



CertusPro-NX Versa Board - Customizable HSB Sensor Interfaces

Reference Design

FPGA-RD-02326-1.0

November 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

Contents	3
Abbreviations in This Document.....	7
1. Introduction	8
1.1. Quick Facts	9
1.1.1. Quick Start Guide	9
1.2. Features	10
1.3. Naming Conventions	10
1.3.1. Nomenclature.....	10
1.3.2. Signal Names	10
2. Directory Structure and Files	11
3. Functional Description.....	12
3.1. Customizable MIPI CSI-2 Sensor Interface	12
3.1.1. MIPI CSI-2 RX DPHY IP	13
3.1.2. MIPI CSI-2 to AXIS Payload Converter	13
3.2. Hololink IP	14
3.2.1. Overview	14
3.3. Ethernet 1G/10G Support in CPNX Device	15
3.3.1. Ethernet 1G Support with 10/100/1000 Mbps SFP Transceiver	15
3.3.2. Ethernet 10G Support with 10GBase-T SFP+ Transceiver	16
3.4. Clocking and Reset Scheme.....	16
4. Customizing the Reference Design	19
4.1. Steps for Customizing the Reference Design	19
4.1.1. IP Generation	19
4.1.2. HDL Parameter and Define Switch Configuration	22
4.1.3. Platform Design Constraint Configuration	23
4.1.4. Driver Configuration and Patches	26
4.1.5. Verifying the Reference Design Configuration	26
4.2. Customizing the Reference Design for a CertusPro-NX Versa Evaluation Board	27
5. Compiling the Reference Design	30
5.1. Configuring the Reference Design Parameters	30
5.2. Assigning a Revision ID for each Bitstream	30
5.3. Configuring the Lattice Radiant Software to Compile the Reference Design	31
5.4. Generating the Bitstream File	32
6. Implementing the Reference Design on the CertusPro-NX Versa Board.....	33
6.1. Requirements.....	33
6.1.1. Hardware.....	33
6.1.2. Software	33
6.2. Jetson AGX ORIN Host Setup.....	35
6.2.1. Setting Up the Jetson AGX ORIN Developer Kit.....	35
6.2.2. Configuring the AGX Orin Developer Kit	39
6.2.3. Building the Holoscan Sensor Bridge Demo Container	42
6.2.4. Applying Patch to Support the IMX258 Camera Module	42
6.3. Testing the System	42
6.3.1. Programming the Bit File.....	42
6.3.2. Camera Video Streaming Test	46
7. Resource Utilization.....	48
7.1. Resource Utilization with Ethernet 1G.....	48
7.2. Resource Utilization with Ethernet 10G.....	49
8. Debug Methodology	50
8.1. Powering Up the Evaluation Board	51
8.1.1. Troubleshooting Common Issues.....	51
8.2. Verifying the Ethernet Connection	52

8.2.1. Troubleshooting Connection Issues	53
8.3. Running the Streaming Script (linux_imx258_player.py).....	54
8.3.1. Preparing the Streaming Script	54
8.3.2. Monitoring the Output of the Streaming Script	54
8.3.3. Troubleshoot Common Issues	55
Appendix A: Known Issues	56
References	57
Technical Support Assistance	58
Revision History.....	59

Figures

Figure 1.1. Overview of Customizable HSB Sensor Interface RD with NVIDIA ecosystem	9
Figure 2.1. Directory Structure	11
Figure 3.1. Overview of Reference Design Block Diagram	12
Figure 3.2. Overview block diagram of MIPI CSI-2 interface	13
Figure 3.3. Holoscan Sensor Bridge IP in FPGA	15
Figure 3.4. Reset and Clock Domain Block Diagram for Ethernet 10G	16
Figure 3.5. Reset and Clock Domain Block Diagram for Ethernet 1G	17
Figure 4.1. Steps Flow to Customize and Verify the Reference Design	19
Figure 4.2. MIPI CSI-2 RX DPHY IP Generation Module	20
Figure 4.3. Ethernet 1G (MAC + SGMII SerDes) IP Generation	21
Figure 4.4. Ethernet 10G MAC IP Generation	21
Figure 4.5. Ethernet 10G PCS IP Generation	22
Figure 4.6. Define Switch Setting in HOLOLINK_def.svh File	23
Figure 4.7. PDC File Setting for 1G or 10G	24
Figure 4.8. Screenshot of Lattice Radiant Software Hierarchy File List	26
Figure 4.9. CertusPro-NX Versa Evaluation Board Block Diagram	27
Figure 4.10. Required HDL Parameter and DEFINE Switch Configuration	28
Figure 4.11. PDC Configuration File	28
Figure 4.12. Lattice Radiant Software Hierarchy File List	29
Figure 5.1. Revision ID assignment in hsb_1chip_cpnx_top.sv	30
Figure 5.2. Revision ID Date Code Representation	31
Figure 5.3. Lattice Radiant Software	31
Figure 5.4. Open Project File	31
Figure 5.5. Place and Route Design Strategy	32
Figure 5.6. Generate and Export Bitstream File	32
Figure 6.1. Hardware Setup	34
Figure 6-2. Camera Sensor Connection	34
Figure 6.3. Ubuntu Host PC and Jetson AGX Orin Developer Kit Setup	35
Figure 6.4. Recovery and Reset Button	35
Figure 6.5. Jetson AGX Orin [64GB developer kit version]	36
Figure 6.6. JetPack 6.2.1 (Rev.1)	36
Figure 6.7. Terms and Conditions – License Agreement	37
Figure 6.8. Download Operation and OS Image Creation	37
Figure 6.9. SDK Manager	38
Figure 6.10. Summary Finalization	38
Figure 6.11. MAXN Power Mode	40
Figure 6.12. Hardware Setup for Bit File Programming	43
Figure 6.13. Radiant Programmer Window	43
Figure 6.14. Select Cable Settings	44
Figure 6.15. Load Bitstream File	44
Figure 6.16. Program Device Toolbar Icon	45
Figure 6.17. Message on Successful Programming	45
Figure 6.18. Console Output — Remote Firmware Update	46
Figure 6.19. Camera Video Streaming Test Setup	47
Figure 8.1. Debug Methodology Flow Chart For RD	50
Figure 8.2. Position of the LED on CPNX Versa Evaluation Board	51
Figure 8.3. Ethernet Link Establishment Status	52
Figure 8.4. Data Packet Received by Host	53
Figure 8.5. Sample Output	53
Figure 8.6. LAN Light Indication on Host	54
Figure 8.7. Revision ID Version	54
Figure 8.8. Camera I2C configuration fail log	55

Tables

Table 1.1. Summary of the Reference Design	9
Table 2.1. Folder Structure Description	11
Table 3.1. Clock Domain Distribution for Reference design with Ethernet 10G	17
Table 3.2. Clock Domain Distribution for Reference design with Ethernet 1G	17
Table 3.3. Reset Distribution applicable for both Ethernet 10G and 1G	18
Table 4.1. Reference design's MIPI CSI-2 RX DPHY IP configuration options	19
Table 4.2. PDC pin assignment for 2-dimension Array for MIPI Data Lane	25
Table 4.3. Configuration required for reference design to fit into CPNX Versa Evaluation Board	27
Table 5.1. Revision ID's Board ID and FW ID Representation	30
Table 6.1. Hardware Needed	33
Table 7.1. Logic Consumption for each Instance Available in the Reference Design	48
Table 7.2. Comparison between Reference Design Utilization and CertusPro-NX Resources Available	48
Table 7.3. Logic Consumption for each Instance Available in the Reference Design	49
Table 7.4. Comparison between Reference Design Utilization and CertusPro-NX Resources Available	49
Table 8.1. LED status upon Board Power-up	51
Table 8.2. Expected Status LED Behavior when running linux_imx258_player.py.script	55

Abbreviations in This Document

A list of abbreviations used in this document.

Abbreviations	Definition
APB	Advanced Peripheral Bus
AXI	Advanced eXtensible Interface
BOOTP	Bootstrap Protocol
CDC	Clock Domain Crossing
CPNX	CertusPro-NX
CRC	Cyclic Redundancy Check
DPHY	Digital Physical Layer
EBR	Embedded Block RAM
ECB	Ethernet Control Bus
FPGA	Field-Programmable Gate Array
FW	Firmware
GUI	Graphical User Interface
HDL	Hardware Description Language
HIF	Host Interface
HSB	Holoscan Sensor Bridge
I2C	Inter-Integrated Circuit
ICMP	Internet Control Message Protocol
IP	Intellectual Property (Core)
MAC	Media Access Control
MIPI	Mobile Industry Processor Interface
MPCS	Multi-Protocol Communication Stack
NGC	NVIDIA GPU Cloud
PCB	Printed Circuit Board
PCS	Physical Coding Sublayer
PDC	Platform Design Constraint
PFU	Programmable Function Unit
PLL	Phase-Locked Loop
PTP	Precision Time Protocol
SDK	Software Development Kit
SERDES	Serializer/Deserializer
SFP	Small Form-factor Pluggable
SPI	Serial Peripheral Interface
STA	Static Timing Analysis
UDP	User Datagram Protocol
YAML	YAML Ain't Markup Language

1. Introduction

This user guide describes the reference design for customizable HSB sensor interfaces, implemented using the Lattice Semiconductor CertusPro™-NX Versa Evaluation Board. The customizable interfaces in this reference design are divided into two categories.

Note: Only a single Ethernet port is used for the CertusPro-NX Versa Evaluation board, as there is only one camera connector available for testing.

- **Sensor Interface**

This reference design targets a MIPI CSI-2 camera sensor. The sensor interface is designed to be flexible and scalable, allowing you to integrate multiple camera sensors based on your requirements. The system supports various configurations to meet diverse imaging needs.

- **Host Interface**

The host interface focuses on Ethernet connectivity and supports multiple link speeds to accommodate varying data throughput requirements. This enables seamless communication between the sensor bridge and host systems.

In this reference design, the primary objective is to establish a robust and reliable bridge between the HSB interfaces and the NVIDIA host. Image tuning is not included and involves defining image quality parameters, configuring processing pipelines, and optimizing the Image Signal Processing (ISP). These tasks are highly specific to the camera sensor and require dedicated resources to achieve optimal results.

The reference design leverages NVIDIA's Holoscan Sensor Bridge solution and integrates smoothly within the broader NVIDIA ecosystem, enabling high-performance sensor data processing and transmission. An overview of the key components of the NVIDIA Holoscan Sensor Bridge solution and ecosystem is as follows:

1. **Sensors**

Targets the MIPI CSI-2 camera sensor.

2. **Holoscan Sensor Bridge (Lattice CertusPro™-NX Versa Evaluation Board)**

- Lattice CertusPro™-NX Versa Evaluation Board functions as the Holoscan Sensor Bridge solution.
- Performs sensor data pre-processing to reduce computational load on the host system.
- The reference design is implemented in the Lattice FPGA targeting CPNX-100 speed grade 9 device.

The key components of the reference design include:

- **Sensor Interface**
 - Supports MIPI CSI-2 camera sensors.
 - Highly configurable to accommodate various camera sensors with different MIPI lanes and lane rates.
- **Hololink IP**
 - Acts as a bridge between the sensor interface and the host system.
 - Handles data encapsulation, synchronization, and buffering to ensure smooth transmission of high-throughput sensor data.
- **Host Interface (Ethernet link)**
 - Provides a high-speed Ethernet link to the host, supporting 1G or 10G speeds.
 - Enables low-latency, and high-bandwidth communication.

3. **Host (NVIDIA Jetson AGX ORIN Platform)**

- Runs the Holoscan SDK to process incoming data.
- Performs real-time AI inference, visualization, and decision-making.
- Streams processed data to the cloud or other edge devices.

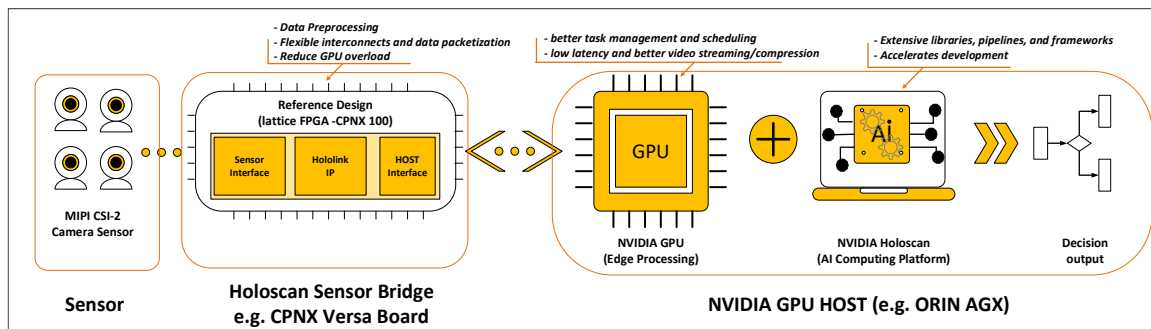


Figure 1.1. Overview of Customizable HSB Sensor Interface RD with NVIDIA ecosystem

1.1. Quick Facts

Table 1.1. Summary of the Reference Design

General	Target Devices	CertusPro-NX FPGA (LFCPNX-100-9LFG672I)
	Source code format	Verilog and SystemVerilog
Simulation	Functional simulation	N/A
	Timing simulation	N/A
	Testbench	N/A
	Testbench format	N/A
	Testbench format	N/A
Software Requirements	Software tool and version	Lattice Radiant™ software, version 2025.1.0.39.0
	IP version (if applicable)	<ul style="list-style-type: none"> • CSI-2/DSI D-PHY Rx IP v1.9.0 • OSC IP v1.4.0 • PLL IP v1.9.1 • 10G Ethernet PCS IP Core v1.3.0 • 10G Ethernet MAC IP Core v1.1.0 • Tri-speed Ethernet v2.1.0 (MAC + SGMII (SERDES)) • Hololink IP v2507
	Host Software	<ul style="list-style-type: none"> • JetPack 6.2.1 • NVIDIA Holoscan SDK 3.3.0 • NVIDIA Jetson Linux 36.4.4 • HSB SDK 2.2.0
Hardware Requirements	Board	<ul style="list-style-type: none"> • CPNX Versa Evaluation Board • IMX258 camera module • Ethernet 10GBase-T (Cat 6 RJ-45) SFP+ module (10G Ethernet) • 10/100/1000BASE-T SFP module (1G Ethernet) • Display Monitor
	Host	<ul style="list-style-type: none"> • NVIDIA Jetson AGX Orin Platform
	Cable	<ul style="list-style-type: none"> • DisplayPort cable • Ethernet Cat 6 cable

1.1.1. Quick Start Guide

Refer to [Implementing the Reference Design on CPNX Versa Board](#) for instructions on setting up and streaming video from the IMX258 camera module to the NVIDIA Jetson AGX Orin developer kit over a 1G/10G Ethernet connection.

1.2. Features

Key features of the reference design include:

- Sensor Interface
 - Supports one or more camera sensors.
 - Designed to be customizable, allowing flexible configuration of the MIPI CSI-2 RX DPHY interface.
 - Enables integration of various camera types.
 - In testing, the CPNX Versa Evaluation Board was used with a single MIPI CSI-2 camera.
- Hololink IP
 - Acts as a bridge between the sensor interface and the host system.
 - Implements Ethernet packetization and supports streaming DMA, control interfaces, and transport abstraction.
 - Enables low-latency, high-throughput data transfer from sensors to compute platforms.
 - Supports Precision Time Protocol (PTP) per IEEE 1588-2019 specification. PTP synchronizes the Hololink IP's internal time with the host time, enabling the host to act as the Time Transmitter and send PTP packets.
- Host Interface (Ethernet link)
 - Supports Ethernet channels with a link speed of 1G or 10G.
 - Uses UDP over Ethernet to stream sensor data directly to GPU memory.
 - Enables real-time AI inference and visualization with minimal latency.
 - In testing, the CPNX Versa Evaluation Board was used with a single Ethernet port configured for either 1G or 10G.

1.3. Naming Conventions

1.3.1. Nomenclature

The nomenclature used in this document is based on Verilog HDL.

1.3.2. Signal Names

- `_n` are active low (asserted when value is logic 0)
- `_i` are input signals
- `_o` are output signals

2. Directory Structure and Files

The following figure illustrates the directory structure of the reference design.

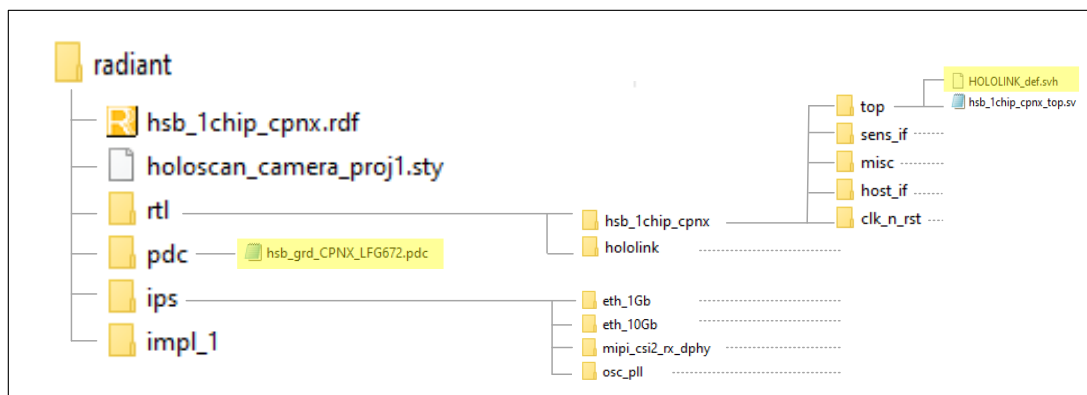


Figure 2.1. Directory Structure

Table 2.1. Folder Structure Description

Folder/File	Description
hsb_1chip_cpnx.rdf	This file is the Lattice Radiant software project file for the CertusPro-NX reference design.
holoscan_camera_proj1.sty	This file is the strategy file used for the project. Custom settings were applied to improve place-and-route results.
rtl/hsb_1chip_cpnx	This folder contains HDL code for the sensor interface, host interface, top level design and other components.
rtl/hololink	This folder contains the encrypted Hololink IP. The current version is 2507.
pdc	This folder contains the PDC file.
ips	This folder contains IP cores used in the reference design, such as the oscillator, PLLs, MIPI CSI-2 RX DPHY IP, Ethernet 10G MAC and PCS, and Tri-speed Ethernet v2.1.0.00 (MAC + SGMII (SERDES)).
imp_1	This folder contains reports and logs generated during FPGA bitstream compilation.

Note: The highlighted files in Figure 2.1 (HOLOLINK_def.svh and hsb_grd_CPNX_LFG672.pdc) are important. You will modify these files to create your intended reference design.

3. Functional Description

This section describes the architecture of the reference design. The system architecture includes several key subsystems working together seamlessly. The reference design includes three main components: the MIPI CSI-2 sensor interface, Hololink IP, and the Ethernet link as host interface. The reference design is implemented on the CPNX Versa Evaluation Board. Figure 3.1 shows the overview of the reference design block diagram.

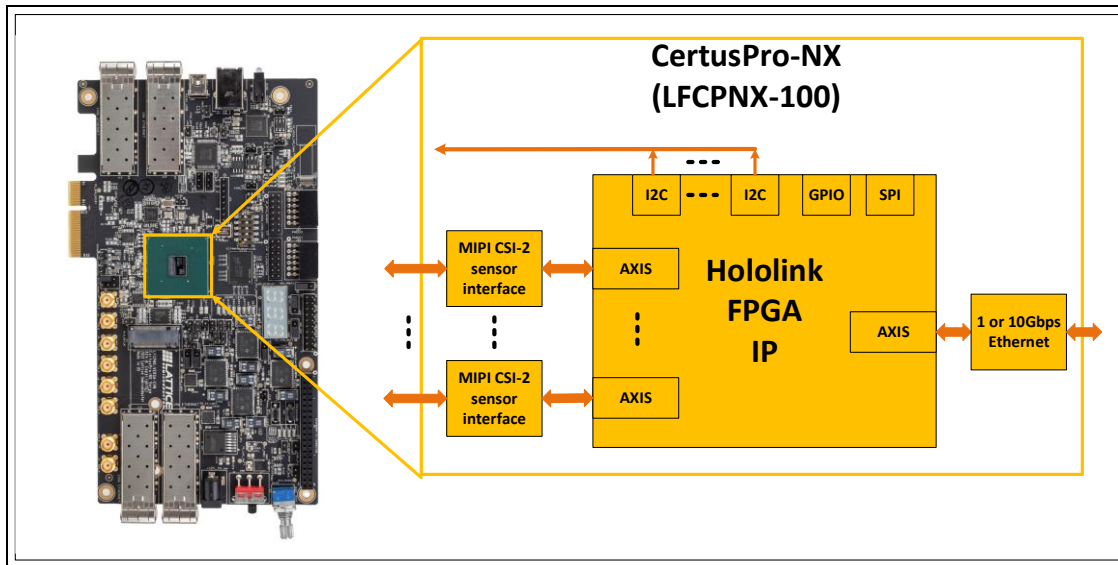


Figure 3.1. Overview of Reference Design Block Diagram

The MIPI CSI-2 sensor interface is highly customizable, supporting a variety of camera configurations based on user platform requirements. Its primary function is to receive raw image data from camera sensors and convert it into AXI-Stream (AXIS) payloads, which are then transmitted to the Hololink IP for further processing. This modularity ensures flexibility in adapting to different camera sensor types and resolutions.

The Hololink IP, provided by NVIDIA, serves as a critical bridge between the sensor interface and the host interface. It offers extensive configurability, including reconfigurable AXIS ports on both the sensor and host sides, and supports control peripherals such as I2C, GPIO, and SPI. This flexibility allows for seamless integration and communication between the sensor subsystem and the host processing platform.

The Ethernet block functions as the host interface. It supports 1G and 10G link speeds: 10G Ethernet uses MAC and PCS IP cores, while 1G Ethernet uses MAC + SGMII (SERDES) IP core. The reference design utilizes a single 1G or 10G Ethernet block. The Ethernet block connects to the Hololink IP via AXI-Stream ports and uses UDP over Ethernet to stream sensor data directly to GPU memory.

3.1. Customizable MIPI CSI-2 Sensor Interface

The customizable MIPI CSI-2 camera sensor interface has two capabilities:

- **Scalability**
Refers to the ability of the MIPI CSI-2 sensor interface to increase or decrease the number of the sensor interfaces based on user platform requirements.
- **Configurability**
Refers to the ability to configure each camera sensor interface individually, such as setting different MIPI CSI-2 lane counts and lane rates.

Figure 3.2 shows the block diagram of MIPI CSI-2 interface. It consists of two main components: the MIPI CSI-2 RX DPHY IP and the MIPI CSI-2 to AXIS payload converter.

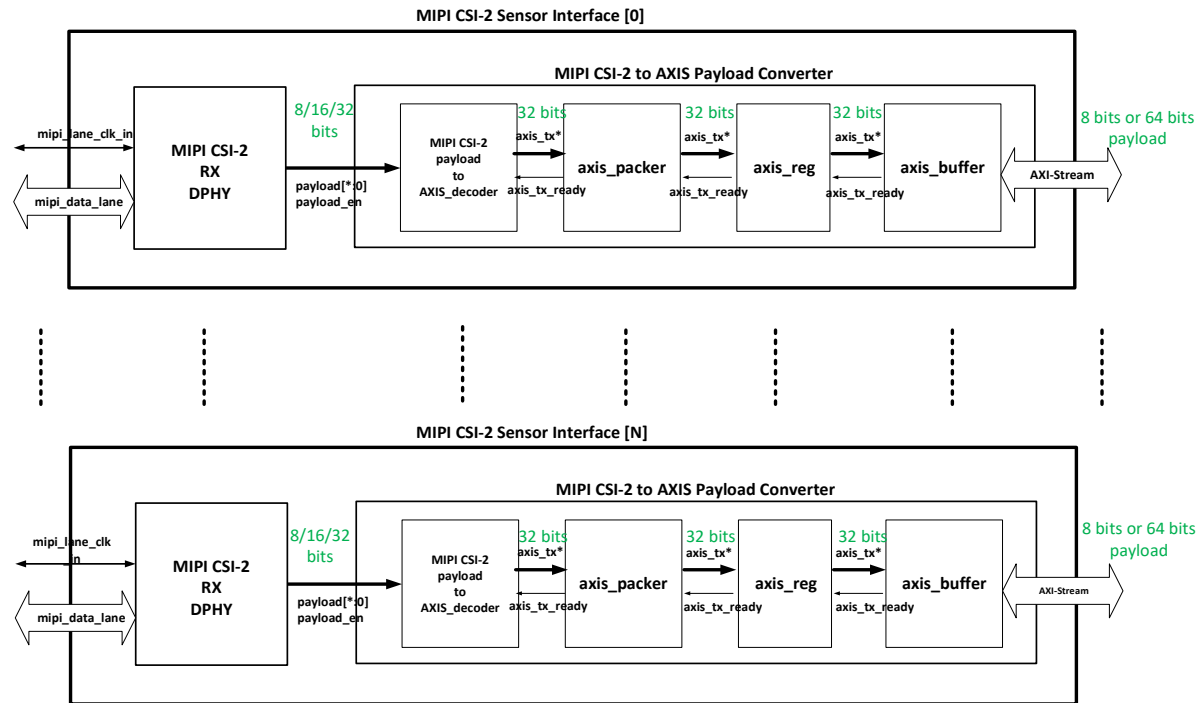


Figure 3.2. Overview block diagram of MIPI CSI-2 interface

3.1.1. MIPI CSI-2 RX DPHY IP

As the reference design targets the CPNX-100 device (package type), only the soft MIPI CSI-2/DSI RX DPHY IP is available. The soft MIPI CSI-2 RX DPHY IP is a configurable block. It receives MIPI CSI-2 data streams and converts them into parallel data formats such as 8-bit, 16-bit or 32-bit, based on the number of lanes configured during the IP generation. The IP provides flexibility to configure the MIPI CSI-2 lane rate according to the camera sensor's requirement. Additionally, it generates an output byte clock based on the data type, which is passed to the AXIS blocks. When the sensor outputs a continuous D-PHY clock, it is recommended to disable RX_FIFO in the settings and drive the `clk_byte_fr_i` input clock directly using the `clk_byte_hs_o` output clock. However, the soft MIPI CSI-2 RX DPHY IP has limitations compared to the hardened version, particularly in performance. These limitations may vary depending on the FPGA device family and package type used. For more detailed specifications and configuration guidelines, refer to the [CSI-2/DSI D-PHY Receiver IP Core - User Guide \(FPGA-IPUG-02081\)](#).

3.1.2. MIPI CSI-2 to AXIS Payload Converter

The MIPI CSI-2 to AXIS Payload Converter module bridges the MIPI CSI-2 protocol and the AXI4-Stream (AXIS) interface. It extracts image data packets from the CSI-2 stream and reformats them into AXIS-compatible payloads for downstream processing.

This converter handles:

- Parsing of CSI-2 protocols elements (such as, long packets, short packets).
- Extracting payload from CSI-2 long packets.
- AXIS framing, including valid signals and handshake logic.
- Detecting errors in packet integrity and protocol compliance.

It enables seamless integration of camera sensor data into FPGA-based image processing pipelines using standard AXI4-Stream interfaces.

Note: Depending on the Hololink IP version, some features listed above may not be used. For example, in Hololink IP version 2507, feature (i) and (iv) are not supported.

3.1.2.1. MIPI CSI-2 Payload to AXIS Decoder Module

This module extracts image data packets from the CSI-2 stream and converts them into AXIS-compatible payloads. It supports CSI-2 stream output in parallel RAW10 formats of 8-bit, 16-bit, or 32-bit, as provided by the MIPI CSI-2 RX D-PHY IP. AXIS-compatible payloads are data packets formatted according to the AXI4-Stream (AXIS) protocol.

1. Structured data words: Payloads are aligned into fixed-width data words.
 - TDATA[31:0]: Carries the payload of the MIPI CSI-2 RAW10
2. Control signals: Each payload includes AXIS control signals:
 - TVALID: Indicates valid data.
 - TREADY: Handshake signal from the receiver.
 - TLAST: Marks the end of a frame or packet.
 - TKEEP[3:0]: Specifies which bytes in the data word are valid.
 - TUSER[1:0] Typically used for user-defined metadata.
 - tuser[0] is asserted during cycles containing embedded data (MIPI Data Type = 0x12)
 - tuser[1] is asserted on the final clock cycle of a long packet, indicating the line end signal.

3.1.2.2. AXIS Packer Module

A key feature of this module is its handling of the TKEEP signal, which indicates the validity of individual bytes within each data word. The AXIS Packer filters out invalid TKEEP bits, ensuring that only valid image data is included in the output stream. This maintains data integrity and prevents downstream modules from processing corrupt or incomplete payloads.

3.1.2.3. AXIS REG Module

The AXIS REG module acts as a simple register stage within the AXIS data path. Its primary function is to shift and hold one cycle of AXIS data, helping to manage timing, pipeline depth, and synchronization between modules.

3.1.2.4. AXIS Buffer Module

The AXIS Buffer Module receives AXIS data from the AXIS REG module and passes it through a dual-clock FIFO (DC_FIFO) to facilitate clock domain crossing (CDC). This ensures reliable data transfer between modules operating in different clock domains.

Key functions include:

- Data buffering: Temporarily stores AXIS data and control signals to manage timing differences.
- Clock domain crossing: Uses DC_FIFO to safely transfer data between asynchronous clock domains.
- Flow control: Maintains AXIS protocol integrity using TVALID, TREADY, and TLAST signals across domains.

This module is essential in systems where the CSI-2 receiver and downstream AXIS processing blocks operate on separate clocks, ensuring smooth and error-free data streaming.

3.2. Hololink IP

3.2.1. Overview

A detail explanation on the Nvidia Hololink IP is provided in the NVIDIA documentation webpage, [Holoscan Sensor Bridge v2.2.0](#) - NVIDIA Docs. You are encouraged to review the document for further understanding. This section provides only an overview of the IP.

The NVIDIA Holoscan Sensor Bridge FPGA IP works with the Holoscan software to provide a sensor-agnostic platform that transfers data from various sensors to an Ethernet-connected host.

The Holoscan Sensor Bridge FPGA IP, comprises three main components: the Sensor Interface IP, the Hololink IP, and the Host Interface IP. [Figure 3.3](#) illustrates the block diagram of the Holoscan Sensor Bridge FPGA IP. The Sensor Interface and Host (Ethernet) Interface blocks are implemented using FPGA vendor-specific logic. In this reference design, Lattice Semiconductor is responsible for the integration and maintaining the IP for both the Sensor Interface and Host Interface components.

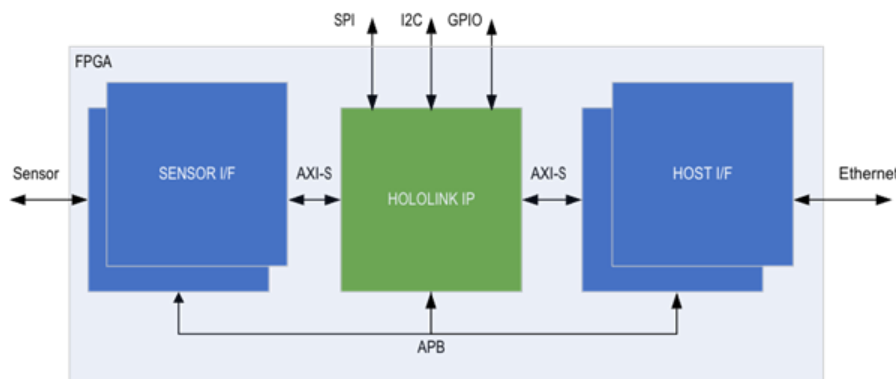


Figure 3.3. Holoscan Sensor Bridge IP in FPGA

The Holoscan Sensor Bridge FPGA IP streamlines FPGA development, offering scalability and configurability to support a wide range of sensor-to-host applications.

Key functions of the Holoscan Sensor Bridge IP include:

- Encapsulation of sensor data: Converts AXI-Stream sensor data into Ethernet UDP AXI-Stream format for host-side processing.
- Network protocol support: Implements BOOTP, ICMP, and NVIDIA-defined Ethernet Control Bus (ECB) protocols.
- Event and control packet transmission: Sends enumeration and control event packets based on predefined conditions.
- Peripheral interface control: Manages SPI, I2C, and GPIO interfaces to configure sensors and other onboard components.

3.3. Ethernet 1G/10G Support in CPNX Device

The CPNX device supports high-speed Ethernet connectivity via integrated IP cores and external transceiver modules. This section outlines the supported configurations for both 1G and 10G Ethernet, including key components, integration details, and functional descriptions.

3.3.1. Ethernet 1G Support with 10/100/1000 Mbps SFP Transceiver

The CPNX device supports 1G Ethernet using 10/100/1000BASE-T SFP transceivers. This configuration leverages the Lattice Tri-Speed Ethernet IP, which provides a flexible, robust solution for Ethernet connectivity at 10 Mbps, 100 Mbps, and 1G speeds.

Key Components:

- **Tri-Speed Ethernet IP Core**
Supports MAC + SGMII (SERDES) mode for high-speed serial data transfer. Compliant with IEEE 802.3 and Cisco SGMII standards, it enables auto-negotiation for speed and duplex and includes statistics counters for monitoring.
- **High-Level Block Diagram Components**
 - MAC Core: Handles Ethernet frame processing (CRC checks, filtering) and interfaces with user logic via AXI4-Stream for both transmit and receive paths.
 - SGMII PCS (SERDES): Converts GMII data into serialized differential signals using 8b/10b encoding at 1.25 Gbps. Supports auto-negotiation and link establishment.
 - Host Interface: Configurable via AXI4-Lite, APB, or AHB-Lite for MAC and PCS settings, including speed, duplex mode, and interrupt handling.

The MAC core processes Ethernet frames and passes GMII data to the SGMII PCS, which serializes it for transmission through the SFP transceiver. This architecture ensures reliable, efficient Ethernet communication for embedded applications.

3.3.2. Ethernet 10G Support with 10GBase-T SFP+ Transceiver

The CPNX device supports 10G Ethernet using SFP+ transceivers, enabling high-speed data communication for bandwidth-intensive applications. The solution integrates multiple IP cores and hardware components.

Key Components:

- **MAC IP Core**
Manages Ethernet frame-level operations, including encapsulation, CRC generation, and flow control. Interfaces with system logic via AXI or Avalon, depending on the FPGA architecture.
- **PCS IP Core (Physical Coding Sublayer)**
Handles low-level physical layer functions and interfaces directly with the SFP+ module.

Features:

- Encoding/Decoding: Supports 8b/10b or 64b/66b schemes.
 - Scrambling/Descrambling: Maintains signal integrity.
 - Clock Data Recovery (CDR): Aligns incoming data with system clock.
 - Link Synchronization: Establishes and maintains stable link.
 - Error Handling: Optional Forward Error Correction (FEC) for enhanced reliability.
 - **SFP+ Transceiver Module**
A compact, hot-swappable module for both 10G and 1G connections.
- Features:
- Hot-Swappable: Enables module replacement without system shutting down the system.
 - Media Support: Compatible with copper and fiber (such as, SR for short-range, LR for long-range).
 - Diagnostics: Supports real-time monitoring of temperature, power, and signal quality.
 - Transmission range: Varies from a few meters (copper) to tens of kilometers (fiber).

The MAC and PCS IP cores work together to prepare and transmit Ethernet frames through the SFP+ module. This setup ensures high-speed, low-latency communication suitable for advanced networking applications.

3.4. Clocking and Reset Scheme

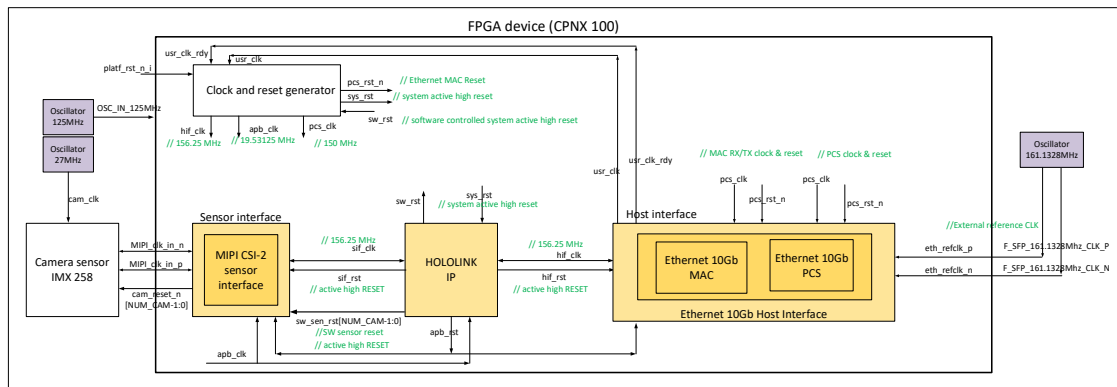


Figure 3.4. Reset and Clock Domain Block Diagram for Ethernet 10G

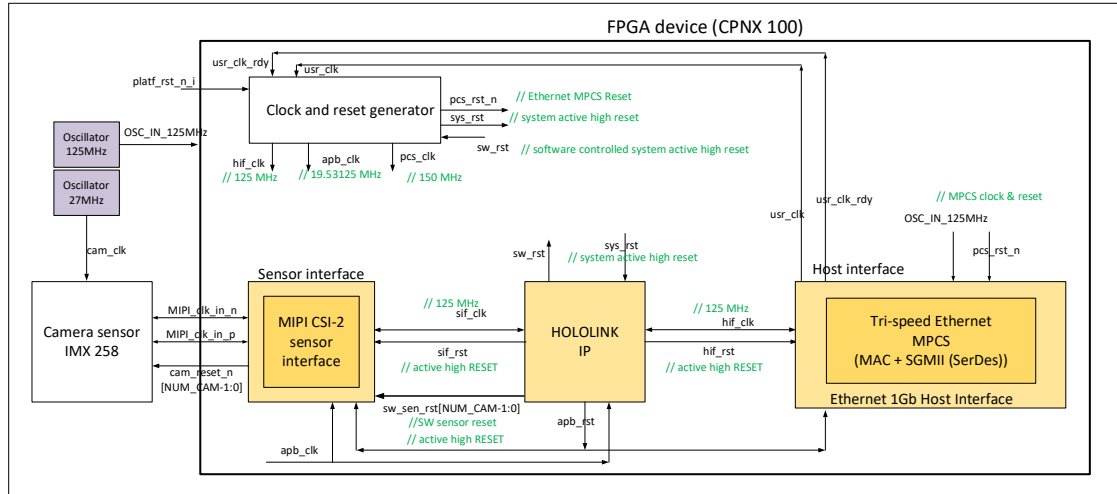


Figure 3.5. Reset and Clock Domain Block Diagram for Ethernet 1G

The following table lists the clock frequencies and their distribution.

Table 3.1. Clock Domain Distribution for Reference design with Ethernet 10G

Clocks	Frequency (MHz)	Description
usr_clk	322.266	PCS output clock used as reference clock for the PLL to generate system clocks (such as, apb_clk and hif_clk).
pcs_clk	150	PCS calibration clock.
OSC_IN_125MHz	125	Platform clock input used for debug hook signals.
hif_clk	156.25	Host interface clock used as AXIS clock for host interface.
sif_clk	156.25	Directly assigned from hif_clk and is used as the AXIS clock for sensor interface.
apb_clk	19.5313	APB bus clock.

Table 3.2. Clock Domain Distribution for Reference design with Ethernet 1G

Clocks	Frequency (MHz)	Description
usr_clk	125	PCS output clock used as reference clock for the PLL to generate system clocks (such as, apb_clk and hif_clk).
OSC_IN_125MHz	125	Platform clock input used for MPICS IP and debug hook signals.
hif_clk	125	Host interface clock used as AXIS clock for host interface.
sif_clk	125	Directly assigned from hif_clk and is used for the AXIS clock for sensor interface.
apb_clk	19.5313	APB bus clock.

Table 3.3. Reset Distribution applicable for both Ethernet 10G and 1G

Clocks	Description
sw_sen_rst[NUM_CAM-1:0]	Register-controlled sensor reset. The reset signal width increases based on the number of assigned cameras. Details are discussed in the Customizing the Reference Design section.
sw_sys_rst	Register-controlled system reset. Used to reset system-level logic. Also triggers reset for hif_rst, apb_rst, and sif_rst.
apb_rst	Reset APB logic
hif_rst	Reset Host logic
sif_rst	Reset Sensor logic

4. Customizing the Reference Design

4.1. Steps for Customizing the Reference Design

This section provides a detailed walkthrough of the steps required to configure the reference design and to verify the customized reference design. [Figure 4.1](#) illustrates the steps involved to enable customization and support for various customer-specific configurations. Detailed explanation is provided in this section. However, details on reference design compilation and implementation are covered separately in [Compiling the Reference Design](#) section and [Implementing the Reference Design on CPNX Versa Board](#) section.

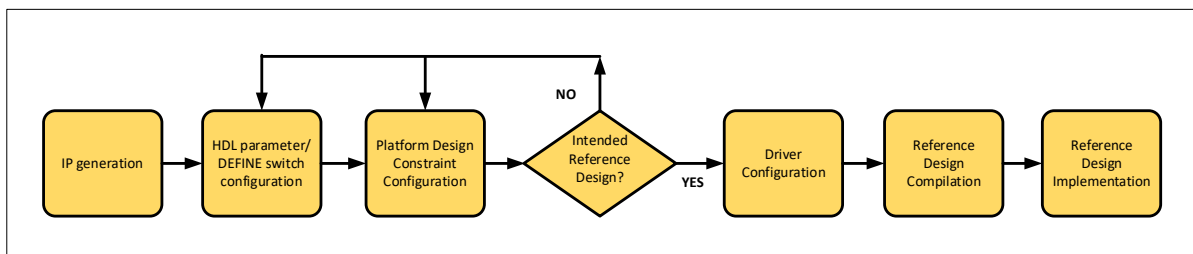


Figure 4.1. Steps Flow to Customize and Verify the Reference Design

4.1.1. IP Generation

4.1.1.1. MIPI CSI-2 RX DPHY IP generation

Each MIPI CSI-2 camera sensor interface requires a corresponding MIPI CSI-2 RX DPHY IP block. This IP must be generated based on the specifications of the camera sensor being used. The key parameters to configure includes:

- Number of data lanes
- MIPI lane rate
- Gear Ratio
- MIPI synchronizer clock frequency

These parameters must be correctly set before generating the IP to ensure compatibility with the camera sensor. The gear ratio for the CertusPro-NX family device is fixed at 8, and the MIPI synchronizer clock frequency in the reference design is set to 150 MHz. The reference design also provides preconfigured MIPI CSI-2 RX DPHY IPs for users to choose from. The table shows the available IP configuration option that the user can select. The IP is selected based on user settings, which will be explained in the [HDL Parameter and Define Switch Configuration](#) section.

Table 4.1. Reference design's MIPI CSI-2 RX DPHY IP configuration options

MIPI Lane Rate (Mbps)	MIPI Lane Number	MIPI RX DPHY IP name
720	1 lane	mipi_csi2_rx_DPHY_720Mbps_1L.ipx
	2 lanes	mipi_csi2_rx_DPHY_720Mbps_2L.ipx
	4 lanes	mipi_csi2_rx_DPHY_720Mbps_4L.ipx
960	2 lanes	mipi_csi2_rx_DPHY_960Mbps_2L.ipx
	4 lanes	mipi_csi2_rx_DPHY_960Mbps_4L.ipx
1440	4 lanes	mipi_csi2_rx_DPHY_1440Mbps_4L.ipx
1500	4 lanes	mipi_csi2_rx_DPHY_1500Mbps_4L.ipx

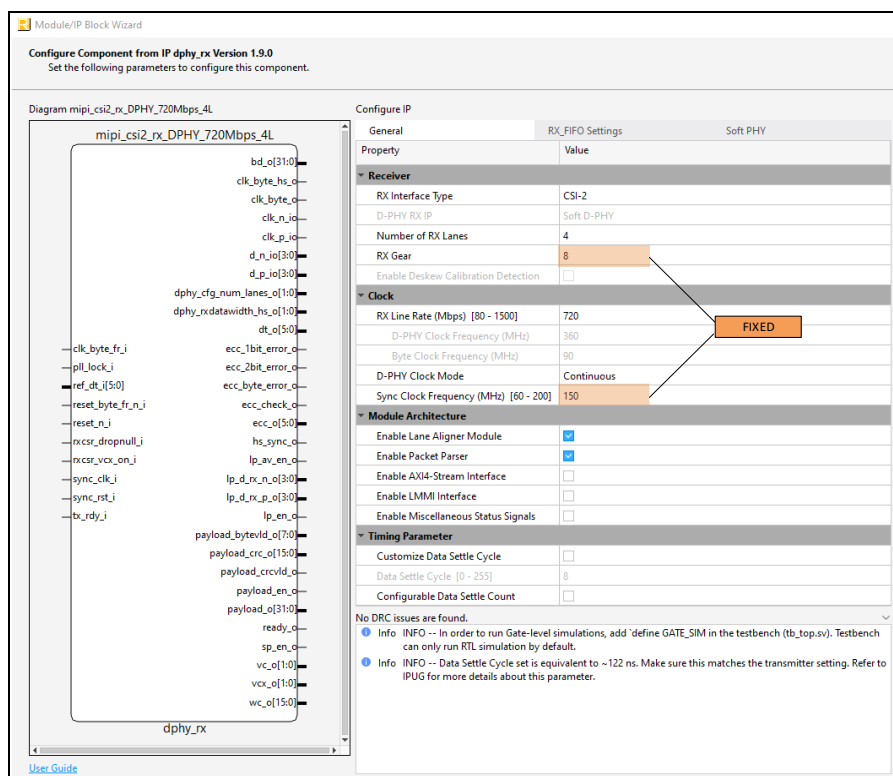


Figure 4.2. MIPI CSI-2 RX DPHY IP Generation Module

4.1.1.2. Hololink IP

Hololink IP is owned by NVIDIA, and NVIDIA updates the latest Hololink IP source code on GitHub at the following link, [GitHub - nvidia-holoscan/holoscan-sensor-bridge](https://github.com/nvidia-holoscan/holoscan-sensor-bridge) at 2.2.0-branch. The IP is encrypted. Currently, the reference design integrates the latest available Hololink IP, which is version 2507.

4.1.1.3. Ethernet IP Generation

Specific Ethernet IPs are required to support 1G or 10G link speeds in the CertusPro-NX FPGA device. To support Ethernet 1G link speed, the Tri-speed Ethernet v2.1.0 (MAC + SGMII (SERDES)) IP is used. To support Ethernet 10G link speed, the Ethernet PCS IP Core v1.3.0 and MAC IP Core v1.1.0 are used. The Ethernet Lane ID must be correctly set (Lane ID = 6) to meet the platform requirements for the CertusPro-NX Versa Evaluation Board before generating the IP. Figure 4.3 shows the Tri-speed Ethernet v2.1.0 (MAC + SGMII (SERDES)) IP generation for Ethernet 1G link speed support. Figure 4.4 shows the Ethernet 10G MAC IP generation, and Figure 4.5 shows the Ethernet 10G PCS IP generation module's GUI.

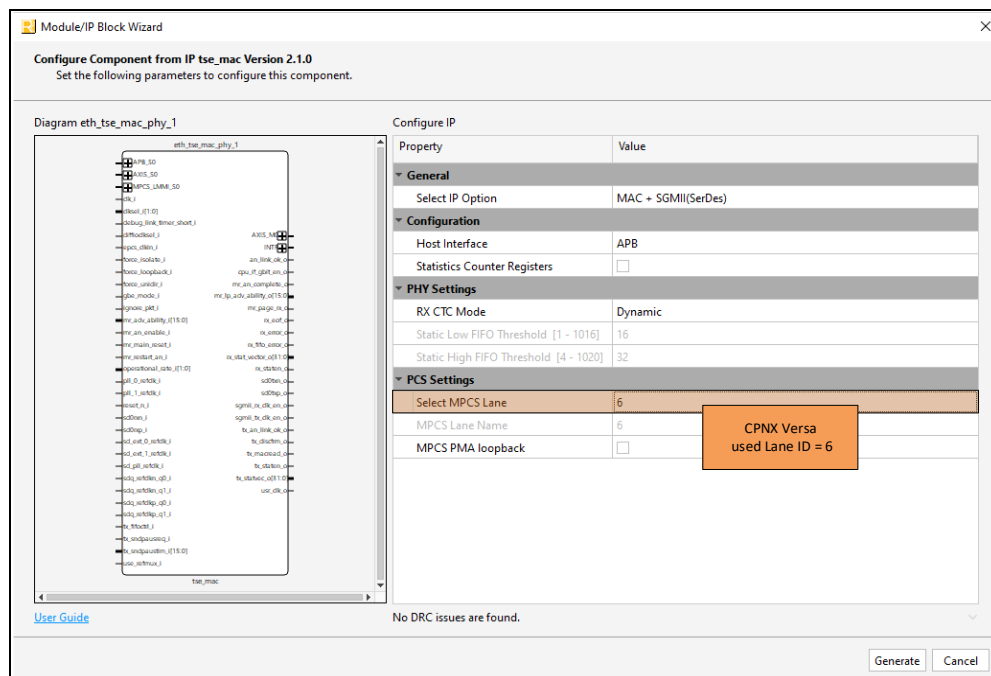


Figure 4.3. Ethernet 1G (MAC + SGMII SerDes) IP Generation

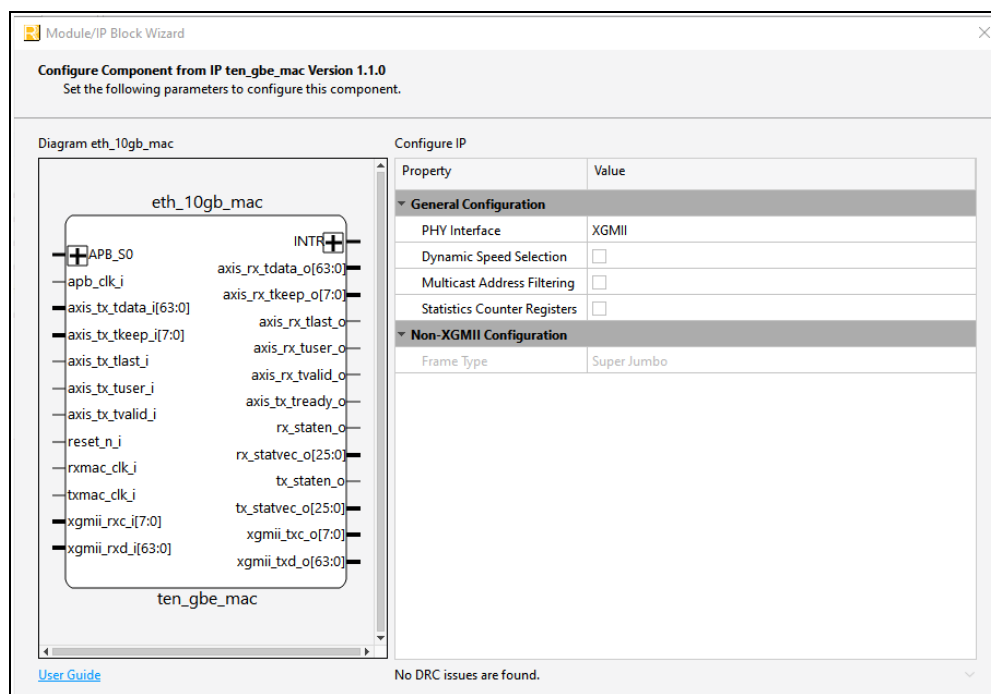


Figure 4.4. Ethernet 10G MAC IP Generation

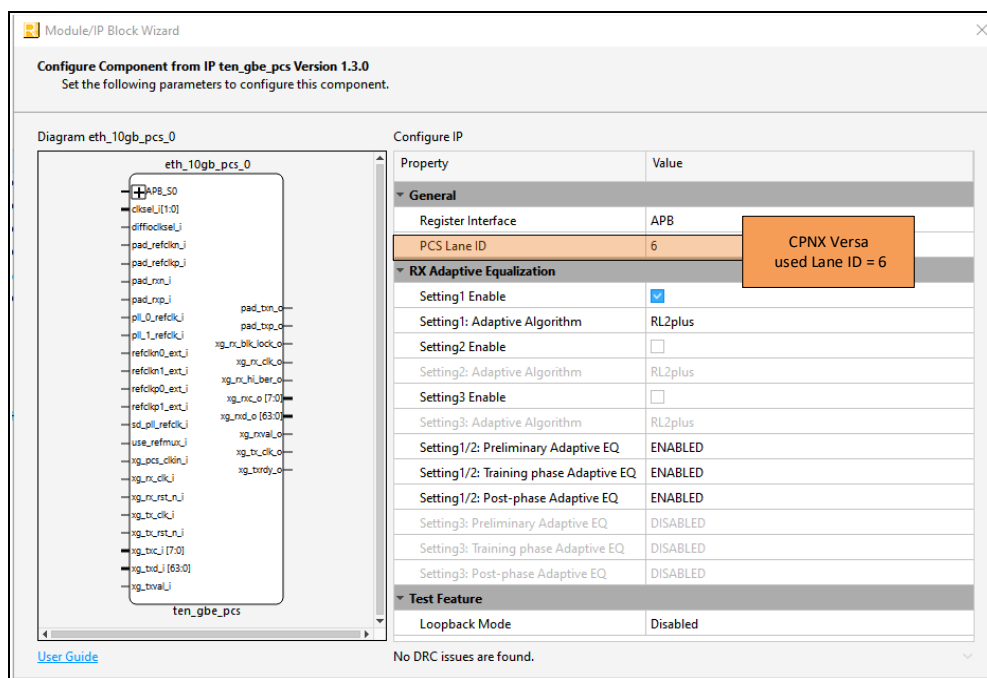


Figure 4.5. Ethernet 10G PCS IP Generation

4.1.2. HDL Parameter and Define Switch Configuration

Configure the HDL parameters and Define switches to reflect the desired hardware setup. These configurations modify the HDL design prior to compilation and affect the following aspects:

- Number of camera sensor interfaces
- Number of lanes per camera sensor
- Number of camera reset outputs
- I2C control peripherals for each camera sensor
- AXI-Stream ports connecting the sensor interface to the Hololink IP
- AXI-Stream ports connecting the Hololink IP to the host interface (Ethernet link)
- Ethernet link speed: 1G or 10G
- Number of Ethernet links used as host interface
- Ethernet MAC IP address

These changes should be made in the DEFINE file to ensure proper integration. The DEFINE file is located at:

`<project_name>/radiant/rtl/hsb_1chip_cpnx/top/HOLOLINK_def.svh`

Configurations such as cameras reset outputs, I2C camera controls, and AXI-Stream ports for the sensor/host interface are automatically configured based on the number of camera sensors and assigned Ethernet 1G/10G links. Figure 4.6 shows the DEFINE switches that need to be changed in HOLOLINK_def.svh file. These settings must be carefully configured to ensure the HDL design reflects the intended hardware setup and supports the required camera sensor interfaces.

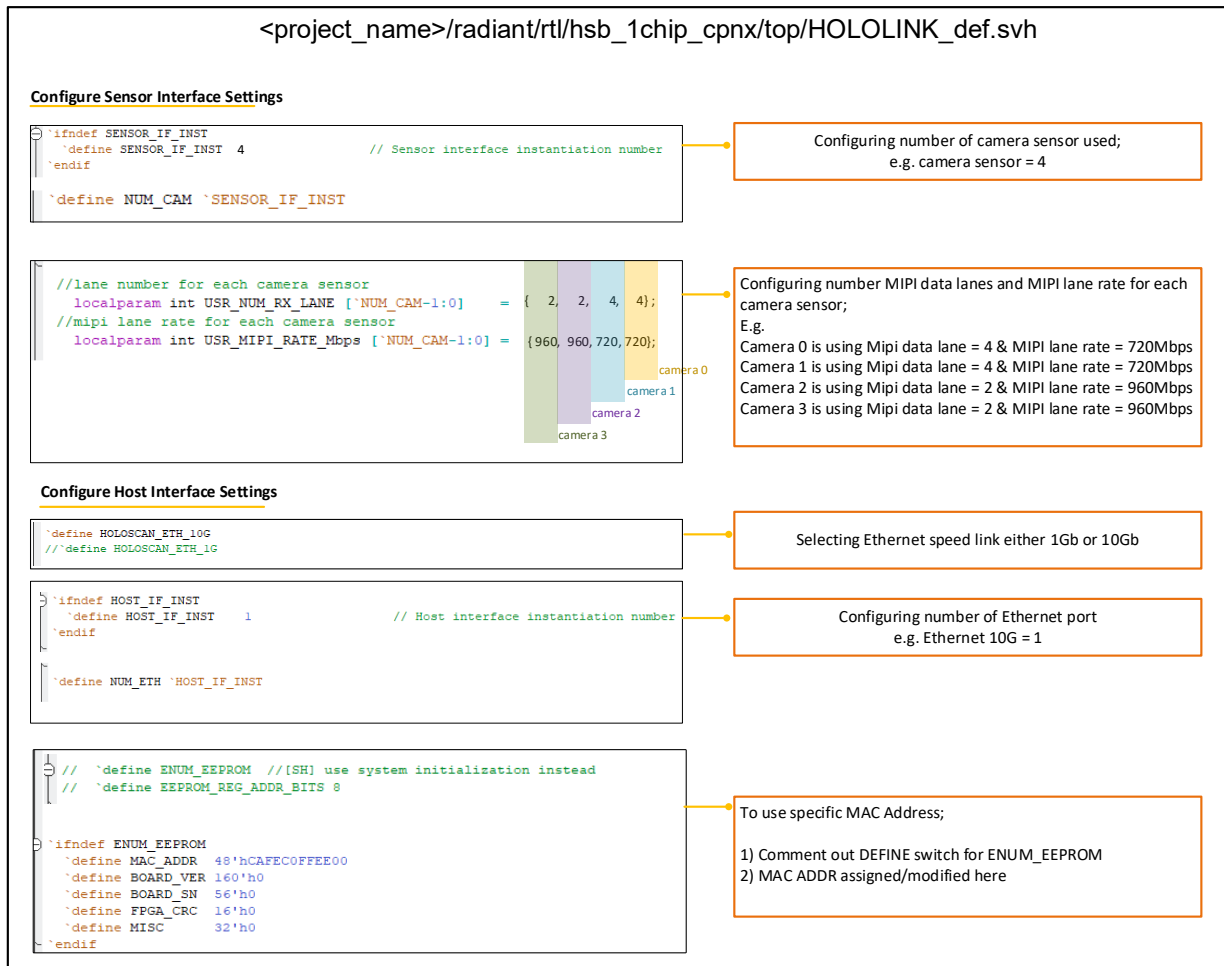


Figure 4.6. Define Switch Setting in HOLOLINK_def.svh File

Note: Figure 4.6 screenshot shows an example setting of four camera sensors. If more or less cameras are required for the intended reference design, the array size for `USR_NUM_RX_LANE[`NUM_CAM-1:0]` and `USR_MIPI_RATE_Mbps[`NUM_CAM-1:0]` change. Therefore, the data assigned to it are trimmed or added based on the intended camera number usage.

4.1.3. Platform Design Constraint Configuration

Once the HDL design configuration is finalized, platform design constraints in the PDC file are updated. A PDC template file is provided with this reference design and is located at:

`<project_name>/radiant/pdc/hsb_grd_CPNX_LFG672.pdc`

```

1 #####
2 # BEGIN: USR CONFIG SETUP
3 #####
4 #
5 # 1) SENSOR IF SETTING: CAM MIPI
6 #-----
7 #
8 # 1.1) available number of CAM
9 #-----
10 set CAM_NUM 4
11 #
12 #-----
13 # 1.2) cam0 config setup
14 #-----
15 set CAM0_LANE0 4
16 set MIPIlane0_inMbps 720
17 set mipigear0 8
18 #-----
19 # 1.3) cam1 config setup
20 #-----
21 set CAM1_LANE0 4
22 set MIPIlane0_inMbps 720
23 set mipigear1 8
24 #-----
25 # 1.4) cam2 config setup
26 #-----
27 set CAM2_LANE0 2
28 set MIPIlane2_inMbps 960
29 set mipigear2 8
30 #-----
31 # 1.5) cam3 config setup
32 #-----
33 set CAM3_LANE0 2
34 set MIPIlane3_inMbps 960
35 set mipigear3 8
36 #
37 #-----
38 # 2) HOST IF SETTING: ETHERNET
39 #-----
40 set ETH_NUM 1
41 #set ETH_SPEED_Gbps 1
42 set ETH_SPEED_Gbps 10
43 #
44 #####
45 # END: USR CONFIG SETUP
46 #####

```

Number of Camera Used

Camera0;
i) lane number
ii) MIPI lane rate

Camera1;
i) lane number
ii) MIPI lane rate

Camera2;
i) lane number
ii) MIPI lane rate

Camera3;
i) lane number
ii) MIPI lane rate

i) Number of Ethernet
ii) Ethernet speed link
"1" or "10"

Figure 4.7. PDC File Setting for 1G or 10G

Figure 4.7 shows the PDC file settings for the number of camera sensors, data lane, and lane rates used for each camera sensor, as well as the number of Ethernet ports and link speed, configurable to 1G or 10G. If fewer cameras are used, the lane numbers for undefined cameras will be ignored. For example, if the number of cameras is 1, the lane number values set for cameras 2, 3, and 4 will be ignored.

The steps to update the platform design constraints include:

- Pin assignment at the top level of the HDL design
- Defining timing constraints

Proper configuration of these constraints is essential to produce intended reference design. However, the PDC file template provided with the reference design allows users to choose 1, 2, or 4 camera sensors. If more camera sensors are required, users must manually add the desired settings.

4.1.3.1. Pin Assignment for MIPI CSI-2 Camera Sensor Interface

In this section, the focus is solely on the MIPI CSI-2 camera sensor interface. The pins that need to be updated at the HDL top level are the camera MIPI clock lane, camera MIPI data lane, camera sensor reset, and camera I2C control. Special attention is required for the MIPI data lane assignment due to the use of two-dimensional array coding method at the HDL top-level design.

MIPI clock and data lanes are differential signals, and the pins assigned to them must be high-speed differential I/O types. For CertusPro-NX FPGAs, the typical I/O type used for MIPI clock and data lanes is LVDS (Low Voltage Differential Signaling) I/O pairs. Special handling is required when setting pin assignments for MIPI data lanes, as the reference design's top level uses a two-dimensional array coding method. This affects how the assignments are defined and mapped in PDC file. Table 4.2 illustrates the correct pin assignment approach under these conditions in the PDC file.

Table 4.2. PDC pin assignment for 2-dimension Array for MIPI Data Lane

Camera	System Verilog pin name	PDC mapped pin name
Camera 0	mipi_cam_data_p[0][0]	mipi_cam_data_p[0]
	mipi_cam_data_n[0][0]	mipi_cam_data_n[0]
	mipi_cam_data_p[0][1]	mipi_cam_data_p[1]
	mipi_cam_data_n[0][1]	mipi_cam_data_n[1]
	mipi_cam_data_p[0][2]	mipi_cam_data_p[2]
	mipi_cam_data_p[0][2]	mipi_cam_data_n[2]
	mipi_cam_data_p[0][3]	mipi_cam_data_p[3]
	mipi_cam_data_p[0][3]	mipi_cam_data_n[3]
Camera 1	mipi_cam_data_p[1][0]	mipi_cam_data_p[4]
	mipi_cam_data_n[1][0]	mipi_cam_data_n[4]
	mipi_cam_data_p[1][1]	mipi_cam_data_p[5]
	mipi_cam_data_n[1][1]	mipi_cam_data_n[5]
	mipi_cam_data_p[1][2]	mipi_cam_data_p[6]
	mipi_cam_data_p[1][2]	mipi_cam_data_n[6]
	mipi_cam_data_p[1][3]	mipi_cam_data_p[7]
	mipi_cam_data_p[1][3]	mipi_cam_data_n[7]
Camera 2	mipi_cam_data_p[2][0]	mipi_cam_data_p[8]
	mipi_cam_data_n[2][0]	mipi_cam_data_n[8]
	mipi_cam_data_p[2][1]	mipi_cam_data_p[9]
	mipi_cam_data_n[2][1]	mipi_cam_data_n[9]
	mipi_cam_data_p[2][2]	mipi_cam_data_p[10]
	mipi_cam_data_p[2][2]	mipi_cam_data_n[10]
	mipi_cam_data_p[2][3]	mipi_cam_data_p[11]
	mipi_cam_data_p[2][3]	mipi_cam_data_n[11]
Camera 3	mipi_cam_data_p[3][0]	mipi_cam_data_p[12]
	mipi_cam_data_n[3][0]	mipi_cam_data_n[12]
	mipi_cam_data_p[3][1]	mipi_cam_data_p[13]
	mipi_cam_data_n[3][1]	mipi_cam_data_n[13]
	mipi_cam_data_p[3][2]	mipi_cam_data_p[14]
	mipi_cam_data_p[3][2]	mipi_cam_data_n[14]
	mipi_cam_data_p[3][3]	mipi_cam_data_p[15]
	mipi_cam_data_p[2][3]	mipi_cam_data_n[15]

Using the PDC template from reference design, you can configure the timing constraint for the MIPI CSI-2 camera sensor interface by assigning the MIPI lane rate, as shown in [Figure 4.7](#). In the PDC file, this value will be used to assign the timing constraint required for the MIPI clock input.

Driver configuration is modified based on the target reference design. For example, each camera sensor has its own I2C control signals, so the driver must be modified to correctly identify and communicate with the target sensor. This ensures successful initialization and operation of the camera.

4.1.5. Verifying the Reference Design Configuration

You are expected to verify whether the configured setting delivers the target reference design. There are two steps required for you to verify: first, check the Lattice Radiant software's hierarchy list to confirm the selected camera number and camera configuration. Secondly, you need to check the clock summary details in the timing analysis report to ensure the assigned timing constraint matches the target configuration. [Figure 4.8](#) shows a screenshot of the reference design implemented on the CertusPro-NX Versa Evaluation Board. If the verified design is not the target reference design, then you are required to review the HDL parameters, DEFINE switch configuration, and PDC configuration, following the configuration flow shown in [Figure 4.1](#).

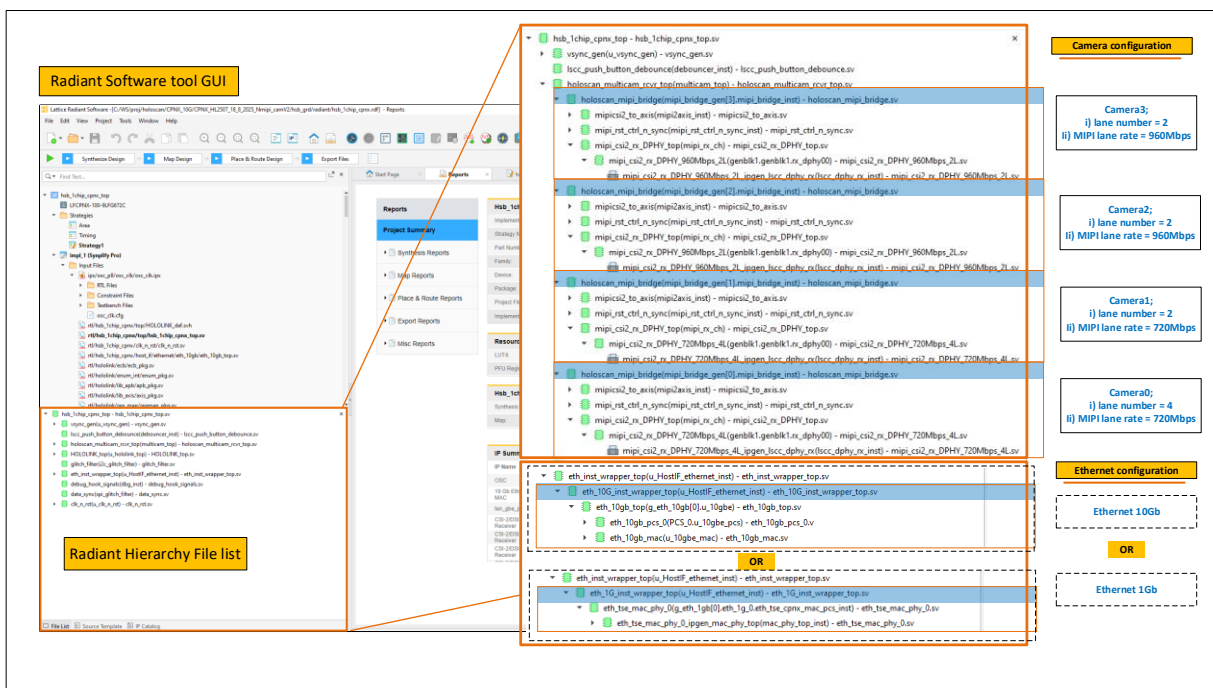


Figure 4.8. Screenshot of Lattice Radiant Software Hierarchy File List

4.2. Customizing the Reference Design for a CertusPro-NX Versa Evaluation Board

Figure 4.9 shows an overview of a configured CertusPro-NX Versa Evaluation board block diagram.

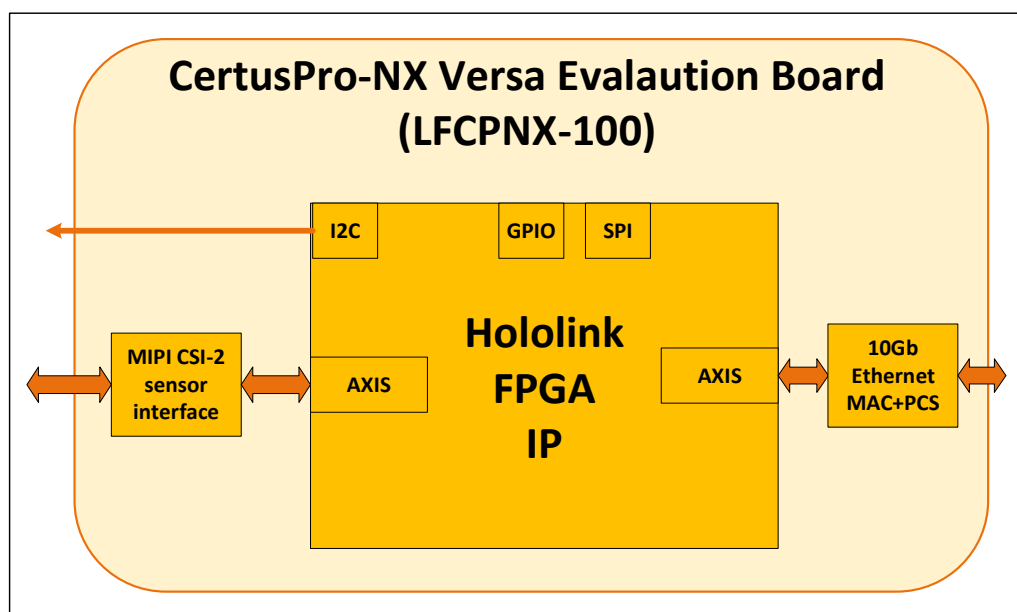


Figure 4.9. CertusPro-NX Versa Evaluation Board Block Diagram

Table 4.3 shows the configuration required for the reference design to fit onto the CertusPro-NX Versa Evaluation Board

Table 4.3. Configuration required for reference design to fit into CPNX Versa Evaluation Board

	Features	Configuration
Camera sensor interface	Number of camera sensors	1
	Camera 0: MIPI lane number	4
	Camera 0: MIPI lane rate	720 Mbps
Ethernet Link	Number of Ethernet port	1
	Ethernet link speed	10G

To customize the reference design, follow the steps below.

Note: This reference design is already integrated with a preset MIPI RX DPHY IP configuration, latest Hololink IP version 2507, and Ethernet IPs.

1. Configure HDL Parameter and DEFINE Switch.

The reference design need to be configured by changing the settings of the HDL parameter and DEFINE switch to meet the CertusPro-NX Versa board requirement using this file:

```
<project_name>/radiant/rtl/hsb_1chip_cpnx/top/HOLOLINK_def.svh
```

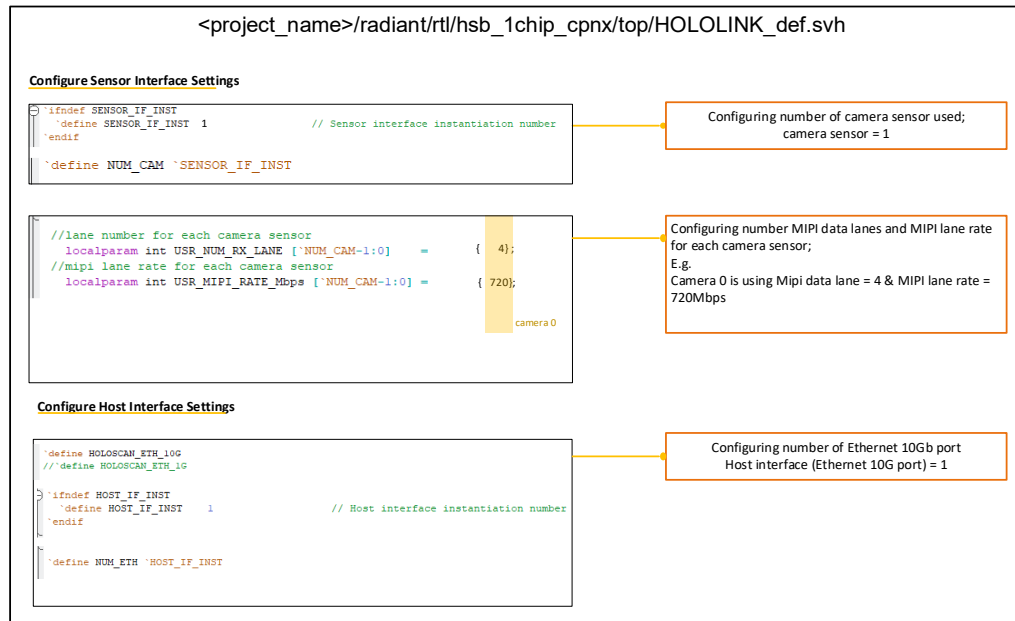


Figure 4.10. Required HDL Parameter and DEFINE Switch Configuration

- Review the PDC configuration file.

Figure 4.11 shows a screenshot of the `<project_name>/radiant/pdc/hsb_grd_CPNX_LFG672.pdc` file.

```

1 #####
2 # BEGIN: USR CONFIG SETUP
3 #####
4 #-----
5 # 1) SENSOR IF SETTING: CAM MIPI
6 #-----
7 #
8 # 1.1) available number of CAM
9 #-----
10 set CAM_NUM 1
11 #
12 #-----
13 # 1.2) cam0 config setup
14 #-----
15 set CAM0_LANE0 4
16 set MIPIlaneRate0_inMbps 720
17 set mipigear0 8
18 #-----
19 # 1.3) cam1 config setup
20 #-----
21 set CAM1_LANE0 4
22 set MIPIlaneRate1_inMbps 720
23 set mipigear1 8
24 #-----
25 # 1.4) cam2 config setup
26 #-----
27 set CAM2_LANE0 4
28 set MIPIlaneRate2_inMbps 720
29 set mipigear2 8
30 #-----
31 # 1.5) cam3 config setup
32 #-----
33 set CAM3_LANE0 4
34 set MIPIlaneRate3_inMbps 720
35 set mipigear3 8
36 #
37 #-----
38 # 2) HOST IF SETTING: ETHERNET
39 #-----
40 set ETH_NUM 1
41 #set ETH_SPEED_Gbps 1
42 set ETH_SPEED_Gbps 10
43 #
44 #####
45 # END: USR CONFIG SETUP
46 #####
            
```

Number of Camera Used

Camera0;
i) lane number
ii) MIPI lane rate

i) Number of Ethernet
ii) Ethernet speed link
10 Gbps

Figure 4.11. PDC Configuration File

3. Configure the driver. For details on how to configure the driver, refer to the [Applying Patch to Support the IMX258 Camera Module](#) section. To enable driver configuration support, follow the steps from the [Configuring the AGX Orin Developer Host Kit](#) section through the [Applying Patch to Support the IMX258 Camera Module](#) section.
4. Verify the reference design configuration. Review the Lattice Radiant software hierarchy file list and the clock summary from the place-and-route timing report. [Figure 4.12](#) shows the reference design in the Lattice Radiant software hierarchy file list. The reference design is now ready for compilation and implementation.

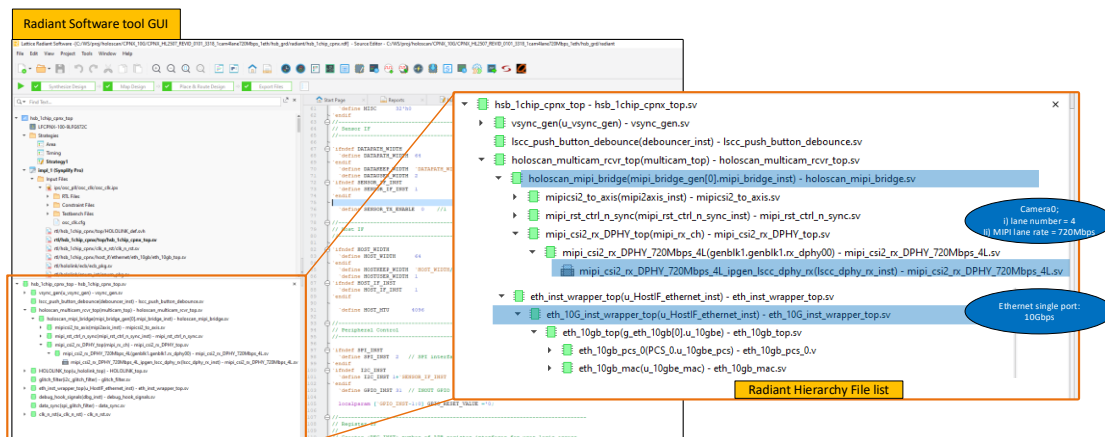


Figure 4.12. Lattice Radiant Software Hierarchy File List

5. Compiling the Reference Design

This section describes how to compile the reference design using Lattice Radiant software. For more details on the Lattice Radiant software, please refer to the [Lattice Radiant Software User Guide](#).

5.1. Configuring the Reference Design Parameters

To configure the reference design parameters, refer to the details in the [Customizing the Reference Design](#) section.

5.2. Assigning a Revision ID for each Bitstream

To assign a revision ID for the bitstream modify the reference design HDL top-level file, *hsb_1chip_cpnx_top.sv* as shown in [Figure 5.1](#).

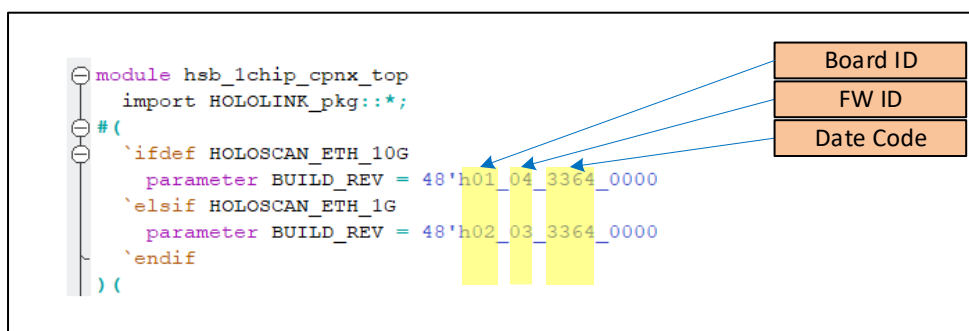


Figure 5.1. Revision ID assignment in *hsb_1chip_cpnx_top.sv*

[Table 5.1](#) shows the board ID and firmware ID in hexadecimal format.

Table 5.1. Revision ID's Board ID and FW ID Representation

Reference design with CPNX Device	Reference Design with 1G Ethernet	Reference Design with 10G Ethernet	Remarks
Board ID (in Hexadecimal)	02	01	Board ID used for CertusPro-NX device with ethernet link 1G or 10G
FW ID (in Hexadecimal)	03	04	FW ID used for FW upgrade that has code change (design integration, bug fix or new feature)

The date code refers to the date used to generate the release bitstream. The date code is represented differently, as shown in [Figure 5.2](#). The example in the figure illustrates the conversion of the date 4th November 2025 from decimal to binary and hexadecimal formats.

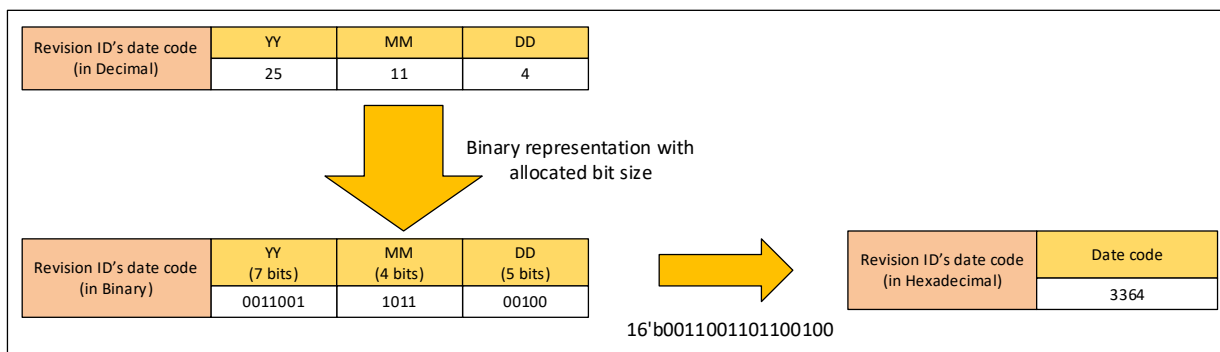


Figure 5.2. Revision ID Date Code Representation

5.3. Configuring the Lattice Radiant Software to Compile the Reference Design

1. Launch the Lattice Radiant software, as shown in Figure 5.3. Note that the project is compiled using the Lattice Radiant software version 2025.1.0.39.0.

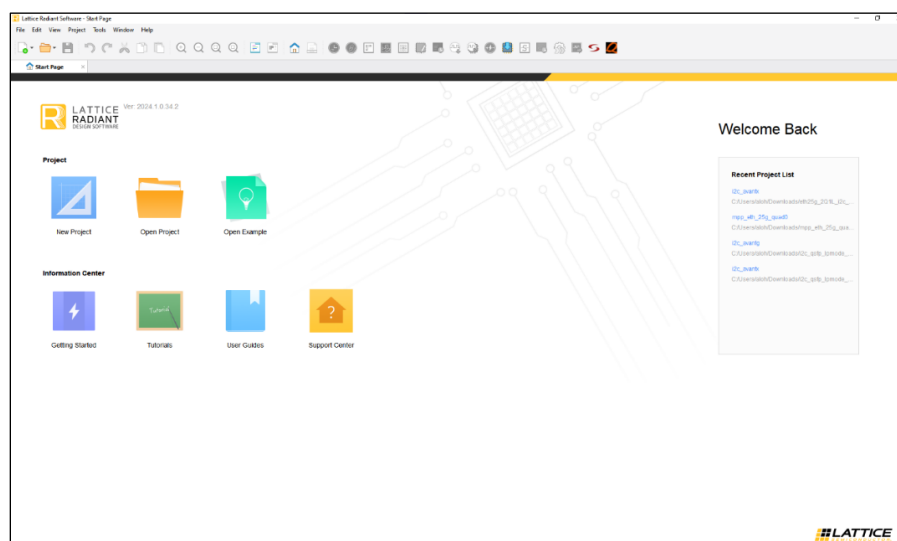


Figure 5.3. Lattice Radiant Software

2. Click **File > Open Project**. From the project database, open the Lattice Radiant software project file (*hsb_1chip_cpnx.rdf*), as shown in Figure 5.4.

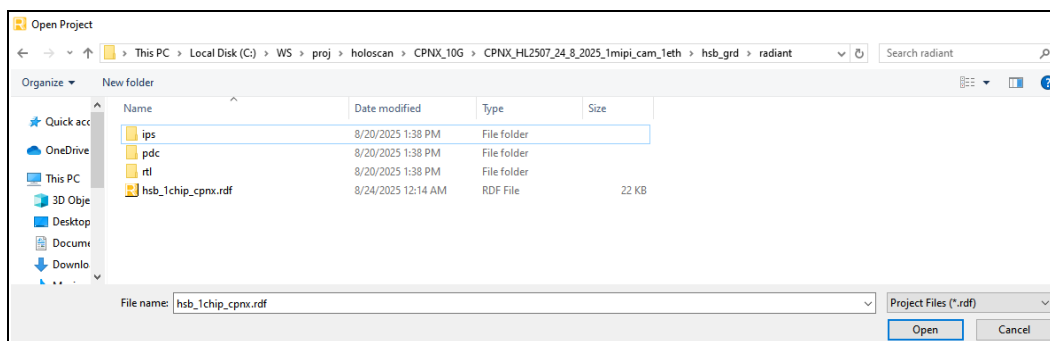


Figure 5.4. Open Project File

3. Modify the Place and Route design strategy details under **Strategy1 > Place & Route Design**. Refer to [Figure 5.5](#). The seed run count has been increased to 4 for this reference design, allowing you to select the reference design with the best place-and-route result and the best STA report. Specify multiple seeds for placement and routing during implementation¹.

Note:

1. This is controlled by the Placement Iteration setting under the Place and Route Design strategy in the Lattice Radiant software, which determines how many placement and routing iterations are performed. Each iteration uses a different cost table, resulting in a unique placement strategy and generating a distinct Uniform Database (.udb) file. By exploring multiple placement options, you increase the likelihood of achieving better timing closure and overall design performance. Using multiple seeds is beneficial for complex designs as it provides alternative implementations that may meet timing more effectively.

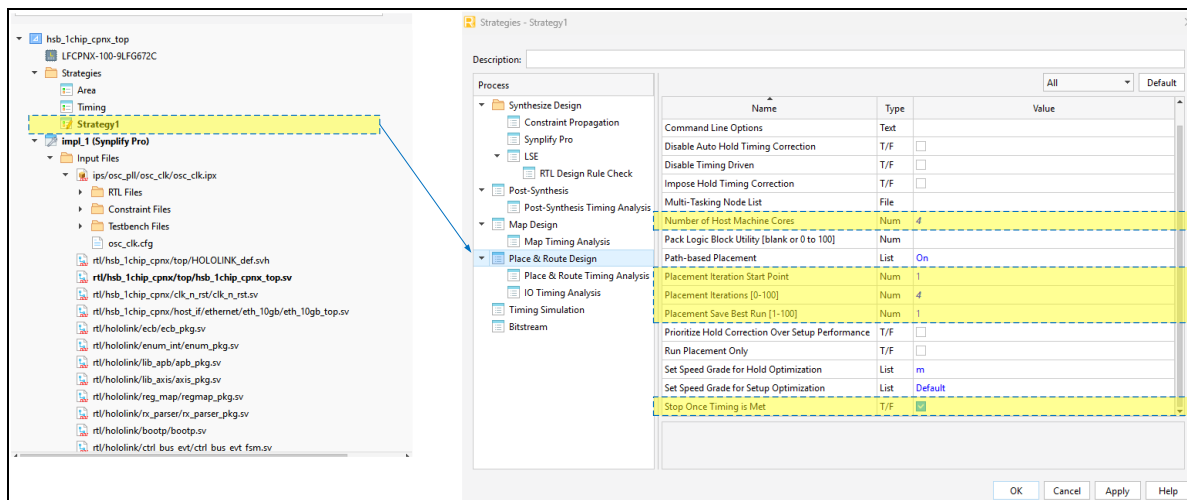


Figure 5.5. Place and Route Design Strategy

For further details on configuring Lattice Radiant Software, refer to [Lattice Radiant User Guide 2025.1](#).

5.4. Generating the Bitstream File

To create the FPGA bitstream file, click **Export Files** to generate the bit file. View the log message in the Export Reports folder for the generated bitstream.



Figure 5.6. Generate and Export Bitstream File

6. Implementing the Reference Design on the CertusPro-NX Versa Board

6.1. Requirements

6.1.1. Hardware

Table 6.1. Hardware Needed

Quantity	Description	Link
1	NVIDIA Jetson AGX Orin Developer Kit	NVIDIA Jetson AGX Orin Developer Kit
1	Lattice CertusPro-NX Versa Board (CPNX)	Lattice CertusPro-NX Versa Board
1	IMX258 Camera Sensor Module	Contact Lattice Sales: mailto:sales@latticesemi.com
1	Ethernet 10GBase-T (Cat 6 RJ-45) SFP+ Module (Ethernet 10G support)	10GBASE SFP+ RJ-45 Module Note: Tested with this device.
1	10/100/1000BASE-T SFP module (Ethernet 1G support)	10/100/1000BASE-T RJ45 SFP Module Note: Tested with this device.
1	CAT6 Ethernet cable (support 1G/10G)	Generic
1	Type-C USB Cable	Generic
1	DisplayPort Cable	Generic
1	Monitor with DisplayPort support (for AGX Orin)	Generic
1	Keyboard (for AGX Orin)	Generic
1	Mouse (for AGX Orin)	Generic
1	Mini USB Type-A cable for programming	Generic
1	12 V Power Supply	Generic
1	Host Orin AGX's power adapter	Generic

6.1.2. Software

- Jetpack 6.2.1
- NVIDIA Jetson Linux 36.4.4
- NVIDIA Holoscan SDK 3.3.0
- Holoscan Sensor Bridge (HSB) SDK 2.2.0

Figure 6.1 shows the hardware setup for the Jetson AGX Orin host and Lattice CertusPro-NX Versa Board.

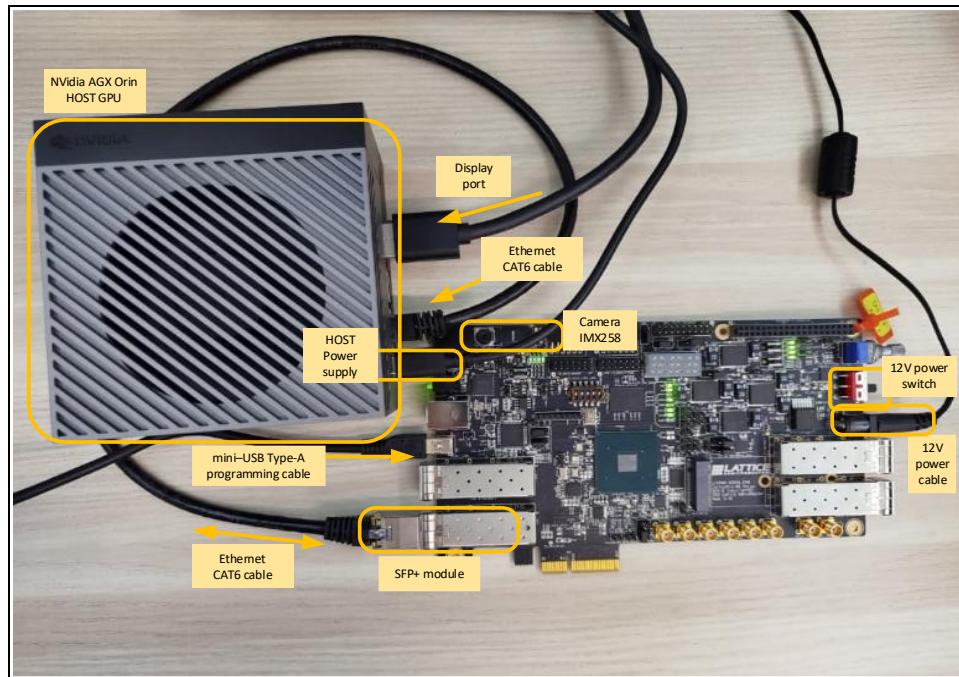


Figure 6.1. Hardware Setup



Figure 6-2. Camera Sensor Connection

6.2. Jetson AGX ORIN Host Setup

6.2.1. Setting Up the Jetson AGX ORIN Developer Kit

To set up the Jetson AGX Orin Developer Kit, follow the steps below. These steps are to be executed on an Ubuntu host PC terminal.

1. Set up an Ubuntu 22.04 host PC with at least 8 GB of RAM and an internet connection. Connect the Ubuntu host to the Jetson AGX Orin using a USB cable as shown in Figure 6.3.

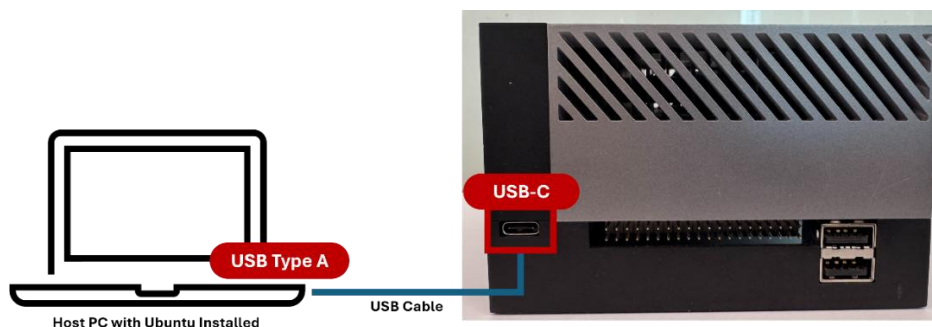


Figure 6.3. Ubuntu Host PC and Jetson AGX Orin Developer Kit Setup

2. Download and install the NVIDIA SDK Manager:

```
wget https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2204/x86_64/cuda-keyring_1.1-1_all.deb  
sudo dpkg -i cuda-keyring_1.1-1_all.deb  
sudo apt-get update  
sudo apt-get -y install sdkmanager
```
3. Place the AGX Orin into recovery mode by pressing and holding the recovery button as shown in Figure 6.4. While holding the recovery button, press the reset button and release it. Wait for about three seconds or more, then release the recovery button.

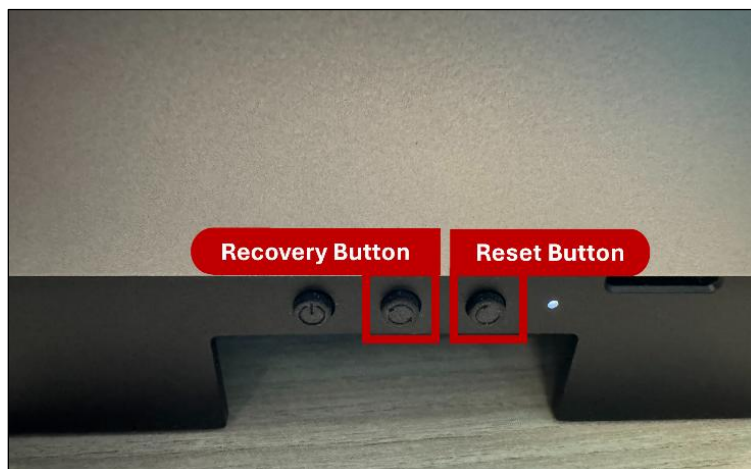


Figure 6.4. Recovery and Reset Button

4. Launch SDK Manager from the terminal, type:
`sdkmanager`
5. Log in using your NVIDIA developer credentials if you are using the SDK Manager for the first time. Select the checkbox, **Stay logged in**, then click the **LOGIN** button.
6. Select **Jetson AGX Orin [64 GB Developer Kit version]** as the target hardware after the SDK Manager window appears. Then click **OK**.

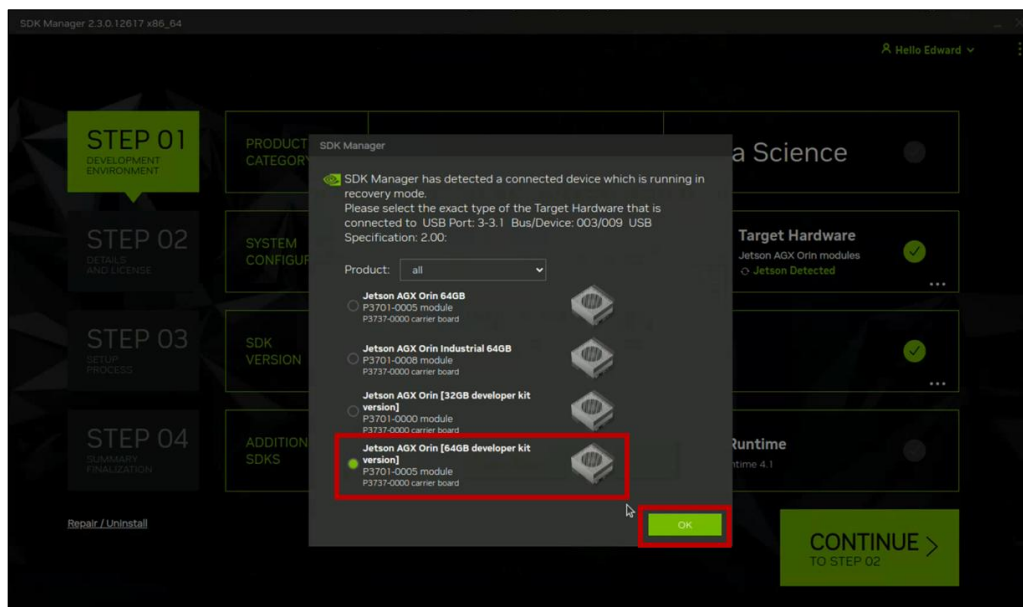


Figure 6.5. Jetson AGX Orin [64GB developer kit version]

7. Click the ... icon at (A) as shown in Figure 6.6, and select **JetPack 6.2.1 (Rev. 1)**. Then, click **CONTINUE** to proceed.

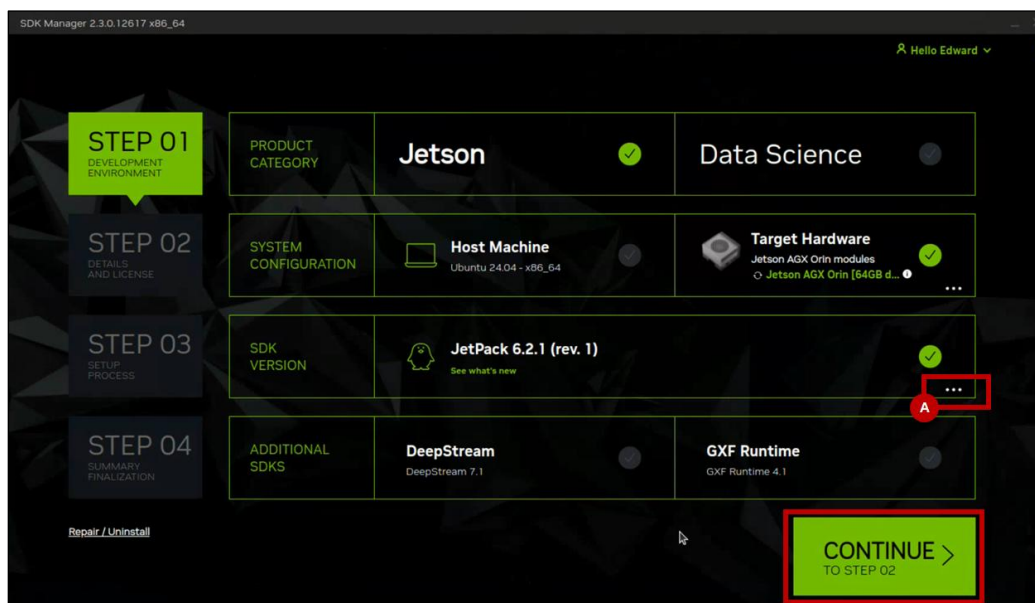


Figure 6.6. JetPack 6.2.1 (Rev.1)

8. Leave all settings as default, and select **I accept the terms and conditions of the license agreements**. Then, click **CONTINUE** to proceed with the installation process.

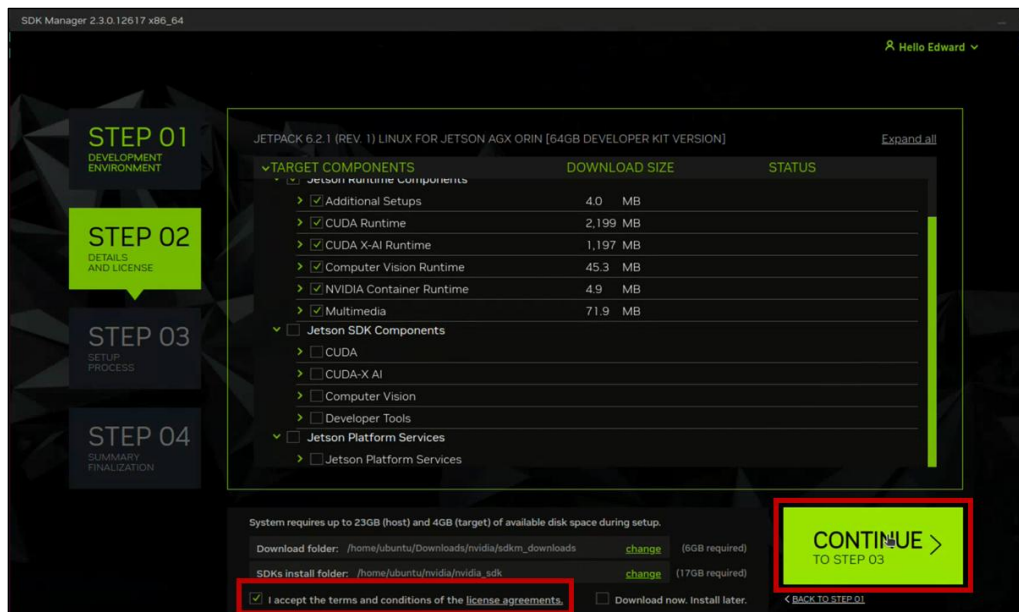


Figure 6.7. Terms and Conditions – License Agreement

9. In **STEP 03 – SETUP IN PROCESS**, as shown in Figure 6.8, wait for the download operation and OS image creation process to complete.



Figure 6.8. Download Operation and OS Image Creation

10. When prompted for a flash setup, enter the desired username to access the AGX Orin system, see letter **(A)** in [Figure 6.9](#), along with the password at **(B)**. Select **Pre-Config** from the dropdown at **(C)** and **EMMC** as the storage device at **(D)**. Then, click **Flash** to proceed.

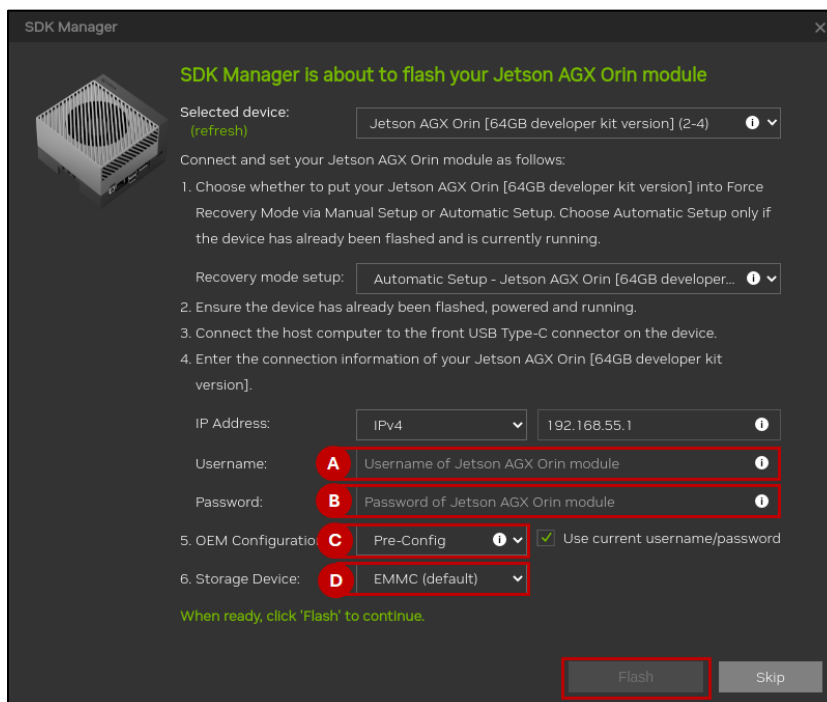


Figure 6.9. SDK Manager

11. Complete **STEP 04 – SUMMARY FINALIZATION** to finish the setup. Reset the Jetson AGX Orin system by pressing the reset button as shown in [Figure 6.4](#). Then proceed to the next section to configure the Jetson AGX Orin itself.

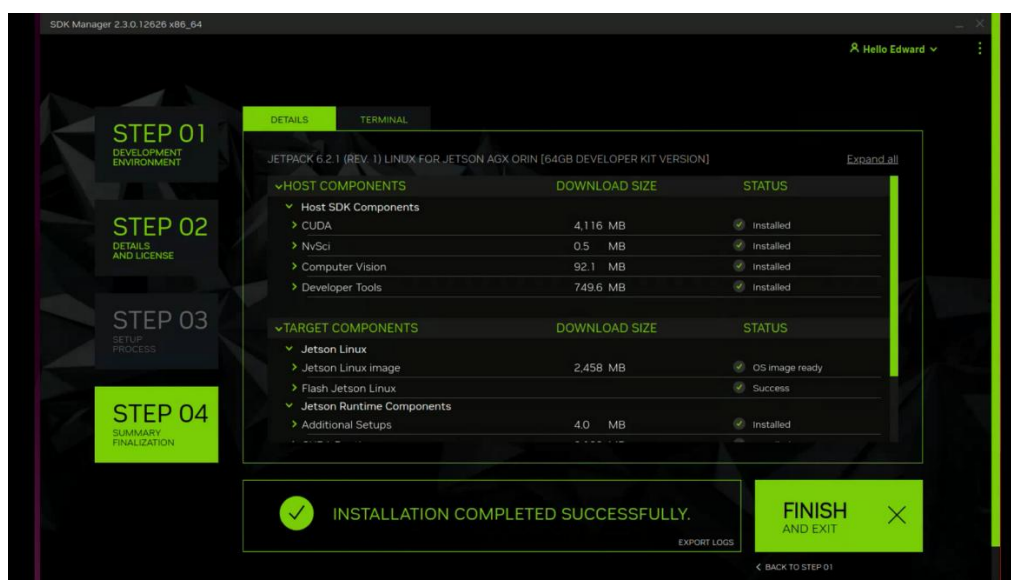


Figure 6.10. Summary Finalization

6.2.2. Configuring the AGX Orin Developer Kit

The customizable HSB sensor interface reference design is supported by the Jetson AGX Orin system running at JetPack 6.2.1. The following steps are one-time setup procedures and should be executed directly on the AGX Orin terminal, outside the Holoscan Sensor Bridge (HSB) docker container. All commands assume the HSB is connected via eno1 (the Ethernet interface on the Jetson AGX Orin).

1. Install git-lfs (some data files in the HSB source repository use it).

```
sudo apt-get update
sudo apt-get install -y git-lfs
```

2. Grant the user permission to the docker subsystem.

```
sudo usermode -aG docker $USER
```

3. Reboot the AGX Orin to apply the changes.

```
sudo reboot
```

4. Increase the network receive buffer size to support Linux sockets.

```
echo 'net.core.rmem_max = 31326208' | sudo tee /etc/sysctl.d/52-hololink-
rmem_max.conf
sudo sysctl -p /etc/sysctl.d/52-hololink-rmem_max.conf
```

5. Assign a static IP address (192.168.0.101) to the onboard network port.

```
EN0=eno1
sudo nmcli con add con-name hololink-$EN0 ifname $EN0 type ethernet ip4
192.168.0.101/24
sudo nmcli connection up hololink-$EN0
```

6. Power-on the CertusPro-NX Versa board and ensure it is properly connected. Then, ping to check connectivity.

```
ping 192.168.0.101
```

7. Isolate a processor core from the Linux kernel when running Linux socket-based examples¹.

To isolate that core, edit the /boot/extlinux/extlinux.conf file.

```
sudo nano /boot/linux/extlinux/extlinux.conf
```

8. Add the setting isolcpus=2 to the end of the line that begins with **APPEND**.

Your file should resemble the following:

```
TIMEOUT 30
DEFAULT primary

MENU TITLE L4T boot options

LABEL primary
    MENU LABEL primary kernel
    LINUX /boot/Image
    ...
    APPEND ${cbootargs} ...<other-settings>... isolcpus=2
```

9. Save and exit by pressing **Ctrl+O**, then **Enter**, followed by **Ctrl+X**.

The sensor bridge applications can run the network receiver process on a different core by setting the environment variable **HOLOLINK_AFFINITY** to the desired core.

For example, to run on the first processor core, type:

```
HOLOLINK_AFFINITY=0 python3 examples/linux_imx258_player.py
```

10. Run the **jetson_clocks** tool at startup to set the core clocks to their maximum.

```
JETSON_CLOCKS_SERVICE=/etc/systemd/system/jetson_clocks.service
cat <<EOF | sudo tee $JETSON_CLOCKS_SERVICE >/dev/null
[Unit]
Description=Jetson Clocks Startup
After=nvpmode1.service

[Service]
Type=oneshot
ExecStart=/usr/bin/jetson_clocks

[Install]
WantedBy=multi-user.target
EOF
sudo chmod u+x $JETSON_CLOCKS_SERVICE
sudo systemctl enable jetson_clocks.service
```

11. Set the Jetson AGX Orin power mode to **MAXN** for optimal performance. You can change this setting via the L4T power dropdown menu in the upper-right corner of the screen, as indicated by (A). Click (B) to select the **MAXN** option.

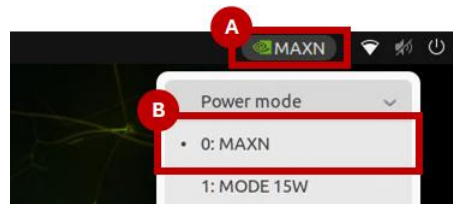


Figure 6.11. MAXN Power Mode

12. Reboot Jetson AGX Orin to apply changes.

```
sudo reboot
```

13. Enable PTP for **\$EN0** to synchronize timestamps in received data with the host system time. Install the **linuxptp** tool, then create a systemd service file to run **phc2sys** at boot time, ensuring the clock in **\$EN0** is synchronized with the system clock.

```
sudo apt update && sudo apt install -y linuxptp
EN0=en01
PHC2SYS_SERVICE=/etc/systemd/system/phc2sys-$EN0.service
cat <<EOF | sudo tee $PHC2SYS_SERVICE >/dev/null
[Unit]
Description=Copy system time to $EN0
Requires=NetworkManager.service
After=NetworkManager.service
After=timemaster.service

[Service]
Type=simple
ExecStartPre=timeout 3m bash -c "until [ \"\$(nmcli -g GENERAL.STATE device show $EN0)\" = \"100 (connected)\" ]; do sleep 1; done"
ExecStart=/usr/sbin/phc2sys -c $EN0 -s CLOCK_REALTIME -O 0 -S 0.0001

[Install]
WantedBy=multi-user.target
EOF
```


14. Configure it for execution at startup, and start it now.

```
sudo chmod u+x $PHC2SYS_SERVICE
sudo systemctl enable phc2sys-$EN0.service
sudo systemctl start phc2sys-$EN0.service
```

15. Next, run **ptp4l** to send PTP SYNC messages to **\$EN0**.

```
cat <<EOF | sudo tee /etc/linuxptp/hsb-ntp.conf >/dev/null
# This configuration is appropriate for NVIDIA Holoscan sensor bridge
# applications, where PTP messages are sent over L2 and a 1/2 second interval.
[global]
logSyncInterval -1
logMinDelayReqInterval -1
network_transport L2
EOF
```

16. Create a systemd service to run **ptp4l**.

```
PTP4L_SERVICE=/etc/systemd/system/ptp4l-$EN0.service
cat <<EOF | sudo tee $PTP4L_SERVICE >/dev/null
[Unit]
Description=Send PTP SYNC messages to $EN0
After=phc2sys-$EN0.service

[Service]
Type=simple
ExecStart=/usr/sbin/ptp4l -i $EN0 -f /etc/linuxptp/hsb-ntp.conf

[Install]
WantedBy=multi-user.target
EOF
```

```
sudo chmod u+x $PTP4L_SERVICE
sudo systemctl enable ptp4l-$EN0.service
sudo systemctl start ptp4l-$EN0.service
```

17. Log in to NVIDIA GPU Cloud (NGC).

- Register for an NGC account at <https://catalog.ngc.nvidia.com/>
- Create an API key at <https://ngc.nvidia.com/setup/api-key>
- Use the API key to log in to **nvcr.io**

```
$ docker login nvcr.io
Username: $oauthtoken
Password: <Your token key to NGC>
WARNING! Your password will be stored unencrypted in
/home/<user>/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

Note:

1. For high-bandwidth applications, such as 4K video acquisition, isolate the network receiver core to ensure optimal performance. Set processor affinity for the example program to the isolated core to improve performance and reduce latency. By default, the sensor bridge software runs the time-critical background network receiver process on core 3. Ensure core 3 is isolated from Linux scheduling so that no other processes run on it without explicit user assignment, thereby improving reliability and performance.

6.2.3. Building the Holoscan Sensor Bridge Demo Container

The MIPI IMX258 camera reference design is based on the HSB release version 2.2.0-GA.

1. Fetch the HSB source code version 2.2.0 from Github:

```
git clone https://github.com/nvidia-holoscan/holoscan-sensor-bridge -b 2.2.0
```

2. Build the HSB demo container for iGPU (default for AGX Orin).

```
cd holoscan-sensor-bridge
sh docker/build.sh --igpu
```

6.2.4. Applying Patch to Support the IMX258 Camera Module

1. Place the patch file **lattice_sensor_bridge_2.2.0.patch** in the same directory level as the **holoscan-sensor-bridge** directory.

2. Apply the patch using the following command:

```
patch -p0 < lattice_sensor_bridge_2.2.0.patch
```

3. Verify that **linux_imx258_player.py** exists after applying **lattice_sensor_bridge_2.2.0.patch** to confirm the patch was applied correctly:

```
ls holoscan-sensor-bridge/examples/linux_imx258_player.py
```

4. Rebuild the HSB demo container using the following command:

```
cd holoscan-sensor-bridge
sh docker/build.sh --igpu
```

6.3. Testing the System

This section provides details of the reference design testing on the CertusPro-NX Versa Evaluation Board with NVIDIA AGX ORIN GPU Host connected.

6.3.1. Programming the Bit File

This section outlines the steps for uploading the *.bit* file. There are two methods to update the reference design firmware into the CertusPro-NX Versa Evaluation board.

1. Local firmware update via USB interface.

Flash the firmware directly into the board using a USB connection, independent of the Holoscan ecosystem. This method is ideal for initial setup, debugging, and hands-on development.

2. Remote firmware update via Ethernet

Update the firmware remotely over an Ethernet connection using the Holoscan ecosystem. This method is suited for field updates and production environments, where remote access and automation are essential.

The following sections describes in detail the steps of the uploading the *.bit* file using the two methods.

6.3.1.1. Local Firmware Update via USB Interface

1. Connect the board to the PC using a mini-USB Type-A cable, as shown in Figure 6.12. Connect the power adaptor to the board power port and turn the switch **ON**. The board is now ready to be programmed.

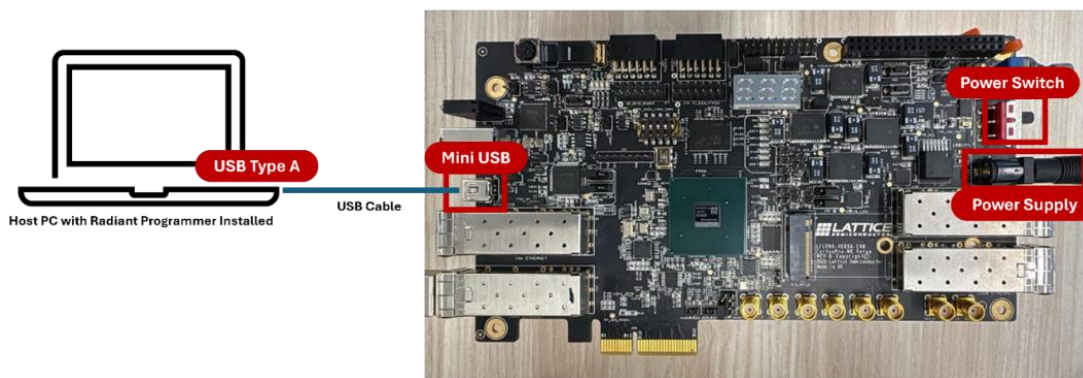



Figure 6.12. Hardware Setup for Bit File Programming

2. Click the Radiant Programmer icon () if the project is already open in the Lattice Radiant software, otherwise, launch the standalone Radiant Programmer. Enter the **Project Location**, **Project Name**, then click **OK**.

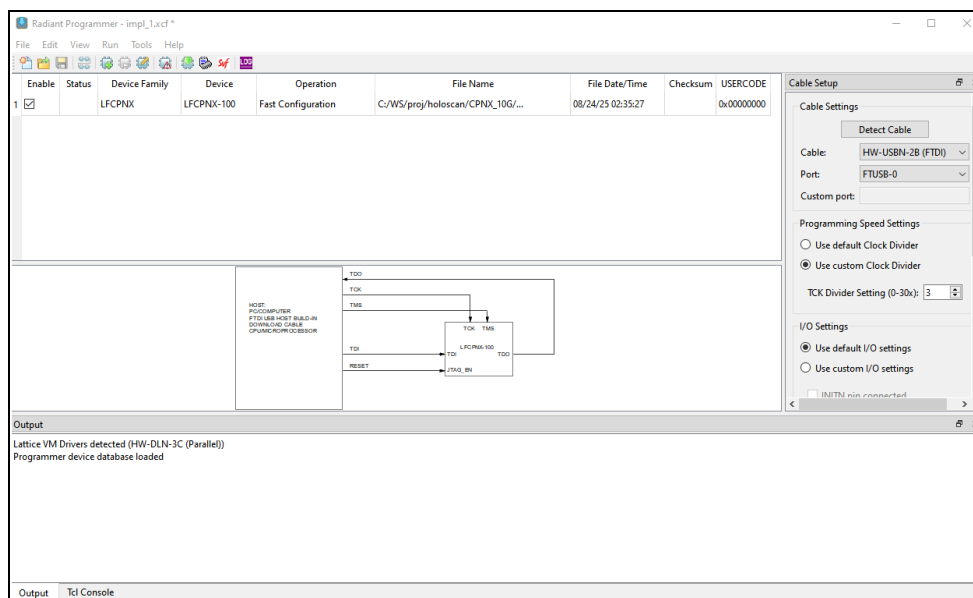


Figure 6.13. Radiant Programmer Window

- Click **Detect Cable** in the cable setup section, select **FTUSB-0**. Then click **OK** to proceed.

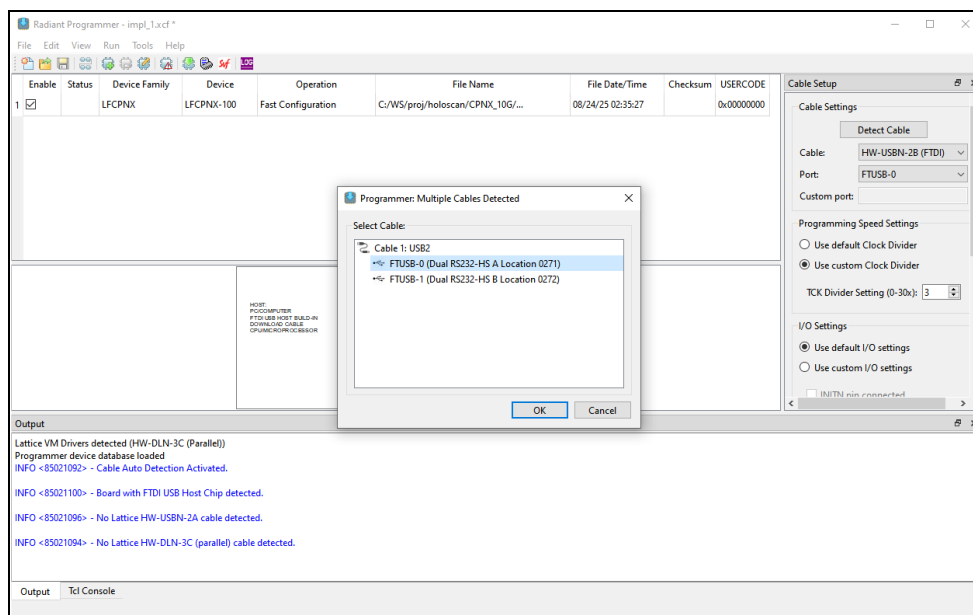


Figure 6.14. Select Cable Settings

- Select the appropriate SPI flash configuration to flash directly to external SPI flash. Refer to Figure 6.15 for the expected settings. Click the ... button under the file name bar and choose the bitstream file to program.

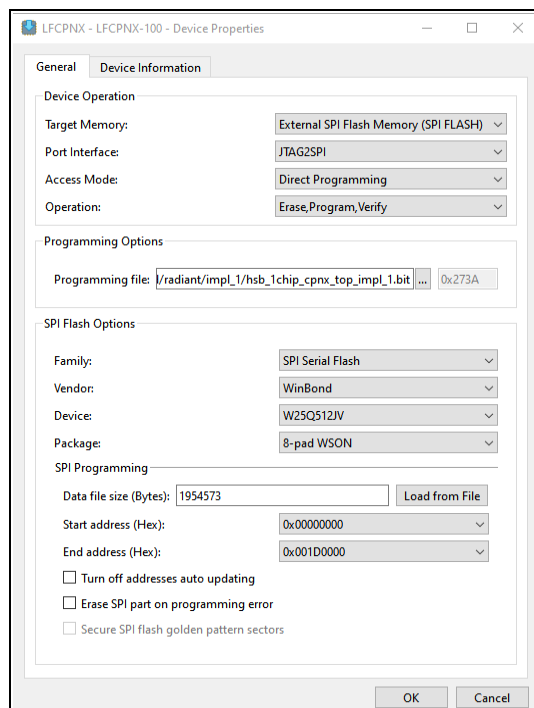


Figure 6.15. Load Bitstream File

5. Click the **Program Device** toolbar icon to load the bitstream into the board, as shown in [Figure 6.16](#).



Figure 6.16. Program Device Toolbar Icon

6. Verify that the programming status is **Operation: successful** in the output window, as shown in [Figure 6.17](#). To activate the flashed bitstream, power cycle the CertusPro-NX Versa Evaluation board by switching it **OFF**, then **ON**.

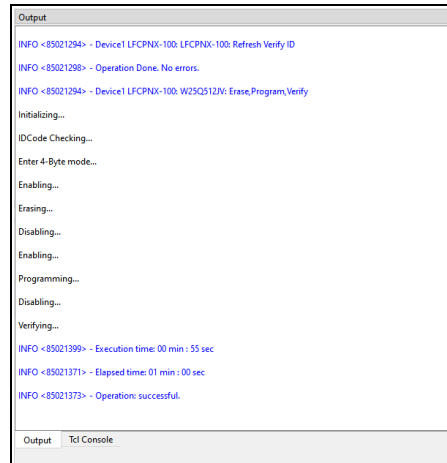


Figure 6.17. Message on Successful Programming

6.3.1.2. Remote Firmware Update via Ethernet

Before using this method note the following prerequisites:

- If you are using the CertusPro-NX Versa board for the first time, perform a [Local Firmware Update via USB](#). After the initial update, you may use the remote firmware update via Ethernet for future updates.
- For firmware changes involving a switch between 10G and 1G bitstreams on the same board, use the [Local Firmware Update via USB](#).
- Before proceeding, complete the steps in [Applying Patch to Support the IMX258 Camera Module](#).

If all prerequisites are met, follow the steps below:

1. Connect the AGX Orin ethernet port to the CertusPro-NX Versa board 10G/1G ethernet port (SFP Module). Refer to [Figure 6.1](#) for the setup.
2. Select and copy the following 10G/1G FPGA bitstream file into the Holoscan-Sensor-Bridge directory .
 - Bitstream file supporting 10G Ethernet: fpga_cpnx_versa_0104_2507.bit
 - Bitstream file supporting 1G Ethernet: fpga_cpnx_versa_0203_2507.bit
3. Open the terminal on the AGX Orin.
4. Change directory to the Holoscan-Sensor-Bridge directory .

```
cd holoscan-sensor-bridge
```
5. Update the FPGA bitstream for ethernet 10G or 1G using the command below. The process takes 20 to 30 minutes to complete.
 - Using bitstream that supports 10G Ethernet:

```
program_lattice_cpnx_versa scripts/manifest_cpnx_versa_10g.yaml
```
 - Using bitstream that supports 1G Ethernet:

```
program_lattice_cpnx_versa scripts/manifest_cpnx_versa_1g.yaml
```

- Wait for the process to complete. When prompted, power cycle the Versa board, then Press **Enter** to exit the program. Refer to [Figure 6.18](#) for the expected output.

```
INFO spi_flash_base.cpp:169 status tid=0xa0 -- CPNX: verify address=0x100000
INFO spi_flash_base.cpp:169 status tid=0xa0 -- CPNX: verify address=0x110000
INFO spi_flash_base.cpp:169 status tid=0xa0 -- CPNX: verify address=0x120000
INFO spi_flash_base.cpp:169 status tid=0xa0 -- CPNX: verify address=0x130000
INFO spi_flash_base.cpp:169 status tid=0xa0 -- CPNX: verify address=0x140000
INFO spi_flash_base.cpp:169 status tid=0xa0 -- CPNX: verify address=0x150000
INFO spi_flash_base.cpp:169 status tid=0xa0 -- CPNX: verify address=0x160000
INFO spi_flash_base.cpp:169 status tid=0xa0 -- CPNX: verify address=0x170000
INFO spi_flash_base.cpp:169 status tid=0xa0 -- CPNX: verify address=0x180000
INFO spi_flash_base.cpp:169 status tid=0xa0 -- CPNX: verify address=0x190000
INFO spi_flash_base.cpp:169 status tid=0xa0 -- CPNX: verify address=0x1A0000
INFO spi_flash_base.cpp:169 status tid=0xa0 -- CPNX: verify address=0x1B0000
INFO spi_flash_base.cpp:169 status tid=0xa0 -- CPNX: verify address=0x1C0000
INFO spi_flash_base.cpp:169 status tid=0xa0 -- CPNX: verify address=0x1D0000
You must now physically power cycle the sensor bridge device.
Press <Enter> to continue:
root@ubuntu-agx:/home/agx/hsb/holoscan-sensor-bridge-2.2.0#
```

Figure 6.18. Console Output — Remote Firmware Update

- Power cycle the Versa board by switching it **OFF**, and then **ON** to activate the newly flashed bitstream .

6.3.2. Camera Video Streaming Test

- Run the Holoscan Sensor Bridge demo container.

```
cd /<path>/<to>/<holoscan-sensor-bridge>
xhost +
sh docker/demo.sh
```

- Run the camera video streaming example .

This example will be streaming from the camera and display the video on AGX Orin.

Note: Once Step 1 is executed in the AGX Orin terminal, the demo container starts, the command prompt will switch to the environment inside the container.

- Run the video streaming demo for IMX258 camera using the following commands:

- When using 10G Ethernet:

```
python examples/linux_imx258_player.py
```

- When using 1G Ethernet:

```
python examples/linux_imx258_player.py --camera-mode=1
```

[Figure 6.19](#) shows the camera video streaming test setup. The video is streamed by the NVIDIA host and captured in an office environment under lighting conditions ranging from 140 lux to 135 lux. The test environment is identical for both 1G and 10G Ethernet configurations. Image quality remains consistent across both.

The primary objective of this reference design is to establish a robust and reliable bridge between the HSB interfaces and the NVIDIA host. Image tuning is not included. This process typically involves defining image quality parameters, configuring processing pipelines, and optimizing the image signal processor (ISP)—tasks that are highly specific to the camera sensor and require dedicated resources.

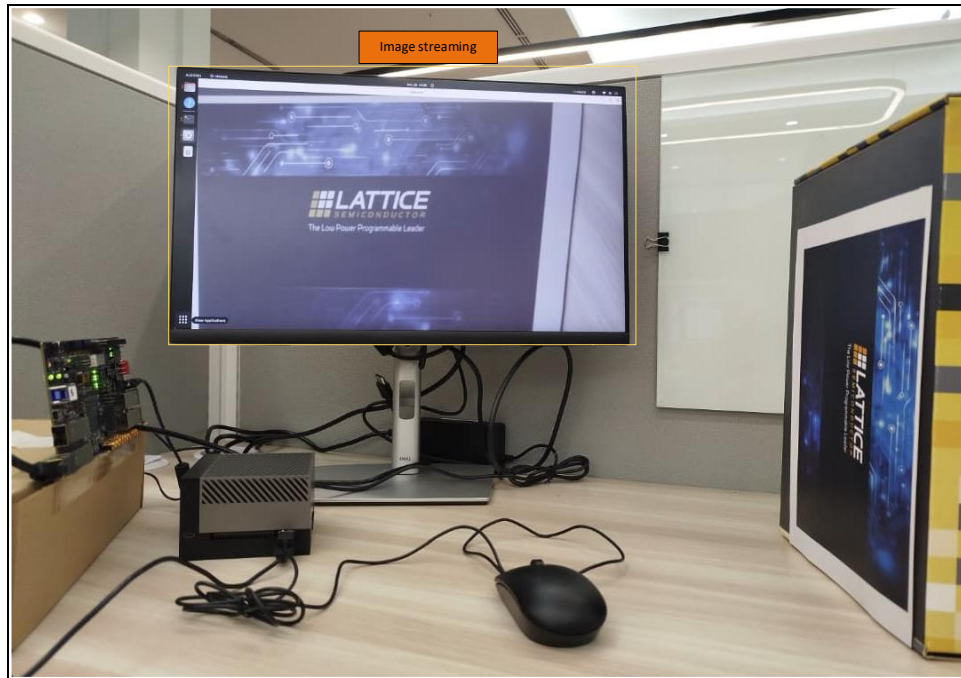


Figure 6.19. Camera Video Streaming Test Setup

7. Resource Utilization

The section shows the resource utilization of the reference design for the CertusPro-NX FPGA (LFCPNX-100-9LFG672I) device using the Synplify Pro tool in the Lattice Radiant software version 2025.1.0.39.0. Refer to the [Compiling the Reference Design](#) section for configuration details. Note that changes in design attributes may affect resource utilization, and values may vary over time due to feature updates, bug fixes, or other modifications.

7.1. Resource Utilization with Ethernet 1G

[Table 7.1](#) shows one camera sensor interface configured with four MIPI lanes at 720 Mbps, paired with a single-port 1G Ethernet host interface.

Table 7.1. Logic Consumption for each Instance Available in the Reference Design

Instance name	LUT4 Logic	LUT4 Distributed RAM	LUT4 Ripple Logic	PFU Registers	DSP MULT	EBR
multicam_top (camera sensor interface)	1602	0	220	1211	0	9
Hololink_top (Hololink IP)	11430	1050	3040	16091	8	33
Ethernet_inst (Ethernet 1G- host interface)	3100	18	910	3463	0	5
Clk_rst	35	0	0	38	0	0
Miscellaneous	283	48	212	783	0	0
Total	16450	1116	4382	21586	8	47

***Note:** Other submodules are categorized as miscellaneous in this table.

[Table 7.2](#) compares the reference designs logic consumption with the available resources of CertusPro-NX (LFCPNX-100-9LFG672I) device, based on the configuration described in [Table 7.1](#).

Table 7.2. Comparison between Reference Design Utilization and CertusPro-NX Resources Available

	LUT	PFU registers	DSP Mult	EBR
CPNX 100 device logic resource	79872	79872	156	208
Reference design logic resource consumed	21948	21586	8	47
Reference design logic resource consumed (%)	27.48	27.03	5.13	22.60

From the place-and-route report, the device SLICE utilization summary after final SLICE packing is 80% used (32,211 out of 39,936 slices).

7.2. Resource Utilization with Ethernet 10G

Table 7.3 shows one camera sensor interface configured with four MIPI lanes at a lane rate of 720 Mbps, and a single-port 10G Ethernet host interface.

Table 7.3. Logic Consumption for each Instance Available in the Reference Design

Instance name	LUT4 Logic	LUT4 Distributed RAM	LUT4 Ripple Logic	PFU Registers	DSP MULT	EBR
multicam_top (camera sensor interface)	1717	0	212	1321	0	8
Hololink_top (Hololink IP)	18446	2052	2948	23395	8	35
Ethernet_inst (Ethernet 10G- host interface)	3858	0	110	2971	0	4
Clk_rst	36	0	0	38	0	0
Miscellaneous	281	48	212	725	0	0
Total	24338	2100	3482	28450	8	47

***Note:** Other submodules are categorized as miscellaneous in this table.

Table 7.4 compares the reference design logic consumption with the available resources of the CertusPro-NX (LFPCPX-100-9LFG672I) device, based on the configuration described in Table 7.3.

Table 7.4. Comparison between Reference Design Utilization and CertusPro-NX Resources Available

	LUT	PFU registers	DSP Mult	EBR
CPNX 100 device logic resource	79872	79872	156	208
Reference design logic resource consumed	29920	28450	8	47
Reference design logic resource consumed (%)	37.46	35.62	5.13	22.60

From the place-and-route report, the device SLICE utilization summary after the final SLICE packing is 89% used (35,941 out of 39,936 slices).

8. Debug Methodology

This section outlines the recommended steps for debugging the camera video streaming setup in the Holoscan Sensor Bridge reference design. By following this procedure, you can verify system functionality and identify potential issues during integration and testing.

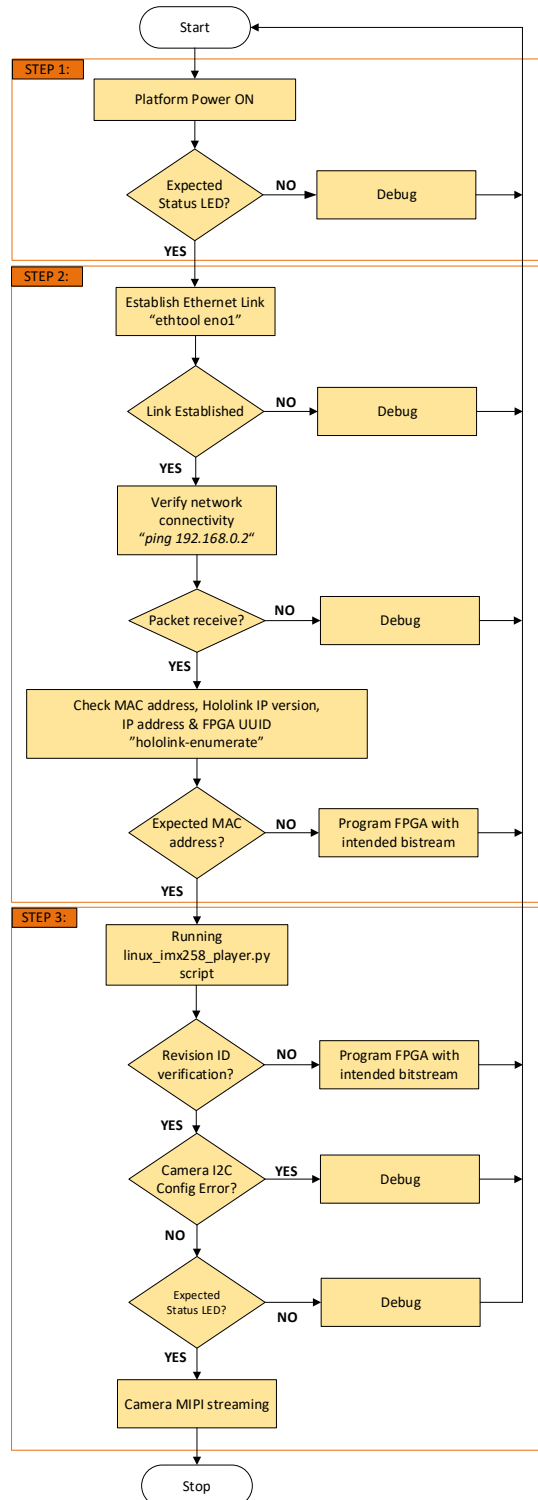


Figure 8.1. Debug Methodology Flow Chart For RD

8.1. Powering Up the Evaluation Board

1. Power cycle the board after programming.
After programming the FPGA firmware into the external SPI flash, fully power down the platform, then power it back up. This power cycle ensures the FPGA correctly loads and configures the new firmware.
2. Verify LED status.
3. After powering up, check the status LEDs on the evaluation board. Refer to [Figure 8.2](#) or LED locations. These LEDs indicate the board operational status. [Table 8.1](#) explains each LED behavior and meaning.
Note: This step assumes the HSB Docker container is already set up on the host system.

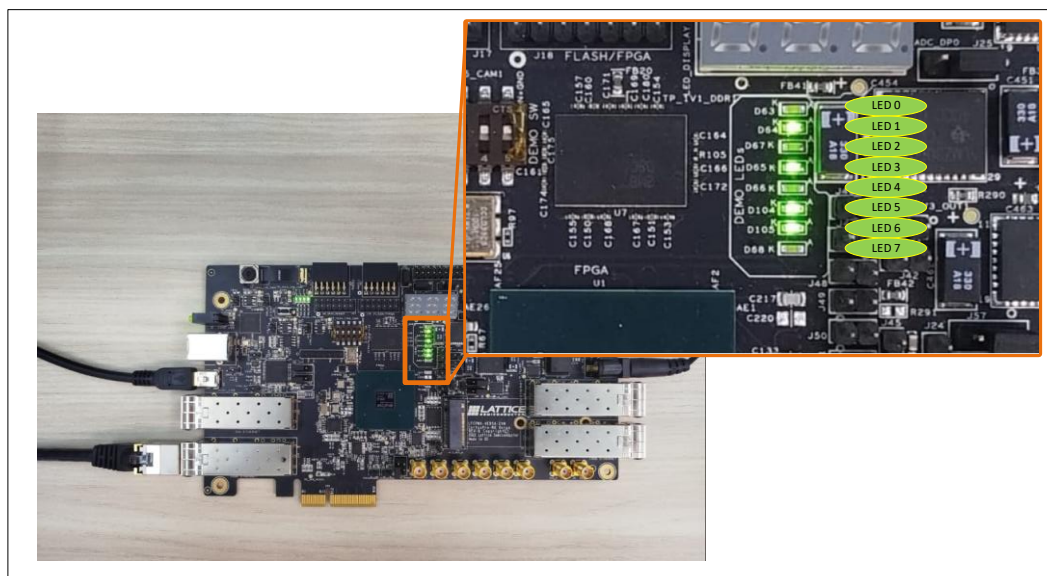


Figure 8.2. Position of the LED on CPNX Versa Evaluation Board

[Table 8.1](#) shows the LED status upon power-up.

Table 8.1. LED status upon Board Power-up

LED	Connected Signal	Board Power UP
LED0	pcs_clk_heart_beat	LED toggles
LED1	hif_clk_heart_beat	LED toggles
LED2	byte_clk_heart_beat	OFF
LED3	plat_rst_n (platform reset)	ON
LED4	sif_rst	OFF
LED5	sw_sen_rst	ON
LED6	cam_rst_n	ON
LED7	hif_rst	OFF

8.1.1. Troubleshooting Common Issues

- **Issue A: No status LEDs are on.**
 - Workaround 1: Check that the CertusPro-NX Versa board power supply is connected and the 12 V switch is turned on.
 - Workaround 2: Power cycle the board using 12 V switch.
 - Workaround 3: Reprogram the FPGA to confirm the correct bitstream is used.
 - Workaround 4: If the issue persists, submit a technical support case through www.latticesemi.com/techsupport.

- **Issue B: LED1 is not toggling.**
Possible cause: Ethernet IP configuration issue.
 - Workaround 1: Regenerate the Ethernet IP with the correct settings. Recompile and reprogram the FPGA device with the newly compiled FPGA bitstream.
 - Workaround 2: If the issue persists, submit a technical support case through www.latticesemi.com/techsupport.
- **Issue C: Reset status LEDs behave unexpectedly.**
Possible cause: Custom changes to HDL source code or software patches.
 - Review your modifications to the reference design HDL source code and any applied software patches.

8.2. Verifying the Ethernet Connection

1. Check the Ethernet link status.

On the host terminal, run `ethtool eno1` command. This verifies whether the Ethernet link is established, and confirms that the link speed is as expected. If successful, the terminal will display the link status, as shown in [Figure 8.3](#).

```

agx_orin@agxorin: ~
agx_orin@agxorin:~$ ethtool eno1
[sudo] password for agx_orin:
Settings for eno1:
    Supported ports: [ ]
    Supported link modes:   100baseT/Half 100baseT/Full
                           1000baseT/Full
                           10000baseT/Full
                           1000baseKX/Full
                           10000baseKX4/Full
                           10000baseKR/Full
                           2500baseT/Full
                           5000baseT/Full
    Supported pause frame use: Symmetric Receive-only
    Supports auto-negotiation: Yes
    Supported FEC modes: Not reported
    Advertised link modes:  100baseT/Half 100baseT/Full
                           1000baseT/Full
                           10000baseT/Full
                           1000baseKX/Full
                           10000baseKX4/Full
                           10000baseKR/Full
                           2500baseT/Full
                           5000baseT/Full
    Advertised pause frame use: Symmetric Receive-only
    Advertised auto-negotiation: Yes
    Advertised FEC modes: Not reported
    Link partner advertised link modes:  10baseT/Full
                                        100baseT/Full
                                        1000baseT/Full
                                        10000baseT/Full
                                        2500baseT/Full
                                        5000baseT/Full
    Link partner advertised pause frame use: No
    Link partner advertised auto-negotiation: Yes
    Link partner advertised FEC modes: Not reported
    Speed: 1000Mb/s
    Duplex: Full
    Auto-negotiation: on
    Port: Twisted Pair
    PHYAD: 0
    Transceiver: external
    MDI-X: Unknown
    Supports Wake-on: g
    Wake-on: d
    Current message level: 0x00000000 (0)
    Link detected: yes
    
```

Figure 8.3. Ethernet Link Establishment Status

2. Test the Ethernet connectivity.

Run `ping 192.168.0.2` command on the host terminal. A successful response confirms that the network connection is active. You should see messages indicating that packets are being received. Refer to [Figure 8.4](#) for an example of successful ping response from the evaluation board.

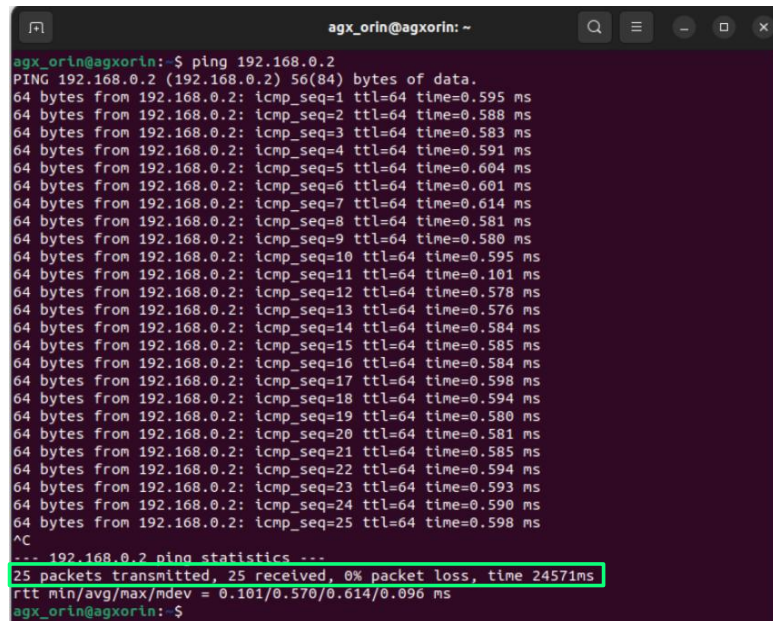


Figure 8.4. Data Packet Received by Host

3. Verify the device information.

On the host HSB container prompt, run `hololink-enumerate` command. This command displays the MAC address, Hololink IP version, IP address, and FPGA UUID. See [Figure 8.5](#) for a sample output.

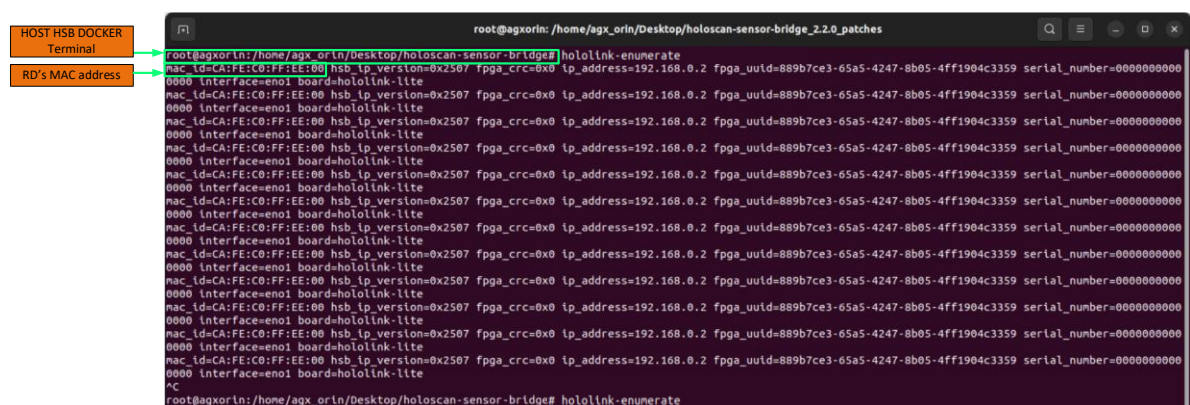


Figure 8.5. Sample Output

8.2.1. Troubleshooting Connection Issues

If you encounter issues during steps 1 or 2, try the following:

- Workaround 1: Ensure the SFP module is securely locked into the SFP cage.
- Workaround 2: Confirm that the LAN cable is properly connected to both the SFP module and the host LAN port. If the connection is correct, the LAN indicator light should be on, as shown in [Figure 8.6](#).

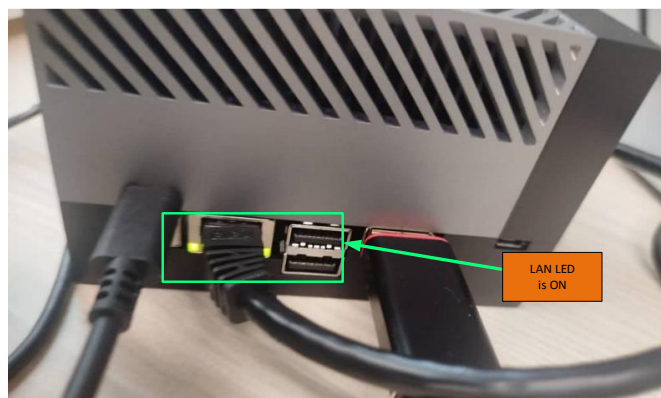


Figure 8.6. LAN Light Indication on Host

8.3. Running the Streaming Script (linux_imx258_player.py)

8.3.1. Preparing the Streaming Script

Run the script. Follow the steps in the [Camera Video Streaming Test](#) section to run the `linux_imx258_player.py` command on the host HSB docker container prompt.

8.3.2. Monitoring the Output of the Streaming Script

Once the script is running, observe the following:

1. Verify the Revision ID.

Check the log for the Revision ID to confirm the FPGA firmware version. [Figure 8.7](#) shows the Revision ID version from the host HSB docker container prompt, which represents the bitstream version for the CertusPro-NX with 10G Ethernet.

```

root@agxorin:/home/agx_orin/Desktop/holoscan-sensor-bridge_2.2.0_patches
root@agxorin:/home/agx_orin/Desktop/holoscan-sensor-bridge_2.2.0_patches# python examples/linux_imx258_player.py
INFO 898 _init_ linux_imx258_player.py:39 tid=MainThread -- _init_
INFO 899 hololink.cpp:1496 configure_hsb tid=0xf1 -- HSB IP version=0x2507 datecode=0x1023336
[info] [fragment.cpp:765] Loading extensions from configs...
INFO 2693 compose linux_imx258_player.py:50 tid=MainThread -- compose
INFO 2704 compose linux_imx258_player.py:83 tid=MainThread -- frame_size=2592000
INFO 3075 setup_base_receiver_op.py:58 tid=MainThread -- setup
[info] [gxf_executor.cpp:326] Creating context
[info] [gxf_executor.cpp:2398] Activating Graph...
[info] [gxf_executor.cpp:2467] Running Graph...
[info] [gxf_executor.cpp:2469] Waiting for completion...
[info] [greedy_scheduler.cpp:191] Scheduling 5 entities
INFO 3102 start_base_receiver_op.py:71 tid=Dummy-1 -- frame_size=2592000 frames=4386484224
INFO 3105 data_channel.cpp:368 configure_socket tid=0xf1 -- (done) DataChannel configure_socket socket_fd=71 local_ip=192.168.0.121.
INFO 3105 start_base_receiver_op.py:76 tid=Dummy-1 -- local_ip='192.168.0.121' local_port=46805
INFO 3105 data_channel.cpp:139 compute_payload_size tid=0xf1 -- header_size=78 payload_size=1408 packets=1841
error: XDG_RUNTIME_DIR is invalid or not set in the environment.
[info] [context.cpp:52] Vulkan Version:
[info] [context.cpp:52] - available: 1.3.275
[info] [context.cpp:52] - requesting: 1.2.0
[info] [context.cpp:52] Used Instance Layers :
[info] [context.cpp:52] Used Instance Extensions :
[info] [context.cpp:52] VK_KHR_surface
[info] [context.cpp:52] VK_KHR_xcb_surface
[info] [context.cpp:52] VK_EXT_debug_utils
[info] [context.cpp:52] VK_KHR_external_memory_capabilities
[info] [context.cpp:52] Compatible Devices :
[info] [context.cpp:52] 0: NVIDIA Tegra Orin (nvgpu)
[info] [context.cpp:52] Physical devices found :
[info] [context.cpp:52] 1

```

Figure 8.7. Revision ID Version

2. Check the camera for I2C configuration errors.
Look for any I2C configuration errors in the log, when the `linux_imx258_player.py` script is executed. Figure 8.8 shows an example of a failed camera I2C configuration.

```

root@agxorin: /home/agx_orin/Desktop/holoscan-sensor-bridge_2.2.0_patches
root@agxorin:/home/agx_orin/Desktop/holoscan-sensor-bridge_2.2.0_patches# python examples/linux_imx258_player.py
<Frozen importlib._bootstrap_external>:1297: FutureWarning: The cuda.cuda module is deprecated and will be removed in a future release, please switch to u
se the cuda.bindings.driver module instead.
INFO 196 main linux_imx258_player.py:185 tid=MainThread -- Initializing.
I2C Controller Address 1
Camera ID 0
INFO 1413 __init__ linux_imx258_player.py:39 tid=MainThread -- __init__
INFO hololink.cpp:1496 configure_hsb tid=0x109 -- HSB IP version=0x2507 datecode=0x1023336
Traceback (most recent call last):
  File "/home/agx_orin/Desktop/holoscan-sensor-bridge_2.2.0_patches/examples/linux_imx258_player.py", line 238, in <module>
    main()
  File "/home/agx_orin/Desktop/holoscan-sensor-bridge_2.2.0_patches/examples/linux_imx258_player.py", line 227, in main
    camera_0.configure()
  File "/usr/local/lib/python3.12/dist-packages/hololink/sensors/imx258.py", line 164, in configure
    self.set_register(reg, val)
  File "/usr/local/lib/python3.12/dist-packages/hololink/sensors/imx258.py", line 209, in set_register
    self.i2c.i2c_transaction(
RuntimeError: I2C port indicates I2C_NAK.
root@agxorin:/home/agx_orin/Desktop/holoscan-sensor-bridge_2.2.0_patches#

```

Figure 8.8. Camera I2C configuration fail log

3. Observe the status LED.
Refer to Table 8.2 for expected LED behavior. The byte clock heartbeat LED should toggle, indicating that the camera sensor is transmitting MIPI data.

Table 8.2. Expected Status LED Behavior when running `linux_imx258_player.py` script

LED	Connected Signal	MIPI camera streaming
LED0	pcs_clk_heart_beat	LED toggles
LED1	hif_clk_heart_beat	LED toggles
LED2	byte_clk_heart_beat	LED toggles
LED3	plat_rst_n (platform reset)	ON
LED4	sif_rst	OFF
LED5	sw_sen_rst	ON
LED6	cam_rst_n	ON

4. Confirm MIPI Camera Streaming
If no errors are detected, the MIPI camera stream should display successfully on the host monitor.

8.3.3. Troubleshoot Common Issues

- Revision ID version mismatch.
 - Workaround: Reprogram the HSB board with the correct FPGA bitstream version.
- Camera I2C Configuration Error
 - Workaround 1: Ensure the `IMX258` camera sensor is properly connected.
 - Workaround 2: If you are using your own reference design HDL code, verify that the camera reset signal is not asserted.
 - Workaround 3: If you are using your own software patch, confirm the I2C configuration port is correctly targeted.
 - Workaround 4: If you are using your own software patch, ensure the camera sensor register settings are correct.

Appendix A: Known Issues

1. When using some type of 1000BASE-T RJ-45 SFP module, the network link fails to establish, and no link LED activity is observed.

References

- [CSI-2/DSI D-PHY Receiver IP Core - User Guide \(FPGA-IPUG-02081\)](#)
- [CSI-2/DSI D-PHY Receiver IP Core](#) web page
- [CSI-2/DSI D-PHY Transmitter IP Core](#) web page
- [Lattice Radiant](#) FPGA design software
- [Lattice Radiant Software User Guide](#)
- [Lattice Insights](#) for Lattice Semiconductor training courses and learning plans
- [NVIDIA Holoscan - NVIDIA Docs](#) NVIDIA Holoscan Docs Hub

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, November 2025

Section	Change Summary
All	Initial release.



www.latticesemi.com