# Lattice Avant and Nexus PCIe Host DMA Driver Software User Guide

## Technical Note

FPGA-TN-02386-1.3

December 2025

## Disclaimers

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS, with all faults, and all associated risk is the responsibility entirely of the Buyer. The information provided herein is for informational purposes only and may contain technical inaccuracies or omissions, and may be otherwise rendered inaccurate for many reasons, and Lattice assumes no obligation to update or otherwise correct or revise this information. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. LATTICE PRODUCTS AND SERVICES ARE NOT DESIGNED, MANUFACTURED, OR TESTED FOR USE IN LIFE OR SAFETY CRITICAL SYSTEMS, HAZARDOUS ENVIRONMENTS, OR ANY OTHER ENVIRONMENTS REQUIRING FAIL-SAFE PERFORMANCE, INCLUDING ANY APPLICATION IN WHICH THE FAILURE OF THE PRODUCT OR SERVICE COULD LEAD TO DEATH, PERSONAL INJURY, SEVERE PROPERTY DAMAGE OR ENVIRONMENTAL HARM (COLLECTIVELY, "HIGH-RISK USES"). FURTHER, BUYER MUST TAKE PRUDENT STEPS TO PROTECT AGAINST PRODUCT AND SERVICE FAILURES, INCLUDING PROVIDING APPROPRIATE REDUNDANCIES, FAIL-SAFE FEATURES, AND/OR SHUT-DOWN MECHANISMS. LATTICE EXPRESSLY DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY OF FITNESS OF THE PRODUCTS OR SERVICES FOR HIGH-RISK USES. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

## Inclusive Language

This document was created consistent with Lattice Semiconductor’s inclusive language policy. In some cases, the language in underlying tools and other items may not yet have been updated. Please refer to Lattice’s inclusive language FAQ 6878 for a cross reference of terms. Note in some cases such as register names and state names it has been necessary to continue to utilize older terminology for compatibility.

# Contents

# Figures



# Tables

# Abbreviations in This Document

A list of abbreviations used in this document.

| Abbreviation | Definition |
|---|---|
| API | Application Programming Interface |
| AXI-MM | Advanced eXtensible Interface Memory-Mapped |
| AXI-S | Advanced eXtensible Interface Stream |
| BAR | Base Address Register |
| BIOS | Basic Input/Output System |
| CPU | Central Processing Unit |
| DMA | Direct Memory Access |
| FPGA | Field Programmable Gate Array |
| GCC | GNU Compiler Collection |
| I/O | Input/Output |
| ID | Identification |
| IP | Intellectual Property |
| OS | Operating System |
| PCIe | Peripheral Component Interconnect Express |
| RAM | Random Access Memory |
| RO | Read-only |
| RW | Read-write |
| SGDMA | Scatter-Gather Direct Memory Access |
| SPI | Serial Peripheral Interface |
| SRAM | Static Random Access Memory |

# 1. Introduction

PCI Express® is a high performance, fully scalable, and well-defined standard for a wide variety of computing and communications platforms. Refer to the PCI Express for Nexus FPGAs web page for more information on the Lattice PCIe ×1 and ×4 IP cores and the PCI Express for Avant FPGAs web page for more information on the Lattice PCIe ×8 IP core.

This guide describes how to use the PCIe host DMA driver software with the PCIe ×1, ×4, and ×8 DMA module.

The software driver is developed using Jungo WinDriver software toolkit. You may create a custom driver or use Jungo WinDriver for your driver development needs. To obtain Jungo WinDriver, contact Jungo for further information.

## 1.1. Purpose

This document is a reference guide for anyone using the driver with the Lattice PCIe ×1, ×4, and ×8 DMA modules. For details on developing a PCIe driver using Jungo WinDriver, refer to the Jungo WinDriver manual.

## 1.2. Audience

The intended audience for this document includes embedded system designers and embedded software developers using the Lattice Avant™ and Nexus™ devices. The technical guide assumes readers have expertise in embedded systems and FPGA technologies.

## 1.3. Driver Version

PCIe Host AXI-MM DMA driver version 25.01.00.

PCIe Host AXI-S DMA driver version 24.02.00.

## 1.4. Driver and IP Compatibility

**Table 1.1. Driver and IP Compatibility**

| Driver | Driver Version | IP Version |
|---|---|---|
| AXI-MM DMA driver:<br>• Linux: latticedma-25.01.00-Linux.sh<br>• Windows: latticedma-25.01.00-win64.exe | 25.01.00 | 3.0.0 (PCIe ×1) |
| | | 4.0.0 (PCIe ×4) |
| | | 3.0.0 (PCIe ×8) |
| AXI-S DMA driver:<br>• Linux: cpnxaxisdma-24.02.00-Linux.sh<br>• Windows: cpnxaxisdma-24.02.00-win64.exe | 24.02.00 | 3.4.0 (PCIe ×4) |

Refer to the PCIe ×1 IP Release Notes (FPGA-RN-02060), PCIe ×4 IP Release Notes (FPGA-RN-02059), and PCIe ×8 IP Release Notes (FPGA-RN-02061) more information on the driver and IP versions.

**Table 1.2. Driver Tested Environment Quick Facts**

| IP | Tested Device | PC Environment |
|---|---|---|
| PCIe ×1 | MachX05-65T Development Board<br>Certus-NX Versa Evaluation Board | Operating System: Windows 10, Windows 11, Linux |
| | | Supported Architecture: x86_64 |
| | | CPU: Intel, AMD |
| PCIe ×4 | CertusPro-NX PCIe Bridge Board | Operating System: Windows 10, Windows 11, Linux |
| | | Supported Architecture: x86_64 |
| | | CPU: Intel, AMD |

| IP | Tested Device | PC Environment |
|---|---|---|
| PCIe ×8 | Avant-X Versa Board | Operating System: Windows 10, Windows 11, Linux |
| | | Supported Architecture: x86_64 |
| | | CPU: Intel, AMD |

# 2. Software Setup

This section describes the steps to install the software driver onto the host machine. The following installation procedures use the AXI-MM DMA driver as an example.

**Note:** The installation steps are consistent for both AXI-MM and AXI-S DMA drivers, as well as across all driver versions. The driver name or version shown in screenshots or file directories may vary depending on the DMA data interface (AXI-MM or AXI-S) and the version used. This variation does not affect the installation process.

## 2.1. Driver File Location

After installing the PCIe IP package file (.ipk) using the Lattice Radiant software, you can find the driver file at the following location on your PC: *C:\Users\<user>\RadiantIPLocal\<ipk name>\driver*.

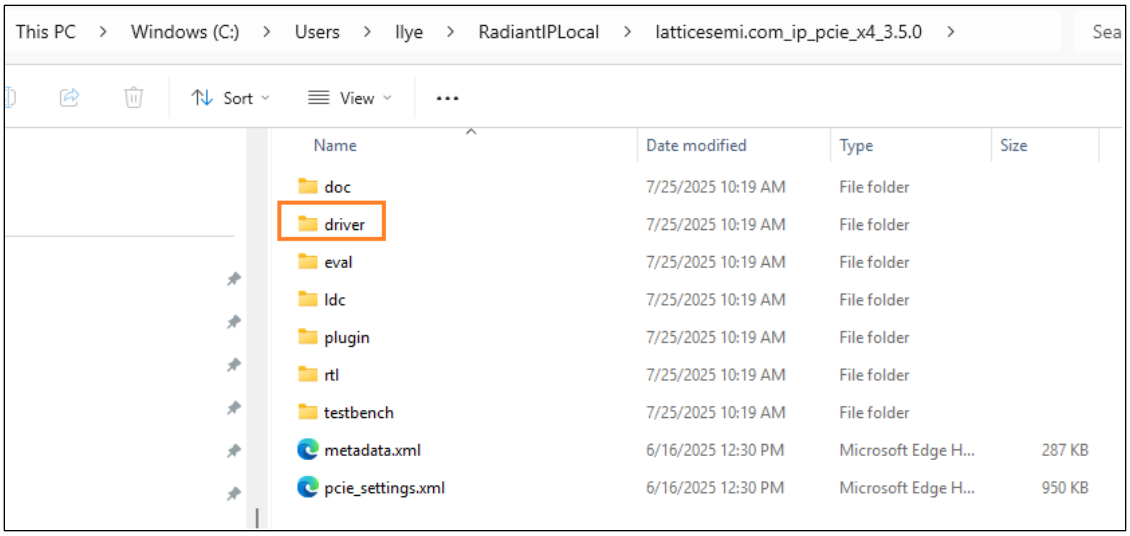


**Figure 2.1. Driver File Location**

## 2.2. Driver Installation on a Windows Machine

1. If you are installing the driver for the first time, skip step 2 and go to step 3.
2. If the driver is already installed previously, run *Uninstall.exe* in the installation folder: *C:\Program Files\latticedma_25.01.00* before you install the driver.

3. Double-click on *latticedma-25.01.00-win64.exe* to install the driver. Select **Next** to continue with the installation.

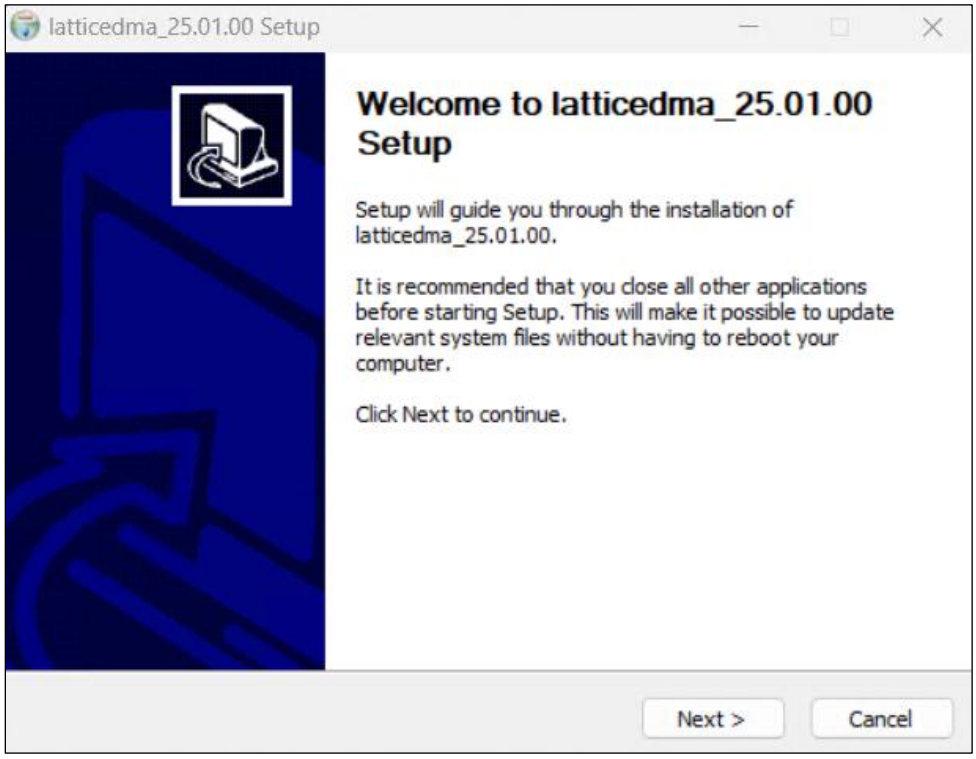


**Figure 2.2. Setup**

4. Click **I Agree** to continue with the installation.

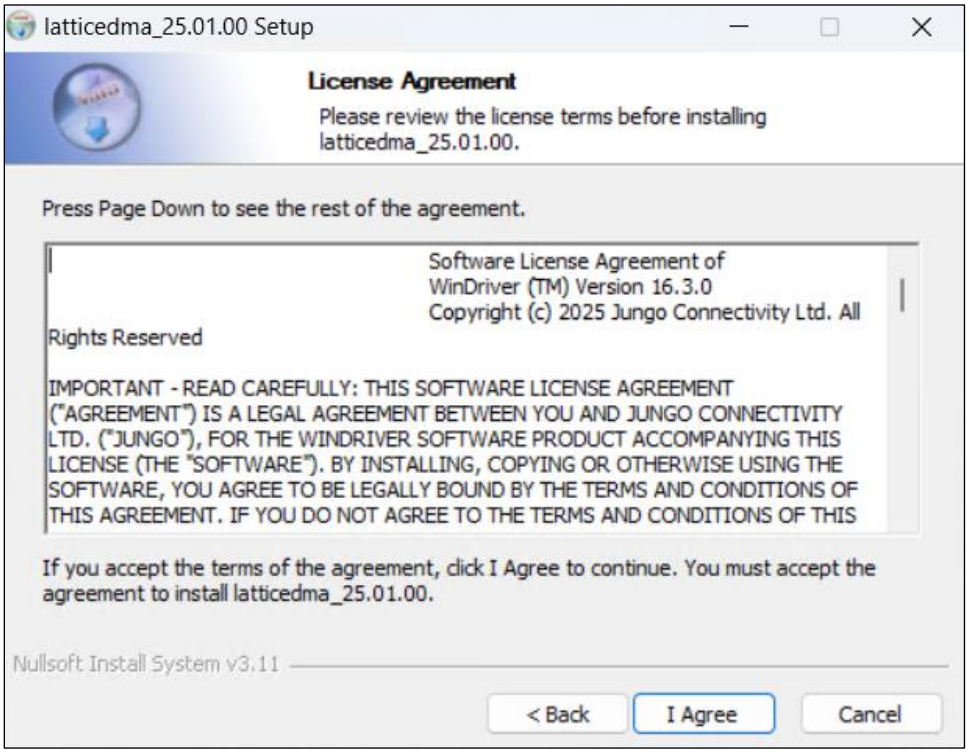


**Figure 2.3. Setup License Agreement**

5. Use the default location in the **Destination Folder** as the installation location and click **Next** to continue.

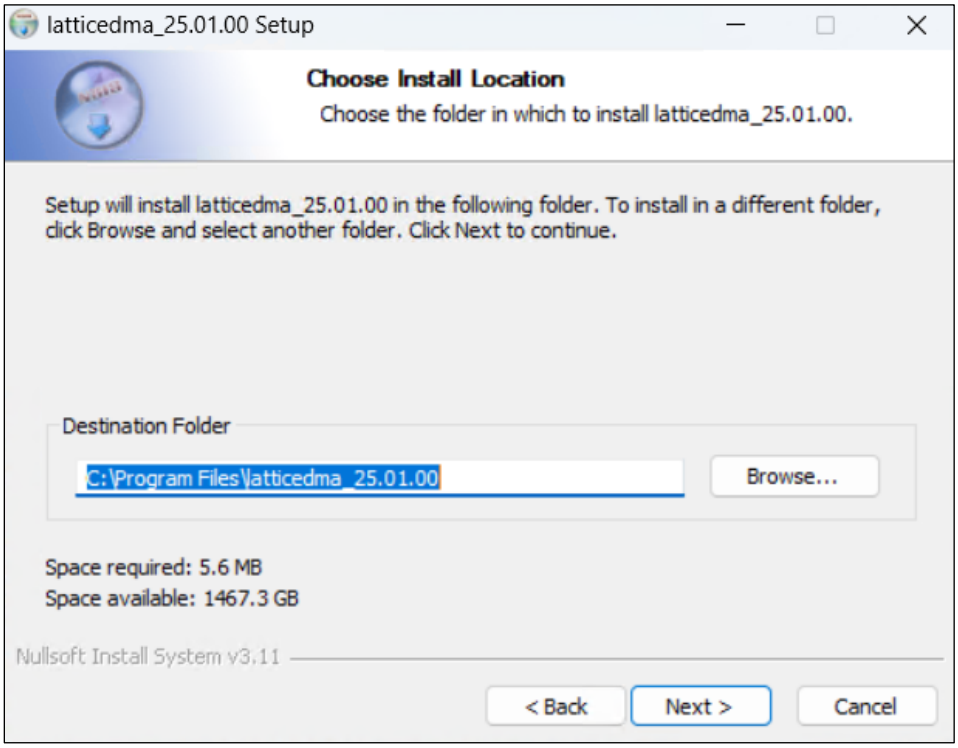


**Figure 2.4. Setup Install Location**

6. Click **Next** until you see the window as shown in Figure 2.5. Click **Install**.

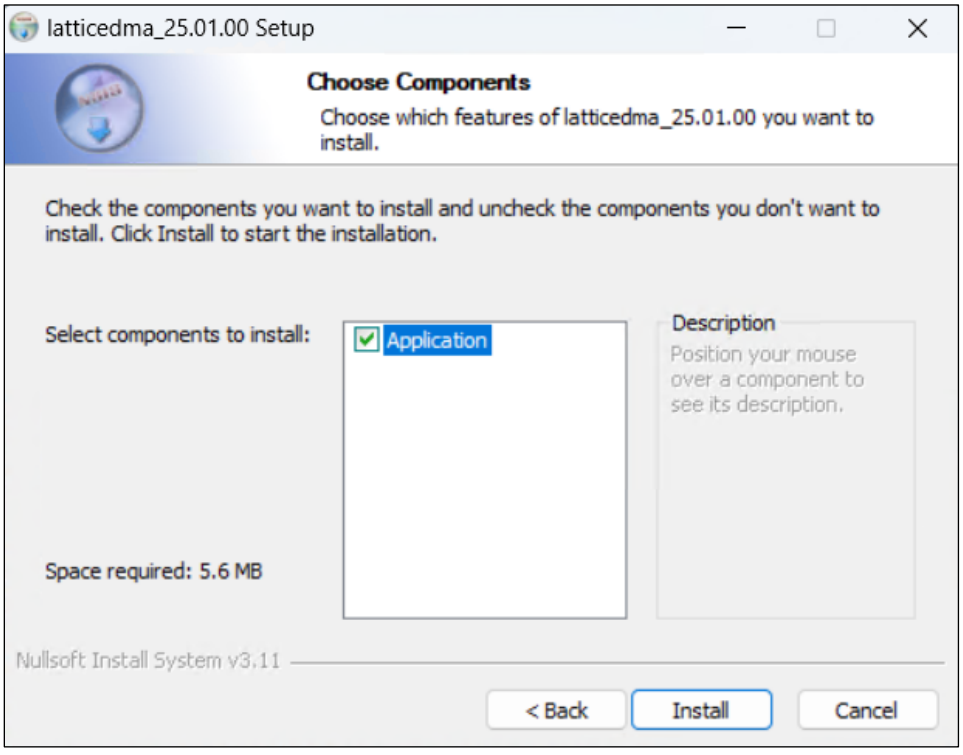


**Figure 2.5. Setup Install**

7. When installation is complete, open **Device Manager**. The following two new devices are listed under **Jungo Connectivity**, for example:
   - For CertusPro-NX or Avant-G/X devices:
     - *latticedma Driver*
     - *PCIe (Lattice - Device ID: 0x9c25 (Requiring driver: latticedma)*
   - For Certus-NX or MachXO5-NX (LFMXO5-65T) devices:
     - *latticedma Driver*
     - *PCIe (Lattice – Device ID: 0x9c1d (Requiring driver: latticedma)*

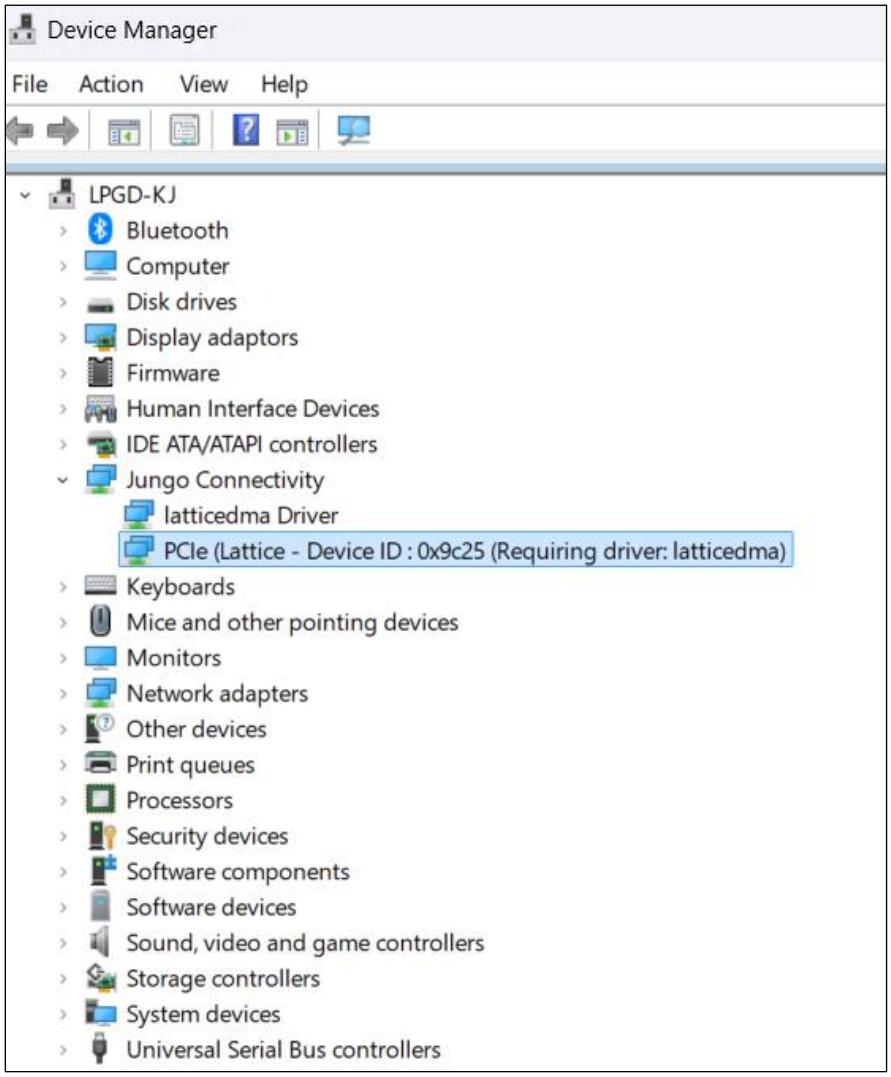


**Figure 2.6. Device Manager for CertusPro-NX or Avant-G/X Devices**

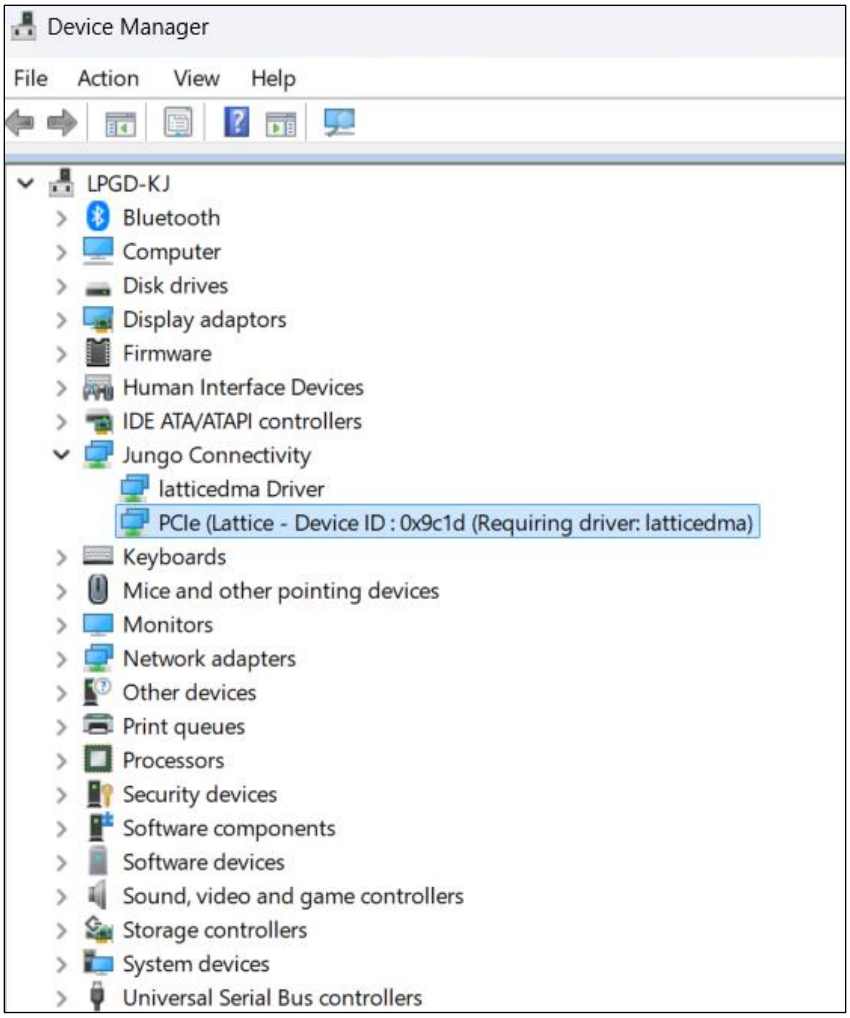


**Figure 2.7. Device Manager for Certus-NX or MachXO5-NX (LFMXO5-65T) Devices**

8. A new folder is created in *C:\Program Files\latticedma_25.01.00*.
9. A shortcut: *latticedma_25.01.00* is added to your desktop.
10. You can run the user app by running the desktop shortcut or by running: *C:\Program Files\latticedma_25.01.00\bin\latticedma.exe*.

**Note**: This driver is built with a demo license and is limited to 30 days. After 30 days, you need to replace the license with a valid license and rebuild the driver. For more information, refer to the Jungo WinDriver documentation. To continue using WinDriver drivers, you can obtain a valid paid annual subscription from Jungo. For more information, contact Jungo.

## 2.3. Driver Installation on a Linux Machine

**Note:** The Linux driver is compatible only with kernel version 6.12.16 or earlier. If you use a newer kernel, you must install version 6.12.16 or below for the driver to function properly.

1. Before installing the driver, check if you have GCC installed by typing the following command in your terminal:
```
gcc –version
```

2. If you see a message indicating that the command is not found, install GCC:
```
sudo apt update
sudo apt install gcc
gcc –version
```

3. Go to the directory where file *latticedma-25.01.00-Linux.sh* is located. Give executable permission to the *.sh* file:
```
sudo chmod +x latticedma-25.01.00-Linux.sh
```

4. Then, run the command below to build the driver and user application and install the driver.
```
sudo ./latticedma-25.01.00-Linux.sh
```

5. Enter *Y* when prompted.
6. Two new modules are installed. Run the command below to verify that *latticedma* is installed:
```
sudo lsmod | grep latticedma
```

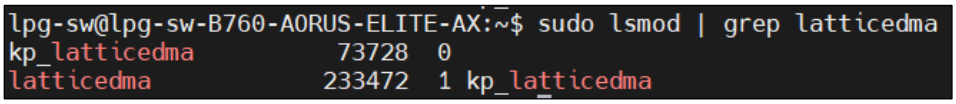


**Figure 2.8. lsmod**

7. A new folder is created in your current directory *latticedma-25.01.00-Linux*.
8. Give permission to all the files in the directory:
```
sudo chmod 777 -R *
```

9. Run the user application:
```
sudo ./latticedma-25.01.00-Linux/bin/latticedma
```

```
pc@pc-desktop:~/lll/Jungo$ sudo ./latticedma-25.01.00-Linux/bin/latticedma

LATTICEDMA diagnostic utility.
Application accesses hardware using WinDriver.

Found 1 matching device [ Vendor ID 0x1204, Device ID 0x9c25 ]:

 1. Vendor ID: 0x1204, Device ID: 0x9c25
    Location: Domain [0x0], Bus [0x9], Slot [0x0], Function [0x0]
    Memory range [BAR 0]: base 0xFCB10000, size 0x10000
    Memory range [BAR 1]: base 0xFCB00000, size 0x10000
    Interrupt: IRQ 255
    Interrupt Options (supported interrupts):
        Message-Signaled Interrupt (MSI)
        Level-Sensitive Interrupt
    PCI Express Generation: Gen2

Using [KP_LATTICEDMA] Kernel-Plugin driver version [1.00 - My Driver V1.00]

LATTICEDMA main menu
--------------------
1. Scan PCI bus
2. Find and open a PCI device
3. Read/write memory and I/O addresses on the device
4. Read/write the PCI configuration space
5. Direct Memory Access (DMA)
6. Register/unregister plug-and-play and power management events
99. Exit Menu

Enter option:
```

**Figure 2.9. Run latticedma**

**Note**: This driver is built with a demo license and is limited to one hour. You need to reinstall the driver after one hour. To continue using WinDriver drivers without having to reinstall every hour, obtain a valid paid annual subscription from Jungo. For more information, contact Jungo.

# 3. Application Overview

The PCIe software driver can run on both Windows and Linux operating systems. To run the software driver, you need to program the FPGA with the example design to the SPI flash or SRAM on the board.

The PCIe software driver is developed using the Jungo WinDriver software toolkit. It comes with a console-based user application that provides the following functionalities:

- Scanning the PCI bus
- Finding and opening a PCI device
- Reading/writing the PCI configuration space
- Reading/writing memory and I/O addresses on the device
- Performing direct memory access (DMA)

The AXI-MM DMA driver is a common driver for PCIe ×1, PCIe ×4, and PCIe ×8 IP. However, the AXI-MM DMA example design has different FPGA internal RAM sizes configured for devices supported by PCIe ×1, PCIe ×4, and PCIe ×8 IP:

- 64 kB for devices supported by PCIe ×1 IP
- 128 kB for devices supported by PCIe ×4 and PCIe ×8 IP

The device internal RAM size is identified by the driver reading the device ID at offset 0x2 of the PCI Configuration Space. If the device ID read is 0x9c1d, the FPGA internal RAM size is assumed to be 64 kB. If the device ID read is 0x9c25, the FPGA internal RAM size is assumed to be 128 kB. If the device ID read does not match either 0x9c25 or 0x9c1d, the FPGA internal RAM size is assumed to be 64 kB.

**Table 3.1. AXI-MM DMA Driver Device ID Requirement**

| IP | Required Device ID |
|---|---|
| PCIe ×1 | 0x9c1d |
| PCIe ×4 | 0x9c25 |
| PCIe ×8 | 0x9c25 |

**Note:** When creating the AXI-MM DMA example design, make sure the PCIe device ID is set according to the table above, based on the IP used. Otherwise, a DMA error occurs if the transfer exceeds the FPGA internal RAM size. This applies to both Windows and Linux AXI-MM DMA drivers.

The AXI-S DMA driver only works with AXI-S DMA example design when the device ID is set to 0x9c25; otherwise, the driver does not recognize the device. Supporting a different device ID requires modifications to the driver files. This applies to both Windows and Linux AXI-S DMA drivers.

## 3.1. DMA Functionalities

Below are the DMA functionalities provided by the user application.

```
LATTICEDMA main menu
--------------------
1. Scan PCI bus
2. Find and open a PCI device
3. Read/write memory and I/O addresses on the device
4. Read/write the PCI configuration space
5. Direct Memory Access (DMA)
6. Register/unregister plug-and-play and power management events
99. Exit Menu

Enter option: 5

LATTICEDMA DMA menu
-------------------
1. Perform DMA transfer
2. Perform DMA transfer to device, perform DMA transfer from device and compare the DMA buffers
3. Perform DMA transfer to device with incremented data, perform DMA transfer from device and compare the DMA buffers
4. Measure DMA performance
99. Exit Menu

Enter option:
```

**Figure 3.1. LATTICEDMA DMA Menu**

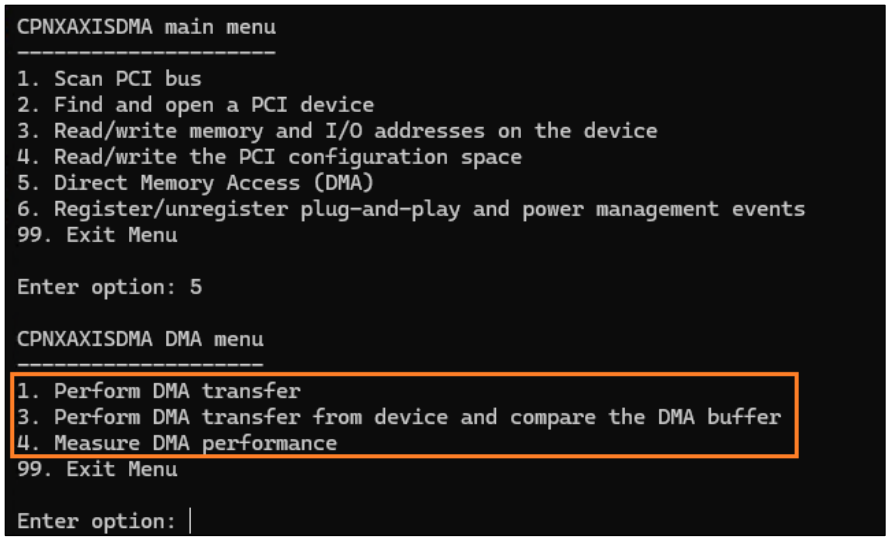


**Figure 3.2. CPNXAXISDMA DMA Menu**

## 3.1.1. DMA Transfer

### 3.1.1.1. AXI-MM DMA Transfer

When selecting **Perform DMA transfer**, you are prompted to select the DMA direction.

The **From device** DMA direction indicates from the FPGA to the host device, while the **To device** DMA direction indicates from the host device to the FPGA.

```
LATTICEDMA DMA menu
-------------------
1. Perform DMA transfer
2. Perform DMA transfer to device, perform DMA transfer from device and compare the DMA buffers
3. Perform DMA transfer to device with incremented data, perform DMA transfer from device and compare the DMA buffers
4. Measure DMA performance
99. Exit Menu

Enter option: 1

Select DMA direction:
---------------------
1. From device
2. To device
99. Cancel
Enter option: 1

Enter number of packets to transfer (4-byte packets) (max value: 32768) or 'x' to cancel: 50

NOTICE: The device memory address should be 8 DWORD aligned, otherwise it is rounded down to the nearest 8 DWORD aligned address

Enter device memory address for transfer (max value: 0x1FF38) or 'x' to cancel: 0x0

Running DMA using Kernel-PlugIn driver [KP_LATTICEDMA]

###
Message Signalled Interrupt #1 received
MSI data 0x0
###

Buffer:

00000000 12345678 12345678 12345678 12345678 12345678 12345678 12345678 12345678
00000020 12345678 12345678 12345678 12345678 12345678 12345678 12345678 12345678
00000040 12345678 12345678 12345678 12345678 12345678 12345678 12345678 12345678
00000060 12345678 12345678 12345678 12345678 12345678 12345678 12345678 12345678
00000080 12345678 12345678 12345678 12345678 12345678 12345678 12345678 12345678
000000a0 12345678 12345678 12345678 12345678 12345678 12345678 12345678 12345678
000000c0 12345678 12345678

DMA transfer from device completed successfully
DMA interrupts disabled
Closed DMA handle
```

**Figure 3.3. LATTICEDMA Transfer Direction**

#### 3.1.1.2. AXI-S DMA Transfer

When selecting **Perform DMA transfer**, only the **From device** DMA direction is supported.

The AXI-S example design does not include FPGA internal RAM. Therefore, you are not prompted to enter a device memory address when performing the DMA transfer.

```
CPNXAXISDMA DMA menu
--------------------
1. Perform DMA transfer
3. Perform DMA transfer from device and compare the DMA buffer
4. Measure DMA performance
99. Exit Menu

Enter option: 1

Select DMA direction:
---------------------
1. From device
99. Cancel
Enter option: 1

Enter number of packets to transfer (4-byte packets) (max value: 26214400) or 'x' to cancel: 10
Running DMA using Kernel-PlugIn driver [KP_CPNXAXISDMA]

###
Message Signalled Interrupt #1 received
MSI data 0x0
###

Buffer:

00000000 e3e2e1e0 e7e6e5e4 ebeae9e8 efeeedec f3f2f1f0 f7f6f5f4 fbfaf9f8 fffefdfc
00000020 03020100 07060504

DMA transfer completed successfully
DMA interrupts disabled
Closed DMA handle
```

**Figure 3.4. CPNXAXISDMA Transfer Direction**

## 3.1.2. DMA Transfer and Data Comparison

### 3.1.2.1. AXI-MM DMA Transfer and Data Comparison

When selecting **Perform DMA transfer to device, perform DMA transfer from device and compare the DMA buffers**, the driver application duplicates the user-entered data pattern to fill the write buffer. It then initiates a DMA transfer from the host to the device, followed by a DMA transfer from the device to the host. The values in the read buffer are then compared to those in the write buffer, and the status of the DMA buffer comparison is reported on the console.

```
LATTICEDMA DMA menu
-------------------
1. Perform DMA transfer
2. Perform DMA transfer to device, perform DMA transfer from device and compare the DMA buffers
3. Perform DMA transfer to device with incremented data, perform DMA transfer from device and compare the DMA buffers
4. Measure DMA performance
99. Exit Menu

Enter option: 2

Enter DMA data pattern as 4-byte packet (max value: 0xFFFFFFFF) or 'x' to cancel: 0x12345678

Enter number of packets to transfer (4-byte packets) (max value: 32768) or 'x' to cancel: 50

NOTICE: The device memory address should be 8 DWORD aligned, otherwise it is rounded down to the nearest 8 DWORD aligned address

Enter device memory address for transfer (max value: 0x1FF38) or 'x' to cancel: 0x0

Running DMA using Kernel-PlugIn driver [KP_LATTICEDMA]

###
Message Signalled Interrupt #1 received
MSI data 0x0
###


DMA transfer to device completed successfully
DMA interrupts disabled
Closed DMA handle
Running DMA using Kernel-PlugIn driver [KP_LATTICEDMA]

###
Message Signalled Interrupt #1 received
MSI data 0x0
###

Buffer:

00000000 12345678 12345678 12345678 12345678 12345678 12345678 12345678 12345678
00000020 12345678 12345678 12345678 12345678 12345678 12345678 12345678 12345678
00000040 12345678 12345678 12345678 12345678 12345678 12345678 12345678 12345678
00000060 12345678 12345678 12345678 12345678 12345678 12345678 12345678 12345678
00000080 12345678 12345678 12345678 12345678 12345678 12345678 12345678 12345678
000000a0 12345678 12345678 12345678 12345678 12345678 12345678 12345678 12345678
000000c0 12345678 12345678

DMA transfer from device completed successfully
DMA interrupts disabled
Closed DMA handle
Write buffer and read buffer are identical
```

**Figure 3.5. LATTICEDMA Data Comparison**

#### 3.1.2.2. AXI-S DMA Transfer and Data Comparison

The AXI-S DMA example design configures the FPGA to transfer a running incremental data when the host initiates a DMA transfer. The data returned by the FPGA increments byte by byte, for example, 0x00, 0x01, 0x02, 0x03, up to 0xFF, then rollover to 0x00 and continues the sequence.

When selecting **Perform DMA transfer from device and compare the DMA buffer**, the driver application executes a DMA transfer from the device to host. The application then compares and verifies that the data follows an incremental pattern, ranging from 0x00 to 0xFF, byte by byte. The status of the data verification is reported on the console.

```
CPNXAXISDMA DMA menu
--------------------
1. Perform DMA transfer
3. Perform DMA transfer from device and compare the DMA buffer
4. Measure DMA performance
99. Exit Menu

Enter option: 3

NOTICE: The DMA transfer size should be DWORD aligned, otherwise it is rounded down to the nearest DWORD aligned address

Enter number of packets to transfer (1-byte packets) (max value: 104857600) or 'x' to cancel: 100
Running DMA using Kernel-PlugIn driver [KP_CPNXAXISDMA]

###
Message Signalled Interrupt #1 received
MSI data 0x0
###

Buffer:

00000000 20 21 22 23 24 25 26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35 36 37 38 39 3a 3b 3c 3d 3e 3f
00000020 40 41 42 43 44 45 46 47 48 49 4a 4b 4c 4d 4e 4f 50 51 52 53 54 55 56 57 58 59 5a 5b 5c 5d 5e 5f
00000040 60 61 62 63 64 65 66 67 68 69 6a 6b 6c 6d 6e 6f 70 71 72 73 74 75 76 77 78 79 7a 7b 7c 7d 7e 7f
00000060 80 81 82 83

DMA transfer completed successfully
DMA data verification successful
DMA interrupts disabled
```

**Figure 3.6. CPNXAXISDMA Data Comparison**

### 3.1.3. DMA Transfer with Incremented Data and Data Comparison

The DMA transfer with incremented data and data comparison functionality is available only with the AXI-MM DMA driver.

When selecting **Perform DMA transfer to device with incremented data, perform DMA transfer from device and compare the DMA buffers**, the driver application fills the write buffer with incremented data, starting from the user-entered initial value. It then initiates a DMA transfer from the host to the device, followed by a DMA transfer from the device to the host. After the transfers are complete, the values in the read buffer are compared to those in the write buffer. The status of the DMA buffer comparison is reported on the console.

```
LATTICEDMA DMA menu
-------------------
1. Perform DMA transfer
2. Perform DMA transfer to device, perform DMA transfer from device and compare the DMA buffers
3. Perform DMA transfer to device with incremented data, perform DMA transfer from device and compare the DMA buffers
4. Measure DMA performance
99. Exit Menu

Enter option: 3

Enter DMA starting value (max value: 0xFF) or 'x' to cancel: 0x0

Enter number of packets to transfer (1-byte packets) (max value: 131072) or 'x' to cancel: 100

NOTICE: The device memory address should be 8 DWORD aligned, otherwise it is rounded down to the nearest 8 DWORD aligned address

Enter device memory address for transfer (max value: 0x1FF9C) or 'x' to cancel: 0x0

Running DMA using Kernel-PlugIn driver [KP_LATTICEDMA]

###
Message Signalled Interrupt #1 received
MSI data 0x0
###


DMA transfer to device completed successfully
DMA interrupts disabled
Closed DMA handle
Running DMA using Kernel-PlugIn driver [KP_LATTICEDMA]

###
Message Signalled Interrupt #1 received
MSI data 0x0
###

Buffer:

00000000 00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f 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 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 50 51 52 53 54 55 56 57 58 59 5a 5b 5c 5d 5e 5f
00000060 60 61 62 63

DMA transfer from device completed successfully
DMA interrupts disabled
Closed DMA handle
Write buffer and read buffer are identical
```

**Figure 3.7. DMA Data Comparison with Incremented Data**

### 3.1.4. DMA Performance Measure

When selecting **Measure DMA performance**, ensure that the PCIe link status displays the expected link speed and link width to achieve maximum throughput. For example, when running the DMA driver with a CertusPro-NX programmed with a gen3 ×4 bitstream, verify that the negotiated link speed is *8.0 GT/s* and the link width is *×4*.

You can check this by using the application driver to read the configuration space:

1. Select **Read/Write PCI configuration space** from the main menu.
2. Choose **Read from a named PCI Express register**.
3. Enter *9* to select **LINK_STS** register.

```
Read/write the device's configuration space
-------------------------------------------
1. Read from an offset
2. Write to an offset
3. Read all configuration registers defined for the device (see list above)
4. Read from a named register
5. Write to a named register
6. Read from a named PCI Express register
7. Write to a named PCI Express register
8. Scan PCI/PCIe capabilities
9. Write to an offset, read from the same offset and compare the values
99. Exit Menu

Enter option: 6

PCI configuration registers:
----------------------------

PCI Registers
-------------
    Name                  BAR  Offset     Size  R/W  Description
    ----                  ---  ------     ----  ---  -----------
 1. PCIE_CAP_ID                0x0         1    RW   PCI Express
                                                     Capability ID
 2. NEXT_CAP_PTR               0x1         1    RW   Next Capabiliy Pointer
 3. CAP_REG                    0x2         2    RW   Capabilities Register
 4. DEV_CAPS                   0x4         4    RW   Device Capabilities
 5. DEV_CTL                    0x8         2    RW   Device Control
 6. DEV_STS                    0xA         2    RW   Device Status
 7. LNK_CAPS                   0xC         4    RW   Link Capabilities
 8. LNK_CTL                    0x10        2    RW   Link Control
 9. LNK_STS                    0x12        2    RW   Link Status
10. SLOT_CAPS                  0x14        4    RW   Slot Capabilities
11. SLOT_CTL                   0x18        2    RW   Slot Control
12. SLOT_STS                   0x1A        2    RW   Slot Status
13. ROOT_CAPS                  0x1C        2    RW   Root Capabilities
14. ROOT_CTL                   0x1E        2    RW   Root Control
15. ROOT_STS                   0x20        4    RW   Root Status
16. DEV_CAPS2                  0x24        4    RW   Device Capabilities 2
17. DEV_CTL2                   0x28        2    RW   Device Control 2
18. DEV_STS2                   0x2A        2    RW   Device Status 2
19. LNK_CAPS2                  0x2C        4    RW   Link Capabilities 2
20. LNK_CTL2                   0x30        2    RW   Link Control 2
21. LNK_STS2                   0x32        2    RW   Link Status 2
22. SLOT_CAPS2                 0x34        4    RW   Slot Capabilities 2
23. SLOT_CTL2                  0x38        2    RW   Slot Control 2
24. SLOT_STS2                  0x3A        2    RW   Slot Status 2

Select a register from the list above to read from or 0 to cancel: 9
Read [0x1043] from register LNK_STS at offset [0x52]
Decoded register data:
Current Link Speed: 8.0 GT/s
Negotiated Link Width: x4
Link Training: False
```

**Figure 3.8. PCIe Link Status Register**

If the Gen3 device negotiated link speed is not 8.0 GT/s, verify that the FPGA card is connected to a slot that supports PCIe Gen3, and that the slot is properly configured to support Gen3. You may need to configure the PCIe speed settings in the system BIOS to ensure Gen3 support.

#### 3.1.4.1. AXI-MM DMA Performance Measure

DMA performance measurement is available for single-direction DMA transfer (host to device or device to host) and bi-directional simultaneous DMA transfer.

For single-direction DMA transfers, the maximum buffer size available in the PCIe ×4 and ×8 example designs is 128 kB, while in the PCIe ×1 example design is 64 kB.

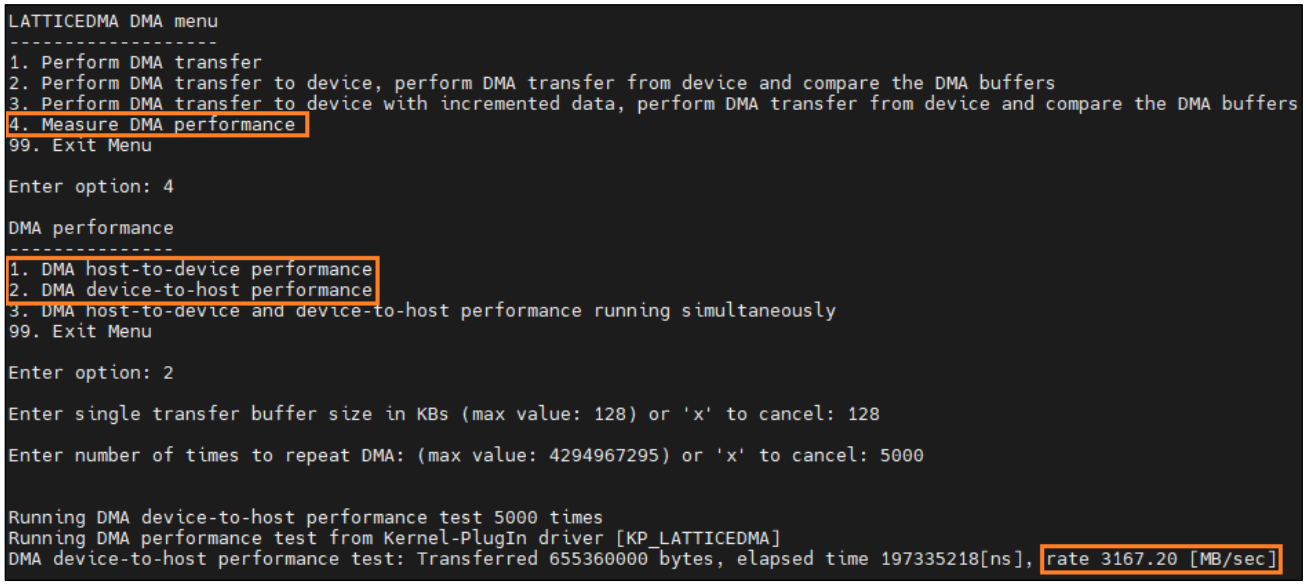


**Figure 3.9. LATTICEDMA Measure DMA Performance for Single-Direction Transfer**

For a bi-directional simultaneous DMA transfer, the maximum buffer size available is half of the maximum buffer size for a single-direction DMA transfer. For example, the maximum buffer size for bi-directional DMA transfer for a CertusPro-NX example design is 128 ÷ 2 = 64 kB. The FPGA internal RAM is divided into two halves: one half for host-to-device DMA transfers and the other half for device-to-host DMA transfers.

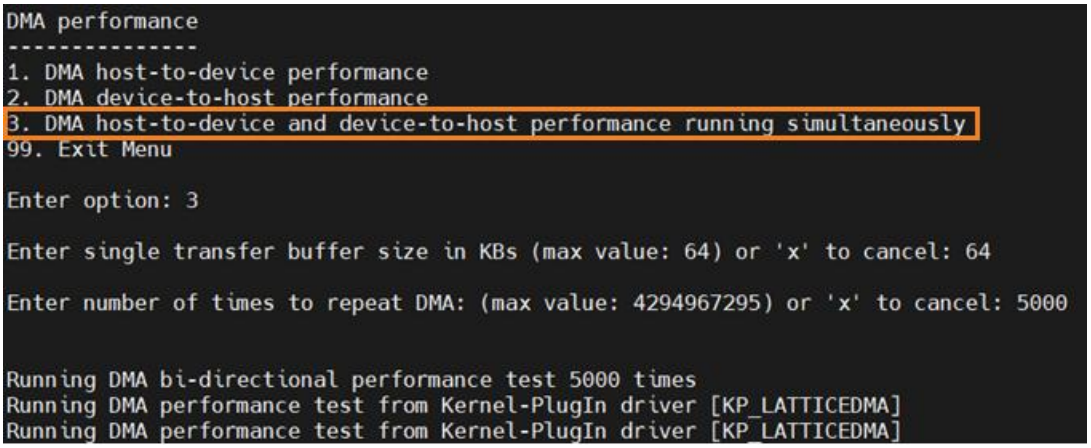


**Figure 3.10. LATTICEDMA Measure DMA Performance for Bi-Directional Simultaneous Transfer**

#### 3.1.4.2. AXI-S DMA Performance Measure

DMA performance measurement is available only with single-direction DMA transfer and with device to host DMA transfer. Bi-directional simultaneous DMA transfer is not supported by the AXI-S DMA driver. The maximum transfer size is 100 MB at a time.

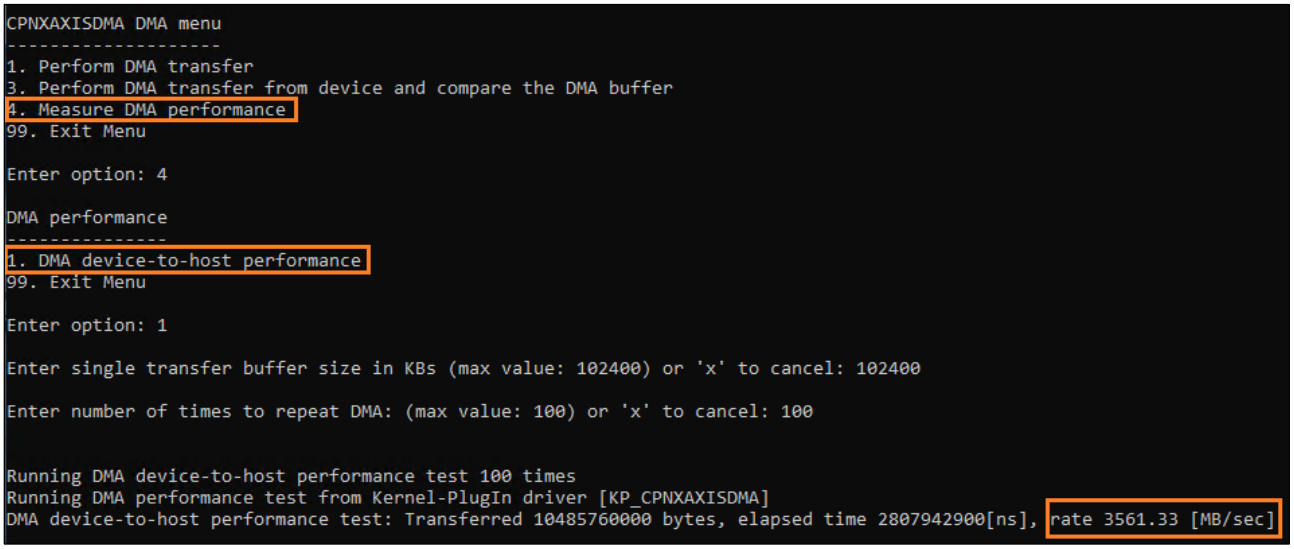


**Figure 3.11. CPNXAXISDMA Measure DMA Performance**

## 3.2. Read/Write Memory and I/O Addresses

The DMA example design has the DMA bypass mode enabled, allowing you to perform memory read/write on the device. When selecting **Read/write memory and I/O addresses on the device**, the default active address space is BAR 0. BAR 0 is mapped to the DMA internal registers for DMA configuration and status. Therefore BAR 0 memory is read-only (RO) to prevent any interference with the internal registers.

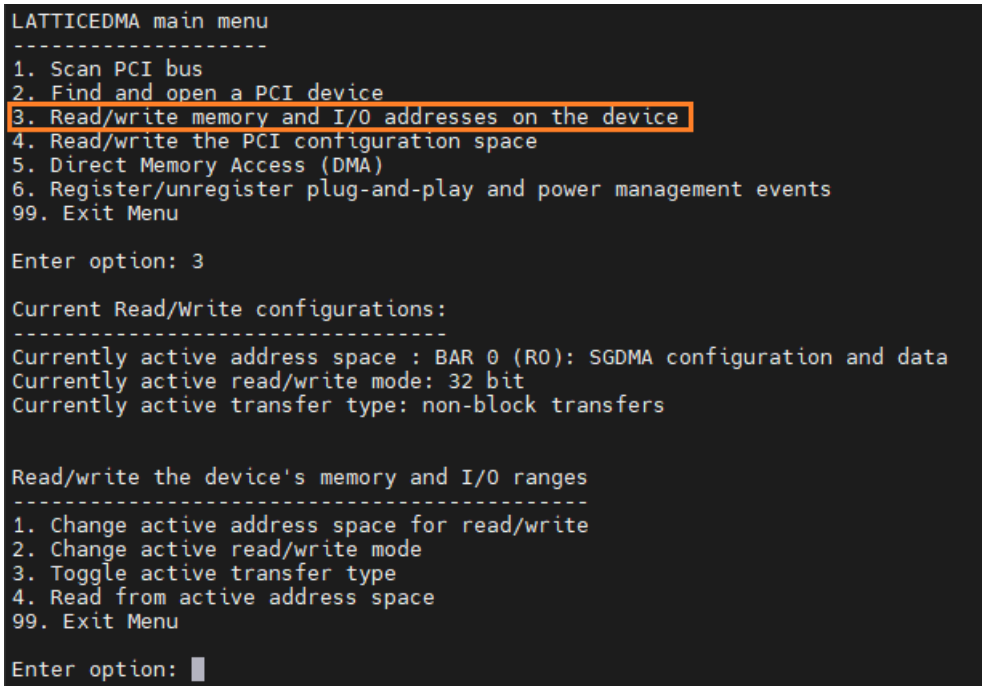


**Figure 3.12. LATTICEDMA Read/Write Memory**

The DMA example design has 2 BARs enabled:

- BAR 0 for DMA registers
- BAR 1 for memory or I/O read/write operations.

BAR 0 memory is read-only (RO), while BAR 1 memory is read-write (RW).

To change the active address space to BAR1, select **Change active address space for read/write.** When BAR 1 is selected, both read and write operations can be performed on the memory.

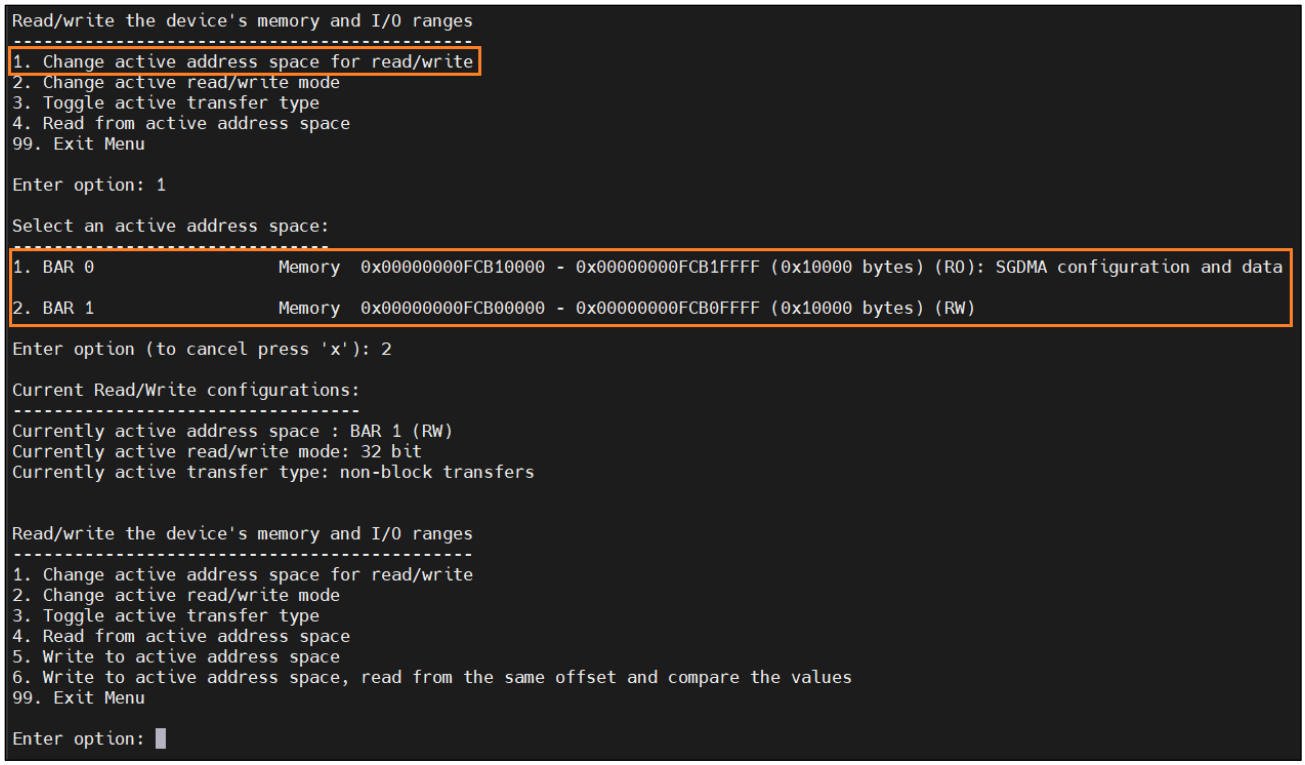


**Figure 3.13. LATTICEDMA Change Active Address Space for Read/Write**

### 3.2.1. Read/Write Mode

The driver application supports read/write modes of 8 bits, 16 bits, 32 bits, and 64 bits. However, the FPGA only supports up to 32 bits data access. You can change the read/write mode by selecting **Change active read/write mode**.

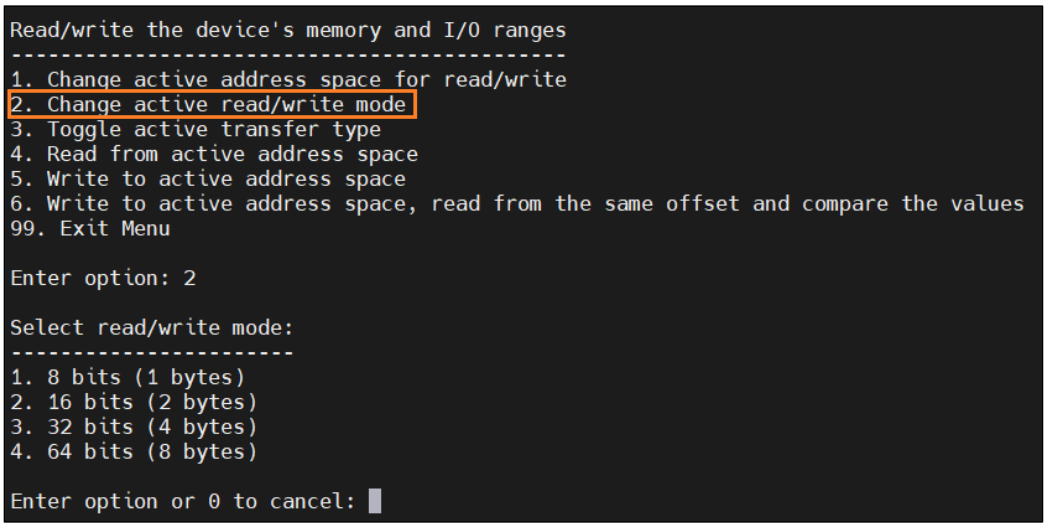


**Figure 3.14. LATTICEDMA Change Active Read/Write Mode**

### 3.2.2. Transfer Type: Non-block or Block Transfers

You can toggle between non-block transfers and block transfers by selecting **Toggle active transfer type**.

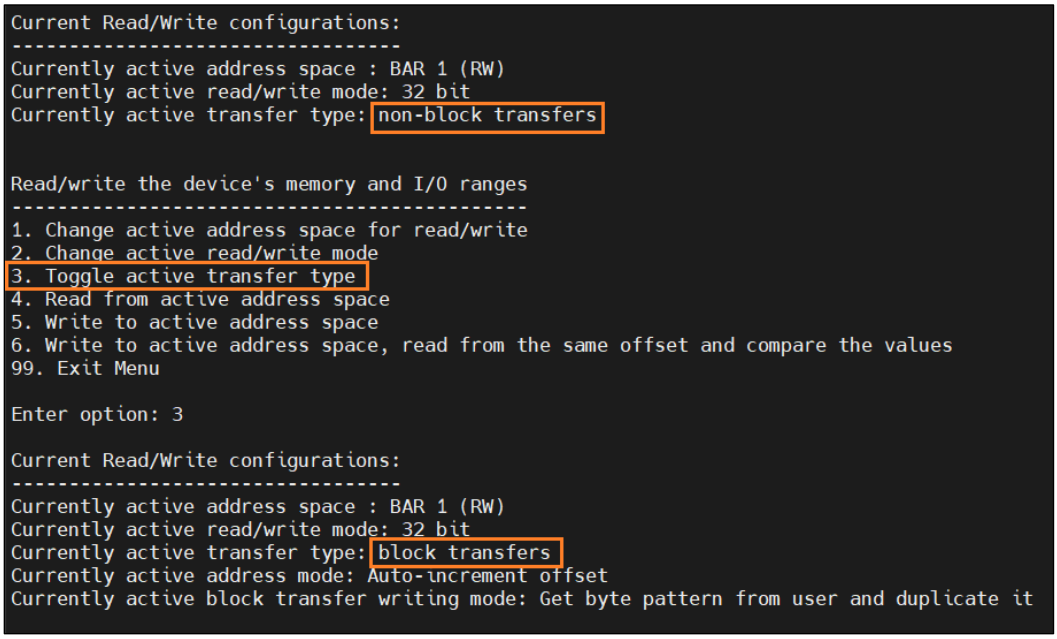


**Figure 3.15. LATTICEDMA Non-block or Block Transfer**

#### 3.2.2.1. Non-block Transfer

Non-block transfer refers to a read/write operation to a single unit of memory depending on the read/write mode:

- If 8-bit mode is selected, non-block transfer reads/writes 1 byte of data.
- If 16-bit mode is selected, non-block transfer reads/writes 2 bytes of data.
- If 32-bit mode is selected, non-block transfer reads/writes 4 bytes of data.
- If 64-bit mode is selected, non-block transfer reads/writes 8 bytes of data.

#### 3.2.2.2. Block Transfer

Block transfer allows you to read/write data to/from a block of memory.

### 3.2.3. Write, Read, and Compare Values

For both block and non-block transfers, there is a function that writes your entered data, then reads it back, and compares the values. Any data mismatches are reported on the console.

```
Current Read/Write configurations:
----------------------------------
Currently active address space : BAR 1 (RW)
Currently active read/write mode: 32 bit
Currently active transfer type: non-block transfers


Read/write the device's memory and I/O ranges
---------------------------------------------
1. Change active address space for read/write
2. Change active read/write mode
3. Toggle active transfer type
4. Read from active address space
5. Write to active address space
6. Write to active address space, read from the same offset and compare the values
99. Exit Menu

Enter option: 6
Notice: Your device must support both read and write of size [0x4] from the selected offset for this option to work
offset (to cancel press 'x'): 0x0
Enter data to write (max value: 0xFFFFFFFF) or 'x' to cancel: 0x12345678
Wrote 0x12345678 to offset 0x0 in BAR 1
Read 0x12345678 from offset 0x0 in BAR 1
Write buffer and read buffer are identical
```

**Figure 3.16. Non-block Transfer – Write, Read, and Compare Values**

```
Current Read/Write configurations:
----------------------------------
Currently active address space : BAR 1 (RW)
Currently active read/write mode: 32 bit
Currently active transfer type: block transfers
Currently active address mode: Auto-increment offset
Currently active block transfer writing mode: Get start value from user and increment it


Read/write the device's memory and I/O ranges
---------------------------------------------
1. Change active address space for read/write
2. Change active read/write mode
3. Toggle active transfer type
4. Read from active address space
5. Write to active address space
6. Write to active address space, read from the same offset and compare the values
7. Toggle active block transfer address increment mode
8. Change active block transfer write mode
99. Exit Menu

Enter option: 6
Notice: Your device must support both read and write of size [0x4] from the selected offset for this option to work
offset (to cancel press 'x'): 0x0
bytes (to cancel press 'x'): 0x100
Enter starting value (max value: 0xFFFFFFFF) or 'x' to cancel: 0x1
Wrote 0x100 bytes to offset 0x0

01 00 00 00 02 00 00 00 03 00 00 00 04 00 00 00
05 00 00 00 06 00 00 00 07 00 00 00 08 00 00 00
09 00 00 00 0A 00 00 00 0B 00 00 00 0C 00 00 00
0D 00 00 00 0E 00 00 00 0F 00 00 00 10 00 00 00
11 00 00 00 12 00 00 00 13 00 00 00 14 00 00 00
15 00 00 00 16 00 00 00 17 00 00 00 18 00 00 00
19 00 00 00 1A 00 00 00 1B 00 00 00 1C 00 00 00
1D 00 00 00 1E 00 00 00 1F 00 00 00 20 00 00 00
21 00 00 00 22 00 00 00 23 00 00 00 24 00 00 00
25 00 00 00 26 00 00 00 27 00 00 00 28 00 00 00
29 00 00 00 2A 00 00 00 2B 00 00 00 2C 00 00 00
2D 00 00 00 2E 00 00 00 2F 00 00 00 30 00 00 00
31 00 00 00 32 00 00 00 33 00 00 00 34 00 00 00
35 00 00 00 36 00 00 00 37 00 00 00 38 00 00 00
39 00 00 00 3A 00 00 00 3B 00 00 00 3C 00 00 00
3D 00 00 00 3E 00 00 00 3F 00 00 00 40 00 00 00

Write buffer and read buffer are identical
Press ENTER to continue
```

**Figure 3.17. Block Transfer – Write, Read, and Compare Values**

# 4. APIs

The APIs are defined by Jungo's WinDriver.

Table 4.1 shows the list of APIs used by the host PCIe driver to enable SGDMA data transfer and its operation. For more details, refer to the links provided in the table below.

**Table 4.1. List of APIs**

| WinDriver API | Description |
|---|---|
| OsEventCreate | Creates an event object. |
| OsEventWait | Waits until a specified event object is in the signaled state or the time-out interval elapses. |
| OsEventSignal | Sets the specified event object to the signaled state. |
| OsEventClose | Closes a handle to an event object. |
| WDC_IntEnable | • Enables interrupt handling for the device.<br>• The software passes in a user-mode interrupt handler callback function, LATTICEDMA_IntHandler/CPNXAXISDMA_IntHandler, which is called after an interrupt is received and processed in the kernel.<br>• The last argument is set to TRUE as the driver uses a Kernel Plugin driver. |
| WDC_IntDisable | Disables interrupt handling for the device. |
| WDC_DMASGBufLock | • Locks a pre-allocated user-mode memory buffer for DMA which is passed in as second argument of the API.<br>• Returns the corresponding physical mappings of the locked DMA pages as the fifth argument of the API. |
| WDC_DMAContigBufLock | • Allocates a contiguous descriptor buffer, locks it in physical memory<br>• Returns mapping of the allocated descriptor buffer to physical address and to user-mode and kernel virtual address spaces. |
| WDC_DMABufUnlock | Unlocks and frees the memory allocated for a DMA buffer by a previous call to WDC_DMASGBufLock and WDC_DMAContigBufLock. |
| WDC_CallKerPlug | Sends a message from a user-mode application to a Kernel PlugIn driver, in this case, the software sends a message to start the DMA transfer. |
| kp_interlocked_init | Initializes a Kernel PlugIn interlocked counter. |
| kp_interlocked_read | Reads to the value of a Kernel PlugIn interlocked counter, in this case, the software driver reads the intReceived flag. |
| kp_interlocked_set | Sets the value of a Kernel PlugIn interlocked counter to the specified value. Here, it is used to set the intReceived flag inside the kernel plugin interrupt handler function. |
| kp_interlocked_uninit | Uninitialized a Kernel PlugIn interlocked counter. |
| WDC_DMASyncCpu | Synchronizes all CPU caches with the DMA buffer by flushing the data from the CPU caches. |

# 5. Software Flow Diagrams

Note that some details of the software are omitted from the flowchart to simplify the diagrams and to describe the operation more clearly.

## 5.1. LATTICEDMA_DmaOpen/CPNXAXISDMA_DmaOpen

This function opens a DMA handle, allocate and initialize DMA information structure, including allocation of the scatter-gather DMA buffer.

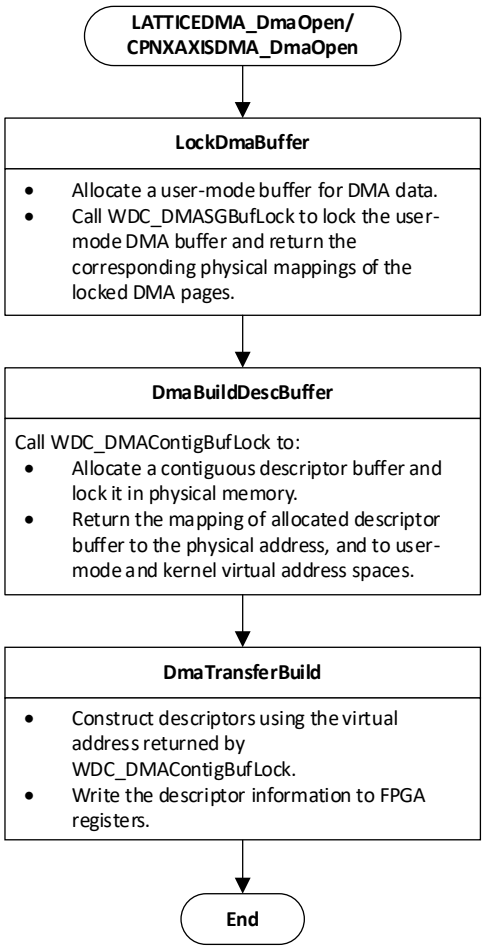


**Figure 5.1. LATTICEDMA_DmaOpen/CPNXAXISDMA_DmaOpen**

## 5.2. DmaPerformanceSingleDir

This function initializes the DMA and starts a thread to initiate DMA transfer and measure the DMA throughput.

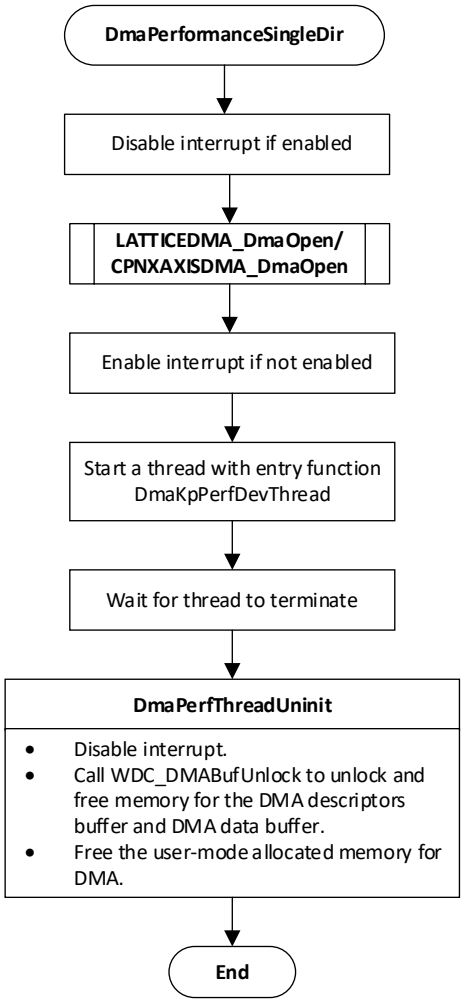


**Figure 5.2. DmaPerformanceSingleDir**

## 5.3. DmaKpPerfDevThread

This is the thread function that sends a message to the kernel mode to start the DMA transfer, get the start time, poll for DMA completion, and get the end time for DMA throughput measurement.

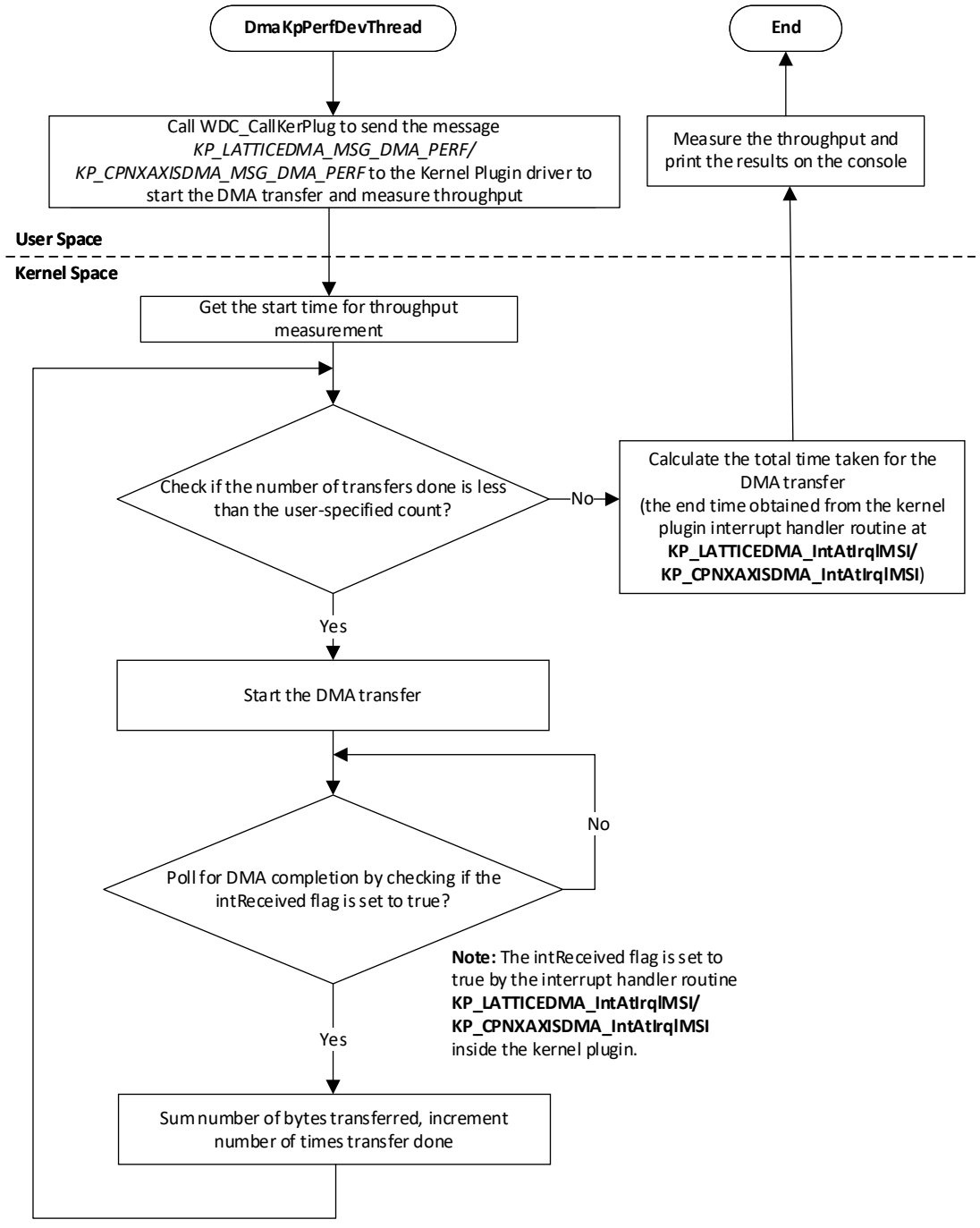


**Figure 5.3. DmaKpPerfDevThread**

# References

- PCI Express ×1/×2/×4 Endpoint IP Core User Guide (FPGA-IPUG-02009)
- PCIe ×1 IP Core User Guide (FPGA-IPUG-02091)
- PCIe ×4 IP Core User Guide (FPGA-IPUG-02126)
- PCIe ×8 IP Core User Guide (FPGA-IPUG-02243)
- PCIe ×1 IP Release Notes (FPGA-RN-02060)
- PCIe ×4 IP Release Notes (FPGA-RN-02059)
- PCIe ×8 IP Release Notes (FPGA-RN-02061)
- Jungo WinDriver: Introduction web page
- PCI Express Endpoint Core web page
- PCI Express for Nexus FPGAs web page
- PCI Express for Avant FPGAs web page
- Avant-G web page
- Avant-X web page
- Certus-NX web page
- CertusPro-NX web page
- MachXO5-NX web page
- Lattice Solutions IP Cores web page
- Lattice Solutions Reference Designs web page
- Lattice Insights web page for Lattice Semiconductor training courses and learning plans

# Technical Support Assistance

Submit a technical support case through www.latticesemi.com/techsupport.

For frequently asked questions, refer to the Lattice Answer Database at www.latticesemi.com/Support/AnswerDatabase.

# Revision History

**Revision 1.3, December 2025**

| Section | Change Summary |
|---|---|
| All | • Updated the document title from *PCIe Host Driver Software for CertusPro-NX AXI-MM and AXI-S DMA User Guide* to *Lattice Avant and Nexus PCIe Host DMA Driver Software User Guide*.<br>• Made editorial fixes. |
| Abbreviations in This Document | Added *BIOS*, *GCC*, *I/O*, *ID*, and *IP*. |
| Introduction | • Updated the descriptions in the Introduction, Purpose, Audience, Driver Version, and Driver and IP Compatibility sections.<br>• Updated Table 1.1. Driver and IP Compatibility and Table 1.2. Driver Tested Environment Quick Facts. |
| Software Setup | • Updated the description in the Software Setup section.<br>• Added the Driver File Location section.<br>• In the Driver Installation on a Windows Machine section:<br>  • Updated the file names and directory paths to reflect the latest naming conventions.<br>  • Added Avant G/X device information.<br>  • Updated all figures.<br>• In the Driver Installation on a Linux Machine section:<br>  • Updated the file names and directory paths to reflect the latest naming conventions.<br>  • Added a note on Linux driver compatibility.<br>  • Updated all figures. |
| Application Overview | • Updated the descriptions in the following sections:<br>  • Application Overview<br>  • AXI-MM DMA Transfer<br>  • AXI-MM DMA Transfer and Data Comparison<br>  • DMA Performance Measure<br>  • AXI-MM DMA Performance Measure<br>  • Read/Write Memory and I/O Addresses<br>  • Read/Write Mode<br>• Added Table 3.1. AXI-MM DMA Driver Device ID Requirement.<br>• Added the DMA Transfer with Incremented Data and Data Comparison section.<br>• Updated the following figures:<br>  • Figure 3.1. LATTICEDMA DMA Menu (including the caption)<br>  • Figure 3.3. LATTICEDMA Transfer Direction (including the caption)<br>  • Figure 3.5. LATTICEDMA Data Comparison (including the caption)<br>  • Figure 3.8. PCIe Link Status Register<br>  • Figure 3.9. LATTICEDMA Measure DMA Performance for Single-Direction Transfer (including the caption)<br>  • Figure 3.10. LATTICEDMA Measure DMA Performance for Bi-Directional Simultaneous Transfer (including the caption)<br>  • Figure 3.12. LATTICEDMA Read/Write Memory (including the caption)<br>• Updated the captions for Figure 3.13. LATTICEDMA Change Active Address Space for Read/Write, Figure 3.14. LATTICEDMA Change Active Read/Write Mode, and Figure 3.15. LATTICEDMA Non-block or Block Transfer. |
| APIs | Updated the description for *WDC_IntEnable*. |
| Software Flow Diagrams | Updated the following sections including all figures:<br>• LATTICEDMA_DmaOpen/CPNXAXISDMA_DmaOpen (including the figure caption)<br>• DmaPerformanceSingleDir<br>• DmaKpPerfDevThread |

| Section | Change Summary |
|---|---|
| References | • Added *PCIe ×1 IP Release Notes (FPGA-RN-02060)*, *PCIe ×8 IP Release Notes (FPGA-RN-02061)*, *PCIe ×1 IP Core User Guide (FPGA-IPUG-02091)*, and *PCIe ×8 IP Core User Guide (FPGA-IPUG-02243)*.<br>• Added *PCI Express Endpoint Core*, *PCI Express for Nexus FPGAs*, *PCI Express for Avant FPGAs*, *Avant-G*, *Avant-X*, *Certus-NX*, *MachXO5-NX*, and *Lattice Solutions Reference Designs* web pages. |

**Revision 1.2, April 2025**

| Section | Change Summary |
|---|---|
| All | • Added *and AXI-Stream* and *User Guide* to the document title.<br>• Made editorial fixes. |
| Abbreviations in This Document | Added *AXI-MM*, *AXI-S*, *BAR*, *RO*, and *RW*. |
| Introduction | • Updated the description in the Introduction section.<br>• Updated the Driver Version and Driver and IP Compatibility sections. |
| Software Setup | • Updated the description in the Software Setup section. |
| Application Overview | • Added the following figures:<br>  • Figure 3.2 CPNXAXISDMA DMA Menu<br>  • Figure 3.4 CPNXAXISDMA Transfer Direction<br>  • Figure 3.6 CPNXAXISDMA Data Compare<br>  • Figure 3.10 CPNXAXISDMA Measure DMA Performance<br>  • Figure 3.11 CPNXAXISDMA Read/Write Memory<br>  • Figure 3.12 CPNXAXISDMA Change Active Address Space for Read/Write<br>  • Figure 3.13 CPNXAXISDMA Change Active Read/Write Mode<br>  • Figure 3.14 CPNXAXISDMA Non-block or Block Transfer<br>  • Figure 3.15 Non-block Transfer – Write, Read, and Compare Values<br>  • Figure 3.16 Block Transfer – Write, Read, and Compare Values<br>• Added the AXI-MM DMA Transfer section header and updated its content.<br>• Updated the following figure captions:<br>  • Figure 3.3. CPNXDMA Transfer Direction<br>  • Figure 3.5. CPNXDMA Data Compare<br>  • Figure 3.8. CPNXDMA Measure DMA Performance for Single-Direction Transfer<br>  • Figure 3.9. CPNXDMA Measure DMA Performance for Bi-Directional Simultaneous Transfer<br>• Added the following sections:<br>  • AXI-S DMA Transfer<br>  • AXI-S DMA Transfer and Compare Data<br>  • AXI-S DMA Performance Measure<br>  • Read/Write Memory and I/O Addresses<br>• Added the following section headers:<br>  • AXI-MM DMA Transfer and Compare Data<br>  • AXI-MM DMA Performance Measure |
| APIs | • Updated the *WDC_IntEnable* description.<br>• Added the *WDC_DMASyncCpu* API. |
| Software Flow Diagrams | • Added *CPNXAXISDMA_DmaOpen* to the CPNXDMA_DmaOpen/CPNXAXISDMA_DmaOpen section header.<br>• Updated the following figures:<br>Figure 5.1. CPNXDMA_DmaOpen/CPNXAXISDMA_DmaOpen and its caption<br>Figure 5.2. DmaPerformanceSingleDir<br>Figure 5.3. DmaKpPerfDevThread |
| References | • Added the *PCIe x4 IP Release Notes (FPGA-RN-02059)*. |

**Revision 1.1, January 2025**

| Section | Change Summary |
|---|---|
| All | Made editorial fixes. |
| Abbreviations in This Document | Added *Field Programmable Gate Array* (*FPGA*) and *Random Access Memory* (*RAM*). |
| Introduction | • Updated the Driver Version section.<br>• Updated *Driver Version* in Table 1.1. Driver and IP Compatibility. |
| Software Setup | • Updated the driver version in the Driver Installation on a Windows Machine and Driver Installation on a Linux Machine sections.<br>• Updated the programming code in step 6 of the Driver Installation on a Linux Machine section.<br>• Updated the following figures and their captions:<br>  • Figure 2.1. Setup<br>  • Figure 2.2. Setup License Agreement<br>  • Figure 2.3. Setup Install Location<br>  • Figure 2.4. Setup Install<br>Figure 2.6. lsmod (caption not updated)<br>Figure 2.7. Run cpnxdma (caption not updated) |
| Application Overview | • Updated Figure 3.2. DMA Transfer Direction and Figure 3.3. DMA Data Compare.<br>• Updated the DMA Performance Measure section.<br>• Added Figure 3.5. Measure DMA performance for Single Direction Transfer.<br>• Updated Figure 3.6. Measure DMA Performance for Bi-Directional Simultaneous Transfer and its caption. |

**Revision 1.0, December 2024**

| Section | Change Summary |
|---|---|
| All | Initial release. |


www.latticesemi.com