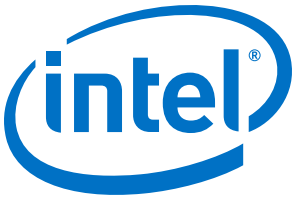


AN 855: PCI Express* High Performance Reference Design for Intel® Cyclone® 10 GX

Subscribe
Send Feedback

AN-855 | 2018.06.08
Latest document on the web: PDF | HTML

Contents

1. PCI Express High Performance Reference Design

The *PCI Express High-Performance Reference Design* highlights the performance of Intel's PCI Express* products. The design includes a high-performance chaining direct memory access (DMA) that transfers data between the a PCIe Endpoint in the FPGA, internal memory and the system memory. The reference design includes a Windows-based software application that sets up the DMA transfers. The software application also measures and displays the performance achieved for the transfers. This reference design enables you to evaluate the performance of the PCI Express protocol in Intel® Cyclone® 10 GX.

Intel offers the Intel Cyclone 10 GX Hard IP for PCI Express. The hard IP implementation is available as a Root Port or Endpoint. Depending on the device used, the hard IP implementation is compliant with *PCI Express Base Specification 2.0,* or *3.0*.

Related Information

PCI Express Base Specification 1.1, 2.0, or 3.0.

1.1. Understanding Throughput in PCI Express

The throughput in a PCI Express system depends on the following factors:

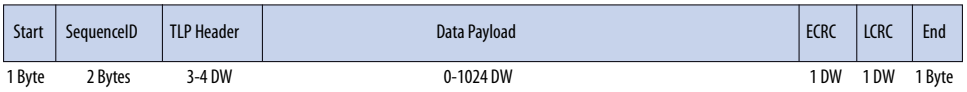
- Protocol overhead
- Payload size
- Completion latency
- Flow control update latency
- Characteristics of the devices that form the link

1.1.1. Protocol Overhead

PCI Express Gen1 and Gen2 IP cores use 8B/10B encoding. Each byte of data is converted into a 10-bit data code, resulting in a 25% overhead. The effective data rate is therefore reduced to 2 Gbps or 250 MBps per lane for Gen1, and 4 Gbps or 500 MBps per lane for Gen2.

An active link also transmits Data Link Layer Packets (DLLPs) and Physical Layer Packets (PLPs). The PLPs are four bytes or one dword and consist of SKP Ordered Sets. The DLLPs are two dwords and consist of the ACK/NAK and flow control DLLPs. The ACKs and flow control update DLLPs are transmitted in the opposite direction from the Transaction Layer Packet (TLP). If link is transmitting and receiving high bandwidth traffic, the DLLP activity can be significant. The DLLPs and PLPs reduce the effective bandwidth available for TLPs. The format of the TLP illustrates that the overhead if a TLP is seven dwords. The overhead is six dwords if the optional ECRC is not included.

Figure 1. TLP Format



The overhead includes the following fields:

- Start and End framing symbols
- A Sequence ID
- A TLP header that is three or four dwords long,
- The link cyclic redundancy check (LCRC).

The rest of the TLP contains 0–1024 dwords of data payload.

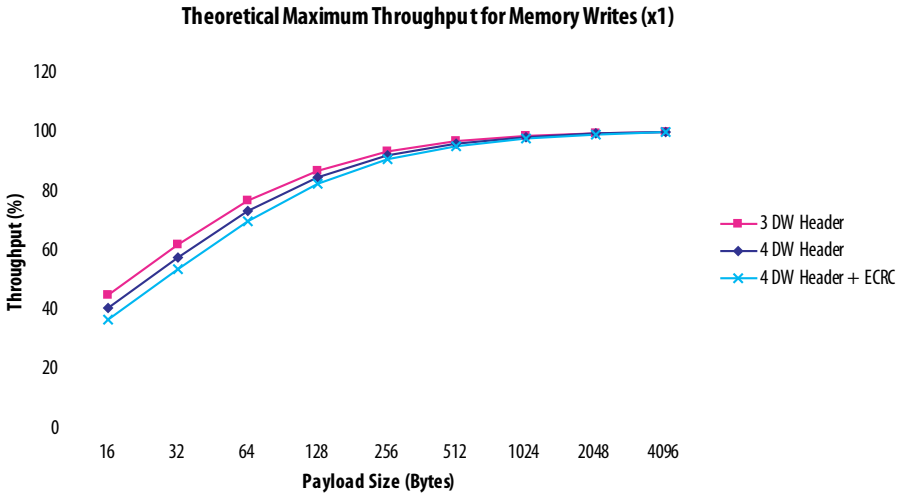
1.1.2. Throughput for Posted Writes

The theoretical maximum throughput is calculated using the following formula:

```
Throughput % = payload size / (payload size + overhead)
```

The following figure shows the maximum throughput possible with different TLP header sizes and ignores any DLLPs and PLPs. For a 256-byte maximum payload size and a three dword TLP header (or five dword overhead), the maximum possible throughput is (256/(256+20)), or 92%.

Figure 2. Maximum Throughput for Memory Writes



The `device control` register (bits 7:5) in the PCI Express Configuration Space specifies maximum TLP payload size. The parameter **maximum payload size** sets the read-only value of the `Maximum Payload Size Supported` field of the `Device Capabilities` register (bits 2:0). The payload size you specify for your variant may be reduced based on the system maximum payload size. This **maximum payload size** parameter affects the resource utilization. To maximize resources, do not specify a the maximum payload size that is greater than the system maximum payload size.

PCI Express uses flow control. A TLPs is not transmitted unless the receiver has enough free buffer space to accept it. Header and data credits track available buffer space. When the application in the completer accepts the TLP, it frees the RX buffer space in the completer's Transaction Layer. The completer sends a flow control update (FC Update DLLP) that returns the credits consumed by the originating TLP. After the device uses all of its initial credits, link bandwidth is limited by how fast it receives credit updates. Flow control updates depend on the maximum payload size and the latencies in the transmitting and receiving devices.

Related Information

Intel Arria® 10 and Intel Cyclone 10 GX Avalon®-ST Interface for PCI Express User Guide

For more information about the flow control update loop and the associated latencies, refer to the *Throughput Optimization* chapter. The information in the *Throughput Optimization* chapter is not specific to a particular device.

1.1.3. Throughput for Reads

PCI Express uses a split-transaction for reads. A requester first sends a memory read request. The completer then sends an ACK DLLP to acknowledge the memory read request. It subsequently returns a completion data that can be split into multiple completion packets.

Read throughput is somewhat lower than write throughput because the data for the read completions may be split into multiple packets rather than being returned in a single packet. The following example illustrates this point. This example uses a read request for 512 bytes and a completion packet size of 256 bytes. The maximum possible throughput is calculated as follows:

Number of completion packets = 512/256 = 2

Overhead for a 3 dword TLP Header with no ECRC = 2*20 = 40 bytes

Maximum Throughput % = 512/(512 + 40) = 92%.

These calculations do not take into account any DLLPs and PLPs. The *PCI Express Base Specification* defines a read completion boundary (RCB) parameter. The RCB parameter determines the naturally aligned address boundaries on which a read request may be serviced with multiple completions. For a root complex, the RCB is either 64 bytes or 128 bytes. For all other PCI Express devices, the RCB is 128 bytes.

Note: A non-aligned read request may experience a further throughput reduction.

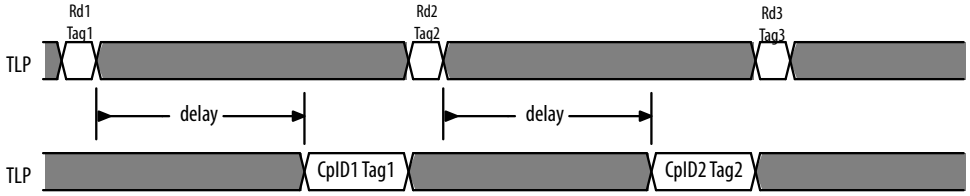
Read throughput depends on the round-trip delay between the following two times:

- The time when the application logic issues a read request
- The time when all of the completion data has been returned.

To maximize throughput, the application must issue enough read requests and process enough read completions. Or, the application must issue enough non-posted header credits to cover this delay.

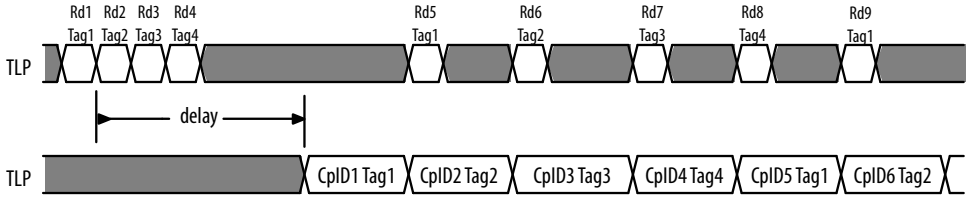
The following figure shows timing diagram for memory read requests (`MRd`) and completions (`CplD`). The requester waits for a completion before making a subsequent read request, resulting in lower throughput.

Figure 3. Low Performance Reads Request Timing Diagram



The following timing diagram eliminates the delay for completions with the exception of the first read. This strategy maintains a high throughput.

Figure 4. High Performance Read Request TimingDiagram



The requester must maintain maximum throughput for the completion data packets by selecting appropriate settings for completions in the RX buffer. All versions of Intel's PCIe IP cores offer five settings for the **RX Buffer credit allocation performance for requests** parameter. This parameter specifies the distribution of flow control header, data, and completion credits in the RX buffer. You should use this parameter to allocate credits to optimize for the anticipated workload.

A final constraint on the throughput is the number of outstanding read requests supported. The outstanding requests are limited by the number of header tags and the maximum read request size. The maximum read request size is controlled by the `device control` register (bits 14:12) in the PCIe Configuration Space. The Application Layer assign header tags to non-posted requests to identify completions data. The **Number of tags supported** parameter specifies number of tags available. A minimum number of tags are required to maintain sustained read throughput. This number is system dependent. On a Windows system, eight tags are usually enough to ensure continuous read completion with no gap for a 4 KByte read request. The *High Performance Request Timing Diagram* uses 4 tags. The first tag is reused for the fifth read.

Related Information

PCI Express Base Specification.
For more information the read completion boundary.

1.2. Deliverables Included with the Reference Design

The reference design includes the following components:

- Software application and Windows driver configured specifically for this reference design
- FPGA programming files for Intel Cyclone 10 GX FPGA Development Kit for x1 Gen1 and x4 Gen2.

Related Information

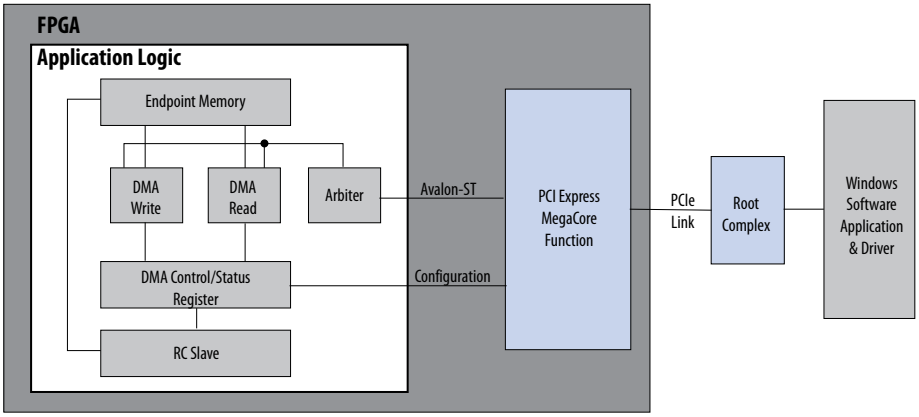
Arria 10 Avalon-ST Interface for PCIe Solutions user Guide
For more information about this example design in Arria 10 devices

1.3. Reference Design Functional Description

The reference design consists of the following components:

- An application layer that consists of the chaining DMA example generated by the IP core
- The IP core variation
- A software application and Windows driver configured specifically for this reference design

Figure 5. Reference Design Components



The chaining DMA example consists of two DMA modules in the application logic and an internal Endpoint memory. The design supports simultaneous DMA read and DMA write transactions. The DMA write module transfers data from Endpoint memory to the Root Complex system memory across the PCIe link. The DMA read module implements read transfers data from the Root Complex system memory across the PCIe link to Endpoint memory.

The reference design is included the FPGA and relies on no other hardware interface except the PCIe link. A chaining DMA provides higher performance than a simple DMA for non-contiguous memory transfers between the system and Endpoint memory. For a simple DMA, the software application programs the DMA registers for every transfer. The chaining DMA uses descriptor tables for each memory page. These descriptor tables contain the following information:

- Transfer length
- Source and destination addresses for the transfer
- Control information that sets the handshaking behavior between the software application and the DMA module

Each descriptor consists of four dwords. The descriptors are stored in a contiguous memory page.

Based on the attributes set in the Parameter Editor, the software application creates the necessary descriptor tables in the system memory. The software application also creates a descriptor header table. This table specifies the total number of descriptors and the address of the first descriptor table. At the beginning of the transfer, the software application programs the DMA registers with the descriptor header table. The DMA module continuously collects these descriptor tables for each DMA read and write and performs the transfers specified.

The DMA module also includes a performance counter. The counter starts when the software writes a descriptor header table to the DMA registers. It continues counting until the last data has been transferred by the DMA module. After the transfer is complete, the software application uses the counter value to compute the throughput for the transfer and reports it. The counter value includes latency for the initial descriptor read. Consequently, the throughput reported by the software application is less than the actual throughput.

1.3.1. Project Hierarchy

Intel Cyclone 10 GX devices use the following directory structure:

- The project directory contains:
  - The top-level module `top.v`
  - The Intel Quartus® Prime project file `top.qpf`
  - The Intel Quartus Prime project settings file `top.qsf`
- The `platform` folder contains all IP files.
- The `master_image` folder contains a working `.sof` configuration file or image.
- The `driver` folder contains the software application and Windows driver for this design.

IP Core Settings

The reference design supports a maximum payload size of 256 Bytes. The desired performance for received completions and requests is set to **Maximum**. The following tables show the settings for supported devices.

Table 1. System Settings for PCI Express IP Core – Intel Cyclone 10 GX Device

Parameter	Value
PCIe IP core type	**PCI Express hard IP**
System Settings	
Number of lanes	**x4**
Lane rate	**Gen2 (5.0 Gbps)**
Port type	**Native endpoint**
Application interface	**Avalon-ST 128-bit**
RX buffer credit allocation	**Low**
Reference clock frequency	**100 MHz**
Hard IP Mode	**Gen2x4, Interface: 128-bit, 125 MHz**

continued...

Parameter	Value
Enable Avalon-ST Reset output port	OFF (Unchecked)
Enable byte parity ports on Avalon-ST interface	OFF (Unchecked)
Enable multiple packets per cycle for the 256-bit interface	OFF (Unchecked)
Enable credit consumed selection port	OFF (Unchecked)
Enable Configuration Bypass (CfgBP)	OFF (Unchecked)
Enable local management interface (LMI)	OFF (Unchecked)

Table 2. PCI Register Settings – Intel Cyclone 10 GX Device

PCI Base Address Registers (Type 0 Configuration Space)		
BAR	**BAR Type**	**BAR Size**
0	32-bit Non-Prefetchable Memory	256 MBytes - 28 bits
1, 3, 4, 5	Disabled	N/A
2	32-bit Non-Prefetchable Memory	1 KBytes - 10 bits
PCI Read-Only Registers		
Register Name	**Value**	**Additional Information**
Vendor ID	0x1172	The Vendor ID can be either 0x1172 or 0xB0D8. This parameter has no effect on design behavior.
Device ID	0xE001	N/A
Revision ID	0x1	N/A
Class Code	0x00FF0000	N/A

Table 3. Capabilities Parameters – Intel Cyclone 10 GX Device

Capability Registers	
Device Capabilities	
Maximum payload size	256 Bytes
Number of tags supported	32
Completion timeout range	ABCD
Implement completion timeout disable	ON
Error Reporting	
Advanced error reporting (AER)	On
ECRC check	Off
ECRC generation	Off
ECRC forwarding	Off
Track receive completion buffer overflow	Off
Link Capabilities	
Link port number	1
Data link layer active reporting	Off

continued...

Capability Registers	
Surprise down reporting	Off
Slot clock configuration	On
MSI Capabilities	
MSI messages requested	4
MSI-X Capabilities	
Implement MSI-X	Off
MSI-X Table size	0
MSI-X Table Offset	0x0
MSI-X Table BAR Indicator (BIR)	0
Pending Bit Array (PBA) Offset	0x0
PBA BAR Indicator	0
Slot Capabilities	
Use Slot Power registers (Root Port only)	Off
Slot power scale	0
Slot power limit	0
Slot number	0

Table 4. Power Management Parameters – Intel Cyclone 10 GX Device

Parameter	Value
Power Management	
Endpoint L0s acceptable latency	Maximum of 64 ns
Endpoint L1 acceptable latency	Maximum of 1 us
PHY Characteristics	
Gen2 transmit deemphasis	6dB

Intel Quartus Prime Settings

The `.qar` files in the reference design package has the recommended synthesis, Fitter, and timing analysis settings. These settings are optimized for the parameters chosen in this reference design.

1.4. Hardware Requirements

The reference design requires the following hardware:

- The Intel Cyclone 10 GX FPGA Development Kit.
- A computer running Windows driver with an x4/x1 PCI Express slot for the Intel Cyclone 10 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 Intel Quartus Prime software for downloading FPGA programming files to the Intel Cyclone 10 GX FPGA development board. This computer is referred to as computer #2 in this document.
- A USB cable or other Intel download cable.
- A PCI Express x4 to x1 lane converter for x1 operation.

Note: Intel provides x4 and x1 design examples for this reference design. If the PCI Express slot on your motherboard has one lane but you want to use the x4 design example, you must use a lane converter to transfer data from the higher lanes to the lower available lane.

1.5. Software Requirements

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

- Reference design software installed on computer #1.
- PCI Express High Performance Reference design package, available as a downloadable compressed file.
- The Intel Quartus Prime software running on computer #2.

The x4 and x1 design examples for this reference design can be downloaded at the links below:

Cyclone 10 GX PCIe Gen1 x1 Design Example

Cyclone 10 GX PCIe Gen2 x4 Design Example

For more information about the flow control update loop and the associated latencies, refer to the *Throughput Optimization* chapter. The information in the *Throughput Optimization* chapter is not specific to a particular device.

1.6. Software Installation

You must have Administrator privileges to install the software application.

The software installation is included with the design files and the application includes an executable driver. The driver configuration is specific to this reference design.

1. Download the installation package from the Design Store and restore the project. The software application and Windows driver are included in the driver folder in the project directory.
2. Before plugging in the PCI Express card, copy the `Windows_for_AVST_On_Chip_Mem` directory to computer #1.
3. Install the Windows driver. Follow the instructions in the `README.txt` file from the `Windows_for_AVST_On_Chip_Mem` directory to install and run the software application.

1.7. Hardware Installation

The development kits include the integrated Intel FPGA Download Cable circuitry for FPGA programming. However, for the host computer and development board to communicate, you must install the Intel FPGA Download Cable on the host computer.

1. Power down computer #1 and plug the development board into the PCI Express slot. For an x1 operation, use a PCI Express lane converter.
2. Install the Intel FPGA Download Cable on the host computer.
3. Program the FPGA with the reference design using the Intel Quartus Prime software on computer #2.
   - If you are using the on-board/integrated Intel FPGA Download Cable, use a USB cable.
   - Otherwise, use the Intel FPGA Download Cable connection between computer #2 and the development board on computer #1.
4. Connect one end of the USB cable to the USB port on the development board.
5. Connect the other end of the cable to the USB port on the computer running the Intel Quartus Prime software, i.e. computer #2.

   To download the Intel FPGA Download Cable driver, go to the Intel support site at *Cable and Adapter Drivers Information*.

   For installation instructions, go to *USB-Blaster Driver Installation for Windows*.

Related Information

- Cable and Adapter Drivers Information
- USB-Blaster Driver Installation for Windows

1.7.1. Programming using the .sof File

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.

1. Start the Intel Quartus Prime programmer on computer #2.
2. Click **Hardware Setup** and select the **USB Blaster**. Click **Close**.
3. In the Intel Quartus Prime Programmer, click **Auto Detect**. This command lists devices attached to the JTAG chain on the development board.
4. Right-click the Intel Cyclone 10 GX (10CX2200YF780E5G) 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 Intel development kit. 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 Ctl-Alt-Delete to perform 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.

1.8. Running the Software Application

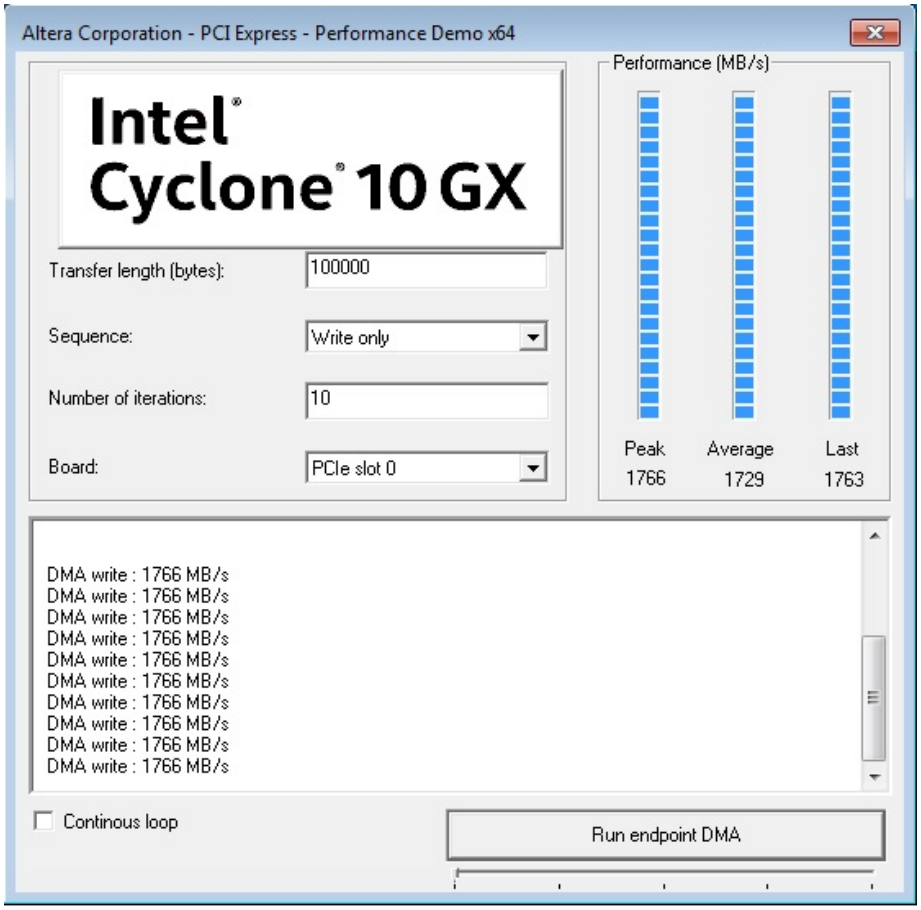
The software GUI has the following control fields:

- **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

1. Set the **Transfer length** to 100,000 bytes and the **Sequence** to **Write only**, Click **Run**.

   When set for **Write only**, the software programs the DMA registers in the FPGA to transfer data from the FPGA to the system memory in chunks of 100,000 bytes. The performance bars report the peak, average, and last throughput. The average throughput is computed across all the iterations.
2. You can use the GUI to change the **Transfer length** and **Sequence** and repeat the test.

Figure 6. Write-Only Options



3. Double-click on the application `Windows_for_AVST_On_Chip_Mem` in the `Windows_for_AVST_On_Chip_Mem` directory.
4. The application reports the board type, the number of active lanes, the maximum read request size, and the maximum payload size.

1.9. Additional Chaining DMA Commands

In addition to the parameter settings to control the chaining DMA, the GUI includes five other commands. The following table describes these commands. The position of the slider control changes the command.

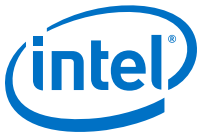
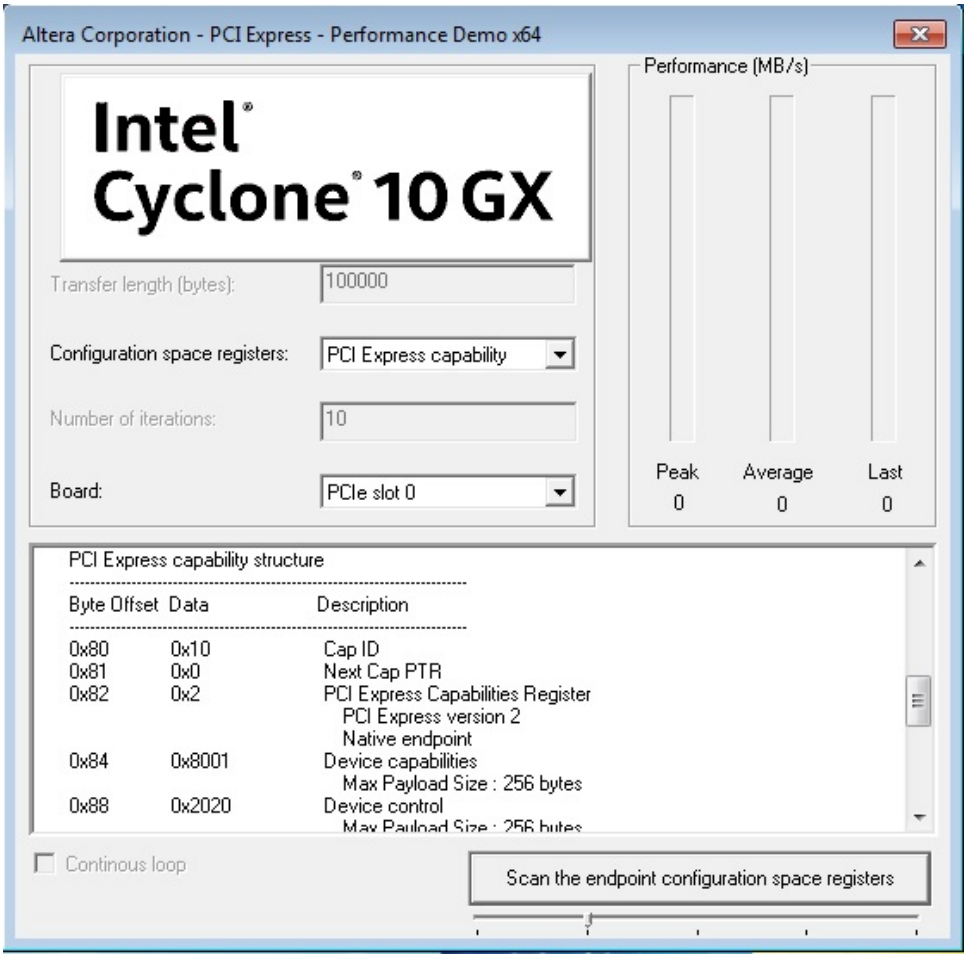


Table 5. PCI Express Performance Demo GUI Commands and Options

Command	Options	Description
Run endpoint DMA	**Write only** **Read only** **Read then write** **Write then read** **Read and write**	Writes transfers data from the FPGA to system memory. Reads transfer data from system memory to the FPGA.
Scan the endpoint configuration space registers	**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.
Run target read	**At endpoint address** **From 0x0 to endpoint address**	Programs the Root Port of the motherboard's PCI Express chipset to read from the FPGA's memory, as follows: • If you select **At Endpoint address**, you can type the starting read address in Endpoint memory in the **Endpoint address** field. • If you select **From 0x0 to Endpoint address**, the **Endpoint address** field specifies the transfer length.
Run target write	**At endpoint address** **From 0x0 to endpoint address**	Programs the Root Port of the motherboard's PCI Express chipset to write to the FPGA's memory, as follows: • If you select **At Endpoint address**, you can type the starting write address in Endpoint memory the **Endpoint address** field. • If you select **From 0x0 to Endpoint address**, the **Endpoint address** field specifies the transfer length.

Figure 7. Scan the Endpoint Configuration Space Registers



1.10. Using SignalTap II

Signal Tap II can provide information on the performance of this design. The `init` signal in the DMA read and write modules transitions to zero at the beginning of the transfer. You can use the `init` signal as a trigger in the SignalTap II file to capture data.

The `tx_st_ready` and `rx_st_valid` are indications of link utilization and throughput. In the transmit direction, the frequent deassertion of the `tx_st_ready` signal typically indicates that the IP core is not receiving enough credits from the device at the far end of the PCI Express link. It could also indicate that a x4 link has trained to x1. In the receive direction, the deassertion of `rx_st_valid` indicates that the IP core is not receiving enough data.

1.11. Performance Benchmarking Results

The following tables list the performance of x1, and x4 operations with the Intel Cyclone 10 GX FPGA development board for the Intel Xeon® E5-2603 Sandy Bridge EP processor using this reference design. The table shows the average throughput with the following parameters:

- 100 KByte transfer
- 20 iterations
- A 256-byte payload
- Maximum 512-byte read request
- 256-byte read completion

Note: Refer to the following web page for other available reference designs and application notes for PCI Express.

- *PCI Express Reference Designs and Application Notes*

Table 6. Intel Cyclone 10 GX Hard IP for PCI Express Performance

Configuration	DMA Read (MB/sec)	DMA Write (MB/sec)	Simultaneous DMA Read/ Write (MB/sec)	Theoretical maximum throughputs (MB/sec)	
				DMA Read (MB/sec)	DMA Write (MB/sec)
Gen2 X4	1706	1766	1652/1446	1855	1855
Gen1 X1	223	223	214/189	231	231

1.12. Document Revision History

Table 7. Document Revision History

Date	Version	Changes
2018.06.15	18.0.1	Initial revision for Intel Cyclone 10 GX