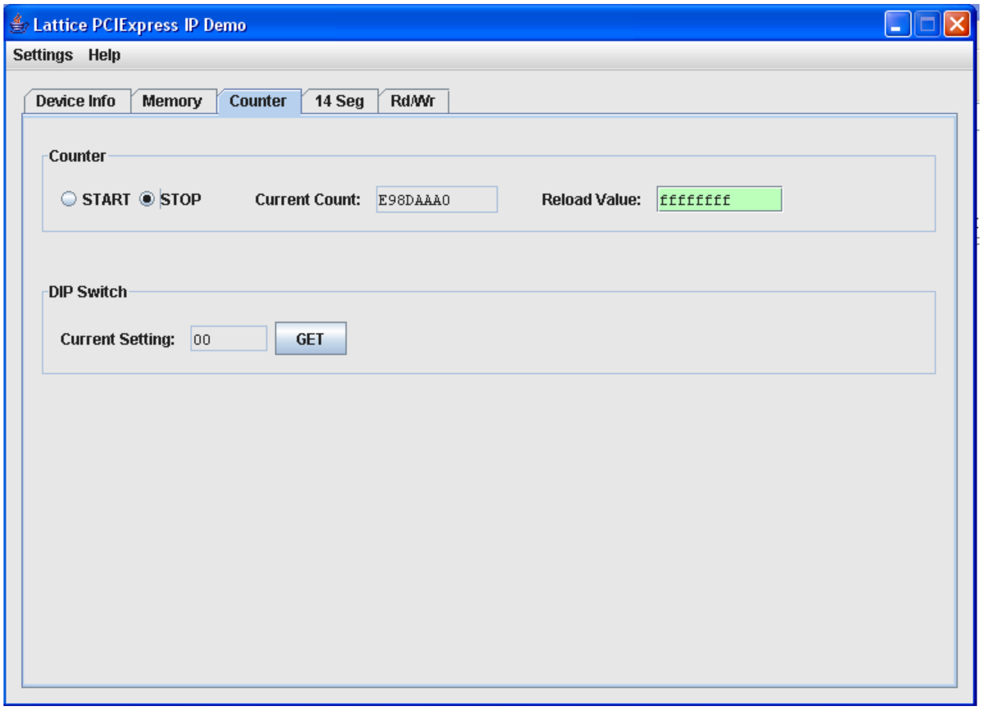
Table 6. Memory Page Features

Feature	Description
Pattern Tests	Pressing **Run** starts a test to check that all locations of the EBR can be read and written and that the contents are correct. First, all 16KB are cleared to 0 and verified. Then various patterns (AA, 55, 01, FF) are written to all locations and verified. If everything passes, PASS is displayed. If a memory location has an incorrect value the test aborts and displays ERRORS! The memory contents are left with an incrementing pattern 00 01 02... that is displayed when the test successfully finishes.
READ	The contents of the EBR memory are read from the value entered in the offset field. 256 bytes are read and displayed in the list box above.
CLEAR	Sets all 16KB to 0.
FILL	Writes the byte value entered in the field to all 16KB locations.
LOAD	Loads 16KB of binary data from the file specified (or as much data as is in the file) into EBR memory, starting at location 0. This can be used to load a known pattern into the EBR memory by using a file created by another tool.
SAVE	Writes all 16KB of EBR memory to the file specified. This can be used to save the contents of EBR memory for off-line processing (i.e., to verify that the pattern loaded in with LOAD is correctly saved in the EBR).

8. Next click on the **Counter tab** to open the Counter page. The Counter page allows you to control a 32-bit down counter in the FPGA hardware. The page is illustrated in Figure 5. Table 7 provides descriptions of the page's features.

   The counter is driven by the 125 MHz clock that feeds the IP. The counter is started by selecting the **Start** radio button. Counting begins from the value entered into the Reload Value field. The current count value is displayed in the Current Count field.

Figure 5. PCI Express Basic Demo Counter Page



Counter

○ START ● STOP Current Count: E98DAAA0 Reload Value: ffffffff

DIP Switch

Current Setting: 00

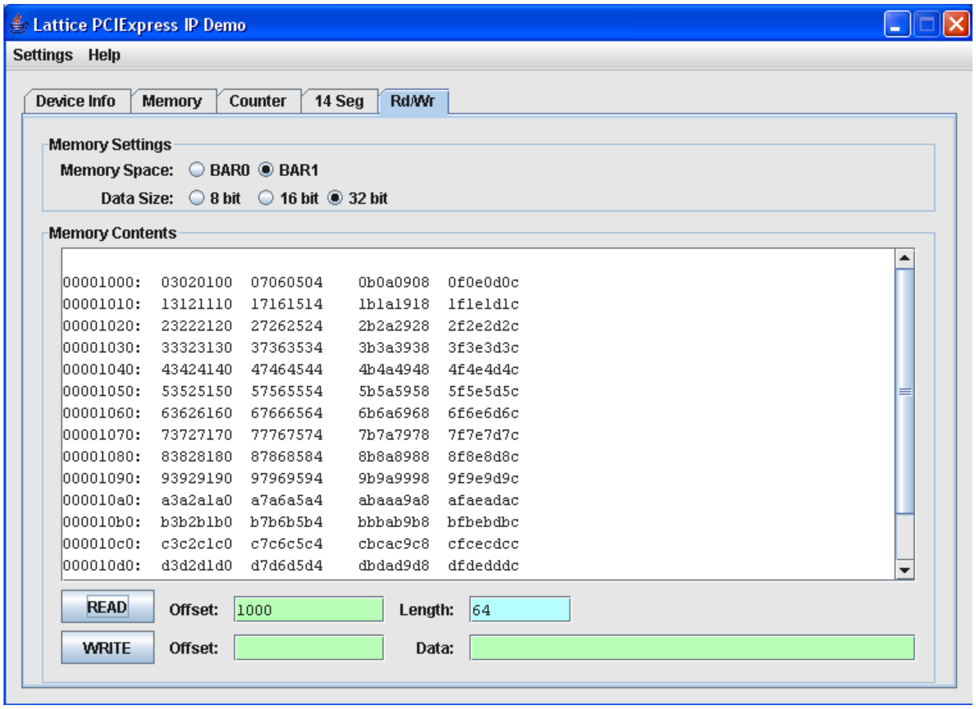
Table 7 provides descriptions of all of the Counter page features.

Table 7. Counter Page Features

Feature	Description
START/STOP	Starts and stops the 32-bit down counter in the FPGA hardware.
Current Count	Displays the current count value.
Reload Value	Sets the number from which counting begins.
DIP Switch	The DIP switch section shows that user changes to the switches on the board are seen by the application software on the PC. The GUI polls the DIP switch register 10 times per second and displays the value read from the 8-bit DIP switch register.
Get button	Used to immediately update the value. This is active if **No Polling** was selected from the Settings drop-down menu.

9. Finally, click on the **Rd/Wr tab** to open the Read/Write page. The Read/Write page is used for looking at and poking at registers and EBR memory values in the application IP. Refer to Figure 6.

Figure 6. PCI Express Basic Demo Read/Write Page



Memory Settings

Memory Space: ○ BAR0 ● BAR1

Data Size: ○ 8 bit ○ 16 bit ● 32 bit

Memory Contents

```
00001000:  03020100  07060504   0b0a0908  0f0e0d0c
00001010:  13121110  17161514   1b1a1918  1f1e1d1c
00001020:  23222120  27262524   2b2a2928  2f2e2d2c
00001030:  33323130  37363534   3b3a3938  3f3e3d3c
00001040:  43424140  47464544   4b4a4948  4f4e4d4c
00001050:  53525150  57565554   5b5a5958  5f5e5d5c
00001060:  63626160  67666564   6b6a6968  6f6e6d6c
00001070:  73727170  77767574   7b7a7978  7f7e7d7c
00001080:  83828180  87868584   8b8a8988  8f8e8d8c
00001090:  93929190  97969594   9b9a9998  9f9e9d9c
000010a0:  a3a2a1a0  a7a6a5a4   abaaa9a8  afaeadac
000010b0:  b3b2b1b0  b7b6b5b4   bbbab9b8  bfbebdbc
000010c0:  c3c2c1c0  c7c6c5c4   cbcac9c8  cfcecdcc
000010d0:  d3d2d1d0  d7d6d5d4   dbdad9d8  dfdedddc
```

Offset: 1000 Length: 64

Offset: Data:

The Read/Write page is primarily used for debugging and diagnosing the application IP registers. Table 8 provides descriptions of all of the Read/Write page features.

Table 8. Read/Write Page Features

Feature	Description
Memory Space	Indicates the base address register (BAR) memory space to access.
Data Size	Indicates bit size. Options are 8-bit, 16-bit, and 32-bit.
Memory Contents	Displays memory contents.
READ	Starts a read data access based on offset and length settings.
WRITE	Starts a write data access based on offset and data settings.

Data accesses can be specified as byte, short or word operations by selecting Data Size. Access is done to the selected Base Address Register (BAR). The memory contents are displayed in the window. In the address, the upper nibble (31:28) specifies the BAR being accessed. The following example shows reading the EBR memory (BAR 1, starting at offset 0x1000) in the application IP and displaying them in word format.

Data can be written to registers using the WRITE button. Specify the BAR Offset to start writing at and the hex data in the Data field. Separate each value with a space. Data size should match the Data Size selected at the top of the page in Memory Settings.

Rebuilding the PCI Express Basic Demo Design

You can rebuild the PCI Express Basic demo IP reference design by running the source HDL design files through a design flow in the Diamond software. All source HDL files and necessary project files are included in the kit installation. This document assumes that you have already installed Diamond and are familiar with basics tasks. Refer to Figure 1 to understand where various files referenced in this section are located.

We recommend that you copy the files from the installation location to a new working location. This allows you to quickly move back to the original configuration without re-installing this kit.

Implementing the PCI Express Basic Demo Design

The top.ldf Diamond project file is included in this kit. This file contains information regarding options to use when implementing the demo design. The top.lpf logical preference file specifies timing constraints and LatticeECP3 I/O pin assignments with respect to the LatticeECP3 Versa Evaluation Board. The working directory is the implementation directory.

To implement the demo design using the HDL source flow:

1. Open Diamond.
2. Click **File > Open Project**.
3. In the Open Project dialog, navigate to and select the top.ldf file in the **<kit_dir>\Hardware\PCIe_x1\Versa_PCIeBasic\Implementation\ecp3-35EA_PCIeBasic** directory path.
4. Click **Open**. All of the Verilog HDL files are imported into the project.
5. Choose **Project > Active Strategy > Translate Design Settings**. Verify that Macro Search Path is set to the directory path **.../Source/ipexpress/ecp3/pciex1 for Windows**. Your path would include forward slashes for Linux. Click **OK**.
6. In the File List pane of Diamond, right-click the device name at the top of the list, and chose **Properties** from the drop-down menu.
7. In the Project Properties dialog box, make sure that the following properties are selected: **LatticeECP3** family, **LFE3-35EA** device, **-8** speed grade, and a **FPBGA484** package. Click **OK**.
8. In the Process pane of Diamond double-click on the **Bitstream** file.

Modifying the PCI Express Basic Demo Design

This section provides a very simple alteration to the HDL to demonstrate a change in the behavior of the function of the LED light in the demo display. It will involve a small change in the HDL code in the source file.

Note: Since the source is being changed, the resulting netlist may be different and the provided start point for Place & Route may no longer produce a design that meets timing. Running more iterations to achieve timing may be required. See the Place & Route Properties.

To modify the PCI Express Basic demo design:

1. Open the **top_basic.v** file with an ASCII editing tool or the internal ASCII editing tool in Diamond. This file is located in the **<kit_dir\Hardware\PCIe_x1\Versa_PCIeBasic\Source\ecp3** path, where **<kit_dir>** represents the path **<install_dir>\DK-ECP3-PCIE-XXX** where XXX is the kit revision number.

2. On or about line 390 as shown below, delete the tilde character (~) that appears before the (led_out_int) parameter.

```
led_out          <= ~(led_out_int);
```

3. After making this small change, click **File > Save** and close the editor.

   This modification to the code will cause the 14-segment LED to operate in reverse, that is, all of the lights will be on when the demo starts instead of off.

4. Open the **top.ldf** project file in Diamond. This file is located in the **Hardware\PCIe x1\Versa_PCIeBasic\Implementation\ecp3-35EA_PCIeBasic** folder.

5. Double click the **Bitstream File** process in the Processes window to generate a top.bit file in the directory <kit_dir>\Hardware\Implementation.

6. Start ispVM and perform the steps described in the Reprogramming the LatticeECP3 Versa Evaluation Board to download the new bitstream to the board's SPI flash memory.

7. Push the **PROGRAM** button on the board to program your device from SPI Flash memory.

8. Reboot the PC so that the BIOS recognizes the new PCI Express endpoint device configuration.

9. Verify that the status LEDs are correct and note that all the LED segments are now on.

10. Rerun the LED test described in the Touring the PCI Express Basic Demo Interface section of this document. Notice that the state of the 14-segment LED on the board is the inverse of what is displayed in the GUI.

You have now completed the Lattice PCI Express Basic demo and have successfully completed all of the learning objectives of this kit.

Running the PCI Express Throughput Demo

This chapter describes the Lattice PCI Express Throughput demo that you can run within this kit on a Windows system (Windows 7, Windows XP, Windows 2000, Server2003).

Before You Begin

Before beginning this demo, you must do the following operations.

- Use ispVM to download the bitstream for this demo to the LatticeECP3 Versa Evaluation Board.

  You can find the x1 bitstream and the XCF file necessary for ispVM in the **<kit_dir>\Demonstration\PCIeThruput\Bitstreams** directory path.

  For general information on ispDOWNLOAD cable and ispVM configuration software usage, see the following:

  – Download procedure described in the Reprogramming the LatticeECP3 Versa Evaluation Board section of this document.
  – General introduction to device configuration with ispDOWNLOAD cable and ispVM software described in the Device Configuration Software and Cable section of this document.
  – ispVM Help system

- Install the board drivers for the application.

  You can find the driver files necessary for proper demo installation in the

<kit_dir>\Demonstration\PCIeThruput\Driver directory path. See the procedure described in the Installing LatticeECP3 Versa Evaluation Board Drivers section of this document for guidance.

Resource References

Please be aware of supplementary companion documentation when using this demo.

Hardware Resources

The PCIe Throughput Demo x1 bitstream is built from the Diamond project located in **Hardware\PCIe x1\Versa_PCIeThruput\Implementation\ecp3-35EA_PCIeThruput**. The Verilog source code is located in the project Source\directory.

The Verilog design architecture is explained in UG07, PCI Express Throughput Demo Verilog Source Code User's Guide. This document describes the purpose and functionality of the Verilog modules used in PCIe Throughput Demo design.

Software Resources

The PCI Express Throughput Demo uses the lscpcie2.sys device driver. The source code for this device driver is located in Software\lscpcie2_Win7\Driver or Software\lscpcie2_Win2kXP\drvr. The architecture of the lscpcie2 device driver is explained in the lscpcie2 Driver Reference Manual which can be accessed through the Software\PCIeDocIndex.html link.

The PCIe Throughput Demo application source code is located in Software\PCIeSFIF_Win2kXP\SFIF_GUI\SFIF_UI. This directory contains the Java project source code to create the user interface. The GUI also uses the PCIeAPI_Lib_Win2kXP API library. Both of these codes are common for Windows 7 and Windows XP.

The architecture of the PCI Express Throughput Demo application is explained in the PCI Express Throughput Demo Reference Manual and the PCI Express API Reference Manual that can be accessed through the Software\PCIeDocIndex.html link.

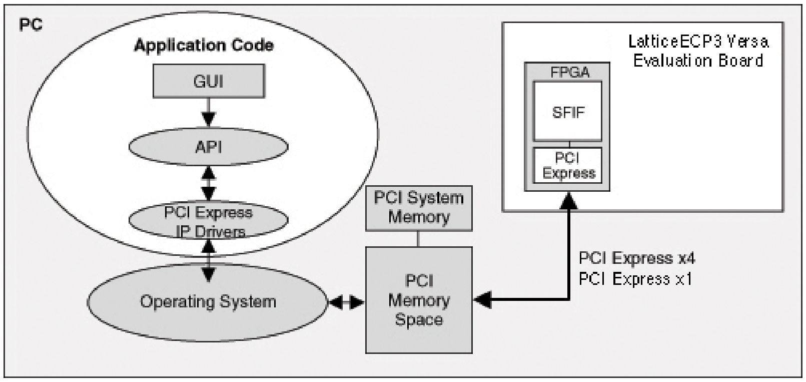
Throughput Demo Operations Overview

The purpose of this demo is to show the performance of the Lattice PCI Express SERDES hardware and PCI Express Endpoint IP core when operating in a PC PCI Express expansion slot. The data rates for writes to the PC system memory and reads from the PC system memory are measured and displayed in a graphical user interface.

The demo performs Direct Memory Access (DMA) operations by transferring data directly to and from the PC memory. The demo uses an IP block named the SFIF (Stored FIFO InterFace) to generate read and write Transaction Layer Packets (TLPs) that will access the PC system memory. The SFIF exercises the PCI Express Endpoint IP core and link with low overhead so the true performance of the PCI Express core and link can be measured.

The PCI Express interface is used for both control plane and data plane traffic. The control plane loads the SFIF memory and sets up the transfer. The data plane transfers the data from the SFIF to the PC memory. Figure 7 shows the relationship of the hardware and software components of the demo. For more details on SFIF IP, register mapping and related topics, see the documents IPUG75, PCI Express 1.1 x1, x4 IP Core User's Guide and UG07, Lattice PCI Express Throughput Demo Verilog Source Code User's Guide.

Figure 7. PCI Express Throughput Block Diagram



The Throughput demo software allows you to set up different types of data transfers to understand the PCI Express link. You can select the type of transfer to perform (e.g., write, read, and write/read) as well as how many bytes of data to transfer. You also have the option of selecting the size of the TLP in which to perform the transfer.

Note: The PCI Express Throughput demo design requires at least 16 posted credits to use 128-byte write TLPs. This requirement is to optimize the throughput of the PCIe link. You can determine the amount of posted credits for the given slot in the GUI. If the posted credits are less than 32, then 64-byte write TLPs are the largest size supported.

If you experience any problems running this demo, please refer to Appendix B.

Running the Throughput Demo

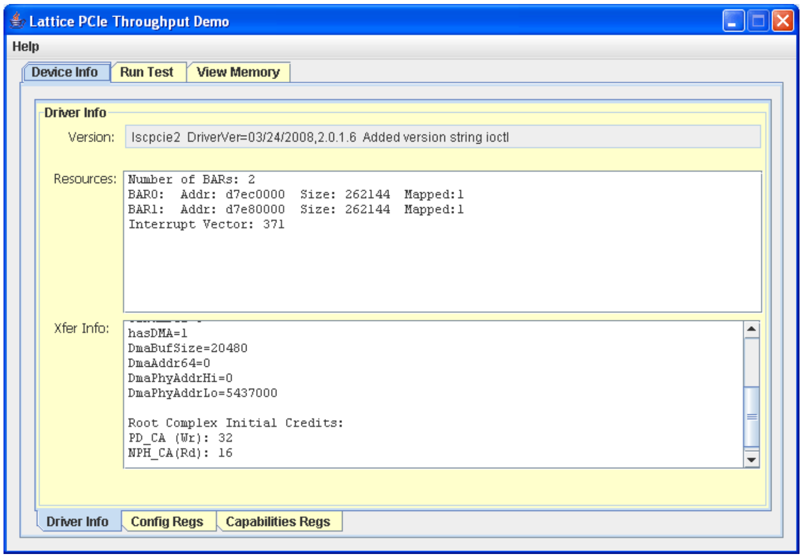
This section describes how to run the PCIe Throughput demo after installation. You can access the demo from the Windows Start Menu.

To run the PCIe Throughput demo from you PC:

1. From the Windows desktop, choose **Start > Programs > LatticeECP3 Versa Evaluation Board > ECP3 Thruput**.

The graphical user interface opens the PCI Express Throughput demo software with the Device Info tab activated as shown in Figure 8.

Figure 8. Throughput Demo Device Info Page



Version: lscpcie2 DriverVer=03/24/2008,2.0.1.6 Added version string ioctl

Resources:

```
Number of BARs: 2
BAR0:  Addr: d7ec0000  Size: 262144  Mapped:1
BAR1:  Addr: d7e80000  Size: 262144  Mapped:1
Interrupt Vector: 371
```

Xfer Info:

```
hasDMA=1
DmaBufSize=20480
DmaAddr64=0
DmaPhyAddrHi=0
DmaPhyAddrLo=5437000

Root Complex Initial Credits:
PD_CA (Wr): 32
NPH_CA(Rd): 16
```

The Device Info page displays information about the board's PCI config space registers, PCI Express capabilities, and root complex buffer sizes. Descriptions of the information you can view in this page are available in the Touring the PCI Express Throughput Demo Interface section of this document.

Touring the PCI Express Throughput Demo Interface

This section describes the pages and features in the PCI Express Throughput Demo interface.

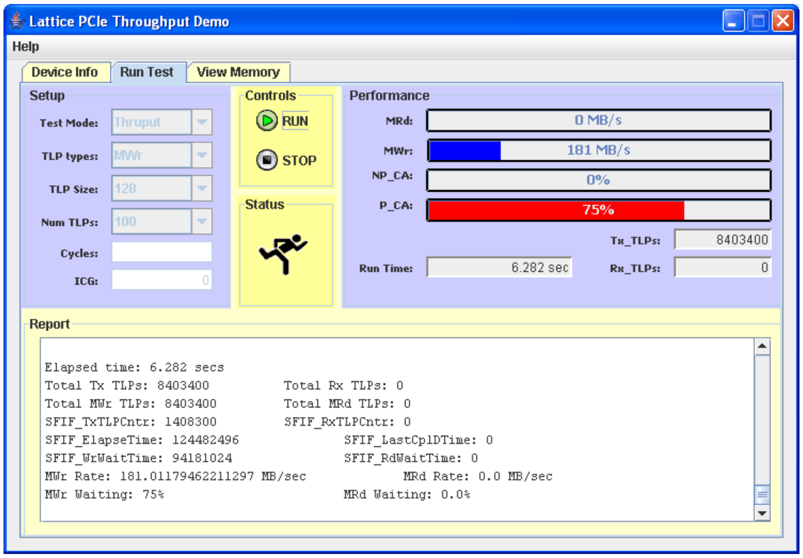
1. In the Device Info page, click on the **Device Info sub tabs** and observe the structure of the information that is displayed in each. Table 9 describes the information available for viewing in the sub tabs at the bottom of the page.

Table 9. Device Info Page Sub Tab Descriptions

Sub Tab Page	Information Description
Driver Info	Provides information about the device driver including the version, the resources used, and the transfer information. The demo design uses the lscpci2 driver. The demo requests two BARs (Base Address Registers) and a single interrupt vector. The Xfer Info box provides the buffer sizes for the root complex for Posted and Non-Posted TLPs. This information is important when considering the amount of credit waiting the demo design demonstrates when running a transfer. A root complex with larger buffers will provide better performance when running the demo since it will not have to release credits as quickly to allow the next TLP.
Config Regs	Provides the standard PCI Type0 space configuration register contents. Things such as Device ID and Vendor ID are displayed and the assigned BARs.
Capabilities Regs	Provides the link list of capability structures and their contents. Key information found in this box is the maximum TLP size supported by the root complex and the negotiated link width.

2. Now click on the **Run Test** tab to see the contents of the Run Test page. This page operates the demo design. In this page, you will be running the demonstration to compute the throughput of the PCI Express link and display the transfer rates with bar graphs. You can select read, write and read-write throughput tests. See Figure 9.

Figure 9. Throughput Run Test Info Page



Test Mode: Thruput
TLP types: MWr
TLP Size: 128
Num TLPs: 100
Cycles:
ICG: 0

MRd: 0 MB/s
MWr: 181 MB/s
NP_CA: 0%
P_CA: 75%
Tx_TLPs: 8403400
Run Time: 6.282 sec
Rx_TLPs: 0

```
Elapsed time: 6.282 secs
Total Tx TLPs: 8403400         Total Rx TLPs: 0
Total MWr TLPs: 8403400        Total MRd TLPs: 0
SFIF_TxTLPCntr: 1408300        SFIF_RxTLPCntr: 0
SFIF_ElapseTime: 124482496             SFIF_LastCplDTime: 0
SFIF_WrWaitTime: 94181024              SFIF_RdWaitTime: 0
MWr Rate: 181.01179462211297 MB/sec            MRd Rate: 0.0 MB/sec
MWr Waiting: 75%                       MRd Waiting: 0.0%
```

3. On the Run Test page, under Setup options, select the following:

   Test Mode: **Thruput**

   TLP Type: **MWr**

   For the rest of the options, take the defaults.

4. On the Run Test page, click the **RUN** button. After running your test, notice the status indicators in the Performance section at right. The two top progress indicator bars for MRd (memory read TLPs), and MWr (memory write TLPs) will contain a percentage of blue which indicates throughput. The two progress indicator bars below that show the wait time for the root complex to accept TLPs over the entire time spent running.

See Table 10 for details about features on the Run Test Feature page.

Table 10. Run Test Page Feature Descriptions

Feature	Description
Setup	Sets the configuration for the specific test.
Test Mode	There are two modes of operation, cycles and throughput. **Throughput** - This mode allows the test to run continuously looping through the tx_fifo and updating the performance numbers every second. This test will run until the you click the STOP button. **Cycles** - This mode allows you to set up a specific number of times the tx_fifo will be looped. Once complete, the test will stop automatically and the performance numbers will be displayed based on the entire run. The cycle consists of all MRd TLPs or all MWr TLPs. The purpose is to validate that the correct number of TLPs was sent/received using the counters and View Memory page. See TLP Types in this table for descriptions. The key difference between the two modes is how the performance data is displayed. A throughput test will provide new performance data every second. A cycle test will provide performance data after the number of cycles completes, or one second, whichever comes first. Note that cycles will not run longer than one second. Cycles tests run once; throughput tests run continuously until stopped.

Table 10. Run Test Page Feature Descriptions (Continued)

Feature	Description
TLP Types	There are four types of TLP types which impact the type of "traffic" sent over the PCI Express link. **MWr** – Memory Write TLPs to write data from the endpoint to the PC system memory. **MRd** – Memory Read TLPs to read data from PC system memory to the endpoint. **MRd+MWr** – Both Memory Read and Memory Write TLPs are sent to the root complex. **R+W+Ctl** – Read, Write, and Control data are present on the PCI Express link. The Read and Write TLPs are sent from the SFIF while the PC is also modifying the GPIO 14-segment display LEDs. This TLP type shows both data and control plane TLPs sharing the PCI Express link.
TLP Size	The TLP size controls allows you to select the size of the TLPs to be sent from the SFIF. The maximum size of the TLP will be dependent on the root complex. In MRd mode, the maximum TLP size is limited by Max Read Request size (512 bytes). In MWr mode, the maximum TLP size is limited by Max TLP Size (128 bytes). In Read/Write mode the following sizes are available for MRd TLP and MWr TLP combinations. 512,128 – 512-byte read requests with 128-byte write TLPs 256,128 – 256-byte read requests with 128-byte write TLPs 128,128 – 128-byte read requests with 128-byte write TLPs 64,64 – 64-byte read requests with 64-byte write TLP 32,32 – 32-byte read requests with 32-byte write TLPs 16,16 – 16-byte read requests with 16-byte write TLPs
Num TLPs	This control allows the user to select the ratio of read requests to write TLPs. **1Rd,1Wr** – This ratio results in one Read Request and 1 Write TLP back-to-back. The completion data will need to be received before another Read Request can be made. **1Rd,4Wr** – This ratio results in one Read Request and four Write TLPs. The completion data must be received before another Read Request can be made. This results in much greater bandwidth since read requests are four times the size of a write TLP. This ratio balances the PCI Express link, but still waits for read data. **4Rd,16Wr** – This ratio results in four Read Requests and 16 Write TLPs. This ratio allows for four read requests to be outstanding (TAGs). This ratio is only recommended on server class motherboards due to the high bandwidth required. With four reads outstanding the root complex can better utilize the read data. **16Rd,64Wr** – This ratio results in 16 Read Requests and 64 Write TLPs. This ratio allows for 16 read requests to be outstanding (TAGs). This ratio is only recommended on server class motherboards due to the high bandwidth required. With 16 read outstanding the root complex can better utilize the read data.
Cycles	This control is only available when the Cycle mode has been selected. This controls the number of times the tx_fifo is looped before ending the test. The software starts the SFIF and waits one second while the SFIF transfers data (number of cycles). After one second, the software stops the SFIF and displays the performance. This has the effect of limiting cycles tests to a maximum of one second of operation. The cycles value cannot be larger than 65535 (it is a 16-bit counter). In Throughput mode this control is not used. In Throughput mode the SFIF is looping the tx_fifo continuously until the user presses the STOP button.
ICG	(Inter Cycle Gap) This control sets the number of 125MHz clock cycles between cycles. You can use this control to model TLP traffic patterns that may be appropriate for your system. The ICG value cannot be larger than 65535 (it is a 16-bit counter).
Controls	Stops and starts test. Status shows an image of a running man to indicate test is in progress.

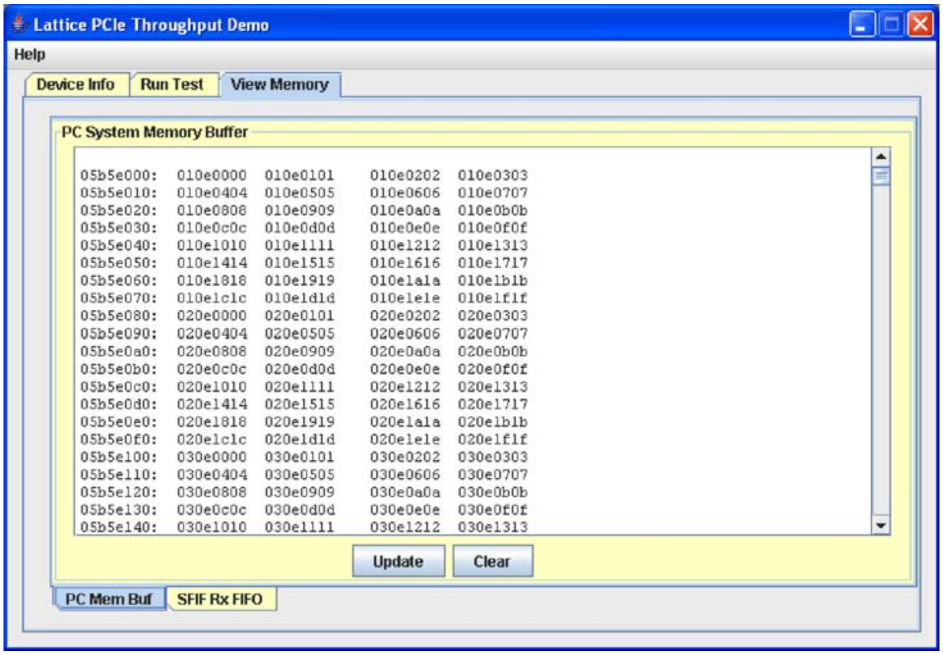
Table 10. Run Test Page Feature Descriptions (Continued)

Feature	Description
Performance	Displays the current data rates and other statistics. Data rates are displayed as progress bars, with the rate (Mbps) displayed in the bar. The bars are updated every second when running in Throughput mode or upon completion in Cycle mode. The rates are computed from the hardware counters in the SFIF. Write rates are computed from the following SFIF hardware counters: Tx TLP Count and Elapsed Count. Write Rate (MB/sec) = (Tx TLP Count * TLP Size) / (Elapsed Count * 8ns) Read rates are computed from the following SFIF hardware counters: Rx TLP Count and CplD Timestamp. Read Rate (MB/sec) = (Rx TLP Count * RCB_Size) / (Elapsed Count * 8ns) RCB_Size is the size in bytes of a CplD. In Throughput mode performance is recalculated every second and the counters are reset. In Cycles mode the performance is calculated once at the end of the run displaying the results for the entire transfer.
NP_CA and P_CA	Time spent waiting for the root complex to accept TLPs is computed and displayed as a bar graph in percentage of time waiting over time spent running. Normal efficient operation should show a small percentage of time spent waiting for credits and more time spent sending TLPs. Counters record when the SFIF wants to send a MRd but the credit available ports of the PCI Express core indicates the root complex has not yet processed the read requests. The PCI Express core is waiting to accept an UpdateFC-NP freeing up Non-Posted credits to send another MRd TLP. Counters record when the SFIF wants to send a MWr but the credit available ports of the PCI Express core indicates the root complex has not yet processed the sent write TLPs. The PCI Express core is waiting to accept an UpdateFC-P freeing up Posted credits to send another MWr TLP.
Report	Logs all of the test information. The Report box provides details about the test. In the Throughput mode this report will be updated every second up to 10 seconds. After 10 seconds the data is no longer updated in the report box to prevent system load and excessive resource usage during long duration tests (over night). In the Cycles mode the report window is updated when the test is complete.

5. Next, click on the **View Memory tab** to open the View Memory page. Refer to Figure 10. Notice that this page allows you to inspect the memory contents of the PC system memory buffer and the SFIF rx_fifo to check for data integrity. See the descriptions of these sub tabs in Table 11.

After inspecting this page, you can move on to the next chapter which describes the PCIe DMA demos.

Figure 10. Throughput Demo View Memory Page



```
05b5e000:  010e0000  010e0101   010e0202  010e0303
05b5e010:  010e0404  010e0505   010e0606  010e0707
05b5e020:  010e0808  010e0909   010e0a0a  010e0b0b
05b5e030:  010e0c0c  010e0d0d   010e0e0e  010e0f0f
05b5e040:  010e1010  010e1111   010e1212  010e1313
05b5e050:  010e1414  010e1515   010e1616  010e1717
05b5e060:  010e1818  010e1919   010e1a1a  010e1b1b
05b5e070:  010e1c1c  010e1d1d   010e1e1e  010e1f1f
05b5e080:  020e0000  020e0101   020e0202  020e0303
05b5e090:  020e0404  020e0505   020e0606  020e0707
05b5e0a0:  020e0808  020e0909   020e0a0a  020e0b0b
05b5e0b0:  020e0c0c  020e0d0d   020e0e0e  020e0f0f
05b5e0c0:  020e1010  020e1111   020e1212  020e1313
05b5e0d0:  020e1414  020e1515   020e1616  020e1717
05b5e0e0:  020e1818  020e1919   020e1a1a  020e1b1b
05b5e0f0:  020e1c1c  020e1d1d   020e1e1e  020e1f1f
05b5e100:  030e0000  030e0101   030e0202  030e0303
05b5e110:  030e0404  030e0505   030e0606  030e0707
05b5e120:  030e0808  030e0909   030e0a0a  030e0b0b
05b5e130:  030e0c0c  030e0d0d   030e0e0e  030e0f0f
05b5e140:  030e1010  030e1111   030e1212  030e1313
```

See Table 11 for details about the features of the View Memory page

Table 11. View Memory Page Sub Tab Descriptions

Sub Tab Page	Description
PC Mem Buf	The PC system memory buffer sub tab allows you to inspect the contents of the PC system memory buffer allocated in the kernel space by the driver and is used for the source of MRd requests and destination for MWr TLPs. This can also be used to verify that the MWr TLPs have worked and that the data was transferred from the LatticeECP3 Versa Evaluation Board into system memory.
SFIF Rx FIFO	The SFIF Rx FIFO sub tab displays the parsed and formatted contents of the SFIF rx_fifo. This can be used to verify that a small burst of MRd TLPs have returned the proper data to the board. The TLPs are parsed and time stamped.

Running the PCI Express Scatter-Gather DMA Demos

This section describes the Lattice PCI Express Direct Memory Access (DMA) demos that you can run within this kit on a Windows system (Windows 7, Windows XP, Windows 2000, Windows Server 2003).

The DMA demos illustrate Lattice PCI Express, DDR3 SDRAM Controller (optional) and Scatter-Gather DMA (SGDMA) IP cores working together to transfer data over the PCI Express bus. The Scatter-Gather DMA operates as a Master DMA, reading and writing data to PC system memory.

One demo illustrates moving large amounts of image data from the LatticeECP3 Versa Evaluation Board to PC system memory and display on the screen. The other demo implements a simple hardware image processor, in which pixel data from a source image on the screen is read by the board, modified by the hardware and written back and redisplayed. A test program is also provided that checks all the driver and IP functionality.

Before You Begin

Before beginning this demo, you must do the following operations.

- Use ispVM to download the bitstream for this demo to the LatticeECP3 Versa Evaluation Board.

  You can find the x1 bitstream and the XCF file necessary for ispVM in the <**kit_dir>\Demonstration\PCIeDMA\Bitstreams** directory path.

  For general information on ispDOWNLOAD cable and ispVM configuration software usage, see the following:

  – Download procedure described in the Reprogramming the LatticeECP3 Versa Evaluation Board section of this document.
  – General introduction to device configuration with ispDOWNLOAD cable and ispVM software described in the Device Configuration Software and Cable section of this document.
  – ispVM Help system

- Install the board drivers for the application.

  You can find the driver files necessary for proper demo installation in the **<kit_dir>\Demonstration\PCIeDMA\Driver** directory path. See the procedure described in the Installing Drivers section of this document.

Resource References

Please be aware of supplementary companion documentation when using this demo.

Hardware Resources

The PCI Express SGDMA Demo x1 bitstream is built from the Diamond project located in **Hardware\PCIe_x1\Versa_PCIeSGDMA\Implementation\ecp3-35EA_PCIeSGDMA**. The Verilog source code is located in the project Source\ directory.

The Verilog design architecture is explained in UG06 Lattice PCI Express x4 Scatter-Gather DMA Demo Verilog Source Code User's Guide. This document describes the purpose and functionality of the Verilog modules used in PCIe SGDMA Demo design.

Software Resources

The PCI Express SGDMA Demo uses the lscdma.sys device driver. The source code for this device driver is located in Software\lscdma_Win7\Driver or Software\lscdma_Win2kXP\drvr. The architecture of the lscdma device driver is explained in the lscdma Driver Reference Manual that can be accessed through the **Software\PCIeDocIndex.html** link.

The PCI Express SGDMA Demo application source code is located in two directories corresponding to the two demos: **Software\PCIeDMA_Win2kXP\ColorBars** and **Software\PCIeDMA_Win2kXP\ImageMove**. These applications use the OpenGL APIs for displaying the image data. The demos also use the PCIeAPI_Lib_Win2kXP API library. Both of these codes are common for Windows 7 and Windows XP.

The architecture of the PCI Express SGDMA Demo application is explained in the PCI Express DMA Demo Reference Manual and PCI Express API Reference Manual that can be accessed through the **Software\PCIeDocIndex.html** link.

DMA Demo Operations Overview

Direct Memory Access (DMA) is a method of transferring data from one memory mapped device to another. The data is transferred by a dedicated device that performs the bus cycle (memory reads and writes). The CPU is not involved in the actual data movement.

Using a dedicated DMA device frees the CPU to do other operations and also shortens the transfer time. If the

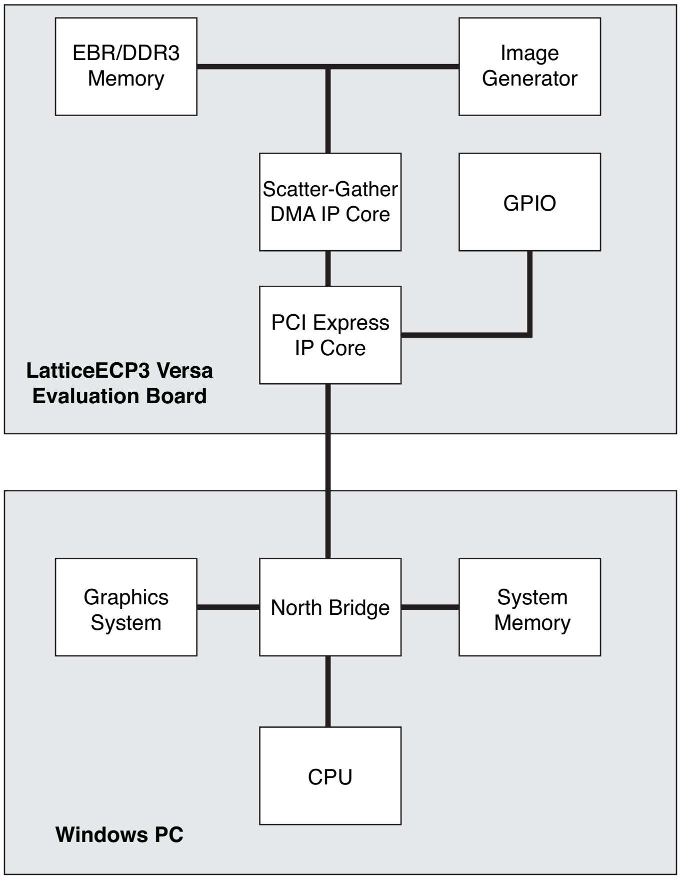
CPU had to move the data, it would be done in a software loop which requires fetching, decoding and executing each instruction involved in the loop. This could easily expand to 10 or more instruction cycles per datum moved. A DMA engine could perform the same datum move operation in one to three bus clocks (depending on bus architecture).

In modern PC systems the DMA engine, the device responsible for performing the bus cycles to implement the transfer, is located on the add-in card. This is known as Bus Master DMA and is the preferred method of operation. The PCI bus is being phased out and replaced with the PCI Express bus. To take advantage of the high bandwidth that PCI Express offers, DMA is used to transfer the data between the add-in card and the system memory. The Lattice Scatter-Gather DMA IP works in conjunction with the Lattice PCI Express Endpoint IP core to transport the data.

The Scatter-Gather DMA IP core, together with the PCI Express Endpoint IP core, demonstrates moving data between the Lattice FPGA and PC system memory using a LatticeECP3 Versa Evaluation Board. The board uses the PCI Express link as both control (setup and operation of the core) and data path (DMA to/from PC system memory). The PC provides the test platform (power, run-time environment) and the user interface.

A PC platform is used because currently PCs are the only readily available, economical and standard platform utilizing PCI Express. A Windows device driver provides the interface to the board's register and memory space. Application software uses the driver to set up and configure the DMA engine, execute it, and verify the results. The demo system is illustrated in the block diagram shown in Figure 11.

Figure 11. DMA Demo Block Diagrams



The demo hardware has the following objectives:

- Acts as a reference design for using the PCI Express and Scatter-Gather DMA IP cores

- Performs actual DMA transfers over the PCI Express bus at an optimal rate
- Provides counters and timers to measure performance
- Provides a platform for demonstration and experimentation

The demo application software has the following objectives:

- Demonstrates accessing, configuring and operating the PCI Express, DDR3 SDRAM Controller and Scatter-Gather DMA IP cores
- Verifies proper operation (ensures all DMA data is transferred from source to destination un-corrupted)
- Demonstrate Windows driver and system programming so users can extend software for their own particular system needs
- System memory allocation (Memory Descriptor Lists)
- Interrupt handling - ISRs and DPCs

Scatter-Gather DMA Overview

Hardware devices perform Direct Memory Access (DMA) by initiating read/write bus transactions. DMA means transferring data to and from system memory directly, without involving the CPU.

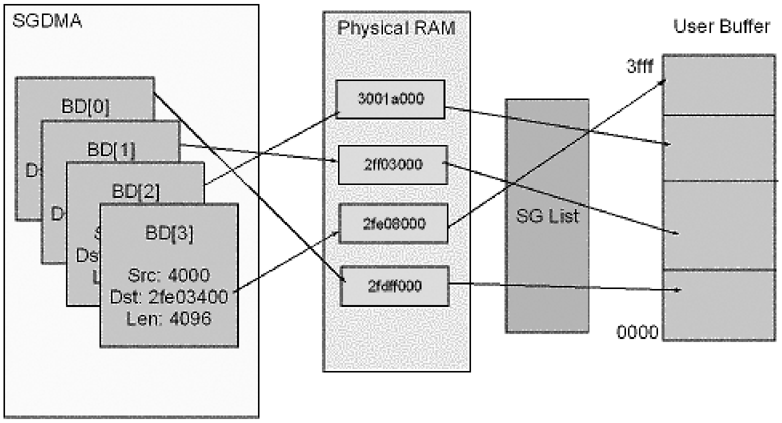
Bus Master DMA means the device (the PCI Express Core on the board) is controlling the bus and doing the data transfers. In order to perform the transfer, an address is needed and a length. The SGDMA is configured by the software driver. The addresses known to software for describing a buffer's location in memory are only relevant in the domain of the CPU. The CPU (and software) view memory as virtual 2 GB address spaces per process. The DMA needs physical memory addresses.

When software allocates a large buffer of memory, the memory manager finds the number of required free pages (4KB per page) in system memory and makes them appear contiguous to software via virtual memory translation tables in hardware. A 1MB buffer allocated by the software appears contiguous to the software, but in reality is scattered throughout physical system memory in discontinuous 4KB chunks. The magic of virtual memory makes it appear contiguous to software.

The SGDMA needs physical addresses to put on the bus and needs contiguous memory. In a simple flat memory architecture, the SGDMA could just take a starting address and a length of 1MB and transfer all data in one continuous operation. In virtual memory machines, the kernel and memory manager need to be enlisted at the driver level to create a map of the virtual memory to physical pages. In Windows, this mapping is known as a Memory Descriptor List (MDL).

The MDL is a Scatter-Gather List that maps virtual memory to physical page addresses. The device driver uses the MDL entries to program the buffer descriptors. Each buffer descriptor is programmed with the physical memory address and the length (usually one page, 4096 bytes). When the Scatter-Gather DMA channel is activated it reads the linked list of buffer descriptors and moves the data to that address, and then moves to the next buffer descriptor and next address until the end of the list is reached. Figure 12 illustrates this operation.

Figure 12. Scatter-Gather DMA Buffer Address Mapping



The buffer descriptors (BD[1, 2,..], shown on the left in Figure 12 have their destination addresses programmed to the start of the physical pages in memory. These pages in Physical RAM to the right of the buffer descriptors may not be contiguous or sequential in memory. The memory manager in the PC hardware uses the MDL or Scatter-Gather List (SG List) to make this set of pages appear contiguous to the application running in user space (Virtual Memory mode).

The Scatter-Gather DMA off-loads the processor and kernel by having the ability to perform this scattering of contiguous data (memory on the LatticeECP3 Versa Evaluation Board) to arbitrary memory pages, or for reading, to gather a set of discontinuous memory pages into a contiguous memory on the board.

See the code links to the DMATest.cpp, ColorBars.cpp and ImageMove.cpp files included with this kit. The source code is the best documentation of what is happening behind the scenes in the demos. To access this documentation, go to the kit **Software** directory and open the **PCIeDocIndex.html** document. Under the **Documentation** section click the hyperlink **PCIe DMA Demo Reference Manual**. Click on the **File List** book in the navigation pane at left or the **Files** tab in the main pane on the page.

Note: The PCI Express Scatter-Gather DMA demo design requires at least 16 posted credits. This requirement is to optimize the throughput of the PCI Express link. You can determine the amount of posted credits for the given slot using the PCI Express Throughput demo and GUI. If the posted credits are less than 32, then the PCI Express Scatter-Gather DMA demo will not be able to run in the given slot.

If you experience problems running this demo, please refer to Appendix B of this document.

Running the DMA Demos

This section describes how to run the DMA demos and refers you to documentation on the demos that describes what these applications demonstrate.

To run the ColorBars graphical DMA demo:

- From the Windows desktop, choose **Start > Programs > LatticeECP3 Versa Evaluation Board > ColorBars**.

  In this demo, image data is transferred from the board to PC memory and displayed. The ColorBars window displays a series of vertical colored bars in a gradient manner. See Figure 13. For details on this demo, see the PCI Express DMA ColorBars Demo section of this document.

To run the ImageMove graphical DMA demo:

- From the Windows desktop, choose **Start > Programs > LatticeECP3 Versa Evaluation Board > ImageMove**.

  In this demo, image data is transferred from the PC to the board and then back to the software, which then displays a modified image on the screen. See Figure 14. For details on this demo, see the PCI Express DMA ImageMove Demo.

Figure 13. ColorBars Demo Window

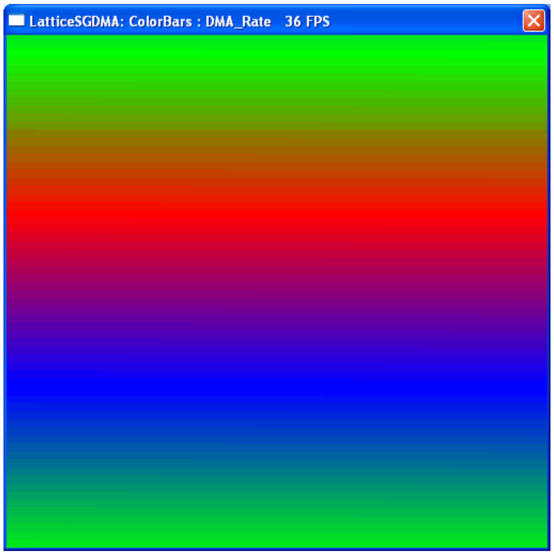
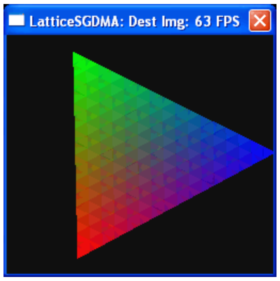


Figure 14. ImageMove Demo Window



The 14-segment LED displays the real-time interrupt processing during execution of the ColorBars and ImageMove. The inner eight segments are the lower eight bits of the ISR routine counter. The outer eight segments are the lower eight bits of the DPC routine counter, where real processing is done. All segments (inner and outer) should be changing at a rapid rate during demo operation (interrupts after each DMA transfer) indicating that the hardware is operating and interrupts are being serviced.

A demo can error out displaying an error dialog under the following circumstances:

- The board is not recognized by hardware or the operating system.
- The driver is not loaded (bitstream not PCI Express demo).
- The PCI Express link is not a x1.
- Driver can not access registers.
- Application or driver can not verify IP register IDs.
- Another demo is running.

Running Multiple DMA Demos

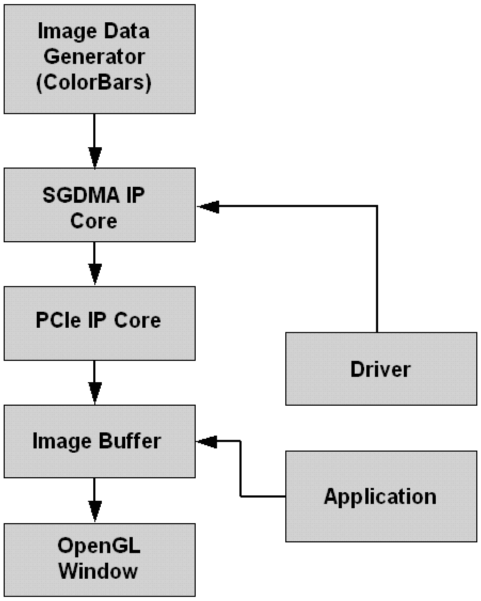
Do not run more than one demo at a time. The ImageMove and ColorBars demos cannot be run at the same time because they are mutually exclusive. Each needs DMA channels in the Scatter-Gather DMA. The driver marks channels as in-use once a demo “opens” the channels. Starting another demo will fail when it attempts to open the same channels.

PCI Express DMA ColorBars Demo

This program demonstrates the Lattice PCI Express Endpoint IP core and the Scatter-Gather DMA IP core operating on a LatticeECP3 Versa Evaluation Board. It transfers image data from the board to PC memory and software, which then displays it on the screen.

The image source is a block of IP operating as a FIFO. The IP tracks how many reads have been requested, and after eight complete rows have been read, it changes the color data provided with the next eight rows.

Figure 15. ColorBars Program Operation Flow



The image is displayed using OpenGL calls. The display rate is therefore also dependent on the OpenGL library and graphics subsystem hardware. Displaying an image is a quick way to illustrate that data has been moved. It would not be practical to display 1MB to the screen in a text dump, or save it to a file. An image provides a quick, visual way to observe a large transfer of data, and it can run continuously.

The image data is 1MB in size. Each DMA Read request is 1MB in size. After the hardware has transferred the pixel data, the API call returns and the software displays the image. This loop is repeated over and over. The data rate (frame rate) is displayed in the window title bar. The frame rate is roughly the throughput rate in Mbps (each frame = 1 MB). Frame rate is governed by the video refresh rate. Most video systems will not draw frames into video memory faster than the frame rate (waste of operations).

To see the key commands available for the ColorBars demo, refer to Table 12.

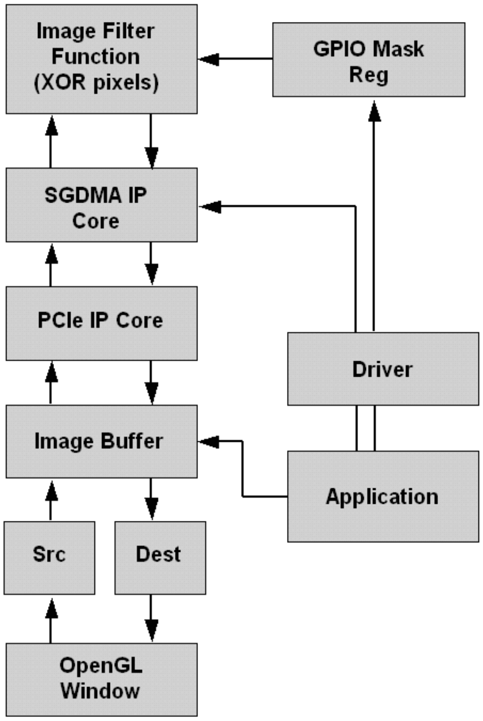
Table 12. DMA ColorBars Demo Keyboard Commands

Key Command	Description
<Esc>	Terminates program and closes the window.
<F1>	Draw blank image buffer only (do not generate data). This is the fastest rate.
<F2>	Generate ColorBars data with the software. This will usually be the slowest data rate.
<F3>	Get image data from board from DMA transfer.
<F4>	Draw a frame each second (slowly) so it can be viewed and the changes are visible.
<Space>	Pause/resume image transfer.

PCI Express DMA ImageMove Demo

This file demonstrates the Lattice PCI Express Endpoint IP core and the Scatter-Gather DMA IP core operating on a LatticeECP3 Versa Evaluation Board. It transfers image data from the PC to the board and then back to the software, which then displays a modified image on the screen.

Figure 16. ImageMove Program Operation Flow



The image is displayed using OpenGL calls. The display rate is therefore also dependent on the OpenGL library and graphics subsystem hardware. Displaying an image is a quick way to illustrate that data has been moved. An image provides a quick, visual way to observe a large transfer of data. Each image is 256KB in size.

Below is the sequence of events as this demo image undergoes processing:

1. The image source is generated by rotating the triangle shape using OpenGL transform matrix. The resulting image is displayed on the screen.

2. The source image is read from the screen into the source buffer.

3. The source buffer is sent to the Image Filter memory on the LatticeECP3 Versa Evaluation Board. The memory is only 64KB in size, so the image is sent in four chunks.

4. After a 64KB chunk is transferred to the board, the 64KB chunk is read back, with the pixels modified by the XOR function in the read path.

5. After four write/read chunks, the destination buffer contains the modified image and it is displayed on the screen.

Table 13. DMA ImageMove Demo Keyboard Commands

Key Command	Description
<Esc>	Terminates program and closes the window
<F1>	XOR filter set to 0xcc33aa55 (changes resulting display)
<F2>	XOR filter set to 0xf0f0f0f0 (changes resulting display)
<F3>	XOR filter set to 0x0f0f0f0f (changes resulting display)
<F4>	No filter is applied
<space>	Pause/remove image display

Simulation

The simulation process includes the PCI Express Bus Functional Model (BFM) as test methodology. A macro script(.do) is provided with the demos. For more on BFM support, see the Trellisys BFM user manual available with the PCI Express documentation from the LatticeECP3 Versa Development Kit web page.

The simulation process uses the **x1_active_basic.do** Aldec Active-HDL simulation script located under the **DK-ECP3-PCIE-XXX\Hardware\PCIe_x1\Versa_PCIeBasic\Simulation\aldec** directory. To ensure this script is used, follow the steps below, as illustrated in Figures 17 and 18:

1. Go to the **DK-ECP3-PCIE-XXX\Hardware\PCIe_x1\Versa_PCIeBasic\Simulation\aldec** directory.

2. Open **x1_active_basic.do** in the first line of the .do file edit the path with your location (e.g. set PROJ_DIR **C:\DK-ECP3-PCIE-XXX\Hardware\PCIe_x1\Versa_PCIeBasic\Simulation\aldec**).

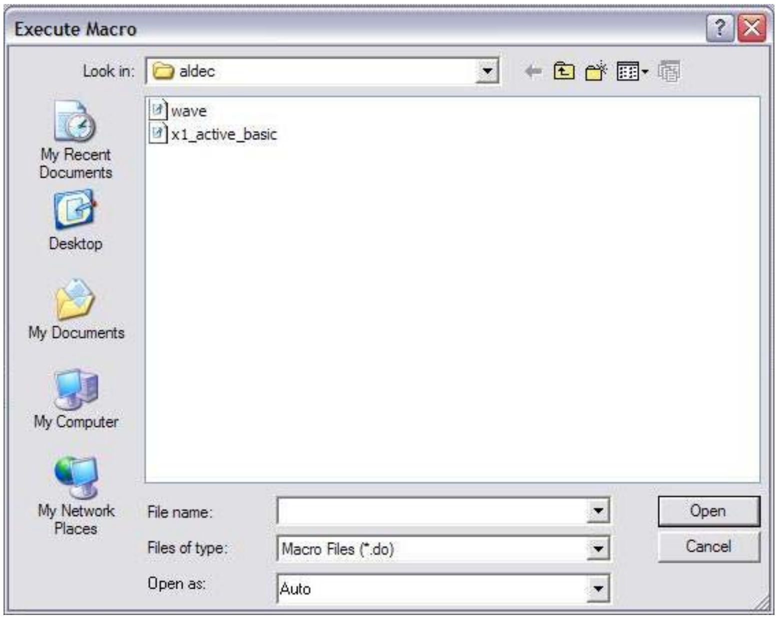
3. Save the file.

4. Open Active-HDL. Go to **Tools > Execute Macro** and select the **x1_active_basic.do** file.

The script compiles all necessary design and test bench files and runs the simulation into the Aldec Waveform window shown in Figure 18.

This procedure pertains specifically to the PCI Express Basic demo application. For the PCI Express Throughput and DMA demos, you will need to load the appropriate macro files (.do) from the respective demo folders.

Figure 17. Execute Macro File



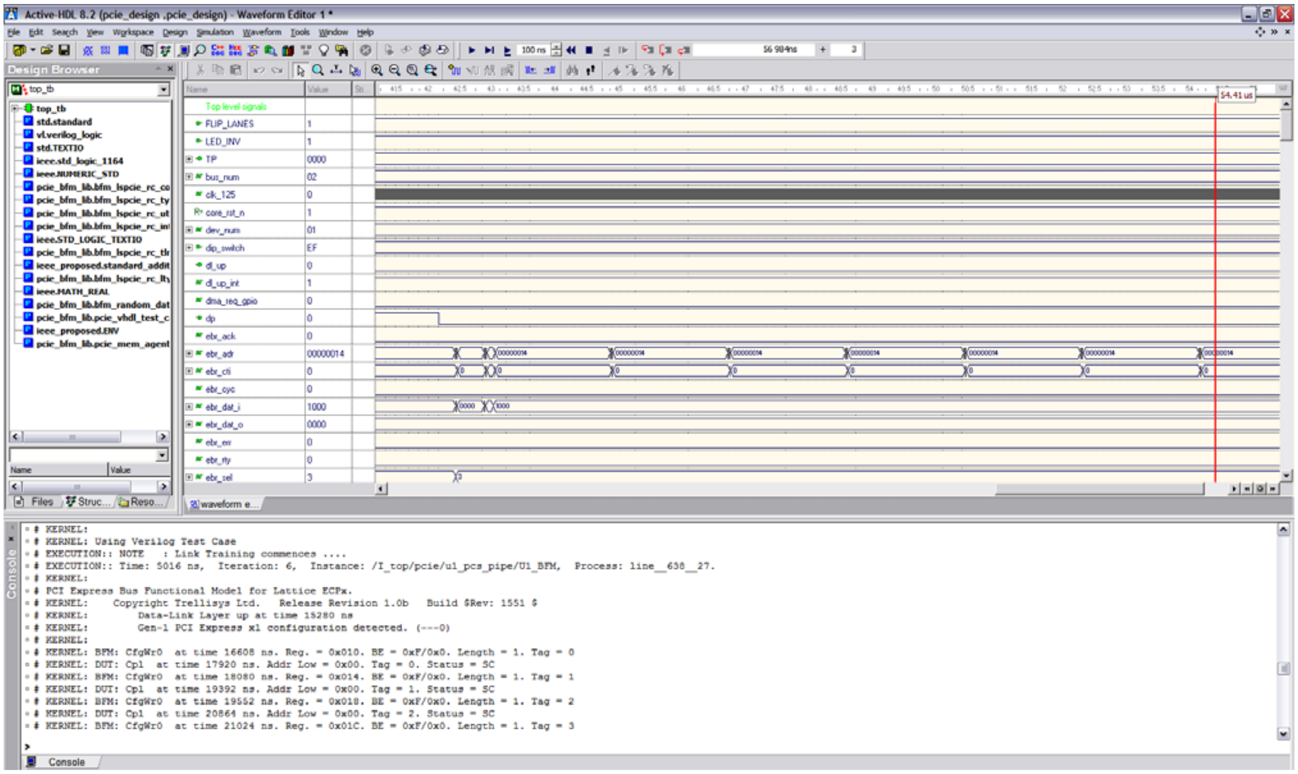
Execute Macro

Look in: aldec

wave
x1_active_basic

File name:
Files of type: Macro Files (*.do)
Open as: Auto

Figure 18. Aldec Active-HDL Simulation



```
# KERNEL:
# KERNEL: Using Verilog Test Case
# EXECUTION:: NOTE   : Link Training commences ....
# EXECUTION:: Time: 5016 ns,  Iteration: 6,  Instance: /I_top/pcie/u1_pcs_pipe/U1_BFM,  Process: line__638__27.
# KERNEL:
# PCI Express Bus Functional Model for Lattice ECPx.
# KERNEL:    Copyright Trellisys Ltd.   Release Revision 1.0b   Build $Rev: 1551 $
# KERNEL:         Data-Link Layer up at time 15280 ns
# KERNEL:         Gen-1 PCI Express x1 configuration detected. (---0)
# KERNEL:
# KERNEL: BFM: CfgWr0  at time 16608 ns. Reg. = 0x010. BE = 0xF/0x0. Length = 1. Tag = 0
# KERNEL: DUT: Cpl  at time 17920 ns. Addr Low = 0x00. Tag = 0. Status = SC
# KERNEL: BFM: CfgWr0  at time 18080 ns. Reg. = 0x014. BE = 0xF/0x0. Length = 1. Tag = 1
# KERNEL: DUT: Cpl  at time 19392 ns. Addr Low = 0x00. Tag = 1. Status = SC
# KERNEL: BFM: CfgWr0  at time 19552 ns. Reg. = 0x018. BE = 0xF/0x0. Length = 1. Tag = 2
# KERNEL: DUT: Cpl  at time 20864 ns. Addr Low = 0x00. Tag = 2. Status = SC
# KERNEL: BFM: CfgWr0  at time 21024 ns. Reg. = 0x01C. BE = 0xF/0x0. Length = 1. Tag = 3
>
```

Technical Support Assistance

Hotline: 1-800-LATTICE (North America)
+1-503-268-8001 (Outside North America)
e-mail: techsupport@latticesemi.com
Internet: www.latticesemi.com

Revision History

Date	Version	Change Summary
April 2011	01.0	Initial release.
March 2012	01.1	Updated Scatter-Gather DMA ImageMove Demo for DDR3.
		Simulation text section – Added information about Bus Functional Model support.
		Updated Execute Macro File screen shot.
		Updated Aldec Active-HDL Simulation screen shot.

© 2012 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal. All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

Appendix A. Installing and Running the Evaluation Board in Linux

This appendix describes the LatticeECP3 Versa Evaluation Board installation and all of the necessary installation to run the PCI Express software demo in the Linux environment. You should have already gone through the Getting Started with the Development Kit section of this document which includes important Linux Diamond instructions.

Once you have successfully installed and opened the demo in Linux, you should refer back to the Running the PCI Express Basic Demo Software section of this document for instructions on using the demo. The demo graphical user interface will be the same in Linux as it appears in Windows for PC.

Installing the Demos on Linux

This section provides Linux installation instructions for the LatticeECP3 Versa Evaluation Board. Refer back to the Installing Diamond or ispLEVER (Windows/Linux) section of this document if you have not properly installed a compatible version of the ispLEVER software.

To install the development kit on your Linux machine:

1. Specify a path location for installing the kit, create a directory for Lattice development kits, and change to that directory.

   ```
   mkdir <path>/<install_path>
   cd <path>/<install_path>
   ```

2. Using a web browser, go to the LatticeECP3 Versa Evaluation Board Kit Tar (Linux) the page, and download the Linux tar.gz file: **DK-ECP3-PCIE-XXX.tar.gz**.

3. Unzip and untar the **DK-ECP3-PCIE-XXX.tar.gz** file to your current working directory.

After you untar the tar file, the directory structure is as follows:

```
DK-ECP3-PCIE-XXX
   Demonstration/
   Hardware/
   Software/
```

The kit directory structure for Linux is the same as Windows with the exception of the top-level Windows default directory **Lattice_DevKits** which can be user-defined for either platform. Figure 1 illustrates the structure and describes the contents of the directories.

Installing the PCI Express Demo Package

This section discusses the installation of the demo package, including software installation and the board hardware. Please read this section completely before attempting to install the package so that you understand the steps involved and how they apply to your situation.

The Lattice PCI Express demo package is released as a compressed Linux tar file. The package includes the Linux evaluation board drivers, FPGA bitstreams, Java GUI demo applications, and all demo source code. The file must first be unzipped/untarred and then extracted into the final destination directory. A script can then be run to install the device drivers and demo files. Once this is complete, you are ready to run the demo executables.

Before You Begin

This demo assumes that you are familiar with basic PCI Express technology and are comfortable installing new hardware and software packages on Linux. Some experience in these areas is helpful when installing the board and software.

Notes on Installing Linux Drivers

The Linux demonstration components and drivers supplied with this kit are built for Red Hat Workstation 4.0, Update 4 (32-bit kernel 2.6.9-42 i686) and successfully verified on this kernel version. Please be aware that the files provided in the <kit_dir>/Demonstration/Software path are only tested for compatibility with this Linux kernel version. Results on other kernel distributions or 64-bit kernel versions have not been fully validated.

You should run the following command initially to determine if you have to rebuild the driver binaries:

```
uname -r -i -m
```

If this command does not show the following information line output:

```
2.6.9-42.ELsmp i686 i386
```

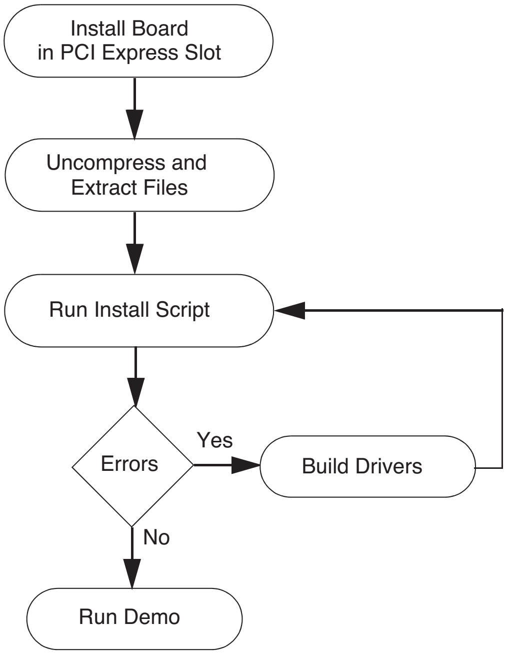
then you will probably have to rebuild the drivers. If you have a 64-bit version of Linux then you will probably also need to rebuild the demos as well. Generally, if you do not have the specific kernel version employed in this kit, you may need to rebuild the drivers (and possibly demo applications) from the source as described in the Building Demo Binaries from Source Code section of this document. The kernel header files must be installed on your system and you should be familiar with building kernel drivers from source code. Installing the driver requires the root password.

If you need more information on this topic, a good resource is *Linux Device Drivers*, by A. Rubini and J. Corbet. This book details all aspects of Linux driver development.

Installation Overview

The following steps are taken to install and run the demo.

Figure 19. Linux Demo Installation Flow



Installing LatticeECP3 Versa Evaluation Board Hardware

After board setup, you can install the hardware. You must have root privileges to perform this installation.

To install the LatticeECP3 Versa Evaluation Board on Linux:

1. Shut down your Linux operating system, turn off the machine and unplug the power cord.

IMPORTANT: This step is necessary because power supplies have standby voltages that are present even when your machine's power light and fan are turned off. Unplugging your machine is the safest way to ensure the board will not be "hot-swapped".

2. Locate an available PCI Express slot. The board can be installed in any slot that is larger than the finger edges in use, x1, x4, or x16.
3. Ensure that the board is not connected any external power supply before proceeding.
4. Using ESD precautions, install the LatticeECP3 Versa Evaluation Board in the PCI Express slot in the x1 position.
5. Power-on your machine and observe that it boots normally to the login screen.
6. Log in as a user with root privileges.

   Caution: Lattice is not liable for any loss of date or damages that may result from the installation of the hardware and execution of the kit demo software tools. Do not install and operate on mission-critical systems.

Demo Package Software Installation

This section describes how to install the demo applications. You must have the board installed or the demos will not run. The drivers will not have anything to open, and the demo applications will not have anything to communicate with.

To install the demos and drivers:

1. Change to the Demonstration/ directory.
2. Execute the driver installation script to install the kernel object driver files and demos using the following command.

   ```
   ./install.sh
   ```

   Note that if you get an error message such as, "insmod error inserting... Invalid module format," refer to the section Building Demo Binaries from Source Code section of this document.
3. (Optional) Verify the driver installed by issuing any of the following commands:

   a. `ls -l /dev/lsc*` (displays a list of installed Lattice evaluation boards)

   b. `cat /proc/modules | grep lsc*` (verifies drivers are installed)

   c. `cat /proc/driver/lsc*` (displays information about the boards installed and what BAR resources they have been assigned)

   d. `dmesg` shows messages from driver installing and finding the LatticeECP3 Versa Evaluation Board

To run the demos:

- Click on the appropriate Quick Launch icon on the desktop for the desired demo. Note that the LatticeECP3 Versa Evaluation Board must be installed with corresponding bitstream.
- Or, run the desired demo from the command line in the appropriate Demonstration directory.

```
./rundemo.sh
```

After starting the demo GUI application, go back to the Running the PCI Express Basic Demo section of this document and follow the instructions. If the demo binaries will not run, see the Building Demo Binaries from Source Code section in this appendix to produce the demo binaries.

Building Demo Binaries from Source Code

In the event that the binaries in this release are incompatible with the installed Linux distribution, the binaries can be built from the source code.

Prerequisites:

- GCC and other compiler tools
- Kernel source header files

To build the demo binaries from source code:

1. Change to the Software directory.

```
cd <PCIeDemo_dir>/Software
```

2. To ensure your environment is free of previously compiled object files that might conflict with any newly built ones, run the following command:

```
make clean
```

3. Build the source files into the executable demos.

```
make demos
```

4. Build the drivers.

```
make drivers
```

5. Install the files with the install script in the Demonstration directory.

```
cd ../Demonstration
./install.sh
```

6. If previous versions were installed, then first run the remove script.

```
./remove.sh
```

Installing a Java Runtime Environment

If you are working in a kernel other than the one tested and supplied with this kit, you may have to install a compatible JRE to execute the demo software.

1. Go to the Sun Microsystems web site and download the JRE 5 version from the Java web site.
2. Choose a Linux self-extracting file compatible with version jre-1_5_0_14-linux for your machine. Do not choose the RPM. Read the installation instructions located there. This file must match the system kernel and a different file may be required.
3. Save the JRE installation file into the <demo>/bin directory (or install globally on the machine).
4. Change permissions on the jre*.bin file.

```
chmod 777
```

5. Run the JRE installation file to install.

```
./jre*.bin
```

6. Tell the demo GUI where to find the newly installed Java JRE:

```
        export DEMO_JRE=”/usr/local./jre1.5.0_14”
```

Replace the directory argument for DEMO_JRE with specific directory if different than one shown above.

Uninstalling the PCI Express Demo Software

If you want to remove the software from your system (i.e., to perform a clean installation of a new version), simply run the remove.sh script in the Demonstration/ directory and delete the directory that the development kit was extracted into.

How to Build the FPGA Demo Designs Using Linux

The Diamond implementation flow for Linux is the same as Windows. Refer to “Rebuilding the PCI Express Basic Demo Design” on page 14 for more information.

Appendix B. Troubleshooting

This appendix outlines some debug procedures to follow when experiencing trouble installing or running a demo on a Windows PC.

Troubleshooting Demo Software Installation

- The most likely installation issue that may arise for the kit demo software will be related to permissions. Depending upon the system security policies, you may need to have administrator privileges to install into certain directories, for example, the Program Files directory in Windows.

Troubleshooting Driver Installation

- The board must be connected to the PC and recognized by Windows for the driver to be successfully installed. If you do not see the “Found New Hardware” message when logging in after installing the board, check the board LEDs. Try a different PCI Express slot.
- Make sure you specify the search location for the driver during installation. Specify that Windows should install from the **Demonstation\<demo_name>\Drivers** directory.
- You must have Administrator privileges to install device driver files.

Troubleshooting Demo Operation

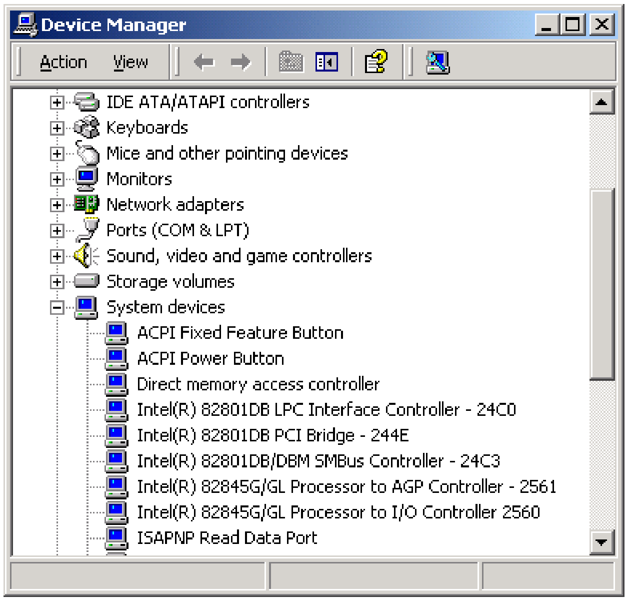
- The LatticeECP3 Versa Evaluation Board must be installed in the PC and recognized by Windows for the driver to be successfully installed/loaded. The driver must be loaded by Windows to run the demo. Verify that Windows sees the board and has loaded a driver for it. See Troubleshooting the LatticeECP3 Versa Evaluation Board section of this appendix.
- If the GUI displays the error message, “ERROR LOADING LIBRARY:Cpp_Jni - running in View Only mode” when executed, then the driver was not found or loaded. There are two causes:
  - The driver was never loaded (or the board is not installed)
  - The board failed to be detected by Windows.

In either case, the board needs to be installed and seen by Windows and the LSC_PCIe driver needs to be associated with the hardware.

Using Device Manager to Debug Installation

Use Device Manager to get basic information on the hardware you have installed. To access Device Manager, right-click on the **My Computer** desktop icon and select **Properties**. In the System Properties dialog, select the **Hardware** tab and **Device Manager** button.

Figure 20. Device Manager



IDE ATA/ATAPI controllers
Keyboards
Mice and other pointing devices
Monitors
Network adapters
Ports (COM & LPT)
Sound, video and game controllers
Storage volumes
System devices
ACPI Fixed Feature Button
ACPI Power Button
Direct memory access controller
Intel(R) 82801DB LPC Interface Controller - 24C0
Intel(R) 82801DB PCI Bridge - 244E
Intel(R) 82801DB/DBM SMBus Controller - 24C3
Intel(R) 82845G/GL Processor to AGP Controller - 2561
Intel(R) 82845G/GL Processor to I/O Controller 2560
ISAPNP Read Data Port

The Device Manager provides the same basic set of software driver information as in the Computer Management window. The Hardware Wizard allows you to install and remove drivers. You must to have administrator privileges to run the Hardware Wizard and install/remove drivers. Again, the most useful thing is to verify that the lscpcie and lscvpci drivers (if enabled) have been installed.