Lattice Avant and Nexus PCIe Basic Memory-Mapped Host Driver (Non-DMA) User Guide

Technical Note

FPGA-TN-02387-1.3

October 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
BAR	Base Address Register
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
RO	Read-Only access
RW	Read-Write access
SGDMA	Scatter-Gather Direct Memory Access
SPI	Serial Peripheral Interface
SRAM	Static Random Access Memory
TLP	Transaction Layer Packet

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 driver software with the Avant and Nexus non-DMA example design.

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 non-DMA module. 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 Driver non-DMA version 25.01.01.

1.4. Driver and IP Compatibility

Table 1.1. Driver and IP Compatibility

Driver Version	IP Version
25.01.01	2.1.0 (PCIe ×1)
	3.6.0 (PCIe ×4)
	2.4.0 (PCIe ×8)

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) for more information on the driver and IP versions.

Table 1.2. Driver Tested Environment Quick Facts

IP	Tested Device	PC Environment
PCIe ×1	Certus-NX Versa Evaluation Board MachXO5-65T Development Board	Operating System: Windows 10, Windows 11, Linux
		Supported Architecture: x86_64
		CPU: Intel, AMD
PCIe ×4	CertusPro-NX PCIe Bridge Board CertusPro-NX Versa Evaluation Board	Operating System: Windows 10, Windows 11, Linux
		Supported Architecture: x86_64
		CPU: Intel, AMD
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.

Note: The installation steps are consistent across all driver versions. The driver version shown in screenshots or file directories may vary depending on 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*.

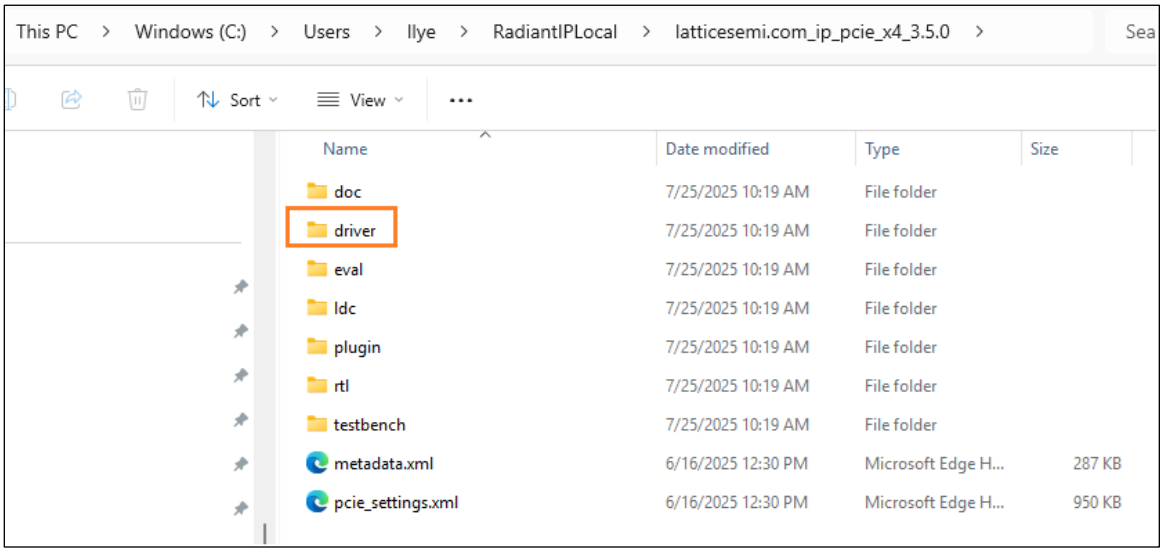

Figure 2.1. Driver File Location

2.2. Driver Installation on 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\lattice_24.02.01* before you install the driver.
3. Double click on *lattice-24.02.01-win64.exe* to install the driver. Select **Next** to continue with the installation.

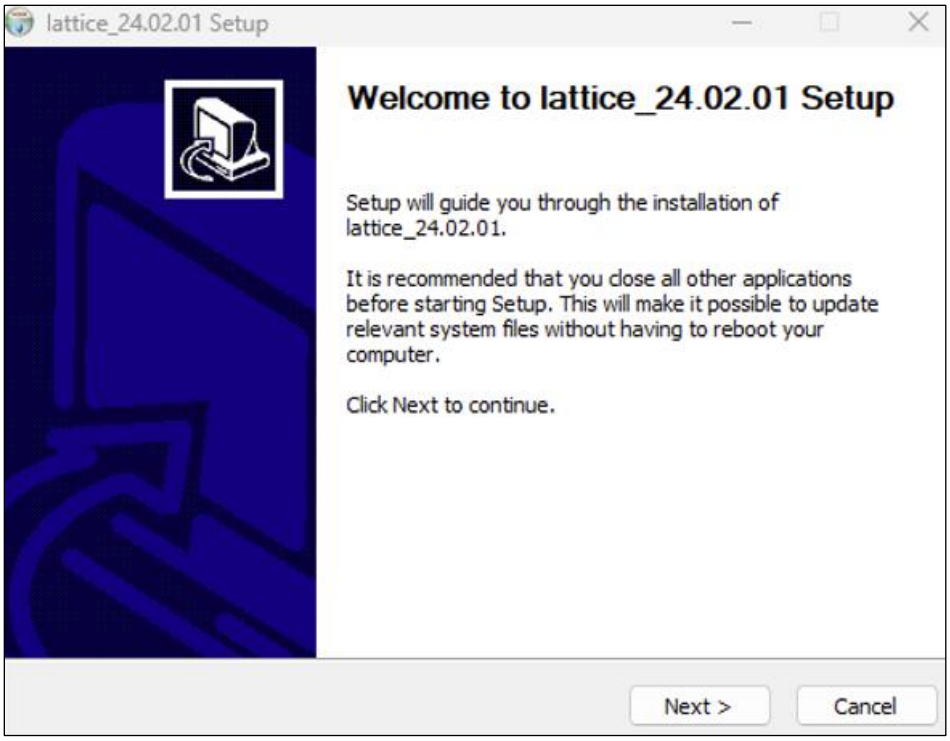

Figure 2.2. Setup

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

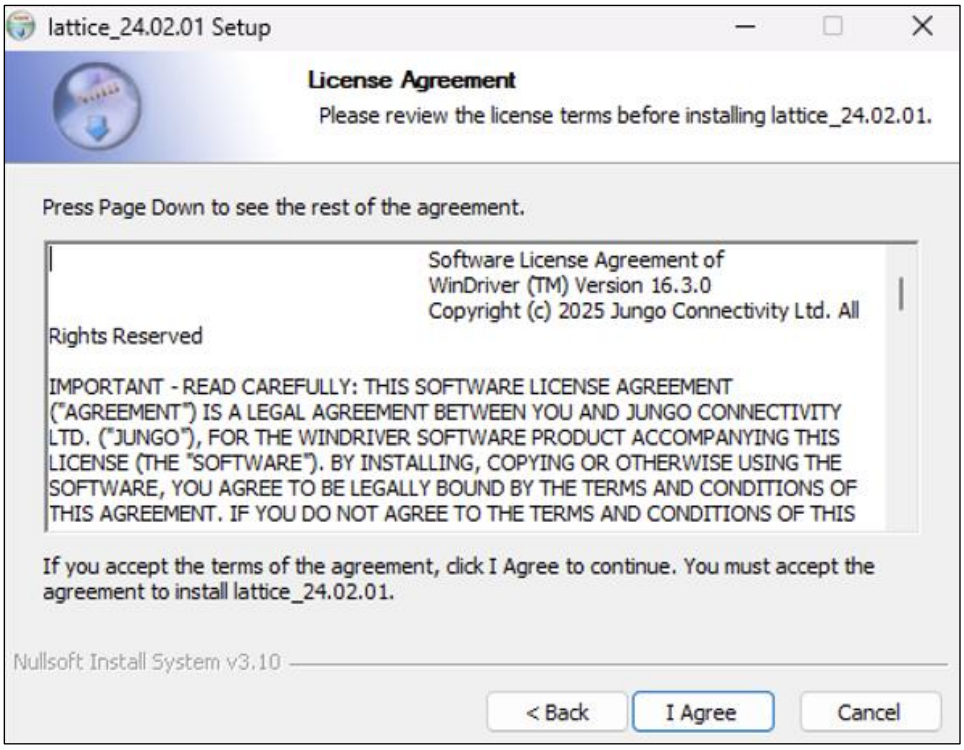

Figure 2.3. Setup License Agreement

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

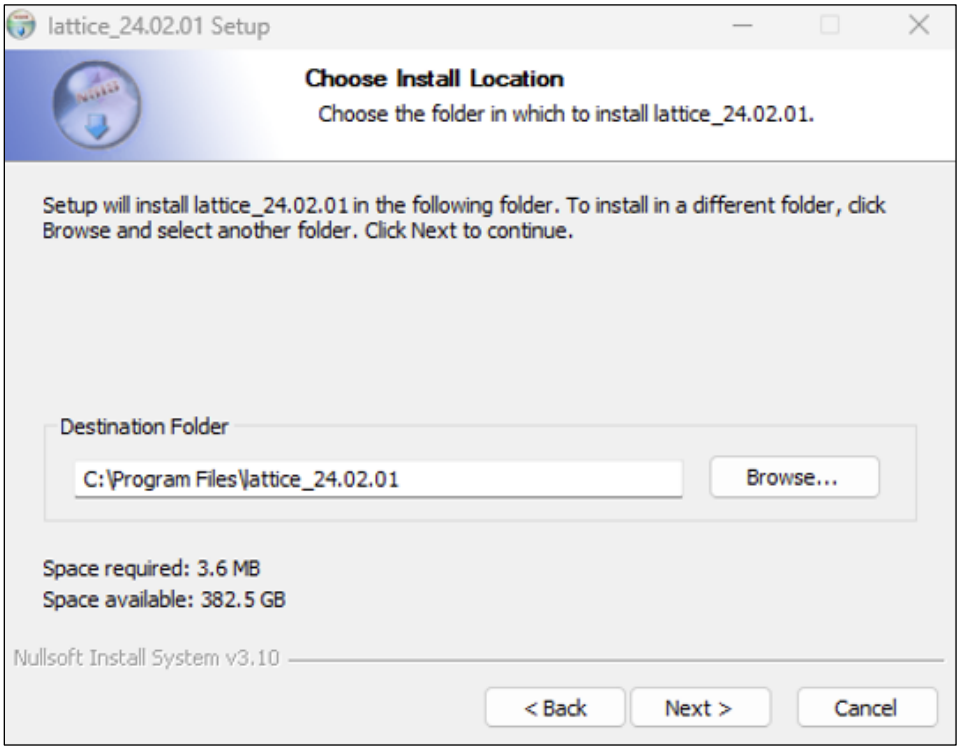

Figure 2.4. Setup Install Location

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

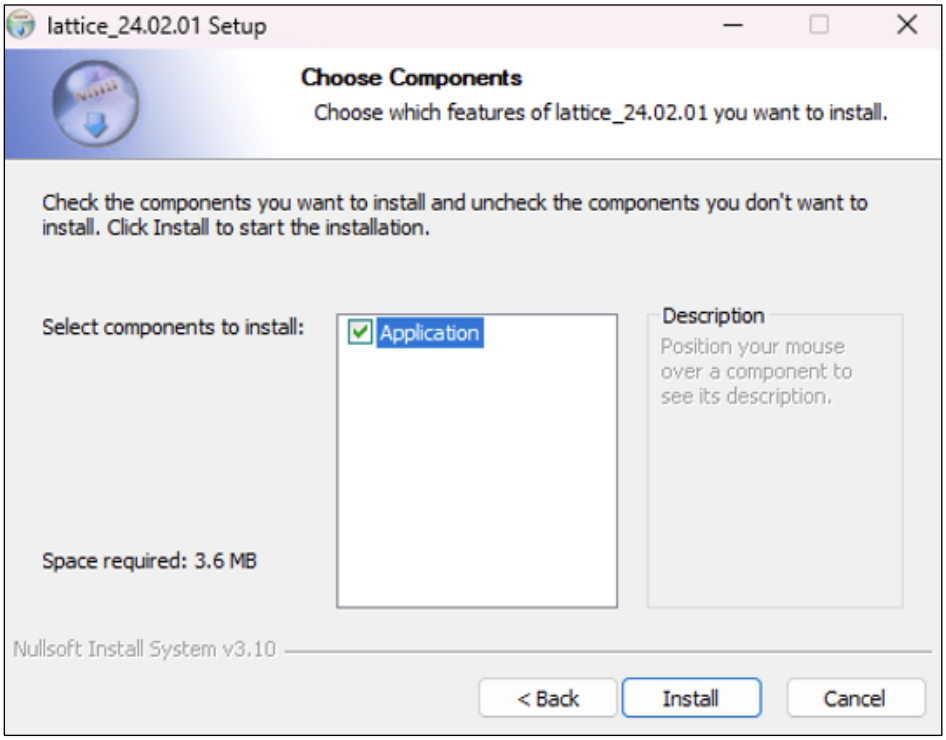

Figure 2.5. Setup Install

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 devices:
     - *lattice Driver*
     - *PCIe (Lattice - Device ID : 0x9c25 (Requiring driver: lattice)*
   - For Certus-NX or MachXO5-NX (LFMXO5-65T) devices:
     - *lattice Driver*
     - *PCIe (Lattice - Device ID : 0x9c1d (Requiring driver: lattice)*

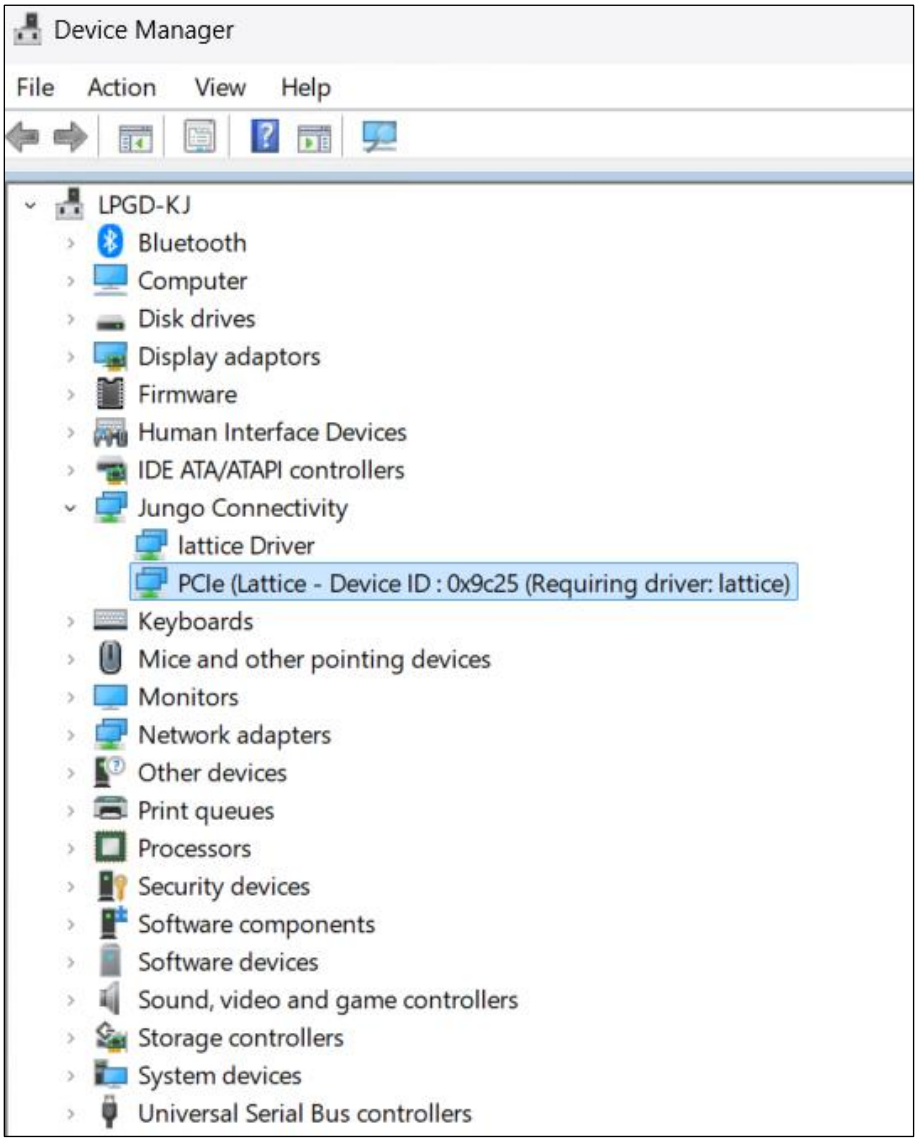

Figure 2.6. Device Manager for CertusPro-NX or Avant Devices

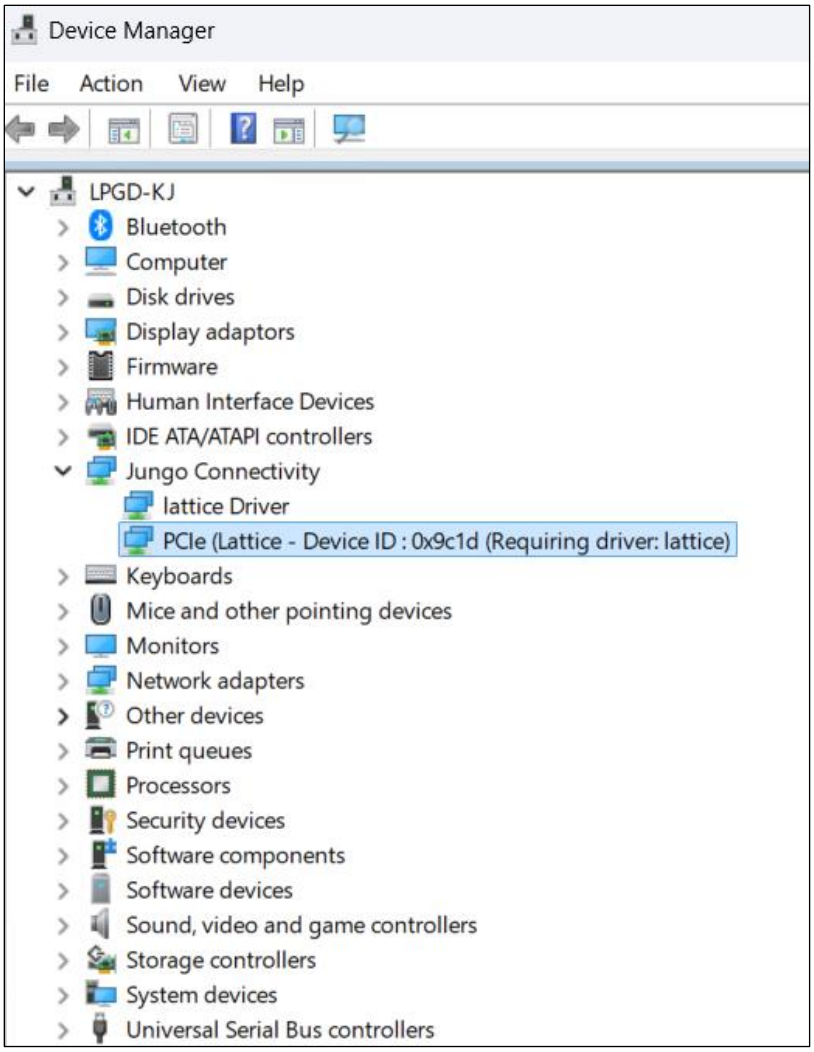

Figure 2.7. Device Manager for Certus-NX or MachXO5-NX (LFMXO5-65T) Devices

8. A new folder is created in *C:\Program Files\lattice_24.02.01*.
9. A shortcut: *lattice_24.02.01* is added to your desktop.
10. You can run the user application by running the desktop shortcut or by running: *C:\Program Files\lattice_24.02.01\bin\lattice.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 Linux Machine

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 *lattice-24.02.01-Linux.sh* is located. Give executable permission to the *.sh* file:
```
sudo chmod +x lattice-24.02.01-Linux.sh
```

4. Then, run the command below to build the driver and user application and install the driver.
```
sudo ./lattice-24.02.01-Linux.sh
```

5. Enter *Y* when prompted.
6. A new module is installed. Run the command below to verify that *lattice* is installed:
```
sudo lsmod | grep lattice
```

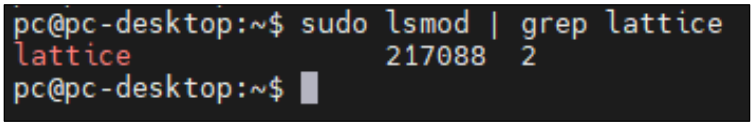

Figure 2.8. lsmod

7. A new folder is created in your current directory *lattice-24.02.01-Linux*.
8. Give permission to all the files in the directory:
```
sudo chmod 777 -R *
```

9. Run the user application:
```
sudo ./lattice-24.02.01-Linux/bin/lattice
```

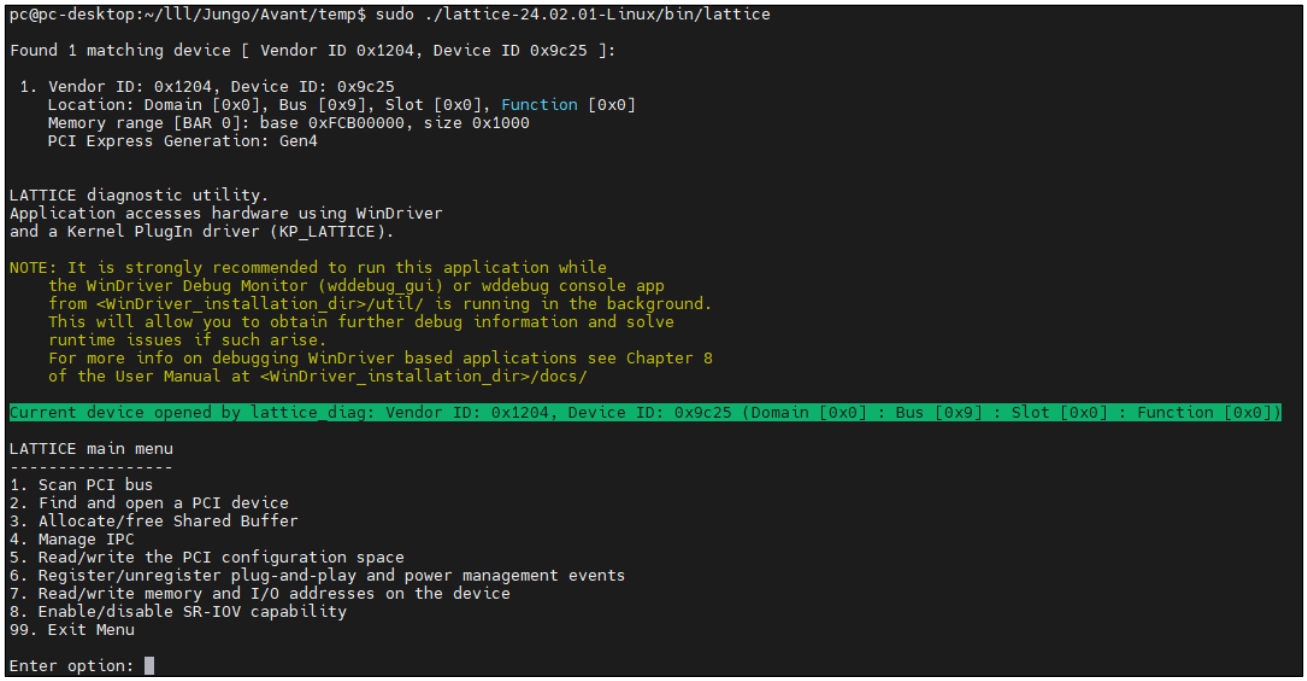

Figure 2.9. Run the Lattice User Application

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.

Table 3.1. Windows Driver Device ID Requirement

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

Note: This applies to the Windows driver only. When creating the example design, make sure the PCIe device ID is set according to the table above, based on the IP used. Otherwise, the driver does not recognize your device. Supporting a different device ID requires modifications to the driver files.

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

```
LATTICE main menu
-----------------
1. Scan PCI bus
2. Find and open a PCI device
3. Allocate/free Shared Buffer
4. Manage IPC
5. Read/write the PCI configuration space
6. Register/unregister plug-and-play and power management events
7. Read/write memory and I/O addresses on the device
8. Enable/disable SR-IOV capability
99. Exit Menu

Enter option: 7

Current Read/Write configurations:
----------------------------------
Currently active address space : BAR 0
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:
```

Figure 3.1. Read/Write Memory Submenu

3.1. Read/Write Memory and I/O Addresses

In the non-DMA TLP mode example design, only BAR0 is enabled for user memory or I/O read and write operations.

In the bridge mode example design, 2 BARs are enabled:

- BAR0: Used for internal bridge mode registers. BAR0 memory is read-only (RO).
- BAR1: Used for user memory or I/O read and write operations. BAR1 memory is read-write (RW).

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

```
Current Read/Write configurations:
----------------------------------
Currently active address space : BAR 0 (RO): Bridge Mode internal registers

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
99. Exit Menu

Enter option: 1

Select an active address space:
-------------------------------
1. BAR 0                  Memory  0x00000000DD2F0000 - 0x00000000DD2FFFFF (0x10000 bytes) (RO): Bridge Mode internal registers

2. BAR 1                  Memory  0x00000000DD2E0000 - 0x00000000DD2EFFFF (0x10000 bytes) (RW)

Enter option (to cancel press 'x'): 2

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:
```

Figure 3.2. Bridge Mode Example Design – Change Active Address Space for Read/Write

3.1.1. Read/Write Mode

The application supports read/write mode of 8-bit, 16-bit, 32-bit, and 64-bit. However, the bridge mode example design does not support 64-bit read/write operations.

You can change the read/write mode by selecting **Change active read/write mode**.

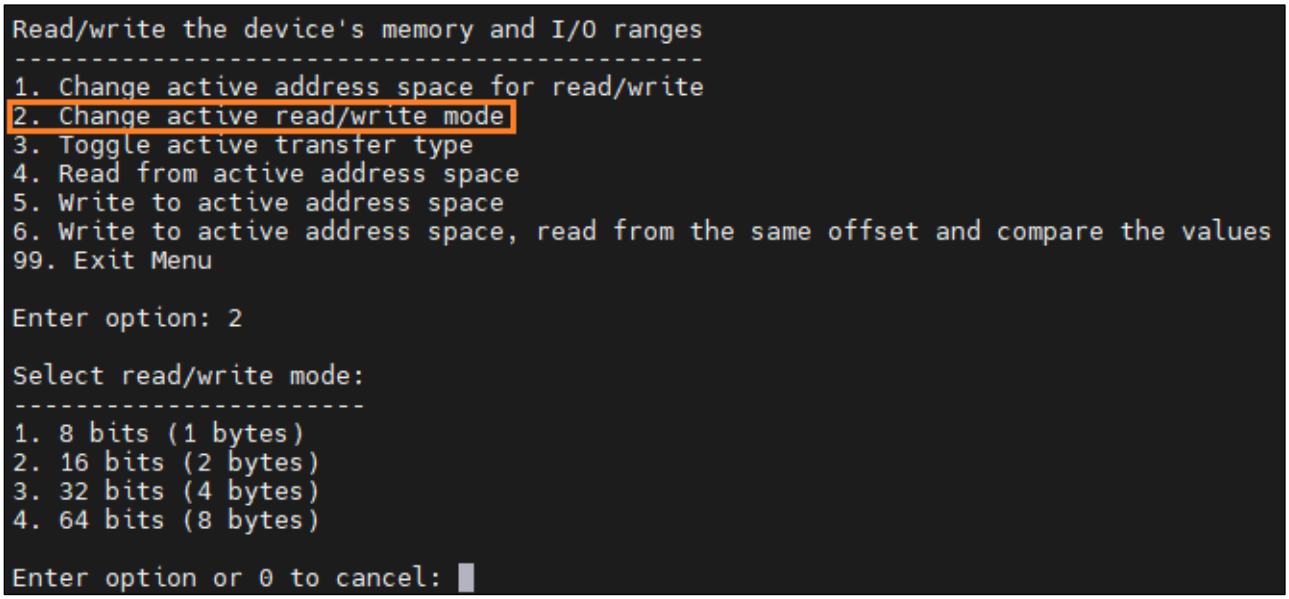

Figure 3.3. Read/Write Mode Selection

3.1.2. Read/Write Offset Address

When performing read/write operations to the active address space, you are prompted to enter the offset address.

For bridge mode design only, ensure the offset address is such that the modulus of the offset by the number of bytes for the selected read/write mode (8-bit, 16-bit, or 32-bit) returns *0*. For example, if the 16-bit mode is selected, the offset address should be entered at 0x0, 0x02, 0x04, 0x06, 0x08, and so on. Here, the offset address is a multiple of 2 bytes, and the offset address modulus 2 returns *0*. If 32-bit mode is selected, the offset address should be entered at 0x0, 0x04, 0x08, 0x0C and so on. Here, the offset address is a multiple of 4 bytes, and the offset address modulus 4 return *0*. This applies to both non-block and block transfers.

```
Current Read/Write configurations:
----------------------------------
Currently active address space : BAR 0 (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: 5
offset (to cancel press 'x'): 0x4
Enter data to write (max value: 0xFFFFFFFF) or 'x' to cancel: 0x12345678
Wrote 0x12345678 to offset 0x4 in BAR 0
```

Figure 3.4. Read/Write Offset

3.1.3. Transfer Type: Non-block or Block Transfers

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

```
Current Read/Write configurations:
----------------------------------
Currently active address space : BAR 0 (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: 3

Current Read/Write configurations:
----------------------------------
Currently active address space : BAR 0 (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 byte pattern from user and duplicate it
```

Figure 3.5. Non-Block or Block Transfers

3.1.3.1. Non-block Transfer

Non-block transfer refers to read/write to a single unit of the 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.

```
Current Read/Write configurations:
----------------------------------
Currently active address space : BAR 0
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: 4
offset (to cancel press 'x'): 0x0
Read 0x78563412 from offset 0x0 in BAR 0
```

Figure 3.6. Non-Block Transfer 32-Bit Read

3.1.3.2. Block Transfer

Block transfer allows you to read/write data to/from a block of memory. When you select block transfer, you are provided with additional options for the *address mode* and *block transfer writing mode*.

```
Current Read/Write configurations:
----------------------------------
Currently active address space : BAR 0 (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 byte pattern from user and duplicate 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:
```

Figure 3.7. Block Transfer

3.1.4. Block Transfer Address Increment Mode

When toggling the active block transfer address increment mode, it changes the address mode between *Auto-increment offset* and *Don't increment offset*.

Don't increment offset address mode selection reads/writes to the same memory location, while the *Auto-increment offset* selection increases the address on each read/write operation.

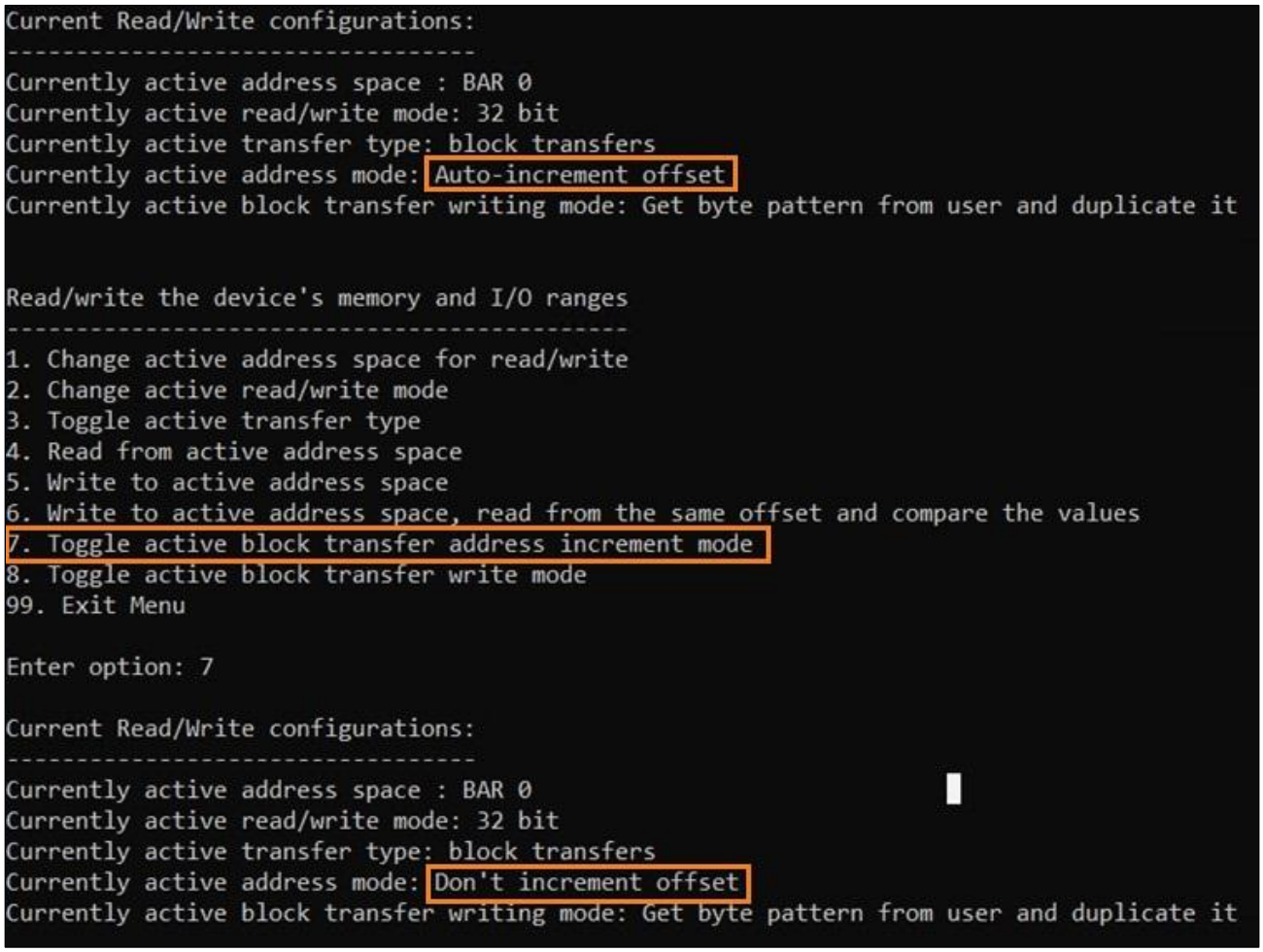

Figure 3.8. Toggling Block Transfer Address Increment Mode

3.1.5. Block Transfer Write Mode

When selecting *Change active block transfer write mode*, you are given the following options: *Get entire buffer from user*, *Get byte pattern from user and duplicate it*, or *Get start value from user and increment it*.

```
Current Read/Write configurations:
----------------------------------
Currently active address space : BAR 0 (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 byte pattern from user and duplicate 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: 8

Select block transfer write mode:
---------------------------------
1. Get entire buffer from user
2. Get byte pattern from user and duplicate it
3. Get start value from user and increment it
99. Exit Menu

Enter option:
```

Figure 3.9. Block Transfer Write Mode

If you select the *Get byte pattern from user and duplicate it* block transfer writing mode, after selecting *Write to active address space* or *Write to active address space, read from the same offset and compare the values,* you are prompted to select a buffer size of 256, 512, 1,024, or 4,096 bytes and to enter the data to write. The driver application then fills the write buffer with duplicated data that you entered.

```
Current Read/Write configurations:
----------------------------------
Currently active address space : BAR 0
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 byte pattern from user and duplicate 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. Toggle active block transfer write mode
99. Exit Menu

Enter option: 5
offset (to cancel press 'x'): 0x0

Select write buffer size:
-------------------------
1. 256 bytes
2. 512 bytes
3. 1024 bytes
4. 4096 bytes
99. Exit Menu

Enter option: 2
Enter data to write (max value: 0xFFFFFFFF) or 'x' to cancel: 0x12345678
Wrote 0x200 bytes to offset 0x0
```

Figure 3.10. Block Transfer Write – Get Byte Pattern from User and Duplicate It

If you select the *Get entire buffer from user* block transfer writing mode, you can manually enter the entire data for the buffer from the console.

```
Current Read/Write configurations:
----------------------------------
Currently active address space : BAR 0
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 entire buffer from user


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. Toggle active block transfer write mode
99. Exit Menu

Enter option: 5
offset (to cancel press 'x'): 0x0
data to write (Bytes in hex format): 0x123456781234567812345678
Wrote 0xc bytes to offset 0x0
```

Figure 3.11. Block Transfer Write – Get Entire Buffer from User

If you select the *Get start value from user and increment it* block transfer writing mode, after selecting *Write to active address space* or *Write to active address space, read from the same offset and compare the values,* you are prompted to enter the starting value for the incremented data. The driver application then fills the write buffer with incremental data, starting with the value that you entered.

```
Current Read/Write configurations:
----------------------------------
Currently active address space : BAR 0 (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: 5
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

Press ENTER to continue
```

Figure 3.12. Block Transfer Write – Get Start Value from User and Increment It

3.1.6. Write, Read, and Compare Values

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

```
Current Read/Write configurations:
----------------------------------
Currently active address space : BAR 0 (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 byte pattern from user and duplicate 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

Select write buffer size:
-------------------------
1. 256 bytes
2. 512 bytes
3. 1024 bytes
4. 4096 bytes
99. Exit Menu

Enter option: 1
Enter data to write (max value: 0xFFFFFFFF) or 'x' to cancel: 0x12345678
Wrote 0x100 bytes to offset 0x0

78 56 34 12 78 56 34 12 78 56 34 12 78 56 34 12
78 56 34 12 78 56 34 12 78 56 34 12 78 56 34 12
78 56 34 12 78 56 34 12 78 56 34 12 78 56 34 12
78 56 34 12 78 56 34 12 78 56 34 12 78 56 34 12
78 56 34 12 78 56 34 12 78 56 34 12 78 56 34 12
78 56 34 12 78 56 34 12 78 56 34 12 78 56 34 12
78 56 34 12 78 56 34 12 78 56 34 12 78 56 34 12
78 56 34 12 78 56 34 12 78 56 34 12 78 56 34 12
78 56 34 12 78 56 34 12 78 56 34 12 78 56 34 12
78 56 34 12 78 56 34 12 78 56 34 12 78 56 34 12
78 56 34 12 78 56 34 12 78 56 34 12 78 56 34 12
78 56 34 12 78 56 34 12 78 56 34 12 78 56 34 12
78 56 34 12 78 56 34 12 78 56 34 12 78 56 34 12
78 56 34 12 78 56 34 12 78 56 34 12 78 56 34 12
78 56 34 12 78 56 34 12 78 56 34 12 78 56 34 12
78 56 34 12 78 56 34 12 78 56 34 12 78 56 34 12

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

Figure 3.13. Block Transfer – Write, Read, and Compare Values with Duplicated Data

```
Current Read/Write configurations:
----------------------------------
Currently active address space : BAR 0 (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 3.14. Block Transfer – Write, Read, and Compare Values with Incremental Data

4. APIs

The APIs are defined by Jungo WinDriver. The following shows the list of APIs used by the host PCIe driver to read/write from/to PCIe memory mapped to the Base Address Register (BAR) as well as to read/write from/to PCIe configuration space. For more details, refer to the links below:

- WDC_DriverOpen
- WDC_ReadAddr8
- WDC_ReadAddr16
- WDC_ReadAddr32
- WDC_ReadAddr64
- WDC_WriteAddr8
- WDC_WriteAddr16
- WDC_WriteAddr32
- WDC_WriteAddr64
- WDC_GetBusType
- WDC_PciReadCfg
- WDC_PciReadCfg8
- WDC_PciReadCfg16
- WDC_PciReadCfg32
- WDC_PciReadCfg64
- WDC_PciWriteCfg
- WDC_PciWriteCfg8
- WDC_PciWriteCfg16
- WDC_PciWriteCfg32
- WDC_PciWriteCfg64
- WDC_PciGetExpressOffset
- PciExpressConfRegData2Str
- PciConfRegData2Str
- WDC_PciGetExpressGen
- WDC_PciGetExpressOffset

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
- Certus-NX web page
- CertusPro-NX web page
- MachXO5-NX web page
- Avant-G web page
- Avant-X web page
- Lattice Solutions IP Cores 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, October 2025

Section	Change Summary
All	• Updated the document title by replacing *CertusPro-NX and Avant-G/X* with *Lattice Avant and Nexus*. • Made editorial fixes.
Abbreviations in This Document	Added *GCC*, *ID*, and *TLP*.
Introduction	• Updated the descriptions in the Introduction, Purpose, Audience, Driver Version, and Driver and IP Compatibility sections. • 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. • Added the Driver File Location section. • Updated step 7 in the Driver Installation on Windows Machine section. • Updated Figure 2.6. Device Manager for CertusPro-NX or Avant Devices and Figure 2.7. Device Manager for Certus-NX or MachXO5-NX (LFMXO5-65T) Devices. • Updated the programming code in step 8 of the Driver Installation on Linux Machine section.
Application Overview	• Added Table 3.1. Windows Driver Device ID Requirement and *Note* to the Application Overview section. • Updated the descriptions in the Read/Write Mode, Read/Write Offset Address, and Non-block Transfer sections.
References	Added *PCIe ×1 IP Core User Guide (FPGA-IPUG-02091)*, *PCIe ×1 IP Release Notes (FPGA-RN-02060)*, *Certus-NX web page*, and *MachXO5-NX web page*.

Revision 1.2, June 2025

Section	Change Summary
Abbreviations in This Document	Added *RO* and *RW*.
Introduction	Updated Table 1.1. Driver and IP Compatibility.
Software Setup	Updated the description in the Software Setup section.
Application Overview	Added Figure 3.2. Bridge Mode Example Design – Change Active Address Space for Read/Write and its description to the Read/Write Memory and I/O Addresses section.

Revision 1.1, March 2025

Section	Change Summary
All	Renamed the document title from *PCIe Basic Memory-Mapped (Non-DMA) Host Driver for CertusPro-NX Devices* to *CertusPro-NX and Avant-G/X PCIe Basic Memory-Mapped Host Driver (Non-DMA) User Guide*.

Introduction	• Updated the from *non-dma module* to *non-DMA example design*.
	• Added the *Avant-G/X* information to the following sections and tables:
	  ◦ Introduction
	  ◦ Audience
	  ◦ Table 1.1. Driver and IP Compatibility
	  ◦ Table 1.2. Driver Tested Environment Quick Facts
	• Added the IP release notes information to the Driver and IP Compatibility section.

Software Setup	• Updated the driver version from *24.02.00* to *24.02.01* in the Driver Installation on Windows Machine and Driver Installation on Linux Machine sections. • Updated Figure 2.1. Setup – Figure 2.4. Setup Install and Figure 2.8. Run the Lattice User Application. • Updated step *7* in the Driver Installation on Windows Machine section.

Section	Change Summary
	• Updated the caption for Figure 2.5. Device Manager for CertusPro-NX Devices. • Added Figure 2.6. Device Manager for Avant Devices. • Updated *gcc -version* to *gcc --version* in the Driver Installation on Linux Machine section.

Application Overview	• Updated the description in the Read/Write Mode section.
	• Added the Read/Write Offset Address section.
	• Added *Increment* to the Block Transfer Address Increment Mode section title.
	• Updated the descriptions in the Block Transfer Write Mode section.
	• Updated the following figures:
	  • Figure 3.2. Read/Write Mode Selection
	  • Figure 3.4. Non-Block or Block Transfers
	  • Figure 3.6. Block Transfer
	  • Figure 3.8. Block Transfer Write Mode and its caption
	  • Figure 3.12. Block Transfer – Write, Read, and Compare Values with Duplicated Data and its caption
	• Added the following figures:
	  • Figure 3.11. Block Transfer Write – Get Start Value from User and Increment It and its description
	  • Figure 3.13. Block Transfer – Write, Read, and Compare Values with Incremental Data

References	Added the *PCIe X8 IP Core User Guide (FPGA-IPUG-02243)*, *PCIe x4 IP Release Notes (FPGA-RN-02059)*, and *PCIe x8 IP Release Notes (FPGA-RN-02061)*, and the *PCI Express for Avant FPGAs*, *Avant-G*, and *Avant-X* web pages.

Revision 1.0, January 2025

Section	Change Summary
All	Initial release.

www.latticesemi.com