Lattice mVision MIPI DSI to DisplayPort Demo

User Guide

FPGA-UG-02224-1.1

March 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
AUX	Auxiliary Channel
AXI	Advanced Extensible Interface
DP	DisplayPort
DSI	Display Serial Interface
FFC	Flat Flexible Cable
FMC	FPGA Mezzanine Card
FPC	Flexible Printed Circuit
HPD	Hot Plug Detect
MIPI	Mobile Industry Processor Interface
MPCS	Multiple-Protocol Physical Coding Sublayer
OS	Operating System
PCIe	Peripheral Component Interconnect Express
RX	Receiver
SPI	Serial Peripheral Interface
TX	Transmitter

1. Introduction

The CertusPro™-NX Mobile Industry Processor Interface (MIPI®) Display Serial Interface (DSI) to DisplayPort (DP) bridge design features a MIPI D-PHY receiver front-end configuration with four lanes. The bridge decodes MIPI DSI 24 bpp RGB888 packets and converts the formatted video data stream to a DisplayPort interface supporting up to four lanes at 5.4 Gbps.

2. Hardware and Software Requirements

2.1. Hardware Requirements

- Raspberry Pi 5 board
- CertusPro-NX Peripheral Component Interconnect Express (PCIe®) Bridge Board RevC
- Lattice™ Modular FPGA mezzanine card (FMC) adapter
- Lattice DisplayPort transmitter daughter card
- 4K monitor with DisplayPort input
- Mini USB Type A cable for programming the bitstream
- DisplayPort cable
- 12 V power supply
- Raspberry Pi 5 22-pin 0.5 mm-pitch FFC/FPC cable

2.2. Software Requirements

- The Lattice Radiant™ Programmer software version 2023.2 or later
- Raspberry Pi operating system (OS) (Debian version 12 – Bookworm, Kernel version 6.6.41 or later)

To download the demo files, refer to the Lattice mVision MIPI DSI to DisplayPort Demonstration web page.

3. Demo Design Overview

3.1. Theory of Operation

In this demo, the Raspberry Pi 5 board serves as the video source, sending the 1080p60 sync pulse mode video signal to MIPI DSI D-PHY receiver (RX). This data is processed through a Byte-to-Pixel Converter, which translates the incoming byte stream into pixel data in the pixel domain based on the DSI synchronization packets. The pixel data is fed into a Video Scaler, which adjusts the resolution and size of image to meet the output requirements. The scaled image is transmitted through the DP transmitter (TX) to a monitor, where the final output is displayed. This demo is targeted to upscale RGB888 8 bits per colour (bpc) video resolution from 1080p60 to 4K60.

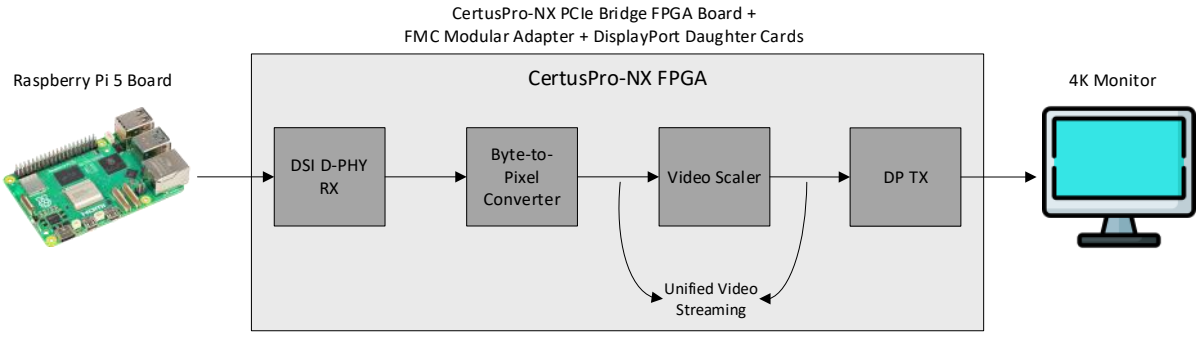

Figure 3.1. Top-Level Architecture of the System Design

3.1.1. MIPI DSI D-PHY RX

In this demo, DSI interface is configured to act as a physical layer interface to connect to the Raspberry Pi 5 board to receive the 1080p60 sync pulse mode video signal.

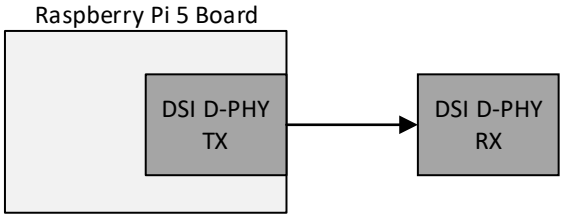

Figure 3.2. MIPI DSI D-PHY IP Block Diagram

3.1.2. Byte-to-Pixel Converter

In this demo, the Byte-to-Pixel Converter IP converts the DSI standard-based video payload packets from MIPI D-PHY Receiver Module and output to pixel format. the Byte-to-Pixel Converter IP also generates video control signals in the pixel domain based on the DSI synchronization packets.

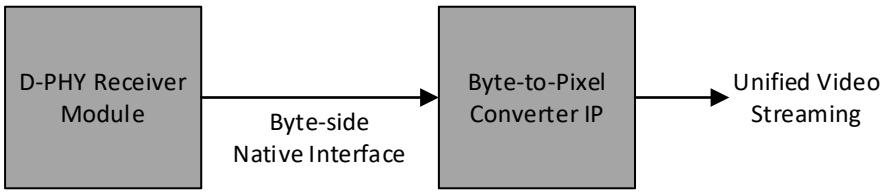

Figure 3.3. Byte-to-Pixel Converter IP Block Diagram

3.1.3. Video Scaler

In this demo, the Video Scaler IP is used to upscale the resolution of a video stream, mapping input video frames to output frames of a different size. The input and output are configured to 1080p60 and 4K60 resolution respectively. This IP is configured to use a unified-streaming interface for both input and output of video data streams. This IP supports two types of scaling algorithm: bilinear and Lanczos.

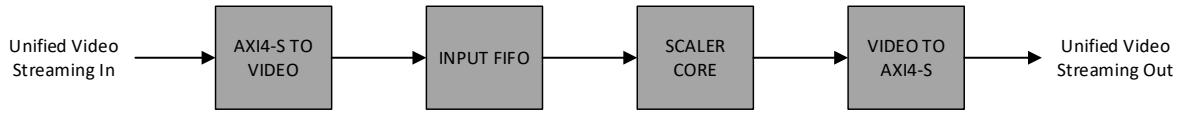

Figure 3.4. Video Scaler IP Block Diagram

3.1.3.1. Bilinear Scaling Algorithm

Bilinear algorithm considers the closest 2×2 neighborhood of known pixel values surrounding the unknown pixel. The algorithm then takes a weighted average of these 4 pixels to arrive at the final interpolated value.

3.1.3.2. Lanczos Scaling Algorithm

Lanczos algorithm uses sinc function for interpolation and variable filter taps ranging from 4 to 12 for both horizontal and vertical filters.

3.1.4. DisplayPort Transmitter

This demo focuses only on the DisplayPort transmitter as described in the following subsections.

3.1.4.1. Configuration

The DisplayPort transmitter can start automatically on power up.

3.1.4.2. Training

When the DisplayPort transmitter is active, the transmitter waits for Hot Plug Detect (HPD) line to go high and starts the training process using the auxiliary channel (AUX) interface. The training sequence includes clock recovery, channel equalization, and symbol lock. Training is initiated by AUX channel, but the training data (TPS1, TPS2, TPS3, or TPS4) is transmitted over the main channel.

3.1.4.3. Normal Operation

After the training is complete, the DisplayPort transmitter starts receiving video data from the Unified Video Streaming interface. In the protocol layer, the pixel data is mapped into lanes based on lane count and colorimetry. The pixel data, along with control signals, is packed according to video timing signals and passed to the DisplayPort packet generator module for packet generation. The data is then passed to the scrambler and subsequently to the Lane Skew module. Finally, the lane skewed data is serialized by the Multiple-Protocol Physical Coding Sublayer (MPCS) module.

4. Setting Up the Demo

4.1. Hardware Setup

This section covers the hardware setup, the steps to program the demo into the serial peripheral interface (SPI) memory of the FPGA board, and the initialization of the Raspberry Pi 5 board.

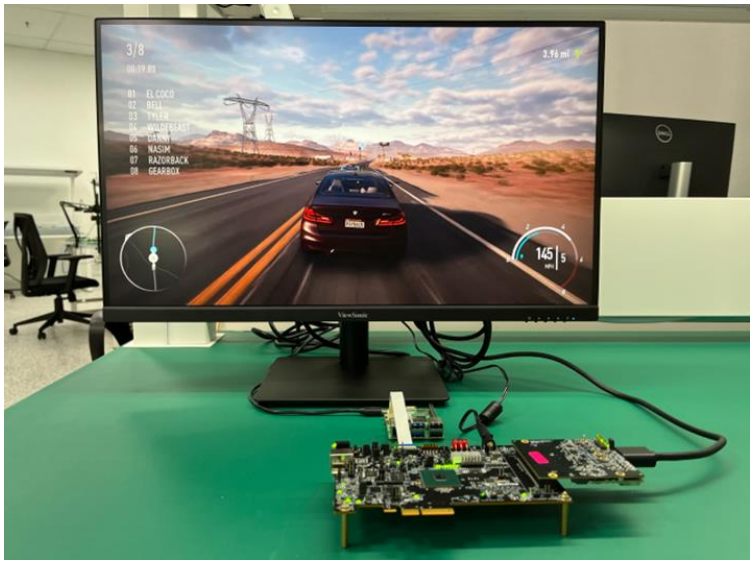

Figure 4.1. MIPI DSI-DP Hardware Setup Overview

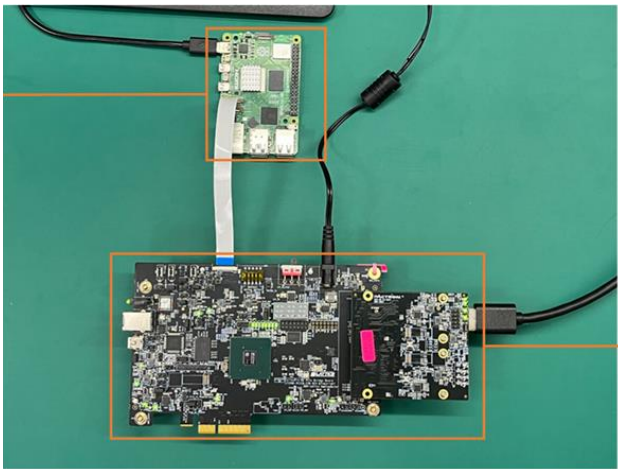

Figure 4.2. MIPI DSI-DP Hardware Setup Top View

4.1.1. Raspberry Pi 5 Board

The Raspberry Pi 5 board can be powered using the USB-C power supply. The DSI ports (CAM/DISP 1) on the Raspberry Pi 5 board and the CertusPro-NX PCIe Bridge Board are connected using the 22-pin 0.5 mm-pitch FFC/FPC cable.

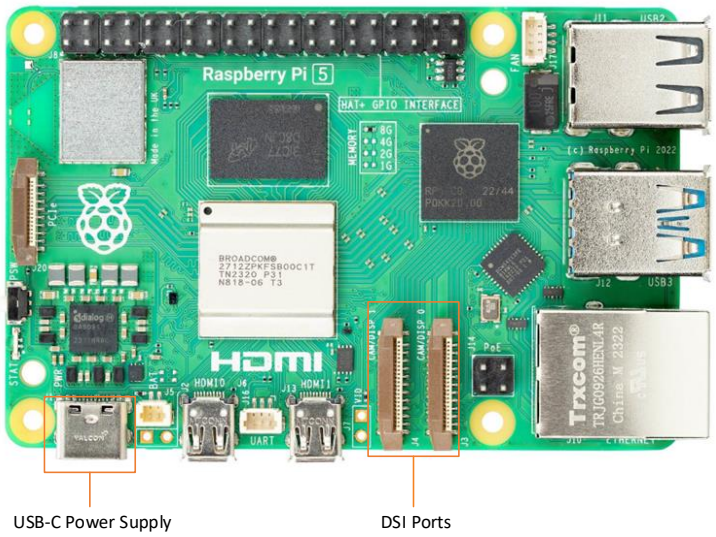

Figure 4.3. Raspberry Pi 5 Board

4.1.2. CertusPro-NX PCIe Bridge Board RevC

With the default jumper configuration, ensure J15 connector is opened. The external 12 V power is provided, and the 12 V source switch receives power from the external 12 V power connection. The laptop that runs the Lattice Radiant Programmer software is connected to the board using the mini USB Type A cable.

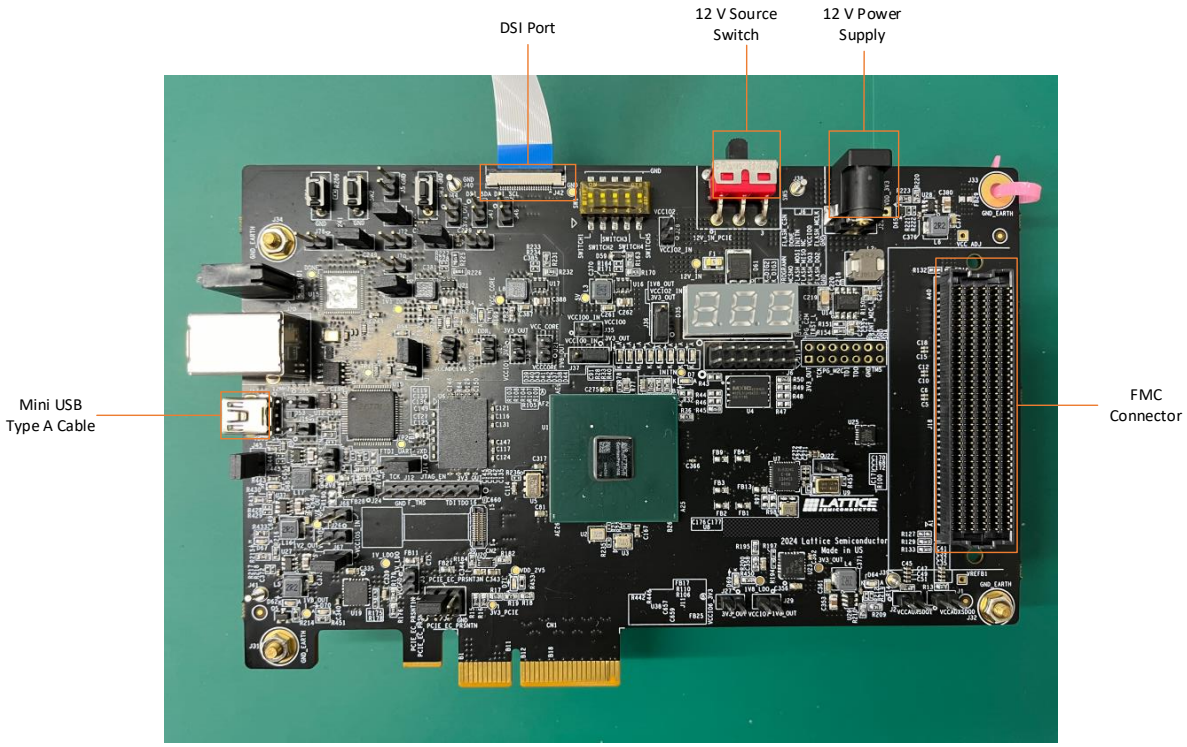

Figure 4.4. CertusPro-NX PCIe Bridge Board RevC

4.1.3. Modular FMC Adapter and Daughter Cards

The modular FMC adapter power is provided from the FMC connector (J1). The DisplayPort transmitter daughter card connects to the modular FMC adapter through the J3 connector.

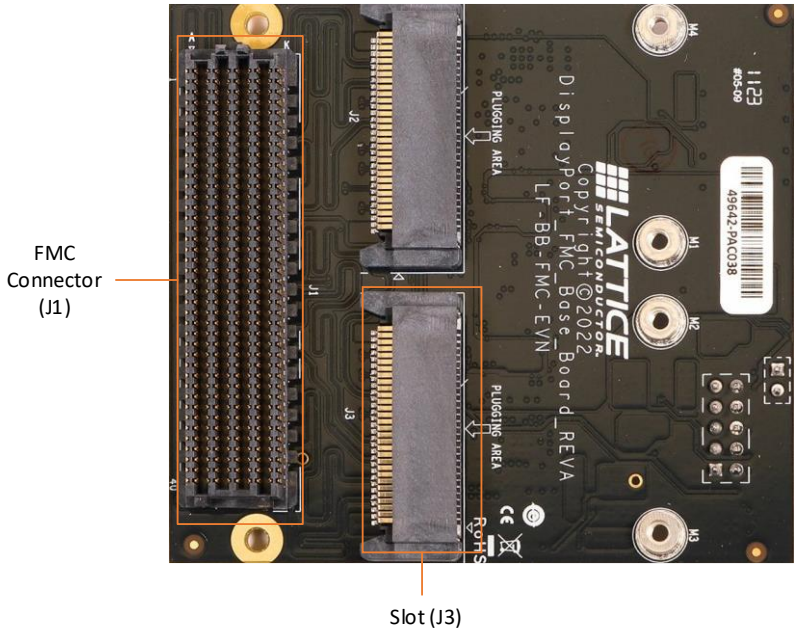

Figure 4.5. Modular FMC Adapter

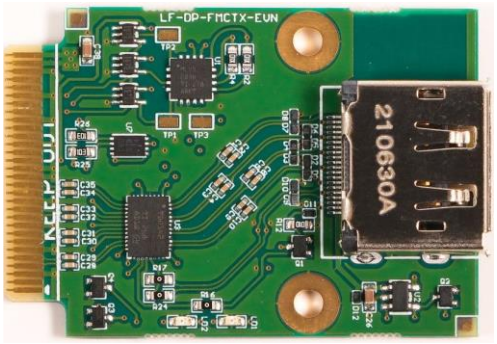

Figure 4.6. DisplayPort Transmitter Daughter Card

4.1.4. 4K Monitor

The output from DisplayPort transmitter daughter card is connected to the DELL™ U2723QE 4K monitor with DisplayPort 1.4 using the DisplayPort 1.4 cable.

Figure 4.7. 4K Monitor

4.2. Software Setup for the Raspberry Pi 5 Board

This section provides the procedure for enabling and configuring DSI output on the host machine.

4.2.1. Preparing the Raspberry Pi 5 Board

1. Ensure the Raspberry Pi 5 board is installed with the Raspberry Pi OS (Debian version 12 – Bookworm, Kernel version 6.6.41).
2. Copy the *vc4-kms-dsi-lattice-demo.dtbo* file in the .zip file provided to */boot/firmware/overlays/* in the Raspberry Pi 5 board in super user mode using the following command:
   ```
   sudo cp <path_to_ vc4-kms-dsi-lattice-demo.dtbo> /boot/firmware/overlays/
   ```

4.2.2. Enabling DSI Output for the Raspberry Pi 5 Board

1. Open */boot/firmware/config.txt* as super user in the Raspberry Pi 5 board for editing.
2. After this command:
   ```
   dtoverlay=vc4-kms-v3d
   ```
   Add the following command:
   ```
   dtoverlay=vc4-kms-dsi-lattice-demo
   ```

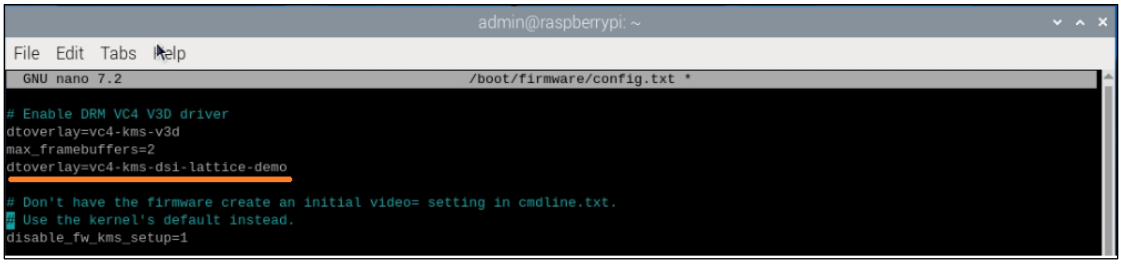

Figure 4.8. DSI Output Enabling Message

- By default, the added command enables the Raspberry Pi 5 board to output DSI signal with following settings:
  - Port: DSI1
  - Pixel clock-frequency: 148500000
  - Horizontal active (hactive):1920
  - Horizontal front-porch (hfp): 88
  - Horizontal sync-len (hsync): 44
  - Horizontal back-porch (hbp): 148
  - Vertical active (vactive): 1080
  - Vertical front-porch (vfp): 4
  - Vertical sync-len (vsync): 5

- Vertical back-porch (vbp): 36
- Color-format: rgb888
- Lane: four-lane
- Video mode: low_power_sync_pulse_mode

3. Reboot the Raspberry Pi 5 board.

4.2.3. Customizing DSI Output for the Raspberry Pi 5 Board

1. To customize different DSI output, open */boot/firmware/config.txt* as super user in the Raspberry Pi 5 board for editing.
2. To override the default setting, add the following parameters as required after this command:
   `dtoverlay=vc4-kms-dsi-lattice-demo`

Table 4.1. Parameters to Customize DSI Output for the Raspberry Pi 5 Board

Wherever applicable, default values are in bold.

Parameter	Selectable Values	Description
DSI Port	dsi0 (DSI Port 0) **dsi1 (DSI Port 1)**	Connected the DSI port on the Raspberry Pi 5 board to the CertusPro-NX PCIe Bridge Board (Rev B). Example: dtparam=dsi0
DSI Timing Data	Calculated value according to the resolution wanted	Configure the value for these DSI timing data parameters: clock-frequency, hactive, hfp, hsync, hbp, vactive, vfp, vsync, and vbp. Example: Configure the Raspberry Pi 5 board DSI output to 1080p60. • dtparam=clock-frequency=148500000 • dtparam=hactive=1920 • dtparam=hfp=88 • dtparam=hsync=44 • dtparam=hbp=148 • dtparam=vactive=1080 • dtparam=vfp=4 • dtparam=vsync=5 • dtparam=vbp=36 **Note:** Currently the CertusPro-NX PCIe Bridge Board (Rev B) supports only 1080p60.
DSI Color Format	**rgb888 (RGB888)**	Color format of the DSI output. Example: dtparam=rgb888
DSI Lane	**four-lane**	DSI data lane to be used for the DSI connection. Example: dtparam=four-lane
DSI Video Mode	**low_power_sync_pulse_mode (Sync Pulse in Low Power Mode)**	Video mode of the DSI output. Example: dtparam=low_power_sync_pulse_mode

An example of overriding the DSI output parameters is shown in the figure below.

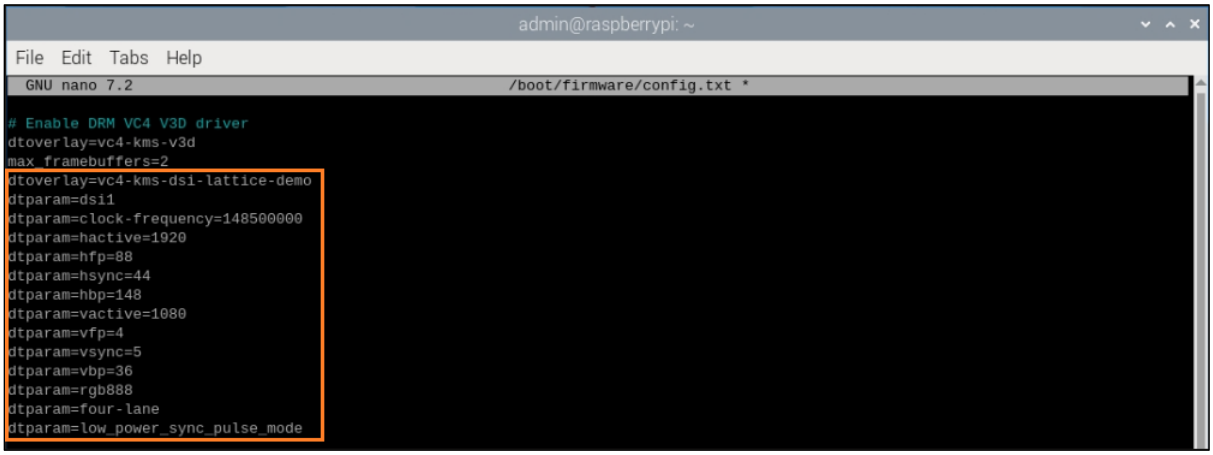

Figure 4.9. DSI Output Parameters Override Message

4.3. Programming the FPGA

To program the FPGA, follow these steps:

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 4.10.
2. Select **Create a new project from a scan**. The values are indicated in the **Cable**, **Port**, and **TCK Divider Setting (0-30×)** fields.
3. Click **OK**.

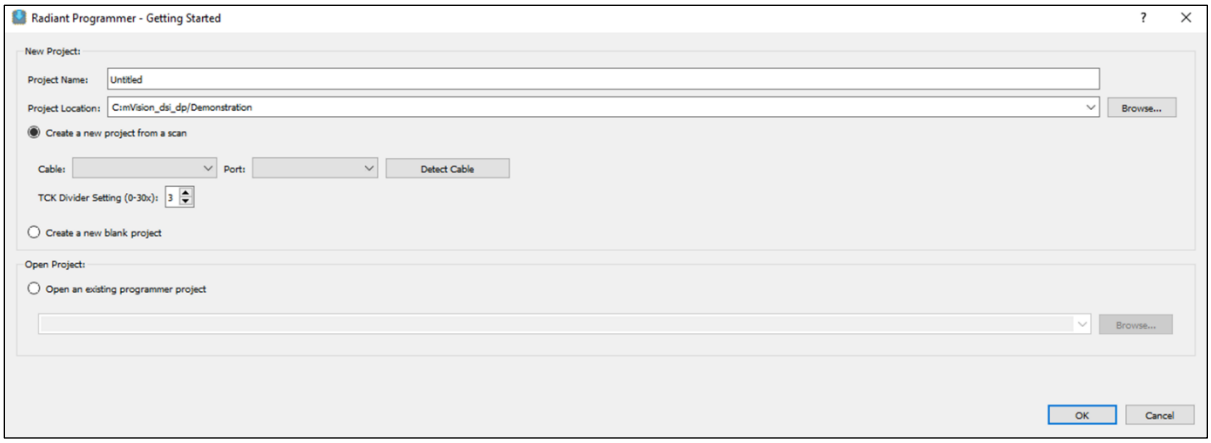

Figure 4.10. Creating a New Project from a Scan

4. The main interface opens as shown in the figure below.

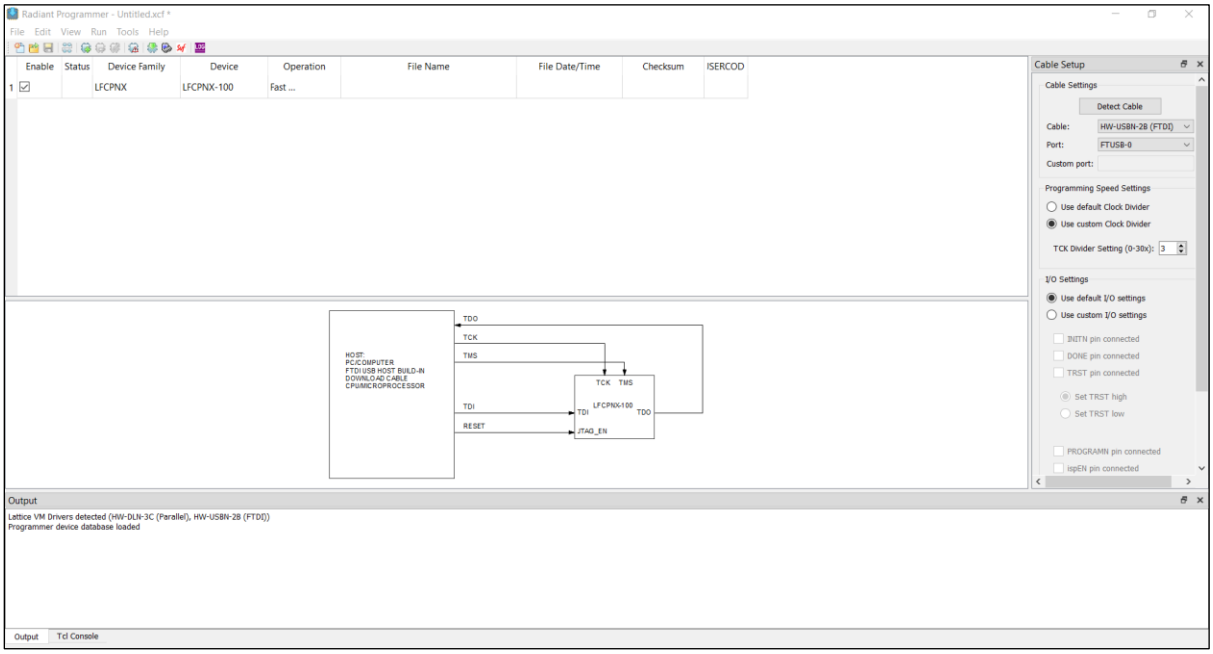

Figure 4.11. Lattice Radiant Programmer Window

5. If the Programmer settings do not match the settings shown in the figure below for CertusPro-NX devices, select these settings manually from the drop-down menu.

Figure 4.12. CertusPro-NX Device Settings

To select the programming settings, follow these steps:

1. Browse and select the Programming file, *dsi_dp_lanczos.bit* or *dsi_dp_bilinear.bit*.
2. Click **OK**.
3. Double-click **Operation** to open the **Device Properties** dialog box.

4. Select the settings as shown in the figure below.

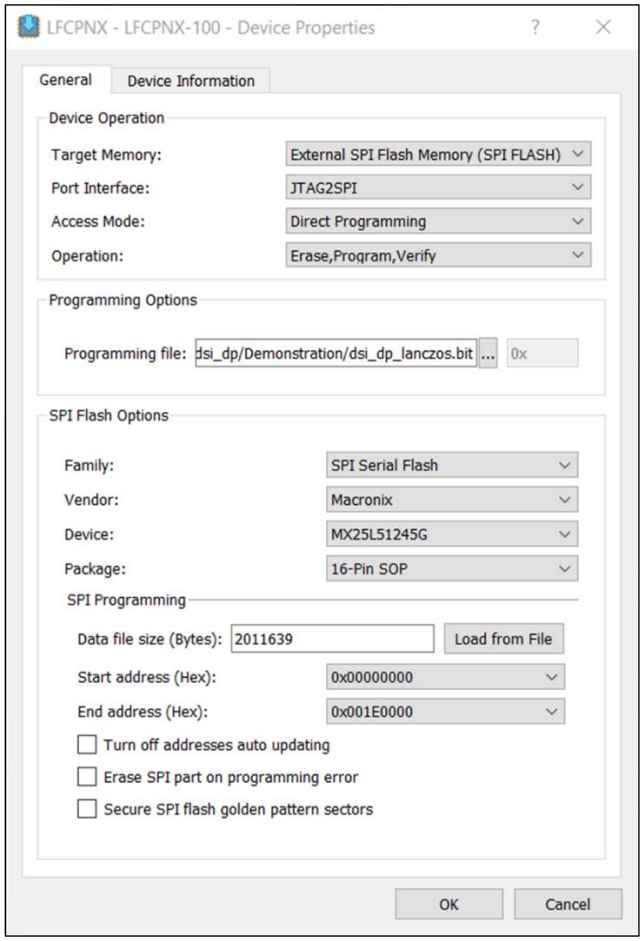

Figure 4.13. Device Properties Window for the CertusPro-NX SPI Flash Programming

5. Click **Programming** from the menu bar to start programming.

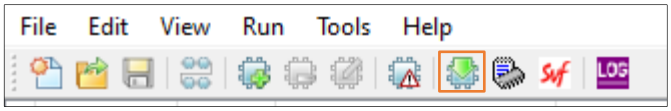

Figure 4.14. Programmer Menu Bar

When the FPGA programming is successful, the output console displays the *Operation: successful* message.

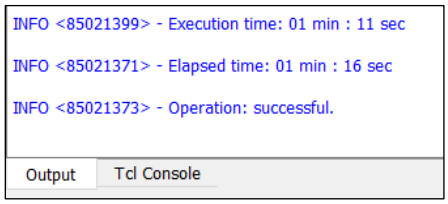

Figure 4.15. Programmer Output Window

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

6. After programming, check the status LEDs on the board and power cycle the board.

4.3.1. Status LED

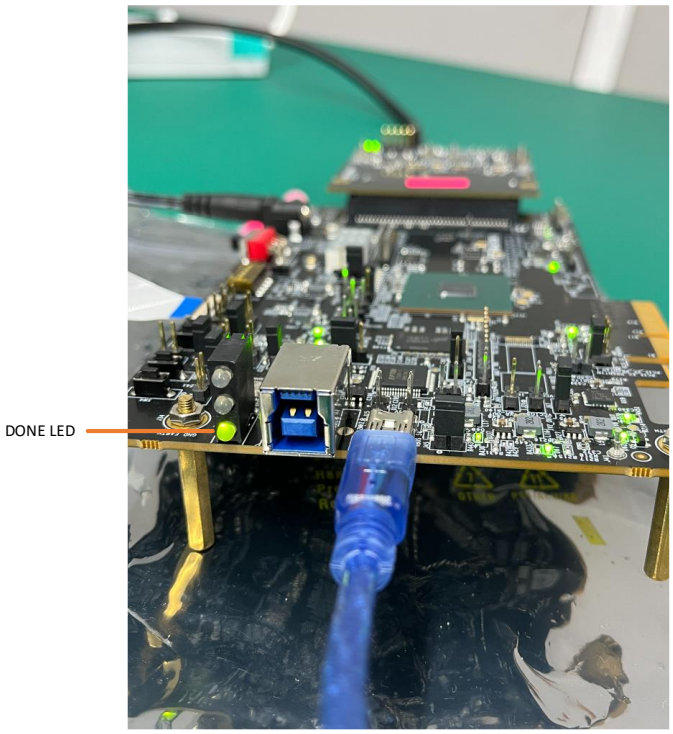

Figure 4.16. CertusPro-NX Status LEDs

Table 4.2. Description of the Status LED

Sl. No	Name	Description
1	DONE	Green – Lights if configuration is successful.

5. Troubleshooting

5.1. Error when Updating SPI Flash

- If there is verification error when programming the .bit file, try changing the TCK frequency by setting **TCK Divider Setting** to greater than 3. Restart programming by clicking the **Program** button.

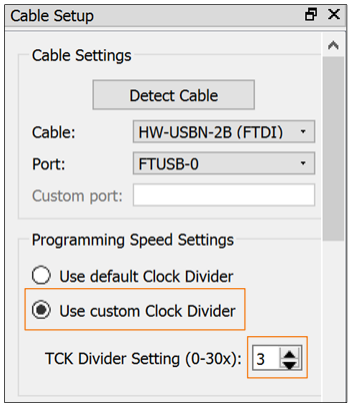

Figure 5.1. TCK Frequency Settings

If the verification error problem still occurs, press and hold the **PROGRAMN** push button before clicking the **Program** button.

- If the device is not detected on Port: **FTUSB-0**, click **Detect Cable** and select Port: **FTUSB-1**.

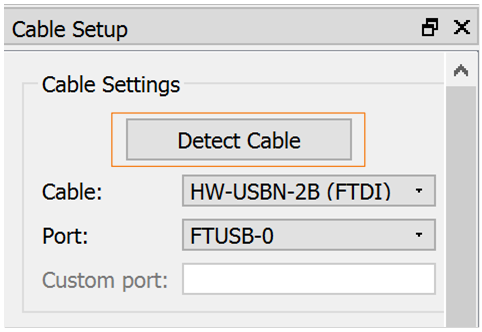

Figure 5.2. Port Selection

Appendix A. Resource Utilization

The table below shows a sample resource utilization of the CertusPro-NX MIPI DSI to DP bridge demo design.

Table A.1. Resource Utilization of the CertusPro-NX MIPI DSI to DP Bridge Demo Design

MIPI DSI to DP Bridge Demo Design	Scaling Method	LUT4	Registers	EBR	DSP
Total	Bilinear	13,438	11,668	58	28
	Lanczos	17,322	13,811	79	100

References

- CertusPro-NX web page
- Modular FMC Adapter and DisplayPort Daughter Cards web page
- Lattice Radiant Software web page
- Lattice Radiant Timing Constraints Methodology (FPGA-AN-02059)
- 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, refer to the Lattice Answer Database at www.latticesemi.com/Support/AnswerDatabase.

Revision History

Revision 1.1, March 2025

Section	Change Summary
Abbreviations in This Document	Added definition for FFC and FPC.
Hardware and Software Requirements	Updated the Hardware Requirements and Software Requirements sections.
Demo Design Overview	Updated the Video Scaler section. Added the subsections.
Setting Up the Demo	• Updated the Raspberry Pi 5 Board section. • Updated Figure 4.4. CertusPro-NX PCIe Bridge Board RevC. • Updated the step about the actions after programming the FPGA in the Programming the FPGA section.

Revision 1.0, December 2024

Section	Change Summary
All	Production release.

www.latticesemi.com