

Introduction

The *Altera® PCI Express to External Memory Reference Design* provides a sample interface between the Altera PCI Express MegaCore® function and a 64-bit external memory. Altera offers this reference design to demonstrate the operation of the PCI Express MegaCore function and either a DDR2 or DDR3 SDRAM memory controller. The reference design has the following features:

- Supports PCI Express (PCIe) endpoint DMA read and write transactions
- Uses the PCI Express hard IP MegaCore function
- Uses the High-Performance SDRAM Controller MegaCore function for DDR2
- Uses the High-Performance SDRAM Controller II MegaCore function for DDR3
- Uses either a Arria® II GX or Stratix® IV GX device with internal transceivers

This document covers the following topics:

- [Reference Design Overview](#)
- [Using the Reference Design](#)

Reference Design Overview

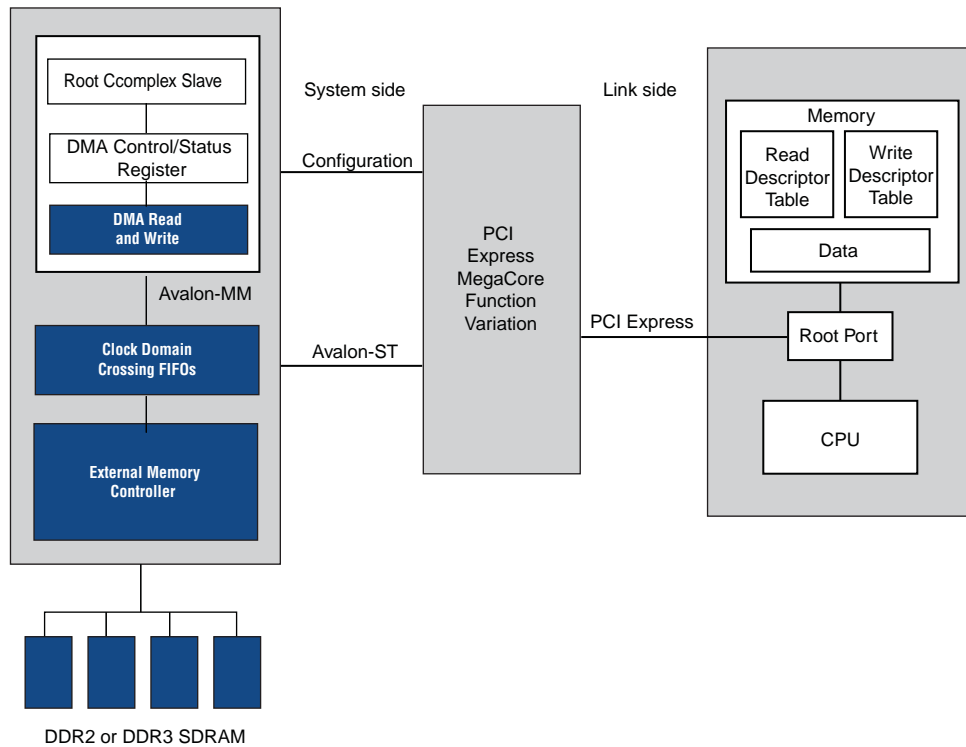
The reference design connects the Altera PCI Express MegaCore function to external memory using the reference design interface circuitry. The design runs on Altera's Arria II GX FPGA Development Kit or Stratix IV GX FPGA Development Kit. Both kits include a PCI Express development board. Altera also provides a software driver, programming information, and a GUI to run the application.

The Altera PCI Express to external memory reference design interfaces to the system side of the Altera PCI Express MegaCore function. (Refer to [Figure 1 on page 2.](#)) The external side of the PCI Express MegaCore function forms half of the PCI Express link. The memory controller accesses external memory. The PCI Express MegaCore function generally operates as a PCIe master, or initiator. When the PCI Express MegaCore function operates as a PCIe master, the DMA engine initiates the transaction, monitors the status, and manages the progress of the data transfers.

This reference design is very similar to the chaining DMA design example that you automatically generate when you create a PCI Express Megacore function using the MegaWizard Plug-In Manager.



Refer to “[Chapter 7 Testbench and Design Example](#)” in the *PCI Express Compiler MegaCore Function User Guide* for a detailed description of the chaining DMA design example.

Figure 1. Block Diagram for the PCI Express to External Memory

The difference between the chaining DMA design example and the PCI Express to external memory reference design is that the reference design uses an external memory to store data instead of internal memory in the FPGA. Consequently, the files in the chaining DMA design example that define the DMA controller and memory accesses are modified. In addition, the top-level Verilog HDL file, `<name>_example_chaining_top.v`, points to the external memory version of these files. Table 1 lists the files in the DMA design example and PCI Express to external memory reference design that access memory.

Table 1. Files that Access Memory

File Name in Chaining DMA Design Example	File Name in PCI Express to DDR3 SDRAM Reference Design
<code>altpcierd_write_dma_requester_128.v</code>	<code>altpcierd_write_dma_requester_128_ddr.v</code>
<code>altpcierd_dma_dt.v</code>	<code>altpcierd_dma_dt_ddr.v</code>
<code>altpcierd_dma_prg_reg.v</code>	<code>altpcierd_dma_prg_reg_ddr.v</code>
<code>altpcierd_cdma_app_icm.v</code>	<code>altpcierd_cdma_app_icm_ddr.v</code>
<code>altpcierd_example_app_chaining.v</code>	<code>altpcierd_example_app_chaining_ddr.v</code>
<code><name>_example_chaining_pipen1b.v</code>	<code><name>_example_chaining_pipen1b_ddr.v</code>

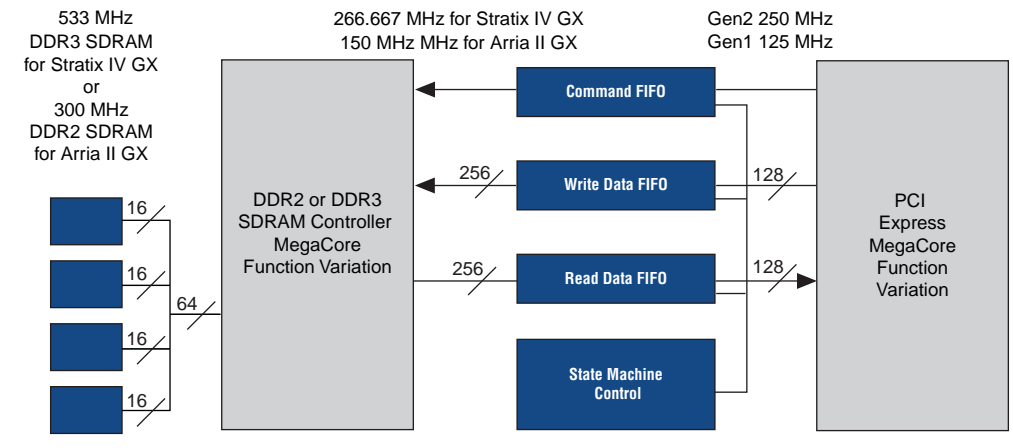
Figure 1 highlights the modules that include these files in dark blue.

The following sections provide an overview of modules in this reference design that differ from the comparable modules in the chaining DMA design example that you automatically generate when you create a PCI Express MegaCore function.

PCIe to Memory Interface Block

The memory interface block interfaces to DDR3 memory on the Stratix IV GX development board and DDR2 memory on the Arria II GX development board. [Figure 2](#) illustrates the PCI Express to external memory subsystem.

Figure 2. Datapath from PCI Express MegaCore Function to External Memory



On the Stratix IV GX development board, DDR3 memory transfers 64 bits on each clock edge with a burst size of 8 for a total of 512 bits per burst transfer. The local interface to the DDR3 High Performance SDRAM Controller II is 256 bits wide and runs at half rate, or 266.667 MHz, so that the local interface must complete two transfers to match the external memory transfer size of 512 bits. On the Arria II GX development board, the DDR2 memory transfers 64 bits on each clock edge with a burst size of 4 for a total of 256 bits per burst transfer. The local interface of the DDR2 high performance controller is 256 bits wide and runs at half rate, or 150MHz.

The reference design uses three FIFOs to pass data between the external memory interface clock domain and the PCI Express clock domain. The three separate FIFOs handle access commands, write data, and read data. The FIFOs are instantiated in a block called `DDR2_Fifo_Interfaces.vhd` or `DDR3_Fifo_Interfaces.vhd` for DDR2 and DDR3 respectively. This file also defines a state machine to control the timing of the data and commands to the memory controller's local interface and monitor the status signals from it.

The following pseudo code summarizes the algorithm that the state machine implements:

1. Wait for the command FIFO to not be empty.
2. Decode the read or write command.
3. For reads, ensure the memory controller is ready to receive a command, then send the read.

4. Writes are different for the Arria II GX and Stratix IV GX development boards.
 - a. For the Arria II GX development board, retrieve 2 consecutive words before sending a 256-bit word to the memory controller. The two PCI Express write commands must be to consecutive addresses. These constraints are built into the state machine.
 - b. For the Stratix IV GX development board, because the PCI Express word size is 128 bits, retrieve 4 consecutive words before sending two, 256-bit words on consecutive clock cycles. To match throughput, four write commands are required by the PCIe MegaCore function for each external memory write in this particular design application. The write commands must be to consecutive addresses. These constraints are built into the state machine. When all data is collected and the memory controller is ready, the state machine writes the data to the memory controller and then decodes the next command.



There are several alternatives when designing hardware that interfaces to external memory. For more information refer to *Volume 3: Implementing Altera Memory Interface IP* in the *External Memory Interface Handbook*.

External Memory

The following sections discuss the performance of the DDR2 and DDR3 memories on the Stratix IV GX and Arria II GX development boards.

DDR2 SDRAM IP Block

The Arria II GX FPGA Development Kit uses a 64-bit Micron DDR2 SODIMM MT8HTF1284H. The entire interface runs at the maximum supported speed of 300 MHz to ensure that the DDR2 memory system does not limit the bandwidth of the reference design.

Example 1 gives the bandwidth calculations for the PCI Express and DDR2 SDRAM interfaces.

Example 1. Bandwidth Calculations

```
Gen1 PCI Express - 128 bits @125 MHz = 16 Gbps (minus some overhead)
DDR2 - 64 bits @ 300 MHz = 38.4 Gbps (minus overhead due to controller and interface
    inefficiencies)
```

As these calculations indicate, by running the DDR2 interface at 300 MHz, the available bandwidth easily accommodates the bandwidth required for Gen1 operation. The reference design uses Altera's DDR2 high performance controller which consists of a memory PHY (ALTMEMPHY) and controller code.

DDR3 SDRAM IP Block

The Stratix IV GX FPGA Development Kit uses a bank of four, 16-bit Micron MT41J64M16 DDR3 components to create a 64-bit memory interface. The interface uses fly-by topology as specified in *JEDEC Standard for DDR3 SDRAM, JES79-3C*. The entire interface runs at the maximum supported speed of 533 MHz to ensure that the DDR3 memory system does not limit the bandwidth of the reference design.

Example 2 gives the bandwidth calculations for the PCI Express and DDR3 SDRAM interfaces.

Example 2. Bandwidth Calculations

Gen2 PCI Express - 128 bits @250 MHz = 32 Gbps (minus some overhead)
Gen1 PCI Express - 128 bits @125 MHz = 16 Gbps (minus some overhead)
DDR3 - 64 bits @ 533 MHz = 68.4 Gbps (minus overhead due to controller and interface inefficiencies)

As these calculations indicate, by running the DDR3 interface at 533 MHz, the available bandwidth easily accommodates the bandwidth required for Gen2 operation. The reference design uses Altera's High-Performance SDRAM Controller II MegaCore function which consists of a memory PHY (ALTMEMPHY) and controller code.



For more information on using external memories with Altera FPGAs, refer to *Chapter 3: Using DDR3 SDRAM in Stratix III and Stratix IV Devices* in *DDR, DDR2, and DDR3 SDRAM Design Tutorials* in Volume of the *External Memory Interface Handbook*. For more information on the high performance DDR3 or DD2 SDRAM memory controllers, refer to the *DDR and DDR2 High-Performance Controllers and ALTMEMPHY IP User Guide* or the *DDR3 High Performance Controller and ALTMEMPHY IP User Guide* and in Volume 3 of the *External Memory Interface Handbook*.

Using the Reference Design

This section describes how to install the reference design and provides instructions for running the software application. The following sections are included:

- [Hardware Requirements](#)
- [Software Requirements](#)
- [Software Installation](#)
- [Hardware Installation](#)
- [Running the Software Application](#)
- [Running the Simulation](#)

Hardware Requirements

The reference design requires the following hardware:

- The Arria II GX or Stratix IV GX FPGA Development Kit.
- A computer running 32-bit Windows XP with an x8/x4/x1 PCI Express slot for the Arria II GX FPGA or Stratix IV GX FPGA development board. The software application and hardware are installed on this computer, referred to as computer #1 in this document.
- A computer with the Quartus II software for downloading FPGA programming files to the development board, referred to as computer #2 in this document.
- A USB cable or other Altera download cable.

Software Requirements

To run the reference design application requires installation of the following software:

- Reference design software installed on computer #1.
- PCI Express to external memory reference design package, available as a downloadable compressed file.



For more information refer to the [PCI Express to External Memory Reference Design](#) product page.

- The Quartus II software version 9.1 running on computer #2.

Software Installation

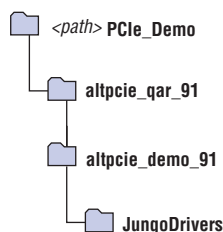
You must have Administrator privileges to install the software application.

The software application only runs on 32-bit Windows XP and includes the WinDriver software from Jungo as part of the application.

Perform the following steps to install the software application and Windows drivers:

1. Download the PCI Express to external memory reference design software to computer #1, and extract the compressed files. [Figure 3](#) shows the directory structure.

Figure 3. Directory Structure



2. Before plugging in the PCI Express card, copy the **altpcie_demo_91** directory to computer #1.
3. In the **JungoDrivers** directory, double-click on **install.bat** to install the Windows XP driver for this application.
4. Run **altpcie_demo.exe** from the **altpcie_demo_91** directory to start the software application.

Hardware Installation

If you are using the Stratix IV GX card, you must check the settings on an eight-position dip switch which controls the PCI Express mode of operation on computer #1 before plugging it in. [Figure 4](#) highlights this component. The right-most position of this dip switch is used to change between normal operation and PCI Express compliance base board (CBB) testing. To run the software included in this application note, this switch must be in the **off** position. When it is in the **on** position, you can use the reset switch labeled PB1 to cycle through various modes required for CBB testing. (The dip switch labels the **on** side on the switch.)


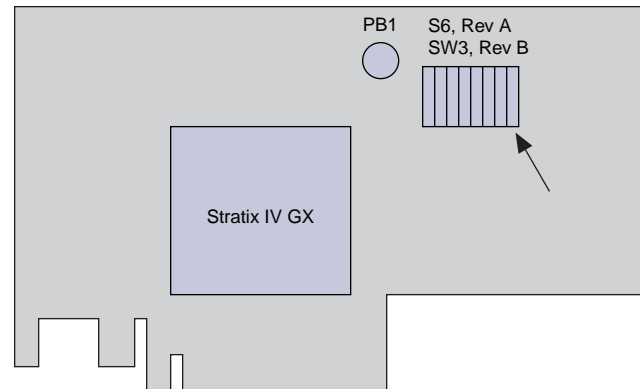
-  The top-level RTL file has been modified to enable CBB testing. If you regenerate the MegaCore function, you may overwrite this top-level file and disable the CBB testing capability.

Figure 4. Location of Components that Control PCI Express Mode of Operation



Perform the following steps to install the hardware.

1. Power down computer #1 and plug the development card into the PCI Express slot. Depending on your hardware, a PCI Express lane converter may be required.
2. Both Development Kits include integrated USB-Blaster circuitry for FPGA programming. However, for the host computer and development board to communicate, you must install the USB-Blaster on the host computer.

To download the USB-Blaster driver, go to the Altera support site at www.altera.com/support/software/drivers/dri-index.html. For installation instructions, go to www.altera.com/support/software/drivers/usb-blaster/dri-usb-blaster-xp.html.

3. Program the FPGA with the reference design using the Quartus II software on computer #2 and an Altera USB-Blaster cable (or other download cable) connection between computer #2 and the development board on computer #1.

Connecting the USB-Blaster Cable

To connect the USB-Blaster cable, complete the following steps:

1. Connect one end of the USB cable the USB port.
2. Connect the other end of the cable to the USB port on the computer running the Quartus II software on computer #2.

Programming with the .sof File

To program the board with the .sof file provided, interrupt the boot sequence on computer #1 to bring up the BIOS System Setup interface. (Pressing the F2 key interrupts the boot sequence on many Windows PCs.)

Perform the following steps to program the FPGA with the .sof file:

1. Start the Quartus II programmer on computer #2.
2. Click **Hardware Setup** and select the **USB Blaster**. Click **Close**.

3. In the Quartus II Programmer, click **Auto Detect** to list the devices attached to the JTAG chain on the development board.
4. Right-click the Arria II GX (EP2AGX125EF35) or Stratix IV GX (EP4SGX230) device and click **Change File**. Select the path to the appropriate **.sof** file.
5. Turn on the **Program/Configure** option for the added file.
6. Click **Start** to download the selected file to the Arria II GX or Stratix IV GX device. The device is configured when the **Progress** bar reaches 100%.
7. On computer #1 exit the BIOS System Setup or boot manager interface.
8. On computer #1, press **Ctrl+Alt+Delete** to start a soft reboot.
9. The operating system detects a new hardware device and displays the Found New Hardware Wizard. In the wizard, select **Install the software automatically (Recommended)**. Click **Next**.
10. Click **Finish** to close the wizard.

Running the Software Application

Complete the following steps to run the software application:

1. Double-click on the application **altpcie_demo.exe** in the **altpcie_demo_91** directory.
2. The application reports the board type, the number of active lanes, the maximum read request size, and the maximum payload size.

The software GUI has the following control fields:

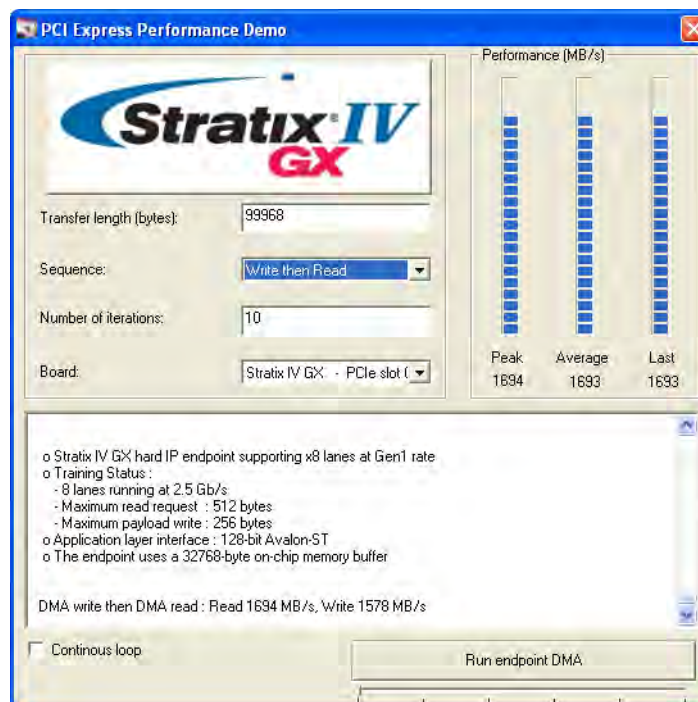
- **Transfer length**—Specifies the transfer length in bytes
 - **Sequence**—Controls the sequence for data transfer or addressing
 - **Number of iterations**—Controls the number of iterations for the data transfer
 - **Board**—Specifies the development board for the software application
 - **Continuous loop**—When this option is turned on, the application performs the transfer continuously
3. Set the **Transfer length** to 99,968 bytes and the **Sequence** to **Write then Read**, Click **Run**.

When set for **Write then Read**, the software programs the DMA registers in the FPGA to transfer data from the FPGA to the external memory. The performance bars report the peak, average, and last throughput. The average throughput is computed across all the iterations.

4. You can use the GUI to change the **Transfer length** and **Sequence** and repeat the test.

Figure 5 illustrates the GUI for the Stratix IV GX device.

Figure 5. Write then Read Options



In addition to the parameter settings to control the chaining DMA, the GUI includes additional commands that you can use to obtain configuration information about the device and board and to perform root port reads and writes. Table 2 outlines all of the available commands. The position of the slider control changes the command.

Table 2. PCI Express Performance Demo GUI Commands and Options (Note 1)

Command	Options	Description
Run endpoint DMA (Figure 5)	Write only Read only Read then write Write then read	Writes transfer data from the FPGA to system memory. Reads transfer data from system memory to the FPGA.
Scan the endpoint configuration space registers (Figure 6)	Type 0 Configuration PCI Express capability MSI capability Power management capability	Reports the byte address offset, value and a description of the selected register set.
Scan the current PCI Express board settings (Figure 7)	—	Reports the configuration settings of the development board.
Scan the motherboard PCI bus	—	Reports the vendor ID, device ID, slot, bus, and function numbers for all devices on the motherboard's PCI bus.

Notes to Table 2:

- (1) This software application is a different version of the software application used in [Application Note 456: PCI Express High-Performance Reference Design](#). To run this reference design, you must use the software version included with this reference design.

Figure 6 illustrates the output of the Scan the current PCI Express board settings command for the Stratix IV GX device.

Figure 6. Scan the Current PCI Express Board Settings

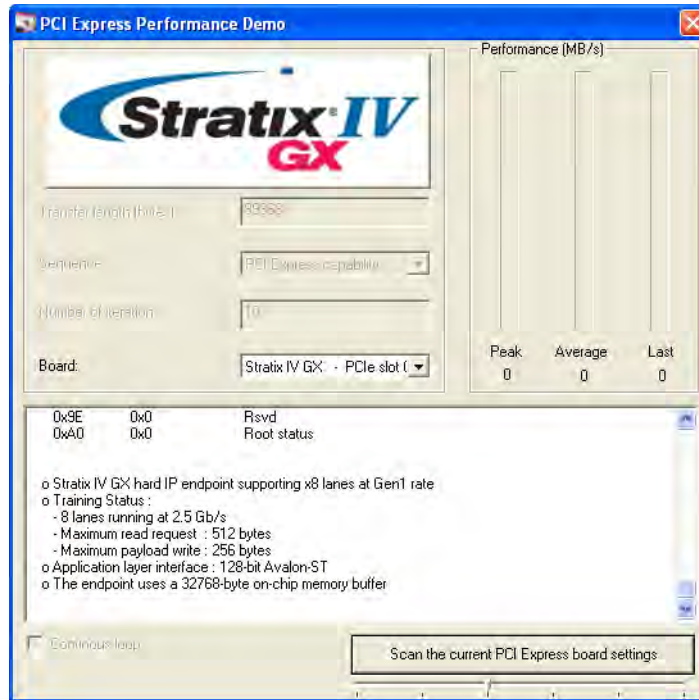


Figure 7 illustrates the output of the Scan the motherboard PCI bus command for the Stratix IV GX device.

Figure 7. Scan the Motherboard PCI Express Bus

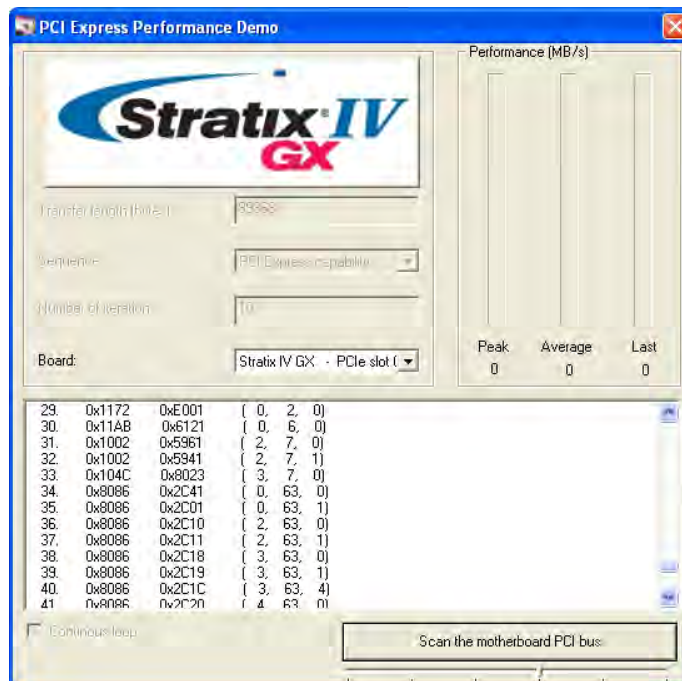


Table 3 gives the performance of the Stratix IV GX and Arria II GX FPGA development boards using this reference design with a motherboard that includes the Intel X58 Chipset. The table shows the average throughput for a transfer size of 99,968 Bytes and 20 iterations with a maximum write payload size of 256 bytes, a maximum read request size of 512 bytes, a read completion size of 256 bytes using a clock multiplier unit (CMU) clock.

Table 3. Stratix IV GX and Arria II GX Performance - Intel X58 Chipset

Configuration	DMA Reads (MB/s)	DMA Writes (MB/s)
Stratix IV GX-Gen1 x8 128-bit interface	1613	1695
Arria II GX-Gen1 x8 128-bit interface	1221	1584

Running the Simulation

To simulate this PCI Express reference design for either the Arria II GX or Stratix IV GX FPGA, complete the following steps:

1. Download **altpcie_zip_91.zip** from the [PCI Express to External Memory Reference Design](#) product page.
2. Unzip the file **altpcie_zip_91.zip** to your project directory. You should then see the following directory:
`<install_dir>\altpcie_zip_91\top_examples\chaining_dma\testbench.`
3. Choose **Programs > Altera > ModelSim <version>** (Windows Start menu) to run the ModelSim® software.
4. Select **File**, then **Change Directory...**
5. Browse to
`<install_dir>\altpcie_zip_91\top_examples\chaining_dma\testbench.`
6. Click **OK**.
7. At the Modelsim> command prompt, type `do runt_b.do` ↵
 This script compiles the all of the Verilog modules necessary to run the simulation and loads the simulation files.
8. To set up the wave viewer, type `do wave.do` ↵
9. To run the simulation, type `run -all` ↵



The simulation begins by initializing simulation models for both the PCI Express hard IP MegaCore function, the external memory controller, and external memory. Then, it runs the DMA tests.

Example 3 shows excerpts from the transcript of a successful simulation.

Example 3. Transcript from Successful Simulation

```
#INFO:26440ns DMA: Read
#INFO:26440ns TASK:dma_rd_test
#INFO:26440ns TASK:dma_set_rd_desc_data
#INFO:26440ns TASK:dma_set_msi READ
#INFO:26440ns Message Signaled Interrupt Configuration
#INFO:26440ns msi_address (RC memory)= 0x07F0
#INFO:7040ns msi_control_register = 0x0084
#INFO:29440ns msi_expected = 0xB0FC
#INFO:29440ns msi_capabilities address = 0x0050
#INFO:29440ns multi_message_enable = 0x0002
#INFO:29440ns msi_number = 0000
#INFO:29440ns msi_traffic_class = 0000
#INFO:29440ns TASK:dma_set_header READ
#INFO:29440ns Writing Descriptor header
#INFO:29480ns data content of the DT header
#INFO:29512ns TASK:msi_poll Polling MSI Address:07F0---> Data:FADE.....
#INFO:29696ns TASK:rcmem_poll Polling RC Address0000090C current data (0000FADE)
expected data (00000002)
#INFO:32296ns TASK:rcmem_poll Polling RC Address0000090C current data (00000000)
expected data (00000002)
#INFO:47096ns TASK:rcmem_poll Polling RC Address0000090C current data (00000002)
expected data (00000002)
#INFO:47096ns TASK:rcmem_poll ---> Received Expected Data (00000002)
#INFO:47144ns TASK:msi_poll Received DMA Read MSI(0000) : B0FC
#INFO:47152ns Completed DMA Read
#INFO:47152ns TASK:chained_dma_test
#INFO:47152ns DMA: Write
#INFO:47152ns TASK:dma_wr_test
#INFO:47152ns TASK:dma_set_wr_desc_data
#INFO:47152ns TASK:dma_set_msi WRITE
#INFO:47152ns Message Signaled Interrupt Configuration
#INFO:47152ns msi_address (RC memory)= 0x07F0
#INFO:47760ns msi_control_register = 0x00A5
#INFO:50160ns msi_expected = 0xB0FD
#INFO:50160ns msi_capabilities address = 0x0050
#INFO:50160ns multi_message_enable = 0x0002
#INFO:50160ns msi_number = 0001
#INFO:50160ns msi_traffic_class = 0000
#INFO:50200ns Shared Memory Data Display:
#INFO:50200ns Address Data
#INFO:50200ns 00000800 10100003 00000000 00000800 CAFEFAD
#INFO:50200ns TASK:dma_set_rclast
#INFO:50200ns Start WRITE DMA : RC issues MWr (RCLast=0002)
#INFO:50228ns TASK:msi_poll Polling MSI Address:07F0---> Data:FADE.....
#INFO:50412ns TASK:rcmem_poll Polling RC Address0000080C current data (0000FADE)
expected data (00000002)
#INFO:52412ns TASK:rcmem_poll Polling RC Address0000080C current data (00000000)
expected data (00000002)
#INFO:62132ns TASK:msi_poll Received DMA Write MSI(0000) : B0FD
#INFO:62212ns TASK:rcmem_poll Polling RC Address0000080C current data (00000002)
expected data (00000002)
#INFO:62212ns TASK:rcmem_poll ---> Received Expected Data (00000002)
#INFO:62220ns Completed DMA Write
#INFO:62220ns TASK:check_dma_data
#INFO:62220ns Passed : 4096 identical dwords.
```

```
#INFO:62220ns TASK:downstream_loop
#INFO:63116ns Passed: 0004 same bytes in BFM mem addr 0x00000040 and 0x00000840
#INFO:63988ns Passed: 0008 same bytes in BFM mem addr 0x00000040 and 0x00000840
#INFO:64868ns Passed: 0012 same bytes in BFM mem addr 0x00000040 and 0x00000840
#INFO:65756ns Passed: 0016 same bytes in BFM mem addr 0x00000040 and 0x00000840
#INFO:66652ns Passed: 0020 same bytes in BFM mem addr 0x00000040 and 0x00000840
#INFO:67556ns Passed: 0024 same bytes in BFM mem addr 0x00000040 and 0x00000840
#INFO:68468ns Passed: 0028 same bytes in BFM mem addr 0x00000040 and 0x00000840
#INFO:69388ns Passed: 0032 same bytes in BFM mem addr 0x00000040 and 0x00000840
#INFO:70324ns Passed: 0036 same bytes in BFM mem addr 0x00000040 and 0x00000840
#INFO:71268ns Passed: 0040 same bytes in BFM mem addr 0x00000040 and 0x00000840
# SUCCESS: Simulation stopped due to successful completion!
# Break in Function ebfm_log_stop_sim at ../../common/testbench//altpcietb_bfm_log.v
line 96
```

SignalTap II Files

The reference design package also includes SignalTap II files (.stp) that you can use with the SignalTap II Embedded Logic Analyzer to obtain information on the performance of this design. The SignalTap II files includes the key signals from the application logic. [Figure 8](#) shows an example of the SignalTap II Embedded Logic Analyzer GUI.


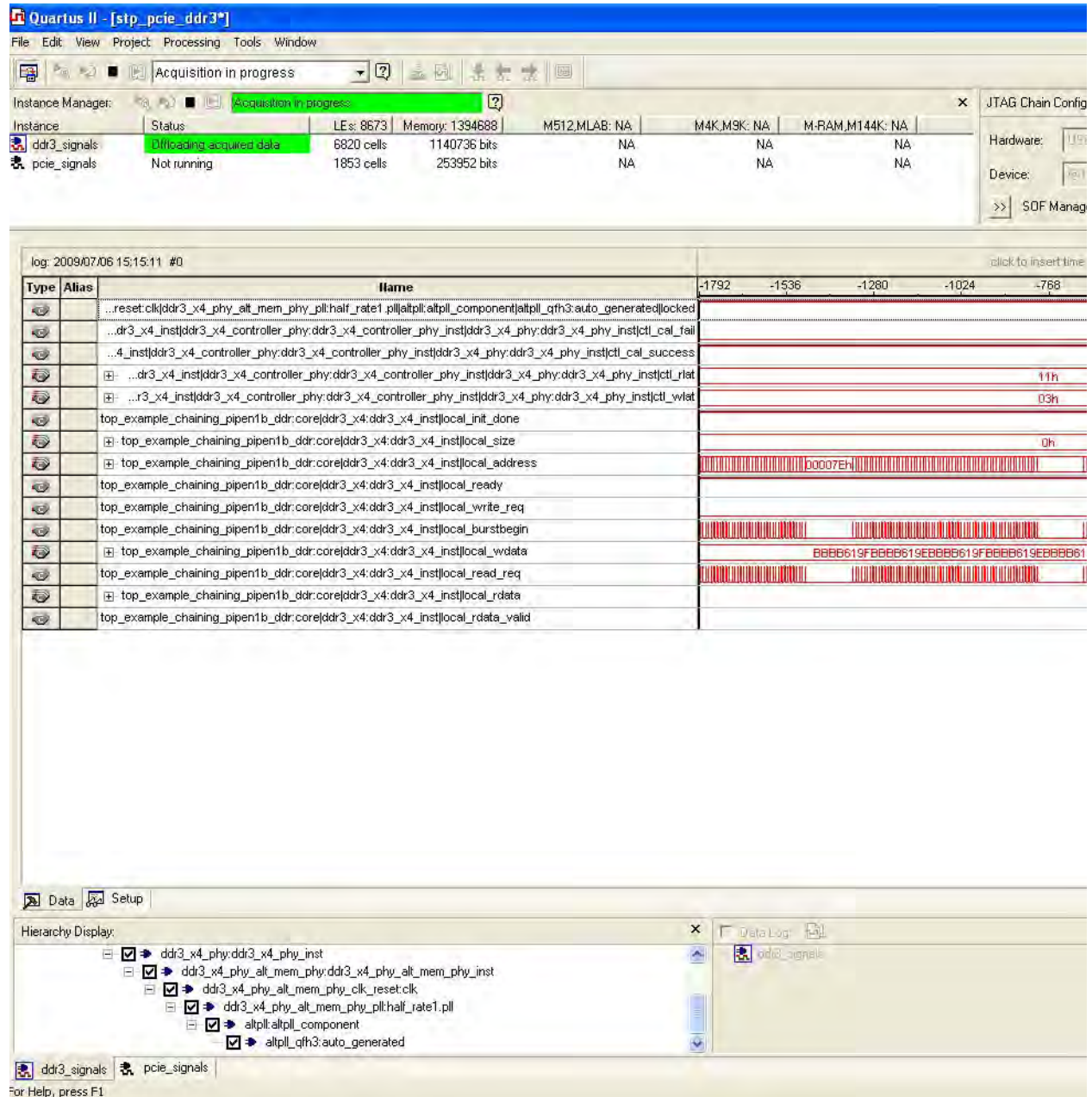
 For more information about the signals displayed in [Figure 8](#) refer to the *DDR and DDR2 High-Performance Controllers and ALTMEMPHY IP User Guide* or the *DDR3 High Performance Controller and ALTMEMPHY IP User Guide* in Volume 3 of the *External Memory Interface Handbook and PCI Express Compiler MegaCore Function User Guide*.

Figure 8. The SignalTap GUI



Design Limitations

This reference design has the following limitations:

- The DMA engine is not designed to transfer data with a size of less than 64 bytes.
- The DMA engine only transfers data in multiples of 64 bytes and the data must be aligned on a 64-byte address.
- The root port must use a DMA to access the SDRAM memory.

References

This application note references the following documents:

- *Chapter 3: Using DDR3 SDRAM in Stratix III and Stratix IV Devices* in *DDR, DDR2, and DDR3 SDRAM Design Tutorials* in Volume of the *External Memory Interface Handbook*.
- *AN 456: PCI Express High-Performance Reference Design*
- *DDR and DDR2 High-Performance Controllers and ALTMEMPHY IP User Guide*
- *DDR3 High Performance Controller and ALTMEMPHY IP User Guide* in Volume 3 of the *External Memory Interface Handbook*
- *JEDEC Standard for DDR3 SDRAM, JES79-3C*
- *PCI Express Compiler MegaCore Function User Guide*
- *Volume 3: Implementing Altera Memory Interface IP* in the *External Memory Interface Handbook*

Revision History

Table 4 shows the revision history for this application note.

Table 4. Revision History

Date and Revision	Changes Made	Summary of Changes
February 2010, version 1.3	The design now uses the High-Performance SDRAM Controller II for DDR3 SDRAM.	Updated to reflect use of newer memory controller.
December 2009, version 1.2	Updated to include the PCI Express hard IP implementation in an Arria II GX device using DDR2 memory. Added instructions for running a DMA simulation in ModelSim.	Updated to include performance for newer development board.
July 2009, version 1.1	Updated to use the PCI Express hard IP implementation in a Stratix IV GX device with DDR3 SDRAM.	Updated to showcase newer technology.
August 2006, version 1.0	Initial Release	—



101 Innovation Drive
San Jose, CA 95134
www.altera.com
Technical Support
www.altera.com/support

Copyright © 2009 Altera Corporation. All rights reserved. Altera, The Programmable Solutions Company, the stylized Altera logo, specific device designations, and all other words and logos that are identified as trademarks and/or service marks are, unless noted otherwise, the trademarks and service marks of Altera Corporation in the U.S. and other countries. All other product or service names are the property of their respective holders. Altera products are protected under numerous U.S. and foreign patents and pending applications, maskwork rights, and copyrights. Altera warrants performance of its semiconductor products to current specifications in accordance with Altera's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Altera assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Altera Corporation. Altera customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.



I.S. EN ISO 9001