PCIe Multifunction Demo for Lattice Nexus-based FPGAs

User Guide

FPGA-UG-02150-1.2

May 2023

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 associated risk the responsibility entirely of the Buyer. Buyer shall not rely on any data and performance specifications or parameters provided herein. 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. No Lattice products should be used in conjunction with mission- or safety-critical or any other application in which the failure of Lattice's product could create a situation where personal injury, death, severe property or environmental damage may occur. 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.

Contents

Figures

Tables

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
API	Application Programming Interface
BAR	Base Address Register
BIOS	Basic Input/Ouput System
DLL	Dynamic Link Library
FDSOI	Fully Depleted Silicon on Insulator
FPGA	Field-Programmable Gate Array
LED	Light-emitting diode
MIPI	Mobile Industry Processor Interface
PCIe	PCI Express
PHY	Physical Layer
SPI	Serial Peripheral Interface
USB	Universal Serial Bus
I^2C	Inter-integrated Circuit
MDIO	Management Data Input Output
GPIO	General Purpose Input Output
OS	Operating System

1. Introduction

This guide describes how to set up and run the PCIe Multifunction Demo using devices built on the Lattice Nexus™ platform, specifically CrossLink™-NX, Certus™-NX, CertusPro™-NX, MachXO5-NX™ devices.

For CrossLink-NX devices, the demo is targeted to the CrossLink-NX PCIe Bridge Board, which features the CrossLink-NX FPGA in a 400-ball caBGA package (LIFCL-40-8BG400C). For Certus-NX devices, the demo is targeted to the Certus-NX Versa Evaluation Board, which features the Certus-NX FPGA in a 256-ball caBGA package (LFD2NX-40-8BG256C). For CertusPro-NX devices, the demo is targeted to the CertusPro-NX Versa Evaluation Board, which features the CertusPro-NX FPGA in a LFG672 package. For MachXO5-NX, the demo is targeted to the MachXO5T-NX Development Board, which features the LFMX05-100T FPGA in the BBG400 package. The above-mentioned FPGAs are built on the Nexus FPGA platform using low power 28 nm FDSOI technology.

This guide familiarizes with the process of setting up the PCI Express development environment. It is assumed that user do not have any associated tools installed on the system.

The demo discussed in this document is the PCI Express Multifunction Demo.

1.1. Learning Objectives

After completing the steps in this guide, user will be able to perform the following:

- Set up and install all applicable development tools and PCI Express demos.
- Establish communication between the FPGA and the system through the PCI Express link.
- Run the PCI Express Multifunction Demo which implements three separate PCI Express functions on a single endpoint device. Each function allows the user to control a different aspect of the FPGA Board.
- Use what the demo teaches about designing Lattice PCI Express solutions.
- Modify and rebuild the PCI Express Multifunction Demo.
- Become familiar with the software development tools and major design flow steps employed in this kit.
- Use other existing documentation in conjunction with this guide.

This document assumes that user have already installed the Lattice Radiant™ design software. This document covers some of the basic functions of the Lattice Radiant software. If user would like to learn more about the Lattice Radiant software, refer to the Lattice Radiant software Help system.

2. Hardware and Software Requirements

2.1. Hardware Requirements

To install the kit design and run the demo software, a computer with a PCI Express ×16, ×8, ×4, or ×1 slot is required. The computer must also have a USB port and be able to run the Lattice Radiant software. All other hardware and drivers are included in the kit.

- Mini-USB to USB-A cable for programming the bitstream
- 12 V Power Adapter
- Evaluation Board
  - CrossLink-NX PCIe Bridge Board for the CrossLink-NX FPGA
  - Certus-NX Versa Evaluation Board for the Certus-NX FPGA
  - CertusPro-NX Versa Evaluation Board for the CertusPro-NX FPGA
  - MachXO5T-NX Development Board for MachXO5-NX

Additionally, for Certus-NX and CertusPro-NX devices, a MIPI CSI Camera Module is needed to test I^2C functionality; this demo uses the Sony IMX258-0AQH5 Camera Module. The CrossLink-NX PCIe Bridge Board should already contain the camera module. The MachXO5T-NX Development Board does not provide an interface to connect a camera module, hence only incorporates test involving DIP switches and LEDs.

2.2. Software Requirements

The following software applications are required to obtain the expected results in the procedures described in this guide:

- Lattice Radiant software version 2.2 or later (available at the Lattice website Design Software and IP page)
- PCIe Multifunction Demo for Windows 10 or Linux
- Windows 10 or Ubuntu 18.04.6
- Bit file for the SPI Flash
- LIFCL40_PCIe_MultiFunction.bit file for CrossLink-NX devices
- LFD2NX_PCIe_MF.bit file for Certus-NX devices
- LFCPNX_100_PCIe_Multifunction.bit file for CertusPro-NX devices
- MachXO5TNX_PCIE_MF_DEMO.bit file for MachXO5-NX devices

3. Demo Design Overview

3.1. Theory of Operation

The demo runs on a standard ×64 PC and accesses the FPGA board installed on a PCIe slot. Figure 3.1 shows the relationship between the hardware and software components of the demo. The PCIe IP on the Lattice FPGA acts as a PCIe endpoint occupying certain ranges of PCI memory space. When the computer boots, the BIOS and OS probe the PCI Express and PCI buses to detect devices present on the buses and assign the devices ranges in the PCI memory space. The BIOS maps the PCI memory space to the memory space in the PC. After the device driver is installed, the software can read from and write to the PCIe device memory (Embedded Block RAM - EBR). The software can also read from the PCIe configuration space registers. The Multifunction demo implements, one (for MachXO5-NX) or two (for CertusPro-NX devices) or three (for CrossLink-NX and Certus-NX devices) separate PCIe functions within a single endpoint.

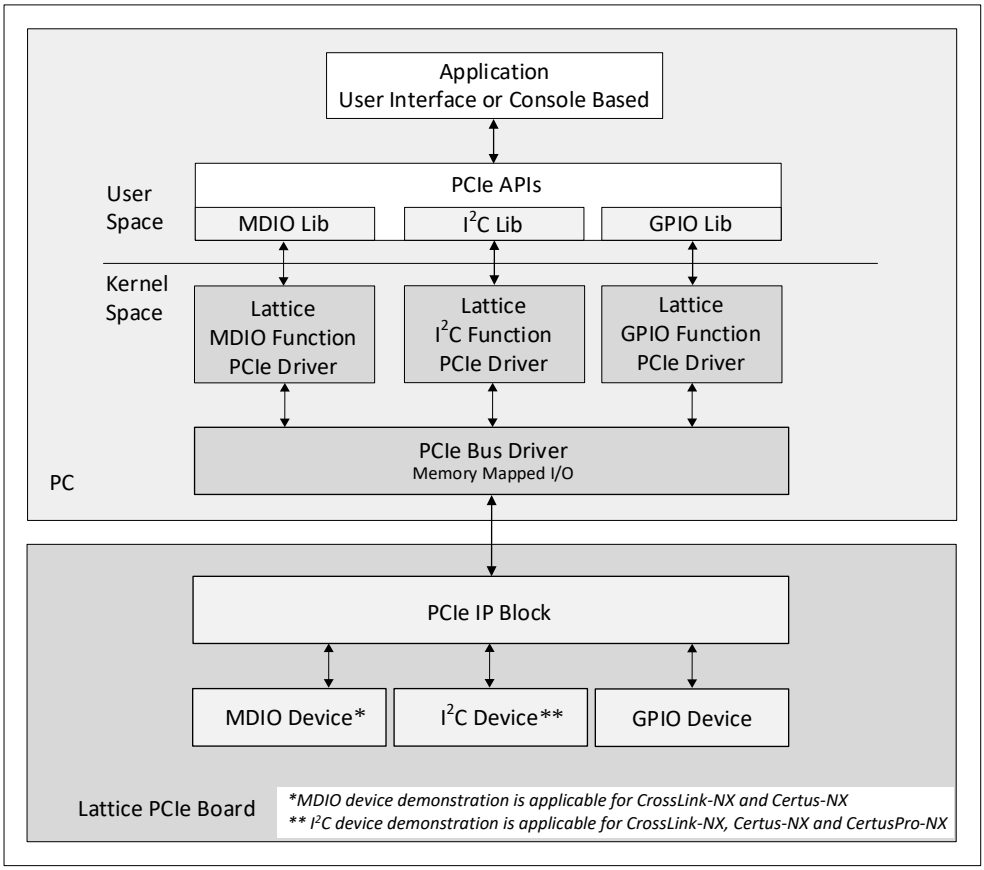


Figure 3.1. Relationship between Hardware and Software Components

3.2. Design Overview

The included user interface application demonstrates the PCIe Multifunction design. The application software is developed using a layered architecture consisting of the following layers:

- User interface application
- Driver API
- Device Drivers
- Device Hardware (FPGA Design)

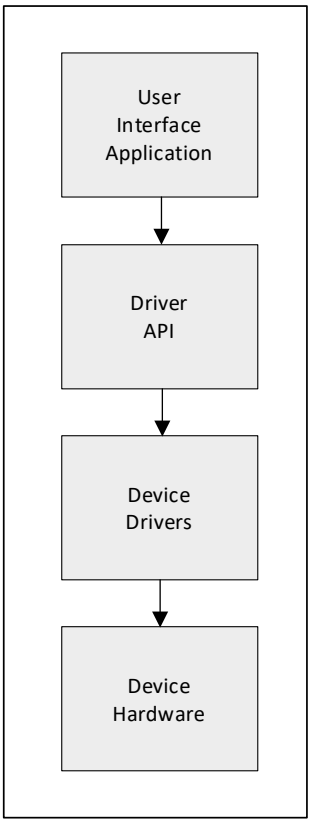


Figure 3.2. PCIe Multifunction Demo SW Design

3.2.1. User Interface Application

The Multifunction demo includes a user interface written in Qt. The user interface uses the driver API to communicate with the device hardware to send control plane read/write instructions to registers in the IP. The Driver API is a C++ dynamic library that provides an interface to access the hardware.

On Windows, the *multifunction_lib.dll* library is a DLL that bridges the user space demo applications to the kernel space driver code and provides routines to control the IP modules.

On Linux, the library is called *libmem_rw.so*.

3.2.2. Device Drivers

Hardware drivers provide access to the FPGA board. On Windows, the *lpcie_gpio.sys*, *lpcie_i2c.sys*, and *lpcie_mdio.sys* drivers support the Multifunction demo. On Linux, the *gpio_main.ko*, *i2c_main.ko*, and *mdio_main.ko* drivers support the demo. Note that the MDIO driver is only applicable to CrossLink-NX and Certus-NX devices.

3.2.3. Device Hardware (FPGA Design)

Figure 3.1 shows the top-level architecture of the FPGA design. The design uses the PCIe hard IP on the FPGA to implement the PCIe endpoint. The endpoint interfaces with the application logic for each function through an arbiter. For initial configuration of the PCIe IP, the LMMI interface is used.

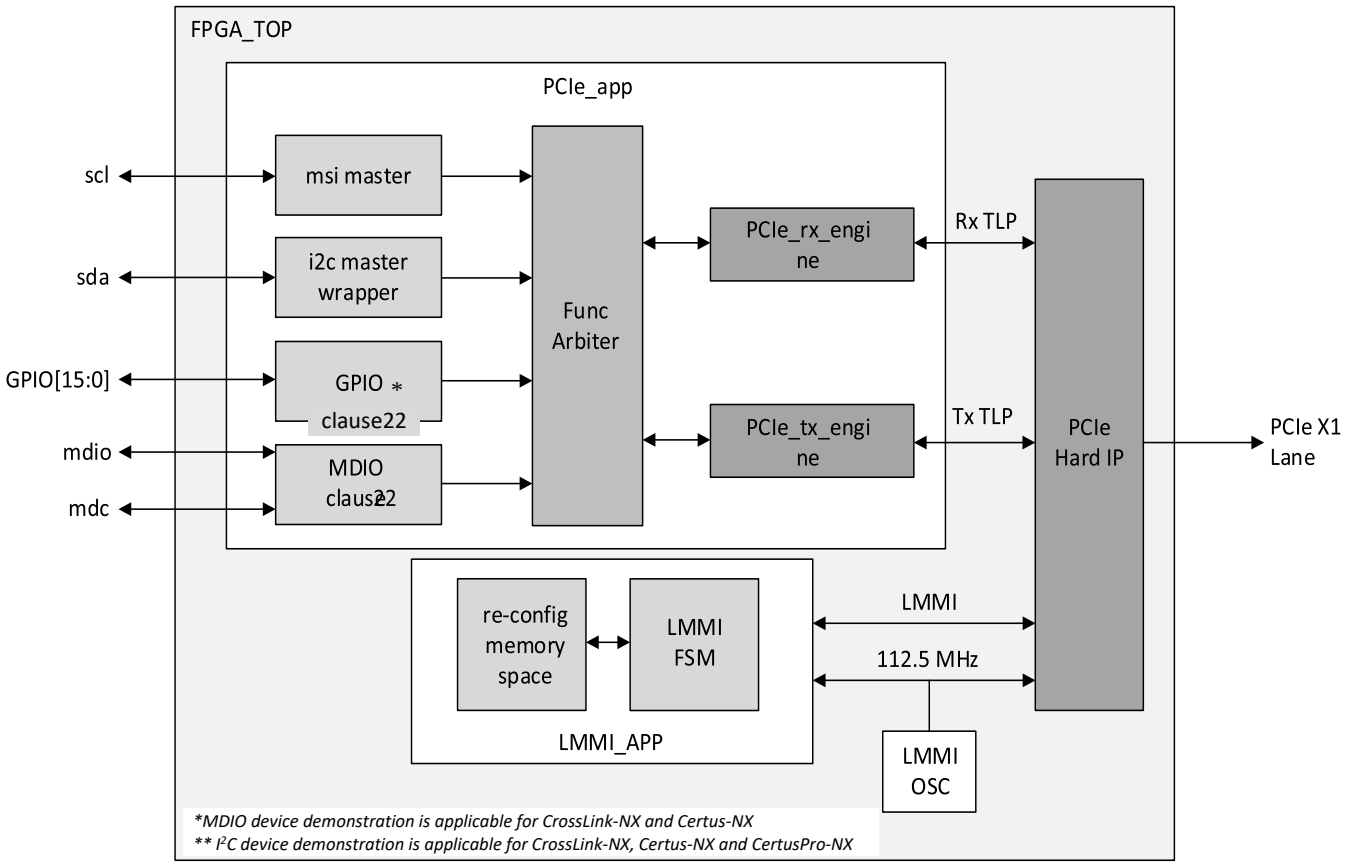


Figure 3.3. PCIe Multifunction FPGA Design

4. Importing and Building the FPGA Demonstration

The design package includes the PCIe IP, .bit file, and synthesis projects created with the Lattice Radiant software.

4.1. Hardware Directory Structure

The Hardware folder inside the package contains the following subfolders.

./CL_NX_BridgeBoard_PCIeMultifunctionDemo, ./CertusNX_PCIe_Multifunction, or ./CertusPro_NX_PCIe_Multifunction

- IP – Contains the pre-generated IPs used in the design. These IPs can be configured by clicking the *.ipx* file after opening the project in the Lattice Radiant software.
- Implementation – Contains the Lattice Radiant project (*.rdf*) file, constraints file (*.pdc*), and implemented design and bit files.
- Source – Contains RTL files required for the design.

4.2. Building a Lattice Radiant Project

To generate the bit stream file:

1. Open the Lattice Radiant software.
2. Click **Open project** and browse to the .rdf file
   a. For CrossLink-NX, the file is *LIFCL_PCIe_MF.rdf*, which is located in the *Hardware\CL_NX_BridgeBoard_PCIeMultifunctionDemo\Implementation\LIFCL-40_PCIe_MF* folder.
   b. For Certus-NX, the file is LFD2NX_PCIe_MF.rdf, which is located in the *Hardware\CertusNX_PCIe_Multifunction\Implementation\LFD2NX-40_PCIe_MF* folder.
   c. For CertusPro-NX, the file is LFCPNX_100_PCIe_Multifunction.rdf, which is located in the *Hardware\CertusPro_NX_PCIe_Multifunction\Implementation\LFCPNX_100_PCIe_Multifunction* folder.
   d. For MachXO5-NX, the file is LFC MachXO5TNX_PCIE_MF_DEMO.rdf, which is located in the *Hardware\MachXO5T-NX-PCIe-Multifunction-Demo folder*.
3. When user project loads, click **Task Detail View**. This shows a list of actions that the Lattice Radiant software performs to build the .bit file.
4. Select the files and reports that user want to generate.

   Note: The software selects options needed to regenerate the .bit file by default.
5. After selecting preferred reports, click **Run All**.

   This creates a .bit file with user project's current name in the *impl_1* folder.

5. Setting Up the Demo

5.1. Hardware Setup

This section covers the steps in programming the demo to the SPI memory of the FPGA Board.

5.1.1. Jumper Configuration

For the CrossLink-NX board, install the jumpers listed in Table 5.1.

Table 5.1. Jumper Configuration

Jumper Checklist	Configuration
J11, J18	Connect pins 1 and 2
J7, J6, J13, J14	Connect pins 1 and 2
J31, J32, J331	Connect pins 1 and 2
J8, J9, J10, J15, J16, J19, J20, J21, J22, J23, J25, J26, J29, J30, J34, J35, J36, J37, J38, J39, J42	*These are current measurement headers and should be kept Open.*
J51	Connect pins 2 and 4
J48	Connect pins 2 and 3
J27	Connect pins 1 and 2
J45, J46	Connect pins 1 and 2
J17	Open

Note:

1. J31, J32, and J33 are different from the default board configuration.

Figure 5.1 shows the location of the jumpers.

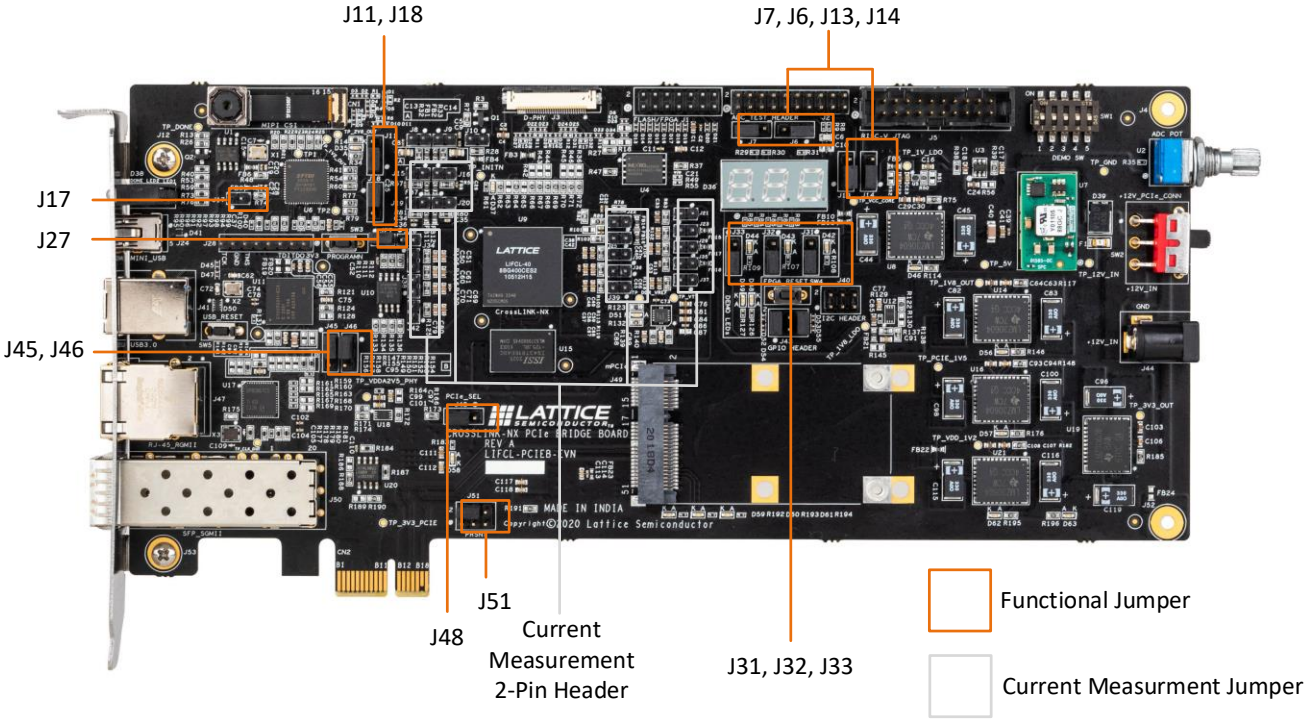


Figure 5.1. CrossLink-NX PCIe Bridge Board Jumper Location

Make sure the card is plugged into a PCIe slot or connected to the 12 V input supply. If the card is plugged into the PCIe slot, external power is provided by the system, and SW2 should be in the *up* position to receive power from the PCIe slot. If external 12 V power is provided, then SW2 should be in the *down* position to receive power from the external 12 V connection. Connect the board to the PC running the Lattice Radiant software with the Mini USB Type A cable as shown in Figure 5.2.

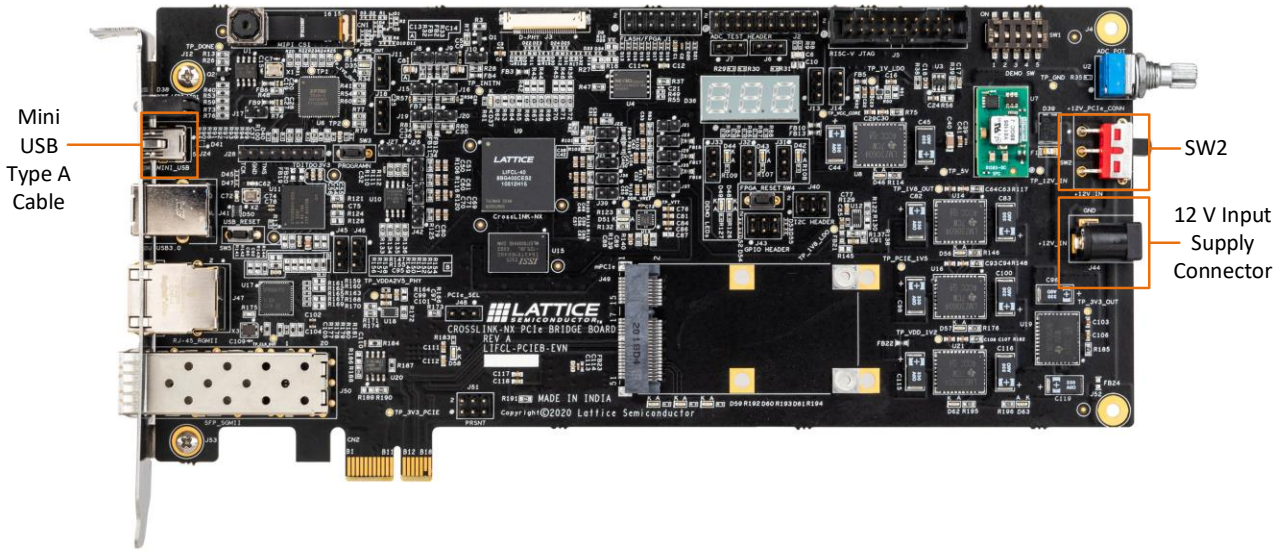


Figure 5.2. CrossLink-NX PCIe Bridge Board Connection

For the Certus-NX board, there are no jumpers to configure beyond the default. Refer to the Certus-NX Evaluation Board User Guide (FPGA-EB-02032) for the default jumper configuration.

There is no toggle switch for selecting the power source. The board can be powered through the PCIe slot or from an external 12 V input supply. If the board is powered from the PCIe slot, the external 12 V input supply connector does not need to be connected. Connect the board to the PC running the Lattice Radiant software with the Mini USB Type A Cable as shown in Figure 5.3.

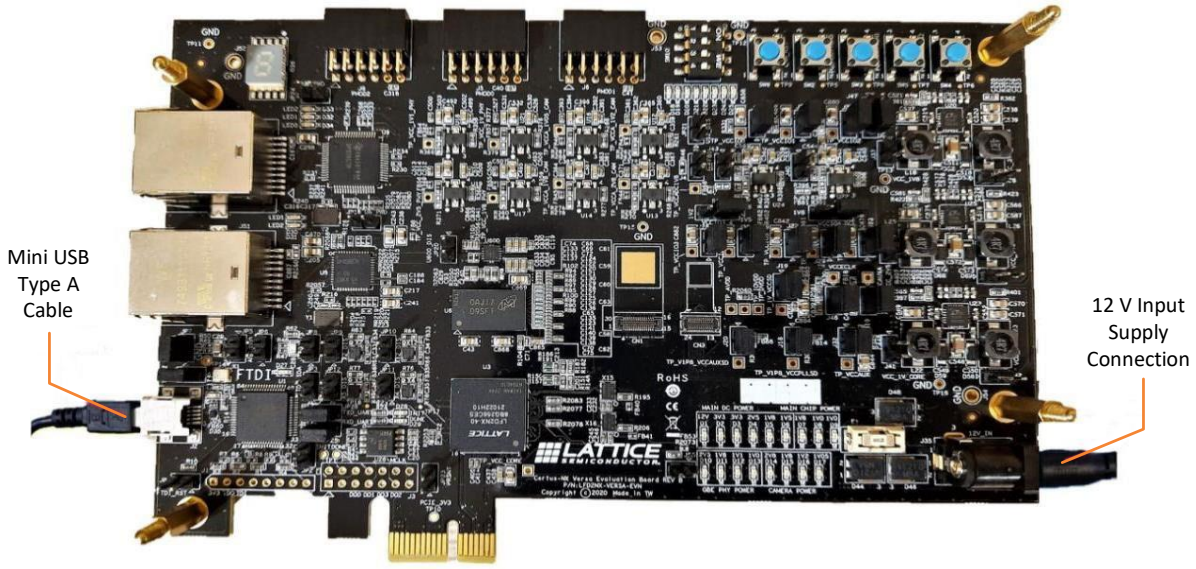


Figure 5.3. Certus-NX PCIe Versa Evaluation Board Connection

For the CertusPro-NX board, there are no jumpers to configure beyond the default. Refer to the CertusPro-NX Versa Board User Guide (FPGA-EB-02053) for a detailed list of jumpers on the board. when the board is plugged into the PCIe slot, external power is provided by the system, and SW6 should be in the *up* position to receive power from the PCIe slot. Connect the board to the PC running the Lattice Radiant software with the Mini USB Type A Cable as shown in Figure 5.4.

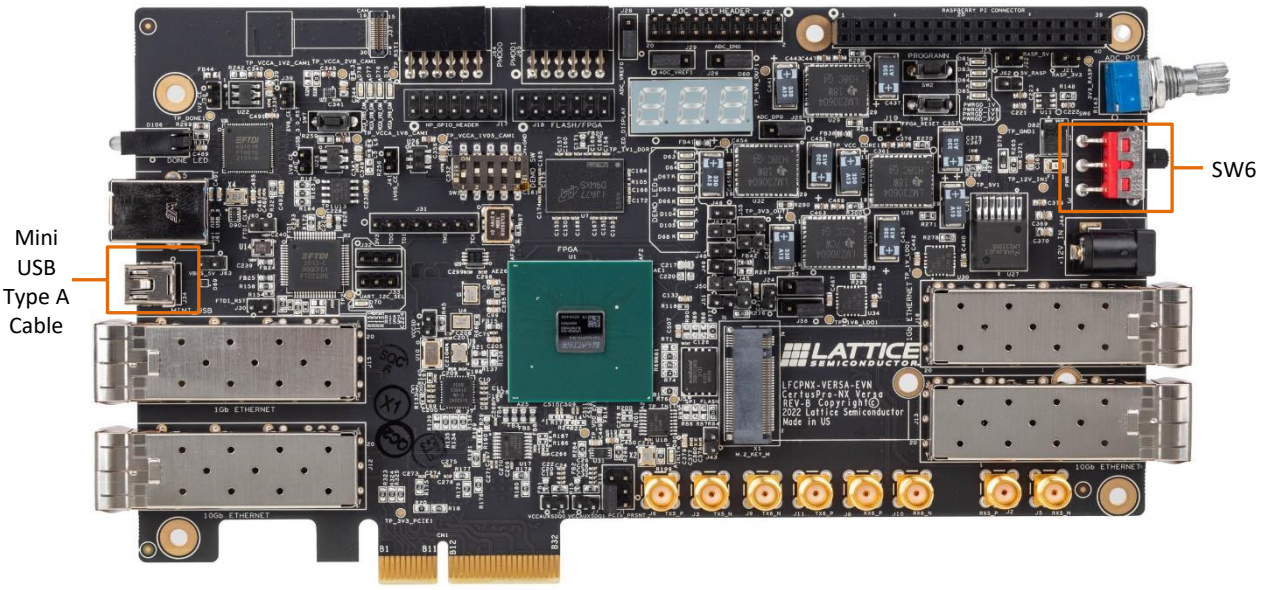


Figure 5.4. CertusPro-NX Versa Evaluation Board Connection

For MachXO5-NX, install the jumpers listed in Table 5.2 beyond the default. Refer to the MachXO5T-NX Development Board User Guide (FPGA-EB-02052) for a detailed list of jumpers on the board. when the board is plugged into the PCIe slot, external power is provided by the system, and SW6 should be in the up position to receive power from the PCIe slot. Connect the board to the PC running the Lattice Radiant software with the Mini USB Type A Cable as shown in Figure 5.5.

Table 5.2. Jumper Configuration

Jumper Checklist	Configuration
JP10	Connect pins 1 and 2
JP4	Connect pins 1 and 2

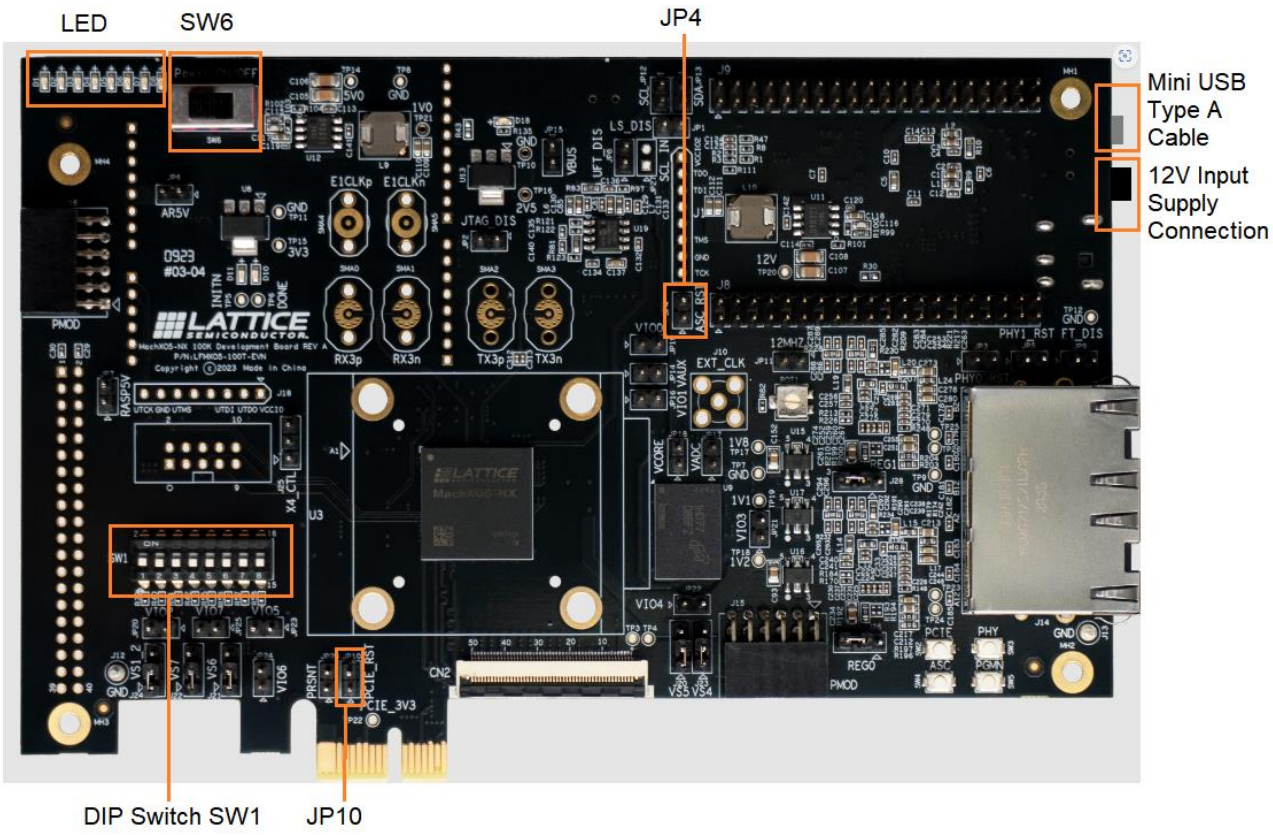


Figure 5.5. MachXO5-NX Development Board Connection

5.1.2. Programming the FPGA

To program the FPGA device:

1. Create a new project using the Lattice Radiant Programmer software. In the **Getting Started** dialog box, input the **Project Name** and **Location** as shown in Figure 5.6.
2. Select **Create a new project from a scan**. Values are indicated in the **Cable**, **Port**, and **TCK Divider Setting (0-30×)** fields.
3. Click **OK**.

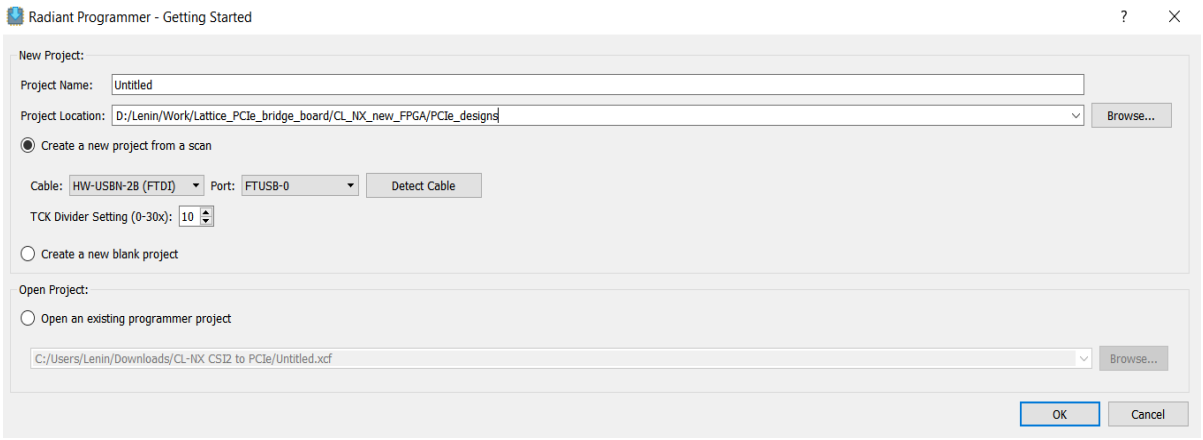


Figure 5.6. Creating a New Project from a Scan

4. The main interface opens as shown in Figure 5.7.

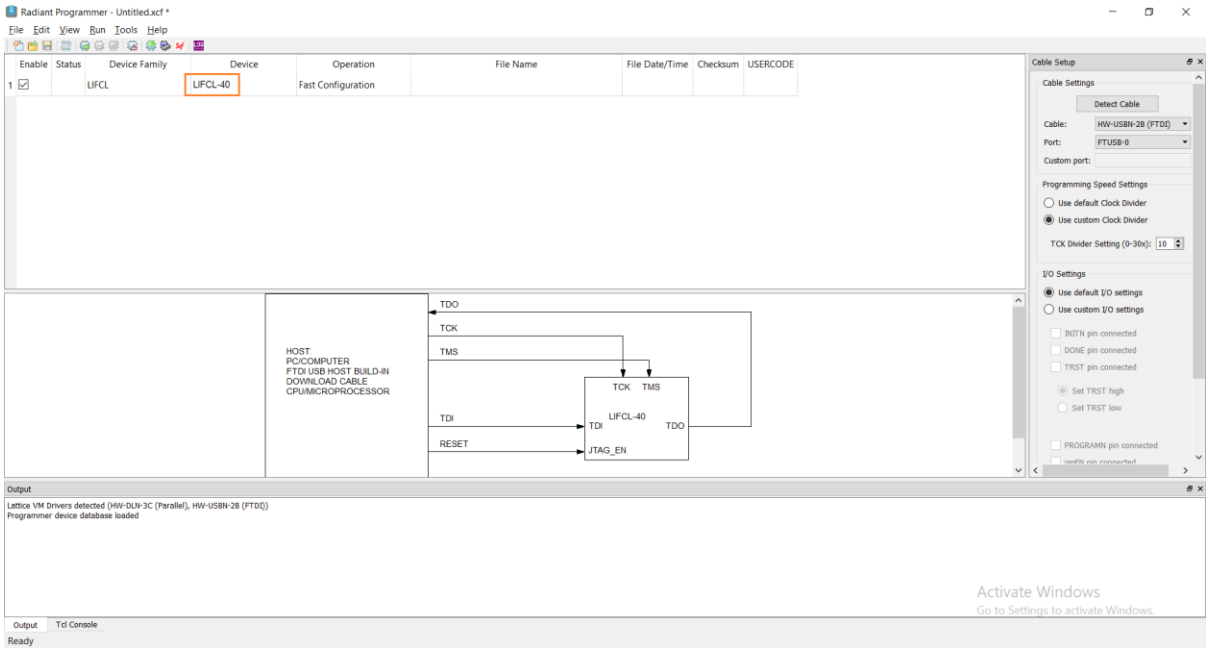


Figure 5.7. Lattice Radiant Programmer Window

5. If the Programmer settings do not match the settings shown in Figure 5.8 for CrossLink-NX devices, Figure 5.9 for Certus-NX devices, or Figure 5.10 for CertusPro-NX devices, or Figure 5.11 for MachXO5-NX select these settings manually from the drop down menu.

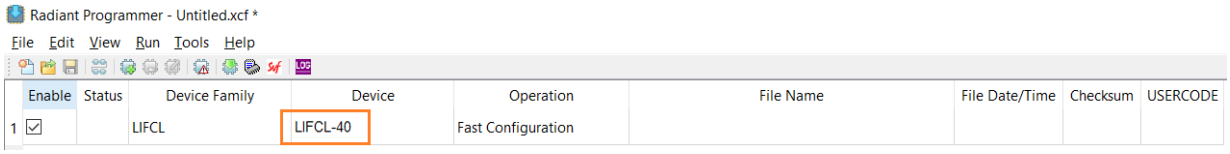


Figure 5.8. CrossLink-NX FPGA Device Settings

Figure 5.9. Certus-NX FPGA Device Settings

Figure 5.10. CertusPro-NX FPGA Device Settings

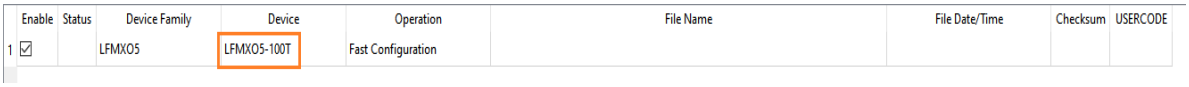


Figure 5.11. MachXO5-NX FPGA Device Settings

To select programming settings:

1. Browse and select the Programming file from the *Demonstration\Bitstream* folder.
   a. For CrossLink-NX devices, select *LIFCL40_PCIe_Multifunction.bit*.
   b. For Certus-NX devices, select *LFD2NX_PCIe_MF.bit*.
   c. For CertusPro-NX devices, select *LFCPNX_100_PCIe_Multifunction.bit*.
   d. For MachXO5-NX, select *MachXO5TNX_PCIE_MF_DEMO.bit*.
2. Click **OK**.
3. Double-click under **Operation** to open the **Device Properties** dialog box.
4. Select the settings as shown in Figure 5.12 for CrossLink-NX devices, Figure 5.13 for Certus-NX devices, or Figure 5.14 for CertusPro-NX devices.
5. For MachXO5-NX, Flash programming, select the Flash Header Programming Options and CFG0 Programming Options. The .msc file and jed.file can be found in the following locations:

   Hardware\MachXO5T-NX-PCIe-Multifunction-Demo\impl_1\MachXO5TNX_PCIE_MF_DEMO_header.mcs

   Hardware\MachXO5T-NX-PCIe-Multifunction-Demo\impl_1\MachXO5TNX_PCIE_MF_DEMO_0.jed

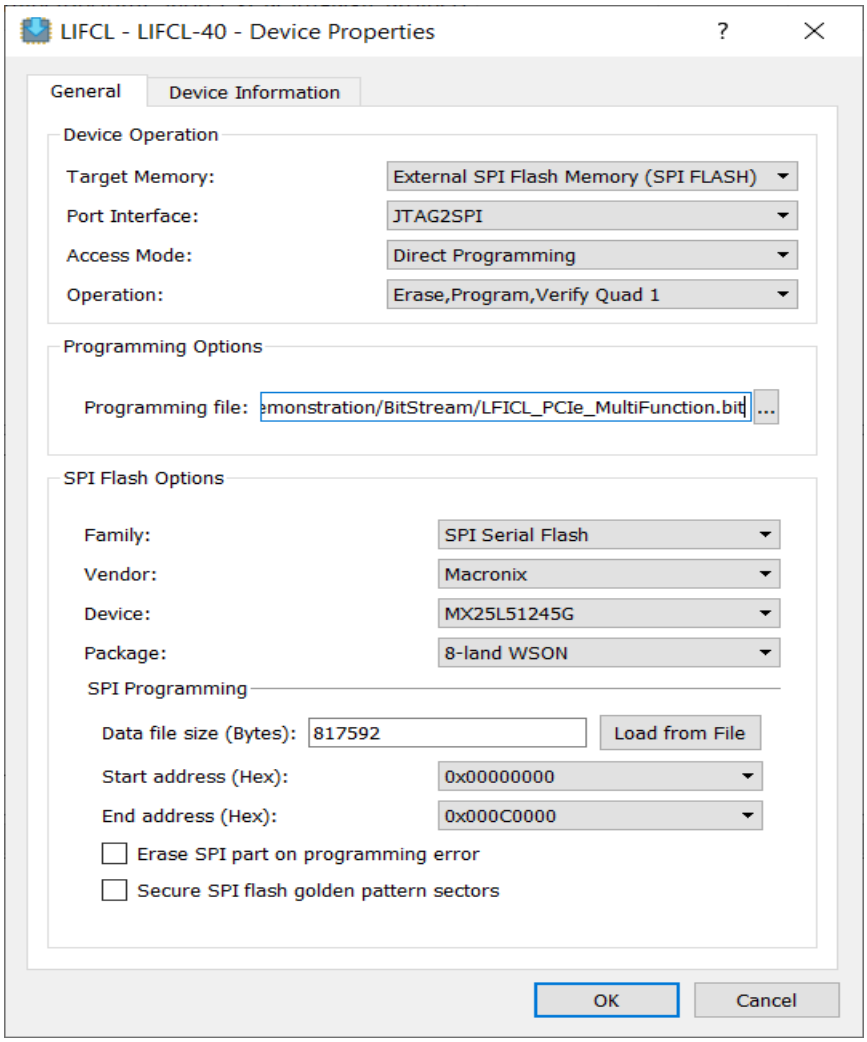


Figure 5.12. Device Properties Window for CrossLink-NX SPI Flash Programming

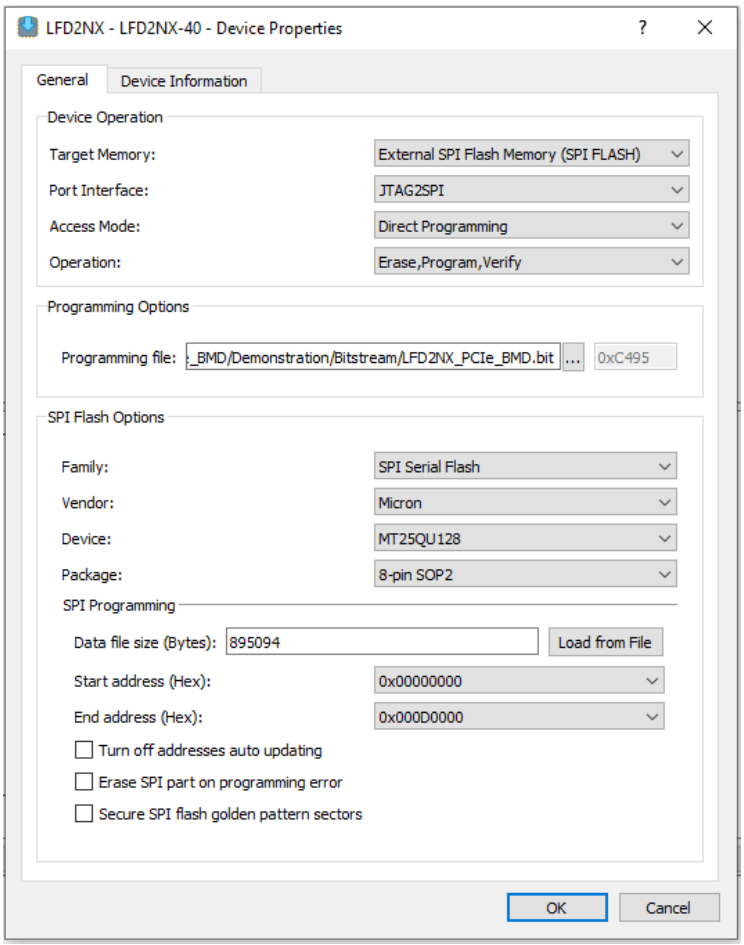


Figure 5.13. Device Properties Window for Certus-NX SPI Flash Programming

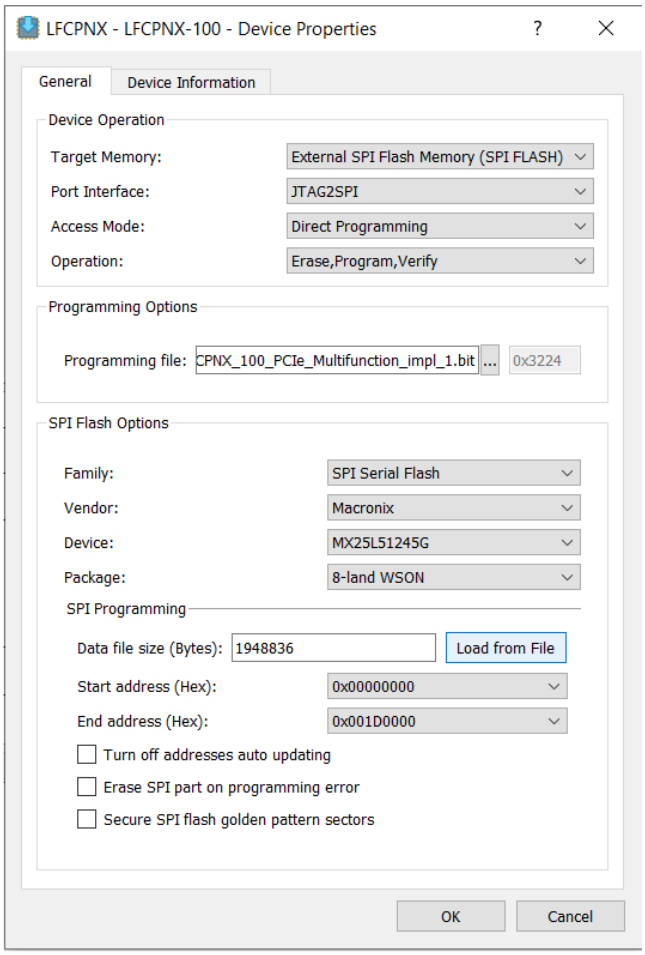


Figure 5.14. Device Properties Window for CertusPro-NX SPI Flash Programming (with Macronix Flash)

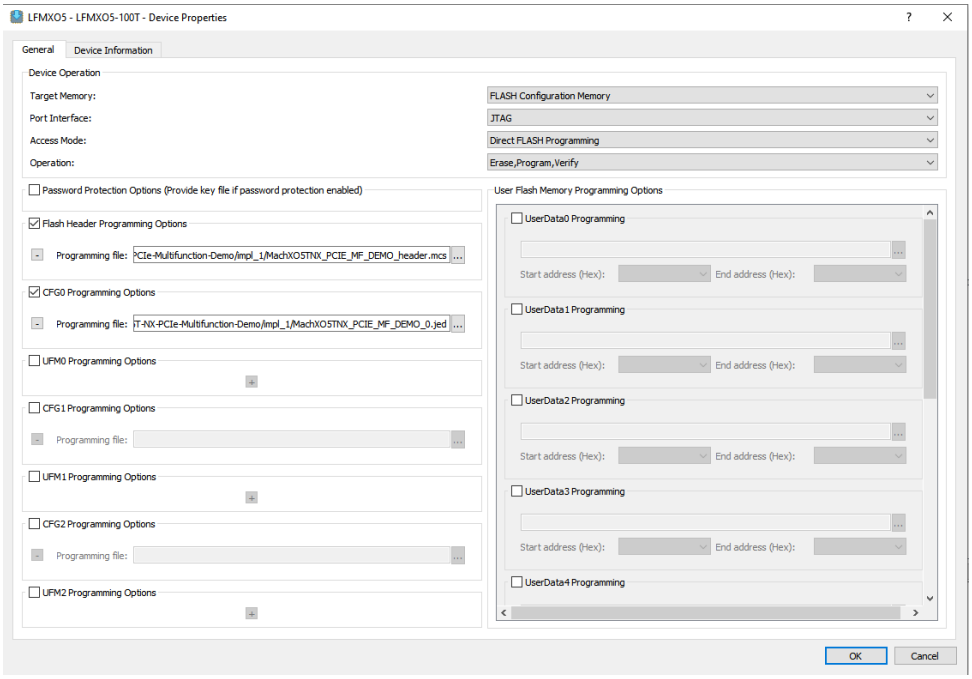


Figure 5.15. Device Properties Window for MachXO5-NX Flash Programming

6. Click the **Programming** button from the menu bar shown in Figure 5.16 to start programming.

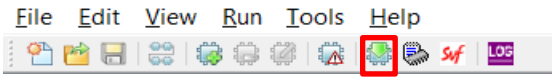


Figure 5.16. Programmer Menu Bar

When the FPGA programming is successful, the output console displays an **Operation: successful** message as shown in Figure 5.17.

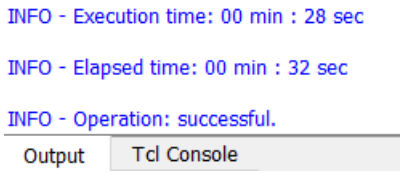


Figure 5.17. Programmer Output Window

If the programming operation is unsuccessful, refer to the Troubleshooting section of this document.

After programming, power cycle the board and check the status LEDs on the board. The LED status is discussed in the next section.

5.1.3. Status LED

For the Crosslink-NX board, the three status LEDs are shown in Figure 5.18.

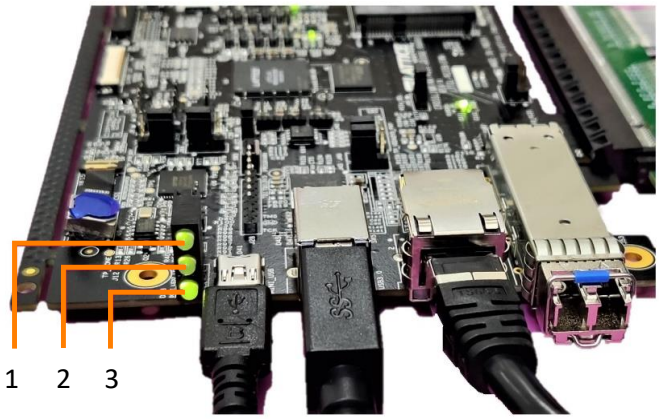


Figure 5.18. CrossLink-NX Status LED

The description of each status LED is provided in Table 5.3.

Table 5.3. Status LED Description

Sl. No	Name	Description
1	125 MHz Clock present	Green – Blinking when clock is present
2	PCIe Link Up	Green – Lights up if PCIe link up is successful
		Red – Lights up if PCIe link up is not successful
3	DONE	Green – Lights if configuration is successful

For the Certus-NX board, the programming done LED lights up in green if the configuration is successful. The LED is located at D29 on the board, as indicated in Figure 5.19.

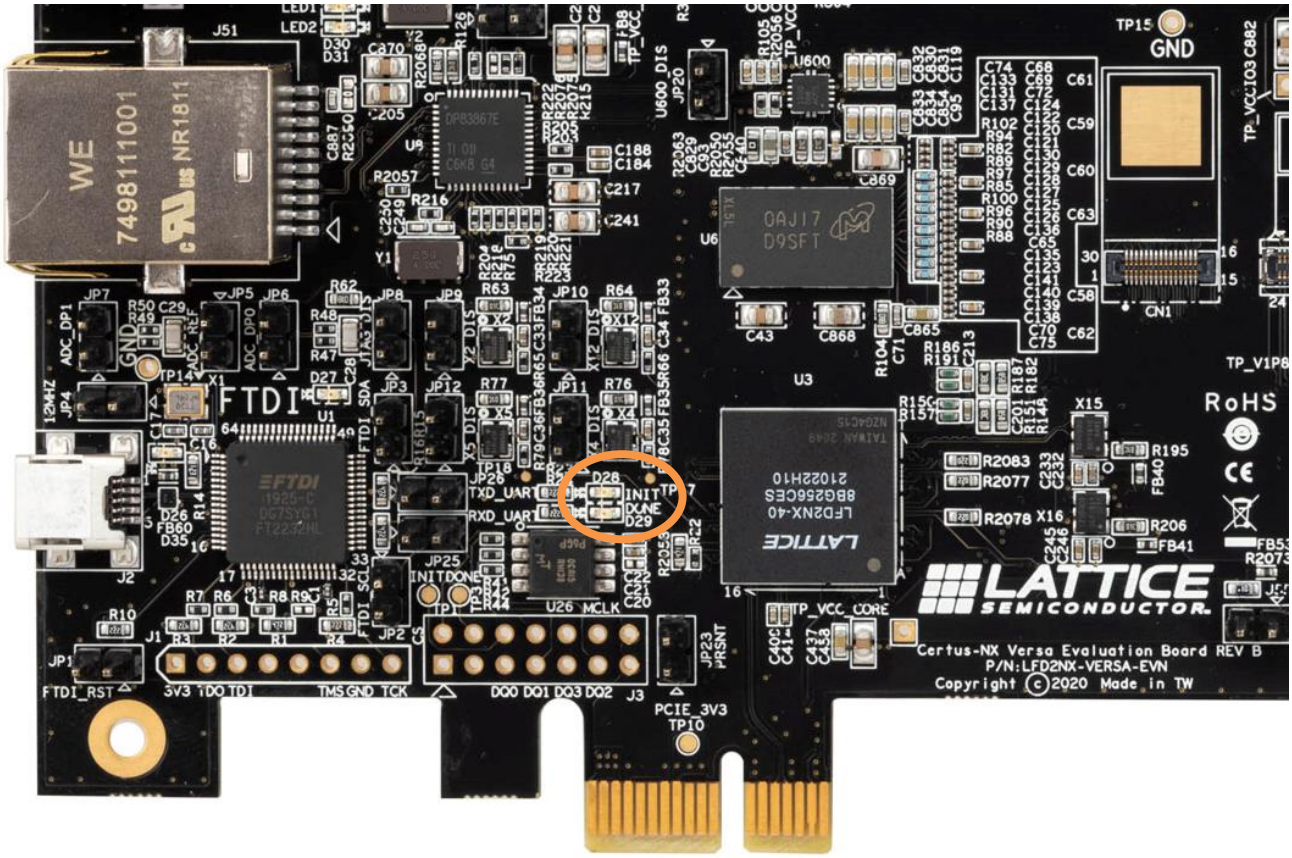


Figure 5.19. Certus-NX Programming Done LED

For the CertusPro-NX board, the programming done LED lights up in green if configuration is successful. The LED is located as shown in Figure 5.20.

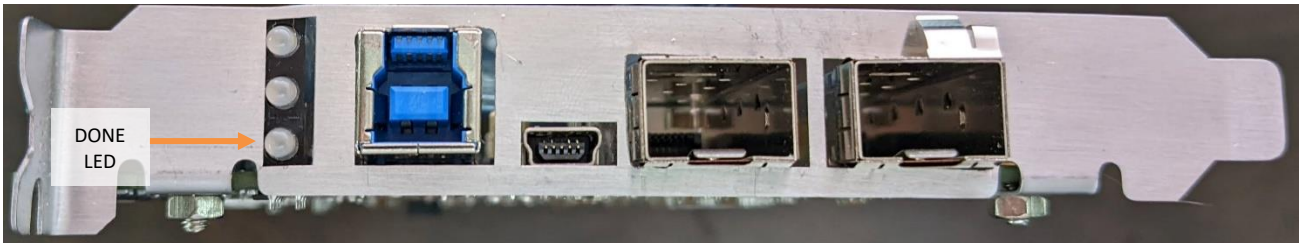


Figure 5.20. CertusPro-NX Programming Done LED

For MachXO5-NX, the programming done LED D10 lights up in green if configuration is successful. The LED is located as shown in Figure 5.21.

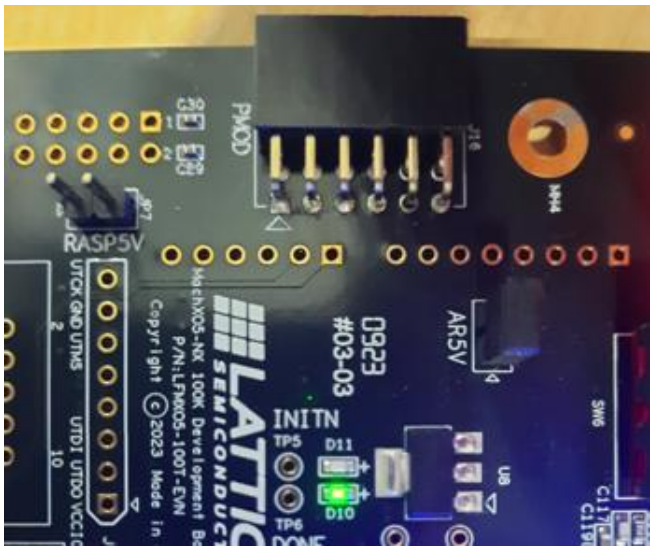


Figure 5.21. MachXO5-NX Programming Done LED

5.2. Software Setup

This section provides the procedure for installing software onto the host machine.

5.2.1. Software Setup and Installation for Windows

Before installing the driver, the Driver Signature Enforcement feature should be disabled.

5.2.1.1. Disabling Driver Signature Enforcement Permanently

To permanently disable Driver Signature Enforcement:

1. Start the Command Prompt as administrator.
2. Enter the following lines and press Enter.

```
bcdedit.exe -set loadoptions DISABLE_INTEGRITY_CHECKS
```

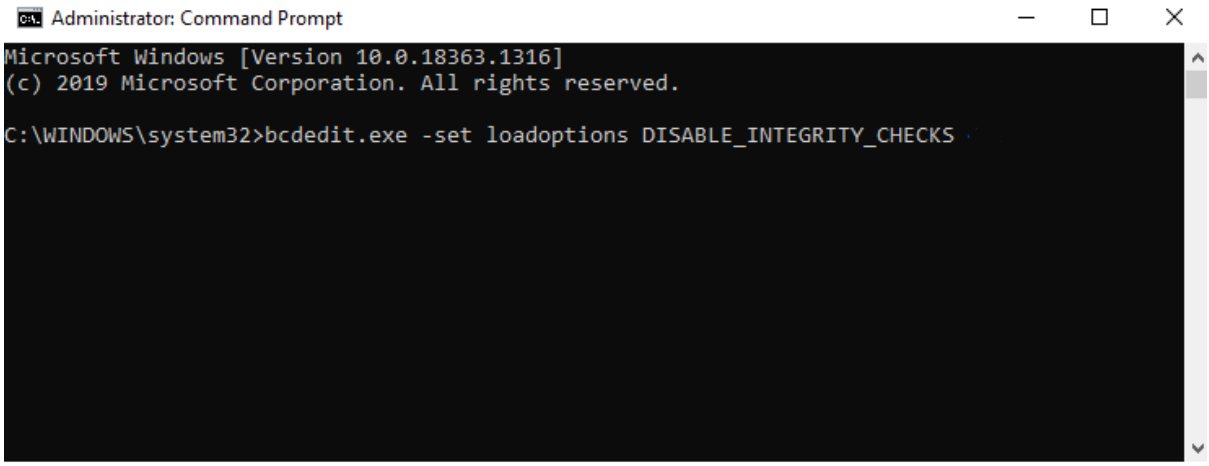


Figure 5.22. Running Disable Integrity Checks Command

```
bcdedit.exe -set TESTSIGNING ON
```

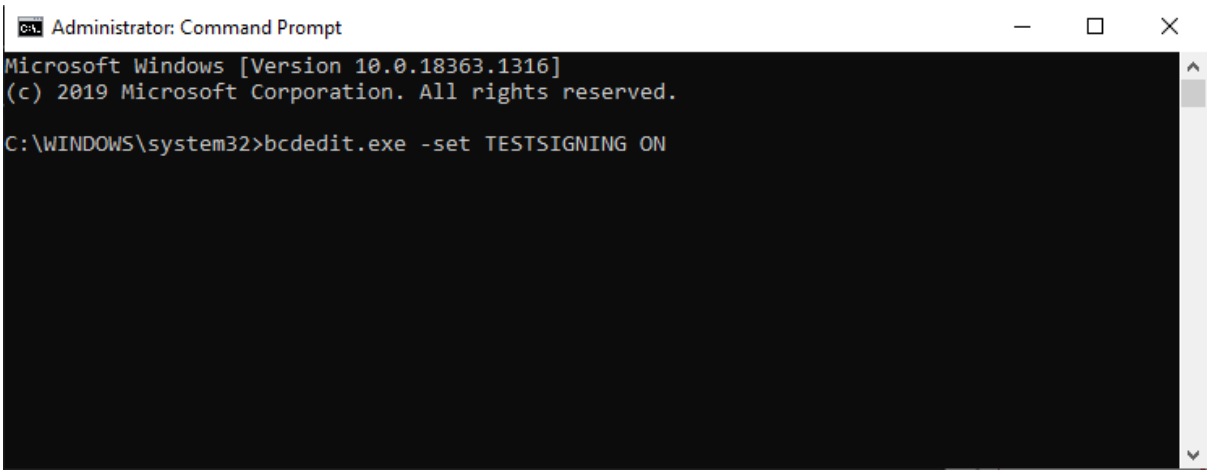


Figure 5.23. Running Test Sign On Command

3. Close the Command Prompt and restart the PC.

5.2.1.2. Disabling Driver Signature Enforcement Temporarily

Note: If Driver Signature Enforcement is already disabled, skip this section and proceed to the Driver Installation section.

To disable Driver Signature Enforcement temporarily on Windows 10:

1. Press the Windows key and click the power button.
2. Press and hold the Shift key and click **Restart**.
3. When the **Choose an option** screen appears as shown in Figure 5.24, release the Shift key.
4. Click **Troubleshoot**.

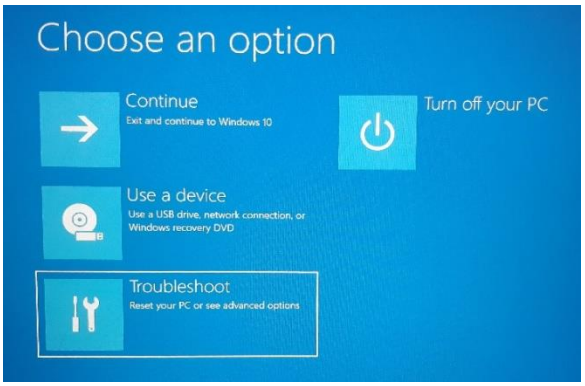


Figure 5.24. Troubleshoot Option

5. Select **Advanced options** and press Enter.

Figure 5.25. Advanced Options

6. Select **Startup Settings** and press Enter.

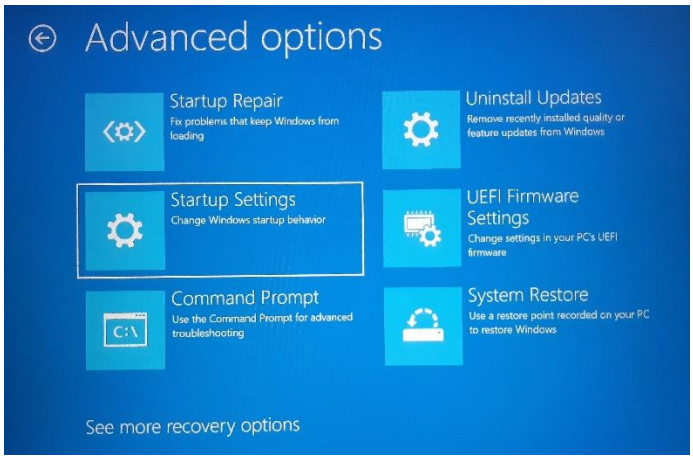


Figure 5.26. Select Startup Settings

7. Click **Restart**.

Figure 5.27. Restarting Windows

8. After restarting, select Option 7 to disable driver signature verification.

5.2.1.3. Driver Installation

There are two ways to install the device driver:

- Install through a user interface installer (.exe) that is included in the demo package
- Install manually with the Device Manager

Installing Multifunction Demo Device Driver with the User Interface Installer

The Multifunction Demo device driver can be installed during the installation of the user interface as described in the following section.

The Installer provides a standard packaging format for applications and a standard method for customizing the applications. The installer helps to install the Multifunction Demo application in the system.

The Framework supported version is Windows 10. Use WDF 1.25 or earlier.

To install the Multifunction Demo device driver through the user interface:

1. In the *Demonstration\Windows10\Application* folder, double click *setup.exe*.
2. The welcome page appears. Click **Next**.

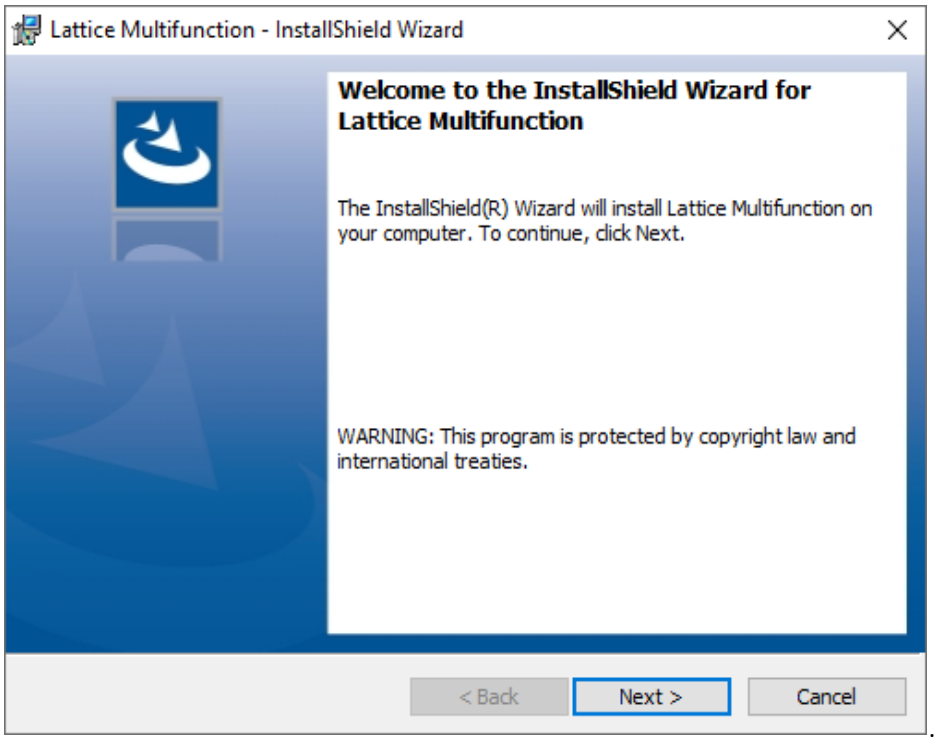


Figure 5.28. Windows Installer: Welcome Page

3. Provide the location where user want to install the application. Click **Next**.

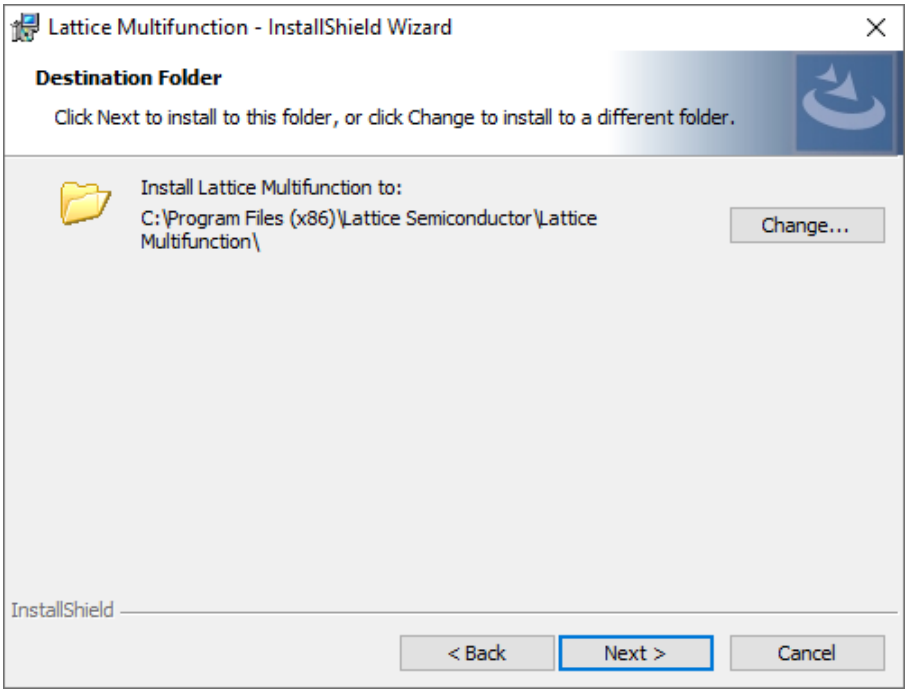


Figure 5.29. Windows Installer: Destination Folder Page

4. The installation summary page is shown. Click **Install.**

   Note: Administrative access is required to run this command.

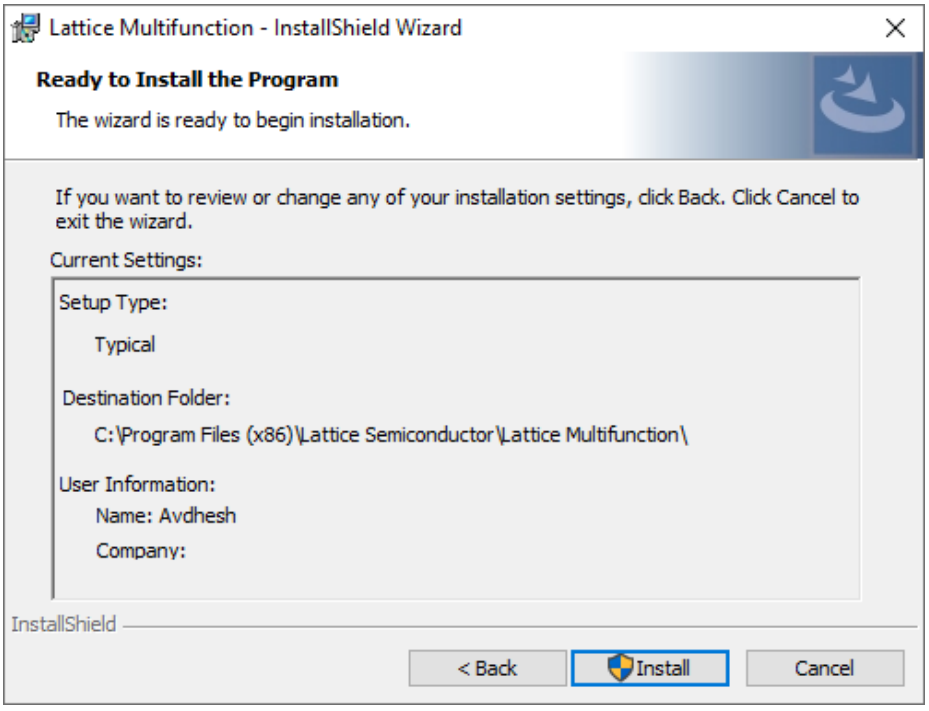


Figure 5.30. Windows Installer: Summary Page

5. The installation of the Multifunction Demo application starts. When the installation of the software is complete, the drivers are installed.

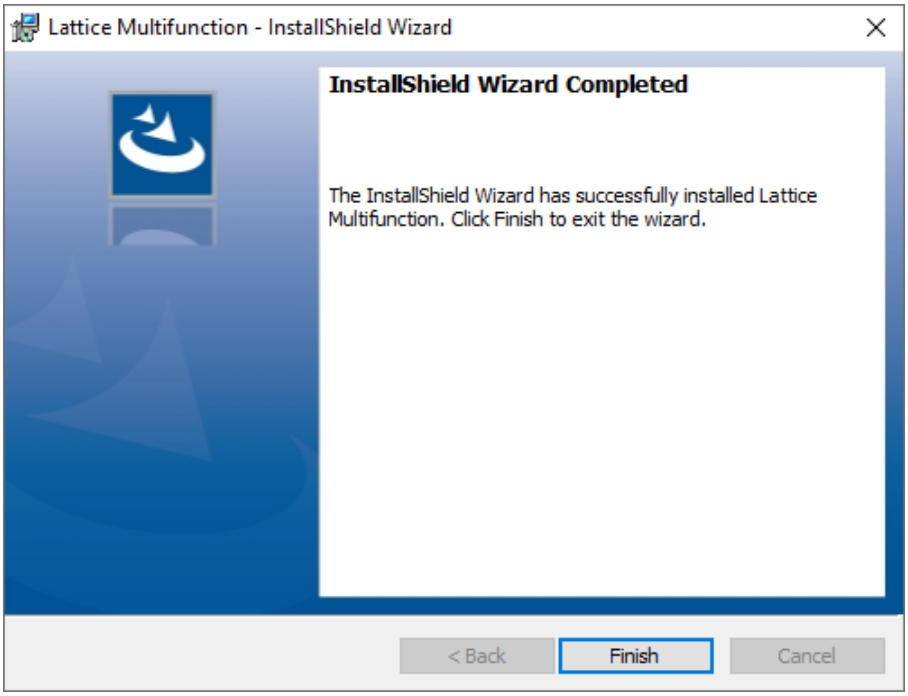


Figure 5.31. Windows Installer: Application Installed

6. A message box appears. Click **Yes**.

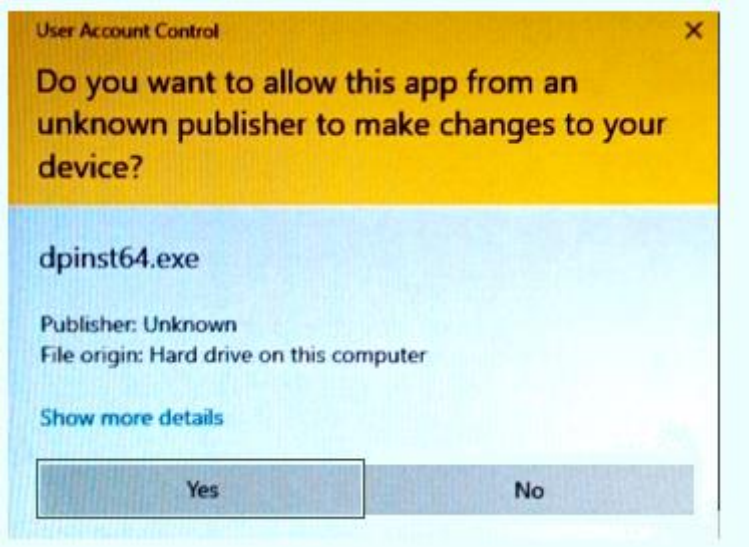


Figure 5.32. Device Configuration Prompt

7. The device driver installation wizard opens. Click **Next**.

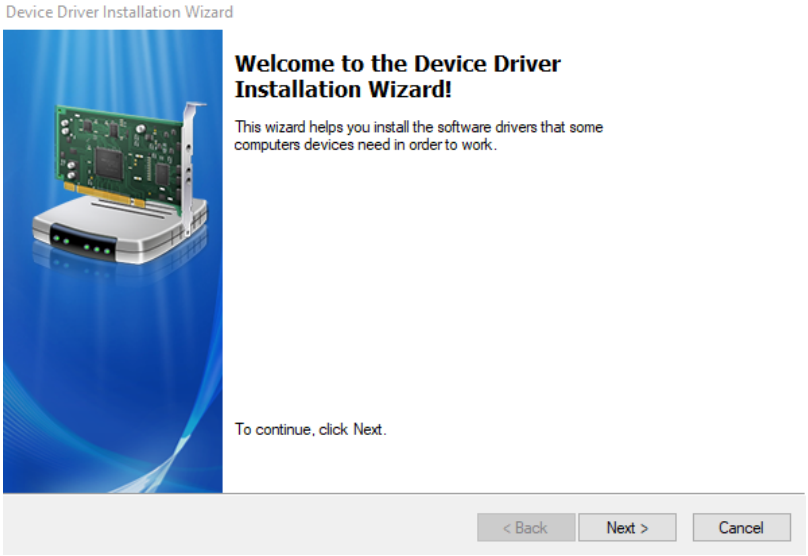


Figure 5.33. Device Driver Installation Wizard

8. If user receive a Windows Security prompt, select **Install this driver software anyway** as shown in Figure 5.34.

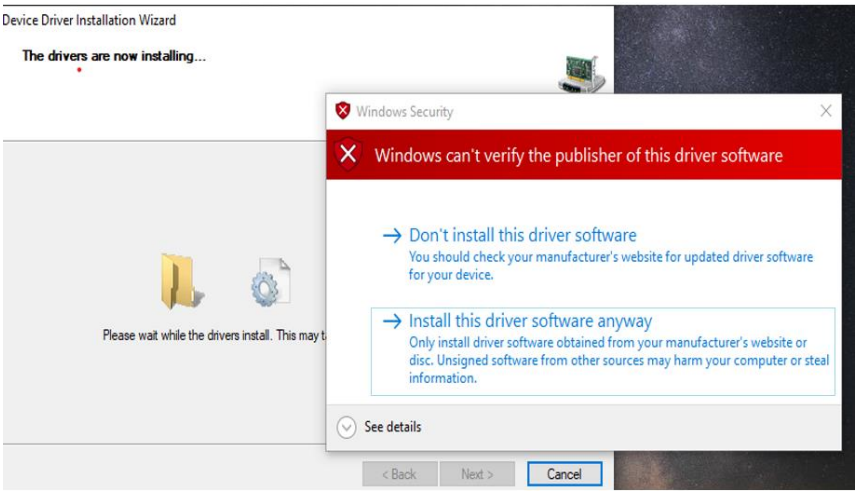


Figure 5.34. Windows Security in Driver Installation

9. If the driver is installed successfully, a message is displayed as shown in Figure 5.35. Click **Finish**.

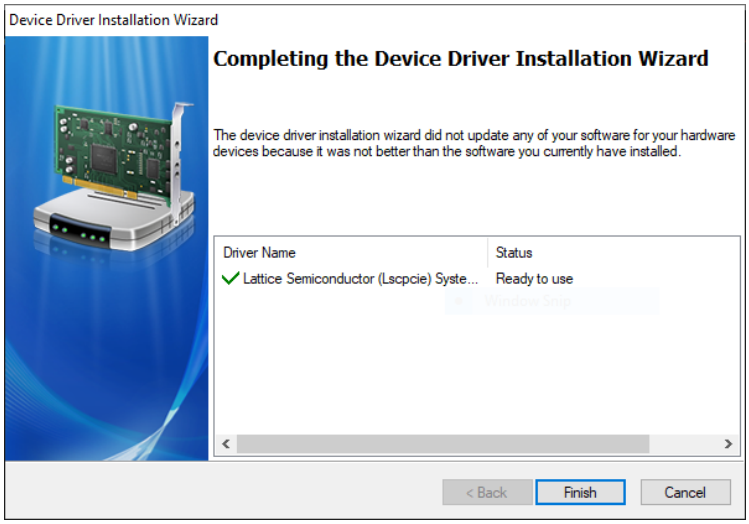


Figure 5.35. Device Driver Installation Completed

Installing the Multifunction Demo Device Driver Manually

The drivers for the Multifunction Demo can be found in the *Demonstration\Windows10\Driver* folder.

To install the Multifunction Demo device driver manually:

1. Open **Device Manager**, which shows the connected PCIe devices, as shown in Figure 5.36.

   Note: If the **Device Manager** does not show the connected PCIe device, see the Troubleshooting section.

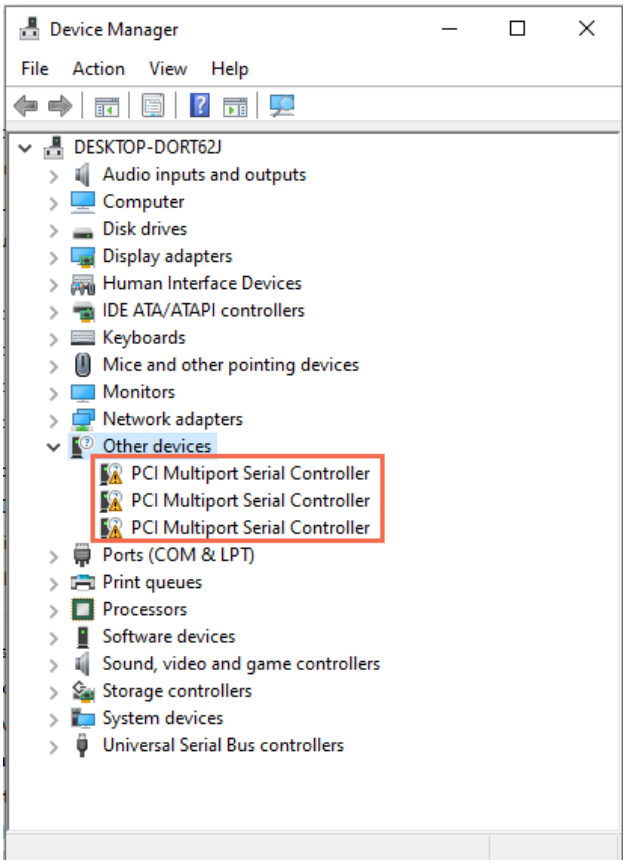


Figure 5.36. Device Manager

2. Right-click on each device and select **Properties** as shown in Figure 5.37.

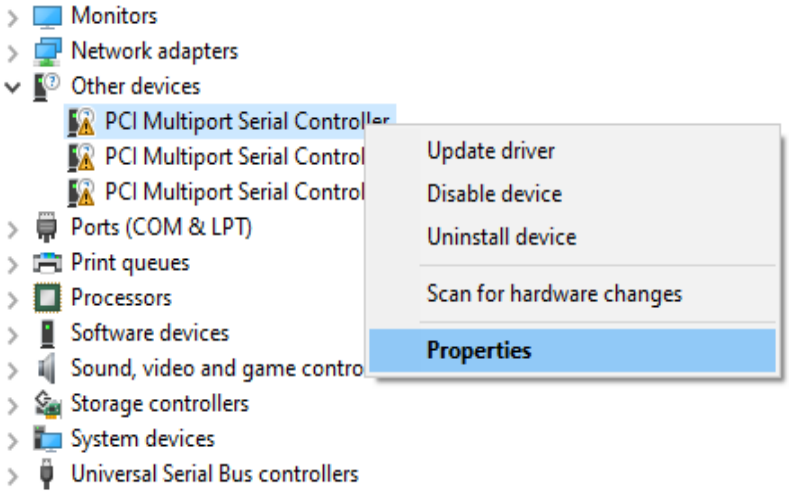


Figure 5.37. Showing Device Properties

3. The information in **Hardware IDs** is needed to install the correct driver for the corresponding device. The Hardware IDs of the MDIO, I^2C, and GPIO devices are shown below:

Table 5.4. Hardware IDs

Sl. No	CrossLink-NX	Certus-NX	CertusPro-NX	MachXO5-NX
MDIO	PCI\VEN_1204&DEV_9C22&SUBSYS_E00419AA	PCI\VEN_1204&DEV_9C32&SUBSYS_E00419AA	N/A	N/A
I²C	PCI\VEN_1204&DEV_9C1F&SUBSYS_E00419AA	PCI\VEN_1204&DEV_9C2F&SUBSYS_E00419AA	PCI\VEN_1204&DEV_9C40&SUBSYS_E00419AA	N/A
GPIO	PCI\VEN_1204&DEV_9C20&SUBSYS_E00419AA	PCI\VEN_1204&DEV_9C30&SUBSYS_E00419AA	PCI\VEN_1204&DEV_9C42&SUBSYS_E00419AA	PCI\VEN_1204&DEV_9C42&SUBSYS_E00419AA

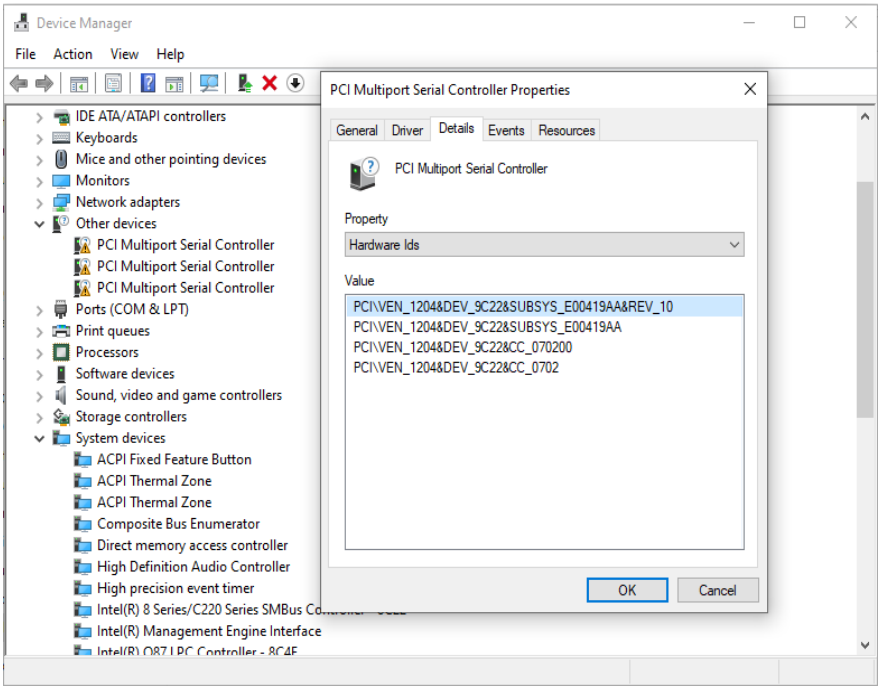


Figure 5.38. Hardware IDs of CrossLink-NX MDIO Device

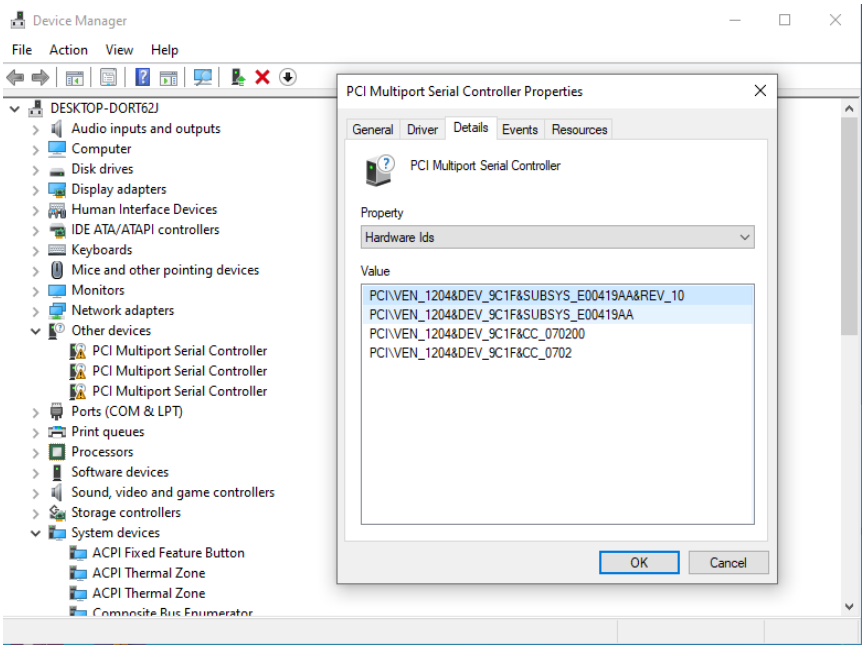


Figure 5.39. Hardware IDs of CrossLink-NX I²C Device

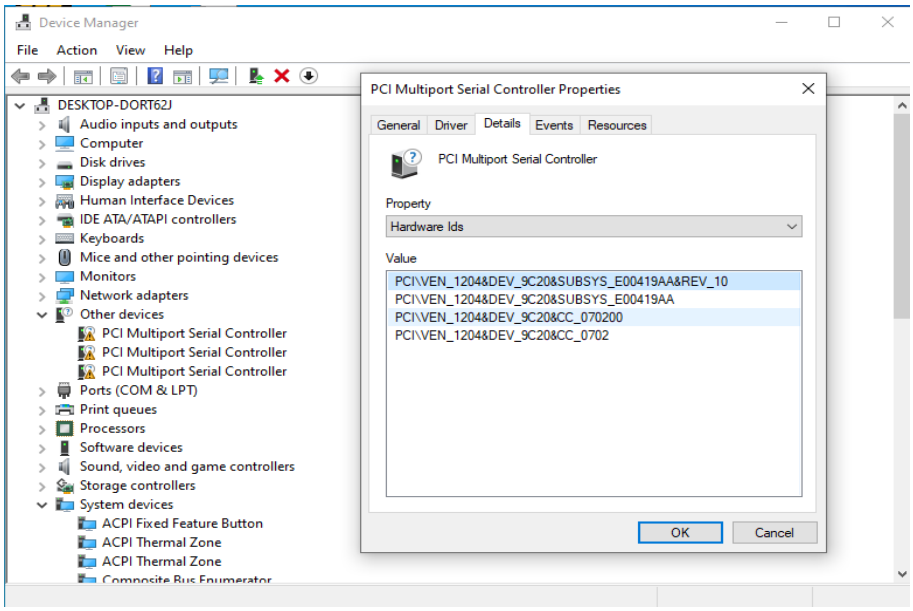


Figure 5.40. Hardware IDs of CrossLink-NX GPIO Device

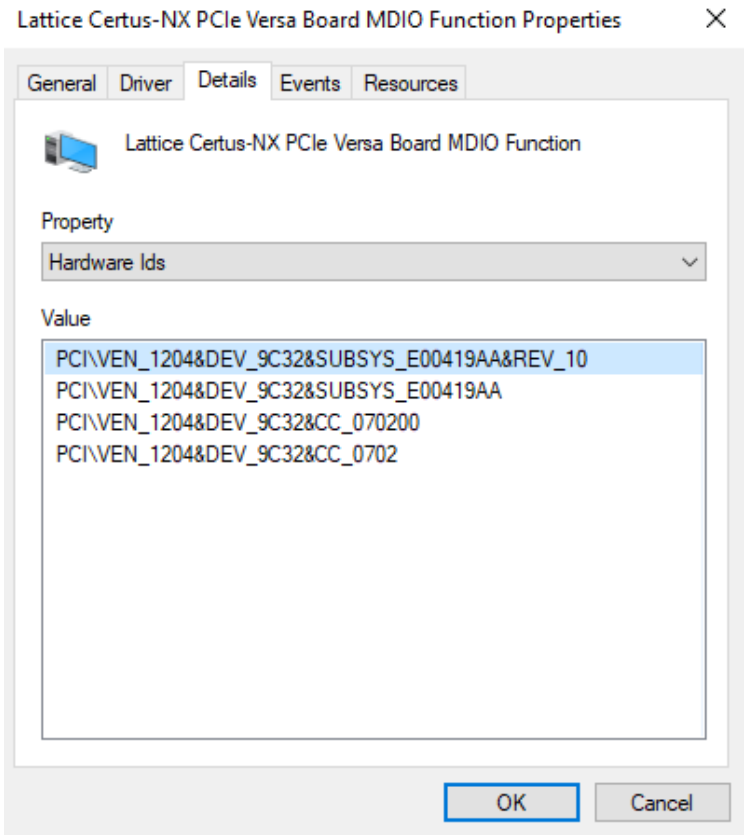


Figure 5.41. Hardware IDs of Certus-NX MDIO Device

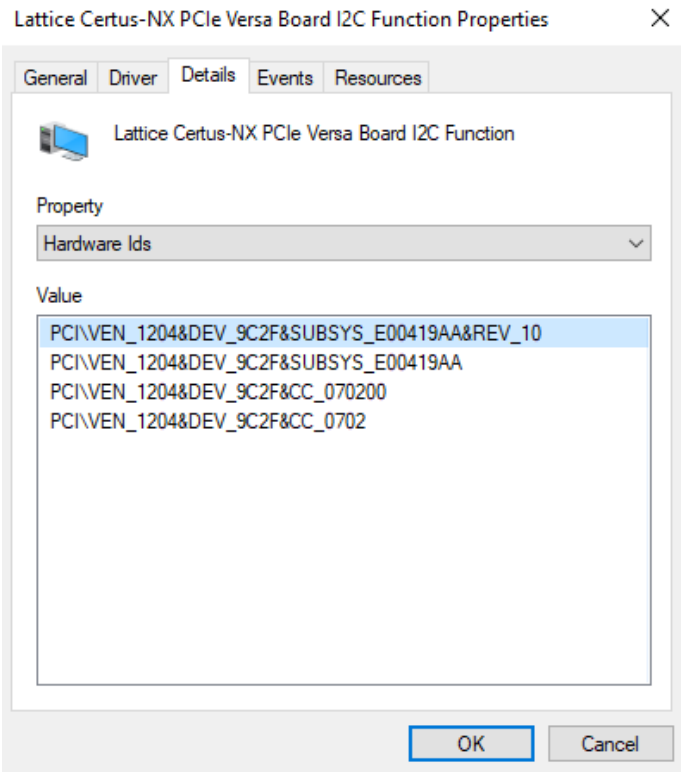


Figure 5.42. Hardware IDs of Certus-NX I²C Device

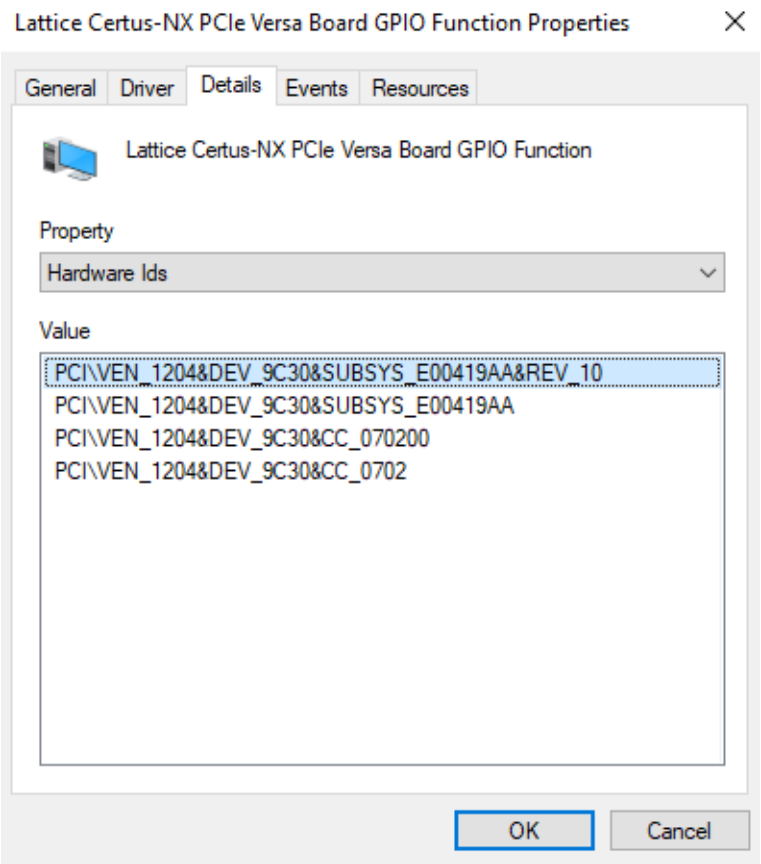


Figure 5.43. Hardware IDs of Certus-NX GPIO Device

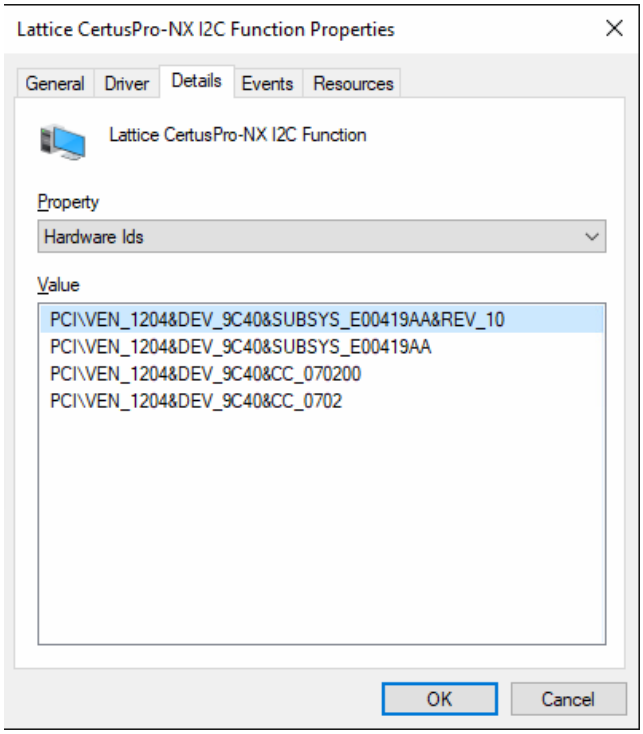


Figure 5.44. Hardware IDs of CertusPro-NX I²C Device

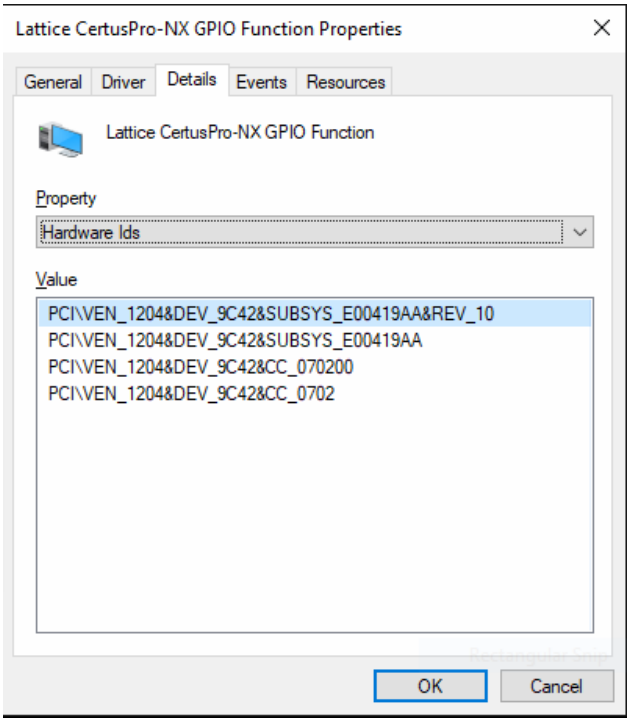


Figure 5.45. Hardware IDs of CertusPro-NX and MachX05-NX Devices

4. Right click on the device and select **Update driver**.

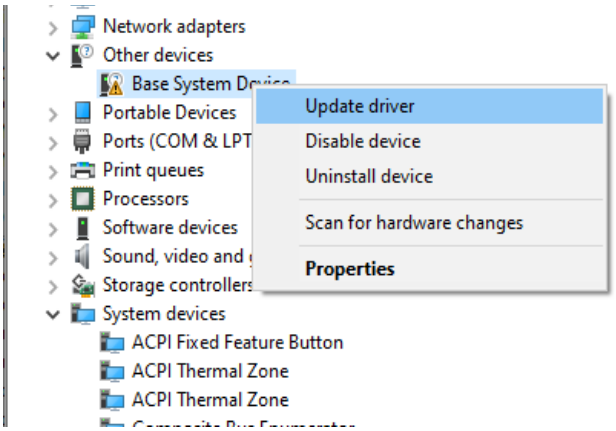


Figure 5.46. Update Driver Menu in Device Manager

5. Select the **Browse my computer for driver software** option as shown in Figure 5.47.

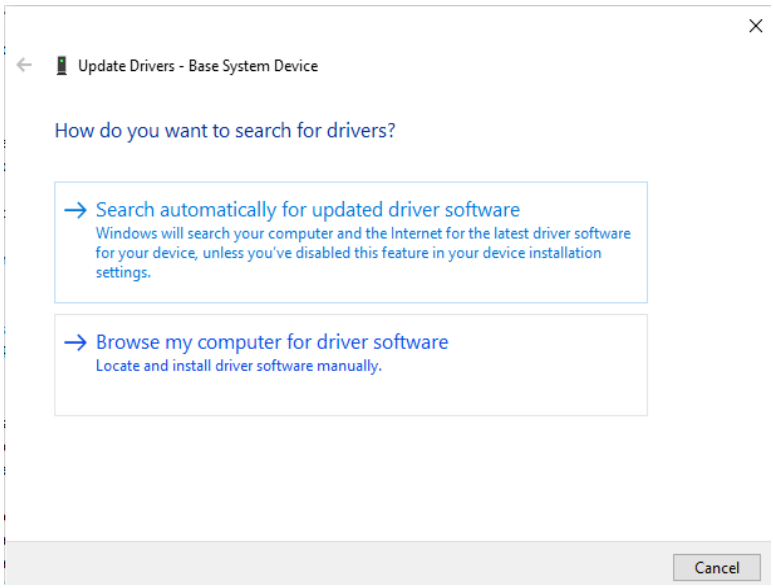


Figure 5.47. Update Driver Options

6. Browse for the MDIO device drivers as shown in Figure 5.48. For the CertusPro-NX board, browse for I^2C or GPIO driver.

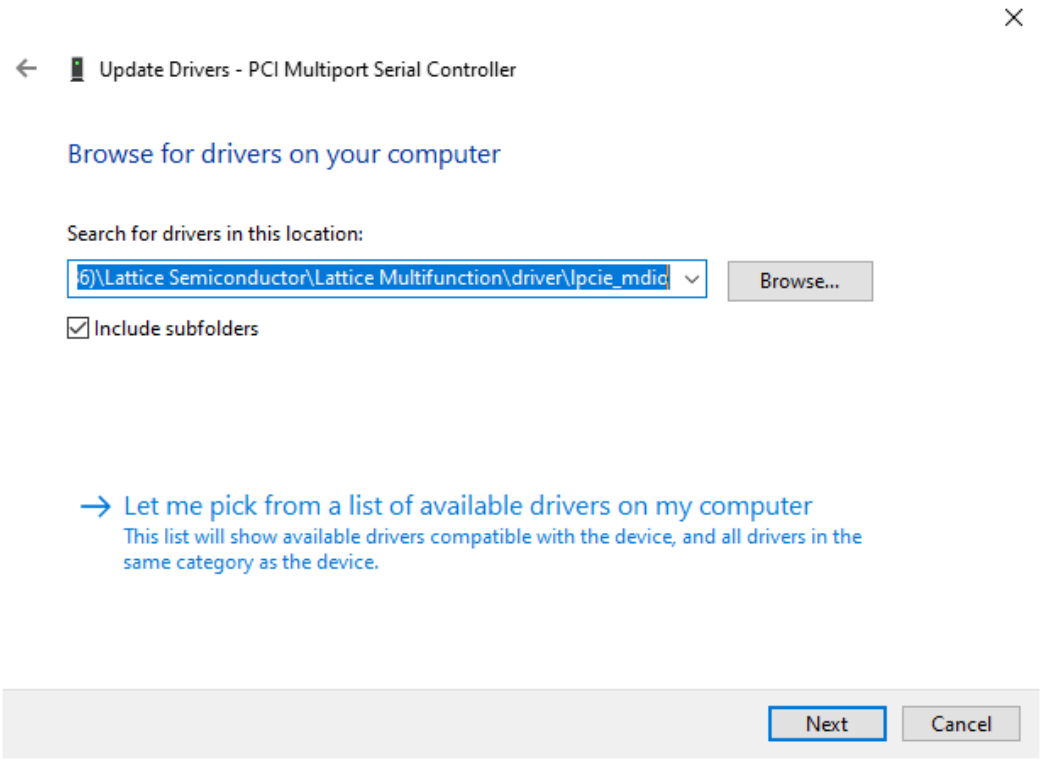


Figure 5.48. Browse the Driver for Device

7. If user receive a Windows Security prompt, select **Install this driver software anyway** as shown in Figure 5.49.

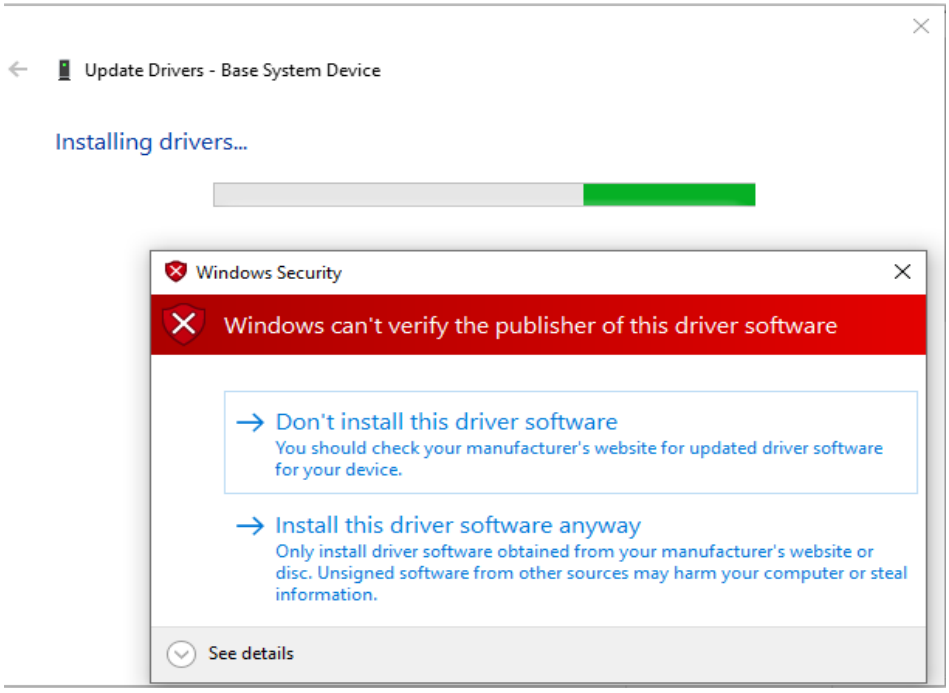


Figure 5.49. Windows Security in Device Manager

8. If the driver is installed successfully, a message is displayed as shown in Figure 5.50 for the CrossLink-NX board, Figure 5.51 for the Certus-NX board, and Figure 5.52 for the CertusPro-NX board.

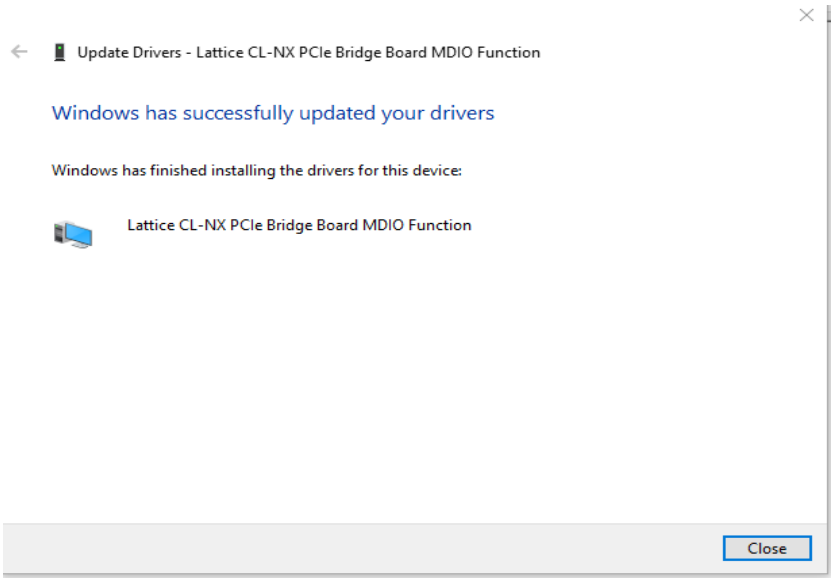


Figure 5.50. MDIO Driver Installation Status Message for the CrossLink-NX Board

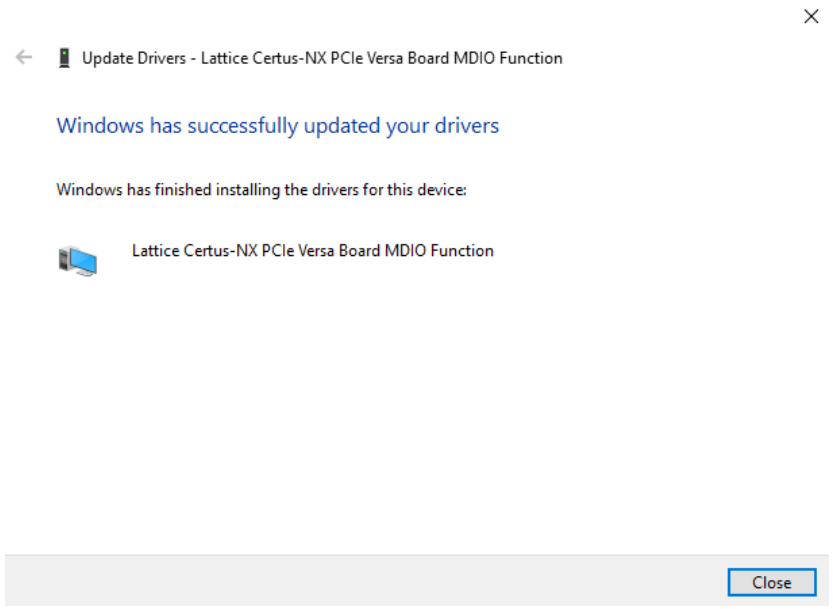


Figure 5.51. MDIO Driver Installation Status Message for the Certus-NX Board

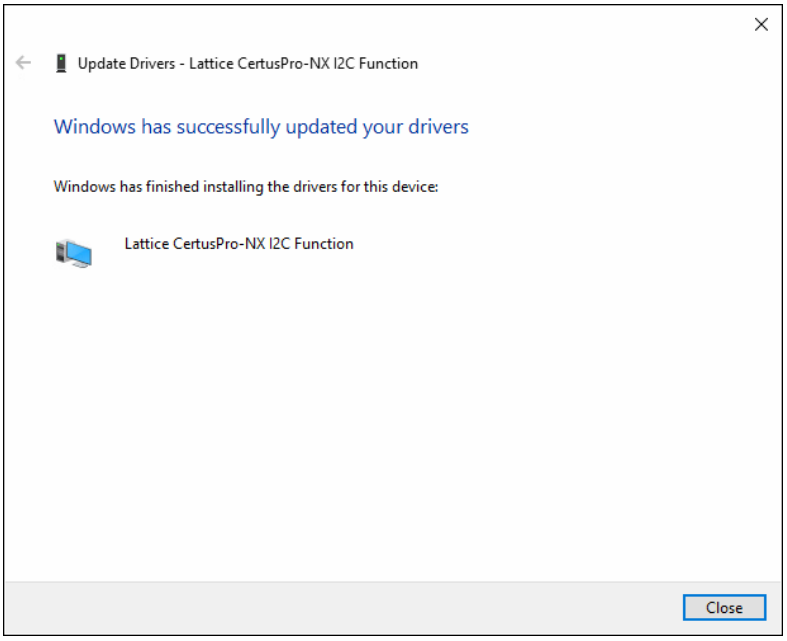


Figure 5.52. I²C Driver Installation Status Message for the CertusPro-NX Board

9. After the driver is installed, the device driver name is displayed in **Device Manager**, as shown in Figure 5.53.

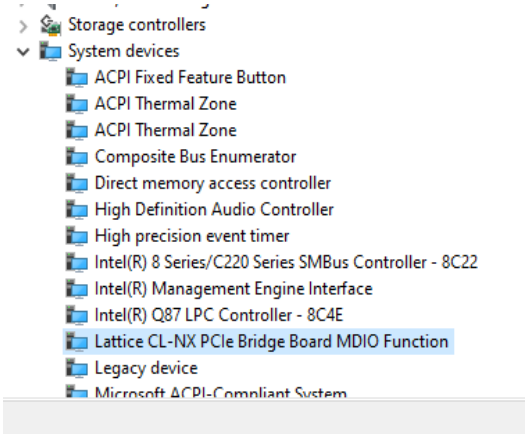


Figure 5.53. Multifunction Demo Device Name Displayed in Device Manager

10. Follow steps 3 through 8 for the I²C and GPIO devices. After successful installation, all the three devices should be displayed in Device Manager as shown in Figure 5.54 for the CrossLink-NX board, Figure 5.55 for the Certus-NX board, and Figure 5.56 for the CertusPro-NX board.

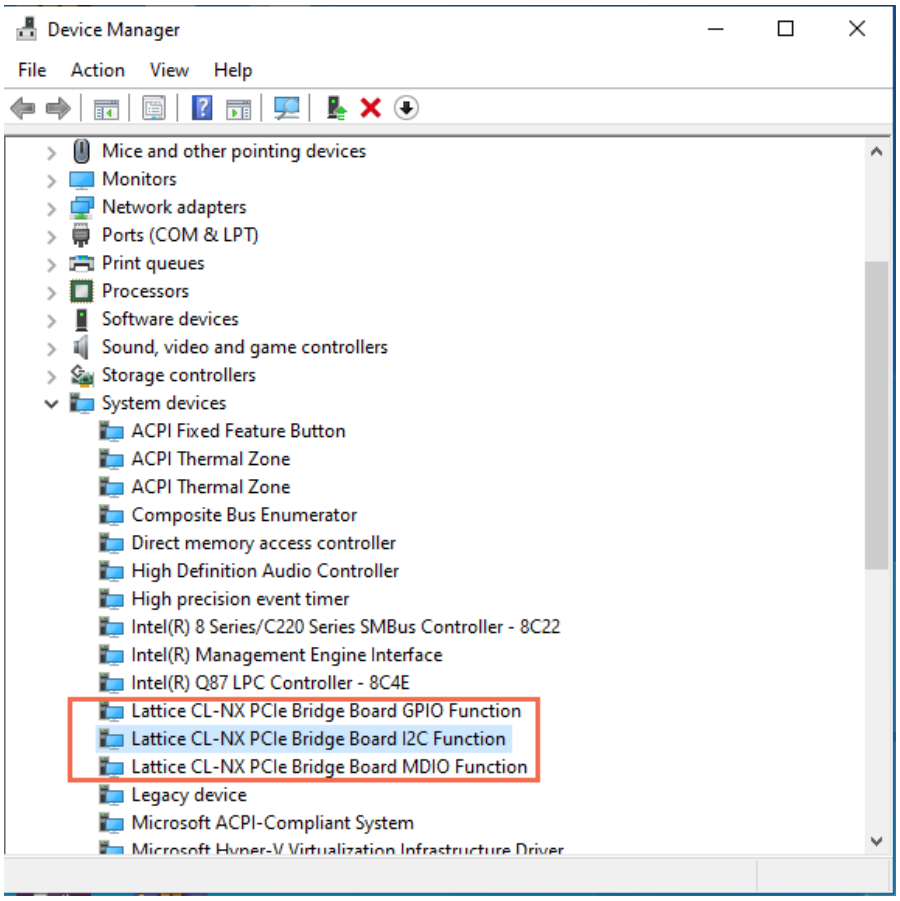


Figure 5.54. MDIO, I²C, and GPIO Device Drivers for the CrossLink-NX Board in Device Manager

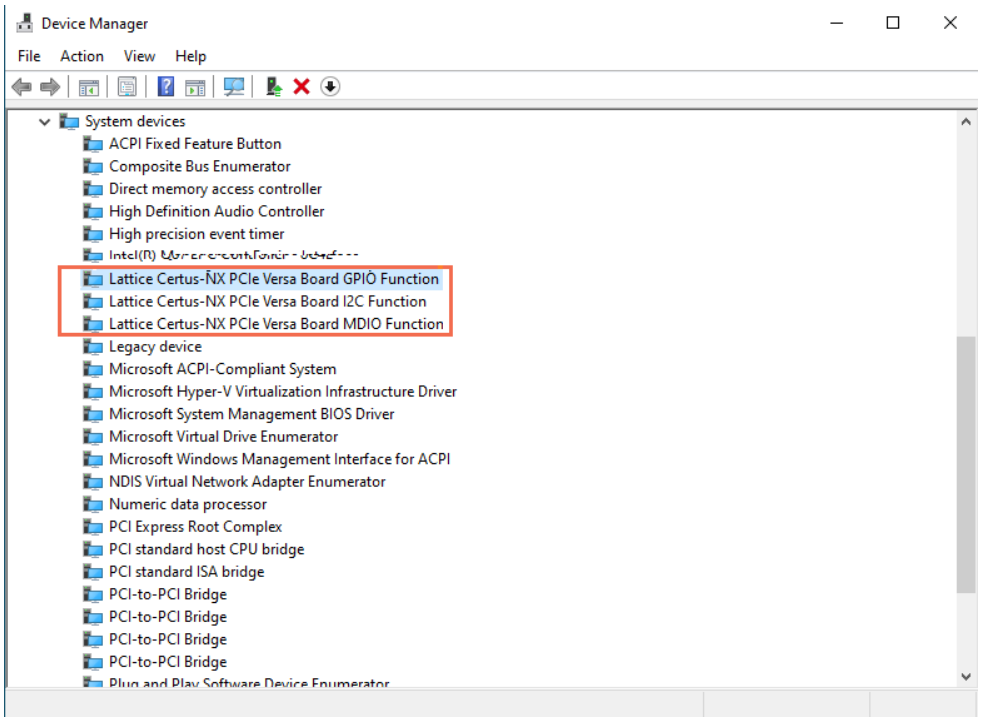


Figure 5.55. MDIO, I²C, and GPIO Device Drivers for the Certus-NX Board in Device Manager

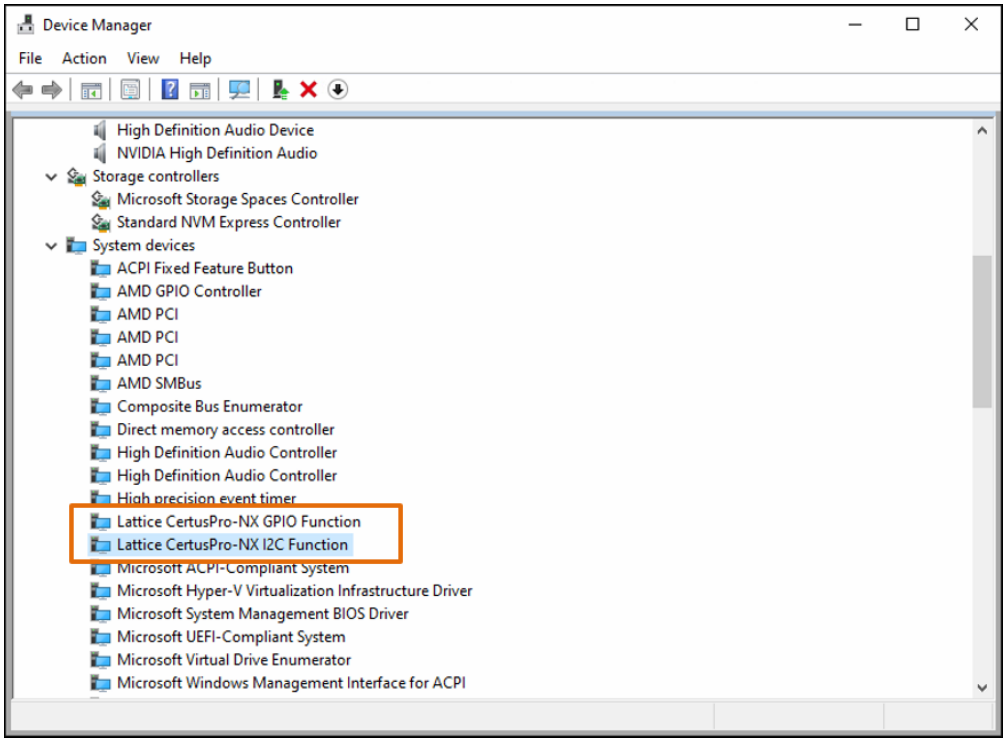


Figure 5.56. I²C, and GPIO Device Drivers for the CertusPro-NX Board in Device Manager

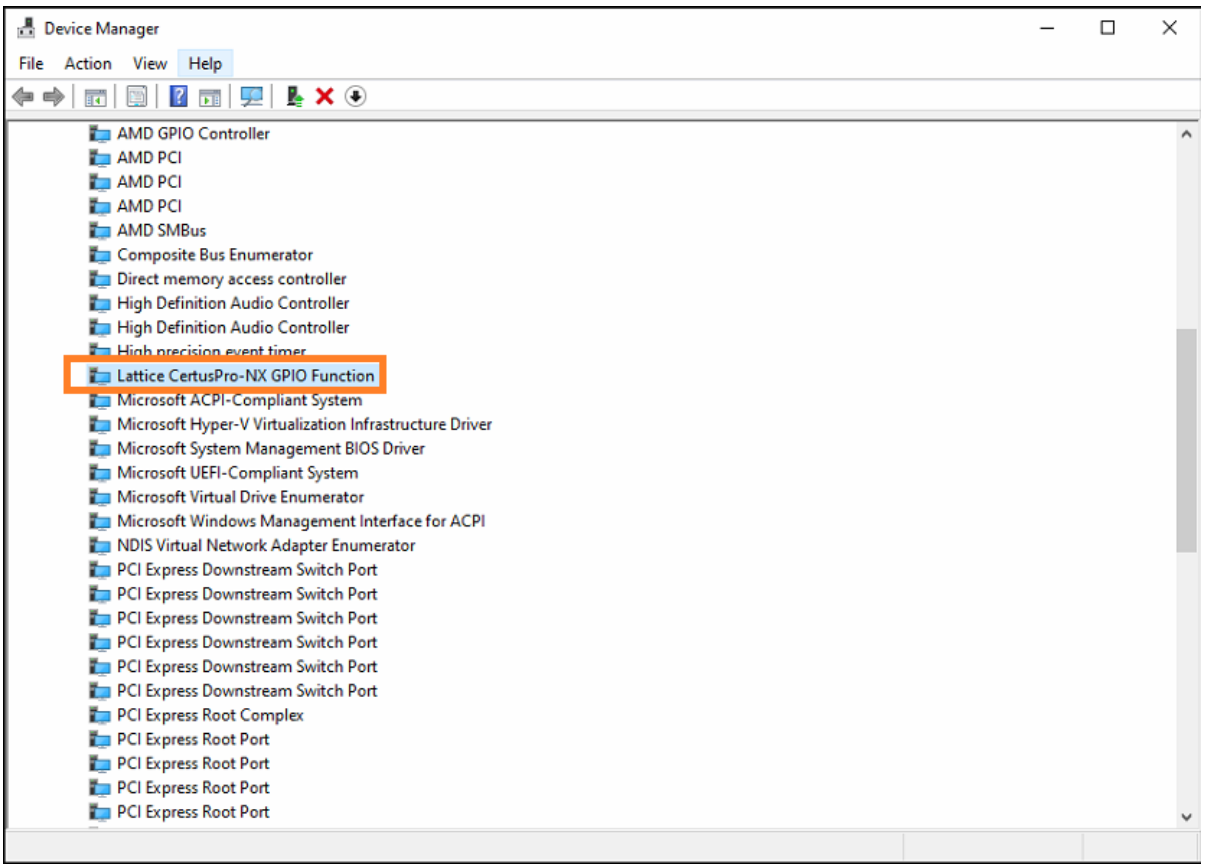


Figure 5.57. GPIO Device Drivers for MachX05-NX in Device Manager

5.2.1.4. Uninstall Device Driver Manually

Follow the steps below to uninstall the device driver.

1. Open **Device Manager** window. Look for the device user want to uninstall.
2. Right click on the device driver and select Uninstall device as shown in Figure 5.58.

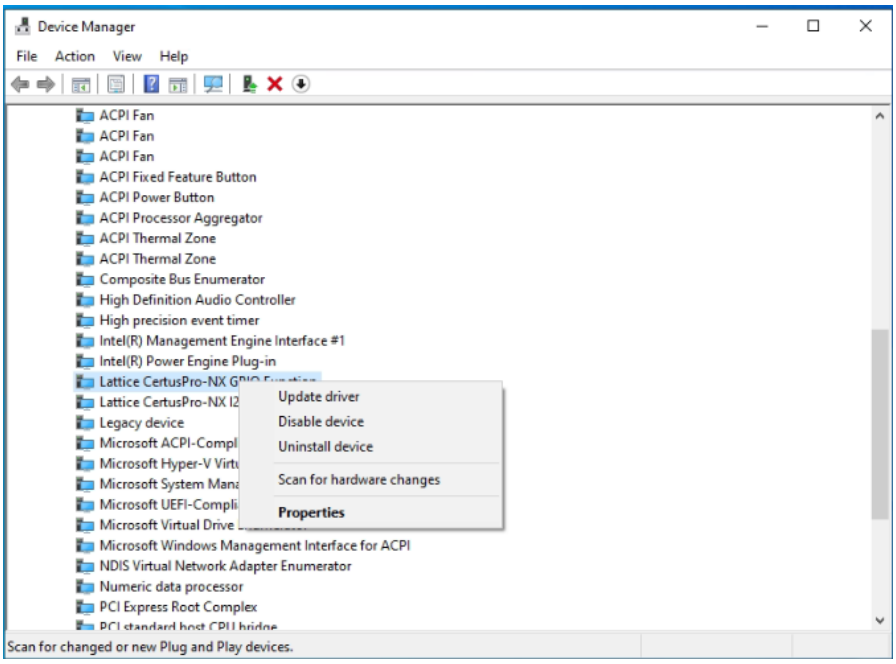


Figure 5.58: Select "Uninstall device" for the device user want to uninstall

3. Check the option **Delete the driver software for this device** and click on **Uninstall** as shown in Figure 5.59.

Figure 5.59: Select the check box and click on Uninstall button.

4. In the **Device Manager**, Click **Action** menu and select **Scan for hardware changes**. Make sure that the uninstalled device is not listed in the **Device Manager**.

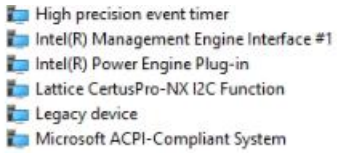


Figure 5.60: Uninstalled device

5.2.2. Software Setup for Linux

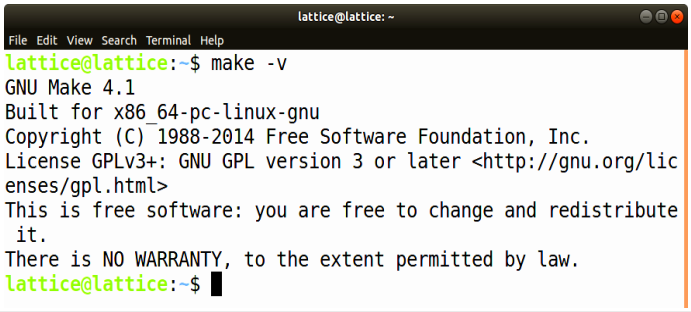
5.2.2.1. Supported Operating System

- Distribution: Ubuntu
- Description: Ubuntu 18.04.6 LTS
- Release: 18.04.6
- OS Type: 64 bit.
- Codename: bionic

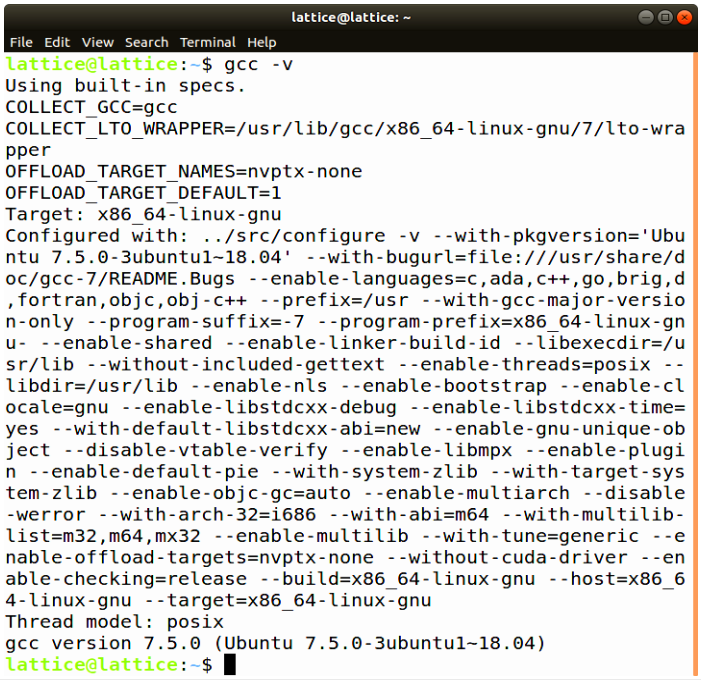
5.2.2.2. Required Packages

To check whether the required packages are installed, run the following commands on a terminal as shown below.

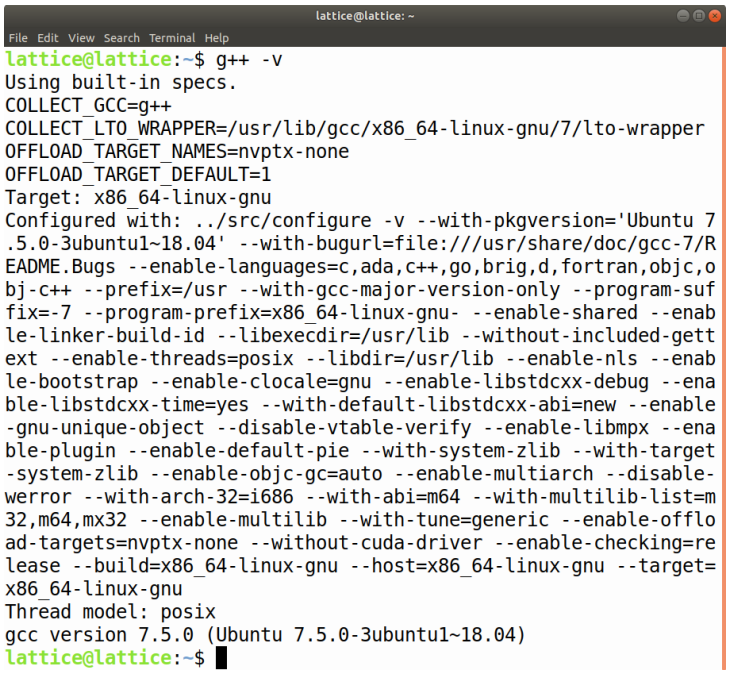
```
make -v
```




```
gcc -v
```




```
g++ -v
```



5.2.2.3. Packages Installation Steps

Run the following commands on a terminal to install the required packages.

```
sudo apt update
sudo apt install build-essential
sudo apt install flex
```

5.2.2.4. Automatic Setup and Installation

To setup the demo in automatic mode:

1. Go to the *Demonstration/Linux* directory.
2. Change the permission of the *install.sh* file by running the chmod command.
   ```
   sudo chmod 777 install.sh
   ```
3. Run the command below, which builds the driver and API library, installs the driver, and launches the QT use interface application.
   ```
   sudo ./install.sh
   ```
4. To uninstall the driver, go to the *Demonstration/Linux* directory and run the command below.
   ```
   sudo ./uninstall.sh
   ```

5.2.2.5. Manual Setup and Installation

Note: User can skip this section if user have installed the driver successfully in the previous section.

Before installing the driver, the driver must be built.

To build the driver:

1. Go to the *Demonstration/Linux* directory.
2. Run the following command on terminal.
   ```
   sudo chmod 777 -R Source_Code
   ```
3. Go to the *Source_Code* directory.
4. Run the following commands.
   ```
   sudo make clean
   sudo make
   ```
   This builds the driver and the API library.
5. Make sure that the driver is not installed.
6. To remove an existing driver, run one or all of the following command in a terminal window:
   ```
   sudo rmmod gpio_main.ko
   sudo rmmod i2c_main.ko
   sudo rmmod mdio_main.ko
   ```

To install the drivers:

1. Go to *drv_src/gpio_drv/* directory.
2. Run the command below.
   ```
   sudo insmod gpio_main.ko
   ```
   a. Repeat the same procedure for the i2c_drv and mdio_drv directories.

      Note: mdio_drv is not applicable to the CertusPro-NX board.
3. To launch the user interface application, go to the *app_src/gui/deploy/* directory and run the command below.
   ```
   sudo ./MFDemo.sh
   ```

6. Application Overview

The PCI Express Multifunction application demonstrates the multifunction capabilities of the FPGA. The application allows user to access the GPIO, I²C, and MDIO registers on the Crosslink-NX and Certus-NX boards. The application provides real time interaction with the FPGA hardware to demonstrate a functional PCI Express communications path between the software and the FPGA IP.

6.1. Running the PCI Express Demo Application

On Linux, user can launch the demo application by running the *./MFDemo.sh script* located in the *Demonstration/Linux/Source_Code/app_src/gui/deploy* directory. On Windows, user can launch the application by clicking the **Lattice Multifunction** shortcut on the desktop.

The graphical user interface opens the PCI Express Test Application software with the Device Info tab selected, as shown in Figure 6.1 for the Crosslink-NX device (similar to the Certus-NX device), Figure 6.2 for the CertusPro-NX, and Figure 6.3 for the MachXO5-NX. Note that some of the information shown may vary between different FPGA devices (such as the Device ID).

Figure 6.1. PCIe Test Application Device Info Tab for Crosslink-NX

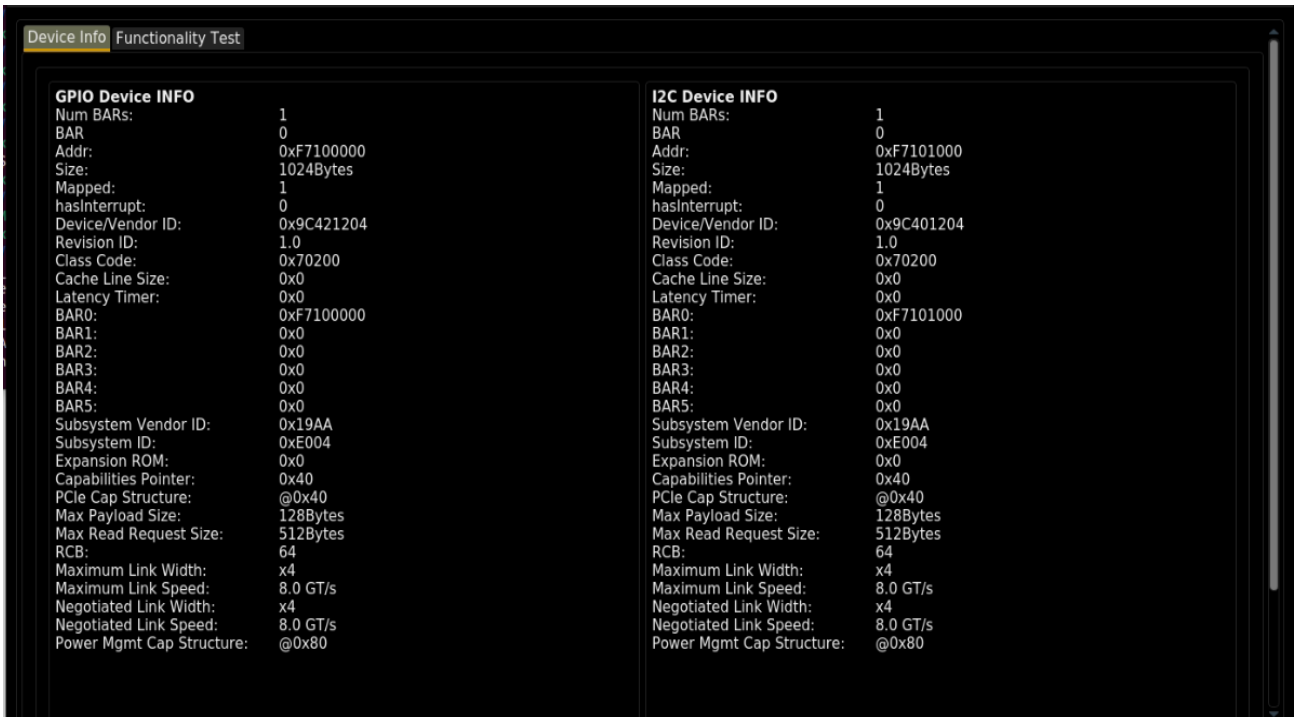


Figure 6.2. PCIe Test Application Device Info Tab for CertusPro-NX

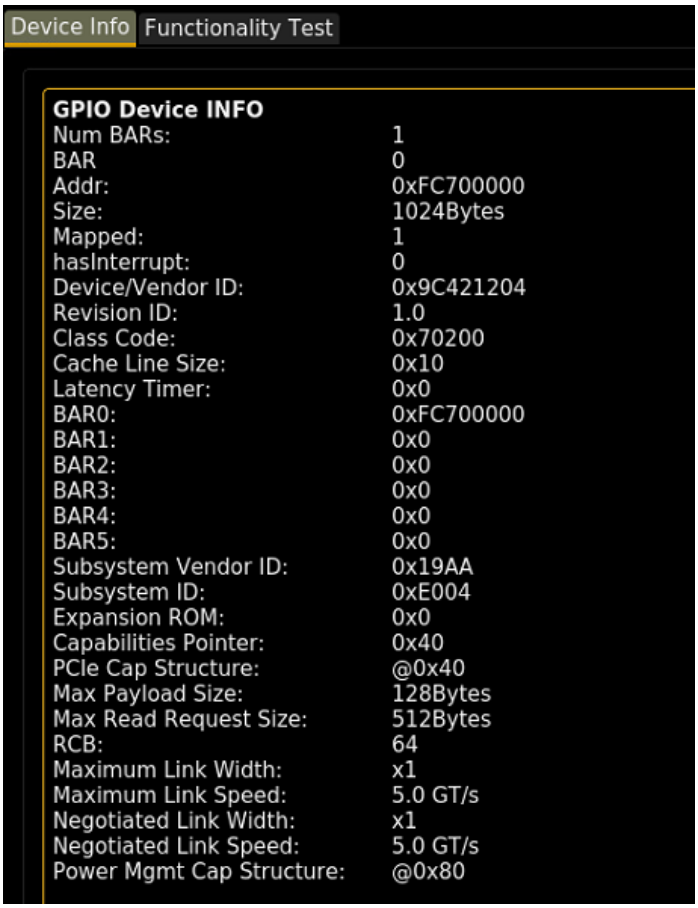


Figure 6.3. PCIe Test Application Device Info Tab for MachXO5-NX

6.2. Using the PCI Express Demo Application User Interface

The PCI Express Device Info page displays information about the device driver and PCI configuration registers. The data displayed is read-only and cannot be edited. If there is a problem loading the driver or accessing the device, an error message appears on this page.

6.2.1. Functionality Test Tab

Figure 6.4 shows the Functionality Test tab for the Crosslink-NX and Certus-NX devices. Figure 6.5 for the CertusPro-NX device, and Figure 6.6 for MachXO5-NX.

User can read from or write to the device using controls present on this tab.

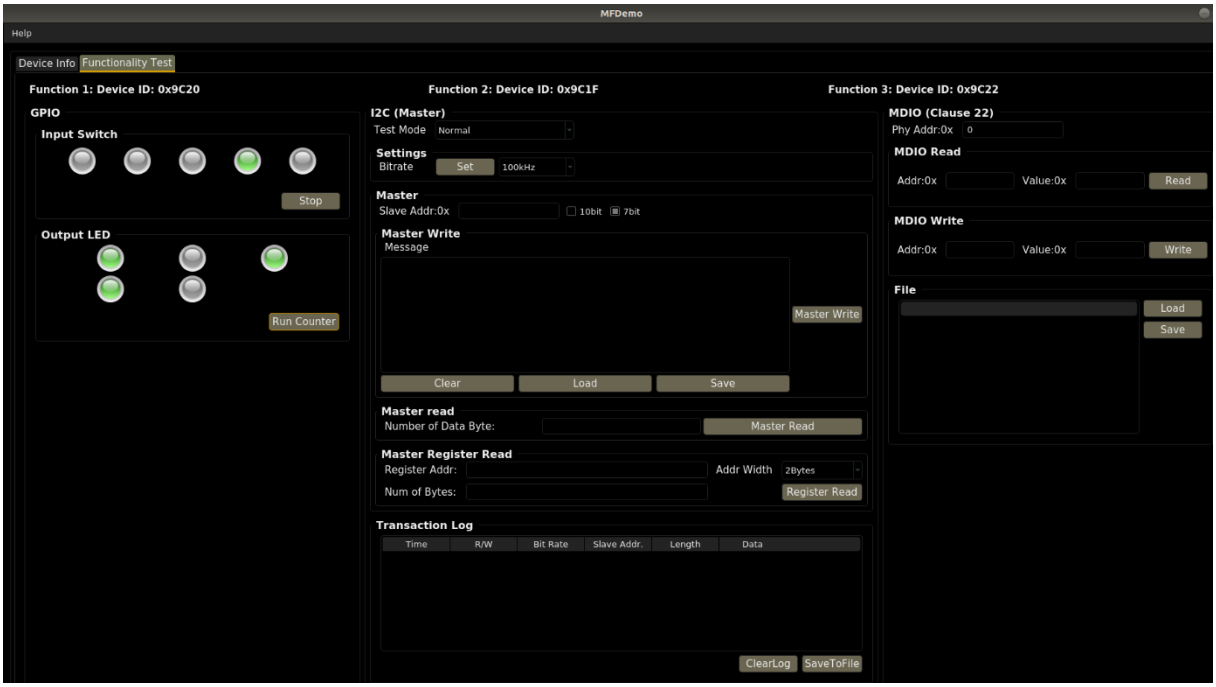


Figure 6.4. Functionality Test Tab for Crosslink-NX/Certus-NX

Figure 6.5. Functionality Test Tab for CertusPro-NX

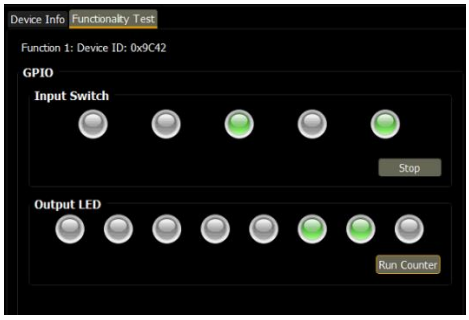


Figure 6.6. Functionality Test Tab for MachXO5-NX

6.2.2. GPIO

This section describes how to interact with the GPIO endpoint function.

6.2.2.1. Input Switch

There are five input LEDs present in the GPIO input box. These LEDs change status when user change the position of the input switches on the board. On CrossLink-NX and CertusPro-NX devices, this is mapped to SW1 on the board, and there is one LED per input switch. On Certus-NX devices, the first four switches are mapped to SW10, and the fifth switch is mapped to SW3. For MachXO5-NX, all five switches are mapped to SW1. To start reading input from the switches, click the Read Input button. An example of the LEDs is shown in Figure 6.7.

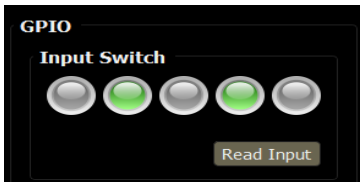


Figure 6.7. GPIO Input Switch.

6.2.2.2. Output LED

The Output LED section shows the status of five output LEDs. These LEDs represent the five bits of a counter, which begin to increment when user click the Run counter button on the user interface, as shown in Figure 6.8.

Figure 6.8. GPIO Output LEDs

6.2.3. I²C (For CrossLink-NX, Certus-NX and CertusPro-NX devices)

This section describes how to use the GUI to read from and write to the I²C device.

6.2.3.1. Test Mode

This control has two options for testing the I²C interface: Normal Mode and Batch Mode. In Normal Mode, user enter each I²C command manually. In Batch Mode, the I²C commands are loaded from a script.

6.2.3.2. Settings

This control is used to configure the I²C bit-rate. Two bit-rates, 100 kHz and 400 kHz, are supported and can be selected from the Bitrate drop down box as shown in Figure 6.9. After selecting the desired speed from the drop down box, user must click the Set button to apply the settings to the device.

Figure 6.9. I²C Bit-Rate Selection

6.2.3.3. Slave addr

This control is used to program the I²C slave address target. The user interface supports both 7-bit and 10-bit addressing modes. Note that CertusPro-NX devices support only 7-bit addressing mode. The address mode can be selected by selecting the appropriate radio button as shown in Figure 6.10. The 7-bit mode is selected by default.

Figure 6.10. I²C Address Program

6.2.3.4. Master Write

The Master write option is used to write data to the I²C device. User can either manually type the address and/or the data information into the Message box, or user can load the data from a hex file by clicking the Load button. Clicking the Master Write button initiates the write operation. The interface is shown in Figure 6.11.

User can clear or save the message box contents into a file. To do so, click the Save button, provide the desired path, enter the name of the file, and click **OK**. This saves the data into the specified hex file. The I²C interface is spot-checked using the camera: Sony IMX258-0AQH5. The camera module is included with the CrossLink-NX board. For Certus-NX and CertusPro-NX boards, the camera module must be obtained separately. The testing procedure is described below.

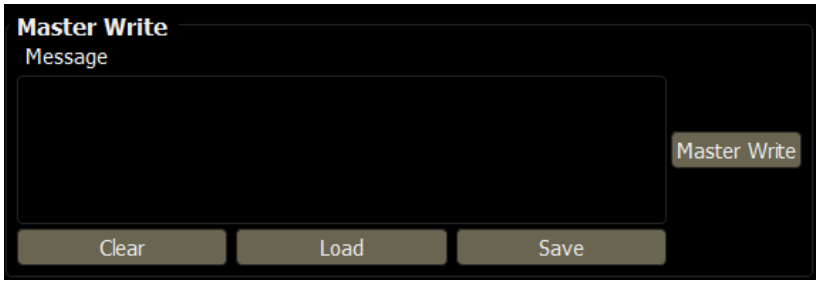


Figure 6.11. I²C Master Write

Table 6.1. Master Write Control Description

Controls	Description
Clear	Clears the input box
Load	Loads the file configuration hex file into the input box
Save	Saves the input box data into the hex file
Master Write	Writes the input box data on the I²C device

IMX258-0AQH5 Camera Write Protocol

Figure 6.12 shows the communication protocol format for the IMX258 camera for a single write to an arbitrary address. Figure 6.13 shows the format for a sequential write starting from an arbitrary address.

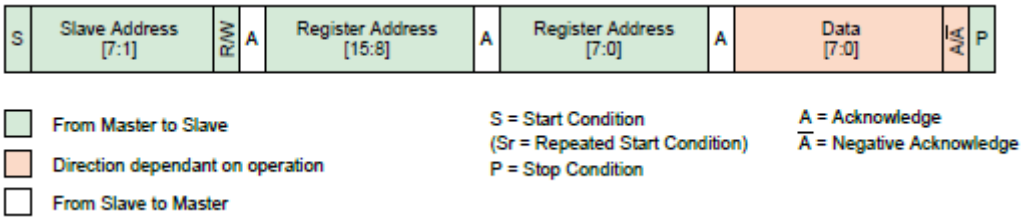


Figure 6.12. Single Write to an Arbitrary Address of IMX258-0AQH5 Camera

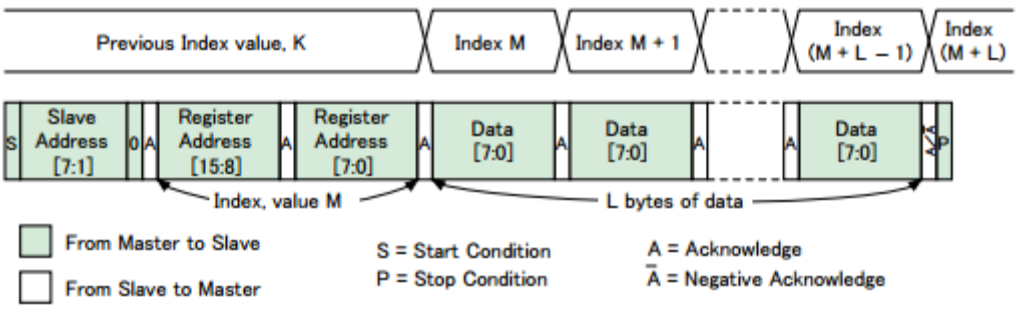


Figure 6.13. Sequential Write Starting from an Arbitrary Address of IMX258-0AQH5 Camera

To test the I²C interface, data can be written to the register 0x3093 of the camera, as shown in Figure 6.14. Additional register information can be obtained through the IMX258-0AQH5 camera data sheet available from Sony. An example of how to execute this transaction is discussed in the next section.

Table 6.2. Registers of MX258-0AQH5 CMOS Camera

Address	Bit	Name	Description
0x3092	[1:0]	MANUAL_DATA PEDESTAL_VALUE [9:8]	Manual setting value for Data Pedestal Default setting value: 0x40
0x3093	[7:0]	MANUAL_DATA PEDESTAL_VALUE [7:0]	
0x3090	[0]	MANUAL_DATA PEDESTAL_ENABLE	Output black level is controlled with Manual_DTA_PEDESTAL_VALUE register value 0x0 : pedestal = 0x40 (fixed value) 0x1 : pedestal = MANUAL_DATA_PEDESTAL_VALUE Else : inhibited value
0x0340	[7:0]	FRM_LENGTH_LINES [15:8]	The length of frame Unit: lines Format: 16-bit unsigned integer *set value for given capture mode ≤ 65525d
0x0341	[7:0]	FRM_LENGTH_LINES [7:0]	
0x0342	[7:0]	LINE_LENGTH_PCK [15:8]	The length of line Unit: pixels Format: 16-bit unsigned integer *Set to 5352d. For any other value change required, confirm with SONY.
0x0343	[7:0]	LINE_LENGTH_PCK [7:0]	

Master Write Procedure

To perform a write operation to the camera's register address 0x3093:

1. Select **Normal Mode** from the **Test mode** drop down box.
2. Select **100 kHz** bit-rate from the **Bitrate** drop down box and click the **Set** button.
3. Enter **0x1A** in the **Slave addr** box.
4. Select the **7-bit mode** radio button.
5. Enter the data and address into the message box – two bytes of address and one byte of data as shown in Figure 6.14. For example, to write address 0x3093 with 0xFF data, then enter "30 93 FF" into the message box.
6. To initiate the transaction, click the **Master Write** button. To verify that the write operation is successful, follow the instructions on how to perform a Master Read operation.

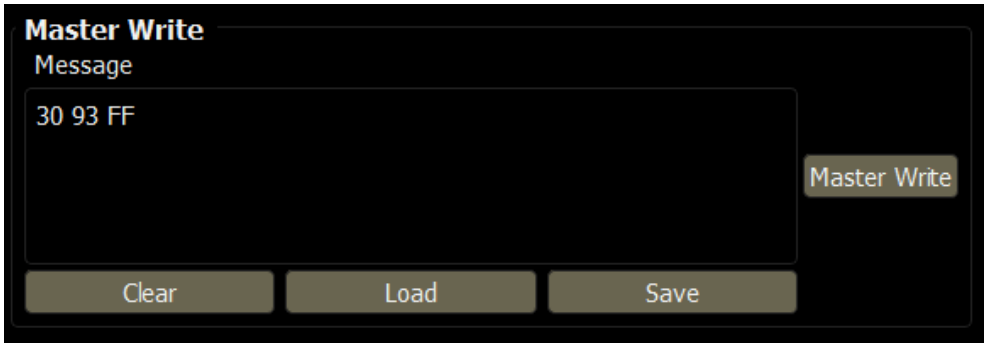


Figure 6.14. I²C Master Write Procedure

6.2.3.5. Master Read Procedure

The Master read option is used to read the data from the I²C device. This section describes the Master read procedure with an example IMX258-0AQH5 camera.

Master Read from a Held Address

The Master Read option is used to read data from the held address of the I²C device. When a master transmits the slave address but does not designate an index or address, the previously read from address value is used or *held*. A single byte or sequential data can be read from the held address.

Figure 6.15. I²C Master Read

IMX258-0AQH5 Camera Read Protocol From a Held Address

Figure 6.16 shows the communication protocol format for the IMX258 camera for a single read from the held address. Figure 6.17 shows the format for a sequential read starting from the held address.

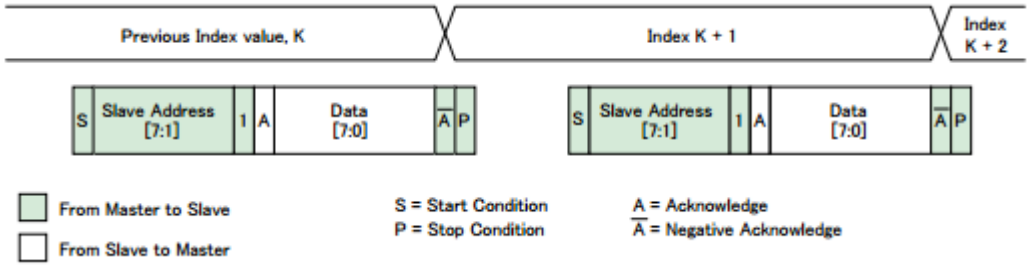


Figure 6.16. Single Read from the Held Address of IMX258-0AQH5 camera

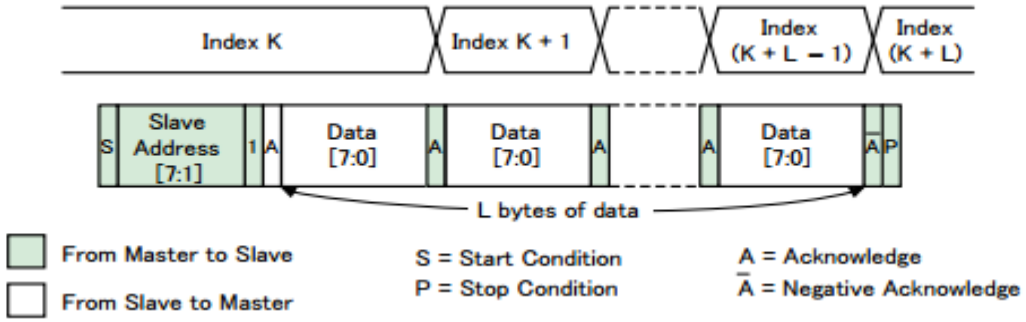


Figure 6.17. Sequential Read Starting from the Held Address of IMX258-0AQH5 Camera

An example of performing a single read from the held address (in this example, 0x3093) is shown below.

To perform a Master Read from a held address:

1. Select **Normal Mode** from the **Test mode** drop down box.
2. Select **100 kHz** bit-rate from the Bitrate drop down box and click the **Set** button.
3. Enter **0x1A** in the **Slave addr** box.
4. Select the **7-bit mode** radio button.
5. Enter the register address in the Master Write message box as shown in below Figure 6.18 to set the current register address. For an example, to read from 0x3093 register address, enter **30 93** in the message box and click the **Master Write** button.

Figure 6.18. I²C Master Write to Write the Register Address

6. Enter the number of bytes to read in the **Number of Data Byte** box as shown in Figure 6.19 and click **Master Read**. The results of the read transaction are shown in the transaction log table, and the data value read out can be found in the data column of the transaction log table.

Figure 6.19. I²C Master Read

Master Register Read from an Arbitrary Address

The Master Register Read option is used to read data from an arbitrary address of the I²C device. A single byte or sequential data can be read from an arbitrary address.

Figure 6.20. I²C Master Register Read

Table 6.3. Master Register Read Control Description

Controls	Description
Register Addr	Specifies the register address to be read.
Addr Width	Specifies the address width. Valid values are 1 Byte and 2 Bytes.
Num of Bytes	Specifies the number of bytes to be read from the I²C device.
Register Read	When this button is pressed, the read operation is performed.

IMX258-0AQH5 Camera Read Protocol from an Arbitrary Address

Figure 6.21 shows the communication protocol format single read from an arbitrary address and Figure 6.22 shows the sequential read starting from an arbitrary address of MX258-0AQH5 Camera.

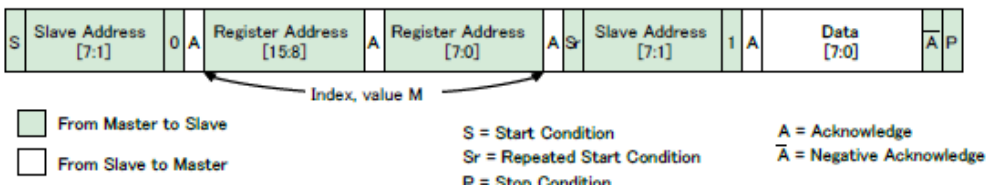


Figure 6.21. Single Read from an Arbitrary Address of MX258-0AQH5 Camera

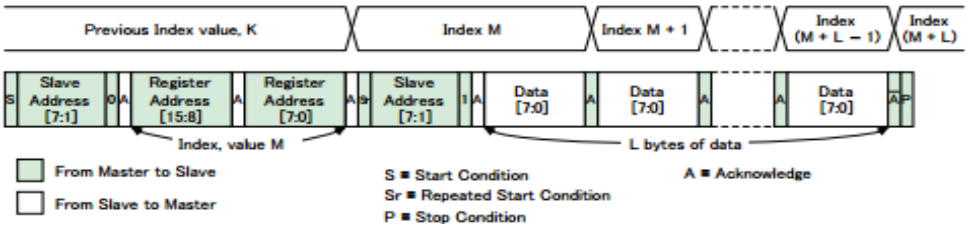


Figure 6.22. Sequential Read Starting from an Arbitrary Address of MX258-0AQH5 Camera

To perform a Master Read operation:

1. Select **Normal Mode** from the **Test mode** drop down box.
2. Select either **100 kHz** or **400 kHz** from the **Bitrate** drop down box and click the **Set** button.
3. Enter **0x1A** in **Slave addr**.
4. Select the **7-bit mode** radio button.
5. Enter **3093** in the **Register Addr** field.
6. Select **2Bytes** in **Addr Width**.
7. Enter **1** in **Num of Bytes**.
8. Click the **Register Read** button and check the transaction log for the readback value.

Figure 6.23. I²C Master Register Read

Note: I²C does not work without power cycle, if any step goes wrong.

6.2.3.6. Transaction Log

The Transaction Log shows the details of all transactions on the I²C device as shown in Figure 6.24. User can save and clear the log. To save the log into file click **SaveToFile**, select the desired location and enter the log file name. The log is saved in text format. To clear the log from the table, click the **ClearLog** button.

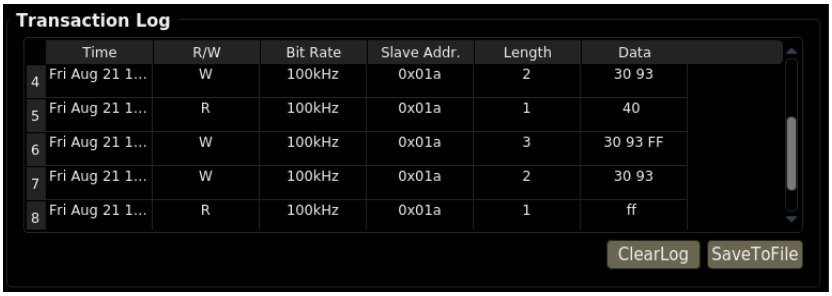


Figure 6.24. I²C Master Register Read

The log table is updated only when the transaction is successfully completed. Failed transactions are not logged in the table. The description of each column of the transaction log table is described in Table 6.4.

Table 6.4. Transaction Log Control Description

Column	Description
Time	The start time of the transaction.
R/W	Specifies whether the transaction was a read or write.
Bit Rate	Specifies what bit rate was chosen for the transaction.
Slave Addr	Specifies the I²C slave address of the transaction.
Length	Specifies the number of bytes transferred during the transaction.
Data	Specifies the transferred data during the transaction.

6.2.3.7. Batch Mode

Batch mode is used to read and write the I²C device using an XML file. User can load an XML file by clicking the **Load** button. Click the **Execute** button to run the commands in the XML file. To abort the operation, press the **Stop** button. After completion, an **Updated successfully** message is displayed. An example of Batch mode XML is shown in Figure 6.25.

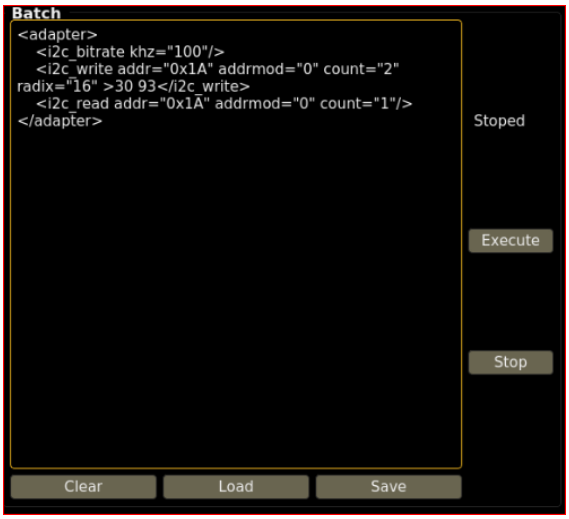


Figure 6.25. I²C Batch Mode Read/Write

XML node description:

- i2c_bitrate – elements used to configure the I²C device bit-rate.
- kHz – attributes to I2C bit rate value.
- i2c_write – elements used to write address and data or address-only.
- addr – attribute that holds the I²C slave device address.
- addrmod – attribute that specifies the addressing mode, with a value of 0 for 7bit and 1 for 10bit.
- count – attribute that specifies the number bytes to be written or read.
- radix – attribute that specifies the radix of the device address and data format.
- i2c_read – elements used to read data from the device.

Table 6.5. Batch Mode Control Description

Control	Description
Clear	Clears the input box.
Load	Loads the xml configuration file in the input box.
Save	Saves the input box configuration to a file.
Stopped	Indicates the Master device status when any operation is run.
Execute	Used to start batch file execution.
Stop	Used to stop the execution of the batch file.

6.2.4. MDIO (for CrossLink-NX and Certus-NX devices)

This section describes how to use the user interface to read and write the MDIO device. This section is only applicable for CrossLink-NX and Certus-NX devices.

6.2.4.1. Phy Addr

Phy Addr specifies the address of the PHY to access. The valid address range is from 0x0 to 0x1F. Both the CrossLink-NX PCIe Bridge Board and the Certus-NX Versa Evaluation board only have a single PHY connected at address 0x0.

Figure 6.26. MDIO Phy Addr

6.2.4.2. MDIO Read

This control is used to read the specified MDIO register, as shown in Figure 6.27. Enter the register address as **0x0** in **Addr** and click the **Read** button. **Value** is updated to show the data read from the specified address. The CrossLink-NX PCIe Bridge Board PHY is the DP83867 Ethernet Phy. The address **0x0** contains the default value of **0x1140**, which appears after clicking the **Read** button.

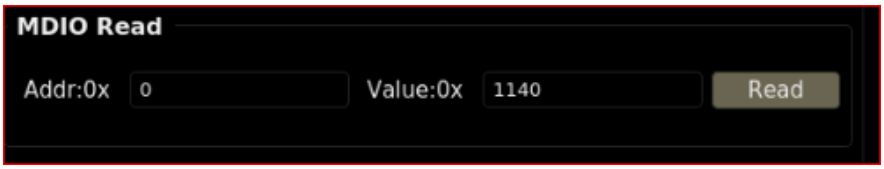


Figure 6.27. MDIO Configuration Register Read

6.2.4.3. MDIO Write

This control is used to write the specified MDIO register as shown in Figure 6.28. After entering the Phy register address in **Addr** and the data in **Value**, click the **Write** button. After clicking the **Write** button, the data is written to that given address. For testing purposes, write **0x40** to the Phy (DP83867 Ethernet Phy) register address **0x1F** and verify by reading back using the MDIO Read function.

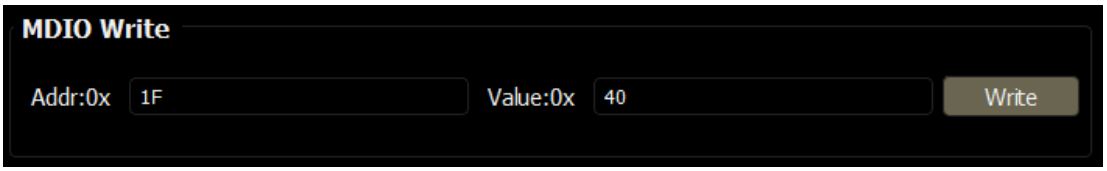


Figure 6.28. MDIO Register Write

6.2.4.4. File

This control is used to load MDIO register commands from a file. User can browse the MDIO configuration file by clicking the **Load** button, which opens a dialog box to browse for a configuration file. After loading the file, the MDIO commands loaded from the file is shown in the table. User can save the table data into a file by clicking the **Save** button. A dialog box opens to allow the user to indicate the file name and location.

Figure 6.29. MDIO Configuration through File

Table 6.6. MDIO Batch Mode Control Description

Control	Description
Load	Used to load an MDIO configuration file.
Save	Used to save table content to a file.

Table 6.7. MDIO Batch Mode Table Description

Column Name	Description
Operation	Specifies whether the command is a read or write operation
Address	Specifies the read/write address.
Data	Specifies the data read from or to be written to the specified address
Status	Specifies the status of the operation as either PASS or Fail

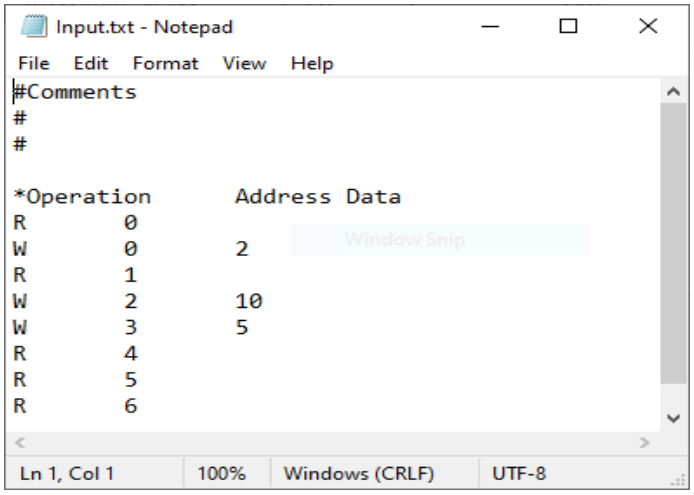


Figure 6.30. MDIO Configuration File Example

The MDIO configuration file is a tab separated text file where each column is separated by a tab as shown in Figure 6.30. Comment lines begins with the '#' symbol while column headers begins with a '*'. These lines are ignored while parsing the file.

Table 6.8. MDIO Input File Description

Column Name	Description
Operation	Specifies the operation type as R for read, and W for write.
Address	Specifies the register address in this column.
Data	For write operations, add a column specifying the data to be written. For read operations, this column needs to be kept blank.

7. Troubleshooting

7.1. SPI Flash Update

- If you are getting a verification error while dumping the .bit file, try changing the TCK frequency to a value greater than 4. The TCK Divider Setting option is in the Cable Setup dialog box of the Lattice Radiant Programmer Window, as shown in Figure 7.1. Restart programming by clicking the Program button as shown in Figure 5.16.

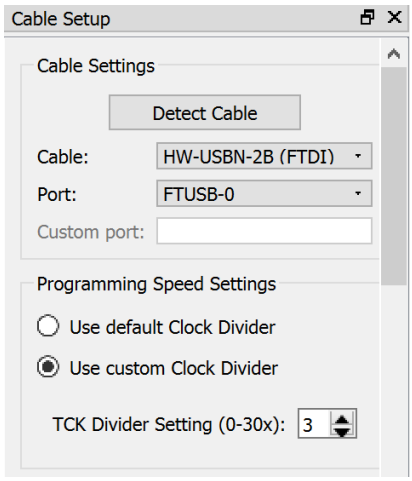


Figure 7.1. TCK Frequency Setting

- If the verification error problem still occurs, press and hold the PROGRAMN push button on the board before clicking the Program button.
- If the device is not getting detected on port FTUSB-0. Click Detect cable and select the port FTUSB-1.

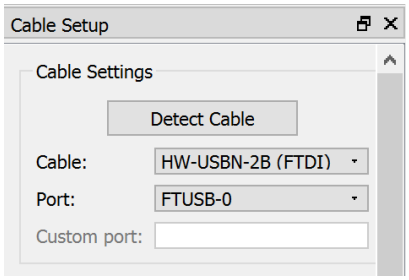


Figure 7.2. Port Selection

7.2. Driver Installation and User Interface Launch for Windows

This section describes troubleshooting steps for the Windows operating system.

7.2.1. Problem in Driver Installation

- If the hardware IDs are not found in Device Manager, ensure that the board is properly installed in a PCIe slot.
- Ensure that the board has a valid PCIe bitstream file loaded in the SPI Flash. Shut down the system and try another slot. If the board is present, check the Properties and the Resource tab to verify if memory is assigned to it. Also, verify if the Vendor ID and Device ID are valid, as seen by Windows Plug-n-Play. If the values are invalid, perhaps the bitstream file is corrupt and needs to be reloaded into SPI flash. If the PCIe board is shown in the list of Device Manager, install the driver manually.
- If the driver is not being installed even though the device is present in the Device Manager, make sure that the driver signature enforcement is disabled permanently. If not, follow the steps described in the Disabling Driver Signature Enforcement Permanently section.

7.2.2. Problem with Launching User Interface

Ensure that the driver is installed properly. Otherwise, the Device Info section of the user interface displays the message shown in Figure 7.3.

The *Driver Open [FAILED]* message is displayed if the driver is not installed properly. The Device information is present if the drivers are installed properly.

Figure 7.3. User Interface with No Device Driver

If any such error message occurs, ensure that the device is properly inserted in the system. After that, verify that drivers are installed. When all the issues are resolved, restart the user application.

When using file saving operations in the Memory portion of the demo, make sure that user have write permissions for the location where the file is being saved. Similarly, for file reading operations, the directory from where the file is being loaded must have Read Permission.

If the Multifunction demo short cut icon is accidentally deleted, then the application can be launched by double clicking on *lattice_bd.exe* file present in *[INSTALLATION FOLDER]/Lattice Semiconductor/Multifunction Demo/bin* folder.

7.3. Driver Installation User Interface Launch for Linux

This section describes troubleshooting steps for the Linux operating system.

7.3.1. Problem in Driver Installation

The following error may occur during software driver installation.

Issue: insmod: ERROR: could not insert module *lattice_main.ko*: Operation not permitted.

Resolution: Ensure that the Secure Boot option is disabled in the BIOS of the system. Refer to the BIOS user manual to learn about the steps for turning off the Secure Boot option.

Issue: insmod: ERROR: could not insert module *drv_src/drvr/lattice_main.ko*: Invalid module format.

Resolution: Run the following commands sequentially in the terminal.

```
sudo apt update && sudo apt upgrade
sudo apt remove --purge linux-headers-*
sudo apt autoremove && sudo apt autoclean
sudo apt install linux-headers-generic
sudo apt-get install build-essential linux-headers-`uname -r`
```

7.3.2. Problem in Driver Loading

After launching the user interface, if the driver is not loaded properly, a *Driver open[Failed]* message is displayed in the Info section.

- Make sure that the board is properly connected to the PC.
- Run the command below in a terminal window to show the list of PCIe devices connected to PC.

  ```
  sudo lspci -vnm
  ```

  Ensure that the Lattice device is present by checking the device and vendor IDs, as shown in Figure 7.4 for the CrossLink-NX board, Figure 7.5 for the Certus-NX board, Figure 7.6 for the CertusPro-NX board, and Figure 7.7 for MachXO5-NX.

```
Device: 01:00.0
Class:  0702
Vendor: 1204
Device: 9c22
SVendor:        19aa
SDevice:        e004
Rev:    10

Device: 01:00.1
Class:  0702
Vendor: 1204
Device: 9c1f
SVendor:        19aa
SDevice:        e004
Rev:    10

Device: 01:00.2
Class:  0702
Vendor: 1204
Device: 9c20
SVendor:        19aa
SDevice:        e004
Rev:    10
```

Figure 7.4. lspci -vnm for CrossLink-NX Output Image

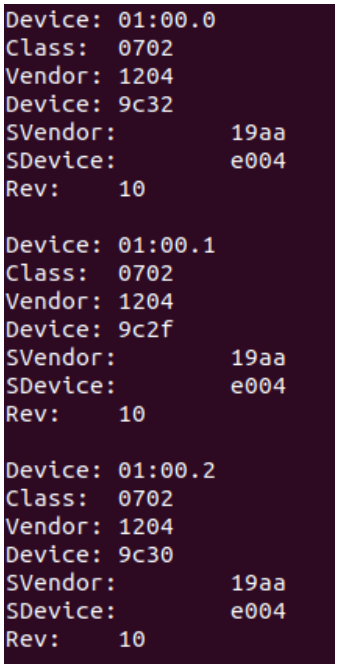


Figure 7.5. lspci -vnm for Certus-NX Output Image

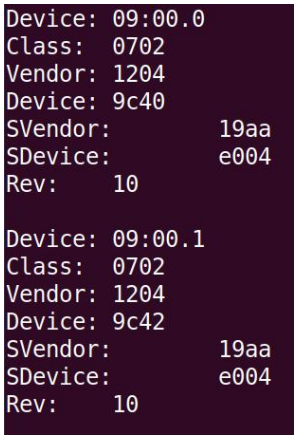


Figure 7.6. lspci -vnm for CertusPro-NX Output Image

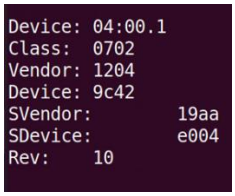


Figure 7.7. lspci -vnm for MachXO5-NX Output Image

- If the device is present, perform the manual setup and installation steps one by one.
- If the driver does not build properly, check for any software or kernel dependencies.

7.3.3. Problem with User Interface Launching

- Go to the *Demonstration/Linux* directory. Check the content of this directory. The default content is shown in Figure 7.8. Check the permissions of these file.

```
lattice@lattice:~/mul6/CL-NX_BridgeBoard_PCIe_Multifunction/Demonstration/Linux$ ls -l
total 16
-rw-r--r-- 1 root root  175 Feb 16 11:53 install.sh
-rw-r--r-- 1 root root  237 Feb 16 11:53 Readme.txt
drwxr-xr-x 6 root root 4096 Feb 16 11:53 Source_Code
-rw-r--r-- 1 root root   47 Feb 16 11:53 uninstall.sh
```

Figure 7.8. Content List of Demonstration/Linux Directory.

- The script files should have *execute* permission – to set permissions on the file, run the command below in the terminal and try to run the application again.

```
sudo chmod 777 filename
```

- If the user interface still does not launch, change directory into the Source_Code directory and verify that all files are present in this directory, according to Figure 7.9.

```
lattice@lattice:~/mul6/CL-NX_BridgeBoard_PCIe_Multifunction/Demonstration/Linux/Source_Code$ ls -l
total 36
drwxr-xr-x 5 root root 4096 Feb 16 11:53 app_src
-rw-r--r-- 1 root root   27 Feb 16 11:53 compile.sh
drwxr-xr-x 5 root root 4096 Feb 16 11:53 drv_src
drwxr-xr-x 2 root root 4096 Feb 16 11:53 include
-rw-r--r-- 1 root root  137 Feb 16 11:53 install_drv.sh
-rw-r--r-- 1 root root   52 Feb 16 11:53 launch_gui.sh
drwxr-xr-x 2 root root 4096 Feb 16 11:53 lib
-rw-r--r-- 1 root root  566 Feb 16 11:53 Makefile
-rw-r--r-- 1 root root  156 Feb 16 11:53 Readme.txt
```

Figure 7.9. Content List of Software/Linux Directory

- If all files are present in this directory, change directory to *app_src/gui/deploy* and try to run the command below directly.

```
sudo ./MFDemo.sh
```

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 1.2, May 2023

Section	Change Summary
All	• Applied formatting and editorial changes throughout the document. • Added CertusPro-NX device support where applicable. • Added MachXO5-NX support. • Rearranged specific sections to the following sequence: Demo Design Overview, Importing and Building the FPGA Demonstration, Setting Up the Demo, Application Overview, and Troubleshooting.
Acryonyms in This Document	• Added new acronyms: API, BIOS, DLL, and OS.
Setting Up the Demo	• Updated Table 5.4. Hardware IDs to add MachXO5-NX information. • Updated Figure 5.16. Programmer Menu Bar to highlight the Programming button. • Updated Figure 5.32. Device Configuration Prompt to highlight *Yes* option in message box. • Updated Figure 5.34. Windows Security in Driver Installation to highlight *Install this driver software anyway* option. • Added Figure 5.5. MachXO5-NX Development Board Connection, Figure 5.11. MachXO5-NX FPGA Device Settings, Figure 5.15. Device Properties Window for MachXO5-NX Flash Programming, Figure 5.21. MachXO5-NX Programming Done LED, Figure 5.57. GPIO Device Drivers for MachX05-NX in Device Manager, and Table 5.2. Jumper Configuration.
Application Overview	• Replaced bullet information from *kHz - attributes hold the I2C bit rate value* to *kHz - attributes to I2C bit rate value* in Batch Mode section. • Added Figure 6.3. PCIe Test Application Device Info Tab for MachXO5-NX and Figure 6.6. Functionality Test Tab for MachXO5-NX.
Troubleshooting	• Added reference to Figure 5.16. Programmer Menu Bar in SPI Flash Update section. • Added Figure 7.7. lspci -vnm for MachXO5-NX Output Image.
Technical Support Assistance	Added a link to the Lattice Answer Database.

Revision 1.1, March 2022SS

Section	Change Summary
All	• Changed document title to PCIe Colorbar Demo for Lattice Nexus-based FPGAs. • Updated the contents to be applicable to Lattice Nexus-platform devices, specifically CrossLink-NX and Certus-NX for this release.

Revision 1.0, February 2022

Section	Change Summary
All	Initial release.

www.latticesemi.com