PCIe Multifunction Demo Reference Design for MachXO5-NX using the LFMXO5-65T Device

User Guide

FPGA-UG-02239-1.0

August 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
BIOS	Basic Input/Output System
CSI	Camera Serial Interface
EBR	Embedded Block RAM
DLL	Dynamic Link Library
FDSOI	Fully Depleted Silicon on Insulator
FPGA	Field-Programmable Gate Array
GPIO	General Purpose Input Output
I2C	Inter-integrated Circuit
IP	Intellectual Property
LED	Light-emitting diode
LMMI	Local Memory-Mapped Interface
MIPI	Mobile Industry Processor Interface
PCIe	PCI Express
PHY	Physical Layer
RTL	Register Transfer Level
SPI	Serial Peripheral Interface
USB	Universal Serial Bus

1. Introduction

This guide describes how to set up and run the PCIe Multifunction Demo using devices built on the MachXO5™-NX using LFMXO5-65T FPGA device.

The demo is targeted for MachXO5-NX LFMXO5-65T Development Board, which features the MachXO5-NX using the LFMXO5-65T FPGA device in the BBG484 package.

This guide familiarizes you with the process of setting up and running a design which incorporates PCI Express. It is assumed that you do not have any associated tools installed on your system.

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

1.1. Learning Objectives

After completing the steps in this guide, you can 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 PCIe demo which implements two separate PCI Express functions on a single endpoint device. Each PCIe function allows you to control a different aspect of the FPGA Board.
- Use what the demo teaches you about designing Lattice PCI Express solutions.
- 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 you have already installed the Lattice Radiant™ design software. This document covers some of the basic of function of the Lattice Radiant software. 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 x6, x8, x4, or x1 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
  - MachXO5-NX using LFMXO5-65T Development Board
- Additionally, a MIPI CSI Camera Module is needed to test I^2C functionality; this demo uses the Sony IMX258-0AQH5 Camera Module.

2.2. Software Requirements

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

- Lattice Radiant software version 2025.1 or later (available at the Lattice website Design Software and IP page).
- PCIe Multifunction Protocol demo for Windows or Linux
- Windows 10 or Windows 11
- Ubuntu 24.04.2 LTS or newer Linux versions
- Bit file for the SPI Flash
- Multifunction_Implementation.bit file for MachX05-NX using LFMXO5-65T device

3. Demo Design Overview

3.1. Theory of Operation

The demo runs on a standard ×64 PC and accesses the FPGA board installed in a PCIe slot. Figure 3.1 shows the relationship between the hardware and software components of the demo. The PCIe IP present in the Lattice FPGA as a PCIe endpoint occupying certain ranges of PCI memory space. When the PC boots, the BIOS and OS probe the PCI Express and PCI buses and detect the devices present on the buses and assign them ranges in the PCI memory space. The PCI memory space is mapped into the PC's memory space by the BIOS. Once the device driver is installed, the application software can read/write from/to the PCIe device memory (Embedded Block RAM - EBR). The application software can also be read from the PCIe configuration space registers.

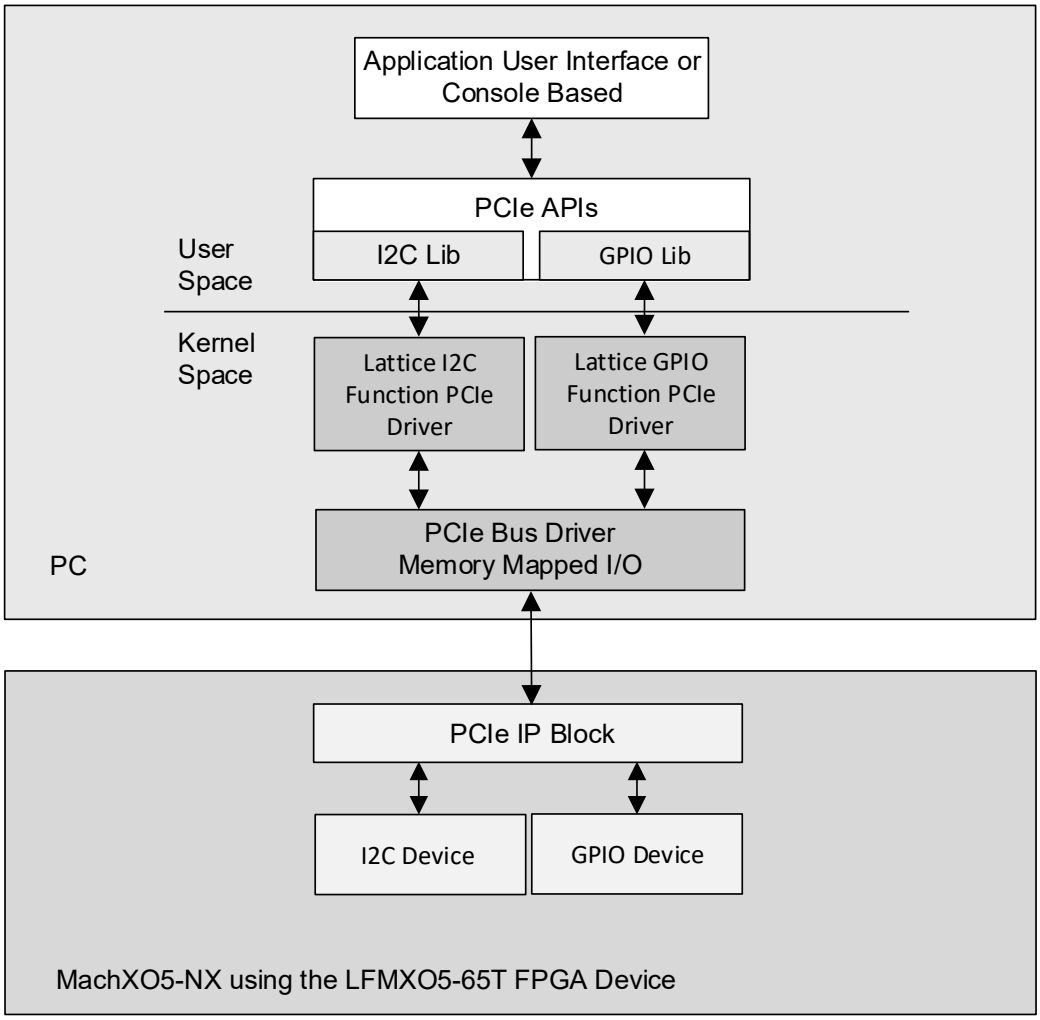

Figure 3.1. Relationship between Hardware and Software Components

3.2. Design Overview

A user interface application is provided for demonstrating the Multifunction demo. 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)

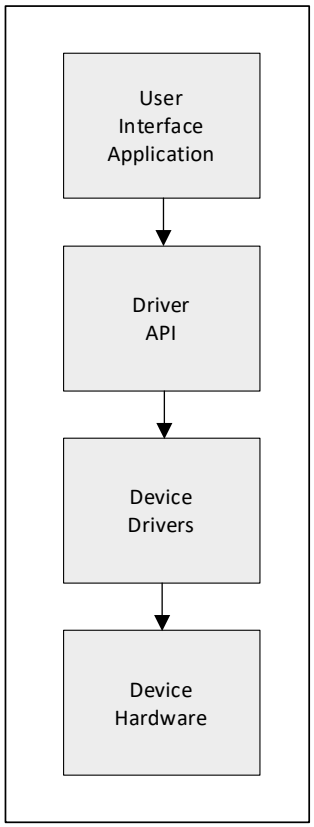

Figure 3.2. Mixed Mode Demo SW Design for PCIe

3.2.1. User Interface Application

The Mixed Mode demo includes a user interface written in Qt and uses the driver API to communicate with the device hardware to send control plane read/writes to registers in the IP. The Driver API is a C++ dynamic library which 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 named *libmem_rw.so*.

3.2.2. Device Drivers

The hardware drivers provide access to the FPGA board. On Windows, the *lpcie_gpio.sys* and *lpcie_i2c.sys*, and drivers support the Mixed Mode demo. On Linux, *gpio_main.ko* and *i2c_main.ko* support the demo.

Figure 3.1 shows the top-level architecture of the FPGA design. The PCIe hard IP is used on the FPGA side 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 shows the block diagram of the FPGA design.

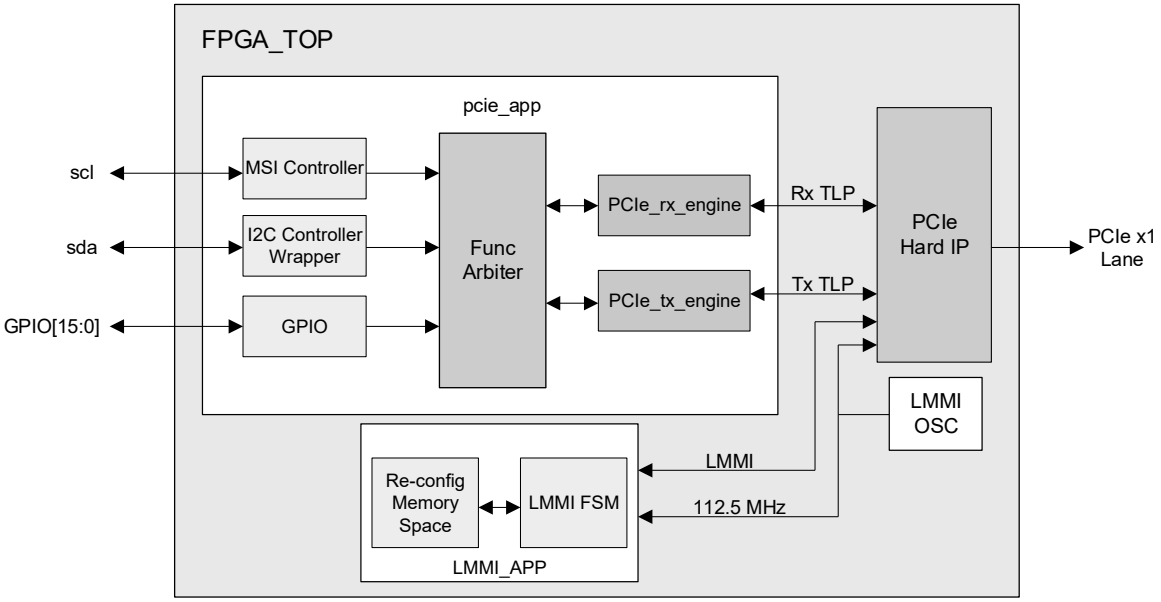

Figure 3.3. Multifunction Demo Design

4. Importing and Building the FPGA Demonstration

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

4.1. Hardware Directory Structure

The Hardware folder inside the package contains the following subfolders.

- 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 Lattice Radiant.
- 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 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
   - The file is MultifunctiomDemo.rdf, which is in the *Hardware*\ folder.
3. Once your project loads, click **Task Detail View**. This shows a list of actions that Lattice Radiant performs to build the .bit file.
4. Select the files and reports that you want to generate.

   Note: The options needed to regenerate the .bit file are selected by default.
5. After selecting your preferred reports, click **Run All**. This creates a .bit file with your 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 FPGA Board.

5.1.1. Jumper Configuration

For MachXO5-NX using LFMXO5-65T, users need to connect JP9 jumpers. The rest please use the default jumpers provided on the board. Refer to the MachXO5-NX using LFMXO5-65T Development board user guide 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 must 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 5.1.

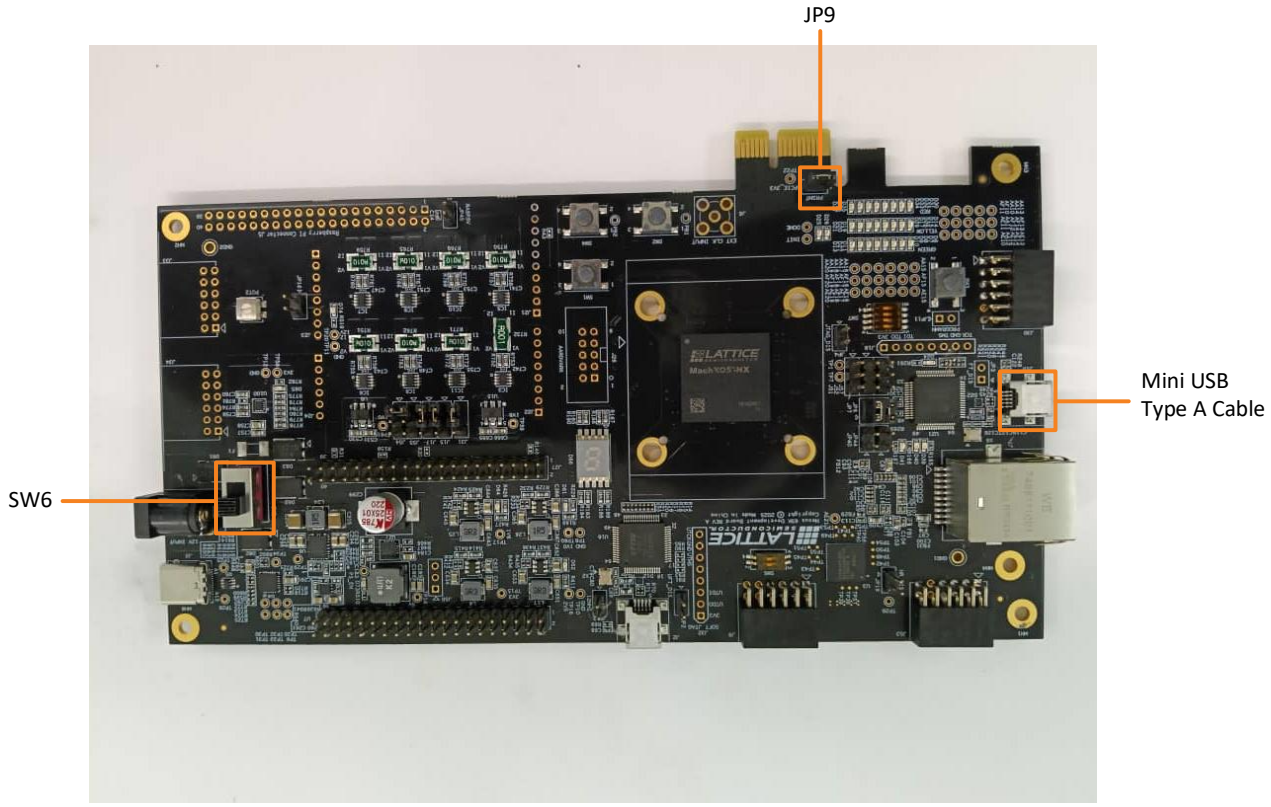

Figure 5.1. MachXO5-NX using LFMXO5-65T Development Board Connection

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, indicate **Project Name** and **Location** as shown in Figure 5.2.
2. Select **Create a new project from scan**. Values are indicated in the **Cable**, **Port**, and **TCK Divider Setting (0-30×)** fields.
3. Click **OK**.

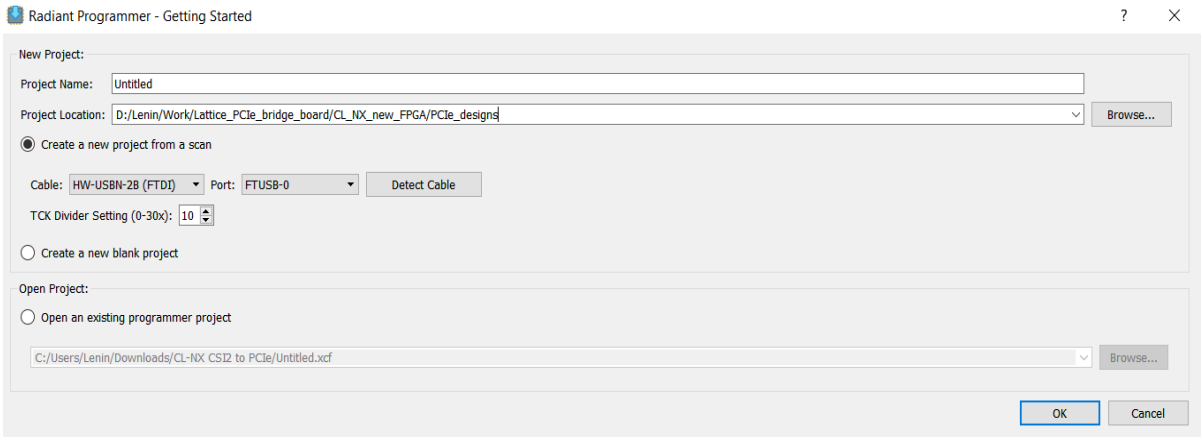

Figure 5.2. Creating a New Project from a Scan

4. The main interface opens as shown in Figure 5.3.

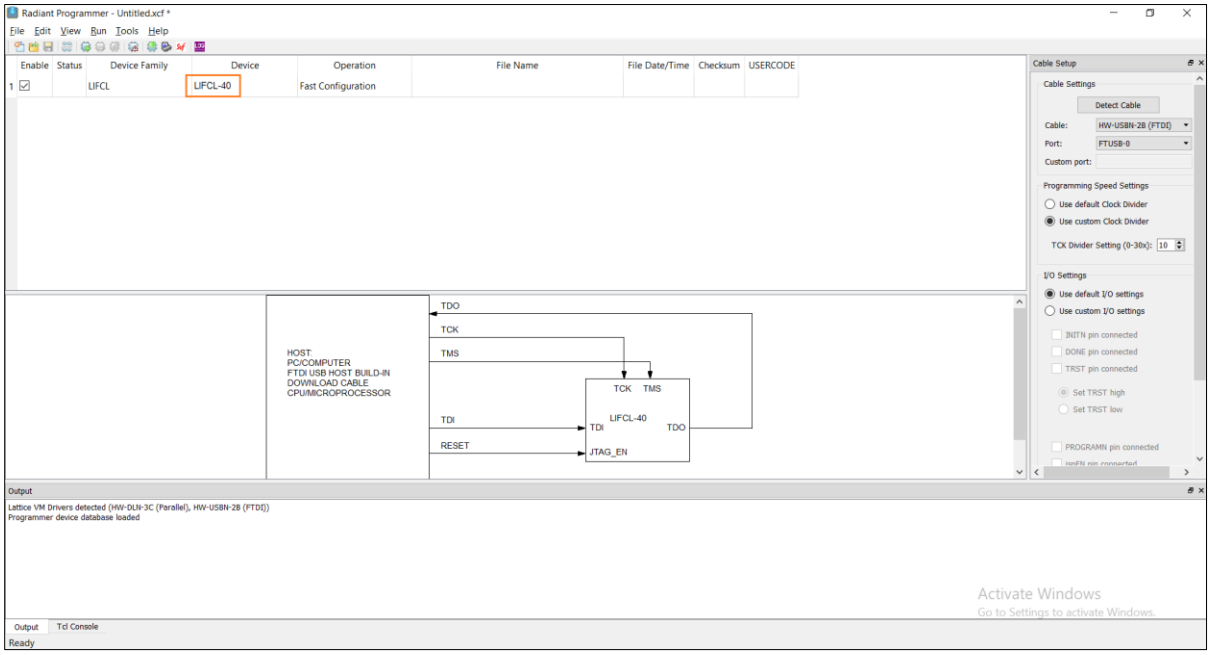

Figure 5.3. Lattice Radiant Programmer Window

5. If the Programmer settings do not match the settings shown in Figure 5.4 for MachXO5-NX using the LFMXO5-65T device, select these settings manually from the drop-down menu.

	Enable	Status	Device Family	Device	Operation	File Name	File Date/Time	Checksum	USERCODE
1	☑		LFMXO5	LFMXO5-65T	Fast Configuration				

Figure 5.4. MachXO5-NX using the LFMXO5-65T FPGA Device Settings

To select the programming settings:

1. Browse and select the Programming file from the *Demonstration\Bitstream* folder.
   a. Select MultifunctionDemo_Implementation.bit.
2. Click **OK**.
3. Under **Operation**, please ensure **Fast Configuration** option is selected.
4. Click the **Programming** button from the menu bar shown in Figure 5.5 to start programming.

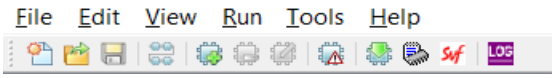

Figure 5.5. Programmer Menu Bar

When the FPGA programming is successful, the output console displays an **Operation: successful** message as shown in Figure 5.6.

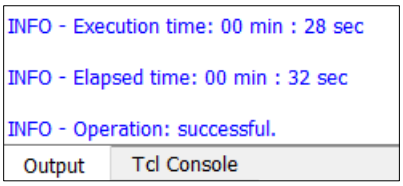

Figure 5.6. Programmer Output Window

If the programming operation is unsuccessful, refer to the Troubleshooting section of this document. After programming, restart the host PC to allow the PCIe Endpoint to link up with the host.

5.2. Software Setup

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

5.2.1. Software Setup and Installation for Windows

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

5.2.1.1. Disabling Driver Signature Enforcement Permanently

To permanently disable the 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
```

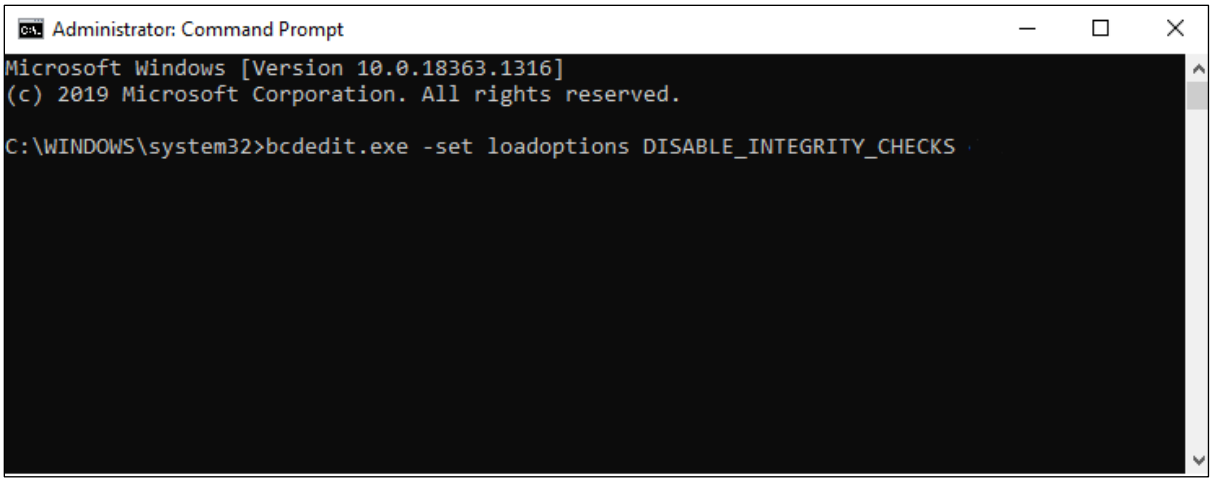

Figure 5.7. Running Disable Integrity Checks Command

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

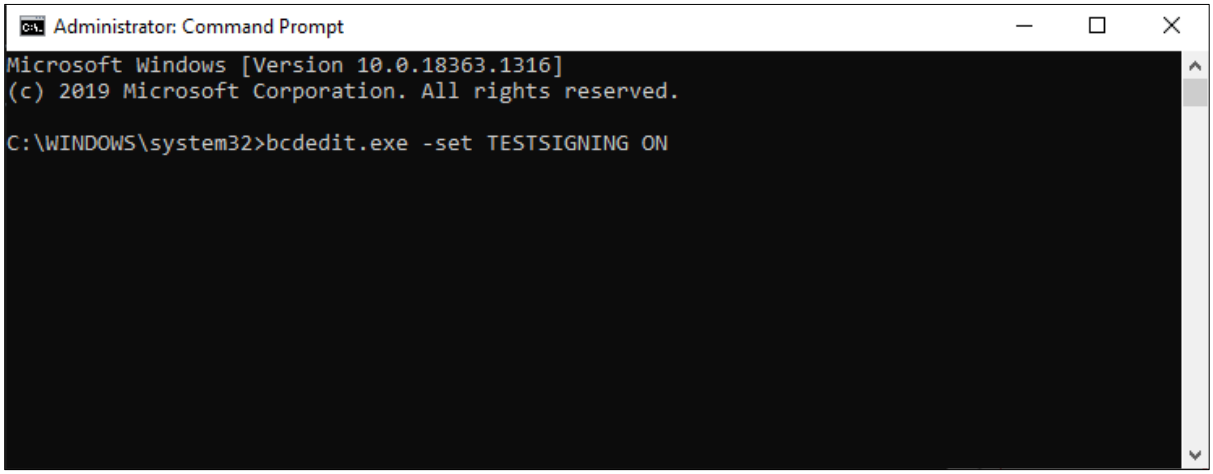

Figure 5.8. Running Test Sign on Command

3. Close the Command Prompt and restart your 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 the Windows operation system:

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 5.9, release the Shift key.
4. Click **Troubleshoot**.

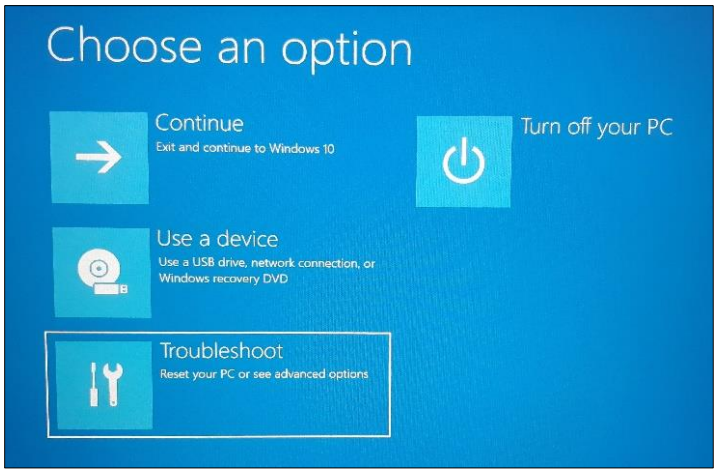

Figure 5.9. Troubleshoot Option

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

Figure 5.10. Advanced Options

6. Select **Startup Settings** and click Restart as shown in Figure 5.12.

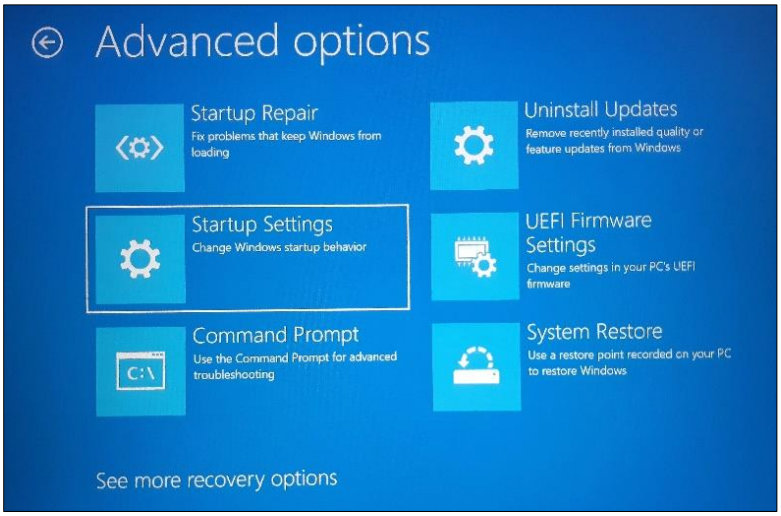

Figure 5.11. Select Startup Settings

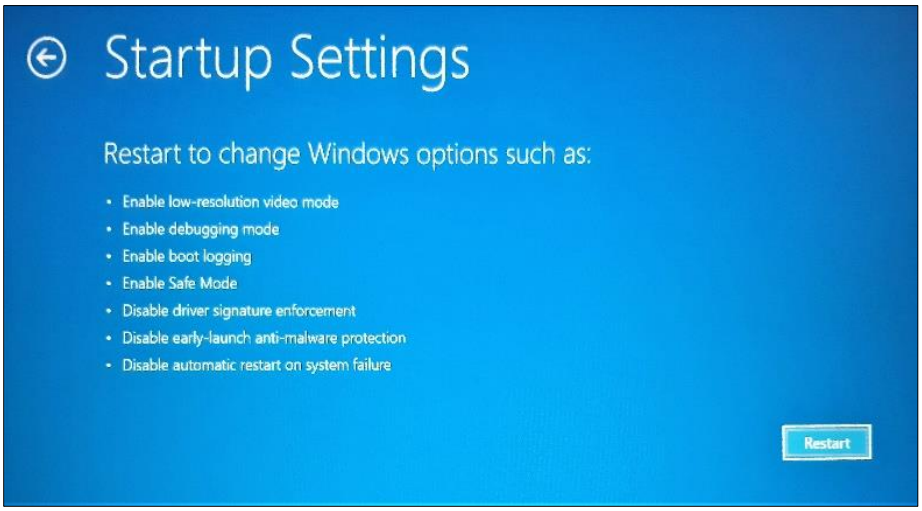

Figure 5.12. Restarting Windows

7. After restarting, select **Option 7** to disable the 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 through Device Manager manually

Installing Multifunction Demo Device Driver through the User Interface Installer

The Multifunction Demo device driver can also 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 your system.

The Framework supported version is Windows operating system.

To install the Multifunction Demo device driver through user interface:

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

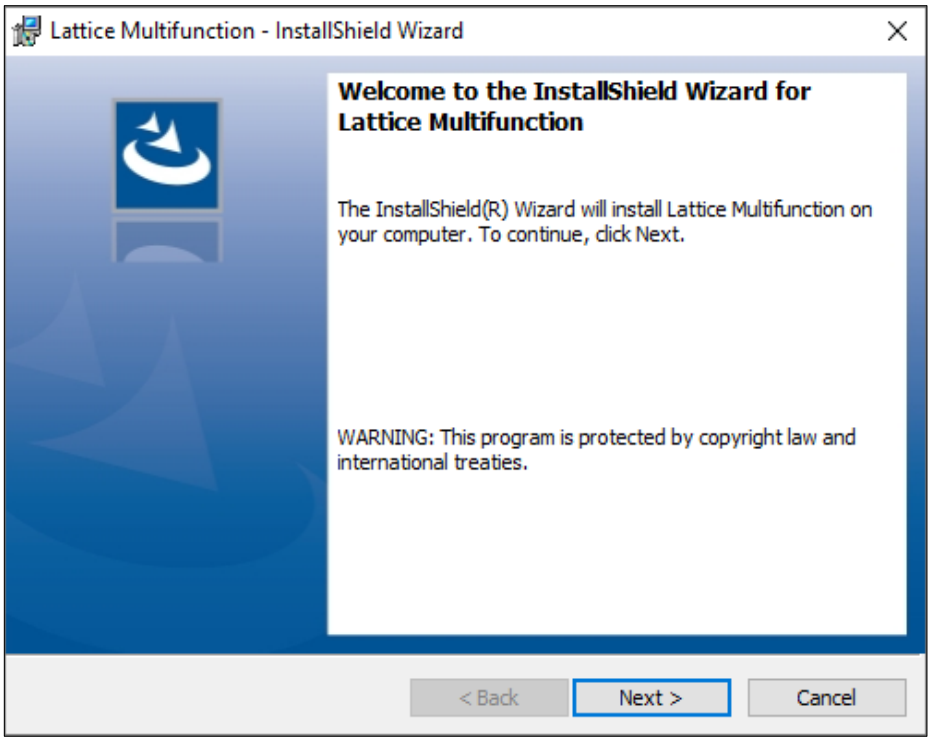

Figure 5.13. Windows Installer: Welcome Page

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

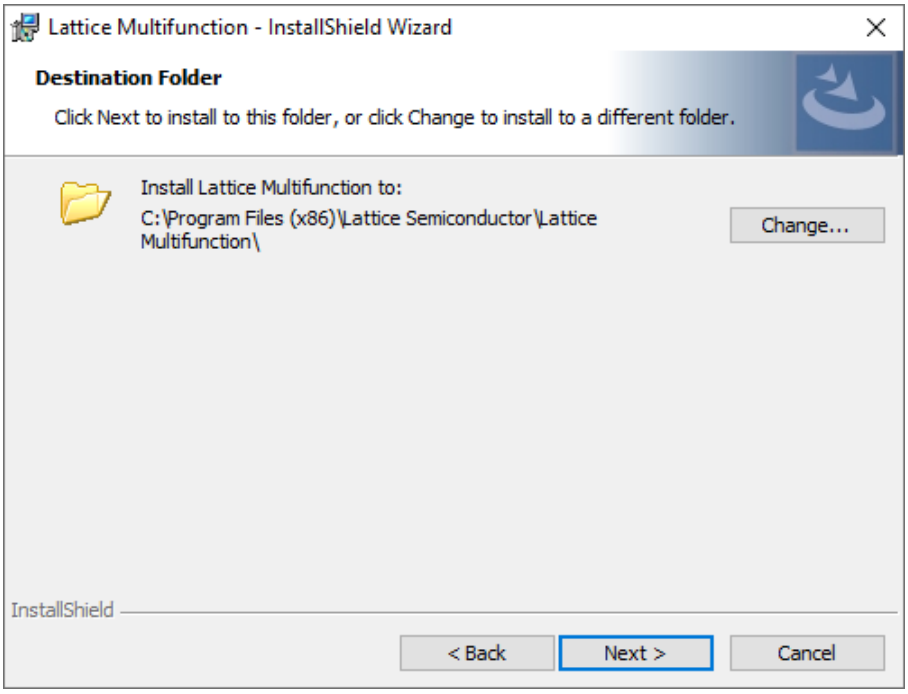

Figure 5.14. Windows Installer: Destination Folder Page

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

   Note: Administrative access is required to run this command.

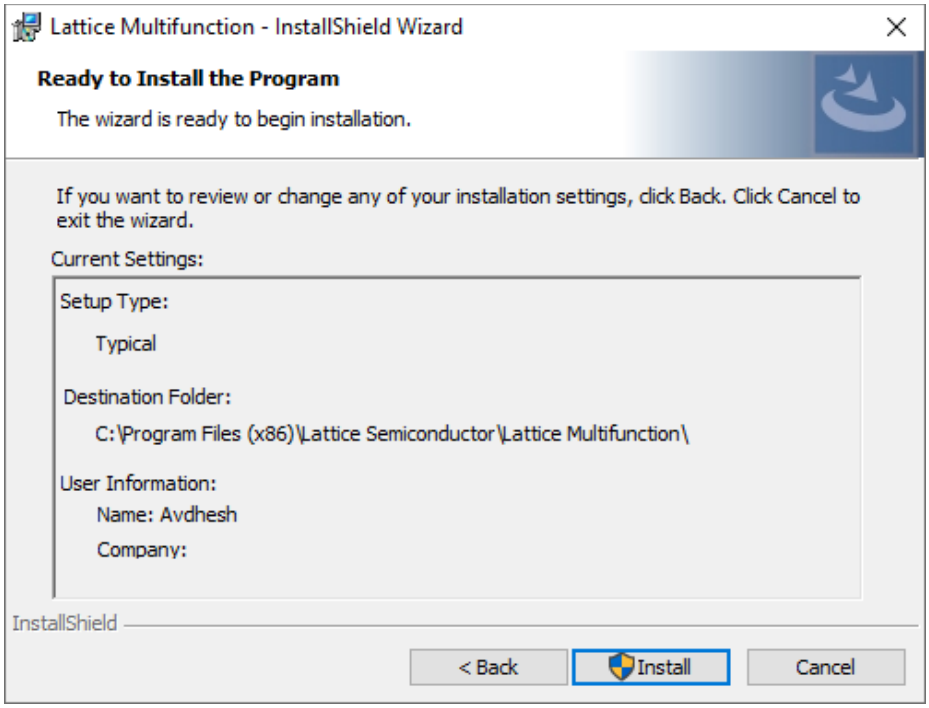

Figure 5.15. Windows Installer: Summary Page

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

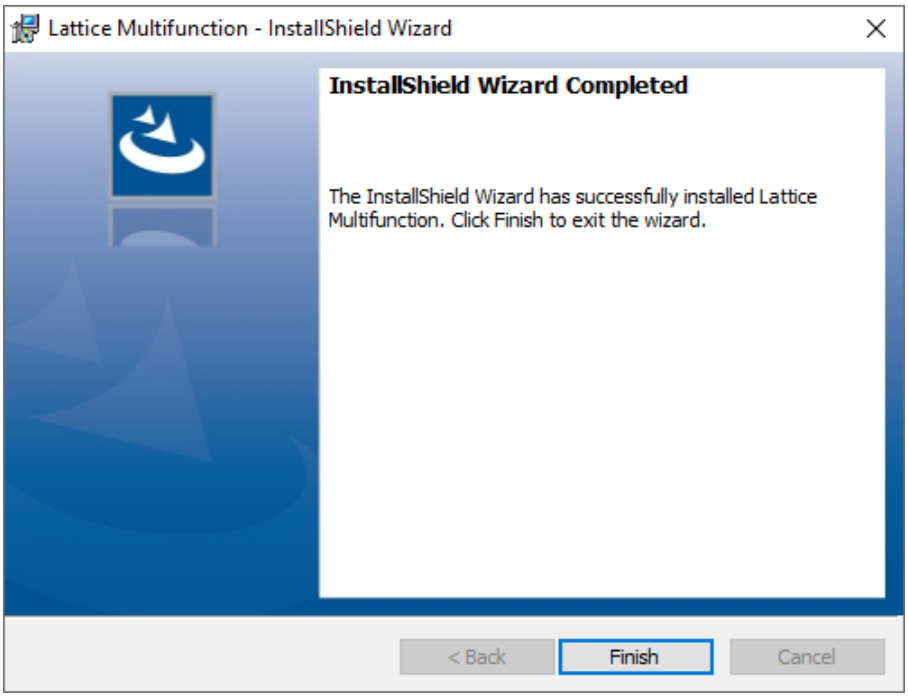

Figure 5.16. Windows Installer: Application Installed

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

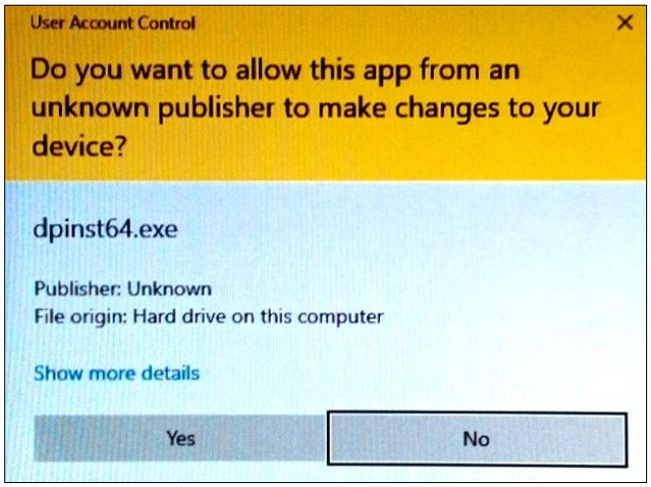

Figure 5.17. Device Configuration Prompt

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

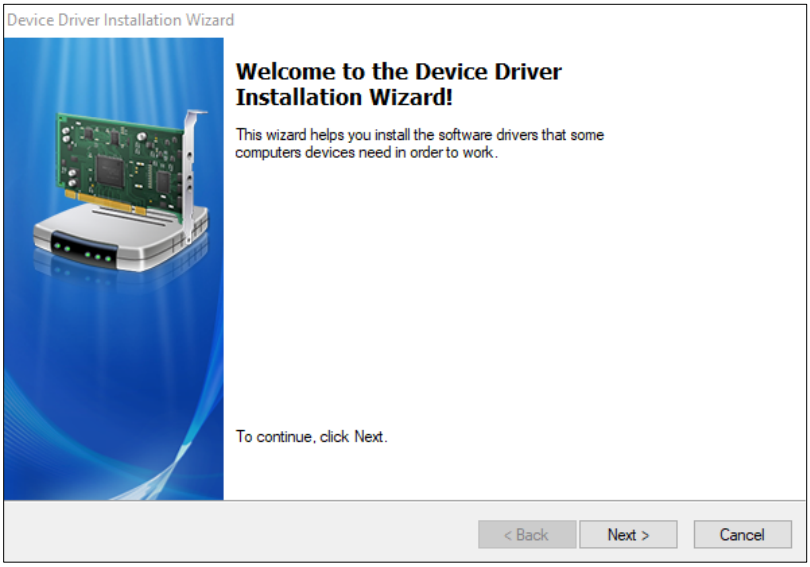

Figure 5.18. Device Driver Installation Wizard

8. If you receive a Windows Security prompt, select **Install this driver software anyway** as shown in Figure 5.19.

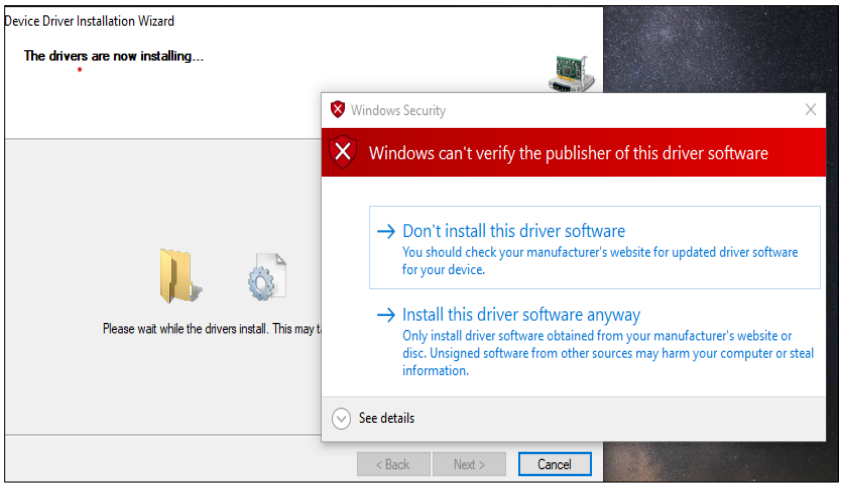

Figure 5.19. Windows Security in Driver Installation

9. If the driver is installed successfully, a message is displayed as shown in Figure 5.20.

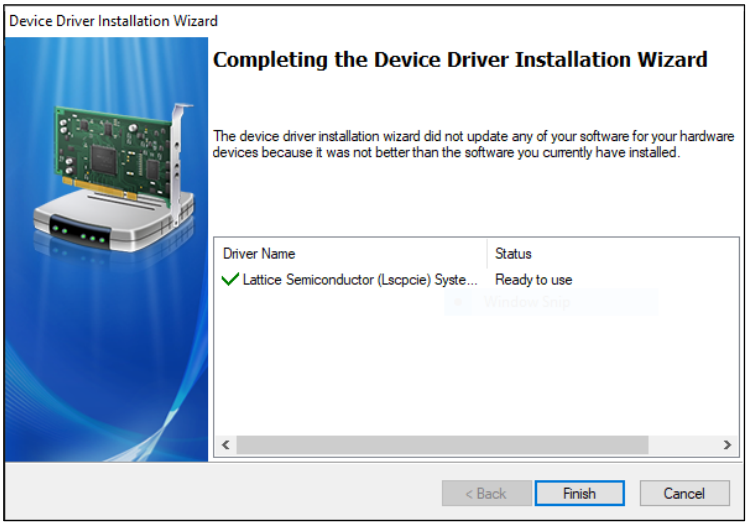

Figure 5.20. Device Driver Installation Completed

Installing 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 5.21.

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

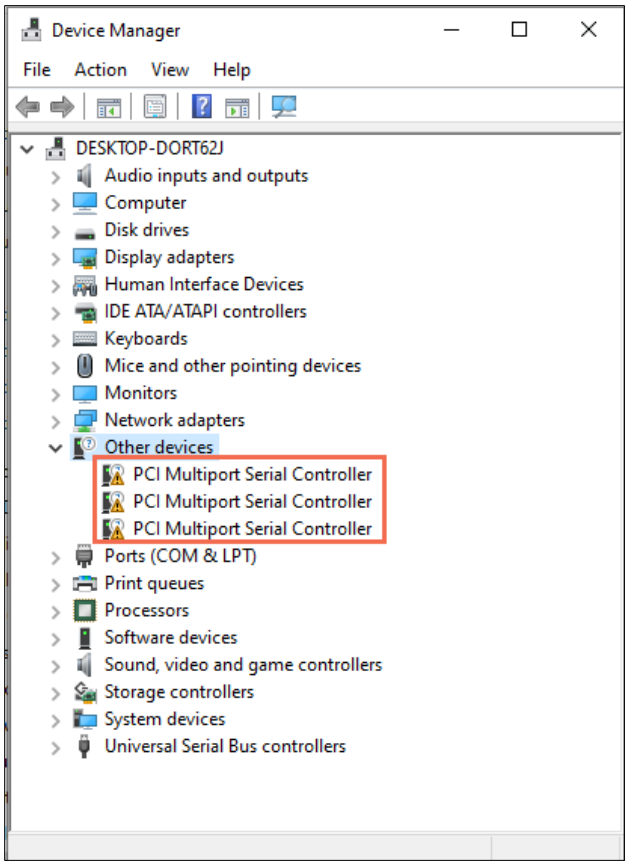

Figure 5.21. Device Manager

2. Right-click each device and select **Properties** as shown in Figure 5.22.

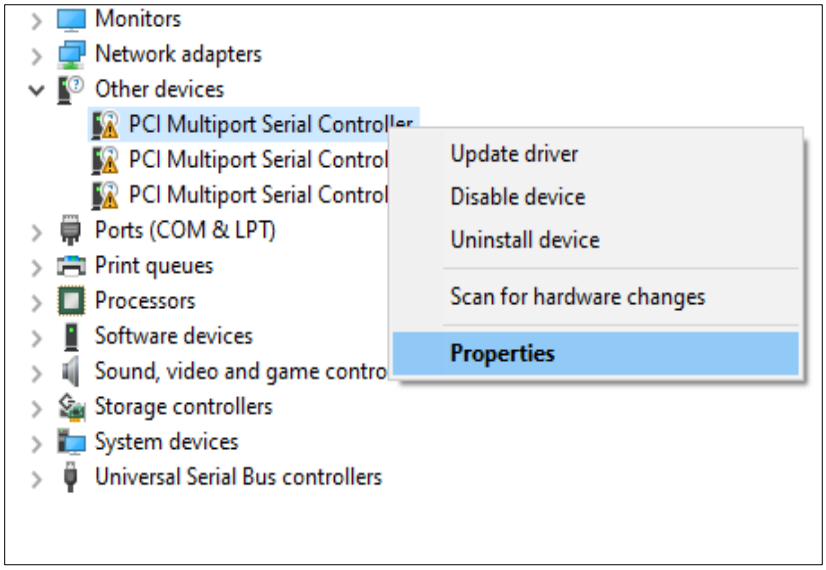

Figure 5.22. Showing Device Properties

3. The information in **Hardware IDs** is needed to install the correct driver for the corresponding device. The Hardware IDs of the I2C, and GPIO devices are shown below:

Table 5.1. Device IDs

Sl. No	MachXO5-NX using the LFMXO5-65T Device
I2C	PCI\VEN_1204&DEV_9C2F&SUBSYS_E00419AA
GPIO	PCI\VEN_1204&DEV_9C30&SUBSYS_E00419AA

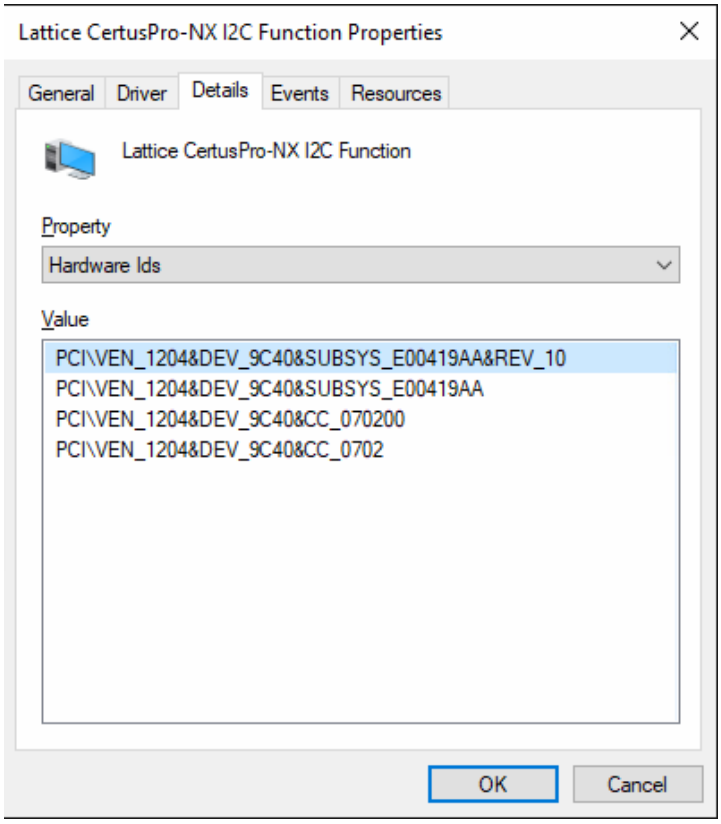

Figure 5.23. Hardware IDs of MachX05 using LFMXO5-65T I2C Device

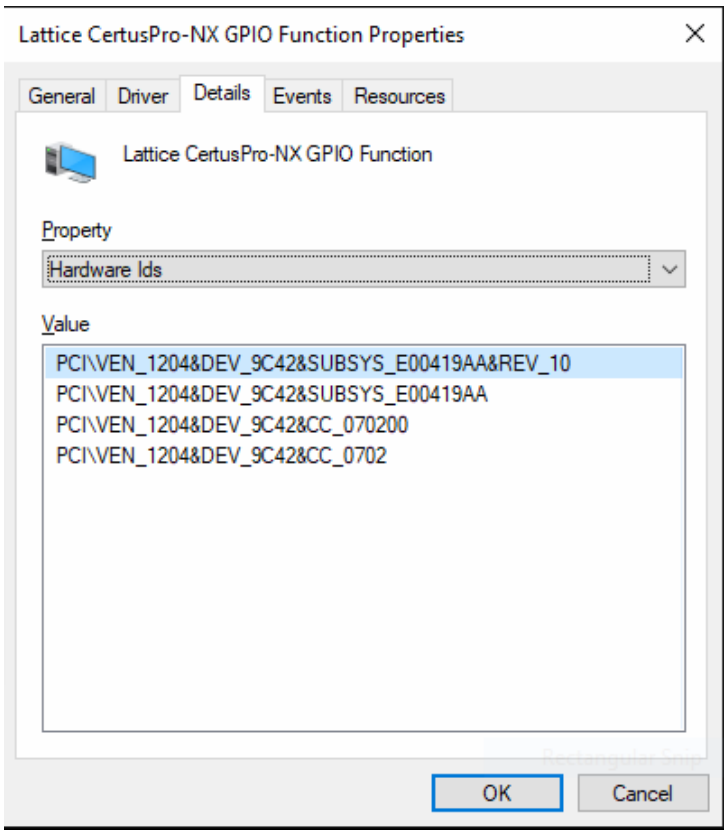

Figure 5.24. Hardware IDs of MachXO5-NX using the LFMXO5-65T GPIO Device

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

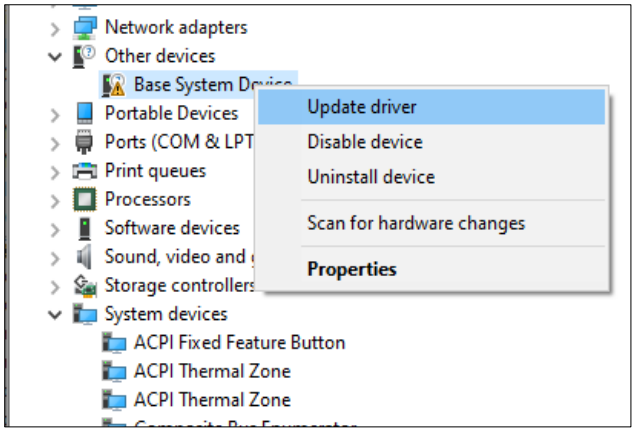

Figure 5.25. Update Driver Menu in Device Manager

5. Select the **Browse my computer for driver software** option as shown in Figure 5.26.

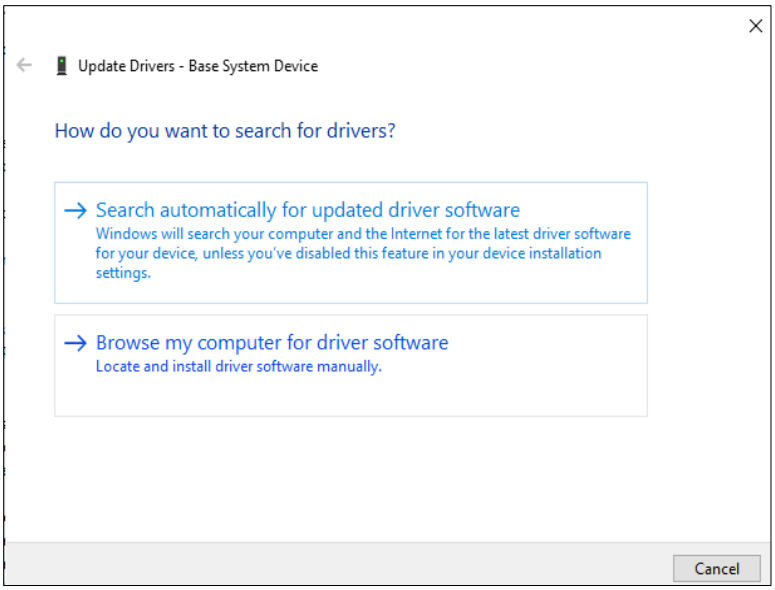

Figure 5.26. Update Driver Options

6. Browse for I2C or the GPIO driver.

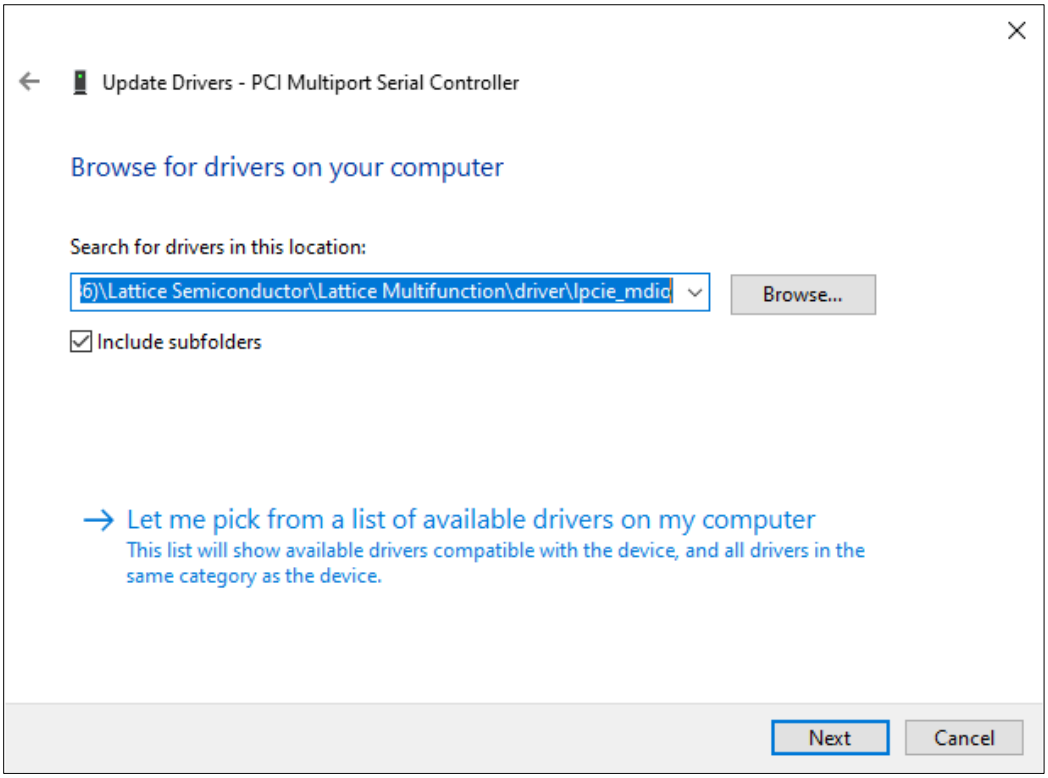

Figure 5.27. Browse the Driver for Device

7. If you receive a Windows Security prompt, select **Install this driver software anyway** as shown in Figure 5.28.

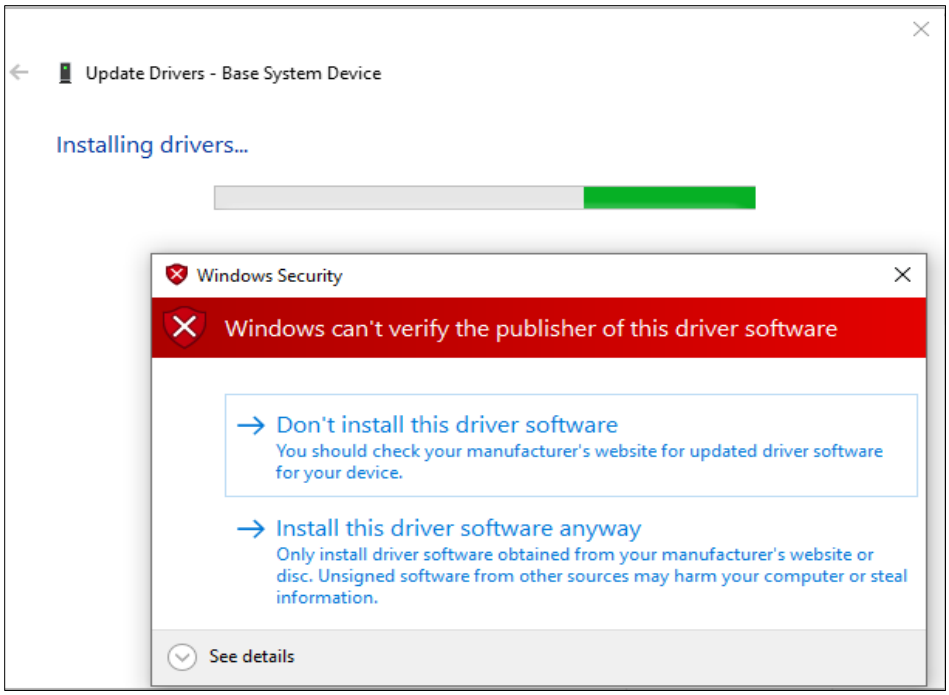

Figure 5.28. Windows Security in Device Manager

8. If the driver is installed successfully, a message is displayed as shown in Figure 5.29.

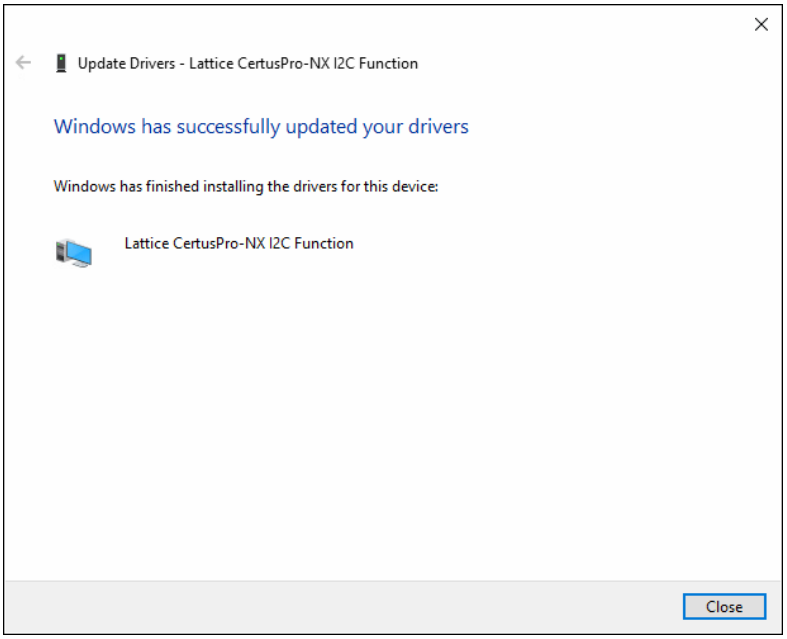

Figure 5.29. I2C Driver Installation Status Message for MachXO5-NX using LFMXO5-65T Device

9. Follow steps 3 through 8 for the GPIO devices. After successful installation, all two devices must be displayed in the Device Manager as shown in Figure 5.30.

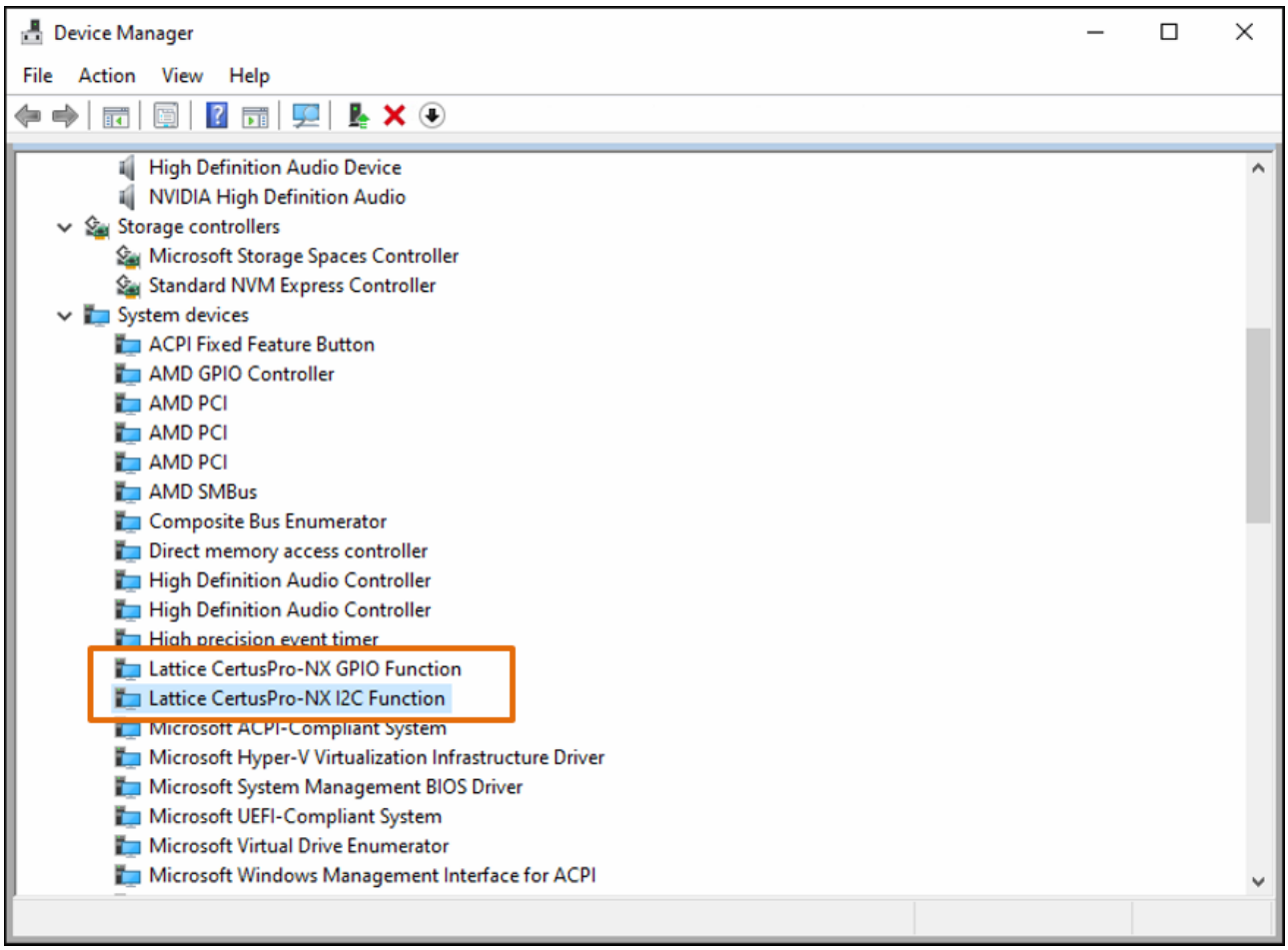

Figure 5.30. I2C and GPIO Device Drivers for MachXO5-NX using LFMXO5-65T Device in Device Manager

5.2.1.4. Driver Uninstall Manually

To uninstall the device driver, perform the following steps:

1. Open **Device Manager** window. Look for the device you want to uninstall.
2. Right-click on the device driver and select **Uninstall device** as shown in Figure 5.31.

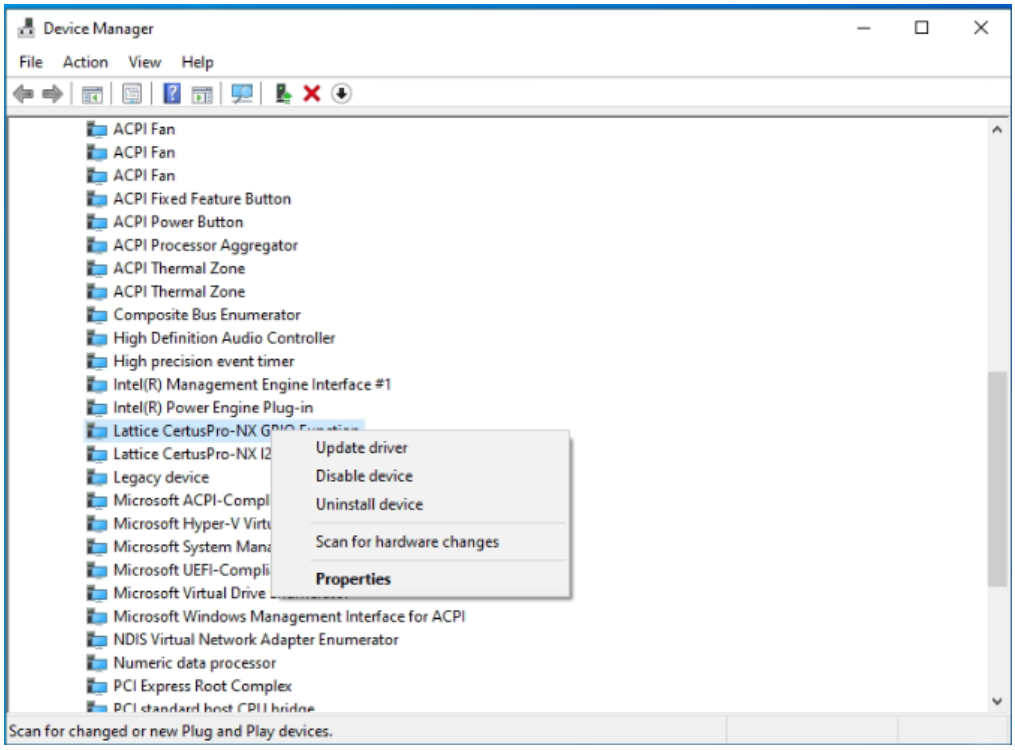

Figure 5.31. Device Manager Window

3. Check the option **Delete the driver software for this device** and click on **Uninstall** as shown in Figure 5.32.

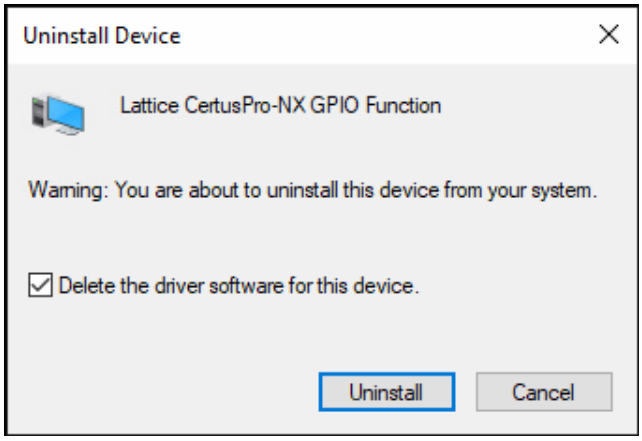

Figure 5.32. Uninstall Device Window

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

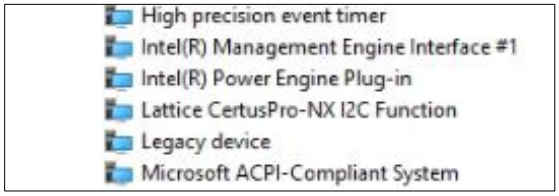

Figure 5.33. Uninstalled Device

5.2.2. Software Setup for Linux

5.2.2.1. Supported Operating System

- Distribution: Ubuntu
- Description: Ubuntu 24.04.2 LTS
- Release: 24.04.2

5.2.2.2. Required Packages

To check whether the required packages are installed or not, run the following commands on a terminal as shown below.

```
make -v
```

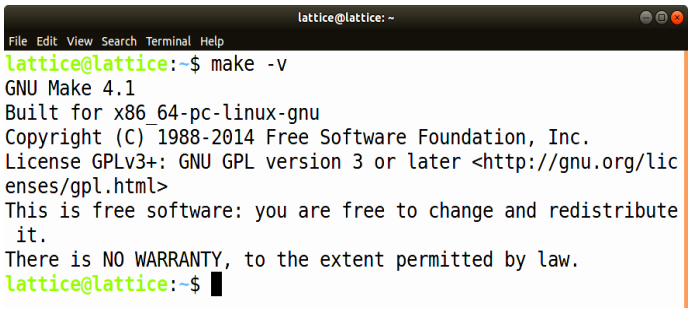

Figure 5.34. Display Make Utility in Linux Terminal

```
gcc -v
```

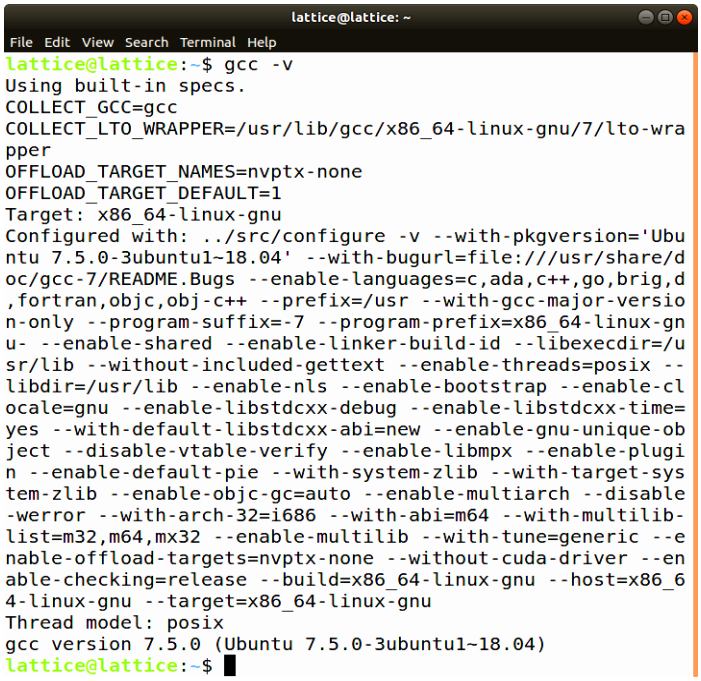

Figure 5.35. Display Compiler Version in Linux Terminal

```
g++ -v
```

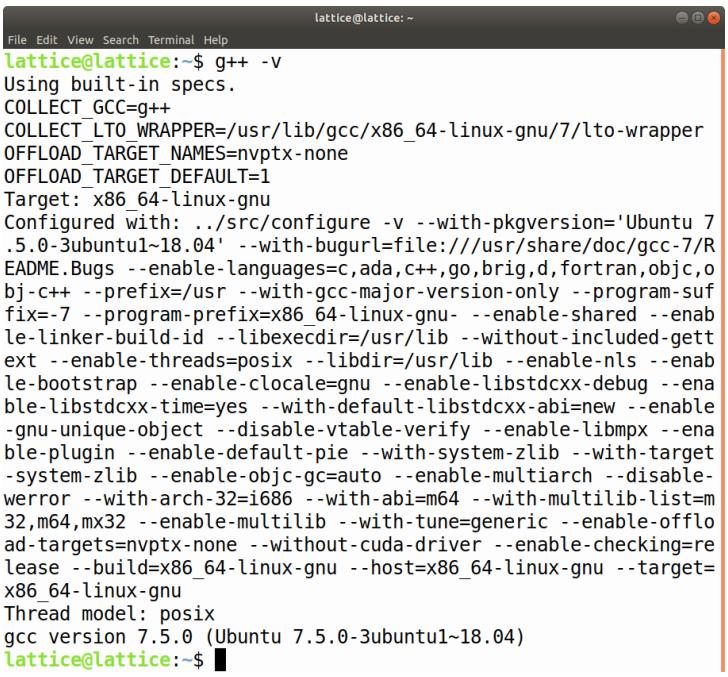

Figure 5.36. Update Compiler Version in Linux Terminal

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 set up the demo in automatic mode:

1. Go to the *Demonstration/Linux* directory.
2. Change the permission of the *install.sh* file by running chmod command.
   ```
   sudo chmod 777 install.sh
   ```
3. Run the command below, which builds the driver and API library, install the driver, and launch 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: You may skip this section if you 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 API library.
5. Make sure that the driver is not installed.
6. To remove an existing driver, run one or all the following commands in the terminal window:
```
sudo rmmod gpio_main.ko
sudo rmmod i2c_main.ko
```

To install the drivers:

1. Go to *drv_src/gpio_drv/* directory.
2. Run the command below.
```
sudo insmod gpio_main.ko
```
Repeat the same procedure for the *i2c_drv*.
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 PCIe Multifunction Application is used to demonstrate the multifunction capabilities of the FPGA PCIe IP. The application software allows you to access GPIO and I^2C registers for PCIe and 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 Mixed Mode Demo Application

On Linux, the demo application can be launched by running the *./MFDemo.sh script*, which is in the *Demonstration/Linux/Source_Code/app_src/gui/deploy* directory. On Windows, the application can be launched by clicking Lattice Multifunction shortcut present 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. Note that some of the information may vary between different FPGA devices (such as the Device ID).

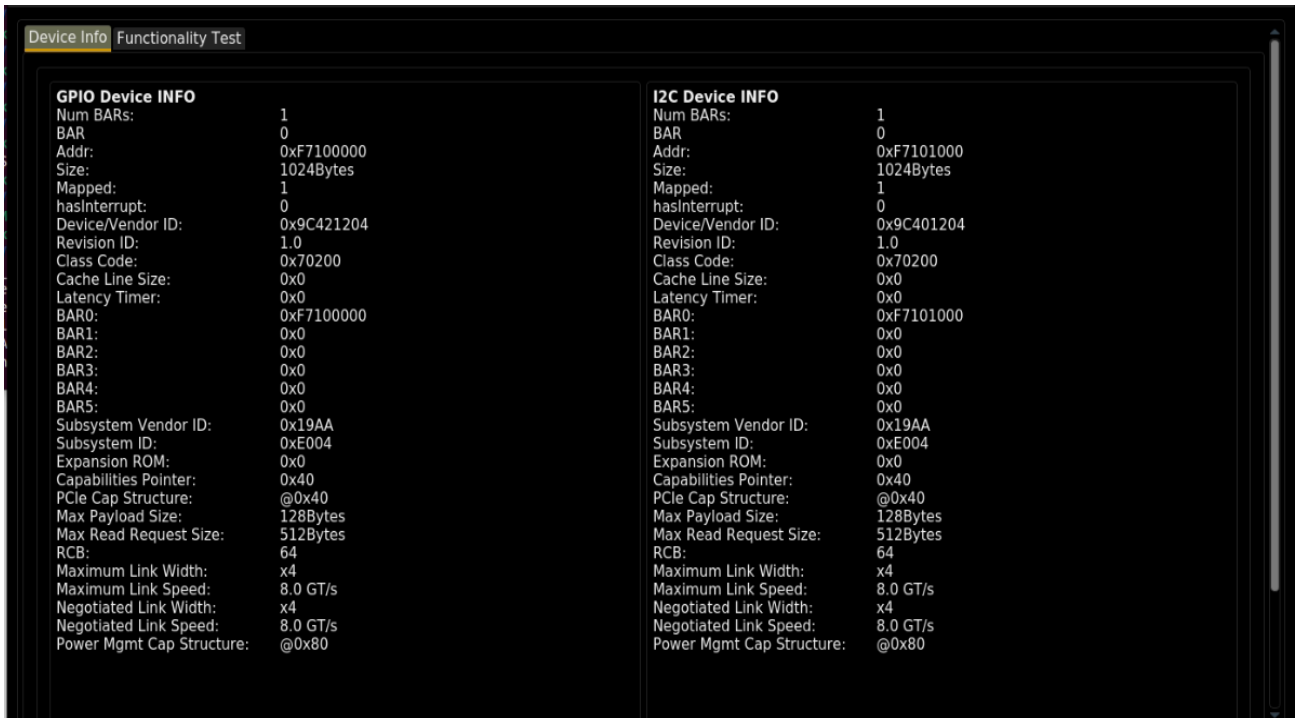

Figure 6.1. PCIe Test Application Device Info Tab

6.2. Using the Multifunction Demo Application User Interface

The PCI Express Device Info page displays information about the device driver and device's 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

You can read or write from or on the device using controls present on this tab. Figure 6.2 shows the Functionality Test tab.

Figure 6.2. Functionality Test Tab

6.2.2. GPIO

This section is used 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 you change the position of the input switches on the board. This is mapped to SW1 on the board and there is one LED per input switch. To start reading input from the switches, click the *Read Input* button. An example of LEDs is shown in Figure 6.3.

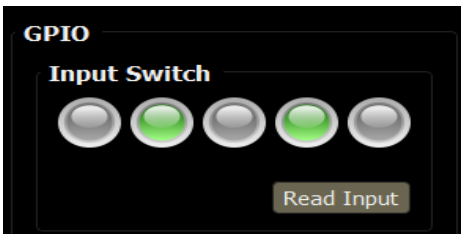

Figure 6.3. GPIO Input Switch

6.2.2.2. Output LED

The Output LED section shows the status of eight output LEDs. These LEDs represent the five bits of a counter, which begin to increment when you click the “Run Counter” button on the user interface, as shown in Figure 6.4.

Figure 6.4. GPIO Output LEDs

Run Counter

The *Run Counter* switches the Demo LED within the MachXO5-NX using LFMXO5-65T Development board to be turned ON and OFF following the GPIO Output LED as shown in Figure 6.5. 7 out of 8 (LED_0 till LED_6) of the Demo LED follows the GPIO Output LED.

Figure 6.5. Demo LEDs

6.2.3. I2C

This section describes how to use the user interface to read and write 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, you can 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 6.6. After selecting the desired speed from the drop-down box, you must click the *Set* button to apply the settings to the device.

Figure 6.6. I2C Bit-Rate Selection

6.2.3.3. Controller Write

The Controller write option is used to write data to the I2C device. You can either manually type the address and/or the data information into the Message box, or you can load the data from a hex file by clicking on the Load button. Clicking on the *Controller Write* button initiates the write operation interface, as shown in Figure 6.7.

You 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. For MachXO5-NX using LFMXO5-65T development board, the camera module must be obtained separately.

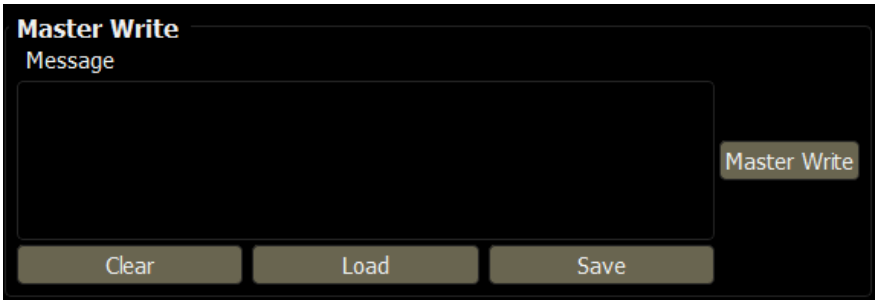

Figure 6.7. I2C Controller Write

Table 6.1. Controller 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
Controller Write	Writes the input box data on the I2C device

IMX258-0AQH5 Camera Write Protocol

Figure 6.8 shows the communication protocol format for the IMX258 camera for a single write to an arbitrary address. Figure 6.9 shows the format for a sequential write starting from an arbitrary address.

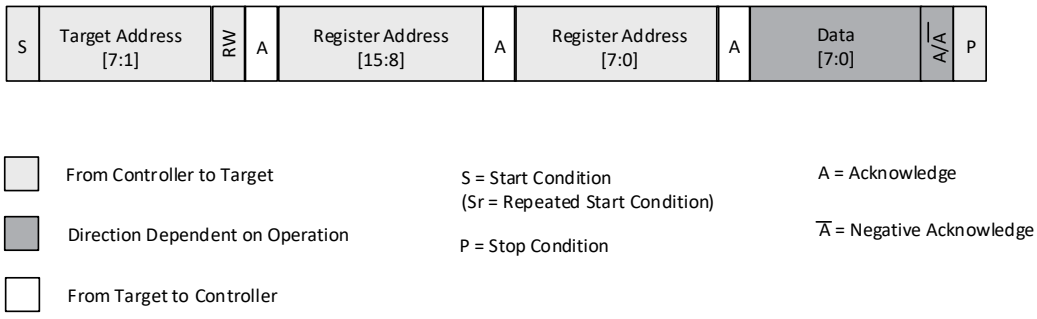

Figure 6.8. Single Write to an Arbitrary Address of IMX258-0AQH5 Camera

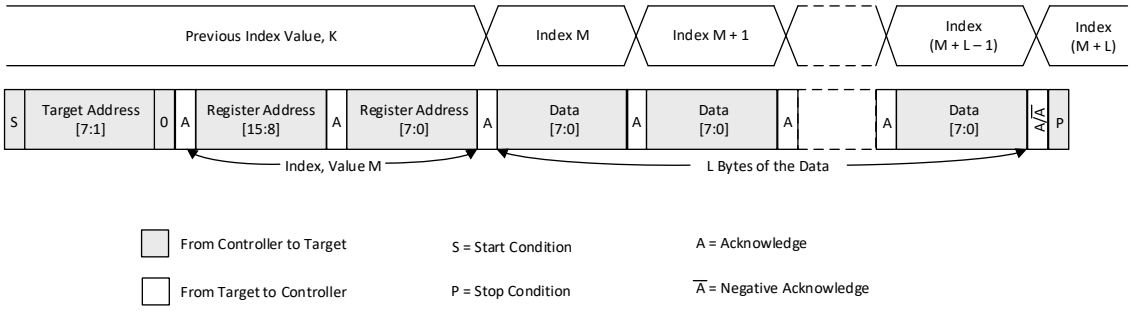

Figure 6.9. Sequential Write Starting from an Arbitrary Address of IMX258-0AQH5 Camera

To test the I2C interface, data can be written to the register 0x3093 of the camera, as shown in Figure 6.10. Additional register information can be obtained through the IMX258-0AQH5 camera data sheet available through 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]	

Controller 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 **Target 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 6.10. 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 **Controller Write** button. To verify that the write operation is successful, follow the instructions on how to perform a Controller Read operation.

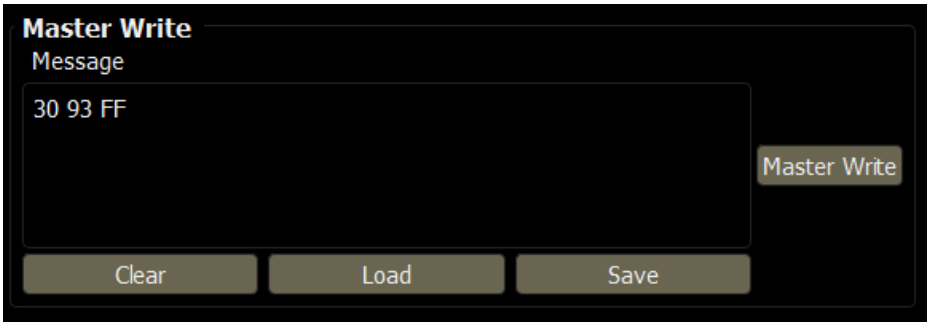

Figure 6.10. I2C Controller Write Procedure

6.2.3.4. Controller Read

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

Controller Read from a Held Address

The Controller Read option is used to read data from the held address of the I2C device. When the controller transmits the target 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 6.11. I²C Controller Read

IMX258-0AQH5 Camera Read Protocol from a Held Address

Figure 6.12Figure 6.12 shows the communication protocol format for the IMX258 camera for a single read from the held address. Figure 6.13 shows the format for a sequential read starting from the address held.

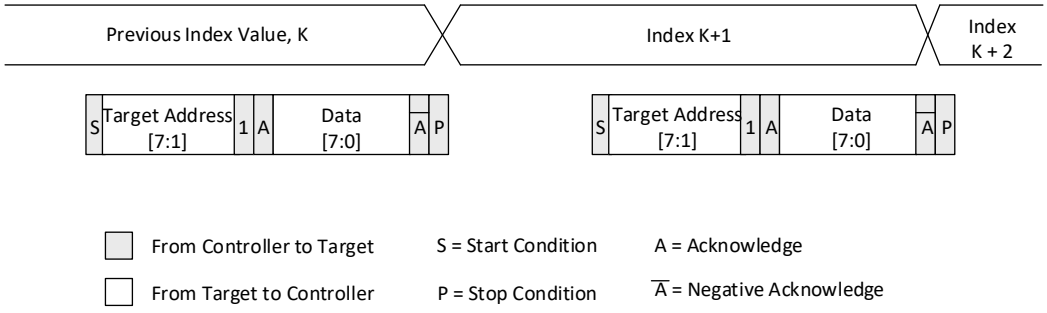

Figure 6.12. Single Read from the Held Address of IMX258-0AQH5 Camera

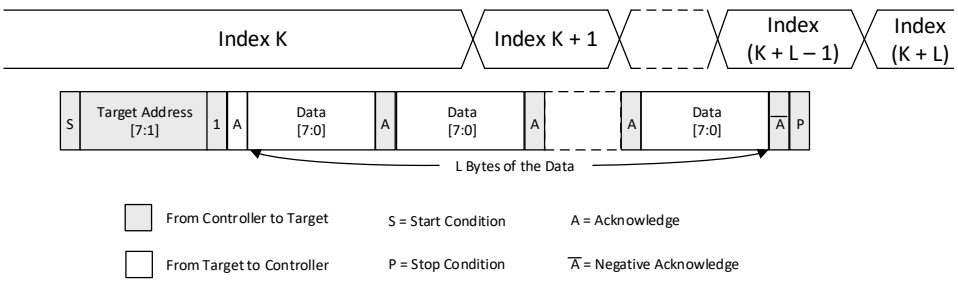

Figure 6.13. Sequential Read Starting from the Held Address of IMX258-0AQH5 Camera

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

To perform a Controller 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 **Target addr** box.
4. Select the **7-bit mode** radio button.
5. Enter the register address in the Controller Write message box as shown in Figure 6.14 to set the current register address. For example, to read from 0x3093 register address, enter **30 93** in the message box and click the **Controller Write** button.

Figure 6.14. I2C Controller 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 6.15 and click **Controller 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 6.15. I2C Controller Read

Controller Register Read from an Arbitrary Address

The Controller 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 6.16. I2C Controller Register Read

Table 6.3. Controller 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 I^2C device.
Register Read	When this button is pressed, the read operation is performed.

IMX258-0AQH5 Camera Read Protocol from an Arbitrary Address

Figure 6.17 shows the communication protocol format single read from an arbitrary address and Figure 6.18 shows the sequential read starting from an arbitrary address of MX258-0AQH5 Camera.

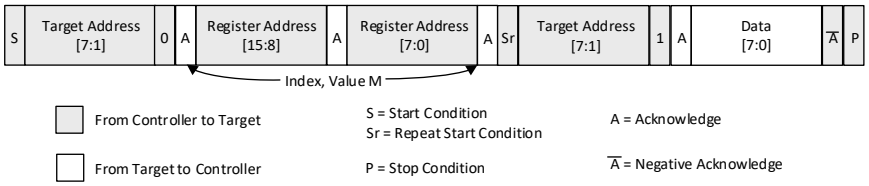

Figure 6.17. Single Read from an Arbitrary Address of MX258-0AQH5 Camera

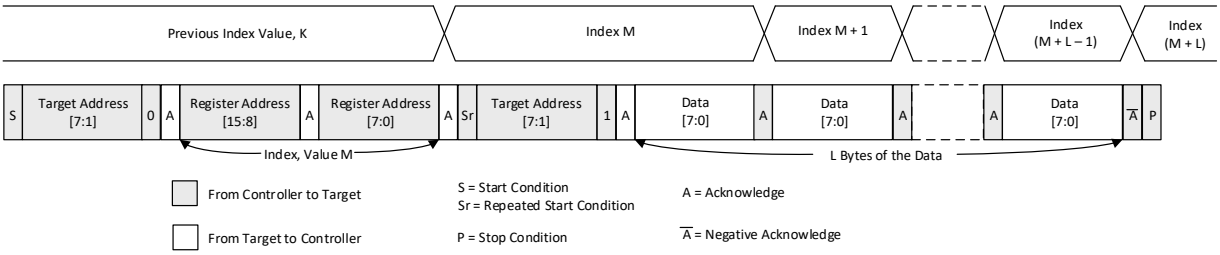

Figure 6.18. Sequential Read Starting from an Arbitrary Address of MX258-0AQH5 Camera

To perform a Controller 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 **Target 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 6.19. I2C Controller Register Read

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

6.2.3.5. Transaction Log

The Transaction Log shows the details of all transactions on the I2C device as shown in Figure 6.20. You 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.

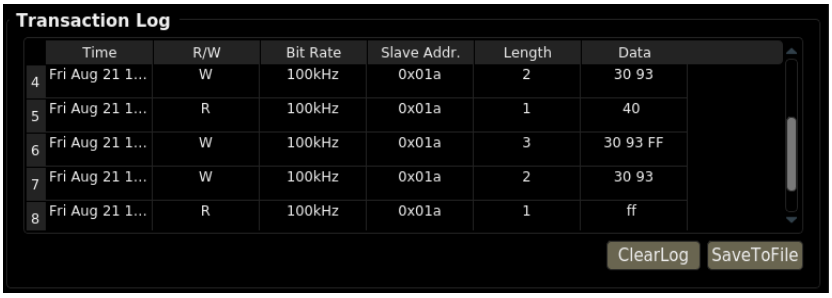

Figure 6.20. I2C Controller Register Read

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.
Target Addr	Specifies the I2C target address of the transaction.
Length	Specifies the number of bytes transferred during the transaction.
Data	Specifies the transferred data during the transaction.

6.2.3.6. Batch Mode

Batch mode is used to read and write the I2C device using an XML file. You 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 6.21.

Figure 6.21. I2C Batch Mode Read/Write

The following are the descriptions of the XML node:

- i2c_bitrate – elements used to configure the I2C device bit-rate.
- kHz – attributes hold the I2C bit rate value.
- i2c_write – elements used to write address and data or address-only.
- addr – attribute that holds the I2C target device address.
- addrmod – attribute that specifies the addressing mode, with a value of 0 for 7-bit mode and 1 for 10-bit mode.
- 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 Controller device status when any operation is run.
Execute	Used to start batch file execution.
Stop	Used to stop the execution of the batch file.

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.

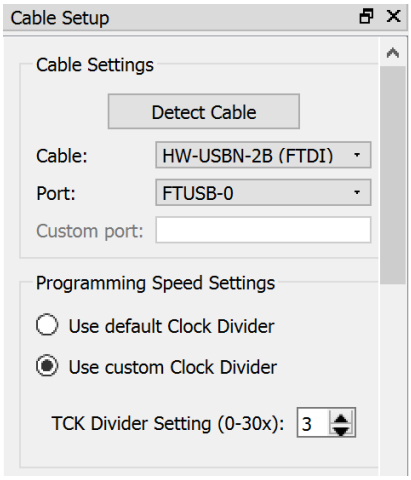

Figure 7.1. TCK Frequency Setting

- 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.

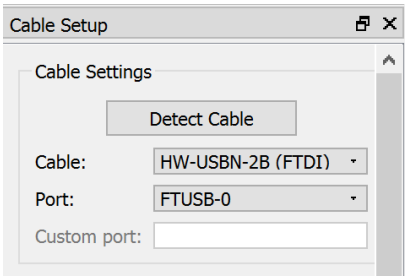

Figure 7.2. Port Selection

7.2. Installing the Driver and User Interface Launch for Windows

7.2.1. Problem with Installing the Driver

- 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 the 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.

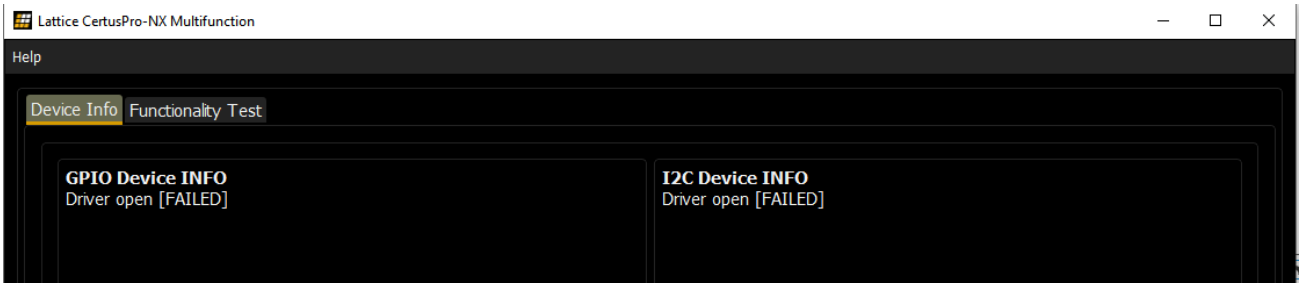

Figure 7.3. User Interface with No Device Driver

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, ensure that you 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. Installing the Driver User Interface Launch for Linux

This section describes troubleshooting steps for Linux.

7.3.1. Problem with Installing the Driver

The following error may occur during the 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 your system. Refer to your 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 with Loading the Driver

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.

```
Device: 09:00.0
Class:  0702
Vendor: 1204
Device: 9c40
SVendor:        19aa
SDevice:        e004
Rev:    10

Device: 09:00.1
Class:  0702
Vendor: 1204
Device: 9c42
SVendor:        19aa
SDevice:        e004
Rev:    10
```

Figure 7.4. lspci -vnm for MachXO5-NX using LFMXO5-65T Device 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 7.5. Check the permissions of these files.

```
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 7.5. Content List of Demonstration/Linux Directory

- The script files must 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 7.6.

```
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 7.6. 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
```

References

- MachXO5-NX web page
- Lattice Radiant Software FPGA web page
- Lattice Insights 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, please refer to the Lattice Answer Database at www.latticesemi.com/Support/AnswerDatabase.

Revision History

Revision 1.0, August 2025

Section	Change Summary
All	Initial release.

www.latticesemi.com